High-speed Apply Engine Reference Manual

Supporting

Version 11.2 of High-speed Apply Engine
Version 11.2 of Log Master for DB2
Version 11.2 of LOADPLUS for DB2
Version 11.2 of Recovery Management for DB2
Version 11.2 of Database Administration for DB2

May 2015
Contacting BMC Software

Several methods are available for contacting BMC Software.

You can access the BMC Software website at http://www.bmc.com. From this website, you can obtain information about the company, its products, corporate offices, special events, and career opportunities.

United States and Canada

<table>
<thead>
<tr>
<th>Address</th>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC SOFTWARE INC 2101 CITYWEST BLVD HOUSTON TX 77042-2827 USA</td>
<td>1 713 918 8800 or 1 800 841 2031</td>
<td>1 713 918 8000</td>
</tr>
</tbody>
</table>

Outside United States and Canada

<table>
<thead>
<tr>
<th>Telephone</th>
<th>Fax</th>
</tr>
</thead>
<tbody>
<tr>
<td>+01 713 918 8800</td>
<td>+01 713 918 8000</td>
</tr>
</tbody>
</table>

© Copyright 1999-2015 BMC Software, Inc. BMC, BMC Software, and the BMC Software logo are the exclusive properties of BMC Software, Inc., are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other BMC trademarks, service marks, and logos may be registered or pending registration in the U.S. or in other countries. All other trademarks or registered trademarks are the property of their respective owners.

Explorer and Together are trademarks or registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

AIX, DB2, DRDA, Distributed Relational Database Architecture, IBM, RETAIN, and z/OS are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

UNIX is the registered trademark of The Open Group in the US and other countries.

The information included in this documentation is the proprietary and confidential information of BMC Software, Inc., its affiliates, or licensors. Your use of this information is subject to the terms and conditions of the applicable End User License agreement for the product and to the proprietary and restricted rights notices included in the product documentation.

Restricted rights legend

U.S. Government Restricted Rights to Computer Software. UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES. Use, duplication, or disclosure of any data and computer software by the U.S. Government is subject to restrictions, as applicable, set forth in FAR Section 52.227-14, DFARS 252.227-7013, DFARS 252.227-7014, DFARS 252.227-7015, and DFARS 252.227-7025, as amended from time to time. Contractor/Manufacturer is BMC SOFTWARE INC, 2101 CITYWEST BLVD, HOUSTON TX 77042-2827, USA. Any contract notices should be sent to this address.
Customer support

Support website
You can obtain technical support from BMC 24 hours a day, 7 days a week at http://www.bmc.com/support. From this website, you can:

■ Read overviews about support services and programs that BMC offers
■ Find the most current information about BMC products
■ Search a database for problems similar to yours and possible solutions
■ Order or download product documentation
■ Download products and maintenance
■ Report a problem or ask a question
■ Subscribe to receive proactive e-mail alerts
■ Find worldwide BMC support center locations and contact information, including e-mail addresses, fax numbers, and telephone numbers

Support by telephone or e-mail
In the United States and Canada, if you need technical support and do not have access to the web, call 1 800 537 1813 or send an e-mail message to customer_support@bmc.com. (In the subject line, enter SupID:yourSupportContractID, such as SupID:12345). Outside the United States and Canada, contact your local support center for assistance.

Before contacting BMC
Have the following information available so that Customer Support can begin working on your issue immediately:

■ Product information
  — Product name
  — Product version (release number)
  — License number and password (trial or permanent)
■ Operating system and environment information
  — Machine type
  — Operating system type, version, and service pack or other maintenance level such as PUT or PTF
  — System hardware configuration
  — Serial numbers
  — Related software (database, application, and communication) including type, version, and service pack or maintenance level
■ Sequence of events leading to the problem
■ Commands and options that you used
■ Messages received (and the time and date that you received them)
  — Product error messages
  — Messages from the operating system
  — Messages from related software
License key and password information

If you have questions about your license key or password, contact Customer Support through one of the following methods:

- Send an e-mail message to customer_support@bmc.com. (In the Subject line, enter SupID:yourSupportContractID, such as SupID:12345.)
- In the United States and Canada, call 1 800 537 1813. Outside the United States and Canada, contact your local support center for assistance.
Contents

About this book ..................................................................................................................... 13
Conventions .......................................................................................................................... 13
Related publications .............................................................................................................. 14
Summary of changes ............................................................................................................ 14
Version 11.2.00, May 2015 ................................................................................................. 14

Chapter 1         Introducing High-speed Apply Engine ......................................................... 17
Overview of High-speed Apply Engine features and functions ........................................... 17
How High-speed Apply Engine works .............................................................................. 19
Input sources for apply requests ...................................................................................... 20
   SQL statements ........................................................................................................... 20
   Logical log ................................................................................................................... 21
When to use High-speed Apply Engine ........................................................................... 22
   Transaction recovery ............................................................................................... 22
   Data migration ......................................................................................................... 22
   SQL processing ....................................................................................................... 22
How to use High-speed Apply Engine ............................................................................. 23
Solution integration ............................................................................................................. 23
Considerations for DB2 on mainframe targets ................................................................. 24
   Operational considerations for mainframe DB2 ......................................................... 24
   Static SQL processing .............................................................................................. 25
   Database access for mainframe DB2 ........................................................................ 26
   DB2 authorizations ................................................................................................. 26
   DB2 data type support ............................................................................................ 26
   DBCS support ......................................................................................................... 27
   Sort work data sets .................................................................................................. 27
Considerations for DB2 LUW targets ................................................................................ 27
   Operational considerations for DB2 LUW targets .................................................... 27
   Database access for DB2 LUW targets .................................................................... 28
   DB2 LUW authorizations ....................................................................................... 28
   DB2 LUW data type support ................................................................................. 29
   Template file ........................................................................................................... 29
Considerations for Oracle targets ...................................................................................... 29
   Operational considerations for Oracle targets ......................................................... 30
   Database access for Oracle targets ....................................................................... 30
   Oracle authorizations ............................................................................................. 31
<table>
<thead>
<tr>
<th>Chapter 2</th>
<th>Defining configurations</th>
<th>39</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>Multiple configuration files for DB2 on mainframe targets</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td>Creating and managing configuration files</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Configuration file management</td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Creating or editing a configuration file</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>Configurations for specific tasks</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>Mapping source object names to target objects</td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>Creating logical partitions for target tables</td>
<td>47</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 3</th>
<th>Running High-speed Apply Engine</th>
<th>51</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>51</td>
</tr>
<tr>
<td>Overriding parameters in an apply request</td>
<td>51</td>
<td></td>
</tr>
<tr>
<td>UNIX and Windows command line</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>Mainframe batch job</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>Creating and running apply requests</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Creating and running an apply request for DB2 on mainframe targets</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Creating and running an apply request for DB2 LUW</td>
<td>58</td>
<td></td>
</tr>
<tr>
<td>Creating and running an apply request for Oracle</td>
<td>60</td>
<td></td>
</tr>
<tr>
<td>Where to go from here</td>
<td>62</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 4</th>
<th>Restart and recovery</th>
<th>63</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>63</td>
</tr>
<tr>
<td>Enabling restart processing</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>Creating a restart table</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>Specifying restart parameters</td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>Restarting an apply request</td>
<td>72</td>
<td></td>
</tr>
<tr>
<td>Summary of restart parameter usage</td>
<td>74</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 5</th>
<th>Conflict resolution</th>
<th>77</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>77</td>
</tr>
</tbody>
</table>
Chapter 6  Configuration file format

Overview of the configuration file's format .................................................. 93
  Sections in configuration files ................................................................. 96
  Format of configuration parameters ....................................................... 97
  Comments in configuration files ............................................................. 98
  Data set and file names in configuration files ....................................... 98
  Object names in configuration files ........................................................ 99

Chapter 7  Configuration parameters

Overview ........................................................................................................ 103
  Quick reference by name ......................................................................... 103
  Quick reference by section ....................................................................... 109
  Agent parameters .................................................................................... 116
    InitialAgents ......................................................................................... 117
    MaxAgents .......................................................................................... 119
    MaxPrepares ....................................................................................... 120
  AnyConflict parameters ........................................................................ 122
    Action .................................................................................................. 123
    Code .................................................................................................... 124
  Bind parameters ..................................................................................... 124
    AuthID ................................................................................................. 125
    BindAction ......................................................................................... 126
    BindOwner ......................................................................................... 129
    CollectionID ..................................................................................... 130
    DBPROTOCOL .................................................................................... 131
    DISP ................................................................................................. 132
    FreeOption ........................................................................................ 133
    Library ............................................................................................... 134
    PackageName ..................................................................................... 135
    PackagePrefix ................................................................................... 136
    PlanName ........................................................................................... 136
    PlanPrefix .......................................................................................... 137
  BindTuning parameters .......................................................................... 138
    Using [BindTuning] parameters for performance ............................. 139
    MaxFailedBinds ................................................................................. 142
    MaxPackages ...................................................................................... 143
<table>
<thead>
<tr>
<th>Parameter Type</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DistributionTuning parameters</td>
<td>182</td>
</tr>
<tr>
<td>Default distribution of work</td>
<td>182</td>
</tr>
<tr>
<td>Precedence for distribution of work by object</td>
<td>183</td>
</tr>
<tr>
<td>MaxExecuteImmediateDML</td>
<td>183</td>
</tr>
<tr>
<td>PartitionClustering</td>
<td>184</td>
</tr>
<tr>
<td>RIClustering</td>
<td>186</td>
</tr>
<tr>
<td>RIRecursionLimit</td>
<td>187</td>
</tr>
<tr>
<td>SimpleTSClustering</td>
<td>189</td>
</tr>
<tr>
<td>URCheckRI</td>
<td>189</td>
</tr>
<tr>
<td>ErrorOutput parameter</td>
<td>191</td>
</tr>
<tr>
<td>FileName (EFN)</td>
<td>191</td>
</tr>
<tr>
<td>InsertConflict parameters</td>
<td>192</td>
</tr>
<tr>
<td>LogicalLog parameters</td>
<td>193</td>
</tr>
<tr>
<td>CCSIDCompatible</td>
<td>194</td>
</tr>
<tr>
<td>IncludeDDL</td>
<td>195</td>
</tr>
<tr>
<td>IncludeLOB</td>
<td>195</td>
</tr>
<tr>
<td>IncludeRI</td>
<td>197</td>
</tr>
<tr>
<td>IncludeTrigger</td>
<td>198</td>
</tr>
<tr>
<td>IncludeXML</td>
<td>199</td>
</tr>
<tr>
<td>IncludeHistory</td>
<td>200</td>
</tr>
<tr>
<td>Qualify</td>
<td>201</td>
</tr>
<tr>
<td>PartitionDistribution</td>
<td>202</td>
</tr>
<tr>
<td>Sort</td>
<td>203</td>
</tr>
<tr>
<td>SQLType</td>
<td>204</td>
</tr>
<tr>
<td>UpdateColumns</td>
<td>205</td>
</tr>
<tr>
<td>UseLLogSysStrings</td>
<td>206</td>
</tr>
<tr>
<td>WORKNUM</td>
<td>207</td>
</tr>
<tr>
<td>LogOutput parameters</td>
<td>208</td>
</tr>
<tr>
<td>FileName (LFN)</td>
<td>208</td>
</tr>
<tr>
<td>LogType</td>
<td>209</td>
</tr>
<tr>
<td>Overlap</td>
<td>210</td>
</tr>
<tr>
<td>Threshold</td>
<td>211</td>
</tr>
<tr>
<td>MonitorTuning parameters</td>
<td>212</td>
</tr>
<tr>
<td>CommitTrigger</td>
<td>213</td>
</tr>
<tr>
<td>CommitValue</td>
<td>214</td>
</tr>
<tr>
<td>InputTrigger</td>
<td>215</td>
</tr>
<tr>
<td>inputValue</td>
<td>216</td>
</tr>
<tr>
<td>ScanInterval</td>
<td>217</td>
</tr>
<tr>
<td>MultiRowInsert parameters</td>
<td>218</td>
</tr>
<tr>
<td>MaxConflicts</td>
<td>219</td>
</tr>
<tr>
<td>MaxRows</td>
<td>220</td>
</tr>
<tr>
<td>MultiRowInsert</td>
<td>.................................................................</td>
</tr>
<tr>
<td>ObjectCluster parameters</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Cluster</td>
<td>.................................................................</td>
</tr>
<tr>
<td>Table</td>
<td>.................................................................</td>
</tr>
<tr>
<td>ObjectMap parameters</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Positional requirements for object mapping</td>
<td>.................................................................</td>
</tr>
<tr>
<td>Using wildcards to map object names</td>
<td>.................................................................</td>
</tr>
<tr>
<td>SourceColumn</td>
<td>........................................................................</td>
</tr>
<tr>
<td>SourceTable</td>
<td>........................................................................</td>
</tr>
<tr>
<td>TargetColumn</td>
<td>........................................................................</td>
</tr>
<tr>
<td>TargetTable</td>
<td>........................................................................</td>
</tr>
<tr>
<td>ObjectPartition parameters</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Positional requirements for ObjectPartition parameters</td>
<td>.................................................................</td>
</tr>
<tr>
<td>Partition key length</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Considerations for Oracle data types</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Column</td>
<td>........................................................................</td>
</tr>
<tr>
<td>LimitKey</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Order</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Partition</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Table</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Value</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Restart parameters</td>
<td>........................................................................</td>
</tr>
<tr>
<td>CheckpointFrequency</td>
<td>........................................................................</td>
</tr>
<tr>
<td>RestartID</td>
<td>........................................................................</td>
</tr>
<tr>
<td>RestartType</td>
<td>........................................................................</td>
</tr>
<tr>
<td>RetainTime</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Retry</td>
<td>........................................................................</td>
</tr>
<tr>
<td>TableName</td>
<td>........................................................................</td>
</tr>
<tr>
<td>SQL parameters</td>
<td>........................................................................</td>
</tr>
<tr>
<td>DecimalPoint</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Format</td>
<td>........................................................................</td>
</tr>
<tr>
<td>StartUp parameters</td>
<td>........................................................................</td>
</tr>
<tr>
<td>BindQualifier</td>
<td>........................................................................</td>
</tr>
<tr>
<td>ConnectionVersion</td>
<td>........................................................................</td>
</tr>
<tr>
<td>Database</td>
<td>........................................................................</td>
</tr>
<tr>
<td>DateFormat</td>
<td>........................................................................</td>
</tr>
<tr>
<td>DBMSType</td>
<td>........................................................................</td>
</tr>
<tr>
<td>DefaultOpts</td>
<td>........................................................................</td>
</tr>
<tr>
<td>DistributionType</td>
<td>........................................................................</td>
</tr>
<tr>
<td>FileName (IFN)</td>
<td>........................................................................</td>
</tr>
<tr>
<td>InputMediaType</td>
<td>........................................................................</td>
</tr>
</tbody>
</table>
Contents

Appendix A    Logical log input  281
Logical log input considerations .................................................. 281

Appendix B    Installing High-speed Apply Engine on mainframe targets  285
Installation overview and prerequisites .................................... 285
DB2 authorizations for plans, packages, and collections .............. 286
   Using the user ID running High-speed Apply Engine ................ 287
   Using BindOwner and a pre-bound plan ................................. 288
   Using a pre-bound plan and package ................................... 290
   Using the AuthId parameter .............................................. 292
   Summary of DB2 authorization requirements ......................... 293
Installation options overview .................................................... 295
Generating installation options ................................................... 297

Appendix C    Installing High-speed Apply Engine on UNIX or Windows  299
Overview and prerequisites ....................................................... 299
   Installation considerations ............................................... 300
About this book

This book contains detailed product information and is intended for system administrators and database administrators (DBAs).

Like most BMC documentation, this book is available in printed and online formats. To request printed books or to view online books and notices (such as release notes and technical bulletins), see the support website at http://www.bmc.com/support.

**Note**

Online books are formatted as PDF or HTML files. To view, print, or copy PDF books, use the free Adobe Reader from Adobe Systems. If your product installation does not install the reader, you can obtain the reader at http://www.adobe.com.

The software also offers online Help. To access Help, press F1 within any product or click the Help button in graphical user interfaces (GUIs).

Conventions

This document uses the following special conventions:

- All syntax, operating system terms, and literal examples are presented in this typeface.

- Variable text in path names, system messages, or syntax is displayed in italic text: `testsys/instance/fileName`

- Menu sequences use a symbol to convey the sequence. For example, Actions => Create Test instructs you to choose the Create Test command from the Actions menu.
Related publications

From the BMC Support Central website, you can use the following methods to access related publications that support your product or solution:


- View Quick Course videos (short overviews of selected product concepts, tasks, or features), which are available from the following locations:
  — Documentation Center (primary center and secured center)
  — Support Central (at http://www.bmc.com/support/mainframe-demonstrations)
  — BMC Mainframe YouTube channel (https://www.youtube.com/user/BMCSoftwareMainframe)


You can order hardcopy documentation from your BMC sales representative or from the support site. You can also subscribe to proactive alerts to receive e-mail messages when notices are issued.

Tip

You can access the BMC Support Central site at http://www.bmc.com/support.

Summary of changes

This section summarizes changes to the functionality of the product, listing the changes by product version and release date. The summary includes enhancements to the product and any major changes to the documentation.

Version 11.2.00, May 2015

This release fixes known problems and includes the following enhancements.
Support for BIGINT and DECFLOAT

High-speed Apply Engine now supports BIGINT and DECFLOAT data types for DB2 LUW.

Support for BINARY_FLOAT and BINARY_DOUBLE

High-speed Apply Engine now supports BINARY_FLOAT and BINARY_DOUBLE data types for Oracle.

Simplified installation wizard for UNIX and Windows

You can now use a single, simplified installation wizard to install High-speed Apply Engine on either a Microsoft Windows system or a UNIX system.

See “Installing High-speed Apply Engine on UNIX or Windows” on page 299.
Introducing High-speed Apply Engine

This chapter explains how the High-speed Apply Engine component works and when you can use it. This chapter also includes platform-based operational considerations for using High-speed Apply Engine.

Overview of High-speed Apply Engine features and functions

The High-speed Apply Engine is a component that is distributed with several solutions and products from BMC Software, including Log Master for DB2. High-speed Apply Engine enables you to rapidly update a DB2, DB2 LUW (previously Universal Database (UDB)), or Oracle database and to control apply processing with flexible configuration options.

High-speed Apply Engine runs on the following database management systems (DBMSs) and operating system platforms:

- IBM DB2 LUW on mainframe operating systems (such as z/OS)
- IBM DB2 LUW on UNIX
- IBM DB2 LUW on Microsoft Windows
- Oracle on UNIX
- Oracle on Windows

As a component, High-speed Apply Engine provides the following functionality to BMC products:

- The execution of generated SQL statements for Log Master for DB2.
The LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY load action (referred to as an SQLAPPLY load action) for LOADPLUS for DB2.

The High-speed Apply Engine provides many features in addition to those used by these products. To use any of these additional features, run High-speed Apply Engine independently of the accompanying product. When High-speed Apply Engine is distributed with BMC Software products, installation of the products includes installation of High-speed Apply Engine.

The High-speed Apply Engine accepts the following sources as input for an apply request:

- SQL statements (including SQL generated by Log Master for DB2 or any standard SQL processor)
- Logical log files created by Log Master for DB2

High-speed Apply Engine provides the following advanced features to improve the performance of your apply request:

- Multi-threaded execution: Use multiple agents to increase parallelism and improve performance.
- Restart processing: If an apply request fails, continue processing from the point of failure. Avoid repeating transactions that have already been processed.
- Conflict resolution rules: Use flexible parameters to define precise responses to SQL codes or database messages during execution. Customize the apply request to match your data with a variety of responses, including the ability to write a statement or unit of recovery to a separate file for subsequent examination or execution.

Using SQL or logical log input, you can use High-speed Apply Engine for the following tasks:

- Apply changes to a database as part of an apply request
- Migrate data between databases
- (DB2 only) Include an apply request as part of a transaction recovery process for a DB2 database

The apply request runs as a batch job on the operating system. This implementation allows you to schedule requests as part of your normal recovery process. During the recovery process, you can use SQL to reverse (undo) the effects of incorrect transactions, or to re-apply (redo) correct transactions after performing a DB2 recovery.

When conflicts occur during the recovery process, you can use conflict resolution rules to specify a different resolution for each type of conflict that might occur.
How High-speed Apply Engine works

The High-speed Apply Engine runs as a command from a UNIX or Windows command line, or as a program step in a mainframe batch job.

Figure 1 on page 19 shows an overview of the High-speed Apply Engine data flow. The numbered steps for the data flow are described following the figure.

Figure 1: High-speed Apply Engine data flow

When you submit an apply request, High-speed Apply Engine processes it as follows:

1. The request processor manages the apply request.

2. The request processor also obtains configuration information for the apply request and determines the processing requirements from that information. You can
specify configuration information in the apply request or in a separate
configuration file.

3 The input processor begins processing the input source that is specified in the
configuration information.

4 If the apply request specifies restart processing, the input processor uses
information in the restart table and the configuration file to determine the location
of the first record to be processed in the input source.

5 The input processor passes SQL statements to an apply agent. Each agent
processes the statements for an object, set of objects (tables or partitions), or a unit
of recovery (UR). You specify the maximum number of agents in the
configuration information.

6 The apply agents execute the SQL statements against the target tables.

7 The apply agents keep track of the changes that have been applied, display
statistics about the request, and update the restart table.

8 If conflicts occur during apply processing, the apply agent coordinates conflict
resolution with the conflict manager. The conflict manager uses the conflict
resolution rules in the configuration to determine the appropriate action.

9 If you configure the apply request to include defer actions and a conflict file, the
conflict manager writes the deferred conflicts to the conflict file.

Input sources for apply requests

This section describes the SQL and logical log input sources that High-speed Apply
Engine uses.

For more information about specifying the input sources in a configuration or an
apply request, see “Defining configurations” on page 39 or “Running High-speed
Apply Engine” on page 51.

SQL statements

You can specify an input file that contains SQL statements.

The SQL can be generated by the following sources:

- Log Master for DB2
other standard SQL generators

SQL input can include data manipulation language (DML) or data definition language (DDL) statements. High-speed Apply Engine can process any DML or DDL statements that are accepted by the target DBMS. To provide restart capability, High-speed Apply Engine requires that SQL input does not contain more than one SQL statement on the same line.

The Log Master for DB2 product generates a separate template file that High-speed Apply Engine can use to efficiently process SQL input as static SQL against DB2 for mainframe targets or DB2 LUW targets. The template file contains one entry for each distinct statement type in the SQL input. If an apply request includes a template file, High-speed Apply Engine processes the template file before it begins processing SQL input.

Logical log

The High-speed Apply Engine running on mainframe operating systems accepts logical logs as an input source. Log Master for DB2 produces logical logs that let you specify the types of operations that you need to perform. The logical log files provide information that High-speed Apply Engine can use to update the target tables.

Logical log input consists of the following files:

- The logical log control file contains information about the format and content of the logical log. Specify the control file as the input for High-speed Apply Engine.

- The logical log data file contains actual change information (updates, deletes, inserts, or DDL objects) in logical log format.

- (optional) The large volume VSAM files contain data from DB2 columns that hold large volumes of data (such as XML columns or LOB columns). To apply changes to large volume columns, the appropriate VSAM files must be available to High-speed Apply Engine along with the control file and the data file.

- (optional) The logical log XMLSTRING control file contains the string IDs and string data that DB2 uses to encode the data in XML columns. If the logical log includes XML data and the target DB2 subsystem is different than the subsystem where the logical log was generated, High-speed Apply Engine needs the string IDs and data to serialize your XML data correctly.

High-speed Apply Engine can often process a logical log file faster than an SQL file that contains the same database changes. This improvement occurs because logical log input is already in host variable format. High-speed Apply Engine can also avoid parsing SQL syntax.
You can create logical logs to perform data migration, UNDO SQL functions, or REDO SQL functions. For more information on the logical log records that High-speed Apply Engine uses, see “Logical log input” on page 281. For more information about logical logs, see the chapter about logical log files in the Log Master for DB2 Reference Manual.

When to use High-speed Apply Engine

The High-speed Apply Engine can improve the performance of operations that require large numbers of updates to your DB2, DB2 LUW, or Oracle databases, including transaction recovery, data migration, and SQL processing.

Transaction recovery

You can use High-speed Apply Engine as part of your recovery strategy for DB2, DB2 LUW, and Oracle databases. You can create an input file for High-speed Apply Engine to use to perform undo and redo processing against the target tables.

For DB2, you can include High-speed Apply Engine as part of the recovery tasks that you develop with the Recovery Management for DB2 solution from BMC Software.

Data migration

When you create the target tables for data migration, you can use High-speed Apply Engine to perform the row updates required to move the data to the target tables.

Because you can process the updates to the target table without stopping the database, you can migrate data without waiting for a recovery window. You can translate table and column names as you run the apply request, but you cannot transform the content of a column from one data type to another.

SQL processing

If you have routine operations that require you to process large numbers of SQL statements, you can use High-speed Apply Engine to speed the process.

Consider High-speed Apply Engine for operations that you perform by using your native database SQL processors (for example, SPUFI, DSNTEP2, DB2CLP, DB2SQL89, DB2BATCH, or SQL*Plus).
How to use High-speed Apply Engine

To apply updates to a target database, the High-speed Apply Engine requires

- an input source that contains the updates
- configuration information that describes how the updates should be processed

To use High-speed Apply Engine, you perform the tasks listed in the following table.

<table>
<thead>
<tr>
<th>Task</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Create an input file for the apply request.</td>
<td>“Input sources for apply requests” on page 20</td>
</tr>
<tr>
<td>2. Define one or more configuration files.</td>
<td>“Defining configurations” on page 39  “Configuration parameters” on page 103</td>
</tr>
</tbody>
</table>
| 3. Create and run the apply request.  
**Note:** You can also include configuration information in the apply request, overriding the settings in the configuration files. | “Running High-speed Apply Engine” on page 51 |
| (optional) Enable restart processing. | “Restart and recovery” on page 63 |

Solution integration

The High-speed Apply Engine is a component of Log Master for DB2. In turn, Log Master for DB2 (including High-speed Apply Engine) is a component of the Recovery Management for DB2 solution.

The Recovery Management for DB2 solution integrates the features of:

- RECOVERY MANAGER for DB2
- RECOVER PLUS for DB2
- COPY PLUS for DB2
- Log Master for DB2
- SNAPSHOT UPGRADE FEATURE
When you acquire this solution, you benefit from all features of these individual products and from more features that are available when one Recovery Management component can rely on the presence of all others. For more information, see the *Recovery Management for DB2 User Guide*.

Considerations for DB2 on mainframe targets

This section describes requirements and considerations for using the High-speed Apply Engine against DB2 on mainframe operating systems.

For more information about installing High-speed Apply Engine, see “Installing High-speed Apply Engine on mainframe targets” on page 285.

Operational considerations for mainframe DB2

When you use High-speed Apply Engine, consider these constraints and recommendations:

- **Authorization requirements**
  The following libraries must be APF authorized:
  - High-speed Apply load libraries
  - Any libraries that you reference in the apply request (in the STEPLIB DD statements)

  Also, the user ID that submits the apply request must have the appropriate authorizations to run the request. For more information, see “DB2 authorizations” on page 26.

- **Operating system requirements**
  If the input source for the apply request is a data set, the data set must be present in the ICF catalog of the operating system.

  High-speed Apply Engine must be installed on the same operating system logical partition (LPAR) as the target DB2 subsystem.

- **DB2 thread requirements**
  The target DB2 subsystem must have enough threads available to process the apply request. Ensure that at least two threads are available, or as many threads as the initial number of agents that High-speed Apply Engine specifies. For more information about setting the initial number of agents, see “InitialAgents” on page 117. For more information about the maximum number of agents, see “MaxAgents” on page 119.
Restart table considerations

A new installation of High-speed Apply creates a new restart table. High-speed Apply does not migrate restart tables from previous versions. If you have an earlier version of High-speed Apply Engine installed, any restarts must be completed on the original version before switching to a new version.

The restart table for an apply request must reside on the same DB2 subsystem as the target tables.

LOADPLUS requirements

If High-speed Apply Engine is called by the LOADPLUS for DB2 product, then High-speed Apply Engine must be installed on the same DB2 subsystem as LOADPLUS, and must be available through the STEPLIB, JOBLIB, or LINKLIST when LOADPLUS is executed.

Static SQL processing

To maximize performance, the High-speed Apply Engine processes SQL statements statically as much as possible, within limits set by your configuration parameters.

*Note*

Static SQL processing pertains to DB2 and DB2 LUW only.

As it processes apply requests, High-speed Apply Engine takes the following actions:

- Generates plans or packages dynamically
- Issues DB2 BIND commands to bind the plans or packages
- Executes the statements as static SQL using the bound plans or packages. (For DB2 LUW targets on UNIX or Windows, High-speed Apply Engine binds only packages.)

If it cannot process an SQL statement statically, High-speed Apply Engine processes it with dynamic SQL or by using EXECUTE IMMEDIATE. You can configure an apply request to dynamically allocate DB2 plans or packages by specifying parameters in the [Bind] section of the configuration file.

Depending on limits that are set by the parameters in the [BindTuning] section, High-speed Apply Engine prepares and executes SQL statements dynamically as it builds database request modules (DBRMs). High-speed Apply Engine switches to static execution after the bind action occurs.
Database access for mainframe DB2

To run an apply request, the High-speed Apply Engine updates the target tables through DB2. The database can continue to run while High-speed Apply Engine performs its updates, so you can update the target tables while they are online.

The access method for DB2 is the Call Attach Facility (CAF). Because High-speed Apply Engine uses multiple connections to the database, the target DB2 subsystem must have enough threads available for the apply request (for more information, see “Operational considerations for mainframe DB2” on page 24).

DB2 authorizations

To execute SQL or logical log input, the user ID that runs the High-speed Apply Engine must have the following DB2 authorizations:

- EXECUTE privilege for the plan that High-speed Apply Engine uses to access its own restart tables and the catalog (see “PublicPlan” on page 266).

- EXECUTE privilege for the restart package.

- appropriate table privileges such as INSERT, UPDATE, or DELETE for the target tables (the specific privileges depend on the actions that the apply request performs)

- appropriate privileges to bind or administer plans, packages, and collections

High-speed Apply Engine provides several methods to grant these privileges. Some techniques avoid granting bind privileges to the user ID that runs High-speed Apply Engine. For more information, see “DB2 authorizations for plans, packages, and collections” on page 286.

DB2 data type support

The High-speed Apply Engine supports the same SQL data types and data type conversions as DB2.

For more information about compatible data types, consult the appropriate DB2 documentation.
DBCS support

The High-speed Apply Engine supports double-byte characters in logical log input and SQL input. High-speed Apply Engine can read logical log files that are created by Log Master for DB2, or an application program that conforms to the published format.

For more information about the logical log format, see the section about logical logs in the Log Master for DB2 technical documentation.

Sort work data sets

The High-speed Apply Engine sorts logical log input (subject to your control).

Under normal circumstances, you do not have to specify any sort parameters or sort work data sets. For more information, see “Sort” on page 203. If you want to specify sort work data sets for performance reasons or because of procedures in your environment, the DD names of sort work data sets must adhere to the following convention:

SW01WKyy

The variable yy represents the sequence number of a sort work data set. It ranges from one up to the maximum number of sort work data sets that High-speed Apply Engine requires. You can specify an exact number of data sets by using the WORKNUM parameter. For more information, see “WORKNUM” on page 207.

Considerations for DB2 LUW targets

This section describes requirements and considerations for using the High-speed Apply Engine for DB2 LUW on the UNIX and Windows platforms.

For more information about installing High-speed Apply Engine, see “Installing High-speed Apply Engine on UNIX or Windows” on page 299.

Operational considerations for DB2 LUW targets

When you use High-speed Apply Engine, consider these constraints and recommendations:
For optimal performance, High-speed Apply Engine should be installed on the DB2 LUW server that contains the target tables.

File names that use mixed case or lowercase are processed exactly as they appear. If your system is case sensitive, specify the file names in the appropriate case. For more information, see “Data set and file names in configuration files” on page 98.

Table names that use mixed case or lowercase must be delimited by quotation marks. For more information, see “Object names in configuration files” on page 99.

Unless the target DB2 LUW database allows automated login, you must provide a user ID and password as part of the configuration. For more information about specifying these values, see “Password” on page 265 and “UserID” on page 271.

The user ID that submits the apply request must have appropriate privileges on target tables. For more information, see “DB2 LUW authorizations” on page 28.

Database access for DB2 LUW targets

Using information from the configuration files and the apply request, the High-speed Apply Engine updates the DB2 LUW target tables. The database can continue to run while High-speed Apply Engine performs its updates, so you can update the target tables while they are online.

DB2 LUW authorizations

You must have the following DB2 LUW authorizations to run the High-speed Apply Engine:

- On the target tables, bind privileges and table privileges (INSERT, UPDATE, and DELETE).
- On the restart table, bind privileges and table privileges (INSERT, UPDATE, DELETE, and SELECT).

These authorizations can be granted during installation or after, but you must have these authorizations before you can run High-speed Apply Engine. The privileges that High-speed Apply Engine requires depend on the transactions that the apply request performs on the target tables.
DB2 LUW data type support

The High-speed Apply Engine supports the same data types and data type conversions as DB2 LUW.

For more information about compatible data types, consult the appropriate DB2 LUW documentation.

Template file

A template file is a separate file that is generated by Log Master for DB2 that the High-speed Apply Engine can use to efficiently process SQL input as static SQL against DB2 for mainframe targets or DB2 LUW targets.

For DB2 LUW targets, perform the following steps to use the template file:

- transfer (download) the SQL file and the corresponding template file to the target DB2 LUW server

- edit the SQL file

  The first statement in the SQL file is a comment that indicates the corresponding template file, as in the following example:

  ```
  -- @@DSNTMPL mainframeFileName
  ```

  Replace the existing file name (the name that was used when the template file was first generated by Log Master for DB2) with the fully qualified path name of the template file on the target DB2 LUW server.

- run an apply request that uses the SQL file as input

Considerations for Oracle targets

This section describes requirements and considerations for using the High-speed Apply Engine for Oracle on the UNIX and Windows platforms.

For more information about installing High-speed Apply Engine, see “Installing High-speed Apply Engine on UNIX or Windows” on page 299.
Operational considerations for Oracle targets

When you use High-speed Apply Engine, consider these constraints and recommendations:

- For optimal performance, High-speed Apply Engine should be installed on the Oracle server that contains the target tables.

- File names that use mixed case or lowercase are processed exactly as they appear. If your system is case-sensitive, be sure to specify the file names in the appropriate case. For more information, see “Data set and file names in configuration files” on page 98.

- Table names that use mixed case or lowercase must be delimited by quotation marks. For more information, see “Object names in configuration files” on page 99.

- Unless the target Oracle database allows automated login, you must provide a user ID and password as part of the configuration. For more information about specifying these values, see “Login” on page 263.

- The user ID that submits the apply request must have appropriate privileges on the target tables. For more information, see “Oracle authorizations” on page 31.

- The value that you specify for the MaxPrepares parameter cannot exceed the value of the OPEN_CURSORS initialization parameter for your Oracle session. If you specify a value that is greater than OPEN_CURSORS, High-speed Apply Engine uses the OPEN_CURSORS value.

Database access for Oracle targets

Using information from the configuration files and the apply request, the High-speed Apply Engine updates the Oracle target tables. The database can continue to run while High-speed Apply Engine performs its updates, so you can update the target tables while they are online.

High-speed Apply Engine uses the Oracle Call Interface (OCI) to access Oracle databases. For some operations (such as dynamically determining the version of the Oracle client), High-speed Apply Engine uses the SQL*Plus application. The directory that contains SQL*Plus must be included in the PATH for the system, so that High-speed Apply Engine can run the program.
Oracle authorizations

To run the High-speed Apply Engine you must have the following Oracle authorizations:

- table privileges (INSERT, UPDATE, or DELETE) on the target tables
- table privileges (INSERT, UPDATE, DELETE, and SELECT) on the restart table

These authorizations can be granted during installation or after, but you must have these authorizations before you can run High-speed Apply Engine. The privileges that High-speed Apply Engine requires depend on the transactions that the apply request performs on the target tables.

Oracle data type support

The High-speed Apply Engine supports the same SQL data types and data type conversions as Oracle. For more information about compatible data types, consult the appropriate Oracle documentation.

If you want to create logical partitions for your target tables, be aware of some restrictions on the data types and functions that can be used as partition keys. For more information, see “Considerations for Oracle data types” on page 233.

Distribution of work

The High-speed Apply Engine uses two basic techniques to balance the workload for an apply request between multiple agents: distribution by object (table or partition) and distribution by unit of recovery (UR).

High-speed Apply Engine uses default values to choose between these techniques. To override the defaults and optimize distribution between agents for your environment, use configuration parameters. For more information about default values and which techniques take precedence over others, see “Distribution Tuning parameters” on page 182.

Distribution by object

The High-speed Apply Engine can distribute work between agents based on the tables or partitions that the SQL statements in the input file update.
High-speed Apply Engine uses the same agent to process all SQL statements that affect the same table or partition, regardless of what units of recovery contain the statements (unless High-speed Apply Engine regroups tables because of RI relationships). High-speed Apply Engine offers several options for this type of distribution (for more information, see “Options for distribution by object” on page 32). High-speed Apply Engine might also use the same agent for all tables in RI relationships (subject to parameters that you can adjust).

- **Advantages of distribution by object**: High-speed Apply Engine can increase parallelism and use multiple agents more efficiently. This type of distribution frequently provides faster performance.

- **Disadvantages of distribution by object**: High-speed Apply Engine provides data integrity for the overall apply request, but does not preserve transaction integrity during execution. With this technique, High-speed Apply Engine does not honor the original COMMIT or ROLLBACK statements (High-speed Apply Engine commits work based on other parameters that you can define). You can encounter more conflicts and you might need to define more conflict resolution rules.

Be aware of the following points regarding distribution by object:

- Use the [CommitTriggers] parameters to control when High-speed Apply Engine commits work. The values of these parameters greatly affect performance.

- To request that High-speed Apply Engine not honor database-defined RI relationships between tables or partitions, use the RIClustering parameter.

- Distribution is based on the base table when an alias or single table view is used to identify the table_name in the SQL statement (for example, `INSERT INTO table_name (col1, col2) values ("xxx", "yyy");`).

### Options for distribution by object

High-speed Apply Engine provides the following options for distribution by object:

- **groups of tables**

  High-speed Apply Engine distributes work between agents based on groups of tables. You can define your own groups or use existing groups that are defined within DB2. High-speed Apply Engine uses the same agent to process all SQL statements that affect the tables in a group. High-speed Apply Engine supports the following techniques to define groups of tables:

  — groups of tables that are defined by users

  Use the [ObjectCluster] parameters to define your own groups of tables. This option provides the most flexibility; you can respond to changing data distribution by changing table groups for each run of an apply request.
— groups of tables that are defined by RI relationships
  By default, High-speed Apply Engine groups tables based on database-defined RI relationships (subject to parameters that you can adjust). For more information, see “RIRecursionLimit” on page 187. This type of distribution preserves RI for all updates that the apply request make. Use the RIClustering parameter to avoid this grouping. For more information, see “RIClustering” on page 186.

— groups of tables that are defined by simple table spaces
  By default, High-speed Apply Engine includes all tables in a simple (nonsegmented) table space in the same group and uses the same agent to process all SQL statements for those tables. Use the SimpleTSClustering parameter to avoid this kind of table grouping. For more information, see “SimpleTSClustering” on page 189.

■ partitions
  High-speed Apply Engine distributes work between agents based on table partitions. You can define your own partitions for a nonpartitioned table, tell High-speed Apply Engine to use partitions that are different than the existing partitions of a table, or use the existing partitions of a table. High-speed Apply Engine uses the same agent to process all SQL statements that affect a partition that you define or use.

  High-speed Apply Engine supports the following techniques to define partitions:

— partitions that are defined by users
  Use the [ObjectPartition] parameters (“ObjectPartition parameters” on page 230) to define your own partitions for nonpartitioned or partitioned tables. (This process is sometimes referred to as logical partitioning.) This option provides the most flexibility; you can respond to changing data distribution by changing your partitioning for each run of an apply request.

— partitions that are defined by partitioned table spaces
  Use the PartitionClustering parameter (“PartitionClustering” on page 184) to direct High-speed Apply Engine to use the same agent to process all SQL statements that affect the same physical partition of a table.

■ individual tables
  High-speed Apply Engine uses the same agent to process all SQL statements that affect an individual table in your input file. High-speed Apply Engine defaults to this option only when

— you do not activate any of the other options for distribution by object

— the tables in your input file are not part of database-defined RI relationships (or you disable RI clustering)
— the tables in your input file are not stored in a simple table space (or you disable simple table space clustering)

— the tables in your input file are not partitioned tables

**Distribution by unit of recovery**

The High-speed Apply Engine can distribute work between agents based on the units of recovery (transactions) contained in the input file.

High-speed Apply Engine uses the same agent to process all SQL statements in an original transaction, regardless of what tables or partitions the statements update. (Two different transactions that update the same tables might not be processed by the same agent.) High-speed Apply Engine accommodates dependencies between transactions and preserves the order of the original transactions for restart purposes. For more information, see “DistributionType” on page 258.

- **Advantages of distribution by unit of recovery (UR):** High-speed Apply Engine preserves the original transactions and honors the original COMMIT or ROLLBACK statements. This preserves transaction integrity during an apply request. High-speed Apply Engine performs faster when there are few dependencies between transactions (for example, when the transactions do not update the same tables).

- **Disadvantages of distribution by unit of recovery:** Performance can be slower when there are dependencies between transactions, because the different agents wait more frequently to accommodate the dependencies and preserve the original order of transactions.

Use this type of distribution when transaction integrity during the apply request is more important than speed. Be aware of the following points regarding distribution by UR:

- Under distribution by UR, configuration parameters that are related to distribution by object do not apply (such as [CommitTriggers], [ObjectPartition], and many [DistributionTuning] parameters, such as PartitionClustering and RIClustering).

- Under distribution by UR, High-speed Apply Engine is more likely to use the maximum number of agents that are available (depending on the number of transactions in your input and the dependencies between them). Each agent requires a separate connection to the database (for DB2 on mainframe targets, a separate thread on the DB2 subsystem). Use the MaxAgents parameter to prevent High-speed Apply Engine from using more database connections, and by extension, more resources than your environment can tolerate. For more information, see “MaxAgents” on page 119.
You can control how High-speed Apply Engine processes referential integrity (RI) dependencies between transactions. When your input contains many RI dependencies, consider using the URCheckRI parameter (for more information, see “URCheckRI” on page 189). When the input contains fewer RI dependencies, consider using a conflict resolution rule with Code parameter value RIConflict and Action parameter value Retry.

Conflict resolution

A number of different conflicts can occur when you update the data in a table. The High-speed Apply Engine lets you specify how to handle the conflicts as they occur.

Based on the type of conflict, you can choose to

- ignore the conflict
- defer resolution until the job is complete
- stop apply processing until the conflict is resolved
- retry an SQL statement that generates a time-related conflict (such as time out conditions or RI-related conflicts)

For more information about conflict resolution, see “Conflict resolution” on page 77.

Restart processing

If the DBMS or operating system fails during apply processing, the High-speed Apply Engine maintains its position in the request that was running when the failure occurred.

When you restart the request, High-speed Apply Engine uses information in the restart table to automatically resume processing. To enable restart processing, you must perform the following tasks:

- include the appropriate [Restart] parameters in the configuration for the apply request

For more information about these parameters, see “Summary of restart parameter usage” on page 74 and “Restart parameters” on page 239.
create one or more restart tables

By default, High-speed Apply Engine uses a single restart table for all apply requests. However, if you run many apply requests simultaneously on the same target database, you might experience resource contention for the single restart table.

If you experience contention, create a restart table for each user that runs apply requests simultaneously on the target database. This practice reduces the possibility of contention for the restart table. For more information about creating restart tables, see “Creating a restart table” on page 64.

rerun the apply request that was in-process at the point of failure by specifying the appropriate restart ID for the request

For more information, see “Restarting an apply request” on page 72.

To optimize restart processing, use configuration parameters to adjust the checkpoint frequency, the number of agents, or the COMMIT actions of an apply request.

File transfer guidelines

Use the following guidelines when preparing and sending SQL files that Log Master for DB2 generates on the mainframe to a Windows or Unix platform for execution by High-speed Apply Engine.

Define the SQL output data set as VB if the generated SQL statements cannot exceed 27 KB in length. For example:

```
SQL MIGRATE
DATASET MY.MIGRATE.SQL NEW CYLINDERS SPACE(50,20) UNIT(SYSDA) RELEASE
RECFM VB
```

If minimizing the elapsed time for the file transmission is important, consider not generating and sending a template file. Using the template file increases the SQL file size by 25 percent or more, which affects transmission time and DASD space usage. If High-speed Apply Engine does not use a template file, some SQL will be executed in dynamic mode before binds occur that allow execution in static mode.

If you are using File Transfer Protocol (FTP), include the following command to ensure that trailing blanks are removed:

```
quote site trail
```

If you are using FTP to send the SQL file to a Windows server, include the following command to ensure that High-speed Apply Engine recognizes the end-of-line character:

```
quote site sbsendeol=lf
```
After sending the file, ensure that all characters were translated correctly from EBCDIC to ASCII. You might need to adjust the translate tables used in the data transfer.

If the SQL output was generated with RECFM set to FB (the default), include the SQL parameter Format with a value of Packed72 in the High-speed Apply Engine configuration file, as shown in the following example:

```
[SQL]
Format=Packed72
```

For more information about the Format parameter, see “Format” on page 250.

**File transfer examples**

The following figure shows sample JCL for running FTP in batch mode on the mainframe:

**Figure 2: JCL for running FTP in batch mode**

```c
//FTP EXEC PGM=FTP,REGION=0M
//OUTPUT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=* 
//OUTPUT DD SYSOUT=* 
//INPUT DD *      
yourServerName
yourLoginID
yourPassword
cd yourdirectory
quote site trail
quote site sbsendeol=lf
put 'your.mainframe.migr.sql.dataset' yourlocalfile.sql
quit
/*
```

The following figure shows an example FTP session on a Windows or UNIX platform. In the example, bold text indicates user entries.

**Figure 3: FTP session**

```
C:\ftptest>ftp themainframe
Connected to themainframe.xyz.com.
220 Connection will close if idle for more than 30 minutes.
User (themainframe.xyz.com:(none)): mylogin
331 Send password please.
Password:  
230 MYLOGIN is logged on. Working directory is "MYLOGIN.".
ftp> quote site trail
200 SITE command was accepted
ftp> quote site sbsendeol=lf
200 SITE command was accepted
ftp> get
Remote file 'MYLOGIN.MIGRATE.SQL'
Local file mylocalfile.sql
200 Port request OK.
125 Sending data set MYLOGIN.MIGRATE.SQL
250 Transfer completed successfully.
```
Where to go from here

If you understand the information in this chapter, you can go on to create configuration files and apply requests for High-speed Apply Engine. For more information about these tasks, see the following chapters:

<table>
<thead>
<tr>
<th>Task</th>
<th>See</th>
</tr>
</thead>
<tbody>
<tr>
<td>instructions on defining and managing configuration files</td>
<td>“Defining configurations” on page 39</td>
</tr>
<tr>
<td>instructions on creating and maintaining apply requests</td>
<td>“Running High-speed Apply Engine” on page 51</td>
</tr>
<tr>
<td>instructions on configuring an apply request for restart and on restarting the request</td>
<td>“Restart and recovery” on page 63</td>
</tr>
<tr>
<td>format of the configuration files</td>
<td>“Configuration file format” on page 93</td>
</tr>
<tr>
<td>information about configuration parameters available for High-speed Apply Engine</td>
<td>“Configuration parameters” on page 103</td>
</tr>
</tbody>
</table>
Defining configurations

This section provides instructions for working with configuration files.

For more information on the format of the configuration file see, “Configuration file format” on page 93. For more information on the configuration parameters, see “Configuration parameters” on page 103.

Overview

The High-speed Apply Engine uses configuration parameters to determine the target database and tables, processing options, and conflict resolution rules for each apply request.

You can set configuration parameters in the following ways:

- create and edit configuration files
  You can then specify one or more configuration files in the apply request.

- override the values specified in the configuration files by specifying different values in the apply request
  For more information about creating apply requests, see “Running High-speed Apply Engine” on page 51.

Multiple configuration files for DB2 on mainframe targets

For DB2 on mainframe targets, a single apply request can refer to several configuration files. This feature gives you the flexibility to create High-speed Apply Engine configurations that meet the needs of your site.

You might use multiple configuration files to
- create a separate configuration file for conflict resolution rules
- specify a standard set of options (for example, temporary file storage options) that can be used by multiple apply requests
- store database-specific information in one configuration file, and specify different types of apply requests for that database in separate configuration files

To include multiple configuration files in an apply request, specify a data definition (DD) statement for the configuration FileName (CFN) parameter, and use the DD statement to include the files. For more information about specifying a DD statement as part of an apply request, see “Creating and running apply requests” on page 55.

**Creating and managing configuration files**

This section provides instructions for creating configuration files for target tables for DB2, DB2 LUW, and Oracle.

They also provide instructions for certain configuration file management tasks, including

- mapping source object names to target objects
  If the table or column names in the input file differ from those in the target database, you can configure the apply request to map the source object names to the target objects. For more information, see “Mapping source object names to target objects” on page 44.

- creating logical partitions for a target table
  You can create logical partitions for a target table to change the workload distribution for an apply request. For more information, see “Creating logical partitions for target tables” on page 47.

For more information about including configuration files in apply requests, see “Creating and running apply requests” on page 55.

**Configuration file management**

You can create as many configuration files as you need for the apply requests that you want to run.

Use the following practices to manage configuration files:
Create a separate library or directory for High-speed Apply Engine configuration files. If you decide to change the location of the configuration files, remember that you must update any apply requests with the new location. For Oracle and DB2 LUW targets, the configuration file must be in the current directory, or you must specify the full path name in the apply request.

Use a naming convention for the files to help you identify them. For example, you might identify the files with the target database name, job or task type (migration, SQL processing, or recovery), or input type (SQL or logical log file).

Organize configuration files in a meaningful way for your site. You might have several data sets or directories in the configuration library, depending on how many configuration files you create.

Because you can use several configuration files for a single apply request, you should include comments in the configuration file that describe how you use it. For more information about entering comments in a configuration file, see “Comments in configuration files” on page 98.

For DB2 on mainframe targets, High-speed Apply Engine recognizes DD names and data set names for the configuration FileName (CFN) parameter. You can use a DD statement in an apply request to override configuration defaults. For more information see “Running High-speed Apply Engine” on page 51.

Creating or editing a configuration file

The High-speed Apply Engine uses one or more configuration files to determine the parameter values for an apply request.

Use this procedure to create or edit configuration files for your apply requests.

Before you begin

High-speed Apply Engine provides sample configuration files for you to copy and edit, as follows:

1. Copy the sample configuration file.
2. Open the copy in a text editor.
3. Edit configuration parameters.
4. Save the configuration file.
For a DB2 mainframe target, the HLQ.DBSAMP library contains sample configuration files and apply requests. HLQ represents a high-level qualifier that you specified when you installed High-speed Apply Engine.

For a DB2 LUW target, the examples/db2udb subdirectory in the installation directory contains the sample files. The installation directory is the UNIX or Windows path that you specified for High-speed Apply Engine during installation.

For an Oracle target, the examples/oracle subdirectory in the installation directory contains the sample files. The installation directory is the UNIX or Windows path that you specified for High-speed Apply Engine during installation.

To help you determine which values to use for the parameters in this procedure, use the parameter descriptions in “Configuration parameters” on page 103. “Configuration file format” on page 93 describes the format of the configuration file.

To create a configuration file for a DB2 mainframe target

1. Create a configuration file with your desired configuration parameters.

   One method of doing this is to use a sample configuration file located in the HLQ.DBSAMP library as a template. If you use one of these files as a template, copy the file, renaming it according to your site requirements. Edit the file with any standard editor.

2. Edit the configuration parameters as required:

   a. You must include the [Startup] section with the SSID parameter.

      This step completes the minimum configuration. If you do not need to specify any other parameters, you can skip to Step 3 on page 43.

   b. (optional) If your input source is a logical log control file, you must include the [Logical Log] section and the SQLType parameter.

   c. (optional) If you include the [ObjectMap] section, you must specify at least one SourceTable parameter and one TargetTable parameter.

   d. (optional) If you include the [ObjectCluster] section, you must define at least one cluster.

   e. (optional) If you include the [ObjectPartition] section, you must define at least one partition.

   f. (optional) Specify other parameters as appropriate for the apply request.
Many of the configuration parameters have default values. Ensure that the
defaults are acceptable for your environment and for the apply requests that
will use the configuration file. For more information on each parameters
default values, see “Configuration parameters” on page 103.

**Tip**

Indent section names one or two spaces, so that you can easily insert comment
characters. Indent parameters under their section headings for readability. If you
include grouped parameters, such as those in the [ObjectMap] section, indent all
of the parameters in a group to the same level.

3 Save the file.

Note the file name and location. You will need this information to specify the
configuration file in an apply request.

**To create a configuration file for an Oracle or DB2 LUW target**

1 Create a configuration file with the your desired configuration parameters.

One method of doing this is to use a sample file as a template. The sample files
are located in the `examples/oracle` or `examples/db2udb` subdirectories of the
installation directory. If you use one of these files as a template, copy the file,
renaming it according to your site requirements. Edit the file with a standard
editor.

2 Edit the configuration parameters as necessary:

   a For Oracle, if you include the [Startup] section, you must include the Login
      parameter. If the target Oracle instance is configured for automated login, the
      [Startup] section is optional.

      For DB2 LUW, you must include the [Startup] section. In the [Startup] section
      you must include the database name for the Database parameter, and specify
      DB2 for the DBMSType.

   b *(optional)* If you include the [ObjectMap] section, you must specify at least one
      SourceTable parameter and one TargetTable parameter.

   c *(optional)* If you include the [ObjectCluster] section, you must define at least
      one cluster.

   d *(optional)* If you include the [ObjectPartition] section, you must define at least
      one partition.

   e *(optional)* Specify other parameters as appropriate for the apply request.
Many of the configuration parameters have default values. Ensure that the defaults are acceptable for your environment and for the apply requests that will use the configuration file. For more information about each parameter’s default values, see “Configuration parameters” on page 103.

Tip
Indent section names one or two spaces, so that you can easily insert comment characters. Indent parameters under their section headings for readability. If you include grouped parameters, such as those in the [ObjectMap] section, indent all of the parameters in a group to the same level.

3 Save the file.

Note the file name and location. You will need this information to specify the configuration file in an apply request.

Where to go from here

You can specify this configuration file in one or more apply requests. For more information about creating apply requests, see “Running High-speed Apply Engine” on page 51.

Configurations for specific tasks

This section provides instructions on how to define configurations for several specific uses of the High-speed Apply Engine.

Mapping source object names to target objects

If the table or column names in the input file differ from those in the target database, you can use this procedure to help you configure the apply request to map the source object names to the target objects.
The following figure shows the relationships described in this procedure.

**Figure 4: Mapping source object names to target objects**

Before you edit the configuration, you should determine which source and target object names will be mapped in the configuration. The target object names that you specify must exist when you run an apply request that requires them. For more information about the parameters in this procedure, see “ObjectMap parameters” on page 224.

**Note**
You can use wildcard patterns to specify groups of objects. For more information, see “Using wildcards to map object names” on page 226.
To map a source table to a target table

1. Create or edit a configuration file as described in “Creating or editing a configuration file” on page 41.

2. Add the [ObjectMap] section name to the configuration file.

3. Specify a SourceTable parameter in the [ObjectMap] section, as follows:
   ```
   SourceTable=tableName
   ```
   A source table name is a table name specified in the input file. This table can reside on a different DB2 subsystem, DB2 LUW database, or Oracle instance than the target table. High-speed Apply Engine does not validate the existence of the source table. The name must conform to the conventions that are described in “Object names in configuration files” on page 99.

4. Specify a TargetTable parameter, as follows:
   ```
   TargetTable=tableName
   ```
   The target table name you specify must be the target for the changes that would otherwise be applied to the source table that you specified in “Mapping source object names to target objects” on page 44. The target table must reside on the same subsystem or server as High-speed Apply Engine. The name must conform to the conventions that are described in “Object names in configuration files” on page 99.

5. If the column names differ between the source and target table, specify a SourceColumn and TargetColumn parameter for each column that needs to be mapped. (If the columns are the same for both tables, you can proceed to Step 6 on page 46.)
   ```
   SourceColumn=source01
   TargetColumn=target01
   SourceColumn=source02
   TargetColumn=target02
   ```
   The column name parameters must follow the corresponding TargetTable parameter. Each SourceColumn parameter must be followed by a TargetColumn parameter. The target column names must exist when you run the apply request.

6. Repeat steps 3,4 and 5 until you have mapped all object names that are required for the apply request.

7. Save the configuration file.

When you run an apply request that refers to the configuration file, High-speed Apply Engine maps the object names in the input file according to the parameters in the [ObjectMap] section.
If a problem occurs

If you receive error messages about the object names specified in the [ObjectMap] section, check the configuration file for the following common problems:

- Verify that the [ObjectMap] parameters are in the correct order.

- Verify that the target objects that are specified by the TargetTable and TargetColumn parameters exist in the DB2 subsystem, DB2 LUW database, or Oracle instance where you ran the apply request.

Creating logical partitions for target tables

Use this procedure to create logical partitions for a target table to change the workload distribution for an apply request.

Figure 5: Creating logical partitions for target tables

Create or edit a configuration file. → Add ObjectPartition parameters to the file. → Save the configuration file.

/ObjectPartition/
Table=ACCT.PAYABLE
Column=AcctNumber
Order=ASC
Partition=CC5700
Value=5700
Partition=CC6000
Value=6000

ObjectPartition Parameters
Before you begin

To define logical partitions, you must know the column name and value that represent the upper limit of each partition. For a DB2 mainframe target, you can also use a limit key value to define the partition.

The order of the parameters in the [ObjectPartition] section is significant. The steps in this procedure describe the required order. For more information, see “Positional requirements for ObjectPartition parameters” on page 232.

If you create logical partitions for a table that is physically partitioned, High-speed Apply Engine ignores the physical partitions and uses the logical partitions for workload distribution.

High-speed Apply Engine can create partition keys from the columns and values that you specify within the partition definition. The partition key length has the following limitations:

- For Oracle and DB2 LUW targets, the partition key is truncated after 256 bytes. Specify unique key values within the first 256 bytes of the partition definition.
- For DB2 mainframe targets, the partition key is truncated after 255 bytes. Specify unique key values within the first 255 bytes of the partition definition.

To define logical partitions for apply processing

1. Create or edit a configuration file as described in “Creating or editing a configuration file” on page 41.

2. Add the [ObjectPartition] section name to the configuration file.

3. Specify the table that you want to partition by using the TableName parameter.

   The table that you specify must exist on the target DB2 subsystem, DB2 LUW database, or Oracle instance.

4. Specify a Column parameter for a column to be used in the partition definition.

5. *(optional)* For DB2 mainframe targets, specify an Order parameter for the column, if necessary.

   This parameter is optional. You can specify either ascending (ASC) or descending (DESC) order for the corresponding column. The Order parameter must immediately follow its associated Column parameter.

   If you do not include an Order parameter for a column, the default sort order is ascending.
6 Repeat Step 4 on page 48 and Step 5 on page 48 for each column that you want to use in the partition definition.

7 Define the upper limits for a partition as follows:

a Specify a Partition parameter.

The Partition parameter indicates the start of a partition definition. You must specify this parameter at least one time in the [ObjectPartition] section, but you do not need to specify a value for it. If you do specify a value, it appears in the informational messages for the apply request.

b Specify the upper limit for the partition with a Value parameter (or with a LimitKey parameter) for each of the Column parameters you defined in Step 4 on page 48.

   ■ The Value parameter represents the upper limit of the partition for the associated column. You must specify the Value parameters in the same order as the Column parameters. If the order is incorrect, High-speed Apply Engine cannot process the apply request.

   ■ If your target is a DB2 table on a mainframe and you have one or more limit-key values that represent a partition point, you can specify the LimitKey parameter as the upper limit of the partition. The LimitKey parameter and the Value parameter are mutually exclusive. For more information, see “LimitKey” on page 235.

   ■ Repeat this step to create additional partitions for the table you specified in Step 3 on page 48. The values for the partition definitions must be defined in ascending order. The values for each succeeding partition must be greater than the values of the previous partition.

8 Repeat this procedure for each target table that you want to process with logical partitions.

If a problem occurs

If you receive error messages about the object names specified in the [ObjectPartition] section, check the configuration file for the following common problems:

   ■ Verify that the [ObjectPartition] parameters are in the correct order.

   ■ Verify that the column names and values for the partition definitions are valid values, and that the partition definitions appear in ascending order in the configuration.
Running High-speed Apply Engine

This chapter provides basic tasks required to use the High-speed Apply Engine.

Overview

An apply request contains the following items:

- command to run the High-speed Apply Engine
- any overrides to the parameters in the configuration file

The format for the request varies with the operating system platform. This section describes overall considerations and the format for each platform.

Overriding parameters in an apply request

You can enter any of the following parameters in an apply request to override the information in the request’s configuration file. Specify all other configuration information in the configuration file. Use the appropriate format for your command override.

For more information, see “Mainframe batch job” on page 52, or “UNIX and Windows command line” on page 52.

The override parameters include:

- any of the following abbreviated FileName parameters:
  - CFN, [Configuration] section
  - DFN, [DisplayOutput] section
  - EFN, [ErrorOutput] section
— LFN, [LogOutput] section
— IFN, [Startup] section

■ any of the following parameters in the [StartUp] section:
  — DefaultOpts, abbreviated as DOPT
  — FileName, abbreviated as IFN
  — InputType, abbreviated as IT
  — PlanName, abbreviated as PLAN
  — SSID
  — Login
  — ConnectionVersion
  — Database
  — UserID
  — Password

**UNIX and Windows command line**

An apply request for UNIX and Windows is a command entered at the command prompt.

You can specify configuration overrides as parameters of the command, as shown in the following figure. Separate multiple configuration overrides with commas. In the figure, *path* represents the path to the High-speed Apply Engine directory.

**Figure 6: Sample apply request for UNIX and Windows**

`path:apemain ifn=may_18_2008.sql,cfn=apply_cfg.ora10.ini`

**Mainframe batch job**

On z/OS systems, an apply request is a batch job that you submit for processing as you would submit any job. You can view the results of the job in the same way that you view any batch job output.
Note
For more information about comments in your configuration, see “Comments in configuration files” on page 98.

The examples in this chapter are for reference only. When you create an apply request, you must include a valid job statement and any processing options that your site requires.

The following figure shows a sample apply request.

Figure 7: Sample apply request for mainframe targets (JCL)

```
//APTMAIN  JOB STATEMENT
//*
//APPLYSQL EXEC PGM=APTBMAIN,PARM='SSID=DBAT,IT=SQL',
//   REGION=0M
//STEPLIB  DD DISP=SHR,DSN= product.libraries
// DD DISP=SHR,DSN=DB2.DSNEXIT
// DD DISP=SHR,DSN=DB2.DSBLOAD
//APTEE R DD SYSS OUT=*
//APTPRIN T DD SYSO U T=* 
//APTplOG DD SYSSOUT=*,LRECL=550,RECFM=VB
//SYSPRINT DD SYSSOUT=* 
//SYSDUMP DD SYSSOUT=* 
//SYSTEM DD SYSSOUT=* 
//APTDUMP DD SYSSOUT=* 
//SYSO U T DD SYSSOUT=*
//*
//SYSSIN DD DDNAME=APTCNFG
// DD DISP=SHR,DSN=BMCAPT.CONFIG.FILES(CFGAGENT)
// DD DISP=SHR,DSN=BMCAPT.CONFIG.FILES(CFG_DISP)
// DD DISP=SHR,DSN=BMCAPT.CONFIG.FILES(CFGLOG)
//*
//APTCNFG DD *
IFN=BMCALP.LOGMSTR.MIGRATE.SQL
//*
```

DD names and data set names

The High-speed Apply Engine recognizes both data definition (DD) names and data set names for configuration information.

It uses the following criteria to determine whether a name is a DD name or a data set name:

- If the name is eight characters or fewer, High-speed Apply Engine treats it as a DD name. The DD name must refer to a DD statement in the same apply request. If the character string is not a valid DD name, High-speed Apply Engine treats it as a data set name.

- If the name is longer than eight characters, High-speed Apply Engine treats it as a data set name. The names that you specify must comply with the name rules for data sets on the current operating system.
Input files can be any data set supported by the operating system, including a partitioned data set (PDS) or generation data group (GDG) data sets.

Specifying DD statements on mainframe targets

On mainframe targets, High-speed Apply Engine uses certain data sets depending on the type of processing that you request.

You must allocate some data sets by using a DD statement in the JCL, while High-speed Apply Engine allocates other data sets dynamically. The data sets that can be allocated are described in the following paragraphs.

- **SYSIN**: Defines the input data set that contains High-speed Apply Engine configuration parameters and their values. Specify this data set either as a DD statement in the JCL or as the value of the FileName parameter in the [Configuration] section (commonly abbreviated as CFN), see “FileName (CFN)” on page 155.

- **APTErr**: Defines the output data set where High-speed Apply Engine displays error messages that are generated during apply processing and additional information. Always specify this data set as a DD statement in the JCL.

- **APTPRINT**: Defines an output data set where High-speed Apply Engine displays all statistics and messages that are generated during apply processing. Always specify this data set as a DD statement in the JCL.

- **APTDump**: *(optional)* Defines the output data set that contains a SNAP dump that High-speed Apply Engine might generate in response to internal errors. BMC Software recommends that you include this optional data set in your JCL as a troubleshooting aid.

- **APTDiag**: *(optional)* Defines the output data set where High-speed Apply Engine writes special diagnostic information used by BMC Software Customer Support in troubleshooting situations. High-speed Apply writes output to this file when the value of the RestartType parameter is other than None. Specify this data set either as a DD statement in the JCL or as the value of the FileName parameter in the [DiagnosticOutput] section, see “FileName” on page 179.

- **SYSOUT**: *(optional)* Defines the output data set that contains messages from the sort routine. Normally, you can specify this data set as SYSOUT=*.

- **SYSUDump**: *(optional)* Defines a data set used for system dumps produced when a system error occurs. BMC Software recommends that you include either SYSUDUMP or SYSMDump in your JCL as a troubleshooting aid.

- **SYSMDump**: *(optional)* Defines a data set used for system dumps produced when a system error occurs. Include either the SYSMDump data set or the SYSUDUMP data set in your JCL as a troubleshooting aid. For SYSMDump, the
product produces a system dump that can be read by the IBM Interactive Problem Control System (IPCS) product. Specify the attributes of the SYSMDUMP data set as indicated in the appropriate JCL documentation.

- **SYSTERM**: (optional) Defines the output data set where High-speed Apply Engine writes messages and other error information when data sets that are defined by other required DD statements are not available.

### Apply request management for DB2 for mainframe targets

You might create a large number of apply requests to perform a variety of tasks using the High-speed Apply Engine.

Use the following practices to help you manage the apply requests:

- Create a separate library for the apply requests and organize them in a meaningful way for your site. You might have several members in the request library, depending on the number of apply requests that you create.

- Use a naming convention for the requests to help you identify them. For example, you might identify the files by target database, request type (migration, SQL processing, or recovery), or input type (SQL or logical log file).

- Include comments in the apply request that describe the tasks it performs. Ensure that the comments appear on a separate line from any configuration parameters.

### Creating and running apply requests

The tasks for initiating an apply request vary with the target database type.

Use the procedures in the following sections to

- Help you write apply requests that fit your requirements

- Manage the requests that you create

### Creating and running an apply request for DB2 on mainframe targets

You create and save an apply request for DB2 in the same way that you write any batch job. You can use any editor to create and save an apply request. The job steps that you include in the apply request depend on the type of apply processing that
you want to perform. Use the procedure in this section to create a basic apply request.

Before you begin

Before you can run an apply request, the target tables must exist if you are using [ObjectMap] parameters. The High-speed Apply Engine load libraries and any libraries referenced in the STEPLIB DD statements must be APF-authorized.

You should also become familiar with the HLQ.DBSAMP library. This library contains a sample configuration file and apply request. HLQ represents a high-level qualifier that you specified when you installed High-speed Apply Engine.

To create an apply request for DB2

1. Create a job that runs an apply request.

   One method of doing this is to use a sample apply request job as a template. (A sample job is stored in the APTSAMP member in the HLQ.DBSAMP library). If you use this template, copy the file, renaming it according to your site requirements. Edit the file with any standard editor.

2. Edit the job statement to conform to your site requirements.

3. Verify that the EXEC statement invokes APTBMAIN.

4. Specify your configuration information. You can specify a single configuration file, multiple configuration files, or no configuration file.
   a. To specify a single configuration file, specify the name of your configuration file in the default DD statement SYSIN. For example:

      ```
      //HSAESQL EXEC PGM=APTBMAIN,
      //      PARM='SSID=DBAT',REGION=0M
      
      //SYSIN DD DISP=SHR,DSN=APT.CNFG.FILE(CONFIG)
      ```

   b. To specify multiple configuration files, specify the names of your configuration files in the default DD statement SYSIN (For an example, see Figure 7 on page 53).

   c. To specify no configuration file, enter None as the value of the FileNane (CFN) parameter within the PARM parameter of EXEC statement in your JCL. For example:

      ```
      //APPLYSQL EXEC PGM=APTBMAIN,
      //      PARM='SSID=DBAT,CFN=NONE',REGION=0M
      ```

5. Add any configuration overrides required for the request.
You can specify the overrides within the PARM parameter of the EXEC statement, or with a separate DD statement that includes configuration sections. For example:

```
//APPLYSQL EXEC PGM=APTBMAIN,
//   PARM='SSID=DBAT,InputType=SQL,IFN=APT.INPUT.SQLFILE2'
```

6. Save the job if you want to reuse it in the future.

7. Run the job to start processing the apply request.

### Verifying High-speed Apply Engine maintenance on mainframe targets

In some situations, you might need to verify maintenance that has been applied to the High-speed Apply Engine. If an apply request does not complete successfully, you might need to provide this information to BMC Software Customer Support.

1. Use one of the following methods to obtain maintenance information:
   - Run a normal apply request
     Examine the output of an apply request in your environment. Message BMCAPT0215 displays a list of the maintenance fixes that have been applied to High-speed Apply Engine and its components. This method requires a connection to DB2.
   - Run a maintenance job
     Use the special EXEC statement parameter called MAINT. An apply request with this parameter prints information from the control sections (CSECT) that track maintenance applied to High-speed Apply Engine and its components. High-speed Apply Engine performs no other processing. You do not need a connection to DB2.

Figure 8 on page 57 shows sample JCL that you can modify to run a maintenance job. Figure 9 on page 57 shows the output information that you can use to verify maintenance.

#### Figure 8: Sample maintenance job for mainframe targets (JCL)

```
//APTMAIN  JOB STATEMENT
//*
//APPLY    EXEC PGM=APTBMAIN,REGION=0M,
//         PARM='MAINT'
//STEPLIB  DD DISP=SHR,DSN=product.libraries
//         DD DISP=SHR,DSN=DB2.DSNEXIT
//         DD DISP=SHR,DSN=DB2.DSNLOAD
//APTERR   DD SYSOUT=*  
//APTPRINT DD SYSOUT=*  
```

#### Figure 9: Sample maintenance output for mainframe targets

```
BMCAPT0001I HIGH-SPEED APPLY ENGINE V09.01.00
Copyright (c) 1999-2008 BMC Software, Inc. as an unpublished license
BMCAPT0001I Date: Sat May 18, 2008
```
Creating and running an apply request for DB2 LUW

Because of the command-line interface, generating an apply request for a DB2 LUW target can be as simple as typing a command.

Use the procedure in this section to simplify the apply request process.

**Before you begin**

Before you perform the tasks in this section, you should be familiar with the default path and configuration file for the High-speed Apply Engine in your environment. For more information, see “File names for UNIX and Windows” on page 98.

When you are configuring High-speed Apply Engine, be sure that the shared library environment variable is set to enable High-speed Apply Engine to run correctly. The name of the environment variable depends on your operating system, as shown in Table 2 on page 58. In this table, `installationDirectory` refers to the directory where High-speed Apply Engine was installed. The `platform` is the UNIX platform chosen during the installation (aix, sol). The `v.r.m` is the version, release, and maintenance level of High-speed Apply Engine.

**Table 2: Shared library environment variable values: DB2 LUW**

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Environment variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solaris</td>
<td>LD_LIBRARY_PATH</td>
<td><code>installationDirectory/HighSpeedApply/platform/v.r.m/bin</code></td>
</tr>
<tr>
<td>AIX</td>
<td>LIBPATH</td>
<td><code>installationDirectory/HighSpeedApply/platform/v.r.m/bin</code></td>
</tr>
<tr>
<td>Windows</td>
<td>PATH</td>
<td><code>installationDirectory\High-speed Apply</code></td>
</tr>
</tbody>
</table>

You should also become familiar with the `examples/db2udb` subdirectory within the installation directory when you are configuring High-speed Apply Engine. The `examples/db2udb` subdirectory contains a sample apply request. The installation directory is the UNIX or Windows directory that you specified for High-speed Apply Engine during installation.

You can manually set these environment variables and run High-speed Apply Engine by invoking `apebmain` (for more information, see “To run an apply request for DB2 LUW” on page 59), or you can use the sample scripts in the installation root directory (`installationDirectory/HighSpeedApply/platform/v.r.m`).

58  High-speed Apply Engine Reference Manual
You must update the BMC Software registry before you can use the sample scripts in the installation root directory. For more information, see “Installing High-speed Apply Engine on UNIX or Windows” on page 299.

The sample scripts in the installation root directory are named apeenv and HighSpeedApply:

- The apeenv script originates from the profile and will set the environment variables required to run High-speed Apply Engine. You need to make specific reference to apeenv only once during your login session by typing . apeenv at the command prompt.

- You can use the HighSpeedApply script to run High-speed Apply Engine without having to set the path and shared library path. Using the HighSpeedApply script is the easiest way to run the component. It accepts the same parameters you specify to the High-speed Apply Engine executable command (apebmain), including the overrides to the parameters in the configuration file. BMC Software recommends that you move the HighSpeedApply script to one of the default directories in your path or your group’s path.

**To run an apply request for DB2 LUW**

1. At the command prompt type apebmain.

2. Verify the name and location of the configuration file for the apply request, as follows:

   - If the file uses the default path and file name, skip to Step 4 on page 59.
   - If the file does not use the default path and file name, go to Step 3 on page 59.
   - If you are not using a configuration file, enter cfn=none and skip to Step 4 on page 59.

   (You do not need a configuration file if you want to accept all High-speed Apply Engine default values, or if you enter all nondefault values as overrides on the command line.)

3. Enter the name of the configuration file, using the FileName (CFN) parameter. For an example, see Figure 6 on page 52.

   If the configuration file is in the current directory, type the file name. Otherwise, you must type the complete path and file name for the configuration file.

4. Add any configuration overrides required for the request.
You can include as many overrides as necessary to complete the apply request. Separate each section name and parameter with a single space.

5 Press **Enter** to start processing the apply request.

### Creating and running an apply request for Oracle

Because of the command-line interface, generating an apply request for an Oracle target can be as simple as typing a command.

Use the procedure in this section to simplify the apply request process.

**Before you begin**

Before you perform the tasks in this section, you should be familiar with the default path and configuration file for the High-speed Apply Engine for UNIX or for Windows. For more information, see “File names for UNIX and Windows” on page 98.

When you are configuring High-speed Apply Engine, be sure that the shared library environment variable is set to enable High-speed Apply Engine to run correctly. The name of the environment variable depends on your operating system, as shown in Table 3 on page 60. In this table, *installationDirectory* refers to the directory where High-speed Apply Engine was installed. The *platform* is the UNIX platform chosen during the installation (aix, sol). The *v.r.m* is the version, release, and maintenance level of High-speed Apply Engine.

<table>
<thead>
<tr>
<th>Operating system</th>
<th>Environment variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solaris</td>
<td>LD_LIBRARY_PATH</td>
<td><em>installationDirectory</em>/HighSpeedApply/platform/v.r.m/bin</td>
</tr>
<tr>
<td>AIX</td>
<td>LIBPATH</td>
<td><em>installationDirectory</em>/HighSpeedApply/platform/v.r.m/bin</td>
</tr>
<tr>
<td>Windows</td>
<td>PATH</td>
<td><em>installationDirectory</em>/HighSpeedApply</td>
</tr>
</tbody>
</table>

You should also become familiar with the **examples/oracle** subdirectory in the installation directory when you are configuring High-speed Apply Engine. The **examples/oracle** subdirectory contains a sample apply request. The installation directory is the UNIX or Windows directory that you specified for High-speed Apply Engine during installation.

You can manually set these environment variables and run High-speed Apply Engine by invoking apebmain (for more information, see “To run an Apply request for Oracle” on page 61), or you can use the sample scripts in the installation root directory (*installationDirectory*/HighSpeedApply/platform/*v.r.m*).
**Note**

You must update the BMC Software registry before you can use the sample scripts in the installation root directory. For more information, see “Installing High-speed Apply Engine on UNIX or Windows” on page 299.

The sample scripts in the installation root directory are named apeenv and HighSpeedApply:

- The apeenv script originates from the profile and sets the environment variables required to run High-speed Apply Engine. You need to make specific reference to apeenv only once during your login session by typing `. apeenv` (a period followed by a space, then apeenv) at the command prompt.

- You can use the HighSpeedApply script to run High-speed Apply Engine without having to set the path and shared library path. Using the HighSpeedApply script is the easiest way to run High-speed Apply Engine. It accepts the same parameters you specify to the High-speed Apply Engine executable command (apebmain), including the overrides to the parameters in the configuration file. BMC Software recommends that you move the HighSpeedApply script to one of the default directories in your path or your group’s path.

**To run an Apply request for Oracle**

1. At the command prompt, type `apebmain`.

2. Verify the name and location of the configuration file, as follows:
   - If the file uses the default path and file name, skip to **Step 4 on page 61**.
   - If the file does not use the default path and file name, go to **Step 3 on page 61**.
   - If you are not using a configuration file, enter `cfn=none` and skip to **Step 4 on page 61**.

   (You do not need a configuration file if you accept all High-speed Apply Engine default values, or if you enter all nondefault values as overrides on the command line.)

3. Enter the name of the configuration file, using the FileName (CFN) parameter. (For an example, see **Figure 6 on page 52**.)

   If the configuration file is in the current directory, type the file name. Otherwise, you must type the complete path and file name for the configuration file. You can specify only one configuration file in an apply request for an Oracle target.

4. Add any configuration overrides required for the request.
You can include as many overrides as necessary to complete the apply request. Separate each section name and parameter with a single space.

5 Press Enter to start processing the apply request.

Where to go from here

You have completed the basic tasks required to use the High-speed Apply Engine. Refer to the following chapters:

- “Restart and recovery” on page 63 provides instructions for creating a restart table and specifying restart parameters for an apply request.

- “Configuration parameters” on page 103 contains reference information about the configuration parameters available for High-speed Apply Engine.
Restart and recovery

This chapter provides an overview of restart processing for the High-speed Apply Engine product. The chapter also explains how to enable restart processing and summarizes different approaches for specifying restart parameters.

Overview

If your operating environment or DBMS fails while the High-speed Apply Engine is running, you can use the information in a restart table to continue apply processing from the failure point. Restart processing prevents High-speed Apply Engine from repeating transactions that have already been processed.

The High-speed Apply Engine restart table resides in your target database. For DB2 on mainframe targets, it must reside on the target DB2 subsystem. If you enable restart processing, High-speed Apply Engine stores information in the restart table during each run of an apply request. The structure of the restart table is shown in Figure 10 on page 66, Figure 11 on page 67, or Figure 12 on page 69, depending on your target database environment. If you need more information about the content of the restart table, contact BMC Software Customer Support.

You must create a default restart table during installation. You can define additional (personal) restart tables as needed in your environment. The procedures in this chapter describe how to create restart tables, specify the appropriate configuration parameters, and restart an apply request. Table 4 on page 75 summarizes two different ways that you can define restart parameters in your environment.

WARNING

In general, do not modify the input file of an apply request between initial run and restart. Modify the configuration file only to the extent required to resolve error conditions. Ensure that changes to the configuration do not alter the distribution of work between agents. For example, do not change the values of configuration parameters such as MaxAgents, DistributionType, PartitionClustering, or RIClustering.
Enabling restart processing

This section describes the tasks that you must perform to enable restart processing for the High-speed Apply Engine:

- You must create one or more restart tables that High-speed Apply Engine can use to record information about the apply requests. Normally, a restart table is created during installation. You can use the instructions in “Creating a restart table” on page 64 to create additional tables, if necessary.

- You must specify the appropriate restart parameters in the configuration information, as instructed in “Specifying restart parameters” on page 70.

- To restart an apply request, you must ensure that the configuration includes the appropriate restart ID and rerun the request, as described in “Restarting an apply request” on page 72.

Creating a restart table

The installation procedures for the High-speed Apply Engine create a restart table that is used if you do not include the TableName parameter in the [Restart] section of the configuration.

Use the procedures in this section to create additional restart tables that High-speed Apply Engine can use to track apply processing.

Before you begin

By default, High-speed Apply Engine uses a single restart table for all apply requests. (This table can be the restart table created during installation, or one that you designate.) In normal use, this practice works well. However, if you run a large number of apply requests simultaneously on the same target database, you might begin to experience resource contention for the single restart table.

If you experience contention, create a restart table for each user that runs apply requests simultaneously on the target database. This practice eliminates contention for the restart table.

High-speed Apply Engine provides sample data definition language (DDL) files to help you create the restart table as follows:

- For a DB2 mainframe target, the HLPQ.DBSAMP library contains a member named APTREST. HLPQ represents a high-level qualifier that you specified when you installed High-speed Apply Engine.
For DB2 LUW targets, the examples/db2udb subdirectory in the installation directory contains a file named CrtUDBRestartTbl.sql. The installation directory is the UNIX or Windows path that you specified for High-speed Apply Engine during installation.

For Oracle targets, the examples/oracle subdirectory in the installation directory contains a file named CrtOraRestartTbl.sql. The installation directory is the UNIX or Windows path that you specified for High-speed Apply Engine during installation.

To create a personal restart table for DB2 on a mainframe target

1. Create a DDL file with the statements to create a restart table.

   One method of doing this is to use the sample DDL file located in a member named APTREST in HLQ.DBSAMP library. If you use this file as a template, copy the file, renaming it according to your site requirements. Edit the file with any standard editor.

   Regardless of how you create the file, the DDL statements should resemble those shown in Figure 10 on page 66 when you have finished.

2. Edit the DDL statements as follows:

   a. Change the database name and table space name to values that are appropriate for your site.

      The table space that you specify must be located in the same DB2 subsystem as the target tables for the apply request.

   b. Change the table name and index name so that they are unique.

   c. (optional) Define the alias (BMCAPT_APTRSCL) so that it uses the name of the personal restart table. If an earlier version of High-speed Apply Engine is already installed, you must drop and re-create the alias.

      Do not redefine this alias unless you want all apply requests that do not specify the TableName parameter in the [Restart] section to use the personal restart table. The alias is the default value for this TableName parameter. If an apply request does not specify a restart table name, High-speed Apply Engine writes restart information to the table that the alias represents.

      When you have finished editing your DDL file, the statements in it should resemble those in Figure 10 on page 66. The databaseName, tableSpaceName,
owner, restartTableName, and indexName variables represent unique values that are appropriate for your environment.

Figure 10: Sample DDL to create a restart table: DB2

```sql
/*--------------------------------------------------------------------*/
/*- DEFINE RESTART DB,TABLESPACE,TABLE,INDEX AND SYNONYM           */
/*--------------------------------------------------------------------*/
* DROP DATABASE databaseName ;
* COMMIT ;
CREATE DATABASE databaseName
  STOGROUP SYSDEFLT
  CCSID EBCDIC;
COMMIT;
CREATE TABLESPACE tableSpaceName
  IN databaseName
  USING STOGROUP SYSDEFLT
  PRIQTY 240
  SECQTY 48
  CLOSE NO
  FREEPAGE 0
  PCTFREE 0
  SEGSIZE 4;
COMMIT;
CREATE TABLE owner.restartTableName
  (  RESTART_ID       CHAR(16)     NOT NULL,
      REQUEST_ID      CHAR(4)      NOT NULL,
      RECORD_TYPE     CHAR         NOT NULL,
      AGENT_ID        SMALLINT     NOT NULL,
      RECORD_SEQUENCE SMALLINT     NOT NULL,
      REQUEST_TIMESTAMP TIMESTAMP    NOT NULL,
      RECORD_TIMESTAMP TIMESTAMP    NOT NULL,
      TOTAL_LENGTH    INTEGER      NOT NULL,
      DATA            LONG VARCHAR NOT NULL
    ) IN databaseName.tableSpaceName;
COMMIT;
CREATE UNIQUE INDEX owner.indexName
  ON owner.restartTableName
  (  RESTART_ID,
      REQUEST_ID,
      RECORD_TYPE,
      AGENT_ID,
      RECORD_SEQUENCE
    ) USING STOGROUP SYSDEFLT;
COMMIT;
CREATE ALIAS BMCAPT_APRSCL FOR owner.restartTableName;
GRANT ALL ON TABLE owner.restartTableName TO PUBLIC;
COMMIT;

3 Save your DDL member, and execute it to create the restart table.

To create a personal restart table for DB2 LUW

1 Create a data definition language (DDL) file with the statements to create a restart table.
One method of doing this is to use the sample DDL file named CrtUDBRestartTbl.sql in the examples/db2udb subdirectory as a template. If you use this file as a template, copy the file, renaming it according to your site requirements. Edit the file with any standard editor.

Regardless of how you create the file, the DDL statements should resemble those shown in Figure 11 on page 67 when you have finished.

2 Edit the DDL statements as follows:

a Edit the table name to a value that is appropriate for your site.

The table name that you specify must be located in the same DB2 LUW database as the target tables for the apply request. Note this name so that you can specify it in the configuration for the apply request.

b Edit the index name so that it is unique.

c (optional) Define the alias (BMCAPE_APERSCL) so that it uses the name of the personal restart table. If an earlier version of High-speed Apply Engine is already installed, you must drop and re-create the alias.

Do not redefine this alias unless you want all apply requests that do not specify the TableName parameter in the [Restart] section to use the personal restart table. The alias is the default value for this TableName parameter. If an apply request does not specify a restart table name, High-speed Apply Engine writes restart information to the table that the alias represents.

d Edit the GRANT statement if necessary. For personal restart tables, you can restrict access if necessary.

e Add a CREATE TABLESPACE statement if this is the standard for your site.

When you have finished editing your DDL file, the statements in it should resemble those in Figure 11 on page 67. The restartTableName, and indexName variables represent unique values appropriate for your environment.

**Figure 11: Sample DDL to create a restart table: DB2 LUW**

```sql
CREATE TABLE restartTableName
  (  RESTART_ID         CHAR(16)     NOT NULL,
  REQUEST_ID         CHAR(4)      NOT NULL,
  RECORD_TYPE        CHAR         NOT NULL,
  AGENT_ID           SMALLINT     NOT NULL,
  RECORD_SEQUENCE    SMALLINT     NOT NULL,
  REQUEST_TIMESTAMP  TIMESTAMP    NOT NULL,
  RECORD_TIMESTAMP   TIMESTAMP    NOT NULL,
  TOTAL_LENGTH       INTEGER      NOT NULL,
  DATA               LONG VARCHAR NOT NULL
  );
```
CREATE UNIQUE INDEX indexName ON restartTableName
    ( RESTART_ID,
    RECORD_TYPE,
    AGENT_ID,
    RECORD_SEQUENCE
    );

GRANT ALL ON TABLE restartTableName TO PUBLIC;
CREATE ALIAS BMCAPE_APEXSL FOR restartTableName;
COMMIT;

3  Save the edited DDL file.

4  Run the DDL file to create the restart table.

   ■ For Windows, type db2cmd from a command prompt to get to the DB2 command level prompt (CLP); then type the following commands:
      db2 connect to <database>
      db2 -t -f CrtUDBRestartTbl
      db2 disconnect

   ■ For UNIX, type the following commands:
      db2 connect to <database>
      db2 -t -f CrtUDBRestartTbl
      db2 disconnect

To create a personal restart table for Oracle

1  Create a data definition language (DDL) file with the statements to create a restart table.

   One method of doing this is to use the sample DDL file named CrtOraRestartTbl.sql in the examples/oracle subdirectory as a template. If you use this file as a template, copy the file, renaming it according to your site requirements. Edit the file with any standard editor.

   Regardless of how you create the file, the DDL statements should resemble those shown in Figure 12 on page 69 when you have finished.

2  Edit the DDL statements as follows:

   a  Edit the table name to a value that is appropriate for your site.

      The table name that you specify must be located in the same Oracle database as the target tables for the apply request. Note this name so that you can specify it in the configuration for the apply request.
b  Edit the index name so that it is unique.

c  *(optional)* Define the public synonym (BMCAPE_APERSCL) so that it uses the name of the personal restart table. If an earlier version of High-speed Apply Engine is already installed, you must drop and re-create the public synonym.

Do not redefine this synonym unless you want *all* apply requests that do not specify the TableName parameter in the [Restart] section to use the personal restart table. The synonym is the default value for this TableName parameter. If an apply request does not specify a restart table name, High-speed Apply Engine writes restart information to the table that the public synonym represents.

d  Edit the GRANT statement if necessary. For personal restart tables, you can restrict access if necessary.

When you have finished editing your DDL file, the statements in it should resemble those in Figure 12 on page 69. The restartTableName and indexName variables represent unique values appropriate for your environment.

**Figure 12: Sample DDL to create a restart table: Oracle**

```sql
CREATE TABLE restartTableName  
  ( RESTART_ID         CHAR(16)     NOT NULL,  
    REQUEST_ID         CHAR(4)      NOT NULL,  
    RECORD_TYPE        CHAR         NOT NULL,  
    AGENT_ID           SMALLINT     NOT NULL,  
    RECORD_SEQUENCE    SMALLINT     NOT NULL,  
    REQUEST_TIMESTAMP  DATE         NOT NULL,  
    RECORD_TIMESTAMP   DATE         NOT NULL,  
    TOTAL_LENGTH       INTEGER      NOT NULL,  
    DATA               LONG VARCHAR NOT NULL  
  ) ;

CREATE UNIQUE INDEX indexName ON restartTableName  
  ( RESTART_ID,  
    RECORD_TYPE,  
    AGENT_ID,  
    RECORD_SEQUENCE  
  ) ;

CREATE PUBLIC SYNONYM BMCAPE_APERSCL FOR restartTableName;

GRANT SELECT, UPDATE, DELETE, INSERT ON restartTableName  
TO PUBLIC;

EXIT
```

3  Save the edited DDL file.

4  Run the DLL file to create the restart table.

   - From SQL*Plus, issue the following command:
     ```
     @CrtOraRestartTbl
     ```
From a UNIX or Windows command prompt, issue the following command:

```
sqlplus userID/ password @CrtOraRestartTbl
```

If you are not already running SQL*Plus, you must supply your user ID and password to start the utility.

**Where to go from here**

With the restart table in place, you can specify the restart parameters in the configuration for the apply request. For more information about the parameters, see “Specifying restart parameters” on page 70. For more information about restarting an apply request, see “Restarting an apply request” on page 72.

**Specifying restart parameters**

Use this procedure to specify the appropriate restart parameters for your apply request.

**Before you begin**

The restart table that you specify in this procedure must exist when you run the apply request. Use the instructions in “Creating a restart table” on page 64 to create a restart table.

If you add the restart parameters to a configuration file, they will be used for all requests that refer to that configuration file. Ensure that the restart parameters which you specify are valid for all requests that use the configuration. For more information about the restart parameters, see “Restart parameters” on page 239.

**To add restart parameters to a configuration**

1. Edit the configuration file or apply request, using one of the following methods:

   - Edit a configuration file or create a new one, as described in “Creating and managing configuration files” on page 40, and add the [Restart] section to the file.

   - Edit or create an apply request, as described in “Creating and running apply requests” on page 55:
     
     — For a Oracle or DB2 LUW target, add restart parameters to the command for the apply request.
     
     — For a DB2 mainframe target, add restart parameters to the EXEC statement by using the PARM parameter, or include them in a separate DD statement.
If you include restart parameters in the apply request and the configuration file, High-speed Apply Engine uses the parameters that you specify in the apply request.

2 If you are creating a configuration file, enter the section heading ([Restart] or /Restart/).

3 Specify the name of the restart table with the TableName parameter, or accept the default value, as follows:
   a Specify a unique table name in the format owner.tableName, where owner is the user ID of the restart-table owner, and tableName is the name of the restart table.
   b If you do not include the TableName parameter, High-speed Apply Engine uses the default value, as follows:
      ■ The default alias for the DB2 restart table is BMCAPIT_APTRSCL.
      ■ The default public synonym or alias for Oracle and DB2 LUW is BMCAPE_APERSCL.

4 Specify the RestartType and RestartID parameters as follows:
   ■ To specify parameters before the initial run of the apply request, take one of the following actions:
      —Specify RestartType as New/Restart and specify a value for RestartID (High-speed Apply Engine uses your restart ID value for the apply request). For advantages and disadvantages of this technique, see Step 2.b on page 73.
      —Specify RestartType as New and do not specify a restart ID (High-speed Apply Engine generates a restart ID).
      —To avoid recording restart information for the apply request, specify RestartType as None.

   Note
To maintain data integrity in case of a system failure, BMC Software recommends that you record restart information for your apply requests by specifying the restart type as New/Restart or New.

   ■ To restart a previously interrupted apply request, perform the steps described in “Restarting an apply request” on page 72.

For more information about using restart parameters, see Table 4 on page 75, “RestartID” on page 242, or “RestartType” on page 243.
5 (optional) Specify a value for the Retry parameter. The Retry parameter determines the number of times that High-speed Apply Engine attempts to access the restart table. For more information, see “Retry” on page 245.

6 (optional) Specify a value for the RetainTime parameter. This parameter determines how long High-speed Apply Engine retains restart information for an apply request in the restart table. For more information, see “RetainTime” on page 244.

7 (optional) Specify a value for the CheckpointFrequency parameter. This parameter determines how often High-speed Apply Engine performs checkpoint processing to update the restart table with a statement ID that represents the progress of all agents through the input file. For more information, see “CheckpointFrequency” on page 240.

8 Save the configuration file or apply request.

9 Run the apply request to enable the restart parameters or to restart apply processing.

**Restarting an apply request**

If the operating environment or DBMS fails while the High-speed Apply Engine is running, you can use this procedure to restart the apply request from the point of failure.

**Before you begin**

To successfully restart an apply request, you must

- know the name or alias of the restart table that tracked the original request
- know the restart ID that you specified or that High-speed Apply Engine generated for the job
- specify the appropriate restart parameters in the configuration, as described in “Specifying restart parameters” on page 70
WARNING

High-speed Apply Engine bases its checkpoint and restart processing on the number and sequence of records in the input file. If you modify an input file between the initial run of an apply request and the subsequent restart and you change the number or sequence of records in the file, the results of the restarted run are unpredictable.

In general, do not modify the input file of an apply request between initial run and restart. Modify the configuration file only to the extent required to resolve error conditions. Ensure that changes to the configuration do not alter the distribution of work between agents. For example, do not change the values of configuration parameters such as MaxAgents, DistributionType, PartitionClustering, or RIClustering.

To restart an apply request

1  Review the display output file or your job output to determine whether there is a restart ID that you can use for the apply request. The restart ID is either specified in the configuration or generated by High-speed Apply Engine (in message BMCAPT0097).

   Note

   The display output file is specified with the FileName (DFN) parameter in the [DisplayOutput] section.

   a  If you find a restart ID, skip to Step 2 on page 73.

   b  If no restart ID exists for the request, the job cannot be restarted from the point of failure.

      You must analyze the target tables to determine where apply processing ended. Contact BMC Software Customer Support to determine if you can use conflict parameters effectively to permit a restart of this apply request.

2  Edit the configuration for the apply request.

   a  If you directed High-speed Apply Engine to generate a restart ID (no value was specified for the RestartID parameter), set the RestartType parameter to Restart and the RestartID parameter to the generated restart ID. (Obtain the generated value from the display output file or from message BMCAPT0097).

   b  If you specified a value for the RestartID parameter, verify that the RestartID parameter contains your specified value. Use one of the following techniques:

      ■ Verify that the RestartType parameter is New/Restart.

      The advantage of this technique is that you do not have to change your configuration when you restart the apply request. The disadvantage is that
you can encounter negative consequences if you accidentally run an apply request in error. For example, if you attempt to apply a new input file but the previous run did not complete successfully, you can fail to apply part of the input file. If you attempt to restart the previous input file but the previous run did complete successfully, you can encounter -803 SQL codes, or corrupt the data in your database.

- Set the RestartType parameter to Restart.

The advantage of this technique is that you encounter fewer negative consequences when you run an apply request in error. The disadvantage is that you must change your configuration to restart an apply request.

3 Verify that the restart table is available for the apply request.

4 Submit the apply request to restart apply processing, as follows:

a For DB2 mainframe targets, use the procedure in “Creating and running an apply request for DB2 on mainframe targets” on page 55.

b For DB2 LUW targets, use the procedure in “Creating and running an apply request for DB2 LUW” on page 58.

c For Oracle targets, use the procedure in “Creating and running an apply request for Oracle” on page 60.

For more information about using restart parameters, see “RestartID” on page 242 or “RestartType” on page 243.

Summary of restart parameter usage

The following table summarizes two approaches for specifying and using restart parameters. One approach allows the High-speed Apply Engine to generate the restart ID; the other approach uses a restart ID that you specify for each apply request (but not for each run).

For more information about these parameters, see “RestartID” on page 242 or “RestartType” on page 243.
Table 4: Specifying High-speed Apply Engine [Restart] parameters

<table>
<thead>
<tr>
<th>Type of run / restart</th>
<th>Using generated RestartID</th>
<th>Using specified RestartID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Run</td>
<td>RestartType=New</td>
<td>RestartType=New / Restart</td>
</tr>
<tr>
<td></td>
<td>Do not specify RestartID</td>
<td>RestartID=yyyyyy where yyyyyyy represents a unique restart ID that you specify</td>
</tr>
<tr>
<td>Subsequent Run</td>
<td>RestartType=New</td>
<td>RestartType=New / Restart</td>
</tr>
<tr>
<td>(Previous Run Successful)</td>
<td>Do not specify RestartID</td>
<td>RestartID=yyyyyy</td>
</tr>
<tr>
<td>No Restart</td>
<td>RestartType=Restart</td>
<td>RestartType=New / Restart</td>
</tr>
<tr>
<td></td>
<td>RestartID=xxxxxx where xxxxxx represents the restart ID generated during the previous unsuccessful run (provided by message BMCAPT0097)</td>
<td>RestartID=yyyyyy For advantages and disadvantages of this technique, see Step 2.b on page 73.</td>
</tr>
<tr>
<td>Subsequent Run</td>
<td>RestartType=New</td>
<td>RestartType=New</td>
</tr>
<tr>
<td>(Previous Run Unsuccessful)</td>
<td>Do not specify RestartID</td>
<td>RestartID=yyyyyy</td>
</tr>
<tr>
<td>Restart</td>
<td>RestartType=Restart</td>
<td>RestartType=New / Restart</td>
</tr>
<tr>
<td></td>
<td>RestartID=yyyyyy</td>
<td>RestartID=yyyyyy For advantages and disadvantages</td>
</tr>
<tr>
<td></td>
<td>For advantages and disadvantages of this technique, see Step 2.b on page 73.</td>
<td></td>
</tr>
<tr>
<td>Subsequent Run</td>
<td>RestartType=New</td>
<td>RestartType=New</td>
</tr>
<tr>
<td>(Previous Run Unsuccessful)</td>
<td>Do not specify RestartID</td>
<td>RestartID=yyyyyy</td>
</tr>
<tr>
<td>Bypass Restart</td>
<td>RestartType=New</td>
<td>RestartType=New</td>
</tr>
<tr>
<td>(Start at beginning of file)</td>
<td>Do not specify RestartID</td>
<td>RestartID=yyyyyy</td>
</tr>
</tbody>
</table>

Summary of restart parameter usage
Conflict resolution

This chapter contains information about conflict resolution rules and procedures for creating conflict resolution rules.

Overview of conflict resolution

Conflict resolution rules determine how the High-speed Apply Engine responds when a conflict occurs during apply processing. You create conflict resolution rules by defining sets of Code and Action parameters for each conflict to which you want to respond.

The following sections of its configuration file control how High-speed Apply Engine handles conflict resolution:

- [AnyConflict] (“AnyConflict parameters” on page 122)
- [DeleteConflict] (“DeleteConflict parameters” on page 178)
- [DDLConflict] (“DDLConflict parameters” on page 177)
- [InsertConflict] (“InsertConflict parameters” on page 192)
- [UpdateConflict] (“UpdateConflict parameters” on page 279)
- [Conflict] (“Conflict parameters” on page 157)
- [ConflictFile] (“ConflictFile parameters” on page 163)

Consider these points regarding the configuration sections for conflict resolution:

- The rules in the [AnyConflict] section apply to all types of statements (INSERT, UPDATE, DELETE, or EXCHANGE statements, as well as data definition language (DDL) statements. A rule that is defined in one of the other conflict sections, such as the [InsertConflict] section, takes precedence over rules in the [AnyConflict] section.
If a statement does not start with INSERT, UPDATE, or DELETE, High-speed Apply Engine will use [DDLConflict] rules if they are defined. High-speed Apply Engine uses [DDLConflict] rules to process EXCHANGE statements.

The [Conflict] section describes general processing options for conflict management, particularly for retry processing.

The [ConflictFile] section defines the conflict file or files where High-speed Apply Engine writes output when the Action parameter of a conflict resolution rule is DeferStatement or DeferUR.

High-speed Apply Engine tracks the number of conflicts that trigger a conflict resolution rule during an apply request and displays the results in message BMCAPT0222.

Format of conflict resolution rules

A conflict resolution rule in any [xxxConflict] section consists of a Code parameter and one or more Action parameters, as shown in the following figure.

**Figure 13: Format of a conflict resolution rule**

<table>
<thead>
<tr>
<th>Code</th>
<th>Action</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>code</td>
<td>action01</td>
<td>action02</td>
</tr>
</tbody>
</table>

The `code` value defines an Oracle message number, a DB2 SQL code, or a conflict type (for more information, see Table 5 on page 79). The `action` value defines the action that the High-speed Apply Engine takes when it encounters the conflict type that is defined by the `code` value (for more information, see Table 6 on page 83).

If you specify more than one action for a code, the actions must be valid in the specified sequence. For example, if `action01` in the preceding figure was specified as Abort, then `action02` could never be performed. The following sections describe the codes and actions that you can define for these parameters.

Several actions cause High-speed Apply Engine to perform a commit action or a rollback action. For more information, see “Actions for conflict resolution” on page 82. The value of the DistributionType parameter or the CommitOnDemand parameter can limit commit or rollback actions to the COMMIT or ROLLBACK statements in your input. High-speed Apply Engine might perform additional commit or rollback actions in response to certain Action parameters of a conflict resolution rule or to resource shortages.
Codes for conflict resolution (SQL codes)

The Code parameter identifies the SQL code, database message, or conflict type that the High-speed Apply Engine receives from the target database when a conflict occurs. You can specify a particular value only once in a configuration section. High-speed Apply Engine ignores any duplicate values.

The following table describes the values that you can specify for the Code parameter. The default actions that are listed in this table show the action that High-speed Apply Engine takes when it encounters the conflict type, but you have not defined a conflict resolution rule for that conflict type.

For more information about the actions that you can specify for the code types, see “Actions for conflict resolution” on page 82.

Table 5: Code parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>DBMS</th>
<th>Description</th>
<th>Default action</th>
</tr>
</thead>
</table>
| `nnnn`     | DB2, Oracle, DB2 LUW  | For DB2 and DB2 LUW, this value indicates an SQL code (SQLCODE) of the type `nnnn`. You can qualify this value with a plus (+) or minus (-) sign to limit the value to a positive or negative SQL code. If you do not specify positive or negative, High-speed Apply Engine assumes that `nnnn` is a positive value. For Oracle, negative SQL codes correspond to error codes listed in the Error Messages documentation. For example, code `-942` would correspond to the 'ORA-00942: table or view does not exist' error condition. | -`nnnn`: Abort  
+`nnnn`: Warn |
<p>| MultipleRows | DB2, Oracle, DB2 LUW | This value indicates conflicts that might occur when multiple rows are changed by an UPDATE or DELETE statement in the target table. If you specify this code in the [InsertConflict] section, High-speed Apply Engine ignores it. | Warn |
| MultiRowConflict | DB2      | This value indicates any conflict that occurs as High-speed Apply Engine processes any insert within a multiple row insert action. For conflicts that occur during a multiple row insert, High-speed Apply Engine resets all code values other than 0 or -911 (TimeOut) to -254 and processes the conflict by using the rule that specifies this code value. Valid actions for this code value are Abort and Retry. If the action is Retry, High-speed Apply Engine retries the multiple row insert that encountered a conflict by performing single row insert actions for all rows. | Retry |</p>
<table>
<thead>
<tr>
<th>Value</th>
<th>DBMS</th>
<th>Description</th>
<th>Default action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negative</td>
<td>DB2, Oracle, DB2 LUW</td>
<td>This value indicates any negative SQL code except those that are subject to the following:</td>
<td>Abort</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ RIConflict</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TimeOut</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ individual -nnnn rule</td>
<td></td>
</tr>
<tr>
<td>NoRows</td>
<td>DB2, Oracle, DB2 LUW</td>
<td>This value indicates conflicts that might occur if no rows are changed by a statement in the apply request. Specifying NoRows is equivalent to specifying the specific SQL codes that result from no affected rows. For example:</td>
<td>Warn</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ For DB2 and DB2 LUW, this value is equivalent to SQL code 100.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ For Oracle, this value is equivalent to SQL code ORA-00100</td>
<td></td>
</tr>
<tr>
<td>Positive</td>
<td>DB2, Oracle, DB2 LUW</td>
<td>This value indicates any positive SQL code except those that are subject to a NoRows rule or an individual nnnn rule.</td>
<td>Warn</td>
</tr>
<tr>
<td>RIConflict</td>
<td>DB2, Oracle, DB2 LUW</td>
<td>This value indicates conflicts that arise from database-defined referential integrity (RI) constraints. Specifying RIConflict is equivalent to specifying the specific SQL codes that result from RI constraints in the target database. For example:</td>
<td>None (High-speed Apply Engine takes action indicated by value of Negative code parameter)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ For DB2 and DB2 LUW, this value is equivalent to SQL codes -530, -531, and -532.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ For Oracle, this value is equivalent to ORA-02291 or ORA-02292.</td>
<td></td>
</tr>
<tr>
<td>SQLWarnx</td>
<td>DB2, DB2 LUW</td>
<td>This value indicates any of the SQL warnings returned to the SQL Communication Area (SQLCA). For DB2 and DB2 LUW, the value x can be from 0 to 9 or the letter A. You can specify this code one time for each value of x.</td>
<td>Warn</td>
</tr>
<tr>
<td>Value</td>
<td>DBMS</td>
<td>Description</td>
<td>Default action</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>TimeOut</td>
<td>DB2, Oracle, DB2 LUW</td>
<td>This value indicates conflicts that can occur when an attempt to update the target database times out. Specifying TimeOut is equivalent to specifying the specific SQL codes that result from a database’s failure to respond. For example: ■ For DB2 and DB2 LUW, this value is equivalent to SQL codes -913, -911, and -904. ■ For Oracle, this value is equivalent to ORA-00060 or ORA-00104.</td>
<td>Retry</td>
</tr>
</tbody>
</table>

For more information, see “Interaction between conflict resolution rules” on page 81 and “Processing order for conflict resolution rules” on page 82.

**Interaction between conflict resolution rules**

Typically, you enter more than one conflict resolution rule in your configuration file. You can specify more than one rule for a given SQL code. For example, you can create a specific rule for the SQL code 100 (Code=100), and a general rule that applies to operations which do not change rows in the target database (Code=NoRows).

*Note*

When you indicate specific codes, remember that some codes might be different, depending on whether the statement is being processed as dynamic or static SQL. You might want to include all applicable codes (for example, SQL codes -302 and -433 represent a column value that is too long for the specified column).

When an SQL code triggers more than one conflict resolution rule, the High-speed Apply Engine attempts to execute *all* actions that are specified in the applicable rules. The rules are processed in order (by their code types), as described in “Processing order for conflict resolution rules” on page 82.

High-speed Apply Engine does *not* execute all of the actions when any of the following conditions exist:

- If you specify a rule for a particular SQL code (Code=nnnn, Code=TimeOut, or Code=RIContlict) and a rule for a positive or negative code type (Code=Negative | Positive), High-speed Apply Engine executes the action of the particular SQL code rule and does *not* execute the action of the Code=Negative | Positive rule.

Consider the Negative and Positive code types as defaults that apply to all numeric SQL codes except those subject to Code=nnnn, Code=TimeOut, or Code=RIContlict rules. (Code=TimeOut and Code=RIContlict resolve to the SQL
codes that represent specific conditions for the database. High-speed Apply Engine treats these rules like rules with particular SQL codes.)

- When the Action parameter of a conflict resolution rule is Abort or Terminate, High-speed Apply Engine does not execute actions that are specified for any rules which are processed after the Abort or Terminate action.

**Processing order for conflict resolution rules**

The order that rules appear in the configuration file is not significant.

The High-speed Apply Engine processes the conflict resolution rules in the following order:

1. rules with nonzero numeric values for the Code parameter
   - Depending on the rules that you specify, this category can be Code=Negative | Positive, Code=nnnn, Code=TimeOut, or Code=RIConflict (TimeOut includes positive or negative values). For more information about these rule types, see “Interaction between conflict resolution rules” on page 81.

2. Code=SQLWarnx (DB2 and DB2 LUW)

3. Code=NoRows | MultipleRows

4. Code=0
   - Rules for SQL code 0 are processed last.

**Actions for conflict resolution**

The value of the Action parameter determines how the High-speed Apply Engine handles the conflict specified by the Code parameter. You can specify one or more actions for each code value.

The following table describes the values that you can specify for the Action parameter. For more information about the code types that you can specify, see “Codes for conflict resolution (SQL codes)” on page 79.
Table 6: Action parameter values

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abort</td>
<td>High-speed Apply Engine rolls back the current unit of recovery (transaction) and terminates the apply request. This value is the default action for Negative conflicts.</td>
</tr>
<tr>
<td></td>
<td>- For distribution by object (“Distribution by object” on page 31), High-speed Apply Engine rolls back all statements processed since the previous commit action, including the statement that generated the conflict.</td>
</tr>
<tr>
<td></td>
<td>- For distribution by UR (“Distribution by unit of recovery” on page 34), High-speed Apply Engine rolls back the entire transaction.</td>
</tr>
<tr>
<td>Continue</td>
<td>High-speed Apply Engine continues processing when it encounters the specified conflict type.</td>
</tr>
<tr>
<td>Defer</td>
<td>High-speed Apply Engine writes the SQL statement that generated the specified conflict type to the conflict file. Processing continues with the next unprocessed statement.</td>
</tr>
<tr>
<td>DeferStatement</td>
<td>(This action is not normally used with distribution by UR because, when you defer one statement in a transaction, and then continue processing other statements, you break the integrity of the transaction.)</td>
</tr>
<tr>
<td>DeferUR</td>
<td>High-speed Apply Engine writes the transaction that generated the specified conflict type to the conflict file. This value is valid only after a Rollback action.</td>
</tr>
<tr>
<td></td>
<td>- For distribution by object, High-speed Apply Engine writes all statements processed since the previous commit action to the conflict file, including the statement that generated the conflict. High-speed Apply Engine continues processing with the next unprocessed statement.</td>
</tr>
<tr>
<td></td>
<td>- For distribution by UR, High-speed Apply Engine writes the entire transaction to the conflict file, including any statements that have not been processed. High-speed Apply Engine continues processing with the next transaction.</td>
</tr>
<tr>
<td>Display</td>
<td>High-speed Apply Engine displays any information available from the DBMS regarding the error.</td>
</tr>
<tr>
<td><strong>Value</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>----------</td>
<td>----------------</td>
</tr>
</tbody>
</table>
| **Retry** | High-speed Apply Engine attempts to apply the statement that generated the conflict again. This value is the default action for TimeOut and MultiRowConflict code values. This action is only valid with the following Code parameters:  
  - Timeout  
  - MultiRowConflict  
  - RIConflict  
  - DB2 SQL codes -913, -911, -904, -530, -531, -532, and -254.  
  - Oracle messages that reflect a timeout or RI conflict condition for the target tables  
  You can specify the number of (or time for) retry attempts by using the Conflict parameters ("Conflict parameters" on page 157). You can also use the Conflict parameters MaxFailedRetries and MaxRetryFail to define the maximum number of retries allowed and the action that the High-speed Apply Engine takes when the maximum number of retries is exceeded.  
  To provide clear output messages when the High-speed Apply Engine retries a statement, add a Warn or Display action before the Retry action in your conflict resolution rules. |
| **Rollback** | High-speed Apply Engine rolls back the current transaction when it encounters the specified conflict type.  
  - For distribution by object, High-speed Apply Engine rolls back all statements since the previous commit action, including the statement that generated the conflict. High-speed Apply Engine continues processing with the next unprocessed statement.  
  - For distribution by UR, High-speed Apply Engine rolls back the entire transaction. High-speed Apply Engine continues processing with the next transaction. |
<p>| <strong>RollbackStatement</strong> | High-speed Apply Engine commits that part of the current transaction that has already been processed and then rolls back the statement that generated the specified conflict type. Processing continues with the next unprocessed statement. |
| <strong>Skip</strong> | High-speed Apply Engine skips the statement that generated the specified conflict type. Processing continues with next unprocessed statement. Normally, the RollbackStatement action precedes the Skip action, so that the statement that caused the conflict is not applied to the target table. |</p>
<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| SkipUR | High-speed Apply Engine skips the unit of recovery that generated the specified conflict type. This value is valid only after a Rollback action.  

- For distribution by object, High-speed Apply Engine skips all statements processed since the previous commit action, including the statement that generated the conflict. High-speed Apply Engine continues processing with the next unprocessed statement.  

- For distribution by UR, High-speed Apply Engine skips the entire transaction, including any statements that have not been processed. High-speed Apply Engine continues processing with the next transaction. |
| Terminate | High-speed Apply Engine commits the current transaction and terminates processing when it encounters the specified conflict type.  

- For distribution by object, High-speed Apply Engine commits the statements processed since the previous commit action, including the statement that generated the conflict.  

- For distribution by UR, High-speed Apply Engine commits the statements processed since the previous commit action, including the statement that generated the conflict, but not including any statements that have not been processed. |
| Warn | High-speed Apply Engine issues a warning message and continues processing when it encounters the specified conflict type. The apply request ends with a return code of 4. This value is the default action for NoRows, Positive, SQLWARNx and MultipleRows conflicts. |

The following actions cause High-speed Apply Engine to perform an implied commit action:

- Defer  
- DeferStatement  
- DeferUR  
- RollbackStatement  
- Skip  
- SkipUR  
- Terminate
The following actions cause High-speed Apply Engine to perform an implied rollback action:

- Rollback
- Abort

Creating conflict resolution rules

You can add conflict resolution rules to the configuration to respond to SQL codes that occur during an apply request. Use the procedures in this section to help you define rules that are appropriate for your environment and your input data.

The following figure shows the relationships described in this procedure.

Figure 14: Creating conflict resolution rules
Before you begin

For more information about the parameters mentioned in these procedures, see “Configuration parameters” on page 103. For more information about the valid SQL code types and action types, see “Overview of conflict resolution” on page 77.

This procedure contains the following basic activities:

- To define the actual conflict resolution rules, define sets of Code and Action parameters in the conflict sections. The following conflict sections are based on the type of SQL statement that generates a conflict:
  - [AnyConflict]
  - [InsertConflict]
  - [UpdateConflict]
  - [DeleteConflict]
  - [DDLConflict]

- (optional) You can define general processing options for all conflict resolutions rules in the [Conflict] section. (If you do not explicitly define the parameters in this section, High-speed Apply Engine uses default values.)

- (optional) If any of your conflict resolution rules specify an Action parameter with a value of Defer, DeferStatement, or DeferUR, you can define parameters for the conflict file (or files) where High-speed Apply Engine writes the deferred output. (If you do not explicitly define the parameters in this section, High-speed Apply Engine uses default values.)

To define conflict resolution rules

1 To define conflict resolution rules, add your desired conflict section (or sections) to the configuration file:

- The [AnyConflict] section describes how High-speed Apply Engine handles conflicts for any SQL statement type in the apply request.

- The other conflict sections specify how High-speed Apply Engine handles conflicts for their respective statement types. The values you specify in these sections override the values in the [AnyConflict] section. The other conflict sections include
  - [InsertConflict]
  - [UpdateConflict]
To define the [Conflict] section

The [Conflict] section contains general processing options for conflict resolution. For more information, see “Conflict parameters” on page 157. This section is optional. If you do not include the [Conflict] section, the High-speed Apply Engine uses the default values for the parameters in the following steps:

1. Create or edit a configuration file as described in “Creating or editing a configuration file” on page 41.

2. Add the [Conflict] section to the configuration file.

3. Specify the RetryLimit parameter to indicate how High-speed Apply Engine keeps track of the retry attempts that are defined by the RetryValue parameter, as follows:

   - Time indicates that High-speed Apply Engine retries the conflict resolution rules as many times as necessary until it reaches the number of seconds that is specified by the RetryValue parameter.
Count indicates that High-speed Apply Engine retries the conflict resolution rules until it reaches the *number of attempts* that is specified by the RetryValue parameter. Count is the default value.

4 Specify the RetryValue parameter to define a number of seconds or a number of attempts for the apply request.

The default RetryValue is five. If you specify RetryLimit as Time, consider specifying a larger RetryValue.

5 Specify the RetryFail parameter to indicate the action that High-speed Apply Engine takes after all retry attempts do not resolve the conflict.

The valid values for RetryFail are as follows:

- Abort rolls back the current unit of work before ending apply processing. Abort is the default value.
- Defer writes the SQL statement that generated the conflict type to the conflict file. Processing continues with the next unprocessed statement.
- Skip skips the SQL statement that generated the conflict. Processing continues with the next unprocessed statement.
- Terminate commits the current unit of work before ending apply processing.

6 Specify the MaxFailedRetries parameter to indicate the maximum number of failed retries the High-speed Apply Engine allows. The default value is 5.

7 Specify the MaxRetryFail parameter to indicate what action the High-speed Apply Engine takes when MaxFailedRetries is exceeded. The default value is Abort.

8 Save the configuration file.

**To define the [ConflictFile] section for output to file**

If any of your conflict resolution rules specify an action of Defer, DeferStatement, or DeferUR, you can define parameters for the conflict file (or files) where the High-speed Apply Engine writes the deferred output. (This section is optional, High-speed Apply Engine uses default values if you do not explicitly define the parameters in this section.) To specify that deferred-conflict output be written to a file, proceed with the following steps:

1 Add the [ConflictFile] section name to the configuration file.
The [ConflictFile] section contains file name and allocation information for the deferred conflict files. For more information, see “ConflictFile parameters” on page 163.

2 Specify the FileNameModel parameter that High-speed Apply Engine uses to name the conflict files.

The FileNameModel parameter accepts variables and text constants that High-speed Apply Engine uses to dynamically create a name for one or more conflict files. The model that you specify must resolve to a unique name for each file that High-speed Apply Engine creates. For more information about this parameter, see “FileNameModel” on page 165.

3 Specify the SingleFile parameter to indicate whether High-speed Apply Engine creates one conflict file for an apply request or a conflict file for each active agent:

- Yes creates a single deferred-conflict file for the entire apply request.
- No creates a deferred-conflict file for each agent that processes the apply request.

For more information about the SingleFile parameter, see “SingleFile” on page 172.

4 For DB2 mainframe targets only, specify the DISP parameter and allocation parameters for the conflict files:

1 Specify the DISP parameter to indicate the disposition for the conflict files:

- OLD if the data is not shared.
- SHR if the data set exists and is shared.
- NEW (default) if the data set is new. High-speed Apply Engine creates a new data set each time it processes the apply request.

2 Specify the allocation parameters for the conflict files:

- If you want to use DFSMS to manage your data sets, you can specify the DATACLAS, STORCLAS, or MGMTCLAS parameters to define the data set allocation requirements for the conflict files.
- If you have already created a conflict file, you can use the LIKE parameter to define the data set allocation for additional conflict files.
- Otherwise, you can specify the UNIT and VOLSER parameters to define the data set allocation requirements for the conflict files.
5  Save the configuration file.

**If a problem occurs**

If you receive error messages about the object names specified in any of the conflict resolution sections, check the configuration file for the following common problems:

- Verify that the [ConflictFile] parameters create appropriate file names, and that the files can be allocated as defined.

- Verify that the conflict resolution rules can be performed in the order that you specified them. For example, if an Abort action is specified before a DeferStatement action, the DeferStatement action will never be reached.
Configuration file format

This chapter provides an overview of configuration files. For example, it explains the configuration categories (sections) within the file and the required format for the configuration parameters that the file contains.

For more information about the configuration parameters, see “Configuration parameters” on page 103.

Overview of the configuration file's format

A configuration file is a text file that contains processing options for one or more apply requests. The parameters in each configuration file are organized into sections.

Figure 15 on page 94, Figure 16 on page 95, and Figure 17 on page 95 show examples of a configuration file. A configuration file contains the following elements:

- **Section names** indicate the configuration categories that are included in the file. The [Startup] section (“StartUp parameters” on page 251) is the only required configuration section. For more information about specifying section names, see “Sections in configuration files” on page 96.

- **Parameters** specify processing options for the request. Each parameter must appear in the appropriate section of the configuration file. For more information about specifying parameters, see “Format of configuration parameters” on page 97.

- **Comments** can provide information about the configuration file and its uses. For more information about including comments in a configuration file, see “Comments in configuration files” on page 98.

- **Conflict Resolution rules** are special sets of configuration parameters that describe how to handle the conflict situations that can occur during apply processing. For more information about conflict resolution rules, see “Overview of conflict resolution” on page 77.
Lines of input in the configuration file can be up to 1024 bytes long. BMC Software recommends that any parameters, section names, or conflict resolution rules should be complete on one line of the file.

If you need to "split" or "wrap" one logical line of configuration information onto multiple physical lines, use the plus sign (+) as the last character on a line and continue the configuration information in column 1 of the next line. High-speed Apply Engine appends any characters (including spaces) onto the end of the previous line. For example, when DB2 objects have long names the Log Master for DB2 product can generate High-speed Apply Engine object mapping rules on multiple lines by using the plus sign as the last character in column 72.

**Figure 15: Sample configuration file for a DB2 mainframe target**

```
;High-speed Apply Sample Config File
/StartUp/
   FileName=USER14.MIGRATE.LLOG.CNTL
   InputType=LogicalLog
/LiteralLog/
   UpdateColumn=CHANGED
   Qualify=CHANGED
   SQLType=MIGRATE
/Bind/
   PackagePrefix=APTK
   PlanPrefix=APT
   BindOwner=DB2ADM1
   FreeOption=ALL
   Library=APTDBRM
   Disp=SHR
/ObjectMap/
   SourceTable=HR.PAYROLL
   TargetTable=ACC1Q.PAYROLL
   SourceColumn=NAME
   TargetColumn=SURNAME
   SourceColumn=HIREDAT
   TargetColumn=DATE
   SourceColumn=SALARY
   TargetColumn=RATE
/TemporaryFiles/
   STORCLAS=TEMP
/Agent/
   MaxAgents=30
/Restart/
   RestartType=New
/AnyConflict/
   Code=NEGATIVE
   Action=ABORT
   Code=POSITIVE
   Action=WARN
   Code=MULTIPLESROWS
   Action=WARN
   Code=NOROWS
   Action=WARN
   Code=TIMEOUT
   Action=RETRY
   Code=SQLWARN0
   Action=WARN
   Code=SQLWARN1
   Action=WARN
   Code=SQLWARN2
   Action=WARN
   Code=SQLWARN3
   Action=WARN
```
Figure 16: Sample configuration file for DB2 LUW

```
[StartUp]
  UserID=hstr
  Password=hstr
  Filename=test.sql
  Dbmstype=DB2
  Database=Webserver

[DisplayOutput]
  FileName=DB2UDBDisplay.txt

[ErrorOutput]
  FileName=DB2UDBError.txt

[Agent]
  MaxAgents=10

[Conflict]
  Code=Negative
  Action=abort
  Code=Positive
  Action=Warn
  Code=Multiplierows
  Action=Warn
  Code=Norows
  Action=Warn
```

Figure 17: Sample configuration file for Oracle

```
[StartUp]
  Login=user14/pwora@Ora10
  FileName=ora10_update.sql

[Agent]
  MaxAgents=8

[Conflict]
```
Sections in configuration files

Each section in a configuration file contains a group of parameters that describe a particular type of information that High-speed Apply Engine uses for apply processing. Each parameter in the file must be specified in its designated section. The following figure shows the configuration section format.

**Note**
You can also include configuration information in an apply request by using overrides. For more information, see “Overriding parameters in an apply request” on page 51.

**Figure 18: [Configuration] section format**

```plaintext
[SectionName]
parameter01=value01
parameter02=value02
```

When you include a section name in the configuration file or apply request, you must use square brackets ([ ]) or slashes (/ /) as section-name delimiters. You can use only one type of delimiter within a configuration file. The following figure shows examples of both types of delimiter.

**Figure 19: Section name delimiters**

```plaintext
[Agent]
or
/Agent/
```

If you use a dedicated 3270 terminal to access DB2 on a mainframe target, you should use the slash character for section names. If you use a terminal emulator that supports square brackets, verify that all square brackets in a configuration file resolve to only one of the following hexadecimal code pairs:

- The [ character resolves to 0xAD, and the ] character resolves to 0xBD.
- The [ character resolves to 0xBA, and the ] character resolves to 0xBB.
Note
Be particularly cautious if you use cut and paste functions to copy information into a terminal emulator. The characters that you paste into the window might resolve differently than the characters that you type directly into the window. If you mix the different types of square brackets in the same configuration file, High-speed Apply Engine issues an error message for the configuration.

To verify that the brackets are the same throughout a file, display the hexadecimal values for the configuration file and examine the section names for discrepancies.

Format of configuration parameters

Configuration parameters specify processing options for an apply request.

Parameter names and values are specified as follows:

ParameterName=value

The parameter name and value are separated by an equal sign (=). Do not put spaces around the equal sign. Parameter names are not case sensitive; you can use any combination of uppercase and lowercase letters.

Throughout this chapter, abbreviations for the parameters are shown in parentheses following the parameter name. The abbreviations are valid in a configuration file or in an apply request. In some cases, you can use the abbreviation to specify the parameter without its section name. For more information, see “Overriding parameters in an apply request” on page 51.

Some configuration sections require that you specify their parameters in a specific order. The following table shows the section names and provides a page reference for the positional requirements.

Table 7: Configuration sections with positional parameters

<table>
<thead>
<tr>
<th>Section</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AnyConflict</td>
<td>“AnyConflict parameters” on page 122</td>
</tr>
<tr>
<td>DDLConflict</td>
<td>“DDLConflict parameters” on page 177</td>
</tr>
<tr>
<td>DeleteConflict</td>
<td>“DeleteConflict parameters” on page 178</td>
</tr>
<tr>
<td>InsertConflict</td>
<td>“InsertConflict parameters” on page 192</td>
</tr>
<tr>
<td>ObjectCluster</td>
<td>“ObjectCluster parameters” on page 222</td>
</tr>
<tr>
<td>ObjectMap</td>
<td>“ObjectMap parameters” on page 224</td>
</tr>
<tr>
<td>ObjectPartition</td>
<td>“ObjectPartition parameters” on page 230</td>
</tr>
</tbody>
</table>
Comments in configuration files

To enter comments in a configuration file, use a comment character and enter the comment text anywhere to the right of the comment character on the same line. Valid comment characters are a semicolon (;), a pound sign (#), or two hyphens (--). You can start comments in any column of a separate line. The following figure shows an example of comments in a configuration file.

Note
If you include configuration parameters in an apply request, comments must appear on a separate line. If you include a comment on the same line as a configuration parameter, High-speed Apply Engine issues an error message.

Figure 20: Sample comments in a configuration file

```
[Startup]
;This contains SQL updates for the SCHEDULE table.
 FileName=SCHED.UPD.SQL
 SSID=DB2E
#
#This is the plan High-speed Apply Engine uses to run.
 --The plan to access the tables is in the
 --[Bind]section.

[Bind]
 PlanName=APTPLAN
```

Data set and file names in configuration files

Several parameters require that you specify data set or data definition (DD) names, or UNIX or Windows file names. In general, these names must conform to the rules that apply to names on the operating system where the High-speed Apply Engine runs.

File names for UNIX and Windows

The file names that you specify in UNIX and Windows configurations must refer to files that can be accessed or created by the user ID that issues the apply request. If the file does not reside in the current directory, you must specify the complete path as part of the parameter value.

High-speed Apply Engine processes file names exactly as they are specified for UNIX and Windows configurations. If your operating system is case sensitive, you must specify file names with the appropriate case.
You can simplify High-speed Apply Engine operations by using the default configuration file name for your apply requests. The default configuration file name for UNIX and Windows is `apeconfig.ini` (lowercase characters).

High-speed Apply Engine uses these file names if no other configuration file is specified for the request. If the default configuration file contains all of the necessary configuration information, you can initiate apply processing by typing the appropriate command with no parameter qualifiers. For more information, see “Creating and running an apply request for Oracle” on page 60.

### Data set names for mainframe targets

The data set names that you specify in the configuration information for existing data sets must be catalogued or in the JCL. Any new data sets will be catalogued. Data set names must be fully qualified and conform to the naming rules for data sets on your operating system. If you specify an invalid data set name, the apply request terminates when it attempts to allocate the data set.

For mainframe targets, the High-speed Apply Engine processes file names exactly as they are specified. BMC Software recommends that you do not specify mixed case or lowercase data set names.

Some file-name parameters default to DD statements in the JCL if you do not specify a file name. The default for the configuration file is the SYSIN DD statement. The default for the Library parameter is the data set that the APTDBRM DD statement specifies. For more information, see “Creating and running an apply request for DB2 on mainframe targets” on page 55.

### Object names in configuration files

You can specify table and column names in several parameters within a High-speed Apply Engine configuration. This section describes the table-name format and wildcard usage for table and column names.

#### Table names

Several configuration parameters require you to specify table names. The tables that you specify must already exist.

You can specify a table name in one of the formats shown in the following table.
### Table 8: Table name formats

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>owner.tableName</code></td>
<td>To fully qualify a table name, specify the owner and table name, as follows:</td>
</tr>
<tr>
<td></td>
<td>- The <em>owner</em> is the user ID of the table owner.</td>
</tr>
<tr>
<td></td>
<td>- The period is a required separator.</td>
</tr>
<tr>
<td></td>
<td>- The <em>tableName</em> is the DB2, DB2 LUW or Oracle table name.</td>
</tr>
<tr>
<td><code>tableName</code></td>
<td>If you specify a table name without an owner, High-speed Apply Engine uses the user ID that runs the apply request as the owner of the table.</td>
</tr>
</tbody>
</table>

**Note**
The TableName parameter in the [Restart] section accepts a table name, a DB2 alias, a DB2 synonym or an Oracle public synonym. For more information, see “TableName” on page 246.

### Delimited table names

The High-speed Apply Engine accepts ordinary or delimited table names. If you include special characters in your table names, or if you require lowercase or mixed-case alphabetic characters, you must use delimiters around the table name.

**Note**
The TableName parameter in the [Restart] section does not accept delimited table names. You must specify the restart table by using an ordinary name, by using a DB2 alias or synonym, or by using an Oracle public synonym.

The following table shows some examples of delimited table names. The use of delimiters must conform to the following rules:

- High-speed Apply Engine supports single (’) or double ("”) quotation marks as delimiters.
- The *owner* and *tableName* are distinct nodes, and they can be delimited separately. If the entire name is delimited, it will be qualified with the current user ID for the apply request.
- The first delimiter that is encountered will be recognized for the node, and it must be matched by the same delimiter.
- Wildcard characters must be outside of the delimiters to be recognized.
Table 9: Delimited table names

<table>
<thead>
<tr>
<th>Delimited name</th>
<th>Resolved name</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;XYZ's&quot;.&quot;TableA&quot;</td>
<td>XYZ's.TableA</td>
</tr>
<tr>
<td>XYZ&quot;s.TableA</td>
<td>XYZ's.TABLEA</td>
</tr>
<tr>
<td>(not delimited)</td>
<td></td>
</tr>
<tr>
<td>&quot;A DAY.MONTH&quot;</td>
<td>owner.A DAY.MONTH</td>
</tr>
<tr>
<td>&quot;A DAY&quot;.??&quot;NTH&quot;</td>
<td>A DAY.MONTH</td>
</tr>
<tr>
<td></td>
<td>or</td>
</tr>
<tr>
<td></td>
<td>A DAY.NINTH</td>
</tr>
</tbody>
</table>

Wildcards in object names

Some parameters allow you to specify a set of database objects by using wildcard characters as part of a table or column name.

The High-speed Apply Engine supports the wildcard characters shown in the following table. For more information, see “Using wildcards to map object names” on page 226.

Table 10: Wildcard characters

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>The asterisk represents a string of zero or more contiguous characters.</td>
</tr>
<tr>
<td>%</td>
<td>The percent sign represents a string of zero or more contiguous characters.</td>
</tr>
<tr>
<td>?</td>
<td>The question mark represents a single character.</td>
</tr>
</tbody>
</table>

Long object names

In later versions of DB2, the maximum lengths of DB2 object names are increased from previous versions. The High-speed Apply Engine supports the maximum currently usable name length for all DB2 objects.
Configuration parameters

This chapter contains detailed descriptions of the configuration parameters that are available with the High-speed Apply Engine.

Overview

“Quick reference by name” on page 103 lists the parameters in alphabetical order by name, and “Quick reference by section” on page 109 lists them by section. The sections appear in alphabetical order, and the parameters appear alphabetically within each section.

This organization does not imply a similar order within a configuration file. Except as noted in this document, you can order the configuration file for your convenience, including only those sections that apply to your configuration requirements. For more information about the format of a configuration file, see “Configuration file format” on page 93. For more information about creating and managing configuration files, see “Defining configurations” on page 39.

Quick reference by name

The following table lists the configuration parameters of the High-speed Apply Engine by name, section, and platform, and gives a page reference for each parameter description.

Table 11: Configuration parameters by name

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Section</th>
<th>DBMS</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>AnyConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Action</td>
<td>DDLConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Parameter</td>
<td>Section</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------</td>
<td>-----------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>Action</td>
<td>DeleteConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Action</td>
<td>InsertConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Action</td>
<td>UpdateConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>AuthID</td>
<td>Bind</td>
<td>DB2</td>
<td>“AuthID” on page 125</td>
</tr>
<tr>
<td>BindAction</td>
<td>Bind</td>
<td>DB2</td>
<td>“BindAction” on page 126</td>
</tr>
<tr>
<td>BindOwner</td>
<td>Bind</td>
<td>DB2</td>
<td>“BindOwner” on page 129</td>
</tr>
<tr>
<td>BindQualifier</td>
<td>StartUp</td>
<td>DB2</td>
<td>“BindQualifier” on page 253</td>
</tr>
<tr>
<td>CCSIDCompatible</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“CCSIDCompatible” on page 194</td>
</tr>
<tr>
<td>CheckpointFrequency</td>
<td>Restart</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CheckpointFrequency” on page 240</td>
</tr>
<tr>
<td>Cluster</td>
<td>ObjectCluster</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Cluster” on page 223</td>
</tr>
<tr>
<td>Code</td>
<td>AnyConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>Code</td>
<td>DDLConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>Code</td>
<td>DeleteConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>Code</td>
<td>InsertConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>Code</td>
<td>UpdateConflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>CollectionID</td>
<td>Bind</td>
<td>DB2</td>
<td>“CollectionID” on page 130</td>
</tr>
<tr>
<td>Column</td>
<td>ObjectPartition</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Column” on page 234</td>
</tr>
<tr>
<td>CommitOnDemand</td>
<td>CommitTriggers</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CommitOnDemand” on page 150</td>
</tr>
<tr>
<td>CommitTrigger</td>
<td>MonitorTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CommitTrigger” on page 213</td>
</tr>
<tr>
<td>CommitValue</td>
<td>MonitorTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CommitValue” on page 214</td>
</tr>
<tr>
<td>ConnectionVersion</td>
<td>StartUp</td>
<td>Oracle</td>
<td>“ConnectionVersion” on page 253</td>
</tr>
<tr>
<td>Database</td>
<td>StartUp</td>
<td>DB2 LUW</td>
<td>“Database” on page 254</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“DATACLAS” on page 164</td>
</tr>
<tr>
<td>Parameter</td>
<td>Section</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>---------------------------</td>
<td>--------------------------</td>
<td>---------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>DATACLAS</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“DATACLAS” on page 273</td>
</tr>
<tr>
<td>DataType</td>
<td>Conflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DataType” on page 157</td>
</tr>
<tr>
<td>DateFormat</td>
<td>StartUp</td>
<td>Oracle</td>
<td>“DateFormat” on page 255</td>
</tr>
<tr>
<td>DBMSType</td>
<td>StartUp</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DBMSType” on page 256</td>
</tr>
<tr>
<td>DBPROTOCOL</td>
<td>Bind</td>
<td>DB2</td>
<td>“DBPROTOCOL” on page 131</td>
</tr>
<tr>
<td>DecimalPoint</td>
<td>SQL</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DecimalPoint” on page 249</td>
</tr>
<tr>
<td>DefaultOpts (DOPT)</td>
<td>StartUp</td>
<td>DB2</td>
<td>“DefaultOpts” on page 257</td>
</tr>
<tr>
<td>Directory (DIR)</td>
<td>TemporaryFiles</td>
<td>DB2 LUW</td>
<td>“Directory” on page 273</td>
</tr>
<tr>
<td>DISP</td>
<td>Bind</td>
<td>DB2</td>
<td>“DISP” on page 132</td>
</tr>
<tr>
<td>DISP</td>
<td>ConflictFile</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DISP” on page 164</td>
</tr>
<tr>
<td>DistributionType</td>
<td>StartUp</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DistributionType” on page 258</td>
</tr>
<tr>
<td>FileName</td>
<td>DiagnosticOutput</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName” on page 179</td>
</tr>
<tr>
<td>FileName (CFN)</td>
<td>Configuration</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (CFN)” on page 155</td>
</tr>
<tr>
<td>FileName (DFN)</td>
<td>DisplayOutput</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (DFN)” on page 181</td>
</tr>
<tr>
<td>FileName (EFN)</td>
<td>ErrorOutput</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (EFN)” on page 191</td>
</tr>
<tr>
<td>FileName (IFN)</td>
<td>StartUp</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (IFN)” on page 259</td>
</tr>
<tr>
<td>FileName (LFN)</td>
<td>LogOutput</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (LFN)” on page 208</td>
</tr>
<tr>
<td>FileNameModel</td>
<td>ConflictFile</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileNameModel” on page 165</td>
</tr>
<tr>
<td>Format</td>
<td>SQL</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Format” on page 250</td>
</tr>
<tr>
<td>FreeOption</td>
<td>Bind</td>
<td>DB2, DB2 LUW</td>
<td>“FreeOption” on page 133</td>
</tr>
<tr>
<td>IncludeDDL</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“IncludeDDL” on page 195</td>
</tr>
<tr>
<td>IncludeLOB</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“IncludeLOB” on page 195</td>
</tr>
<tr>
<td>IncludeRI</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“IncludeRI” on page 197</td>
</tr>
<tr>
<td>Parameter</td>
<td>Section</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>--------------------</td>
<td>------------------------</td>
<td>---------------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>IncludeTrigger</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“IncludeTrigger” on page 198</td>
</tr>
<tr>
<td>IncludeXML</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“IncludeXML” on page 199</td>
</tr>
<tr>
<td>IncludeHistory</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“IncludeHistory” on page 200</td>
</tr>
<tr>
<td>InitialAgents</td>
<td>Agent</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InitialAgents” on page 117</td>
</tr>
<tr>
<td>InputMediaType</td>
<td>StartUp</td>
<td>DB2</td>
<td>“InputMediaType” on page 261</td>
</tr>
<tr>
<td>InputTrigger</td>
<td>MonitorTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InputTrigger” on page 215</td>
</tr>
<tr>
<td>InputType (IT)</td>
<td>StartUp</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InputType” on page 262</td>
</tr>
<tr>
<td>InputValue</td>
<td>MonitorTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InputValue” on page 216</td>
</tr>
<tr>
<td>LargeStatementSupport</td>
<td>StartUp</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“LargeStatementSupport” on page 263</td>
</tr>
<tr>
<td>Library</td>
<td>Bind</td>
<td>DB2</td>
<td>“Library” on page 134</td>
</tr>
<tr>
<td>Like</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“LIKE” on page 169</td>
</tr>
<tr>
<td>Like</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“LIKE” on page 274</td>
</tr>
<tr>
<td>LimitKey</td>
<td>ObjectPartition</td>
<td>DB2</td>
<td>“LimitKey” on page 235</td>
</tr>
<tr>
<td>Login</td>
<td>StartUp</td>
<td>DB2 LUW, Oracle</td>
<td>“Login” on page 263</td>
</tr>
<tr>
<td>LogType</td>
<td>LogOutput</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“LogType” on page 209</td>
</tr>
<tr>
<td>MaxAgents</td>
<td>Agent</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxAgents” on page 119</td>
</tr>
<tr>
<td>MaxConflicts</td>
<td>MultiRowInsert</td>
<td>DB2</td>
<td>“MaxConflicts” on page 219</td>
</tr>
<tr>
<td>MaxExecuteImmediateDML</td>
<td>DistributionTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxExecuteImmediateDML” on page 183</td>
</tr>
<tr>
<td>MaxFailedBinds</td>
<td>BindTuning</td>
<td>DB2, DB2 LUW</td>
<td>“MaxFailedBinds” on page 142</td>
</tr>
<tr>
<td>MaxFailedRetries</td>
<td>Conflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxFailedRetries” on page 158</td>
</tr>
<tr>
<td>MaxPackages</td>
<td>BindTuning</td>
<td>DB2, DB2 LUW</td>
<td>“MaxPackages” on page 143</td>
</tr>
<tr>
<td>MaxPrepares</td>
<td>Agent</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxPrepares” on page 120</td>
</tr>
<tr>
<td>MaxRetries</td>
<td>BindTuning</td>
<td>DB2, DB2 LUW</td>
<td>“MaxRetries” on page 145</td>
</tr>
<tr>
<td>MaxRetryFail</td>
<td>Conflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxRetryFail” on page 159</td>
</tr>
<tr>
<td>MaxRows</td>
<td>MultiRowInsert</td>
<td>DB2</td>
<td>“MaxRows” on page 220</td>
</tr>
<tr>
<td>Parameter</td>
<td>Section</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------------</td>
<td>---------------</td>
<td>---------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>MaxVol</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“MAXVOL” on page 170</td>
</tr>
<tr>
<td>MediaType</td>
<td>ConflictFile</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MediaType” on page 169</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“MGMTCLAS” on page 171</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“MGMTCLAS” on page 275</td>
</tr>
<tr>
<td>MultiRowInsert</td>
<td>MultiRowInsert</td>
<td>DB2</td>
<td>“MultiRowInsert” on page 221</td>
</tr>
<tr>
<td>Order</td>
<td>ObjectPartition</td>
<td>DB2</td>
<td>“Order” on page 235</td>
</tr>
<tr>
<td>Overlap</td>
<td>LogOutput</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Overlap” on page 210</td>
</tr>
<tr>
<td>PackageName</td>
<td>Bind</td>
<td>DB2</td>
<td>“PackageName” on page 135</td>
</tr>
<tr>
<td>PackagePrefix</td>
<td>Bind</td>
<td>DB2, DB2 LUW</td>
<td>“PackagePrefix” on page 136</td>
</tr>
<tr>
<td>PackageUsage</td>
<td>BindTuning</td>
<td>DB2, DB2 LUW</td>
<td>“PackageUsage” on page 146</td>
</tr>
<tr>
<td>Partition</td>
<td>ObjectPartition</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Partition” on page 236</td>
</tr>
<tr>
<td>PartitionClustering</td>
<td>DistributionTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“PartitionClustering” on page 184</td>
</tr>
<tr>
<td>PartitionDistribution</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“PartitionDistribution” on page 202</td>
</tr>
<tr>
<td>Password</td>
<td>StartUp</td>
<td>DB2 LUW</td>
<td>“Password” on page 265</td>
</tr>
<tr>
<td>PlanName</td>
<td>Bind</td>
<td>DB2</td>
<td>“PlanName” on page 136</td>
</tr>
<tr>
<td>PlanName (PLAN)</td>
<td>StartUp</td>
<td>DB2</td>
<td>“PlanName” on page 266</td>
</tr>
<tr>
<td>PlanPrefix</td>
<td>Bind</td>
<td>DB2</td>
<td>“PlanPrefix” on page 137</td>
</tr>
<tr>
<td>PublicPlan</td>
<td>StartUp</td>
<td>DB2</td>
<td>“PublicPlan” on page 266</td>
</tr>
<tr>
<td>Qualify</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“Qualify” on page 201</td>
</tr>
<tr>
<td>Release</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“RELEASE” on page 172</td>
</tr>
<tr>
<td>RestartID</td>
<td>Restart</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RestartID” on page 242</td>
</tr>
<tr>
<td>RestartType</td>
<td>Restart</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RestartType” on page 243</td>
</tr>
<tr>
<td>RetainTime</td>
<td>Restart</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetainTime” on page 244</td>
</tr>
<tr>
<td>Retry</td>
<td>Restart</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Retry” on page 245</td>
</tr>
<tr>
<td>RetryDelay</td>
<td>Conflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryDelay” on page 159</td>
</tr>
<tr>
<td>Parameter</td>
<td>Section</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>-------------------------</td>
<td>--------------------</td>
<td>-----------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>RetryFail</td>
<td>Conflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryFail” on page 160</td>
</tr>
<tr>
<td>RetryLimit</td>
<td>Conflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryLimit” on page 161</td>
</tr>
<tr>
<td>RetryValue</td>
<td>Conflict</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryValue” on page 162</td>
</tr>
<tr>
<td>RIClustering</td>
<td>DistributionTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RIClustering” on page 186</td>
</tr>
<tr>
<td>RIRecursionLimit</td>
<td>DistributionTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RIRecursionLimit” on page 187</td>
</tr>
<tr>
<td>ScanInterval</td>
<td>MonitorTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“ScanInterval” on page 217</td>
</tr>
<tr>
<td>SimpleTSClustering</td>
<td>DistributionTuning</td>
<td>DB2</td>
<td>“SimpleTSClustering” on page 189</td>
</tr>
<tr>
<td>SingleFile</td>
<td>ConflictFile</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“SingleFile” on page 172</td>
</tr>
<tr>
<td>Sort</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“Sort” on page 203</td>
</tr>
<tr>
<td>SourceColumn (SC)</td>
<td>ObjectMap</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“SourceColumn” on page 227</td>
</tr>
<tr>
<td>SourceTable (ST)</td>
<td>ObjectMap</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“SourceTable” on page 228</td>
</tr>
<tr>
<td>SPACE</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“SPACE” on page 173</td>
</tr>
<tr>
<td>SPACE</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“SPACE” on page 275</td>
</tr>
<tr>
<td>SpacePri</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“SpacePri” on page 174</td>
</tr>
<tr>
<td>SpacePri</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“SpacePri” on page 276</td>
</tr>
<tr>
<td>SpaceSec</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“SpaceSec” on page 174</td>
</tr>
<tr>
<td>SpaceSec</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“SpaceSec” on page 277</td>
</tr>
<tr>
<td>SQLTerminator</td>
<td>StartUp</td>
<td>DB2, Oracle</td>
<td>“SQLTerminator” on page 267</td>
</tr>
<tr>
<td>SQLType</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“SQLType” on page 204</td>
</tr>
<tr>
<td>SSID</td>
<td>StartUp</td>
<td>DB2</td>
<td>“SSID” on page 268</td>
</tr>
<tr>
<td>StatementCount</td>
<td>BindTuning</td>
<td>DB2, DB2 LUW</td>
<td>“StatementCount” on page 147</td>
</tr>
<tr>
<td>StatementCount</td>
<td>CommitTriggers</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“StatementCount” on page 152</td>
</tr>
<tr>
<td>StatementUsage</td>
<td>BindTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“StatementUsage” on page 148</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“STORCLAS” on page 175</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“STORCLAS” on page 277</td>
</tr>
<tr>
<td>Parameter</td>
<td>Section</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------</td>
<td>-------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>Synchronous</td>
<td>BindTuning</td>
<td>DB2</td>
<td>“Synchronous” on page 149</td>
</tr>
<tr>
<td>Table</td>
<td>ObjectCluster</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Table” on page 224</td>
</tr>
<tr>
<td>Table</td>
<td>ObjectPartition</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Table” on page 237</td>
</tr>
<tr>
<td>TableName</td>
<td>Restart</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“TableName” on page 246</td>
</tr>
<tr>
<td>TargetColumn (TC)</td>
<td>ObjectMap</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“TargetColumn” on page 229</td>
</tr>
<tr>
<td>TargetTable (TT)</td>
<td>ObjectMap</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“TargetTable” on page 230</td>
</tr>
<tr>
<td>Threshold</td>
<td>LogOutput</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Threshold” on page 211</td>
</tr>
<tr>
<td>Time</td>
<td>CommitTriggers</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Time” on page 153</td>
</tr>
<tr>
<td>TimestampFormat</td>
<td>StartUp</td>
<td>Oracle</td>
<td>“TimestampFormat” on page 269</td>
</tr>
<tr>
<td>TimestampTzFormat</td>
<td>StartUp</td>
<td>Oracle</td>
<td>“TimestampTzFormat” on page 270</td>
</tr>
<tr>
<td>UNIT</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“UNIT” on page 176</td>
</tr>
<tr>
<td>UNIT</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“UNIT” on page 278</td>
</tr>
<tr>
<td>UpdateColumns</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“UpdateColumns” on page 205</td>
</tr>
<tr>
<td>URCheckRI</td>
<td>DistributionTuning</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“URCheckRI” on page 189</td>
</tr>
<tr>
<td>UseLLogSysStrings</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“UseLLogSysStrings” on page 206</td>
</tr>
<tr>
<td>UserID</td>
<td>StartUp</td>
<td>DB2 LUW</td>
<td>“UserID” on page 271</td>
</tr>
<tr>
<td>Value</td>
<td>ObjectPartition</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Value” on page 237</td>
</tr>
<tr>
<td>VOLSER</td>
<td>ConflictFile</td>
<td>DB2</td>
<td>“VOLSER” on page 176</td>
</tr>
<tr>
<td>VOLSER</td>
<td>TemporaryFiles</td>
<td>DB2</td>
<td>“VOLSER” on page 279</td>
</tr>
<tr>
<td>WORKNUM</td>
<td>LogicalLog</td>
<td>DB2</td>
<td>“WORKNUM” on page 207</td>
</tr>
</tbody>
</table>

**Quick reference by section**

Configuration files are organized into sections.
See “Overview of the configuration file's format” on page 93 for a description of the configuration file format. The following table lists the parameters for each section and gives a page reference for each parameter description.

Table 12: Configuration parameters by section

<table>
<thead>
<tr>
<th>Section</th>
<th>Parameter</th>
<th>DBMS</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent Parameters</td>
<td>InitialAgents</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InitialAgents” on page 117</td>
</tr>
<tr>
<td></td>
<td>MaxAgents</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxAgents” on page 119</td>
</tr>
<tr>
<td></td>
<td>MaxPrepares</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxPrepares” on page 120</td>
</tr>
<tr>
<td>AnyConflict</td>
<td>Action</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Parameters</td>
<td>Code</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>Bind Parameters</td>
<td>AuthID</td>
<td>DB2</td>
<td>“AuthID” on page 125</td>
</tr>
<tr>
<td></td>
<td>BindAction</td>
<td>DB2</td>
<td>“BindAction” on page 126</td>
</tr>
<tr>
<td></td>
<td>BindOwner</td>
<td>DB2</td>
<td>“BindOwner” on page 129</td>
</tr>
<tr>
<td></td>
<td>CollectionID</td>
<td>DB2</td>
<td>“CollectionID” on page 130</td>
</tr>
<tr>
<td></td>
<td>DBPROTOCOL</td>
<td>DB2</td>
<td>“DBPROTOCOL” on page 131</td>
</tr>
<tr>
<td></td>
<td>DISP</td>
<td>DB2</td>
<td>“DISP” on page 132</td>
</tr>
<tr>
<td></td>
<td>FreeOption</td>
<td>DB2, DB2 LUW</td>
<td>“FreeOption” on page 133</td>
</tr>
<tr>
<td></td>
<td>Library</td>
<td>DB2</td>
<td>“Library” on page 134</td>
</tr>
<tr>
<td></td>
<td>PackageName</td>
<td>DB2</td>
<td>“PackageName” on page 135</td>
</tr>
<tr>
<td></td>
<td>PackagePrefix</td>
<td>DB2, DB2 LUW</td>
<td>“PackagePrefix” on page 136</td>
</tr>
<tr>
<td></td>
<td>PlanName</td>
<td>DB2</td>
<td>“PlanName” on page 136</td>
</tr>
<tr>
<td></td>
<td>PlanPrefix</td>
<td>DB2</td>
<td>“PlanPrefix” on page 137</td>
</tr>
<tr>
<td>BindTuning Parameters</td>
<td>MaxFailedBinds</td>
<td>DB2, DB2 LUW</td>
<td>“MaxFailedBinds” on page 142</td>
</tr>
<tr>
<td></td>
<td>MaxPackages</td>
<td>DB2, DB2 LUW</td>
<td>“MaxPackages” on page 143</td>
</tr>
<tr>
<td></td>
<td>MaxRetries</td>
<td>DB2, DB2 LUW</td>
<td>“MaxRetries” on page 145</td>
</tr>
<tr>
<td></td>
<td>PackageUsage</td>
<td>DB2, DB2 LUW</td>
<td>“PackageUsage” on page 146</td>
</tr>
<tr>
<td></td>
<td>StatementCount</td>
<td>DB2, DB2 LUW</td>
<td>“StatementCount” on page 147</td>
</tr>
<tr>
<td></td>
<td>StatementUsage</td>
<td>DB2, DB2 LUW</td>
<td>“StatementUsage” on page 148</td>
</tr>
<tr>
<td></td>
<td>Synchronous</td>
<td>DB2</td>
<td>“Synchronous” on page 149</td>
</tr>
<tr>
<td>Section</td>
<td>Parameter</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------</td>
<td>-----------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>CommitTriggers</td>
<td>CommitOnDemand</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CommitOnDemand” on page 150</td>
</tr>
<tr>
<td>Parameters</td>
<td>StatementCount</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“StatementCount” on page 152</td>
</tr>
<tr>
<td></td>
<td>Time</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Time” on page 153</td>
</tr>
<tr>
<td>Configuration</td>
<td>FileName (CFN)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (CFN)” on page 155</td>
</tr>
<tr>
<td>Parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conflict</td>
<td>DataType</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DataType” on page 157</td>
</tr>
<tr>
<td>Parameters</td>
<td>MaxFailedRetries</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxFailedRetries” on page 158</td>
</tr>
<tr>
<td></td>
<td>MaxRetryFail</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxRetryFail” on page 159</td>
</tr>
<tr>
<td></td>
<td>RetryDelay</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryDelay” on page 159</td>
</tr>
<tr>
<td></td>
<td>RetryFail</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryFail” on page 160</td>
</tr>
<tr>
<td></td>
<td>RetryLimit</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryLimit” on page 161</td>
</tr>
<tr>
<td></td>
<td>RetryValue</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetryValue” on page 162</td>
</tr>
<tr>
<td>ConflictFile</td>
<td>DATACLAS</td>
<td>DB2</td>
<td>“DATACLAS” on page 164</td>
</tr>
<tr>
<td>Parameters</td>
<td>DISP</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DISP” on page 164</td>
</tr>
<tr>
<td></td>
<td>FileNameModel</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileNameModel” on page 165</td>
</tr>
<tr>
<td></td>
<td>Like</td>
<td>DB2</td>
<td>“LIKE” on page 169</td>
</tr>
<tr>
<td></td>
<td>MaxVol</td>
<td>DB2</td>
<td>“MAXVOL” on page 170</td>
</tr>
<tr>
<td></td>
<td>MediaType</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MediaType” on page 169</td>
</tr>
<tr>
<td></td>
<td>MGMTCLAS</td>
<td>DB2</td>
<td>“MGMTCLAS” on page 171</td>
</tr>
<tr>
<td></td>
<td>Release</td>
<td>DB2</td>
<td>“RELEASE” on page 172</td>
</tr>
<tr>
<td></td>
<td>SingleFile</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“SingleFile” on page 172</td>
</tr>
<tr>
<td></td>
<td>SPACE</td>
<td>DB2</td>
<td>“SPACE” on page 173</td>
</tr>
<tr>
<td></td>
<td>SpacePri</td>
<td>DB2</td>
<td>“SpacePri” on page 174</td>
</tr>
<tr>
<td></td>
<td>SpaceSec</td>
<td>DB2</td>
<td>“SpaceSec” on page 174</td>
</tr>
<tr>
<td></td>
<td>STORCLAS</td>
<td>DB2</td>
<td>“STORCLAS” on page 175</td>
</tr>
<tr>
<td></td>
<td>UNIT</td>
<td>DB2</td>
<td>“UNIT” on page 176</td>
</tr>
<tr>
<td></td>
<td>VOLSER</td>
<td>DB2</td>
<td>“VOLSER” on page 176</td>
</tr>
<tr>
<td>DDLConflict</td>
<td>Action</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Parameters</td>
<td>Code</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
</tbody>
</table>

Chapter 7  Configuration parameters  111
<table>
<thead>
<tr>
<th>Section</th>
<th>Parameter</th>
<th>DBMS</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteConflict</td>
<td>Action</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Parameters</td>
<td>Code</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>DiagnosticOutput</td>
<td>FileName</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName” on page 179</td>
</tr>
<tr>
<td>Parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DisplayOutput</td>
<td>FileName (DFN)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (DFN)” on page 181</td>
</tr>
<tr>
<td>Parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DistributionTuning</td>
<td>MaxExecuteImmediateDML</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“MaxExecuteImmediateDML” on page 183</td>
</tr>
<tr>
<td>Parameters</td>
<td>PartitionClustering</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“PartitionClustering” on page 184</td>
</tr>
<tr>
<td></td>
<td>RIClustering</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RIClustering” on page 186</td>
</tr>
<tr>
<td></td>
<td>RIRecursionLimit</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RIRecursionLimit” on page 187</td>
</tr>
<tr>
<td></td>
<td>SimpleTSClustering</td>
<td>DB2</td>
<td>“SimpleTSClustering” on page 189</td>
</tr>
<tr>
<td></td>
<td>URCheckRI</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“URCheckRI” on page 189</td>
</tr>
<tr>
<td>ErrorOutput</td>
<td>FileName (EFN)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (EFN)” on page 191</td>
</tr>
<tr>
<td>Parameters</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>InsertConflict</td>
<td>Action</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Parameters</td>
<td>Code</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Code” on page 124</td>
</tr>
<tr>
<td>Section</td>
<td>Parameter</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------</td>
<td>--------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>LogicalLog Parameters</td>
<td>CCSIDCompatible</td>
<td>DB2</td>
<td>“CCSIDCompatible” on page 194</td>
</tr>
<tr>
<td></td>
<td>IncludeDDL</td>
<td>DB2</td>
<td>“IncludeDDL” on page 195</td>
</tr>
<tr>
<td></td>
<td>IncludeLOB</td>
<td>DB2</td>
<td>“IncludeLOB” on page 195</td>
</tr>
<tr>
<td></td>
<td>IncludeRI</td>
<td>DB2</td>
<td>“IncludeRI” on page 197</td>
</tr>
<tr>
<td></td>
<td>IncludeTrigger</td>
<td>DB2</td>
<td>“IncludeTrigger” on page 198</td>
</tr>
<tr>
<td></td>
<td>IncludeXML</td>
<td>DB2</td>
<td>“IncludeXML” on page 199</td>
</tr>
<tr>
<td></td>
<td>IncludeHistory</td>
<td>DB2</td>
<td>“IncludeHistory” on page 200</td>
</tr>
<tr>
<td></td>
<td>Qualify</td>
<td>DB2</td>
<td>“Qualify” on page 201</td>
</tr>
<tr>
<td></td>
<td>PartitionDistribution</td>
<td>DB2</td>
<td>“PartitionDistribution” on page 202</td>
</tr>
<tr>
<td></td>
<td>Sort</td>
<td>DB2</td>
<td>“Sort” on page 203</td>
</tr>
<tr>
<td></td>
<td>SQLType</td>
<td>DB2</td>
<td>“SQLType” on page 204</td>
</tr>
<tr>
<td></td>
<td>UpdateColumns</td>
<td>DB2</td>
<td>“UpdateColumns” on page 205</td>
</tr>
<tr>
<td></td>
<td>UseLLogSysStrings</td>
<td>DB2</td>
<td>“UseLLogSysStrings” on page 206</td>
</tr>
<tr>
<td></td>
<td>WORKNUM</td>
<td>DB2</td>
<td>“WORKNUM” on page 207</td>
</tr>
<tr>
<td>LogOutput Parameters</td>
<td>FileName (LFN)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (LFN)” on page 208</td>
</tr>
<tr>
<td></td>
<td>LogType</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“LogType” on page 209</td>
</tr>
<tr>
<td></td>
<td>Overlap</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Overlap” on page 210</td>
</tr>
<tr>
<td></td>
<td>Threshold</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Threshold” on page 211</td>
</tr>
<tr>
<td>MonitorTuning Parameters</td>
<td>CommitTrigger</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CommitTrigger” on page 213</td>
</tr>
<tr>
<td></td>
<td>CommitValue</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CommitValue” on page 214</td>
</tr>
<tr>
<td></td>
<td>InputTrigger</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InputTrigger” on page 215</td>
</tr>
<tr>
<td></td>
<td>InputValue</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InputValue” on page 216</td>
</tr>
<tr>
<td></td>
<td>ScanInterval</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“ScanInterval” on page 217</td>
</tr>
<tr>
<td>MultiRowInsert Parameters</td>
<td>MaxConflicts</td>
<td>DB2</td>
<td>“MaxConflicts” on page 219</td>
</tr>
<tr>
<td></td>
<td>MaxRows</td>
<td>DB2</td>
<td>“MaxRows” on page 220</td>
</tr>
<tr>
<td></td>
<td>MultiRowInsert</td>
<td>DB2</td>
<td>“MultiRowInsert” on page 221</td>
</tr>
<tr>
<td>Section</td>
<td>Parameter</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------</td>
<td>---------------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>ObjectCluster Parameters</td>
<td>Cluster</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Cluster” on page 223</td>
</tr>
<tr>
<td></td>
<td>Table</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Table” on page 224</td>
</tr>
<tr>
<td>ObjectMap Parameters</td>
<td>SourceColumn (SC)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“SourceColumn” on page 227</td>
</tr>
<tr>
<td></td>
<td>SourceTable (ST)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“SourceTable” on page 228</td>
</tr>
<tr>
<td></td>
<td>TargetColumn (TC)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“TargetColumn” on page 229</td>
</tr>
<tr>
<td></td>
<td>TargetTable (TT)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“TargetTable” on page 230</td>
</tr>
<tr>
<td>ObjectPartition Parameters</td>
<td>Column</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Column” on page 234</td>
</tr>
<tr>
<td></td>
<td>LimitKey</td>
<td>DB2</td>
<td>“LimitKey” on page 235</td>
</tr>
<tr>
<td></td>
<td>Order</td>
<td>DB2</td>
<td>“Order” on page 235</td>
</tr>
<tr>
<td></td>
<td>Partition</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Partition” on page 236</td>
</tr>
<tr>
<td></td>
<td>Table</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Table” on page 237</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Value” on page 237</td>
</tr>
<tr>
<td>Restart Parameters</td>
<td>CheckpointFrequency</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“CheckpointFrequency” on page 240</td>
</tr>
<tr>
<td></td>
<td>RestartID</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RestartID” on page 242</td>
</tr>
<tr>
<td></td>
<td>RestartType</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RestartType” on page 243</td>
</tr>
<tr>
<td></td>
<td>RetainTime</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“RetainTime” on page 244</td>
</tr>
<tr>
<td></td>
<td>Retry</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Retry” on page 245</td>
</tr>
<tr>
<td></td>
<td>TableName</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“TableName” on page 246</td>
</tr>
<tr>
<td>SQL Parameters</td>
<td>DecimalPoint</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DecimalPoint” on page 249</td>
</tr>
<tr>
<td></td>
<td>Format</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“Format” on page 250</td>
</tr>
<tr>
<td>Section</td>
<td>Parameter</td>
<td>DBMS</td>
<td>Reference</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-------------------</td>
<td>---------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>StartUp Parameters (SU)</td>
<td>BindQualifier</td>
<td>DB2</td>
<td>“BindQualifier” on page 253</td>
</tr>
<tr>
<td></td>
<td>ConnectionVersion</td>
<td>Oracle</td>
<td>“ConnectionVersion” on page 253</td>
</tr>
<tr>
<td></td>
<td>Database</td>
<td>DB2 LUW</td>
<td>“Database” on page 254</td>
</tr>
<tr>
<td></td>
<td>DateFormat</td>
<td>Oracle</td>
<td>“DateFormat” on page 255</td>
</tr>
<tr>
<td></td>
<td>DBMSType</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DBMSType” on page 256</td>
</tr>
<tr>
<td></td>
<td>DefaultOpts (DOPT)</td>
<td>DB2</td>
<td>“DefaultOpts” on page 257</td>
</tr>
<tr>
<td></td>
<td>DistributionType</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“DistributionType” on page 258</td>
</tr>
<tr>
<td></td>
<td>FileName (IFN)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“FileName (IFN)” on page 259</td>
</tr>
<tr>
<td></td>
<td>InputMedia-Type</td>
<td>DB2</td>
<td>“InputMedia-Type” on page 261</td>
</tr>
<tr>
<td></td>
<td>InputType (IT)</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“InputType” on page 262</td>
</tr>
<tr>
<td></td>
<td>LargeStatementSupport</td>
<td>DB2, DB2 LUW, Oracle</td>
<td>“LargeStatementSupport” on page 263</td>
</tr>
<tr>
<td></td>
<td>Login</td>
<td>DB2 LUW, Oracle</td>
<td>“Login” on page 263</td>
</tr>
<tr>
<td></td>
<td>Password</td>
<td>DB2 LUW</td>
<td>“Password” on page 265</td>
</tr>
<tr>
<td></td>
<td>PlanName (PLAN)</td>
<td>DB2</td>
<td>“PlanName” on page 266</td>
</tr>
<tr>
<td></td>
<td>PublicPlan</td>
<td>DB2</td>
<td>“PublicPlan” on page 266</td>
</tr>
<tr>
<td></td>
<td>SQLTerminator</td>
<td>DB2, Oracle</td>
<td>“SQLTerminator” on page 267</td>
</tr>
<tr>
<td></td>
<td>SSID</td>
<td>DB2</td>
<td>“SSID” on page 268</td>
</tr>
<tr>
<td></td>
<td>TimestampFormat</td>
<td>Oracle</td>
<td>“TimestampFormat” on page 269</td>
</tr>
<tr>
<td></td>
<td>TimestampTzFormat</td>
<td>Oracle</td>
<td>“TimestampTzFormat” on page 270</td>
</tr>
<tr>
<td></td>
<td>UserID</td>
<td>DB2 LUW</td>
<td>“UserID” on page 271</td>
</tr>
</tbody>
</table>
Agent parameters

[Agent] parameters influence whether and how the High-speed Apply Engine performs multi-threaded execution in your environment. By adjusting these parameters, you can influence how many agents High-speed Apply Engine uses to process your apply request.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>InitialAgents</td>
<td>“InitialAgents” on page 117</td>
</tr>
<tr>
<td>MaxAgents</td>
<td>“MaxAgents” on page 119</td>
</tr>
<tr>
<td>MaxPrepares</td>
<td>“MaxPrepares” on page 120</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [Agent] parameters.

**Figure 21: Example of [Agent] parameter values**

```
[Agent]
InitialAgents=2
MaxAgents=10
MaxPrepares=50
```

High-speed Apply Engine provides multi-threaded execution to increase parallelism and improve the overall performance of an apply request. Using more agents can often improve performance, but several factors affect this general rule.
The distribution of work in your input greatly influences performance. Distribution can reduce or eliminate the advantages of multiple agents. For example, if ninety percent of your input affects one table and your configuration parameters require that the same agent process all updates to that table, additional agents provide little benefit.

High-speed Apply Engine uses only the number of agents that it needs to complete an apply request. You can specify a minimum and maximum number of agents, but the actual number that High-speed Apply Engine uses depends on the following criteria:

— Type of processing that is required by the request
— Number of unique objects that are affected by the request
— How the target objects are grouped (for example, by referential integrity or data sharing requirements)

Be aware of the following points regarding the [Agent] parameters:

— Each agent that High-speed Apply Engine uses requires a separate connection to the target database (for DB2 on mainframe targets, a separate thread on the DB2 subsystem). Consider how your [Agent] parameters match the number of connections available in your environment. In some environments, the number of connections is constrained by workload or policy.

— The AIX operating system limits the amount of shared memory segments that any one process can have. For DB2 LUW on AIX, you can have only up to 8 agents.

— Under distribution by UR, High-speed Apply Engine is more likely to use the maximum number of agents available (especially if your input contains a large number of transactions with many dependencies between them). Use the MaxAgents parameter to prevent High-speed Apply Engine from using more database connections than your environment can tolerate. (For more information, see “MaxAgents” on page 119.)

InitialAgents

The InitialAgents parameter defines the number of agents that the High-speed Apply Engine must start at the beginning of an apply request. The number of agents that you specify depends on the resources which are available on the system where the target database resides.

Attributes

This parameter has the following attributes:
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Agent]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer greater than or equal to 1 and less than or equal to the value of MaxAgents (“MaxAgents” on page 119)</td>
</tr>
<tr>
<td>Default value</td>
<td>depends on the value of MaxAgents</td>
</tr>
<tr>
<td></td>
<td>In general, the default for InitialAgents is one half the value of MaxAgents (rounded down), except when MaxAgents is less than four. For example:</td>
</tr>
<tr>
<td></td>
<td>■ MaxAgents=1, InitialAgents=1</td>
</tr>
<tr>
<td></td>
<td>■ MaxAgents=2 to 5, InitialAgents=2</td>
</tr>
<tr>
<td></td>
<td>■ MaxAgents=6, InitialAgents=3</td>
</tr>
<tr>
<td></td>
<td>■ MaxAgents=7, InitialAgents=3</td>
</tr>
</tbody>
</table>

### Usage

Specify a value representing a number of agents. Consider the following points regarding the InitialAgents parameter:

- Each started agent requires a connection to the target database.

- This parameter defines the minimum number of agents that High-speed Apply Engine requires to start processing. If sufficient connections are not available to start this number of agents, the apply request terminates.

- If you do not have information about the number of database connections available in your environment, BMC Software recommends consulting your database administration personnel before changing the default value of this parameter.

- By setting this parameter higher, you can reduce the small delay that occurs when High-speed Apply Engine starts additional agents during the apply request.

- If you set this parameter too high, you can
  - limit the flexibility of High-speed Apply Engine
— use more database resources than you need

**MaxAgents**

The MaxAgents parameter defines the maximum number of agents that the High-speed Apply Engine can start for the apply request. The number of agents that you specify depends on the resources available on the system where the target database resides.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Agent]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>■ DB2, Oracle, DB2 LUW (except on AIX)</td>
</tr>
<tr>
<td></td>
<td>■ an integer greater than or equal to 1</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW on AIX</td>
</tr>
<tr>
<td></td>
<td>■ an integer from 1 to 8</td>
</tr>
<tr>
<td>Default value</td>
<td>■ DB2, Oracle, DB2 LUW (except on AIX)</td>
</tr>
<tr>
<td></td>
<td>■ 10</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW on AIX</td>
</tr>
<tr>
<td></td>
<td>■ 8</td>
</tr>
</tbody>
</table>

**Usage**

Specify a value representing the maximum number of agents.

Consider the following points regarding the MaxAgents parameter:
High-speed Apply Engine starts with the number of agents specified by the InitialAgents parameter, and then starts as many agents as it needs, up to the number specified by the MaxAgents parameter.

Each started agent requires a connection to the target database. If there are not enough threads available to support the maximum number of agents, High-speed Apply Engine continues processing using the current number of agents, but it does not attempt to start up any more agents.

The BMC Software product LOADPLUS for DB2 passes the value of the APMAXAGENTS option to High-speed Apply Engine as the MaxAgents parameter during an SQLAPPLY load action. For more information, see “LOADPLUS and High-speed Apply Engine” on page 315.

Use this parameter to prevent High-speed Apply Engine from using more database resources than you desire. For example, if your input comes from an ad hoc process, the amount can be greater than expected. Specify this parameter to ensure that High-speed Apply Engine does not use more database connections (and by extension, more resources) than your environment can tolerate.

Under distribution by UR, High-speed Apply Engine is more likely to use the maximum number of agents available (especially if your input contains a large number of transactions with many dependencies between them). Use this parameter to limit the database connections.

You can use this parameter to disable multi-threaded execution. If you specify MaxAgents as 1, High-speed Apply Engine does not use multiple agents. (You must specify MaxAgents as 1 if the value of the CommitOnDemand parameter is Yes.)

There is a limitation with AIX on the amount of shared memory segments that any one process can have. For DB2 LUW on AIX, you can only have up to 8 agents.

### MaxPrepares

The MaxPrepares parameter defines limits on dynamic SQL processing that the High-speed Apply Engine performs during the apply request. It defines the number of prepared statements that High-speed Apply Engine maintains in memory at the same time.

### Attributes

This parameter has the following attributes:
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Agent]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
| Valid values       | ■ DB2
                      0 to 32767
■ DB2 LUW
                      0 to 32767
■ Oracle
                      1 to 32767, not to exceed the maximum number of open cursors allowed for the Oracle instance |
| Default value      | 50                                                                    |

**Usage**

Specify a number that represents the maximum number of prepared statements for High-speed Apply Engine.

Consider the following points regarding the MaxPrepares parameter:

- Set this parameter to a lower value to
  - limit the amount of cached memory that High-speed Apply Engine uses for dynamic SQL processing
  - prevent High-speed Apply Engine from needlessly reusing the cached memory for dynamic SQL processing
    (For example, if your input does not have many repeated distinct statement types, there is no performance benefit to retaining prepared statements in memory.)
  - disable dynamic SQL with prepared statements
    (If you set this value to zero, High-speed Apply Engine will process all statements as EXECUTE IMMEDIATE until it binds static SQL.)

- Set this parameter to a higher value to
— match the dynamic SQL processing of High-speed Apply Engine to the number of distinct statement types in your input

— improve dynamic SQL processing during the time before it binds static SQL

- For DB2 and DB2 LUW, DB2 buffer pool limitations might force prepared statements to be discarded regardless of the value that you specify for MaxPrepares. When it discards prepared statements because of buffer pool limitations, High-speed Apply Engine re-prepares the statement.

  High-speed Apply Engine uses the value of the MaxPrepares parameter to create a database request module (DBRM) or bind file that contains PREPARE/EXECUTE pairs for unique SQL statements in the apply request. If you specify zero, High-speed Apply Engine executes each statement with EXECUTE IMMEDIATE until the statement is bound in a static SQL package (if MaxPackages is greater than 0). For more information, see “MaxPackages” on page 143.

- For Oracle targets, the value of MaxPrepares represents the number of cursors that High-speed Apply Engine opens for each agent. High-speed Apply Engine uses the open cursors to create statement handles for unique SQL statements in the apply request. The value that you specify must be between one and the number of open cursors allowed for your Oracle session. The number of open cursors is determined by the OPEN_CURSORS initialization parameter specified for the Oracle session.

  If you specify a value for MaxPrepares that is greater than the value of OPEN_CURSORS, High-speed Apply Engine uses the value of OPEN_CURSORS for MaxPrepares. For more information about the OPEN_CURSORS parameter, consult the Oracle server documentation.

### AnyConflict parameters

The [AnyConflict] parameters describe how the High-speed Apply Engine handles any conflicts that the apply request generates. You can specify as many codes and actions as required for the types of conflicts that might occur.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Code</td>
<td>“Code” on page 124</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [AnyConflict] parameters.

**Figure 22: Example of [AnyConflict] parameter values**

```
[AnyConflict]
Code=TimeOut
Action=Retry
```
Your configuration file can contain any of the following sections to define different types of conflict resolution rules:

- [AnyConflict]
- [DeleteConflict]
- [DDLConflict]
- [InsertConflict]
- [UpdateConflict]

Be aware of the following points regarding the conflict sections:

- Use the same parameters (Action, Code) in all of these conflict sections.
- *The parameters in the conflict resolution rules sections are positional.* You must specify a Code parameter, and then one or more Action parameters. You can specify as many Code/Action groups as necessary for the apply request.
- You can override the values that you specify in the [AnyConflict] section by specifying values for the other conflict rules sections.

### Action

The Action parameter indicates the action to take for the Code parameter that precedes it. Specify one or more actions for each code.

For more information, see “Actions for conflict resolution” on page 82.

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[AnyConflict] (or any [xxxConflict] section)</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any of the values described in “Actions for conflict resolution” on page 82</td>
</tr>
</tbody>
</table>
The Code parameter indicates the SQL code or code type that the conflict generates when you run the apply request. One or more Action parameters must immediately follow the Code parameter, or the High-speed Apply Engine ignores the code that you specify.

For more information, see “Codes for conflict resolution (SQL codes)” on page 79.

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[AnyConflict] (or any [xxxConflict] section)</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any of the values described in “Codes for conflict resolution (SQL codes)” on page 79</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

### Bind parameters

The [Bind] parameters control the BIND commands that High-speed Apply Engine uses to bind plans and packages on the target DB2 subsystem. These parameters are affected by your environment’s DB2 authorizations. Coordinate your settings for these parameters with the personnel responsible for maintaining DB2 authorizations in your environment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>AuthID</td>
<td>“AuthID” on page 125</td>
</tr>
<tr>
<td>BindAction</td>
<td>“BindAction” on page 126</td>
</tr>
</tbody>
</table>
The following figure shows an example of the [Bind] parameters.

**Figure 23: Example of [Bind] parameter values**

```
[Bind]
  BindOwner=USER3
  DISP=NEW
  FreeOption=All
  Library=APTDBRM
  PlanPrefix=APTL
  PackagePrefix=APTK
```

The following sections describe the parameters that you can specify in the [Bind] section.

**AuthID**

The AuthID parameter specifies an authorization ID that the High-speed Apply Engine uses to bind static SQL for a target DB2 database. The specified authorization ID (possibly a secondary authorization ID) must have the appropriate privileges described in the following paragraphs.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>Attribute</td>
<td>Value</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>valid authorization ID with appropriate bind privileges 1-8 characters long</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

### Usage

Use this parameter to specify a different user ID (or secondary authorization ID) that High-speed Apply Engine uses to bind plans and packages. High-speed Apply Engine runs under this authority only for bind-related activities; for all other processing High-speed Apply Engine runs under the authority of the user ID that runs High-speed Apply Engine.

This parameter is part of one method of granting authorizations and privileges related to plans, packages, and collections. For more information on this and other methods, see “DB2 authorizations for plans, packages, and collections” on page 286.

Specify an authorization ID that has either SYSADM authority or SYSCTRL authority. Be aware of the following points regarding the AuthID parameter:

- Specify either a primary or secondary authorization ID. Ensure that you specify a valid TSO logon ID, not a group ID. If this parameter is not a valid TSO logon ID, security software can issue warning messages or prevent required processing.

- If you specify this parameter, High-speed Apply Engine does not perform any other processing (such as dynamic SQL) at the same time as it binds static SQL. High-speed Apply Engine overrides the value of the Synchronous parameter if necessary, issues message BMC0198, and ensures that no other processing occurs during bind actions.

- Use this parameter to avoid granting privileges over plans, packages, and collections to the user ID that runs High-speed Apply Engine. With this method, you can restrict the privileges to bind plans, bind packages, or update collections to a single or limited number of authorization IDs (possibly secondary authorization IDs).

### BindAction

The BindAction parameter determines the value of the ACTION parameter that the High-speed Apply Engine uses in the DB2 command BIND PLAN. High-speed...
Apply Engine issues the BIND command to bind plans and packages for a target DB2 database. During restart processing High-speed Apply Engine uses a value of Replace for BindAction if BindAction is set to Add in your configuration.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Replace, Add, or Use</td>
</tr>
<tr>
<td>Default value</td>
<td>Replace</td>
</tr>
</tbody>
</table>

**Usage**

Specify one of the following values:

- If you specify Replace, High-speed Apply Engine sets the value of the ACTION parameter to REPLACE. (The new plan replaces any existing plan with the same plan name. If no plan with the same name currently exists in the DB2 catalog, the bind action creates a new plan and corresponding entries in the DB2 catalog.)

- If you specify Add, High-speed Apply Engine sets the value of the ACTION parameter to ADD. (The new plan does not replace an existing plan with the same plan name. If a plan with the same name exists, the bind action terminates with an error message. If no plan with the same name currently exists, the bind action creates a new plan.)

- If you specify Use with a pre-bound plan (“PlanName” on page 136) and package (“PackageName” on page 135), High-speed Apply Engine runs without any binds, and all statements are dynamically executed.

To use the pre-bound plan and package, you must set the following configuration parameters as shown:

```plaintext
[Bind]
PlanName=MyPlan
PackageName=MyPackage
BindAction=Use
```

—You must set PlanName to the pre-bound plan name. You can find the plan name in the bind output displayed by High-speed Apply Engine from the bind job in message BMCAPT0032I.
— You must set PackageName to the pre-bound base application package name. You can find the package name in the bind output displayed by High-speed Apply Engine from the bind job in message BMCAPT0032I.

— You must set BindAction to Use, which indicates that the application plan and package has been bound and should be used to apply the statements. If BindAction=Use is not set, a bind is attempted.

For more information, see “Using BindOwner and a pre-bound plan” on page 288.

- If you specify Use, High-speed Apply Engine avoids binding a plan and uses the existing plan specified by the PlanName parameter to bind only the packages required for the apply request. To use this value, the configuration must meet the following requirements:
  
  — you must specify a value for the CollectionID parameter ("CollectionID" on page 130)
  
  — you should specify a value for the BindOwner parameter ("BindOwner" on page 129)
  
  — you must specify a value for the PlanName parameter ("PlanName" on page 136)
  
  — the plan specified by the PlanName parameter must have been previously bound using special bind options by a user ID with sufficient DB2 authorizations

To bind a plan for use with this value, use a DB2 BIND command similar to the following:

```
BIND PLAN(planName) -
   OWNER(planOwner) -
   KEEPDYNAMIC(YES) -
   PKLIST(BMCAPT.APTREB2,collectionID.*) -
   RELEASE(COMMIT) -
   ACTION(REPLACE) RETAIN -
   ISOLATION(CS) -
   VALIDATE(RUN)
```

This example uses the following variables:

— `planName` is the name that will be specified in the PlanName parameter (for more information, see “PlanName” on page 136)

— `planOwner` is the authorization ID that will be specified in the BindOwner parameter
collectionID is the identifier that will be specified in the CollectionID parameter.

When you specify a value of Use for BindAction, High-speed Apply Engine retains the plan after use, regardless of the value of the FreeOption parameter (for more information, see “FreeOption” on page 133).

**BindOwner**

The BindOwner parameter determines the value of the OWNER parameter that the High-speed Apply Engine uses in the DB2 command BIND PLAN. High-speed Apply Engine issues the BIND command to bind plans and packages for a target DB2 database. The authorization ID that you specify as the value of this parameter must have appropriate privileges to bind plans, bind packages, and administer collections.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
| Valid values  | valid authorization ID with appropriate bind privileges  
|               | 1-128 characters long                                |
| Default value | none                                                 |

**Usage**

Use this parameter to specify a different user ID (or secondary authorization ID) that High-speed Apply Engine uses to bind plans and packages. By using this parameter, you can avoid granting bind privileges to the user ID that runs High-speed Apply Engine.

When the value of the BindAction parameter is Use, BindOwner must be the authorization ID that was used to generate a pre-bound plan that High-speed Apply Engine uses (for more information, see “BindAction” on page 126). The BindOwner parameter is part of one method of granting authorizations and privileges for plans, packages, and collections. For more information about this and other methods, see “DB2 authorizations for plans, packages, and collections” on page 286.
The BMC Software product LOADPLUS for DB2 passes the value of the APOWNER option to High-speed Apply Engine as the BindOwner parameter during an SQLAPPLY load action. For more information, see “LOADPLUS and High-speed Apply Engine” on page 315.

CollectionID

For DB2, the CollectionID parameter specifies a collection name that the High-speed Apply Engine uses in any BIND PACKAGE or FREE PACKAGE commands that it issues.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any valid DB2 collection name</td>
</tr>
<tr>
<td>Default value</td>
<td>value of the PlanName parameter in the [Bind] section (“PlanName” on page 136)</td>
</tr>
</tbody>
</table>

Usage

Use this parameter to limit the scope of the authority that you must grant to either

- the user ID that runs High-speed Apply Engine
- a different user ID (possibly a secondary authorization ID) that the High-speed Apply Engine uses to administer collections.

How you use this parameter depends on how you grant DB2 authority for High-speed Apply Engine to administer collections. For more information, see “DB2 authorizations for plans, packages, and collections” on page 286.

The BMC Software product LOADPLUS for DB2 passes the value of the APCOLLECTION option to High-speed Apply Engine as the CollectionID parameter during an SQLAPPLY load action. For more information, see “LOADPLUS and High-speed Apply Engine” on page 315.
High-speed Apply Engine generates collection names as follows:

- If you specify a value for this parameter, High-speed Apply Engine creates only the collection name you enter. You can grant either user ID listed in the preceding paragraphs authority over only the specific collection name you enter.

- If you do not specify a value, High-speed Apply Engine uses the current plan name as a collection ID. If you specify the PlanName parameter (“PlanName” on page 136), you can grant either user ID authority over only the collection with the same name as the plan name.

- If you do not specify either the CollectionID or PlanName parameters, High-speed Apply Engine dynamically generates a plan name and uses that plan name as a collection name. Because you cannot predict the generated plan name, you might need to grant either user ID authority over all collections. Many DB2 security environments do not permit this type of operation.

**DBPROTOCOL**

For DB2, the DBPROTOCOL parameter enables the High-speed Apply Engine to override a subsystem default setting for the protocol used to update target tables at a remote location by using three-part names.

High-speed Apply is not designed for remote system updating, but can accomplish it with the PRIVATE protocol.

BMC Software might remove support for this parameter and the PRIVATE protocol when High-speed Apply supports only versions of DB2 that no longer allow for this method.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>PRIVATE</td>
</tr>
<tr>
<td>Default value</td>
<td>High-speed Apply does not specify a default value but uses the DB2 subsystem’s ZPARM setting.</td>
</tr>
</tbody>
</table>
**Usage**

High-speed Apply Engine accepts a value of PRIVATE for this parameter to use DB2 PRIVATE protocol to process remote updates that contain three-part names.

---

**Note**

High-speed Apply does not accept a value of DRDA for Distributed Relational Database Architecture for this parameter. Specifying DRDA results in an error. Three-part-name updates will fail with DRDA.

The only support that High-speed Apply provides with DBPROTOCOL DRDA is if the input only updates the local system.

---

**DISP**

For DB2, the DISP parameter specifies the disposition of the data set that you define with the Library parameter.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>NEW, OLD, or SHR</td>
</tr>
<tr>
<td>Default value</td>
<td>SHR, if you specify a value for the Library parameter</td>
</tr>
</tbody>
</table>

**Usage**

The DISP parameter represents the JCL DISP parameter that is used for data set allocation. If you do not specify a value for the Library parameter, the High-speed Apply Engine ignores the value of the DISP parameter. You can specify one of the following values for the DISP parameter:

- NEW indicates that the library data set is new. High-speed Apply Engine creates a new data set each time it processes the apply request.
- OLD indicates that the library data set exists, and that it is not a shared data set.
- SHR indicates that the library data set exists, and that it is a shared data set. This value is the default if you specify a value for the Library parameter.

**FreeOption**

The FreeOption parameter specifies whether to free the DB2 or DB2 LUW plan or packages after the apply request has been completed.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
| Valid values | ■ DB2
  Plan, Packages, None, or All
  ■ DB2 LUW
  Packages, None, or All                                         |
| Default value| All                                                                  |

**Usage**

For DB2, set the FreeOption parameter as follows:

- Specify Plan to free the plan.
- Specify Packages to free the packages.
- Specify None to retain the plan and packages.
- Specify All to free both the plan and the packages. All is the default value.

If the value of BindAction is Use, High-speed Apply Engine retains the plan after use, regardless of the value of FreeOption (for more information, see “BindAction” on page 126).
If you configure High-speed Apply Engine to use a pre-bound plan and package (you set PlanName, PackageName, and BindAction=Use), High-speed Apply Engine retains the plan and package regardless of the value of FreeOption. For more information, see “BindAction” on page 126.

For DB2 LUW, set the FreeOption parameter as follows:

- Specify Packages or All to free the packages.
- Specify None to retain the packages.

**Library**

For DB2, the Library parameter specifies the name of the partitioned data set (PDS) that the High-speed Apply Engine uses to store the database request modules (DBRMs) for the apply request. You can also specify a DD name that appears in the JCL for the apply request.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid PDS or DD name</td>
</tr>
<tr>
<td>Default value</td>
<td>APTDBRM DD statement</td>
</tr>
</tbody>
</table>

**Usage**

You can specify any partitioned data set name or a DD name. When you specify a data set name with the Library parameter, the default value of the DISP parameter becomes SHR.

If you do not specify a data set name for the Library parameter, High-speed Apply Engine uses the data set that is specified in the APTDBRM DD statement. If you do not include that DD statement in the JCL, High-speed Apply Engine creates a temporary data set to store the DBRMs.
If you specify a data set with this parameter (or with the APTDBRM DD statement), maintain the data set to avoid bind problems. Periodically delete old DBRMs that are no longer used by High-speed Apply Engine and compress the data set. If you do not maintain the data set properly, High-speed Apply Engine can encounter bind errors because of insufficient directory space or physical space on DASD storage.

**PackageName**

The PackageName parameter specifies the DB2 package that the High-speed Apply Engine uses during apply processing.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>valid DB2 package name</td>
</tr>
<tr>
<td>Default value</td>
<td>generated package name, based on value of PackagePrefix parameter and 0000</td>
</tr>
</tbody>
</table>

**Usage**

Specify the name of a DB2 package that High-speed Apply Engine can use to dynamically apply SQL to DB2.

- If BindAction is set to Replace or Add, High-speed Apply Engine generates and binds the package.

- If BindAction is set to Use, High-speed Apply Engine uses the pre-bound package; all statements execute as dynamic SQL.

For more information, see “BindAction” on page 126.
PackagePrefix

For DB2 and DB2 LUW, the PackagePrefix parameter specifies the prefix to use for package names that are associated with the apply request. The High-speed Apply Engine uses this prefix and a sequential, numeric suffix as the package name.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>1 to 4 alphanumeric characters. (The first character must be alphabetic.)</td>
</tr>
<tr>
<td>Default value</td>
<td>the apply request ID (generated by High-speed Apply Engine, the request ID is a four-character, alphanumeric string, with a range of AAAA to Z999)</td>
</tr>
</tbody>
</table>

Usage

The format of the package name is PPPPnnnn. PPPP is the value of the PackagePrefix parameter (which you can specify explicitly or by default). The nnnn suffix is a sequential number that High-speed Apply Engine generates automatically. High-speed Apply Engine updates the number for each package that the apply request generates.

You can specify up to four alphanumeric characters for the prefix. Special characters are not allowed.

To prevent potential naming conflicts, allow High-speed Apply Engine to generate the prefix, or use a consistent naming scheme. (Naming conflicts can occur if High-speed Apply Engine processes more than one request with the same PackagePrefix.)

PlanName

The PlanName parameter specifies the DB2 plan that the High-speed Apply Engine uses during apply processing.
**Note**
High-speed Apply Engine does not use this plan to access the DB2 catalog or the restart tables. To specify the plan name that accesses the restart tables, use the PlanName parameter in the [StartUp] section (“PlanName” on page 266).

**Attributes**
This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>1 to 8 alphanumeric characters. (The first character must be alphabetic.)</td>
</tr>
<tr>
<td>Default value</td>
<td>generated plan name, based on value of PlanPrefix parameter and the apply request ID</td>
</tr>
</tbody>
</table>

**Usage**
Specify the name of a DB2 plan that High-speed Apply Engine can use to bind packages for the target tables. If you specify both PlanName and PlanPrefix, High-speed Apply Engine uses the value that you specify for PlanName.

To prevent potential naming conflicts, do not run concurrent jobs that specify the same DB2 application plan.

If you do not specify a value for the CollectionID parameter, High-speed Apply Engine uses the PlanName as the collection ID for any packages that it creates. For more information, see “CollectionID” on page 130 and “DB2 authorizations” on page 26.

If the value of BindAction is Use, High-speed Apply Engine uses (but does not bind) the plan specified by PlanName. For more information, see “BindAction” on page 126.

**PlanPrefix**
The PlanPrefix parameter specifies the prefix that the High-speed Apply Engine uses for the DB2 plan that is generated during apply processing.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Bind]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>1 to 4 alphanumeric characters. (The first character must be alphabetic.)</td>
</tr>
<tr>
<td>Default value</td>
<td>APT</td>
</tr>
</tbody>
</table>

Usage

The format of the plan name is as follows:

**PPPPxxxx**

*PPPP* is the prefix that you specify for the PlanPrefix parameter. The *xxxx* suffix is the apply request ID generated by High-speed Apply Engine. (The request ID is a four-character, alphanumeric string, with a range of AAAA to Z999.)

If you specify both PlanName and PlanPrefix, uses the value that you specify for PlanName.

BindTuning parameters

For DB2 and DB2 LUW, the [BindTuning] parameters determine how the High-speed Apply Engine handles bind processing.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxFailedBinds</td>
<td>“MaxFailedBinds” on page 142</td>
</tr>
<tr>
<td>MaxPackages</td>
<td>“MaxPackages” on page 143</td>
</tr>
<tr>
<td>MaxRetries</td>
<td>“MaxRetries” on page 145</td>
</tr>
<tr>
<td>PackageUsage</td>
<td>“PackageUsage” on page 146</td>
</tr>
<tr>
<td>StatementCount</td>
<td>“StatementCount” on page 147</td>
</tr>
<tr>
<td>StatementUsage</td>
<td>“StatementUsage” on page 148</td>
</tr>
</tbody>
</table>
The following figure shows an example of the [BindTuning] parameters.

**Figure 24: Example of [BindTuning] parameter values**

<table>
<thead>
<tr>
<th>[BindTuning]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synchronous=YES</td>
</tr>
<tr>
<td>MaxPackages=3</td>
</tr>
<tr>
<td>MaxFailedBinds=5</td>
</tr>
<tr>
<td>MaxRetries=2</td>
</tr>
<tr>
<td>StatementCount=200</td>
</tr>
<tr>
<td>StatementUsage=50</td>
</tr>
</tbody>
</table>

High-speed Apply Engine provides two basic groups of [BindTuning] parameters:

- **limits on bind actions for static SQL**
  This group determines how High-speed Apply Engine binds static SQL and when it stops binding static SQL. The group includes the MaxFailedBinds, MaxRetries, and Synchronous parameters. For more information, see the individual parameter descriptions that are included in this section.

- **performance of bind actions for static SQL**
  This group enables you to influence the frequency of bind actions, the size of bound packages, and the overall balance between static SQL and other methods (dynamic SQL with prepared statements and EXECUTE IMMEDIATE). This group includes the MaxPackages, PackageUsage, StatementCount, and StatementUsage parameters.
  When you adjust one parameter, consider the values of the others. Because these parameters are used together, the following paragraphs contain information about their interaction.

### Using [BindTuning] parameters for performance

Consider the following points as you adjust your [BindTuning] parameters. These points cover general principles; the circumstances of your data and your environment can vary. These points assume that static SQL provides a performance benefit in your environment. This assumption is most true when your input contains many repeated statements of the same distinct statement types.

- Use the PackageUsage, StatementCount, and StatementUsage parameters as a group. All three parameters set thresholds that determine when the High-speed Apply Engine performs a bind action.

- When you set a threshold higher, High-speed Apply Engine typically performs fewer bind actions, less frequently, with more distinct statement types in each...
bound package. Conversely, when you set a threshold lower, High-speed Apply Engine performs more bind actions, more frequently, with fewer distinct statement types in each bound package.

- The MaxPackages parameter value should vary inversely with the PackageUsage, StatementCount, and StatementUsage parameters. When the thresholds are low, High-speed Apply Engine binds more packages and MaxPackages should be higher.

- If an apply request includes a template file, the StatementCount and MaxPackages parameters also apply to the template file. Each entry in the template file counts as one occurrence of a distinct statement type. Any packages bound as High-speed Apply Engine processes the template file count against your MaxPackages limit.

- To force High-speed Apply Engine to use static SQL for all input statements, set the Synchronous parameter to Yes and set the StatementCount parameter to 1. If you use this technique, ensure that the MaxPackages value is high enough to accommodate the total number of distinct statement types in your input file.

The following table summarizes the effects of changing values for these parameters.
Table 13: Summary of [BindTuning] parameter usage

<table>
<thead>
<tr>
<th>StatementCount, StatementUsage, and PackageUsage settings</th>
<th>Effects / usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Higher Values</td>
<td>Effects:</td>
</tr>
<tr>
<td></td>
<td>High-speed Apply Engine</td>
</tr>
<tr>
<td></td>
<td>■ performs fewer bind actions.</td>
</tr>
<tr>
<td></td>
<td>■ performs less frequent bind actions.</td>
</tr>
<tr>
<td></td>
<td>■ binds packages that contain more distinct statement types.</td>
</tr>
<tr>
<td></td>
<td>■ binds packages that are larger.</td>
</tr>
<tr>
<td></td>
<td>■ executes more dynamic SQL and less static SQL.</td>
</tr>
<tr>
<td></td>
<td>■ performs bind actions and executes static SQL later in the apply request.</td>
</tr>
<tr>
<td></td>
<td>Usage:</td>
</tr>
<tr>
<td></td>
<td>■ MaxPackages can be set to a lower value (not required; no penalty with higher value).</td>
</tr>
<tr>
<td></td>
<td>■ Use higher values when the environment dictates more dynamic SQL (for example, if bind actions cause other activity to perform poorly).</td>
</tr>
<tr>
<td>StatementCount, StatementUsage, and PackageUsage settings</td>
<td>Effects / usage</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Lower Values</td>
<td>Effects: High-speed Apply Engine</td>
</tr>
<tr>
<td></td>
<td>■ performs more bind actions.</td>
</tr>
<tr>
<td></td>
<td>■ performs more frequent bind actions.</td>
</tr>
<tr>
<td></td>
<td>■ binds packages that contain fewer distinct statement types.</td>
</tr>
<tr>
<td></td>
<td>■ binds packages that are smaller.</td>
</tr>
<tr>
<td></td>
<td>■ executes more static SQL and less dynamic SQL.</td>
</tr>
<tr>
<td></td>
<td>■ performs bind actions and executes static SQL sooner in the apply request.</td>
</tr>
<tr>
<td></td>
<td>Usage:</td>
</tr>
<tr>
<td></td>
<td>■ MaxPackages should be set to a higher value.</td>
</tr>
<tr>
<td></td>
<td>■ Use lower values when environment dictates static SQL (for example, if dynamic SQL performs poorly).</td>
</tr>
<tr>
<td></td>
<td>■ Use lower values when input contains fewer distinct statement types (set StatementCount lower).</td>
</tr>
<tr>
<td></td>
<td>■ Use lower values when input includes many repeated statements of the same distinct statement types (set StatementUsage and PackageUsage lower).</td>
</tr>
</tbody>
</table>

**MaxFailedBinds**

For DB2 and DB2 LUW, the MaxFailedBinds parameter specifies how many bind actions can fail before the High-speed Apply Engine stops processing the apply request.

**Attributes**

This parameter has the following attributes:
### Usage

When a bind action fails for reasons other than unavailable DB2 resources (for more information, see “MaxRetries” on page 145), High-speed Apply Engine counts the action as a failed bind action and continues to process the apply request, using dynamic SQL or EXECUTE IMMEDIATE. The MaxFailedBinds parameter defines the point when High-speed Apply Engine stops processing because of failed bind actions. When High-speed Apply Engine exceeds this limit, it issues message BMCAPT0072 and terminates processing with an error.

Specify a number representing how many failed bind actions High-speed Apply Engine accepts before it terminates processing.

Consider these points regarding the MaxFailedBinds parameter:

- Use this parameter to stop High-speed Apply Engine from processing large amounts of input dynamically when a problem in your environment prevents bind actions from succeeding. Frequent failed bind actions can indicate a serious problem with the target DBMS.

- When a bind action fails because of an unavailable DB2 resource condition and High-speed Apply Engine exceeds the MaxRetries limit (“MaxRetries” on page 145), High-speed Apply Engine does not count the action as a failed bind for purposes of this parameter. High-speed Apply Engine continues processing with dynamic SQL or EXECUTE IMMEDIATE.

- High-speed Apply Engine does count a failed bind action for purposes of this parameter if the bind fails because of allocation errors, I/O errors, or any other conditions that are not related to unavailable DB2 resources.

### MaxPackages

For DB2 and DB2 LUW, the MaxPackages parameter specifies the maximum number of additional packages that the High-speed Apply Engine can add to the base plan.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[BindTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 0 to 9999</td>
</tr>
<tr>
<td>Default value</td>
<td>25</td>
</tr>
</tbody>
</table>

Usage

Use this parameter to expand or limit the amount of static SQL that High-speed Apply Engine can bind and process. The PackageUsage, StatementCount, and StatementUsage parameters determine how many distinct statement types High-speed Apply Engine includes in a package. The MaxPackages parameter determines how many packages High-speed Apply Engine can bind.

Specify a number of packages. Take into consideration the number of distinct input statement types and how quickly you want High-speed Apply Engine to start processing statically rather than dynamically.

- If you set this parameter to zero, High-speed Apply Engine cannot process any static SQL. In general, the lower you set this parameter, the smaller the number of distinct statement types High-speed Apply Engine can process as static SQL (subject to the values of the PackageUsage, StatementCount, and StatementUsage parameters).

- When High-speed Apply Engine has bound the number of packages specified by MaxPackages, it stops binding static SQL. High-speed Apply Engine continues processing. If an input statement type has not already been bound, High-speed Apply Engine processes it using dynamic SQL (either with prepared statements or EXECUTE IMMEDIATE).

- If you set the PackageUsage, StatementCount, and StatementUsage parameters to lower values, High-speed Apply Engine will perform more bind actions more frequently, with fewer distinct statement types in each bind. In this situation, set the The MaxPackages parameter to a higher value to accommodate the higher number of bind actions.

- If your SQL input includes a template file generated by Log Master for DB2, High-speed Apply Engine can bind a package or packages for all distinct
statement types at the beginning of the apply request. The packages that High-speed Apply Engine binds as it processes the template file count against the MaxPackages limit.

- For DB2, the DBRM library must be large enough to hold the DBRM members. High-speed Apply Engine will factor in the MaxPackages parameter if you use a temporary DBRM library. For more information about the use of temporary data sets, see “Library” on page 134.

**MaxRetries**

For DB2 and DB2 LUW, the MaxRetries parameter specifies how many times the High-speed Apply Engine attempts to bind static SQL if the bind action fails due to unavailable DB2 resources (for example, DB2 catalog contention).

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[BindTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 32767</td>
</tr>
<tr>
<td>Default value</td>
<td>5</td>
</tr>
</tbody>
</table>

**Usage**

When a bind action fails because of unavailable DB2 resources, High-speed Apply Engine tries the bind action again, in case the resource (such as the DB2 catalog) becomes available. When the number of retry attempts exceeds this value, High-speed Apply Engine stops attempting to bind static SQL and continues processing with dynamic SQL or EXECUTE IMMEDIATE.

Specify a number representing how many times High-speed Apply Engine retries the bind action before it stops attempting to bind static SQL.

Consider these points regarding the MaxRetries parameter:
Use this parameter to adapt High-speed Apply Engine’s retry attempts to the amount of DB2 catalog contention in your environment. If you have high amounts of catalog contention, set this parameter to a higher value.

The MaxRetries parameter handles only bind actions that fail because of unavailable DB2 resources. High-speed Apply Engine does not retry bind actions that fail because of other reasons.

The MaxFailedBinds parameter handles bind actions that fail for other reasons (for example, allocation failures or I/O errors). High-speed Apply Engine does not count bind actions that fail because of unavailable resources as failed binds for purposes of the MaxFailedBinds parameter. For more information, see “MaxFailedBinds” on page 142).

**PackageUsage**

For DB2 and DB2 LUW, the PackageUsage parameter specifies the total number of times that the High-speed Apply Engine can execute any of the distinct statement types in an unbound package before it performs a bind action.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[BindTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 32767</td>
</tr>
<tr>
<td>Default value</td>
<td>500</td>
</tr>
</tbody>
</table>

**Usage**

Specify a value to represent a number of dynamic execution attempts. (High-speed Apply Engine executes statement types in an unbound package dynamically.)

Specify a lower value to direct High-speed Apply Engine to perform more bind actions more frequently, with fewer distinct statement types in each bound package. For example, if your input includes many repeated statements of the same statement type, specify a lower value to make High-speed Apply Engine bind that statement type and execute it with static SQL sooner.
The PackageUsage parameter works in conjunction with the StatementCount and StatementUsage parameters to determine when to bind a package. You can adjust the value to expedite or delay binding of a package. For more information about using these parameters, see “Using [BindTuning] parameters for performance” on page 139.

**StatementCount**

For DB2 and DB2 LUW, the StatementCount parameter specifies the total number of distinct statement types that the High-speed Apply Engine can add to an unbound package before it performs a bind action.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[BindTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 32767</td>
</tr>
<tr>
<td>Default value</td>
<td>200</td>
</tr>
</tbody>
</table>

**Usage**

Specify a value to represent the number of distinct statement types. Specify a lower value to direct High-speed Apply Engine to perform more bind actions more frequently, with fewer distinct statement types in each bound package. For example, if your input includes a small number distinct statement types, specify a lower value to make High-speed Apply Engine bind the statement types and execute them with static SQL sooner.

Consider the following points as you specify this parameter:

- The StatementCount parameter works with the StatementUsage and PackageUsage parameters to determine when to bind a package. For more information about using these parameters, see “Using [BindTuning] parameters for performance” on page 139.
If an apply request includes a template file, High-speed Apply Engine counts the entries in the template file against your StatementCount value. Each entry in the template file counts as one occurrence of a distinct statement type.

To force High-speed Apply Engine to use static SQL for all input statements, set this parameter to 1 and the Synchronous parameter to Yes. If you use this technique, ensure that the MaxPackages value is high enough to accommodate the total number of distinct statement types in your input file.

**Note**
The total number of statically bound distinct statement types will not exceed StatementCount times MaxPackages.

### StatementUsage

For DB2 and DB2 LUW, the StatementUsage parameter specifies the number of times that the High-speed Apply Engine can execute one distinct statement type in an unbound package before it performs a bind action.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[BindTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 32767</td>
</tr>
<tr>
<td>Default value</td>
<td>200</td>
</tr>
</tbody>
</table>

### Usage

Specify a value to represent the number of dynamic execution attempts. (High-speed Apply Engine executes statement types in an unbound package dynamically.)

Specify a lower value to direct High-speed Apply Engine to perform more bind actions more frequently, with fewer distinct statement types in each bound package. For example, if your input includes many repeated statements of the same statement type, specify a lower value to make High-speed Apply Engine bind that statement type and execute it with static SQL sooner.
The StatementUsage parameter works with the StatementCount and PackageUsage parameters to determine when to bind a package. Adjust the value of this parameter to expedite or delay the binding of a package. For more information about using these parameters, see “Using [BindTuning] parameters for performance” on page 139.

Synchronous

For DB2, the Synchronous parameter specifies whether the High-speed Apply Engine can perform other processing (such as dynamic SQL) at the same time as it binds static SQL.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[BindTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Usage

Use this parameter to adapt bind processing to your environment.

- Specify a value of No to allow High-speed Apply Engine to perform other processing during bind actions. This setting can slightly improve performance because it increases parallelism. It also increases the amount of dynamic SQL processing. However, this setting can slightly increase the possibility of authorization conflicts during bind actions.

- Specify a value of Yes (the default value) to prevent High-speed Apply Engine from performing other processing. This setting reduces the amount of dynamic SQL processing.

Be aware of the following points regarding the Synchronous parameter:

- To force High-speed Apply Engine to use static SQL for all input statements, set this parameter to Yes and set the StatementCount parameter to 1. If you use this
technique, ensure that the MaxPackages value is high enough to accommodate the total number of distinct statement types in your input file.

- If you specify a value for the AuthID parameter, High-speed Apply Engine forces the value of the Synchronous parameter to Yes and issues message BMC0198.

## CommitTriggers parameters

The [CommitTriggers] parameters determine when the High-speed Apply Engine issues SQL COMMIT statements during an apply request.

These parameters apply only when High-speed Apply Engine distributes work between multiple agents based on tables or partitions (when the DistributionType parameter is Object). High-speed Apply Engine ignores these parameters when the DistributionType parameter is UR.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommitOnDemand</td>
<td>“CommitOnDemand” on page 150</td>
</tr>
<tr>
<td>StatementCount</td>
<td>“StatementCount” on page 152</td>
</tr>
<tr>
<td>Time</td>
<td>“Time” on page 153</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [CommitTriggers] section.

**Figure 25: Example of [CommitTriggers] parameter values**

```plaintext
[CommitTriggers]
StatementCount=1000
Time=120
```

The [CommitTriggers] parameters are optional. They apply to each active agent for the apply request. Depending on the apply request, a given agent might not reach the commit threshold until the end of the job.

If you specify both StatementCount and Time parameters, High-speed Apply Engine issues a COMMIT statement when either limit is reached, and resets the counters for both parameters.

### CommitOnDemand

The CommitOnDemand parameter specifies whether the High-speed Apply Engine processes or ignores any COMMIT or ROLLBACK statements included in SQL input (when the value of the DistributionType parameter is Object).
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[CommitTriggers]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

Usage

Use this parameter as one method to make High-speed Apply Engine to honor the original COMMIT or ROLLBACK statements in your SQL input.

- Specify No to direct High-speed Apply Engine to ignore COMMIT or ROLLBACK statements in the SQL input and issue COMMIT statements at points determined by other parameters (such as “StatementCount” on page 152 and “Time” on page 153). This value is the default.

- Specify Yes to direct High-speed Apply Engine to apply COMMIT or ROLLBACK statements to the target database at the same points where the COMMIT statements occur in the SQL input.
  
  — If you specify Yes, you must set the value of the MaxAgents parameter to 1.
  
  — You can specify Yes only for SQL input.
  
  — If you specify Yes, High-speed Apply Engine honors the input COMMIT or ROLLBACK statements, but it might perform additional commit or rollback actions in response to certain Action parameters of a conflict resolution rule or to resource shortages.

This parameter causes High-speed Apply Engine to honor the original COMMIT or ROLLBACK statements with only one agent and for SQL input. To honor these statements with more than one agent or in logical log input, set the DistributionType parameter to UR. For more information, see “DistributionType” on page 258.
StatementCount

The StatementCount parameter specifies the number of SQL statements that an apply agent processes before that agent issues a COMMIT statement.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[CommitTriggers]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 0 to 32767 is recommended</td>
</tr>
<tr>
<td>Default value</td>
<td>1000</td>
</tr>
</tbody>
</table>

Usage

The High-speed Apply Engine ignores this parameter when it distributes work between multiple agents based on units of recovery (when the DistributionType parameter is UR).

Specify a value for StatementCount to indicate the number of SQL statements that each agent of a High-speed Apply Engine request processes before issuing a COMMIT statement.

If you specify zero, High-speed Apply Engine commits all statements as a single transaction in most cases. There are, however, circumstances under which High-speed Apply Engine ignores the statement count and forces commits to occur. These circumstances include

- conflict rules
- use of time as a commit trigger
- internal resource shortages (see Note 1)
- tables specified in input, but nonexistent in target DBMS
- partitioned tables where the partition key is not included in the SQL

Be aware of the following points regarding the StatementCount parameter:
The value of the StatementCount parameter applies to each agent.

**Note**
An internal resource shortage can occur as High-speed Apply Engine runs, depending upon the number of agents and the amount of data that the SQL statements contain.

- If you specify both the StatementCount and Time parameters, High-speed Apply Engine issues a COMMIT statement when either parameter reaches its specified limit. Both counters are reset after a commit or rollback occurs. At the end of a request, High-speed Apply Engine issues a final COMMIT statement as part of termination processing.

- The value of this parameter affects checkpoint and restart processing. For more information, see “Usage” on page 240.

- The BMC Software product LOADPLUS for DB2 passes the value of the APCOMMIT option to High-speed Apply Engine as the StatementCount parameter during an SQLAPPLY load action. For more information, see “LOADPLUS and High-speed Apply Engine” on page 315.

### Time

The Time parameter specifies the number of seconds that can pass before an apply agent issues a COMMIT statement during apply processing.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[CommitTriggers]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 32400</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>
**Usage**

Specify a value for the Time parameter to indicate the number of seconds between COMMIT statements during apply processing. Each time that an agent reaches or exceeds the value of the Time parameter, the agent issues a COMMIT statement.

Be aware of the following points regarding the Time parameter:

- The value of the Time parameter applies to each agent.

- If you specify both the StatementCount and Time parameters, the High-speed Apply Engine issues a COMMIT statement when either parameter reaches its specified limit. Both counters are reset after a commit or rollback occurs. At the end of a request, High-speed Apply Engine issues a final COMMIT statement as part of termination processing.

- High-speed Apply Engine ignores this parameter when it distributes work between multiple agents based on units of recovery (when the DistributionType parameter is UR).

- The value of this parameter affects checkpoint and restart processing. For more information, see “Usage” on page 240.

**Note**

Some conflict actions might cause a commit to occur, overriding the Time value (see “Usage” on page 152).

---

**Configuration parameters**

The [Configuration] section contains the location of the configuration files for the apply request. This section can be specified only in the apply request; you cannot include this section in your configuration file.

If you do not specify a configuration file in the apply request, the High-speed Apply Engine uses a default UNIX or Windows file name, or a DD name. For more information about including a configuration file in the apply request, see “Creating and running apply requests” on page 55.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileName (CFN)</td>
<td>“FileName (CFN)” on page 155</td>
</tr>
</tbody>
</table>
The following figures show sample configuration parameters for z/OS (mainframe targets), for UNIX, and for Windows.

**Figure 26: Example of [Configuration] parameters for mainframe targets**

```
//APTBMAIN EXEC PGM=APTBMAIN,
PARM='CFN=APTCFG'
```

**Figure 27: Example of [Configuration] parameters for UNIX**

```
apebmain CFN=/home/apply/apeconfig.ini
```

**Figure 28: Example of [Configuration] parameters for Windows**

```
apebmain CFN=E:\home\apply\apeconfig.ini
```

### FileName (CFN)

The FileName (CFN) parameter specifies the name of a data set or file that contains configuration parameters for an apply request.

#### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Configuration]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>CFN</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
### Attribute

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB2 a fully qualified data set name or a DD name</td>
</tr>
<tr>
<td>DB2 LUW or Oracle a file name, including the path name if outside of the current directory</td>
</tr>
<tr>
<td>None indicates that there is no configuration file (High-speed Apply Engine uses default values)</td>
</tr>
</tbody>
</table>

### Default values

<table>
<thead>
<tr>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>mainframe targets SYSIN</td>
</tr>
<tr>
<td>UNIX and Windows apeconfig.ini</td>
</tr>
</tbody>
</table>

## Usage

This parameter is frequently specified as an override in the apply request. If you do not specify this parameter in the request, the High-speed Apply Engine attempts to locate a file name, data set name, or DD name with the default value.

Because this parameter accepts a value of None, High-speed Apply Engine ignores any file, data set, or a DD statement with a name of "none" (regardless of uppercase or lowercase characters).

For UNIX and Windows, the file that you specify must exist on the server where you run High-speed Apply Engine. If the file is not in the current directory, you must provide the complete path and file name. For more information, see “Data set and file names in configuration files” on page 98.

For the configuration file, when you do not specify a file name, High-speed Apply Engine uses the default configuration file name (apeconfig.ini).

For mainframe targets, you can specify either a data set name or a DD name, as follows:

- If you specify a name that is longer than eight characters, High-speed Apply Engine treats it as a fully qualified data set name. The data set must already be allocated and cataloged.
If you specify a name that is eight characters or fewer, High-speed Apply Engine treats it as a DD name. The DD name must refer to a DD statement in the same apply request. If the character string is not a valid DD name, High-speed Apply Engine treats it as a data set name.

For the configuration file, you can use the DD statement to refer to multiple configuration files. For more information, see “Multiple configuration files for DB2 on mainframe targets” on page 39.

Conflict parameters

The [Conflict] parameters define general procedures for conflict management. The High-speed Apply Engine uses these parameters to determine how often to retry conflict resolutions, and where to store deferred conflicts. If you do not include this section in the configuration, High-speed Apply Engine uses default values for conflict management.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataType</td>
<td>“DataType” on page 157</td>
</tr>
<tr>
<td>MaxFailedRetries</td>
<td>“MaxFailedRetries” on page 158</td>
</tr>
<tr>
<td>MaxRetryFail</td>
<td>“MaxRetryFail” on page 159</td>
</tr>
<tr>
<td>RetryDelay</td>
<td>“RetryDelay” on page 159</td>
</tr>
<tr>
<td>RetryFail</td>
<td>“RetryFail” on page 160</td>
</tr>
<tr>
<td>RetryLimit</td>
<td>“RetryLimit” on page 161</td>
</tr>
<tr>
<td>RetryValue</td>
<td>“RetryValue” on page 162</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [Conflict] parameters.

Figure 29: Example of [Conflict] parameter values

```
[Conflict]
  RetryFail=abort
  RetryLimit=count
  RetryValue=5
  MaxFailedRetries=5
  MaxRetryFail=Abort
```

DataType

The DataType parameter specifies which types of data that the High-speed Apply Engine sends to the conflict file.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Conflict]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>ID (for Statement Sequence ID)</td>
</tr>
<tr>
<td></td>
<td>Code (for Conflict code)</td>
</tr>
<tr>
<td></td>
<td>SQLCODE (for DBMS SQL code)</td>
</tr>
<tr>
<td></td>
<td>Message (for DBMS SQL message)</td>
</tr>
<tr>
<td></td>
<td>Statement (for SQL statement)</td>
</tr>
<tr>
<td>Default value</td>
<td>all information</td>
</tr>
</tbody>
</table>

Usage

You can specify multiple DataType values. The default is to provide all information.

MaxFailedRetries

The MaxFailedRetries parameter specifies the maximum number of failed retries that the High-speed Apply Engine allows.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Conflict]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 0 to 32767</td>
</tr>
<tr>
<td>Default value</td>
<td>5</td>
</tr>
</tbody>
</table>
Usage

You can specify the maximum number of failed retries that you want High-speed Apply Engine attempt (on a per agent basis). By default, High-speed Apply Engine attempts 5. A value of 0 indicates an unlimited number of retries is allowed.

MaxRetryFail

The MaxRetryFail parameter specifies what action the High-speed Apply Engine performs when MaxFailedRetries is exceeded.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Conflict]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Terminate or Abort</td>
</tr>
<tr>
<td>Default value</td>
<td>Abort</td>
</tr>
</tbody>
</table>

Usage

You can specify what action the High-speed Apply Engine takes when the maximum number of retries (“MaxFailedRetries” on page 158) is reached. The default is Abort.

RetryDelay

The RetryDelay parameter specifies the number of seconds that the High-speed Apply Engine waits between retry attempts for conflict resolution. High-speed Apply Engine uses this parameter only when the Code parameter of a conflict resolution rule is RIConflict (or the individual SQL codes defined by RIConflict) and the Action parameter is Retry.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Conflict]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 0 to 32767, representing a number of seconds</td>
</tr>
<tr>
<td>Default value</td>
<td>0 (no delay)</td>
</tr>
</tbody>
</table>

Usage

Specify the number of seconds that you want High-speed Apply Engine to wait between retry attempts. By default, High-speed Apply Engine does not wait (RetryDelay=0).

Use this parameter only when your apply request encounters time-dependent RI-related conflicts. This parameter changes the timing of any retry attempts for conflict resolution.

For example, assume a conflict resolution rule has a Code parameter of RIConflict. As the apply request runs, one agent attempts to insert a row in a child table before another agent has inserted the required row in the parent table. Conflict occurs. The RetryValue and RetryLimit parameters cause High-speed Apply Engine to retry the child insert action five times (in case the other agent performs the parent insert action), but the conflict is not resolved. You can increase your RetryValue and RetryLimit values, but that action increases the overhead with more retry attempts. Alternately, you can use this parameter to slow down the existing five retry attempts by waiting a given number of seconds between each attempt.

RetryFail

The RetryFail parameter specifies the action that the High-speed Apply Engine takes if all retry attempts for conflict resolution fail.

Attributes

This parameter has the following attributes:
### Usage

Specify Defer, Skip, Terminate, or Abort.

Use this parameter to define the action that High-speed Apply Engine takes when it reaches the limits specified by the RetryLimit and RetryValue parameters. For more information about the Defer, Skip, Terminate, and Abort actions, see “Actions for conflict resolution” on page 82.

### RetryLimit

The RetryLimit parameter indicates the measurement type used for the RetryValue parameter. You can specify whether the limit on conflict resolution retry attempts is based on time or on the number of retry attempts.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Conflict]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Defer, Skip, Terminate, or Abort</td>
</tr>
<tr>
<td>Default value</td>
<td>Abort</td>
</tr>
</tbody>
</table>
Usage

Specify one of the following values for the RetryLimit parameter. For more information about the RetryValue parameter, see “RetryValue” on page 162.

- Time indicates that the High-speed Apply Engine will issue retry attempts until it reaches the number of seconds specified by the RetryValue parameter.

- Count indicates that High-speed Apply Engine will issue retry attempts until it reaches the number of attempts specified by the RetryValue parameter.

The BMC Software product LOADPLUS for DB2 passes the value of the APRETRYLIM option to High-speed Apply Engine as the RetryLimit parameter during an SQLAPPLY load action. For more information, see “LOADPLUS and High-speed Apply Engine” on page 315.

RetryValue

The RetryValue parameter specifies the conflict resolution retry limit for each unit of recovery (transaction) in the request.

When a transaction exceeds the specified limit, the High-speed Apply Engine takes the action specified by the RetryFail parameter. The measurement for this parameter is based on the value that you specify for the RetryLimit parameter. For more information, see “RetryLimit” on page 161.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Conflict]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer that represents either the number of seconds or the number of retry attempts, depending on the value specified for the RetryLimit parameter</td>
</tr>
<tr>
<td>Default value</td>
<td>5</td>
</tr>
</tbody>
</table>
Usage

Specify a value for the number of seconds or the number of attempts that High-speed Apply Engine should retry the conflict resolution rules for each transaction in the apply request.

The BMC Software product LOADPLUS *for DB2* passes the value of the APRETRYVAL option to High-speed Apply Engine as the RetryValue parameter during an SQLAPPLY load action. For more information, see “LOADPLUS and High-speed Apply Engine” on page 315.

ConflictFile parameters

The [ConflictFile] parameters define how the files that contain deferred conflicts are allocated and managed. If you specify a conflict resolution rule with any of the defer actions, the deferred conflicts are stored in the files that you define with the parameters in this section.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATACLAS</td>
<td>“DATACLAS” on page 164</td>
</tr>
<tr>
<td>DISP</td>
<td>“DISP” on page 164</td>
</tr>
<tr>
<td>FileNameModel</td>
<td>“FileNameModel” on page 165</td>
</tr>
<tr>
<td>Like</td>
<td>“LIKE” on page 169</td>
</tr>
<tr>
<td>MediaType</td>
<td>“MediaType” on page 169</td>
</tr>
<tr>
<td>MaxVol</td>
<td>“MAXVOL” on page 170</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>“MGMTCLAS” on page 171</td>
</tr>
<tr>
<td>Release</td>
<td>“RELEASE” on page 172</td>
</tr>
<tr>
<td>SingleFile</td>
<td>“SingleFile” on page 172</td>
</tr>
<tr>
<td>SPACE</td>
<td>“SPACE” on page 173</td>
</tr>
<tr>
<td>SpacePri</td>
<td>“SpacePri” on page 174</td>
</tr>
<tr>
<td>SpaceSec</td>
<td>“SpaceSec” on page 174</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>“STORCLAS” on page 175</td>
</tr>
<tr>
<td>UNIT</td>
<td>“UNIT” on page 176</td>
</tr>
<tr>
<td>VOLSER</td>
<td>“VOLSER” on page 176</td>
</tr>
</tbody>
</table>
The following figure shows an example of the [ConflictFile] parameters.

**Figure 30: Example of [ConflictFile] parameter values**

<table>
<thead>
<tr>
<th>[ConflictFile]</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileNameModel=&amp;SYSUID..APT.&amp;REQUESTID..D&amp;DATE..T&amp;TIME.</td>
</tr>
<tr>
<td>SingleFile=No</td>
</tr>
<tr>
<td>Storclas=CONFLICT</td>
</tr>
</tbody>
</table>

**DATAACLAS**

For z/OS, the DATAACLAS parameter identifies a data class defined by DFSMS. The High-speed Apply Engine uses the data class when it allocates conflict files.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a data class that is defined in DFSMS for your environment</td>
</tr>
<tr>
<td>Default value</td>
<td>the value of the DATAACLAS parameter specified in the [TemporaryFiles] section</td>
</tr>
</tbody>
</table>

**Usage**

The data class defines data attributes for DFSMS-managed data sets. This parameter applies only to systems that use DFSMS. If your environment does not use DFSMS, High-speed Apply Engine specifies the DATAACLAS parameter in its allocation request, but the operating system does not honor its value.

**DISP**

For z/OS, the DISP parameter specifies the disposition of the files allocated for deferred conflicts.
### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>NEW, OLD, or SHR</td>
</tr>
<tr>
<td>Default value</td>
<td>NEW</td>
</tr>
</tbody>
</table>

### Usage

The DISP parameter specifies the disposition of the conflict file. You can specify one of the following values for the DISP parameter:

- NEW indicates that the data set is new. The High-speed Apply Engine creates a new data set each time it processes the apply request. NEW is the default value.
- OLD indicates that the data set exists and that it is not a shared data set.
- SHR indicates that the data set exists and that it is a shared data set.

**Note**

When you restart an apply request, High-speed Apply Engine ignores this value and conflict information is appended to the original conflict file, which must exist.

### FileNameModel

The FileNameModel parameter defines the file name that the High-speed Apply Engine uses to store information about conflicts that you specify as deferred. You can use symbolic variables as part of the file name.

### Attributes

This parameter has the following attributes:
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>- Text, symbolic variables, or both. Specify variables in uppercase.</td>
</tr>
<tr>
<td></td>
<td>- mainframe targets</td>
</tr>
<tr>
<td></td>
<td>The name must resolve to a maximum length of 44 characters.</td>
</tr>
<tr>
<td></td>
<td>This parameter supports GDG and PDS data set names.</td>
</tr>
<tr>
<td>Default value</td>
<td>- mainframe targets</td>
</tr>
<tr>
<td></td>
<td>&amp;SYSUID..BMCAPT..REQUESTID..D&amp;DATE..T&amp;TIME.</td>
</tr>
<tr>
<td></td>
<td>- UNIX and Windows</td>
</tr>
<tr>
<td></td>
<td>&amp;SYSUID..BMCAPE..REQUESTID..D&amp;DATE..T&amp;TIME.</td>
</tr>
</tbody>
</table>

**Usage**

You can use symbolic variables, character strings, or a combination of both as the value of the FileNameModel parameter. For an example of the file-name model, see “ConflictFile parameters” on page 163.

In some circumstances, you might need to re-apply SQL statements that have been previously written to a conflict file. If you do, remember to ensure that the name of the former conflict file (now being read as the input file) is different than the name of the current conflict file.

*Table 14 on page 167* describes the symbolic variables available for High-speed Apply Engine. The variables that you specify are replaced with actual values when you run an apply request. They are resolved according to standard substitution rules.
Note
If the value for SingleFile is set to No, the agent ID will be appended to the name. For more information, see “SingleFile” on page 172.

Symbolic variables have the following format:

- All variables require an ampersand (&) prefix.
- Qualifiers in the model are separated by a period.
- If necessary for substitution, use a trailing period to delimit the variable name.
- For mainframe targets
  - the resulting name must be less than or equal to 44 characters long.
  - date and time variables must be prefixed by an alphabetic character.

You must specify the variables in uppercase. You can specify character strings in lowercase or mixed case if you enclose the value of FileNameModel in quotation marks.

Table 14: Symbolic variables for the FileNameModel parameter

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;REQUESTID</td>
<td>the job ID assigned by the system when the apply request is processed</td>
</tr>
</tbody>
</table>
| &DATE | the current date in the format yyyyddd, where
  - yyyy is the year
  - ddd is the numeric day (001 to 366)
    For mainframe targets, when the &DATE variable is part of the file name model, you must include an alphabetic prefix (for example, D&DATE). |
### ConflictFile parameters

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
</table>
| &DBSERVER      | Oracle: the SID of the current Oracle server  
If you specify a value for the Login parameter, High-speed Apply Engine uses that value to determine the SID. If you do not specify the Login parameter, High-speed Apply Engine determines the current SID from the ORACLE_SID environment variable.  

DB2 for mainframe targets: the SSID of the current DB2 subsystem  
You can specify either &DBSERVER, or &SSID for DB2 mainframe targets. High-speed Apply Engine determines this value from the SSID parameter  

DB2 LUW: High-speed Apply Engine uses the database parameter from configuration. |
| &JOBNAME       | the batch job name for the apply request                                                                                                                                                                   |
| &SSID          | the subsystem ID (SSID) of the target DB2 subsystem  
This variable is not valid for Oracle and DB2 LUW targets.                                                                                                                                               |
| &SYSUID        | the user ID associated with the apply request                                                                                                                                                             |
| &TIME          | the current time in the format \( hhmmss \), where  
- \( hh \) is the hour  
- \( mm \) is the minute  
- \( ss \) is the second  
For mainframe targets, when the &TIME variable as part of the file name model, you must include an alphabetic prefix (for example, T&TIME). |
before you execute High-speed Apply Engine. A typical GDG model name is shown in the following sample:

&SYSUID..BASENAME..D&DATE..T&TIME.MIGR(+1)

- partitioned data set (PDS) member names

High-speed Apply Engine does not support PDS data sets for multiple conflict files. A typical PDS model name is shown in the following sample:

&SYSUID..BASENAME..D&DATE..T&TIME.MIGR(WKLY01)

LIKE

For z/OS, the LIKE parameter lets you specify an existing data set as a model for the allocation attributes for the conflict files.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an existing data set that has the allocation attributes that you want to use for the conflict files</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

If you have already defined and created a data set to contain conflicts, you can use that data set name to define new conflict data sets. The data set name that you specify must refer to an existing data set on the system where the apply request runs.

MediaType

The MediaType parameter specifies the storage media for conflict information.
**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>None</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>No</td>
</tr>
<tr>
<td>Valid values</td>
<td>■ DB2 File</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW and Oracle File</td>
</tr>
<tr>
<td>Default value</td>
<td>None</td>
</tr>
</tbody>
</table>

**Usage**

The MediaType parameter allows you to specify the type of storage media that will be used for the deferred-conflict information. For Oracle and DB2 LUW the only valid value is File.

**MAXVOL**

For z/OS, the MAXVOL parameter specifies the maximum number of disk volumes that the High-speed Apply Engine can use to allocate conflict files.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
### Attribute and Value

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 255</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

*If you do not specify a value, High-speed Apply Engine does not include any maximum number of volumes as it allocates conflict files.*

### Usage

If your environment uses DFSMS, High-speed Apply Engine specifies the MaxVol parameter in its allocation request, but the operating system does not honor its value.

### MGMTCLAS

For z/OS, the MGMTCLAS parameter identifies a management class defined in DFSMS. The High-speed Apply Engine uses the management class when it allocates conflict files.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a management class that is defined in DFSMS for your environment</td>
</tr>
<tr>
<td>Default value</td>
<td>the value of the MGMTCLAS parameter specified in the [TemporaryFiles] section</td>
</tr>
</tbody>
</table>

### Usage

The management class defines how DFSMS manages the data sets in the class. This parameter applies only to systems that use DFSMS. If your environment does not use DFSMS, High-speed Apply Engine specifies the MGMTCLAS parameter in its allocation request, but the operating system does not honor its value.
**RELEASE**

For z/OS, the RELEASE parameter determines whether the High-speed Apply Engine requests the operating system to release any unused disk space when it closes conflict files.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Usage**

Use this parameter to manage any disk space that is allocated to conflict files, but not used during processing.

- Specify Yes to request that the operating system release any unused disk space at the end of processing.
- Specify No to avoid requesting that the operating system release any unused disk space.

**SingleFile**

The SingleFile parameter indicates whether the High-speed Apply Engine creates a single defer file or queue for all conflicts, or one for each apply agent.

**Attributes**

This parameter has the following attributes:
### ConflictFile parameters

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

### Usage

- Specify Yes to create a single conflict file for the entire apply request. Specify No to create a conflict file for each agent that processes the apply request.

- Specify No to direct High-speed Apply Engine to append the conflict file name to the agent ID, in the format .Inn. The I indicates that the identifier is an agent ID. The nn variable is a sequential number between 1 and the maximum number of agents.

For more information about specifying the model for deferred-conflict file or queue names, see “FileNameModel” on page 165 or “RELEASE” on page 172.

### SPACE

For z/OS, the SPACE parameter identifies the allocation unit to be used for conflict files.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>CYLINDER, CYL, TRACK, or TRK</td>
</tr>
<tr>
<td>Default value</td>
<td>Track</td>
</tr>
</tbody>
</table>
**Usage**

Specify an appropriate allocation unit for your environment, or accept the default value (track). The value that you specify determines the unit of measurement for the SpacePri and SpaceSec parameters.

**SpacePri**

For z/OS, the SpacePri parameter specifies the primary amount of space that is allocated for each conflict file. The unit of measurement for the SpacePri parameter is determined by the SPACE parameter.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer greater than or equal to 1</td>
</tr>
<tr>
<td>Default value</td>
<td>15</td>
</tr>
</tbody>
</table>

**Usage**

Specify a value that allocates enough primary storage space for the conflict files that the apply request might generate. The amount of storage that you need depends on the number of conflicts that will be generated, and whether you send all conflicts to a single file or have a file generated for each agent.

**SpaceSec**

For z/OS, the SpaceSec parameter specifies the secondary amount of space that is allocated for each conflict file. The unit of measurement for the SpaceSec parameter is determined by the SPACE parameter.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer greater than or equal to 0</td>
</tr>
<tr>
<td>Default value</td>
<td>5</td>
</tr>
</tbody>
</table>

Usage

Specify a value that allocates enough secondary storage space for the conflict files that the apply request might generate. The amount of storage that you need depends on the number of conflicts that will be generated, and whether you send all conflicts to a single file or have a file generated for each agent.

STORCLAS

For z/OS, the STORCLAS parameter specifies a storage class defined by DFSMS. The High-speed Apply Engine uses the storage class to allocate conflict files.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a storage class that you have defined for DFSMS</td>
</tr>
<tr>
<td>Default value</td>
<td>the value of the STORCLAS parameter specified in the [TemporaryFiles] section</td>
</tr>
</tbody>
</table>
Usage

This parameter only applies to environments that use DFSMS. If your environment does not use DFSMS, High-speed Apply Engine specifies the STORCLAS parameter in its allocation request, but the operating system does not honor its value.

UNIT

For z/OS, the UNIT parameter specifies a unit type that the High-speed Apply Engine uses to allocate conflict files for apply processing.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any valid storage unit identifier for your environment</td>
</tr>
<tr>
<td>Default value</td>
<td>the value of the UNIT parameter specified in the [TemporaryFiles] section</td>
</tr>
</tbody>
</table>

Usage

If you specify a value for the STORCLAS parameter, you do not need to specify a unit type. If your environment uses DFSMS, High-speed Apply Engine specifies the UNIT parameter in its allocation request, but the operating system does not honor its value.

VOLSER

For z/OS, the VOLSER parameter specifies the volume serial numbers that the High-speed Apply Engine uses to allocate conflict files for apply processing.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ConflictFile]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid volume serial number for a storage device</td>
</tr>
<tr>
<td>Default value</td>
<td>the value of the VOLSER parameter specified in the [TemporaryFiles] section</td>
</tr>
</tbody>
</table>

Usage

You can specify the VOLSER parameter as many times as necessary to define storage devices for the conflict files. If you specify a value for the STORCLAS parameter, you do not need to specify a volume serial number.

DDLConflict parameters

The [DDLConflict] parameters determine how the High-speed Apply Engine handles conflicts that occur for data definition language (DDL) statements in the input file. You can specify as many codes and actions as required for the types of DDL conflicts that might occur.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Code</td>
<td>“Code” on page 124</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [DDLConflict] parameters.

Figure 31: Example of [DDLConflict] parameter values

```
[DDLConflict]
Code=TimeOut
Action=Retry
```

If an SQL statement does not start with INSERT, UPDATE or DELETE, High-speed Apply Engine uses the rules defined the [DDLConflict] parameters. High-speed Apply Engine uses [DDLConflict] rules to process EXCHANGE statements.
DeleteConflict parameters

The [DeleteConflict] parameters describe how the High-speed Apply Engine handles conflicts that occur for DELETE statements in the apply request. You can specify as many codes and actions as required for the types of conflicts that might occur. The values that you specify in the [DeleteConflict] section override the values specified in the [AnyConflict] section (for DELETE statements).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Code</td>
<td>“Code” on page 124</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [DeleteConflict] parameters.

**Figure 32: Example of [DeleteConflict] parameter values**

```
[DeleteConflict]
  Code=Positive
  Action=DeferStatement
```

DiagnosticOutput parameter

The [DiagnosticOutput] parameter controls where the High-speed Apply Engine writes special diagnostic information used by BMC Software Customer Support in troubleshooting situations.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileName</td>
<td>“FileName” on page 179</td>
</tr>
</tbody>
</table>

High-speed Apply Engine writes output to this file.
- when it terminates with an error (for mainframe targets, a return code greater than or equal to 8)
- when a restart occurs and the value of the RestartType parameter is other than None

High-speed Apply Engine writes diagnostic information to a standard, default location, depending on the operating system. You can use the FileName parameter to override the default location for this information. The following figure shows an example of the [DiagnosticOutput] parameters.

**Figure 33: Example of [DiagnosticOutput] parameter values**

```
[DiagnosticOutput]
FileName=APPLY.DIAG
```

**FileName**

The FileName parameter in the [DiagnosticOutput] section specifies the location where the High-speed Apply Engine writes special diagnostic information used by BMC Software Customer Support in troubleshooting situations. Use this parameter to override the default location for the diagnostic output file.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[DiagnosticOutput]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
| Valid values  | ■ DB2  
                a fully qualified data set name or a DD name  
                ■ DB2 LUW or Oracle  
                a file name, including the path name if outside of the default path |
### Attribute

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default values</td>
<td>■ DB2</td>
</tr>
<tr>
<td></td>
<td>APTDIAG</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW, Oracle apediag</td>
</tr>
</tbody>
</table>

### Usage

High-speed Apply Engine writes output to this file only when a restart occurs and the value of the RestartType parameter is other than None.

You can specify any valid data set or DD name, or a UNIX or Windows file name for the FileName parameter. For more information about specifying file names on different operating systems, see “Usage” on page 156.

For mainframe targets, you can use a DD statement to allocate and catalog a new data set. The default DD name for the diagnostic output file is **APTDIAG**.

### DisplayOutput parameter

The [DisplayOutput] parameter controls how the High-speed Apply Engine displays statistics and messages that are generated during apply processing. High-speed Apply Engine always writes messages and statistics to a standard location for the operating system, or the file defined by the APTPRINT DD statement. You can use the FileName parameter to identify an additional, optional location for these messages.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileName (DFN)</td>
<td>“FileName (DFN)” on page 181</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [DisplayOutput] parameters.

**Figure 34: Example of [DisplayOutput] parameter values**

```
[DisplayOutput]     
FileName=APPLY.DISPLAY
```
FileName (DFN)

The FileName (DFN) parameter specifies an optional, supplemental location where the High-speed Apply Engine sends the messages that are generated during apply processing. On UNIX and Windows, this is a copy of stdout. For mainframe targets, this is a copy of APTPRINT.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[DisplayOutput]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>DFN</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>- DB2&lt;br&gt;a fully qualified data set name or a DD name&lt;br&gt;- DB2 LUW or Oracle&lt;br&gt;a file name, including the path name if outside of the default path</td>
</tr>
<tr>
<td>Default values</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

You can specify any valid data set or DD name, or a UNIX or Windows file name for the FileName parameter. The display output file also contains messages generated by the apply request. You specify the display output file as described in “DisplayOutput parameter” on page 180.

For more information about specifying file names on different operating systems, see “Usage” on page 156. You can use a DD statement to allocate and catalog a new data set.
DistributionTuning parameters

The [DistributionTuning] parameters control how the High-speed Apply Engine distributes work between multiple agents during apply processing.

For more information about the different techniques you can use to distribute work between agents, see “Distribution of work” on page 31.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxExecuteImmediateDML</td>
<td>“MaxExecuteImmediateDML” on page 183</td>
</tr>
<tr>
<td>PartitionClustering</td>
<td>“PartitionClustering” on page 184</td>
</tr>
<tr>
<td>RIClustering</td>
<td>“RIClustering” on page 186</td>
</tr>
<tr>
<td>RIRecursionLimit</td>
<td>“RIRecursionLimit” on page 187</td>
</tr>
<tr>
<td>SimpleTSClustering</td>
<td>“SimpleTSClustering” on page 189</td>
</tr>
<tr>
<td>URCheckRI</td>
<td>“URCheckRI” on page 189</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [DistributionTuning] parameters.

**Figure 35: Example of [DistributionTuning] parameter values**

```
[DistributionTuning]
  PartitionClustering=No
  RIClustering=Yes
  RIRecursionLimit=3
  SimpleTSClustering=Yes
```

**Default distribution of work**

If you do not change any default values, the High-speed Apply Engine distributes work between multiple agents according to the following criteria:

- By default, High-speed Apply Engine always uses distribution by object. You must explicitly request distribution by UR. For more information, see “DistributionType” on page 258.

- For tables that are part of RI relationships, High-speed Apply Engine uses the same agent to process updates to both parent and child tables (subject to parameters that you can adjust). For more information, see “RIRecursionLimit” on page 187.

- For partitioned tables, High-speed Apply Engine distributes work by partition.
For DB2, for tables that reside in a simple (nonsegmented) table space, High-speed Apply Engine uses the same agent to process all updates to all tables in the table space.

For tables that reside in segmented table spaces and are not subject to RI, High-speed Apply Engine distributes work by table. High-speed Apply Engine uses the same agent for all updates to each table.

### Precedence for distribution of work by object

The following list shows how the High-speed Apply Engine resolves conflicts between different types of distribution by object (when you define more than one type in your configuration).

A type of distribution higher on this list takes precedence over a type that is lower on the list:

- user-defined groups of tables (defined by [ObjectCluster] parameters)
- groups of tables defined by RI relationships (when RIClustering is Yes)
- user-defined partitions (defined by [ObjectPartition] parameters)
- partitions defined by partitioned table spaces (when PartitionClustering is No)
- for DB2 LUW, range partitions (when both range and hash partitioning are specified)
- for DB2, groups of tables defined by simple table spaces (when SimpleTSClustering is No)
- individual tables

### MaxExecuteImmediateDML

The MaxExecuteImmediateDML parameter defines a limit on how many data manipulation language (DML) statements the High-speed Apply Engine can process as EXECUTE IMMEDIATE.

### Attributes

This parameter has the following attributes:
### Usage

Normally, High-speed Apply Engine processes data definition language (DDL) statements by issuing an EXECUTE IMMEDIATE command. High-speed Apply Engine also processes a few types of complex DML statements this way (for example, an INSERT statement with an embedded SELECT clause). However, when High-speed Apply Engine experiences problems processing statements with static or dynamic SQL, it attempts to continue processing by using EXECUTE IMMEDIATE as a last resort.

Specify a number of statements to impose a limit. By default, High-speed Apply Engine does not impose a limit.

- Use this parameter to stop High-speed Apply Engine from executing an unusually large number of DML statements in a less than optimal manner. When High-speed Apply Engine exceeds the limit set by this parameter, it issues message BMCAPT0196 and terminates processing.

- Depending on the MaxExecuteImmediateDML value you specify, message BMCAPT0196 can indicate that High-speed Apply Engine is not using static or dynamic SQL processing as much as it should be.

- For DB2, the message can also indicate that you should adjust the values of the [BindTuning] parameters, particularly MaxPackages. For more information, see “Using [BindTuning] parameters for performance” on page 139 and “MaxPackages” on page 143.

### PartitionClustering

The PartitionClustering parameter determines how the High-speed Apply Engine distributes a request for partitioned target tables.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[DistributionTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

Usage

Use this parameter to enable or disable the distribution of work between multiple agents based on the physical partitions of a target table (as defined in the DBMS). Select one of the following values:

- Specify Yes to process all of the partitions in a target table with a single agent.
- Specify No to process each partition in a target table with a separate agent. The default value is No.

For more information, see “Distribution of work” on page 31. Be aware of the following points when you specify this parameter:

- High-speed Apply Engine ignores this parameter when it distributes work between multiple agents based on units of recovery (when the DistributionType parameter is UR).

- BMC Software recommends setting this parameter to Yes if any SQL statement in your input updates the partitioning key so that a row in the target table moves from one partition to another. If you set the parameter to No in this situation, you increase the possibility that High-speed Apply Engine can execute SQL statements in an incorrect order or that multiple agents can encounter deadlock conditions.

- If you specify No, exercise caution when the partitioning key of a target table is not the same as the primary key. High-speed Apply Engine can encounter synchronization errors between the agents that process different partitions. To avoid synchronization errors, define logical partitions (with [ObjectPartition] parameters) based on the table’s primary key.

- High-speed Apply Engine supports the standard range partitions available on DB2 for mainframe targets.
High-speed Apply Engine supports the standard range and hash partitions available on DB2 LUW targets on UNIX or Windows.

High-speed Apply Engine supports range partitions available on Oracle on UNIX or Windows. High-speed Apply Engine does not support hash, list, composite range-hash, or composite range-list partitioning.

**RIOClustering**

The RIOClustering parameter determines whether the High-speed Apply Engine distributes work between multiple agents based on the referential integrity (RI) settings of the target tables.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[DistributionTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Usage**

Use this parameter to group tables based on database-defined RI relationships, and to send the SQL statements affecting any table in an RI group to the same agent. This type of work distribution reduces the possibility of RI-related conflicts. To create the groups, High-speed Apply Engine follows the chain of parent tables in RI relationships, up to the number of generations specified by the RIRecursionLimit parameter (for more information, see “RIRecursionLimit” on page 187). As it encounters new tables, High-speed Apply Engine dynamically updates the groups of tables. Select one of the following values:

- Specify Yes if you want High-speed Apply Engine to group target tables based on database-defined RI relationships and distribute work for each group to a different agent. The default value is Yes.

- Specify No to permit multiple agents to process the target tables in an RI relationship. If you specify No, RI-related conflicts can occur because High-speed
Apply Engine does not always process the statements in the same order in which they were input. You might want to set up conflict rules to handle such errors by deferring them, then reprocessing the deferred SQL. If you specify No, High-speed Apply Engine ignores the value of the RIRecursionLimit parameter.

This parameter does not apply under distribution by unit of recovery (when the DistributionType parameter is UR). To control RI processing under distribution by unit of recovery, use the URCheckRI parameter. If you include the [ObjectCluster] section in your configuration, High-speed Apply Engine sets this parameter to No, regardless of the value specified in the configuration.

Because this parameter is Yes by default, the most common reason to specify it is to disable this type of distribution and prevent High-speed Apply Engine from grouping tables based on database-defined RI. You might want to disable RI distribution for the following reasons:

- You know there are RI relationships between the target tables in your apply request, but you know that either 1) there are no RI conflicts in your input data, or 2) you can safely ignore any RI conflicts. By disabling distribution based on RI, you can
  - enable multi-threaded execution for faster performance
  - avoid unnecessary processing as High-speed Apply Engine interrogates the target DBMS to determine RI relationships.

- Your apply request includes partitioned tables that are part of RI relationships, but none of the other RI-related tables are included in the apply request. In this situation, you disable RI distribution to let High-speed Apply Engine distribute work between agents based on partitions instead of RI relationships. For more information, see “Precedence for distribution of work by object” on page 183.

**RIRecursionLimit**

The RIRecursionLimit parameter determines how many generations of referential integrity (RI) relationships the High-speed Apply Engine examines as it creates groups of tables based on database-defined RI.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[DistributionTuning]</td>
</tr>
<tr>
<td>Attribute</td>
<td>Value</td>
</tr>
<tr>
<td>--------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 0 to 32767, or the keyword Max</td>
</tr>
<tr>
<td>Default value</td>
<td>Max</td>
</tr>
</tbody>
</table>

**Usage**

Use this parameter to control how much processing High-speed Apply Engine performs to determine the RI relationships of the target tables in the apply request. With this parameter, you can avoid unnecessary RI-related processing when your target database contains many generations of RI relationships, but your target tables are related by only a few generations of RI relationships. Select one of the following values:

- Specify 0 to disable RI-related processing. (Alternately, you can set the value of the RIClustering parameter to No).
- Specify a number between 1 and 32767 to define a number of generations in the chain of RI parent relationships.
- Specify Max to direct High-speed Apply Engine to determine all of the "parent" RI relationships beginning with the target tables of the apply request. For compatibility with earlier versions, the default value is Max.

Be aware of the following points regarding this parameter:

- If either the RIClustering parameter or the URCheckRI parameter are set to No, High-speed Apply Engine ignores the value of the RIRecursionLimit parameter.
- High-speed Apply Engine looks at only the parent tables in RI relationships. High-speed Apply Engine starts with the first target table it encounters. If the RIClustering parameter is Yes and this parameter is greater than zero, High-speed Apply Engine interrogates the target DBMS to find the immediate parent tables of the target table. Then it determines the parents of the parents and so forth, depending on the number of generations that you specify. When High-speed Apply Engine encounters a new target table, it follows the chain of RI parent relationships again, up to the number of generations you specify.

For example, if you know that your input includes only child tables and their immediate RI parents, specify 1 to prevent High-speed Apply Engine from acquiring RI information for any grandparent tables. Match the scope of RI-related processing to the scope of the RI relationships in your input data.
SimpleTSClustering

For DB2, the SimpleTSClustering parameter indicates whether the High-speed Apply Engine uses the same apply agent to process updates to all tables in a simple table space.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[DistributionTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Usage

Select one of the following values:

- Specify Yes to direct High-speed Apply Engine to cluster the tables of a simple table space together and send all statements for the tables to the same apply agent.

- Specify No to direct High-speed Apply Engine to distribute the statements by table. The consequence of specifying No is a greater risk of deadlocks between the agents.

High-speed Apply Engine ignores this parameter when it distributes work between multiple agents based on units of recovery (when the DistributionType parameter is UR).

URCheckRI

The URCheckRI parameter determines whether the High-speed Apply Engine accommodates dependencies between units of recovery (transactions) based on database-defined referential integrity (RI) relationships. High-speed Apply Engine uses this parameter only when it distributes work between multiple agents based on transactions (when the DistributionType parameter is UR).
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[DistributionTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Usage

When you select distribution by UR, High-speed Apply Engine can direct multiple agents to wait as needed when statements in two different transactions update tables that are part of a database-defined RI relationship. (High-speed Apply Engine directs the agent processing the later transaction to wait until the agent processing the earlier transaction finishes and performs a commit action.) For more information about distribution by UR, see “Distribution by unit of recovery” on page 34 or “DistributionType” on page 258. Select one of the following values:

- Specify Yes to accommodate transaction dependencies based on database-defined RI relationships. This value is the default.

- Specify No to ignore transaction dependencies based on database-defined RI relationships. High-speed Apply Engine does not examine the RI relationships of the target tables in your input data. Use this value to avoid RI-related processing when you know that 1) your input data does not contain any RI dependencies or that 2) you can safely ignore the RI dependencies in your input data.

Be aware of the following points regarding the URCheckRI parameter:

- This parameter does not apply under distribution by object (when the DistributionType parameter is Object). To control RI processing under distribution by object, use the RIClustering parameter.

- Set this parameter to Yes when your input contains a larger number of RI dependencies. If the input contains fewer RI dependencies, consider setting this parameter to No and using a conflict resolution rule with an Code parameter of RIConflict and an Action parameter of Retry.
The [ErrorOutput] parameter controls where the High-speed Apply Engine displays error messages that are generated during apply processing. The messages are always written to a standard location for the operating system, or the file defined by the APTERR DD statement. You can use the FileName parameter to identify an additional, optional location for these messages.

### FileNmae (EFN)

The FileName (EFN) parameter lets you specify an optional, supplemental location to which the High-speed Apply Engine sends error messages that the apply request generates. On UNIX and Windows, this is a copy of stderr. For mainframe targets, this is a copy of APTERR.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ErrorOutput]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>EFN</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
### Usage

You can specify any valid data set or DD name, or a UNIX or Windows file name for the **FileName** parameter. The error output file also contains messages and statistics that the apply request generates. You specify the error output file as described in “**ErrorOutput parameter**” on page 191.

For more information about specifying file names on different operating systems, see “**Usage**” on page 156. For mainframe targets, you can use a DD statement to allocate and catalog a new data set.

### InsertConflict parameters

The [InsertConflict] parameters describe how the High-speed Apply Engine handles conflicts that occur for INSERT statements in the apply request. You can specify as many codes and actions as required for the types of conflicts that might occur. The values that you specify in the [InsertConflict] section override the values specified in the [AnyConflict] section (for INSERT statements).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Code</td>
<td>“Code” on page 124</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [InsertConflict] parameters.

**Figure 37: Example of [InsertConflict] parameter values**

```plaintext
[InsertConflict]
  Code=NoRows
  Action=Abort
```

For more information about the parameters that you can specify in the conflict sections, see “**AnyConflict parameters**” on page 122. For a list of valid values for the
LogicalLog parameters

When you apply logical log input created by Log Master for DB2 (or an application program that conforms to the published format), the [LogicalLog] parameters control how the High-speed Apply Engine processes your input.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCSIDCompatible</td>
<td>“CCSIDCompatible” on page 194</td>
</tr>
<tr>
<td>IncludeDDL</td>
<td>“IncludeDDL” on page 195</td>
</tr>
<tr>
<td>IncludeLOB</td>
<td>“IncludeLOB” on page 195</td>
</tr>
<tr>
<td>IncludeRI</td>
<td>“IncludeRI” on page 197</td>
</tr>
<tr>
<td>IncludeTrigger</td>
<td>“IncludeTrigger” on page 198</td>
</tr>
<tr>
<td>IncludeXML</td>
<td>“IncludeXML” on page 199</td>
</tr>
<tr>
<td>IncludeHistory</td>
<td>“IncludeHistory” on page 200</td>
</tr>
<tr>
<td>Qualify</td>
<td>“Qualify” on page 201</td>
</tr>
<tr>
<td>PartitionDistribution</td>
<td>“PartitionDistribution” on page 202</td>
</tr>
<tr>
<td>Sort</td>
<td>“Sort” on page 203</td>
</tr>
<tr>
<td>SQLType</td>
<td>“SQLType” on page 204</td>
</tr>
<tr>
<td>UpdateColumns</td>
<td>“UpdateColumns” on page 205</td>
</tr>
<tr>
<td>UseLLogSysStrings</td>
<td>“UseLLogSysStrings” on page 206</td>
</tr>
<tr>
<td>WORKNUM</td>
<td>“WORKNUM” on page 207</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [LogicalLog] parameters. For more information about logical log file format, see the chapter about logical log files in the Log Master for DB2 Reference Manual.

**Figure 38: Example of [LogicalLog] parameter values**

```plaintext
[LogicalLog]
  IncludeRI=Yes
  IncludeTrigger=Yes
  Qualify=CHANGED
  Sort=YES
  SQLType=MIGRATE
  UpdateColumns=CHANGED
```
CCSIDCompatible

For DB2, the CCSIDCompatible parameter indicates whether the High-speed Apply Engine should bypass character conversion in a specific situation.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

Usage

Use this parameter to bypass character conversion when you know that the Coded Character Set Identifiers (CCSIDs) of two different sets of host variables are compatible, but High-speed Apply Engine encounters an error as it attempts to convert characters. Select one of the following values:

- Specify Yes to assume the CCSIDs are compatible and bypass character conversion.
- Specify No to assume the CCSIDs are not compatible and cause High-speed Apply Engine to report an error when it cannot find character conversion information in the SysStrings table of the DB2 catalog.

High-speed Apply Engine checks the value of this parameter only when all of the following conditions are true:

- The CCSID of the host variables for the source database is different than the CCSID of the host variables for the target database

- High-speed Apply Engine cannot locate either
  — Sysstrings information from the DB2 catalog
  — a string conversion utility for the two specified CCSIDs
IncludeDDL

The IncludeDDL parameter determines whether the High-speed Apply Engine processes or ignores any data definition language (DDL) information that is present in logical log input. This parameter is valid only for logical log input generated by Log Master for DB2.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Usage

High-speed Apply Engine uses EXECUTE IMMEDIATE to process all DDL statements. Select one of the following values:

- Specify Yes to direct High-speed Apply Engine to process any DDL information in the logical log input.
- Specify No to direct High-speed Apply Engine to ignore any DDL information in the logical log input.

For more information, see the chapter about Log Master for DB2 syntax in the Log Master for DB2 Reference Manual.

IncludeLOB

For DB2, the IncludeLOB parameter specifies whether the High-speed Apply Engine processes or ignores records in the logical log that apply to large object (LOB) columns. If you need to update LOB columns in the target tables, set this parameter to Yes.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes, No, Inline</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Usage

Specify Yes to include LOB column data in the changes to target tables. Specify No when the source tables do not contain LOB columns or when they contain LOB columns, but you do not want to attempt to update the LOB columns in the target tables. Specify Inline to include only inline LOB data in the changes to target tables, ignoring the auxiliary LOB tables. The default value of this parameter is Yes.

Be aware of the following points regarding High-speed Apply Engine and LOB processing:

- Valid logical log input files include LOB column data in separate LOB VSAM files (one set of LOB VSAM files for each LOB column, or each partition of each LOB column). Ensure that the appropriate LOB VSAM files are available to High-speed Apply Engine along with the logical log input file.

- If your input file contains LOB column data, consider specifying a value of Yes for the LargeStatementSupport parameter to accommodate the larger amount of data.

- A LOB table space that corresponds to a LOB column (or a partition of a LOB column) can be logged or not logged at the discretion of database administration personnel. If a LOB table space is not logged when the logical log input files are generated, High-speed Apply Engine inserts default values into the LOB column (or the partition of the LOB column) of the target tables.

- When the value of the SQLType parameter is Undo, High-speed Apply Engine cannot apply changes to undo the effect of a delete action on a row that contains a LOB column. The logical log records that exist for a delete action on a LOB table space associated with a LOB column do not contain LOB column data. Consequently, High-speed Apply Engine cannot insert the data into the LOB column to undo the delete action. In this situation High-speed Apply Engine
issues a warning message and either inserts default values for the LOB column or skips the LOB column and allows DB2 to insert default values.

- When the value of the SQLType parameter is Undo, High-speed Apply Engine might not be able to apply changes to undo the effect of an update action on a row that contains a LOB column. The logical log records that exist for an update to a LOB table space associated with a LOB column might not contain the "before image" of the LOB column data. If it cannot undo the effects of an update action, High-speed Apply Engine issues a warning message and either includes default values in the LOB column or skips the LOB column and allows DB2 to insert default values.

### IncludeRI

For DB2, the IncludeRI parameter specifies whether the High-speed Apply Engine uses the referential integrity (RI) records in the logical log when it processes the apply request. If you need to maintain referential integrity for the target tables, set this parameter to Yes.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>If the value of the SQLType parameter is Undo, the default value of IncludeRI is Yes. If the value of the SQLType parameter is Redo or Migrate, the default value of IncludeRI is No.</td>
</tr>
</tbody>
</table>

### Usage

Select one of the following values:
Specify Yes when the source tables are defined with RI and the target tables are not defined with RI.

To maintain RI, High-speed Apply Engine must sort the logical log file before it processes it. If the value of IncludeRI is Yes, High-speed Apply Engine sets the Sort parameter to Yes, overriding the value set in the configuration if necessary. For more information, see “Sort” on page 203.

Specify No when the source tables are defined with RI and the target tables are defined with identical RI relationships.

The default value of this parameter depends on the value of the SQLType parameter (for more information, see “SQLType” on page 204).

IncludeTrigger

For DB2, the IncludeTrigger parameter specifies whether the High-speed Apply Engine uses records in the logical log that result from trigger activity.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Usage

Use this parameter to apply (or avoid applying) input records that reflect database changes that result from activity defined within a trigger. The activity can be performed directly within the trigger or by other processes nested beneath the trigger (for example, in a stored procedure called from within a trigger).

This parameter is frequently used when a target database environment defines the same triggers and relationships as the source environment where a logical log input file was created. In this situation, set the IncludeTrigger parameter to No. This action causes High-speed Apply Engine to ignore any trigger records in the input file. As
High-speed Apply Engine processes the input, the triggers defined in the target environment are activated as they were in the source environment, and you can avoid any duplication of database changes.

Select one of the following values:

- Specify Yes to include (apply) trigger records. For consistency with past versions of High-speed Apply Engine, this value is the default.
- Specify No to exclude (ignore) trigger records.

To distinguish input records resulting from trigger activity, High-speed Apply Engine requires input logical log files to be generated by a Log Master.

**IncludeXML**

For DB2, the IncludeXML parameter specifies whether the High-speed Apply Engine processes or ignores records in the logical log that apply to XML columns. If you need to update XML columns in the target tables, set this parameter to Yes.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Usage**

Specify Yes to include data from XML columns in the changes to your target tables. Specify No when the source tables do not contain XML columns or when they contain XML columns, but you do not want to update the corresponding columns in the target tables.

Be aware of the following points regarding High-speed Apply Engine and XML processing:
Valid logical log input files include XML column data in separate VSAM files (one set of VSAM files for each XML column, or each partition of each XML column). Ensure that the appropriate VSAM files are available to High-speed Apply Engine along with the logical log input file.

If the DB2 subsystem of the target tables is different than the subsystem where your logical log input file was generated, ensure that an XMLSTRING control file — was generated along with your logical log input file — is available to High-speed Apply Engine with the logical log input file.

The logical log XMLSTRING control file contains the string IDs and string data that DB2 uses to encode the data in XML columns. The product or program that generates the logical log must obtain this data from the DB2 catalog table SYSIBM.SYSXMLSTRINGS. On a different DB2 subsystem, High-speed Apply Engine requires this data to serialize your XML data correctly.

If your input file contains XML column data, consider specifying a value of Yes for the LargeStatementSupport parameter to accommodate the larger amount of data.

XML columns (and the XML table spaces that correspond to them) can be logged or not logged at the discretion of database administration personnel. If an XML column is not logged when the logical log input files are generated, High-speed Apply Engine inserts default values into that XML column in the target table.

**IncludeHistory**

For DB2, the IncludeHistory parameter specifies whether the High-speed Apply Engine uses history records in the logical log when it processes the apply request. If you need to maintain versioning updates for the target tables, set this parameter to Yes.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Attribute</td>
<td>Value</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

**Usage**

Select one of the following values:

- Specify Yes to process the logical log records that are associated with updates to the history table that is associated with a system-maintained temporal table.
- Specify No to exclude (ignore) history records.

**Qualify**

For DB2, the Qualify parameter specifies how the High-speed Apply Engine selects the rows in the target table to be updated. High-speed Apply Engine uses the value that you specify to generate a WHERE clause as it applies SQL statements to the target table.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Key, Changed, or All</td>
</tr>
<tr>
<td>Default value</td>
<td>Key</td>
</tr>
</tbody>
</table>

**Usage**

Specify one of the following values for the Qualify parameter:

- Key indicates that only those columns designated as key values are included in the WHERE clause. Key is the default value.
- Changed indicates that all key columns and any column with changes are included in the WHERE clause of the SQL statement.

- All indicates that all columns referenced in the input file are included in the WHERE clause.

For more information about what columns can be included in the generated WHERE clauses, see the chapter about Log Master for DB2 syntax in the *Log Master for DB2 Reference Manual*.

## PartitionDistribution

When the High-speed Apply Engine distributes work between agents based on the physical partitions of a table, it must determine the partition that is affected by each record in the input. The PartitionDistribution parameter specifies how High-speed Apply Engine determines the partition or group of partitions that are affected by an input record. This parameter is valid only for logical log input generated by Log Master for DB2.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>UseHeader or Calculate</td>
</tr>
<tr>
<td>Default value</td>
<td>If the value of the SQLType parameter is Migrate, the default value of PartitionDistribution is Calculate</td>
</tr>
<tr>
<td></td>
<td>If the value of the SQLType parameter is Redo or Undo, the default value of PartitionDistribution is UseHeader</td>
</tr>
</tbody>
</table>

### Usage

High-speed Apply Engine can determine the affected partition in the following ways:

- Specify UseHeader to direct High-speed Apply Engine to determine the partition by using the partition number provided in the header of each logical log record in...
the input. This value can improve performance of the initial run and any restarted runs of an apply request.

- Specify Calculate to direct High-speed Apply Engine to determine the partition by reading the values of partitioning key columns in the input record, building a key value, and comparing that value to the high limit key.

Consider the following points as you use the PartitionDistribution parameter:

- High-speed Apply Engine ignores this parameter when any of the following conditions is true:
  - the value of the PartitionClustering parameter is Yes
  - the value of the DistributionType parameter is UR

- When you specify logical partitions for a table with [ObjectPartition] parameters, High-speed Apply Engine determines the partition by reading the values of partitioning key columns (Calculate) for all input statements that affect that table.

- The value of this parameter applies to the entire logical log input file.

- Do not set this parameter to UseHeader if your input contains data definition language (DDL) objects that change the partitioning limit keys of the target table.

- If the partitioning limit keys of the target table are different than those of the source table from which the input was generated, but you execute the input because the differences are compatible, the partition distribution statistics in messages such as BMCAPT0048 reflect the distribution of work based on the partitions of the target table.

**Sort**

For DB2, the Sort parameter specifies whether to sort the data in a logical log file before processing the apply request.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
</tbody>
</table>
**LogicalLog parameters**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>No</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Usage**

BMC Software recommends that you accept the default value for this parameter. The default value is Yes. Specifying IncludeRI=Yes or SQLType=Undo overrides the value of the Sort parameter if necessary, so that the logical log is always sorted. If you need to change the value of the Sort parameter to No, contact BMC Software Customer Support.

**SQLType**

For DB2, the SQLType parameter determines which records in your logical log input the High-speed Apply Engine uses to generate and apply SQL statements.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Redo, Undo, or Migrate</td>
</tr>
<tr>
<td>Default value</td>
<td>Migrate</td>
</tr>
</tbody>
</table>

**Usage**

Logical log input that conforms to the published format includes a field named SQLTYPE in the header of each record in the logical log data file. The SQLType parameter causes High-speed Apply Engine to select a different set of records based on the value in this field. High-speed Apply Engine generates SQL statements from the records it selects. For more information about logical log data records, see the chapter that discusses logical log files in the *Log Master for DB2 Reference Manual*. 
Specify one of the following values for the SQLType parameter:

- Specify Redo to cause High-speed Apply Engine to generate SQL statements from redo records in the logical log input (where the SQLTYPE field is 'R'). A logical log input file does not contain any redo records unless a REDO recovery point was specified when the logical log was generated.

- Specify Undo to cause High-speed Apply Engine to generate SQL statements from undo records in the logical log input (where the SQLTYPE field is 'U'). If this parameter is UNDO, High-speed Apply Engine
  — reverses the statement type of the original data changes (for example, if the original statement was an INSERT, High-speed Apply Engine generates and executes a DELETE statement)
  — reverses the order of the original data changes. (To accomplish this, High-speed Apply Engine must sort the logical log input. When SQLType is UNDO, High-speed Apply Engine sets the value of the Sort parameter to Yes, regardless of the value in your configuration. For more information, see “Sort” on page 203.)

- Specify Migrate to cause High-speed Apply Engine to generate SQL statements from undo records in the logical log input (where the SQLTYPE field is 'U'). If this parameter is Migrate, High-speed Apply Engine does not reverse the statement type or the order of the original data changes.

For more information about these values, see the chapter about Log Master for DB2 syntax in the Log Master for DB2 Reference Manual.

The value of this parameter affects the default value of the IncludeRI parameter (for more information, see “IncludeRI” on page 197).

**UpdateColumns**

For DB2, the UpdateColumns parameter specifies whether the apply request updates all of the columns in the target table, or only the changed columns.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
</tbody>
</table>
### LogicalLog parameters

**Attribute**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>All or Changed</td>
</tr>
<tr>
<td>Default value</td>
<td>Changed</td>
</tr>
</tbody>
</table>

#### Usage

Specify Changed to improve the performance of apply requests that update tables with large numbers of columns.

### UseLLogSysStrings

A logical log file generated by the BMC Software product Log Master for DB2 contains SysStrings records that define the translation information, including the coded character set identifiers (CCSIDs), that was in use when the file was generated. The UseLLogSysStrings parameter determines whether the High-speed Apply Engine uses the translation information from the logical log to translate input data before it applies the data to the target DB2 subsystem.

#### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

#### Usage

BMC Software recommends that you use the default value of No for normal processing.

- Specify Yes to direct High-speed Apply Engine to *translate* the input data before applying it to the target subsystem. Use this value when DB2 would not be able to successfully translate between the source column and target column encoding.
schemes by itself. This situation can occur when a translation scheme is either undefined or different in the SysStrings information on the target subsystem. For example, you can specify Yes when the SYSIBM.SYSSTRINGS table on the source DB2 subsystem contains a user-defined translation scheme that is not available on the target subsystem. High-speed Apply Engine performs the translation using information from the logical log file. In this example, DB2 would still need to be able to successfully translate from the source application encoding scheme to the target column encoding scheme.

- Specify No to direct High-speed Apply Engine to apply the input data directly to the target subsystem as is (*without* translation). Use this value for normal processing when the conditions described for the Yes value do not exist. Specifying No can improve performance in some cases by avoiding unnecessary translation actions. This value is the default.

**WORKNUM**

For z/OS, the WORKNUM parameter specifies the number of sort work data sets that the High-speed Apply Engine uses to process your apply request.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogicalLog]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 0 to 255</td>
</tr>
<tr>
<td>Default value</td>
<td>0</td>
</tr>
</tbody>
</table>

**Usage**

Specify either of the following values:

- a number of sort work data sets from 1 to 255
- a value of 0
  This value indicates that the sort routine sets the number of sort work data sets dynamically, depending on conditions in your environment. This value is the default.

For more information about sort work data sets, see “Sort work data sets” on page 27.

LogOutput parameters

The [LogOutput] parameters control the types of information that the High-speed Apply Engine records in the log file and the location of the log file. BMC Software Customer Support can use the information in the log file for diagnostic and troubleshooting purposes.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>FileName (LFN)</td>
<td>“FileName (LFN)” on page 208</td>
</tr>
<tr>
<td>LogType</td>
<td>“LogType” on page 209</td>
</tr>
<tr>
<td>Overlap</td>
<td>“Overlap” on page 210</td>
</tr>
<tr>
<td>Threshold</td>
<td>“Threshold” on page 211</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [LogOutput] parameters.

Note

BMC Software recommends consulting BMC Software Customer Support before changing the default values of these parameters.

Figure 39: Example of [LogOutput] parameter values

```
[LogOutput]
FileName=APTLOG
Overlap=Yes
Threshold=0
LogType=CONFLICT
LogType=WORK_ITEM
LogType=DISPLAY
```

FileName (LFN)

The FileName (LFN) parameter specifies the file name, data set name, or DD name of the file where the High-speed Apply Engine writes the log entries that the apply request generates.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogOutput]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>LFN</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
| Valid values | ■ DB2 for mainframe targets  
a fully qualified data set name or a DD name  
■ DB2 LUW or Oracle  
a file name, including the path name if outside of the default path |
| Default values | ■ Mainframe operating systems  
APTLOG  
■ UNIX and Windows  
no default value |

Usage

You can specify any valid data set or DD name, or a UNIX or Windows file name for the FileName parameter.

For more information about specifying file names on different operating systems, see “Usage” on page 156. For mainframe targets, you can use a DD statement to allocate and catalog a new data set. The default DD name is APTLOG.

LogType

The LogType parameter indicates the type of information that is captured for an apply request.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogOutput]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Conflict, Display, Exec_immed, Host_vars, Work_item</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

You specify the LogType parameter for each type of log entry that you want to capture. Consider the number of types that you specify for this parameter when you specify the Threshold and Overlap parameters.

Specify one of the following values for the LogType parameter. You can repeat the LogType parameter for as many values as necessary.

- **Conflict** indicates that High-speed Apply Engine logs the conflicts that are encountered during apply processing.

- **Display** indicates that High-speed Apply Engine logs the messages that are written to the location specified by the [DisplayOutput] parameters.

- **Exec_immed** indicates that High-speed Apply Engine logs those SQL statements that will be applied to the target table without validation or preparation.

- **Host_vars** indicates that High-speed Apply Engine logs the host variables associated with the target DB2 subsystem or Oracle instance.

- **Work_item** indicates that High-speed Apply Engine logs the work queue items generated for the apply request.

Overlap

The Overlap parameter indicates whether the log file can be overwritten when it exceeds the value specified for the Threshold parameter. If you do not allow the log file to overlap, log processing ends when the file exceeds the threshold.
### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogOutput]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Usage

- Specify Yes to ensure that log processing continues until the end of the apply request, and that the most recent log entries are available.

- Specify No to ensure that log processing ends when the file exceeds the threshold. Be sure that you specify an adequate amount of space for the log file by using the Threshold parameter. Otherwise, log processing stops when the log file exceeds the threshold value for the log.

### Threshold

The Threshold parameter specifies the amount of space (in bytes) allocated for the log file. If you set the Overlap parameter to Yes, the log entries are overwritten from the top of the file when the log size reaches the threshold value.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[LogOutput]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>number of bytes, greater than or equal to 0 (zero)</td>
</tr>
</tbody>
</table>
### Usage

Specify the number of bytes, greater than or equal to zero, that can be allocated for the log file. If you specify zero, the size of the log file is unlimited.

When you specify the Threshold parameter, consider the resources available on the system where the apply request runs. If you anticipate a large number of log entries for the apply request, consider setting a reasonable size for the log file (1,000,000 bytes, for example). Then, use the Overlap parameter to guarantee that log processing continues until the apply request ends. This practice allows you to log the problems that lead to a failure, but conserves the system resources during apply processing.

### MonitorTuning parameters

The [MonitorTuning] parameters help you control how the High-speed Apply Engine displays progress messages during an apply request. The progress messages display statistics about the SQL statements that High-speed Apply Engine has processed as input and committed against the target tables.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommitTrigger</td>
<td>“CommitTrigger” on page 213</td>
</tr>
<tr>
<td>CommitValue</td>
<td>“CommitValue” on page 214</td>
</tr>
<tr>
<td>InputTrigger</td>
<td>“InputTrigger” on page 215</td>
</tr>
<tr>
<td>InputValue</td>
<td>“InputValue” on page 216</td>
</tr>
<tr>
<td>ScanInterval</td>
<td>“ScanInterval” on page 217</td>
</tr>
</tbody>
</table>

High-speed Apply Engine displays progress messages *during* an apply request, so that you can determine

- how many SQL statements in the input file High-speed Apply Engine has examined and dispatched to an agent for application to the target tables
- how many SQL statements High-speed Apply Engine has committed against the target tables

Use the following guidelines to adjust progress messages with these parameters:
Set the parameters to display fewer progress messages for existing apply requests that you run frequently in production.

Set the parameters to display more frequent progress messages as you tune the performance of a new apply request, or to diagnose an existing apply request that experiences performance or other problems.

The following figures show examples of [MonitorTuning] parameters.

**Figure 40: Example of production [MonitorTuning] parameters**

```
[MonitorTuning]
ScanInterval=1
InputTrigger=Time
InputValue=300
CommitTrigger=Count
CommitValue=10000
```

**Figure 41: Example of diagnostic [MonitorTuning] parameters**

```
[MonitorTuning]
ScanInterval=1
InputTrigger=Time
InputValue=10
CommitTrigger=Count
CommitValue=5000
```

**CommitTrigger**

The CommitTrigger parameter indicates the type of units used in the CommitValue parameter. You can specify whether the number defined by the CommitValue parameter represents a number of seconds or a number of SQL statements committed by the High-speed Apply Engine.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MonitorTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Time, Count, None</td>
</tr>
<tr>
<td>Default value</td>
<td>Count</td>
</tr>
</tbody>
</table>
Usage

Use this parameter along with the CommitValue parameter to define how often High-speed Apply Engine displays progress messages for committed SQL statements. You can also use this parameter to prevent High-speed Apply Engine from displaying these messages. Specify one of the following values.

- Time indicates that the number defined by the CommitValue parameter represents a number of seconds.
- Count indicates that the number defined by the CommitValue parameter represents a number of SQL statements committed against the target tables.
- None prevents High-speed Apply Engine from displaying progress messages for committed SQL statements.

If the value of this parameter is Time or Count, you must specify the CommitValue parameter. For information about the CommitValue parameter, see “CommitValue” on page 214.

CommitValue

The CommitValue parameter specifies a numeric value that determines how often the High-speed Apply Engine displays progress messages for committed SQL statements. The units for this parameter are based on the value that you specify for the CommitTrigger parameter.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MonitorTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>

MonitorTuning parameters
### Usage

Use this parameter along with the CommitTrigger parameter to define how often High-speed Apply Engine displays progress messages for committed SQL statements.

Specify a numeric value for the number of seconds or the number of committed SQL statements.

If the value of the CommitTrigger parameter is Time or Count, you must specify this parameter. For more information about the CommitTrigger parameter, see “CommitTrigger” on page 213.

### InputTrigger

The InputTrigger parameter indicates the type of units used in the InputValue parameter. You can specify whether the number defined by the InputValue parameter represents a number of seconds or a number of input SQL statements that the High-speed Apply Engine has processed.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MonitorTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Time, Count, None</td>
</tr>
<tr>
<td>Attribute</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------</td>
</tr>
<tr>
<td>Default value</td>
<td>Time</td>
</tr>
</tbody>
</table>

**Usage**

Use this parameter along with the InputValue parameter to define how often High-speed Apply Engine displays progress messages for SQL statements that High-speed Apply Engine has processed as input. You can also use this parameter to prevent High-speed Apply Engine from displaying these messages. Specify one of the following values.

- Time indicates that the number defined by the InputValue parameter represents a number of seconds.
- Count indicates that the number defined by the InputValue parameter represents a number of SQL statements in the input file.
- None prevents High-speed Apply Engine from displaying progress messages for input SQL statements.

If the value of this parameter is Time or Count, you must specify the InputValue parameter. For more information about the InputValue parameter, see “InputValue” on page 216.

**InputValue**

The InputValue parameter specifies a numeric value that determines how often the High-speed Apply Engine displays progress messages for SQL statements that it has processed as input. The units for this parameter are based on the value that you specify for the InputTrigger parameter.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MonitorTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Attribute</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 32767 that represents</td>
</tr>
<tr>
<td></td>
<td>■ a number of seconds</td>
</tr>
<tr>
<td></td>
<td>■ a number of input SQL statements</td>
</tr>
<tr>
<td></td>
<td>depending on the specified value of the InputTrigger parameter</td>
</tr>
<tr>
<td>Default value</td>
<td>10</td>
</tr>
</tbody>
</table>

**Usage**

Use this parameter along with the InputTrigger parameter to define how often High-speed Apply Engine displays progress messages for input SQL statements.

Specify a numeric value for the number of seconds or the number of input SQL statements.

If the value of the InputTrigger parameter is Time or Count, you must specify this parameter. For more information about the InputTrigger parameter, see “InputTrigger” on page 215.

**ScanInterval**

The ScanInterval parameter determines how frequently the High-speed Apply Engine scans its own internal processes to determine if any of the triggers defined by other [MonitorTuning] parameters have been activated.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MonitorTuning]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer from 1 to 32767</td>
</tr>
<tr>
<td>Default value</td>
<td>1</td>
</tr>
</tbody>
</table>
**Usage**

Use this parameter to determine the precision with which High-speed Apply Engine monitors the triggers set by other [MonitorTuning] parameters. Normally, the scan interval should be short enough to ensure that High-speed Apply Engine will not activate one of the progress message triggers more than once during the scan interval.

For example, assume that for diagnostic reasons, you set the CommitTrigger and CommitValue parameters to display progress messages after every 10 committed SQL statements. If you set the ScanInterval parameter to 30 seconds, High-speed Apply Engine might commit 100 statements before High-speed Apply Engine checks to determine if your trigger has been activated. Your progress messages would not have the scope you desire.

**Note**

BMC Software recommends that you do not change the default value of this option unless you have a specific reason to do so. You can consult BMC Software Customer Support before changing this option.

---

**MultiRowInsert parameters**

For DB2, the [MultiRowInsert] parameters determine whether the High-speed Apply Engine inserts more than one row into a target table in one insert action (by using host variable arrays and the FOR n ROWS form of the INSERT statement). Additional parameters enable you to control the number of rows in each insert action and to set the level of insert conflicts that High-speed Apply Engine tolerates before it switches to single row insert actions.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxConflicts</td>
<td>“MaxConflicts” on page 219</td>
</tr>
<tr>
<td>MaxRows</td>
<td>“MaxRows” on page 220</td>
</tr>
<tr>
<td>MultiRowInsert</td>
<td>“MultiRowInsert” on page 221</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [MultiRowInsert] section.

**Figure 42: Example of [MultiRowInsert] parameter values**

```
[MultiRowInsert]
MultiRowInsert=Yes
MaxRows=500
MaxConflicts=5
```

The [MultiRowInsert] parameters are optional. If you do not specify any of the parameters, High-speed Apply Engine does not perform multiple row insert actions.
MaxConflicts

For DB2, the MaxConflicts parameter enables you to control how many insert-related conflicts High-speed Apply Engine tolerates during all multiple row insert actions of an apply request.

If the number of insert-related conflicts exceeds the value that you specify, High-speed Apply Engine stops using multiple row insert actions and switches to single row insert actions. When the insertion of any row within a multiple row insert action violates a constraint or causes a conflict, High-speed Apply Engine performs the actions defined by the conflict rule in the [InsertConflict] section that specifies a Code value of MultiRowConflict. The rule can be defined explicitly or by default values. For more information, see “Codes for conflict resolution (SQL codes)” on page 79.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MultiRowInsert]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any integer value greater than or equal to 0</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

Specify any integer value greater than or equal to zero. A value of zero indicates that High-speed Apply Engine does not tolerate any conflicts and switches to single row insert actions after it encounters the first conflict.

Be aware of the following points regarding this parameter:

- High-speed Apply Engine only supports multiple row insert actions for mainframe targets.

- As it processes multiple row insert actions, High-speed Apply Engine uses the ATOMIC clause of the multiple row INSERT statement. If an insert action for any row fails, all changes made to the target table by any of the insert actions are backed out (including changes made by successful insert actions).
To simply conflict processing, when High-speed Apply Engine performs a multiple row insert action and encounters an SQL code other than 0 or -911, it resets the SQL code to -254 so that it can process the conflict rule defined by the Code value of MultiRowConflict in the [InsertConflict] section.

MaxRows

For DB2, the MaxRows parameter enables you to control how many rows High-speed Apply Engine includes in each multiple row insert action of an apply request. Use this parameter to balance the improved performance of multiple row insert actions against the increased overhead that can occur when High-speed Apply Engine must process conflicts and possibly retry the insertion of large numbers of rows.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MultiRowInsert]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer value between 1 and 32,767</td>
</tr>
<tr>
<td>Default value</td>
<td>100</td>
</tr>
</tbody>
</table>

Usage

Specify any integer value greater between 1 and 32,767. The default value is 100 rows in each multiple row insert action.

Be aware of the following points regarding this parameter:

- The performance benefits of multiple row insert actions depend on conditions in your environment. BMC Software recommends that you measure results for several values of this parameter in a test environment before implementing changes in a production environment.

- When the insertion of any row during a multiple row insert action violates a constraint or causes a conflict, High-speed Apply Engine performs the actions defined by the conflict rule in the [InsertConflict] section that specifies a Code
value of MultiRowConflict. For more information, see “Codes for conflict resolution (SQL codes)” on page 79.

**MultiRowInsert**

For DB2, the MultiRowInsert parameter enables or disables High-speed Apply Engine’s processing of multiple row insert actions.

When the value of this parameter is Yes, High-speed Apply Engine can insert more than one row into a target table in one insert action (by using host variable arrays and the FOR n ROWS form of the INSERT statement).

Multiple row insert actions can provide performance benefits, but they can also increase the overhead of conflict resolution processing when insert-related conflicts occur. Use the MaxRows and MaxConflicts parameters to balance increased performance against increased conflict resolution overhead.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[MultiRowInsert]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

**Usage**

Specify Yes to enable multiple row insert actions. The default value of No causes High-speed Apply Engine to perform single row insert actions for all inserts in the input file.

Be aware of the following points regarding this parameter:

- High-speed Apply Engine supports multiple row insert actions only under DB2 Version 9 and later for mainframe targets.

- The performance benefits of multiple row insert actions depend on conditions in your environment. BMC Software recommends that you measure results for
several values of the MaxRows and MaxConflicts parameters in a test environment before implementing changes in a production environment.

ObjectCluster parameters

[ObjectCluster] parameters let you specify a group of target tables to be processed by the same apply agent. You can use the parameters in this section to define groups of related tables or to optimize workload distribution.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster</td>
<td>“Cluster” on page 223</td>
</tr>
<tr>
<td>Table</td>
<td>“Table” on page 224</td>
</tr>
</tbody>
</table>

When you define a group of target tables, the High-speed Apply Engine uses the same agent to process all updates to that group. However High-speed Apply Engine does not guarantee that the agent processes *only* that group. If the number of groups you define is greater than the number of agents you permit, High-speed Apply Engine can use one agent to process more than one group. For more information, see “MaxAgents” on page 119.

The position of the [ObjectCluster] parameters is significant. The Cluster parameter must appear first in the section, and it must be followed by at least one table name. For an example of [ObjectCluster] parameters, see the following figure.

**WARNING**

If you include the [ObjectCluster] section in the configuration, the RIClustering parameter is set to No when you run the apply request, regardless of the value specified in the configuration. If you define an object cluster, High-speed Apply Engine ignores any referential integrity constraints on the members of the cluster.

**Figure 43: Example of [ObjectCluster] parameter values**

```plaintext
[ObjectCluster]
Cluster=HUMANRES
 Table=HR.*
Cluster=ACC1Q
 Table=ACC.JAN
 Table=ACC.FEB
 Table=ACC.MAR
Cluster=ACC2Q
 Table=ACC.APR
 Table=ACC.MAY
 Table=ACC.JUN
```
Cluster

The Cluster parameter lets you specify a name for the group of target tables that you define with the Table parameters. You can specify this parameter as many times as necessary to define the clusters that you want to use for the apply request. The Cluster parameter must be the first parameter in the [ObjectCluster] section.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectCluster]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a character string</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

Specify any character string to identify the group of target tables in the cluster, or leave the value for the parameter blank. If you do not specify a value, the High-speed Apply Engine processes the cluster correctly, but the cluster is not identified in the output messages for the request.

You can specify as many clusters as you need to distribute the workload for the apply request. You define a table as a member of a cluster by using the Table parameter, as described in “Table” on page 224. A cluster includes all of the Table parameters that follow it, until another Cluster parameter is specified. Each Cluster parameter indicates the start of a new object cluster.

High-speed Apply Engine groups all of the tables defined for a cluster to be processed by a single agent. When you define clusters, consider the amount of work required for the target tables. If you group large tables or tables that require large numbers of updates, you can negatively impact the performance of High-speed Apply Engine.
Table

The Table parameter specifies the name of a target table that is a member of an object cluster. This parameter must appear after a Cluster parameter in the [ObjectCluster] configuration section.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectCluster]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid target table name</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

Specify the name of a target table for the apply request. You can repeat this parameter as many times as necessary to define the tables for an object cluster. The format of the table name is described in “Object names in configuration files” on page 99. Wildcards are supported for this table name. For more information, see “Wildcards in object names” on page 101.

ObjectMap parameters

The [ObjectMap] parameters let you specify target table and column names that differ from the source table and column names.

For each target table that has a different definition than your source table, you must specify a set of [ObjectMap] parameters. If you do not include an [ObjectMap] section, the High-speed Apply Engine uses the source object names as the target names.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>SourceColumn (SC)</td>
<td>“SourceColumn” on page 227</td>
</tr>
<tr>
<td>SourceTable (ST)</td>
<td>“SourceTable” on page 228</td>
</tr>
</tbody>
</table>
To specify object mapping parameters for DB2 objects with long names, you might need to use multiple lines in your configuration file. For more information, see “Overview of the configuration file’s format” on page 93.

**Note**

Be aware of the following considerations when specifying [ObjectMap] parameters:

- Do not specify [ObjectMap] parameters if your input contains data definition language (DDL) statements in SQL input (or DDL objects in logical log input). Due to processing constraints, High-speed Apply Engine does not support the combination of object mapping and DDL input.

- A logical log file generated with overtime mode might contain multiple instances of the same object, (for example, a table that was dropped and then recreated). Therefore, you should not specify [ObjectMap] parameters for a table with multiple instances.

**Figure 44 on page 225** shows an example of the [ObjectMap] parameters. **Figure 45 on page 225** shows how to change the name of a column without changing the name of the table that contains the column.

**Positional requirements for object mapping**

*The position of the [ObjectMap] parameters is significant.* If the parameters are out of order, the High-speed Apply Engine issues messages that describe the problem, and apply processing terminates.
You must specify the parameters in the following order. For an example of the parameter order, see Figure 44 on page 225.

1. SourceTable must be the first parameter in the section.
2. TargetTable must follow its corresponding SourceTable parameter.
3. SourceColumn must follow the SourceTable/TargetTable pair that contains its source table.
4. TargetColumn must follow its corresponding SourceColumn parameter.

You can repeat the SourceTable and TargetTable parameters as often as necessary to complete the table-name mapping for the apply request. Within a SourceTable/TargetTable pair, you can repeat the SourceColumn and TargetColumn parameters as many times as necessary to complete the column-name mapping.

### Using wildcards to map object names

You can use wildcard patterns as part of the table and column names for the [ObjectMap] parameters.

The wildcards that the High-speed Apply Engine supports are described in “Wildcards in object names” on page 101. For object-name mapping, the High-speed Apply Engine attempts to match the source object names to the target object names. If no match exists, High-speed Apply Engine processes the request by using the source object name as the target. The following table shows how you might use wildcards to perform specific object-mapping tasks.

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SourceTable=TEST.*</code></td>
<td>This example maps all of the tables owned by the TEST user ID to the equivalent tables owned by the PROD user ID.</td>
</tr>
<tr>
<td><code>TargetTable=PROD.*</code></td>
<td></td>
</tr>
<tr>
<td><code>SourceTable=AP1.DS*</code></td>
<td>This example represents data migration between tables owned by the application AP1. Tables that start with DS will be mapped to tables that start with AT.</td>
</tr>
<tr>
<td><code>TargetTable=AP1.AT*</code></td>
<td></td>
</tr>
</tbody>
</table>
This example shows how data from one or more source tables can be applied to a single target table. If the input file included updates from tables AP2.WK01, AP2.WK02, and AP2.WK22, those updates would be applied to the AP3.WEEK table. Because of the single-character wildcards, updates from table AP2.WK101 would not be applied to the AP3.WEEK table.

Wildcards specified for the source object must match the type and format of the wildcards specified for the target object. The following table describes examples of valid and invalid mappings.

Table 16: Valid and invalid wildcard pattern matches

<table>
<thead>
<tr>
<th>Example</th>
<th>Valid/invalid</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SourceTable=ACCT.WE??</td>
<td>Valid</td>
<td>The wildcards are the same type (single character) and format.</td>
</tr>
<tr>
<td>TargetTable=ACCT.AT??</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SourceTable=*.SALARY</td>
<td>Invalid</td>
<td>The wildcards are not the same type. The pattern for the source</td>
</tr>
<tr>
<td>TargetTable=??.RATE</td>
<td></td>
<td>table is a string, but the pattern for the target table is three</td>
</tr>
<tr>
<td></td>
<td></td>
<td>single characters.</td>
</tr>
<tr>
<td>SourceTable=??.SALARY</td>
<td>Invalid</td>
<td>The wildcards are not the same format. The different lengths render</td>
</tr>
<tr>
<td>TargetTable=??.RATE</td>
<td></td>
<td>this mapping invalid.</td>
</tr>
<tr>
<td>SourceTable=*.R??</td>
<td>Valid</td>
<td>The wildcards are the same type and format. Note that the constants</td>
</tr>
<tr>
<td>TargetTable=*.JED??</td>
<td></td>
<td>do not need to be the same length for the mapping to be valid.</td>
</tr>
</tbody>
</table>

**SourceColumn**

The SourceColumn (SC) parameter lets you specify a column name in the source table that differs from the column name in the target table. Specifying the SourceColumn parameter with a corresponding TargetColumn parameter defines how the High-speed Apply Engine maps column names. You can specify this parameter as many times as necessary to map the columns for the table named in the SourceTable parameter.

**Attributes**

This parameter has the following attributes:
### Usage

If you specify SourceColumn, the corresponding TargetColumn parameter must immediately follow it. You can include wildcard characters as part of the column name. For more information about using wildcard characters in object-name mapping, see “Using wildcards to map object names” on page 226.

This parameter is not valid if your input contains data definition language (DDL) statements in SQL input (or DDL objects in logical log input).

### SourceTable

The SourceTable (ST) parameter specifies the name of the source table for the apply request. You must specify SourceTable only if object-name mapping is required between the source and target tables.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectMap]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>ST</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>the table name of the source table</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>
**Usage**

This parameter must precede the corresponding TargetTable parameter. The format of the table name is described in “Object names in configuration files” on page 99. You can include wildcard characters as part of the table name. For more information about using wildcard characters in object-name mapping, see “Using wildcards to map object names” on page 226.

This parameter is not valid if your input contains data definition language (DDL) statements in SQL input (or DDL objects in logical log input).

**TargetColumn**

The TargetColumn (TC) parameter lets you specify a column name in the target table that differs from the column name in the source table. Specifying the TargetColumn parameter with a corresponding SourceColumn parameter defines how the High-speed Apply Engine maps the column name for the apply request. You can specify this parameter as many times as necessary to map the columns defined with the SourceColumn parameter.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectMap]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>TC</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid column name from the target table</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

**Usage**

If you specify TargetColumn, the SourceColumn parameter must immediately precede it. You can include wildcard characters as part of the column name. For more information about using wildcard characters in object-name mapping, see “Using wildcards to map object names” on page 226.

This parameter is not valid if your input contains data definition language (DDL) statements in SQL input or DDL objects (in logical log input).
TargetTable

The TargetTable (TT) parameter specifies the name of the target table for the apply request. You must specify a TargetTable parameter for each corresponding SourceTable parameter.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectMap]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>TT</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>the table name of the target table</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

The TargetTable parameter must immediately follow the corresponding SourceTable parameter in the configuration file. The format of the table name is described in “Object names in configuration files” on page 99. You can include wildcard characters as part of the table name. For more information about using wildcard characters in object-name mapping, see “Using wildcards to map object names” on page 226.

This parameter is not valid if your input contains data definition language (DDL) statements in SQL input (or DDL objects in logical log input).

ObjectPartition parameters

[ObjectPartition] parameters enable you to specify logical data partitions for the target tables. The High-speed Apply Engine uses the values that you specify in this section to distribute the workload between apply agents.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column</td>
<td>“Column” on page 234</td>
</tr>
<tr>
<td>LimitKey</td>
<td>“LimitKey” on page 235</td>
</tr>
</tbody>
</table>
You can use these parameters to

- optimize High-speed Apply Engine performance for tables that are not physically partitioned
- optimize performance by using partitions that are different than the physical partitions of the target tables

The following figures show examples of the [ObjectPartition] parameters for a DB2 target on a mainframe, for a DB2 LUW target, and for an Oracle target.

**Figure 46: Example, [ObjectPartition] parameters for DB2 mainframe target**

```
/ObjectPartition/
Table=ACCT.PAYBLE
  Column=AcctNumber
  Column=Date
  Order=DESC
  Partition=CC5700_YTD
    Value=5700
    Value=2008-05-01
  Partition=CC5700_MTD
    Value=5700
    Value=2008-05-18
Table=ACCT.MONTHLY
  Column=TTIME
  Partition=Part1
    LimitKey='000101010080000003'
  Partition=Part3
    LimitKey='00010101008000000A'
```

**Figure 47: Example, [ObjectPartition] parameters for DB2 LUW target**

```
[ObjectPartition]
Table=ACCT.PAYBLE
  Column=AcctNumber
  Column=Date
  Partition=CC5700_YTD
    Value=5700
    Value=2008-05-01
  Partition=CC5700_MTD
    Value=5700
    Value=2008-05-18
```

**Figure 48: Example, [ObjectPartition] parameters for Oracle target**

```
[ObjectPartition]
Table=ACCT.PAYBLE
```
Positional requirements for ObjectPartition parameters

The position of the [ObjectPartition] parameters is significant. If the parameters are out of order, the High-speed Apply Engine issues message number BMCAPT0060E, and apply processing terminates.

You must specify the parameters in the following order. For an example of the parameter order, see Figure 46 on page 231.

1. A Table parameter must be the first parameter in the section.

2. The Column parameter set follows the Table parameter. The Column parameters must also be ordered so that they correspond to their associated Value parameters in each definition that starts with a Partition parameter.

3. If you specify the Order parameter, it must directly follow its associated Column parameter. The Order parameter is valid only for DB2 targets on a mainframe.

4. The first Partition parameter follows all of the Column parameters specified for the table.

5. A set of Value parameters or a LimitKey parameter must come after a Partition parameter. The LimitKey parameter is valid only for DB2 targets on a mainframe. You cannot include both a set of Value parameters and a LimitKey parameter after the same Partition parameter (the two are mutually exclusive).

Partition key length

The High-speed Apply Engine can create partition keys from the columns and values that you specify within a partition definition.

High-speed Apply Engine derives partition keys as follows:
Considerations for Oracle data types

For Oracle targets, there are some limits on the data types and functions that you can use as partition keys.

Date data types

The following considerations apply to partition keys that contain date data types:

- The High-speed Apply Engine accepts dates specified with the TO_DATE function.
- You can also specify a date enclosed in single quotation marks.
- The J date format (Julian date) is not supported.
- When High-speed Apply Engine starts, it reads the date and time formats that are specified in the Oracle NLS settings. If the input file contains an ALTER SESSION command that modifies the formats, the results of the apply request are unpredictable. To define a different default format for an apply request, specify configuration parameters. For more information, see “DateFormat” on page 255, “TimestampFormat” on page 269, or “TimestampTzFormat” on page 270.

Character data types

The High-speed Apply Engine does not support the TO_CHAR function for partition keys that contain character data types (VARCHAR, VARCHAR2, or CHAR).

Numeric data types

The following considerations apply to partition keys that contain numeric data types:
- High-speed Apply Engine does not support the TO_NUMBER function for partition keys that include the FLOAT or NUMBER data types.

- High-speed Apply Engine converts numeric data types to Oracle internal formats.

- If you use a decimal notation that is not a period (.), delimit numeric literals with single quotation marks (for example, '5,123'). Delimiting the literals prompts High-speed Apply Engine to use the decimal notation specified by the NLS_NUMERIC_CHARACTERS initialization parameter.

## Column

The Column parameter specifies a key column name for the logical partition. You can specify this parameter as many times as necessary to define the key columns for the partition.

## Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectPartition]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, Oracle, DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid target column name</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

## Usage

Specify a valid column name in the target table. For each Column parameter that you include, you must also include a Value parameter, or for DB2, compose a single aggregate limit key. For more information about the Value parameter, see “Value” on page 237.

Ensure that the column or columns you specify as the partitioning key resolve to a unique partition key value within the length determined by your target DBMS. For more information, see “Partition key length” on page 232.
LimitKey

The LimitKey parameter corresponds to the LIMITKEY column in the SYSIBM.SYSTABLEPART table of the DB2 catalog. If you know this value, you can define the partition key by specifying the LimitKey parameter (instead of specifying a set of Value parameters for each set of columns).

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectPartition]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>No</td>
</tr>
<tr>
<td>Valid values</td>
<td>a hexadecimal limit key that represents the column and value of the partition key</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

The LimitKey parameter follows the Partition parameter in the [ObjectPartition] section. The value of the LimitKey parameter is an aggregate hexadecimal notation of the values for the Column set. The High-speed Apply Engine converts the value that you specify to an internal format and uses it as the upper limit for the partition key.

Order

For DB2, the Order parameter specifies whether the column that you identify with the Column parameter is sorted in ascending or descending order. The Order parameter must immediately follow its corresponding Column parameter in the [ObjectPartition] section.

Attributes

This parameter has the following attributes:
### Usage

Specify Asc to sort the column in ascending order, or Desc to sort in descending order.

If you do not include the Order parameter for a given column, the sort order is ascending (Asc).

### Partition

The Partition parameter lets you label the partition definition with a specific name. The name that you specify will be used to identify the partition in the output messages related to the apply request.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectPartition]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Asc (ascending) or Desc (descending)</td>
</tr>
<tr>
<td>Default value</td>
<td>Asc</td>
</tr>
</tbody>
</table>
**Usage**

Specify a name that describes the partition. If you include a partition definition without a value for the Partition parameter, the partition will process correctly, but the output messages for the apply request cannot identify the partition.

**Table**

The Table parameter specifies the name of a target table to be partitioned. You can repeat this parameter as many times as necessary to include the tables that you want to logically partition.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectPartition]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid target table name</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

**Usage**

Specify the name of a target table for the apply request. The format of the table name is described in “Object names in configuration files” on page 99. Note that wildcard characters are not permitted for the Table parameter when it is used in the [ObjectPartition] section.

**Value**

The Value parameter identifies the limit of the partition key for a corresponding Column parameter. Together, the Value and Column parameters define a partition key for the target table.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[ObjectPartition]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>the limit or boundary of the partition for the corresponding Column parameter</td>
</tr>
<tr>
<td></td>
<td>Oracle</td>
</tr>
<tr>
<td></td>
<td>you can also specify MAXVALUE</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

The value that you specify must be the same data type as the column of the corresponding Column parameter. If you specify more than one partition, the values that you specify must be unique and in ascending order in the configuration file.

For Oracle, the value represents the partition boundary for the column. The partition contains all keys less than the limit that you specify with the Value parameter. If you specify MAXVALUE for a partition, the partition contains all remaining keys in the table, greater than or equal to the limit specified by the previous partition.

For DB2 and DB2 LUW, the value represents the upper limit of the partition. The partition contains all keys less than or equal to the limit that you specify with the Value parameter.

The product supports the TIMESTAMP WITH TIME ZONE data type. Enter a value in local time; the product converts the time value to Coordinated Universal Time (UTC) when taking the following actions:

- generating the partition key
- before using the data to determine the partitioning group for distribution purposes

The product supports the use of columns defined as TIMESTAMP WITH TIME ZONE as part of a partitioning key with [ObjectPartition]. Enter a value that adheres to the proper format for the column type, using the precision and time zone offset.
Ensure that the column that you specify as the partitioning key resolves to a unique value within the length determined by your target DBMS. For more information, see “Partition key length” on page 232.

## Restart parameters

The [Restart] parameters control processing during a restart.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CheckpointFrequency</td>
<td>“CheckpointFrequency” on page 240</td>
</tr>
<tr>
<td>RestartID</td>
<td>“RestartID” on page 242</td>
</tr>
<tr>
<td>RestartType</td>
<td>“RestartType” on page 243</td>
</tr>
<tr>
<td>RetainTime</td>
<td>“RetainTime” on page 244</td>
</tr>
<tr>
<td>Retry</td>
<td>“Retry” on page 245</td>
</tr>
<tr>
<td>TableName</td>
<td>“TableName” on page 246</td>
</tr>
</tbody>
</table>

The [Restart] section is optional, but BMC Software recommends that you record restart information for your apply requests, so that you can successfully complete apply processing in the event of an operating environment or DBMS failure.

**WARNING**

In general, do not modify the input file of an apply request between the initial run and restart. Modify the configuration file only to the extent required to resolve error conditions. Ensure that changes to the configuration do not alter the distribution of work between agents. For example, do not change the values of configuration parameters such as MaxAgents, DistributionType, PartitionClustering, or RIClustering.

For more information about restarting the High-speed Apply Engine, see “Restart and recovery” on page 63. For more information about using these parameters, see “Summary of restart parameter usage” on page 74.

The following figure shows an example of the [Restart] parameters.

**Figure 49: Example of [Restart] parameter values**

```plaintext
[Restart]
  RestartType=New
  TableName=BMCAPT_APERSCL
  CheckpointFrequency=3
  RetainTime=20
  Retry=5
```
CheckpointFrequency

The CheckpointFrequency parameter determines how often the High-speed Apply Engine performs checkpoint processing to update the restart table with a statement ID that reflects the overall progress of all agents through the input file. When you restart an apply request, High-speed Apply Engine uses this information to speed up restart processing.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Restart]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>■ a positive integer that represents the number of transactions each agent processes between checkpoints</td>
</tr>
<tr>
<td></td>
<td>■ 0 (do not perform checkpoint processing)</td>
</tr>
<tr>
<td>Default value</td>
<td>5</td>
</tr>
</tbody>
</table>

Usage

Specify the number of commit actions that each agent performs before it performs checkpoint processing. Specify 0 to turn off checkpoint processing.

Note

The checkpoint frequency is a number of commit actions for each agent, but High-speed Apply Engine updates the restart table with a statement ID only after all agents have reached their checkpoint threshold.

Use this parameter to balance the overhead of checkpoint processing against the benefits of shorter run time when you restart an apply request. Checkpoint processing imposes a small amount of overhead during an apply request. When you restart an apply request, however, previous checkpoint processing can provide a large benefit because High-speed Apply Engine can begin processing more quickly. (High-speed Apply Engine uses the statement ID in the restart table to read through
Consider the following situations:

- An apply request processes a large input file and failures occur frequently because of communication problems in your environment. In this case, set the CheckpointFrequency parameter to a lower value. Each agent performs checkpoint processing more frequently. The apply request can run slightly slower, but if it fails and you restart it, High-speed Apply Engine reads through the initial portion of the large input file quickly before it begins performing the extra processing. The restarted request runs faster.

- An apply request processes a smaller input file and you know from previous runs that failures are rare. In this case, set the CheckpointFrequency parameter to a higher value. Each agent performs checkpoint processing less frequently. The apply request runs slightly faster. If you have to restart the request, the restarted run takes longer.

Consider the following points as you set a value for CheckpointFrequency.

- Your checkpoint frequency is a number of commit actions. With distribution by object, your commit actions are determined by the values of the StatementCount and Time parameters. (For more information, see “StatementCount” on page 152 and “Time” on page 153.) Consider the values of these parameters as you set the checkpoint frequency. (With distribution by UR, the commit actions are determined by the original transactions in the input SQL statements.) The number of commit actions is also influenced by the number of agents that process an apply request. Consider the values of your [Agent] parameters when you select a value for CheckpointFrequency.

- When High-speed Apply Engine processes data definition language (DDL) statements in an input file, it uses only one agent and issues a COMMIT statement after each DDL statement. This behavior can affect your checkpoint frequency.

**WARNING**

High-speed Apply Engine bases its checkpoint and restart processing on the number and sequence of records in the input file. If you modify an input file between the initial run of an apply request and a subsequent restart and you change the number or sequence of records in the file, the results of the restarted run are unpredictable.

In general, do not modify the input file of an apply request between initial run and restart. Modify the configuration file only to the extent required to resolve error conditions. Ensure that changes to the configuration do not alter the distribution of work between agents. For example, do not change the values of configuration parameters such as MaxAgents, DistributionType, PartitionClustering, or RIClustering.
**RestartID**

The RestartID parameter identifies information in the restart table associated with a specific apply request.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Restart]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>■ RestartType = Restart yes</td>
</tr>
<tr>
<td></td>
<td>■ RestartType = New/Rerstart yes</td>
</tr>
<tr>
<td></td>
<td>■ Other Restart Types no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a character string up to 16 characters long (You cannot specify the string none.)</td>
</tr>
<tr>
<td>Default value</td>
<td>If RestartType = New, a default restart ID will be generated</td>
</tr>
</tbody>
</table>

**Usage**

You can specify a restart ID and RestartType = New for an initial restart ID. After a restart ID exists, if the RestartType parameter is set to New/Rerstart or Restart, the High-speed Apply Engine uses the user-specified restart ID value as the starting point for restart processing.

For more information about using the RestartType parameter and the RestartID parameter, see “Summary of restart parameter usage” on page 74.
RestartType

The RestartType parameter specifies whether the High-speed Apply Engine records restart information for the apply request.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Restart]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>New, None, New/Restart, Restart</td>
</tr>
<tr>
<td>Default value</td>
<td>New</td>
</tr>
</tbody>
</table>

Usage

If you include the [Restart] section in your configuration file, you must specify the RestartType parameter. You can choose one of the following values for this parameter:

- **New** indicates that High-speed Apply Engine saves restart information for this apply request under a specified or generated restart ID. If you specify RestartType=New and you specify a value for RestartID, High-speed Apply Engine uses your restart ID value. If you specify RestartType=New and do not specify a restart ID, High-speed Apply Engine generates a restart ID.

  You can subsequently specify the restart ID to restart the request after a system failure.

- **None** indicates that High-speed Apply Engine does not collect restart information or generate a restart ID for the apply request. If a failure occurs, no restart information is available for the apply request.

- **New/Restart** indicates that High-speed Apply Engine treats this request as either New or Restart, depending on whether restart information exists for the restart ID. If you use RestartType=New/Restart, you must use the RestartID parameter to specify a value for restart ID. If restart information exists for the specified restart ID, restart processing will be performed, otherwise it is treated as a new request.
Restart indicates that High-speed Apply Engine will use the restart information that exists for the restart ID for processing. If you use RestartType=Restart, you must use the RestartID parameter to specify a value for restart ID.

For more information about using the RestartType parameter and the RestartID parameter, see “Summary of restart parameter usage” on page 74.

RetainTime

The RetainTime parameter specifies how long the High-speed Apply Engine retains records in the restart table. This parameter is optional.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Restart]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
</tbody>
</table>
| Valid values    | ■ a positive integer that represents the number of days to retain restart records  
                  ■ All                                                             |
| Default value   | 7                                                                    |

Usage

Specify the number of days that you want High-speed Apply Engine to keep restart records in the current restart table. Specify All to bypass record deletion and retain all records in the current restart table.

When you specify a number of days, consider the following criteria:

■ how many of your apply requests use the same restart table
■ the number of transactions processed by your apply requests
■ the available storage space on the system that contains the restart table
High-speed Apply Engine issues message BMCAPT0191 and deletes information from the restart table when either of the following conditions are true:

- High-speed Apply Engine is not restarting a previous apply request and any restart information in the current restart table is older than the number of days specified by the RetainTime parameter. This situation can occur either when RestartType is New, or when RestartType is New/Restart and the previous run of the request was successful.

  For this condition, High-speed Apply Engine deletes all restart information in the current restart table that is older than the RetainTime value. (To preserve restart information for one restart ID when information for other restart IDs is older than the RetainTime value, use separate restart tables. For more information, see “Creating a restart table” on page 64.)

- Any restart information (regardless of its age) exists in the restart table for the current restart ID, but RestartType is New.

  If restart information exists, the previous run of the request was unsuccessful, but a restart type of New indicates that you want to bypass the restart and start from the beginning of the input file. For this condition, High-speed Apply Engine deletes only the restart information for the current restart ID.

### Retry

The Retry parameter specifies the number of times that the High-speed Apply Engine tries to access the restart table if the table is unavailable. (For example, High-speed Apply Engine tries to access the restart table again if it receives a resource-unavailable status from DB2, such as -904 or -913).

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Restart]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer that represents the number of retry attempts</td>
</tr>
<tr>
<td>Default value</td>
<td>3</td>
</tr>
</tbody>
</table>
**Usage**

If the restart table is unavailable when you start the request, High-speed Apply Engine attempts to access the table repeatedly. It continues to try to access the restart table until one of the following situations occurs:

- The restart table becomes available.
- The number of attempts to access the restart table equals the value that you specified for the Retry parameter.

If you do not include the Retry parameter, High-speed Apply Engine uses the default value.

**TableName**

The TableName parameter specifies the name of the DBMS table that contains restart information for the apply request. To enable restart processing, a restart table must be available for the High-speed Apply Engine to store restart information.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[Restart]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid table name (nondelimited), an alias, a DB2 synonym, or an Oracle public synonym</td>
</tr>
</tbody>
</table>
| Default value | - Mainframe operating systems  
BMCAPT_APTRSCL (alias)  
- UNIX and Windows  
BMCAPE_APERSCL (public synonym) |
Usage

The table that you specify must exist when you run the apply request. It must reside in the same DB2 subsystem, DB2 LUW database, or Oracle instance as the target tables. Use one of the formats shown in the following table.

If you specify an unqualified name for the restart table, be aware of the value of the BindQualifier parameter in your configuration (for more information, see “BindQualifier” on page 253).

Table 17: Restart table name formats

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner.tableName</td>
<td>To fully qualify a table name, specify the owner and table name, as follows:</td>
</tr>
<tr>
<td></td>
<td>- The owner is the user ID of the table owner.</td>
</tr>
<tr>
<td></td>
<td>- The period is a required separator.</td>
</tr>
<tr>
<td></td>
<td>- The tableName is the DB2, DB2 LUW or Oracle table name.</td>
</tr>
<tr>
<td>tableName</td>
<td>If you specify a table name without an owner, High-speed Apply Engine uses</td>
</tr>
<tr>
<td></td>
<td>the user ID that runs the apply request as the owner of the table.</td>
</tr>
<tr>
<td>synonym</td>
<td>This can be either an unqualified DB2 synonym or an Oracle public synonym.</td>
</tr>
<tr>
<td></td>
<td>- DB2</td>
</tr>
<tr>
<td></td>
<td>High-speed Apply Engine uses the user ID that runs the apply request as</td>
</tr>
<tr>
<td></td>
<td>the owner of the synonym.</td>
</tr>
<tr>
<td></td>
<td>- DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>if the default name is specified or defaulted,</td>
</tr>
<tr>
<td></td>
<td>High-speed Apply Engine uses the user ID that runs the apply request as</td>
</tr>
<tr>
<td></td>
<td>the owner of the synonym.</td>
</tr>
<tr>
<td></td>
<td>If the table is not found, High-speed Apply Engine uses the user ID</td>
</tr>
<tr>
<td></td>
<td>used to BIND the bind files as the owner of the synonym.</td>
</tr>
</tbody>
</table>
You can fully qualify a synonym for a table name by specifying owner and synonym, as follows:

- The owner is the user ID of the synonym owner.
- The period is a required separator.
- The synonym refers to a table.

For DB2, if the default name is specified or defaulted, High-speed Apply Engine uses the user ID that runs the apply request as the owner of the alias. If the table is not found, High-speed Apply Engine uses the owner of the plan as the owner of the alias.

You can fully qualify an alias for a table name by specifying owner and alias, as follows:

- The owner is the user ID of the alias owner.
- The period is a required separator.
- The alias refers to a table.

By default, the High-speed Apply Engine uses a single restart table for all apply requests. (This table can be the restart table created during installation, or one that you designate.) In normal use, this practice works well. However, if you run a large number of apply requests simultaneously on the same target database, you might begin to experience resource contention for the single restart table.

If you experience resource contention, create a restart table for each apply request that runs simultaneously on the target database. This practice ensures that resource contention for the restart table will not occur.

For more information, see “Creating a restart table” on page 64.

**SQL parameters**

[SQL] parameters define values for SQL input for the apply request.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DecimalPoint</td>
<td>“DecimalPoint” on page 249</td>
</tr>
<tr>
<td>Format</td>
<td>“Format” on page 250</td>
</tr>
</tbody>
</table>
The following figure shows an example of the [SQL] parameters.

**Figure 50: Example of [SQL] parameter values**

```
[SQL]
DecimalPoint=,
Format=STANDARD
```

**DecimalPoint**

The DecimalPoint parameter specifies which character the High-speed Apply Engine expects to represent the decimal point in the input SQL file.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[SQL]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Period (.), or comma (,)</td>
</tr>
<tr>
<td>Default value</td>
<td>■ DB2</td>
</tr>
<tr>
<td></td>
<td>High-speed Apply Engine uses character defined as decimal point in the DSNHDECP module of current subsystem</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>High-speed Apply Engine extracts character from the locale setting obtained from the current operating system</td>
</tr>
<tr>
<td></td>
<td>■ Oracle</td>
</tr>
<tr>
<td></td>
<td>High-speed Apply Engine obtains character from the nls_session_parameters (specifically, nls_numeric_characters)</td>
</tr>
</tbody>
</table>
Usage

Some environments use target DBMS options to define a comma (,) instead of a period (.) as the decimal point in any numeric values contained within SQL statements. If your input SQL file is generated in such an environment, use this parameter to direct High-speed Apply Engine to interpret a comma within a numeric value as a decimal point character.

**WARNING**

Exercise caution when the decimal point character of the input file is not supported by the target database. If High-speed Apply Engine processes an input statement as EXECUTE IMMEDIATE, the statement contains the input decimal point character. If that character is not supported by the target database, the results are unpredictable.

Format

The Format parameter specifies the format of SQL input statements.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[SQL]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Standard or Packed72</td>
</tr>
<tr>
<td>Default value</td>
<td>■ DB2</td>
</tr>
<tr>
<td></td>
<td>Packed72</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>Standard</td>
</tr>
<tr>
<td></td>
<td>■ Oracle</td>
</tr>
<tr>
<td></td>
<td>Standard</td>
</tr>
</tbody>
</table>
Usage

Specify the format of the SQL statements that will serve as input for the apply request.

- Specify Standard if your input is coming from Oracle or DB2 LUW for UNIX or Windows.
  
  Standard refers to a format of SQL variable-length statements. This value indicates that the input file contains regular SQL with statements on one or more lines. (Quoted strings cannot span multiple lines.)

- Specify Packed72 if your input is coming from a mainframe source. (Most SQL from mainframe sources is generated in fixed-length data sets.) This format is compatible with the DB2 SPUFI utility. If the file does not match this format, results are unpredictable.
  
  Packed72 refers to a format of SQL fixed-length records which are in packed 72 bytes. This format indicates that the input file contains SQL was generated by Log Master for DB2 on a mainframe source or formatted by the user. This format is processed as though all rows in the file are 72 bytes long and quoted strings can span multiple lines.

StartUp parameters

The [StartUp] (SU) parameters specify initialization options for the High-speed Apply Engine. The parameters that you specify vary depending on the DBMS and operating system platform where you run High-speed Apply Engine.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>BindQualifier</td>
<td>“BindQualifier” on page 253</td>
</tr>
<tr>
<td>ConnectionVersion</td>
<td>“ConnectionVersion” on page 253</td>
</tr>
<tr>
<td>Database</td>
<td>“Database” on page 254</td>
</tr>
<tr>
<td>DateFormat</td>
<td>“DateFormat” on page 255</td>
</tr>
<tr>
<td>DBMSType</td>
<td>“DBMSType” on page 256</td>
</tr>
<tr>
<td>DefaultOpts (DOPT)</td>
<td>“DefaultOpts” on page 257</td>
</tr>
<tr>
<td>DistributionType</td>
<td>“DistributionType” on page 258</td>
</tr>
<tr>
<td>FileName (IFN)</td>
<td>“FileName (IFN)” on page 259</td>
</tr>
<tr>
<td>InputMediaType</td>
<td>“InputMediaType” on page 261</td>
</tr>
<tr>
<td>InputType (IT)</td>
<td>“InputType” on page 262</td>
</tr>
<tr>
<td>LargeStatementSupport</td>
<td>“LargeStatementSupport” on page 263</td>
</tr>
</tbody>
</table>
For DB2 targets on a mainframe, the following figure shows an example of the [StartUp] parameters. The [StartUp] section is required. You must specify the FileName, SSID, and PlanName parameters.

**Figure 51: Example of [StartUp] parameters for DB2 on mainframe targets**

```
/StartUp/
  FileName=USER14.MIGRATE.SQL
  InputType=SQL
  SSID=DBAM
  PlanName=APTSMDB2
```

For DB2 LUW, the following figure shows an example of the [StartUp] parameters. The [Startup] section is required for DB2 LUW targets. You must specify the FileName, Database, and DbmsType parameters.

**Figure 52: Example of [StartUp] parameters for DB2 LUW**

```
[StartUp]
  FileName=/home/operations/migrate.sql
  Database=ACCT
  DbmsType=DB2
```

For Oracle, the following figure shows an example of the [StartUp] parameters. If the target database is configured for automated login, and the input file name is the default name, the [Startup] section is not required. If you do not use automated login, you must include the Login parameter.

**Figure 53: Example of [StartUp] parameters for Oracle**

```
[StartUp]
  FileName=User14.Migrate.SQL
  Login=user14/******@Ora10
```
BindQualifier

For DB2, the BindQualifier parameter specifies a qualifier value that High-speed Apply Engine uses to execute dynamic SQL against the restart table if the restart table is defined in your configuration with an unqualified name. High-speed Apply Engine does not use this parameter when it binds packages to apply user data (from either SQL or logical log input).

Other BMC Software products for DB2 use the value of this parameter during execution to dynamically bind internal packages that are distributed with the products (frequently as the QUALIFIER value in a BIND PACKAGE command). High-speed Apply Engine does not use this parameter in this way because all SQL distributed with High-speed Apply Engine contains fully-qualified names.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>A string up to 16 characters long that is a valid DB2 qualifier value</td>
</tr>
<tr>
<td>Default value</td>
<td>APT vv&lt;sub&gt;r&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

Usage

Specify any value that is valid for the QUALIFIER option of a DB2 BIND command, up to 16 bytes in length. The default value is APT vv<sub>r</sub>, where vv represents a two-character version number and r represents a one-character release number.

You can specify this parameter in the installation options module during the install process or you can specify it in the configuration file.

ConnectionVersion

The ConnectionVersion parameter identifies the version number of the Oracle client and server.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any supported version of the Oracle client and server software. For</td>
</tr>
<tr>
<td></td>
<td>a list of supported versions, see “Installing High-speed Apply Engine</td>
</tr>
<tr>
<td></td>
<td>on UNIX or Windows” on page 299.</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

Specify the version number of the Oracle client and server for the target tables. Enter the version number in the format v.r.m, separated by periods, for example:

```
ConnectionVersion=11.2.0
```

If you do not specify this parameter, the High-speed Apply Engine attempts to determine the Oracle version for the target tables by using the SQL*Plus program. The directory that contains SQL*Plus must be included in the path for the server, so that High-speed Apply Engine can run the program. If High-speed Apply Engine cannot determine the Oracle version number from the configuration or the server, it cannot process the apply request.

You need to specify this parameter only if High-speed Apply Engine cannot determine the version. In that case High-speed Apply Engine generates the message BMCAPT0123, Unable to determine connection version.

Database

For DB2 LUW, the Database parameter specifies the name of the target database.

Attributes

This parameter has the following attributes:
**Usage**

Specify the database name of the DB2 LUW database that contains the target tables for the apply request.

**DateFormat**

The DateFormat defines the default date format that the High-speed Apply Engine uses when it updates date columns in an Oracle target database.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Any valid Oracle date format mask for example: 'MM/DD/YYYY'</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

**Usage**

Specify a valid Oracle date format mask that matches the dates in your input.
Consider the following points regarding this parameter:

- Use this parameter when your input was generated in another Oracle environment (or on a DBMS other than Oracle) that uses a *different* date format than the format of the date columns in your target tables. When you specify this parameter, the High-speed Apply Engine issues an ALTER SESSION SET statement to change the date format (the `nls_date_format` parameter).

- For valid date format masks, consult the Oracle DBMS technical documentation.

### DBMSType

The DBMSType parameter specifies the database type for the tables that contain information for the apply request.

#### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>DB2, Oracle</td>
</tr>
<tr>
<td></td>
<td>Oracle</td>
</tr>
<tr>
<td></td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>yes</td>
</tr>
<tr>
<td>Valid values</td>
<td>DB2, Oracle</td>
</tr>
<tr>
<td>Default value</td>
<td>Mainframe operating systems</td>
</tr>
<tr>
<td></td>
<td>DB2</td>
</tr>
<tr>
<td></td>
<td>UNIX, Windows</td>
</tr>
<tr>
<td></td>
<td>Oracle</td>
</tr>
</tbody>
</table>
Usage

The DBMSType parameter is not required. If you choose to indicate the DBMSType, specify Oracle for an Oracle database or DB2 for a mainframe or a DB2 LUW target.

If you are using DB2 LUW on a UNIX or Windows platform, the [StartUp] section of your configuration file must specify DBMSType=DB2.

DefaultOpts

For DB2, the DefaultOpts (DOPT) parameter specifies the name of an installation options module that you have created that contains default configuration values.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>DOPT</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid module name or the word None</td>
</tr>
<tr>
<td>Default value</td>
<td>APTDOPTS</td>
</tr>
</tbody>
</table>

Usage

Use the DefaultOpts parameter to specify the name of a user-created installation options module. Use None as the value if you do not want to use an installation options module.

You can generate a default configuration options module during installation. For more information, see “Installing High-speed Apply Engine on mainframe targets” on page 285.

Note

Any configuration information specified in the apply request or the configuration file will override the values in the installation options module.
DistributionType

The DistributionType parameter selects one of two major techniques that the High-speed Apply Engine can use to distribute work between multiple agents.

For more information about these two options, see “Distribution of work” on page 31.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Object, UR</td>
</tr>
<tr>
<td>Default value</td>
<td>Object</td>
</tr>
</tbody>
</table>

Usage

Specify one of the following values:

- Object directs High-speed Apply Engine to distribute work based on the tables or partitions that the SQL statements in the input file update. This technique does not preserve transaction integrity during the apply request. This is the default value.

- UR directs High-speed Apply Engine to distribute work based on the units of recovery (transactions) contained in the input file. High-speed Apply Engine uses the same agent to process all SQL statements in an original transaction, regardless of what tables or partitions the statements update. This technique preserves transaction integrity during the apply request.

Use this parameter to override the High-speed Apply Engine default value and select distribution by UR. With distribution by UR, the same agent processes a given transaction, but different agents can process two different transactions that update the same tables or partitions. Distribution by UR causes High-speed Apply Engine to honor the original COMMIT or ROLLBACK statements that occur in the input.

Under distribution by UR, High-speed Apply Engine ignores the values of configuration parameters related to distribution by object (such as
Under distribution by UR, High-speed Apply Engine ensures that it can restart the apply request by directing multiple agents to wait as needed to preserve the order of the original transactions. High-speed Apply Engine can also direct agents to wait to accommodate dependencies between transactions. The following dependencies cause High-speed Apply Engine to coordinate the actions of multiple agents.

- when statements in two different transactions update the same table
  High-speed Apply Engine directs the agent processing the later transaction to process all statements up to the first statement that updates the common table. Then the agent processing the later transaction waits until the agent processing the earlier transaction finishes and performs a commit action.

- when statements in two different transactions update tables that are part of a database-defined referential integrity (RI) relationship
  High-speed Apply Engine directs the agent processing the later transaction to wait until the agent processing the earlier transaction finishes and performs a commit action. Use the URCheckRI parameter to control whether High-speed Apply Engine accommodates this type of dependency. (For more information, see “URCheckRI” on page 189.) When URCheckRI is Yes, the RIClustering parameter does not apply.

- when any transaction contains a data definition language (DDL) statement
  Other agents continue to process transactions without DDL statements. However, High-speed Apply Engine does not assign any additional transactions to agents for processing until after the agent processing the DDL statement finishes and performs a commit action. High-speed Apply Engine takes this action in case the DDL statement changes the structure of the tables being updated by other agents.

Under distribution by UR, High-speed Apply Engine honors the COMMIT or ROLLBACK statements in your input, but it might perform additional commit or rollback actions in response to certain Action parameters of a conflict resolution rule or to resource shortages.

**FileName (IFN)**

The FileName (IFN) parameter specifies the name of the input file for the apply request.

The High-speed Apply Engine accepts the following types of input:

- SQL generated by any standard SQL generator
logical log control files generated by Log Master for DB2 (or other application programs that conform to the published format)

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>IFN</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>■ DB2&lt;br&gt;a fully qualified data set name or a DD name&lt;br&gt;■ DB2 LUW or Oracle&lt;br&gt;a file name, including the path name if outside of the default path</td>
</tr>
<tr>
<td>Default value</td>
<td>■ Mainframe operating systems&lt;br&gt;SQLIN for an SQL input file and LLOGIN for a logical log input file&lt;br&gt;■ UNIX and Windows&lt;br&gt;no default value</td>
</tr>
</tbody>
</table>

**Usage**

For the FileName parameter, you can specify any valid data set or DD name, or a UNIX or Windows file name. For more information about file names on different operating systems, see “Usage” on page 156.

For z/OS, you can specify any valid data set, including a partitioned data set (PDS) or a generation data group (GDG) name.

As with other parameters that require a file name, when you specify a name that is eight characters or fewer, High-speed Apply Engine treats the name as a DD name. The DD name must refer to a DD statement in the apply request. However, in the case of the input file, the default DD name depends on the input type, as follows:
- For SQL input, the default is SQLIN.
- For a logical log file, the default is LLOGIN.

If the character string is not a valid DD name, High-speed Apply Engine treats it as a data set name.

**Note**
If you include the SQLIN or LLOGIN DD statement in the apply request, and you do not specify the InputType parameter, High-speed Apply Engine uses the DD name to determine whether the input file is SQL or a logical log control file. If you include both DD statements in an apply request, you must specify the InputType parameter, or the request cannot be processed.

### InputMediaType
For DB2, the InputMediaType parameter specifies the media type of the input to the High-speed Apply Engine.

#### Attributes
This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>None</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>No</td>
</tr>
<tr>
<td>Valid values</td>
<td>File or Queue (Message Queue)</td>
</tr>
<tr>
<td>Default value</td>
<td>By default, High-speed Apply Engine attempts to determine the media type of the input source by examining the value of other parameters.</td>
</tr>
</tbody>
</table>

#### Usage
In most situations, High-speed Apply Engine can determine the media type of the input from the value of other parameters. Specify this parameter only if the input media cannot be determined by other parameters.

Specify File if the input media is a file.
InputType

The InputType (IT) parameter specifies whether the input file is a logical log control file or an SQL file.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>IT</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Parameter</td>
<td>Reference</td>
</tr>
<tr>
<td>Valid values</td>
<td>■ DB2</td>
</tr>
<tr>
<td></td>
<td>LogicalLog or SQL</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>SQL</td>
</tr>
<tr>
<td></td>
<td>■ Oracle</td>
</tr>
<tr>
<td></td>
<td>SQL</td>
</tr>
<tr>
<td>Default value</td>
<td>SQL</td>
</tr>
</tbody>
</table>

Usage

Specify one of the following values for the InputType parameter:

- LogicalLog indicates that the input file is a logical log control file generated by Log Master for DB2 (or by an application program that conforms to the published format).

- SQL indicates that the input file contains only SQL statements. To provide restartability, ensure that your input file does not contain more than one SQL statement on the same line.

If you do not specify the InputType parameter, and High-speed Apply Engine cannot determine the input type from the default DD name for the input file, High-speed Apply Engine cannot process the apply request.
LargeStatementSupport

The LargeStatementSupport parameter specifies whether the High-speed Apply Engine can process SQL statements or logical log input records that are longer than 65,536 bytes (64K).

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes, No</td>
</tr>
<tr>
<td>Default value</td>
<td>No</td>
</tr>
</tbody>
</table>

Usage

Specify one of the following values for the LargeStatementSupport parameter:

- No directs High-speed Apply Engine to issue error message BMCAPI0046 if either an SQL statement or a logical log record in the input file is longer than 64K bytes. No is the default value.

- Yes directs High-speed Apply Engine to extend the limit on an input statement or record to 327,680 bytes (320K).

If you are processing logical log input that includes data from large volume columns (such as XML or LOB columns), consider specifying Yes for this parameter to accommodate the larger amount of data. If processing in your environment requires that you process an SQL statement or a logical log record that is longer than 320K, contact BMC Software Customer Support for assistance.

Login

For an Oracle or DB2 LUW target, the Login parameter specifies all of the startup information required for an apply request. The parameter includes the user ID, password, and database name for the target database.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>■ Oracle</td>
</tr>
<tr>
<td></td>
<td>yes (unless the Oracle login is automated)</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>■ Oracle</td>
</tr>
<tr>
<td></td>
<td>a user ID and password in the format:</td>
</tr>
<tr>
<td></td>
<td>'userId/password'</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>a user ID, password, and database name (formats):</td>
</tr>
<tr>
<td></td>
<td>@ databaseName</td>
</tr>
<tr>
<td></td>
<td>' userID / password@ databaseName '</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

Specify values for the Login parameter as follows:

■ Specify a user ID and password with the appropriate database authority for the apply request.

■ Specify the name of the instance or database that contains the target tables for the request.

If the target Oracle instance is configured for automated login, you do not need to include the Login parameter in your configuration. If your Oracle user ID or password include lowercase characters, enclose the values in quotes to prevent the High-speed Apply Engine from converting the characters. For example,
' userID / password ' 

By default, High-speed Apply Engine converts the user ID and password to uppercase characters. If your DB2 LUW user ID or password include lowercase characters, enclose the values in quotes to prevent High-speed Apply Engine from converting the characters. For example,

' userID / password@databaseName '

**Password**

For DB2 LUW, the Password parameter specifies the password associated with the user ID used to access the target database.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>When accessing DB2 LUW through a client yes</td>
</tr>
<tr>
<td></td>
<td>When running DB2 LUW on the server no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any valid password in the format 'password'</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

**Usage**

The password is used in conjunction with the user ID. For more information, see “UserID” on page 271.

The need for Password and UserID depends on the authentication settings on the server to which the High-speed Apply Engine connects. For more information about the authentication settings, consult the server documentation.

By default, High-speed Apply Engine converts the value of both UserID and Password to uppercase characters. If your DB2 LUW user ID or password includes...
lowercase characters, enclose the value in quotes to prevent High-speed Apply Engine from converting the characters (for example ‘password’).

**PlanName**

The PlanName (PLAN) parameter specifies the name of the DB2 plan that the High-speed Apply Engine uses to access the DB2 catalog tables and the restart tables.

*Note*

High-speed Apply Engine does not use this plan to access your target DB2 tables. To specify the plan name that accesses the target tables, use the PlanName parameter in the [Bind] section. (For more information, see “PlanName” on page 136).

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>PLAN</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid DB2 plan name</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
<tr>
<td></td>
<td>(default value in installation options module is APTBvvv, where vvvv represents the version and release level of High-speed Apply Engine)</td>
</tr>
</tbody>
</table>

**Usage**

This plan name can be specified in the installation options module during the install process or you can specify it with this parameter, but High-speed Apply Engine must have this plan name in order to run an apply request.

**PublicPlan**

For DB2, the PublicPlan parameter determines whether the High-speed Apply Engine grants the authority to run High-speed Apply Engine (execute against the base plan) to PUBLIC.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>None</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>No</td>
</tr>
<tr>
<td>Valid values</td>
<td>Yes or No</td>
</tr>
<tr>
<td>Default value</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Usage

Use this option to avoid specific grants of authority to individual users that run High-speed Apply Engine. Specify one of the following values for the PublicPlan parameter:

- Specify a value of Yes (the default value) to direct High-speed Apply Engine to grant EXECUTE authority on the base plan to PUBLIC. All users have authority to run High-speed Apply Engine.

- Specify a value of No to prevent High-speed Apply Engine from granting EXECUTE authority on the base plan to PUBLIC. If you specify No, you must grant EXECUTE authority on the base plan to all specific users that run High-speed Apply Engine.

Be aware of the following points regarding the PublicPlan parameter and the High-speed Apply Engine base plan:

- In most cases, BMC Software recommends accepting the default value of Yes. You can specify this parameter in the installation options module during the install process or you can specify it in the configuration file.

- The name of the High-speed Apply Engine base plan is determined by the PlanName parameter (“PlanName” on page 266).

SQLTerminator

The SQLTerminator parameter specifies the character that ends each SQL statement in the input file. This parameter allows you to set the SQL terminator character to something other than a semicolon (;), which is the default.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2, DB2 LUW, Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>■ DB2</td>
</tr>
<tr>
<td></td>
<td>any character except space, comma, single quote ('), double quote (&quot;), left parenthesis, right parenthesis, or underscore (_)</td>
</tr>
<tr>
<td></td>
<td>■ DB2 LUW</td>
</tr>
<tr>
<td></td>
<td>any character except space, left parenthesis, or right parenthesis</td>
</tr>
<tr>
<td></td>
<td>■ Oracle</td>
</tr>
<tr>
<td></td>
<td>any character except space, single quote ('), or double quote (&quot;)</td>
</tr>
<tr>
<td>Default value</td>
<td>semicolon (;)</td>
</tr>
</tbody>
</table>

Usage

If your SQL statements terminate with a character other than a semicolon, use the SQLTerminator parameter to specify that character. The High-speed Apply Engine also uses the current terminator (from the configuration or from the SQL file) for any statements written to the conflict file.

You can also designate a terminator character in your input SQL. For more information, consult the DBMS documentation.

SSID

The SSID parameter specifies the DB2 subsystem ID or the group attachment name for a data sharing group.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>yes</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid DB2 SSID or group attachment name</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

Usage

Specify the SSID of the DB2 subsystem or the group attachment name of the data sharing group that contains the target tables. If you do not specify this parameter, the apply request cannot be processed.

TimeStampFormat

The TimeStampFormat defines the default timestamp format that the High-speed Apply Engine uses when it updates timestamp columns in an Oracle target database.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Any valid Oracle datetime format mask for example: 'YYYY-MM-DD HH:MI:SS.FF'</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>
Usage

Specify a valid Oracle datetime format mask that matches the timestamps in your input.

Consider the following points regarding this parameter:

- Use this parameter when your input was generated on a DBMS other than Oracle (or a different Oracle environment) that uses a different timestamp format than the format of the timestamp columns in your target tables. When you specify this parameter, High-speed Apply Engine issues an ALTER SESSION SET statement to change the timestamp format (the nls_timestamp_format parameter).

- For valid datetime format masks, consult the Oracle DBMS technical documentation.

- High-speed Apply Engine uses this parameter only when the target version of the Oracle DBMS is 9i or later. For earlier versions, High-speed Apply Engine ignores this parameter.

TimestampTzFormat

The TimestampTzFormat defines the default timestamp with time zone format that the High-speed Apply Engine uses when it updates timestamp columns in an Oracle target database.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>Oracle</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>Any valid Oracle datetime format mask for example: 'YYYY-MM-DD HH:MI:SS.FF TZH:TZM'</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>
Usage

Specify a valid Oracle datetime format mask that matches the timestamps and time zones in your input.

Consider the following points regarding this parameter:

- Use this parameter when your input was generated on a DBMS other than Oracle (or in a different Oracle environment) that uses a different timestamp with time zone format than the format of the timestamp columns in your target tables. When you specify this parameter, High-speed Apply Engine issues an ALTER SESSION SET statement to change the timestamp with time zone format (the nls_timestamp_tz_format parameter).

- For valid datetime format masks, consult the Oracle DBMS technical documentation.

- High-speed Apply Engine uses this parameter only when the target version of the Oracle DBMS is 9i or later. For earlier versions, High-speed Apply Engine ignores this parameter.

UserID

For DB2 LUW, the UserID parameter specifies the user ID that the High-speed Apply Engine uses to access the target database.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[StartUp]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>When accessing DB2 LUW through a client yes</td>
</tr>
<tr>
<td></td>
<td>When running DB2 LUW on the server no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid user ID</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>
Usage

Specify a user ID and password with the appropriate database authority for the apply request.

Generally, the UserID parameter is required if you are accessing DB2 LUW through a client. It is not required if you are running DB2 LUW on the server. However, the need for Password and UserID depends on the authentication settings on the server to which High-speed Apply Engine connects. For more information about the authentication settings, consult the DB2 LUW server documentation.

By default, High-speed Apply Engine converts the value of both UserID and Password to uppercase characters. If your DB2 LUW user ID or password includes lowercase characters, enclose the value in quotes to prevent High-speed Apply Engine from converting the characters (for example ‘userID’).

TemporaryFiles parameters

The [TemporaryFiles] parameters specify how to store the temporary files created during apply processing.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATACLAS</td>
<td>“DATACLAS” on page 273</td>
</tr>
<tr>
<td>Directory (DIR)</td>
<td>“Directory” on page 273</td>
</tr>
<tr>
<td>Like</td>
<td>“LIKE” on page 274</td>
</tr>
<tr>
<td>MGMTCLAS</td>
<td>“MGMTCLAS” on page 275</td>
</tr>
<tr>
<td>SPACE</td>
<td>“SPACE” on page 275</td>
</tr>
<tr>
<td>SpacePri</td>
<td>“SpacePri” on page 276</td>
</tr>
<tr>
<td>SpaceSec</td>
<td>“SpaceSec” on page 277</td>
</tr>
<tr>
<td>STORCLAS</td>
<td>“STORCLAS” on page 277</td>
</tr>
<tr>
<td>UNIT</td>
<td>“UNIT” on page 278</td>
</tr>
<tr>
<td>VOLSER</td>
<td>“VOLSER” on page 279</td>
</tr>
</tbody>
</table>

The following figure shows an example of the [TemporaryFiles] parameters.

Figure 54: Example of [TemporaryFiles] parameter values

[TemporaryFiles]
STORCLAS=TEMP
For z/OS, the DATACLAS parameter identifies a data class defined by DFSMS. The High-speed Apply Engine uses the data class when it allocates temporary files.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Section</strong></td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td><strong>Abbreviation</strong></td>
<td>none</td>
</tr>
<tr>
<td><strong>DBMS</strong></td>
<td>DB2</td>
</tr>
<tr>
<td><strong>Required?</strong></td>
<td>no</td>
</tr>
<tr>
<td><strong>Valid values</strong></td>
<td>a data class that is defined in DFSMS for your environment</td>
</tr>
<tr>
<td><strong>Default value</strong></td>
<td>none</td>
</tr>
</tbody>
</table>

**Usage**

The data class defines data attributes for DFSMS-managed data sets. This parameter applies only to systems that use DFSMS. If your environment does not use DFSMS, High-speed Apply Engine specifies the DATACLAS parameter in its allocation request, but the operating system does not honor its value.

The value of this parameter is also used as the default value of the DATACLAS parameter in the [ConflictFile] section.

**Directory**

For DB2 LUW, the Directory (Dir) parameter specifies where to store the temporary files created during apply processing.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Section</strong></td>
<td>[TemporaryFiles]</td>
</tr>
</tbody>
</table>
### Attribute

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abbreviation</td>
<td>Dir</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2 LUW</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid path name</td>
</tr>
<tr>
<td>Default value</td>
<td>High-speed Apply Engine checks environment variables TEMP, TMP or TMPDIR; if these variables are not set, High-speed Apply Engine uses the current working directory.</td>
</tr>
</tbody>
</table>

#### Usage

High-speed Apply Engine uses this parameter to determine where to store any temporary files created during processing.

#### LIKE

For z/OS, the LIKE parameter lets you specify an existing data set as a model for the allocation attributes of all temporary files.

#### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>the name of an existing data set that has the allocation attributes that you want to use for all temporary files</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>
Engine allocates new temporary files, it will use the same allocation attributes as those of the existing data set.

**MGMTCLAS**

For z/OS, the MGMTCLAS parameter identifies a management class defined in DFSMS. The High-speed Apply Engine uses the management class when it allocates temporary files.

**Attributes**

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a management class that is defined in DFSMS for your environment</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

**Usage**

The management class defines how DFSMS manages the data sets in the class. This parameter applies only to systems that use DFSMS. If your environment does not use DFSMS, High-speed Apply Engine specifies the MGMTCLAS parameter in its allocation request, but the operating system does not honor its value.

The value of this parameter is also used as the default value of the MGMTCLAS parameter in the [ConflictFile] section.

**SPACE**

For z/OS, the SPACE parameter identifies the allocation unit to be used for all temporary files.
Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>CYLINDER, CYL, TRACK, or TRK</td>
</tr>
<tr>
<td>Default value</td>
<td>TRACK</td>
</tr>
</tbody>
</table>

Usage

Specify an appropriate allocation unit for your environment, or accept the default value (TRACK). The value that you specify determines the unit of measurement for the SpacePri and SpaceSec parameters.

SpacePri

For z/OS, the SpacePri parameter specifies the primary amount of space that is allocated for each temporary file. The unit of measurement for this parameter is determined by the SPACE parameter.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer greater than or equal to 1</td>
</tr>
<tr>
<td>Default value</td>
<td>15</td>
</tr>
</tbody>
</table>
Usage

Specify a value that allocates enough primary storage space for the temporary files that the apply request might generate.

SpaceSec

For z/OS, the SpaceSec parameter specifies the secondary amount of space that is allocated for each temporary file. The unit of measurement for the SpaceSec parameter is determined by the SPACE parameter.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>an integer greater than or equal to 0</td>
</tr>
<tr>
<td>Default value</td>
<td>5</td>
</tr>
</tbody>
</table>

Usage

Specify a value that allocates enough secondary storage space for the temporary files that the apply request might generate.

STORCLAS

For z/OS, the STORCLAS parameter specifies a storage class defined by the IBM product Data Facility Storage Management System (DFSMS). The High-speed Apply Engine uses the storage class to allocate temporary files for apply processing.

Attributes

This parameter has the following attributes:
### Usage

This parameter applies only to systems that use DFSMS. If your environment does not use DFSMS, High-speed Apply Engine specifies the STORCLAS parameter in its allocation request, but the operating system does not honor its value.

The value of this parameter is also used as the default value of the STORCLAS parameter in the [ConflictFile] section.

### UNIT

For z/OS, the UNIT parameter specifies a storage unit type that the High-speed Apply Engine uses to allocate temporary files for apply processing.

### Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>any valid storage unit identifier for your environment, except VIO</td>
</tr>
<tr>
<td>Default value</td>
<td>SYSALLDA</td>
</tr>
</tbody>
</table>
Usage

If you specified a value for the STORCLAS parameter, you do not need to specify a unit type. If your environment uses DFSMS, High-speed Apply Engine specifies the UNIT parameter in its allocation request, but the operating system does not honor its value.

VOLSER

For z/OS, the VOLSER parameter specifies the volume serial number that the High-speed Apply Engine uses to allocate temporary files for apply processing. If you specified a value for the STORCLAS parameter, you do not need to specify a volume serial number.

Attributes

This parameter has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section</td>
<td>[TemporaryFiles]</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>none</td>
</tr>
<tr>
<td>DBMS</td>
<td>DB2</td>
</tr>
<tr>
<td>Required?</td>
<td>no</td>
</tr>
<tr>
<td>Valid values</td>
<td>a valid volume serial number for a storage device</td>
</tr>
<tr>
<td>Default value</td>
<td>none</td>
</tr>
</tbody>
</table>

UpdateConflict parameters

The [UpdateConflict] parameters describe how the High-speed Apply Engine handles conflicts that occur for UPDATE statements in the apply request. You can specify as many codes and actions as required for the types of conflicts that might occur. The values that you specify in the [UpdateConflict] section override the values specified in the [AnyConflict] section (for UPDATE statements).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>“Action” on page 123</td>
</tr>
<tr>
<td>Code</td>
<td>“Code” on page 124</td>
</tr>
</tbody>
</table>
The following figure shows an example of the [UpdateConflict] parameters.

**Figure 55: Example of [UpdateConflict] parameter values**

<table>
<thead>
<tr>
<th>[UpdateConflict]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code=Positive</td>
</tr>
<tr>
<td>Action=Rollback</td>
</tr>
</tbody>
</table>

For more information about the parameters that you can specify in the conflict sections, see “AnyConflict parameters” on page 122. For a list of values for the Code parameter, see “Codes for conflict resolution (SQL codes)” on page 79. For a list of values for the Action parameter, see “Actions for conflict resolution” on page 82.
Logical log input

This appendix provides more information about logical log input considerations.

For more information, see the chapter on logical log files in the Log Master for DB2 Reference Manual.

Logical log input considerations

The High-speed Apply Engine can read logical log input produced by either Log Master for DB2 or an application program that conforms to the published format.

Be aware of the following points regarding High-speed Apply Engine and logical log input:

- High-speed Apply Engine reads logical log input only for apply requests for DB2 on mainframe targets.

- Logical log input consists of the following files:

  - The logical log control file contains information about the format and content of the logical log.

  - The logical log data file contains actual change information (updates, deletes, inserts, or DDL objects) in logical log format.

  - *(optional)* The large volume VSAM files contain data from DB2 columns that hold large volumes of data (such as XML columns or LOB columns). The product or program that generates the logical log must generate these files to enable High-speed Apply Engine to apply data to large volume columns.

  - *(optional)* The logical log XMLSTRING control file contains the string IDs and string data that DB2 uses to encode the data in XML columns. High-speed Apply Engine needs this file when the logical log includes XML data and the DB2 subsystem of the target tables is different than the subsystem where the logical log was generated. The product or program that generates the logical log must obtain this information from the DB2 catalog table SYSIBM.SYSSXMLSESSIONS. To apply the changes to a different target
subsystem, High-speed Apply Engine uses the string IDs and data to serialize your XML data correctly.

- The published format for logical log input does not define an order for the records in either the logical log control file or the logical log data file.

- High-speed Apply Engine does not use all of the records in logical log input. It reads the following records in either the logical log control file or the logical log data file:
  - CLUS LOB/XML Cluster Data Set Record
  - DCCL Clustering Index Column Description Record
  - DCDX Clustering Index Description Record (deprecated)
  - DCL2 DB2 Column Description Record Part Two
  - DCLI DB2 Column Information Record
  - DCOL DB2 Column Description Record (deprecated)
  - DCXI Clustering Index Information Record
  - DDSM Data Sharing Member Record
  - DLCI Logical Log Column Information Record
  - DLCO Logical Log Column Description Record (deprecated)
  - DLDS Logical Log Data Set Record
  - DLGE Log Information End Record
  - DLGI Log Information Record
  - DLOB LOB Column Description Record
  - DSS1 SysStrings Base Record
  - DSS2 SysStrings TransTable Record
  - DTBC Table Record Count Record
  - DTBI Table Information Record
  - DTBL Table Description Record (deprecated)
  - DTBR Table RBA Record (deprecated)
— DTS2 Table Space Description Record Part Two (deprecated)
— DTSI Table Space Information Record
— DTSP Table Space Description Record (deprecated)
— DXML XML Column Description record
— DXSF XMLSTRING Data Set Name Record
— LLDF Data Change Record (in logical log data file)
— XPTH Current Path Record
— XTYP Type Record
Installing High-speed Apply Engine on mainframe targets

This section describes the installation and configuration of High-speed Apply Engine for versions that run against IBM DB2 on mainframe operating systems.

Installation overview and prerequisites

The High-speed Apply Engine is a component of the BMC Log Master for DB2 product. As a component, High-speed Apply Engine is also used by the BMC LOADPLUS for DB2 product.

High-speed Apply Engine is automatically installed when you install the following solutions and products from BMC:

- Log Master for DB2
- Recovery Management for DB2
- LOADPLUS for DB2
- Database Administration for DB2

You can also install High-speed Apply separately by selecting it from the list in the Installation System. The High-speed Apply Engine honors passwords for Log Master, the Recovery Management for DB2 solution, or itself.

**Note**

If installed separately, High-speed Apply cannot process logical log input that contains data definition language information called DDL objects. To activate this capability, contact BMC Customer Support.

To install High-speed Apply Engine to run against DB2 on mainframe operating systems, use the BMC Software Installation System. This program is an ISPF application that provides a common interface for installing BMC Software products.
for DB2. Before installing and using the High-speed Apply Engine, review the constraints and recommendations in “Considerations for DB2 on mainframe targets” on page 24.

To install and run the High-speed Apply Engine successfully, your software environment must meet the requirements shown in the following table.

**Table 18: High-speed Apply Engine installation requirements for mainframe targets**

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>The DB2 relational database system from IBM, Version 10.1 NFM, or later</td>
</tr>
<tr>
<td>Operating system</td>
<td>An IBM operating system that supports your release of DB2 (such as z/OS, or successors)</td>
</tr>
<tr>
<td>Related products (one or both)</td>
<td>■ A supported version of the BMC Software Log Master for DB2 product</td>
</tr>
<tr>
<td></td>
<td>■ A supported version of the BMC Software LOADPLUS for DB2 product to perform an SQLAPPLY load action</td>
</tr>
</tbody>
</table>

**DB2 authorizations for plans, packages, and collections**

The High-speed Apply Engine creates plans, packages, and collections. Depending on the privileges that you are willing to grant to the user ID that runs High-speed Apply Engine, you can grant the DB2 authorizations and privileges for these activities using one of the methods described in this section.

**Note**

The user ID that runs the High-speed Apply Engine must have the authorizations described in “DB2 authorizations” on page 26.

The following table defines the variables that appear in all of the GRANT examples in this section.

**Table 19: Variables used in DB2 authorization examples**

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>aptPlan</td>
<td>name of High-speed Apply Engine plan that is specified during installation</td>
</tr>
<tr>
<td></td>
<td>For more information, see “PlanName” on page 266.</td>
</tr>
<tr>
<td>Variable name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| collectionIDs | names of collections to which High-speed Apply Engine dynamically binds packages during processing  
This name can be:  
- one specific package designated for use by High-speed Apply Engine  
- list of specific packages designated for use by High-speed Apply Engine  
- "*"  
This variable represents all collections. Your security policies might not permit this specification. |
| databaseName | target database being changed by the apply request |
| tableNames | target tables being changed by the apply request |
| userid01 | authorization ID of the user running the apply request  
You can specify PUBLIC or a specific authorization ID. |
| userid02 | authorization ID (different than userid01) with authority to bind plans, bind packages, and administer collections  
This authorization ID can be a secondary authorization ID. The privileges that are granted to this authorization ID vary, depending on how you enable High-speed Apply Engine bind processing. |
| userPlan01 | name of a pre-bound plan that is bound by using special bind options (optional, when BindAction parameter is Use)  
For more information, see “PlanName” on page 136. |

**Using the user ID running High-speed Apply Engine**

With this method, you must grant authority and privileges to the user ID running the High-speed Apply Engine.

Table 20 on page 294 contains a row that summarizes the requirements for this method. This method has the following requirements:
The user ID that runs High-speed Apply Engine (userid01) must have BINDADD authority, and one of the following statuses:

- PACKADM authority
- CREATE privileges on all packages (*)
- CREATE privileges on a specific collection or list of collections designated for use by High-speed Apply Engine

If userid01 has CREATE privileges only on specific collections, the apply request must specify one of those collection names as the value of the CollectionID parameter (see “CollectionID” on page 130).

**Authorization examples for the user ID running High-speed Apply Engine**

The examples in this section show the grant actions that are normally done during and after installation. The authorizations that you grant depend on your security policies.

For definitions of the variables shown in these examples, see “DB2 authorizations for plans, packages, and collections” on page 286.

The following example shows the authorizations that provide access to the High-speed Apply Engine plan and restart table. These authorizations are normally granted during installation.

```sql
GRANT EXECUTE ON PLAN aptPlan TO userid01;
GRANT EXECUTE ON PACKAGE BMCAPT.APTREB2 TO userid01;
```

The following example shows additional authorizations that are required to run High-speed Apply Engine. These authorizations are normally granted after installation.

```sql
GRANT INSERT, UPDATE, SELECT, DELETE ON tableNames TO userid01;
GRANT BINDADD TO userid01;
GRANT PACKADM ON COLLECTION collectionIDs TO userid01;
or
GRANT CREATE ON COLLECTION collectionIDs TO userid01;
```

**Using BindOwner and a pre-bound plan**

With this method, High-speed Apply Engine uses a pre-bound plan that was created under the authority of a different user ID.
The pre-bound plan is validated at run time; therefore, it must have been previously bound by a different user ID with appropriate privileges. For a sample BIND command, see “BindAction” on page 126.

This method has the following requirements:

- The user ID that runs High-speed Apply Engine (userid01) must have
  - EXECUTE privilege on a specific pre-bound plan
  - BINDAGENT authority
- To be validated at run time, the plan must have been previously bound by a different user ID (userid02) with appropriate privileges.
  For a sample BIND command, see “BindAction” on page 126.
- userid02 must have BINDADD authority, and one of the following statuses:
  - PACKADM authority
  - CREATE privileges on all packages (*)
  - CREATE privileges on a specific collection or list of collections that is designated for use by High-speed Apply Engine
- The apply request must specify the following parameter values:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>BindAction</td>
<td>Use</td>
<td>“BindAction” on page 126</td>
</tr>
<tr>
<td>BindOwner</td>
<td>user ID that bound the plan</td>
<td>“BindOwner” on page 129</td>
</tr>
<tr>
<td>CollectionID</td>
<td>name of the High-speed Apply Engine collection</td>
<td>“CollectionID” on page 130</td>
</tr>
<tr>
<td>PlanName</td>
<td>name of the specific prebound plan</td>
<td>“PlanName” on page 136</td>
</tr>
</tbody>
</table>

**Authorization examples for using a pre-bound plan**

The examples in this section show the authorizations that are normally granted during and after installation. The authorizations that you grant depend on your own security policies.

For definitions of the variables shown in these examples, see “DB2 authorizations for plans, packages, and collections” on page 286.
The following example shows the authorizations that provide access to the High-speed Apply Engine plan and restart table. These authorizations are normally granted during installation:

```
GRANT EXECUTE ON PLAN aptPlan TO userid01;
GRANT EXECUTE ON PACKAGE BMCAPT.APTREB2 TO userid01;
```

The following example shows additional authorizations that are required to run High-speed Apply Engine. These authorizations are normally granted after installation:

```
GRANT INSERT, UPDATE, SELECT, DELETE ON tableNames TO userid01;
GRANT EXECUTE ON PLAN userPlan01 TO userid01;
GRANT BINDAGENT TO userid01;
GRANT BINDADD TO userid02;
GRANT PACKADM ON COLLECTION collectionIDs TO userid02;
or
GRANT CREATE ON COLLECTION collectionIDs TO userid02;
```

**Using a pre-bound plan and package**

With this method, High-speed Apply Engine uses a pre-bound plan and package.

For more information, see “BindAction” on page 126.

This method has the following requirements:

- The user ID that runs High-speed Apply Engine (userid01) must have
  - EXECUTE privilege on the pre-bound plan
  - INSERT, UPDATE, and/or DELETE authority on the tables
- The apply request must specify the following parameter values:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>BindAction</td>
<td>Use</td>
<td>“BindAction” on page 126</td>
</tr>
<tr>
<td>PackageName</td>
<td>name of the specific package</td>
<td>“PackageName” on page 135</td>
</tr>
<tr>
<td>PlanName</td>
<td>name of the specific prebound plan</td>
<td>“PlanName” on page 136</td>
</tr>
</tbody>
</table>
The generation and the binding of the plan and package is done by a standalone High-speed Apply Engine job. The configuration parameters to complete this process are as follows:

```
[Agent]
MaxPrepares=50

[Bind]
PlanName=MyPlan
PackageName=MaxPre50
FreeOption=None
```

- **MaxPrepares** ("MaxPrepares" on page 120), which defaults to 50, is an optional parameter.
- **PlanName**, which defaults to a generated plan name based on value of PlanPrefix ("PlanPrefix" on page 137) and the apply request ID, is an optional parameter.
- **PackageName**, which defaults to generated package name based on the value of PackagePrefix ("PackagePrefix" on page 136) and 0000, is an optional parameter.
- You must set FreeOption to None; otherwise, the plan and package will be dropped during termination of the High-speed Apply Engine job.
- You should define the SQLIN DD with no data.

To use the pre-bound plan and package, you must set the following configuration parameters as shown:

```
[Bind]
PlanName=MyPlan
PackageName=MyPackage
BindAction=Use
```

- You must set **PlanName** to the pre-bound plan name. You can find the plan name in the bind output displayed by High-speed Apply Engine from the bind job in message BMCAPT0032I.
- You must set **PackageName** to the pre-bound base application package name. You can find the package name in the bind output displayed by High-speed Apply Engine from the bind job in message BMCAPT0032I.
- You must set **BindAction** to Use, which indicates that the application plan and package has been bound and should be used to apply the statements. If BindAction=Use is not set, a bind is attempted.
Using the AuthId parameter

With this method, High-speed Apply Engine binds by using the authority of a specified user ID. High-speed Apply Engine uses this user ID only for bind processing.

This method has the following requirements:

- The user ID that runs High-speed Apply Engine (userid01) must have EXECUTE privilege for the High-speed Apply Engine plan and restart table package. This user ID does not require special privileges for bind actions.

- The user ID that you specify for bind processing (userid02) can be a primary or secondary authorization ID, and
  - must have SYSADM authority or SYSCTRL authority
  - must be a valid TSO logon ID; otherwise, your security software can issue warning messages or prevent required processing
  - cannot be a group ID

- The apply request must specify userid02 as the value of the AuthId configuration parameter ("AuthId" on page 125).

Authorization examples for using the AuthID parameter

The examples in this section show the authorizations that are normally granted during and after installation.

The authorizations that you grant depend on your own security policies. For definitions of the variables shown in these examples, see “DB2 authorizations for plans, packages, and collections” on page 286.

The following example shows the authorizations that provide access to the High-speed Apply Engine plan and restart table package. These authorizations are normally granted during installation.

```
GRANT EXECUTE ON PLAN aptPlan TO userid01;
GRANT EXECUTE ON PACKAGE BMCAPT.APTREB2 TO userid01;
```

The following shows additional authorizations that are required to run High-speed Apply Engine. These authorizations are normally granted after installation.

```
GRANT INSERT, UPDATE, SELECT, DELETE ON tableName TO userid01;
GRANT SYSADM TO userid02;
```
Summary of DB2 authorization requirements

The following table summarizes the DB2 authorization requirements for different methods of specifying the [Bind] parameters to run the High-speed Apply Engine.

Note the following authorization considerations:

- Though any of the listed DB2 authorizations or privileges can be granted to PUBLIC, many of them normally are not; for example, SYSADM, SYSCTRL, BINDADD, and PACKADM.

- The BindOwner value must be one of the following:

  — a valid primary or secondary authorization ID of the user running High-speed Apply Engine

  — an authorization ID (with sufficient authority) that has granted BINDAGENT authority to the user running High-speed Apply Engine

- The AuthID value

  — must be a valid TSO logon ID, not a group ID

  — does not have to be a valid secondary authorization ID of the user running High-speed Apply Engine
### Table 20: Summary of DB2 authorization requirements for High-speed Apply Engine

<table>
<thead>
<tr>
<th>[Bind] parameter usage method</th>
<th>DB2 authorization</th>
<th>Granted to one of listed IDs or to PUBLIC</th>
</tr>
</thead>
</table>
| Default [Bind] parameters (if you do not specify any parameters in your configuration) | EXECUTE privilege for High-speed Apply Engine plan (for example, APTBvvr) | ■ primary authorization ID (user ID)  
■ secondary authorization ID  
EXECUTE privilege for restart table package (for example, APTBvvr.APTREB2) | BINDADD authority  
PACKADM authority or CREATE IN privilege for collection  
SELECT, INSERT, UPDATE, and DELETE privileges on target tables | primary authorization ID (user ID) |
| Specify value for BindOwner (APOWNER) parameter | EXECUTE privilege for High-speed Apply Engine plan (for example, APTBvvr) | ■ primary authorization ID (user ID)  
■ secondary authorization ID  
EXECUTE privilege for restart table package (for example, APTBvvr.APTREB2) | BINDADD authority  
PACKADM authority or CREATE IN privilege for collection  
SELECT, INSERT, UPDATE, and DELETE privileges on target tables | authorization ID specified by BindOwner parameter |
Installation options overview

During installation, you specify several options for the High-speed Apply Engine. The Installation System allows you to customize the default values for your site and saves them in the Installation System profile variables. This section describes installation options and their default values.

For quick reference, the following table presents the options in alphabetical order. The variable \( vvr \) shown in the Default value column represents the 3-digit version and release number for High-speed Apply Engine. For example, for version 11.2.00, the default value for Creator Name is APT112.
### Table 21: High-speed Apply Engine installation options for mainframe targets

<table>
<thead>
<tr>
<th>High-speed Apply Engine installation option</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creator Name</td>
<td>APTvvr</td>
<td>qualifier for all tables and indexes that High-speed Apply Engine creates <em>during installation</em>. Ensure that the Creator Name is the same as the Plan owner. High-speed Apply Engine uses the Plan owner as the Creator of the default restart table.</td>
</tr>
<tr>
<td>Database Name</td>
<td>APTvvr</td>
<td>name of a database that will be created during installation. The database contains the DB2 objects that High-speed Apply Engine creates during installation. Do not specify an existing database.</td>
</tr>
<tr>
<td>Plan Name</td>
<td>APTBvvr</td>
<td>name of the plan that High-speed Apply Engine uses to access the temporary and restart tables that it creates during apply processing. This option is equivalent to the PlanName parameter in the [Startup] section of the configuration file. You can specify any valid DB2 plan name. The default value is APTBvvr which represents the version and release level of High-speed Apply Engine. You must specify PlanName (PLAN) in the [Startup] section of the installation options. Or, if you are not using a installation options module, you must specify the PlanName (PLAN) in the [Startup] section of the configuration.</td>
</tr>
<tr>
<td>Bind Qualifier</td>
<td>APTvvr</td>
<td>qualifier for the internal plans and packages that High-speed Apply Engine binds (<em>not</em> the qualifier for packages bound <em>during execution</em>). This option is equivalent to the BindQualifier parameter in the [Startup] section of the configuration file.</td>
</tr>
<tr>
<td>Public Plan</td>
<td>Yes</td>
<td>determines whether High-speed Apply Engine grants the authority to run the product (execute against the High-speed Apply base plan) to PUBLIC.</td>
</tr>
<tr>
<td>Restart Table Space Name</td>
<td>APTRSCLS</td>
<td>DB2 table space name for the default restart table</td>
</tr>
<tr>
<td>Restart Table Name</td>
<td>APTRSCLT</td>
<td>table name for the default restart table</td>
</tr>
<tr>
<td>STOGROUP</td>
<td>SYSDEFLT</td>
<td>type of DFSMS storage allocation to be used for the table spaces and index spaces that contain the restart tables. Specify an existing storage group.</td>
</tr>
</tbody>
</table>
Generating installation options

When you install the High-speed Apply Engine using the BMC Software Installation System, the installation process generates a customized installation data set. This data set contains all jobs required to install High-speed Apply Engine into your specific DB2 environment.

One of these jobs establishes the installation option values (also known as DOPTs). The $C30DOPT member of the installation data set contains a job that runs the assembler with an options macro that establishes the installation processing values for High-speed Apply Engine. The format of the installation options specification is the same as for an High-speed Apply Engine configuration file. (For more information about the High-speed Apply Engine configuration file format, see “Configuration file format” on page 93.) The PlanName (PLAN) parameter in the [Startup] section is a required value. You can optionally tailor other default configuration information as needed.

When you submit the generated job containing the installation options, it assembles and links this option macro call to create an installation options module in the APF authorized library that you designated during installation. The default name for this module is APT$OPTS. When High-speed Apply Engine runs, it must be able to locate and read an installation options module (either the default module or a module that you have edited, assembled and linked). If it cannot read an installation options module, High-speed Apply Engine displays an error message and terminates processing.

If you modify any of the values in the installation options after High-speed Apply Engine has been installed, you must rerun the installation job for these changes to take effect. If you have previously installed High-speed Apply Engine and want to keep the same installation option values, you can

- gather your existing option values before you start to install and then enter the same values as you run the Install System

- use the Options Migration feature of the Installation System (for more information, see the the Installation System documentation)
Installing High-speed Apply Engine on UNIX or Windows

This appendix describes the requirements and procedures for installing the High-speed Apply Engine product on a UNIX or a Microsoft Windows system. This information applies to IBM DB2 LUW and Oracle targets.

Overview and prerequisites

To install and run High-speed Apply Engine, your software environment must meet the requirements shown in the following table.

Note
For the most recent information about installing High-speed Apply Engine, see the High-speed Apply Engine release notes and the readme file (on the software CD, in the root directory). The release notes and readme contain the most current updates to the installation information.

Table 22: High-speed Apply Engine installation requirements for Unix and Windows

<table>
<thead>
<tr>
<th>Item</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system</td>
<td>For a Windows environment, either:</td>
</tr>
<tr>
<td></td>
<td>■ Windows 2007</td>
</tr>
<tr>
<td></td>
<td>■ Windows 2008</td>
</tr>
<tr>
<td></td>
<td>For a UNIX environment, either:</td>
</tr>
<tr>
<td></td>
<td>■ IBM AIX, Version 6.1 and 7.1</td>
</tr>
<tr>
<td></td>
<td>■ Oracle Solaris, version 10 or later</td>
</tr>
</tbody>
</table>
### Database

One of the following versions of Oracle server and client:
- 11.1
- 11.2

One of the following versions of DB2 LUW server and client, running on IBM AIX or Oracle Solaris:
- 9.7
- 9.8
- 10

### Storage

100 MB of available storage for each installed copy of High-speed Apply Engine

### Installation considerations

When you install the High-speed Apply Engine on a UNIX or Microsoft Windows system, consider the following constraints and recommendations:

- On Windows you must have, and be logged in with, Administrator privileges. On UNIX you must have permission to update the `/etc` and `/opt` directories.

- To run the Windows installer, you must have Oracle Java version 5 or later installed.

- High-speed Apply Engine must be installed on the same server or client system as the target database server or client.

- The target database must have enough threads available to process the apply request. Ensure that at least two threads are available, or one thread more than the maximum number of agents for the request.

- The restart table must reside in the same database as the target tables.

- The user ID that submits the apply request must have the appropriate authorizations to run the request. For more information, see “Controlling access” on page 301.

- Before you begin the High-speed Apply Engine installation, BMC recommends the following actions:
If you have a previous version of High-speed Apply Engine installed on your system, complete the following procedures before installing the current version:

— Uninstall (remove) the previous version (see “Uninstalling High-speed Apply Engine” on page 312)

— Drop the existing restart table (see “Dropping an existing restart table” on page 310)

High-speed Apply Engine does not migrate restart tables from previous versions and does not drop existing restart tables.

— Review the High-speed Apply Engine readme file, which contains the most recent changes and additions to the documentation, including installation procedures. The file is located on the CD with the software, in the root directory. The file name begins with readme and is available in ASCII text and HTML format.

**Controlling access**

To run the High-speed Apply Engine, you must have a valid DB2 LUW or Oracle user ID and a password with the following privileges:

- appropriate table privileges (INSERT, UPDATE, and DELETE) on the target tables that are changed by the apply request
  
  The privileges that High-speed Apply Engine requires depend on the transactions that an apply request performs on the target tables.

- INSERT, UPDATE, SELECT and DELETE privileges on the restart table that an apply request uses

**Installing High-speed Apply Engine**

This topic explains how to use the installation wizard to:

- Install High-speed Apply Engine for the first time
- Replace an existing version of High-speed Apply Engine
- Create a default restart table, and bind for DB2 LUW
- Verify the installation
View the readme file

View the log

The installation wizard copies the program files to the installation directory that you specify, creates the subdirectories shown in the following table, and registers High-speed Apply Engine and its components by updating the registry on Windows and creating `/etc/bmc.profile` and updating `/etc/profile` on UNIX platforms.

### Table 23: Installation directory contents

<table>
<thead>
<tr>
<th>Subdirectory</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\InstallationFolder\highspeedapply</code></td>
<td>Windows program files</td>
</tr>
<tr>
<td><code>/InstallationDirectory/highspeedapply/bin</code></td>
<td>AIX or Solaris executables and shared libraries</td>
</tr>
</tbody>
</table>
| `\High-speed Apply\doc` | - readme file (in text and HTML formats)
  This file contains the most recent changes and additions to the documentation, including installation procedures.
| `\High-speed Apply\examples\db2udb`  
(For DB2 LUW targets) | - Sample configuration files |
| `\High-speed Apply\examples\oracle`  
(For Oracle targets) | - Sample SQL to create restart tables (and bind files for DB2 LUW) |

### To start the installation wizard

1. Insert the CD in the CD-ROM drive of the computer where you want to install High-speed Apply Engine.

   The installation wizard begins and displays the Welcome window.
Note

If the installation wizard does not begin automatically, complete the appropriate steps:

- For Windows:
  1. In Windows Explorer, open the **Windows** directory on the CD.
  2. Double-click **setup.cmd** (the Windows Command Script).
  3. If prompted, click **Run**.

- For UNIX:
  1. Go to one of the appropriate directories.
     a. For Solaris: **cdrom/unix/solaris**
     b. For AIX: **cdrom/unix/aix**

Figure 56: High-speed Apply Engine Installer Welcome window

2. Click **Next**.

The installation wizard displays the License Agreement.
3 Read the license agreement, click I agree to the terms of the license agreement, and click Next.

The installation wizard displays the Directory Selection dialog box.

4 Accept the displayed directory or click Browse to select a directory, and then click Next.

The installation preview window shows the destination directory, the features to be installed, and the size of the files.

5 Click Install.

6 In The Update PATH Environment Variable dialog box, select Yes to insert the High-speed Apply Engine directory at the beginning of the system-level PATH environment variable, and then click Next.

The installation wizard displays the Select Options dialog box.

7 Continue with the steps in the following topic.
Note
If you are installing the product on a UNIX server (Solaris or AIX), the Installer will create or update file `/etc/bmc.profile` and add the environment variable `HIGHSPEEDAPPLY_HOME` to point to the directory where the product is installed. This environment variable is used by the product's `apeenv` and the Setup/Verify (`SV.sh`) scripts found in the examples subdirectories. It will also update the `/etc/profile` to `source bmc.profile`. If you choose to set the PATH variables during installation, the PATH and `LD_LIBRARY_PATH` (`LDLIBPATH` in AIX) will also be updated and set in `bmc.profile`. The `/etc/bmc.profile` is also used by other BMC products that are installed on the server.

To create a restart table (Setup Product)

Use this procedure to create the default restart table *during* the installation process.

Note
For information about creating the restart table *after* installation, see “Creating a default restart table after installation” on page 311.

1 On the Select Options dialog box, select Setup Product, and click Next.

2 In the Database Type selection window, select DB2 or Oracle as the database type, and then click Next.

3 Enter information about the specific DB2 LUW database or Oracle instance to which High-speed Apply Engine will apply changes, and then click Next.

The user ID and password are not required if:

- You use Oracle External Credentials (OPSS) in your environment
- You use DB2 LUW and you are running on the server where the database resides

Tip
If necessary, obtain this information from database administration personnel at your site.

The installation wizard creates the restart table, and displays the Setup Output window.

4 Examine the messages to verify that the restart table was created successfully, and then click Next.
Note
If the restart table was not created successfully, minimize the installation wizard and resolve the error. Ensure that you have the correct privileges and connectivity for the DB2 LUW database or Oracle instance. For more information, see . If you cannot resolve the error, contact BMC Software Customer Support.

The installation wizard displays the ReadMe File Selection dialog box.

5 Select Yes, and then click Next.

The installation wizard opens the readme file (with Notepad as the text editor), and displays the Installation Summary window. If you are installing on a UNIX server, the readme file is displayed in the Installer's dialog.

Tip
You can go back to previous screens to run the Verify after the Restart table is created and before exiting the Installer (for example, before selecting Next to go to installation summary screen). In addition, on AIX and Solaris, you can run the Setup/Verify step on the command line outside the Installer by going to the desired DBMS subdirectory (db2usdb or oracle) under the examples directory and invoking SV.sh script, as follows:

Usage: SV.sh SETUP/VERIFY db_instance [user_ID] [user_password]

6 Click Done to close the installation wizard.

Additional procedures

Complete the tasks in this section, if necessary.

Verifying High-speed Apply Engine installation

Use the procedure in this section to verify installation of the High-speed Apply Engine.

To verify installation

1 Complete Step 1 through Step 3 in “To start the installation wizard” on page 302.

The installation wizard displays the Installation Preview window, showing no features to be installed.
2 Click **Install**.

The installation wizard displays the Select Options dialog box.

3 Select **Verify Product**, and then click **Next**.

The installation wizard displays the Database Type selection window.

4 Select **DB2** or **Oracle** as the database type, and then click **Next**.

5 Enter information about the specific DB2 LUW database or Oracle instance to which High-speed Apply Engine will apply changes.

**Tip**

If necessary, obtain this information from database administration personnel at your site.

The user ID and password are not required if:

- You use Oracle External Credentials (OPS$) in your environment
- You use DB2 LUW and you are running on the server where the database resides

6 Click **Next**.

The installation wizard displays the Verify Output window.

7 Examine the messages to verify that the installation completed successfully, and then click **Next**.

**Note**

If the installation did not complete successfully, minimize the installation wizard and resolve the error. Ensure that you have the correct privileges for and connectivity to the DB2 LUW database or Oracle instance that you entered in Step 5 on page 307. For more information, see “Viewing logs” on page 308. If you cannot resolve the error, contact BMC Customer Support.

The installation wizard displays the Installation Summary window.

8 Click **Done** to complete the installation and exit.
Viewing logs

Each time that you run the High-speed Apply Engine installer or uninstaller, a log file is created. A tab name reflects the date and time when the installer or uninstaller was run.

To view logs

1 From the Installation Summary or Uninstallation Summary window, click View log.

A tab for each installation log or uninstallation log is displayed.

**Note**
You can also use the High-speed Apply Maintenance Tool to view a log (see “To view logs with the High-speed Apply Engine Maintenance Tool” on page 308).

2 Click the tab for the log that you want to view.

3 Search for rows highlighted in red (errors) or yellow (warnings).

**Tip**
To sort columns, click the column header.
To reverse sort a column, press Shift while clicking the column header.

To view logs with the High-speed Apply Engine Maintenance Tool

1 Double-click High-speed ApplyMaintenanceTool.cmd in one of the following locations:
   - For Windows: installationDirectory\HighSpeedApply
   - For UNIX: installationDirectory/highspeedapply folder

2 Click the Logs tab.

3 To view a log, click Install Log or Uninstall Log, or to browse for a specific log file, click Browse to Log.

A tab for each installation log or uninstallation log is displayed.

Contents of the logs

The general format of all log messages is:
Example

(Feb 17 2011 01:10:28.123 AM -05:00),CONFIG,com.bmc.install.builder.installanywhere.
InstallationTaskAction,
LOG EVENT [Description=[CREATED InstallationTask],
Detail=[com.bmc.install.product.arsuitekit.ARSuiteKitProjectInitializationInstallationTask]]

This example indicates a LOG EVENT message that occurred on February 17, 2011, at 1:10:28 a.m. with a severity level of CONFIG from an InstallationTaskAction source class.

The severity levels from highest to lowest values are:

- SEVERE (appears in red in the log viewer)
- WARNING (appears in yellow in the log viewer)
- INFO
- CONFIG
- FINE
- FINER
- FINEST

The following table describes the main log message types.

Table 24: Log message types

<table>
<thead>
<tr>
<th>Message type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOG EVENT</td>
<td>Describes the following types of information:</td>
</tr>
<tr>
<td></td>
<td>- The start and end of installation tasks (installer logic and behavior)</td>
</tr>
<tr>
<td></td>
<td>and installation screens.</td>
</tr>
<tr>
<td></td>
<td>- The setting of installation properties. These properties can be set and</td>
</tr>
<tr>
<td></td>
<td>controlled by the platform itself, or they can be set based on user</td>
</tr>
<tr>
<td></td>
<td>input fields.</td>
</tr>
<tr>
<td></td>
<td>- General informative data</td>
</tr>
<tr>
<td>PROGRESS EVENT</td>
<td>Describes how much of a task is completed</td>
</tr>
<tr>
<td>THROWABLE EVENT</td>
<td>Describes failures with stack traces</td>
</tr>
</tbody>
</table>
### Verifying server status and response time (Health Check)

The Health Check function of the High-speed Apply Maintenance Tool calls an API to verify the server status, and records the server response time, as well as the results of the Health Check in the `highspeedapply_configuration_log.txt` file in the `temp` directory. This log file can be collected by Log Zipper.

**To run a Health Check**

1. In the High-speed Apply Maintenance Tool, click the **Health Check** tab.
2. Click **Next**.
3. Click **View Log** to view the `highspeedapply_configuration_log.txt` file in the `temp` directory, or click **Done** to end the Health Check.

### Dropping an existing restart table

Use this procedure to drop an existing restart table. If you have a previous version of High-speed Apply Engine installed on your system, you must drop the existing default restart table before creating a new one.

**To drop an existing restart table**

1. Open a command prompt.
2. Locate the appropriate script for your database platform from the directory `installationDirectory\HighSpeedApply\highspeedapply\examples\`:
   - For Windows: `installationDirectory\highspeedapply\examples`
   - For UNIX: `installationDirectory/highspeedapply/examples`
3. Edit the script to include the name of the existing restart table.
4. Run the script.

---

<table>
<thead>
<tr>
<th>Message type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary</td>
<td>Provides a summary at the end of the log that contains the same type of data shown on the installation summary screen (for example, whether the installation was successful or unsuccessful, and if unsuccessful, what features failed to install)</td>
</tr>
</tbody>
</table>
Creating a default restart table after installation

Use the procedure in this section to create a default restart table after installation by entering commands at a command prompt.

You can also run the installation wizard again to create the default restart table (“To start the installation wizard” on page 302). Completing this procedure is useful when you have multiple target databases and are using one High-speed Apply Engine installation.

To create a restart table by using a command prompt

1. Locate the appropriate sample file for your database platform from the directory:
   \installationDirectory\HighSpeedApply\highspeedapply\examples\n   - For Windows: \installationDirectory\highspeedapply\examples
   - For UNIX: \installationDirectory\highspeedapply\examples
   - oracle\CrtOraDefaultRestartTbl.sql
   - db2udb\CrtUDBDefaultRestartTbl.sql

   The following figure shows a sample DDL for a restart table.

   Figure 57: Sample DDL to create the default restart table on Oracle

   ```sql
   CREATE TABLE APERSCLT
   (   RESTART_ID         CHAR(16)     NOT NULL,
   REQUEST_ID         CHAR(4)      NOT NULL,
   RECORD_TYPE        CHAR         NOT NULL,
   AGENT_ID           SMALLINT     NOT NULL,
   RECORD_SEQUENCE    SMALLINT     NOT NULL,
   REQUEST_TIMESTAMP  DATE         NOT NULL,
   RECORD_TIMESTAMP   DATE         NOT NULL,
   TOTAL_LENGTH       INTEGER      NOT NULL,
   DATA               LONG VARCHAR NOT NULL
   )  ;
   CREATE UNIQUE INDEX APERSCLX ON APERSCLT
   (   RESTART_ID,
   RECORD_TYPE,
   AGENT_ID,
   RECORD_SEQUENCE
   )  ;
   CREATE PUBLIC SYNONYM BMCAPE_APERSCL FOR APERSCLT;
   GRANT SELECT, UPDATE, DELETE, INSERT ON APERSCLT TO PUBLIC;
   ```

2. Accept the default names and skip to the next step, or if necessary, edit the DDL as appropriate for your site.
**Note**

Do not edit the name of the public synonym. This synonym is the default value for the TableName parameter in the [Restart] section. If an apply request does not specify a restart table name, High-speed Apply Engine uses the table referenced by the public synonym.

3 Run the DDL file by issuing one of the following commands:

- For DB2 LUW, from the DB2CMD prompt:
  ```
  Db2-t-f CrtUDBDefaultRestartTbl.sql
  ```

- From a UNIX command prompt:
  ```
  sqlplus userid/password@CrtUDBRestartTbl tableName
  ```

- For Oracle, from SQL*Plus:
  ```
  @CrtOraDefaultRestartTbl tableName
  ```

**Note**

If you are not already running SQL*Plus, you must supply your user ID and password to start the utility from a UNIX command prompt:

```
sqlplus userid/password@CrtOraRestartTbl tableName
```

---

**Uninstalling High-speed Apply Engine**

Removing High-speed Apply Engine deletes all of the program files and Windows registry entries. However, it does not remove any DB2 LUW or Oracle objects that are associated with High-speed Apply Engine, such as the restart tables.

**To uninstall Windows**

1 From the Windows Start menu, select **Control Panel**.

2 Double-click **Add/Remove Programs**.

3 Highlight High-speed Apply Engine in the list of programs, and then click **Change/Remove**.

   The installation wizard starts the uninstaller.

4 Click **Next**.

   The installation wizard displays the Confirm File Deletion window.
5. Select **High-speed Apply Engine vrm Uninstallation**, and then click **Next**.

   The installation wizard displays the Uninstallation Preview window.

6. Click **Uninstall**.

   The installation wizard displays the Uninstallation Summary window.

7. Click **Done**.

8. Drop any High-speed Apply Engine restart tables from the DB2 LUW or Oracle database.

   For more information, see “Dropping an existing restart table” on page 310.

**To uninstall for UNIX**

1. Go to directory `installationDirectory/UninstallBMCHighSpeedApply`.

2. Run `uninstall.sh`. 
LOADPLUS and High-speed Apply Engine

This appendix summarizes the relationship between the LOADPLUS for DB2 product from BMC and the High-speed Apply Engine product (which is also distributed with the Log Master for DB2 product).

For more information about LOADPLUS, see the LOADPLUS for DB2 Reference Manual.

How LOADPLUS uses High-speed Apply Engine

LOADPLUS calls the High-speed Apply Engine to perform a load action that retains a DB2 object in read/write status and that does not replace data (LOAD RESUME YES SHRLEVEL CHANGE SQLAPPLY). This action is sometimes referred to as a SQLAPPLY load. When LOADPLUS calls High-speed Apply Engine, it has access to only the High-speed Apply Engine capabilities required for the load action.

LOADPLUS passes your load data to High-speed Apply Engine, which applies the load records to the DB2 object as if they were SQL INSERT statements. High-speed Apply Engine passes any error conditions that it encounters back to LOAD PLUS, which displays the error information in a LOADPLUS error message.

Requirements for using High-speed Apply Engine with LOADPLUS

To use the High-speed Apply Engine with LOADPLUS, ensure that

- LOADPLUS is installed correctly with a valid LOADPLUS password

To perform an SQLAPPLY load action, BMC Software does not require a license (or password) for either the Log Master for DB2 product or High-speed Apply Engine.
• High-speed Apply Engine is installed on the same DB2 subsystem as LOADPLUS and is available through the STEPLIB, JOBLIB, or LINKLIST when LOADPLUS is executed. (High-speed Apply Engine is installed automatically when you install LOADPLUS.)

LOADPLUS options and High-speed Apply Engine parameters

LOADPLUS provides several syntax options and passes the values of those options to the High-speed Apply Engine as configuration parameters.

For more information about the LOADPLUS options, see the LOADPLUS for DB2 Reference Manual.

Table 25: LOADPLUS options and corresponding High-speed Apply Engine parameters

<table>
<thead>
<tr>
<th>LOADPLUS syntax option</th>
<th>Corresponding High-speed Apply Engine configuration parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>APCOLLECTION</td>
<td>CollectionID: specifies a collection name that High-speed Apply Engine uses in any BIND PACKAGE or FREE PACKAGE commands that it issues to execute SQL statements (“CollectionID” on page 130).</td>
</tr>
<tr>
<td>APCOMMIT</td>
<td>StatementCount: specifies the number of SQL statements that an apply agent processes before that agent issues a COMMIT statement (StatementCount on page 152).</td>
</tr>
<tr>
<td>APMAXAGENTS</td>
<td>MaxAgents: defines the maximum number of agents that High-speed Apply Engine can start for an apply request (“MaxAgents” on page 119).</td>
</tr>
<tr>
<td>APMULTIROW</td>
<td>MultiRowInsert: enables or disables High-speed Apply Engine’s processing of multiple row insert actions. MaxRows: enables you to control how many rows High-speed Apply Engine includes in each multiple row insert action of an apply request.</td>
</tr>
<tr>
<td>APOWNER</td>
<td>BindOwner: specifies an authorization ID that High-speed Apply Engine can use to bind the DB2 plan and packages for the apply request (“BindOwner” on page 129).</td>
</tr>
<tr>
<td>APRETRYLIM</td>
<td>RetryLimit: indicates the measurement type used for the RetryValue parameter (“RetryLimit” on page 161).</td>
</tr>
<tr>
<td>APRETRYVALUE</td>
<td>RetryValue: specifies the retry limit for each unit of recovery (transaction) in the request (“RetryValue” on page 162).</td>
</tr>
</tbody>
</table>
Glossary

A

APF

authorized program facility. An IBM facility that allows you to establish authorization for programs to use certain functions that are otherwise restricted.

apply agent

a High-speed Apply Engine task that updates a target table

apply processing

the process of updating the target tables by using the information provided in the configuration file and apply request

apply request

the batch job or command that is used to start and run High-speed Apply Engine. The apply request can contain job processing information and overriding values for some of the configuration parameters.

C

configuration

information about the database, input files, and processing options used during apply processing. The configuration information can be specified in the configuration file or the apply request.
configuration file
the file that contains configuration information for an apply request

configuration manager
the part of High-speed Apply Engine that validates and processes the configuration information

conflict
the situation that occurs when High-speed Apply Engine attempts to update a target table, but the target database prevents the update. Conflicts can occur if the target table definition does not match the source table definition, if resources are unavailable, or if a time out occurs during apply processing.

conflict manager
the part of High-speed Apply Engine that receives conflict information from the apply agent, and processes conflicts according to the resolution rules defined in the configuration

conflict resolution rule
a set of configuration parameters that describes how High-speed Apply Engine responds to a particular type of conflict. Each rule consists of a conflict type (specified with the Code parameter) and one or more actions (specified with the Action parameter).

D
data migration
the process of moving data from one database environment (table space or instance) to another

DBCS
double-byte character set

DBRM
database request module (DB2 only). High-speed Apply Engine creates a DBRM that contains all unique SQL statements in an apply request. This DBRM facilitates static SQL processing for DB2 targets.

DDL

data definition language. A subset of SQL statements that define data objects, such as the CREATE statements.

DFSMS

Data Facility Storage Management System. An IBM storage management product that lets you define the characteristics of data sets and storage devices.

distributed system

For purposes of this document, a computer system other than a mainframe. A distributed system can be a minicomputer, a LAN server, or a personal computer. This document assumes that the operating system of a distributed system is UNIX or Windows.

DML

data manipulation language. A subset of SQL statements that manipulate data, such as the INSERT, UPDATE and DELETE statements.

GDG

generation data group

input file

the file that contains the updates for the target tables. This file can be a Log Master for DB2 logical log file or an SQL file.
J

JCL

Job Control Language

P

parameter

an element of an High-speed Apply Engine configuration file, also referred to as a configuration parameter. Parameters define processing information for an apply request.

PDS

partitioned data set

R

resolution rule

See conflict resolution rule on page 318.

restart processing

the processing that High-speed Apply Engine performs to start an apply request at an appropriate point after an interruption.

restart table

the table that High-speed Apply Engine uses to store restart information about apply requests. You must create one or more restart tables before High-speed Apply Engine can perform restart processing.
section name

the label that identifies a group of parameters. Section names are delimited by slash characters (/SectionName/) or square brackets ([SectionName]).

SID

system identifier (Oracle only). A unique name that identifies an Oracle instance.

source table

for data migration or transaction recovery, the table that originally contained the data that will be migrated or recovered.

SPUFI

SQL Processor Using File Input. The IBM utility that lets you execute SQL statements without embedding them in an application program.

SQL

Structured Query Language

SSID

subsystem identifier (DB2 only). A unique name that identifies a DB2 subsystem.

target table

any table that High-speed Apply Engine updates during an apply request

template file
a separate file generated by the Log Master for DB2 product that High-speed Apply Engine can use to efficiently process SQL input as static SQL against DB2 for mainframe or DB2 UDB targets. The template file contains one entry for each distinct statement type in the SQL input. If an apply request includes a template file, High-speed Apply Engine processes the template file before it begins processing SQL input.

**transaction recovery**

the process of recovering database transactions that have occurred since the last image copy was taken. This process can require undo processing, redo processing, or both.
Index

A

Abort action 81, 82
Action parameter
   in [AnyConflict] section 123
   in [DDLConflict] section 177
   in [DeleteConflict] section 178
   in [InsertConflict] section 192
   in [UpdateConflict] section 279
processing condition 81
valid values 82
actions, conflict resolution
   Abort 82
   Continue 82
   DeferStatement 82
   DeferUR 82
   Display 82
   Retry 82
   Rollback 82
   RollbackStatement 82
   Skip 82
   SkipUR 82
   Terminate 82
   Warn 82
[Agent] parameters
   APMAXAGENTS 316
   InitialAgents 117
   MaxAgents 119
   MaxPrepares 120
agents, distribution of work between. See distribution of work
[AnyConflict] parameters
   Action 123
   Code 124
APCOLLECTION parameter 316
APCOMMIT parameter 316
apebmain command 58, 60
APF
   definition 317
APF authorization 24
APMAXAGENTS parameter 316
APMULTIROW parameter 316
APOWNER parameter 293, 316
application recovery, overview 22
apply agent
   definition 317
apply processing
   definition 317
apply request
   definition 317
apply requests
   adding Restart parameters 70
   comments 98
   data flow 19
   distributing by partition 184
   distributing by RI settings 186, 187, 189, 197, 200
   file format 51
   including DDL 195, 202
   including LOBS in 195
   including XML in 199
   information needed 41
   request ID, generated 136, 137
   restarting 72
   running 56, 58, 60
   samples 56, 60
   specifying configuration parameters 93
APRETRYLIM parameter 316
APRETRYVALUE parameter 316
APTBMAIN command 56
APTDIAG DD statement 54
APTDUMP DD statement 54
APTERR DD statement 54
APTPRINT DD statement 54
APTREST sample DDL member 64
AuthID parameter Bind 292
AuthID parameter [Bind] 125, 149, 293
authorizations, DB2. See DB2 authorization
balancing the workload 31
batch jobs
  creating 56
  running apply requests 56
[Bind] parameters
  APOCOLLECTION 316
  APOWNER 293, 316
  AuthID 125, 292, 293
  BindAction 127, 288, 290
  BindOwner 129, 288, 290, 293
  CollectionID 130, 288, 290
  DBPROTOCOL 131
  DISP 132
  FreeOption 133
  Library 134
  PackageName 135
  PackagePrefix 136
  PlanName 136
  PlanPrefix 137
bind privileges
  DB 286
  DB2 LUW 28
BindAction parameter [Bind] 127, 288, 290
BINDADD authority, DB2 288, 290
BindOwner parameter [Bind] 129, 288, 290, 293
BindQualifier parameter [StartUp] 253
[BindTuning] parameters
  MaxFailedBinds 142
  MaxPackages 143
  MaxRetries 145
  PackageUsage 146
  StatementCount 147
  StatementUsage 148
  Synchronous 149
BMC Software, registry 58, 60
brackets in section names 96

CAF (call attach facility) 26
calculating partitions in logical log input 202
CCSID
  CCSIDCompatible parameter 194
  overriding translation 194
  CCSIDCompatible parameter [LogicalLog] 194

CFN parameter [Configuration] (synonym for FileNme) 155
changes to the product 14
checkpoint processing
  parameters for 152, 153, 240
  with restart 70, 240
CheckpointFrequency parameter [Restart] 240
Cluster parameter [ObjectCluster] 223
Code parameter
  in [AnyConflict] section 124
  in [DDLConflict] section 177
  in [DeleteConflict] section 178
  in [InsertConflict] section 192
  in [UpdateConflict] section 279
  interaction 81
  processing order 82
  valid values 79
coded character set identifier. See CCSID
codes, conflict resolution
  MultipleRows 79
  MultiRowConflict 79
  Negative 79
  nnnn (specific SQL code) 79
  NoRows 79
  Positive 79
  RIConflict 79
  SQLWarn 79
  TimeOut 79
CollectionID parameter [Bind] 130, 288, 290
column names, wildcard patterns 101
Column parameter [ObjectPartition] 234
command line, specifying configuration parameters 51
comments
  description 98
  in an apply request 98
  including in a configuration file 40
  supported characters 98
CommitOnDemand parameter [CommitTriggers] 150
CommitTrigger parameter [MonitorTuning] 213
[CommitTriggers] parameters
  APCOMMIT 316
  CommitOnDemand 150
  StatementCount 152
  Time 153
CommitValue parameter [MonitorTuning] 214
configuration
  definition 317
  configuration file
definition 318
configuration files
adding Restart parameters 70
CFN parameter [Configuration] 155
comments 40, 98
conflict resolution 77
creating 41
data set names 53
DB2 mainframe target 41
DD name 53
editing 41
examples 41, 64
format of 93
line length 93
multiple lines in 93
naming convention 40
Oracle target 41
quick reference by name 103
quick reference by section 109
saving 41
specifying in an apply request 41
tip for formatting 41
using multiple files 39
wrapping lines in 93
configuration manager
definition 318
configuration parameters, by name 103
APCOLLECTION 316
APCOMMIT 316
APMAXAGENTS 316
APMULTIROW 316
APOWNER 293, 316
APRETRYLIM 316
APRETRYVALUE 316
AuthID [Bind] 292, 293
BindAction [Bind] 288, 290
BindOwner [Bind] 288, 290, 293
CollectionID [Bind] 288, 290
See also individual parameter names
configuration parameters, by section 109
[Configuration] parameters, FileName (CFN) 155
configuration parameters, general
Action 82
Code 79
format 97
in command lines 51
in EXEC statements 51
multiple lines of 93
overview 39
quick reference by name 103
quick reference by section 109
section names 93
configuring conflict resolution 86
conflict
definition 318
conflict files, defining 86
Conflict log type 209
collision manager
definition 318
[Conflict] parameters
APRETRYLIM 316
APRETRYVALUE 316
DataType 157
defining 86
MaxFailedRetries 158
MaxRetryFail 159
RetryDelay 159
RetryFail 160
RetryLimit 161
RetryValue 162
conflict resolution
actions 82
configuring conflict resolution 86
overview 35
processing 82
rule format 77
rule interaction 81
SQL codes 79
conflict resolution rule
definition 318
conflict types
adding rules to the configuration 86
descriptions 79
conflict-resolution rules
defining 86
interaction 81
processing order 82
[ConflictFile] parameters

Index 325
DATACLAS 164
DISP 164
FileNameModel 165
LIKE 169
MAXVOL 170
MediaType 169
MGMTCLAS 171
RELEASE 172
SingleFile 172
SPACE 173
SpacePri 174
SpaceSec 174
STORCLAS 175
UNIT 176
VOLSER 176

ConnectionVersion parameter [StartUp] 253
continuation character in configuration file 93
Continue action 82
control file, logical log 281
controlling conflicts 86
conventions
  configuration file names 40
  naming batch jobs 55
conventions, documentation 13
converting data types 26, 31
creating a configuration file 41
creating a conflict file 86
creator name, DB2 installation option 295
CrtOraRestartTable.sql sample DDL 64

D
data file, logical log 281
data flow of an apply request 19
data migration 22
definition 318
data set names

APTDIAG 54
APTDUMP 54
APTERR 54
APTPRINT 54
DD names and 53
list of DD names 54
sort work data sets 27
SYSIN 54
SYSMDUMP 54
SYSOUT 54
SYSTYPEM 54
SYSUDUMP 54
See also file names
data type conversion 26, 31
data type support
  DB2 26
  DB2 LUW 29
  Oracle 31
database authorization
  DB2 286
  DB2 LUW 28
  Oracle 301
databases
  access considerations 26, 30
  database name, DB2 installation option 295
  supported versions 285, 299
  updates, by object 31
  updates, by unit of recovery (UR) 34
DATACLAS parameter
  in [ConflictFile] section 164
  in [TemporaryFiles] section 273
DataType parameter [Conflict] 157
DateFormat parameter [StartUp] 255
DB2
  authorization. See DB2 authorization
call attach facility 26
creating a configuration file 41
installation options 295
object name length 101
partition key length 47
SQLCA 79
supported versions 285
DB2 authorization
  bind privileges 286
  BINDADD authority 288, 290
methods of granting 286
PACKADM authority 286
table privileges 26
DB2 installation options
bind qualifier 295
creator name 295
database name 295
plan name 295
public plan 295
restart table name 295
restart table space name 295
STOGROUP 295

DB2 LUW
creating a configuration file 41
authorization 28
considerations for target databases 27
DELETE privilege 28
INSERT privilege 28
partition key lengths 47
template file with SQL 29
UPDATE privilege 28

DBCS
definition 318

DBMSType parameter [StartUp] 256

DBPROTOCOL parameter [Bind] 131

DBRM
definition 318

DD names
APTDIAG 54
APTDUMP 54
APTERR 54
APTPRINT 54
default for input file 259
list of valid 54
specifying in configuration files 53
SYSIN 54
SYSDUMP 54
SYSOUT 54
SYSTERM 54
SYSUOMP 54

DDL
definition 319

DDL (data definition language), in logical log input 195, 202

[DDLConflict] parameters
Action 177
Code 177

DecimalPoint parameter [SQL] 249

DefaultOpts parameter [StartUp] (DOPT) 257
defer file. See conflict files
DeferStatement action 82
DeferUR action 82
defining

conflict resolution actions 82
conflict resolution codes 79
logical partitions for source tables 47
object clusters 222
restart table 246

DELETE privilege 26, 31, 301

[DeleteConflict] parameters
Action 178
Code 178
delimiters in table names 100
DFN parameter [DisplayOutput] (synonym for FileName) 181

DFSMS
definition 319
DFSMS storage class
conflict file 175
temporary files 277
diagnostic information, display of output 179

[DiagnosticOutput] parameters, FileName 179

DIR parameter [TemporaryFiles] (synonym for Directory) 273

Directory parameter [TemporaryFiles] 273
disk requirements
Windows 299

DISP parameter
in [Bind] section 132
in [ConflictFile] section 164

Display action 82

Display log type 209
display output messages 181

[DisplayOutput] parameters, FileName 180
distributed system
definition 319
distribution of work between agents
by object 31
by UR (unit of recovery) 34, 258
overview 31
partitions 184, 230
RI (referential integrity) 186, 187, 189
table space, simple 189

[DistributionTuning] parameters
MaxExecuteImmediateDML 183
PartitionClustering 184
RIClustering 186
RIRecursionLimit 187
SimpleTSClustering 189
URCheckRI 189

DistributionType parameter [StartUp] 258

DML
definition 319

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
documentation
readme file 300
documentation information 14
DOPT parameter [StartUp] (synonym for DefaultOpts) 257
dynamic SQL
  balancing with static SQL 139
  fallback to 25, 143
  MaxPrepares parameter 120
  preparing 120
  problems with 183

E

ing a configuration file 41
EFN parameter [ErrorOutput] (synonym for FileName) 191
electronic documentation 13
[ErrorOutput] parameters, FileName (EFN) 191
eamples of DB2 authorizations 286
eamples subdirectory 41, 58, 60, 64
excluding changes from LOBs 195
excluding changes from RI 197
excluding changes from triggers 198
excluding changes from XML columns 199
EXEC statement
  APTBMAIN 56
  PARM parameter 51
  specifying configuration parameters 51
executable files, apebmain 58, 60
EXECUTE IMMEDIATE
  balancing with static SQL 143
  log type 209
  maximum for an apply request 183

F

de formats
  configuration files 93
  logical log input 21, 281
  SQL input 20
file management
  apply requests DB2 55
  configuration files 40
file names
  configuration files 53
  conventions 53
  defaults 99
  diagnostic output file 179
  display output file 181
  error file 191
  input file 259
  log file 208
  specifying in configuration files 99
FileName parameter
  in [Configuration] section (CFN) 155
  in [DiagnosticOutput] section 179
  in [DisplayOutput] section (DFN) 181
  in [ErrorOutput] section (EFN) 191
  in [LogOutput] section (LFN) 208
  in [StartUp] section (IFN) 259
FileNameModel parameter [ConflictFile] 165
fixes, verifying product maintenance 56
Format parameter [SQL] 250
formats
  comments 98
  configuration files 93
  configuration parameters 97
  conflict resolution rules 77
  data set names 99
  DB2 object names 99
  file names 99
  section names 96
  table names 99
FreeOption parameter [Bind] 133
functions, overview 17

G

generation package names 136
generated request ID 136, 137
granting DB2 authorizations 286
grouping target tables 31

H

Help
  online 13
High-speed Apply Maintenance Tool 308
HLQ.DBSAMP library 41, 56, 64
host_vars log type 209
IT parameter [StartUp] (synonym for InputType) 262

J

JCL
definition 320
JCL (job control language)
data set names in 99
procedure for running apply request 56
sample DB2 apply request 52
specifying configuration file 56

L

large volume columns
LOB columns 195
longer statement lengths 263
VSAM files for 281
XML columns 199
LargeStatementSupport parameter [StartUp] 263
LFN parameter [LogOutput] (synonym for FileName) 208
Library parameter [Bind] 134
Like parameter
in [ConflictFiles] section 169
in [TemporaryFiles] section 274
limit on EXECUTE IMMEDIATE statements 183
LimitKey parameter [ObjectPartition] 235
line length in configuration file 93
LLOGIN DD name 259
LOADPLUS for DB2 (BMC Software product) 315
LOBs (large objects)
including or excluding changes from 195
longer statement lengths 263
log entry types
Conflict 209
Display 209
Exec_immed 209
Host_vars 209
Work_item 209
log files
entry types 209
LFN parameter [LogOutput] 208
overwriting 210
Log Master for DB2
supported versions (mainframe targets) 285
with High-speed Apply 17
logical logs
format 281
input to High-speed Apply 17, 21
overview 21, 281
parameters for 193
partitions within 202
parts of 21, 281
records 281
SQL type 204
VSAM files, large volume columns 195, 199, 281
XMLSTRING control file 199, 281
logical partitions, creating 47
[LogicalLog parameters]
   including 41
Login parameter [StartUp] (Oracle only) 263
LogType parameter [LogOutput] 209

M
mainframe operating systems
   executable command (APTBMAIN) 56
   supported versions 285
MAINT parameter, verifying maintenance 56
   maintenance, verifying 56
managing apply requests, DB2 55
mapping
   source objects to target objects 44
      with wildcards 226
MaxAgents parameter [Agent] 119
MaxConflicts parameter [MultiRowInsert] 219
MaxExecuteImmediateDML parameter
   [DistributionTuning] 183
MaxFailedBinds parameter [BindTuning] 142
MaxPackages parameter [BindTuning]
   description 143
      forcing static SQL 147
MaxPrepares parameter [Agent] 120
MaxRetries parameter [BindTuning] 145
MaxRows parameter [MultiRowInsert] 220
MaxVol parameter [ConflictFile] 170
MediaType parameter [ConflictFiles] 169
   messages
      display of output 181
      progress messages 212
         verifying maintenance with 56
MGMTCLAS parameter
   in [ConflictFile] section 171
   in [TemporaryFiles] section 275
monitoring High-speed Apply 212
   progress messages, 212
   statistics messages, 212
   multiple line configuration parameters 93
   multiple row insert processing
      Action parameter 82
      Code parameter 79
      conflict resolution 79
      enabling and disabling 221
      MaxConflicts parameter 219
      MaxRows parameter 220
      MultiRowConflict code value 79
      MultiRowInsert parameter 221
      performance and 220
      Retry action value 82
      summary 221
MultipleRows code type 79, 82
MultiRowConflict code type 79
MultiRowInsert parameter [MultiRowInsert] 221
   [MultiRowInsert] parameters
      APMULTIROW 316

N
names of DB2 objects, length of 101
naming convention, configuration files 40
Negative code type 79, 81
NoRows code type 79, 82

O
object names
   length of 101
      mapping from source to target 44
   [ObjectCluster parameters]
      including 41
   [ObjectCluster] parameters
      including 41
   SourceTable parameter 228
   ST parameter 228
   [ObjectMap] parameters
      defining 44
      including 41
      troubleshooting 44
   [ObjectMap parameters]
      including 41
   [ObjectPartition] parameters
      defining 47
      including 41
      troubleshooting 47
objects, distributing work based on 31
OCI (Oracle Call Interface) 30
online Help 13
operating High-speed Apply, considerations 24, 27, 30
operating systems, supported
mainframe targets 285
Windows 299
Oracle
authorization 31
call interface 30
ConnectionVersion parameter [StartUp] 253
creating a configuration file 41
database authorizations 301
date/time formats 255, 269, 270
DELETE privilege 31, 301
INSERT privilege 31, 301
installation requirements 299
Login parameter [StartUp] 263
MaxPrepares restriction 30
partition key length 47
supported versions 299
table privileges 31
UPDATE privilege 31, 301
Order parameter [ObjectPartition] 235
output messages
diagnostic information 179
display of 181
Overlap parameter [LogOutput] 210
owner, table 99

P
PACKADM authority, DB2 130, 286
package name, generated by High-speed Apply 136
package, pre-bound 290
PackagePrefix parameter [Bind] 136
PackageUsage parameter [BindTuning] 146
parameter
definition 320
parameters, configuration. See configuration parameters
PARM parameter, specifying configuration parameters 51
PARM parameter, verifying maintenance 56
partition keys
DB2 47
Oracle 47
Partition parameter [ObjectPartition] 236
PartitionClustering parameter [DistributionTuning] 184
PartitionDistribution parameter [LogicalLog] 202
partitions
distributing the workload 183, 184
labeling 236
logical log input 202
specifying logical partitions for target tables 230
PDS
definition 320
performance tuning
bind performance (DB2) 139
parameters for statistics messages 212
plan name, DB2 installation option 295
PLAN parameter [StartUp] (synonym for PlanName) 266
plan, pre-bound 288, 290
PlanName parameter
in [Bind] section (DB2 only) 136
in [StartUp] section (DB2 only) 266
PlanPrefix parameter [Bind] (DB2 only) 137
Positive code type 79, 81
pre-bound package 290
pre-bound plan 288, 290
product changes 14
product requirements
DB2 24
DB2 LUW 27
Oracle 24
PTFs, verifying 56
publications, related 14
PublicPlan parameter [StartUp] 266

Q
qualified name, restart table 253
Qualify parameter [LogicalLog] 201
quick reference, configuration parameters
by name 103
by section 109

R
readme file 300
referential integrity (RI). See RI (Referential Integrity)
registry, BMC Software 58, 60
related publications 14
Release parameter [ConflictFile] 172
request ID, generated 136, 137
required parameters 41, 93
resolution rule
definition 320
resolving conflicts 86, 157
[Restart] parameters
  TableName 100
restart processing
definition 320
restart table
definition 320
  for DB2 mainframe targets 64
  for Oracle targets 64
  installation considerations 300
  name format 246
  synonym 246
  unqualified name for 253
restart table name, DB2 installation option 295
restart table space name, DB2 installation option 295
RestartID parameter [Restart] 242
restarting High-speed Apply
  adding configuration parameters 70
  apply requests 72
  checkpoint processing 240
  CheckpointFrequency parameter 240
  configuration parameters 239
  creating a restart table 64
  overview 35, 63
  restart table 246
  RestartID parameter 242
  RestartType parameter 243
  techniques for 74
RestartType parameter [Restart] 243
RetainTime parameter [Restart] 244
Retry action 82
Retry parameter [Restart] 245
RetryDelay parameter [Conflict] 159
RetryFail parameter [Conflict] 160
RetryLimit parameter [Conflict] 161
RetryValue parameter [Conflict] 162
RI (referential integrity)
  controlling scope of 187, 189
  including or excluding changes from 197
  RIClustering parameter [DistributionTuning] 186
  RIConflict code 79
  RIRecursionLimit parameter
    [DistributionTuning] 187
  URCheckRI parameter [DistributionTuning] 189
    with object clusters 222
RI (referential integrity) specifying in a configuration file 186
RIClustering parameter [DistributionTuning] 186
RIConflict code type 79
RIRecursionLimit parameter [DistributionTuning] 187
Rollback action 82
RollbackStatement action 82
running
  DB2 apply request 56
  DB2 LUW apply request 58
  High-speed Apply 23
  Oracle apply request 60

S
samples
  apply request for Oracle 60
  commands 60
  DB2 apply request 52
  examples subdirectory 41, 64
  HLQ.DBSAMP 41, 56, 64
  saving the configuration file 41
SC parameter [ObjectMap] (synonym for SourceColumn) 227
ScanInterval parameter [MonitorTuning] 217
section name
definition 321
section names
  brackets 96
delimiters 96
  in apply requests 96
SID
definition 321
SimpleTSClustering parameter
  [DistributionTuning] 189
SingleFile parameter [ConflictFile] 172
Skip action 82
SkipUR action 82
Sort parameter [LogicalLog] 203
ssing
Sort parameter [LogicalLog] 203
sort work data sets for mainframe targets 27
WORKNUM parameter [LogicalLog] 207
source file types 20
source table
definition 321
source tables, defining partitions 47
SourceColumn parameter [ObjectMap] 227
SourceTable parameter [ObjectMap] 41
SPACE parameter
in [ConflictFile] section 173
in [TemporaryFiles] section 275
SpacePri parameter
in [ConflictFile] section 174
in [TemporaryFiles] section 276
SpaceSec parameter
in [ConflictFile] section 174
in [TemporaryFiles] section 277
special characters, configuration file 96
splitting lines in configuration file 93
SPUFI
definition 321
SQL
batch processing 22
code types 79
data types 26, 31
definition 321
dynamic. See dynamic SQL
input file 17
specifying codes for conflict resolution 79
statements, as input 20
static. See static SQL
template file for input. See template file
SQL codes
order of conflict resolution 82
specifying for conflict resolution 81
SQL*Plus
called from High-speed Apply 30
including in path 30
SQLAPPLY load action 315
SQLCA 79
SQLIN DD name 259
SQLTerminator parameter [StartUp] 267
SQLType parameter [LogicalLog]
affects of 41, 202
description 204
SQLWarn code type 79, 82
SSID
definition 321
SSID parameter [StartUp] 41, 268
[StartUp] parameters
required section 93
StatementCount parameter
in [BindTuning] section 147
in [CommitTriggers] section 152
StatementUsage parameter [BindTuning] 148
static SQL
balancing with dynamic SQL 139
balancing with EXEC IMMEDIATE 143
efficiency of 20
forcing 147
overview 25, 138
performance of 139
processing 25
related parameters 120, 125, 143, 145, 149
template with 20
triggers for binding 146–148
STOGROUP, DB2 installation option 295
storage requirements
Windows 299
STORCLAS parameter
in [ConflictFile] section 175
in [TemporaryFiles] section 277
summary of changes 14
support, customer
verifying maintenance 56
symbolic variables (substitutions) 165
Synchronous parameter [BindTuning] 125, 149
synonym, for the restart table 246
SYSIN DD statement 54
SYSMDUMP DD statement 54
SYSOUT DD statement 54
SYSTERM DD statement 54
SYSUDUMP DD statement 54
T
table names
delimited 100
format 99
format for restart table 246
synonym, restart table 246
wildcard patterns 101
table owner 99
Table parameter
in [ObjectCluster] section 224
in [ObjectPartition] section 237
table privileges
TableName parameter [Restart]
description 246
limitation for restart table 100
tables
grouping targets 31
restart 64	
target table
definition 321
TargetColumn parameter [ObjectMap] 229
TargetTable parameter [ObjectMap] 41, 230
task summary 23
TC [ObjectMap] parameter (synonym for TargetColumn) 229
template file
definition 321
template file, for SQL input
overview 20
related parameters 139, 143, 147
under DB2 LUW 29
Terminate action 81, 82
Threshold parameter [LogOutput] 211
Time parameter [CommitTriggers] 153
TimeOut code type 79, 81
TimestampFormat parameter [StartUp] 269
TimestampTzFormat parameter [StartUp] 270	
transaction recovery 22
definition 322
triggers, including or excluding changes from 198
troubleshooting
ObjectMap section 44
ObjectPartition parameters 47
TT [ObjectMap] parameter (synonym for TargetTable) 230

UNIT parameter
in [ConflictFile] section 176
in [TemporaryFiles] section 278
UNIX
executable command 58, 60
PATH variable 30

unqualified name, restart table 253
UPDATE privilege 26, 31, 301
UpdateColumns parameter [LogicalLog] 205
URCheckRI parameter [DistributionTuning] 189
UseLLogSysStrings parameter [LogicalLog] 206

V
V10.1 (QM001646391) INLINE LOBs
added value Inline to Attributes section 195
Value parameter [ObjectPartition] 237
variables, symbolic 165
verifying product maintenance 56
viewing installation logs 308
VOLSER parameter
in [ConflictFile] section 176
in [TemporaryFiles] section 279
VSAM files, large volume columns 195, 199, 281

W
Warn action 82
wildcard patterns
in object names 101
Windows operating system
installation considerations 300
installation procedure 300
PATH setting 30
Work_item log type 209
work, distribution of. See distribution of work
workload balancing 31
WORKNUM parameter [LogicalLog] 207
wrapping lines in configuration file 93

X
XML columns
including or excluding changes from 199
longer statement lengths 263
VSAM files for 199, 281
XMLSTRING control file 199, 281