IBM Systems - iSeries

Database Administration

Version 5 Release 4

IBM Systems - iSeries

Database Administration

Version 5 Release 4

Note

Before using this information and the product it supports, read the information in "Notices," on page 9.

Ninth Edition (February 2006)

This edition applies to version 5, release 4, modification 0 of IBM i5/OS (product number 5722-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

The Administration topic provides the key information for DB2 Universal Database™ for iSeries™. Follow these links to find a specific topic in one of the DB2® UDB for iSeries information center resources.

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

Printable PDF

View and print this topic in PDF format.

To view or download the PDF version of this document, select Administration (about 219 KB).

Saving PDF files

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

1. Right-click the PDF in your browser (right-click the link above).
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

See several topics that can help you to manage your database.

Alter and manage database objects

The iSeries server provides both SQL and system methods for altering and managing the database objects you create. These links show you how to work with library, table, view, journal, alias, and index objects.

There are several methods available for working with database objects. You can use the iSeries Navigator interface, SQL methods, or the traditional file interface. The following table shows the available options for each task. Click the appropriate "X" for more information about performing the task.

The iSeries Navigator tasks are documented in the Online help. For more information about accessing iSeries Navigator objects and using Online help, see iSeries Navigator database tasks.

Task	iSeries Navigator	SQL	Traditional file interface
Display table (file) attributes (catalog)	X	X	X
Drop database objects	X	X	X
Change schema (library) list	X		X
Add a column to a table	X	X	X

Task	iSeries Navigator	SQL	Traditional file interface
Add or alter an identity column	X	X	
Change a table (file) definition	X	X	X
Copy a table (file)	X		X
Copy column definitions	X		
Display contents of tables (files) and views	X	X	X
Display locked rows (records)	X		X
Edit table (file) data	X	X	X
Move a table (file)	X		X
Reorganize a table (file)	X		X
Altering a sequence	X	X	

Create database objects

The first step in developing your database is to create the objects that hold your data. This category covers both SQL and file-based methods for creating tables, views, and indexes (with SQL) as well as physical and logical files (with file-based methods).

The first step in developing any DB2 Universal Database database is to create the database objects that hold the actual data. For SQL-based databases, you create tables, views, and indexes; for system file databases, you create physical or logical files.

You can create objects using iSeries Navigator, SQL, or the traditional file interface. The following table shows the available options for each task. Click the appropriate "X" for more information about performing the task.

The iSeries Navigator tasks are documented in the Online help. For more information about accessing iSeries Navigator objects and using Online help, see iSeries Navigator database tasks.

Task	iSeries Navigator	SQL	Traditional file interface
Create a map of database object relationships	X		
Create a package	X	X	X
Create a schema (library)	X	X	
Create a stored procedure	X	X	
Create a table (file)	X	X	X
Create a user-defined function	X	X	
Create a user-defined type	X	X	
Create a view or logical file	X	X	X
Create an alias	X	X	X
Create an index or access path	X	X	X
Create an object using Run SQL Scripts	X		

Task	iSeries Navigator	SQL	Traditional file interface
Generate SQL for database objects	X		
Create a sequence	X	X	

Ensure data integrity

iSeries provides several different integrity measures, including constraints, trigger programs, and commitment control.

Protecting your database against inadvertent insertions, deletions, and updates is accomplished using commitment control, constraints, and triggers. 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.

See the related topics to learn about different integrity measures.

Related concepts

Commitment control

"Work with triggers and constraints" on page 4
Triggers and constraints can be used to manipulate and manage data in your database tables. These links describe different methods for adding and using triggers and constraints.

Import and export 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 Universal Database for iSeries and copying the data to another server.

Importing data into DB2 UDB for iSeries can be a one-time event or it can be an ongoing task, like weekly updates for business reporting purposes. These types of data moves are typically accomplished through import, export, or load functions.

See the related topics to learn loading and unloading data between systems.

Related concepts

Copy a file

Copy files

Copy source file data

Move a file

Query Management Programming PDF

Related tasks

Load and unload data from non-iSeries systems

Work with multiple databases

Set up multiple independent user databases on your iSeries. This topic describes how to use independent disk pools to configure multiple databases.

With iSeries servers, you can work with multiple databases. The iSeries server provides a system database (often referred to as *SYSBAS*) and the ability to work with one or more user databases. User databases are implemented on the iSeries server through the use of independent disk pools, which are set up in the Disk Management function of iSeries Navigator. After an independent disk pool is set up, it appears as another database under the Databases function of iSeries Navigator.

When you expand an iSeries server in iSeries Navigator and then expand Databases, a list of databases appears that you can work with. To establish a connection to a database, expand the database that you want to work with.

Related concepts

Disk management

Work with triggers and constraints

Triggers and constraints can be used to manipulate and manage data in your database tables. These links describe different methods for adding and using triggers and constraints.

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

Constraints are restrictions or limitations 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 iSeries Navigator, SQL, or the traditional file interface. The following table shows the available options for each task. Click the appropriate "X" for more information about performing the task.

The iSeries Navigator tasks are also documented in the Online help. For more information about accessing iSeries Navigator objects and using Online help, see iSeries Navigator database tasks.

Task	iSeries Navigator	SQL	Traditional file interface
Write an external trigger program	X		X
Add an external trigger	X		X
Add an SQL trigger	X	X	
Write an SQL trigger program	X	X	
Enable and disable a trigger	X		X
Remove a trigger	X	X	X
Create a key constraint	X	X	X
Remove a key constraint	X	X	X
Create a referential constraint	X	X	X
Enable and disable a referential constraint	X		X
Remove a referential constraint	X	X	X
Create a check constraint	X	X	X
Remove a check constraint	X	X	X
Manage check pending constraints	X		

Write DB2 programs

DB2 Universal Database for iSeries provides various methods for writing applications that access or update iSeries data.

See the Related topics to learn about these methods.

Related concepts

Write a program that uses embedded SQL

Write an external function program

Write an external trigger

Write an external procedure program

Write a DB2 UDB CLI application

Database backup and recovery

Backing up your data regularly is an important practice in the event that data recovery is necessary. See this topic for methods of 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 do a server recovery. Use the topics listed here to properly back up your data and recover it if necessary.

Distributed database administration

Learn how to set up and maintain databases distributed across several systems.

Queries and reports

Create and run queries using various methods, including SQL, the OPNQRYF command, the i5/OS™ Create Query API (QQQQRY), ODBC, and Query for iSeries.

One of the most common tasks that you perform with your database is to retrieve information. iSeries servers provide 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. For detailed information about creating SQL statements, see the SQL programming topic.

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

- Query for iSeries
- Query Management Programming
- iSeries Query Manager

In addition, the iSeries Navigator interface provides SQL Assist, which can be used to create SQL statements. SQL Assist helps you to build SQL Select, Insert, Update, and Delete statements. See Build SQL statements with SQL Assist for more information.

Related concepts

Build SQL statements with SQL Assist

SQL programming

Additional information

This topic contains additional information for query for iSeries.

Changes to Query for iSeries

In V5R3, the Query for iSeries manual was not updated, but numerous functional updates were made to the product. These updates are summarized here:

- Support for BINARY, VARBINARY, BLOB, CLOB, DBCLOB, and ROWID data types.
- Support for larger numeric and decimal numbers, plus support for large numeric literals.
- The BINARY, VARBINARY, HEX, and LENGTH built-in functions were added. They are described in this topic.
- The VARCHAR function now supports conversion from CLOB to VARCHAR.
- The VARGRAPHIC function now supports conversion from CLOB to DBCS graphic, CLOB to UCS2 graphic, DBCLOB to DBCS graphic, and DBCLOB to UCS2 graphic.

A binary constant is used for comparing a literal with a binary field (BINARY, VARBINARY, or BLOB). Binary constants are represented with an X followed by a sequence of characters that starts and ends with a string delimiter. The characters between the string delimiters must be an even number of hexadecimal digits. A hexadecimal digit is a digit or any of the letters A through F (uppercase and lowercase), as shown in the following example:

```
Field     Test  Value
binarycol eq    X'12AF'
```

BINARY built-in function for Query (added):

The BINARY function returns a BINARY representation of a string of any type. The form is:

```
►►──BINARY──(──string-expression──┬────────────┬──)───────────────────────────►◄
                                  └─, integer──┘
```

The result of the function is a fixed-length binary string. If the first argument can be null, the result can be null; if the first argument is null, the result is the null value.

The first argument must be a string-expression whose value must be a built-in character string, graphic string, binary string, or row ID data type.

The second argument specifies the length attribute for the resulting binary string. The value must be between 1 and 32766. If the second argument is not specified, the following rules apply:

- If the string-expression is the empty string constant, the length attribute of the result is 1.
- Otherwise, the length attribute of the result is the same as the length attribute of the first argument, unless the argument is a graphic string. In this case, the length attribute of the result is twice the length attribute of the argument.

The actual length is the same as the length attribute of the result. If the length of the string-expression is less than the length of the result, the result is padded with hexadecimal zeros up to the length of the result. If the length of the string-expression is greater than the length attribute of the result, truncation is performed.

VARBINARY built-in function for Query (added):

The VARBINARY function returns a VARBINARY representation of a string of any type. The form is:

```
►►──VARBINARY──(──string-expression──┬────────────┬──)────────────────────────►◄
                                     └─, integer──┘
```

The result of the function is VARBINARY. If the first argument can be null, the result can be null; if the first argument is null, the result is the null value.

The first argument is a string-expression whose value can be a character string, graphic string, binary string, or row ID.

The second argument specifies the length attribute for the resulting binary string. The value must be between 1 and 32740 (32739 if nullable). If the second argument is not specified, the following rules apply:

- If the string-expression is the empty string constant, the length attribute of the result is 1.
- Otherwise, the length attribute of the result is the same as the length attribute of the first argument, unless the argument is a graphic string. In this case, the length attribute of the result is twice the length attribute of the argument.

The actual length is the same as the length attribute of the result. If the length of the string-expression is less than the length of the result, the result is padded with hexadecimal zeros up to the length of the result. If the length of the string-expression is greater than the length attribute of the result, truncation is performed.

HEX built-in function for Query (added):

The HEX function returns a hexadecimal representation of a value. The form is:

```
►►──HEX──(──expression──)──────────────────────────────────────────────────────►◄
```

The argument can be of any built-in data type. The result of the function is a character string. If the argument can be null, the result can be null; if the argument is null, the result is the null value.

The result is a string of hexadecimal digits. The first two digits represent the first byte of the argument, the next two digits represent the second byte of the argument, and so forth. If the argument is a datetime value, the result is the hexadecimal representation of the internal form of the argument.

The length attribute of the result is twice the storage length attribute of the argument. The length attribute of the result cannot be greater than 32766 for fixed-length results or greater than 32740 for varying-length results. If the argument is a varying-length string, the result is a varying-length string. Otherwise, the result is a fixed-length string.

The CCSID of the string is the default SBCS CCSID at the current server.

LENGTH built-in function for Query (added):

The LENGTH function returns the length of a value. The form is:

```
►►──LENGTH──(──expression──)───────────────────────────────────────────────────►◄
```

The argument must be an expression that returns a value of any built-in data type. The result of the function is a large integer. If the argument can be null, the result can be null; if the argument is null, the result is the null value.

The result is the length of the argument. The length of strings includes blanks. The length of a varying-length string is the actual length, not the length attribute. The length of a graphic string is the number of double-byte characters (the number of bytes divided by 2). The length of all other values is the number of bytes used to represent the value:

- 2 for small integer
- 4 for large integer
- 8 for big integer
- The integral part of `(p/2)+1` for packed decimal numbers with precision `p`
- `p` for zoned decimal numbers with precision `p`
- 4 for single-precision float
- 8 for double-precision float
- The length of the string for strings
- 3 for time
- 4 for date
- 10 for timestamp
- 26 for row ID

Security

Authorizing users to data at the system and data levels allows you to control access to your database. These links describe various security methods for controlling your database.

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

Related concepts

Security

Appendix. 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:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, 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 information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, IBM License Agreement for Machine Code, 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.

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.

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_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

This Database administration publication documents intended Programming Interfaces that allow the customer to write programs to obtain the services of IBM i5/OS.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

| DB2
| DB2 Universal Database
| eServer

| e(logo)server
| IBM
| IBM (logo)
| iSeries

Other company, product, and service names may be trademarks or service marks of others.

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.

Printed in USA