IBM i
Version 7.2

Database
Database Administration

IBM

Note

Before using this information and the product it supports, read the information in "Notices" on page 25.

This edition applies to IBM i 7.2 (product number 5770-SS1) and to all subsequent releases and modifications until otherwise indicated in new editions. This version does not run on all reduced instruction set computer (RISC) models nor does it run on CISC models.

Contents

Administration... 1

What's new for IBM i 7.2..1

PDF file for Database administration...1

Database administration..2

Accessing data through client interfaces...2

Altering and managing database objects...3

Creating database objects...3

Ensuring data integrity.. 4

Importing and exporting data between systems..4

Working with multiple databases..4

Working with triggers and constraints.. 4

Writing DB2 programs.. 5

Database backup and recovery.. 5

Distributed database administration...5

Queries and reports... 5

Security..6

Authority Options for SQL Analysis and Tuning 6

Row and column access control (RCAC).. 11

Overview...12

Permissions and masks..13

SQL statements 14

Secure functions...14

Secure triggers.. 14

Administrative authority...14

Best practices when using permissions and masks..15

Classic Query Engine (CQE) and SQL Query Engine (SQE)..24

Native and open query differences...24

Result set ordering... 24

Notices..25

Programming interface information.. 26

Trademarks... 26

Terms and conditions...27

Database administration

Db2® for i provides database administration, backup and recovery, query, and security functions.

You can also explore other database information using the main navigation tree or Database information finder.

What's new for IBM i 7.2

Read about new or significantly changed information for the Database administration topic collection.

IBM Advanced Data Security for i

IBM® Advanced Data Security for i introduces row and column access control (RCAC), as a data-centric security alternative.

Native and open query differences

This section explains the differences when a file with RCAC is opened by native compared to open query file.

How to see what's new or changed

To help you see where technical changes have been made, the information center uses:

- The » image to mark where new or changed information begins.
- The « image to mark where new or changed information ends.

In PDF files, you might see revision bars (|) in the left margin of new and changed information.

To find other information about what's new or changed this release, see the Memo to users.

PDF file for Database administration

You can view and print a PDF file of this information.

To view or download the PDF version of this document, select Database administration.

Saving PDF files

To save a PDF on your workstation for viewing or printing:

1. Right-click the PDF link in your browser.
2. Click the option that saves the PDF locally.
3. Navigate to the directory in which you want to save the PDF.
4. Click **Save**.

Downloading Adobe Reader

You need Adobe Reader installed on your system to view or print these PDFs. You can download a free copy from the Adobe Web site (www.adobe.com/products/acrobat/readstep.html).

Database administration

Db2 for i provides various methods for setting up and managing databases.

Related concepts

Journal management

Accessing data through client interfaces

You can access Db2 for i data through client interfaces on the server, such as the Java Database Connectivity (JDBC) driver, the Open Database Connectivity (ODBC) driver, IBM i Portable Application Solutions Environment (IBM i PASE), OLE DB Provider, .Net Provider, Net.Data®, or Distributed Relational Database Architecture™ (DRDA).

Accessing data with Java

You can access Db2 for i data in your Java™ programs by using the Java Database Connectivity (JDBC) driver that is included with the IBM Developer Kit for Java licensed program.

The driver allows you to perform the following tasks:

- Access database files.
- Access JDBC database functions with embedded Structured Query Language (SQL) for Java.
- Run SQL statements and process results.

Related concepts

Accessing your System i5 database with the IBM Developer Kit for Java JDBC driver

Accessing data with Domino

You can use IBM Lotus® Domino® for i5/OS to integrate data from Db2 for i databases and Domino databases in both directions.

To take advantage of this integration, you need to understand and manage how authorizations work between the two types of databases.

Related concepts

Lotus Domino for i5/OS

Accessing data with ODBC

You use the IBM i Access for Windows Open Database Connectivity (ODBC) driver to enable your ODBC client applications to effectively share data with each other and with the server.

Related concepts

ODBC administration

Accessing data with IBM i PASE

IBM i Portable Application Solutions Environment (IBM i PASE) is an integrated runtime environment for AIX®, UNIX, or other applications that are running on the IBM i operating system. IBM i PASE supports the Db2 for i call level interface (CLI).

Related concepts

Database

Accessing data with IBM i Access for Windows OLE DB Provider

The IBM i Access for Windows OLE DB Provider, along with the Programmer's Toolkit, facilitates the IBM i client/server application development from the Microsoft Windows client PC.

The IBM i Access for Windows OLE DB Provider gives programmers record-level access interfaces to Db2 for i database files. In addition, it provides support for SQL, data queues, programs, and commands.

Related reference

System i Access for Windows OLE DB Provider

Accessing data with IBM i Access for Windows .Net Provider

The IBM i Access for Windows .Net Provider access.

The IBM i Access for Windows .Net Provider allows access to DB2 for IBM i through the Microsoft ADO.NET interface.

Accessing data with Net.Data

Net.Data is an application that runs on a server. You can use Net.Data to easily create dynamic Web documents that are called Web macros. Web macros that are created for Net.Data have the simplicity of HTML with the functionality of CGI-BIN applications.

Net.Data makes it easy to add live data to static Web pages. Live data includes information that is stored in databases, files, applications, and system services.

Related concepts

Net.Data applications for the HTTP Server

Accessing data through a Linux partition

IBM and a variety of Linux distributors have cooperated to integrate the Linux operating system with the reliability of the IBM i architecture.

Linux brings a new generation of Web-based applications to the IBM i product. IBM has changed the Linux PowerPC® kernel to run in a secondary logical partition and contributed the kernel back to the Linux community.

Accessing data using Distributed Relational Database Architecture (DRDA)

A *distributed relational database* consists of a set of SQL objects that are spread across interconnected computer systems. Each relational database has a relational database manager to manage the tables in its environment.

The database managers communicate and cooperate with each other in a way that allows a given database manager access to run SQL statements on a relational database on another system.

Related reference

Distributed relational database function and SQL

Altering and managing database objects

Db2 for i provides both Structured Query Language (SQL) and system methods for altering and managing database objects.

Several methods are available for working with database objects. You can use the IBM Navigator for i interface, SQL statements, or IBM i commands.

Related concepts

System i Navigator database tasks

Related reference

Terminology: SQL versus traditional file access

Creating database objects

The first step in developing your database is to create the objects that hold your data. You can create tables, views, and indexes with SQL. You can also create physical and logical files using the traditional system interface.

You can create database objects using IBM Navigator for i, SQL, or the traditional system interface.

Related concepts

System i Navigator database tasks

Related reference

Terminology: SQL versus traditional file access

Ensuring data integrity

Db2 for i provides several integrity measures, such as constraints, trigger programs, and commitment control.

Constraints, triggers, and commitment control can protect your database against inadvertent insertions, deletions, and updates. Constraints basically govern how data values can change, while triggers are automatic actions that start, or *trigger*, an event, such as an update of a specific table.

Related concepts

Commitment control

Working with triggers and constraints
You can use triggers or constraints to manage data in your database tables.

Importing and exporting data between systems

Importing data is the process of retrieving data from external sources, while *exporting data* is the process of extracting data from Db2 for i and copying data to another system.

Importing data into Db2 for i can be a one-time event or it can be an ongoing task, like weekly updates for business reporting purposes. These types of data move operations are typically accomplished through import, export, or load functions.

Related concepts

Copying a file

Copying files

Copying source file data

Moving a file

Related tasks

Importing and exporting data

Loading and unloading data from systems other than System i

Working with multiple databases

The system provides a system database (identified as *SYSBAS*) and the ability to work with one or more user databases.

User databases are implemented through the use of independent disk pools, which are set up in the disk management function of System i® Navigator. After an independent disk pool is set up, it appears as another database in the Databases folder of System i Navigator.

When you expand a system in System i Navigator and then expand **Databases**, a list of databases that you can work with is shown. To establish a connection to a database, expand the database that you want to work with.

Related concepts

Disk management

Working with triggers and constraints

You can use triggers or constraints to manage data in your database tables.

A *trigger* is a type of program that is automatically called whenever a specified action is performed on a specific table. Triggers are useful for keeping audit trails, detecting exceptional conditions, maintaining relationships in the database, and running applications and operations that coincide with the change operation.

A *constraint* is a restriction or limitation that you place on your database. Constraints are implemented at the table level. You can use constraints to create referential integrity in your database.

You can work with triggers and constraints using IBM Navigator for i, SQL, or the traditional system interface.

Related concepts

System i Navigator database tasks

Writing DB2 programs

Db2 for i provides various methods for writing applications that access or update data.

You can write embedded SQL programs, external functions, external procedures, Db2 for i CLI applications, and trigger programs.

Related concepts

Embedded SQL programming

Writing a DB2 for i5/OS CLI application

Related tasks

Creating trigger programs

Related reference

Defining an external procedure

Writing UDFs as external functions

Database backup and recovery

Saving your data can be time-consuming and requires discipline. However, it is crucial that you back up your data because you never know when you might need to recover it.

Related concepts

Backup and recovery

Journal management

Recovering and restoring your database

Distributed database administration

With Db2 for i, you can work with databases that are distributed across several systems.

Related concepts

Distributed database programming

Queries and reports

You can use SQL, the Open Query File (OPNQRYF) command, the Query (QQQQRY) API, Open Database Connectivity (ODBC), or the IBM Query for i licensed program to create and run queries.

One of the most common tasks that you perform with your database is to retrieve information. The system provides several methods to create and run queries and reports.

You can use an SQL statement to retrieve information. This SQL statement is called a *query*. The query searches the tables stored in your database to find the answer to the question that you posed with your SQL statement. The answer is expressed as a set of rows, which is referred to as the result set. After a query has been run, you can also create a report to display the data provided in your result set.

In addition to using SQL, you can use other functions and products to create and run queries and reports. See the following information for details.

- IBM DB2® Web Query for IBM i overview
- Query for IBM i
- Query Management Programming

- Query Manager Use

In addition, you can build SELECT, INSERT, UPDATE, and DELETE SQL statements in the SQL Assist window of System i Navigator.

Related concepts

SQL programming

Related tasks

Building SQL statements with SQL Assist

Related reference

Open Query File (OPNQRYF) command

Query (QQQQRY) API

Security

Authorizing users to data at the system and data levels allows you to control access to your database.

Securing your database requires you to establish ownership and public authority to objects and specific authority to your applications.

Related concepts

DRDA server access control exit programs

Granting file and data authority

Limiting access to specific fields in a database file

Security

Specifying public authority

Using database file capabilities to control I/O operations

Using logical files to secure data

Authority Options for SQL Analysis and Tuning

This topic describes the authority options for SQL analysis and tuning.

Db2 for i has a rich set of commands, stored procedures, APIs and tools for analysis and tuning of the performance aspects of database applications. Previously, a system security officer would need to grant *JOBCTL user special authority to enable database analysts and database administrators to use the database tools. Since *JOBCTL authority allows a user to change many system critical settings that are unrelated to database activity, it was not an easy decision for security officers to grant this authority. In some cases, it was an easy decision and *JOBCTL was not granted to database analysts, thus prohibiting the use of the full set of database tools.

Note: For more information about setting overrides for the QAQQINI file refer to the following link: QAQQINI file override support.

Now the security officer has additional capability to authorize access to database analysis tools and the SQL Plan Cache. Db2 for i which takes advantage of the function usage capability available in the operating system. A new function usage group called QIBM_DB has been created with function IDs in the QIBM_DB group:

1. QIBM_DB_SQLADM (Database Administrator tasks)
2. QIBM_DB_SYSMON (Database Information tasks)
3. QIBM_DB_DDMDRDA (DDM & DRDA Application Server Access)
4. QIBM_DB_ZDA (Toolbox Application Server Access)
5. QIBM_DB_SECADM (Database Security Administrator)

The security officer now has flexibility to grant authorities by either; granting *JOBCTL special authority or authorizing a user or group to the IBM i Database Administrator Function through Application

Administration in System i Navigator of IBM Navigator for i. The Change Function Usage (CHGFCNUSG) command, with a function ID of QIBM_DB_SQLADM, can also be used to change the list of users that are allowed to perform Database Administration operations. The function usage controls allow groups or specific users to be allowed or denied authority. The CHGFCNUSG command also provides a parameter which can be used to grant function usage authority to any user that has *ALLOBJ user special authority. (e.g. ALLOBJAUT(*USED))

The **Database Administrator** function is needed whenever a user is analyzing and viewing SQL performance data. Some of the more common functions are displaying statements from the SQL Plan Cache, analyzing SQL Performance Monitors and SQL Plan Cache Snapshots, and displaying the SQL details of a job other than your own.

The database administrator function usage is an alternative to granting *JOBCTL, but it does not replace the requirement of having the correct object authority. To enable database administrator tasks which are unrelated to performance analysis, refer to the specific task for details on the authorization requirements. For example, to allow an administrator to reorganize a table, they must have object authorities granted, which are not covered by QIBM_DB_SQLADM.

In addition to QIBM_DB_SQLADM, the Change Function Usage (CHGFCNUSG) command, with a function ID of QIBM_DB_SYSMON, can also be used to change the list of users that are allowed to perform Database Information operations.

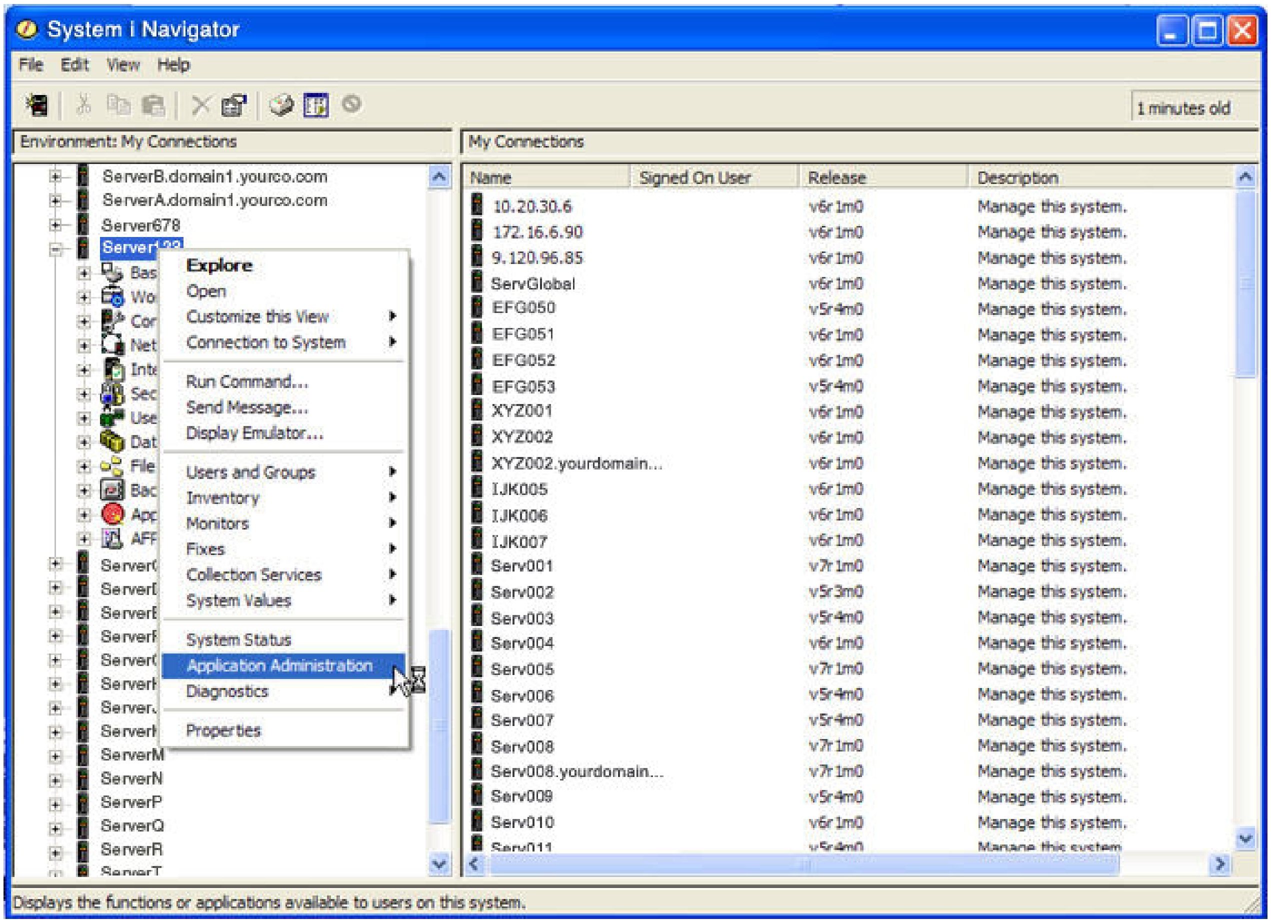
The **Database Information** function provides much less authority than Database Administrator.The primary use is to allow a user to examine high-level database properties. For example, a user that does not have *JOBCTL or QIBM_DB_SQLADM, could be allowed to view the SQL Plan Cache properties if granted authority to QIBM_DB_SYSMON.

To work with QIBM_DB database group function usage from System i Navigator, follow these steps:

1. Launch Application Administration as shown in figure 1.
2. Expand the ‘IBM i' and ‘Database' folders under the Host Applications tab as shown in figure 2.
3. Customize the Database Administrator (QIBM_DB_SQLADM) function usage as shown in figure 3.

In this example, the security officer determined that they wanted to set up a group called Dbagroup that would contain all the users that they wanted to give this level of authority. And they explicitly wanted to deny access to Slfuser. Now the security officer has one convenient and easily monitored place to view and authorize users to these functions.

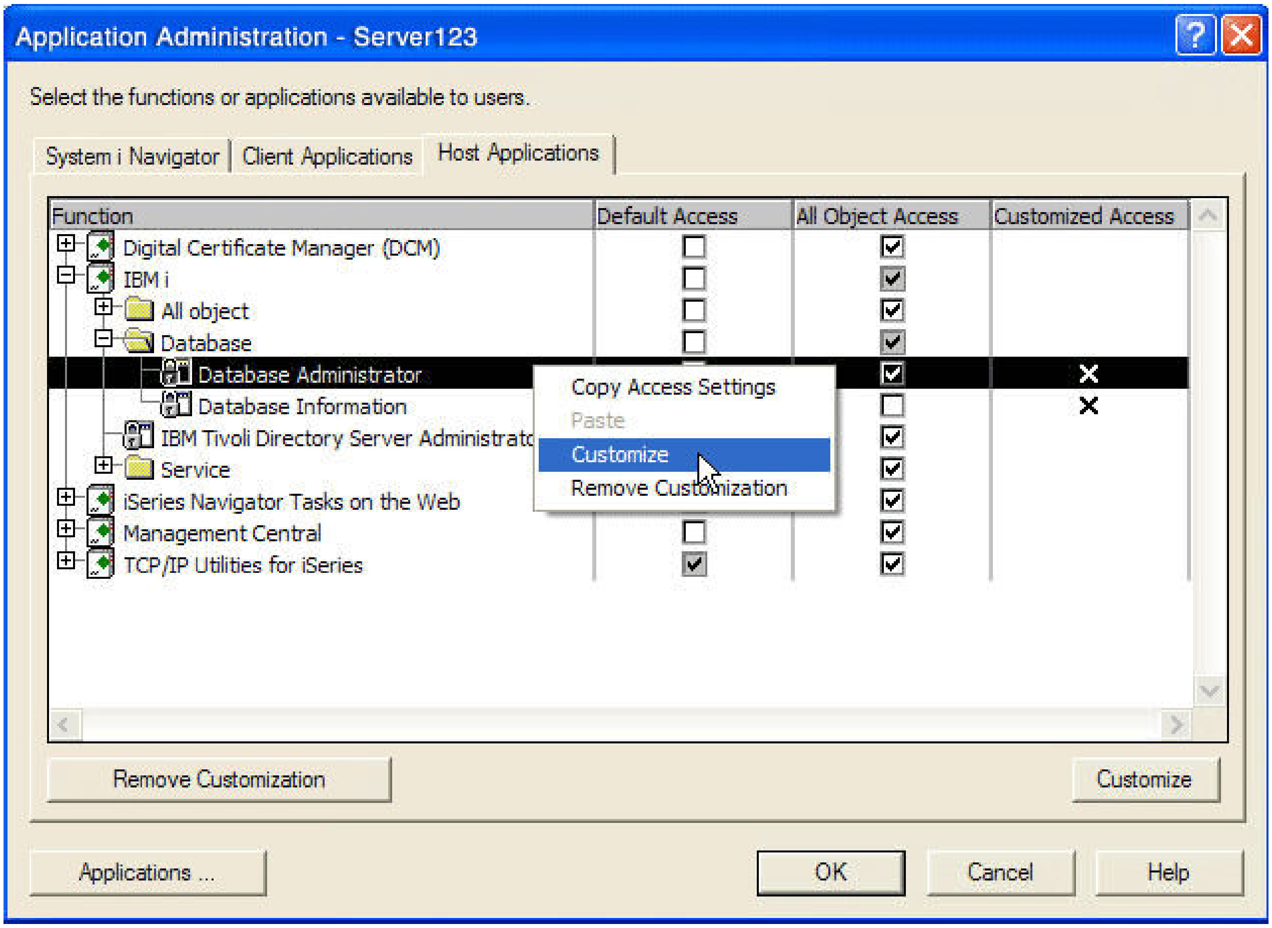
Figure 1. Launch Application Administration.



Name	Signed On User	Release	Description
10.20.30.6		v6r1m0	Manage this system.
172.16.6.90		v6r1m0	Manage this system.
9.120.96.85		v6r1m0	Manage this system.
ServGlobal		v6r1m0	Manage this system.
EFG050		v5r4m0	Manage this system.
EFG051		v6r1m0	Manage this system.
EFG052		v6r1m0	Manage this system.
EFG053		v5r4m0	Manage this system.
XYZ001		v6r1m0	Manage this system.
XYZ002		v6r1m0	Manage this system.
XYZ002.yourdomain...		v6r1m0	Manage this system.
IJK005		v6r1m0	Manage this system.
IJK006		v6r1m0	Manage this system.
IJK007		v6r1m0	Manage this system.
Serv001		v7r1m0	Manage this system.
Serv002		v5r3m0	Manage this system.
Serv003		v5r4m0	Manage this system.
Serv004		v6r1m0	Manage this system.
Serv005		v7r1m0	Manage this system.
Serv006		v5r4m0	Manage this system.
Serv007		v5r4m0	Manage this system.
Serv008		v7r1m0	Manage this system.
Serv008.yourdomain...		v7r1m0	Manage this system.
Serv009		v5r4m0	Manage this system.
Serv010		v6r1m0	Manage this system.
Serv011		v5r4m0	Manage this system

Displays the functions or applications available to users on this system.

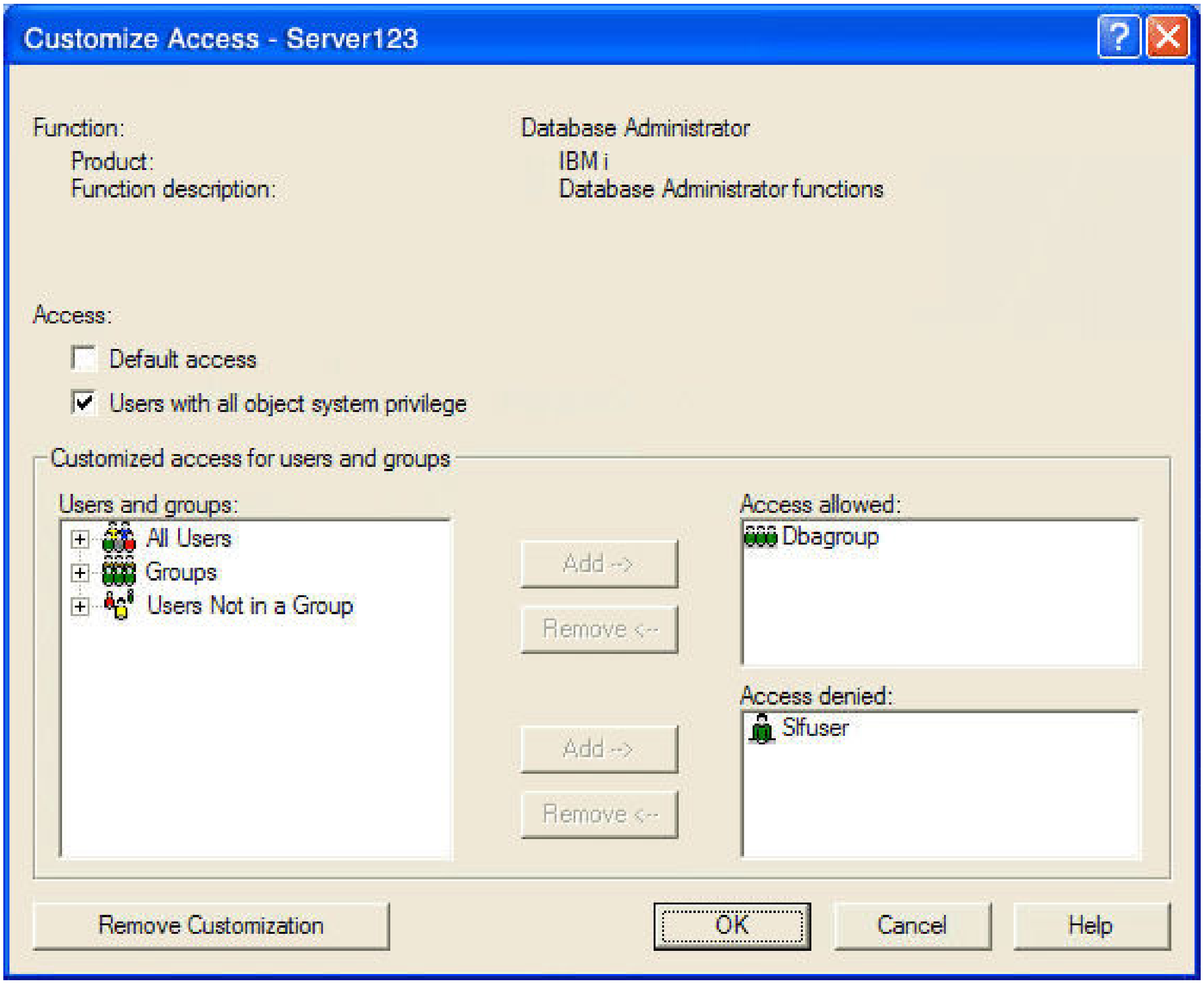
Figure 2. Expand the Database group



Select the functions or applications available to users.

Function	Default Access	All Object Access	Customized Access
Digital Certificate Manager (DCM)	☐	☑	
IBM i	☐	☑	
All object	☐	☑	
Database	☐	☑	
Database Administrator		☑	✕
Database Information		☐	✕
IBM Tivoli Directory Server Administrato		☑	
Service		☑	
iSeries Navigator Tasks on the Web		☑	
Management Central	☐	☑	
TCP/IP Utilities for iSeries	☑	☑	

Figure 3. Change the QIBM_DB_SQLADM function usage settings



Customize Access - Server123

Function: Database Administrator
Product: IBM i
Function description: Database Administrator functions

Access:
□ Default access
☑ Users with all object system privilege

Customized access for users and groups

Users and groups:
- All Users
- Groups
- Users Not in a Group

Access allowed: Dbagroup

Access denied: Slfuser

Table 1 describes some of the authorization changes related to DB2 commands, Stored Procedures, and APIs.

Table 1. Authorization requirements for Database performance and analysis

User Action	*JOBCTL	QIBM_DB_SQLADM	QIBM_DB_SYSMON	No Authority
SET CURRENT DEGREE (SQL statement)	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
CHGQRYA command targeting a different user's job	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
STRDBMON or ENDDBMON commands targeting a different user's job	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
STRDBMON or ENDDBMON commands targeting a job that matches the current user	**Allowed**	**Allowed**	**Allowed**	**Allowed**
QUSRJOBI() API format 900 or System i Navigator's SQL Details for Job	**Allowed**	**Allowed**	**Allowed**	**Not Allowed**
DUMP PLAN CACHE PROPERTIES procedure	**Allowed**	**Allowed**	**Allowed**	**Not Allowed**

Table 1. Authorization requirements for Database performance and analysis (continued)

User Action	*JOBCTL	QIBM_DB_SQLADM	QIBM_DB_SYSMON	No Authority
Visual Explain within Run SQL Scripts	**Allowed**	**Allowed**	**Allowed**	**Allowed**
Visual Explain outside of Run SQL Scripts	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
ANALYZE PLAN CACHE procedure	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
DUMP PLAN CACHE procedure	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
MODIFY PLAN CACHE procedure	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
MODIFY PLAN CACHE PROPERTIES procedure (currently does not check authority)	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
CHANGE PLAN CACHE SIZE procedure (currently does not check authority)	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
START PLAN CACHE EVENT MONITOR procedure	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
END PLAN CACHE EVENT MONITOR procedure	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**
END ALL PLAN CACHE EVENT MONITORS procedure	**Allowed**	**Allowed**	**Not Allowed**	**Not Allowed**

Row and column access control (RCAC)

Row and column access control (RCAC) provide a data-centric alternative to achieve data security.

RCAC places access control at the table level around the data itself. SQL rules that are created on rows and columns are the basis of the implementation of this capability.

RCAC terms

- Base table - The table (physical file) the permission or mask is added to.
- Dependent object - Any object (file, schema, function, or other object) the permission or mask references.
- QIBM_DB_SECADM – The function usage identifer the user must be authorized to in order to manipulate all actions that are related to permissions and masks.
- Row and Column Access Control (RCAC) – Access control is the ability to control the access to data by using permissions and masks.
- Permission - A row permission defines a row access control rule for rows of a table.
- Mask - A column mask defines a column access control rule for a specific column in a table.
- RULETEXT – The expression to be used by the permission or mask.
- 5770-SS1 IBM Advanced Data Security for i (Option 47) – Product that needs to be ordered and installed to be able to:
  - create row permissions.
  - create column masks.
  - execute database access over objects that have active RCAC.

Overview

IBM Advanced Data Security for i introduces RCAC as an extra layer of data security.

RCAC provides access control to a table at the row level, column level, or both. RCAC can be used to complement the table privileges model. To comply with various government regulations, you might implement procedures and methods to ensure that information is adequately protected. Individuals in your organization are permitted access to only the subset of data that is required to perform their job tasks. For example, government regulations in your area might state that a doctor is authorized to view the medical records of their own patients, but not of other patients. The same regulations might also state that, unless a patient gives their consent, a healthcare provider is not permitted access to patient personal information, such as the patients home phone number. You can use RCAC to ensure that your users only have access to the data that is required for their work. For example, RCAC can filter patient information and data to include only that data, which a particular doctor is authorized to view.

Other patients do not exist as far as the doctor is concerned. Similarly, when a patient service representative queries the patient table at the same hospital, they are able to view the patient name and telephone number columns, but the medical history column is masked for them. If data is masked, a NULL or an alternate value is displayed instead of the actual medical history. RCAC has the following advantages:

1. No database user is inherently exempted from the RCAC rules. Even high-level authorities such as users with all object authority (special authority (such as *ALLOBJ)) authority are not exempt from these rules. Only users with QIBM_DB_SECADM authority can manage RCAC within a database. Therefore you can use RCAC to prevent users with all object authority from freely accessing all data in a database.
2. Table data is protected regardless of how a table is accessed. Applications, improvised query tools and report generation tools are all subject to the access control rules. The enforcement is data-centric.
3. No application changes are required to take advantage of this additional layer of data security. RCAC is established and defined in a way that is not apparent to existing applications. However RCAC represents an important shift in paradigm in the sense that it is no longer what is being asked but rather who is asking. Even though two users can execute what appears to be identical queries, when row permission predicates are added to the query, those two users might observe a different result set. This behavior is the exact intent of the solution. It means that application designers and DBAs must be conscious that queries do not see the whole picture in terms of the data in the table unless granted RCAC authorization.
4. Prior to RCAC controls for data-centric data protection, DB2 for i users would protect the data through the creation of several to many SQL views or Select-omit logical files. While this technique of relying upon a view/logical file to limit data achieves the goal, it creates several problems:
   a. Applications had to be coded to work with specialized views, instead of a common object.
   b. In large installations, the number of views which exist for this purpose quickly grows to a large number, resulting in additional object management considerations like Save/Restore.
   c. The security officer has to spend time adjusting authorizations to many objects.
   d. For select-omit logical files, DB2 for i has to spend processing cycles to keep each select-omit logical file up to date as the underlying object(s) change.

Besides achieving the benefits of innately secure data when deploying RCAC, DB2 for i customers can retire the many views which exist solely to protect data.

IBM Advanced Data Security for i

IBM Advanced Data Security for i is an installable option that is used to manage security policies by enforcing RCAC with permissions and masks.

If IBM Advanced Data Security for i , is not installed, see Installing, upgrading, or deleting IBM i/OS® and related software for information about installing extra licensed programs. To install IBM Advanced Data Security for i, use option 47 in the list of installable options for the operating system.

Tables which contain enabled RCAC permissions or masks can be restored regardless of whether the IBM Advanced Data Security for i is installed. However if the option is not installed, permissions and masks cannot be created and tables, views, or indexes cannot be accessed which contain active permissions or masks.

Separation of duties

Separation of duties helps businesses comply with industry regulations or organizational requirements and simplifies the management of authorities. Separation of duties is commonly used to prevent fraudulent activities or errors by a single person. It provides the ability for administrative functions to be divided across individuals without overlapping responsibilities, so that one user does not possess unlimited authority, such as with *ALLOBJ authority.

For example, assume that a business has assigned the duty to manage security on IBM i to Theresa. Prior to release IBM i 7.2, in order to grant privileges, Theresa had to have the same privileges Theresa was granting. Thus, in order to grant *USE privileges to the PAYROLL table, Theresa had to have *OBJMGT and *USE authority (or a higher level of authority such as *ALLOBJ). This requirement allowed Theresa to access data in the PAYROLL table even though Theresa's job description was only to manage security.

In IBM i 7.2, the function usage, QIBM_DB_SECADM, provides a user with the ability to grant authority, revoke authority, change ownership, or change primary group. This is done without giving access to the object or, in the case of a database table, to the data that is in the table or allowing other operations on the table. QIBM_DB_SECADM function usage can only be granted by a user with *SECADM special authority and can be given to a user or a group.

QIBM_DB_SECADM is also responsible for administering RCAC. RCAC restricts which rows a user is allowed to access in a table and whether a user is allowed to see information in certain columns of a table.

The best practice is that the RCAC administrator has QIBM_DB_SECADM function usage and absolutely no data privileges. The RCAC administrator can deploy and maintain the RCAC constructs and would be unable to grant themselves unauthorized access to data.

Permissions and masks

RCAC is a model in which a security administrator manages privacy and security policies.

RCAC permits all users to access the same table, as opposed to alternative views of a table. RCAC, however, restricts access to the data in the table based on individual user permissions or rules as specified by a policy that is associated with the table. There are two sets of rules. One set of rules operates on rows(permissions) and the other on columns(masks). In order to create permissions and masks the IBM Advanced Data Security for i must be installed.

Row permission

- A row permission defines a row access control rule for a specific table.
- A row access control rule is an SQL search condition that describes what set of rows a user can access.
- The definition of each row permission may reference the user or group in the search condition. If multiple row permissions are defined for a table and row access control is activated, the search condition in each row permission is connected by the logical OR operator to form the row access control search condition. This row access control search condition is applied whenever the table is accessed. It acts as a filter to the table before any other user-specified operations, such as predicates and ordering are processed. It acts like the WITH CHECK OPTION clause of a view to ensure that a row to be inserted or updated conforms to the definitions of the row permissions in an INSERT, UPDATE, or MERGE statement.

Column mask

- A column mask defines a column access control rule for a specific column in a table.
- A column access control rule is an SQL CASE expression that describes what column values a user is permitted to see and under what conditions.

- The definition of each column mask may reference the user or group in the search conditions in the CASE WHEN clause. While multiple columns in a table may have column masks, only one column mask can be created for a single column. When column access control is activated for the table, the CASE expression in the column mask definition is applied to the output column to determine the masked values that are returned to an application. The application of column masks affects the final output only. It does not impact the operations, such as predicates and ordering in an SQL statement.

RCAC can be activated for a table before or after row permissions or column masks are created for the table. If row permissions or column masks exist, activating row and column access control simply makes the permissions or masks become effective. If row permissions do not yet exist, activating row access control for a table means that Db2 for i generates a default row permission that prevents any access to the data in the table.

SQL statements

The SQL create, alter, and drop statements support the implementation of RCAC with permissions and masks.

- Create Permission
- Alter Permission
- Drop Permission
- Create Mask
- Alter Mask
- Drop Mask
- Alter Function
- Alter Trigger
- Alter Table

Authorization

The authorization ID of the SQL statement must be authorized to the Database Security Administrator function of IBM i. See Administrative authority.

Secure functions

Functions must be defined as secure before they can be called within RCAC definitions.

The SECURED attribute is required if the UDF is referenced in the definition of a row permission or column mask because the UDF will have access to data prior to the application of RCAC. The SECURED attribute is also required for a UDF that is invoked in an SQL statement when the function arguments reference columns that are activated with column access control.

Secure triggers

Triggers defined on a table with RCAC activated must be secure.

The SECURED attribute is required for a trigger when the associated table has RCAC activated or the associated view whose underlying table is activated with RCAC. If a trigger exists but is not secure, RCAC cannot be activated for the associated table.

Administrative authority

Authorization to the Database Security Administrator function of IBM i can be assigned through Application Administration in IBM Navigator for i.

The Change Function Usage Information (CHGFCNUSG) command, with a function ID of QIBM_DB_SECADM, can be used to change the list of authorized users.

Best practices when using permissions and masks

Permissions and masks can be created for a table in a number of different implementations. This section will explain some of the implementations that can be used to create permissions and masks.

Creating permissions and masks
A number of considerations need to be determined to decide the best way to create permissions or masks.

Creating permissions or masks when row or column access control is active

The job creating the permission or mask obtains an exclusive lock on the base table. If row or column access control is active, the base table is not allowed to be read until the creating of the permission or mask is complete. The applications reading the base table need to be ended until the permissions or masks are added.

Creating permissions or masks when row or column access control is not active

The job creating the permission or mask obtains an exclusive lock on the base table. However, the base table is allowed to be read until the row or column access control is activated. The applications reading the base table do not have to be ended.

Single permission with all users
Example 1: Using a single permission with all the users defined in the permission.

```
CREATE SCHEMA MY_LIB
CREATE TABLE  MY_LIB.PERMISSION_TABLE (COLUMN1 INT)
CREATE PERMISSION MY_LIB.PERM1 ON MY_LIB.PERMISSION_TABLE FOR ROWS WHERE
       VERIFY_GROUP_FOR_USER(CURRENT_USER,'USER1','USER2','USER3') = 1
       ENFORCED FOR ALL ACCESS ENABLE

ALTER TABLE MY_LIB.PERMISSION_TABLE ACTIVATE ROW ACCESS CONTROL
/***************************************************************/
/*     Sign on as USER1                                        */
/***************************************************************/
INSERT INTO MY_LIB.PERMISSION_TABLE VALUES(1)  /* Allowed.     */
```

The advantage of a single permission is the best query performance for applications. The disadvantage is adding another user, the permission has to be dropped and created to add the new user.

Single permission with a group profile
Example 2: Using a single permission with all the users defined in a group profile in the permission.

```
CREATE SCHEMA MY_LIB
CREATE TABLE MY_LIB.PERMISSION_TABLE (COLUMN1 INT)
CREATE PERMISSION MY_LIB.PERM1 ON MY_LIB.PERMISSION_TABLE
       AS P_GROUP  FOR ROWS WHERE
       VERIFY_GROUP_FOR_USER(SESSION_USER,'PERM_GROUP') = 1
         ENFORCED FOR ALL ACCESS ENABLE
ALTER TABLE MY_LIB.PERMISSION_TABLE ACTIVATE ROW ACCESS CONTROL
/********************************************************************/
/* Sign on as USER1 which is a member of the user group PERM_GROUP  */
/********************************************************************/
INSERT INTO MY_LIB.PERMISSION_TABLE  VALUES(1)  /* Allowed.         */
```

The advantage of a single permission checking a group profile means the permission does not have to change adding another user. The disadvantage for every query of the base table, the VERIFY_GROUP_FOR_USER function is checked.

Single permission with a dependent table
Example 3: Using a single permission with the users defined in a dependent table.

```
CREATE SCHEMA MY_LIB
CREATE SCHEMA RCAC_DEPENDENT
CREATE TABLE MY_LIB.PERMISSION_TABLE (COLUMN1 INT)
CREATE TABLE RCAC_DEPENDENT.USERS (USERNAME CHAR (10))
```

```
      INSERT INTO RCAC_DEPENDENT.USERS
             VALUES('USER1     '),('USER2     '),('USER3     ')
      CREATE TABLE MY_LIB.PERMISSION_TABLE (FIELD1 INT)
      CREATE PERMISSION MY_LIB.PERM1 ON MY_LIB.PERMISSION_TABLE
             FOR ROWS WHERE
             CURRENT_USER IN (SELECT USERNAME FROM RCAC_DEPENDENT.USERS)
             ENFORCED FOR ALL ACCESS ENABLE
      ALTER TABLE MY_LIB.PERMISSION_TABLE ACTIVATE ROW ACCESS CONTROL
      /***********************************************************************/
      /* Sign on as USER1                                                    */
      /***********************************************************************/
      INSERT INTO MY_LIB.PERMISSION_TABLE  VALUES(1)    /* Allowed.           */
```

The advantage of a single permission checking a dependent table is that when adding another user, the permission does not have to change. The disadvantage is the performance consideration of querying the dependent table.

Single permission with a UDF
Example 4: Using a single permission with a User Defined Function (UDF).

```
            CREATE SCHEMA RCAC_DEPENDENT
            CREATE SCHEMA MY_LIB
            CREATE TABLE MY_LIB.PERMISSION_TABLE (COLUMN1  INT)
            CREATE OR REPLACE FUNCTION RCAC_DEPENDENT.UDF_PERMISSION
                    ()
                      RETURNS CHAR(10)
                      LANGUAGE SQL
                      MODIFIES SQL DATA
                      NO EXTERNAL ACTION
                      DETERMINISTIC
                      NOT FENCED
                      SECURED
                      BEGIN
                        DECLARE ALLOWS CHAR(10);
                         IF  (CURRENT_USER = 'USER1') THEN
                            SET ALLOWS = 'ALLOWED   ';
                         ELSE
                            SET ALLOWS = 'DISALLOWED';   END IF;
                         RETURN ALLOWS;
                      END

            CREATE PERMISSION MY_LIB.PERMISSION_USER
                                ON MY_LIB.PERMISSION_TABLE
                      FOR ROWS WHERE
                       RCAC_DEPENDENT.UDF_PERMISSION()  = 'ALLOWED   '
                       ENFORCED FOR ALL ACCESS  ENABLE

             ALTER TABLE MY_LIB.PERMISSION_TABLE ACTIVATE ROW ACCESS CONTROL
```

The advantage of a single permission checking a UDF is adding another user, the permission does not have to change. The disadvantage appears when the UDF changed. During the next open of the table with the permission, verification must be done to allow the new UDF to be used with the permission. The verification causes the permission or mask to be regenerated once for the table.

Permissions for each user
Example 5: Using multiple permissions, a permission for each user.

```
        CREATE SCHEMA MY_LIB
        CREATE TABLE MY_LIB.PERMISSION_TABLE (COLUMN1 INT)

        CREATE PERMISSION MY_LIB.P1 ON MYLIB.PERMISSION_TABLE
                     FOR ROWS WHERE
                     CURRENT_USER  = 'USER1     '
                     ENFORCED FOR ALL ACCESS ENABLE

        CREATE PERMISSION MY_LIB.P2 ON MY_LIB.PERMISSION_TABLE
                     FOR ROWS WHERE
                     CURRENT_USER  = 'USER2     '
                     ENFORCED FOR ALL ACCESS ENABLE
```

```
CREATE PERMISSION MY_LIB.P3 ON MY_LIB.PERMISSION_TABLE
             FOR ROWS WHERE
             CURRENT_USER  = 'USER3      '
             ENFORCED FOR ALL ACCESS ENABLE

ALTER TABLE MY_LIB.PERMISSION_TABLE ACTIVATE ROW ACCESS CONTROL
```

The advantage of multiple permissions is the ease of use of having individual permissions. The disadvantage is having to add another user, a new permission has to be added. The new permission causes a regeneration of the composite permission used for the table.

Attributes of multiple permissions

The attributes for each permission of the base table need to be the same.

The attributes need to be the same because when the permission is executed, the data (rows of the base table) is checked for each permission. For example, take the case where one permission is using a *PERIOD as the decimal point and another permission is using a *COMMA. The permissions are different because the type of decimal point that is expected by each permission is not the same. The following attributes can change the execution of the permission:

- DATFMT, TIMFMT, DATSEP, TIMSEP DECMPT
- SRTSEQ and LANGID
- DECFLTRND
- Decimal point and DECRESULT

If the attributes listed are not the same for each permission, an unexpected result may be returned.

Unqualified object names

Unqualified object names in the RULETEXT become schema qualified during the creation of the permissions or masks.

For example, creating permissions or masks in a test environment cause the object names to become qualified with the test schema name. Therefore, it is best to qualify the schema name to avoid confusion of the schema name.

```
CREATE SCHEMA  MY_LIB
CREATE SCHEMA RCAC_LIB
CREATE TABLE MY_LIB.PERMISSION_TABLE (COLUMN1 INT)
CREATE TABLE RCAC_LIB.DEPENDENT_TABLE (COLUMN1 INT)
SET SCHEMA  RCAC_LIB
CREATE PERMISSION MY_LIB.PERMISSION_USE
                  ON MY_LIB.PERMISSION_TABLE FOR ROWS
                  WHERE
                     COLUMN1 IN  (SELECT COLUMN1 FROM DEPENDENT_TABLE)
                  ENFORCED FOR ALL ACCESS ENABLE

/*******************************************************************/
/* The select statement will show the RULETEXT as being qualified. */
/*******************************************************************/
SELECT CHAR(RULETEXT,200) FROM QSYS2.SYSCONTROL
                          WHERE SCHEMA = 'MY_LIB'

/*******************************************************************/
/* The RULETEXT is now qualified.                                  */
/*******************************************************************/
PERMISSION_TABLE.COLUMN1 IN
(SELECT RCAC_LIB.DEPENDENT_TABLE.COLUMN1 FROM RCAC_LIB.DEPENDENT_TABLE)
```

Dependent objects
A number of considerations must be determined to decide how to handle dependent objects of the permissions and masks.

Ownership
Dependent objects of a permission or mask should be owned by the user profile with the QIBM_DB_SECADM functional authority and no object management authority should be granted to other users.

This restricts the possibility of the dependent object being manipulated by an authorized user to change a permission or mask to allow unintended access to data.

Schema
All dependent tables or views of a permission or mask should be created in a different schema than the schema of the base table.

If the user executes a Create Duplicate Object (CRTDUPOBJ), or Restore (RSTOBJ) of the base table to a new schema, the schema names of the dependent objects are not changed. By keeping the dependent tables and views in a different schema after the CRTDUPOBJ or RSTOBJ of the base table, the newly created base table references the same dependent objects as the original base table.

If the dependent objects of the permissions and masks are in the same schema, if the user duplicates the schema, the duplicated permissions and masks reference the objects of the original schema. Therefore, when cloning a schema and the objects within, the best practice is to use the Generate SQL feature within IBM i Navigator. By de-selecting the "Schema Qualify Objects" option, the resulting SQL script will no longer contain schema qualified references within the permissions and masks. The user can precede execution of the SQL script with a SET SCHEMA statement specifying the target schema.

Schema authority
The schema that contains the dependent objects should not allow object management authority to users.

By not granting object management authority to users, the dependent objects will not be allowed to be manipulated by users.

Secured UDFs
An SQL user-defined function (UDF) used in the RULETEXT of a permission or mask must be marked as SECURE.

This same rule applies for any function that may be invoked with a masked column specified as an argument. The SECURE attribute is stored in the *PGM or *SRVPGM executable that is called when the UDF is invoked. When the *PGM/*SRVPGM for a SECURED SQL function is restored, the SECURE attribute that is associated with the function may be lost unless one of the following is true:

- The user doing the restore is authorized to the QIBM_DB_SECADM function.
- The user doing the restore has *SAVSYS special authority.
- The user named QSECOFR is doing the restore.

Old Program Model (OPM) programs cannot be used for functions (UDFs) defined in permissions or masks. This is because the system cannot verify the program during other database operations such as restore or rename.

When creating a UDTF or UDF, the default is FENCED, meaning the UDTF or UDF is executed in a secondary thread. Certain SQL special registers like CURRENT USER may not behave as expected when referenced in a FENCED UDF. Therefore, when UDTFs or UDFs are used in the RCAC text, use NOT FENCED.

ALWCPYDTA and isolation level
The expressions in the RULETEXT of the permission or mask runs with the same ALWCPYDTA and isolation level attributes when opening a base table, index, or view with an active permission or mask.

For native opens the ALWCPYDTA attribute is *NO. This prevents temporary copies of the data from being used to execute the permission or mask expressions. If the permission or mask requires a temporary copy of the data, it is recommended that the corresponding expressions be moved to a secure UDF that runs with an ALWCPYDTA attribute of *YES or *OPTIMIZE. The RULETEXT of the permission, or mask could then be changed to reference the UDF instead of the expression that needed a temporary copy of the data.

Restoring objects

Restoring a different version of a dependent object of the base table can impact the existing permissions and masks.

The process to verify the dependent objects for permissions and masks is done the first time the base table is opened and not during the restore process.

Therefore, after restoring the dependent objects for the permissions or masks, the system administrator should include in the process a simple open operation of the base table. This allows the verification to be completed and avoid verification at application run time.

It is important to ensure that the proper dependent objects of the permissions and masks are restored when restoring the base tables with the permissions and masks.

Additional operations

A number of considerations must be reviewed creating permissions or masks for a table.

Adding application profile to permissions and masks

Some existing applications might need to add the profile of the job running the application to the permissions and masks of the base table.

Some examples of these applications would be the Data Propagator, High Availability (HA) software, and similar applications. If the application profile is not added to the permissions and masks, the permissions and masks are enforced and the application may use partial rows and masked data.

Reclaim Storage

After completing a Reclaim Storage (RCLSTG), any data spaces that are orphaned and found by the reclaim storage operation are added to the QRCL library as a table.

Since these data spaces could be the result of a base table that had RCAC, the data spaces that are now tables in QRCL do not have any RCAC. After the RCLSTG completes, the system administrator needs to query the tables in QRCL and handle (copy the data and delete the table) the tables that need to be protected with RCAC.

Query Reports

This recommendation applies to query report writer functions such as Query for i or DB2 for i Query Manager.

When using a web query report writer function, it is recommended, for consistent results, that a sort is also applied to any column that is used for report break processing. With the application of column masks, the sorting is done on a column before masks are applied, but the break processing that is done by the report writer function may be done using masked values. As a result, inconsistent break groupings and different summary values may be seen when running a query report after masks are defined on the based table.

MQTs

When populating or refreshing an MQT, it does not account for any predicates or expressions from masks or permissions on dependent tables.

When the MQT is used for optimization in a query, the underlying row permissions and column masks are built into the query that uses the MQT. In order for the MQT to be used for optimization, the MQT must include any columns that are used by the masks or permissions.

In the following example, the MQT TOTALSALES cannot be used by any query that includes CreditCardNum because CustID is used by the mask for CreditCardNum but it is not in the select list from the MQT.

```
CREATE SCHEMA MY_LIB
CREATE TABLE MY_LIB.SALES(CustID INT,
                          CreditCardNum VARCHAR(12),
                          Amount DEC(6,2))

CREATE MASK MY_LIB.CCN_MASK ON SALES FOR COLUMN CreditCardNum
RETURN
CASE
   WHEN (CustID < 10) THEN CreditCardNum
   ELSE ‘b*******’ || SUBSTR(CreditCardNum, 9, 4)
END
ENABLE;

CREATE TABLE MY_LIB.TOTALSALES
AS (SELECT CreditCardNum AS SCCN, SUM(Amount) AS SSUM
 FROM SALES
 GROUP BY CreditCardNum)
DATA INITIALLY DEFERRED
REFRESH DEFERRED
MAINTAINED BY USER

SELECT CreditCardNum, Sum(Amount)
FROM MY_LIB.SALES
GROUP BY CreditCardNum
```

Group Profiles and QIBM_DB_SECADM

Authorization IDs that are authorized to the QIBM_DB_SECADM function should not be added to a group profile.

Such an authorization ID can transfer ownership or grant privileges for an object to any authorization ID other than itself. However, the authorized ID still can transfer or grant to the group of which the authorized ID is a member.

Users who have the necessary authorities to delete, move, copy, rename, or replace the *PGM/*SRVPGM objects are unable to do those operations when the *PGM/*SRVPGM object corresponds to a SECURE FUNCTION and the user is authorized to the QIBM_DB_SECADM function. A user that is allowed to use the QIBM_DB_SECADM function can use the Create SQL ILE CL commands (CRTSQLCBLI, CRTSQLCI, CRTSQLCCPPI, or CRTSQLRPGI) or any of the Create Bound Program CL commands (CRTBNDC, CRTBNDCBL, CRTBNDCL, CRTBNDCPP, CRTBNDRPG) to replace a *PGM/*SRVPGM associated with a SECURE FUNCTION.

After the object is created, the object can be copied to the QRPLOBJ library. The QRPLOBJ copy of the SECURE FUNCTION can be copied or moved to another library, but will not be allowed to be used as a SECURE FUNCTION unless the program is renamed, moved, copied, or saved/restored by a user with QIBM_DB_SECADM authority. Remember, a user without QIBM_DB_SECADM authority is allowed to delete, move, or copy the object in QRPLOBJ, but is not allowed to delete it from the library to which it was moved or copied.

Copy File (CPYF) parameters

The Copy File (CPYF) command can compare returned values from the FROMFILE TOKEY, INCCHAR and INCREL parameters.

If a mask is defined for the column that is used by any of these parameters, the mask value is returned from the FROMFILE and used by the parameter that could result in unexpected results.

OmniFind Text Search Server for DB2 for i

The OmniFind Text Search Server for DB2 for i (5733-OMF) version 1.3 or higher allows customers to create a text search index over a column of a table that is protected by RCAC.

After a text search index is created, the CONTAINS and SCORE built-in SQL functions can be used to perform full text searches over the indexed column. Customers should be aware of the following considerations when creating a text search index over a column that is protected by RCAC.

- A text search server performs the task of indexing and searching documents; the indexed data is stored outside of DB2 as stream files in the integrated file system. Because the indexed data is stored outside of DB2, users that have access to the text search server could possibly reconstruct sensitive documents from the index.
- Data is exchanged with the text search server using network protocols that are not encrypted, digital certificates are not verified.
- A text search index requires that the base table contain one or more identifying columns that are a primary key, unique index, or ROWID. The identifying column is used to identify a specific row when interacting with the text search server or an administrator; the values are stored in the staging table, and may be returned from administrative procedures. When a text search index is created over a table that is protected by RCAC, the identifying column should contain a generated value, such as a ROWID or an identity column. This allows individual rows to be identified using non-sensitive information. For more information, please refer to the OmniFind Text Search Server for DB2 for i .

Using RCAC on Multi-Formatted Logical Files

A multiple format logical file contains either more than one record format or has more than one file that is specified on the PFILE keyword (DDS) of a logical file.

In order to open a logical file where the logical file has more than one file that is specified on the PFILE keyword, the following criteria must be met:

1. Each permission or mask on the same based on physical file must have a unique correlation name.
2. Since permissions and mask names in the same library must be unique and cannot use the mask or permission name for determining a match between two tables. Instead, the match is using the correlation name. The correlation name that is used for the “same” permission or mask that is applied to multiple based on physical files must be the same for each file.
3. The RULETEXT for a matching permission or mask must be the same. In cases where no correlation name is specified on the permission or mask, the RULETEXT is normalized to use the table name as the correlation name. Therefore, the only way to force RULETEXT to be the same between two permissions and masks is to use the same explicit correlation name.
4. Each matching mask or permission between tables must be defined with the same parser options:
   - Date/time format and separator
   - SRTSEQ and LANGID
   - DECFLTRND
   - Decimal point and DECRESULT
   - CCSID of RULETEXT
5. RCAC for every based on physical file must be in the same active, or deactive state.
6. Each mask or permission must be in the same ENABLED/DISABLED state as its match on the other based on physical files.

In this example LF1 is based on PF1, PF2 and PF3. and each definition uses the correlation name PERM1 so that the SQL checking code can identify them as being equivalent.

```
     CREATE SCHEMA MY_LIB
     SET    SCHEMA MY_LIB
     CREATE TABLE  MY_LIB.PF1  (COLUMN1 INT)
     CREATE TABLE  MY_LIB.PF2  (COLUMN1 INT)
     CREATE TABLE  MY_LIB.PF3  (COLUMN1 INT)

DDS for LF1
 FMT LF .....A..........T.Name++++++.Len++TDp......Functions++++++++++++++
                        R RECORD1                  PFILE(PF1 PF2 PF3)
                          COLUMN1
                        K COLUMN1

     ADDLIBLE MY_LIB
     CRTLF FILE(MY_LIB/LF1) SRCFILE(MY_LIB/QDDSSRC)

     CREATE PERMISSION PF1_P1 ON MY_LIB.PF1 AS PERM1 FOR ROWS WHERE
```

```
       CURRENT_USER = 'USER3' ENFORCED FOR ALL ACCESS

     CREATE PERMISSION PF2_P2 ON MY_LIB.PF2 AS PERM1 FOR ROWS WHERE
       CURRENT_USER = 'USER3' ENFORCED FOR ALL ACCESS

     CREATE PERMISSION PF3_P3 ON MY_LIB.PF3 AS PERM1 FOR ROWS WHERE
       CURRENT_USER = 'USER3' ENFORCED FOR ALL ACCESS

     CREATE MASK PF1_M1 ON MY_LIB.PF1 AS MASK1
                 FOR COLUMN COLUMN1 RETURN
                 CASE WHEN COLUMN1 > 55000  THEN  0 END

     CREATE MASK PF2_M2 ON MY_LIB.PF2 AS MASK1
                 FOR COLUMN COLUMN1 RETURN
                 CASE WHEN COLUMN1 > 55000  THEN  0 END

     CREATE MASK PF3_M3 ON MY_LIB.PF3 AS MASK1
                 FOR COLUMN COLUMN1 RETURN
                 CASE WHEN COLUMN1 > 55000  THEN  0 END

     ALTER TABLE PF1 ACTIVATE ROW ACCESS CONTROL
     ALTER TABLE PF2 ACTIVATE ROW ACCESS CONTROL
     ALTER TABLE PF3 ACTIVATE ROW ACCESS CONTROL

     ALTER TABLE PF1 ACTIVATE COLUMN ACCESS CONTROL
     ALTER TABLE PF2 ACTIVATE COLUMN ACCESS CONTROL
     ALTER TABLE PF3 ACTIVATE COLUMN ACCESS CONTROL
```

Propagation of masked data

Performing an insert or update operation into a base table with active column access control, the operation may fail because the data is the masked data.

This can happen when the data to be inserted or updated contains the masked value, and the masked data was selected from a table with active column access control and the select was done in the same SQL statement. As an example, assume that both TABLE1 and TABLE2 have active column access control and for the insert, selecting from TABLE2 would return the masked data. The following statement would return an error:

```
INSERT INTO TABLE1 SELECT * FROM TABLE2
```

The statement would fail with SQ20478 – Row or column access control is not valid.

However, assume for this example, TABLE1 and TABLE2 contain two columns, NAME and SSN. For the user doing the INSERT, the mask is defined to return the string ‘XXX-XX-nnnn’ when querying TABLE2.

```
SELECT NAME, SSN INTO :name, :ssn FROM TABLE2;
INSERT INTO TABLE1 VALUES(:name, :ssn);
```

This same type of problem can also occur if the user is running a native database application. A READ from TABLE2 followed by a WRITE into TABLE1 could result in masked data that is written to the file. Or in the case of an update, even if the SSN column is not intended to change on the UPDATE, the record being updated contains the masked value for the SSN column and the SSN column changes.

Two solutions to prevent masked data are provided:

1. BEFORE trigger.
2. CHECK constraint.

Before Trigger Solution

The trigger solution checks the new data that is written into a column and conditionally sets the column to the current value, or sets it to the DEFAULT.

This is an example of a before insert/update trigger for preventing masked data:

```
     CREATE SCHEMA MY_LIB
     CREATE TABLE  MY_LIB.EMP_INFO
```

```
             (COL1_name CHAR(10) WITH DEFAULT 'DEFAULT',
              COL2_ssn  CHAR(11) WITH DEFAULT 'DEFAULT')


/*********************************************************************/
/* Create a mask to give COL2_ssn for DBMGR, but for any other user */
/* mask the column. This table will contain a trigger to ensure the */
/* column can never contain a masked value.                          */
/*********************************************************************/

CREATE MASK MASK_SSN ON MY_LIB.EMP_INFO
            FOR COLUMN COL2_ssn
            RETURN
            CASE
              WHEN VERIFY_GROUP_FOR_USER(SESSION_USER, 'DBMGR') = 1
                   THEN COL2_ssn
                   ELSE 'XXX-XX-'||SUBSTR(COL2_ssn,8,4)
            END
            ENABLE

ALTER TABLE MY_LIB.EMP_INFO ACTIVATE COLUMN ACCESS CONTROL


CREATE TRIGGER PREVENT_MASK_SSN BEFORE INSERT OR UPDATE ON MY_LIB.EMP_INFO
               REFERENCING NEW ROW AS N OLD ROW AS O
               FOR EACH ROW MODE DB2ROW
               SECURED
               WHEN(SUBSTR(N.COL2_ssn,1,7) = 'XXX-XX-')
                BEGIN
                  IF INSERTING THEN SET N.COL2_ssn = DEFAULT;
                  ELSEIF UPDATING THEN SET N.COL2_ssn = O.COL2_ssn;
                  END IF;
                END
```

Attempting an insert or update operation causes the before trigger to be executed and ensure the correct data into column COL2_ssn.

Check Constraint Solution

The check constraint-based solution provides new SQL syntax to allow the specification of an action to perform when a violation of the check constraint’s check-condition occurs instead of returning a hard error. However, if the check-condition continues to fail after the action is taken, a hard error will be returned and the SQL statement fails with the existing constraint failure, (SQLSTATE=23513, SQLCODE=-545).

A check constraint with the on-violation-clause is allowed on both the CREATE TABLE and ALTER TABLE statements.

In the following example, the mask is defined to return a value of ‘XXX-XX-nnnn’ for any query that is not done by a user profile in the ‘DBMGR’ group. The constraint checks that the column SSN does not have the masked value.

```
CREATE SCHEMA MY_LIB
SET    SCHEMA MY_LIB
CREATE TABLE  MY_LIB.EMP_INFO
             (COL1_name CHAR(10) WITH DEFAULT 'DEFAULT',
              COL2_ssn  CHAR(11) WITH DEFAULT 'DEFAULT')

CREATE MASK MASK_ssn ON  MY_LIB.EMP_INFO
              FOR  COLUMN COL2_ssn     RETURN
              CASE
                 WHEN VERIFY_GROUP_FOR_USER ( SESSION_USER , 'DBMGR' ) = 1
                 THEN COL2_ssn
                 ELSE 'XXX-XX-'||SUBSTR(COL2_ssn,8,4)
              END
              ENABLE


/* Check constraint for the update and insert.*/
ALTER TABLE MY_LIB.EMP_INFO
            ADD CONSTRAINT  MASK_ssn_preserve
            CHECK(SUBSTR(COL2_ssn,1,7)<>’XXX-XX-’)
            ON UPDATE VIOLATION PRESERVE  COL2_ssn
            ON INSERT VIOLATION SET COL2_ssn = DEFAULT
```

Classic Query Engine (CQE) and SQL Query Engine (SQE)

The section explains the native open and query processing differences between CQE and SQE.

Native and open query differences

Some files with RCAC are not allowed to be accessed.

An attempt to use the native environment to open a file with active RCAC involving any of the following is not allowed:

- A logical file with multiple formats if the open attempt is for more than one format.
- A distributed file.
- A file with read triggers.
- A program described file.
- A file or query that specifies an ICU 2.6.1 sort sequence.
- The Query (QQQQRY) API.

An attempt to use SQL to query a table with active RCAC involving any of the following is not allowed:

- A distributed file.
- A file with read triggers.
- A file or query that specifies an ICU 2.6.1 sort sequence.

Result set ordering

SQE implementation may result in a different result set ordering for WRKQRY, RUNQRY, or OPNQRYF.

When a query is performed without explicitly specifying that the results be returned in a specific order, both the SQE and CQE optimizers will choose whatever plan will perform the best. This means that both SQE and CQE may or may not return the results in a keyed file order. Since CQE has far less advanced capability than SQE, it is more likely to return the results in a keyed order and SQE is less likely to return the results in a keyed order. Hence, if a query is specified with WRKQRY, RUNQRY, or OPNQRYF and the row ordering is important, explicitly specify the key field(s) and key field ordering.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Software Interoperability Coordinator, Department YBWA
3605 Highway 52 N
Rochester, MN 55901
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_.

Programming interface information

This Database administration publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of IBM i.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks of Oracle, Inc. in the United States, other countries, or both.

Other product and service names might be trademarks of IBM or other companies.

Terms and conditions

Permissions for the use of these publications is granted subject to the following terms and conditions.

Personal Use: You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative works of these publications, or any portion thereof, without the express consent of IBM.

Commercial Use: You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM®