SQL Explorer for DB2
User Guide

Supporting

Version 11.2.00 of SQL Explorer for DB2
Version 11.2.00 of SQL Performance 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>
</table>
| BMC SOFTWARE INC  
2101 CITYWEST BLVD  
HOUSTON TX 77042-2827 USA | 1 713 918 8800 or 1 800 841 2031 | 1 713 918 8000 |

**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 2006-2015 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 is a trademark or registered trademark of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

C/370, DB2, DB2 Universal Database, DRDA, IBM, MVS, OS/390, 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.

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 UNPUBLISHED—RIGHTS RESERVED UNDER THE COPYRIGHT LAWS OF THE UNITED STATES.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
- Related publications ......................................................... 11
- Conventions ........................................................................ 12
- Summary of changes .......................................................... 12

### Part 1  Getting started with SQL Explorer

#### Chapter 1  Overview of SQL Explorer
- Common problems when working with SQL ............................ 17
- What you can do with SQL Explorer ...................................... 18
  - Explain SQL statements .................................................. 19
  - Compare SQL statements ................................................. 20
  - Analyze mismatched data types ....................................... 20
  - Analyze the impacts of schema changes .......................... 20
  - Compare timestamp matches before production ............... 21
  - Migrate access path statistics to another subsystem .......... 21
  - Generate automatic DECLARE TABLE statements ............ 21
- SQL Explorer requirements ................................................ 22
  - Operating environment .................................................. 22
  - SQL Explorer and DB2 authority requirements ................ 22
- Integration with other BMC products .................................. 23
  - SQL Performance for DB2 solution ................................. 24
  - BMC Performance for DB2 SQL solution ......................... 24
  - CATALOG MANAGER for DB2 ........................................ 26
- SQL Explorer interface ..................................................... 26
  - SQL Explorer main menu .............................................. 26
  - ISPF conventions .......................................................... 28
  - Online Help ................................................................. 29

#### Chapter 2  Setting up SQL Explorer
- Logging on .......................................................................... 31
- Attaching SQL Explorer to local or remote DB2 subsystems ...... 31
  - Attaching to a DB2 subsystem on the same MVS system ...... 32
  - Attaching to an additional DB2 subsystem on a different MVS system ...... 33
- Expert rules ........................................................................ 34
  - Using predefined expert rules ........................................... 34
### Appendix C
Sample data set members for adding analysis to job steps 227

### Appendix D
Sample history table Explain queries 229

### Appendix E
Available variables for expert rules 231

<table>
<thead>
<tr>
<th>List of variables</th>
<th>231</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
About this book

This book contains detailed information about the BMC SQL Explorer for DB2 product and is intended for IBM DB2 Universal Database (DB2) system administrators, database administrators (DBAs), and DB2 application programmers.

To use this book, you should be familiar with the following items:

- DB2 for z/OS and OS/390
- IBM Multiple Virtual Storage (MVS), z/OS, or OS/390 systems
- Job control language (JCL) and the Interactive System Productivity Facility (ISPF)

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 Customer Support website at http://www.bmc.com/support. Most product shipments also include the books on a documentation CD.

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 PF1 within any product or click the Help button in graphical user interfaces (GUIs).

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

— 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.

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.

Summary of changes

This topic summarizes product changes and enhancements by version number and release date.
Version 11.2 May 2015

This release fixes known problems and includes the following changes or enhancements:

- End of support for IBM DB2 Version 9 systems
  Starting with this release, SQL Explorer for DB2 does not support IBM DB2 Version 9. Earlier releases continue to support Version 9.

- Expert rules
  The expert rules, used by the Common Explain component of SQL Explorer for DB2, have been upgraded. Obsolete rules have been deleted. All other rules are consistent with updates to the DB2 Optimizer.

- DB2 Analytics Accelerator (DAA) support in the Explain function
  The Explain component in SQL Explorer now reports SQL statements' eligibility to run on a DB2 Analytics Accelerator.

- Generation of the DSN_STAT_FEEDBACK table in DB2 Version 11 in the Explain function

Version 11.1 June 2013

This release of SQL Explorer for DB2 includes the following product enhancements and changes:

- Explain package—An Explain package compares the current plan and any or all of the saved plans for the statement. This feature lets you see the access paths for previous binds. The statement inserts the available plans into the plan table.

- REXX exec—Common Explain calls a REXX exec (capable of adding new facts to the rules engine knowledge base) before firing rules. This feature lets you define your own expert rule variables.

- Available variables for expert rules—SQL Explorer has an updated list of available variables for expert rules. See “List of variables” on page 231.

- Migrating access path statistics from a remote DB2—The Migrate Access Path Statistics feature now lets you migrate access path statistics from one DB2 subsystem to another subsystem from a remote LPAR. You can specify the subsystems (SSID) and distributed data facility (DDF) locations for the migrate operation.

- End of support for DB2 Version 8—Starting with this release, SQL Explorer does not support IBM® DB2 Version 8. Earlier releases continue to support Version 8.
Version 10.1 April 2011

This release of SQL Explorer for DB2 includes the following product enhancements and changes:

- Expert rules—The expert rules have a new format and use the Policy Rules engine. See “Expert rules” on page 34.

- Migration statistics—Modifications have been made to the fields for migration statistics. See “Migrating access path statistics” on page 169.
Getting started with SQL Explorer

This part provides an overview of SQL Explorer and describes how to set up the product.

Chapters:

- Overview of SQL Explorer
- Setting up SQL Explorer
Overview of SQL Explorer

This chapter provides a high-level overview of SQL Explorer.

Common problems when working with SQL

Inefficient SQL statements can have a major impact on application throughput and response time.

Even minor changes in application or DB2 data structures can significantly affect performance and availability. For example, subtle statistical changes in the DB2 catalog can cause the DB2 optimizer to change access path selections. If the changes are not detected before the application moves to production, the application might perform poorly or be unavailable.

The SQL Explorer for DB2 product is an SQL analysis tool. Application developers can use it to identify performance problems caused by inefficient SQL statements and correct them before the problems reach production.

Similarly, DBAs can identify and manage performance impacts resulting from data structure changes before implementing those changes in production. SQL Explorer also helps resolve problems that are already in production.

Table 1 on page 17 describes how SQL Explorer addresses common problems that application developers and DBAs face.

Table 1: Common problems

<table>
<thead>
<tr>
<th>Problem</th>
<th>How SQL Explorer helps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plan table information is cryptic and requires expert knowledge of the relational database management system (RDBMS).</td>
<td>SQL Explorer offers expert rules that you can tailor for your environment or technical audience. DBAs and performance experts can customize the expert rules to establish installation-specific rules, set or change thresholds, and issue warnings on keywords to be avoided. Application developers can use SQL Explorer to parse and check SQL code against the expert rules base.</td>
</tr>
</tbody>
</table>
## What you can do with SQL Explorer

You can prevent or resolve common SQL-related performance problems by using SQL Explorer.

You can use SQL Explorer to:

- Explain SQL statements
- Compare SQL statements
- Analyze mismatched data types
- Analyze the impacts of schema changes
- Compare timestamp matches before production

<table>
<thead>
<tr>
<th>Problem</th>
<th>How SQL Explorer helps</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBAs need to control application performance within their change-</td>
<td>Impact analysis enables you to see the effects of anticipated changes before making the changes. Also, you can use SQL Explorer to compare database request module (DBRM) libraries to DB2 catalogs. This type of comparison reduces testing time before promoting an application into production. This approach also prevents the need for rolling back the DB2 catalog to remove data that was added during testing.</td>
</tr>
<tr>
<td>management procedures.</td>
<td></td>
</tr>
<tr>
<td>Application design reviews sometimes fail to produce quantified,</td>
<td>Through integration with the BMC CATALOG MANAGER product, SQL Explorer can run automatic Explain jobs on binds or rebinds. Also, online and batch reporting offers ad hoc or regular reporting capabilities for planning and impact analysis. Recommendations provide a source of expert knowledge for DBAs, application developers, and performance experts.</td>
</tr>
<tr>
<td>objective information to justify change recommendations.</td>
<td></td>
</tr>
<tr>
<td>Application developers want more involvement in performing application-</td>
<td>Developers can check their SQL in the development phase of the application, rather than waiting until the application has gone to test or production and caused a problem. By offering explanations of plans, packages, DBRMs, or DBRMLIBs, SQL Explorer enables you to check entire applications. DBRMLIB explanations enable you to explain SQL stored outside of the DB2 catalog. SQLX edit macro enables quick, online explanation or execution of a single SQL statement from a TSO edit session.</td>
</tr>
<tr>
<td>specific tuning.</td>
<td></td>
</tr>
<tr>
<td>Analyzing SQL can be time-consuming. Rewriting SQL statements without</td>
<td>You can add or modify rules to enforce installation standards and to detect SQL statements that should be avoided in specific circumstances.</td>
</tr>
<tr>
<td>expert knowledge of the RDBMS can be a trial-and-error process, but</td>
<td></td>
</tr>
<tr>
<td>maintaining expert knowledge can be difficult when new releases</td>
<td></td>
</tr>
<tr>
<td>introduce SQL changes that affect the DB2 optimizer.</td>
<td></td>
</tr>
</tbody>
</table>
Migrate access path statistics to another subsystem
Generate automatic DECLARE TABLE statements

**Note**
With SQL Explorer, you can analyze objects on local or remote DB2 subsystems through the Call Attach Facility (CAF) or through the DB2 Distributed Data Facility (DDF).
Also, many of the reports that SQL Explorer produces are available either online or in batch mode.

For more information, view the Quick Course "SQL Explorer for DB2—Getting Started."

**Related Information**
- "Attaching SQL Explorer to local or remote DB2 subsystems“ on page 31
- "Understanding online versus batch processing” on page 45

**Explain SQL statements**

In SQL Explorer, running an Explain job explains the steps that the DB2 optimizer must take to execute the relevant SQL statements.

You can run an Explain job to see an explanation of any of the following types of statements:

- All explainable SQL statements in a plan, package, or database request module (DBRM)
- SQL statements stored outside the DB2 catalog (referred to as a DBRM library Explain or DBRMLIB Explain)
- A single SQL statement (online or in batch mode)

The Explain option includes the cost and filter factor for each process step of the SQL statement. This capability enables the DBA or developer to determine quickly which part of the query is estimated to consume the most resources.

**Related Information**
- “Explaining SQL statements” on page 63
Compare SQL statements

The Compare option compares versions of plans or packages through historic baselines (Explain results) that you created with the Explain option. Running a Compare job compares the Explain results for two different versions of a plan, package, or DBRM.

Related Information

- “Explaining and comparing plans, packages, and DBRMs” on page 113

Analyze mismatched data types

Using the Mismatch Analysis option, SQL Explorer analyzes a plan or package, comparing the data type and length of host variables or expressions to the DB2 catalog and reporting any differences. You can use this feature to identify the effects of anticipated column changes.

For example, the DB2 optimizer can change access paths based solely on the data-type differences between the column and the host variable or expression. However, the DB2 optimizer does not provide any warning that a data-type mismatch occurred. The Mismatch Analysis option detects data-type mismatches and reports whether they could result in changes in the access path.

Related Information

- “Performing mismatch analysis” on page 133

Analyze the impacts of schema changes

Impact Analysis reports each SQL statement that would be affected by a change to a column, table, synonym, view, or alias. The report also lists each plan, package, or DBRM that contains the affected SQL statement.

Related Information

- “Analyzing for impact of schema changes” on page 141
Compare timestamp matches before production

You can compare a DBRM to a load library to check for timestamp matches between a module and that module's corresponding member in a list of specified data sets. For example, you can

- Compare a single DBRM member to one or more load libraries
- Compare a single load member to multiple DBRM libraries
- Compare a DBRMLIB to a DB2 catalog to ensure that timestamps match and no binds are missing
  This option also generates binds for which the catalog timestamp is earlier than the DBRMLIB timestamp.

Related Information

- “Preventing timestamp mismatches before production” on page 147

Migrate access path statistics to another subsystem

You can use the From Location and To Location fields in the Migrate Access Path Statistics option to copy statistics from one DB2 catalog to another catalog that is running on a different system. Alternatively, you can copy statistics to a DB2 catalog that is running on the same subsystem.

Example

Assume that you migrate production statistics to a test environment. You can view the actual access paths that the DB2 optimizer selects without having to replicate the production environment. You can preserve the test data while viewing the access paths based on production data.

Related Information

- “Migrating access path statistics” on page 169

Generate automatic DECLARE TABLE statements

For COBOL, C, or PL/I applications, you can use the Declarations Generator to automatically generate a DECLARE TABLE statement, host variable structure, and
indicator structure for nullable columns. The generated statements are inserted into a partitioned data set (PDS) member that you can include in various applications. This option offers several advantages:

- Reduces application development time
- Avoids manual coding errors
- Ensures that host variable definitions match DB2 column types

**Related Information**

- “Generating table declarations automatically” on page 191

---

**SQL Explorer requirements**

SQL Explorer has the following environment and authorization requirements:

**Operating environment**

SQL Explorer requires the following operating environment:

- DB2 Version 10 or 11
- An IBM-supported version of OS/390 or z/OS that is active on all systems in the sysplex

**SQL Explorer and DB2 authority requirements**

SQL Explorer uses DB2 security to ensure that any access or updates to DB2 are appropriately authorized.

SQL Explorer uses the following authority:

- Execution of an SQL statement via the SQLX edit macro or the online ad-hoc explain interface uses the DB2 authority of your TSO ID.
Migrate Access Path Statistics uses Install SYSADM authority when running the job from an APF-authorized library. BMC recommends that you back up your catalog before you perform the Update step.

The Explain component uses the following authority:

- INSTALL SYSADM authority in the following situations:
  - When running Explain jobs online while connected to the Data Collector
  - When running Explain jobs in batch mode from an APF-authorized library

- Authority of your TSO ID in the following situations:
  - When running Explain jobs online without being connected to the Data Collector
  - When running Explain jobs in batch mode from a library that is not APF-authorized

**Note**

DB2 writes only to plan tables for the ID under which you are set. The Explain component initiates a SET CURRENT SQLID statement to the specified plan table owner in order to write to the plan tables.

If you receive a -553 SQL code when the SET CURRENT SQLID statement is issued against the plan table owner name, Explain processing continues (using your AUTHID) and writes to the `authID.PLAN_TABLE`. If needed, the Common Explain component of SQL Explorer dynamically builds the required plan tables in a BMC database and table space.

**Related Information**

- "Explaining and executing an individual SQL statement" on page 105
- "Migrating access path statistics" on page 169

### Integration with other BMC products

SQL Explorer is a component of the following solutions:

- **SQL Performance for DB2**
- **BMC Performance for DB2SQL**

You can integrate the Common Explain component of SQL Explorer with and launch it from the BMC CATALOG MANAGER for DB2 product.
SQL Performance for DB2 solution

With the SQL Performance solution, you can fine-tune an application from its inception through its growth and retirement. SQL Performance users benefit from the features of the following components:

- APPTUNE for DB2 product
- SQL Explorer for DB2 product
- Index component
- Performance Advisor component
- Workload Access Path Compare component
- Index Advisor component
- Exception Advisor component
- Reorg Advisor component
- Workload Statistics Advisor component

SQL Performance users benefit from additional features that are available when one SQL Performance for DB2 component can rely on the presence of the other components. SQL Performance enables you to:

- Identify large, resource-consuming or performance-constrained SQL statements
- Proactively identify and correct problems before changes reach production
- Identify what to fix (SQL text, update statistics, insufficient indexes, and so on)
- Quickly correct performance problems in production that result from updated catalog statistics and executing binds
- Monitor SQL performance across your sysplex
- Improve performance by identifying high-impact tuning operations, such as reorganizing objects, eliminating obsolete indexes, and identifying degraded objects
- Compare and analyze workload access paths to improve performance
- Provide recommendations for indexes to improve performance
- Identify which catalog statistics will, if updated, reduce the cost of running a set of SQL statements

For more information, see the SQL Performance for DB2 User Guide.

BMC Performance for DB2 SQL solution

With the BMC Performance for DB2 SQL solution, you can fine-tune an application from its inception through its growth to its retirement. BMC Performance for DB2 SQL users benefit from the features of the following components:

- APPTUNE for DB2 product
- SQL Explorer for DB2 product
BMC Workbench for DB2 product
This premium feature is available only if you have a valid license for the BMC Performance for DB2SQL solution.

- Index component
- Performance Advisor component
- Workload Access Path Compare component
- Index Advisor component
- What-if Index component
This premium feature is available only if you have a valid license for the BMC Performance for DB2SQL solution.

- Exception Advisor component
- Reorg Advisor component
- Statistics Advisor component
This premium feature is available only if you have a valid license for the BMC Performance for DB2SQL solution.

BMC Performance for DB2SQL users benefit from additional features that are available when one BMC Performance for DB2SQL component can rely on the presence of the other components. BMC Performance for DB2SQL enables you to:

- Identify large, resource-consuming or performance-constrained SQL statements
- Proactively identify and correct problems before changes reach production
- Identify what to fix (SQL text, update statistics, insufficient indexes, and so on)
- Quickly correct performance problems in production that result from updated catalog statistics and executing binds
- Monitor SQL performance across your sysplex
- Improve performance by identifying high-impact tuning operations (such as reorganizing objects) eliminating obsolete indexes, and identifying degraded objects
- Compare and analyze workload access paths to improve performance
- Provide recommendations for indexes to improve performance
- Use the What-if Index feature to evaluate the impact of applying a different index
- Identify which catalog statistics will, if updated, reduce the cost of running a set of SQL statements

For more information, see the SQL Performance for DB2 User Guide.
CATALOG MANAGER for DB2

As discussed in the *CATALOG MANAGER for DB2 User Guide*, CATALOG MANAGER includes numerous integration provisions to accommodate plan or package explanations.

---

**Related Information**

- “Integrating CATALOG MANAGER for DB2 with SQL Explorer” on page 157

---

**SQL Explorer interface**

The user interface for SQL Explorer for DB2 consists of Interactive System Productivity Facility (ISPF) panels.

The interface accommodates foreground processing of jobs or batch processing (that is, saving jobs in a partitioned data set for subsequent submission).

---

**SQL Explorer main menu**

On the SQL Explorer main menu, you select an option by entering its number or letter in the **Option** field to the left of the option list.

*Figure 1 on page 26* illustrates the main menu.

*Figure 1: SQL Explorer main menu (PSSPF000)*

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.</td>
<td>Options</td>
</tr>
<tr>
<td>1.</td>
<td>Explain</td>
</tr>
<tr>
<td>2.</td>
<td>Explain and Compare</td>
</tr>
<tr>
<td>3.</td>
<td>Compare</td>
</tr>
<tr>
<td>4.</td>
<td>Mismatch Analysis</td>
</tr>
<tr>
<td>5.</td>
<td>Impact Analysis</td>
</tr>
<tr>
<td>6.</td>
<td>DBRM / Load Compare</td>
</tr>
<tr>
<td>7.</td>
<td>Migrate Access Path Statistics</td>
</tr>
<tr>
<td>8.</td>
<td>Declarations Generator</td>
</tr>
<tr>
<td>R.</td>
<td>SQL Explorer Rules Help</td>
</tr>
<tr>
<td>M.</td>
<td>Maintenance</td>
</tr>
</tbody>
</table>

SSID . . . . . DIY4
DDF Location
If you have other System and SQL Performance for DB2 products or solutions installed, a common panel (Figure 2 on page 27 shows an example) might precede the SQL Explorer main menu.

**Figure 2: SQL Performance for DB2 menu**

![SQL Performance for DB2 menu](image)

- **Note**
  The options that are available on the SQL Performance for DB2 menu might vary from those shown in Figure 2 on page 27. The options on the panel change depending on the product licenses that you have activated.

Selecting option **S** (for SQL Explorer Component) on the common panel displays the SQL Explorer main menu.

*Table 2 on page 27* briefly describes each option on the SQL Explorer main menu.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Options</td>
<td>Enables you to change general, analysis, and JCL options.</td>
</tr>
<tr>
<td>Explain</td>
<td>Produces online or batch information about a plan, package, DBRM, DBRMLIB, or ad hoc SQL and suggests recommendations for improving performance</td>
</tr>
<tr>
<td>Explain and Compare</td>
<td>Analyzes and evaluates a plan, package, or DBRM and suggests recommendations for improving performance</td>
</tr>
<tr>
<td>Compare</td>
<td>Evaluates a plan, package, or DBRM Explain against previous Explains and suggests recommendations for improving performance</td>
</tr>
<tr>
<td>Mismatch Analysis</td>
<td>Reports mismatches (in data type, length, or both) between host variables or expressions and a DB2 column or expression</td>
</tr>
<tr>
<td>Impact Analysis</td>
<td>Lists all SQL statements that are affected by a schema change</td>
</tr>
<tr>
<td>DBRM/Load Compare</td>
<td>Finds matching timestamps between DBRM and load libraries to avoid -805 and -818 SQL errors</td>
</tr>
</tbody>
</table>
### ISPF conventions

This section describes how to use the ISPF **Command** line, ISPF keys and commands, and wildcards.

#### Command line

In SQL Explorer, the main menu and most other panels have a **Command** line that appears at the top left of the panel (beneath the title):

**Command ===>

You can issue a SQL Explorer command by typing it in uppercase or lowercase characters on the **Command** line.

#### ISPF keys and commands

In SQL Explorer, most ISPF commands work the same as in other ISPF applications.

Use the following keys to move through the panels:

- **DOWN (PF8)** displays the next Help topic. When you are viewing a report, use DOWN to scroll down.
- **END (PF3)** saves any changes and returns to the previous panel. The object selection panels use the END command to save object selections and then proceed to the next panel.

- **Enter** accepts default entries or changes, and displays the next panel.

- **HELP (PF1)** displays Help panels or detailed error messages.

- **UP (PF7)** displays the previous Help topic. When you are viewing a report, use UP to scroll up.

### Wildcards

You can use the following wildcard characters to search for and list DB2 objects:

- `_` (underscore) matches any single character.

- `%` (percent sign) or `*` (asterisk) matches a string of zero or more characters.

To specify a group of objects, type a qualified name or a name with a wildcard character.

### Online Help

SQL Explorer provides an online Help system, which includes detailed information about panels and fields.

To access the Help system, press HELP (PF1) or type HELP on the **Command** line from any panel.

When you are viewing a Help panel, press **Enter**, DOWN (PF8), or LEFT (PF10) to display the next Help panel, or press UP (PF7) or RIGHT (PF11) to display the previous Help panel.

The online Help also provides additional tips and suggestions that are associated with each expert rule.

---

**Related Information**

- "Displaying Help for expert rules" on page 44
Setting up SQL Explorer

This chapter describes how to set up SQL Explorer.

Logging on

The SQL Explorer product installation creates one of the following logon mechanisms:

- A menu option for logging on to SQL Explorer under ISPF
  See “SQL Explorer main menu” on page 26.

- A CLIST for logging on to CLIST under TSO

Select the ISPF option or execute the CLIST to log on to SQL Explorer. Contact your SQL Explorer administrator for the proper procedure to use at your site.

Note
If you are using multiple System and SQL Performance for DB2 products, a common main menu is displayed.

Attaching SQL Explorer to local or remote DB2 subsystems

A SQL Explorer session is always attached to a DB2 subsystem, either through the Call Attach Facility (CAF) or through the DB2 Distributed Data Facility (DDF).

During a session, you can also attach to an additional DB2 subsystem, as follows:

- You can use CAF to attach to a subsystem on the same MVS system.
- You can use DDF to attach to a subsystem on any MVS system.
The following table describes the requirements of CAF and DDF:

<table>
<thead>
<tr>
<th>Access method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAF</td>
<td>The target DB2 subsystem must be in the same MVS system as the current SQL Explorer session.</td>
</tr>
</tbody>
</table>
| DDF           | The target DB2 subsystem has the following requirements:  
  - Can be in a different MVS system from the current SQL Explorer session  
  - Must be defined to DB2 in the communications database |

**Attaching to a DB2 subsystem on the same MVS system**

The subsystem to which you connect during a SQL Explorer session is determined by the value that you specify for the SSID.

Figure 3 on page 32 illustrates the process of attaching a session through CAF.

**Figure 3: Attaching to an additional subsystem (same MVS system)**

*Note*

To submit the JCL on another MVS system, you must have a route card.

**To attach to an additional subsystem on the same MVS system**

Change the SSID on the SQL Explorer main menu. You do not have to exit the product to make this change.
Attaching to an additional DB2 subsystem on a different MVS system

From a SQL Explorer session, you can attach to a DB2 subsystem on another MVS system if you have the DB2 DDF.

**Example**

Assume that you have a production system on one MVS system and a development system on another MVS system. With SQL Explorer running on the development system, you can use DDF to run a SQL Explorer job on the production system. Figure 4 on page 33 illustrates the process.

**Figure 4: Attaching to an additional subsystem (different MVS system)**

### Before you begin

Ensure that the following requirements are met:

- Remote locations are defined in the SYSIBM.LOCATIONS table of the local DB2 subsystem and are connected using the DB2 DDF.
SQL Explorer is installed on the local SSID, and the object is bound with SQL Explorer on the remote DB2 subsystems.

SQL Explorer is installed on the remote DB2 subsystems.

The same version of SQL Explorer is on all DB2 subsystems.

The collection IDs are the same for all SQL Explorer installations (in order to use DDF to connect).

**Note**

If SQL Explorer has not been installed for a SSID that you entered, SQL Explorer issues a CONFIG ERROR (such as UNABLE TO DETERMINE DB2 LOADLIBS FOR SSID 'XXXX'). Press F1 to see a more detailed description of the error. For more information, see the *BMC Products and Solutions for DB2 Customization Guide*.

---

**To attach to an additional subsystem on a different MVS system**

In the **DDF Location** field on the SQL Explorer main menu, enter the location of the DDF.

---

**Expert rules**

The SQL Explorer Explain feature is driven by a set of expert rules.

These rules trigger messages that tell you where performance problems exist and what design changes are needed. Each expert rule has an associated severity value and threshold.

---

**Using predefined expert rules**

The product provides the following predefined sets of rules:

- DEFAULT contains rules that are related to performance issues. These rules are primarily relevant to DBAs.
- APPLDEV contains rules for application developers.
The rules files are available in the *hlq.LKSAMP* data set. The name of *hlq.LKSAMP* varies depending on your installation options and might be *hlq.BMCSAMP*. The name of the members that contain the rules files are as follows:

- **PSSDFLT** contains the DEFAULT rules.
- **PSSAPPL** contains the APPLDEV rules.
- **PSSJAPAN** contains the JAPAN rules.

**To use predefined rules**

1. On the Explain Options panel, specify the rules file to use by entering one of the following:
   - The member name (for example: **PSSDFLT**, **PSSAPPL**, **PSSJAPAN**)
   - Rule set name (for example: **DEFAULT**, **APPLDEV**, **JAPAN**)

   **Note**
   To use a rule set that you create, you must specify the member name on the Explain Options panel.

### Severity codes in rules

Each message that is associated with a rule has a severity code.

The severity code determines the order in which the messages are displayed and the color used to display them. Severe messages are displayed first, followed by warning and informational messages.

The severity code is displayed as the tenth character in the message identifier (as shown in the following example).

**Example**

BMC184032W

Table 3 on page 35 shows the possible severity codes.

**Table 3: Severity codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>S (severe error)</td>
<td>Correct the issue before continuing.</td>
</tr>
</tbody>
</table>
Customizing expert rules

You can customize or create your own custom rule sets. This task describes how to copy one of the default rule sets and modify its rules to meet your needs.

Consider the following when defining your own rules set:

- The facts and variables within the predefined rules are set internally. You can use the same facts within rules you define. For a list of facts, see “List of variables” on page 231.

- You can modify the severity codes on messages to adjust them to the severity level that you require.

- If you do not want a rule to trigger, use one of the following methods to disable it:
  - Enclose rules that you do not want to fire within a comment statement. The syntax for a comment is /* commentText */.
  - Set the predicate comparison to A=0. For example, you could modify a rule as follows to ensure that it is never triggered:

    ```sql
    RULE1008: IF A = 0
    THEN OUT=OUT||"BMC184008X-A Correlated Subquery will be evaluated for each row in the composite table. It performs like a Nested Loop Join."
    
    /**
    * Added by SQL Explorer for DB2 User Guide
    * 
    */
    ``

  - Set the severity code of the message to X.

- If you want a rule to always trigger, ensure that the predicate always evaluates to true. For example, your predicate could be as follows:

    ```sql
    RULE1008: IF 0 = 0
    THEN OUT=OUT||"BMC184008I-A Correlated Subquery will be evaluated for each row in the composite table. It performs like a Nested Loop Join."
    
    /**
    * Added by SQL Explorer for DB2 User Guide
    * 
    */
    ``

- If your message text is longer than a single line, ensure that you use the || characters to append the next line. Enclose each line in the message within quotation marks and end the message with 
.

    ```sql
    RULE5056: IF FULLOUTER = "Y"
    THEN OUT=OUT||"BMC184620I-The clause FULL OUTER JOIN includes unmatched rows from both tables. Missing values in the result table are filled with nulls."
    
    /**
    * Added by SQL Explorer for DB2 User Guide
    * 
    */
    ```
You can use variable values within the message text:

— If you want to display a variable value in the message, insert the variable name within || characters. Do not include the variable name within quotation marks. For example, the following message includes the variable value for LOCKSIZE on the final line:

```sql
RULE2054P: IF LOCKRULE = "P" THEN LOCKSIZE = "PAGE"
RULE2054R: IF LOCKRULE = "R" THEN LOCKSIZE = "ROW"
RULE2054T: IF LOCKRULE = "T" THEN LOCKSIZE = "TABLE"
RULE2054S: IF LOCKRULE = "S" THEN LOCKSIZE = "TABLESPACE"
RULE2054L: IF LOCKRULE = "L" THEN LOCKSIZE = "LOB"
RULE2054X: IF LOCKRULE = "X" THEN LOCKSIZE = "XML"
RULE2054: IF LOCKRULE <> "A"
   THEN OUT=OUT||"The tablespace has a LOCKSIZE of "||LOCKSIZE||".
```

— If the value that you want to display is a numerical value, check to see if an equivalent string variable exists. (See “List of variables” on page 231.) The equivalent variable name has a C appended. For example, the following message includes the variable value for SEGSIZE on the final line:

```sql
RULE2056: IF TBTYPE = "T" AND SEGSIZE > 2
   THEN OUT=OUT||"The tablespace has a SEGSIZE of "||SEGSIZEC||".
```

To customize expert rules

1. Copy one of the following members in the BMCSAMP data set to create your own version of the rule set:
   - PSSDFLT (for the DEFAULT rule set)
   - PSSAPPL (for the APPLDEV rule set)
   - PSSJAPAN (for the JAPAN rule set)

   **Note**
   Do not modify the sample members directly. These files will be overwritten and any changes lost the next time that you install the product. Always create a copy of the file and save it in the hlq.UBMCSAMP library to modify.

2. Modify the copy of the rule set in the hlq.UBMCSAMP data set as needed.

   Ensure that the rules conform to the structure described in “Structure of rules” on page 39. Ensure also that the RULESET name matches the name of the new rules member, as does NEWRULE1 in the example below:

   ```sql
   Command ===> BROWSE HLQ.LLOCSAMP(NEWRULE1) -
   ```
3 Update the Explain options to identify this rule set.

   a At the SQL Explorer main menu, select 0 (Options).

   b At the Options panel, select 1 (General).

   c At the General Default Options panel, press Enter.

   d At the Explain Options panel, specify the member name of the rule set that you want to use in the Rule set field and press Enter.

---

**Note**

For information on the other options that you can enter on the Explain Options panel, see information relating to specifying Explain options.
You can also access the Explain Options panel from the Explain Object Specification panel by selecting the Explain Options field.

---

**Related Information**

- “List of variables” on page 231
- “Specifying Explain options” on page 76

---

**Defining new expert rules**

You can define new expert rules variables (facts) by creating a REXX EXEC. Common Explain creates the set of rules variables based on the SQL statement that is being explained. You can set new values for those variables and you can make your own variables.

**To define new expert rules**

1 Using a text editor, create a REXX exec.

   **Note**

   The BMCSAMP data set contains a sample REXX exec (PSSREXI) that you can edit to create a customized rule set.

2 Save the edited REXX exec as a member of the same data set as the rule sets.
The name of the member must be PSSREXIT.

If PSSREXIT exists in the data set, Common Explain executes the REXX exec before running the rules engine. The REXX exec replaces and adds variable values to which the rules engine will subsequently refer.

REXX exec can call the following functions:

- SETS and SETF functions take two arguments, the variable name and the variable value.
- GETS and GETF functions take one argument, the variable name.
- PSSRXSQL function takes one argument, an SQL statement.

By calling the PSSRXSQL function, the REXX exec can perform the following SQL statements:

- FETCH INTO
- SELECT INTO
- COMMIT
- ROLLBACK
- OPEN
- CLOSE
- EXECUTE IMMEDIATE (for executing INSERT, UPDATE, or DELETE SQL statements)

PSSRXSQL sets the following REXX variables after executing an SQL statement:

- SQLCODE
- SQLSTATE
- SQLERRM (contains the text of an error message)
- “INTO” variables
- cursor_name.n variables

For more information, see “PSSRXSQL external function” on page 41.

Structure of rules

You store rules in a data set member.

The rules are stored in the BMCSAMP data set, in the following members:

- PSSDFLT for default rules
- PSSAPPL for rules for application developers
- PSSJAPAN for rules in Japanese
**Standard format for a rule set**

The first line of the data set member contains the rule set declaration:

```
RULESET ruleSetName
```

Following the RULESET declaration, you can have one or more rules. Rules are processed in the order in which they are defined in the rule set. Each rule adheres to the following standard format, though the ELSE clause is optional.

```
name: IF predicate
  THEN OUT=action1
  ELSE OUT=action2
```

In this format, the variables are as follows:

- **name** indicates the name of the rule and must be unique within the rule set. Do not include a space between the name of the rule and the colon.

- **predicate** is an IF statement that specifies the value that is being evaluated and the value to which it is being compared. The predicate can include multiple conditions separated by AND or OR.

  **Example**
  - IF UPDATE_NO_WHERE = "Y"
  - IF MIN = "Y" OR MAX = "Y"
  - IF OPTIMIZE = "Y" AND (MIN = "Y" OR MAX = "Y" OR AVG = "Y" OR SUM = "Y")
  - IF PARTITIONS > 0

- **action1** value indicates the action that will be performed if the condition within the predicate is true. Usually, the action specifies issuing a message. The message must end with `\n` and be enclosed in quotes. Vertical bars (| |) indicate that the message continues across lines.

  **Example**
  ```
  THEN OUT=OUT||"BMC184024I-Consider the use of BETWEEN in "||"place of >= and <=. DB2 may consider using an index to provide the"||"answer.\n"
  ```

- **action2** indicates the action that will be performed if the condition within the predicate is false. This value follows the same rules as **action1**.
Samples from the default rule set

Following are descriptions of sample rules (RULE1032, RULE2002, and RULE1026) from the default rule set.

RULE1032 triggers if OPTIMIZE and UNION are both used. The rule then produces a message that indicates that the rows will not be optimized due to this condition:

RULE1032: IF OPTIMIZE = "Y" AND UNION = "Y"
    THEN OUT=OUT||"BMC184032W-The OPTIMIZE FOR N ROWS will be"||
        " ignored due to the use of a UNION, UNION ALL, or UNION DISTINCT.\n"

RULE2002 triggers if VALIDATE is set to R. The rule then produces a message to indicate that this setting can impact performance:

RULE2002: IF VALIDATE = "R"
    THEN OUT=OUT||"BMC184172I-The SQL and DB2 authorizations "||
        "will be checked at execution time. This can adversely impact perfo"||
        "rmance. Review with DBA on this usage.\n"

RULE1026 provides two different message outputs by providing an ELSE statement. If QUALIFIED is set to N, one message is generated; otherwise, a different message is generated.

RULE1026: IF QUALIFIED = "N"
    THEN OUT=OUT||"BMC184026I-The tables referenced in the FR"||
        "OM clause are not qualified. These will be resolved at bind time b"||
        "y the QUALIFIER name, by a SET CURRENT SQLID/SCHEMA if the query i"||
        "s dynamic, or it will default to the executing AUTH ID.\n"
    ELSE OUT=OUT||"BMC184028I-The tables referenced in the FR"||
        "OM clause are fully qualified. This will limit the flexibility for"||
        " the qualifier to be resolved at bind time.\n"

PSSRXSQL external function

The PSSRXSQL external function sets values according to the SQL statement. You can call PSSRXSQL from REXX exec whenever you need to execute SQL.

In each example in Table 4 on page 41, the argument in parentheses is an SQL statement. When PSSRXSQL runs, it sets the variable rc to the return code of the command that ran. The value in the rc variable determines subsequent processing. Using this external PSSRXSQL function, you can perform any SQL statement.

### Table 4: PSSRXSQL external function

<table>
<thead>
<tr>
<th>SQL statement</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN CURSOR</td>
<td>rc = pssrxsql(&quot;OPEN cursorName CURSOR FOR&quot; sqlStatement)</td>
<td>Prepare the SQL SELECT statement and open the specified cursor.</td>
</tr>
<tr>
<td>SQL statement</td>
<td>Example</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>FETCH CURSOR</td>
<td><code>rc = pssrxsql(&quot;FETCH cursorName&quot;)</code></td>
<td>Fetch the first or next row. Return SQLCODE = 100 when all rows have been fetched. The first form of the function puts the data into variables whose names are <code>cursorName.columnName</code>. It also puts values into variables like <code>cursorName.n</code>, where <code>n</code> is the column number of the query. The second form puts values into the specified variables. The maximum length of a variable name is 250 characters. Variable names can contain @, #, $, ¢, ., !, ?, or _.</td>
</tr>
<tr>
<td></td>
<td><code>rc = pssrxsql(&quot;FETCH cursorName INTO v1, v2, ... vn&quot;)</code></td>
<td></td>
</tr>
<tr>
<td>CLOSE CURSOR</td>
<td><code>rc = pssrxsql(&quot;CLOSE cursorName&quot;)</code></td>
<td>Close the specified cursor.</td>
</tr>
<tr>
<td>EXECUTE</td>
<td><code>rc = pssrxsql(sqlStatement)</code></td>
<td>Prepare and run the SQL statement. If the statement is SELECT, return no more than one row. The function sets variables as it does for FETCH. If the SELECT statement has no INTO clause, the stem name is always SELECT.</td>
</tr>
<tr>
<td>COMMIT</td>
<td><code>rc = pssrxsql(&quot;COMMIT&quot;)</code></td>
<td>Commit uncommitted work, and close all open cursors.</td>
</tr>
<tr>
<td>ROLLBACK</td>
<td><code>rc = pssrxsql(&quot;ROLLBACK&quot;)</code></td>
<td>Back out relational database changes made since the last commit.</td>
</tr>
</tbody>
</table>

**Note**

The current version of the PSSRXSQL external function does not support the following items:
- More than three open cursors at the same time
- Parameter markers
- The DESCRIBE command
- Mixed-case commands
- LOBs

Commands must be all lowercase or all uppercase. For example, either `open` or `OPEN` is acceptable, but `oPeN` is not valid.

**PSSRXSQL return codes**

If PSSRXSQL encounters non-SQL errors, it returns one of the following return codes:

<table>
<thead>
<tr>
<th>Return code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Success</td>
</tr>
<tr>
<td>Return code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
</tr>
<tr>
<td>30</td>
<td>No command in the argument string</td>
</tr>
<tr>
<td>31</td>
<td>Already disconnected</td>
</tr>
<tr>
<td>32</td>
<td>Already connected</td>
</tr>
<tr>
<td>33</td>
<td>Cursor already open</td>
</tr>
<tr>
<td>34</td>
<td>Too many open cursors</td>
</tr>
<tr>
<td>35</td>
<td>Unknown cursor</td>
</tr>
<tr>
<td>36</td>
<td>Out of memory</td>
</tr>
<tr>
<td>37</td>
<td>SQL failure</td>
</tr>
</tbody>
</table>
The variable SQLCODE is set, and the variable SQLERRM contains a printable error message. |
| 38          | Argument string length greater than 32,767 characters |
| 39          | An ‘INTO’ was expected but not found |

## Printing error messages

The following code formats and prints SQL error messages to the SYSTSPRT data set when the product is running in batch mode or TSO:

```plaintext
rc = pssrxsql("select current_timestamp into :TS from sysibm.sysdummy1")
if rc <> 0 then do
  msg = "BMC184001S-PSSREXIT rc=" || rc || "\n"
  if sqlcode <> 0 then do
    m = substr(sqlerrm,i+1,79)
    if m = ' ' then leave
    msg = msg || "BMC184001S-" || m || "\n"
  end
  rc = sets("OUT", msg)
end
```

**Note**

BMC does not recommend using the SAY command instead of the STEPS function because the DBC does not allocate a SYSTSPRT data set.

You can use the REXX variable SQLSTATE instead of SQLCODE. SQLSTATE indicates both warnings and errors.

If the product issues a warning without an error, PSSRXSQL returns $rc=0$, SQLCODE=0, ignoring warnings so that all the row values are returned. If you want to inspect warnings, use the SQLSTATE variable.
Displaying Help for expert rules

SQL Explorer offers online explanations and recommendations for any SQL Explorer rule.

You can display rules Help by using the SQL Explorer Rules Help option on the main menu or by using the PSSMSG command.

To display Help for a rule (SQL Explorer main menu)

1. On the SQL Explorer main menu, type R (for SQL Explorer Rules Help) and press Enter.

   The BMC MESSAGE PROCESSOR: SPECIFY SEARCH ARGUMENT panel is displayed.

2. In the BMC Message ID field, type the ID of the rule that you want to display (for example, type BMC184326), and press Enter.

   The message explanation and recommendation are displayed (Figure 5 on page 44).

   Figure 5: BMC Message Processor: Message Text panel (PSSMP003)

<table>
<thead>
<tr>
<th>PSSMP003</th>
<th>----------</th>
<th>BMC MESSAGE PROCESSOR : MESSAGE TEXT</th>
<th>----------</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMC184326</td>
<td>- Rule # 2156</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message:</td>
<td>Tip: Consider using OPTIMIZE FOR 1 ROW if the desired answer set is small and you want to avoid the cost of List Prefetch.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Explanation:</td>
<td>OPTIMIZE FOR 1 ROW is a clause that will discourage the Optimizer from using List Prefetch.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recommendation:</td>
<td>Convert your SQL to use OPTIMIZE FOR 1 ROW.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. Press Enter to display subsequent message panels or END to exit.

To display Help for a rule (PSSMSG command)

1. On the product's COMMAND line, enter TSO PSSMSG msgNum. Replace msgNum with a valid message number.

   For example, enter TSO PSSMSG BMC184000.
Understanding online versus batch processing

SQL Explorer typically gives you the choice of running a job in batch mode or online except when you are analyzing a particularly large amount of data.

For example, you can run the combined Explain and Compare process only in batch mode.

The following methods are available for running a job in batch mode:

- You can navigate through the panels, selecting options and letting the product build the JCL for you.
- You can edit the parameters of an existing batch job to suit your needs.

Allocating a data set for a specific job

If an analysis job requires data sets that are not already allocated, the Allocate Data Set panel is displayed.

Use this panel to enter or change information about the data set allocation.

Figure 6: Allocate Data Set panel (PSSPDSNA)

PSSPDSNA ---------------------- Allocate Data Set -----------------------------
Command ==> 
Allocate Data Set with the Following Values   Y (Y/N)
DD Name . . . SYSIN
Data Set Name. RDAKNN.SQLXPLR.SYSPRINT(SQLX0044)

<table>
<thead>
<tr>
<th>Volume Serial . . .</th>
<th>(Blank for authorized default volume)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic Unit . . .</td>
<td>(Generic group name or unit address)</td>
</tr>
<tr>
<td>Space Units . . .</td>
<td>CYL</td>
</tr>
<tr>
<td>Primary Quantity .</td>
<td>1</td>
</tr>
<tr>
<td>Secondary Quantity .</td>
<td>1</td>
</tr>
<tr>
<td>Directory Blocks . .</td>
<td>64</td>
</tr>
<tr>
<td>Record Format . . .</td>
<td>FB</td>
</tr>
<tr>
<td>Record Length . . .</td>
<td>80</td>
</tr>
<tr>
<td>Block Size . . .</td>
<td>6240</td>
</tr>
<tr>
<td>Management Class .</td>
<td>(Blank for default management class)</td>
</tr>
<tr>
<td>Data Class . . .</td>
<td>(Blank for default data class)</td>
</tr>
<tr>
<td>Storage Class . . .</td>
<td>(Blank for default storage class)</td>
</tr>
</tbody>
</table>

To allocate a data set

1 In the Allocate Data Set with the Following Values field, accept the default (Y) to allocate the data set.

Specifying N cancels the allocation.
Note

The **DD Name** and **Data Set Name** fields show the DD name and data set name of the data set to be allocated. You cannot modify this information.

2 In the **Volume Serial** field, specify the volume serial (VOLSER) of the direct access volume where you want the data set to reside.

Leave this field blank if you want Storage Management to select an eligible volume. (The volume must already be defined for your environment.)

3 In the **Generic Unit** field, specify the generic group name or unit address for the direct access volume where you want the data set to reside (for example, SYSDA, 3380, 3390).

Leave this field blank if you want Storage Management to select the unit.

4 In the **Space Units** field, specify the unit in which the data set size is expressed:
   - BLKS for blocks
   - TRKS for tracks
   - CYLS for cylinders (default)

5 In the **Primary Quantity** and **Secondary Quantity** fields, specify the primary and secondary allocation quantities.

For both fields, the default is 1, and the maximum value that you can specify is 16777215. The unit is the unit type that you specified in Step 4 on page 46.

6 In the **Directory Blocks** field, specify the number of 256-byte directory blocks to be provided for a partitioned data set.

If you leave this field blank, the number defaults to 64. To allocate a sequential data set, specify 0. The maximum value you can specify for a partitioned data set is 99999.

7 In the **Record Format** field, specify one of the following formats:
   - FB (fixed length, blocked records)
   - FBA (fixed length, blocked records with ASA printer control characters for SYSPRINT data sets)

8 In the **Record Length** field, specify the logical record length, in bytes, of the records to be stored in the data set.
In the **Block Size** field, specify the block size (physical record length), in bytes, of blocks to be stored in the data set:

- If the specified record length is 80, the record format defaults to FB and the block size defaults to 3120.
- If the specified record length is 100, the record format defaults to FB and the block size defaults to 3200.
- If the specified record length is 132, the record format defaults to FBA and the block size defaults to 3300.

Specify the management, data, and storage classes:

- In the **Management Class** field, specify the name of the SMS management class that contains the migration and backup information for allocating the data set.
- In the **Data Class** field, specify the name of the SMS data class that contains the SPACE and LRECL information for allocating the data set.
- In the **Storage Class** field, specify the name of the SMS storage class for allocating the data set.

**Using online mode**

To run a SQL Explorer job online (this applies to all product functions but Explain), you must allocate data sets to store the resulting output.

You can specify additional data set options by typing **OPTIONS** (or **OPT**) on the **Command** line (available on most SQL Explorer panels) and completing the fields on the Data Set Options and Additional Data Set Options panels. For more information see “**Setting data set options**” on page 56

If the Allocate Data Set panel is displayed, provide allocation information as discussed in “**Allocating a data set for a specific job**” on page 45.

When you specify to run a job in the foreground and complete all of the required job options, the results are reported online.

**Using batch mode**

When you navigate through the SQL Explorer panels, the parameters are supplied. Alternatively, you can use these parameters to edit your JCL if you want to run a quick analysis.
The input and execution parameters described in “Parameter reference” on page 199 are used in batch mode for any SQL Explorer job.

You can specify multiple objects in a single batch job step execution for most product functions. To do so, include multiple requests separated by the - -NEWOBJ and - -ENDOBJ parameters. The product processes each request individually and places no limit on the number of requests.

You can include all of the available parameters. If no value is displayed after the equal sign (=), the product uses the default value for that parameter (unless blank is a valid value for the parameter).

You can provide the parameters by adding them instream in the SYSIN DD, or by pointing the SYSIN to a data set. The report can be written to a SYSOUT class or to a data set (the default). The report is written to the SYSPRINT DD statement.

**Specifying JCL options in batch mode**

When you run a job in batch mode, the Batch Job panel is displayed. Use this panel to specify JCL options before executing a job in batch mode.

**Figure 7: Batch Job panel (PSSPB000)**

PSSPB000  ---  Batch Job  ---

Command ==>

JCL Data Set . . . 'RDAKNN.SQLXPLR.CNTL(SQLX####)'
Current Counter 37 (Replaces #### in member name, then incremented)

Set JCL Options N (Y/N - Change options for creating JCL)
Build Job . . . Y (Y/N - Create JCL, save in JCL data set)
Edit Data Set . . Y (Y/N - Edit JCL data set)
Submit . . . N (Y/N - Submit JCL data set)

**To specify batch JCL options**

1 In the **JCL Data Set** field, specify the data set to which SQL Explorer writes the generated JCL.

You can type the name of a partitioned or sequential file, or you can specify the TEMP keyword. If you specify TEMP, SQL Explorer uses the data set specified in ZTEMPF.

**Note**

If the Allocate Data Set panel is displayed during this procedure, provide allocation information as discussed in “Allocating a data set for a specific job” on page 45.

2 In the **Current Counter** field, enter any value from 1 through 9999 to replace the '####' suffix in the PDS member name.

If needed, SQL Explorer adds leading zeros to fill the four-digit suffix.
3 (optional) In the **Set JCL Options** field, specify Y to review or change options for creating JCL.

4 In the **Build Job** field, specify whether to generate the JCL into the data set that you specified in the **JCL Data Set** field.

**Note**

After you build and edit the JCL the first time, an asterisk (*) is shown in both the **Build Job** and **Edit Data Set** fields to ensure that you do not accidently regenerate and write over your existing JCL.

5 In the **Edit Data Set** field, specify whether to edit the data set online.

If you specify Y, the JCL is displayed in an ISPF edit session.

6 In the **Submit** field, specify whether to submit the JCL for execution.

BMC recommends that you specify N the first time the Batch Job panel is displayed. After SQL Explorer generates the JCL, make any necessary changes to your data set. When the Batch Job panel is displayed again, specify Y in the Submit field to execute the JCL.

7 Press **Enter** to complete the actions that you specified.

### Specifying execution parameters

The execution parameters are shown at the beginning of each batch job step.

Some of these parameters might also appear as keywords in the SYSIN DD. The values in the SYSIN DD override any values that you enter for the execution parameters. The highlighted text in **Figure 8 on page 49** shows the execution parameters from a sample batch job step.

You can provide the parameters by adding them instream in the SYSIN DD, or by pointing the SYSIN to a data set. The report can be written to a SYSOUT class or to a data set (the default). The report is written to the SYSPRINT DD statement.

**Figure 8: Execution parameters in a batch job step**

```plaintext
//RDADACP  JOB (5213),CLASS=Q,MSGCLASS=X
/*ROUTE XEQ  BMCPLX1
/*JBPARM SYSAFF=DB2A
**
/** EXECUTION OF PLAN,PACKAGE,OR DBRM ANALYSIS
/**
//ANALYSIS EXEC PGM=PSSMAIN,
// PARM='01DEAE 02DAAVRMD1',
// REGION=0M,
// COND=(4,LT)
//STEPLIB DD DISP=SHR,DSN=BMCPERF.LOAD
```
Verifying applied maintenance

Complete this task if you need to identify what maintenance has been applied to SQL Explorer.

**To verify applied maintenance**

1. On the SQL Explorer main menu, type **M** and press **Enter**.

   The Product Maintenance panel is displayed (Figure 9 on page 50).

   **Figure 9: Product Maintenance panel (PSSPMNTD)**

<table>
<thead>
<tr>
<th>Command ==&gt;&gt;</th>
<th>Product Maintenance ==&gt;&gt;</th>
<th>Scroll ==&gt;&gt; CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product</td>
<td>SQL EXPLORER</td>
<td>Maintenance Date</td>
</tr>
<tr>
<td>Release</td>
<td>11.2.00</td>
<td></td>
</tr>
<tr>
<td>The following maintenance has been applied to the product:</td>
<td><strong>NONE</strong></td>
<td></td>
</tr>
</tbody>
</table>

2. Examine the maintenance list to identify what maintenance has been applied.

3. When you finish, press **END**.

**Reviewing and changing options**

SQL Explorer uses various options to define the operating environment and to control how product components work.

For example, the product options control aspects of how the product generates JCL, allocates data sets, and formats reports.

SQL Explorer provides default values for these options. However, you can change the defaults as needed to suit your environment or target audience.

- You can navigate through all of the options panels in one session, review the values, and make changes as needed.
You can adjust defaults later while working with the relevant product functions. From most panels in the product, you can specify SQL Explorer options by typing OPTIONS (or OPT) on the Command line and pressing Enter.

In either case, any changes that you make to the default values apply to all SSIDs, and the changes remain in effect until the next time you change them.

Subsequent sections explain the first approach, which uses the main Options panel (Figure 10 on page 51). You can access this panel by selecting Options from the SQL Explorer main menu.

Figure 10: Options panel (PSSPO000)

```
PSSPO000 ---------------------- Options ---------------------------------------
Command ===> Options
Specify one of the options and press ENTER to continue.
Options  1. General
         2. Analysis
         3. JCL Options
```

**Related Information**

- “Setting general options” on page 51
- “Setting analysis options” on page 53
- “Setting JCL options” on page 54

---

**Setting general options**

This section guides you through reviewing and changing options related to the following items:

- Report titles
- Breaks for columns listed in SQL clauses
- High-level qualifier (HLQ) for product data sets
- Truncation for long object names

**To review or change general options**

1. On the SQL Explorer main menu, type 0 (zero) for Options.

2. Change the SSID and DDF locations, if necessary, and press Enter.
The main Options panel is displayed.

3 In the **Options** field, type 1 (for **General**) and press **Enter**.

The General Default Options panel is displayed (Figure 11 on page 52).

![Figure 11: General Default Options panel (PSSPOGEN)](PSSPOGEN

| Command ====> | General Default Options |-------------------------- |
|-----------------------------------|--------------------------|
| Specify options and press ENTER to continue. |                     |
| Generate Report Title . . . . . . Y (Y/N) |                     |
| Column Break . . . . . . . . N (Y/N) Format 1 column per line in SQL text |                     |
| Data Set High-Level Qualifier RDAJXH.SQLXPLR |                     |
| Long Name Truncation Position R (Left, Middle, Right) |                     |
| Long Name Truncation Chars >> |                     |

4 In the **Generate Report Title** field, specify whether to include a report title in the output.

By default (Y), SQL Explorer includes the title. However, if you plan to review the report online from the output data set, BMC recommends that you specify N to omit the title.

5 In the **Column Break** field, specify whether to insert a break between columns listed as part of the SELECT, INSERT, DELETE, or UPDATE clause in the SQL text.

6 In the **Data Set High-Level Qualifier** field, type the high-level qualifiers for the data sets that the SQL Explorer analysis routines require.

7 In the **Long Name Truncation Position** field, specify where to begin dropping characters from object names that are too long to be displayed:

- **L** drops characters from the name’s left side.
- **M** drops characters from the middle.
- **R** drops characters from the right.

8 In the **Long Name Truncation Chars** field, specify the characters that replace the truncated string.

The default is >>, but any characters are valid in this field.

9 Press **Enter** or **END** to store changes and leave the panel:

- Press **Enter** to continue to the next set of options in the menu sequence.
Press **END** once to return to the previous panel, or twice to return to the main Options panel.

If you made changes, SQL Explorer stores them in your profile for use in the current session and future sessions.

**Setting analysis options**

This section guides you through reviewing and changing options related to SQL analysis:

- Explain options
- Compare options
- Mismatch analysis object options
- Impact analysis options
- Migrate access path statistics options
- Declarations Generator options

**To review or change analysis options**

1. On the SQL Explorer main menu, type 0 (zero) for Options.

2. Change the SSID and DDF locations, if necessary, and press Enter.

   The main Options panel is displayed.

3. In the Options field, type 2 (for Analysis) and press Enter.

   The Analysis Options panel is displayed:

   **Figure 12: Analysis Options panel (PSSPOAN0)**

<table>
<thead>
<tr>
<th>Options</th>
<th>1. Explain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2. Compare</td>
</tr>
<tr>
<td></td>
<td>3. Mismatch</td>
</tr>
<tr>
<td></td>
<td>4. Impact</td>
</tr>
<tr>
<td></td>
<td>5. Migrate Access Path Statistics</td>
</tr>
<tr>
<td></td>
<td>6. Declarations Generator</td>
</tr>
</tbody>
</table>

4. In the Options field, select one of the values and press Enter to go to that panel.
5 Press Enter or END to store changes and leave the panel:

- Press Enter to continue to the next set of options in the menu sequence.
- Press END once to return to the previous panel, or twice to return to the main Options panel.

Related Information

- "Specifying Explain options" on page 76
- "Specifying Compare options for a Compare job" on page 126
- "Reviewing or changing options for the Mismatch Analysis report" on page 139
- "Setting Impact Analysis Options" on page 144
- "Specifying options for the Migrate Access Path Statistics job" on page 171
- "Setting Declarations Generator options" on page 195

Setting JCL options

This section guides you through reviewing and changing the following options that are related to generating JCL with SQL Explorer:

- SMS classes for allocating data sets (see “Setting SMS class options” on page 55)
- Data set options for JCL, SYSPRINT, summary, and SYSIN data sets (see “Setting data set options” on page 56)
- Job card options for batch analysis jobs (see “Setting job card options” on page 58)

To review or change JCL options

1. On the SQL Explorer main menu, type 0 (zero) for Options.

2. Change the SSID and DDF locations, if necessary, and press Enter.
   The main Options panel is displayed.

3. In the Options field, type 3 (for JCL Options) and press Enter.
The JCL Options panel is displayed (Figure 13 on page 55).

**Figure 13: JCL Options panel (PSSPOJ00)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>==&gt;</td>
<td>1. SMS Classes</td>
</tr>
<tr>
<td></td>
<td>2. Data Sets</td>
</tr>
<tr>
<td></td>
<td>3. Job Cards</td>
</tr>
</tbody>
</table>

4 In the **Options** field, select one of the values and press **Enter** to go to that panel.

5 Press **Enter** or **END** to store changes and leave the panel:
   - Press **Enter** to continue to the next set of options in the menu sequence.
   - Press **END** once to return to the previous panel, or twice to return to the main Options panel.

**Setting SMS class options**

This task describes how to specify options for SMS management, data, and storage classes for your data sets.

1 Access the JCL Options panel using the procedure described in “Setting JCL options” on page 54.

2 In the **Options** field, type **1** (for SMS Classes) and press **Enter**.

The SMS Class Options panel is displayed (Figure 14 on page 55).

**Figure 14: SMS Class Options panel (PSSPOSMS)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>==&gt;</td>
<td>Management Class . . . . (Blank for default management class)</td>
</tr>
<tr>
<td></td>
<td>Data Class . . . . . . . (Blank for default data class)</td>
</tr>
<tr>
<td></td>
<td>Storage Class . . . . . (Blank for default storage class)</td>
</tr>
</tbody>
</table>

3 In the **Management Class** field, accept the default (blank) or type the name of an SMS management class that contains the migration and backup information for allocating the data set.

4 In the **Data Class** field, accept the default (blank) or type the name of an SMS data class that contains the SPACE and LRECL information for allocating the data set.

5 In the **Storage Class** field, accept the default (blank) or type the name of an SMS storage class for allocating the data set.
Press **Enter** or **END** to store changes and leave the panel:
- Press **Enter** to continue to the next set of options in the menu sequence.
- Press **END** once to return to the previous panel, or twice to return to the main Options panel.

If you made changes, SQL Explorer stores them in your profile for use in the current session and future sessions.

### Setting data set options

This task describes how to specify options for JCL, SYSPRINT, summary, and SYSIN data sets.

**Note**

If the data sets do not exist, they are created the first time that a particular task requires them. The data sets are created as partitioned if you specified a member name. Otherwise, they are created as sequential.

You can enclose the data set and member name with single quotation marks. If you use quotation marks, the product uses the exact data set and name that you specify. Otherwise, the product attempts to append the data set and member name with the TSO prefix (if you have a prefix set) as the first node.

### To review or change data set options

1. Access the JCL Options panel using the procedure described in “Specifying JCL options in batch mode” on page 48.

2. In the **Options** field, select 2 (for **Data Sets**) and press **Enter**.

Figure 15 on page 56 is displayed.

**Figure 15: Data Set Options panel (PSSPODS2)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Specify options and press ENTER to continue.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Use ### in member name to have counter variable generated into member name.</td>
</tr>
<tr>
<td>Counter</td>
<td>Counter . . . . . . 1 (Replaces ### in member name, then incremented)</td>
</tr>
<tr>
<td>Note:</td>
<td>JCL, SYSPRINT, Summary, and SYSIN data sets can be partitioned or sequential.</td>
</tr>
</tbody>
</table>

**JCL Data Set** . . . . . 'RDAJLG4.SQLXPLR.CNTL(SQLX###)'  
**SYSPRINT Data Set** . . . 'RDAJLG4.SQLXPLR.SYSPRINT(SQLX###)'  
**SYSPRINT SYSOUT** . . . Y (Y/N) SYSPRINT Class . . . *  
**Summary Data Set** . . . 'RDAJLG4.SQLXPLR.SUMMARY(SQLX###)'  
**Summary SYSOUT** . . . Y (Y/N) SYSOUT Class . . . *  
**SYSIN Data Set** . . . . 'RDAJLG4.SQLXPLR.SYSIN'

### Additional Data Set (Update)
3 In the **Counter** field, enter any value from 1 through 9999 to replace the '#####' suffix in the PDS member name.

If needed, SQL Explorer adds leading zeros to fill the four-digit suffix.

4 In the **JCL Data Set** field, type the name of the partitioned or sequential data set to which SQL Explorer writes the JCL and selected options for the batch job.

SQL Explorer creates partitioned data sets if you specify a member name or sequential data sets if you omit the member name. You can enclose the data set and member name with single quotation marks. If you use quotation marks, the product uses the exact data set and name that you specify. Otherwise, the product attempts to append the data set and member name with the TSO prefix (if you have a prefix set) as the first node.

5 Specify options for SYSPRINT data, as follows:
   
   a In the **SYSPRINT Data Set** field, type the name of the partitioned or sequential data set to which SQL Explorer writes SYSPRINT data for the job.
   
   b In the **SYSPRINT SYSOUT** field, type Y to use a SYSOUT class or N to use a SYSPRINT data set.
   
   c In the **SYSOUT Class** field, specify the JES message class for the SYSPRINT data set.

   You can use a wildcard in this field.

6 Specify options for summary data:
   
   a In the **Summary Data Set** field, specify the name of the partitioned or sequential data set to which SQL Explorer writes summary data for the job.
   
   b In the **Summary SYSOUT** field, type Y to use a SYSOUT class or N to use a summary data set.
   
   c In the **SYSOUT Class** field, specify the JES message class for the summary data set.

   You can use a wildcard in this field.
   
   d In the **SYSIN Data Set** field, type the name of the partitioned or sequential data set to which SQL Explorer writes SYSIN data for an online job.

7 Specify options for additional data sets:
   
   a Type any character in the space beside **Additional Data Sets**, and press Enter.
The Additional Data Set Options panel is displayed (Figure 16 on page 58).

**Figure 16: Additional Data Set Options panel (PSSPODS3)**

PSSPODS3 ----------------- Additional Data Set Options ------------------------
Command ===>
Specify options and press ENTER to continue.

NOTE: The Update data set can only be sequential.
The Update data set is needed only for Migrate Access Path Statistics.

Update Data Set . . . . 'RDAJLG4.SQLXPLR.UPDATE'

b In the **Update Data Set** field, type the name of the *sequential* data set to which the access path statistics for the Migrate Access Path Statistics job are written.

**Note**
You can enclose the data set and member name with single quotation marks. If you use quotation marks, the product uses the exact data set and name you specify. Otherwise, the product attempts to append the data set and member name with the TSO prefix (if you have a prefix set) as the first node.

8 Press **Enter** or **END** to store changes and leave the panel:

- Press **Enter** to continue to the next set of options in the menu sequence.
- Press **END** once to return to the previous panel, or twice to return to the main Options panel.

If you made changes, SQL Explorer stores them in your profile for use in the current session and future sessions.

**Related Information**

- "Allocating a data set for a specific job" on page 45

---

**Setting job card options**

This task describes how to specify job cards and additional JCL for batch analysis jobs.

1 Access the JCL Options panel using the procedure described in “Setting JCL options” on page 54.

2 In the **Options** field, select 3 (for **Job Cards**) and press **Enter**.
The Job Card Options panel is displayed (Figure 17 on page 59).

**Figure 17: Job Card Options panel (PSSPOJOB)**

<table>
<thead>
<tr>
<th>PSSPOJOB</th>
<th>----------------------- Job Card Options -----------------------------</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>===&gt;</td>
</tr>
</tbody>
</table>

Specify Job Card information and press ENTER to continue.

**Job Card Information:**

- //RDAKNNP JOB (5213),CLASS=Q,MSGCLASS=X
- /*ROUTE XEQ BMCPLX1
- /*JOBPARM SYSAFF=DB2A

3. In the **Job Card Information** field, specify job cards and other JCL to add to the batch analysis jobs.

4. Press **Enter** or **END** to store changes and leave the panel:

   - Press **Enter** to continue to the next set of options in the menu sequence.
   - Press **END** once to return to the previous panel, or twice to return to the main Options panel.

If you made changes, SQL Explorer stores them in your profile for use in the current session and future sessions.
Explaining and comparing SQL statements

The topics in this part cover aspects of explaining and comparing SQL statements.

Chapters:

- Explaining SQL statements
- Explaining and executing an individual SQL statement
- Explaining and comparing plans, packages, and DBRMs
- Comparing Explain history
Explaining SQL statements

In SQL Explorer, running an Explain job Explains the steps that the DB2 optimizer must take to execute the relevant SQL statements. This chapter presents instructions for explaining an SQL statement in SQL Explorer.

Overview of the Explain function

Given the heavy dynamic SQL requirements that are generated by many applications, being able to Explain a problematic SQL statement quickly and navigate to the DB2 PLAN_TABLE information is imperative.

Explain provides you with the ability to:

■ View the most important information (such as access path data) quickly
■ Display only the Explain data that you are interested in at a given time
■ Compare access paths from different types of Explains
■ Show the number of indexes in order by index access
■ Highlight the indexes that are used in an Explain operation
■ Highlight the key columns that are matched in an Explain operation
■ Show the tables in the order that they were accessed
■ Identify mismatches in data type or length during Explain processing
■ Toggle between various levels of detail and related information while still displaying this data on a single panel
■ Exclude information that you do not need by customizing the report layout for the Explain results
■ Compare the current plan and all saved plans for the Explain package statement
**How Explain works**

Explain is a powerful function that creates a set of unique, historical baseline snapshots that contain key DB2 catalog statistics for an application.

The snapshot information is stored in DB2 tables and includes:

- SQL text
- Dependent objects
- Catalog information that influences the DB2 optimizer
- Access path information

Explain derives access path information from the DB2 plan table. The product puts this information in context with the database and system environments at a particular point in time. By capturing the environmental variables and correlating them to the access path information, Explain provides additional information that helps you to understand the performance of your applications.

When an application has performance problems, you normally look for the access paths in the contents of the last plan table and then look for environmental statistics in the DB2 catalog. Because the catalog is always changing, the most recent statistics differ from the statistics that were applicable when the application was first implemented—and possibly when it was last bound. When the application is implemented again, it might perform poorly due to differences in the environment. In such a case, a dynamic Explain can analyze the most current statistics and access paths that the DB2 optimizer chose.

You could compare the results of this dynamic Explain to a *static* Explain if you had already performed a bind with EXPLAIN(YES). After analyzing the differences between the two Explains, you could perform a *What-If Edit SQL Explain* by editing the SQL and comparing that Explain to the existing dynamic and static Explains.

---

**Note**

SQL Performance for DB2 users also have a *What-If Index* feature that allows them to add, drop, or update statistics on one or more indexes using cloned structures. This feature is available only as part of the SQL Performance solution. For more information, see the *SQL Performance for DB2 User Guide*.

Explain results are based on the environment that existed when the DB2 optimizer selected the access paths. If you specify Y for the HISTORY parameter when you analyze an object in batch, Explain saves the access path information and related catalog statistics to the product history tables for future reporting or for use by the SQL Explorer Compare function. The historical information can be correlated to the access paths that the DB2 optimizer chose. You can use this information to develop additional reports that are designed to find potential problems.
For example, if you want to find applications that are performing table space scans on tables larger than 100 rows, a query against the history tables provides the answer.

Explain can process plans, packages, database request modules (DBRMs), DBRM libraries (DBRMLIBs), or any ad hoc SQL statement. Explain generates a formatted report that includes the basic information from the plan table and catalog statistics, supplemented by text taken from a set of predefined expert rules.

---

**Related Information**

- “Sample history table Explain queries” on page 229

---

**Explain types**

An Explain takes one of the following forms:

- A *dynamic Explain* that asks the DB2 optimizer for an explanation of the access path. This process evaluates access paths, SQL text, and key catalog statistics in real time.

  **Note**

  If you upgrade to DB2 Version 10 or later, you must ensure that your existing plan tables are in Unicode format before running the dynamic Explain process. See your IBM product documentation for information about migrating your plan tables.

  The dynamic Explain process in DB2 Version 10 or later alters the Unicode back-leveled PLAN_TABLE that you specify to include all of the columns that are required by your version of DB2. This alteration ensures that your PLAN_TABLE remains compatible with the DB2 version that you are using. If the PLAN_TABLE has a padded index, automatic ALTERs for the back-level PLAN_TABLE are disabled.

- A *static Explain* from a bind with EXPLAIN(YES). This process reports the access path information that was derived when you performed a bind with EXPLAIN(YES).

- An *Explain Compare* on a static SQL statement from the SQL workload. This process performs a dynamic Explain and a static Explain, matching the STMTNO from the catalog to QUERYNO in the PLAN_TABLE.

- A *What-If Edit SQL Explain*, in which the SQL is edited and a dynamic Explain is performed. You can continue to edit and Explain the SQL as many times as
needed, and you can also compare the resulting access paths to determine the performance impact of an SQL change.

- An **Explain Statement Cache** on a dynamic SQL statement. This process performs an EXPLAIN STMTCACHE STMTID on a dynamic SQL statement that was executed and is still in the statement cache. The access path in use when the statement was executed is retrieved from the statement cache and reported.

- An **Explain Package**, in which the current access path for a static SQL statement is retrieved and reported even if a BIND with EXPLAIN(YES) was not previously performed.

When you perform multiple Explains on a statement or object, the differences are shown in white. An asterisk (*) next to the Explain label indicates that there was a change on that line.

The Explains are listed in order, based on when they were performed, with the most recent Explain at the top of the list. The Explains are compared from the bottom up, such that a difference between Timestamp 1 and Timestamp 2 is indicated by an asterisk on the label for Timestamp 2. The columns where the differences occurred are shown in white.

### Extended Explain

The product enables the user to exploit DB2's Extended Explain facilities.

DB2 has additional plan tables that provide information from the DB2 optimizer. This information aids in understanding each processing step within an SQL statement.

Explain externalizes this information for your use and interpretation. The use of the Extended Explain facilities does not incur any measurable additional cost, because the DB2 optimizer already generates this information to select the best access paths.

### Explain entry points

Several BMC Software products and solutions share the Explain function.

Table 6 on page 67 describes how you can access the Explain function from these products.
### Table 6: Entry points for the Explain function

<table>
<thead>
<tr>
<th>Product</th>
<th>Entry point</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPTUNE and Index Component (SQL Performance)</td>
<td><strong>Explain Interface</strong> option on the product main menu</td>
<td>You can perform a dynamic or static Explain on a plan, DBRM, package, or on a statement from one of these objects.</td>
</tr>
<tr>
<td></td>
<td><strong>SQL Workload Analysis Menu</strong></td>
<td>You can drill down to a statement that was executed and captured by the Data Collector. For static statements, you can run a dynamic, static, or Compare Explain. For dynamic statements, you can run a dynamic Explain.</td>
</tr>
<tr>
<td></td>
<td>Statement Cache</td>
<td>You can drill down to a statement that was executed and still resides in the statement cache. You can run a dynamic Explain or you can display Explain information reflecting the current access path for the cache entry.</td>
</tr>
<tr>
<td><strong>MainView for DB2</strong></td>
<td><strong>Explain Interface</strong> option on the MainView for DB2 - DC Main Menu</td>
<td>You can perform a dynamic or static Explain on a plan, DBRM, package, or on a statement from one of these objects.</td>
</tr>
<tr>
<td></td>
<td><strong>Current Activity</strong> hyperlink on the THDDETL, THDACTV, THHDDBAT, THDSAPD, THDSPAS, THDTSO, and THDWLM views</td>
<td>You can access Explain information for an active SQL statement in a current thread if the MainView for DB2 - Data Collector component is installed.</td>
</tr>
<tr>
<td></td>
<td><strong>CACHE ID</strong> hyperlink on the ASUTIME view</td>
<td>You can access Explain information for a long-running, dynamic SQL statement that is cached in the EDM pool if the MainView for DB2 - Data Collector component is installed.</td>
</tr>
<tr>
<td></td>
<td><strong>EXPLAIN</strong> button on the SQL summary pop-up of the STRAC display</td>
<td>You can access Explain information for a static SQL statement if the MainView for DB2 - Data Collector component is installed.</td>
</tr>
<tr>
<td></td>
<td><strong>EXPLAIN</strong> button on the SQL event pop-up of the DTRAC display</td>
<td>You can access Explain information for a dynamic SQL statement if the MainView for DB2 - Data Collector component is installed.</td>
</tr>
<tr>
<td></td>
<td><strong>EXPLAIN</strong> button on the DUSER display</td>
<td>You can access Explain information for a dynamic SQL statement if the MainView for DB2 - Data Collector component is installed.</td>
</tr>
<tr>
<td></td>
<td><strong>Statement ID</strong> hyperlink on the SCSQLD view</td>
<td>You can access Explain information for a static SQL statement if the MainView for DB2 - Data Collector component is installed.</td>
</tr>
<tr>
<td>SQL Explorer for DB2</td>
<td>SQL Explorer main menu</td>
<td>You can perform a dynamic or static Explain on a plan, DBRM, package, or on a statement from one of these objects.</td>
</tr>
<tr>
<td></td>
<td>SQLX edit macro</td>
<td>You can mark existing SQL text in a source program for dynamic Explain or execution, or type ad hoc SQL text into an edit session and mark it for dynamic Explain or execution.</td>
</tr>
<tr>
<td>Product</td>
<td>Entry point</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>----------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CATALOG MANAGER for DB2</td>
<td>BMCEXPLORE command</td>
<td>You can specify BMCEX on a plan, DBRM, or package for Explain processing.</td>
</tr>
<tr>
<td>Confirm SQL panel</td>
<td></td>
<td>You can perform a dynamic Explain on a single statement by specifying Y for the Analysis option on the Confirm SQL panel. You can access the Confirm SQL panel as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Type ANALYZE on SQL that was saved to CATALOG MANAGER's SQL table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Type DESCRIBE on a DBRM or package, and type GET nnn, where nnn is the statement number.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Type a DML statement on an object in a table list.</td>
</tr>
<tr>
<td>BMC Workbench for DB2</td>
<td>DB2 Navigator tab</td>
<td>You can select a statement to Explain from:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ The DB2 Statement Cache</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ A package, by using the DB2 Object View</td>
</tr>
<tr>
<td>Performance tab</td>
<td></td>
<td>You can manually type in an ad hoc statement to Explain.</td>
</tr>
</tbody>
</table>

You can also perform a dynamic Explain on ad hoc SQL text that you enter, or on a DBRMLIB member or one of its statements. From a list of packages, you can perform an explain package which shows the current access path for static SQL statements even if a BIND with EXPLAIN(YES) has not been done.

Main panels

The main panels for the Explain process are as follows:

- **Explain Object Specification** allows you to specify criteria to retrieve an object or statement to Explain. Use this panel to:
  - Specify whether to list objects that you can select to Explain
  - Specify to Explain the specified objects in batch
  - Specify whether to provide additional Explain options

  **Note**
  The appearance of this panel varies, depending on the route by which you enter the Explain function. For example, you might be asked to specify a subsystem ID for your Explain.
- **Explain Options** is displayed when you select **Explain Options** on the Explain Object Specification panel or from the OPTIONS task bar. Use the Explain Options panel to specify or change Explain processing options.

- **Explain Object Selection List** is displayed when you specify **Processing Mode L, List object(s)**, on the Explain Object Specification panel, or when you execute BMCEX on objects from CATALOG MANAGER. Use the Explain Object Selection List to specify objects for an Explain operation. A task bar and action codes provide more room on this panel for displaying Explain information. You can also use the FIND, RFIND, and ZOOM commands on this panel.

  Because a plan can consist of DBRMs, a package list of collection IDs, or a mixture of both DBRMs and packages, the Object Selection List for plans shows both the DBRMs and package list.

  The appearance of the Explain Object Selection List is controlled by a member in a layout data set.

- **Explain Results** displays the results of online Explain operations, including the following information:

  - The estimated cost (in timerons) of executing the SQL statement
  - The predicates used in the SQL statement
  - Whether a mismatch in data type or length might be causing the DB2 optimizer to choose a bad access path
  - Access path information
  - Expert rules regarding the syntax of the statement, the access path, and current statistics on the objects
  - Statistics from the DB2 catalog for tables referenced in the access path steps

  In addition, indexes referenced in the access path steps are displayed and highlighted with the table statistics, followed by indexes that are available but not referenced in the step. You can drill down further to display table and index columns, and their related statistics.

  The information on the Explain Results panel is divided into the following sections:

  - The **Statement area** is located at the top of the panel, this section shows information such as the type of Explain that was performed, the statement number, the cost in timerons, the statement text, and the cost in service units.
The Access path area is located at the bottom of the panel, this section displays information from the DB2 PLAN_TABLE. You can drill down to table and index columns and statistics for each query block.

A task bar and action codes provide more room on this panel for displaying Explain information. From this panel, you can drill down to get more information, display reports on demand, edit SQL, and perform additional Explains. You can also use the FIND, RFIND, and ZOOM commands on this panel.

The appearance of the Explain Results panel is controlled by the same layout data set that controls the display of the Explain Object Selection List.

Related Information

- “Explaining a package online” on page 85
- “Working with Explain results” on page 91
- "Layout data set" on page 73

Task bar

The Explain Object Selection List and Explain Results panels use a task bar containing the following options.

To display these options, put your cursor on a task bar item, and press Enter.

- **FILE task bar** enables you to manage the data sets used by the Explain operation. Specify one of the following options, and press Enter:
  - OPEN LAYOUT allows you to browse, edit, or change the layout data set, which controls the display of this panel.
  - PRINT writes the Explain results to the ISPF data set. This option is available only on the Explain Results panel.
  - EXIT returns you to the previous panel.

- **COMMANDS task bar** enables you to specify processing options for all objects on the panel.

- **OPTIONS task bar** enables you to specify processing options for the Explain operation or for the JCL used in a batch Explain. Specify one of the following options, and press Enter:
— **EXPLAIN** displays the Explain Options panel.

— **JCL** displays the JCL Options panel.

- **HELP task bar** enables you to get more information about the Explain function.
  Specify one of the following options, and press **Enter**:

  — **ACTIONS** displays information about the Explain action codes.
  
  — **COMMANDS** displays information about the Explain commands.

---

**Related Information**

- "Commands" on page 72

---

**Actions**

You can use the following action codes in your Explain in one of the following ways:

- Type the action code in the space beside an object, and press **Enter**.

- Put the cursor in the space beside an object, and press **Enter**. A list of action codes is displayed. Specify an action code, and press **Enter**.

The action codes are listed on the Explain Object Selection List and the Explain Results panels. To specify an action code for multiple objects, type the action code in the space beside an object, type an equal sign (=) in the space beside any subsequent objects on which you want to perform the same action, and press **Enter**.

Table 7 on page 71 lists the action codes.

**Table 7: Action codes for Explain**

<table>
<thead>
<tr>
<th>Code</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>Hide</td>
<td>Collapses the results of the previous action code</td>
</tr>
</tbody>
</table>
| P    | Predicate | Displays a predicate report  
  Explain automatically displays a predicate report if a statement has a mismatch in data type or length in the predicate. The predicates containing the mismatch are shown in red. |
<table>
<thead>
<tr>
<th>Code</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Rules</td>
<td>Displays severe, warning, and informational rules for BIND parameters, SQL statements, access path steps, and statistics. Severe rules are shown in red, warning rules are shown in yellow, and informational rules are shown in green. To learn more about a rule, type <code>TSO PSSMSG BMC msgNum</code>, where <code>msgNum</code> is the six-digit BMC message number. Do not include the severity code in this number.</td>
</tr>
<tr>
<td>RI</td>
<td>Rules:</td>
<td>Displays informational rules for BIND parameters, SQL statements, access path steps, and statistics</td>
</tr>
<tr>
<td></td>
<td>Informational</td>
<td></td>
</tr>
<tr>
<td>RS</td>
<td>Rules:</td>
<td>Displays severe rules for BIND parameters, SQL statements, access path steps, and statistics</td>
</tr>
<tr>
<td></td>
<td>Severe</td>
<td></td>
</tr>
<tr>
<td>RW</td>
<td>Rules:</td>
<td>Displays warning rules for BIND parameters, SQL statements, access path steps, and statistics</td>
</tr>
<tr>
<td></td>
<td>Warning</td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>Show</td>
<td>Expands a list of statements, shows a formatted version of statement text, or shows statistics or columns for the objects referenced in the access path step</td>
</tr>
<tr>
<td>T</td>
<td>Text Compare</td>
<td>Compares the text of two SQL statements</td>
</tr>
<tr>
<td>W</td>
<td>What-If Edit</td>
<td>Performs a What-If Edit SQL Explain</td>
</tr>
<tr>
<td></td>
<td>SQL</td>
<td></td>
</tr>
<tr>
<td>XB</td>
<td>Explain:</td>
<td>Performs a batch Explain</td>
</tr>
<tr>
<td></td>
<td>Batch</td>
<td></td>
</tr>
<tr>
<td>XC</td>
<td>Explain:</td>
<td>Compares the results of a dynamic Explain with one or more static Explains</td>
</tr>
<tr>
<td></td>
<td>Compare</td>
<td></td>
</tr>
<tr>
<td>XD</td>
<td>Explain:</td>
<td>Performs a dynamic Explain</td>
</tr>
<tr>
<td></td>
<td>Dynamic</td>
<td></td>
</tr>
<tr>
<td>XS</td>
<td>Explain:</td>
<td>Performs a static Explain</td>
</tr>
<tr>
<td></td>
<td>Static</td>
<td></td>
</tr>
<tr>
<td>XP</td>
<td>Explain:</td>
<td>Performs an Explain package, which reports on plan stability rows. <strong>Note:</strong> This option is currently available only with DB2 Version 10 or later.</td>
</tr>
<tr>
<td></td>
<td>Package</td>
<td></td>
</tr>
</tbody>
</table>

**Note**

SQL Performance *for DB2* users have additional action codes for the *What-If Index* feature of the SQL Performance solution. For more information about these solution-specific action codes, see the *SQL Performance for DB2 User Guide*.

## Commands

The following table lists commands that apply to all objects in the Explain Object Selection List or the Explain Results panel.

You can specify a command in one of the following ways:

- Type a command on the **COMMAND** line, and press **Enter**.
Put your cursor on the **COMMANDS** task bar, and press **Enter**. Specify a command, and press **Enter**.

### Table 8: Commands for Explain

<table>
<thead>
<tr>
<th>Commands</th>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>H ALL</td>
<td>Hide All</td>
<td>Collapses the results of all previous action codes and commands</td>
</tr>
<tr>
<td>P ALL</td>
<td>Predicates All</td>
<td>Displays all predicate reports</td>
</tr>
<tr>
<td>R ALL</td>
<td>Rules All</td>
<td>Displays all severe, warning, and informational rules</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Severe rules are shown in red, warning rules are shown in yellow, and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>informational rules are shown in green.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To learn more about a rule, type <strong>TSO PSSMSG BMC messageNumber</strong>, where</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>messageNumber</strong> is the six-digit BMC message number. Do not include the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>severity code in this number.</td>
</tr>
<tr>
<td>RI ALL</td>
<td>Rules:</td>
<td>Displays all informational rules</td>
</tr>
<tr>
<td></td>
<td>Informational</td>
<td>All</td>
</tr>
<tr>
<td>RS ALL</td>
<td>Rules: Severe</td>
<td>Displays all severe rules</td>
</tr>
<tr>
<td>RW ALL</td>
<td>Rules: Warning</td>
<td>Displays all warning rules</td>
</tr>
<tr>
<td>S ALL</td>
<td>Show All</td>
<td>Expands a list of statements for all of the displayed objects, or shows</td>
</tr>
<tr>
<td></td>
<td></td>
<td>formatted versions of all of the statement text, or shows statistics or</td>
</tr>
<tr>
<td></td>
<td></td>
<td>columns for the objects referenced in the access path steps</td>
</tr>
<tr>
<td>XB ALL</td>
<td>Explain: Batch</td>
<td>Performs a batch Explain for all objects in the list</td>
</tr>
</tbody>
</table>

### Layout data set

The appearance of the Explain Object Selection List and the Explain Results panel is controlled by a layout data set member.

The Explain function is shipped with a default layout member (PSSREPA) in the `hlq.SAMP` data set. The DOMCLIST allocates the product `hlq.llqSAMP` data set to the PSSCNTL DD. The product does not allow customization of PSSREPA, but you can copy the default layout to a new data set name and edit that member. You can save one or more layout members.

*Note*

The name of the delivered SAMP data set is `hlq.llqSAMP`. (The runtime library `llq` might use a different low-level qualifier (LLQ), depending on what values were entered during installation.) Place all customized members in the `hlq.UBMCSAMP` data set.

To customize a default layout member:
1 Copy the PSSREPA member to one of the following locations:
   - The userlib data set (hlq.UBMCSAMP)
   - A different data set
     If you copy the member to a different data set, first allocate the target data set (for example, new.dsn) in the DOMCLIST on the PSSCNTL DD:

     **Example**
     ```plaintext
     ALLOC F(PSSCNTL) SHR REU DA('new.dsn' 'hlq.UBMCSAMP' 'hlq.BMCSAMP')
     ```

2 Edit the new data set.

3 Change the name of the member and then reference the new name as the layout data set.

The Layout Data Set panel contains the following options:

- **Edit layout data set** opens the layout data set for editing.
- **Browse layout data set** opens the layout data set for viewing. In this mode, you cannot edit the layout data set.
- **Use default data set instead of layout data set** reverts to using the default data set to display the Explain Object Specification List and the Explain Results panel online.
- **Overwrite layout data set with default data set and edit** replaces the contents of the layout data set with the contents of the default data set and opens the revised layout data set for editing.

The default layout data set contains comments to help you edit your layout data set.

**Related Information**

- “Editing the Explain report layout” on page 93

---

### Generating a list of packages to Explain

Use the following procedure to generate a list of packages to Explain.

1 Access the Explain Object Specification panel.
The method varies depending on the product:

- On the SQL Explorer Main Menu, type 1 (for Explain), change the SSID and DDF locations if necessary, and press Enter.
- On the APPTUNE Main Menu, select Explain Interface and press Enter.
- On the MainView for DB2-DC Main Menu, select Explain Interface and press Enter.

The Explain Object Specification panel is displayed (Figure 18 on page 75).

**Note**

The product recognizes only the information that applies to the object type that you specify. For example, if you specify to Explain a package, any information existing in the plan, DBRM, or DBRMLIB fields is ignored. You do not need to remove any of the unrelated values before proceeding.

---

**Figure 18: Explain Object Specification panel (PSSPA110)**

| PSSPA110 ------------------- Explain Object Specification ------------------- |
|-------------------------------|--------------------------------------------------------------------------------|
| Command ===>| Type . . 2 (1=Plan, 2=Package, 3=DBRM, 4=DBRMLIB, 5=Ad Hoc SQL) |
| Plan: Name DYNSTAPL |
| Package: COLLID C9AN31C Name . . * |
| Version * |
| DBRM: Plan V10LGUPL Name . . * |
| DBRMLIB: (Specify PDS with member name or wildcard member.) |
| DSN . . 'PSS.TESTING.V10.DBRM(V10TEST1)' |
| Processing Mode: L (L=List object(s), B=Batch Explain with specified objects) |

2 In the **Type** field, select **Package**.

3 In the **Package: COLLID** field, type the name of the package collection.

You can use wildcards in this field.

**Tip**

If the information is truncated, place your cursor on the field, and press **F4** or **F6** (ZOOM). The contents of the entire field are displayed in a pop-up window.

4 In the **Package: Name** field, type the name of the package that you want to Explain.
This name is the one used when the package was bound. You can enter a specific name, or use wildcards to display a list and select the package from the list. If the Name information is truncated, place your cursor on the field, and press F4 or F6 (ZOOM). The contents of the entire field are displayed in a pop-up window.

5 In the **Package: Version** field, type the version identifier for the package.

You can type a specific version, use wildcards, or leave the field blank (for packages that do not use a version). If the package that you want to Explain has a version but you leave this field blank, the package does not appear in your results.

**Tip**

BMC recommends using a wildcard because package versions can contain mixed-case characters or timestamps and can be long and difficult to remember.

6 In the **Processing Mode** field, select **List object(s)** to view a list of object specification criteria.

7 In the **Explain Options** field, type any character in the space beside the field, and press **Enter**.

The Explain Options panel is displayed.

8 Specify options for the Explain.

For instructions, see “**Specifying Explain options**” on page 76

The Explain Object Selection List panel is displayed (Figure 19 on page 76).

---

**Specifying Explain options**

Use the following procedure to specify processing options for your Explain.
1. Access the Explain Options panel, as described in “Generating a list of packages to Explain” on page 74.

Figure 20: Explain Options panel (PSSPA020)

PSSPA020 ------------------------ Explain Options ------------------------
Command ===> Specify options and press ENTER to continue.
Press F8 to scroll down and F7 to scroll up.

Rule Set . . . . . . . . . DEFAULT (DEFAULT, APPLDEV, other)

Dynamic Explain:
- Plan Table Owner . . . . USERID (USERID, authid)
- Qualifier Name . . . . . QUAL (QUAL, authid) For unqualified SQL
- Degree . . . . . . . . . (Blank/1/Any)

Static Explain:
- Plan Table Rows . . . . L (L=Last bind time, A=ALL rows)
- Read-Only Application . . N (Y/N)
- Online Application . . . . N (Y/N)
- Acceleration Detail . . . . Y (Y/N)
- Currency Code . . . . . USD (USD, EUR, etc.)
- Cost Translation Rate . . 1.0 Cost per timeron
- . . . . . . . . . . . . . . 1.0 Cost per service unit

Rule Display Option
1 1 BMC message ID, rule severity, and the message text
2 Message text only
3 No rule messages

Column Break . . . . . . . N (Y/N) Format 1 column per line in SQL text
Recall DBRMLIB Data Set . . . Y (Y/N) Restore PDS for DBRM and DBRMLIB

2. In the Rule Set field, type the name of the rule set to use:

- **DEFAULT** contains rules concerning performance issues, primarily geared for DBAs.
- **APPLDEV** contains rules designed especially for application developers.
- **other** is the name of a customized rule set. This value corresponds to the member name of the rule set that you created and should match the RULESET declaration.

You might have several modified versions of the rule set for your installation. For example, you might have a rule set for the accounting application and another for human resources, or you might have one rule set for the test environment and another for the production environment.

3. Select options for a Dynamic Explain, a Static Explain, or an Explain Package:

- For a **Dynamic Explain**, specify the following options shown in Table 9 on page 78.
### Table 9: Options for Dynamic Explain

<table>
<thead>
<tr>
<th>Field</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Plan Table Owner</strong></td>
<td>Type a qualifier for the plan table used by the Explain operation, as follows:</td>
</tr>
<tr>
<td></td>
<td>— USERID uses the job submitter’s user ID for the plan table. Explain reverts to this value if you do not have the authority to perform the SET CURRENT SQLID command.</td>
</tr>
<tr>
<td></td>
<td>— authID uses authID.Plan_Table, where authID can be any valid DB2 qualifier.</td>
</tr>
<tr>
<td></td>
<td>— OWNER specifies the owner of the package or DBRM</td>
</tr>
<tr>
<td><strong>Tip:</strong> BMC recommends that you point to BMC or empty plan tables. If your plan table has a lot of rows, you can avoid performance problems by adding the recommended indexes in the DAADB2IX member in the SAMP library.</td>
<td></td>
</tr>
<tr>
<td><strong>Qualifier Name</strong></td>
<td>Specify a qualifier for objects and tables in the Explain operation, as follows:</td>
</tr>
<tr>
<td></td>
<td>— QUAL uses the Qualifier Name provided at bind time. This value is not valid for DBRMLIB or ad hoc SQL text explanation.</td>
</tr>
<tr>
<td></td>
<td>— authID is any valid DB2 qualifier used to qualify any unqualified objects.</td>
</tr>
<tr>
<td><strong>Note:</strong> For online Explains, the product updates this field with the qualifier of a bound object. For batch Explains, the product uses the value you specify here.</td>
<td></td>
</tr>
<tr>
<td><strong>Degree</strong></td>
<td>Specify whether the Explain operation should be run using parallel processing to maximize performance, as follows:</td>
</tr>
<tr>
<td></td>
<td>— blank issues a SET CURRENT DEGREE command to the value that the object was bound to before running the dynamic Explain. This value is not valid for DBRMLIB or ad hoc SQL text explanation. (default)</td>
</tr>
<tr>
<td></td>
<td>— 1 issues a SET CURRENT DEGREE = 1 command before running the dynamic Explain.</td>
</tr>
<tr>
<td></td>
<td>— Any issues a SET CURRENT DEGREE = ANY command before running the dynamic Explain.</td>
</tr>
<tr>
<td><strong>Note:</strong> For a Static Explain, specify a Plan Table Rows value:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>— Last bind time uses the most recent rows in the owner.Plan_Table from the last time you ran a bind with EXPLAIN(YES) for the object that you want to Explain.</td>
</tr>
</tbody>
</table>

---

For a Static Explain, specify a Plan Table Rows value:
—**ALL rows** uses all of the rows in the *owner*.Plan_Table for the object that you want to Explain.

- For an **Explain Package**, use the **Plan Table Rows** value to specify the package versions that you want to display:

  — **Last bind time** displays only the current copy of the package that you want to Explain.

  — **ALL rows** displays current, previous, and original copies of the package that you want to Explain.

  **Note**
  
  If the last bind on the object did not specify EXPLAIN(YES), no static Explain information is retrieved from the PLAN_TABLE.

4 In the **Read-Only Application** field, specify whether the access intent is read only and no data manipulation is involved:

- Type **Y** to trigger a set of specific DB2 consideration rules.

- Type **N** if your application inserts, updates, or deletes data.

5 In the **Online Application** field, specify whether to trigger specific rules for table space scan, list and sequential prefetch, and multiple index access paths (MIAP) considerations:

- Type **Y** if the application usage is online.

- Type **N** if the application usage is batch.

6 In the **Acceleration Detail** field, specify whether to report the SQL’s eligibility to run on an IBM DB2 Analytics Accelerator (DAA):

- Type **Y** (the default) to show the SQL’s eligibility to run on a DAA.

- Type **N** to omit the SQL’s eligibility to run on a DAA.

A BMC message rule indicates the name of the accelerator, if applicable.

  **Note**
  
  If you have enabled Accelerator Modeling (via the **ACCELMODEL** system parameter), this option supports Accelerator Modeling and actual accelerators.

Depending on the parameters that you set, SQL Explorer displays the applicable BMC message rules that are listed here:
If you enabled the DAA, SQL Explorer displays this message rule:

BMC184401I-The query will be sent to the accelerator.

If, in addition, you set the Acceleration Detail parameter to Y, SQL Explorer displays one of the following message rules:

BMC184397I-The query will be sent to Accelerator acceleratorServerName acceleratorServerLocation.

BMC184398I-This query is not eligible for Acceleration. Reason: reasonCode-queryText

Values for the variables correspond to the REASON_CODE and QI_DATA columns of the DSN_QUERYINFO_TABLE.

If Accelerator Modeling is turned on, SQL Explorer displays one of the following message rules:

BMC184399I-ACCELMODEL {ELIGIBLE | INELIGIBLE}

BMC184398I-This query is not eligible for Acceleration. Reason: reasonCode-queryText

The product's Common Explain component retrieves this information from the following tables:

- DSN_STATEMNT_TABLE
- DSN_QUERYINFO_TABLE
- PLAN_TABLE

If these tables do not exist at the time of the Explain, Common Explain creates them.

For more information, view the Quick Course "APPTUNE for DB2—DB2 Analytic Accelerator."

7 Specify a three-digit **Currency Code** for the monetary unit used to calculate the cost translation rate.

**Best practice**

You can use any characters except spaces in this field, though BMC Software recommends that you use ISO 4217 standard codes for the representation of currencies. The default is USD.

8 Specify the **Cost Translation Rate** used to translate the estimated processor cost of executing an SQL statement into monetary units, as follows:

- Specify the rate used to translate the timeron cost into a monetary unit, in the form of currency/timerons, where currency is the value you specified for Currency Code.
Specify the rate used to translate the service unit cost into a monetary unit, in the form of currency/service units, where currency is the value you specified for Currency Code.

9 Specify the Rule Display Option, as follows:

- BMC message ID, rule severity, and the message text causes the Explain output to contain only the BMC message ID, the message severity level, and the descriptive text of the message.

  Example
  
  BMC184028I-The tables referenced in the FROM clause are fully qualified. This will limit the flexibility for the qualifier to be resolved at bind time.

- Message text only causes the Explain output to contain only the descriptive text of the message.

  Example
  
  The tables referenced in the FROM clause are fully qualified. This will limit the flexibility for the qualifier to be resolved at bind time.

- No rule messages generally causes a batch or online Explain to bypass rule message processing. Because this option bypasses the rule message processing, no rules are listed in the batch or online output.

  However, if you specify this option for an online Explain in conjunction with one of the rule-related action codes shown in “Actions” on page 71, the product overrides this option and displays the rules associated with the specified action.

10 Specify whether to insert a Column Break between columns listed as part of the SELECT, INSERT, DELETE, or UPDATE clause in the SQL text:

- Y formats the SQL text with a single column per line.

- N formats the SQL text with multiple columns per line.

11 In the Recall DBRMLIB Data Set field, specify whether to restore the partitioned data set (PDS) when performing an Explain on a DBRM or DBRMLIB.

12 Depending on whether you want to select objects to Explain or use the specified object, take one of the following actions:
### Specifying batch Explain options

Use the following procedure to specify processing options for your batch Explain operation.

**To specify batch Explain options**

1. Access the Explain Options panel (Figure 21 on page 82).

   (For instructions, see “Generating a list of packages to Explain” on page 74).

   **Figure 21: Explain Options panel–Batch Explain Options (PSSPA020)**

<table>
<thead>
<tr>
<th>Explain type</th>
<th>Action</th>
</tr>
</thead>
</table>
   | Online       | 1 Press **Enter**.  
The Explain Object Selection List is displayed.  
2 Proceed to “Explaining a package online” on page 85. |
   | Batch        | 1 Press **F8** to scroll down to the batch Explain options.  
2 Specify options for a batch Explain. For instructions, see “Specifying batch Explain options” on page 82. |

   **Related Information**
   - “Customizing expert rules” on page 36
   - “Actions” on page 71

2. In the **Explain INSERT Statements** field, select one of the following options:
Y Explains the INSERT statements.
N skips the INSERT statements.

3 Specify the **Rule Message Level** to display in the report:
- **All** includes Informational, Warning, and Severe messages in the report. (default)
- **Severe** flags only the statements and situations that might cause severe performance problems, such as a SELECT * statement.
- **Warning** flags statements that can be improved to enhance performance.
- **Informational** includes messages that provide information that might help you make changes to improve performance.

The Column Functions (MIN/MAX) on Composite index can take advantage of One-Fetch access when there exists a matching predicated on the high order column other than a LIKE predicate.

4 In the **Suppress Statement Detail When No Rules Fire** field, specify whether to suppress detail in batch Explain reports. This option is used in conjunction with **Rule Message Level**. It does not affect the analysis object summary information nor the summary report.

- Y suppresses the detail in a batch Explain report when no rules are issued, based on the **Rule Message Level** you specify.
- N displays the detail in a batch Explain report, based on the **Rule Message Level** you specify. (default)

Table 10 on page 83 lists the results of various combinations of these options.

**Table 10: Results of Suppress Statement Detail option**

<table>
<thead>
<tr>
<th>Rule Message Level is</th>
<th>and</th>
<th>and Suppress Statement Detail is either</th>
</tr>
</thead>
<tbody>
<tr>
<td>Severe</td>
<td>Severe rules are issued</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>No severe rules are issued</td>
<td>All batch report sections and severe rules are displayed.</td>
</tr>
<tr>
<td>Rule Message Level is</td>
<td>and</td>
<td>and Suppress Statement Detail is either</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Y</td>
</tr>
<tr>
<td>Warning</td>
<td>Both warning and severe rules are issued</td>
<td>All batch report sections and warning and severe rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>Neither warning nor severe rules are issued</td>
<td>No batch report sections nor rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>Only severe rules are issued</td>
<td>All batch report sections and only severe rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>Only warning rules are issued</td>
<td>All batch report sections and only warning rules are displayed.</td>
</tr>
<tr>
<td>Informational</td>
<td>Informational, warning, and severe rules are issued</td>
<td>All batch report sections and severe, warning, and informational rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>Only informational and warning rules are issued</td>
<td>All batch report sections and only warning and informational rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>Only informational rules are issued</td>
<td>All batch report sections and only informational rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>No informational, warning, or severe rules are issued</td>
<td>All batch report sections are displayed, but no rules are shown.</td>
</tr>
<tr>
<td></td>
<td>Only warning rules are issued</td>
<td>All batch report sections and only warning rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>Only severe rules are issued</td>
<td>All batch report sections and only severe rules are displayed.</td>
</tr>
<tr>
<td></td>
<td>Only warning and severe rules are issued</td>
<td>All batch report sections and only warning and severe rules are displayed.</td>
</tr>
</tbody>
</table>

5 In the **Maintain History** field, specify whether to create and save a history of the Explain operation on the object for later use in the Compare process.

**Note**

If you are explaining a DBRMLIB, an ad hoc SQL statement, or by statement only, Explain ignores this option, regardless of what you specify.

6 In the **Histories Retained** field, specify the number of history records to retain for the object, for later use in the Compare process.

When the specified number of histories is reached, the oldest entries are dropped and the newest entries are retained. The range for this option is from 0 to 99.
Note
You must specify Y for the Maintain History option to enable this option. A value of 0 deletes all existing history rows for the object, or saves none.

7 Select a Summary Report Sort Order:

- **Object, Statement** shows the report sequenced by object and statement.
- **Cost (desc)** shows the report sequenced by cost in descending order.
- **Object, Cost (desc)** shows the report sequenced by object and cost in descending order.

This report appears at the top of the output, and contains one line for each object and statement, and information such as counts for scans, joins, and so on.

8 In the **Generate Report Title** field, specify whether to include a report title in the output.

9 Type a **Title** for the report, and press **Enter**.

Note
You must specify Y for the **Generate Report Title** field to enable this option.

Related Information

- “Generating a list of packages to Explain” on page 74

---

Explaining a package online

Use the following procedure to perform an online Explain on a package.

**To Explain a package online**

1 Generate a list of packages to Explain.

   For instructions see “Generating a list of packages to Explain” on page 74. Be sure to specify **List object(s)** for the **Processing Mode**.

2 Specify options for the Explain.

   For instructions see “Specifying Explain options” on page 76.
On the Explain Object Selection List (Figure 22 on page 86), specify the action that you want to perform on an object.

**Figure 22: Explain Object Selection List—Dynamic Explain of a Statement (PSSPE100)**

<table>
<thead>
<tr>
<th>Actions: S H R RS RW RI XD XS XC XB XP</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEMOCJ TRF1AFSV RDAMCG2 Y Y N B S</td>
</tr>
<tr>
<td>DEMOCJ TRF1BEFD RDAMCG2 Y Y N B S</td>
</tr>
<tr>
<td>DEMOCJ TRF1BEFI RDAMCG2 Y Y N B S</td>
</tr>
<tr>
<td>DEMOCJ TRF1BEFU RDAMCG2 Y Y N B S</td>
</tr>
<tr>
<td>DEMOCJ TRF1XX RDAMCG2 Y Y N B S</td>
</tr>
<tr>
<td>DEMOCOLT DEM0PKG1 V1.02 RDAJLG4 Y Y Y R S</td>
</tr>
</tbody>
</table>

**To do this** | **Perform this action**
--- | ---
Display severe, warning, and informational rules for the package, based on the bind information | Type R in the space beside a package.
Display the statements for the package | Type S in the space beside a package.
Display a formatted version of the statement text | Type S in the space beside a statement.
Display severe, warning, and informational rules for the statement, based on SQL syntax | Type R in the space beside a statement.
Perform a dynamic Explain on the statement | Type XD in the space beside a statement.
Perform a static Explain on the statement | Type XS in the space beside a statement. 
Perform an Explain Compare on a statement to compare a dynamic Explain to a static Explain | Type XC in the space beside a statement.

To find a specific string of text, type **FIND string** on the **COMMAND** line, and press **Enter**. Press **F5** to move the cursor to a subsequent instance of the specified text.

**Tip**
If the information is truncated, place your cursor on the field, and press **F4** or **F6** (ZOOM). The contents of the entire field are displayed in a pop-up window.
Explaining a package in batch

This task describes how you can perform a batch Explain on a package.

1 Generate a list of packages to Explain.

For instructions, see “Generating a list of packages to Explain” on page 74.

2 Specify options for the Explain.

For instructions, see “Specifying Explain options” on page 76 and “Specifying batch Explain options” on page 82.

3 In the space beside an object or statement, type XB, and press Enter.

The Batch Explain Type panel is displayed (Figure 23 on page 87).

**Figure 23: Batch Explain Type panel (PSSPA116)**

<table>
<thead>
<tr>
<th>Explain Type . . .</th>
<th>1. XD=Explain Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2. XS=Explain Static from bind with EXPLAIN(YES)</td>
</tr>
<tr>
<td></td>
<td>3. XC=Explain Compare Dynamic and Static Explains</td>
</tr>
<tr>
<td></td>
<td>4. XP=Explain Package</td>
</tr>
</tbody>
</table>

4 Specify the batch Explain Type as follows:

- XD performs a dynamic Explain.
- XS performs a static Explain.
- XC compares a dynamic Explain and a static Explain.
- XP compares the current plan and saved plans for the Explain package statement.
5 Press Enter.

The Batch Job panel is displayed.

6 Specify options for the batch Explain.

For instructions, see “Specifying JCL options in batch mode” on page 48. You can also change the layout data set using the Data Set Options panel.

**Note**

If any of the data sets do not exist, the Allocate Data Set panel is displayed. For instructions, see “Allocating a data set for a specific job” on page 45.

7 *(optional)* To customize the display of your batch Explain, follow these steps:

a In the Set JCL Options field, type Y, and press Enter.

The JCL Options panel is displayed (Figure 24 on page 88).

**Figure 24: JCL Options panel (PSSPOJ00)**

```
PSSPOJ00 ------------------------- JCL Options --------------------------------
Command ===>
Specify one of the options and press ENTER to continue.
Options
  1. SMS Classes
  2. Data Sets
  3. Job Cards
```

b In the Options field, select Data Sets, and press Enter.

The Data Set Options panel is displayed (Figure 25 on page 88).

**Figure 25: Data Set Options panel (PSSPODS1)**

```
PSSPODS1 ----------------------- Data Set Options -----------------------------
Command ===>
Specify options and press ENTER to continue.
Use #### in member name to have counter variable generated into member name.
Counter . . . . . . . . 13   (Replaces #### in member name, then incremented)
Note: The JCL, SYSPRINT, and Summary data sets can be partitioned or sequential. The Layout data set must be partitioned with a member name specified.
JCL Data Set ....... 'RDAKNN.SQLXPLR.CNTL(SQLX####)'
SYSPRINT Data Set .... 'RDAKNN.SQLXPLR.SYSPRINT(SQLX####)'
SYSPRINT SYSOUT .... Y (Y/N)  SYSPRINT Class . . . *
Summary Data Set .... 'RDAKNN.SQLXPLR.SUMMARY(SQLX####)'
Summary SYSOUT ....... Y (Y/N)  Summary Class . . . *
Layout Data Set ....... 'BMCPERF.BBSAMP'
```

c Verify that the options on the panel are correct, and press Enter.
The Job Card Options panel is displayed.

d Specify job card options as needed, and press Enter.

For more information see “Setting job card options” on page 58.

The JCL Options panel is displayed.

8 Press F3 to display the generated job.

9 Review and submit the batch Explain job.

---

**Related Information**

- “Specifying JCL options in batch mode” on page 48
- “Allocating a data set for a specific job” on page 45

---

**Explaining an ad hoc SQL statement**

Use the following procedure to Explain an ad hoc SQL statement.

1 Access the Explain Object Specification panel. The method varies depending on the product.

- On the SQL Explorer main menu, select Explain. Review the SSID and DDF Location, revise if necessary, and press Enter.

- On the APPTUNE Main Menu, select Explain Interface and press Enter.

- On the MainView for DB2 - DC Main Menu, select Explain Interface and press Enter.

The Explain Object Specification panel is displayed (Figure 26 on page 90).
**Note**

Explain recognizes only the information that applies to the object Type that you specify. For example, if you specify to Explain a package, any information that exists in the plan, DBRM, or DBRMLIB fields is ignored. You do not need to remove any of the unrelated values before performing the Explain.

---

**Figure 26: Explain Object Specification panel (PSSPA110)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Explain Object Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>5 (1=Plan, 2=Package, 3=DBRM, 4=DBRMLIB, 5=Ad Hoc SQL)</td>
</tr>
<tr>
<td>Plan:</td>
<td>Name</td>
</tr>
<tr>
<td>Package:</td>
<td>COLLID DEMO% Name % Version %</td>
</tr>
<tr>
<td>DBRM:</td>
<td>Plan Name .</td>
</tr>
<tr>
<td>DBRMLIB:</td>
<td>(Specify PDS with member name or wildcard member.) DSN .</td>
</tr>
<tr>
<td>Processing Mode:</td>
<td>L (L=List object(s), B=Batch Explain with specified objects)</td>
</tr>
</tbody>
</table>

---

2. In the **Type** field, select **Ad Hoc SQL**, and press **Enter**.

An edit session is displayed (**Figure 27 on page 90**).

---

**Figure 27: Ad hoc SQL Explain edit session**

<table>
<thead>
<tr>
<th>Command</th>
<th>EXPLAIN</th>
<th>Columns 00001 00072</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISREDDE2</td>
<td>SELECT * FROM SYSTABLES</td>
<td></td>
</tr>
</tbody>
</table>

---

3. In the edit window, type an SQL statement, and press **F3**.

The Explain or Execute Parameters panel is displayed (**Figure 28 on page 90**).

---

**Figure 28: Explain or Execute Parameters panel (PSSPA117)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Explain or Execute Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option</td>
<td>1. Explain 2. Execute 3. Edit</td>
</tr>
<tr>
<td>Qualifier Name</td>
<td>SYSIBM</td>
</tr>
</tbody>
</table>

---

4. In the **Option** field, select **Explain**.
5 Specify a **Qualifier Name**, and press **Enter**.

The Explain Results panel is displayed (Figure 29 on page 91).

**Figure 29: Explain Results panel (PSSPE200)**

<table>
<thead>
<tr>
<th>FILE</th>
<th>COMMANDS</th>
<th>OPTIONS</th>
<th>HELP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSSPE200</td>
<td>Explain Results for SQLTEXT</td>
<td>Scroll ===&gt;</td>
<td>CSR</td>
</tr>
</tbody>
</table>

More: >

Actions: S H R RS RW RI XD XS W P T

LBL STMTNO COST*RATE SQL-STATEMENT

XD01 13489.976562 SELECT * FROM SYSTABLES ;

Xd01 13489.976 1 1 0 SELECT 0 I 0 N SYSIBM SYSTABLES

************************************************ Bottom of Data ************************************************

6 Specify options on the Explain Results panel.

For instructions, see “**Working with Explain results**” on page 91.

**Note**

For a SQL Performance installation, the identifier of this panel is PSSPW200. In this situation, additional action codes are available only as part of the SQL Performance solution.

7 Review the results of your Explain.

**Working with Explain results**

Use the following procedure to specify Explain results options on the Explain Results panel.

**Figure 30: Explain Results panel (PSSPE200)**

<table>
<thead>
<tr>
<th>FILE</th>
<th>COMMANDS</th>
<th>OPTIONS</th>
<th>HELP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSSPE200</td>
<td>Explain Results for PACKAGE DEMOCOLT.DEMOPKG1.V1.02</td>
<td>Scroll ===&gt;</td>
<td>CSR</td>
</tr>
</tbody>
</table>

More: >

Actions: S H R RS RW RI XD XS W P T

LBL STMTNO COST*RATE SQL-STATEMENT

__ XD01 228 20.621376 DECLARE DEMOCUR1 CURSOR WITH HOLD FOR SELECT SN ,
__ SN EQUAL NONCOLEXP N N N 0.04
__ PN EQUAL NONCOLEXP N N N 0.04
__ JN EQUAL VALUE N Y N N 0.04
__ COST*RATE QB PL MIX QTYPE METH ACC MTCX IX TCREATOR TNAME
__ XD01 20.621376 1 1 0 SELECT 0 I 0 N SPJ SPJNX01

************************************************ Bottom of Data ************************************************
Note
For a SQL Performance installation, the identifier of this panel is PSSPW200. Also, additional action codes are available only as part of the SQL Performance solution.

Tip
- If the Explain results panel information is truncated, place your cursor on the field, and press F4 or F6 (ZOOM). The contents of the entire field are displayed in a pop-up window.
- To find a specific string of text, type FIND string on the COMMAND line, and press Enter. Press F5 to move the cursor to a subsequent instance of the specified text.

To specify Explain results

1. On the Explain Results panel, specify the action that you want to perform and press Enter:

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Perform this action...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collapse the results of that previous action code</td>
<td>Type H or the previous action code in the space beside an object.</td>
</tr>
<tr>
<td>Display a formatted version of the statement text</td>
<td>Type S in the space beside a statement.</td>
</tr>
<tr>
<td>Display the predicate report for the statement</td>
<td>Type P in the space beside a statement.</td>
</tr>
<tr>
<td>Display severe, warning, and informational path rules</td>
<td>Type R in the access path area.</td>
</tr>
<tr>
<td></td>
<td>For the query block line, these rules are based on DB2 PLAN_TABLE information.</td>
</tr>
<tr>
<td></td>
<td>On the table line, these rules are based on the table and index statistics.</td>
</tr>
<tr>
<td>Display table and index statistics for the referenced</td>
<td>Type S in the access path area.</td>
</tr>
<tr>
<td>objects in the Explain</td>
<td>All indexes for the table are shown. The referenced index appears first and is</td>
</tr>
<tr>
<td></td>
<td>highlighted.</td>
</tr>
<tr>
<td>Display column statistics</td>
<td>Type S on a table in the access path area.</td>
</tr>
<tr>
<td>Display key columns</td>
<td>Type S on an index in the access path area.</td>
</tr>
<tr>
<td></td>
<td>The referenced key columns are displayed in white.</td>
</tr>
<tr>
<td>Display severe, warning, and informational rules,</td>
<td>Type R on a table in the access path area.</td>
</tr>
<tr>
<td>based on the table and index statistics</td>
<td></td>
</tr>
<tr>
<td>If you want to...</td>
<td>Perform this action...</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Perform a What-If Edit SQL Explain</td>
<td>Type <strong>W</strong> on a statement. Depending on the type of statement you choose, one of the following events occurs:</td>
</tr>
<tr>
<td></td>
<td>■ <strong>XD</strong>, <strong>XP</strong> or <strong>XS</strong> opens an edit session for the SQL text as it currently appears in the catalog.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>W</strong> opens an edit session for the SQL text as it appeared in the selected statement.</td>
</tr>
</tbody>
</table>

2. Repeat Step 1 on page 92 until you have finished examining the Explain results.

### Editing the Explain report layout

Use this procedure to customize the layout of the Explain Object Selection List or the Explain Results panel.

1. Put your cursor on the **FILE** task bar, and press **Enter**.

   A list of **FILE** options is displayed.

2. Select **Open Layout** and press **Enter**.

   The Layout Data Set Panel is displayed (Figure 31 on page 93).

   **Figure 31: Layout Data Set panel (PSSPLODS)**

   PSSPLODS ----------------------- Layout Data Set ------------------------------
   Command ===> 
   Layout Data Set 'ROADAC.SQLXPLR.CNTL(PSSREPA)'
   Default Data Set 'BMCPERF.BMCSAMP(PSSREPA)' (Used if layout data set is blank)
   Option  1. Edit layout data set  
            2. Browse layout data set  
            3. Use default data set instead of layout data set  
            4. Overwrite layout data set with default data set and edit

3. Type the name of the **Layout Data Set** used to control the appearance of the Explain report.

4. In the **Option** field, specify **Overwrite layout data set with default data set and edit** and press **Enter**.
The product replaces the contents of the layout data set with the contents of the default data set and opens the revised layout data set for editing (Figure 32 on page 94).

Figure 32: Example of a layout data set

File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT RDADAC.SQLXR.CNTRL(PSSREPA) - 01.31 Columns 00001
000072
Command ===> Scroll ===> CSR
000086 [PACKAGE]
000087 * THIS IS INFORMATION FROM SYSPACKAGE.
000088 *WIDTH SHOW COL_NAME COLUMN HEADING
000089 16 S COLLID
000090 8 S NAME
000091 8 S VERSION VERS
000092 8 S OWNER
000093 1 S VALID
000094 1 S OPERATIVE OPER
000095 1 S REOPTVAR REOPT
000096 1 S VALIDATE
000097 1 S ISOLATION
000098 1 S RELEASE
000099 1 S EXPLAIN
000100 1 S QUOTE
000101 1 S COMMA

You can perform the following actions in the various sections of the layout data set. The changes you make here apply to the Explain Object Selection List or the Explain Results panel.

- Specify a new value for the WIDTH of a column.
  If the display width of a field is too short to show all of the values in the field, one of the following events occurs:
  - For numbers, the product attempts to show the number up to the decimal point. If this is not possible, asterisks (*) are used to fill the entire field.
  - For characters, the product truncates the character text from the right.

  **Note**
  If the information is truncated, place your cursor on the field, and press F4 or F6 (ZOOM). The contents of the entire field are displayed in a pop-up window.

- Hide a column by typing H beside a COL_NAME in the SHOW column.

- Display an existing column with a different name by specifying a new COLUMN HEADING.

  **Note**
  You cannot change the actual names of the columns.

- Change the order in which the columns are displayed by moving a row within its section.

6 Press F3 to save your changes.
Explain report fields

The Explain Report Field labels table lists the fields that are on the Explain reports in alphabetic order by field label name. Each field is shown with the DB2 column name, the DB2 table or other source from which it comes, and the section of the Explain report in which it can be found.

The field labels are controlled by the hlq.IICsAMP (PSSREPA) layout data set member. The labels shown below are the default labels that are shipped with SQL Explorer. For information about customizing the layout of your reports, see “Editing the Explain report layout” on page 93.

Table 11: Explain Report field labels (from PSSREPA layout data set)

<table>
<thead>
<tr>
<th>Report field label</th>
<th>DB2 column name</th>
<th>DB2 table or other source</th>
<th>Report section</th>
</tr>
</thead>
<tbody>
<tr>
<td>1STKEY</td>
<td>FIRSTKEYCARDF</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>ACC</td>
<td>ACCESSSTYPE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>ADEG</td>
<td>ACCESS_DEGREE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>APG</td>
<td>ACCESS_PGROUP_ID</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>BIND_EXPLAIN_ONLY</td>
<td>BIND_EXPLAIN_ONLY</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>BIND_TIME</td>
<td>BIND_TIME</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>BINDTIME</td>
<td>BINDTIME</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>BINDTIME</td>
<td>BINDTIME</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>BPOOL</td>
<td>BPOOL</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>BPOOL</td>
<td>BPOOL</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>CARD</td>
<td>COLCARDF</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>CARD</td>
<td>CARDF</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>CAT</td>
<td>COST_CATEGORY</td>
<td>DSN_STATEMNT_TABLE</td>
<td>STMT_COST</td>
</tr>
<tr>
<td>CFE</td>
<td>COLUMN_FN_EVAL</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>CHARSET</td>
<td>CHARSET</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>CHARSET</td>
<td>CHARSET</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>CHARSET</td>
<td>CHARSET</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>CHECKFLAG</td>
<td>CHECKFLAG</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>CHILD</td>
<td>CHILDREN</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>CLOSERULE</td>
<td>CLOSERULE</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>CLUSTERED</td>
<td>CLUSTERED</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>Report field label</td>
<td>DB2 column name</td>
<td>DB2 table or other source</td>
<td>Report section</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>--------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>CLUSTERING</td>
<td>CLUSTERING</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>CLUSTERRATIO</td>
<td>CLUSTERRATIOF</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>COLLID</td>
<td>COLLID</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>COLLID</td>
<td>COLLID</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>COLNO</td>
<td>COLNO</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>COLNO</td>
<td>COLNO</td>
<td>SYSKEYS</td>
<td>KEYCOLS</td>
</tr>
<tr>
<td>COLUMN NAME</td>
<td>NAME</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>COMMA</td>
<td>COMMA</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>COMMA</td>
<td>COMMA</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>COMMA</td>
<td>COMMA</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>COMP</td>
<td>PCTROWCOMP</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>COMPRESS</td>
<td>COMPRESS</td>
<td>SYSTABLEPART</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>CONTOKEN</td>
<td>CONTOKEN</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>CONTOKEN</td>
<td>CONTOKEN</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>CORR</td>
<td>CORRELATION_NAME</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>COST*RATE</td>
<td>COMP*COST</td>
<td>DSN_DETCOST_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>COST*RATE</td>
<td>COST</td>
<td>DSN_STATEMNT_TABLE</td>
<td>STMT_COST</td>
</tr>
<tr>
<td>CREATOR</td>
<td>CREATOR</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>CREATOR</td>
<td>CREATOR</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>CREATOR</td>
<td>CREATOR</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>CREATOR</td>
<td>CREATOR</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>CREATOR</td>
<td>CREATOR</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>CREATOR</td>
<td>CREATOR</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>CU</td>
<td>SORTC_UNIQ</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>DATABASE</td>
<td>DBNAME</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>DBPROTOCOL</td>
<td>DBPROTOCOL</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>DBPROTOCOL</td>
<td>DBPROTOCOL</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>DEC31</td>
<td>DEC31</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>DEC31</td>
<td>DEC31</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>DEC31</td>
<td>DEC31</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>DEC31</td>
<td>DEC31</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>Report field label</td>
<td>DB2 column name</td>
<td>DB2 table or other source</td>
<td>Report section</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>---------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>DEFERPREP</td>
<td>DEFERPREP</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>DEFERPREP</td>
<td>DEFERPREP</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>DEFERPREP</td>
<td>DEFERPREP</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>DEFERPREPARE</td>
<td>DEFERPREPARE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>DEGREE</td>
<td>DEGREE</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>DEGREE</td>
<td>DEGREE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>DEGREE</td>
<td>DEGREE</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>DYNAMICRULES</td>
<td>DYNAMICRULES</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>DYNAMICRULES</td>
<td>DYNAMICRULES</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>DYNAMICRULES</td>
<td>DYNAMICRULES</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>EDPROC</td>
<td>EDPROC</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>ENCODING_CCSID</td>
<td>ENCODING_CCSID</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>ENCODING_CCSID</td>
<td>ENCODING_CCSID</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>ENCODING_CCSID</td>
<td>ENCODING_CCSID</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>EXPLAIN</td>
<td>EXPLAIN</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>EXPLAIN</td>
<td>EXPLAIN</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>EXPLAIN</td>
<td>EXPLAIN</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>EXPLAIN_TIME</td>
<td>EXPLAIN_TIME</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>EXPREDICATE</td>
<td>EXPREDICATE</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>EXPREDICATE</td>
<td>EXPREDICATE</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>FARINDREF</td>
<td>FARINDREF</td>
<td>SYSTABLEPART</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>FAROFF</td>
<td>FAROFFPOSF</td>
<td>SYSINDEXPART</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>FILTER</td>
<td>FILTER_FACTOR</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>FREEPAGE</td>
<td>FREEPAGE</td>
<td>SYSTABLEPART</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>FULLKEY</td>
<td>FULLKEYCARDDF</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>FUNCTIONTS</td>
<td>FUNCTIONTS</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>FUNCTIONTS</td>
<td>FUNCTIONTS</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>FUNCTIONTS</td>
<td>FUNCTIONTS</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>G</td>
<td>SORTN_GROUPBY</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>G</td>
<td>SORTC_GROUPBY</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>GME</td>
<td>GROUP_MEMBER</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>GROUP_MEMBER</td>
<td>GROUP_MEMBER</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>Report field label</td>
<td>DB2 column name</td>
<td>DB2 table or other source</td>
<td>Report section</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>--------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>GROUP_MEMBER</td>
<td>GROUP_MEMBER</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>GROUP_MEMBER</td>
<td>GROUP_MEMBER</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>HI2KEY</td>
<td>HI2KEY</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>HOSTLANG</td>
<td>HOSTLANG</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>HOSTLANG</td>
<td>HOSTLANG</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>IMMEDWRITE</td>
<td>IMMEDWRITE</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>IMMEDWRITE</td>
<td>IMMEDWRITE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>IMMEDWRITE</td>
<td>IMMEDWRITE</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>ISOLATION</td>
<td>ISOLATION</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>IX</td>
<td>INDEXONLY</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>IXCREATOR</td>
<td>ACCESSCREATOR</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>IXNAME</td>
<td>NAME</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>IXNAME</td>
<td>ACCESSNAME</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>J</td>
<td>SORTN_JOIN</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>J</td>
<td>SORTC_JOIN</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>JDEG</td>
<td>JOIN_DEGREE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>JOIN</td>
<td>JOIN</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>JPG</td>
<td>JOIN_PGROUP_ID</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>JTYPE</td>
<td>JOIN_TYPE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>KEEPDYNAMIC</td>
<td>KEEPDYNAMIC</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>KEEPDYNAMIC</td>
<td>KEEPDYNAMIC</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>KEEPDYNAMIC</td>
<td>KEEPDYNAMIC</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>KEY</td>
<td>KEYFIELD</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>KEY COLUMN</td>
<td>COLNAME</td>
<td>SYSKEYS</td>
<td>KEYCOLS</td>
</tr>
<tr>
<td>KEYCOLS</td>
<td>KEYCOLS</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>LCK</td>
<td>TSLOCKMODE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>LEAFDIST</td>
<td>LEAFDIST</td>
<td>SYSINDEXPART</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>LEFT-HAND</td>
<td>LEFT_HAND_SIDE</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>LEN</td>
<td>LENGTH</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>LHS_TABNO</td>
<td>LHS_TABNO</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>Report field label</td>
<td>DB2 column name</td>
<td>DB2 table or other source</td>
<td>Report section</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>--------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>LO2KEY</td>
<td>LOW2KEY</td>
<td>SYSOCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>LOCK</td>
<td>LOCKRULE</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>MERC</td>
<td>MERGC</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>MERGE</td>
<td>MERGE_JOIN_COLS</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>MERGN</td>
<td>MERGN</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>METH</td>
<td>METHOD</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>MISMATCH</td>
<td>MISMATCH</td>
<td>None</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>MIX</td>
<td>MIXOPSEQ</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>MIXED</td>
<td>MIXED</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>MIXED</td>
<td>MIXED</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>MIXED</td>
<td>MIXED</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>MIXED</td>
<td>MIXED</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>MTCH</td>
<td>MATCHCOLS</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>NACTIVE</td>
<td>NACTIVEF</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>NAME</td>
<td>NAME</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>NAME</td>
<td>NAME</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>NAME</td>
<td>NAME</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>NAME</td>
<td>NAME</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>NEARINDREF</td>
<td>NEARINDREF</td>
<td>SYSTABLEPART</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>NEAROFF</td>
<td>NEAROFFPOSF</td>
<td>SYSINDEXPART</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>NLEAF</td>
<td>NLEAF</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>NLEVELS</td>
<td>NLEVELS</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>NO</td>
<td>TABNO</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>NPAGES</td>
<td>NPAGESF</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>NU</td>
<td>SORTN_UNIQ</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>NULLS</td>
<td>NULLS</td>
<td>SYSOCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>O</td>
<td>SORTN_ORDERBY</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>O</td>
<td>SORTEC_ORDERBY</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>OPER</td>
<td>OPERATIVE</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>OPER</td>
<td>OPERATIVE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>OPER</td>
<td>OPERATIVE</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>OPTHINT</td>
<td>OPTHINT</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>Report field label</td>
<td>DB2 column name</td>
<td>DB2 table or other source</td>
<td>Report section</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>--------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>OPTHINT</td>
<td>OPTHINT</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>OPTHINT</td>
<td>OPTHINT</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>OPTHINT</td>
<td>OPTHINT</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>ORDERING</td>
<td>ORDERING</td>
<td>SYSKEYS</td>
<td>KEYCOLS</td>
</tr>
<tr>
<td>OWNER</td>
<td>OWNER</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>PADDED</td>
<td>PADDED</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>PAR</td>
<td>PARALLELISM_MODE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>PARENT</td>
<td>PARENTS</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>PARENT_PLANNO</td>
<td>PARENT_PLANNO</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>PARENT_QBLOCKNO</td>
<td>PARENT_QBLOCKNO</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>PARTS</td>
<td>PARTITIONS</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>PCTFREE</td>
<td>SYSINDEXEXPART</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>PCTFREE</td>
<td>SYSTABLEPART</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>PCTPAGES</td>
<td>PCTPAGES</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>PDSNAME</td>
<td>PDSNAME</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>PDSNAME</td>
<td>PDSNAME</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>PDSNAME</td>
<td>PDSNAME</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>PERCACTIVE</td>
<td>PERCATIVE</td>
<td>SYSTABLEPART</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>PERCDROP</td>
<td>PERCDROP</td>
<td>SYSTABLEPART</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>PGSIZE</td>
<td>PGSIZE</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>PL</td>
<td>PLANNO</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>PLAN</td>
<td>PLNAME</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>PLAN</td>
<td>PLANNAME</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>PRANGE</td>
<td>PAGE_RANGE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>PRE</td>
<td>PREFETCH</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>PROCMS</td>
<td>PROCMS</td>
<td>DSN_STATEMENT_TABLE</td>
<td>STMT_COST</td>
</tr>
<tr>
<td>PTYPE</td>
<td>PRIMARY_ACCESTYPE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>QB</td>
<td>QBLOCKNO</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>QTYPE</td>
<td>QBLOCK_TYPE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>QUALIFIER</td>
<td>QUALIFIER</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>QUALIFIER</td>
<td>QUALIFIER</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>QUALIFIER</td>
<td>QUALIFIER</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>QUALIFIER</td>
<td>QUALIFIER</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>Report field label</td>
<td>DB2 column name</td>
<td>DB2 table or other source</td>
<td>Report section</td>
</tr>
<tr>
<td>--------------------</td>
<td>----------------</td>
<td>----------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>QUERYNO</td>
<td>QUERYNO</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>QUERYNO</td>
<td>QUERYNO</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>QUOTE</td>
<td>QUOTE</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>QUOTE</td>
<td>QUOTE</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>QUOTE</td>
<td>QUOTE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>REASON</td>
<td>REASON</td>
<td>DSN_STATEMENT_TABLE</td>
<td>STMT_COST</td>
</tr>
<tr>
<td>RECLength</td>
<td>RECLength</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>RELBOUND</td>
<td>RELBOUND</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>RELBOUND</td>
<td>RELBOUND</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>RELBOUND</td>
<td>RELBOUND</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>RELEASE</td>
<td>RELEASE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>RELEASE</td>
<td>RELEASE</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>REMOTE</td>
<td>REMOTE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>REOPT</td>
<td>REOPTVAR</td>
<td>SYSPLAN</td>
<td>DBRM</td>
</tr>
<tr>
<td>REOPT</td>
<td>REOPTVAR</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>REOPT</td>
<td>REOPTVAR</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>RIGHT-HAND</td>
<td>RIGHT_HAND_SIDE</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>S.U.*RATE</td>
<td>PROCsu</td>
<td>DSN_STATEMENT_TABLE</td>
<td>STMT_COST</td>
</tr>
<tr>
<td>SCALE</td>
<td>SCALE</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>SCPG</td>
<td>SORTC_PGROUP_ID</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>SECTNOI</td>
<td>SECTNOI</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>SEGSIZE</td>
<td>SEGSIZE</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>SNP.G</td>
<td>SORTN_PGROUP_ID</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>SQLERROR</td>
<td>SQLERROR</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>SQL-STATEMENT</td>
<td>STMT</td>
<td>SYSSTMT, SYSPACKSTMT, DBRMLIB member</td>
<td>SQLTEXT</td>
</tr>
<tr>
<td>SQL-STATEMENT</td>
<td>STMT</td>
<td>SYSSTMT, SYSPACKSTMT, DBRMLIB member, APPTUNE trace data</td>
<td>STMT_COST</td>
</tr>
<tr>
<td>STATFDBK</td>
<td>REASON</td>
<td>DSN_STAT_FEEDBACK</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>STATFDBK</td>
<td>REASON</td>
<td>DSN_STAT_FEEDBACK</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>Report field label</td>
<td>DB2 column name</td>
<td>DB2 table or other source</td>
<td>Report section</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------</td>
<td>--------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>STATFDBK</td>
<td>REASON</td>
<td>DSN_STAT_FEEDBACK</td>
<td>KEYCOLS</td>
</tr>
<tr>
<td>STATFDBK</td>
<td>REASON</td>
<td>DSN_STAT_FEEDBACK</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>STATTIME</td>
<td>STATTIME</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>STATTIME</td>
<td>STATTIME</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>STATTIME</td>
<td>STATTIME</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>STG1</td>
<td>SEARCHARG</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>STMT_TYPE</td>
<td>STMT_TYPE</td>
<td>DSN_STATEMNT_TABLE</td>
<td>STMT_COST</td>
</tr>
<tr>
<td>STMTNO</td>
<td>STMTNO</td>
<td>SYSSSTMT, SYSPACKSTMT,</td>
<td>STMT_COST</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBRMLIB member</td>
<td></td>
</tr>
<tr>
<td>STMTNO</td>
<td>STMTNO</td>
<td>SYSSSTMT, SYSPACKSTMT,</td>
<td>SQLTEXT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DBRMLIB member, APPTUNE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>trace data</td>
<td></td>
</tr>
<tr>
<td>TABLESPACE</td>
<td>TSNAPNAME</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>TBCREATOR</td>
<td>CREATOR</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>TBNNAME</td>
<td>TNAME</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>TBNNAME</td>
<td>NAME</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>TIMESTAMP</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>TSTYPE</td>
<td>TSTYPE</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>TTYPE</td>
<td>TABLE_TYPE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>TYPE</td>
<td>COLTYPE</td>
<td>SYSCOLUMNS</td>
<td>COLUMNS</td>
</tr>
<tr>
<td>TYPE</td>
<td>INDEXTYPE</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>TYPE</td>
<td>TYPE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>TYPE</td>
<td>TYPE</td>
<td>DSN_PREDICAT_TABLE</td>
<td>PREDICATE</td>
</tr>
<tr>
<td>TYPE</td>
<td>TYPE</td>
<td>SYSTABLESPACE</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>UNIQUE</td>
<td>UNIQUERULE</td>
<td>SYSINDEXES</td>
<td>INDEX_STATS</td>
</tr>
<tr>
<td>USED</td>
<td>HINT_USED</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
<tr>
<td>VALID</td>
<td>VALID</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>VALID</td>
<td>VALID</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>VALIDATE</td>
<td>VALIDATE</td>
<td>SYSPACKLIST</td>
<td></td>
</tr>
</tbody>
</table>

102  SQL Explorer for DB2 User Guide
<table>
<thead>
<tr>
<th>Report field label</th>
<th>DB2 column name</th>
<th>DB2 table or other source</th>
<th>Report section</th>
</tr>
</thead>
<tbody>
<tr>
<td>VALIDATE</td>
<td>VALIDATE</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>VALIDATE</td>
<td>VALIDATE</td>
<td>SYSPACKLIST</td>
<td>PACKLIST</td>
</tr>
<tr>
<td>VALPROC</td>
<td>VALPROC</td>
<td>SYSTABLES</td>
<td>TABLE_STATS</td>
</tr>
<tr>
<td>VERS</td>
<td>VERSION</td>
<td>SYSDBRM</td>
<td>DBRM</td>
</tr>
<tr>
<td>VERS</td>
<td>VERSION</td>
<td>DBRMLIB member</td>
<td>MEMBER</td>
</tr>
<tr>
<td>VERS</td>
<td>VERSION</td>
<td>SYSPACKAGE</td>
<td>PACKAGE</td>
</tr>
<tr>
<td>WHEN</td>
<td>WHEN_OPTIMIZE</td>
<td>PLAN_TABLE</td>
<td>PLAN_TABLE</td>
</tr>
</tbody>
</table>
Explaining and executing an individual SQL statement

This chapter describes how to use SQL Explorer to explain and execute an individual SQL statement.

This chapter includes the following sections.

- “Overview of the SQLX edit macro” on page 105
- “Explaining an individual statement” on page 106
- “Executing an individual statement” on page 107

Overview of the SQLX edit macro

The SQLX edit macro provides interactive explanation and execution of individual SQL statements and is valid anywhere you can edit Data Manipulation Language (DML) SQL.

The SQLX edit macro can be used in an ISPF edit session. You can access the text you want to process from a data set or enter the SQL in a TSO edit session.

The SQLX edit macro enables you to ensure that the data is being processed as intended by displaying the requested data rows. When the SQL statement is processing the data correctly, you can switch to the Explain function and tune the statement for optimal performance.

You can view the results of an SQLX Explain operation only online. For more information, see “Main panels” on page 68.

Using the execution function of the SQLX edit macro, you can input values for host variables and execute an SQL statement in the foreground.
Explaining an individual statement

This task describes how you can analyze a single SQL statement in the foreground by using the SQLX edit macro.

1. In ISPF, navigate to the data set containing the SQL you want to Explain. Alternatively, you can open a TSO edit session and enter the SQL there.

2. Mark the SQL text you want to analyze by entering the block QQ command at the statement begin line and at the statement end line.

Depending on the programming language, the begin and end statements may have different keywords. The example in Figure 33 on page 106 is from a COBOL program with EXEC SQL as the beginning text and END-EXEC as the ending text.

In the example shown in Figure 33 on page 106, the member SQL in RDADAC.JCL is being edited.

Note
SQLX does not support the use of a sequential text file greater than 80 bytes.

Figure 33: Specify SQL for the SQLX Edit Macro

<table>
<thead>
<tr>
<th>ISREDE2</th>
<th>RDADAC.JCL(SQL) - 01.69</th>
<th>Columns 00001 00072</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command ====&gt; SQLX DEAH</td>
<td>Scroll ====&gt; CSR</td>
<td></td>
</tr>
<tr>
<td>099001</td>
<td>EXEC SQL</td>
<td></td>
</tr>
<tr>
<td>000002</td>
<td>DECLARE CSR2 CURSOR FOR</td>
<td></td>
</tr>
<tr>
<td>000003</td>
<td>SELECT EMPNO, LASTNAME,</td>
<td></td>
</tr>
<tr>
<td>000004</td>
<td>FIRSTNAME, PHONENO</td>
<td></td>
</tr>
<tr>
<td>000005</td>
<td>FROM EMP</td>
<td></td>
</tr>
<tr>
<td>000006</td>
<td>WHERE EMPNO &gt; :WS-EMPNO</td>
<td></td>
</tr>
<tr>
<td>000007</td>
<td>END-EXEC.</td>
<td></td>
</tr>
</tbody>
</table>

Note
If the text spans multiple panels, you might get an INVALID COMMAND message when you press F8 (DOWN) because ISPF does not recognize QQ as a valid command. Ignore the INVALID COMMAND message while you continue to mark the statement, then go to Step 3 on page 106 to initiate the macro. Alternatively, you can use the Q nn command at the beginning of the statement, where nn is the number of lines to search to locate the end of the statement.

3. On the Command line, type SQLX ssid, where ssid is the ID of the subsystem on which you want to run the Explain.

For subsequent SQLX operations, if the statement you want to Explain is on the subsystem you specified on the previous SQLX operation, you can simply type SQLX.
4 Press **Enter** to display the SQLX Parameters panel (Figure 34 on page 107).

**Figure 34: SQLX Parameters panel (PSSPE010)**

```
PSSPE010 ----------------- SQL EXPLORER 11.2.00 SQLX Parameters ----------------
Command ===>  

Specify the options below and press ENTER to continue.

SQLX Option . . . . . . 1  1. Explain  
                    2. Execute

Qualifier Name . . . . IOD

Source Language . . . . 1  1. COBOL (VS-COBOL II, COBOL/370)  
                         2. PL/I  
                         3. C  
                         4. ASSEMBLER  
                         5. FORTRAN  
                         6. SQL Text Only

DDF Location . . . . . ________________ (Optional parameter)
```

5 Complete the SQLX Parameters panel:

   a In the **SQLX Option** field, select **Explain**.

   b Specify the **Qualifier Name** to use to resolve unqualified table references.

   c Specify the **Source Language** for your TSO edit session.

       The language types expect the SQL to be in the format required by those High Level Programming languages.

   d *(optional)* Specify a different **DDF Location** from which you want to run the Explain operation.

   e Press **Enter**.

       The Explain Results panel is displayed.

---

**Executing an individual statement**

This task describes how to execute a single SQL statement in the foreground using the SQLX edit macro.

1 On the TSO edit panel, edit a member containing the SQL you want to execute, or enter an SQL statement.

2 Mark the statement to be executed by using the block QQ commands.

   For more information, see **Step 2 on page 106**
3 On the **Command** line, type **SQLX ssid**, where *ssid* is the ID of the subsystem on which you want to run SQLX to execute the SQL statement.

For subsequent SQLX operations, if the statement you want to execute is on the subsystem you specified on the previous SQLX operation, you can simply type **SQLX**.

4 Press **Enter** to display the SQLX Parameters panel.

5 Complete the SQLX Parameters panel:

   a In the **SQLX Option** field, select **Execute**.
   
   b Specify a **Qualifier Name** to use to resolve unqualified table references.
   
   c Specify the **Source Language** for your TSO edit session.

      The language types expect the SQL to be in the format required by those High Level Programming languages.
   
   d *(optional)* Specify a **DDF Location** from which you want to execute the SQL.
   
   e Press **Enter**.

If your statement contains host variables, the SQL Host Variables List panel is displayed (Figure 35 on page 108). On this panel, you can indicate host variable replacements.

**Figure 35: SQL Host Variables List panel (PSSPLIHV)**

<table>
<thead>
<tr>
<th>Command</th>
<th>SQL EXPLORER 11.2.00 SQL Host Variables List</th>
<th>Row 1 to 1 of 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display SQL . . . . . N</td>
<td><em>(Y/N) Display the SQL statement</em></td>
<td></td>
</tr>
<tr>
<td>Execute SQL . . . . . N</td>
<td><em>(Y/N) Substitute values and execute SQL</em></td>
<td></td>
</tr>
<tr>
<td>Select a variable with 'S' to enter long character values.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S Host Variable</td>
<td>Substitute Value</td>
<td></td>
</tr>
</tbody>
</table>

| S WS-EMPNO |

6 If one of your replacement values is longer than 44 characters and wraps to the next line as you are typing, type **S** in column 1 to the left of the host variable you are editing, and press **Enter**.
The SQL Host Variable Edit panel is displayed (Figure 36 on page 109).

**Figure 36: SQL Host Variable Edit panel (PSSPEDHV)**

<table>
<thead>
<tr>
<th>Command</th>
<th>SQL Explorer 11.2.00 SQL Host Variable Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Edit the value for the host variable, then press END to continue.</td>
<td></td>
</tr>
<tr>
<td>Host Variable . . . : WS-EMPNO</td>
<td></td>
</tr>
<tr>
<td>Host Variable Value   '000010'</td>
<td></td>
</tr>
</tbody>
</table>

7 In the **Host Variable Value** field, type a replacement value for the host variable.

The replacement value you enter is parsed and evaluated to determine whether it meets the criteria for a NUMERIC value. If the replacement value meets these criteria, it is treated as numeric and is not enclosed in apostrophes. Decimal values are treated as numeric.

8 Press **END**.

The SQL Host Variables List panel is displayed (Figure 37 on page 109).

**Figure 37: SQL Host Variables List panel (PSSPLIHV)**

<table>
<thead>
<tr>
<th>Command</th>
<th>SQL EXPLORER 11.2.00 SQL Host Variables List</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display SQL . . . . Y (Y/N) Display the SQL statement</td>
<td></td>
</tr>
<tr>
<td>Execute SQL . . . . Y (Y/N) Substitute values and execute SQL</td>
<td></td>
</tr>
</tbody>
</table>

Select a variable with 'S' to enter long character values.

<table>
<thead>
<tr>
<th>S Host Variable</th>
<th>Substitute Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS-EMPNO</td>
<td>'000010'</td>
</tr>
</tbody>
</table>

9 Complete the SQL Host Variables List panel:

a In the **Display SQL** field, perform one of the following actions:
   - Y displays the SQL with the substituted host variable values.
   - N does not display the SQL on a separate TSO edit panel.

   **Note**

   BMC Software recommends that you review the substituted values by using the **Display SQL** option before running the SQL.

b In the **Execute SQL** field, perform one of the following actions:
   - Y runs the SQL statement without reviewing it.
   - N enables you to review the statement before submitting it for execution.
Note

If you type Y for both Display SQL and Execute SQL, the product first displays the SQL with the substituted values and then executes the SQL without returning to the SQL Host Variables List panel.

c) Press Enter.

The SQL Statement Display panel is displayed (Figure 38 on page 110).

Figure 38: SQL Statement Display panel (PSSPLIHS)

---

Press END to return to Edit Host Variables.

---

The SQL Statement Display panel is displayed (Figure 38 on page 110).

10 Review the value substitutions, and press END.

The results of executing the SQL statement are displayed (Figure 39 on page 110).

Figure 39: Results of SQL statement execution

---

110 SQL Explorer for DB2 User Guide
11 Review the results of the SQL statement execution.

12 Press END.

The TSO edit panel is displayed.
Explaining and comparing plans, packages, and DBRMs

This chapter describes how to use SQL Explorer to Explain and compare plans, packages, and DBRMs.

It includes the following sections:

■ “Overview of explaining and comparing in the same job” on page 113
■ “Explaining and comparing a package” on page 114
■ “Specifying Compare options for an Explain and Compare job” on page 117

Overview of explaining and comparing in the same job

You can explain and compare objects in the same job. The basic steps are identical for each part of the process. The Explain job runs, then the Compare job runs.

The concepts behind the Explain and Compare function are the same as those for the Explain function and the Compare function.

The Explain and Compare function allows the following wildcard characters:

<table>
<thead>
<tr>
<th>Wildcard character</th>
<th>Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ (underscore)</td>
<td>Any single character</td>
</tr>
<tr>
<td>% (percent sign)</td>
<td>A string of zero or more characters</td>
</tr>
<tr>
<td>* (asterisk)</td>
<td></td>
</tr>
</tbody>
</table>

For example, if you know that the plan you are looking for begins with the characters BMC, type BMC% to return a list of all plans beginning with the characters BMC.
Explaining and comparing a package

This task describes how to Explain and Compare a package. Explaining and Comparing at the same time enables you to compare the last analysis with your most recent changes.

Explain and Compare jobs are processed only in batch mode. The Explain job runs first, followed by the Compare job.

1. On the SQL Explorer main menu, select Explain and Compare.
2. Review the SSID and DDF Location, revise if necessary, and press Enter.

The Explain and Compare Object Specification panel is displayed (Figure 40 on page 114).

**Note**

Explain and Compare recognizes only information that is applicable to the object Type you specify. For example, if you specify to Explain and Compare a package, any information in the plan or DBRM fields is ignored. You do not need to remove any of the unrelated values before running the job.

3. Specify the Explain and Compare Object Specification options:

---

**Related Information**

- “Explaining SQL statements” on page 63
- “Comparing Explain history” on page 119
a In the **Type** field, select **Package**.

b In the **Package: COLLID** field, type the name of the package collection.

You can use wildcards in this field.

c If the information is truncated, place your cursor on the field, and press **F4** or **F6** (ZOOM). The contents of the entire field are displayed in a pop-up window.

d In the **Package: Name** field, type the name of the package you want to Explain and Compare.

Specify the name that was used when the package was bound. You can enter a specific package name or you can use wildcards in the name field to display a list of matching packages, from which you can select particular packages to Explain and Compare.

If the information is truncated, place your cursor on the field, and press **F4** or **F6** (ZOOM). The contents of the entire field are displayed in a pop-up window.

e In the **Package: Version** field, specify the version identifier for the package.

You can type a specific package version, use wildcards in this field, or leave the field blank (for packages that do not use a version). If the package that you want to Explain and Compare has a version but you leave the version field blank, the package does not appear in your results.

If the information is truncated, place your cursor on the field, and press **F4** or **F6** (ZOOM). The contents of the entire field are displayed in a pop-up window.

---

**Best practice**

Because package versions can contain mixed-case characters or timestamps, and can be rather lengthy and difficult to remember, BMC Software recommends that you use a wildcard in this field.

---

f In the **Object Selection List** field, type **Y**.

g In the **Explain Options** field, type **Y**.

h In the **Compare Options** field, type **Y**, and press **Enter**.

The Explain and Compare Object Selection List panel is displayed (Figure 41 on page 116). This panel might show only one or two object names, or it might contain a list spanning several panels.
Note
At least one object name must appear in the selection list for the Explain and Compare to occur. If the product cannot locate any object names, it issues an error message.

Figure 41: Explain and Compare Object Selection List panel (PSSPOLST)

<table>
<thead>
<tr>
<th>COLLECTION ID</th>
<th>PACKAGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS0SQL</td>
<td>PSSB1_BPU6992</td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS0SQL</td>
<td>PSSB1_BPU6853</td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS0SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSSOSQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSSESQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS5SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS6SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS7SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS8SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS9SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS10SQL</td>
<td>PSSB1_BPU7158</td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS11SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS12SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS13SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS14SQL</td>
<td></td>
</tr>
<tr>
<td>DAA102_Q_MAIN</td>
<td>PSS15SQL</td>
<td></td>
</tr>
</tbody>
</table>

4 Select the packages to Explain and Compare as follows:

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Take this action...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select one or more packages individually</td>
<td>Type any character in the space beside each package name.</td>
</tr>
<tr>
<td>Select all packages in the list</td>
<td>Type one of the following commands on the Command line:</td>
</tr>
<tr>
<td></td>
<td>■ SELECT ALL</td>
</tr>
<tr>
<td></td>
<td>■ S ALL</td>
</tr>
<tr>
<td></td>
<td>■ S *</td>
</tr>
<tr>
<td></td>
<td>■ SELECT *</td>
</tr>
<tr>
<td>Note: The list of packages could span several panels. Selecting an action to apply to all packages applies the action to all packages in the list, not just the packages listed on the currently displayed panel.</td>
<td></td>
</tr>
<tr>
<td>Cancel selection of a package</td>
<td>Delete the selection character in the space beside the package name.</td>
</tr>
</tbody>
</table>

5 Press Enter and then press END.

The Explain Options panel is displayed.

6 Specify options for the Explain.

For instructions, see “Specifying Explain options” on page 76.
Specify options for the Compare.

For instructions, see “Specifying Compare options for an Explain and Compare job” on page 117.

Specify options for the batch job.

For instructions, see “Specifying JCL options in batch mode” on page 48.

The Explain output is sent to SYSOUT or to a data set you specify.

---

Note
If any of the data sets do not exist, the Allocate Data Set panel is displayed. For instructions, see “Allocating a data set for a specific job” on page 45.

---

**Specifying Compare options for an Explain and Compare job**

This task describes how to specify Compare options for an Explain and Compare job.

For more information about specifying options for a Compare job, see “Specifying Compare options for a Compare job” on page 126.

1. Access the Compare Options panel.

   For instructions, see “Explaining and comparing a package” on page 114.

   **Figure 42: Compare Options panel (PSSPC040)**

<table>
<thead>
<tr>
<th>Compare Explain History to 1</th>
<th>(1 - Most recent relative history)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Differences Only</td>
<td>Y (Y/N)</td>
</tr>
<tr>
<td>SQL % Matching Threshold</td>
<td>100 (0-100)</td>
</tr>
<tr>
<td>Cost Summary Report</td>
<td>... N (Y/N)</td>
</tr>
</tbody>
</table>

2. In the Compare Explain History to field, specify the relative history entry to be compared to the history entry that the Explain created, which has a relative history of 0.

   A relative history of 1 represents the most recent history prior to running the Explain.
3 In the **Report Differences Only** field, type Y to report only the statements with a difference in access paths between the two histories selected.

4 In the **SQL % Matching Threshold** field, specify a percentage to determine whether a match exists when comparing SQL statements to each other.

   For example, a value of 50 means to compare any SQL statements that match at 50% or greater. The default is 100.

5 In the **Cost Summary Report** field, type Y to list in the report the matching percentage for each pair of statements, along with an indication of whether differences were found in access paths or statistics or both.

6 Press **Enter**.
Comparing Explain history

This chapter contains the following sections:

- “Overview of historical comparisons” on page 119
- “Comparing a package” on page 121
- “Specifying Compare options for a Compare job” on page 126
- “Comparing Explain history for two different objects” on page 127

Overview of historical comparisons

The SQL Explorer for DB2 product compares the access paths of the historical versions of a plan, package, or DBRM Explain to identify differences in access paths, cost, SQL text, and key catalog statistics that can impact selection of access paths by the DB2 optimizer.

The comparison highlights the differences between two sets of Explain results to enable you to determine which changes have an impact on performance. For example, although an SQL statement in a previous application version was resolved by an efficient access path (such as a matching index access), in a new version of the application, the DB2 optimizer might select an inefficient access path (such as a table space scan).

SQL Explorer can identify many reasons for a change in access path. Many changes might have occurred in the environmental statistics. For instance, the index keys could have changed, the index might no longer be clustered, or the index has been dropped. The report layout highlights what is different to help you locate the reasons for these changes.

The Compare Explain History function matches on the SQL text and can find matches even if the statement numbers have changed. Return codes can be monitored to prevent other steps from running after the Compare Explain History
The batch Compare process can compare two objects with a different name, collection ID, or version.

**Note**
The Workload Access Path Compare component of the SQL Performance solution is an automated tool that identifies changed access paths and provides analysis and drill down capability for either static or dynamic SQL statements in a user-defined workload.

Workload data, including column distribution statistics and SQL text, are gathered for comparison from one of the following sources:

- The DB2 catalog
- APPTUNE archived trace data sets
- A DBRM library
- BMC Performance Database
- SQL text data set
- Statement cache

This feature is available only as part of the SQL Performance solution. For more information, see the *SQL Performance for DB2 User Guide*.

### SQL statement matches and match percentages

The Compare Explain History function uses an SQL parser that breaks the SQL statement into lines.

Each major or subordinate clause of the SQL statement begins a new line. Subordinate keywords (like **AND** or a column name in the first clause) also begin a new line.

Each line of the first SQL statement is checked against a line from the corresponding SQL statement of the second object. For example, the FROM clause of the first object is compared to the FROM clause of the second object. Any differences that are found affect the matching percentage. If the matching percentage is within the threshold in the **MATCHPCT** parameter, the two statements are compared. Matching is done using a top-down method in which all SQL statements matching at 100% are done, then 99%, then 98%, and so forth.

In the following example, the SQL statements matched at 50%:

```sql
SELECT EMPNO ,
   +      ,FIRSTNME
   -      ,MIDINIT
   ,LASTNAME
   ,MIDINIT
   +      ,FIRSTNME
 FROM   IOD.EMP
 WHERE  (EMPNO = ?)
```
The formula for calculating the matching percentage per statement is:

\[
\frac{\text{(# matching lines)} \times 100}{\text{(# lines from either statement, whichever is greater)}}
\]

In this example, the number of matching lines is 4. The number of lines from the first object is 7 and from the second object is 8. The maximum number (8) is used as follows: \(4 \times 100 / 8 = 50\%\).

**Cost comparisons**

The Compare Explain History function reads strictly from the SQL Explorer history tables.

To enable Compare, you must have run an Explain with HISTORY=YES to populate the history tables for that object. If the DSN_DETCOST_TABLE did not exist when the Explain function was issued, the cost appears as zero in the Compare report. The comparison of cost data can affect how the return code gets set.

For information about return codes for Compare, see “Compare return codes” on page 222.

**Tip**

For a dynamic explain (XD), the product creates the DSN_DETCOST_TABLE if needed. For a static explain (XS), if the DSN_DETCOST_TABLE did not exist at the time you issue the BIND with EXPLAIN(YES), the product reports zero cost because the optimizer was not able to record the cost information at the time of the BIND.

BMC Software recommends that you ensure that the DSN_DETCOST_TABLE exists before doing a BIND with EXPLAIN(YES).

**Comparing a package**

This task describes how to Compare the Explain history for a package.

If the information in a field is truncated, place your cursor on the field, and press F4 or F6 (ZOOM). The contents of the entire field are displayed in a pop-up window.

1. On the SQL Explorer main menu, select Compare.
2 Review the **SSID** and **DDF Location**, revise if necessary, and press **Enter**.

The Compare Object Specification panel is displayed (Figure 43 on page 122).

---

**Note**

Compare recognizes only the information that is applicable to the object type that you specify. For example, if you Compare a package, any information existing in the plan or DBRM fields is ignored. You do not need to remove any of the unrelated values before performing the comparison.

---

### Figure 43: Compare Object Specification panel (PSSPA130)

PSSPA130 --------------- Compare Object Specification -------------------------

Command ===> Type . . (1=Plan, 2=Package, 3=DBRM)

Plan:
Name

Package:
COLLID Name . .
Version

DBRM:
Plan Name . .

Object Selection List Y (Y/N) Plan, package, or DBRM object(s)
History Selection Y (Y/N) (Y=Show List of Histories)
(N=Specify relative history records below where 0=Most Recent)
   0 1st Relative History
   1 2nd Relative History

/ Compare Options

---

3 In the **Type** field, select **Package**.

4 In the **Package: COLLID** field, type the name of the package collection.

You can use wildcards in this field.

5 In the **Package: Name** field, type the name of the package that you want to Compare.

This is the package name used when the package was bound. You can enter a specific package name or you can use wildcards in the name field to display a list of matching objects, from which you can select particular packages to Compare.

6 In the **Package: Version** field, specify the version identifier for the package, as follows:
   - Use wildcards in this field (recommended)
   - Type a specific package version
   - Leave the field blank (for packages that do not use a version)
   - Specify **AUTO** as the value of the field (if packages are versioned automatically in your environment)
If you specify **AUTO**, SQL Explorer Compares two instances of packages with the same collection ID and name without considering version number.

**WARNING**

If the package that you want to Compare has a version but you leave the version field blank, the package does not appear in your results.

If you enter a version incorrectly, the package does not appear in your results. Because package versions can contain mixed-case characters or timestamps, and can be rather lengthy and difficult to remember, BMC Software recommends that you use a wildcard in this field unless packages are versioned automatically in your environment, in which case you should use **AUTO**.

1. In the **Object Selection List** field, specify whether you want to select objects from a selection list:
   - If you want to select objects from a selection list, type **Y**.
   - If you specified **AUTO** as the package version or if you want to use the specific criteria that you entered on the panel, type **N**.

2. In the **History List** field, specify whether you want to view a list of histories from which to select:
   - If you want to view a list of histories, type **Y**.
   - If do not want to view a list or histories or if you specified **AUTO** as the package version, type **N**. If you specify **N**, use the **first Relative History** and **second Relative History** fields to specify the versions of the package that you want to Compare, relative to the most recent instance of the package. The most recent instance of the package is considered **0**.

   **Example**

   If you want to Compare the most recent instance of the package with the previous instance of the package, enter **0** and **-1**. If you want to Compare the most recent instance of the package with the package two instances back, enter **0** and **-2**. If you have two instances with the same timestamp, SQL Explorer Compares those two instances.

3. In the **Compare Options** field, type any character in the space beside the field, and press **Enter**.

   The Compare Options panel is displayed ([Figure 46 on page 126](#)).

4. Specify options for the Compare and press **Enter**.
For instructions, see “Specifying Compare options for a Compare job” on page 126.

11 If you specified to display the object selection list, perform the following substeps. Otherwise, continue to Step 12.a on page 125.

a At the Compare Object Selection List panel (Figure 44 on page 124), specify the objects that you want to Compare.

**Figure 44: Compare Object Selection List panel (PSSPOLST)**

```
PSSPOLST ---------------- Compare Object Selection List ---------- 1 objects selected
Command ==>                                                  Scroll ==> CSR
Select object(s) and press ENTER and then F3 to END.

<table>
<thead>
<tr>
<th>COLLECTION ID</th>
<th>PACKAGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>_</td>
<td>HVM6PK01</td>
<td>IODWL410 V1.01</td>
</tr>
<tr>
<td>_</td>
<td>HVM6PK01</td>
<td>IODWL413 V1.01</td>
</tr>
<tr>
<td>_</td>
<td>HVM6PK01</td>
<td>IODWL414 V1.01</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODUDF01 V1.01</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODWLX02 V101</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODWLX03 V101</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODWLX04 V101</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODWL002 V101</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODWL003 V101</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODWL004 V101</td>
</tr>
<tr>
<td>_</td>
<td>IODPKG01</td>
<td>IODWL005 V101</td>
</tr>
</tbody>
</table>
```

This panel might display only one or two object names, or it might contain a list spanning several panels.

**Note**

At least one object name must appear in the Compare Object Selection List panel for the Compare to occur. If the product cannot locate an object name, it issues an error message.

You can specify the packages to Compare in the following ways:

<table>
<thead>
<tr>
<th>If you want to...</th>
<th>Take this action...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select one or more packages individually</td>
<td>Type any character in the space beside each package name.</td>
</tr>
</tbody>
</table>
| Select all packages in the list | Type one of the following commands on the Command line:  
  ■ SELECT ALL  
  ■ S ALL  
  ■ S *  
  ■ SELECT *  
  Note: The list of packages could span several panels. Selecting an action to apply to all packages applies to all packages in the list, not just the packages listed on the currently displayed panel. |
If you want to... | Take this action...
---|---
Cancel the selection of an package | Delete the selection character in the space beside the package name.

b  Press Enter and then press END.

12 If you specified to display the history selection list, perform the following substeps.

a  At the History Selection List panel (Figure 45 on page 125), select the two version entries for the comparison by typing any character in the space beside the version.

**Figure 45: History Selection List panel (PSSPHLST)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Scroll</th>
<th>CSR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSSPHLST ------------ History Selection List ------------</td>
<td>2 entries selected</td>
<td></td>
</tr>
</tbody>
</table>

1 of 1

OBJECT: PACKAGE IODPKG01.IODWL002.V101

Select 2 history entries and press ENTER and then F3 to END.

ANALYSIS TIME

/ 2010-01-08-11.43.14.785219
/ 2010-01-08-11.43.12.502537
- 2010-12-27-12.00.23.237637

Note
At least two Explain histories must appear in the History Selection List panel for a comparison to occur. If the product cannot locate any history versions, it issues an error message.

The history versions are shown on a separate panel for each object selected. If you selected more than one object to Compare, the versions available for the next object are displayed after you press END.

b  Press Enter.

Depending on the **Process Mode** that you selected, one of the following events occurs:

<table>
<thead>
<tr>
<th>Process mode</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>O (online)</td>
<td>The Compare job runs in the foreground and the online report is displayed.</td>
</tr>
<tr>
<td>B (batch)</td>
<td>The Batch Job panel is displayed. For instructions, see “Specifying JCL options in batch mode” on page 48.</td>
</tr>
</tbody>
</table>
Specifying Compare options for a Compare job

This task describes how to specify options for a Compare job.

For information about specifying Compare options for an Explain and Compare job, see “Specifying Compare options for an Explain and Compare job” on page 117.

1. Access the Compare Options panel (Figure 46 on page 126).

   **Figure 46: Compare Options panel (PSSPC020)**

   PSSPC020 ------------------------ Compare Options ------------------------
   Command ===> Specify options and press ENTER to continue.
   Process Mode . . . . . . O (O=Online, B=Batch)
   Report Differences Only Y (Y/N)
   SQL % Matching Threshold 100 (0-100)
   Currency Code . . . . . USD (USD, EUR, etc.)
   Cost Translation Rate 2 Cost per timeron
   Show SQL . . . . . . . . Y (Y/N)
   Cost Summary Report . . N (Y/N)
   Generate Report Title Y (Y/N)
   Title COMPARE REPORT

   Refer to one of the following procedures for instructions:

   ■ “Setting analysis options” on page 53.

   ■ “Comparing a package” on page 121

2. On the Compare Options panel, specify a **Process Mode** for the comparison:

   ■ **Online** runs the Compare online.

   ■ **Batch** creates JCL and runs the Compare in batch.

3. In the **Report Differences Only** field, type Y to report only the statements with a difference in access paths between the two histories selected.
4 In the **SQL % Matching Threshold** field, specify a percentage to determine whether a match exists when comparing SQL statements to each other.

For example, a value of 50 means to Compare any SQL statements that match at 50% or greater. The default is 100.

5 Specify a three-digit **Currency Code** for the monetary unit used to calculate the cost translation rate.

---

**Best practice**

You can use any characters except spaces in this field, though BMC Software recommends that you use ISO 4217 standard codes for the representation of currencies. The default is USD.

---

6 In the **Cost Translation Rate** field, specify the rate used to translate the estimated processor cost of executing an SQL statement into a monetary unit.

The rate is monetary units / timeron. The default is 1.

7 In the **Show SQL** field, type Y to display the SQL text in the report.

8 In the **Cost Summary Report** field, type Y to list in the report the matching percentage for each pair of statements, along with an indication of whether differences were found in access paths or statistics or both.

9 In the **Generate Report Title** field, type Y to include a report title in the report.

---

**Note**

If you plan to review the report online from the output data set, BMC Software recommends that you set this option to N.

---

10 Type a **Title** for the report.

11 Press Enter.

### Comparing Explain history for two different objects

The Compare Explain History function enables you to compare history records for two differently named objects.

Because the ISPF interface does not currently support this comparison, you must manually add or change the parameters in a previously generated comparison job.
The Compare panels allow you to compare multiple history records of the same object and version. You do not need to make additional manual changes to the comparison job. However, if you want to compare different objects or versions, some manual intervention is required.

**Before you begin**

Verify whether you have existing Compare JCL, and take the corresponding action, as follows:

- If you have existing Compare JCL, go to Step 1 on page 128.
- If you do not have existing Compare JCL, you must create it before you proceed. For instructions, see “Comparing a package” on page 121. When you have created the JCL, complete the following steps.

**To compare Explain history for two different objects**

1. Edit the JCL to include the new Compare parameters for Object A (the first object) and Object B (the second object). Define the SYSIN parameters listed in Table 12 on page 128.

<table>
<thead>
<tr>
<th>SYSIN parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLLID</td>
<td>Specify the collection ID of Object A when the object is a package.</td>
</tr>
<tr>
<td>COLLID2</td>
<td>Specify the collection ID of Object B when the object is a package. If you do not specify a value, this parameter defaults to the value of the COLLID parameter.</td>
</tr>
<tr>
<td>COMPNEW</td>
<td>Specify the relative history number of Object B.</td>
</tr>
<tr>
<td>COMPOLD</td>
<td>Specify the relative history number of Object A.</td>
</tr>
<tr>
<td>COSTSUM</td>
<td>For information, see “Compare parameters” on page 206.</td>
</tr>
<tr>
<td>CURRENCY</td>
<td>Specify the three-character currency code for the monetary unit used to calculate the cost translation rate. You can use any characters except spaces in this field, though BMC Software recommends that you use ISO 4217 standard codes.</td>
</tr>
<tr>
<td>DDFLOC</td>
<td>Specify the location name of the subsystem when using Distributed Data Facility processing.</td>
</tr>
<tr>
<td>DIFF</td>
<td>For information, see “Compare parameters” on page 206.</td>
</tr>
<tr>
<td>--ENDOBJ</td>
<td>Indicates the end of each set of input parameters.</td>
</tr>
<tr>
<td>MATCHPCT</td>
<td>For information, see “Compare parameters” on page 206.</td>
</tr>
<tr>
<td>NAME</td>
<td>Specify the object name (Object A) of the first plan, package, or DBRM.</td>
</tr>
</tbody>
</table>
### SYSIN parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME2</td>
<td>Specify the object name (Object B) of the second plan, package, or DBRM. If you do not specify a value, this parameter defaults to the value of the NAME parameter.</td>
</tr>
<tr>
<td>--NEWOBJ</td>
<td>Indicates the beginning of each set of input parameters.</td>
</tr>
<tr>
<td>OBJECT</td>
<td>Specify the type of object to be compared: plan, package, or DBRM.</td>
</tr>
<tr>
<td>PLNAME</td>
<td>Specify the plan name of Object A when the object is a DBRM.</td>
</tr>
<tr>
<td>PLNAME2</td>
<td>Specify the plan name of Object B when the object is a DBRM. If you do not specify a value, this parameter defaults to the value of the PLNAME parameter.</td>
</tr>
<tr>
<td>RATE</td>
<td>For information, see “Compare parameters” on page 206.</td>
</tr>
<tr>
<td>RPTHDR</td>
<td>Specify whether to print headings for the report (YES or NO).</td>
</tr>
<tr>
<td>SHOWSQL</td>
<td>Specify whether to show SQL Text in the report (YES or NO).</td>
</tr>
<tr>
<td>TITLE</td>
<td>Specify a title for the report (64 bytes).</td>
</tr>
<tr>
<td>VERSION</td>
<td>Specify the version of Object A when the object is a package.</td>
</tr>
<tr>
<td>VERSION2</td>
<td>Specify the version of Object B when the object is a package. If you do not specify a value, this parameter defaults to the value of the VERSION parameter.</td>
</tr>
</tbody>
</table>

2 Submit the job and review your output.
Comparing Explain history for two different objects
Analyzing mismatches and change impacts

This part presents topics relating to mismatches and schema change impacts.

Chapters:

- Performing mismatch analysis
- Analyzing for impact of schema changes
- Preventing timestamp mismatches before production
- Integrating CATALOG MANAGER for DB2 with SQL Explorer
Performing mismatch analysis

This chapter contains the following sections:

- “Overview of mismatch analysis for access paths” on page 133
- “Analyzing data-type mismatches” on page 135

Overview of mismatch analysis for access paths

SQL access paths are driven by the predicates used for selecting and filtering data from a table, or from multiple tables in the case of a join or subselects.

The SQL Explorer for DB2 product is sensitive to the attributes of the columns in the tables and the variables or expressions supplied from the application program.

The DB2 optimizer can change the access paths based solely on the differences in data types between the column and the host variable or expression. However, the DB2 optimizer does not provide any warning that a data-type mismatch has occurred. The Mismatch Analysis function provides the necessary analysis and comparison to detect when mismatches could cause a change in the access path. The Mismatch Analysis function evaluates the left and right sides of the operand in the predicate, and also evaluates SELECT INTO and UPDATE SET clauses.

The following list describes the mismatch categories that are used when differences in data type or length are detected.

NO IMPACT

NO IMPACT means that changes you make to this category do not improve the access path any further. As such, you need not make any changes in this instance.
The following conditions result in a NO IMPACT category:

- The data type or length is different but cannot be improved further, because it always remains Stage 2 processing (as when a function is used).

  **Example**

  ```sql
  WHERE ABS(EDLEVEL*(- 1.5)) = :WS-TEMP-01.WS-EDLEVEL
  ```

  results in the following mismatch information:

<table>
<thead>
<tr>
<th>SRC</th>
<th>COLUMN</th>
<th>DATA TYPE AND LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREDICATE</td>
<td>EXP</td>
<td>ABS(EDLEVEL *(- 1.5))</td>
</tr>
<tr>
<td>HOST</td>
<td>WS-TEMP-01.WS-EDLEVEL</td>
<td>SMALLINT(2)</td>
</tr>
</tbody>
</table>

- The data type or length is different but occurs in a SELECT column INTO host-variable clause that does not affect the access path.

  **Example**

  ```sql
  SELECT CONCAT(FIRSTNME,LASTNAME) INTO :WS-TEMP-01.WS-TEMP
  ```

  results in the following mismatch information:

<table>
<thead>
<tr>
<th>SRC</th>
<th>COLUMN</th>
<th>DATA TYPE AND LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTO-CLAUSE</td>
<td>EXP</td>
<td>SELECT CONCAT(FIRSTNME,LASTNAME)</td>
</tr>
<tr>
<td>HOST</td>
<td>WS-TEMP-01.WS-TEMP</td>
<td>CHAR(18)</td>
</tr>
</tbody>
</table>

- The data type or length is different but compatible.

  **Example**

  ```sql
  WHERE BIRTHDATE = :WS-TEMP-01.WS-BIRTHDATE
  ```

  results in the following mismatch information:

<table>
<thead>
<tr>
<th>SRC</th>
<th>COLUMN</th>
<th>DATA TYPE AND LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREDICATE</td>
<td>DB2</td>
<td>BIRTHDATE</td>
</tr>
<tr>
<td>HOST</td>
<td>WS-TEMP-01.WS-BIRTHDATE</td>
<td>CHAR(10)</td>
</tr>
</tbody>
</table>

- The data type or length is different but does not require improvement, because it is already at Stage 1 processing.

  **Example**

  ```sql
  WHERE C.TBCREATOR = 'SYSIBM'
  ```

  results in the following mismatch information:

<table>
<thead>
<tr>
<th>SRC</th>
<th>COLUMN</th>
<th>DATA TYPE AND LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREDICATE</td>
<td>DB2</td>
<td>TBCREATOR</td>
</tr>
<tr>
<td>EXP</td>
<td>'SYSIBM'</td>
<td>VARCHAR(6)</td>
</tr>
</tbody>
</table>
**Analyzing data-type mismatches**

This section describes how to perform mismatch analysis and set options to modify the report.

**Analyzing access-path mismatches on a package**

Mismatch Analysis on a package compares the data type and length of the left side of the operand in the predicate to the right side of the operand.

SELECT INTO and INSERT INTO clauses are also checked for differences. If a difference exists, an entry is made in the report.

---

**MISMATCH**

*MISMATCH* indicates the existence of a problem that could have an adverse effect on the DB2 optimizer. In this instance, you can improve performance by making a change.

This category indicates that the data type or length is different and affects the access path. You can improve performance by making a change.

---

**Example**

WHERE SLLNG_DSTRBTR_CD LIKE :WS-TEMP-01.HOSTVAR

results in the following mismatch information:

<table>
<thead>
<tr>
<th>SRC</th>
<th>COLUMN</th>
<th>DATA TYPE AND LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREDICATE</td>
<td>SLLNG_DSTRBTR_CD</td>
<td>CHAR(7)</td>
</tr>
<tr>
<td>HOST</td>
<td>WS-TEMP-01.HOSTVAR</td>
<td>CHAR(8)</td>
</tr>
</tbody>
</table>

**Example**

WHERE COMM = A.COL1

results in the following mismatch information:

<table>
<thead>
<tr>
<th>SRC</th>
<th>COLUMN</th>
<th>DATA TYPE AND LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREDICATE</td>
<td>COMM</td>
<td>DECIMAL(9,2)</td>
</tr>
<tr>
<td>DB2</td>
<td>COL1</td>
<td>INTEGER(4)</td>
</tr>
</tbody>
</table>

For more information, view the Quick Course "SQL Explorer for DB2—Using Mismatch Analysis."
If the plan has only packages, select option 2 (Package) as the object Type.

To analyze mismatches on a package

1 On the SQL Explorer main menu, type 4 (Mismatch Analysis).

2 Change the SSID and DDF locations, if necessary, and press Enter.

   The Mismatch Analysis Object Specification panel is displayed (Figure 47 on page 136).

   Mismatch Analysis recognizes only the information that is applicable to the object Type that you specify. For example, if you perform Mismatch Analysis on a plan, the analysis ignores any information existing in the package fields. You do not need to remove the unrelated values before performing the Mismatch Analysis.

   **Figure 47: Mismatch Analysis Object Specification panel (PSSPH010)**

   PSSPH010 ------------ Mismatch Analysis Object Specification ------------------
   Command ===> Process Mode 0 (O=Online, B=Batch)
   Type . . . . 1 (1=Plan, 2=Package)
   Plan: Name . . . PSS*
   Package: COLLID . . Name . . %
   Version %
   Object Selection List N (Y/N) Plan or package object(s)
   _ Additional Options

3 Complete the Mismatch Analysis Object Specification panel:

   a Specify a Process Mode for the Mismatch Analysis:
      - Online runs the Mismatch Analysis job online.
      - Batch creates a member containing the JCL and enables you to edit and then submit the Mismatch Analysis job in batch.

   b In the Type field, select Package.

   c In the Package: COLLID field, type a collection ID.

      You can use wildcards in this field.

      If the information is truncated, place your cursor on the field, and press F4 or F6 (ZOOM). The contents of the entire field are displayed in a pop-up window.
In the **Package: Name** field, type the name of the package you want to analyze.

This is the name used when the package was bound. You can enter a specific package name or you can use wildcards in the name field to display a list of matching packages, from which you can select particular packages to analyze.

In the **Package: Version** field, specify the version identifier for the package, as follows:

- Use wildcards in this field (recommended).
- Type a specific package version.
- Leave the field blank (for packages that do not use a version).
- Specify **AUTO** as the value of the field (if packages are versioned automatically in your environment).

If you specify **AUTO**, SQL Explorer compares two instances of packages with the same collection ID and name without considering version number.

---

**WARNING**

If the package you want to compare has a version but you leave the version field blank, the package does not appear in your results.

If you enter a version incorrectly, the package does not appear in your results. Because package versions can contain mixed-case characters or timestamps, and can be rather lengthy and difficult to remember, BMC Software recommends that you use a wildcard in this field unless packages are versioned automatically in your environment, in which case you should use **AUTO**.

In the **Object Selection List** field, type **Y**.

*(optional)* Type any character in the **Additional Options** field to update options for Mismatch Analysis.

For more information, see “Reviewing or changing options for the Mismatch Analysis report” on page 139.

Press **Enter**.
The Mismatch Analysis Object Selection List panel is displayed (Figure 48 on page 138).

**Figure 48: Mismatch Analysis Object Selection List panel (PSSPOLST)**

<table>
<thead>
<tr>
<th>COLLECTION ID</th>
<th>PACKAGE</th>
<th>VERSION</th>
</tr>
</thead>
<tbody>
<tr>
<td>BMCACP</td>
<td>ACPCSQL1</td>
<td>ACP_0910_PGMR_RWC</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL1</td>
<td>ACP_0910_PGMR_DFB</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL1</td>
<td>ACP_0910</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL1</td>
<td>ACP_0810_BPU0414</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL1</td>
<td>ACP_0810_BPU0294</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL1</td>
<td>ACP_0810</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL2</td>
<td>ACP_0910_PGMR_RWC</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL2</td>
<td>ACP_0910_PGMR_DFB</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL2</td>
<td>ACP_0910</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL2</td>
<td>ACP_0810_BPU0220</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL2</td>
<td>ACP_0810</td>
</tr>
<tr>
<td>BMCACP</td>
<td>ACPCSQL3</td>
<td>ACP_0910_PGMR_RWC</td>
</tr>
</tbody>
</table>

4 Select the objects to analyze, as follows:

<table>
<thead>
<tr>
<th>To accomplish this...</th>
<th>Take this action...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select one or more packages individually</td>
<td>Type any character in the space beside each package name.</td>
</tr>
</tbody>
</table>
| Select all packages in the list | Type one of the following commands on the Command line:  
    ■ SELECT ALL  
    ■ S ALL  
    ■ S *  
    ■ SELECT *  
    Note: The list of packages could span several panels. Selecting an action to apply to all packages applies the action to all packages in the list, not just the packages listed on the currently displayed panel. |
| Cancel the selection of a package | Delete the selection character in the space beside the package name. |

5 Press **Enter** and then press **END**.

Depending on the **Process Mode** that you selected one of the following events occurs:

<table>
<thead>
<tr>
<th>Process mode</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>O (online)</td>
<td>The Mismatch Analysis job runs in the foreground and the online report is displayed.</td>
</tr>
<tr>
<td>B (batch)</td>
<td>The Batch Job panel is displayed. For instructions, see “Specifying JCL options in batch mode” on page 48.</td>
</tr>
</tbody>
</table>
If any of the data sets do not exist, the Allocate Data Set panel is displayed. For instructions, see “Allocating a data set for a specific job” on page 45.

6 Press **Enter** to generate the JCL.

## Reviewing or changing options for the Mismatch Analysis report

This section guides you through reviewing and changing options related to the Mismatch Analysis feature.

### To review or change Mismatch Analysis options

1. Access the Mismatch Analysis Additional Options panel in one of the following ways:

   - From the SQL Explorer main menu:
     
     — Type 0 (zero) for **Options**.
     
     — In the **Options** field, type 2 (for **Analysis**) and press **Enter**.
     
     — In the **Options** field, type 3 (for **Mismatch**) and press **Enter**.

   - By typing any character in the **Additional Options** field on the Mismatch Analysis Object Specification panel.

   The Mismatch Analysis Additional Options panel is displayed (Figure 49 on page 139).

   **Figure 49: Mismatch Analysis Additional Options panel (PSSPH020)**

   PSSPH020 ----------- Mismatch Analysis Additional Options-----------------------------
   Command ===> Specify options and press ENTER to continue.
   Column Break . . . . . . Y (Y/N) Format 1 column per line in SQL text
   Report Differences Only Y (Y/N)
   Generate Report Title . . Y (Y/N)
   Title MISMATCH ANALYSIS REPORT

2. In the **Column Break** field, specify whether to insert a break between columns listed as part of the SELECT, INSERT, DELETE, or UPDATE clause in the SQL text.

3. In the **Report Differences Only** field, specify one of the following options:
Type Y to report only items with a category of Mismatch (items with a NO IMPACT category are not included).

Type N to report all statements, flagging the statements that are different.

4 In the Generate Report Title field, specify Y to include a report title in the output.

**Note**
If you plan to review the report online from the output data set, BMC Software recommends that you set this option to N.

5 Type a Title for the report.

The maximum length for this field is 64 characters.

6 Press Enter to save your changes and move to the next panel in the series.
Analyzing for impact of schema changes

This chapter includes the following sections:

- “Overview of analyzing schema-change impact” on page 141
- “Analyzing for impact” on page 142
- “Setting Impact Analysis Options” on page 144

Overview of analyzing schema-change impact

When you are considering schema changes, the SQL Explorer for DB2 product enables you to examine the impact of those intended object changes.

Impact Analysis goes beyond simple dependency and actually shows the SQL statement text and number that are affected. Using the report that is produced, you can see where and how much those changes affect the performance and operation of the application.

The Impact Analysis process works against the DB2 catalog and analyzes the SQL statements stored within the catalog tables. Impact Analysis needs only the column name that is going to be converted. If the schema uses a naming convention that can benefit from the use of wildcards, Impact Analysis can find all instances containing the text string. The process then searches the DB2 catalog for all columns that meet the criteria (for example, qualified by creator and table or wildcards, including views, aliases, or synonyms). Using this resulting list, the analysis navigates through several DB2 catalog tables to identify the plans, packages, and DBRMs that are candidates.

The process then scans each SQL statement from candidate DBRMs and packages to see whether the column is referenced within the SQL. If the column is used, the SQL statement is added to the detail report and the appropriate counter is increased by one increment in the summary section.
Analyzing for impact

This task describes how to run an Impact Analysis on objects you are considering changing.

Impact Analysis produces a report showing you the impact results of your intended changes.

To analyze the impact of changes

1. On the SQL Explorer main menu, type 5 (for Impact Analysis).

2. Change the SSID and DDF locations, if necessary, and press Enter.

   The Impact Analysis Object Specification panel is displayed (Figure 50 on page 142).

   **Figure 50: Impact Analysis Object Specification panel (PSSPI010)**

   ```
   Command ===> 
   Process Mode  O   (O=Online, B=Batch)
   Objects to Analyze (Case-Sensitive):  (Table, View, Alias, and Synonym)
   Creator . . IOD               Name . . . EMP
   Columns:         (Comma-Delimited List)
   Name . . . LAST*, FIRST*, MIDINIT
   For the specified table objects, search only these DBRMs and packages:
   Type . . . . 3   (1=DBRM, 2=Package, 3=DBRMs and Packages)
   DBRM:
   Plan . . %               Name . . %
   Package:
   COLLID . . %            Name . . %
   Version   %
   / Options
   ```

3. In the **Process Mode** field, select a processing option for the analysis:

   - **Online** runs the Impact Analysis job online.
   - **Batch** creates a member containing the JCL and enables you to edit and then submit the Impact Analysis job in batch.

4. Specify table objects to analyze, as follows:
**Note**

Be aware of the following information:

- If information is truncated, place your cursor on the field, and press **PF4** or **PF6** (ZOOM). The contents of the entire field are displayed in a pop-up window.
- The **Creator** and **Name** fields are case-sensitive.
- You can use wildcards in any text-entry field.

**a** In the **Creator** field, specify the owner of an alias, synonym, table, or view.

**b** In the **Name** field, specify an alias, synonym, table, or view.

**c** In the **Columns: Name** field, specify one or more column names, separated by commas, to be used in the search.

5 Specify the packages or DBRMs to include in the search:

**a** In the **Type** field, specify **DBRM**, **Package**, or **DBRMs and Packages**.

**b** In the **DBRM: Plan** field, type the name of a plan.

**c** In the **DBRM: Name** field, type the name of a DBRM.

**d** In the **Package: COLLID** field, type the name of the package collection.

**e** In the **Package: Name** field, type the name of the package you want to include.

This is the package name used when the package was bound. You can enter a specific package name or you can use wildcards.

**f** In the **Package: Version** field, specify the version identifier for the package, as follows:

- Use wildcards in this field (recommended).
- Type a specific package version.
- Leave the field blank (for packages that do not use a version).
**WARNING**
If the package you want to compare has a version but you leave the version field blank, the package does not appear in your results.
If you enter a version incorrectly, the package does not appear in your results. Because package versions can contain mixed-case characters or timestamps, and can be rather lengthy and difficult to remember, BMC Software recommends that you use a wildcard in this field.

6  *(optional)* Type any character in the **Options** field to update options for Impact Analysis.

For more information, see “Setting Impact Analysis Options” on page 144.

7  Press **Enter**.

Depending on the **Process Mode** you selected, one of the following events occurs:

<table>
<thead>
<tr>
<th>Process mode</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>O (online)</td>
<td>The Impact Analysis job runs in the foreground and the online report is displayed.</td>
</tr>
<tr>
<td>B (batch)</td>
<td>The Batch Job panel is displayed. For instructions, see “Specifying JCL options in batch mode” on page 48.</td>
</tr>
</tbody>
</table>

**Note**
If any of the data sets do not exist, the Allocate Data Set panel is displayed. For instructions, see “Allocating a data set for a specific job” on page 45.

8  Press **Enter** to generate the JCL.

---

**Setting Impact Analysis Options**

This task guides you through reviewing and changing options related to the Impact Analysis feature.

**To review or change Impact Analysis options**

1  Access the Impact Analysis Additional Options panel in one of the following ways:

   - From the SQL Explorer main menu:

     1  Type **0** (zero) for **Options**.
2 In the **Options** field, type 2 (for **Analysis**) and press **Enter**.

3 In the **Options** field, type 4 (for **Impact**) and press **Enter**.

- By typing any character in the **Options** field on the Impact Analysis Object Specification panel.

The Impact Analysis Object Options panel is displayed (Figure 51 on page 145).

**Figure 51: Impact Analysis Object Options panel (PSSPI020)**

```
PSSPI020 ---------------- Impact Analysis Object Options ----------------
Command ===> 

Specify options and press ENTER to continue.

SQL Statement(s) to Analyze
 /  ALL
   SELECT
   UPDATE
   INSERT
   DELETE

Recall ............ N   (Y/N) Recall Archived DBRM libraries
Report Level ........ B   (O=Object, S=SQL text, B=Both)
Generate Report Title .. Y   (Y/N)
   Title  IMPACT ANALYSIS REPORT
```

2 Complete the Impact Analysis Object Options panel:

a In the **SQL Statement(s) to Analyze** field, type any character in the space beside each of the SQL statements you want to analyze.

You can select individual statements or all statements.

b In the **Recall** field, specify whether to restore the partitioned data sets containing the DBRMs for the specified plans and packages.

c In the **Report Level** field, specify the information to display in the report, as follows:

- **Object** displays a report providing a summary of the impact on DBRMs, packages, and tables in the Impact Analysis job.

- **SQL text** displays a report providing a list of all impacted packages and DBRMs, including the SQL statement numbers and the SQL text.

- **Both** displays a report providing a summary of the impacted objects and a report providing a list of all impacted SQL statements.

d In the **Generate Report Title** field, specify whether to include a report title in the output.
If you plan to view the report online from the output data set, BMC Software recommends that you set Generate Report Title to N.

In the Title field, type a title for the report.

The maximum length for this field is 64 characters.
Preventing timestamp mismatches before production

This chapter includes the following sections:

■ “Overview of timestamp mismatches” on page 147
■ “Preventing timestamp mismatches” on page 148

Overview of timestamp mismatches

Synchronizing the DB2 precompiled code of the DBRM with the appropriate load module can often be a difficult task.

The timestamp of the DBRM is an encoded stored clock value, and the precise location of the corresponding timestamp in the load module varies, depending on the application language type. Mismatches of the load module’s timestamp and the DBRM that is bound to the DB2 subsystem can cause problems at execution time, such as -805 and -818 SQL errors.

Before you promote an application into production, you must ensure that your DBRMs and load modules have been correctly promoted to your production libraries. You must also ensure that all your DBRMs or packages are bound to your plans in each subsystem.

The SQL Explorer product provides the following functions to help you fulfill these requirements:

■ The **DBRM to Load Compare** [DBRM library (DBRMLIB) and load library comparison] function ensures that the timestamps in your DBRMLIB and load libraries match, and alerts you to any promotion problems in your production libraries. When you use this feature, SQL Explorer produces a report that lists the timestamp matches between the specified DBRM libraries and corresponding load module libraries.
The DBRMLIB to DB2 Catalog Compare (DBRMLIB and DB2 catalog comparison) function ensures that the timestamps in your DBRMLIB and DB2 catalog match, and alerts you when binds are missing. Binds are generated when the timestamp in the catalog is earlier than the timestamp in the DBRMLIB.

Preventing timestamp mismatches

Preventing timestamp mismatches prior to production is a two-stage process.

In the first stage, you compare the DBRMLIB and the load library to ensure that their timestamps match. In the second stage, you compare the DBRMLIB and the DB2 catalog to ensure that the objects were bound to your production plan.

When your DBRMs are synchronized with their corresponding load modules and they have been bound to your DB2 subsystems, your application is ready to be promoted to your production environment.

Comparing DBRMLIBs and load libraries

The DBRM and load library comparison process searches for matching members and report on whether timestamp mismatches exist.

Before rolling an application out to a user community, you must ensure that your DBRMLIBs and load libraries are synchronized.

You can compare different configurations of DBRMLIBs and load libraries. This function supports composite members, which are load library members containing more than one DBRM, or load library members with a different name than the DBRM.

For each DBRM data set member, the report returns one of the following possible statuses:

- MATCH—indicates that the DBRM member is referenced in the load module and that the contokens match.
- NO MATCH—indicates that the DBRM member is referenced in the load module, but that the contokens do not match.
- NO REF—indicates that the DBRM member is not referenced in the load module at all.
To compare DBRMLIBs and load libraries

1. On the SQL Explorer main menu, type 6 (for DBRM / Load Compare) and press Enter.

   The DBRM / Load Compare panel is displayed (Figure 52 on page 149).

   **Figure 52: DBRM / Load Compare panel (PSSPD010)**

   
PSSPD010 -------------- DBRM / Load Compare Specification -------------------
Command ===>  
Process Mode . . B (O=Online, B=Batch)  
DBRM Name . . . . PSS%  
Load Name . . . . PSS*  
_ DBRM / Load Libraries

2. In the **Process Mode** field, select the processing option for the comparison:
   
   - **Online** runs the comparison job online.
   
   - **Batch** creates JCL and runs the comparison job in batch. (default)

3. In the **DBRM Name** field, specify the name of the member to be used for the DBRM.

   You can use wildcard characters.

4. In the **Load Name** field, enter the load module name that contains the DBRM.

   You can use wildcard characters.

5. Type any character in the space beside **DBRM / Load Libraries** to specify DBRMLIB and load library data set names for the comparison and press Enter.
The DBRM / Load Compare Libraries panel is displayed (Figure 53 on page 150).

Figure 53: DBRM / Load Compare Libraries panel (PSSPD020)

On the DBRM / Load Compare Libraries panel, complete the following fields. You can enter up to seven data sets of each type.

Note

Enclose the data set name with single quotation marks. If you use quotation marks, DBRM/Load Compare uses the exact data set name you specify. Otherwise, DBRM/Load Compare attempts to append the data set name with the TSO prefix (if you have a prefix set) as the first node.

a In the DBRMLIB DSN fields, type the name of each DBRMLIB data set that you want to include.

b In the LOADLIB DSN fields, type the name of each load library data set that you want to include.

7 Press Enter.

Depending on the Process Mode you selected, one of the following events occurs:

<table>
<thead>
<tr>
<th>Process mode</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>O (online)</td>
<td>The Compare job runs in the foreground and the online report is displayed.</td>
</tr>
<tr>
<td>B (batch)</td>
<td>The Batch Job panel is displayed. For instructions, see “Specifying JCL options in batch mode” on page 48.</td>
</tr>
</tbody>
</table>
Comparing a DBRMLIB and the DB2 catalog

The DBRMLIB to DB2 Catalog comparison process verifies whether the DBRMLIB members for the application plan you are promoting to production have been bound to your production subsystems.

The process also indicates any DBRMs that have not been bound to the specified plans. Such DBRMs are either new DBRMs that have not yet been bound, or obsolete DBRMs that must be deleted.

Based on existing bind parameters, binds are generated when the timestamp in the catalog is earlier than the timestamp in the DBRMLIB.

The process also alerts you when the catalog contoken is newer than the contoken in your production library. In this situation, you must analyze your promotion process to determine whether the bind was performed in error, or the DBRM member was not promoted properly. In the latter case, you might need to repeat the DBRM/Load comparison process in “Comparing DBRMLIBs and load libraries” on page 148.

Each member in the specified DBRMLIBs is analyzed and placed into one of the message categories shown in Table 13 on page 151.

Table 13: Message categories for the DBRMLIB to DB2 Catalog Compare function

<table>
<thead>
<tr>
<th>Message category</th>
<th>Explanation and action</th>
</tr>
</thead>
</table>
| THE FOLLOWING DBRMLIB MEMBERS ARE NOT CURRENTLY BOUND TO ANY OF THE PLANS SPECIFIED IN YOUR INPUT PARMS | **Explanation:** Members were found in the concatenation of the DBRMLIBs, but were not found in any of the plans specified in the input.  
**Action:** Review these members to determine whether they represent new DBRMs that should be bound, or whether they are obsolete DBRMs that you should should remove from the DBRMLIB. |
<table>
<thead>
<tr>
<th>Message category</th>
<th>Explanation and action</th>
</tr>
</thead>
</table>
| SEVERE ERROR: PLAN NOT BOUND - MEMBER NOT FOUND IN THE SPECIFIED DBRMLIBS: DBRM DBRMLibName IN PLAN planName LAST BOUND FROM catalogPDSName | **Explanation:** A DBRM was bound to the plan you specified, but SQL Explorer could not find the DBRM in the DBRMLIB concatenation in the input.  
When a plan contains DBRMS instead of packages, binding the plan without all of the DBRMs previously bound to it drops the DBRMs from the plan, and the DBRMs are no longer valid under the plan. Instead of taking this action, SQL Explorer prints a severe message including the DBRMLIB, plan name, and DBRM name to warn you about this potential problem. It is possible to get this error more than once for the same plan, if more than one DBRM was not found in the DBRMLIBs.  
**Action:** Review these DBRMs to determine whether they should be dropped from the plan, whether you should specify their DBRMLIBs in the input, or whether they were promoted improperly. |
| SEVERE ERROR: OBJECT NOT BOUND - CATALOG CONTOKEN NEWER: inputDBRMLIBMember TO PLAN planName LAST BOUND FROM catalogPDSName inputDBRMLIBMember TO COLLID collectionID LAST BOUND FROM catalogPDSName | **Explanation:** The contoken for this member was more recent in the catalog than in the DBRMLIB. No bind is generated for the plan or package containing the member.  
**Action:** Review this list to determine whether an incorrect DBRMLIB was specified, or whether there was a problem with the promotion process. |
| WARNING: MEMBER NOT BOUND - DUPLICATE MEMBERS: successiveInputDBRMLIBMember DUPLICATE OF previousInputDBRMLIBMember | **Explanation:** A DBRM member was found in more than one DBRMLIB in the input. SQL Explorer processes only the first DBRM member found in the concatenation and reports all successive DBRM members as duplicates.  
**Action:** Review this list to determine whether the correct DBRMLIB concatenation was specified. |
| WARNING: MEMBERS NOT BOUND - INVALID DBRM: inputDBRMLIBMember | **Explanation:** The listed DBRM member did not have the format of a valid DBRM.  
**Action:** Review this list to determine whether the member can be deleted, or whether the program was pre-processed without containing any SQL statements. |
| WARNING: PACKAGE NOT BOUND - MEMBER NOT FOUND IN THE SPECIFIED DBRMLIBS: PACKAGE packageName IN COLLID collectionID LAST BOUND FROM catalogPDSName | **Explanation:** The specified plan had a package bound to it, but the name of the package did not match any member names in the DBRMLIB. A bind for the plan may have been generated, depending on the comparison of contokens for other packages and DBRMs included in the plan. In this instance, a warning is issued because binding the plan does not invalidate the package from the plan.  
**Action:** Review this list to determine whether the correct DBRMLIB concatenation was specified. |
To compare a DBRMLIB and the DB2 catalog

1. Copy the PSSD2CJB member from the hlq.llqSAMP data set to a new data set.  
   
   **Note**  
   The name of hlq.llqSAMP varies depending on your installation options and might be hlq.BMCSAMP.

2. Edit the tokens in your copied PSSD2CJB member.

   On the **Command** line, type `C token newToken`, and press **Enter**, where `token` is the existing token name and `newToken` is the new token name you specify. Editable tokens are defined in Table 14 on page 153.

**Table 14: Editable tokens in the PSSD2CJB member**

<table>
<thead>
<tr>
<th>Token</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>jobname</td>
<td>Name of job</td>
</tr>
<tr>
<td>acct</td>
<td>Job account number</td>
</tr>
<tr>
<td>msgclass</td>
<td>Job message class</td>
</tr>
<tr>
<td>class</td>
<td>Job class</td>
</tr>
<tr>
<td>ssid</td>
<td>DB2 subsystem</td>
</tr>
<tr>
<td>PSS_plan</td>
<td>Name of SQL Explorer plan</td>
</tr>
</tbody>
</table>
Preventing timestamp mismatches

<table>
<thead>
<tr>
<th>Token</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>dsnload</td>
<td>DB2 DSNLOAD data set</td>
</tr>
<tr>
<td>bmcpswd</td>
<td>BMCPSWD data set</td>
</tr>
<tr>
<td>BMC_bblink</td>
<td>Name of the LOAD data set where PSSD2C module resides</td>
</tr>
<tr>
<td>user_sysprint</td>
<td>User data set for SYSPRINT reports or specify SYSOUT</td>
</tr>
<tr>
<td>user_summary</td>
<td>User data set for SUMMARY report or specify SYSOUT</td>
</tr>
<tr>
<td>user_bind</td>
<td>User data set for generated BIND statements or specify SYSOUT</td>
</tr>
<tr>
<td>DBRMLIB1</td>
<td>Name of the first DBRM data set to be searched</td>
</tr>
<tr>
<td>DBRMLIB2</td>
<td>Name of the second DBRM data set to be searched</td>
</tr>
<tr>
<td>DBRMLIB3</td>
<td>Name of the third DBRM data set to be searched</td>
</tr>
<tr>
<td></td>
<td>Include additional DBRMLIB lines for other DBRM data sets as needed.</td>
</tr>
<tr>
<td>planname</td>
<td>Application plan to be compared to DBRMLIBs</td>
</tr>
</tbody>
</table>

The following guidelines apply to editing the PSSD2CJB data set member:

- You must change all occurrences of the tokens you specify, or the comparison job fails.
- You can add any number of DBRMLIBs in ascending DD sequence order. Delete or comment out any DBRMLIB lines you do not need.
- You can replicate the `planname` parameter as needed to include all of the plans that make up your application.
- You can use the following wildcard characters:

<table>
<thead>
<tr>
<th>Wildcard character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ (underscore)</td>
<td>Matches any single character</td>
</tr>
<tr>
<td>% (percent sign)</td>
<td>Matches a string of zero or more characters</td>
</tr>
<tr>
<td>* (asterisk)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 54 on page 154 shows an example of an edited PSSD2CJB member:

**Figure 54: Edited PSSD2CJB member**

```plaintext
//RDAJGLPW JOB (5216),MSGCLASS=X,CLASS=A
//*
//*-----------------------------------------------------------------
//*     DBRMLIB TO DB2 CATALOG COMPARE JCL
//*-----------------------------------------------------------------
//PSSD2C EXEC PGM=PSSD2C,PARM='DECA DAAVRMD1'
//STEPLIB    DD DISP=SHR,DSN=BMCPERF.LOAD
//          DD DISP=SHR,DSN=SYS2.DSNLOAD
//          DD DISP=SHR,DSN=AFDQA.AFD.PASSWORD
```
Submit the DBRMLIB to DB2 Catalog comparison job by typing **SUB** on the **Command** line, and pressing **Enter**.

The job produces a report containing a BIND statement, as shown in Figure 55 on page 155, and SYSPRINT and SUMMARY sections, as shown in Figure 56 on page 155 and Figure 57 on page 156 respectively.

**WARNING**

Running long name objects in PSSD2CJB JCL with a data set logical record length (LRECL) setting that is less than 255 causes the BIND statement to wrap the long names and generate an error when the BIND statement is executed.

---

**Figure 55: BIND statement generated by PSSD2CJB job**

```sql
DSN SYSTEM(DEBF) 
BIND PLAN(JLG7PGDM) MEMBER(APPTST01,APPTST41,APPTST50) - 
PKLIST(PK01HM07.*) LIBRARY('RDAEXN.HVM7.DBRMLIB'- 
,'RDAJLG.TESTING.DBRM') QUALIFIER(IOD) NODEFER(PREPARE) VALIDATE(BIND) - 
ISOLATION(CS) CACHESIZE(1024) CURRENTDATA(NO) - 
DEGREE(ANY) DISCONNECT(EXPLICIT) SQLRULES(DB2) - 
ACQUIRE(USE) RELEASE(DEALLOCATE) EXPLAIN(NO) - 
NOREOPT(VARS) KEEP_DYNAMIC(NO) DBPROTOCOL(DRDA) - 
ENCODING(37) IMMEDWRITE(NO) - 
BIND PACKAGE(PK01HM07) MEMBER(IODSTAG1) - 
LIBRARY('RDAEXN.HVM7.DBRMLIB') QUALIFIER(IOD) SQLERROR(NOPACKAGE) VALIDATE(BIND) - 
ISOLATION(CS) CURRENTDATA(NO) DEGREE(ANY) - 
RELEASE(COMMIT) EXPLAIN(NO) NOREOPT(VARS) - 
KEEP_DYNAMIC(NO) DBPROTOCOL(DRDA) ENCODING(37) - 
IMMEDWRITE(NO) - 
END
```

**Figure 56 on page 155** shows the SYSPRINT.

---

**Figure 56: SYSPRINT generated by PSSD2CJB job**

```sql
SQL EXPLORER DBRMLIB TO CATALOG COMPARE 
(C) COPYRIGHT 1996 - 2015 BMC SOFTWARE, INC. 
RUN DATE:  MON MAR 16 23:54:51 2015 
DB2 SSID:  DIY6   DB2 VERSION: 10.1 
WARNING: PACKAGE NOT BOUND - MEMBER NOT FOUND IN THE SPECIFIED DBRMLIBS: 
PACKAGE IODWL926 IN COLLID PK01HM07 LAST BOUND FROM RDAEXN.C7.DBRM 
BINDS GENERATED: CATALOG CONTOKENS OLDER: 
RDAJLG.TESTING.DBRM(APPTST01) TO PLAN JLG7PGDM LAST BOUND FROM SAME PDS 
RDAEXN.HVM7.DBRMLIB(IODSTAG1) TO COLLID PK01HM07 LAST BOUND FROM SAME PDS
```
Objects not bound - contokens match:
RDAJLG.TESTING.DBRM(APPTST41) TO PLAN JLG7PGDM LAST BOUND FROM SAME PDS
RDAJLG.TESTING.DBRM(APPTST50) TO PLAN JLG7PGDM LAST BOUND FROM SAME PDS
RDAEXN.HVM7.DBRMLIB(IODOPT77) TO COLLID PK01HMD7 LAST BOUND FROM SAME PDS
RDAEXN.HVM7.DBRMLIB(IODWLOA1) TO COLLID PK01HMD7 LAST BOUND FROM SAME PDS
RDAEXN.HVM7.DBRMLIB(IODWLTTT) TO COLLID PK01HMD7 LAST BOUND FROM SAME PDS
RDAEXN.HVM7.DBRMLIB(IODWLV72) TO COLLID PK01HMD7 LAST BOUND FROM SAME PDS
RDAEXN.HVM7.DBRMLIB(IODWL411) TO COLLID PK01HMD7 LAST BOUND FROM SAME PDS

Figure 57 on page 156 shows the SUMMARY.

Figure 57: SUMMARY generated by PSSD2CJB job

--SQL EXPLORER 11.02.00 PARMS- RELEASED MAY 2015 COMPILED MAR 11 2015
(C) COPYRIGHT 1996 - 2015 BMC SOFTWARE, INC.

RUN DATE:  MON MAR 16 23:54:51 2015
DB2 SSID:  DIY6   DB2 VERSION:  10.1

The following DBRMLIB members are not currently bound to any of the plans specified in
your input parms:
ALIASPG  RDAJLG.TESTING.DBRM
BIVBPLTB RDAJLG.TESTING.DBRM
COMPSQLR RDAJLG.TESTING.DBRM
CURSCHEM RDAJLG.TESTING.DBRM
DOTCCA99 RDAJLG.TESTING.DBRM
DOTCCBND RDAJLG.TESTING.DBRM
DOTCCRUN RDAJLG.TESTING.DBRM
DOTCC999 RDAJLG.TESTING.DBRM
DOTDBCSK RDAJLG.TESTING.DBRM
DOTMLPSQ RDAJLG.TESTING.DBRM
DOTITBC RDAJLG.TESTING.DBRM

Preventing timestamp mismatches
Integrating CATALOG MANAGER for DB2 with SQL Explorer

This chapter includes the following sections:

- “Overview of CATALOG MANAGER integration” on page 157
- “Before you begin” on page 158
- “Commands to access SQL” on page 158
- “Explaining a DBRM, package, or plan” on page 160
- “Explaining a statement from a DBRM or package” on page 161
- “Explaining a statement from the SQL_Table” on page 164
- “Editing the SQL statement by using the SQLX Edit Macro” on page 165

Overview of CATALOG MANAGER integration

By integrating the SQL Explorer for DB2 product with CATALOG MANAGER, you can access and explain SQL in several ways.

To access SQL statements in CATALOG MANAGER, you can create lists of DB2 objects and navigate through the objects to SQL statements that are stored in the DB2 catalog, or you can use the CATALOG MANAGER SQL_Table to access SQL statements that were saved in a CATALOG MANAGER session.

This section describes how to use CATALOG MANAGER commands to navigate to SQL Explorer. For detailed information about the functions of CATALOG MANAGER, see the CATALOG MANAGER for DB2 User Guide.
Before you begin

Before you attempt to use SQL Explorer with CATALOG MANAGER, verify that the ACTPSS CLIST has been installed and customized for SQL Explorer.

For more information, see the *Installation System Reference Manual*.

Commands to access SQL

CATALOG MANAGER enables you to issue simple line commands to access SQL, as follows:

- Issue the BMCEXPLORE command to Explain a DBRM, package, or plan (Figure 58 on page 158).

Figure 58: Using the BMCEXPLORE command
- Issue the DESCRIBE and GET commands to Explain an SQL statement that is stored in a DBRM or a package (Figure 59 on page 159).

**Figure 59: Using the DESCRIBE and GET commands**
Issue the ANALYZE command to Explain or edit a DML statement from the CATALOG MANAGER SQL_Table (Figure 60 on page 160).

Figure 60: Using the ANALYZE command

---

Explaining a DBRM, package, or plan

In this task, you use the BMCEXPLORERE command to access SQL Explorer to Explain DBRMs, packages, and plans.

1. Generate a list of eligible objects.
In the Command (Cm) column beside the source object, type BMCEXPLORE (BMCEX) as shown in Figure 61 on page 161.

Figure 61: BMCEXPLORE command issued against plan

```
ACTPJNL1 -------------------------  PLAN LIST  -------------------------------
Command ===>                                                  Scroll ===> PAGE
01
CMD will show commands for this list. Type command and press ENTER
LISTS: AL CA CI DB DM DP IM IS IX MQT MX PA PG PI PL SY TB TS UA US VW
LIKE ACT%
Cm Plan     Owner   Valdat Isolat Valid Operat Acq Rel Bound      Member   Dyn
----v----1----v----2----v----3----v----4----v----5----v----6----v----7----v----
BMCEX    T91DDB RDATLF     B      S     Y      Y    U   C  2010-04-27
ACT91DDB RDATLF B S Y Y U C 2015-04-27
ACT91DDG RDATLF B S Y Y U C 2015-04-27
ACT91DDH RDATLF B S Y Y U C 2015-04-27
ACT91DDK RDATLF B S Y Y U C 2015-04-27
ACT91DDL RDATLF B S Y Y U C 2015-04-27
ACT91DLM RDATLF B S Y Y U C 2015-04-27
ACT91DLS RDATLF B S Y Y U C 2015-04-27
ACT91DML RDATLF B S Y Y U C 2015-04-27
ACT91DMM RDATLF B S Y Y U C 2015-04-27
ACT910DB RDATLF B S Y Y U C 2015-04-24
ACT910DE RDATLF B S Y Y U C 2015-04-24
ACT910DG RDATLF B S Y Y U C 2015-04-24
ACT910DH RDATLF B S Y Y U C 2015-04-24
ACT910DK RDATLF B S Y Y U C 2015-04-24
ACT910DL RDATLF B S Y Y U C 2015-04-24
ACT910DM RDATLF B S Y Y U C 2015-04-24
```

3 Press Enter to begin SQL Explorer Explain processing.

For more information, see “Explaining SQL statements” on page 63.

## Explaining a statement from a DBRM or package

In this task, you use the DESCRIBE command to access SQL Explorer to Explain individual SQL statements in DBRMs and packages.

The following types of statements are valid:

- SELECT
- INSERT
- UPDATE
- DELETE
- DECLARE CURSOR (static SQL only)

### To Explain a statement from a DBRM or package

1 Generate a list of eligible objects.

2 In the Command (Cm) column beside the source object, type DESCRIBE.
3 Press Enter.

The DESCRIBE panel for the source object is displayed (Figure 62 on page 162).

Figure 62: DESCRIBE panel

<table>
<thead>
<tr>
<th>Command</th>
<th>Scroll</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package = ACT_DEV0910.ACTALAL</td>
<td>PAGE</td>
</tr>
<tr>
<td>FROM SYSIBM.SYSPACKAGE</td>
<td></td>
</tr>
<tr>
<td>Location . . . .</td>
<td>Collid . . . .</td>
</tr>
<tr>
<td>Name . . . .</td>
<td>Contoken . . .</td>
</tr>
<tr>
<td>Owner . . . .</td>
<td>Creator . . .</td>
</tr>
<tr>
<td>Timestamp . . .</td>
<td>Bindtime . . .</td>
</tr>
<tr>
<td>Qualifier . . .</td>
<td>ACT91D</td>
</tr>
<tr>
<td>Avgsize . . . .</td>
<td>Pksize . . . .</td>
</tr>
<tr>
<td>Valid . . . .</td>
<td>Oper . . . .</td>
</tr>
<tr>
<td>Validate . . .</td>
<td>Isolation . .</td>
</tr>
<tr>
<td>Release . . .</td>
<td>Explain . .</td>
</tr>
<tr>
<td>Quote . . . .</td>
<td>N(‘)</td>
</tr>
<tr>
<td>Hostlang . . .</td>
<td>Charset . . .</td>
</tr>
<tr>
<td>Mixed . . . .</td>
<td>Dec31 . . .</td>
</tr>
<tr>
<td>Deferprep . . .</td>
<td>SQLerror . .</td>
</tr>
<tr>
<td>Remote . . .</td>
<td>PCTimestamp . .</td>
</tr>
<tr>
<td>IBMreqd . . .</td>
<td>Version . . .</td>
</tr>
<tr>
<td>PDSname . . .</td>
<td>Degree . . .</td>
</tr>
<tr>
<td>Group Member . . .</td>
<td>Dynamicrules . .</td>
</tr>
<tr>
<td>Reoptvar . . .</td>
<td>Deferprepare . .</td>
</tr>
<tr>
<td>Keepdynamic . . .</td>
<td>Pathschemas . .</td>
</tr>
<tr>
<td>Type . . .</td>
<td>DBprotocol . .</td>
</tr>
<tr>
<td>Functions . . .</td>
<td>Opthint . .</td>
</tr>
<tr>
<td>Encoding CCSID . .</td>
<td>Immedwrite . .</td>
</tr>
<tr>
<td>Relbound . . .</td>
<td>Catencode . .</td>
</tr>
<tr>
<td>Remarks . . .</td>
<td>OownerType . .</td>
</tr>
<tr>
<td>Rounding . . .</td>
<td>Distribute . .</td>
</tr>
<tr>
<td>Lastused . . .</td>
<td>01/01/0001</td>
</tr>
<tr>
<td>Stmtno</td>
<td>Stmt</td>
</tr>
<tr>
<td>396</td>
<td>PREPARE ALOOSRCH</td>
</tr>
<tr>
<td>400</td>
<td>DECLARE ALOOCRCSR CURSOR FOR ALOOSRCH</td>
</tr>
<tr>
<td>403</td>
<td>OPEN ALOOCRCSR</td>
</tr>
<tr>
<td>410</td>
<td>FETCH ALOOCRCSR USING DESCRIPTOR :* sqlda</td>
</tr>
<tr>
<td>416</td>
<td>CLOSE ALOOCRCSR</td>
</tr>
<tr>
<td>451</td>
<td>DECLARE ALOOCRCSR CURSOR FOR</td>
</tr>
</tbody>
</table>
| 4 | Scroll down the DESCRIBE panel to locate the statement.
5 On the Command line, type GET nnn, where nnn is the number of the statement to be Explained. (as illustrated in Figure 63 on page 163).

**Figure 63: Use of GET subcommand**

```
AUBMODS5 Package: ACT_DEV0910.ACTALAL Line 22 of 104 Col 1 80
Command ====> GET 451 Scroll ====> PAGE

Reoptvar . . . . N
Keepdynamic. . N
Type . . . . . . D
Functions . . . . 2010-06-17-23.37.01.. Opthint. . . .
Encoding CCSID : 37 Immedwrite . . . . N
Relbound . . . . M Catencode. . . .
Remarks . . . . . Owntype. . . .
Rounding . . . . E Distribute . . . N
Lastused . . . . 01/01/0001

<table>
<thead>
<tr>
<th>Stmtno</th>
<th>Stmt</th>
</tr>
</thead>
<tbody>
<tr>
<td>396</td>
<td>PREPARE ALOOSRCH FROM :pSEARCHSTR</td>
</tr>
<tr>
<td>400</td>
<td>DECLARE ALOOCR SR CURSOR FOR ALOOSRCH</td>
</tr>
<tr>
<td>403</td>
<td>OPEN ALOOCR SR</td>
</tr>
<tr>
<td>410</td>
<td>FETCH ALOOCR SR USING DESCRIPTOR :* sqlda</td>
</tr>
<tr>
<td>416</td>
<td>CLOSE ALOOCR SR</td>
</tr>
<tr>
<td>451</td>
<td>DECLARE ALO1CR SR CURSOR FOR SELECT A.* FROM AUDIT_LOG A FOR FETCH ONLY</td>
</tr>
</tbody>
</table>
```

6 Press Enter.

The Confirm SQL panel is displayed (Figure 64 on page 163).

**Figure 64: Confirm SQL panel**

```
AUBDYN Confirm SQL 1 to 3 of 3 Scroll ====> PAGE
Command ====> RDACRJ2

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current SQLID</td>
<td>RDACRJ2</td>
</tr>
<tr>
<td>Edit Options</td>
<td>N</td>
</tr>
<tr>
<td>Edit SQL</td>
<td>Y/N Modify SQL processing options</td>
</tr>
<tr>
<td>Save in SQL table</td>
<td>N</td>
</tr>
<tr>
<td>Name of saved SQL</td>
<td>20100525_151539</td>
</tr>
<tr>
<td>Save in PDS</td>
<td>N</td>
</tr>
<tr>
<td>PDS(member)</td>
<td>RDACRJ2.BMCCAT.WORK(TEST)</td>
</tr>
<tr>
<td>Analysis</td>
<td>N</td>
</tr>
<tr>
<td>Execute SQL</td>
<td>Y/N Execute the SQL</td>
</tr>
<tr>
<td>FROM</td>
<td>AUDIT_LOG A</td>
</tr>
<tr>
<td>FOR</td>
<td>FETCH ONLY</td>
</tr>
</tbody>
</table>
```

7 To continue, choose one of the following procedures:

- To edit the SQL statement before Explaining it, or to Explain or execute the SQL statement on another subsystem, see “Editing the SQL statement by using the SQLX Edit Macro” on page 165.
- To begin SQL Explorer Explain processing, in the Analysis field, enter Y. For more information, see “Explaining SQL statements” on page 63.
Explaining a statement from the SQL_Table

In this task, you access SQL Explorer to explain SQL statements that have been saved in the CATALOG MANAGER SQL_Table.

The following types of statements are valid:

- SELECT
- INSERT
- UPDATE
- DELETE
- DECLARE CURSOR (static SQL only)

To Explain a statement from the SQL_Table

1. Display the CATALOG MANAGER SQL_Table List panel.

   For more information, see the CATALOG MANAGER for DB2 User Guide.

2. In the Command (Cmd) column beside the source statement, type ANALYZE (as shown in Figure 65 on page 164).

Figure 65: SQL_Table List panel

3. Press Enter.
The Confirm SQL panel is displayed (Figure 66 on page 165).

**Figure 66: Confirm SQL panel (AUBDYN)**

<table>
<thead>
<tr>
<th>Command</th>
<th>Confirm SQL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current SQLID</td>
<td>RDACRJ2</td>
</tr>
<tr>
<td>Edit Options</td>
<td>N Y/N Modify SQL processing options</td>
</tr>
<tr>
<td>Edit SQL</td>
<td>Y/N Edit SQL before executing</td>
</tr>
<tr>
<td>Save in SQL table</td>
<td>N A/Y/R/N A/Y-Append, R-Replace</td>
</tr>
<tr>
<td>Name of saved SQL</td>
<td>20100525_151539</td>
</tr>
<tr>
<td>Save in PDS</td>
<td>Y/N Save SQL in PDS</td>
</tr>
<tr>
<td>PDS(member)</td>
<td>RDACRJ2.BMCCAT.WORK(TEST)</td>
</tr>
<tr>
<td>Analysis</td>
<td>Y/N Call SQL Explorer for EXPLAIN</td>
</tr>
<tr>
<td>Edit/Browse data</td>
<td>E/B/N Call the Table Editor</td>
</tr>
<tr>
<td>Execute SQL</td>
<td>Y/N Execute the SQL</td>
</tr>
<tr>
<td>SQL</td>
<td>SELECT DISTINCT INSTANCE FROM SYSIBM.SYSINDEXSPACESTATS; SELECT PARTITION, INSTANCE, DBNAME, SUBSTR(NAME,1,18), UPDATESTATSTIME FROM SYSIBM.SYSINDEXSPACESTATS WHERE INSTANCE = 2 ORDER BY 3, 4, 1, 2, 5 ;</td>
</tr>
</tbody>
</table>

4 To continue, choose one of the following procedures:

- To edit the SQL statement before explaining it, or to explain or execute the SQL statement on another subsystem, see “Editing the SQL statement by using the SQLX Edit Macro” on page 165.
- To begin SQL Explorer Explain processing, in the Analysis field, enter Y. For more information, see “Explaining SQL statements” on page 63.

**Editing the SQL statement by using the SQLX Edit Macro**

1 Explain a statement from the SQL_Table.

For information, see “Explaining a statement from the SQL_Table” on page 164.

2 On the Confirm SQL panel, in the Edit SQL field, type Y and press Enter.
An ISPF edit panel is displayed (Figure 67 on page 166).

**Figure 67: ISPF Edit panel**

![ISPF Edit panel image]

---

1. **Note**
SQLX supports datasets with a maximum width of 256 bytes.

2. Depending on the programming language, the begin and end statements may have different keywords.

3. **Note**
If the text spans multiple panels, you might get an INVALID COMMAND message when you press **PF8** (DOWN) because ISPF does not recognize QQ as a valid command. Ignore the INVALID COMMAND message while you continue to mark the statement, then proceed to **Step 5 on page 166** to initiate the macro. Alternatively, you can use the Q **nn** command at the beginning of the statement, where **nn** is the number of lines to search to locate the end of the statement.

4. On the **Command** line, type **SQLX ssid**, where **ssid** is the ID of the subsystem on which you want to Explain the SQL statement.

5. Press **Enter** to begin SQL Explorer Explain processing.

For more information, see “Explaining SQL statements” on page 63.
Working with statistics and table declarations

This part presents topics relating to statistics and table declarations.

Chapters:

- Migrating access path statistics
- Generating table declarations automatically
Migrating access path statistics

This chapter includes the following sections:

- “Overview of statistics migration” on page 169
- “Migrate Access Path Statistics process” on page 170
- “Specifying options for the Migrate Access Path Statistics job” on page 171
- “Setting Migrate Access Path Statistics options” on page 176
- “Specification of patterns for translating object names” on page 179
- “Editing JCL to include additional translations or objects to migrate” on page 180
- “Running the Migrate Access Path Statistics Extract job” on page 182
- “Editing the Update file” on page 186
- “Running the Update job” on page 186

Overview of statistics migration

DB2 allows users with the proper authority to update certain columns in the DB2 catalog.

These columns, normally populated by the RUNSTATS utility, are statistical in nature and indicate the size and organizational state of the data and indexes.

The Migrate Access Path Statistics function of the SQL Explorer product enables you to copy statistics from one DB2 catalog to another, or to the same subsystem’s DB2 catalog. For example, you can migrate production statistics to a test environment to view the actual access paths that the DB2 optimizer selects, without the cost of replicating the production environment. This example assumes that your DB2 environments are the same—for instance, that buffer pools, EDM pools, and so on.
are of comparable sizes in both environments. You can preserve the test data and still have the benefit of seeing the access paths based on production data.

You can use the Explain function to analyze SQL in a test environment and store the access paths in the History tables. After you migrate the production statistics to the test environment, you can re-Explain the SQL in combination with the Compare function to get information about changes in access paths that occur prior to actually binding the SQL in production. Alternatively, if you run the RUNSTATS utility and perform the rebinds in production, then note a significant or sudden drop in performance, you can quickly copy your saved statistics to the catalog. Then, using SQL Explorer, you can Explain the production statistics later in a non-production situation to determine why the access paths changed.

**Note**
Any subsequent execution of the RUNSTATS utility alters the values that were updated as part of the Migrate Access Path Statistics process. As an audit facility, you can specify a STATSTIME to be included in the Update process. If STATSTIME is omitted, CURRENT TIME is used.

For more information, view the Quick Course "SQL Explorer for DB2—Migrating Access Path Statistics".

### Migrate Access Path Statistics process

Migrating access path statistics is a two-step operation.

1. The first step is to extract the statistics from the source DB2 subsystem. This process creates a script containing the access path statistics to update on the target DB2 catalog.

2. This script file is the input to the second step, which is a DB2 Update step, run against the target subsystem (SSID).

The user administering the Extract or the Update jobs must have the appropriate authorities to access the DB2 catalog.

Because of the method used to extract the statistics, SQL Explorer does not validate that the objects exist in the target environment. If the objects do not exist, the Update job receives the following message: The following row not found for update.

**Note**
Migrate Access Path Statistics cannot process temporary tables, because temporary tables have no column definitions to examine.
In this task, you use the initial panels of the Migrate Access Path Statistics function to specify processing options for the job.

**Before you begin**

Because the Update job updates the DB2 catalog’s statistics, you might want to back up the statistics before you update them. Use one of the following methods to back up the DB2 catalog statistics:

- Before running the Update statements, run Migrate Access Path Statistics, using the target subsystem as the target and source. These original catalog statistics are preserved in the Update file. You can use this file to restore the values to their original state.

- Run a DSN1COPY of DSNDDB06. This action provides a point-in-time backup of your DB2 catalog.

**WARNING**

Migrate Access Path Statistics uses the UPDATE data set, which is a sequential data set. Be careful not to overwrite any data in this data set that might contain statistics that you want to save.

**To specify options for the Migrate Access Path Statistics job**

1. On the SQL Explorer main menu, type 7 (for Migrate Access Path Statistics).

2. Change the SSID locations, if necessary, and press Enter.

The Migrate Access Path Statistics Specification panel (Figure 68 on page 172) is displayed.
You cannot edit the **Process Mode**, because Migrate Access Path Statistics jobs run in batch mode.

**Figure 68: Migrate Access Path Statistics Specification panel (PSSPM010)**

<table>
<thead>
<tr>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSSPM010 --------- Migrate Access Path Statistics Specification ---------------</td>
</tr>
<tr>
<td>Process Mode   B (B=Batch)</td>
</tr>
<tr>
<td>From SSID   . . DEDR</td>
</tr>
<tr>
<td>To SSID    . . DHZB</td>
</tr>
<tr>
<td>From Location   DEBF</td>
</tr>
<tr>
<td>To Location    DIY</td>
</tr>
<tr>
<td>Database and Table Space Name to Migrate (wildcards may be used)</td>
</tr>
<tr>
<td>Database.Tablespace  PSSQX91.PSSS0391</td>
</tr>
<tr>
<td>(Optional) Translate Current Object Names to New Names (press F1 for Help)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Current Value</th>
<th>New Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB Name</td>
<td>PSSQX91</td>
</tr>
<tr>
<td>TS Name</td>
<td>PSSS0391</td>
</tr>
<tr>
<td>TB Creator</td>
<td>PSS</td>
</tr>
<tr>
<td>IX Creator</td>
<td>PSS</td>
</tr>
<tr>
<td>IX Name</td>
<td>PSSX0%_D91S03T01</td>
</tr>
<tr>
<td></td>
<td>TSSX0%_D91S03T01</td>
</tr>
</tbody>
</table>

**Note**

- In this example, the extract process uses DDF from subsystem DEDR to connect to and extract statistics from DEBF. Similarly, the update process uses DDF from subsystem DHZB to connect to and update DIY with the statistics from DEBF.

- In the JCL that SQL Explorer generates, the **From Location** and **To Location** fields appear as DDFLOC SYSIN cards.

3 In the **From SSID** field, type the subsystem name (the *source* subsystem) from which you want to migrate access path statistics.

4 In the **To SSID** field, type the name of the subsystem (the *target* subsystem) to which you want to migrate access path statistics.

The target subsystem can be on a version of DB2 that is different from the source subsystem.

**Note**

If SQL Explorer has not been installed for a SSID that you entered, SQL Explorer issues a **CONFIG ERROR** (such as **UNABLE TO DETERMINE DB2 LOADLIBS FOR SSID 'XXXX'**). Press **F1** to see a more detailed description of the error.

For more information, see the *BMC Products and Solutions for DB2 Customization Guide*. 

---

**Note**

- In the JCL that SQL Explorer generates, the **From Location** and **To Location** fields appear as DDFLOC SYSIN cards.
5 In the **From Location** field, type the name of the DDF location from which subsystem access path statistics is migrated.

6 In the **To Location** field, type the name of the DDF location to which access path statistics is migrated.

*Note*

The **To Location** and **From Location** fields represent the DDF location names defined in the DDF table SYSIBM.LOCATIONS.

- The **To Location** is the DDF location name defined in the SYSIBM.LOCATIONS table associated with the To SSID subsystem.
- The **From Location** is the DDF location name defined in the SYSIBM.LOCATIONS associated with the From SSID subsystem.

These parameters let you extract or update DB2 statistics from the defined DB2 subsystems to the specified locations using DB2. The fields are optional, and can be specified separately with either or both fields left blank.

7 In the **Database and Table Space Name to Migrate** field, type the name of the table space that you want to migrate, in `databaseName.tableSpaceName` format.

Table 15 on page 173 shows the wildcard characters you can use in this field:

<table>
<thead>
<tr>
<th>Wildcard character</th>
<th>Matches</th>
</tr>
</thead>
<tbody>
<tr>
<td>_ (underscore)</td>
<td>Any single character</td>
</tr>
<tr>
<td>% (percent sign)</td>
<td>A string of zero or more characters</td>
</tr>
<tr>
<td>* (asterisk)</td>
<td></td>
</tr>
</tbody>
</table>

8 *(optional)* If the target object names are different from the source object names, apply them at **Translate Current Object Names to New Names**.

Specify a **Current Value** for each of the source object names and a **New Value** for each of the target object names. You can specify values for the following objects:

- **DB Name** (database name)
- **TS Name** (table space name)
- **TB Creator** (table creator)
- **IX Creator** (index creator)
- **IX Name** (index name)

You can use the wildcard characters shown in Table 15 on page 173.
Note
The objects that you want to update must already exist on the target subsystem. If the objects do not exist, the Update job issues the following message: The following row not found for update.

In most cases, the panel fields used to enter object names are translated to uppercase characters. However, the TB Creator, IX Creator, and IX Name object names are case sensitive. To enter an object name containing lowercase characters, delimit the object name by enclosing the name in quotation marks ("."). Such delimited object names are used as they are specified and are not translated to uppercase.

Note
If the object name includes a literal question mark, enclose the question mark in a double set of quotation marks, and enclose the object name in quotation marks. For example, to represent table creator “RDAKG?” enter "RDAKG""?" in the TBCREATOR field.

The Migrate Access Path Statistics Specification panel allows for only one set of pattern translation input parameters. You can manually insert additional translation sets by editing the Migrate Access Path Statistics JCL. For more information, see “Specification of patterns for translating object names” on page 179.

9 (optional) Type any character in the space beside the Additional Options field to select additional Migrate Access Path Statistics options.

For more information, see “Setting Migrate Access Path Statistics options” on page 176.

10 Press Enter to display the Batch Job panel.

For information about completing this panel, see “Specifying JCL options in batch mode” on page 48.

11 Press Enter to view the generated JCL.

Figure 69 on page 174 shows an example of the sample generated JCL.

Note
If any of the data sets do not exist, the Allocate Data Set panel is displayed. For instructions, see “Allocating a data set for a specific job” on page 45.

Figure 69: Generated JCL for Migrate Access Path Statistics

```
//MVSIXTM JOB (PDOM.1605A), 'MIGRATE STATS',REGION=0K,
//             MSGCLASS=X,CLASS=A,NOTIFY=SYSUID
/*ROUTE XEQ BMCPLX1
//*JOBPARM SYSAFF=DB2A
//*-------------------------------------------------------------
```

174 SQL Explorer for DB2 User Guide
Note

- SQL Explorer generates a DDFLOC card as the first SYSIN card. The DDFLOC card uses the From Location and To Location values entered on Figure 68 on page 172.
- The UPDATE step includes the reference to target subsystem. You should still revise the Route and Jobparm Sysaff cards.

Setting Migrate Access Path Statistics options

This topic guides you through reviewing and changing options that are related to the Migrate Access Path Statistics feature.

To review or change Migrate Access Path Statistics options

1. Access the Migrate Access Path Statistics Options panel in one of the following ways:
   - From the SQL Explorer main menu:
     1. Type 0 (zero) for Options.
     2. In the Options field, type 2 (for Analysis) and press Enter.
     3. In the Options field, type 5 (for Migrate Access Path Statistics) and press Enter.
   - By typing any character in the Additional Options field on the Migrate Access Path Statistics Specification panel
The Migrate Access Path Statistics Options panel is displayed (Figure 70 on page 177).

**Figure 70: Migrate Access Path Statistics Options panel (PSSPM020)**

```
PSSPM020 ------------ Migrate Access Path Statistics Options ------------------
Command ===> Specify options and press ENTER to continue.

Commit . . . NOERROR          (NOError/NOWarning/Any)
Commit Freq    TS               (TS/DB/TB/EOJ) - On SQL error
Retries . . . 5                (0-10)          - On SQL timeout
Stats Time . . CURRENT          (Current/Source)
Delete . . . N                (Y/N) - Delete SYSCOLDIST, SYSTABSTATS,
                                SYSKEYTGTDIST and SYSKEYTGTDISTSTATS
                                entries for selected tablespace(s)

Specify Catalog Tables to Update/Insert
/ All                   Index Statistics for DB2 version 9 and later
SYSTABLESPACE   SYSTABLES
SYSTABSTATS     SYSKEYTARGETS
SYSCOLUMNS      SYSKEYTGTDIST
SYSCOLDIST      SYSKEYTGTDISTSTATS
SYINDEXES       SYSINDEXES
Include Additional Catalog Tables : S SYSCOLSTATS S SYSROUTINES
```

2. Complete the fields shown in Table 16 on page 177.

**Table 16: Migrate Access Path Statistics Options**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Commit</strong></td>
<td>Options for processing the Update job</td>
</tr>
<tr>
<td>Specify one of the following options:</td>
<td></td>
</tr>
<tr>
<td>■ NOError runs a COMMIT command if there are warnings but no errors. (default)</td>
<td></td>
</tr>
<tr>
<td>■ NOWarning runs a COMMIT command if there are no errors or warnings.</td>
<td></td>
</tr>
<tr>
<td>■ Any runs a COMMIT command regardless of errors or warnings.</td>
<td></td>
</tr>
<tr>
<td><strong>Commit Freq</strong></td>
<td>Frequency for running a COMMIT command during the Update job</td>
</tr>
<tr>
<td>Specify one of the following options:</td>
<td></td>
</tr>
<tr>
<td>■ TS runs a COMMIT command after the statistics relating to all objects in the table space are updated and before the next table space occurs. (default)</td>
<td></td>
</tr>
<tr>
<td>■ DB runs a COMMIT command after the statistics relating to all objects in the database are updated and before the next database occurs.</td>
<td></td>
</tr>
<tr>
<td>■ TB runs a COMMIT command after the statistics relating to all objects in the table are updated and before the next table occurs.</td>
<td></td>
</tr>
<tr>
<td>■ EOJ runs a COMMIT command at the end of the update statistics job.</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| On Error                     | Specifies how to proceed if errors are encountered  
  Specify one of the following options:  
  ■ CONTINUE keeps processing even if an SQL error occurs.  
  ■ QUIT stops processing when any SQL error occurs, including an SQL error of +100 Not Found on an Update statement. (default)  
  **Note:** This option has no effect if you select to run a COMMIT command regardless of errors or warnings (Any). The entries you make on the Migrate Access Path Statistics Options panel dictate the statistics migration output. For example, if you select to run a COMMIT command if there are no errors or warnings (NOWarning) and select to continue processing even if an SQL error occurs (CONTINUE), the results show all of the errors you get, but no updates are committed. |
| Retries                      | Number of times to retry executing the UPDATE or INSERT statement before terminating  
  Enter a value from 0 to 10. The default is 5. If a data set is not available or needs to be restored, retry the SQL statement.  
  **Note:** This parameter applies only to the Update job.                                                                                                                                                                                                                                             |
| Stats Time                   | Timestamp to use for updating the STATTIME on the target objects  
  ■ Current uses the actual machine time as the target timestamp value.  
  ■ Source uses the source object’s timestamp as the target timestamp value.  
  **Note:** In each of these instances, the timestamp microseconds is changed to 999999 to identify the statistics as migrated.                                                                                                                                                                       |
| Delete                       | Specifies whether all previous instances of the target table spaces SYSCOLDIST, SYSTABSTATS, SYSKEYTGTDIST and SYSKEYTGTDISTSTATS are deleted before inserting the migrated statistic rows  
  This parameter is for the UPDATE job.  
  Specify one of the following options:  
  ■ Y deletes the entries.  
  ■ N retains the entries. (default)                                                                                                                                                                                                                                                                  |
| Specify Catalog Tables to Update/ Insert | Specifies which target DB2 object types in the generated UPDATE file are updated or inserted  
  Only the statistics for the selected object types are updated or inserted, although all access path statistics are extracted. Index statistics are included for DB2 Version 9 and later. You can select multiple object types.  
  **Note:** Statistics for SYSCOLSTATS and SYSROUTINES are not included for the target object types unless you specify them in the INCLUDE field.                                                                                                                                                                 |
| Include Additional Catalog Tables | Specifies whether to include SYSCOLSTATS, SYSROUTINES or both in the update process  
  You can update these statistics without updating the target object types listed above.                                                                                                                                                                                                                                                                       |
Specification of patterns for translating object names

The translation function of the Migrate Access Path Statistics function enables you to specify patterns for translating object names. This capability is useful when the names of the objects on your target subsystem are different from the object names on your source subsystem.

This procedure is optional.

**Note**
The objects that you want to update must already exist on the target subsystem.

On the Migrate Access Path Statistics Specification panel, you can specify a single set of pattern translation input parameters (see Figure 71 on page 179).

**Figure 71: Example of pattern translation using a single set of input parameters**

You can manually insert additional translation sets by editing the Migrate Access Path Statistics JCL. For more information, see “Editing JCL to include additional translations or objects to migrate” on page 180. You can also specify patterns for translating object names by using the wildcard characters shown in “Specifying options for the Migrate Access Path Statistics job” on page 171.
Figure 72 on page 180 shows examples of pattern translation using these wildcards.

**Figure 72: Example of pattern translation using wildcard characters**

<table>
<thead>
<tr>
<th>Current Value</th>
<th>New Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DB Name . . PRODDB1</td>
<td>DB1TEST</td>
</tr>
<tr>
<td>TS Name . . PROD1%</td>
<td>%TEST1</td>
</tr>
<tr>
<td>TB Creator . <em>PROD</em></td>
<td><em>TEST</em></td>
</tr>
<tr>
<td>IX Creator . <em>PROD</em></td>
<td><em>TEST</em></td>
</tr>
<tr>
<td>IX Name . . PROD1%</td>
<td>%TEST1</td>
</tr>
</tbody>
</table>

By using wildcards, you can obtain a number of creator names on different objects. Wildcards are used only for changing the object names, and thus do not act as a filtering mechanism.

**Note**

When using wildcards for pattern translation, you must use the same number of wildcard characters for both the current and new values. Also, the process assumes that a portion of the source object name is the same as the target object name.

---

**Editing JCL to include additional translations or objects to migrate**

Use the following procedure to specify a single set of pattern translation input parameters, and manually insert additional translation sets into the generated JCL.

1. In the **Translate Current Object Names to New Names** field of the Migrate Access Path Statistics Specification panel, specify the values shown for a single set of translation patterns.

The product generates SYSIN parameters in the Extract job, based on the values that you specified (see Figure 73 on page 180):

**Figure 73: Example of SYSIN parameters from the Extract job**

```
DDDFLOC DEDR
MIGSTATS TABLESPACE PSSQX91.PSSS0391
```
2. Manually insert additional translation sets by editing the generated JCL (shown starting from line 9 in Figure 74 on page 181):

Figure 74: Example of JCL edited to insert additional translation sets

```
DDFLOC DEDR
MIGSTATS TABLESPACE PSSQX91.PSSS0391
DBNAME PSSQX91 TSSQX91
TSNAME PSSS0391 TSSS0391
TBCREATOR PSS TSS
IXCREATOR PSS TSS
IXNAME PSSX0%_D91S03T01 TSSX0%_D91S03T01
INCLUDE (CS,)
```

3. To complete this process, submit your edited JCL for execution and run the Extract job.

For more information, see “Running the Migrate Access Path Statistics Extract job” on page 182.

The example in Figure 75 on page 181 shows the results of the completed Extract job.

Figure 75: Results of completed Extract job

```
--SQL EXPLORER 11.2.00 PARMS- RELEASED MAY 2015
(C) COPYRIGHT 1996 - 2015 BMC SOFTWARE, INC.
SSID=DEDR PLAN-DAAVRM01
DDFLOC DEDR
MIGSTATS TABLESPACE PSSQX91.PSSS0391
DBNAME PSSQX91 TSSQX91
TSNAME PSSS0391 TSSS0391
TBCREATOR PSS TSS
IXCREATOR PSS TSS
IXNAME PSSX0%_D91S03T01 TSSX0%_D91S03T01
INCLUDE (CS,)
REMOTE CONNECTION TO DEDR SUCCESSFUL
REMOTE DB2 VERSION DSN09015
TS PSSQX91.PSSS0391
TB PSS.PST01.D91S03
IX PSS.PSSX01_D91S03T01
IX PSS.PSSX02_D91S03T01
IX PSS.PSSX03_D91S03T01
IX PSS.PSSX04_D91S03T01
MIGSTATS TABLESPACE PSSQX91.PSSS0291
DBNAME PSSQX91 TSSQX91
TSNAME PSSS0291 TSSS0291
TBCREATOR PSS TSS
IXCREATOR PSS TSS
IXNAME PSSX0%_D91S02T01 TSSX0%_D91S02T01
INCLUDE (CS,RT)
```
### Running the Migrate Access Path Statistics Extract job

Two jobs are contained in the JCL that results from running a Migrate Access Path Statistics operation.

The first job extracts the statistics from the source DB2 subsystem. This process creates a file containing the access path statistics. This file is the input to the second job, which is a DB2 Update job that is run against the target subsystem (SSID).
If you submit the job directly from the JCL without making any edits, only the Extract job executes. By default, the Update job is commented out in order to avoid any unintentional manipulation of the DB2 catalog. To run the second job, you might want to change the statistics or move your JCL to the IBM MVS image containing the target SSID.

**Note**

To run the Migrate Access Path Statistics jobs, you must be on the subsystem on which SQL Explorer is installed.

1. **Type SUB on the Command line of the JCL generated file, and press Enter to submit the Extract job that extracts the data from the source SSID.**

   The Extract job writes source object information to the UPDATE data set, along with the catalog statistic information for those objects.

   **Note**

   The comments contained in the generated UPDATE file refer to FLOAT data in generic terms. (For example, both CARD and CARDF are referenced as "CARD.") The statistics in the UPDATE file always include the correct value (for example, CARD or CARDF) for the version of DB2 from which it was extracted. The statistics that are updated are the correct fields for the version of DB2 specified as the target subsystem.

An example of the UPDATE data set is shown in Figure 76 on page 183.

**Figure 76: Example of UPDATE data set**

```sql
# SQL EXPLORER 11.2.00
# STATISTICS OBTAINED FROM DEDR 2014-12-20-15.30.14.474716
# RECORD FORMAT FOR EACH OBJECT TYPE
# :TS,DBNAME,NAME,NACTIVE,STATSTIME
# :TB,CREATOR,NAME,CARD,NPAGES,PCTROWCOMP,STATSTIME
# :TA,PARTITION,CARD,NPAGES,PCTROWCOMP,STATSTIME
# :CD,NAME,COLCARD,STATSTIME
# :CD,NAME,TYPE,NUMCOLUMNS,CARD,FREQUENCY,IBMREQD,STATSTIME
# :CD (FOR DB2) V9
# :CD,NAME,TYPE,NUMCOLUMNS,CARD,FREQUENCY,IBMREQD,COLGROUPCOLNO,COLVALUE,STATSTIME
# :CS,10,CNAME,PARTITION,COLCARD,STATSTIME,IBMREQD,STATS_FORMAT,COLCARDDATA
# :CS,20,CNAME,PARTITION,HIGHKEY
# :CS,21,CNAME,PARTITION,HIGH2KEY
# :CS,22,CNAME,PARTITION,LOWKEY
# :IX,CREATOR,NAME,NLEAF,NLEVELS,CLUSTERRATIO,FIRSTKEYCARD,FULLKEYCARD,STATSTIME
# :IX (FOR DB2) V9
# :IX,CREATOR,NAME,NLEAF,NLEVELS,CLUSTERRATIO,FIRSTKEYCARD,FULLKEYCARD,STATSTIME,DATAREPEATFACTORF
# :KT,10,SCHEMA,NAME,KEYSEQ,STATSTIME,STATS_FORMAT
# :KT,20,SCHEMA,NAME,KEYSEQ,HIGHKEY
# :KT,21,SCHEMA,NAME,KEYSEQ,LIMKEY
# :KT,22,SCHEMA,NAME,KEYSEQ,HIGHVALUES
# :KT,23,SCHEMA,NAME,KEYSEQ,HIGHVALUE
# :KS,10,SCHEMA,NAME,KEYSEQ,PARTITION,STATSTIME,CARDF,IBMREQD,STATS_FORMAT
# :KS,20,SCHEMA,NAME,KEYSEQ,PARTITION,HIGHKEY
# :KS,21,SCHEMA,NAME,KEYSEQ,PARTITION,HIGH2KEY
# :KS,22,SCHEMA,NAME,KEYSEQ,PARTITION,LOWKEY
# :KS,23,SCHEMA,NAME,KEYSEQ,PARTITION,LOW2KEY
```
Running the Migrate Access Path Statistics Extract job

SQL Explorer for DB2 User Guide
Running the Migrate Access Path Statistics Extract job

Chapter 11  Migrating access path statistics    185
Editing the Update file

Before running the Update portion of the Migrate Access Path Statistics job, you might want to modify the statistics that are loaded into the DB2 catalog for the target objects. You can do this by manually changing the statistics in the UPDATE data set that was created in the Migrate Access Path Statistics job.

The record format of each of the object types is listed in the header of the UPDATE data set. Once you have made your changes, press **F3** to save the modified UPDATE data set. These modified statistics are loaded into the target objects as you complete the Migrate Access Path Statistics process.

Running the Update job

Use the following procedure to run the Update job.

1. *(optional)* Update one or more values in the input file.

   For more information, see “Editing the Update file” on page 186.

2. Delete the Extract portion of the job from the JCL.

   In the example in Figure 77 on page 186, the Extract job ends after the first SYSIN. The Update job begins with the job card information that follows.

**Figure 77: Update job step in generated JCL**

```jcl
//MVSIXT1M JOB (PDOM,1605A), 'MIGRATE STATS',REGION=0K,
// MSGCLASS=X,CLASS=A,NOTIFY=&SYSUID
/*ROUTE XEQ BMCPMLX1
/*JOBPARM SYSAFF=DB2A
//-------
//* MEMBER: PSSSMIGR
//* EXECUTION OF MIGRATE ACCESS PATH STATISTICS
----------
//MIGRATE EXEC PGM=PSSAWK,REGION=OM,
// PARM='-f SRC(PSSMIEXT) -v SSID=DEDR -v PLAN=DAAVRMD1'
//STEPLIB DD DISP=SHR,DSN=BMCPERF.LOAD
// DD DISP=SHR,DSN=CSGI.SASC.V700C.LINKLIB
// DD DISP=SHR,DSN=SYS3.DEDR.DSNEXIT
// DD DISP=SHR,DSN=SYS2.DSNLOAD
//BMCPWSW DD DISP=SHR,DSN=AFDQA.AFD.PASSWORD
//ABNLIGNR DD DUMMY
//SRC DD DISP=SHR,DSN=BMCPERF.CLIST
//UPDATE DD DISP=SHR,DSN=MVSIXT3.SQLXPLR.UPDATE
//SYSTERM DD SYSOUT=* 
//SYSDUMP DD SYSOUT=* 
//SYSPRINT DD SYSOUT=* 
//SYIN DD *
DDFLOC DEDR
MIGSTATS TABLESPACE PSSQX91.PSSS0391
DBNAME PSSQX91 TSQX91
TSNAME PSSS0391 TSSS0391
```
Running the Update job

For the example shown in Figure 77 on page 186, you would remove the first job and the lines preceding the job card.

3 Uncomment the information about the second job card by deleting the three leading characters (//*) in each line that is part of the second job card.
4 If the target SSID is on a different MVS system from the source SSID, move the JCL to the correct MVS system and run the JCL on the target SSID.

You might also need to verify that the DSNEXIT and DSNLOAD libraries are correct for the target SSID.

5 Submit the Update job.

Figure 78 on page 188 shows an example of the output from the Update job.

Figure 78: Update job output listing

```
SQL EXPLORER 11.2.00 PARMS- RELEASED MAY 2015
(C) COPYRIGHT 1996 - 2015 BMC SOFTWARE, INC.
SSID=DEDR  PLAN=DAAVRLD1
DDFSRC DDBR
MIGSTATS UPDATE ALL
COMMIT NOERROR
COMMIFREQ TS
ONERROR QUIT
RETRIES 5
DELETE YES
STATSTIME CURRENT
REMOTE CONNECTION TO DEDR SUCCESSFUL
REMOTE DB2 VERSION DSN09015
TS PSSQX91, PSSS0391 UPDATED
TB PSS, PST01_D91S03 UPDATED
IX PSS, PSSX01_D91S03T01 UPDATED
IX PSS, PSSX02_D91S03T01 UPDATED
IX PSS, PSSX03_D91S03T01 UPDATED
IX PSS, PSSX04_D91S03T01 UPDATED
**COMMIT UPDATES/INSERTS**
RECORD FORMAT FOR EACH OBJECT TYPE
:TS, DBNAME, NAME, NACTIVE, STATSTIME
:TB, CREATOR, NAME, CARD, NPAGES, PERTROWCOMP, STATSTIME
:TA, PARTITION, CARD, NPAGES, PCTPAGES, NACTIVE, PERTROWCOMP, IBMREQD, STATSTIME
:CD, NAME, COLCARD, HIGH2KEY, LOW2KEY, STATSTIME
:CD, NAME, TYPE, NUMCOLUMNS, CARD, FREQUENCY, IBMREQD, COLGROUPCOLNO, COLVALUE, STATSTIME
:**CD (FOR DB2) V9
:**CD, NAME, TYPE, NUMCOLUMNS, CARD, FREQUENCY, IBMREQD, COLGROUPCOLNO, COLVALUE, STATSTIME, QUANTILENO
:**DB2 V9 CD TYPE H, THE TWO FOLLOWING RECORDS
:C1, LOWVALUE
:C2, HIGHVALUE
:C3, COLCARD, PARTITION, COLNAME, STATSTIME, IBMREQD, STATS_FORMAT, COLCARDATA
:C4, COLNAME, PARTITION, HIGHKEY
:C5, COLNAME, PARTITION, HIGH2KEY
:C6, COLNAME, PARTITION, LOWKEY
:C7, COLNAME, PARTITION, LOW2KEY
:IX, CREATOR, NAME, NLEAF, NLLEVELS, CLUSTERRATIO, FIRSTKEYCARD, FULLKEYCARD, STATSTIME
:**IX (FOR DB2) V9
:**IX, CREATOR, NAME, NLEAF, NLLEVELS, CLUSTERRATIO, FIRSTKEYCARD, FULLKEYCARD, STATSTIME, DATAREPEATFACTOR
:KT, SCHEMA, NAME, KEYSEQ, STATSTIME, STATS_FORMAT
:KT, SCHEMA, NAME, KEYSEQ, HIGHKEY
:KT, SCHEMA, NAME, KEYSEQ, LOWKEY
:KD, SCHEMA, NAME, KEYSEQ, KEYVALUE
:KD, SCHEMA, NAME, KEYSEQ, KEYGROUPKEYNO
:KD, SCHEMA, NAME, KEYSEQ, KEYGROUPKEYNO
:KA, SCHEMA, NAME, KEYSEQ, KEYVALUE
:KA, SCHEMA, NAME, KEYSEQ, KEYGROUPKEYNO
:KA, SCHEMA, NAME, KEYSEQ, KEYVALUE
:KA, SCHEMA, NAME, KEYSEQ, KEYGROUPKEYNO
:KA, SCHEMA, NAME, KEYSEQ, KEYVALUE
:RT, SCHEMA, NAME, ROUTINTYPE, SPECIFICNAME, PARM_COUNT, CARDINALITY, INITIAL_INSTS, INITIAL_IOS, INSTS_PER_INVOC, IOS_PER_INVOC
MIGSTATS UPDATE SUMMARY 01/01/2015 06:28 PM
OBJECT   INPUT  UPDATED  ERROR  WARNING
------------------ ------ ------- ------- -------
TABLESPACE     1      1       0       0
```

SQL Explorer for DB2 User Guide
### Running the Update job

#### Chapter 11  Migrating access path statistics  189

<table>
<thead>
<tr>
<th>Table/Statistics</th>
<th>Count</th>
<th>Old Count</th>
<th>Mismatch</th>
<th>New Mismatch</th>
</tr>
</thead>
<tbody>
<tr>
<td>TABLES</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TABSTATS</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>COLUMNS</td>
<td>20</td>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>COLDIST</td>
<td>48</td>
<td>48</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>INDEXES</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SYSKEYTARGETS</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SYSKEYTARGETSTATS</td>
<td>12</td>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SYSKEYTGTDIST</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SYSKEYTGTDISTSTATS</td>
<td>8</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>COLSTATS</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ROUTINES</td>
<td>1058</td>
<td>1058</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Running the Update job
Generating table declarations automatically

This chapter includes the following sections:

- “Overview of generating declarations” on page 191
- “Using the Declarations Generator function” on page 193

Overview of generating declarations

For application programmers, the Declarations Generator function of the SQL Explorer product simplifies the process of creating a table declaration.

The Declarations Generator automatically generates a DECLARE TABLE statement, a host variable structure, and an indicator structure for nullable columns for a COBOL, C, or PL/I program. The generated statements are inserted into a partitioned data set (PDS) member or a sequential data set that you can include in any number of applications. This reduces application development time and enables you to avoid the errors that often occur with manual coding. It also ensures that your host variable definitions match the DB2 column types, so you know that your table declarations are correct even before you begin precompiling an application.

Other generator products generate an array for indicator variables. Such an array can make it difficult to remember the column order and the array position for the indicator variable to be checked. As a further complication, the array includes all columns, rather than just the columns containing null values. All of these factors can lead to costly mistakes.

As an alternative solution, the Declarations Generator appends an alternative indicator structure to the bottom of your generated output. This structure uses the column name with a suffix of IND for the null indicator name. Only nullable columns are included in the indicator structure. This simplifies the process of coding null indicators, because you do not need to remember the column order.
The Declarations Generator accommodates existing programs by including an array in the output, so that such programs do not have to be changed for the new indicator variables, even though the programs may have other coding changes that require a recompile. This means that the same Declarations Generator file can be used in both existing and new programs, even as newer programs begin to take advantage of the new indicator names.

Figure 79 on page 192 shows a sample of the array and the additional indicator structure output produced by the Declarations Generator.

Figure 79: Output from Declarations Generator operation

```
EXEC SQL DECLARE DEPT TABLE
( DEPTNO                         CHAR(3) NOT NULL,
  DEPTNAME                       VARCHAR(36) NOT NULL,
  MGRNO                          CHAR(6),
  ADMRDEPT                       CHAR(3) NOT NULL,
  LOCATION                       CHAR(16) ) END-EXEC.
```

For more information, view the Quick Course "SQL Explorer for DB2—Using DCLGEN."
Using the Declarations Generator function

This task describes how you can use the Declarations Generator to generate a table declaration and insert the generated table declaration into a partitioned or sequential data set.

**Note**
The Declarations Generator function does not allow wildcard characters.

1. On the SQL Explorer main menu, type 8 (for **Declarations Generator**).
   
2. Change the SSID location, if necessary, and press **Enter**.

   The Declarations Generator Specification panel is displayed (Figure 80 on page 193).

   **Note**
The **DDF Location** field is not applicable for the Declarations Generator function.

3. On the Declarations Generator Specification panel, specify values for fields as described in “**Declarations Generator specifications**” on page 194.

4. Press **Enter**.

   Depending on the **Process Mode** you selected, one of the following events occurs:

<table>
<thead>
<tr>
<th>Process mode</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>O (online)</td>
<td>The Declarations Generator job runs in the foreground and a message appears on the panel that the declaration has been generated.</td>
</tr>
</tbody>
</table>
Process mode | Event
---|---
B (batch) | The Batch Job panel is displayed. For instructions, see “Specifying JCL options in batch mode” on page 48.

5 Press **Enter** to generate the JCL.

**Declarations Generator specifications**

This topic describes each field on the Declarations Generator Specification panel.

Table 17: Declarations Generator Specifications

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Process Mode** | O runs the Declarations Generator operation online  
 | B (default) creates JCL that you can edit before running the Declarations Generator operation in batch |
| **Creator**   | The name of the table or view’s owner. If the table is a local table and you do not specify this value, DB2 uses your TSO logon ID as the table qualifier. If the table is at a remote location, you must specify a value for this option.  
 | If the **Creator** information is truncated, place your cursor on the field, and press **PF4 (ZOOM)**. The contents of the entire field are displayed in a pop-up window. |
| **Table Name** | The name of the table or view for which you want the Declarations Generator to produce table declarations.  
 | This field is case sensitive. If you use special or lowercase characters in this field, you must enclose the table name in quotation marks.  
 | If you specify an apostrophe or quotation mark as part of a table name, you must double each one and use the delimited identifiers ("'" or "") . You can also include spaces in a table name by enclosing the name in apostrophes or quotation marks. However, you cannot use wildcards in this field.  
 | If the **Table Name** information is truncated, place your cursor on the field, and press **PF4 (ZOOM)**. The contents of the entire field are displayed in a pop-up window. |
| **Library**   | The name of the data set in which the table declarations are to be stored. This data set must already exist and be defined with LRECL=80 for correct formatting, and the Declaration Generator must be able to access it. The data set can be sequential or partitioned. If you specify a partitioned data set, you must also specify a PDS member name.  
<p>| You can use apostrophes as delimited identifiers in this field. If you do not use delimiters, the Declarations Generator applies a standard TSO prefix (user ID) to the data set name that you provide. |</p>
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>The action that the Declarations Generator should take with the generated output when it is sent to a partitioned or sequential data set.</td>
</tr>
<tr>
<td></td>
<td>■ A (ADD)—(default) adds the declarations to the sequential data set or adds a new member to the partitioned data set, if it does not already exist.</td>
</tr>
<tr>
<td></td>
<td>■ R (REPLACE)—replaces an existing data set or member with the new data set or member. The Declarations Generator creates a new member for a partitioned data set lacking a member with the specified name.</td>
</tr>
<tr>
<td></td>
<td><strong>WARNING:</strong> If you specify A (ADD) for a sequential data set containing data, the Declarations Generator overwrites the existing data, rather than appending it.</td>
</tr>
<tr>
<td>Language</td>
<td>The language of the generated table declaration, as follows:</td>
</tr>
<tr>
<td></td>
<td>■ COBOL (default)—any COBOL language</td>
</tr>
<tr>
<td></td>
<td>■ PLI—PL/I</td>
</tr>
<tr>
<td></td>
<td>■ C—C/370 or C++</td>
</tr>
<tr>
<td>Delimiter</td>
<td>The string delimiter character used in the host language. This option is only applicable for COBOL programs. Use one of the following options:</td>
</tr>
<tr>
<td></td>
<td>■ A—use an apostrophe (’) as the string delimiter character</td>
</tr>
<tr>
<td></td>
<td>■ Q—use a quotation mark (‘) as the string delimiter character</td>
</tr>
<tr>
<td>Indicator Var</td>
<td>Specifies whether to create an indicator variable for each column that might contain a null value. Y (Yes) is the default.</td>
</tr>
<tr>
<td>Qualify Table</td>
<td>Specifies whether to include the name of the table or view’s owner in the generated DECLARE statement. Y (Yes) is the default.</td>
</tr>
<tr>
<td>Structure</td>
<td>The name of the generated data structure. To use special or lowercase characters in the data structure name, enclose the data structure name in quotation marks.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If you leave this field blank, the default structure name is the name of the table or view, with a prefix of DCL. If the table or view name is a DBCS string and the language is COBOL or PL/I, the prefix consists of DBCS characters.</td>
</tr>
<tr>
<td>Additional Options</td>
<td><em>(optional)</em> Type any character in the space beside this field to select additional options for the Declarations Generator operation. For more information, see “Setting Declarations Generator options” on page 195.</td>
</tr>
</tbody>
</table>

**Setting Declarations Generator options**

This section guides you through reviewing and changing options related to the Declarations Generator feature.

**To review or change Declarations Generator options**

1. Access the Declarations Generator Additional Options panel in one of the following ways:
From the SQL Explorer main menu:

1. Type 0 (zero) for **Options**.

2. In the **Options** field, type 2 (for **Analysis**) and press **Enter**.

3. In the **Options** field, type 6 (for **Declarations Generator**) and press **Enter**.

By typing any character in the **Additional Options** field on the Declarations Generator Specification panel.

The Declarations Generator Additional Options panel is displayed.

**Figure 81: Declarations Generator Additional Options panel (PSSPDG20)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Label</strong></td>
<td>Specify whether to include column labels in the output as comments. <strong>N</strong> is the default.</td>
</tr>
<tr>
<td><strong>DBCS Symbol</strong></td>
<td>Specify either <strong>G</strong> (the default) or <strong>N</strong> to be used as the symbol to denote a graphic data type in a COBOL picture clause.</td>
</tr>
<tr>
<td><strong>DBCS Delimited</strong></td>
<td>Specify whether the double-byte character set (DBCS) table and column names in the generated DECLARE table statement are enclosed in SQL delimiters. <strong>Y</strong> is the default.</td>
</tr>
<tr>
<td><strong>Col Suffix</strong></td>
<td>Specify whether to form field names by concatenating the column name to the prefix that you specify in the <strong>Names</strong> field. <strong>N</strong> is the default and specifies that the field names are controlled by the <strong>Names</strong> value.</td>
</tr>
<tr>
<td><strong>Names</strong></td>
<td>Location name of table or view</td>
</tr>
</tbody>
</table>

2. On the Declarations Generator Additional Options panel, complete the following fields as required:

   **Note:** If you do not specify a value for the **Names** option, the Declarations Generator issues a warning message and uses the column names as the field names.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Names</td>
<td>Specify the field name prefix that is formed in the declaration. If you specify this option and the Col Suffix value is N, the field names contains the prefix, concatenated with a number that is one to three digits in length. For example, if the prefix is ABCD, the field names are ABCD1, ABCD2, and so on, up to a maximum value of ABCD999. If you leave this field blank, the column names in the table or view are used as the default names for the fields in the generated output. <strong>Note:</strong> If you use special or lowercase characters in this field, you must enclose the field name prefix in quotation marks.</td>
</tr>
<tr>
<td>At Location</td>
<td>Specify the location of the table or view name on another DB2 subsystem. You can use an _ (underscore) in this field. If you specify this option, the location is used as a prefix for the fully-qualified table name. If you leave this field blank, the Declarations Generator assumes that the object is located on the DB2 SSID that you specified on the SQL Explorer main menu, and the location name does not appear (or is not generated) as part of the name in the DECLARE TABLE statement.</td>
</tr>
</tbody>
</table>
Parameter reference

This appendix presents topics relating to parameters.

Overview of parameters

This appendix lists each of the parameters or options you can use in foreground or batch jobs of various SQL Explorer functions.

The parameters are presented by function, and in alphabetical order within each function. All product functions except the Declarations Generator use parameters; the Declarations Generator uses options.

Note

For all of the listed product functions except Migrate Access Path Statistics and the Declarations Generator, the equal sign (=) is required for parameters where indicated. If you omit the equal sign when it is required, the product ignores the parameter and might generate an error message. You can use blank values. If you omit the parameter entirely, the product uses the default value of the parameter. Do not use an equal sign (=) with the Migrate Access Path Statistics parameters and Declarations Generator options.

You can include more than one object in a batch job by using a wildcard character or by repeating the --NEWOBJ/--ENDOBJ set of parameters in the SYSIN.

Explain parameters

You can use the following parameters in your Explain job.

The equal sign (=) is required for these parameters where shown. You can include more than one object in a batch job by using a wildcard character or by repeating the --NEWOBJ/--ENDOBJ set of parameters in the SYSIN.
### Table 18: Explain parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| ACCEL= (optional) | NA | Whether Analytics are displayed:  
- Y (default) specifies that Accelerator rules are displayed.  
- N specifies that Accelerator rules are not displayed. |
| --BEGSQL (optional; required only for OBJECT=SQLTEXT) | NA | Start-delimiter for inline SQL associated with OBJECT=SQLTEXT.  
When Explain encounters this parameter, it starts processing the input lines as inline SQL.  
**Note:** You must also delimit the end of the inline SQL with the --ENDSQL parameter. |
| BMCMSG= (optional) | NA | Use and display format of the BMC rule messages produced in the report.  
Specify one of the following values:  
- 1 displays the rule message with the message text, the severity of the rule message, and the BMC Message ID (default).  
- 2 produces the rule messages with only the message text.  
- 3 bypasses all rule message processing.  
**Note:** This parameter is for use only in batch jobs. |
| COLBRK= (optional) | NA | Whether a break occurs between columns listed as part of the SELECT clause in the SQL text.  
Specify one of the following values:  
- Y specifies that the break occurs.  
- N specifies that the break does not occur. (default) |
| COLLID= (optional) | 08 | Collection ID for packages.  
If you omit this value, this parameter is not part of the selection criteria.  
The maximum length is 128. |
| COMPARE= (optional) | NA | Whether to compare the dynamic Explain with the static Explain for the object.  
Specify one of the following values:  
- Y specifies that the comparison occurs.  
- N specifies that the comparison does not occur (default). |
| CONTOKEN= (optional) | NA | Contoken for packages, used to provide the CHAR value of the consistency token.  
The length must be 16. |
| CURRENCY= (optional) | NA | Three-character currency code for the monetary unit used to calculate the cost translation rate.  
You can use any characters except spaces in this field, though BMC Software recommends that you use ISO 4217 standard codes. The default is USD (U.S. dollars). |
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DBRM</strong>=</td>
<td>NA</td>
<td>Name of the DBRM to Explain</td>
</tr>
<tr>
<td><strong>DBRMLIB</strong>=</td>
<td>NA</td>
<td>Whether to Explain a DBRMLIB member</td>
</tr>
<tr>
<td><strong>DDFLOC</strong>=</td>
<td>NA</td>
<td>Name of DB2 location, if you plan to access data on a different subsystem using DDF</td>
</tr>
<tr>
<td><strong>DEGREE</strong>=</td>
<td>NA</td>
<td>Whether to run a query using parallel processing for better performance</td>
</tr>
<tr>
<td><strong>--ENDOBJ</strong></td>
<td>NA</td>
<td>End-delimiter for a set of Explain parameters associated with a specific object</td>
</tr>
<tr>
<td><strong>--ENDSQL</strong></td>
<td>NA</td>
<td>End-delimiter for inline SQL associated with OBJECT=SQLTEXT</td>
</tr>
<tr>
<td><strong>HISTORY</strong>=</td>
<td>NA</td>
<td>Whether the history tables are to be updated with the results of an Explain</td>
</tr>
</tbody>
</table>

**Parameter name**
- **DBRM**= *(optional; required only for OBJECT= DBRM)*
- **DBRMLIB**= *(optional)*
- **DDFLOC**= *(optional)*
- **DEGREE**= *(optional)*
- **--ENDOBJ** *(required)*
- **--ENDSQL** *(optional; required only for OBJECT= SQLTEXT)*
- **HISTORY**= *(optional)*

**Description**
- Name of the DBRM to Explain
- You can use DB2 wildcards with this parameter. The maximum length is 8. The default is %.
- Whether to Explain a DBRMLIB member
- Specify one of the following values:
  - **Y** specifies that the Explain occurs.
  - **N** specifies that the Explain does not occur (default).
- Name of DB2 location, if you plan to access data on a different subsystem using DDF
  - The maximum length is 16.
- Whether to run a query using parallel processing for better performance
  - For plans, this parameter applies only to the DBRMs bound directly to the plan (named in the MEMBER option on BIND PLAN), and has no effect on PKLIST names. This parameter has no effect on dynamic SQL statements, because they use the value of the special register CURRENT DEGREE.
  - Specify one of the following values:
    - **1** prohibits parallel processing.
    - **ANY** allows parallel processing (default for SQLTEXT and DBRMLIB Explains).
    - **blank** uses the value specified at bind time (default for all other Explain types).
- End-delimiter for a set of Explain parameters associated with a specific object
  - When Explain encounters this parameter, it processes the preceding Explain parameters (delimited by the --NEWOBJ parameter) as a completed Explain request.
- End-delimiter for inline SQL associated with OBJECT=SQLTEXT
  - When Explain encounters this parameter, it stops processing the preceding input lines (delimited by the --BEGSQL parameter) as inline SQL.
- Whether the history tables are to be updated with the results of an Explain
  - Specify one of the following values:
    - **Y** specifies to update the history tables with Explain results.
    - **N** specifies not to update the history tables with Explain results.
  - *(default)*
  - **Note:** This parameter does not apply to single SQL text (OBJECT=SQLTEXT) or DBRMLIB Explains.
### Explain parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| INSERT= (optional) | NA | Whether to Explain INSERT statements  
  - Y specifies to Explain the INSERT statements (default).  
  - N specifies not to Explain the INSERT statements. |
| MSGLEVEL= (optional) | NA | Minimum severity level of the messages produced in the report  
  Specify one of the following values:  
  - A (ALL) specifies to display all Explain messages. (default)  
  - S (SEVERE) specifies to display only messages indicating severe problems.  
  - W (WARNING) specifies to display only warnings and severe messages.  
  - I (INFORMATIONAL) specifies to display informational, warning and severe messages. |
| NAME= (optional; required only for OBJECT=PLAN or PACKAGE, or when DBRMLIB=YES) | 07 | Name of the package or plan to Explain  
  You can use DB2 wildcards with this parameter. The default is %.  
  - The maximum length for plan names is 8.  
  - The maximum length for package names is 128. |
| --NEWOBJ (optional) | NA | Start-delimiter for a set of Explain parameters associated with a specific object  
  You must also delimit the end of the set with the --ENDOBJ parameter. The delimited set of parameters are treated as a single request. |
| OBJECT= (required) | 04 | Object type to be analyzed  
  Specify one of the following values:  
  - PLAN  
  - PACKAGE  
  - DBRM  
  - DBRMLIB  
  - SQLTEXT |
| ONLINE= (optional) | NA | Whether the object is used in online or batch mode  
  Specify one of the following values:  
  - Y specifies that the Explain occurs online.  
  - N specifies that the Explain occurs in batch (default).  
  **Note:** Setting this value to Y triggers specific rules for table space scan, List/Sequential Prefetch, and multiple index access path (MIAP) considerations. |
<p>| PLAN= (required) | 02 | Product plan name used for executing the product function |</p>
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| PLANTBL=          | NA                  | Whether to process a dynamic Explain or a static Explain. Specify one of the following values:  
- **Y** specifies a static Explain and causes the Explain engine to search an existing PLAN_TABLE for the latest version of the requested object and use the access paths reported there.  
- **N** specifies a dynamic Explain. (default)  
Note: The Explain function does not perform a bind or rebind with EXPLAIN(YES) on the object, because this could affect currently executing objects. |
| PLNAME=           | 06                  | Name of the plan containing DBRMs and packages to process. You can use DB2 wildcards to display a list of matching plans in batch. The maximum length is 8. The default is %. |
| PTOWNR=           | NA                  | Qualifier for the plan table used by the dynamic Explain operation. Specify one of the following values:  
- **authID** specifies to use authID.Plan_Table, where authID can be any valid DB2 qualifier.  
- **USERID** specifies to use the job submitter’s user ID for the plan table. Use this value if you do not have the authority to perform the SET CURRENT SQLID command.  
- **OWNER** specifies the owner of the package or DBRM. |
| RATE=             | NA                  | Rate used to translate the timeron cost into a monetary unit, in the form of monetary unit/timerons. |
| RATESU=           | NA                  | Rate used to translate the service units cost into a monetary unit, in the form of monetary unit/service units. |
| READONLY=         | NA                  | Whether the object’s access intent is read only. Specify one of the following values:  
- **Y** triggers specific rules for plans, DBRMs, and DBRMLIBs.  
- **N** specifies that the object’s access intent is not read only. (default) |
| RECALL=           | NA                  | Whether to recall an archived DBRMLIB data set when processing a DBRM or DBRMLIB. Specify one of the following values:  
- **Y** recalls the archived DBRMLIB data set (default).  
- **N** does not recall the archived data set. |
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RETAIN= (optional)</td>
<td>NA</td>
<td>Number of Explain results to retain. The purge process for history uses this value to clean up the history table during the Explain operation. If the HISTORY parameter is set to N, this value is ignored and no purge processing is performed. This parameter is a value from 0 to 99. A value of 0 deletes all existing rows for the object, or saves none. The default is 3.</td>
</tr>
</tbody>
</table>
| RPTHDR= (optional) | NA | Whether to include report headings in the batch report. Specify one of the following values:  
  - Y specifies to include the report headings (default).  
  - N specifies not to include the report headings; use this option if you plan to review the report online from an output data set. |
| RULES= (optional) | NA | Name of a rule set from which to provide expert rules in the report. Specify one of the following values:  
  - DEFAULT specifies one of the product’s predefined rule sets (default).  
  - APPLDEV specifies one of the product’s predefined rule sets.  
  - A specific rule set name associated with a customized rules set. The maximum length is 8. |
| SQLTEXT= (optional) | NA | Used when OBJECT=SQLTEXT, where SQLTEXT is the SQL text you want to print; should be the last parameter in the JCL. Note: You can use the SQLTEXT parameter instead of the –BEGSQL and –ENDSQL parameters. |
| SSID= (required) | 01 | DB2 subsystem name on which to perform the Explain. The subsystem value cannot be changed within a single job step. The maximum length is 4. |
| STATIC= (optional) | NA | Rows in the PLAN_TABLE used for a static Explain. Specify one of the following values:  
  - L uses the most recent rows in the owner.PLAN_TABLE from the last time you ran a bind with EXPLAIN=YES for the object you want to Explain (default).  
  - A uses all of the rows in the owner.PLAN_TABLE for the object you want to Explain. |
<p>|_STMTNO= (optional) | NA | Restricts the process to a single preprocessor SQL statement number associated with the object |</p>
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| SUMRPT= (optional) | NA | Sort order of the batch Explain report, which contains one line for each object and statement. Specify one of the following values:  
- 1 specifies to sort the report sequenced by object, statement. (default)  
- 2 specifies to sort the report sequenced by cost, ranking all statements processed in the job step by cost in descending order.  
- 3 specifies to sort the report sequenced by object, cost. |
| SUPPRESSDET= (optional) | NA | Level of report detail printed during a batch Explain (used in conjunction with MSGLEVEL). Specify one of the following values:  
- Y suppresses the detail in a batch Explain report when no rules are issued, based on the MSGLEVEL you specify.  
- N displays the detail in a batch Explain report, based on the MSGLEVEL you specify. (default) |
| TBQUAL= (optional) | 82 | Qualifier for objects and tables in the Explain. Specify one of the following values:  
- QUAL specifies that Explain uses the qualifier name provided at bind time. (default)  
- authID specifies a valid DB2 qualifier used to qualify any unqualified objects. |
| TITLE= (optional) | NA | Title to display on each page of the batch report. The maximum length is 64. |
| VERSION= (optional) | 09 | Version value associated with a package. The maximum length of this value is 122. |

---

**Related Information**

- "Specifying batch Explain options" on page 82
Explain and Compare parameters

The input and execution parameters for the Explain and Compare function are the same as the parameters for the individual functions themselves.

**Related Information**
- “Explain parameters” on page 199
- “Compare parameters” on page 206

Compare parameters

You can use the following parameters in your foreground or batch Compare job.

**Table 19: Compare parameters**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLLID= (optional)</td>
<td>08</td>
<td>Collection ID for packages If you omit this value, this parameter is not part of the selection criteria.</td>
</tr>
<tr>
<td>COLLID2= (optional)</td>
<td>NA</td>
<td>Collection ID of the second package used in the Compare Explain History process</td>
</tr>
<tr>
<td>COMPNEW= (required)</td>
<td>NA</td>
<td>Version of an Explain history to be Compared to an older history</td>
</tr>
<tr>
<td>COMPOLD= (required)</td>
<td>NA</td>
<td>Version of an Explain history to be Compared to a newer history</td>
</tr>
<tr>
<td>COSTSUM= (optional)</td>
<td>NA</td>
<td>Whether to produce a cost summary report Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Y specifies to produce the report.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>N specifies not to produce the report. (default)</td>
</tr>
<tr>
<td>CURRENCY= (optional)</td>
<td>NA</td>
<td>Three-character currency code for the monetary unit used to calculate the cost translation rate You can use any characters except spaces in this field, though BMC Software recommends you use ISO 4217 standard codes. The default is USD (U.S. dollars).</td>
</tr>
<tr>
<td>DDFLOC= (optional)</td>
<td>NA</td>
<td>Name of DB2 location, if you plan to access data on a different subsystem using DDF The maximum length is 16.</td>
</tr>
<tr>
<td>Parameter name</td>
<td>Execution parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DIFF= (optional)</td>
<td>NA</td>
<td>How the output displays statements that are different Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y specifies that the output lists only the statements that have a difference in access paths. (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N specifies that the output lists all of the statements, but flags those statements that have a difference in access paths.</td>
</tr>
<tr>
<td>--ENDOBJ (required)</td>
<td>NA</td>
<td>End-delimiter for a set of Compare parameters associated with a specific object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When Explain encounters this parameter, it processes the preceding Explain parameters (delimited by the --NEWOBJ parameter) as a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>completed Explain request.</td>
</tr>
<tr>
<td>MATCHPCT= (optional)</td>
<td>NA</td>
<td>Percentage threshold used for comparing SQL statements to each other to determine whether a match exists</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, MATCHPCT=50. This parameter is a value from 0 to 100. The default is 100.</td>
</tr>
<tr>
<td>NAME= (required)</td>
<td>07</td>
<td>Object name used when the object was bound</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This value must match the value stored in the catalog. You can use a DB2 wildcard (%) and the product performs an Explain for all objects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>selected from the catalog.</td>
</tr>
<tr>
<td>NAME2= (optional)</td>
<td>NA</td>
<td>Name of the second package, plan, or DBRM for the Compare Explain History process</td>
</tr>
<tr>
<td>--NEWOBJ (optional) NEWOBJ parameter</td>
<td>NA</td>
<td>Start-delimiter for a set of analysis Compare parameters associated with a specific object</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(You must also delimit the end of the set with the --ENDOBJ parameter.)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The delimited set of parameters are treated as a single request. If the job step has only one request, you can omit this parameter.</td>
</tr>
<tr>
<td>OBJECT= (required)</td>
<td>04</td>
<td>Object type to be analyzed Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ PLAN</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ PACKAGE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ DBRM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter is required; there is no default value.</td>
</tr>
<tr>
<td>PLAN= (required)</td>
<td>02</td>
<td>Product plan name used for executing the product function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This parameter is required; there is no default value.</td>
</tr>
<tr>
<td>PLNAME= (required)</td>
<td>06</td>
<td>Name of the plan containing the DBRM to process</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DB2 wildcards can be used to display a list of matching plans. The maximum length is 8.</td>
</tr>
<tr>
<td>PLNAME2= (optional)</td>
<td>NA</td>
<td>Plan name of the second object for the Compare Explain History process when OBJECT=DBRM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The maximum length is 8.</td>
</tr>
<tr>
<td>Parameter name</td>
<td>Execution parameter</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>---------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>RATE=</td>
<td>NA</td>
<td>Rate used to translate the cost into a monetary unit, in the form of CURRENCY/timerons. This parameter value can be any number. The default is 1.0.</td>
</tr>
</tbody>
</table>
| RPTHDR=        | NA                  | Whether to include report headings in the batch report. Specify one of the following values:  
  - Y specifies to include the report headings. (default)  
  - N specifies not to include the report headings; use this option if you plan to review the report online from an output data set. |
| SHOWSQL=       | NA                  | Whether to include SQL text in the report. Specify one of the following values:  
  - Y specifies to show the SQL text. (default)  
  - N specifies to omit the SQL text. |
| SSID=          | 01                  | Name of DB2 to which you are attached. The subsystem value cannot be changed within a single job step. The maximum length is 4. |
| TITLE=         | NA                  | Title to display on each page of the report. The maximum length is 64. |
| TXTLEN=        | NA                  | Width of the text displayed in the report. This parameter is a value from 60 to 132. The default is 72. |
| VERSION=       | 09                  | Version value associated with a package. The maximum length is 122. |
| VERSION2=      | NA                  | Version of the second object for the Compare Explain History process when OBJECT=PACKAGE. The maximum length is 122. |

- The maximum length is 128.
- You can use any combination of execution parameters, as long as the total byte count of all execution parameters and their values does not exceed 100.
- Use a value from -99 to 0, with 0 being the more current history.
- The maximum length for all object names except packages is 8. The maximum length for package names is 128.
# Mismatch Analysis parameters

You can use the following parameters in your foreground or batch Mismatch Analysis job.

## Table 20: Mismatch Analysis parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| COLBRK= (optional) | NA | Whether a break occurs between columns listed as part of the SELECT clause in the SQL text. Specify one of the following values:  
  - Y specifies that the break occurs.  
  - N specifies that the break does not occur. (default) |
| COLLID= (optional) | 08 | Collection ID for packages. If you omit this value, this parameter is not part of the selection criteria. The maximum length is 128. |
| DDFLOC= (optional) | NA | Name of DB2 location, if you plan to access data on a different subsystem using DDF. The maximum length is 16. |
| DIFF= (optional) | NA | Whether the output lists only the statements that have a mismatch in data type or length that could cause a change to the access path.  
  - Y specifies that the output lists only the statements that are different. (default)  
  - N specifies that the output lists all of the statements, but flags those statements that are different. |
| --ENDOBJ (required) | NA | End-delimiter for a set of Mismatch parameters associated with a specific object. When Explain encounters this parameter, it processes the preceding Explain parameters (delimited by the --NEWOBJ parameter) as a completed Explain request. |
| NAME= (required) | 07 | Object name used when the object was bound. This value must match the value stored in the catalog. You can use a DB2 wildcard (%) and the product performs an Explain for all objects selected from the catalog.  
  - The maximum length for all object names except package names is 8.  
  - The maximum length for package names is 128. |
<p>| --NEWOBJ (optional) | NA | Start-delimiter for a set of Mismatch parameters associated with a specific object. You must also delimit the end of the set with the --ENDOBJ parameter. The delimited set of parameters are treated as a single request. If the job step has only one request, you can omit this parameter. |</p>
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| OBJECT= (required) | 04 | Object type to be analyzed
Specify one of the following values:
- PLAN
- PACKAGE
- DBRM |
| PLAN= (required) | 02 | Product plan name used for executing the product function |
| RPTHDR= (optional) | NA | Whether to include report headings in the batch report
Specify one of the following values:
- Y specifies to include the report headings. (default)
- N specifies not to include the report headings; use this option if you plan to review the report online from an output data set. |
| SSID= (required) | 01 | DB2 subsystem name
The subsystem value cannot be changed within a single job step. The maximum length is 4. |
| STMTNO= (optional) | NA | Restricts the process to a single preprocessor SQL statement number associated with the object |
| TITLE= (optional) | NA | Title to display on each page of the report
The maximum length is 64. |
| TXTLEN= (optional) | NA | Width of the text displayed in the report
This parameter is a value from 60 to 132. The default is 72. |
| VERSION= (optional) | 09 | Version value associated with a package
The maximum length is 122. |

You can use any combination of execution parameters, as long as the total byte count of all execution parameters and their values does not exceed 100.

This parameter is required; there is no default value.

**Impact Analysis parameters**

You can use the following parameters in your foreground or batch Impact Analysis job.
Table 21: Impact Analysis parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COLLID= (required)</td>
<td>08</td>
<td>Collection ID for packages a b This parameter is required when the object type is PACKAGE or BOTH; the default value is %.</td>
</tr>
<tr>
<td>COLNAMES= (optional)</td>
<td>NA</td>
<td>Column name list indicated for change on the report a b</td>
</tr>
<tr>
<td>CREATOR= (optional)</td>
<td>05</td>
<td>Creator name of the table, view, synonym, or alias a b</td>
</tr>
<tr>
<td>DBRMNAME = (required)</td>
<td>NA</td>
<td>Name of the DBRM to be analyzed This parameter is required when the object type is DBRM or BOTH; there is no default value.</td>
</tr>
<tr>
<td>DDFLOC= (optional)</td>
<td>14</td>
<td>Name of DB2 location, if you plan to access data on a different subsystem using DDF a The maximum length is 16.</td>
</tr>
<tr>
<td>--ENDOBJ (required)</td>
<td>NA</td>
<td>End-delimiter for a set of Impact parameters associated with a specific object When Explain encounters this parameter, it processes the preceding Explain parameters (delimited by the --NEWOBJ parameter) as a completed Explain request.</td>
</tr>
<tr>
<td>NAME= (required)</td>
<td>07</td>
<td>Name of the package to be analyzed a b This parameter is required when the object type is PACKAGE or BOTH; the default value is %.</td>
</tr>
<tr>
<td>--NEWOBJ (required)</td>
<td>NA</td>
<td>Start-delimiter for a set of Impact parameters associated with a specific object You must also delimit the end of the set with the --ENDOBJ parameter. The delimited set of parameters are treated as a single request. If the job step has only one request, you can omit this parameter.</td>
</tr>
</tbody>
</table>
| OBJECT= (required)   | 04                  | Object type to be analyzed a Specify one of the following values:  
|                      |                     | ■ PACKAGE  
|                      |                     | ■ DBRM  
|                      |                     | ■ BOTH  
<p>|                      |                     | This parameter is required; there is no default value. |
| PLAN= (required)     | 02                  | Product plan name used for executing the product function a This parameter is required; there is no default value. |
| PLNAME= (required)   | 06                  | Plan name of the DBRM to be analyzed a This parameter is required when the object type is DBRM or BOTH; the default value is %. |</p>
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| RECALL= (optional) | 97 | Whether to restore the partitioned data sets containing the DBRMs for the specified plans and packages. Specify one of the following values:  
- **Y** specifies to restore the data sets.  
- **N** specifies not to restore the data sets. (default) |
| RPTHDR= (optional) | 21 | Whether to include report headings in the batch report. Specify one of the following values:  
- **Y** specifies to include the report headings. (default)  
- **N** specifies not to include the report headings; use this option if you plan to review the report online from an output data set. |
| RPTOPT= (optional) | 75 | Type of information displayed in a report. Specify one of the following values:  
- **BOTH** displays a report that provides a summary of impacted objects and a report that provides a list of all impacted SQL statements. (default)  
- **OBJECT** displays a report that provides a summary of impact on DBRMs, packages, and tables in an Impact Analysis job.  
- **SQLTEXT** displays a report that provides a list of all impacted packages and DBRMs, including the SQL statement numbers and SQL text. |
| SQLTYPE= (required) | NA | Type of SQL statements to include in the Impact Analysis process with an occurrence of the specified column name. Specify one of the following DML statement types:  
- **ALL**  
- **DELETE**  
- **INSERT**  
- **SELECT**  
- **UPDATE**  
There is no default value. |
| SSID= (required) | 01 | DB2 subsystem name to which you are attached. You cannot change the subsystem value within a single job step. The maximum length is 4. |
| TBNNAME= (required) | NA | Name of the table, view, synonym, or alias containing the specified column for Impact Analysis. The table name can contain wildcards; there is no default value. |
| TITLE= (optional) | NA | Title to display on each page of the report. The maximum length is 64. The default is blank. |
DBRM/Load Compare parameters

You can use the following parameters in your foreground or batch DBRM/Load job.

Table 22: DBRM/Load parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VERSION= (required)</td>
<td>09</td>
<td>Version value associated with a package. This parameter is required when the object type is PACKAGE or BOTH; the default value is %. The maximum length is 122.</td>
</tr>
</tbody>
</table>

You can use any combination of execution parameters, as long as the total byte count of all execution parameters and their values does not exceed 100.

b The maximum length is 128.

Migrate Access Path Statistics parameters

You can use the following parameters in your foreground or batch Migrate Access Path Statistics job.
Note

Do not use an equal sign (=) with the Migrate Access Path Statistics parameters and Declarations Generator options.

Table 23: Migrate Access Path Statistics parameters

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Execution parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMIT (optional)</td>
<td>NA</td>
<td>Processing option for an Update job Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ ANY runs a COMMIT command regardless of errors or warnings.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOERROR runs a COMMIT command if there are warnings but no errors. (default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ NOWARNING runs a COMMIT command if there are no errors or warnings.</td>
</tr>
<tr>
<td>COMMITFREQ (optional)</td>
<td>NA</td>
<td>Frequency for running a COMMIT command during an Update job Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TS runs a COMMIT command after the statistics relating to all objects in the table space are updated and before the next table space occurs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ DB runs a COMMIT command after the statistics relating to all objects in the database are updated and before the next database occurs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ TB runs a COMMIT command after the statistics relating to all objects in the table are updated and before the next table occurs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ EOJ runs a COMMIT command at the end of the update statistics job.</td>
</tr>
<tr>
<td>DBNAME (optional)</td>
<td>NA</td>
<td>Used for renaming database when new (target) database name is different from current (source) database name</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use the following format for this parameter: DBNAME currentDBName newDBName.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: The database you rename must already exist on the target subsystem.</td>
</tr>
<tr>
<td>DDFLOC</td>
<td>NA</td>
<td>Name of the DB2 location, used when accessing data on a different subsystem using DDF The maximum length is 16.</td>
</tr>
<tr>
<td>DELETE (optional)</td>
<td>NA</td>
<td>Whether to delete all previous instances of the target table spaces’ SYSCOLDIST and SYSTABSTATS entries before inserting the migrated statistics rows</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ Y deletes the entries.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>■ N retains the entries. (default)</td>
</tr>
<tr>
<td>Parameter name</td>
<td>Execution parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>IXCREATOR (optional)</td>
<td>NA</td>
<td>Used for renaming index creator when new (target) index creator is different from current (source) index creator. Use the following format for this parameter: <strong>IXCREATOR currentIXCreator newIXCreator</strong>. The index creator names are case sensitive. Not: The index creator you rename must already exist on the target subsystem.</td>
</tr>
<tr>
<td>IXNAME (optional)</td>
<td>NA</td>
<td>Used for renaming index when new (target) index name is different from current (source) index name. Use the following format for this parameter: <strong>IXNAME currentIXName newIXName</strong>. The index names are case sensitive. To use an index name containing lowercase characters, delimit the name by enclosing it in quotation marks. If you specify question marks as part of an index creator name, you must double the quotation marks in the object name and enclose it in quotation marks. Not: The index you rename must already exist on the target subsystem.</td>
</tr>
<tr>
<td>MIGSTATS TABLESPACE (optional)</td>
<td>NA</td>
<td>Name of the table space you want to migrate, in <strong>databaseName.tablespaceName</strong> format.</td>
</tr>
<tr>
<td>MIGSTATS UPDATE (optional)</td>
<td>NA</td>
<td>One or more target object types in the generated Update file to be updated or inserted. Update the object type by typing any character in the space beside the object type. You can select individual objects or all of the objects. Only the statistics for the selected object types are updated or inserted.</td>
</tr>
<tr>
<td>ONERROR (optional)</td>
<td>NA</td>
<td>Method for proceeding if the product encounters an SQL error. Specify one of the following values:  ■ CONTINUE keeps processing even if an SQL error occurs.  ■ QUIT stops processing when any SQL error occurs, including an SQL error of +100 Not Found on an Update statement. (default) Not: This option has no effect if you also specify the COMMIT parameter with value ANY. If you specify the COMMIT parameter with value NOWARNING and specify the ONERROR parameter with value CONTINUE, the results show all SQL errors, but no updates are committed.</td>
</tr>
<tr>
<td>RETRIES (optional)</td>
<td>NA</td>
<td>Number of times to retry executing the UPDATE or INSERT statement before terminating. You might need to retry the SQL statement if a data set is not available or needs to be restored. This parameter is a value from 0 to 10; the default is 5. Not: This parameter applies only to Update jobs.</td>
</tr>
<tr>
<td>Parameter name</td>
<td>Execution parameter</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| STATSTIME     | NA                  | Timestamp to use for updating the STATSTIME on the target objects. Specify one of the following values:  
- **Current** uses the actual machine time as the target timestamp value.  
- **Source** uses the source object’s timestamp as the target timestamp value.  
**Note:** In each of these instances, the timestamp microseconds is changed to 999999 to identify the statistics as migrated. |
| TBCREATOR     | NA                  | Used for renaming table creator when new (target) table creator is different from current (source) table creator.  
Use the following format for this parameter: **TBCREATOR currentTBCreator newTBCreator**. The table creator names are case sensitive.  
To use a table creator name containing lowercase characters, delimit the name by enclosing it in quotation marks. If you specify question marks as part of a table creator name, you must double the quotation marks in the object name and enclose it in quotation marks.  
**Note:** The table creator that you rename must already exist on the target subsystem. |
| TSNAME        | NA                  | Used for renaming table space when the new (target) table space name is different from the current (source) table space name.  
Use the following format for this parameter: **TSNAME currentTSName newTSName**.  
**Note:** The table space you rename must already exist on the target subsystem. |

---

**Declarations Generator options**

You can use the following options in your foreground or batch Declarations Generator job.  
**Note**  
Do not use an equal sign (=) with the Migrate Access Path Statistics parameters or Declarations Generator options.
### Table 24: Declarations Generator Options

<table>
<thead>
<tr>
<th>Option name</th>
<th>Execution option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ACTION</strong> <em>(optional)</em></td>
<td>NA</td>
<td>What the Declarations Generator does with the generated output when sending it to a partitioned or sequential data set Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>A</strong> (ADD) adds the declarations to a sequential data set or adds a new member to a partitioned data set. <em>(default)</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>R</strong> (REPLACE) replaces an existing data set or member with the new data set or member. The Declarations Generator creates a new member for a partitioned data set lacking a member with the specified name.</td>
</tr>
<tr>
<td><strong>AT LOCATION</strong> <em>(optional)</em></td>
<td>NA</td>
<td>Location of the table or view name on another DB2 subsystem You can use an _ (underscore) in this field. If you specify this option, the location is used as a prefix for the fully-qualified table name. If you do not specify this option, the Declarations Generator assumes that the object is located on the DB2 SSID that you specified on the SQL Explorer main menu, and that the location name does appear (or is not generated) as part of the name in the DECLARE TABLE statement.</td>
</tr>
<tr>
<td><strong>COLSUFFIX</strong> <em>(optional)</em></td>
<td>NA</td>
<td>Whether to form field names by concatenating the column name to the prefix that you specify using the NAMES option Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Y</strong> specifies that the field names are formed in this manner.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>N</strong> specifies that the column name is not used as a suffix, and that field names are controlled by the NAMES value. <em>(default)</em></td>
</tr>
<tr>
<td><strong>DBCSDELIM</strong> <em>(optional)</em></td>
<td>NA</td>
<td>Whether the double-byte character set (DBCS) table and column names in the generated DECLARE table statement are delimited</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>Y</strong> specifies that the DBCS table and column names are enclosed in SQL delimiters. <em>(default)</em></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>N</strong> specifies that the DBCS table and column names are not delimited in the table declaration.</td>
</tr>
<tr>
<td><strong>DBCSSYMBOL</strong> <em>(optional)</em></td>
<td>NA</td>
<td>Symbol that is used to denote a graphic data type in a COBOL picture clause Specify one of the following values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>G</strong> uses the letter G as the notation for graphic data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- <strong>N</strong> uses the letter N as the notation for graphic data.</td>
</tr>
<tr>
<td>Option name</td>
<td>Execution option</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| INDVAR         | NA               | Whether to create an indicator variable for each column that might contain a NULL value  
Specify one of the following values:  
- Y creates the indicator variable (default)  
- N does not create the indicator variable |
| LABEL          | NA               | Whether to include column labels in the output as comments  
Specify one of the following values:  
- Y includes the column labels.  
- N bypasses the column labels. (default) |
| LANGUAGE       | NA               | Language of the generated table declaration  
Specify one of the following values:  
- COBOL specifies any COBOL language. (default)  
- PLI specifies PL/I.  
- C specifies C/370 or C++. |
| LIBRARY        | NA               | Name of the data set in which the table declarations are to be stored  
The Declarations Generator must be able to access this data set, which must also already exist. The data set can be sequential or partitioned.  
If you specify a partitioned data set, you must also specify a PDS member name.  
You can use apostrophes as delimited identifiers for this option. If you do not use delimiters, the Declarations Generator applies a standard TSO prefix (user ID) to the data set name that you provide. |
| NAMES          | NA               | Field name prefix that is formed in the declaration  
If you specify this option and the COLSUFFIX value is N, the field names contain the prefix, concatenated with a number that is one to three digits in length.  
**Note:** If you do not specify a value for this option, the Declarations Generator issues a warning message and uses the column names in the table or view as the field names.  
If you use special or lowercase characters for this option, you must enclose the field name prefix in quotation marks. |
| OWNER          | NA               | Name of the table or view’s owner  
If the table is a local table and you do not specify this value, DB2 uses your TSO logon ID as the table qualifier. If the table is at a remote location, you must specify a value for this option. |
| QUALIFYTB      | NA               | Whether to include the name of the table or view’s owner in the generated DECLARE statement  
Specify one of the following values:  
- Y includes the name. (default)  
- N does not include the name. |
<table>
<thead>
<tr>
<th>Option name</th>
<th>Execution option</th>
<th>Description</th>
</tr>
</thead>
</table>
| QUOTE          | NA               | String delimiter character used in the host language  
                        This option is only applicable for COBOL programs. Use one of the following keywords:  
                        ■ A specifies an apostrophe (’ ) as the string delimiter character.  
                        ■ Q specifies a quotation mark (" ) as the string delimiter character. |
| STRUCTURE      | NA               | Name of the generated data structure  
                        To use special or lowercase characters in the data structure name, enclose the data structure name in quotation marks.  
                        If you do not specify this option, the default structure name is the name of the table or view, with the prefix DCL. If the table or view name is a DBCS string and the language is COBOL or PL/I, the prefix consists of DBCS characters. |
| TABLE          | NA               | Name of the table or view for which you want the Declarations Generator to produce table declarations  
                        This option is case sensitive. If you use special or lowercase characters in this field, you must enclose the table name in quotation marks.  
                        If you specify an apostrophe or quotation mark as part of a table name, you must double each one and use the delimited identifiers (" '' " or " " ). You can also include spaces in a table name by enclosing the name in apostrophes or quotation marks. However, you cannot use wildcards for this option. |
Return codes

This appendix presents the following topics:

- “Overview of return codes” on page 221
- “Explain return codes” on page 222
- “Compare return codes” on page 222
- “Mismatch return codes” on page 223
- “Impact Analysis return codes” on page 223
- “DBRM/Load Compare return codes” on page 224
- “DBRMLIB to DB2 Catalog Compare return codes” on page 224
- “Migrate Access Path Statistics return codes” on page 225
- “Declarations Generator return codes” on page 225

Overview of return codes

A return code alerts you to possible performance problems you can improve, or to certain events occurring during a particular operation.

Return codes are set based on the severity of the rules that are issued. You can customize the severity of the rules.

Note

Return codes for the Explain and Compare function are the same as those for the individual functions themselves.
Related Information

- “Explain return codes” on page 222
- “Compare return codes” on page 222
- “Customizing expert rules” on page 36

Explain return codes

The following table lists the return codes for the Explain function and gives information about the events that cause Explain to issue a particular return code.

Table 25: Return codes for the Explain function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only informational messages were issued.</td>
<td>0</td>
</tr>
<tr>
<td>At least one severe or warning message was issued.</td>
<td>4</td>
</tr>
<tr>
<td>A connection error, unexpected SQL errors, or memory errors occurred.</td>
<td>12</td>
</tr>
</tbody>
</table>

Return codes are set based on the severity of the rules that are issued. You can customize the severity of the rules.

Related Information

- “Customizing expert rules” on page 36

Compare return codes

The return code alerts you if the access path got worse when the Compare job was executed.

You can monitor the return code to prevent other steps from executing.

The factors affecting return codes are access path differences and the total cost. If at least one statement in the job has a difference in access path or total cost, the return code is affected.
Table 26 on page 223 lists the return codes for the Compare function and gives information about the events that cause Compare to issue a particular return code.

Table 26: Return codes for the Compare function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>No differences in access paths and total cost did not increase.</td>
<td>0</td>
</tr>
<tr>
<td>Input parameters did not match any objects in the history table.</td>
<td>1</td>
</tr>
<tr>
<td>Note: The following message prints at the bottom of the detail report when the number statements processed equals zero: SETTING COND CODE TO 1 BECAUSE THE INPUT PARAMETERS DID NOT MATCH ANY OBJECTS IN THE HISTORY TABLE.</td>
<td></td>
</tr>
<tr>
<td>No differences in access paths, but total cost for at least one statement increased.</td>
<td>2</td>
</tr>
<tr>
<td>At least one statement had differences in access paths, but the total cost for the statement did not increase.</td>
<td>4</td>
</tr>
<tr>
<td>The total cost for at least one statement increased, and there was a change in the access path or statistics.</td>
<td>8</td>
</tr>
<tr>
<td>Connection problems exist.</td>
<td>12</td>
</tr>
</tbody>
</table>

Mismatch return codes

A mismatch return code alerts you when a mismatch occurs that might benefit from a change to the column, host variable, or expression in the SQL statement.

Table 27 on page 223 lists the return codes for the Mismatch Analysis function and gives information about the events that cause Mismatch Analysis to issue a particular return code.

Table 27: Return codes for the Mismatch Analysis function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>No conditions occurred that resulted in a category of NO IMPACT or MISMATCH.</td>
<td>0</td>
</tr>
<tr>
<td>At least one category of NO IMPACT occurred.</td>
<td>4</td>
</tr>
<tr>
<td>At least one mismatch occurred, resulting in a category of MISMATCH.</td>
<td>8</td>
</tr>
<tr>
<td>Connection problems exist.</td>
<td>12</td>
</tr>
</tbody>
</table>

Impact Analysis return codes

The return code alerts you when certain events occur during the execution of an Impact Analysis job.
Table 28 on page 224 lists the return codes for the Impact Analysis function and gives information about the events that cause Impact Analysis to issue a particular return code.

Table 28: Return codes for the Impact Analysis function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Impact Analysis found objects and some statements.</td>
<td>0</td>
</tr>
<tr>
<td>Impact Analysis did not find the object, or found the object but no statements.</td>
<td>4</td>
</tr>
<tr>
<td>An input parameter error or SQL error occurred.</td>
<td>12</td>
</tr>
</tbody>
</table>

DBRM/Load Compare return codes

The return code alerts you when certain events occur during a DBRM/Load comparison.

Table 29 on page 224 lists the return codes for the DBRM/Load Compare function and gives information about the events that cause this function to issue a particular return code.

Table 29: Return codes for the DBRM/Load Compare function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>The DBRM/Load Compare job ran to completion and the job results were written to a report.</td>
<td>0</td>
</tr>
<tr>
<td>An input parameter error was found.</td>
<td>8</td>
</tr>
<tr>
<td>A severe error has occurred. Refer to messages in the job log for more information.</td>
<td>12</td>
</tr>
</tbody>
</table>

DBRMLIB to DB2 Catalog Compare return codes

The return code alerts you when certain events occur during a DBRMLIB to DB2 Catalog comparison.

Table 30 on page 225 lists the return codes for the DBRMLIB to DB2 Catalog Compare function and gives information about the events that cause this function to issue a particular return code.
Table 30: Return codes for the DBRMLIB to DB2 Catalog Compare function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only informational messages were issued.</td>
<td>0</td>
</tr>
<tr>
<td>Only warning messages were issued.</td>
<td>4</td>
</tr>
<tr>
<td>At least one severe message was issued.</td>
<td>8</td>
</tr>
<tr>
<td>An input parameter or SQL error occurred, or the open process failed for the data set.</td>
<td>12</td>
</tr>
</tbody>
</table>

Migrate Access Path Statistics return codes

The return code alerts you when certain events occur during the execution of a Migrate Access Path Statistics job.

Table 31 on page 225 lists the return codes for the Migrate Access Path Statistics function and gives information about the events that cause Migrate Access Path Statistics to issue a particular return code.

Table 31: Return codes for the Migrate Access Path Statistics function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Migrate Access Path Statistics job ran to completion.</td>
<td>0</td>
</tr>
<tr>
<td>A possible external condition might have affected the run.</td>
<td>2</td>
</tr>
<tr>
<td>A warning condition was encountered.</td>
<td>4</td>
</tr>
<tr>
<td>A severe issue was encountered.</td>
<td>8</td>
</tr>
<tr>
<td>No object was found.</td>
<td>10</td>
</tr>
<tr>
<td>An input parameter error was found.</td>
<td>12</td>
</tr>
<tr>
<td>A connection error, unexpected SQL errors, or memory errors occurred.</td>
<td>30-39</td>
</tr>
</tbody>
</table>

Declarations Generator return codes

The return code alerts you when certain events occur during a Declarations Generator operation.

Table 32 on page 226 lists the return codes for the Declarations Generator function and gives information about the events that cause the Declarations Generator to issue a particular return code.
Table 32: Return codes for the Declarations Generator function

<table>
<thead>
<tr>
<th>Condition</th>
<th>Return code</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Declarations Generator operation ran to completion.</td>
<td>0</td>
</tr>
<tr>
<td>The Declarations Generator encountered a warning.</td>
<td>4</td>
</tr>
<tr>
<td>An input parameter error or SQL error occurred.</td>
<td>8</td>
</tr>
<tr>
<td>Connection problems exist.</td>
<td>12</td>
</tr>
</tbody>
</table>
Sample data set members for adding analysis to job steps

You can use the Explain and Compare functions as an instream part of your precompile or BIND jobs. Processing the DBRMLIB member following the precompile enables the developer to perform a dynamic Explain on the embedded application SQL prior to binding the DBRM to a plan or package. Processing the plan, package, or DBRM following the BIND job enables the developer to perform a dynamic Explain using the current statistics and to compare the access path to a prior explain saved to the product history tables.

The members listed in Table 33 on page 227 are delivered in the hlq.llqSAMP data set:

<table>
<thead>
<tr>
<th>Member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSSPCOMP</td>
<td>Input parameters for Compare</td>
</tr>
<tr>
<td>PSSPDM</td>
<td>PROCLIB member to process a DBRM</td>
</tr>
<tr>
<td>PSSPDMILI</td>
<td>PROCLIB member to process a DBRMLIB</td>
</tr>
<tr>
<td>PSSPEXP</td>
<td>Input parameters for Explain</td>
</tr>
<tr>
<td>PSSPJDM</td>
<td>Job to process a DBRM</td>
</tr>
<tr>
<td>PSSPJDMILI</td>
<td>Job to process a DBRMLIB member</td>
</tr>
<tr>
<td>PSSPJPG</td>
<td>Job to process a package</td>
</tr>
<tr>
<td>PSSPJPL</td>
<td>Job to process a plan</td>
</tr>
<tr>
<td>PSSPPG</td>
<td>PROCLIB member to process a package</td>
</tr>
<tr>
<td>PSSPPL</td>
<td>PROCLIB member to process a plan</td>
</tr>
</tbody>
</table>

Note
The name of hlq.llqSAMP varies depending on your installation options and might be hlq.BMCSAMP.
Sample history table Explain queries

Before you can run these queries, you must first have run an Explain with the HISTORY parameter set to YES, and the RETAIN parameter set to a number greater than zero.

For instructions about running the query, see the comment in each query member.

Note
Be sure to verify the creator name for the synonyms used during the installation of SQL Explorer. You must use this creator name as the synonym qualifier in the SET CURRENT SQLID statement in each query. Use the query PSSGETSY to locate the synonym qualifier.

The following members are delivered in the hlq.lkSAMP data set:

Table 34: Sample history table Explain queries

<table>
<thead>
<tr>
<th>Query member</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSSGETSY</td>
<td>Finds the synonym to use for the SQL Explorer history table</td>
</tr>
<tr>
<td>PSSHYBRD</td>
<td>Finds queries performing hybrid joins, which are normally poor performers</td>
</tr>
<tr>
<td>PSSIXCNT</td>
<td>Returns a count of the number of rows using a specified index</td>
</tr>
<tr>
<td>PSSIXNOM</td>
<td>Finds queries performing nonmatching index scans</td>
</tr>
<tr>
<td>PSSIXPTM</td>
<td>Finds queries where match columns are not equal to all index columns</td>
</tr>
<tr>
<td>PSSIXRAT</td>
<td>Finds queries where the cluster ratio is less than a specified percent</td>
</tr>
<tr>
<td>PSSLPREF</td>
<td>Finds queries using list prefetch</td>
</tr>
<tr>
<td>PSSMTCOL</td>
<td>Finds queries using a specified index, and shows the number of columns that match</td>
</tr>
<tr>
<td>PSSTBOIX</td>
<td>Finds queries accessing tables that have no indexes</td>
</tr>
<tr>
<td>PSSTSSCA</td>
<td>Finds queries performing table space scans on tables with CARD greater than a specified value</td>
</tr>
<tr>
<td>Query member</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>PSSUSEIX</td>
<td>Finds queries using a specified index</td>
</tr>
</tbody>
</table>

**Note**

The name of `hlq.IIqSAMP` varies depending on your installation options and might be `hlq.BMCSAMP`. 
Available variables for expert rules

This appendix lists the variables that you can use in expert rules.

List of variables

The following table lists the variables that are available for use in expert rules.

For more information about expert rules, see “Customizing expert rules” on page 36.

Table 35: Variables in expert rules

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Default fact that is used as a placeholder and always results in a false condition</td>
</tr>
<tr>
<td>ACCESSCREATOR</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>ACCESSNAME</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>ACCESTYPE</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>ACCESS_DEGREE</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>ACCESS_DEGREEC</td>
<td>String equivalent of the ACCESS_DEGREE</td>
</tr>
<tr>
<td>ACCESS_PGROUP_ID</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>ACCESS_PGROUP_IDC</td>
<td>String equivalent of ACCESS_PGROUP_ID</td>
</tr>
<tr>
<td>ACQUIRE</td>
<td>Column from the SYSIBM.SYSPLAN table</td>
</tr>
<tr>
<td>AMBIGUOUS_CURSOR</td>
<td>Condition detected in which the cursor is ambiguous</td>
</tr>
<tr>
<td>AVG</td>
<td>Condition detected in which the AVG function was used on a column</td>
</tr>
<tr>
<td>BPOOL</td>
<td>Value that contains the name of the buffer pool in which the table resides</td>
</tr>
<tr>
<td>CARDF</td>
<td>Column from the SYSIBM.SYSTABLES table</td>
</tr>
<tr>
<td>Variable name</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>CARDFC</td>
<td>String equivalent of CARDF</td>
</tr>
<tr>
<td>CHECKFLAG</td>
<td>Column from the SYSIBM.SYSTABLES table</td>
</tr>
<tr>
<td>CHILDREN</td>
<td>Column from the SYSIBM.SYSTABLES table</td>
</tr>
<tr>
<td>CHILDRENC</td>
<td>String equivalent of CHILDREN</td>
</tr>
<tr>
<td>CLOSERULE</td>
<td>Column from the SYSIBM.SYSTABLESPACE table</td>
</tr>
<tr>
<td>CLUSTERED</td>
<td>Column from the SYSIBM.SYSINDEXES table</td>
</tr>
<tr>
<td>CLUSTERING</td>
<td>Column from the SYSIBM.SYSINDEXES table</td>
</tr>
<tr>
<td>CLUSTERRATIOF</td>
<td>Column from the SYSIBM.SYSINDEXES and SYSIBM.SYSINDEXSTATS tables</td>
</tr>
<tr>
<td>CLUSTERRATIOFC</td>
<td>String equivalent of CLUSTERRATIOF</td>
</tr>
<tr>
<td>COALESCE</td>
<td>Value indicating that the query used the COALESCE function</td>
</tr>
<tr>
<td>COL43</td>
<td>Value that contains a column identifier for index screening</td>
</tr>
<tr>
<td>COL45</td>
<td>Value that contains a column identifier for index screening</td>
</tr>
<tr>
<td>COLLID</td>
<td>Column from the SYSIBM.SYSPACKAGE table</td>
</tr>
<tr>
<td>COLUMN_FN_EVAL</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>COMPRESS</td>
<td>Column from the SYSIBM.SYSTABLEPART table</td>
</tr>
<tr>
<td>CORSUB</td>
<td>Condition detected in which a correlated subquery exists</td>
</tr>
<tr>
<td>COST_CATEGORY</td>
<td>Column from the DSN_STATEMENT_TABLE table</td>
</tr>
<tr>
<td>CREATOR</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>DATAROWS</td>
<td>Value that contains the number of data rows</td>
</tr>
<tr>
<td>DATAROWSC</td>
<td>String equivalent of DATAROWS</td>
</tr>
<tr>
<td>DBNAME</td>
<td>Column from the SYSIBM.SYSTABLESPACE table</td>
</tr>
<tr>
<td>DBRM</td>
<td>Column from the SYSIBM.SYSDBRM table</td>
</tr>
<tr>
<td>DECLARE</td>
<td>Condition detected in which the statement defines a cursor</td>
</tr>
<tr>
<td>DEGREE</td>
<td>Column from the SYSIBM.SYSPACKAGE and SYSIBM.SYSPLAN tables</td>
</tr>
<tr>
<td>DELETE_NO_WHERE</td>
<td>Condition detected in which a DELETE statement without a predicate was detected</td>
</tr>
<tr>
<td>DIFFTIME</td>
<td>Value that contains the amount of time since RUNSTATS was run</td>
</tr>
<tr>
<td>DISTINCT</td>
<td>Condition detected in which the DISTINCT keyword was used</td>
</tr>
<tr>
<td>EDITPROC</td>
<td>Condition detected in which a table used an EDITPROC</td>
</tr>
<tr>
<td>EQALL</td>
<td>Condition detected in which the COL=ALL operator was used</td>
</tr>
<tr>
<td>EQANY</td>
<td>Condition detected in which the COL=ANY or COL=SOME operator was used</td>
</tr>
<tr>
<td>Variable name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>EXISTS</td>
<td>Condition existed in which the EXISTS keyword was used</td>
</tr>
<tr>
<td>EXPREDICATE</td>
<td>Column from the SYSIBM.SYSPLAN table</td>
</tr>
<tr>
<td>EXPRESSION</td>
<td>Condition detected in which a COL= expression operator was used</td>
</tr>
<tr>
<td>FAROFFPOSF</td>
<td>Column from the SYSIBM.SYSINDEXPART table</td>
</tr>
<tr>
<td>FAROFF_RATIO</td>
<td>Value that contains the far-off index ratio</td>
</tr>
<tr>
<td>FF43</td>
<td>Value that contains a filter factor within a predicate</td>
</tr>
<tr>
<td>FF43C</td>
<td>This value is used when managing filter factors.</td>
</tr>
<tr>
<td>FF45</td>
<td>Value that contains a filter factor within a predicate</td>
</tr>
<tr>
<td>FF45C</td>
<td>This value is used when managing filter factors.</td>
</tr>
<tr>
<td>FIRSTKEYCARDDF</td>
<td>Column from the SYSIBM.SYSINDEXES table</td>
</tr>
<tr>
<td>FIRSTKEYCARDDFC</td>
<td>String equivalent of FIRSTKEYCARDDF</td>
</tr>
<tr>
<td>FORUPDATEOF</td>
<td>Condition detected in which SELECT FOR UPDATE syntax was used</td>
</tr>
<tr>
<td>FREEPAGE</td>
<td>Column from the SYSIBM.SYSTABLEPART table</td>
</tr>
<tr>
<td>FREEPAGEC</td>
<td>String equivalent of FREEPAGE</td>
</tr>
<tr>
<td>FULLKEYCARDDF</td>
<td>Column from the SYSIBM.SYSINDEXES table</td>
</tr>
<tr>
<td>FULLKEYCARDDFC</td>
<td>String equivalent of FULLKEYCARDDF</td>
</tr>
<tr>
<td>FULLOUTER</td>
<td>Condition detected in which the FULL OUTER JOIN operation was used</td>
</tr>
<tr>
<td>GROUPBY</td>
<td>Condition detected in which the GROUPBY clause was used</td>
</tr>
<tr>
<td>HIGH2KEY</td>
<td>Column from the SYSIBM.SYSCOLUMNS table</td>
</tr>
<tr>
<td>HINTUSED</td>
<td>Value that contains the name of the hint used to determine the access path</td>
</tr>
<tr>
<td>INDEXES</td>
<td>Value that contains the number of indexes on a table</td>
</tr>
<tr>
<td>INDEXESC</td>
<td>String equivalent of INDEXES</td>
</tr>
<tr>
<td>INDEXONLY</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>INSELECT</td>
<td>Condition detected in which a COL IN operator was used</td>
</tr>
<tr>
<td>IPCARDF</td>
<td>Value that contains the total numbers of rows in the index</td>
</tr>
<tr>
<td>ISNOTNULL</td>
<td>Condition detected in which the IS NOT NULL operator is used</td>
</tr>
<tr>
<td>ISOLATION</td>
<td>Column from the SYSIBM.SYSPACKAGE and SYSIBM.SYSPLAN tables</td>
</tr>
<tr>
<td>IXBPOOL</td>
<td>Value that contains the address of the buffer pool in which the index resides</td>
</tr>
<tr>
<td>IXCARDF</td>
<td>Condition detected in which an index is missing RUNSTATS statistics</td>
</tr>
</tbody>
</table>
### List of variables

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IXCARDF_CREATOR</td>
<td>Value that contains the creator name of an index table that is missing RUNSTATS statistics</td>
</tr>
<tr>
<td>IXCARDF_NAME</td>
<td>Value that contains the name of an index table that is missing RUNSTATS statistics</td>
</tr>
<tr>
<td>IXCOLCNT</td>
<td>The number of index keys from the SYSIBM.SYSKEYS table The number of key columns in the index</td>
</tr>
<tr>
<td>IXCR37</td>
<td>Value that contains the creator of an index</td>
</tr>
<tr>
<td>IXCR43</td>
<td>Value that contains the creator of an index</td>
</tr>
<tr>
<td>IXCR45</td>
<td>Value that contains the creator of an index</td>
</tr>
<tr>
<td>IXCREATOR</td>
<td>Column from the SYSINDEXES table</td>
</tr>
<tr>
<td>IXDIFF</td>
<td>Value that contains the time difference between when the table RUNSTATS and the index RUNSTATS were generated</td>
</tr>
<tr>
<td>IXFREEPAGE</td>
<td>Value that contains number of the index freepage</td>
</tr>
<tr>
<td>IXFREEPAGEC</td>
<td>String equivalent of IXFREEPAGE</td>
</tr>
<tr>
<td>IXKEYLEN</td>
<td>Value that contains the total length of the index keys</td>
</tr>
<tr>
<td>IXKEYLENC</td>
<td>String equivalent of IXKEYLEN</td>
</tr>
<tr>
<td>IXMATCOLS</td>
<td>A string that contains the names of the matching index keys from the SYSIBM.SYSKEYS table The names of the matching key columns</td>
</tr>
<tr>
<td>IXNAME</td>
<td>Column from the SYSINDEXES table</td>
</tr>
<tr>
<td>IXNM37</td>
<td>Value that contains an index name</td>
</tr>
<tr>
<td>IXNM43</td>
<td>Value that contains an index name</td>
</tr>
<tr>
<td>IXNM45</td>
<td>Value that contains an index name</td>
</tr>
<tr>
<td>IXPCTFREE</td>
<td>Column from the SYSIBM.SYSINDEXEXPART table</td>
</tr>
<tr>
<td>IXPCTFREEC</td>
<td>String equivalent of IXPCTFREE</td>
</tr>
<tr>
<td>IXSCAN</td>
<td>Condition detected in which an index scan is performed</td>
</tr>
<tr>
<td>IXSCANPAGES</td>
<td>Sum of columns NLEAF and NLEVELS from SYSIBM.SYSINDEXES minus 1 The number of index pages scanned</td>
</tr>
<tr>
<td>IXSTATSTIME</td>
<td>Value that contains the time difference between running RUNSTATS on a table and RUNSTATS on the index for that table</td>
</tr>
<tr>
<td>IXVARCHAR</td>
<td>Condition detected in which the index contains a VARCHAR key</td>
</tr>
<tr>
<td>JOIN_DEGREE</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>JOIN_DEGREEC</td>
<td>String equivalent of JOIN_DEGREE</td>
</tr>
<tr>
<td>Variable name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>JOIN_NO_WHERE</td>
<td>Condition detected in which two tables were joined where at least one table does not use a local predicate</td>
</tr>
<tr>
<td>JOIN_PGROUP_ID</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>JOIN_PGROUP_IDC</td>
<td>String equivalent of JOIN_PGROUP_ID</td>
</tr>
<tr>
<td>JOIN_TYPE</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>LEAFDIST</td>
<td>Column from the SYSIBM.SYSINDEXPART table</td>
</tr>
<tr>
<td>LEAFDISTC</td>
<td>String equivalent of LEAFDIST</td>
</tr>
<tr>
<td>LEFTOUTER</td>
<td>Condition detected in which the LEFT OUTER JOIN operation was used</td>
</tr>
<tr>
<td>LE_AND_GE</td>
<td>Condition detected in which &lt;= and &gt;= were used instead of the BETWEEN clause</td>
</tr>
<tr>
<td>LOCKRULE</td>
<td>Column from the SYSIBM.SYSTABLESPACE table</td>
</tr>
<tr>
<td>LOCKSIZE</td>
<td>Value that indicates the size of the lock&lt;br&gt;Values can be P(page), R(row), T(table), or S(tablespace).</td>
</tr>
<tr>
<td>LOW2KEY</td>
<td>Column from the SYSIBM.SYSCOLUMNS table</td>
</tr>
<tr>
<td>MATCHCOLS</td>
<td>Column from the PLAN_TABLE table&lt;br&gt;The number of matching key columns</td>
</tr>
<tr>
<td>MAX</td>
<td>Condition detected in which the MAX function was used on a column</td>
</tr>
<tr>
<td>MAXTABLES</td>
<td>Value that contains the highest numbers of tables that any query block in the SQL statement joined together</td>
</tr>
<tr>
<td>METHOD</td>
<td>Column from PLAN_TABLE table</td>
</tr>
<tr>
<td>MIN</td>
<td>Condition detected in which the MIN function was used on a column</td>
</tr>
<tr>
<td>MISMATCH</td>
<td>Condition detected in which two columns of different types were compared</td>
</tr>
<tr>
<td>MIXOPSEQ</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>MIXOPSEQC</td>
<td>String equivalent of MIXOPSEQ</td>
</tr>
<tr>
<td>NACTIVEF</td>
<td>Column from the SYSIBM.SYSTABLESPACE table</td>
</tr>
<tr>
<td>NACTIVEFC</td>
<td>String equivalent of NACTIVEF</td>
</tr>
<tr>
<td>NAME</td>
<td>Column from the SYSIBM.SYSPACKAGE or the SYSIBM.SYSPLAN tables</td>
</tr>
<tr>
<td>NEALL</td>
<td>Condition detected in which the COL &lt;&gt; ALL operator was used</td>
</tr>
<tr>
<td>NEAROFFPOSF</td>
<td>Column from the SYSIBM.SYSINDEXPART table</td>
</tr>
<tr>
<td>NEAROFF_RATIO</td>
<td>Value that contains the ratio of near-off index rows to the total number of rows</td>
</tr>
<tr>
<td>NLEAF</td>
<td>Column from the SYSIBM.SYSINDEXES table&lt;br&gt;The number of leaf pages in the index</td>
</tr>
<tr>
<td>NLEAFC</td>
<td>String equivalent of NLEAF</td>
</tr>
<tr>
<td>Variable name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NLEVELS</td>
<td>Column from the DB2SYSINDEXES table</td>
</tr>
<tr>
<td>NLEVELSC</td>
<td>String equivalent of NLEVELS</td>
</tr>
<tr>
<td>NOTBETWEEN</td>
<td>Condition detected in which a NOT BETWEEN operator was used</td>
</tr>
<tr>
<td>NOTCOLBETWEEN</td>
<td>Condition detected in which a NOT COL BETWEEN operator was used</td>
</tr>
<tr>
<td>NOTCOLEQ</td>
<td>Condition detected in which a NOT COL = operator was used</td>
</tr>
<tr>
<td>NOTCOLLIKE</td>
<td>Condition detected in which a NOT COL LIKE operator was used</td>
</tr>
<tr>
<td>NOTCOLRANGE</td>
<td>Condition detected in which a NOT COL &lt;,&lt;=,=&gt; operator was used</td>
</tr>
<tr>
<td>NOTEQUAL</td>
<td>Condition detected in which a NOT EQUAL operator was used</td>
</tr>
<tr>
<td>NOTIN</td>
<td>Condition detected in which a NOT IN operator was used</td>
</tr>
<tr>
<td>NOTNULL</td>
<td>Condition detected in which a NOT NULL operator was used</td>
</tr>
<tr>
<td>NPAGESF</td>
<td>Column from the DB2SYSINDEXES table</td>
</tr>
<tr>
<td>NPAGESFC</td>
<td>String equivalent of NPAGESF</td>
</tr>
<tr>
<td>NTABLES</td>
<td>Column from the DB2SYSTABLESPACE table</td>
</tr>
<tr>
<td>NTABLES1</td>
<td>Value that contains the number of tables referenced in the outermost FROM clause of the statement</td>
</tr>
<tr>
<td>OBJECT</td>
<td>Contains the type of object that supplied the variables for the current invocation of the rules engine OBJECT = “PLAN”, “PACKAGE”, “STMT”, “EXPLAIN”, “SQLTEXT”, “TABLE”</td>
</tr>
<tr>
<td>ONLINE</td>
<td>Condition detected in which ONLINE only usage was specified</td>
</tr>
<tr>
<td>OPERATIVE</td>
<td>Column from the DB2SYSPACKAGE and DB2SYSPLAN tables</td>
</tr>
<tr>
<td>OPTHINT</td>
<td>Column from the DB2PLAN_TABLE table</td>
</tr>
<tr>
<td>OPTIMIZE</td>
<td>Condition detected in which the OPTIMIZE FOR n ROWS clause was used</td>
</tr>
<tr>
<td>ORDERBY</td>
<td>Condition detected in which the ORDER BY keyword was used</td>
</tr>
<tr>
<td>OUT</td>
<td>Value that specifies the output of the rule</td>
</tr>
<tr>
<td>PAGE_RANGE</td>
<td>Column from the DB2PLAN_TABLE table</td>
</tr>
<tr>
<td>PARALLELISM_MODE</td>
<td>Column from the DB2PLAN_TABLE table</td>
</tr>
<tr>
<td>PARENTS</td>
<td>Column from the DB2SYSTABLES table</td>
</tr>
<tr>
<td>PARENTSC</td>
<td>String equivalent of PARENTS</td>
</tr>
<tr>
<td>PARTITIONS</td>
<td>Column from the DB2SYSTABLES table</td>
</tr>
<tr>
<td>PARTITIONSC</td>
<td>String equivalent of PARTITIONS</td>
</tr>
<tr>
<td>PCTFREE</td>
<td>Column from the DB2SYSTABLEPART table</td>
</tr>
<tr>
<td>PCTFREEC</td>
<td>String equivalent of PCTFREE</td>
</tr>
<tr>
<td>Variable name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PCTROWCOMPC</td>
<td>Value that contains the percentage of rows that are compressed in a table</td>
</tr>
<tr>
<td>PREFETCH</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>PROC_SU</td>
<td>Column from the DSN_STMTNT_TABLE table</td>
</tr>
<tr>
<td>QBLOCK_TYPE</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>QUALIFIED</td>
<td>Condition detected in which table references in a FROM clause were qualified</td>
</tr>
<tr>
<td>READONLY</td>
<td>Condition detected in which a READ ONLY clause was used</td>
</tr>
<tr>
<td>REASON_HOST</td>
<td>Condition detected in which the query used host variables</td>
</tr>
<tr>
<td>REASONREFERENTIAL</td>
<td>Condition detected in which a statement was used on a table that has referential constraints of CASCADE or SET NULL</td>
</tr>
<tr>
<td>REASON_TABLE_CARD</td>
<td>Condition detected in which one or more tables are missing cardinality statistics</td>
</tr>
<tr>
<td>REASON_TRIGGER</td>
<td>Condition detected in which a particular statement type used triggers</td>
</tr>
<tr>
<td>REASON_UDF</td>
<td>Condition detected in which a query references one or more user-defined functions</td>
</tr>
<tr>
<td>RELEASE</td>
<td>Column from the SYSIBM.SYSPACKAGE and SYSIBM.SYSPLAN tables</td>
</tr>
<tr>
<td>RELEASEC</td>
<td>String equivalent of RELEASE</td>
</tr>
<tr>
<td>ROPTVAR</td>
<td>Column from the SYSIBM.SYSPACKAGE or SYSIBM.SYSPLAN tables</td>
</tr>
<tr>
<td>RIGHTOUTER</td>
<td>Condition detected in which the RIGHT OUTER JOIN operator includes rows from the table after it and the columns values do not match</td>
</tr>
<tr>
<td>SEGSIZE</td>
<td>Column from the SYSIBM.SYSTABLESPACE table</td>
</tr>
<tr>
<td>SEGSIZEC</td>
<td>String equivalent of SEGSIZE</td>
</tr>
<tr>
<td>SELECT_STAR</td>
<td>Condition detected in which the SELECT * FROM TABLE statement was used</td>
</tr>
<tr>
<td>SEQ43C</td>
<td>Value that contains the column sequence number of the index screening column</td>
</tr>
<tr>
<td>SORTC_GROUPBY</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>SORTC_JOIN</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>SORTC_ORDERBY</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>SORTC_PGROUP_ID</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>SORTC_PGROUP_IDC</td>
<td>String equivalent of SORTC_PGROUP_ID</td>
</tr>
<tr>
<td>SORTC_UNIQ</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>SORTN_JOIN</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>SORTN_PGROUP_ID</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>Variable name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SORTN_PGROUP_IDC</td>
<td>String equivalent of SORTN_PGROUP_ID</td>
</tr>
<tr>
<td>STATSTIME</td>
<td>Value that indicates the last time that RUNSTATs were generated</td>
</tr>
<tr>
<td>STEPNO</td>
<td>Value that contains an access path step number</td>
</tr>
<tr>
<td>STEPNOC</td>
<td>String equivalent of STEPNO</td>
</tr>
<tr>
<td>STMTDYN</td>
<td>Condition is detected in which the package or DBRM can perform dynamic SQL</td>
</tr>
<tr>
<td>STMT_TYPE</td>
<td>Column from the DSN_STATEMNT_TABLE table</td>
</tr>
<tr>
<td>SUBSELECTS</td>
<td>Value that contains the number of subselects in the statement</td>
</tr>
<tr>
<td>SUM</td>
<td>Condition detected in which the SUM function was used on a column</td>
</tr>
<tr>
<td>TBTYPE</td>
<td>Column TABLE_TYPE from the PLAN_TABLE table</td>
</tr>
<tr>
<td>TNAME</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>TSLOCKMODE</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>TSNAME</td>
<td>Column from the SYSIBM.SYSTABLESPACE table</td>
</tr>
<tr>
<td>UNION</td>
<td>Condition detected in which an UNION operator was used</td>
</tr>
<tr>
<td>UNIONALL</td>
<td>Condition detected in which an UNION ALL operator was used</td>
</tr>
<tr>
<td>UNIONS</td>
<td>Value that contains the number of times that a UNION operator was used</td>
</tr>
<tr>
<td>UNIQUERULE</td>
<td>Column from the SYSIBM.SYSINDEXES table</td>
</tr>
<tr>
<td>UPDATECUR</td>
<td>Condition detected in which an UPDATE WHERE CURRENT OF statement was used</td>
</tr>
<tr>
<td>UPDATE_KEYCOL</td>
<td>Condition detected in which the SET value clause in an UPDATE statement updates columns that are used in an index</td>
</tr>
<tr>
<td>UPDATE_NO_WHERE</td>
<td>Condition detected in which an UPDATE statement updates all rows in a table</td>
</tr>
<tr>
<td>VALID</td>
<td>Column from the SYSIBM.SYSPACKAGE and SYSIBM.SYSPLAN tables</td>
</tr>
<tr>
<td>VALIDATE</td>
<td>Column from the SYSIBM.SYSPACKAGE and SYSIBM.SYSPLAN tables</td>
</tr>
<tr>
<td>VALPROC</td>
<td>Column from the SYSIBM.SYSTABLES table</td>
</tr>
<tr>
<td>VERSION</td>
<td>Column from the SYSIBM.SYSPACKAGE table</td>
</tr>
<tr>
<td>WHEN_OPTIMIZE</td>
<td>Column from the PLAN_TABLE table</td>
</tr>
<tr>
<td>WILDCARD</td>
<td>Condition detected in which a LIKE operator used a wildcard character in the first position</td>
</tr>
<tr>
<td>ZEROEQONE</td>
<td>Condition detected in which a 1=0 predicate was evaluated</td>
</tr>
</tbody>
</table>
Index

Symbols

_(underscore) as a wildcard 29
--NEWOBJ
  Compare parameters 207

A

acceleration detail
  Explain options panel 76, 79
Accelerator 79
Accelerator rules
  Explain parameters 200
access paths
  area in Explain results 70
accessing Explain with other BMC products 66
action codes in Explain 71
ACTPSS CLIST 158
ad hoc SQL Explain 89
adding objects to batch Explain job 199
additional data set options, specifying 58
ANALYZE command 160, 164
APPLDEV rules 34
array, Declarations Generator 191

B

backup copy of statistics 171
BEGSQL parameter 200
block QQ command 106, 107, 166
BMC Performance for DB2 SQL
  solution 23, 24
BMCEXPLORER (BMCEX) command 68, 158, 160
BMCMSC parameter 200

C

calculating matching percentage 120
changes to the product 12
CLOSE cursor command in PSSRXSQL 41
COLBRK parameter 200
COLLID (08) parameter 200
COLLID parameter
  Compare parameters 206
  Impact Analysis parameters 211
  Mismatch Analysis parameters 209
COLLID2 parameter 206
COLNAMES parameter 211
Command line 28
commands
  ANALYZE 160, 164
  BMCEXPLORER (BMCEX) 158, 160
  DESCRIBE 159, 161
  DOWN 28
  END 28
  FIND 69
  GET 159, 163
  HELP 28, 29
  panels
    SQL Host Variables List 107
    QQ 106, 107, 166
    RFIND 69
    UP 28
    ZOOM 69
commands in Explain 72
COMMIT command in PSSRXSQL 41
Compare
  parameters 206
  process, defined 119
  return codes 222
COMPARE parameter 200
COMPNEW parameter 206
COMPOLD parameter 206
CONTOKEN parameter 200
conventions, documentation 12
COSTSUM parameter 206
CREATOR parameter 211
CURRENCY (87) parameter 200
CURRENCY parameter 206
customizable reports
  printing SQL error messages 43
  PSSRXSQL external function 41
  SYSTERM data set 43
customizing a default layout member in Explain 199

d
DAA 79
data set options, specifying 56
data-type mismatch, detecting 133
DB2 Analytics Accelerator 79
DBRM parameter 201
DBRM/Load Compare
  defined 147
  overview 148
  parameters 213, 216
  return codes 224
  wildcard characters 149
DBRMLIB parameter 201
DBRMLIB/DB2 Catalog Compare
  defined 147
  editable tokens 151
  guidelines for editing PSSD2CJB member 153
  message categories 151
  overview 151
  PSSD2CJB member 153
  return codes 224
  wildcard characters 154
DBRMNAME parameter 211
DBRM
  Explaining statements with SQL Explorer 161
  explaining with SQL Explorer 160
  DDFLOC parameter 201, 206, 209, 211
Declarations Generator
  benefits to application programmers 191
  delimited identifiers 194
  options 216
  DECLARE TABLE statement 191
default layout data set for Explain 73
DEFAULT rules 34
definition, product 17
DEGREE parameter 201
delimited identifiers, Declarations Generator 194
describe command 159, 161
diagnosing performance problems with Explain 64
DIFF parameter 207, 209
documentation information 11
DOMEPNL3 panel 27
DOWN command 28
DSNEXIT library 186
DSNLOAD library 186
dynamic Explain 65
ed
editing the Explain report layout 93
ingedit tokens in DBRMLIB/DB2 Catalog Compare 151
END command 28
ENDOBJ parameter 201
  Compare parameters 207
  DBRM/Load Compare parameters 213
  Impact Analysis parameters 211
  Mismatch Analysis parameters 209
ENDSQL parameter 201
error messages
  PSSRXSQL 42
examples
  BIND statement generated by PSSD2CJB job 155
  edited PSSD2CJB Member 153
  Explain return codes 222
  generated JCL for Migrate Access Path
    Statistics 174
  output from Declarations Generator operation 191
  results of SQL statement execution 110
  sample update file 183
  sample update job output listing 186
  sample update job step in generated JCL 186
  specifying SQL for SQLX edit macro 106
  SYSPRINT generated by PSSD2CJB job 155
EXECUTE command in PSSRXSQL 41
expert rule variable 38
expert rules
  defined 34
  help 44
  overview 34
explain
  commands 72
  Explain
access path information 64
action codes 71
action codes for multiple objects 71
ad hoc SQL edit session 89
adding objects to batch job 199
alter back-level plan table 65
batch parameters 199
comments in default layout data set 73
dynamic 65
editing report layout 93
entry points 66
equal sign for actions 71
Explain Compare 65
Extended 66
functionality 63
generating a list of objects 74
generating a list of packages 74
highlighting results 65
HISTORY parameter 64
ISO 4217 standards 80
layout data set 73
main panels 68
objects 64
online 85
online parameters 199
parameters 199
plan table compatibility 65
PSSREPA member 73
results 91
results in access path area 70
return codes 221
specifying batch options 76
specifying options 76
stacking of results 65
static 65
taskbar 70
using to diagnose performance problems 64
What-If Edit SQL 65
What-If Index 64
Explain and Compare
return codes 221
Explain Compare 65
Explain options panel
acceleration detail 76, 79
Explain parameters
  Accelerator rules 200
Explain Report fields 95
Explain Results, statement area 69
explaining a package in batch 87
explaining a package online 85
explaining an ad hoc SQL statement 89
Extended Explain
description of 66
external functions
PSSRXSQL 41
extract job, migrate access path statistics 183
G
GET command 159, 163
H
HELP command 28
HISTORY parameter 201
host variable values, adding 105
I
Impact Analysis
column changes 141
defined 141
parameters 210
return codes 224
indicator structure, Declarations Generator 191
INSERT parameter 202
INSTALL SYSADM authority 22
INVALID COMMAND message 106, 166
ISO 4217 standards in Explain 80
ISPF conventions 28
ISPF keys and commands 28
M
maintenance
  applying to SQL Explorer 28
MATCHPCT parameter 207
message categories for DBRMLIB/DB2 Catalog
  Compare 151
message Help, accessing 44
message, invalid command 106, 166
Migrate Access Path Statistics
inserting additional translation sets or objects 180
components 170
overview 21, 169
steps in process 170
temporary tables 170
wildcard characters 180
mismatch
DBRM/Load Compare 149
host variable 133
Mismatch Analysis
parameters 209
categories 133
performing on packages 135
MSGLEVEL parameter 202

NAME parameter 202
Compare parameters 207
Impact Analysis parameters 211
Mismatch Analysis parameters 209
NAME2 parameter 207
NEWOBJ parameter 202
DBRM/Load Compare parameters 213
Impact Analysis parameters 211
Mismatch Analysis parameters 209

OBJECT parameter 202
Compare parameters 207
Impact Analysis parameters 211
Mismatch Analysis parameters 209
online Explain 85
online interface 26
ONLINE parameter 202
OPEN cursor command in PSSRXSQL 41
operating environment 22
options
Additional Data Set 58
data set 56
Declarations Generator 216
Declarations Generator Additional 195

packages

comparing the Explain history 121
generating a list of to Explain 74
performing a batch Explain on 87
performing an online Explain on 85
using the DESCRIBE command to Explain 161
panels
Batch Explain Type (PSSPA116) 87
Batch Job 48
Compare Object Specification 122
Compare Object Specification List 124
Compare Options 126
Data Set Options 56
DBRM / Load Compare 149
DBRM / Load Compare Libraries 150
Declarations Generator Additional Options 196
Declarations Generator Specification 193
Explain Object Selection List (PSSPE100) 76
Explain Object Specification (PSSPA100) 76
Explain Object Specification (PSSPA110) 87, 90
Explain Options (batch) (PSSPA020) 82
Explain Options (PSSPA020) 77
Explain or Execute Parameters (PSSPA117) 90
Explain Results (PSSPE200) 91
History Selection List 125
Impact Analysis Additional Options 144
Impact Analysis Object Specification 142
Layout Data Set (PSSPLODS) 93
Message Text 44
Migrate Access Path Statistics 171
Migrate Access Path Statistics Options 176, 179
Mismatch Analysis Additional Options 139
Mismatch Analysis Object Selection List 138
Mismatch Analysis Object Specification 136
SQL Host Variables List 108
SQLX Parameters 106
parameters
DBRM/Load Compare 213, 216
Explain and Compare 206
Migrate Access Path Statistics 213
Compare 206
DBRM/Load Compare 213, 216
Explain 199
Impact Analysis 210
Mismatch Analysis 209
PLAN parameter 202
Compare parameters 207
Impact Analysis parameters 211
Mismatch Analysis parameters 210
PLANTBL parameter 203
PLNAME parameter 203
Compare parameters 207
Impact Analysis parameters 211
product changes 12
product definition 17
PSSD2CJB member
guidelines for editing 153
tokens 153
PSSMSG command 44
PSSPA110 panel 90
PSSPA116 panel 87
PSSPA117 panel 90
PSSPA130 panel 122
PSSPC020 panel 126
PSSPD010 panel 149
PSSPD020 panel 150
PSSPDG10 panel 193
PSSPDG20 panel 196
PSSPE010 panel 106
PSSPE100 panel 76, 85
PSSPE200 panel 91
PSSPEDHV panel 109
PSSPF000 panel 27
PSSPH010 panel 136
PSSPH020 panel 139
PSSPHLST panel 125
PSSPI010 panel 142
PSSPI020 panel 144
PSSPLIHS panel 110
PSSPLIHV panel 108
PSSPLODS panel 93
PSSPM010 panel 171
PSSPM020 panel 176
PSSPODS1 panel 88
PSSPOJ00 panel 88
PSSPOLST panel 124, 138
PSSREPA layout data set
fields 95
member 73
PSSRXSQL external function 41
PTOWNR parameter 203
publications, related 11

Q

QQ command 106, 107, 166

R

RATE parameter 203, 208
RATESU parameter 203
READONLY parameter 203
RECALL parameter 203, 212
related publications 11
replicating translation sets or objects 180
requirements, software 22
results of Explain 91
RETAIN parameter 204
return codes
  DBRMLIB/DB2 Catalog Compare 224
  Declarations Generator 225
  Explain and Compare 221
  Migrate Access Path Statistics 225
  Mismatch Analysis 223
  PSSRXSQL 42
REXX exec 38
ROLLBACK command in PSSRXSQL 41
RPTHDR parameter 204
  Compare parameters 208
  Impact Analysis parameters 212
  Mismatch Analysis parameters 210
RPTOPT parameter 212
rules 34
  APPLDEV 34
  DEFAULT 34
ingoing editing 39
  help for 44
  predefined 34
rules engine 38
RULES parameter 204
ruleset 38

S

sample proc to add analysis to job step 227
search argument, specifying 44
security requirements 22
SHOWSQL parameter 208
specifying batch Explain options 82
specifying Explain options 76
SQL code -805 147
SQL code +100 178, 215
SQL Explorer (for DB2)
  commands to access SQL 158
  integrating with CATALOG MANAGER 157
  SQLX edit macro 165
SQL Performance (for DB2)
  solution 23, 24
  main menu 27
SQL Table, explaining a statement with SQL Explorer 164
SQLTEXT parameter 204
SQLTYPE parameter 212
SQLX edit macro
  defined 105
  marking text 106
SSID parameter 204
  Compare parameters 208
  Impact Analysis parameters 212
  Mismatch Analysis parameters 210
statement area, Explain Results 69
static Explain 65
STATIC parameter 204
static SQL 161, 164
STMTNO parameter 204, 210
subsystems, switching 26
summary of changes 12
SUMRPT parameter 205
SUPPRESSDET parameter 205
SYSIN 199
System Performance (for DB2)
  main menu 27

T

task bar in Explain 70
tasks
  accessing message Help 44
  analyzing objects for impact 142
  comparing Explain history for two different objects 127
  comparing objects 122
  editing rules 39
  explaining a package online 85
  explaining an ad hoc SQL statement 89
  preventing timestamp mismatches prior to production 148
  running the migrate access path statistics extract job 183
  running the migrate access path statistics update job 186
  specifying batch Explain options 82
  specifying Explain options 76
  specifying migrate access path statistics options 171
  using the Declarations Generator function 193
  using the SQLX edit macro 106
  working with Explain results 91

TBNAME parameter 212
TBQUAL parameter 205
temporary tables in Migrate Access Path Statistics 170
TITLE parameter 205
  Compare parameters 208
  Impact Analysis parameters 212
  Mismatch Analysis parameters 210
transformation patterns, specifying 179
TXTLEN parameter 208, 210

U

UP command 28
using Explain to diagnose performance problems 64
using the Explain task bar 70

V

VERSION parameter 205
  Compare parameters 208
  Impact Analysis parameters 213
  Mismatch Analysis parameters 210
VERSION2 parameter 208

W

What-If Edit SQL Explain 65
What-If Index Explain 65
wildcard characters
  DBRM/Load Compare 149
  DBRMLIB/DB2 Catalog Compare 154
working with Explain results 91
Workload Access Path Compare component 119