Meta5

Administering Databases with Meta5

Version 4.3

Note

Before using this information and the product it supports, be sure to read the general information under “Notices”.

Third Edition (January 2013)

This edition applies to Version 4 Release 3 Modification Level 3 of Meta5, and to any subsequent releases until otherwise indicated in new editions or technical newsletters. Make sure you are using the correct edition for the level of the product.

Order publications through your Meta5 representative or the Meta5 branch office serving your locality.

Contents

Notices

Trademarks

About This Book

This book describes database administration concepts and tasks for the Meta5 system.

Who Should Read This Book

This book is intended for the use of Meta5 database administrators. This book covers the following tasks:

- Creating, modifying, backing up, and restoring Workstation Tools Data Dictionary (WTDD) tables
- Transferring and loading data into tables
- Using the DBA tool
- Configuring data access tools for users
- Using the System Administration (SA) tool to run SQL statements and macros
- Using the SetDBAccess transformer
- Configuring and using the Metadata transformer

Prerequisites

Before attempting the tasks described in this book, you must have a working knowledge of:

- Meta5 desktop functions and capabilities, as described in *Getting Started with the Meta5 Developer's Desktop*
- Structured Query Language (SQL)

You must also be familiar with the SA, Query, Reporter, Forms Design, Data Entry, SQL Entry, and Spreadsheet tools. Detailed documentation for most of these tools can be found in the *Data Access Tools User's Guide*, *Forms User's Guide*, *System Administration Guide and Reference*, and the *Spreadsheet User's Guide*.

We'd Like Your Comments

Your feedback is important to us in providing you with the most accurate and high-quality information. If you have any comments about this book or any other Meta5 documentation, please visit our Website at:

`http://www.meta5.com/`

There you'll find a feedback page where you can enter comments and send them to us.

Chapter 1. Introducing Meta5 Database Administration

As database administrator, you are responsible for defining data, loading it into a database, and ensuring that applications that use the data perform satisfactorily. This chapter describes database-administrative tasks and tools that are unique to the Meta5 environment. This chapter contains the following topics:

- “Database Administration Tasks”
- “Data Access Tools”

Database Administration Tasks

You can perform a number of database administration tasks with Meta5, such as:

- Design a database
- Define categories, tables, columns, default joins, and default constraints in a logical database
- Create tables
- Add data to tables
- Create user accounts
- Grant table privileges to users
- Extract data and format data
- Index tables to improve performance
- Configure and test data access icons
- Enable users to access database categories by creating views and defining synonyms, depending on your relational database management system (RDBMS)

Data Access Tools

Meta5 provides several workstation tools to help you administer and access data. This book describes certain functions that these tools can perform.

These tools use the Workstation Tools Data Dictionary (WTDD) to get information about database system tables. The WTDD is described in “Chapter 2. Working with the Workstation Tools Data Dictionary,” on page 5. The DBA tool is the only data access tool that updates information about the WTDD in real time. All other data access tools make copies of the WTDD the first time you open them. When you reopen one of these data access tools, it verifies that its version of the WTDD is current.

DBA Tool

Using the DBA tool, you can define data as the user sees it by providing DBA access to the WTDD. You can also use the DBA tool to run Structured Query Language (SQL) statements. To run macros, however, you must use the SA tool. See “Chapter 4. Getting Started With The DBA Tool,” on page 35 for information on using the DBA tool.

System Administration Tool

The System Administration (SA) tool provides a graphical interface to administrative functions for several Meta5 features, including database access. You can enter or copy SQL statements in the SA tool window by using the SQL Input Service. You can also use the DBA and SQL Entry tools to run SQL statements. See “Chapter 6. Accessing Databases with the SA Tool,” on page 73 for instructions on using the SA tool to connect to a database and run macros and SQL statements. The SA tool is the only Meta5 tool which you can run Meta5 DBA macros.

Query Tool

You use the Query tool to retrieve data from a database. You can also update data or insert data into a database by using the Query tool with a Spreadsheet tool. See “Chapter 7. Transferring Data,” on page 77 for instructions on how to use the Query tool to load tables. For more information about the Query tool, see the *Data Access Tools User's Guide*.

Spreadsheet Tool

The Spreadsheet tool has no direct data access. However, you can use it with the Query tool as a template to define data types that match the data types of columns in a table that you want to modify. See “Chapter 7. Transferring Data,” on page 77 for information on using the Spreadsheet tool to load tables. For more information about the Spreadsheet tool, see the *Spreadsheet User's Guide*.

Forms Design and Data Entry Tools

Use the Forms Design and Data Entry tools to perform database maintenance.

You can also use the Forms Design tool to create data entry forms that enter, retrieve, update, and delete data in a database. Because you create Data Entry icons by using the Forms Design tool, you will not find a Data Entry icon in the *Blank Icons file drawer.

For information on using these tools, see the *Forms User's Guide*.

Browser Tool

You can use the Browser tool to retrieve, organize, and display data from database categories. With it, you can set constraints on database items, view the results, and transfer the constraints to other Meta5 tools. See the *Data Access Tools User's Guide* for more information.

Reporter Tool

The Reporter tool performs calculations on data it retrieves from a database. It then displays the data in a report format that you specify. This report can then be printed or transferred to other tools, such as the Spreadsheet tool. For more information about the Reporter tool, see the *Data Access Tools User's Guide*.

SQL Entry Tool

You can use the SQL Entry tool to run SQL statements. You can run the SQL Entry tool as a stand-alone tool and as a supplement to tools that cannot run SQL statements. Also, you can use the SQL Entry tool to run certain logic statements and to apply functions and @-variables to SQL statements. See the *Data Access Tools User's Guide* for more information.

Chapter 2. Working with the Workstation Tools Data Dictionary

The Workstation Tools Data Dictionary (WTDD) is a data dictionary, exclusive to Meta5, that provides a logical view of a database. This chapter contains:

- A description of the WTDD
- A description of logical database and physical database interaction
- A list of the tables in the WTDD
- Instructions on how to create the WTDD
- Information on various mechanics of the WTDD
- Information on how changes to the WTDD can affect data access tools

This chapter also describes other database structures that are unique to Meta5, including:

- Categories, fact tables, and dimension tables
- Key columns
- Default joins

The WTDD

A *physical database* is a set of physically stored data tables, used by a host's database management system (DBMS). A *logical database* is a user-arranged representation of a physical database. The WTDD provides a logical view of databases that is standard across Meta5-supported relational database management systems (RDBMS).

The WTDD displays information and translates depictions into SQL statements that run in the background. The RDBMS does not use WTDD information directly. Instead, as a supplement to the physical data dictionary of the RDBMS, the WTDD translates between the depictions users see and the commands that the RDBMS understands.

The WTDD stores the *print names* of database items in data access tools. A print name is the name for a database item that is used in Meta5 windows. The WTDD translates the print names in the logical database into *internal names* when accessing the physical database. Internal names are usually abbreviated due to restrictions in the physical database on the size of object names. Print names allow users to access data in an easy-to-understand format instead of through internal names. For example, you can name a physical column PRODNUM and use the WTDD to give it the logical column name Product ID.

Internal names cannot include embedded spaces. Print names are not subject to this restriction.

The WTDD also contains information about key columns, joins between tables, domains, and constraints. See:

- “Key Columns” on page 14 for information about key columns.
- “Joins” on page 15 for information about joins and domains.
- “Default Constraints” on page 17 for information about constraints.

Each Meta5 logical database has a WTDD. To maintain the WTDD, use the DBA tool. See “Chapter 4. Getting Started With The DBA Tool,” on page 35 for instructions on using the DBA tool.

Logical and Physical Database Representations in the DBA Tool

Using the DBA tool, you define how data in the logical database is presented to users. The logical definitions you create are stored in the WTDD, which Meta5 data access tools use to create queries and reports.

Based on specific information needs, the logical database groups tables into categories. Joins between tables are defined in the logical database so that the data access tool user does not have to specify the joins when querying the database. The form and requirements of the logical database are consistent among all supported host systems.

You can also use the DBA tool to create physical tables to contain the actual data. The DBA tool groups physical data tables by user. Table names and column names in the physical database tend to be somewhat cryptic abbreviations that are not meaningful to workstation users. The form and requirements of the physical database can vary from one kind of database management system to another. SQL statements are run on the physical database.

The logical and physical databases are independent of each other with respect to changes you make to either. If you delete a table from the physical database, the corresponding logical definition of the table remains. Likewise, if you delete the logical description of a table, the physical table remains.

Figure 1 on page 7 shows how the DBA tool presents logical and physical data. See “Chapter 4. Getting Started With The DBA Tool,” on page 35 for information on the features and functions of the tool.

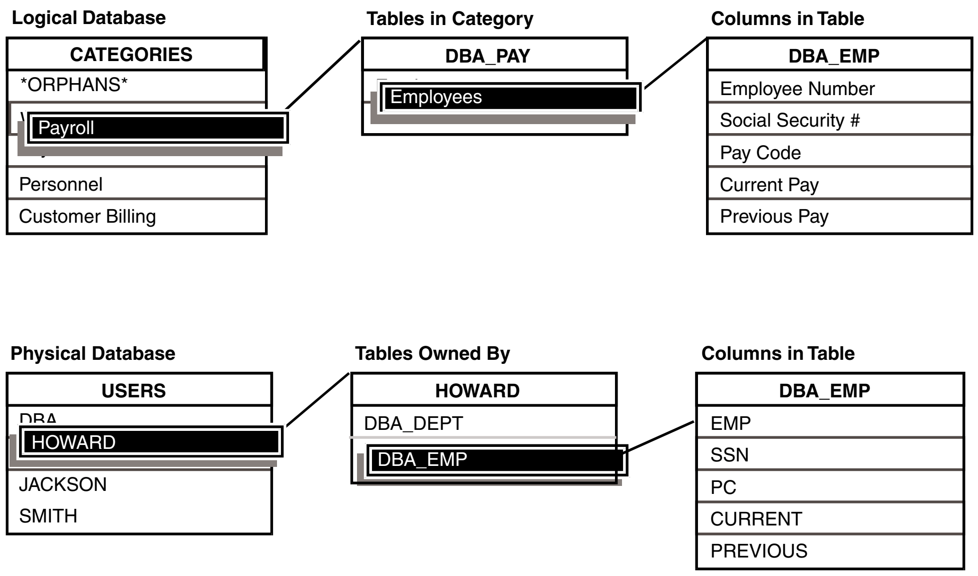


Figure 1. Logical and physical database representations

WTDD Tables

The WTDD consists of 11 physical tables that contain logical database information. WTDD tables are located in a predefined database category called Workstation Tools DD. You can view and manage the WTDD through the Workstation Tools DD category, which is displayed whenever you use the DBA tool to view the logical database. When you open a DBA tool, the Workstation Tools DD is listed in the Logical Database Categories menu. To learn how to use the DBA tool, see “Chapter 4. Getting Started With The DBA Tool,” on page 35.

Table 1 lists each of the tables in the WTDD and their contents. The tables are listed in the sequence that the DBA tool displays them.

Table 1. WTDD tables

Table name	Contents
WT_CATALOG	Database categories
WT_COLUMNS	Logical column definitions
WT_CONTENTS	Associations between tables and views in a category
WT_DCONSTR	Default constraints
WT_DJOINS	Default joins
WT_DOMAINS	Domain definitions
WT_FUNDEP	Functional dependencies between columns

Table 1. WTDD tables

Table name	Contents
WT_TABLES	Logical table definitions
WT_TTBLNAMES	Root names for temporary tables
WT_VARS	Assignments of values to database-specific control variables
WT_VERSIONS	Version numbers of each of the WTDD tables

Category Information and Column Information

Categories and columns are central to the logical database. Figure 2 shows the associations between tables that contain category information. The internal names specified by SQLNAME are used in SQL statements. The print names are displayed in the Query tool and in the logical database that the DBA tool depicts.

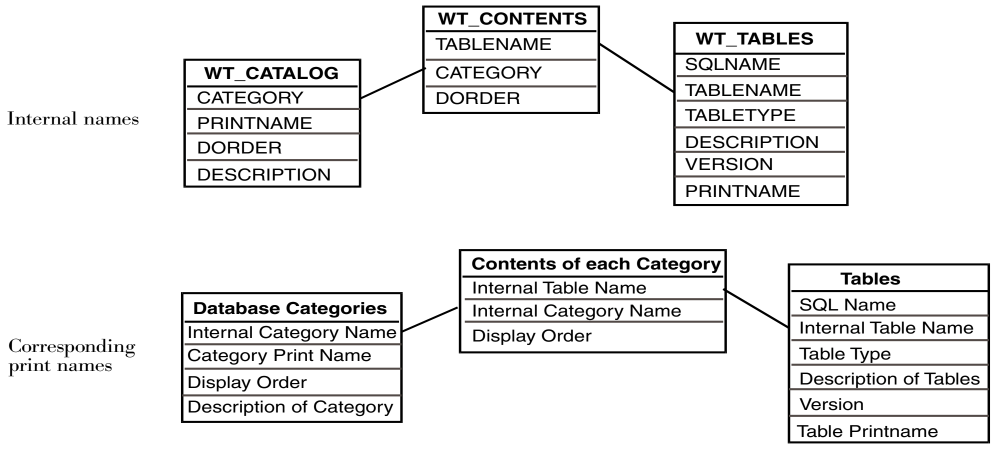


Figure 2. Associations between tables that contain category information

Figure 3 on page 9 shows the associations between tables that contain column information. The table names and column names in this figure are the internal names that you use in SQL statements.

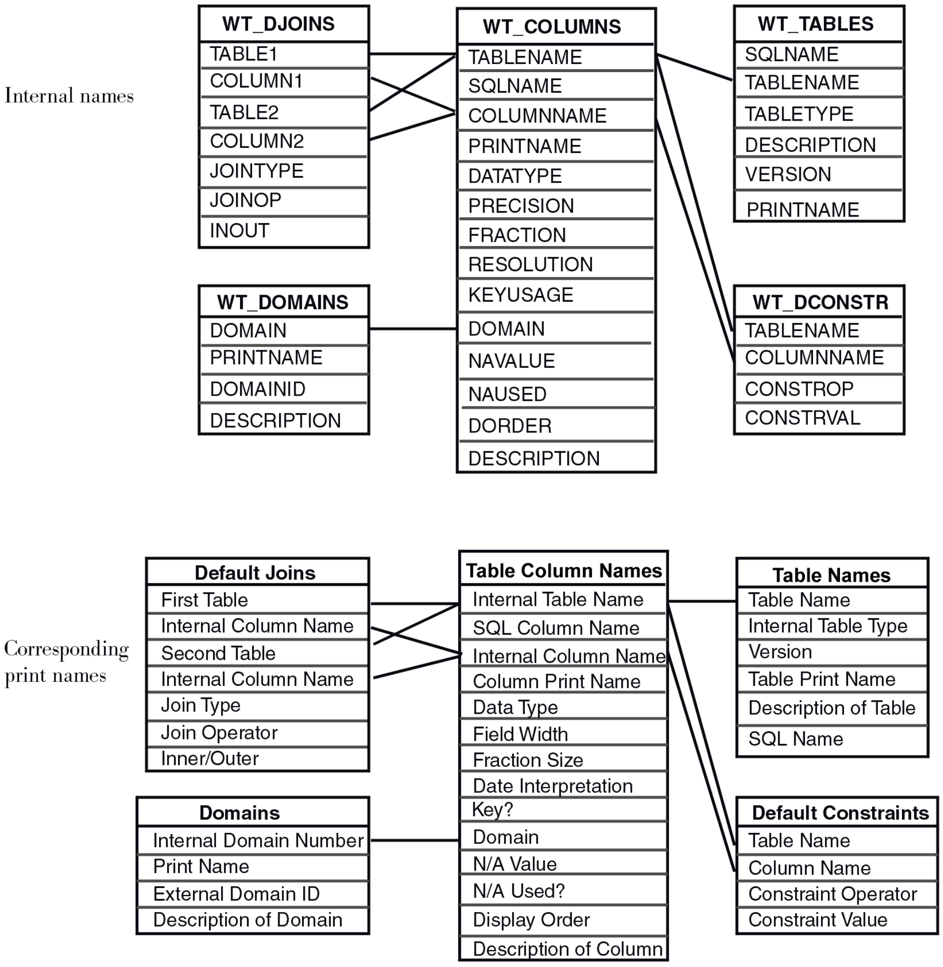


Figure 3. Associations between tables that contain column information

Creating WTDD Tables

There are three macros available in the DBA file drawer that let you create WTDD tables. The macros vary according to your database:

`createwtdd`	Used with DB2 for MVS, DB2 UDB for Windows, ODBC, DB2 for VM, DB2 UDB for AIX, and DataJoiner databases
`newdict`	Used with SYBASE**, and ORACLE** databases
`install`	Used with DB2 UDB for MVS, DB2 UDB for Windows, DB2 UDB for VM, DB2 UDB for AIX, and DataJoiner databases

Ensure that you are connected to the appropriate database before calling these macros.

The `createwtdd` and `newdict` macros also grant select privileges on the WTDD tables to PUBLIC, define categories, and create indexes on the tables. The `newdict` macro creates table descriptions.

The `createwtdd` macro calls an SQLMACRO facility supported by the Meta5 database gateways to send SQL statements to the database management system. You can enter the SQLMACRO command from any Meta5 tool that allows you to directly enter SQL statements. See “Appendix E. Using the SQLMACRO Facility,” on page 121 for more information about the SQLMACRO facility.

A third macro called `install`, used with Meta5 databases, is also located in the DBA file drawer. The `install` macro typically calls the `createwtdd` macro, as well as macros that are specific to your database system. In addition to creating WTDD tables, the `install` macro inserts entries into the WTDD to describe the system catalog and the WTDD tables. This provides users access to the system catalog, which you might not want to allow. In that case, use the `createwtdd` macro.

As an alternative to running the `createwtdd` macro, you can create WTDD tables in an MVS environment by using the METWTBLD program. The METWTBLD program creates WTDD tables on local DB2 systems and on any system that is accessible through Distributed Relational Database Architecture (DRDA). See *Administering Host Services for MVS: DB2 and Cooperative Application Services* for information on how to run the METWTBLD program.

DB2 UDB for MVS

To create a WTDD in the DB2 for MVS environment, run the `createwtdd` macro and provide the required parameters:

```
CALL createwtdd db, dsgn, isgn, wto, wtts, wtprim, wtsec, wtgrant
```

where:

db	The Meta5 database name.
dsgn	The name of the DB2 storage group for containing tables.
isgn	The name of the DB2 storage group for containing indexes.
wto	The WTDD owner ID.
wtts	The table space name for the WTDD.
wtprim	The amount of primary space to be allocated to the WTDD.
wtsec	The amount of secondary space to be allocated to the WTDD.
wtgrant	PUBLIC, or the ID to be granted access to the WTDD.

DB2 UDB for VM

To create a WTDD in the DB2 for VM environment, run the `createwtdd` macro and provide the required parameters, as follows:

```
CALL createwtdd wto, wtds, wtpages, wtgrant
```

where:

wto	The WTDD owner ID.
wtds	The name of the database space.
wtpages	The number of pages in the database space.
wtgrant	PUBLIC, or the ID to be granted access to the WTDD.

DB2 UDB for NT and DB2 UDB for AIX

To create a WTDD in the DB2 UDB for Windows NT and DB2 UDB for AIX environments, run the `createwtdd` macro and provide the required parameters:

```
CALL createwtdd wto, wtgrant
```

where:

wto	The WTDD owner ID.
wtgrant	PUBLIC, or the ID to be granted access to the WTDD.

SYBASE and ORACLE

To create a WTDD in the SYBASE and ORACLE environments, run the `newdict` macro (this macro does not require parameters):

```
CALL newdict
```

Attention:
The `newdict` macro drops all WTDD tables before it creates new ones. Never run the `newdict` macro for a database if that database's WTDD tables are already defined.

WTDD Database Elements

The WTDD defines a number of database elements that support data access tools. Specifically, these elements are used when you use data access tools to design database tables, create views of the database, apply constraints to database tables, and search for and retrieve data.

Facts, Dimensions, and Categories

In the Meta5 environment, data is based on a multidimensional model that consists of fact and dimension table associations:

- A *fact* is data that consists of measures such as unit price or volume.
- A *dimension* is data, such as periods of time, products, and locations, that defines the context of facts.
- A *category* is a group of tables that are used together. Tables contain either facts or dimensions.

Categories in the Logical Database

In the Meta5 environment, you create categories within an existing database system. Categories are strictly logical entities and do not define how or where data is stored. The ways in which tables are used in an application determine the tables' categories. For example, you might specify that a personnel application uses the tables that contain employee, department, and job-related information. You might also specify that the table that contains the employee information is used with a table containing salary information in a payroll application.

Grouping tables into categories makes it easier for users to locate the tables they want to access. Tables can also be grouped to accommodate user groups who need access to certain tables but not to the entire database. A table can belong to more than one category.

The WT_CATALOG table defines database categories. The WT_CONTENTS table stores information about which tables are associated with each category. "Chapter 4. Getting Started With The DBA Tool," on page 35 provides instructions on how to create categories.

You can use certain data access tools to retrieve different types of category information.

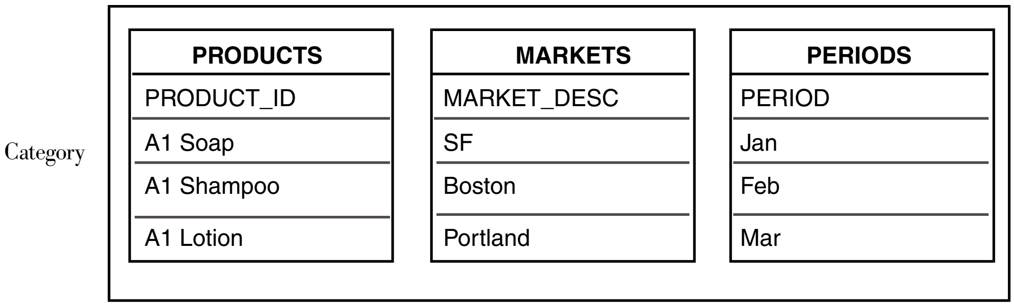
For example, you can use the DBA catalog in the Query tool to list database categories. See the *Data Access Tools User's Guide* for details on specifying categories.

You can use the DBA tool to display categories in the logical view of the database.

The Reporter, Forms Design, and Browser tools can access a specific category at any one time. You must configure these tools in the `Other Data` parameters in the Icon Options window to give them category table access. When a user opens one of these configured data access tools, the icon displays only the tables defined for the specified category. These tools do not open if a category is not specified.

Facts and Dimensions

Sets of facts and sets of dimensions are stored in separate tables. Dimension and fact tables are stored within a category. Figure 4 shows an example of dimension tables within a category.


Category

PRODUCTS
PRODUCT_ID
A1 Soap
A1 Shampoo
A1 Lotion

MARKETS
MARKET_DESC
SF
Boston
Portland

PERIODS
PERIOD
Jan
Feb
Mar

Figure 4. Dimension tables within a category

Dimension tables consist of data that define facts. Figure 5 on page 13 shows an example of dimensions that correspond to the tables in Figure 4. Using a category containing three dimension tables, you can create multidimensional views of data.

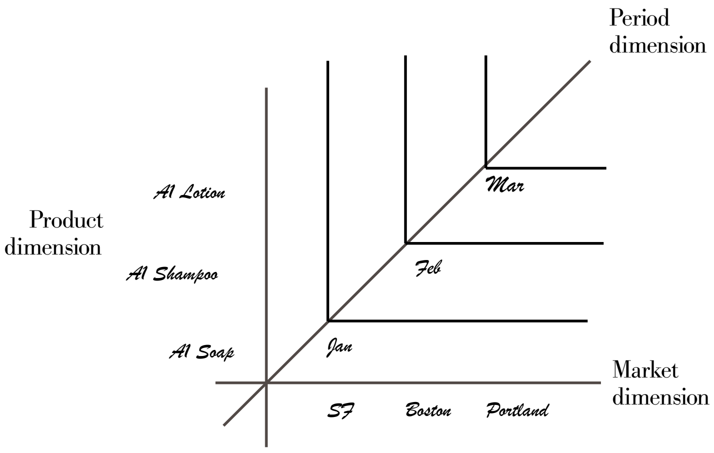


Figure 5. Dimensions formed by a category.

Fact and Dimension Table Associations in the Reporter Tool

The Reporter tool provides access to multidimensional databases. With it, users can specify the layout, format, and content of a report without any knowledge of the structure of the relational database. The Reporter tool works by sending several queries to the database server. The queries retrieve the data, and the Reporter tool builds a report based on this information.

The fact table contains the data that is used in the body of the report, or in calculations for a report. Unit Sales and Dollar Sales are examples of the type of information stored in fact tables.

A dimension table usually contains the information that defines the context of the individual facts. Names of products, markets, and periods of time are examples of information stored in dimension tables.

The Reporter tool works with only one database category at a time. See the *Data Access Tools User's Guide* for a list and description of Reporter icon options.

For the Reporter tool to function, you must follow these rules within a category:

- When you set up a data table, specify one of the following items in the Table Options or New Table Options window of the DBA tool:
  - `Not Used By Reporter`
  - `Reporter Fact Table`
  - `Reporter Dimension Table`
- For every key column in a fact table, there must be a corresponding key column in a dimension table and a default join specified for the relationship between the two key columns.

To support the rules above, you might need to create additional dimension tables, such as a period dimension table.

When a user opens a Reporter tool, the Reporter tool displays the fact and dimension tables and ignores tables defined as `Not Used By Reporter`. At the same time, the Reporter tool also retrieves tables that are used in joins.

Key Columns

Most tables have a set of columns, called *key columns*, that uniquely identify each of its rows. When these columns are grouped together, they are called the *primary key* of a table. For example, the fact table in Figure 6 on page 15 has a primary key that consists of the PRODUCT_ID, MARKET_DESC, and PERIOD columns. Three dimension tables are joined to the fact table, each with its own key column that corresponds to one of the columns in the primary key.

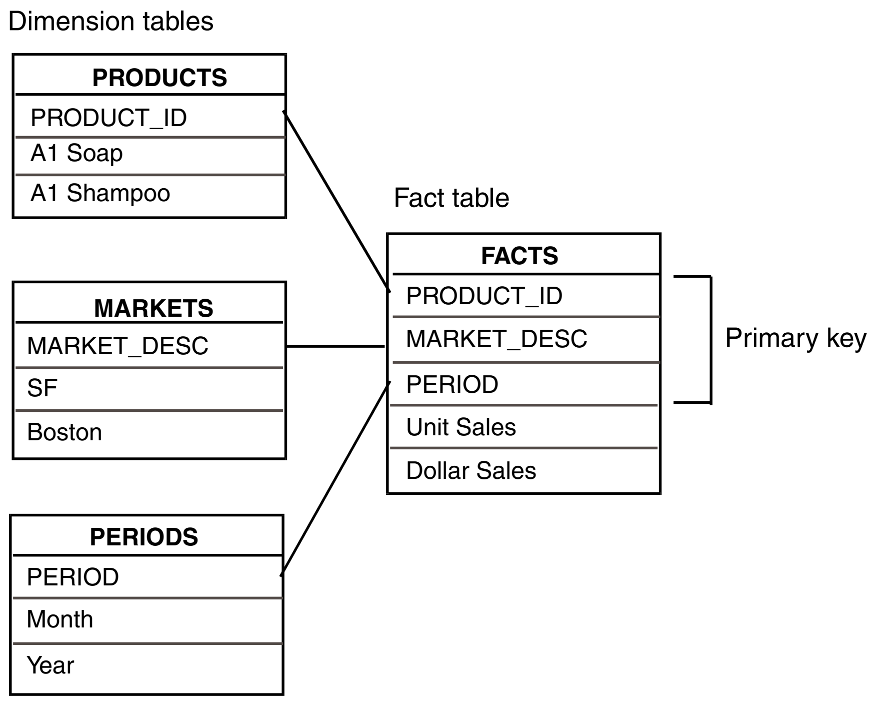


Figure 6. One or more key columns define a primary key

Some Meta5 data access tools must know which columns are primary key columns. These must be defined in the WTDD. The WTDD stores this information in the WT_COLUMNS table.

Joins

One of the most important features of a relational database system is the ability to store and maintain data in separate tables and to link those tables during data retrieval. Links between columns on different tables are called *joins*.

Joins between tables are automatically defined in the logical database so that the data access tool user does not have to specify them when querying the database. These joins are called *default joins*.

If you want to customize your joins, you can:

- Use the DBA tool to specify how tables are joined. These joins then become the default joins. You can specify up to 400 joins per category. See “Chapter 4. Getting Started With The DBA Tool,” on page 35 for more information.
- Use the Query tool to temporarily create joins between any two columns that are assigned the same data type. These joins are called *inner joins*. Using the Query tool to create an inner join has no permanent effect on the default

joins in the database. See the *Data Access Tools User's Guide* for more information.

A character column can be joined to another character column, a number column to another number column, and a column that displays a date to another column displaying a date.

You can use the Query tool to join two columns that have no logical connection. For example, both the Employee Number in the Employees table and the Department Number in the Departments table are integers. Although it makes no sense to do so, a user could join these columns because they are both defined as numbers. This is called a *nonsense join* because it makes no sense to use it.

To prevent users from creating nonsense joins, you must build data domains that you assign as domain numbers to each column. For example, you might assign Manager and Employees columns to the same domain (both refer to people), while excluding the Department Number column.

To assign a domain number, use either the Column Options window (for existing columns) or the New Column Options window (for a new column) in the DBA tool.

The Query tool allows only columns with the same domain number to be joined. Domain number 0 is the default domain number. It is used for all columns that do not have a domain number assigned to them. Use domain numbers carefully to ensure that you do not prevent users from applying joins that are truly needed.

The Employees and Departments tables in Figure 7 demonstrate how you might assign domain numbers to each of their columns so that Query tool users can join the tables only on the Department Number column. The domain numbers are shown in parentheses after the column names.

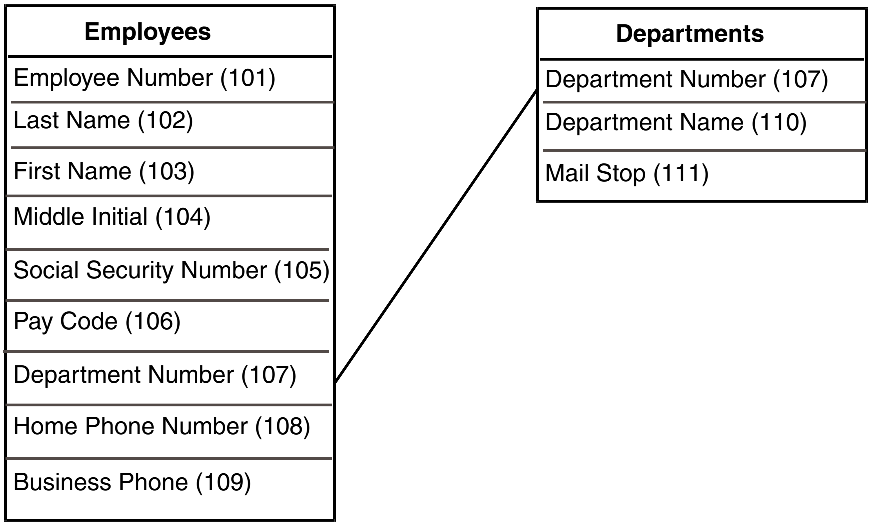


Figure 7. Domain numbers prevent nonsense joins

See the *Data Access Tools User's Guide* for more information about using the Query tool.

The Reporter tool supports both inner joins and two-way outer joins, where values that are not common to the joined columns are listed in the results.

The WT_DJOINS table stores information about joins between columns. The WT_COLUMNS table stores information about the domain number that is given to each column. The WT_DOMAINS table stores information that describes each of the domain numbers.

N/A Values

Meta5 supports nulls, so it is not necessary to use the N/A value. N/A values are supported for existing applications that use them. The WT_COLUMNS table stores information about N/A values.

Default Constraints

If a specific constraint is usually applied to a column, you can define that constraint as a *default constraint*. A default constraint is automatically applied in the Reporter tool unless a user changes it. The WT_DCONSTR table stores information about default constraints.

Important:
If the WT_DCONSTR table is modified, the only way to pick up the changes is to copy a new Reporter icon from the *Blank Icons file drawer.

Version Numbers

While making changes to the WTDD, you probably do not want data access tools to read the WTDD until after you test your changes. The WTDD has a set of version numbers associated with it to help you keep track of the current version of each WTDD table. The version numbers indicate to data access tools when you have changed the WTDD. The version number only applies to WTDDs that were read by previously opened data access tools.

When you want data access tools to use the new version of the WTDD, run the `wtupdate` macro. This macro increments the WTDD version number by one and places the new version on the database server.

When you change the version number, all data access tools that are connected to the affected database update their copy of the WTDD tables the next time they are opened. Any new tools that are opened read the current WTDD contents.

Handling Views

You might need to create *views* of the database for users or groups of users. A view is a subset of a table or joined tables. You create views to:

- Present data from two or more tables to a user as if the data were combined into one table.
- Restrict the use of sensitive or privileged data by displaying part of the data in one table and omitting fields such as salaries or other sensitive information.

The DBA tool treats views like a database table. You define views in the WTDD as logical tables.

If you want the Reporter tool to be able to use the table or view, you must indicate fact or dimension information when you create views. The WT_TABLES table stores information about views.

Modifying the WTDD

After you modify the WTDD, you must run the `wtupdate` macro to make the new version accessible to data access tools. Existing data access tools read a new copy of the WTDD the next time they are opened.

When a newly configured data access tool is first opened, it reads a copy of the WTDD tables from the database server and records the current version number. Each time the tool is opened after this, it compares the version number of its stored WTDD with the version number of the WTDD on the database server. If the number is unchanged, the tool uses its current copy of the WTDD. However, if the tool's WTDD version number differs from that of the WTDD on the database server, the tool reads a new copy of the WTDD.

Effect on Reporter Tools

The Reporter tool has an `Update Category` item, located in the `Special` menu, that lets you force an update of the icon's copy of the WTDD. The `Update Category` choice is useful to database administrators and applications developers who want to test their changes before implementing them for use by an entire user group.

When you update a category, the Reporter tool rereads the WTDD tables and displays an updated setup window. This action updates the categories for the current Reporter tool only, leaving other Reporter tools on your desktop (and throughout the system) running on the previous version of the WTDD.

If the WTDD updates successfully, the Reporter tool generates a confirmation message. If the WTDD cannot update successfully, a message identifies the function that failed. If the WT_DCONSTR table is modified, the only way to pick up the changes is to copy a new Reporter icon from the *Blank Icons file drawer.

Effect on Query Tools

Whenever a Query tool reads a new copy of the WTDD from the database server, the icon's original query structure or report structure is preserved. The Query tool has a function that lets the user force an update of the icon's copy of the WTDD.

Phantoms

Inconsistencies can result between the categories, tables, and columns that are retrieved by a query or a report and those in the newly changed database. The Query tool indicates such inconsistencies by displaying deleted or changed database items as *phantoms*. An overstrike through the print name of a database item indicates that it is a phantom. In addition, if the Query tool detects a phantom, a message is displayed in the Important Message window on the user's desktop.

Preventing Phantoms

Phantoms occur when you delete a database element or change its internal name. To prevent phantoms, never change the internal names of tables or columns. Change the SQL name instead.

Using the DBA tool, you or authorized users specify SQL names for database items in options windows for tables and columns. Meta5 tools use the SQL name for tables and columns to retrieve data.

To change the name of a table or column, use the `SQL Name` field in the options windows for tables and columns. To get an updated copy of the WTDD, you must run the `wtupdate` macro.

The WT_COLUMNS and WT_TABLES tables store the SQL names.

Backing Up and Restoring WTDD Tables

You can back up and restore WTDD tables by using the WTDD Backup capsule located in the DBA file drawer. Use the capsule to save the contents of each WTDD table to a spreadsheet as a backup before you modify WTDD tables. If you want to restore the backup version of WTDD tables, you can do so individually or all at once using the WTDD Backup capsule.

You can find the WTDD Backup capsule icon in the DBA file drawer on the system desktop.

To back up WTDD tables:

1. Open the WTDD Backup capsule icon, and click on the `Show Controls` button in the window header. The User Input Control window opens.
2. Specify ID information for the displayed @-variables for the database you want to access.

The capsule window contains an icon for each WTDD table that directs the contents of WTDD tables to an inner capsule, as shown in Figure 8 on page 20.

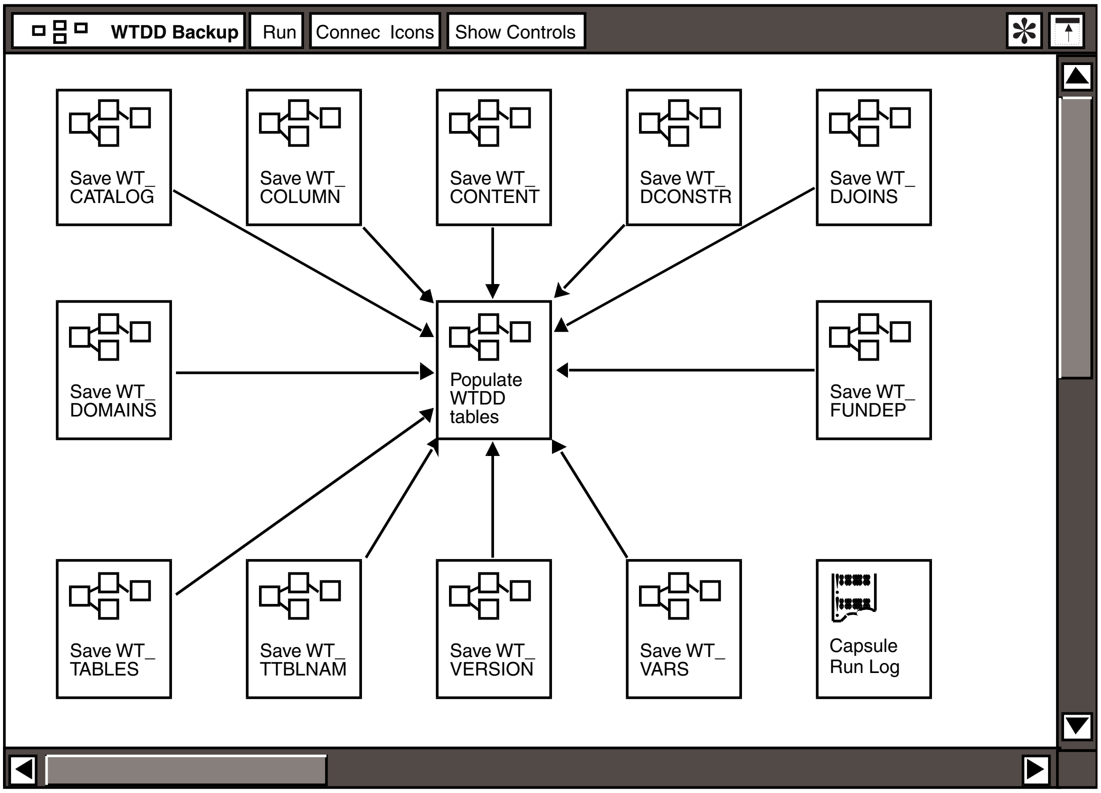


Figure 8. WTDD Backup Capsule

3. Click on the `Run` button in the window header to run the outer capsule.
4. Click on the `Populate WTDD Tables` capsule icon to open the inner capsule.

   As shown in Figure 9 on page 21, for each WTDD table represented in the inner capsule, an `In` icon directs the table contents to a spreadsheet, which in turn directs the table contents into a second spreadsheet and then to a Query tool. The Query tool reloads the backed up version of the WTDD table.

   The `In` icon and the first spreadsheet represent the backup function of the capsule. The second spreadsheet and the Query represent the restore function.

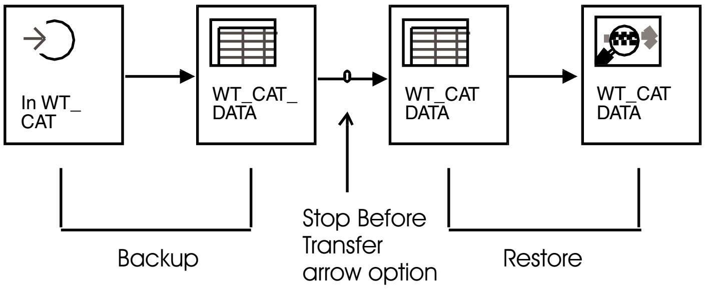


Figure 9. Icons for backing up and restoring a WTDD table

The WTDD Backup capsule is set up to stop data transfer between the two spreadsheets via arrow option settings. To back up WTDD table contents, specify the Arrow options in the Arrow Options window.

You can restore the backup version of WTDD tables either individually or all at once. To restore WTDD tables:

1. Select the arrow between two spreadsheets that indicates stopped transfer of WTDD contents, and open the Arrow Options window.
2. Specify **No** for the `Stop Before Transfer` option.
3. If you want to restore just one table, close the Arrow Options window and click on the `Run` button in the window header.

   To immediately restore multiple tables or all of the tables, repeat steps 1 and 2 for each table that you want to restore. When all arrows are specified, click on the `Run` button.

Chapter 3. Configuring Data Access Tools

Meta5 data access tools are applications that let authorized users retrieve information from databases. Typically, you configure the tools that users need and place them in a file drawer within the New Icons file drawer.

You can set up the Icon Options windows of data access tools so users can configure their own tools. These users can specify which database they want to access. In most cases, the users can switch easily from one WTDD to another without specifying synonyms for the tables.

This chapter describes the information provided in each field in the Icon Options window.

An example of how a data access tool retrieves data is provided in “How Data Access Tools Retrieve Data” on page 32.

Preparing a Data Access Tool for Configuration

Before you can configure a data access tool, you must first select a data access tool icon and name it. Configuration tasks for the data access tool are performed in the Other Data page of the Icon Options window.

To prepare a data access tool for configuration:

1. Select a data access tool icon.
2. Press and hold down both mouse buttons on the data access tool icon. Drag the mouse down to highlight the Icon Options window icon and release both mouse buttons. The Icon Options window opens.
3. In the `Name` field, enter a name for the icon.

   You should select a name for the icon that is similar to the name of the database and the application that you want to access. For example, if you are configuring a Browser tool for a DB2 database, you might call it DB2 Browser.
4. Select `Other Data` in the `Display` field.

The specific configuration tasks you perform depend on what type of database and gateway you are using. See “Setting Parameters for Data Access Tools” for specific instructions.

Setting Parameters for Data Access Tools

To connect to a database, you must first configure a data access tool by providing information in the Other Data page of the Icon Options window.

This section describes input for each field in the Other Data page.

Specifying a Gateway and a Database

There are different ways to specify a database depending on whether you want to access:

- A local or a remote DB2 UDB database
- A database using ODBC
- A database in the SYBASE, and ORACLE environments

Accessing a Local DB2 UDB Database

In the IBM environment, the term *local database* refers to the RDBMS that is running on the same system as Meta5 Host Services, the DB2 UDB for Windows gateway, or the DB2 UDB for AIX gateway.

The format for specifying a database is similar among databases. To access a local database, you must identify a gateway and a database:

1. In the `Gateway` field of the Icon Options window, enter the name of the database gateway that you want the tool to access. If the gateway is on a different realm from the one you are on, immediately follow the gateway name with a colon and then the realm name. For example:

   ```
   mydbs:HQ
   ```

2. In the `Database Name` field, the default database in both the IBM and UNIX environments is always local. If you are connecting to the local default database, you do not need to enter anything in this field. However, you must specify a database name if your server stores more than one physical database or if you want to connect to a database other than the default.

   If only one physical database exists, you do not need to supply a local database name. For DB2 UDBfor MVS, specify the name of a DB2 subsystem. For DB2 UDB for VM, specify the name of the virtual machine running DB2 UDB for VM.

Accessing an ODBC Database

The format for specifying a database is similar among databases. To access an ODBC datasource, you must identify a gateway and a database:

1. In the Gateway field of the Icon Options window, enter the name of the database gateway that you want the tool to access. If the gateway is on a different realm from the one you are on, immediately follow the gateway name with a colon and then the realm name. For example:

   ```
   mydbs:HQ
   ```

2. In the Database Name field, enter the Data Source Name (DSN) for the database. The DSN must be defined on the server running the UDB/ODBC Gateway Service. You should create the DSN as a System DSN.

Accessing a Remote IBM Database Using DRDA

This section describes how data access tools connect to a *remote database* that you specify. In the IBM environment, a remote database is an RDBMS that is accessed through a local database using Distributed Relational Database Architecture (DRDA).

DRDA connections must be set up between the hosts before Meta5 is installed, and WTDD tables must be installed in the remote databases. To access a remote IBM database, you must identify a gateway and a database in one of three ways:

Using the Icon Options Window:

1. In the `Gateway` field of the Icon Options window, enter the name of the database gateway that you want the icon to access. If the gateway is on a different realm from the one you are on, immediately follow the gateway name with a colon and then the realm name. For example:

   ```
   mydbs:HQ
   ```

2. In the `Database Name` field, type the name of the local database in the `Database Name` field, followed by a period and the DRDA server name:

   ```
   local_database.drda_server
   ```

   If only one physical database exists, as with the IBM AS/400, it is unnecessary to supply a local database name. Instead, type:

   ```
   .drda_server
   ```

   Be sure to include the period before the DRDA server name.

   To drop a DRDA connection, use the SQL CONNECT RESET command. After this command is issued, subsequent SQL statements are run by the local database.

As an alternative to using the Icon Options window to establish a DRDA connection, you can use the SQL CONNECT TO command or the Meta5 CONNECT command. You can use the DBA, SA, and SQL Entry tools to enter SQL statements.

Using the SQL CONNECT TO command: To use the SQL CONNECT TO command, type:

```
CONNECT TO drda_server
```

where *drda_server* is the name of the DRDA server that connects the DRDA to the remote database.

Using the Meta5 CONNECT Command: To use the Meta5 CONNECT command, type:

```
CONNECT userid{/ | IDENTIFIED BY} password DATABASE
database_name.drda_server
```

userid and *password*	Your user ID and password on the local database through which you want to access a DRDA server.
database_name.drda_server	The name of the local database that you want to access, followed by a period and the DRDA server name.

For example, type:

```
CONNECT guest/demo DATABASE mydb.drda400
```

The Meta5 CONNECT command returns a message that tells you the type of DRDA server and its version number. For example, you might receive the following message upon connecting to a DB2 for MVS V3.1 server:

```
Application server is DSN03010
```

Accessing a SYBASE Default Database

To access a SYBASE or Red Brick database through the default SQL server, specify the database name in the `Database Name` field in the Icon Options window.

Accessing SYBASE Databases through a Non-Default Server

To access a database through an SQL server other than the default, specify the database name in the `Database Name` field of the Icon Options window in the following format:

```
servername|databasename
```

servername	The name of the non-default SQL server.
databasename	The name of the database.

The interfaces file must be located in the Interface File directory that is specified in the Database Gateways - Gateway Details window.

To open the Database Gateways - Gateway Details window:

1. Open an SA tool to connect to the database, as described in “Chapter 6. Accessing Databases with the SA Tool,” on page 73.
2. Select a gateway and click on the **Show Gateway Configuration** button. The DB Gateways - Gateway Config window opens.
3. Select a gateway and click on the `Modify Gateway` button in the DB Gateways - Gateway Config window.

The directory listed in the `Interface File Directory` field must contain the Interfaces file.

Accessing an Oracle Database

To access an Oracle database located on the same UNIX system as the database gateway, leave the `Database Name` field blank in the Icon Options window.

Accessing a Non-Default Oracle Database

When the Meta5 UNIX gateway is running on a different workstation than your Oracle server, the Meta5 UNIX gateway is the client of the Oracle RDBMS. There are different procedures for connecting to a remote database from an Oracle server, depending on which version of SQL*NET** you use.

Using SQL*NET Version 1.12: To access a database other than that defined by the default system ID (SID) using SQL*NET Version 1.12, specify the database alias, or enter the full database name using the following steps:

1. Type the name of your gateway in the **Gateway** field.
2. Type the name of the database you want to access in the **Database Name** field the following format:

   ```
   T:host1:oracledb
   ```

`T`	Indicates the standard network prefix for the remote connection; in this example, `T` indicates TCP/IP. To find out what prefix you should use, refer to the appropriate SQL*Net documentation for network connection.
`host1`	The name of the target host defined in the UNIX /etc/hosts file.
`oracledb`	The system ID of the target ORACLE database.

The database alias can be defined in the UNIX /etc/sqlnet file.

The `Default SID` field in the Database Gateways - Gateway Details window specifies the default system ID.

Using SQL*NET Version 2.0 and higher: In SQL*NET Version 2.0 and higher, you must create the SQLNET.ORA file and the TNSNAMES.ORA file before attempting to connect to a Meta5 gateway. The contents of these files are described in your Oracle SQL*NET documentation.

Important:
An environmental variable, *TNS_ADMIN*, must be included in the Meta5 gateway user's login file. This specifies the directory that includes the SQLNET.ORA file and the TNSNAMES.ORA file.

To access a database other than that defined by the Default System ID using SQL*NET Version 2.0, specify the database alias using this procedure:

1. Type the name of your gateway in the **Gateway** field.
2. Type the name of the database you want to access in the **Database Name** field. This name is the SID_ALIAS specified in the TNSNAMES.ORA file.

The `Default SID` field in the Database Gateways - Gateway Details window specifies the default system ID.

Accessing a DB2 UDB for AIX Database

To access a database through the DB2 instance, specify the name of the database in the `Database Name` field of the Icon Options window. For more information on the DB2 instance, see your DB2 documentation.

To access a remote DB2 for AIX database, use either the DB2 Client Application Enabler (CAE) for AIX or the DB2 Software Developer's Kit (SDK) for AIX. In either case, the DB2 instance specified in your database configuration should be the local DB2 instance. Catalog the remote database locally before you specify the database name of your data access icon. The default DB2 instance is specified in the Database Gateways - Gateway Details window. Use the DB2 instance in this window for your DB2 for AIX instance.

Accessing a DataJoiner Gateway

Accessing a DataJoiner database from Meta5 is identical to accessing a DB2 for AIX database from Meta5. See your DataJoiner documentation for directions on how to configure different sources to DataJoiner.

If you currently use a SYBASE or an Oracle database gateway, use the `nnsyb` and `nnora` macros to create nicknames for your SYBASE and Oracle WTDDs, respectively.

Specifying a Category

The user specifies which tables the icon can access by entering the name of a category. When a user opens a configured data access icon, the icon displays only the tables defined for the specified category. The Reporter, Forms Design, and Browser tools will not open if you do not specify a category in the Icon Options window.

Specifying a WTDD Owner

The `WTDD Owner` field is used with databases that support qualified table names and that handle multiple WTDDs.

If you use this field, you do not need to create synonyms. If this field is blank, you must create synonyms for all the WTDD tables that users want to access. If users want to access a different WTDD, you need to drop the synonyms (using the SQL DROP command) and create synonyms for the other WTDD tables.

By using the `WTDD Owner` field, users can switch from one WTDD to another between tool sessions. The only exception to this rule occurs when users own their tables. In that case, there is no need to create synonyms.

When a tool builds queries, it uses the information in the `WTDD Owner` field to qualify the WTDD table name. For example, if a user has a WTDD in which the WTDD tables are qualified by MYWTDD, the tool constructs a query such as:

```
SELECT * FROM MYWTDD.WT_VERSIONS
```

In addition, if you are working with a database system that supports three-part names for routing queries to other database systems, you can specify the subsystem (or server name) and the table owner in the `WTDD Owner` field. For example, if the WTDD in the previous example resides on a remote DB2 system called DB2PROD, specifying `DB2PROD.MYWTDD` in the `WTDD Owner` field results in the following query:

```
SELECT * FROM DB2PROD.MYWTDD.WT_VERSIONS
```

Specifying an SQL ID

The `SQL ID` field provides the high-level qualifier for table names in the DB2 environment and makes it unnecessary for you to create synonyms for users to access tables. On all other systems, this field is ignored.

The SQL ID field redirects data access tools to use a different table qualifier than the default defined by the `User Name` field. The SQL ID can be an @-variable.

For example, a user is trying to access tables owned by user Marketing. Rather than requiring you to create a synonym to enable access for the user, the user can simply type `Marketing` in the SQL ID field.

To take full advantage of SQL IDs, you should not use fully qualified table names when adding tables to the logical database.

Specifying a User Name and Password

When a Meta5 tool connects to the database system, the database system checks the Other Data page `User Name` and `Password` fields.

These fields must contain a valid name and password. These entries might be the same as a user's desktop user ID and password, depending on how database security is set up at your site. In the environment, the user name becomes the default table qualifier for tables that the tool is accessing.

For DB2, the user ID is usually the primary authorization ID.

Using a Single ID or Multiple (Group) IDs

If you routinely configure icons for use by a department or other group, you might want to create a shared user ID and password for the group. The advantages of a single ID are easy implementation and less administration - you grant access privileges to fewer users and create fewer synonyms or views because users share all the data. The major disadvantage of a single user ID is that database security decreases. Also, it becomes more difficult to gather cost information for database usage.

If multiple user groups share access to a database but certain groups should not have access to other departments' data, you can create multiple user IDs, also called group IDs, in which user groups share a user ID and password. The advantages of having group IDs are a higher level of database security and easy tracking of database usage cost information. The disadvantage of group IDs is the amount of maintenance required. You must grant privileges to many users and create multiple synonyms or views according to user requirements.

Meta5 Security Features for Data Access Tools

To prevent unauthorized use of data access tools, Meta5 provides security features that you can implement in two ways:

- Through the Data Access Controls window
- By securing icons

The Data Access Controls Window

Users can configure a data access tool to read the database user ID and password from the Data Access Controls window of the desktop where the icon resides. This serves as an alternative to providing and securing a user ID and password in the Icon Options window. Because icons mailed to other users do not retain user and password information, using the Data Access Controls window provides a higher level of security. The desktop Data Access Controls facility also simplifies the process of changing expired passwords.

Before a user can add an entry to the Data Access Controls window, you must assign the user an appropriate database user ID and password. For detailed information on using the Data Access Controls feature, see *Getting Started with the Meta5 Developer's Desktop*.

When an icon that is configured to read user ID and password information from the Data Access Controls window is mailed to another user, the recipient must provide user ID and password information in the Data Access Controls window on the receiving desktop.

Securing a Data Access Tool Icon

Instead of using the Data Access Controls window, users can secure an icon by:

1. Ensuring that the **User Name** field is set to the name of the desktop using it.
2. Selecting **Yes** in the `Secured` field.

A message is displayed, informing the user that this is a permanent change. The user's desktop name is displayed in the **User Name** field.

Secured icons require that you create database user names (or IDs) for each desktop using the icon. You must also grant each of these users access to the database and to the tables within the database. The desktop names cannot be longer than 8 characters.

With secured icons, the database user ID and the desktop name must be the same. However, the database user password and the desktop password do not have to be the same.

To use secured icons, users' desktop names must correspond to an account that has access to the database specified in the Icon Options window. If a secured icon is mailed between users, the workstation receiving the icon automatically changes the user name in the Icon Options window to the recipient's desktop name. The recipient cannot change this name in the Icon Options window. If the database does not recognize the recipient's user ID, the recipient cannot open the icon.

Securing an icon is an irreversible operation. After you secure an icon, you cannot change the user name. Therefore, before you secure an icon, make a copy of it and verify that the users are defined and have access to the database.

Using Special Characters in Desktop Names

If a secured data access tool icon is opened from a desktop whose name contains a slash character, the workstation translates the slash to a dollar sign when the icon is opened and a connection is made to the database gateway or server. This happens because the host does not permit the slash character in a host user ID. Therefore, if a desktop name contains a slash, the corresponding host user ID should contain a dollar sign instead of the slash. For example, if the desktop name is smith/j, you should create a database user ID called smith$j.

Making Desktop Names Unique

Desktop names should be unique across all interconnected Meta5 systems. If desktop names are not unique, outside users might gain access to a database without your permission. You can prevent this from happening by defining unique names for all desktops. For DB2 for MVS only, you can prevent this occurrence by using the desktop ID-to-account ID mapping facility.

For example, assume a user on network Alpha has a desktop named Smith, and this user ID has access to a database. User Smith Alpha mails a secured icon to another desktop Smith on network Beta. If the icon already has a valid database password in its Icon Options window, user Smith Beta can also use that icon to access the database, as long as the passwords match. This gives user Smith Beta access to a database that user Smith Alpha can access, even if user Smith Beta has not been given access privileges.

Making Global Changes to Icons

If the SetDBAccess transformer is available to your users, they can use it to globally change (or set) the configuration options for data access icons that are located in a specified folder, including the password associated with the icon. For information about the SetDBAccess transformer, see “Chapter 8. Desktop Transformers,” on page 81.

Users can also run the MakeSecure transformer to secure all the data access icons in a desktop, file drawer, folder, capsule, or envelope. You can run MakeSecure from the system desktop only. See the *System Administration Guide and Reference* for details about the MakeSecure transformer.

How Data Access Tools Retrieve Data

Figure 10 on page 32 shows how the WTDD retrieves data from a database. The following steps correspond to the circled numbers in Figure 10 on page 32:

1. Use the Query tool to build and send a query from the graphical interface. In the example, `=A241` is a constraint on Brand ID.
2. The WTDD converts the query into an SQL statement.
3. The SQL statement is sent to the database. The database's physical data dictionary determines the location of the data.
4. The RDBMS retrieves the data and sends it to the requesting workstation.

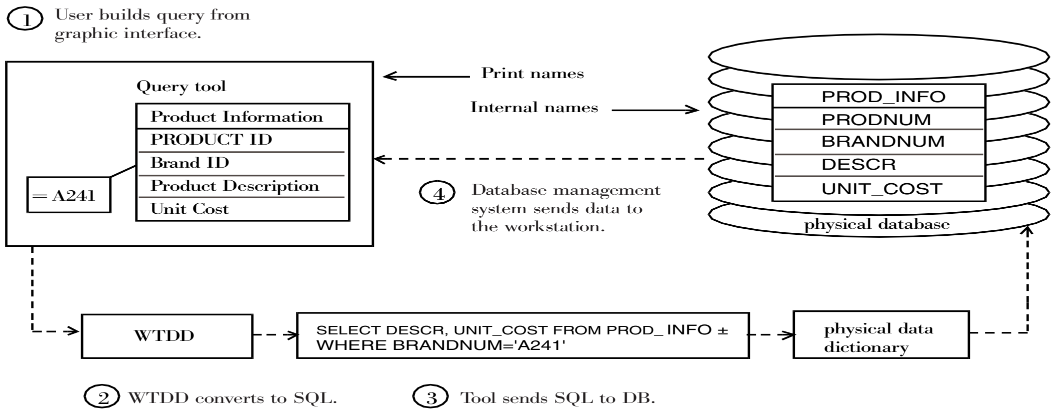
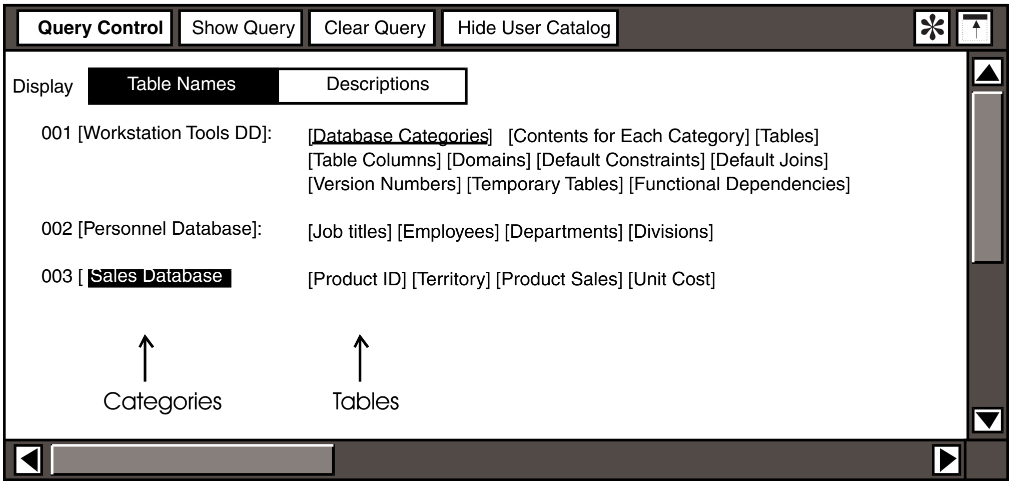


Figure 10. Workstation tool retrieving data from a database

Because the logical representation of the data is stored in the WTDD, users can access data through a single graphical interface regardless of the relational RDBMS.

A Query tool user who wants to see the logical categories of a database can view the DBA catalog within the Query Control window. The DBA catalog shows a list of all database categories and their tables, as shown in Figure 11. To learn more about this feature, see the *Data Access Tools User's Guide*.



001 [Workstation Tools DD]: [Database Categories] [Contents for Each Category] [Tables] [Table Columns] [Domains] [Default Constraints] [Default Joins] [Version Numbers] [Temporary Tables] [Functional Dependencies]

002 [Personnel Database]: [Job titles] [Employees] [Departments] [Divisions]

003 [Sales Database [Product ID] [Territory] [Product Sales] [Unit Cost]

Figure 11. DBA catalog in the Query Control window

Chapter 4. Getting Started With The DBA Tool

Use the Meta5 DBA tool to logically structure relational data, insert data definitions into the WTDD for user access, and create physical data tables.

The DBA tool automatically generates SQL statements from the selections you make in its windows. These SQL statements are run on the physical database. If you want to run your own SQL statements, the DBA tool also provides an SQL Control window, which is described in “Running SQL statements” on page 37.

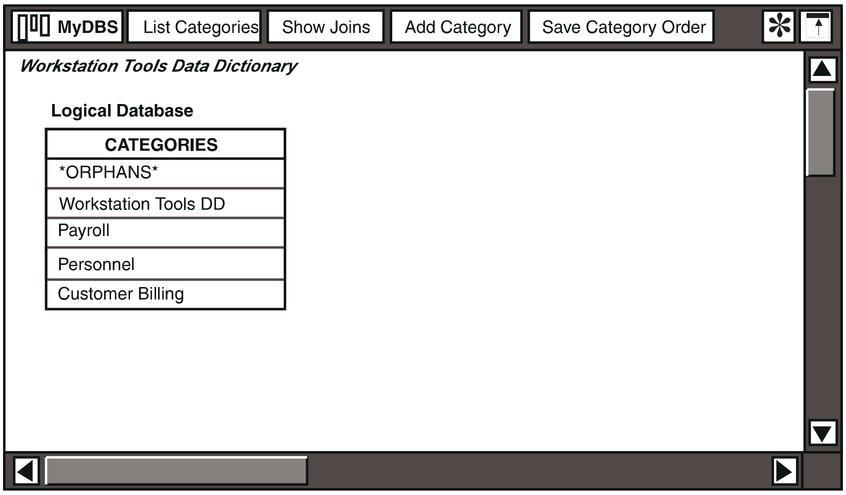
You must configure the DBA tool before you can connect to a database. See “Chapter 3. Configuring Data Access Tools,” on page 23 for information on how to configure the DBA tool.

You can secure a DBA tool to prevent its unauthorized use. To secure a DBA tool, or any other data access tool, see “Meta5 Security Features for Data Access Tools” on page 30.

If a DBA tool icon is not already on your desktop, you can find one in the *Administrator Icons file drawer.

Examining the DBA Tool Window

When you first open the DBA tool, categories are displayed in the logical database, as shown in Figure 12.



CATEGORIES
ORPHANS
Workstation Tools DD
Payroll
Personnel
Customer Billing

Figure 12. The DBA tool window

The *ORPHANS* category and the Workstation Tools DD category are always listed under the CATEGORIES heading. Listed after the Workstations Tools DD category are user categories.

The *ORPHANS* category is the parent category for tables whose categories have been deleted. You can reuse the table descriptions in *ORPHANS* by moving or copying them from *ORPHANS* into an existing or new category.

The Workstations Tools DD category contains the tables in the WTDD and enables Query tool access to the logical table information.

Window header buttons let you access information about the logical and physical tables in the WTDD without writing SQL statements. Table 2 lists the DBA tool window header buttons and describes their functions. Several window header buttons apply only at certain selection levels.

Table 2. DBA tool window header buttons

Button	**Description**
List Categories	Displays, in a DBA Log window, a list of the categories in the logical database by their internal name and their print name and includes a brief description of the element. The following buttons work similarly: • The `List Tables` and `List Columns` buttons are displayed at the tables or columns selection levels in the logical database. • The `List Joins` button is displayed in the default-join control window header when you display joins in the logical database. • The `List Indexes` button is displayed when you show table indexes in the physical database.
Show Joins	Displays a list of database tables and the joins between them in one or more selected categories.
Add Category	Lets you add a new category to the logical database. Similarly, the `Add Table` and Add Column buttons, which are displayed at the tables or columns selection levels, also add new items to the logical database.
Save Category Order	Saves the order in which the categories are displayed (selected after you rearrange the category sequence). Similarly, the `Save Table Order` and `Save Column Order` buttons, which are displayed at the table or column selection levels, also save the order in which the database items are displayed.

Table 2. DBA tool window header buttons

Button	Description
Special	Displays a menu with the options `Run SQL`, `Show Physical Database`, and `Capture SQL`.

You can display the physical database by selecting `Show Physical Database` in the `Special` menu.

When the physical database is displayed, you can redisplay the logical database by selecting `Show Logical Database` in the `Special` menu.

When you select window header buttons to access the WTDD, SQL runs in the background. You can capture this SQL code from both the logical and physical databases, as described in “Capturing SQL With the DBALog Window” on page 38.

If you want to type your own SQL statements to access the WTDD instead of having the DBA tool generate them automatically, you can open the SQL Control window and type commands directly into it. “Running SQL statements” on page 37 contains a description of how to use the SQL Control window.

Running SQL statements

Use the DBA tool to run your own SQL statements through the SQL Control window. This way, you do not have to open an SQL Entry tool each time you need to access the database server.

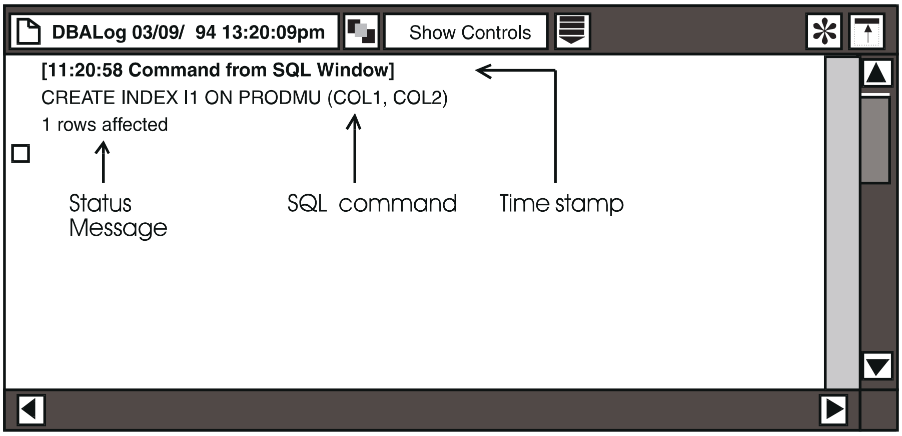
To run SQL statements in the DBA tool:

1. Select `Run SQL` in the `Special` menu of the main window header. The SQL Control window and the DBA Log window open. For a description of the DBA Log window, see “Capturing SQL With the DBALog Window” on page 38.
2. Type or copy an SQL statement into the `SQL statement` field. You can run only one command at a time.
3. Click on the `Run` button in the window header.

The DBA tool sends the command to the database. The DBA Log window displays the results of your SQL statement.

If you run a query through the DBA tool, the DBALog window displays three sections.

If you submit any SQL statement other than a query through the DBA tool, as shown in Figure 13, the DBALog window displays two sections upon completion of the statement run. The first section displays a copy of the statement and the time and date it was sent; the second section tells you how many rows were retrieved.



[11:20:58 Command from SQL Window]
CREATE INDEX I1 ON PRODMU (COL1, COL2)
1 rows affected

Figure 13. DBALog window

When running SQL statements in the DBA tool:

- To ensure that applications are transportable to other gateways or servers, do not use an IBM SQL extension as the name of a database object. Also, do not use a reserved word or keyword as the name of a database object. For a list of the words reserved for IBM and UNIX databases, see “Appendix A. Reserved Words for DB2 Databases,” on page 87 and “Appendix B. Reserved Words for Non-IBM UNIX Databases,” on page 93.
- To enter commands to the gateway, you do not need to preserve the capitalization you find in this publication, unless otherwise noted. The SQL interpreter is generally not case sensitive to commands or keywords. However, quoted text in the SQL statement is case sensitive.
- To run a command, you must click on the `Run` button in the SQL Control window header.
- To stop a currently processing SQL statement, press the Stop function key.

You cannot run macros using the DBA tool.

Capturing SQL With the DBALog Window

If you want to keep a record of database transactions during the current session, you can open a DBALog window to capture the SQL statements, as shown in Figure 13 on page 38.

To open a DBALog window, select `CaptureSQL` in the `Special`menu of the DBA tool window.

The DBALog window is a text window that provides edit and search functions. To access the edit menu, click on the `More` button. You can use this menu search for text and spelling errors.

The `More` button is displayed when the window is too narrow to display all the window header buttons. For additional information about these functions, see the *Text User's Guide*.

The DBALog window is updated whenever you close it, or for each screen of information that it captures.

The `Show Controls` button lets you specify page format.

To close a DBALog window, select the title block in the DBALog window header.

When the DBALog window is closed, the date and time is included in the icon name, showing when the DBALog window was opened. When this text document becomes full, the DBA tool automatically closes the full DBALog window and opens another one whose name contains the date and time when it was first opened.

Chapter 5. Using the DBA Tool to Work With Logical and Physical Databases

This chapter describes how to navigate through various selection levels of the logical database, how to display joins, and how to list logical database elements. This chapter also describes how to add or change information in the logical database, such as:

- Modifying options for database elements
- Changing the database structure by adding database elements
- Creating physical tables from logical definitions
- Changing the display order of database elements
- Deleting logical definitions

This chapter also describes how to create a physical table from a logical definition, navigate through levels of the physical database, and delete physical database elements. The examples show the physical database window for DB2 for MVS. If you are using a different RDBMS, its name is displayed instead of DB2 for MVS.

This chapter also explains how to verify changes made to the logical database and how to make changes accessible to database users.

Displaying Elements in the Logical Database

When you open the DBA tool icon, the logical database is displayed at the categories selection level. The upper left corner of the DBA tool window is titled `Workstation Tools Data Dictionary`, indicating that you are working with logical descriptions of the actual data. From this window, you can display logical database elements such as tables, columns, column definitions, and joins.

Displaying Tables

Figure 14 on page 42 shows how tables that are part of a selected category are displayed under the Tables in Category heading in the DBA Tool window.

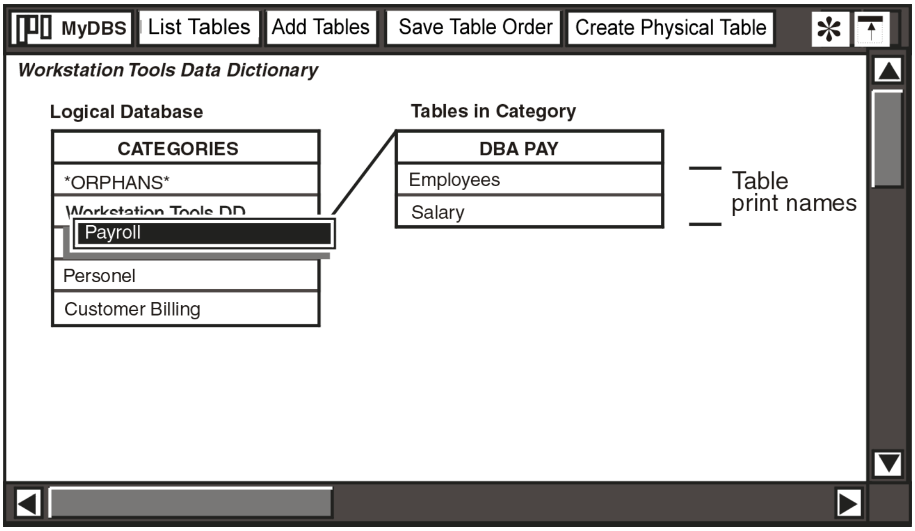


Figure 14. Tables selection level in the logical database

The internal name of the selected category, such as DBA_PAY, is displayed at the top of the table listing. The items listed below the internal name of the category are the print names of the tables in that category.

At the table selection level, the window header buttons indicate the functions that you can perform on tables. At this selection level, you can:

- List information in a DBALog window about all the tables in a category
- Add tables to a category
- Rearrange the order of the tables in a category and save the new order
- Create a physical table from its logical definition

To display the tables in a category, double-click on the category name.

Displaying Columns

Figure 15 on page 43 shows how columns that are part of a selected table are displayed under the Columns in Table heading.

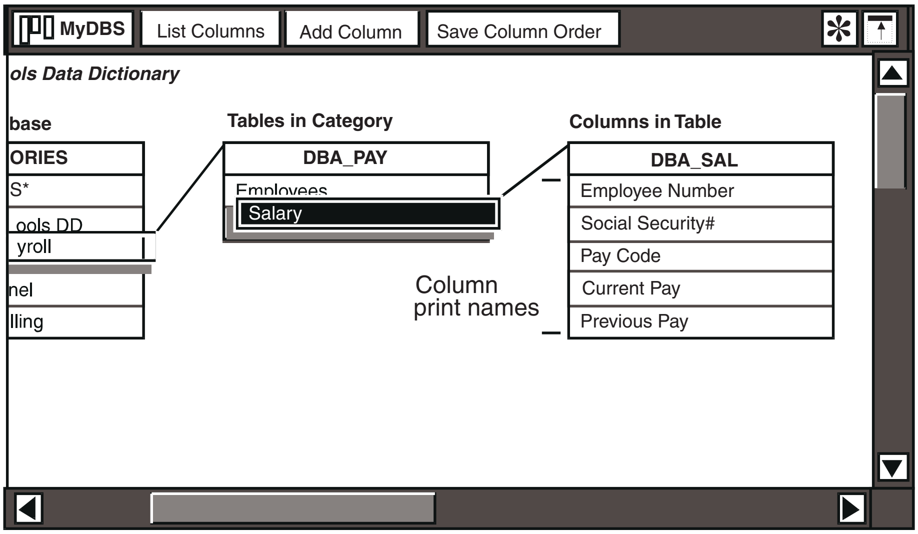


Figure 15. Columns selection level in the logical database

The internal name of the selected table, such as DBA_SAL, is displayed at the top of the list. The items listed below the internal name of the table are the print names of the columns in that table.

At the column selection level, the window header buttons indicate the functions that you can perform on columns. At this selection level, you can:

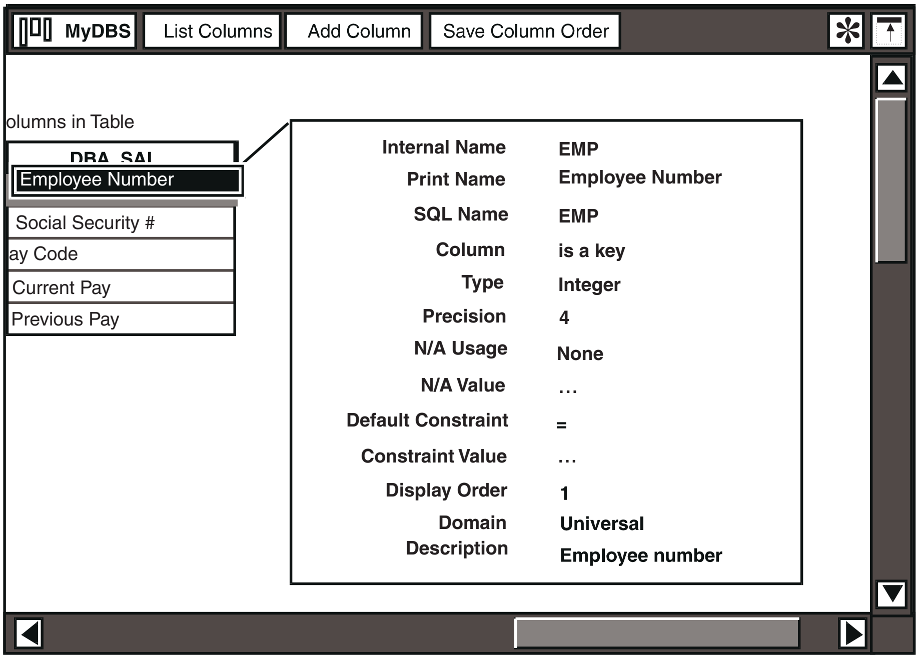
- List information in a DBALog window about the columns in a table
- Add columns to table
- Rearrange the order of columns in a table and save the new order

To display the columns in a table:

1. Double-click on a category name.
2. Double-click on a table name.

Displaying Column Definitions

The logical definition of the column is displayed in an information box, as shown in Figure 16 on page 44. The fields displayed in the logical column information box are identical to the fields displayed in the Column Options and New Column Options windows that you use to add a new column or modify an existing one. Table 4 on page 50 describes these fields.



Internal Name	EMP
Print Name	Employee Number
SQL Name	EMP
Column	is a key
Type	Integer
Precision	4
N/A Usage	None
N/A Value	…
Default Constraint	=
Constraint Value	…
Display Order	1
Domain	Universal
Description	Employee number

Figure 16. Logical column definition

The `Precision` field in the logical column definition applies to integer and real data types. The field `Size` is displayed for the character data type, and `Resolution` is displayed for the date data type.

To display the logical definition of a column:

1. Double-click on a category name.
2. Double-click on a table name.
3. Double-click on a column name.

You might have to scroll the window horizontally to see the entire column definition.

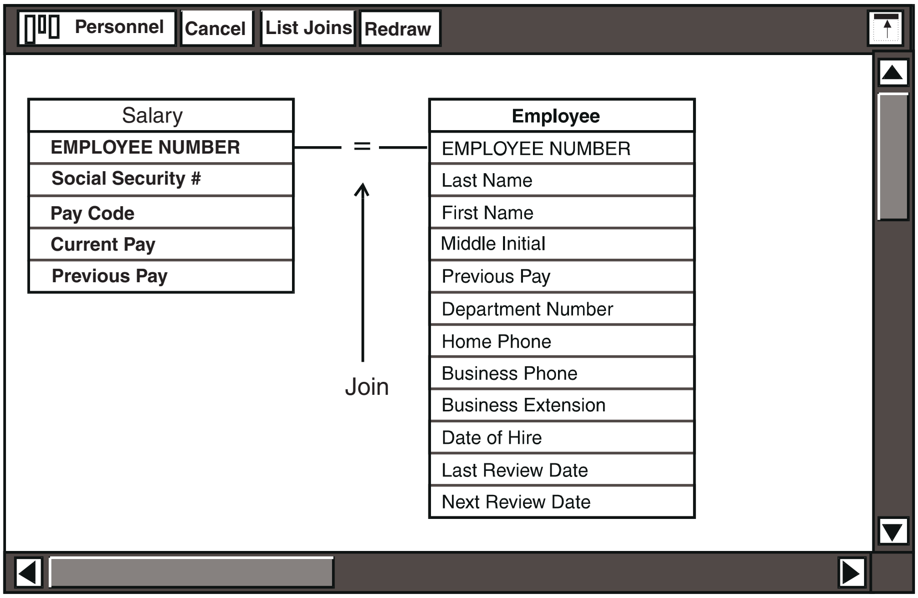
Displaying Default Joins

Use the DBA tool to display the default joins for tables in a category. To learn about default joins, see “Joins” on page 15.

To display default joins using the DBA Tool:

1. At the categories selection level, select a category.

2. Click on the `Show Joins` button in the window header. The default join control window opens, as shown in Figure 17 on page 45, displaying the default joins defined for the table you selected. The window takes on the name of the selected category. In this example, the category name is Personnel, which is displayed in the window title block.



Salary
EMPLOYEE NUMBER
Social Security #
Pay Code
Current Pay
Previous Pay

Employee
EMPLOYEE NUMBER
Last Name
First Name
Middle Initial
Previous Pay
Department Number
Home Phone
Business Phone
Business Extension
Date of Hire
Last Review Date
Next Review Date

Figure 17. Default join control window

To display default joins for more than one category at a time, select two or more categories, and click on the `Show Joins` button in the window header.

Modifying Elements in the Logical Database

Database elements are defined when you enter information into the Category Options or Default Join Options windows. Any information entered into these options windows is stored in the WTDD exactly as you enter it. The DBA tool does not modify data to conform to the syntax requirements of your RDBMS. For this reason, if you use the DBA tool with an RDBMS that converts physical names to uppercase, you should enter internal names in uppercase so they are consistent with the names stored in the physical data dictionary.

When you change information in the logical database, the DBA tool ensures that the WTDD remains consistent. For example, if you change the print name of a table, and that table belongs to more than one category in the logical database, the DBA tool reflects the change in all the categories to which the table belongs.

The WTDD allows internal table names up to 80 characters long and internal column names up to 18 characters long. However, the RDBMS you are using might have lower limits for name lengths. For example, if you enter an internal table name that is 18 characters long, the DBA tool stores all 18 characters in the WTDD, even if the RDBMS you are using accepts no more than 10 or 12 characters for table names, and even though some data access tools can only display the first 15 characters of a column name. Refer to your RDBMS documentation for the maximum length allowed so that you do not exceed it. Internal names cannot include embedded spaces; print names, however, are not subject to this restriction.

Important:
Some RDBMSs, such as DB2 for MVS, let you define category, table, and column names that use the at-sign @. Doing so, however, causes a problem if that category, table, or column is used in a capsule application or a data access tool, such as Query, which recognizes @-variables. Therefore, you should not use the at-sign when naming database elements.

Modifying Categories

Use the Category Options window to change the internal name or print name of a category, change its display order in the categories table in the DBA tool or in the DBA catalog of the Query tool, or provide a description of its contents. The description can be up to 200 characters long.

The Category Options window, used for modifying existing categories, and the New Category Options window, used for defining options for a new category, are identical. See “Adding Categories” on page 53 for a description of the New Category Options window.

To modify category options:

1. Select a category and open the Category Options window.
2. Type appropriate values for each field.

Modifying Tables

Use the Table Options window to change the internal name, print name, or SQL name of an existing table, specify the table's use by the Reporter tool, assign it a version number, and provide a description of its contents. Table 3 on page 46 defines these options.

Table 3. Table options

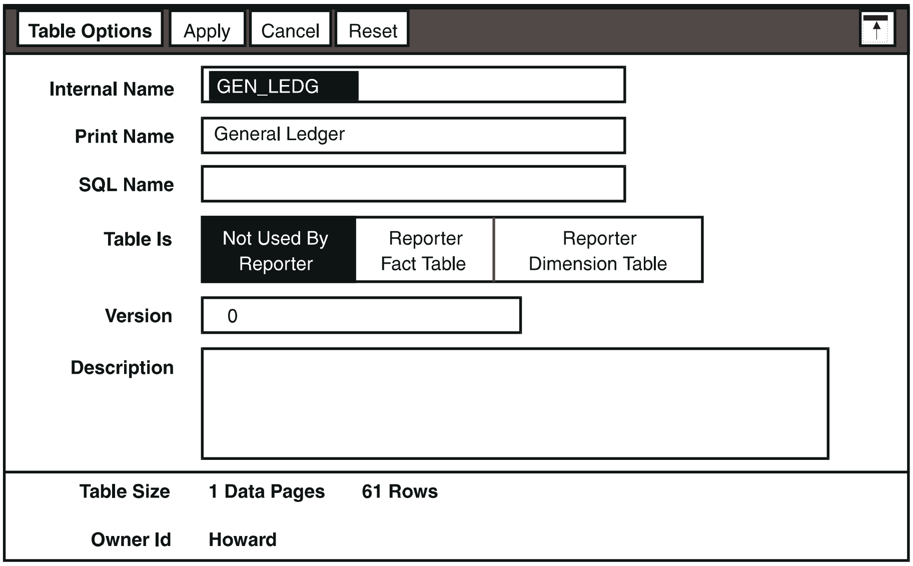
Option	Choice	Description
Internal Name	—	The physical name of the table, with a maximum of 80 characters. However, individual database servers can limit the internal name to fewer characters.

Table 3. Table options

Option	Choice	Description
Print Name	—	The name of the table as displayed in the logical database in the DBA tool or in the DBA catalog of the Query tool. It has a maximum of 30 characters.
SQL Name	—	The table name used for generating SQL statements. The SQL name is used to avoid phantoms in database queries. It has a maximum of 80 characters. If an SQL name is not provided, the data access tools use the internal name.
Table Is	Not Used By Reporter	Specifies that the Reporter tool cannot use this table.
	Reporter Fact Table	Specifies that the table is a fact table that the Reporter tool can use.
	Reporter Dimension Table	Specifies that the table is a dimension table that the Reporter tool can use. For more information on fact and dimension tables, see “Facts, Dimensions, and Categories” on page 12 and the *Data Access Tools User's Guide*.
Version	—	A number assigned by the DBA to track the table's update version.
Description	—	The user's description of the table, having a maximum of 200 characters.

To modify table options:

1. Display the tables selection level.
2. Select a table and open the Table Options window, as shown in Figure 18 on page 48.



Internal Name	GEN_LEDG
Print Name	General Ledger
SQL Name	
Table Is	[✓ Not Used By Reporter] Reporter Fact Table Reporter Dimension Table
Version	0
Description	
Table Size	1 Data Pages 61 Rows
Owner Id	Howard

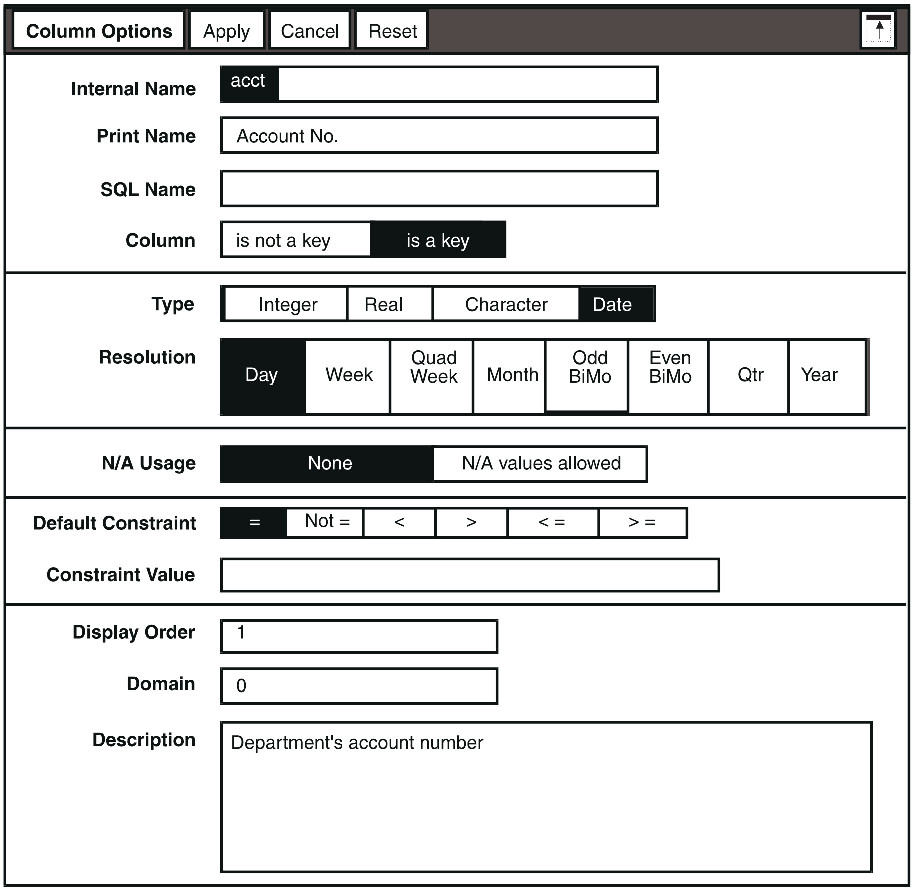
Figure 18. Table Options window

3. Select and type appropriate values for each field.

The Table Options window is identical to the New Table Options window, which is used to define options for a new table. See “Adding Tables” on page 53 for a description of the New Table Options window.

Modifying Columns

Use the Column Options window to modify a set of parameters that define an existing column, as shown in Figure 19 on page 49.



Column Options | Apply | Cancel | Reset

Internal Name	acct
Print Name	Account No.
SQL Name	
Column	is not a key [✓ is a key]
Type	Integer Real Character [✓ Date]
Resolution	[✓ Day] Week Quad Week Month Odd BiMo Even BiMo Qtr Year
N/A Usage	[✓ None] N/A values allowed
Default Constraint	[✓ =] Not = < > < = > =
Constraint Value	
Display Order	1
Domain	0
Description	Department's account number

Figure 19. Column Options window

To modify column options:

1. Display the columns selection level.
2. Select a column and open the Column Options window.
3. Select and type appropriate values for each field. See Table 4 on page 50 for information about each field.

When you specify that a column is a primary key, the DBA tool does not check to see whether that key is unique.

Some column options listed in the table are displayed in the Column Options window only when they are applicable. For example, `Resolution` is displayed only when `Type` is set to `Date`.

Table 4. Column options

Option	Choice	Description
Internal Name		The physical name of the column, with a maximum of 18 characters. However, individual database servers can limit the internal name to fewer characters.
Print Name		The name of the column as displayed in the logical database in the DBA tool, or in the DBA catalog of the Query tool. It has a maximum of 80 characters.
SQL Name		The table name used for generating SQL statements. The SQL name is used to avoid phantoms in database queries. It has a maximum of 18 characters. If an SQL name is not provided, the data access tools use the internal name.
Column		Specifies whether or not the column is part of the table's primary key.
Type		Specifies whether the data type is `Integer`, `Real`, `Character`, or `Date`.
	Precision	If `Type` is `Integer`, the user sets the precision to indicate the maximum allowable digits. If `Type` is `Real`, the `Precision` choice specifies the maximum number of decimal digits that can be contained in the physical column. For example, if 10.2 is entered, it means 12345678.90, not 1234567890.00
	Size	If `Type` is `Character`, the `Size` choice specifies the maximum number of characters that can be in the physical column.
	Resolution	If `Type` is `Date`, the `resolution` choice displays the date as day, week, quad week, month, odd bimonth, even bimonth, quarter, or year. Although all date resolutions can be displayed as non-numeric date types, the date resolutions `Week`, `Quad Week`, and `Qtr` do not translate correctly to date format when used as constraints in the Query and Reporter tools.
N/A Usage		Indicates if N/A values are used. If you select `N/A Values allowed`, the user must provide a value that the DBA tool will recognize as a null. Only use the N/A values field if you have existing applications that use them.
Default Constraint		The type of operator applied to the column. For example, `=` or `not=`.

Table 4. Column options

Option	Choice	Description
	Constraint Value	A user-specified value to be used with the `Default Constraint` that is applied to the column.
Display Order		The relative position of the column when the physical table was created.
Domain		A number that prevents users from creating invalid joins between table columns. It allows only columns with the same domain number to be joined. Domain number 0 is the default number, or universal domain; it is used for all columns that do not have an assigned domain. Domains 1 through 99 are reserved for system use.
Description		A user-provided description of the column, with a maximum of 200 characters.

The Column Options window is identical to the New Column Options window, which is used to define options for a new column. See “Adding Columns” on page 54 for a description of the New Column Options window.

Modifying Default Joins

Use the Default Join Options window to define the type of comparison operator applied between the columns in a join and whether the join can be used by the Reporter tool, as shown in Figure 20.

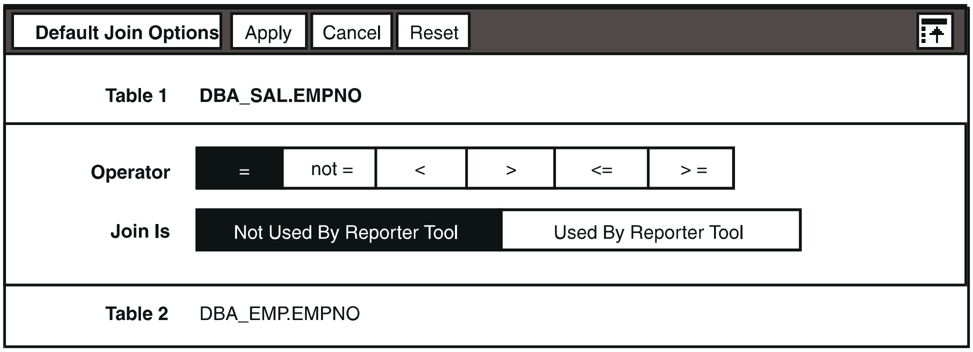


Table 1 DBA_SAL.EMPNO

Operator: =

Join Is: Not Used By Reporter Tool

Table 2 DBA_EMP.EMPNO

Figure 20. Default Join Options window

To modify join options:

1. Display the categories selection level.
2. Select one or more categories.
3. Click on the `Show Joins` button in the window header.
4. Select a join and open the Default Join Options window.

5. Select appropriate values for each field.

Table 5 on page 52 defines the options that apply to default joins.

Table 5. Default join options

Option	Choice	Description
Table1	—	The physical name of the first table and column in the join. Table and column names are separated by a period. For example: `DBA_SAL.EMPNO`
Operator	any choice	The operator that shows how the joined column in the first table is compared with the joined column in the second table.
Join Is	Not Used by Reporter	Specifies that the Reporter tool cannot use this join.
	Used by Reporter Tool	Specifies that the Reporter tool can use this join.
Table2	—	The physical name of the second table and column in the join. Table and column names are separated by a period.

Listing Database Elements

You can use database listings to display or print a summary of database elements defined in your database. The DBA tool generates a list of database elements in the DBALog window.

When you select the `List Categories`, `List Tables`, `List Columns`, `List Joins`, or `List Indexes` buttons, which are displayed at various selection levels in the logical and physical databases, the DBALog window opens and displays the list of elements. If the DBALog window is already open, the list in the window is appended with the new selection.

Database elements are listed by their internal or physical name. The print names of the element are included with a brief description taken from the `Description` field in the Options window for the related database element.

Adding Elements to the Logical Database

You can change the structure of the logical database by adding categories, tables, and columns. You can do this by adding new database elements or by copying and then modifying them. You can also change the order of database elements.

Adding Categories

You can add a new category to a logical database by specifying fields in the New Category Options window.

The New Category Options window is identical to the Category Options window, which you use to modify category options for an existing window.

To add a category to the logical database:

1. Display the categories selection level.
2. Click on the `Add Category` button in the window header. The New Category Options window opens.
3. Type appropriate values in each field.
4. If you are adding only one category, close the New Category Options window after you enter the category information.

   If you want to add another category to the database, click on the `Apply` button, then select `Reset`, and define the next category before closing the window.

   See “Modifying Categories” on page 46 for a description of category options.

Adding Tables

You can add a new table to a category, or you can copy an existing table from one category into another category and modify the table options and default constraints.

If a table already exists in the physical database and you want to create a logical definition for it, you can add the table in the logical database and specify the same table name as the existing physical table.

To add a new table to a category:

1. Select the category that will contain the new table.
2. Click on the `Add Table` button. The New Table Options window opens. It is identical to the Table Options window. See “Modifying Tables” on page 46 for a description of the Table Options window.
3. Enter the table options. See “Modifying Tables” on page 46 for a description of table options.
4. If you are adding only one table, close the New Tables Options window after you enter the table information.

   If you want to add another table to the database, click on the `Apply` button, then click on the `Reset` button and define the next table before closing the window.

You can copy an existing table to a different category if the table is common to one or more categories. When you copy an existing table to a different category, the same logical definition for the table exists in two different categories. That is, two different table references point to the same table. For a description of table references and how they differ from the logical definitions of a table, see “Deleting Definitions and Tables from a Logical Database” on page 60.

To copy an existing table from one category into another:

1. Select the table to be copied.
2. Copy it into another category.
3. Verify that the table was copied by displaying the category into which you copied the table. The copied table is listed there.

When you copy a table into another category, the table retains the same logical definition but now exists in two categories. If you modify one definition, both tables will change. All column definitions within that table are also copied.

Adding Columns

The method for adding columns to a table depends on whether the corresponding physical table already exists.

When you add a table to the logical database and the corresponding physical table already exists, the DBA tool uses the physical definition of the columns from the physical data dictionary that the RDBMS maintains to create the logical definition of the columns. Because the physical data dictionary contains information about the physical columns only, the DBA tool uses defaults for some of the column options in the logical database. For example, it uses Internal Name for Print Name. If you do not want to use the defaults, you can modify them as described in “Modifying Columns” on page 48.

To add columns to a table when the physical table exists:

1. Display the tables selection level.
2. Add a new table to the category, specifying the same table name as the existing physical table.

   The DBA tool uses the physical definition of the table's columns from the physical data dictionary to create the logical definition of those columns.
3. Modify the table and column options settings as desired.

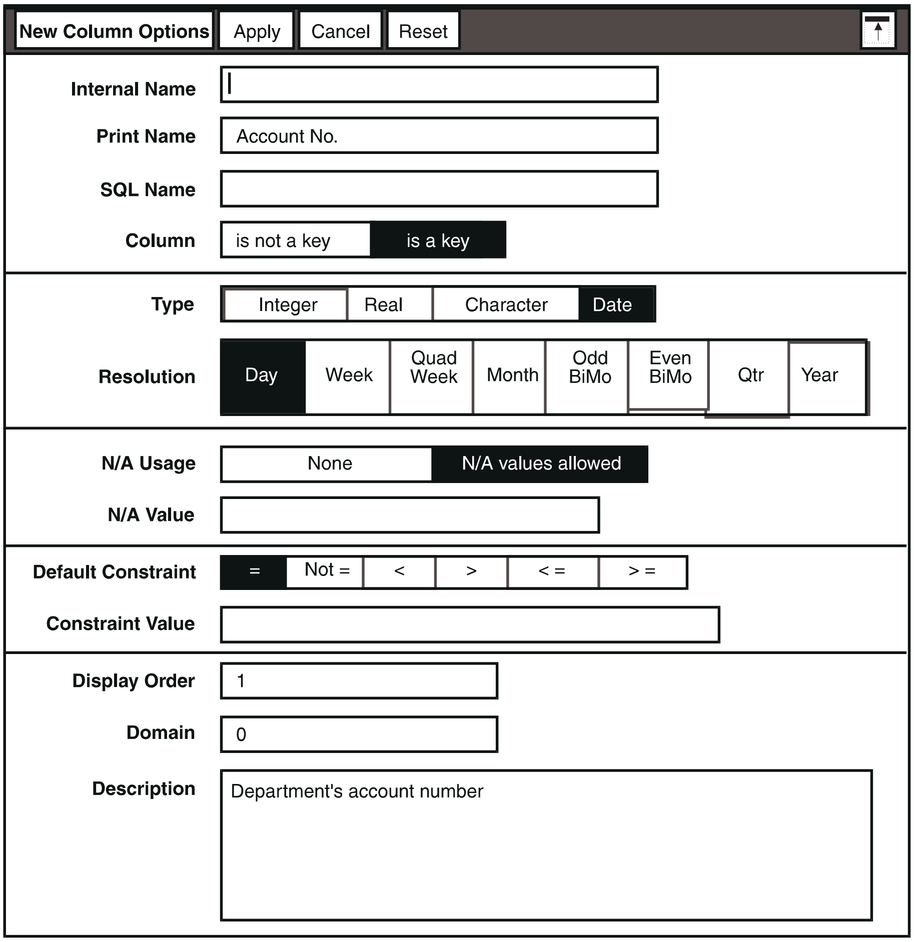
When the physical table does not exist, you can modify the existing logical table definition by adding new columns or copying logical column definitions from another table.

To add a new column:

1. Display the table for which you want to add column definitions.
2. Click on the `Add Column` button in the window header. The New Column Options window opens, as shown in Figure 21 on page 55. It is identical to the Column Options window.

3. Type appropriate values in each field. See Table 4 on page 50 for a description of the column options.
4. If you are adding only one column, close the New Column Options window after you enter the column information.

   If you want to additional columns to the table, click on the `Apply` button, then click on the `Reset` button, and define the next column before closing the window.



Internal Name	
Print Name	Account No.
SQL Name	
Column	is not a key [✓ is a key]
Type	Integer Real Character [✓ Date]
Resolution	[✓ Day] Week Quad Week Month Odd BiMo Even BiMo Qtr Year
N/A Usage	None [✓ N/A values allowed]
N/A Value	
Default Constraint	[✓ =] Not = < > < = > =
Constraint Value	
Display Order	1
Domain	0
Description	Department's account number

Figure 21. New Column Options window

You can create a new logical table and use columns from an existing table. This procedure is not the same as copying a table, because the result is a new and distinct logical table. After you complete the procedure and you modify the new table, the old table does not change.

To copy column definitions from one table to another:

1. Copy an existing table into the category to which the new table will belong. For example, copy Table A.
2. Click on the `Add Table` button in the window header, add a new table to the destination category, and specify a different name than the original table. For example, specify `Table B`.

   There should now be a Table A (a copy of the original Table A from a different category) and a new Table B.
3. Display the columns in Table A.
4. Select and copy the desired columns to Table B.
5. Delete the copy of Table A, if desired.

   For information on deleting tables, see “Deleting Definitions and Tables from a Logical Database” on page 60.
6. Modify Table B as desired.

Adding Default Joins

Default joins are used to connect matching columns in database tables. You define default joins when you design the database. To add default joins to the tables in a category:

1. Select a category.
2. Click on the `Show Joins` button in the window header. The default-join control window for the category opens, displaying tables in that category. The name of the category is displayed in the title block of the window, as shown in Figure 22 on page 57.
3. Select a column in the first table to be joined, and then select a column in the second table to be joined.

   The join is usually represented by a line with an equal sign (=) in the middle.

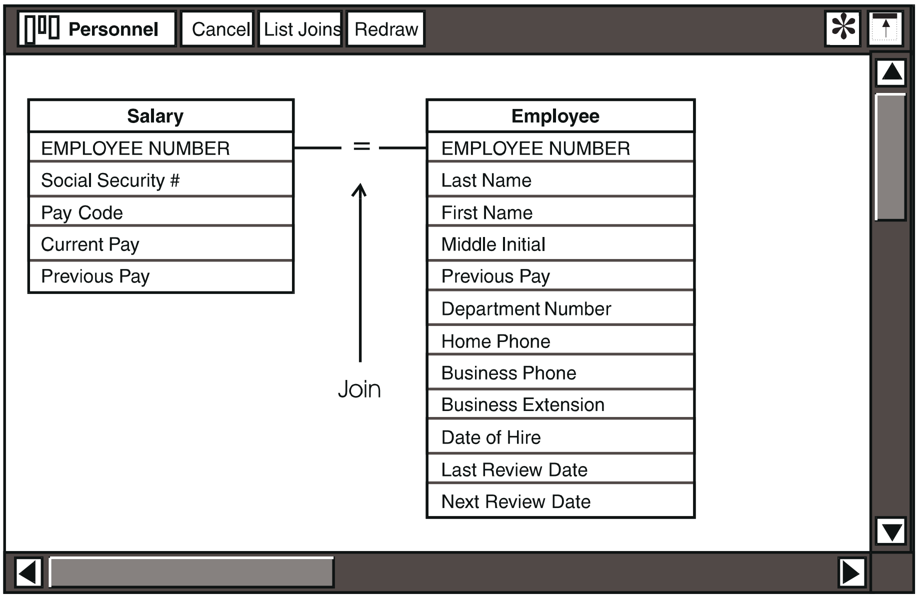


Figure 22. Default-join control window

Creating Physical Tables from Logical Definitions

When you add a table and its columns to the logical database, you must specify information for the physical table, such as the internal names of the table and its columns. The DBA tool uses this information to create the corresponding physical table.

You must have columns defined in a logical table before you can create a physical table.

To create a physical table from a logical table definition:

1. Define a table in the appropriate category.
2. Define columns for a logical table.
3. Display the tables selection level.
4. Select the logical table from which you will create a physical table.
5. Click on the `Create Physical Table` button in the window header.

At this point, the procedure varies according to the RDBMS you are using. The following section describes the procedure for creating a physical table in Meta5-supported databases.

DB2 Family of Databases

This section describes what happens when you create physical tables from logical definitions in IBM databases that Meta5 supports. Your database administrator must authorize your user ID to create tables in the database.

If the table exists when you select `Create Physical Table`, an Important Message window opens informing you that the table you tried to create already exists. If you want to create a new table with the same name that is different than the existing table, you must first drop the existing table. For information about deleting a physical table, see “Deleting Elements from the Physical Database” on page 70.

DB2 for MVS

When you create a physical table in a DB2 for MVS database, the Create Physical Table window opens. You can specify where you want the table stored:

- You can specify a table space.
- You can use the default database by leaving the `Table Space` field blank. DB2 then generates a table space for you in the default database.

To specify a table space:

1. Enter the database name and the table space name, separated by a period, in the Create Physical Table window. For example, type:

   ```
   dbname.tsname
   ```

   where `dbname` is the database name and `tsname` is the table space name.

   Before you can use a new table space, you must create it. You can select `Run SQL` in the `Special` menu to open the SQL Control window, where you can enter the appropriate SQL statements to create a table space.
2. Click on the `Proceed` button.

   A confirmation message is displayed when the table is created. If you do not want to create the table, either close the window or click on the `Cancel` button in the window header.

DB2 for VM

When you create a physical table in a DB2 for VM database, the Create Physical Table window opens. You can specify where you want the table stored:

- In the `DBSpace` field, specify the database that will store the table.
- Use the default database by leaving the `DBSpace` field blank. DB2 for VM then generates a DBSpace name for you.

To specify a DBSpace:

1. Open the Create Physical Table window.

2. Click on the `Proceed` button. A confirmation message is displayed when the table is created. If you do not want to create the table, either close the window or click on the `Cancel` button in the window header.

Before you can use a DBSpace, you must create it. You can click on the `Run SQL` button to display the SQL Control window where you can enter the appropriate SQL statements to create a DBSpace.

DB2 UDB for Windows

For DB2 for Windows databases, there are no options for you to specify. Clicking on the `Create Physical Table` button in the window header creates the new table. A confirmation message is displayed when the table is created.

UNIX Databases

When you create physical tables from logical definitions in Oracle, and SYBASE databases, there are no options for you to specify. Clicking on the `Create Physical Table` button in the window header creates the new table. A confirmation message is displayed when the table is created.

If the table exists when you select `Create Physical Table`, an Important Message window opens informing you that the table you tried to create already exists. If you want to create a new table with the same name that is different than the existing table, you must first drop the existing table. For information about deleting a physical table, see "Deleting Elements from the Physical Database" on page 70.

Changing the Display Order of Database Elements

You can change the order of existing database elements by rearranging them in the display area of the DBA tool. You can also create new database elements based on existing ones.

To permanently change the display order of categories, tables, or columns, you must first rearrange the items in a listing, and then click on the `Save...Order` button in the main DBA tool window header. If you rearrange elements without clicking on the `Save...Order` button, the new order is not permanently changed.

To change the order of categories in the logical database:

1. Select a category.
2. Move the category to a new position in the category listing.
3. Click on the `Save Category Order` button in the main window header.

Although the DBA tool lets you move the *ORPHANS* category temporarily to another location within the categories listing, *ORPHANS* does not actually move to that location when you click on the `Save Category Order` button. *ORPHANS* always remains first in the list of categories.

Table 6 on page 60 summarizes how you can rearrange database elements in the logical database by moving and copying them.

Table 6. Rearranging database elements

Database Element	Moving	Copying
Category	You can move categories in the logical database to change their display order.	You cannot copy a category.
Table	You can move tables within a category to change their display order, or you can move them from one category to another.	You can copy tables from one category to another. However, tables cannot be copied within the same category.
Column	You can move columns within a table to change their display order.	You can copy columns from one table to another in the same category.

Deleting Definitions and Tables from a Logical Database

When you delete an element from the logical database, you only delete the element from the WTDD, not from the physical database. The corresponding physical table remains in the database, along with all of its data and indexes.

Table 7 on page 60 summarizes what happens when you delete various elements from the logical database.

Table 7. Results of deleting database elements

If you delete this element:	The result is:
Category definition	The category definition is removed from the logical database. • If a table that belonged to the deleted category is in other categories as well, the table remains in those other categories. • If a table that belonged to the deleted category is not in any other category, it is moved into the *ORPHANS* category. In either case, tables belonging to the deleted category remain in the logical database.
A table reference that is not in the *ORPHANS* category	The table is removed from the category. • If the table is in other categories, it remains in those other categories. • If the table is not in other categories, it is moved into the *ORPHANS* category.

Table 7. Results of deleting database elements

If you delete this element:	The result is:
A table reference in the *ORPHANS* category	The definition of the table, its columns, and default joins are completely removed from the logical database.
Table definition	The logical table definition, table reference, columns, and default joins are completely removed from the logical database. Before you delete tables from the logical database, be sure you understand the difference between a table reference and the logical table definition. In the DBA tool, the table reference is the print name of the table in a category, and the table definition is the listing that shows the table columns. Figure 23 on page 62 shows both items.
Column definition	The column definition is removed from the table, and any default joins and default constraints based on that column are also deleted. Columns are deleted only through the logical database.

When you delete an element from the logical database, the DBA tool opens a Confirmation window. The display verifies that you want to proceed with the deletion. Click on the `Proceed` button to delete the selected element. Close the Confirmation window or click on the `Cancel` button if you do not want to delete the item.

When deleting a table from the logical database, you either delete the table reference or the logical table definition from the WTDD.

In the DBA tool, the table reference is the print name of the table in a category, and the logical table definition is the listing that shows the table columns. Figure 23 on page 62 shows the difference between a table reference and the logical table definition.

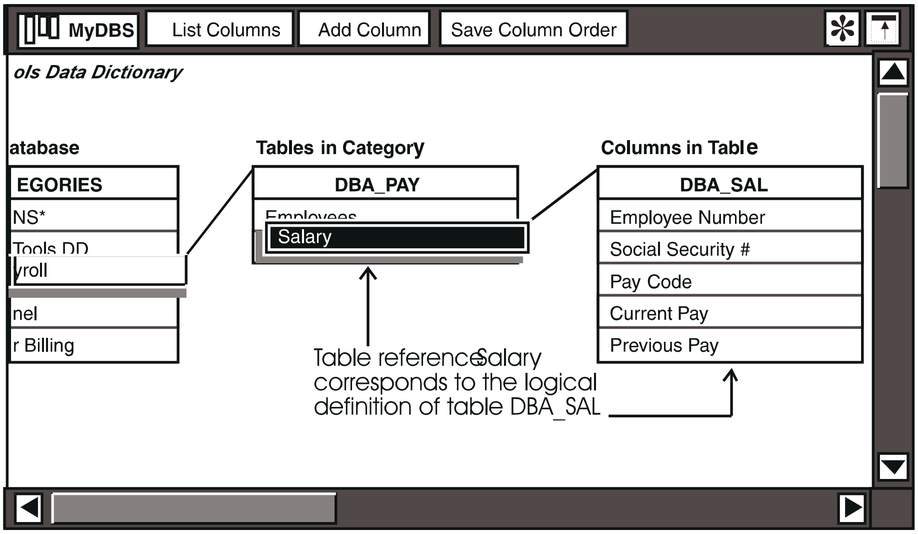


Figure 23. A table reference and the logical table definition

To remove a table reference and retain the table definition in the logical database:

1. Display the table selection level.
2. Select the table reference in the Tables in Category listing, for example, Salary in Figure 23 on page 62, and delete it. The following message is displayed in the Confirmation window:

   ```
   Delete references to selected tables from category?
   ```

3. Click on the Confirmation window. The table reference is still displayed in the *ORPHANS* category.

Important:
If you delete the table reference from the *ORPHANS* category, the table is removed entirely from the logical database.

If you want to remove a table from the logical database entirely, including the definition of the logical table and all of its references:

1. Display the columns selection level.
2. Click on the table heading. All the columns become highlighted.
3. Delete the columns. The following message is displayed in the Confirmation window:

   ```
   Delete table 'table_name' and all logical references to it?
   ```

4. Click on the Confirmation window header.

Testing Changes Made to the Logical Database

When you change the logical database, you are changing the WTDD. You can test changes you make to the WTDD by opening a Query tool and a Reporter tool. Changes to the WTDD do not show up automatically in an open Query or Reporter icon unless the WT_VERSIONS table in the WTDD is updated.

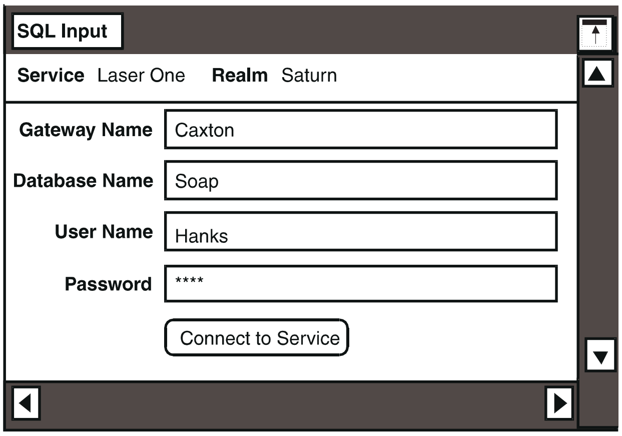
When a user closes a Query or Reporter icon, a copy of the then-current WTDD is saved along with it. However, if a user opens a new (blank) icon, a copy of the latest WTDD is retrieved. Query and Reporter tool users can update WTDD information that is stored with opened Query and Reporter tools. For more information on these features, see the *Data Access Tools User's Guide*.

Updating the WTDD

To ensure that an open data access tool obtains the latest version of the WTDD, the version number associated with WTDD tables must be incremented by one. This is done using the `wtupdate` macro. You can run the macro from the SQL Input window in the SA tool. The next time the tools are opened, they read the latest version of the WTDD.

To update the WTDD using the `wtupdate` macro:

1. Log on to system desktop.
2. If an SA tool icon is not already on the desktop, you can find one in the *Administrator Icons file drawer.
3. Open the SA tool window.
4. Select a realm from the menu in the `Realm` field.
5. Select `SQL Input` from the menu in the `Service Type` field.
6. In the `Select Service` field, select a service from the menu in the `Service` field. Alternatively, you can click on `By Name` and type the name of a service in the `Service` field.
7. Click on the `Connect to Service` button. The SQL Input Service window opens, as shown in Figure 24 on page 64.



Service Laser One Realm Saturn

Gateway Name: Caxton

Database Name: Soap

User Name: Hanks

Password: ****

Figure 24. SQL Input Service window

8. In the fields in this window, provide the name of the gateway service and database to which you want to connect, and the database user ID and password.
9. Click on the `Connect` button to start an SQL Input session. If you typed valid information, an SQL Input Log window also opens.
10. In the SQL Input window, type the following macro call:

```
CALL wtupdate
```

Displaying Elements in the Physical Database

The physical database displays the internal, or physical names of database elements grouped by user. To display the physical database, open the DBA tool icon and select `Show Physical Database` in the `Special` menu.

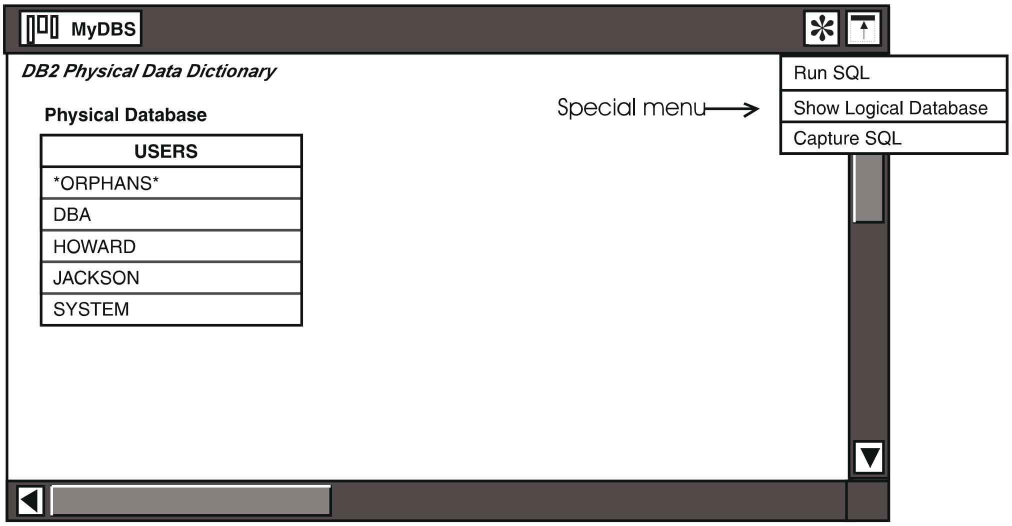
In the physical database, DB2 for MVS, DB2 for Windows, and DB2 for VM convert all user, table, column, and index names to uppercase letters. For example, if you create a physical table named dba_emp in one of these databases, the RDBMS converts the table name to DBA_EMP. The DBA tool does not modify data to conform to the syntax requirements of your RDBMS. For this reason, if you use the DBA tool with an RDBMS that converts physical names to uppercase, you should enter internal names in uppercase to make them consistent with the names stored in the physical data dictionary.

Information about table indexes can be displayed in the physical database. For information about indexes, see “Displaying Table Index Information” on page 69.

Displaying Users

When you first display the physical database, an alphabetical list of the names of the users who created tables in the database opens, as shown in Figure 25 on page 65. These are table owners. The user's selection level is the highest level in the physical database.

The upper left corner of the DBA tool window shows the name of the physical data dictionary for the database being accessed. In Figure 25 on page 65, the name of the data dictionary, DB2 Physical Data Dictionary, indicates that the DBA tool is connected to a DB2 database server.



USERS
ORPHANS
DBA
HOWARD
JACKSON
SYSTEM

Figure 25. The physical database

For SYBASE databases, *ORPHANS* is the first item listed in the physical database.

Displaying User Tables

To display tables owned by a certain user, double-click on a user ID, as shown in Figure 26.

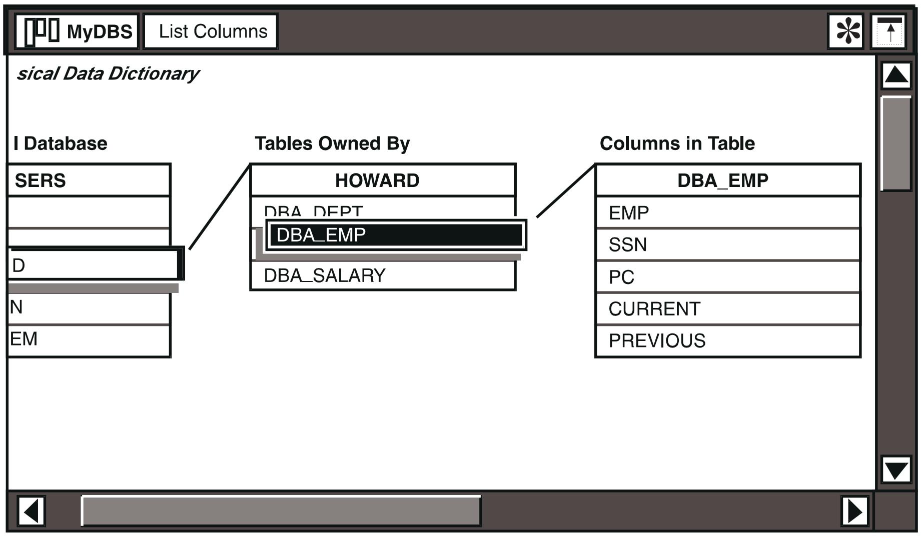


Figure 26. Tables selection level in the physical database

The Tables Owned By heading shows the user ID you selected; in this example, it is HOWARD. The listed tables are owned by the selected user.

At the tables selection level, the window header buttons indicate the functions that you can perform on tables. At this selection level, you can list information in a DBALog window about all the tables that belong to the user, and show table indexes.

You can also delete tables and indexes from the physical database. For information on deleting physical database elements, see “Deleting Elements from the Physical Database” on page 70.

Displaying Columns

Figure 27 shows how the columns in a table owned by a selected user are displayed.

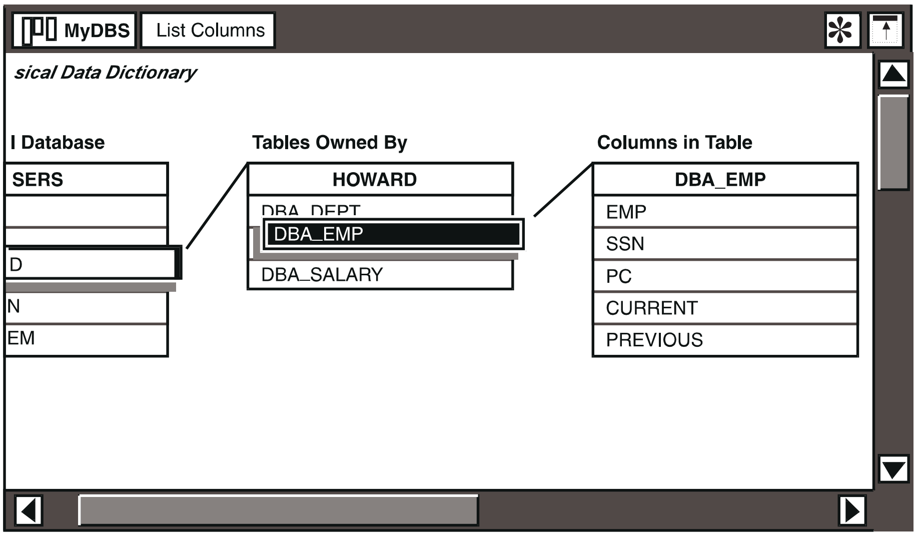


Figure 27. Columns selection level in the physical database

To display the columns in a table:

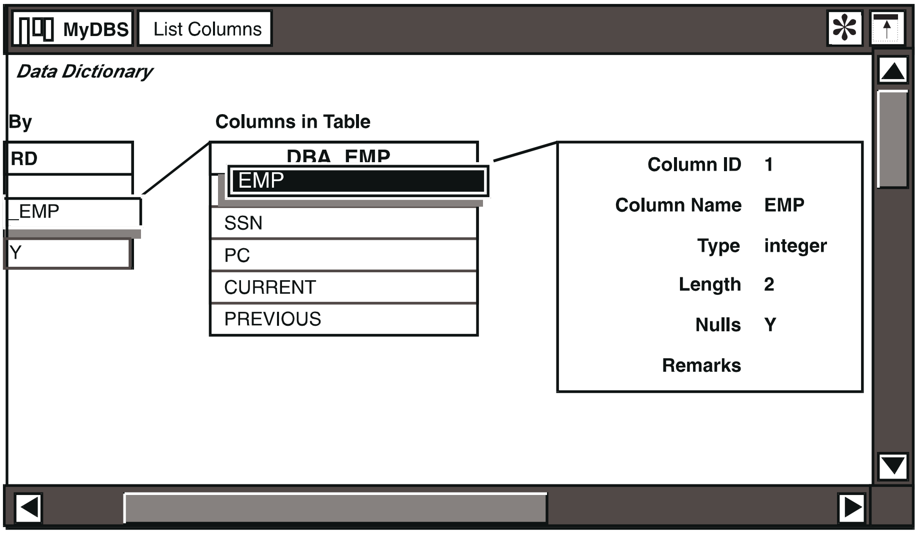
1. Double-click on a user ID.
2. Double-click on a table name.

The Columns in Table heading shows the internal name of a selected table; in this example, it is DBA_EMP. The items listed below the table name are the internal names of the columns in that table.

At the column selection level, the window header buttons indicate the functions that you can perform on columns. At this selection level, you can list information in a DBALog window about the columns in a selected table.

Displaying Physical Column Definitions

Figure 28 on page 68 shows the physical definition of a column for a table in a DB2 database.



Column ID	1
Column Name	EMP
Type	integer
Length	2
Nulls	Y
Remarks	

Figure 28. Physical column definition

The physical definition of a column is displayed in an information box. Table 8 on page 68 describes the fields in the column information box for DB2. The contents of the column definition will vary for different RDBMSs.

Table 8. Physical column information

Field	**Description**
Column ID	The relative position of the column when the table was created.
Column Name	The internal (that is, physical) name of the column.
Type	The data type
Length	The maximum number of bytes required to store the data.
Nulls	Whether nulls can be stored in this column.
Remarks	Remarks provided by the user.

To display the physical definition of a column:

1. Double-click on a user ID.
2. Double-click on a table name.
3. Double-click on a column name.

You might have to scroll the window horizontally to see the entire column definition.

Displaying Table Index Information

You create indexes by running SQL statements in the SQL Control window. For information on how to use the SQL Control window, see “Running SQL statements” on page 37. The SQL statement for creating indexes differs depending on the RDBMS.

Displaying Table Indexes

When you display table indexes, the display contains the name of each of the indexes for that table. To display indexes defined for a table:

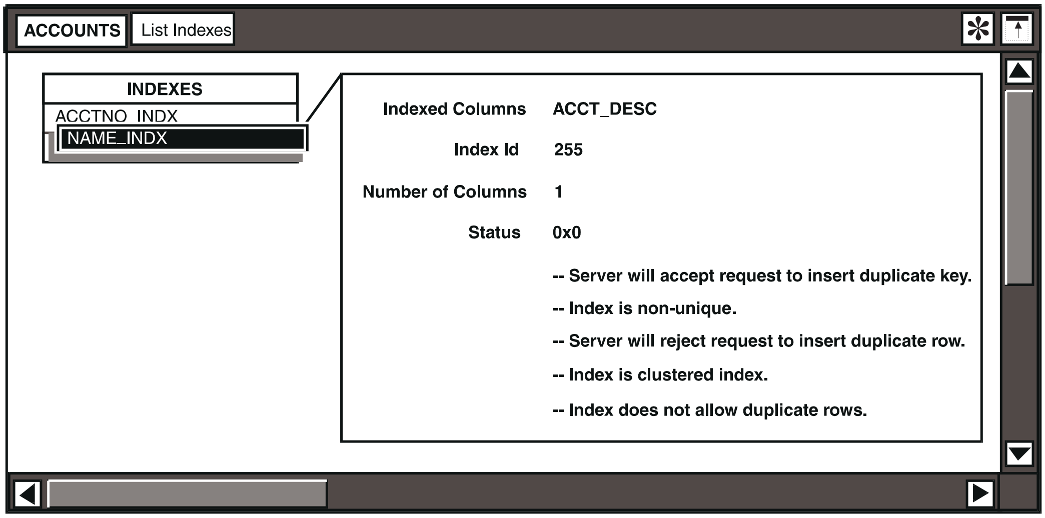
1. Display the tables selection level.
2. Select a table.

   You can show indexes for several tables at once by selecting more than one table. If you selected more than one table, a separate index window opens for each table.
3. Click on the `Show Indexes` button in the window header. An index window opens, listing indexes for that table. The index window takes on the name of the selected table.

Displaying Index Definitions

You can display index definitions in the physical database. Index definitions are displayed in an information box, as shown in Figure 29 for a table in a SYBASE database. Index definitions differ depending on the RDBMS.

To display index definitions, double-click on one of the listed indexes.



Indexed Columns ACCT_DESC

Index Id 255

Number of Columns 1

Status 0x0

-- Server will accept request to insert duplicate key.

-- Index is non-unique.

-- Server will reject request to insert duplicate row.

-- Index is clustered index.

-- Index does not allow duplicate rows.

Figure 29. Example index definition

Table 9 describes the fields shown in this example for a SYBASE database.

Table 9. Example index definition for a SYBASE database

Field	**Description**
Indexed Columns	The column on which the index was created.
Index Id	The ID of the index.
Number of Columns	The number of columns to which the index refers.
Status	Index status is identified in SYBASE databases by a series of hexadecimal representations that correspond to characteristics of each index, such as whether duplicate keys are accepted and if the index is unique.

Listing Index Definitions

You can list index definitions to display or print the indexes defined for various tables.

To list table indexes, click on the `List Indexes` button in an index window.

The DBA tool automatically opens a DBALog window, if one is not already open.

Deleting Elements from the Physical Database

You can delete tables and table indexes from the physical database.

When you delete a table or table index from the physical database, the Confirmation window opens, displaying a message to verify that you want to proceed with the deletion.

You cannot delete users from the physical database using the database depictions in the DBA tool. To add and delete users, you must use the SQL Control window in the DBA tool, the SQL Entry tool, or the SQL Input service in the SA tool to enter SQL statements. The SQL statements to add and delete users vary depending on the RDBMS. Also, you cannot delete columns from a table in the physical database; columns are deleted only through the logical database.

To delete a physical database, select the database element and click on the `Proceed` button. If you do not want to delete the item, close the Confirmation window or click on the `Cancel` button.

Deleting a table from the physical database does not affect the logical database. Because the logical definition of the table is retained, you can delete a physical table and recreate it without having to change the WTDD. Users can also see the logical structure of the deleted table in the Query or Reporter tools, but they cannot display any data from it. If you change a table in the logical database when the physical table exists, the database definitions might not correspond.

Chapter 6. Accessing Databases with the SA Tool

In addition to running SQL statements, the SA Tool provides several functions for performing maintenance tasks on the database. With it, you can start and stop Meta5 services on a host system, obtain status information, control resources and run macros.

This chapter contains a description of how you can use the SA Tool icon to:

- Connect to a database and run SQL statements
- Run macros
- Log your session

You can also use the SA Tool to connect to a database and perform administrative tasks. To do this, see the *Database Gateway Services Guide*.

If an SA Tool icon is not already on the system desktop, you can find one in the Administrator Icons file drawer.

Running SQL Statements with the SA Tool

This section provides instructions on connecting to a database and running SQL statements with the SA Tool. For information on how to use the DBA and SQL Entry tools to run SQL statements, see “Chapter 4. Getting Started With The DBA Tool,” on page 35 and the *Data Access Tools User's Guide*, respectively. These tools also provide the capability to run SQL statements, but they do not let you run macros.

You can enter SQL statements directly into the SA Tool window or copy them into the SA Tool from a text document.

To enter commands directly into the SA Tool window:

1. Open an SA Tool window.
2. Select a realm in the `Realm` field.
3. Select `SQL Input` in the `Service Type` field.
4. In the `Select Service` field, select a service in the `Service` field, or type the name if you know it.
5. Click on the `Connect to Service` button. The SQL Input Service Selection window opens.
6. In this window, provide the name of the gateway and the database to which you want to connect, and the database user ID and password.

7. Click on the `Connect to Service` button to start an SQL input session. If the information you typed was valid, an SQL Input window and an SQL Input Log window open.
8. Type your statement in the SQL Input window, and click on the `Apply` button. The command results are displayed in the SQL Input Log window.
9. Repeat the previous for each statement that you want to run.

If you plan to run several statements, or if your statements are complex, you might want to enter the statements in a text document first and then copy them into the SA Tool window.

To copy commands into the SA Tool, highlight them in the text document and copy them to the SQL Input window.

When you copy commands into the SA Tool from a text document, avoid including return characters from multiple-line commands. These characters create a confusing format in the SQL Input window.

When you run SQL statements:

- You cannot use a reserved word or keyword as the name of a database object.

  To ensure that applications can be transported to other gateways or servers, do not use an extended SQL statement as the name of a database object. See "Appendix A. Reserved Words for DB2 Databases," on page 87 and "Appendix B. Reserved Words for Non-IBM UNIX Databases," on page 93 for a list of the reserved words for DB2 and UNIX databases.
- The SQL interpreter is generally not case sensitive to commands or keywords. Therefore, when entering statements to the gateway, you do not need to preserve the capitalization you find in this book, unless otherwise noted. However, quoted text in the SQL statement is case sensitive.
- To run a statement, click on the `Apply` button in an SQL Input window header.
- To enter multiple-line statements, type a space followed by a hyphen at the end of each line, then press Enter. This indicates that the statement continues on the next line. For example, type a multiple-line statement as follows:

  ```
  SELECT soaps, bleaches, softeners -
  FROM products
  ```
- When using a hyphen to continue a line, be sure to enter a space between the SQL statement and the hyphen. Otherwise, the words concatenate from the end of the first line to the beginning of the next line.
- To stop a currently running SQL statement, press the Stop function key. You might have to press the Stop function key more than once. Pressing the Stop function key breaks the connection to the SQL server and the gateway. To reconnect to the SQL server, close the SQL Input window and use the SQL Input Service Selection window.

Running Macros

By using macros, you can insert run-time parameters in SQL statements to perform lengthy or repetitive SQL sequences for table manipulation and database management. In Meta5, you must call macros from the SQL Input window in the SA Tool. To display the SQL Input window, follow the instructions in “Running SQL Statements with the SA Tool” on page 73.

At the prompt in the SQL Input window, type a macro call statement by using this format:

```
CALL macro_name parameters
```

where *macro_name* identifies the macro you want to run, and *parameters* are parameters you provide as necessary.

The default macro folder that is to be searched is part of the gateway configuration and is specified in the database gateway's Gateway Details window.

Use the same format to call a user-written macro. If a user-written macro is stored inside a folder in the DBA file drawer, separate the folder name from the macro name with a vertical bar, as in this example:

```
CALL folder_name|macro_name parameters
```

Do not include extra spaces before or after the vertical bar. The spaces will affect the macro call.

Macro folders inside the DBA file drawer store Meta5-supplied macros for each database. Each of these macro folders contains a help document that describes its macros and provides syntax examples. Each macro file lists and briefly describes its respective macro parameters. The following macro folders are located in the DBA file drawer:

- `DBS - DB2`
- `DBS - ORACLE7`
- `DBS - OS2`
- `DBS - OS400`
- `DBS - NT-UDB`
- `DBS - NT-ODBC`
- `DBS - SQLDS`
- `DBS - SYBASE`
- `DBS - DB2/6000`
- `DBS - DB2/6000 PE`
- `DBS - DataJoiner`

The message bar at the bottom of the desktop displays each command in the macro as it is run. The SQL Input Log window displays error messages or other responses from the RDBMS.

If you selected a remote realm, then that realm's gateway configuration file is read to determine which macro folder should be read.

Notes on Using Macros

You can include macros and SQL statements as commands in macro files. Meta5 supports nested macros to a depth of eight.

The file server is case sensitive. You must preserve case for macro names, which are lowercase.

Macro parameters are positional. If you omit a parameter in the macro, you must indicate the omission with a comma, which would otherwise delimit that parameter. For example, a macro that takes up to 4 parameters might omit an optional parameter, as shown in the following example:

```
CALL  macro_name  PARAMETER,PARAMETER,,PARAMETER
                                      ↑
                                   omitted
                                   parameter
```

Some macros require you to specify internal names of data categories, tables, and columns. These internal names, unlike print names, cannot include embedded spaces.

To stop a macro, press the Stop function key. You might have to press the Stop function key more than once. Pressing the Stop function key breaks the connection to the SQL server and the gateway. To reconnect to the SQL server, close the SQL Input window and use the SQL Input Service Selection window.

Chapter 7. Transferring Data

To transfer data between databases, you can use a site-specific database load utility (if you have one available).

For smaller amounts of data, you can use Meta5 data access tools to extract, format, and load data into tables. This chapter describes how to:

- Load data into tables using Meta5 tools

This chapter also provides a list of the data types used by Meta5-supported databases and describes how the data types convert to the logical database format that the DBA tool uses.

Loading Data into Tables Using the Query and Spreadsheet Tools

You can update data in a table or add data to table by using the Query and Spreadsheet tools. You use the Spreadsheet tool as a clipboard or template to which you copy table information that you retrieve with the Query tool.

To load data into tables using the Query and Spreadsheet tools:

1. Open a configured Query icon. The Query Catalog window opens. Categories are listed on the left; tables are listed on the right.
2. Click on the `Show DBA Catalog` button in the window header.
3. Select the category in which you want to work. You can work with only one category at a time.
4. Click on the `Show User Catalog` button in the Query Catalog window to return to the user catalog.

   The tables associated with the category you selected are displayed in the user catalog, with their columns, as shown in Figure 30 on page 78.

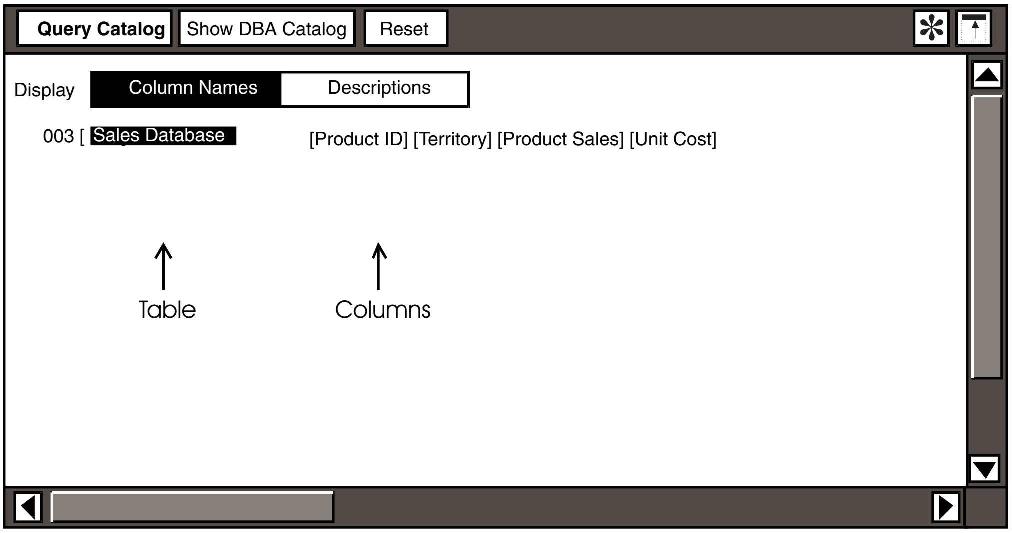


Figure 30. User Catalog in the Query Catalog window

5. Select a table name and copy it into the results section of the Query window. The columns in the table are displayed.

   To learn more about copying information within the Query tool, see the *Data Access Tools User's Guide*.

6. Copy the table information to a spreadsheet in the Spreadsheet tool.

   a. Open a Spreadsheet icon.

   b. Click on the `Show Data` button in the Query window where you copied the table data to display table data.

      If your table contains a large amount of data, you might want to constrain the query to retrieve a smaller amount. To learn more about constraining a query, see the *Data Access Tools User's Guide*.

   c. Click on the `Select All` button in the Query window where you copied the table data, and copy the column information to the opened spreadsheet. Copy the data to the spreadsheet by clicking in the first spreadsheet cell.

      When column names are used in a spreadsheet, they dictate corresponding columns in the target table. If you enter table headings in the same format as your data, the headings are considered as data.

   d. Verify that the data types of the spreadsheet cells are accurate for the type of data in the columns.

      If you modify information directly in the spreadsheet cells, the data types might not be accurate. The Spreadsheet tool recognizes when you key in numbers and defines them as such, even if they were originally character data types. To ensure that data types are accurate, modify entries in the Spreadsheet Options window.

7. Select `Enable Edits` from the `Special` menu in the Query window where you copied the table data.
8. To load data into the table, click on the `Select All` button in the Spreadsheet window and copy it to the table by selecting the table name in the Query Catalog window.
9. Close the Spreadsheet icon.

Mapping Data Types between Supported Databases and Meta5

Because different database management systems support data types that the DBA tool does not, some conversion is necessary when data is transferred to Meta5. The Query, Reporter, Forms Design, and Browser data access tools use the following data types, which the DBA tool supports:

- Integer
- Real
- Character
- Date

Tools that do not use the WTDD might not display data the same way as other tools. For example, the DBA tool normally maps the SYBASE *datetime* data type to *date*. However, the SQL Entry tool does not perform this mapping.

Table 10 on page 79 lists the data types that the DBA tool defines for logical tables when the physical table exists for each of the listed database.

In DB2 database management systems, the DBA tool ignores these data types for logical tables when the physical table already exists:

- fixed graphic
- vargraphic
- long vargraphic
- large object
- very large object

Table 10. Data conversion from RDBMS to Meta5 format

DB2	SYBASE	ORACLE	DBA tool (Meta5 type)
integer, int	int	number (n, 0), where n $\leq$ 9	Integer
smallint	smallint		
	tinyint		

Table 10. Data conversion from RDBMS to Meta5 format

DB2	SYBASE	ORACLE	DBA tool (Meta5 type)
float packed decimal double zoned decimal	float real double precision numeric decimal money smallmoney	number (n, m), where n > 9 or m ≠ 0	Real
char varchar long varchar time timestamp	char nchar varchar nvarchar text timestamp binary varbinary image bit	char long varchar2 rowid date	Character
date	datetime smalldatetime		Date

In Oracle databases, The DBA tool ignores the raw long and raw mislabel data types for logical tables when the physical table already exists.

The DBA tool does not ignore any data types in SYBASE database management systems.

Chapter 8. Desktop Transformers

This chapter describes transformers used to configure Meta5 desktop tools and to display and manipulate WTDD table data from the desktop. For information on other transformers, see the *Transformers Guide*.

The SetDBAccess Transformer

The SetDBAccess transformer searches a container icon for all icons that are used to access database gateways and changes the access options according to your specifications. A container icon is a folder, file drawer, capsule, envelope, or desktop.

You can copy a new SetDBAccess transformer icon to your desktop from the Administrator Icons file drawer on the system desktop in `system|file drawers`.

If users have access to a large number of data access icons, you might want to make the SetDBAccess transformer program available to them. Users can place their icons in a folder and then use the SetDBAccess transformer to globally change the configuration options associated with the group of icons.

The SetDBAccess transformer does not use any input or output regions. Before using the transformer, you must first enter data into the Transformer Controls window.

To open the Transformer Controls window, click on the **Show Controls** button in the window header.

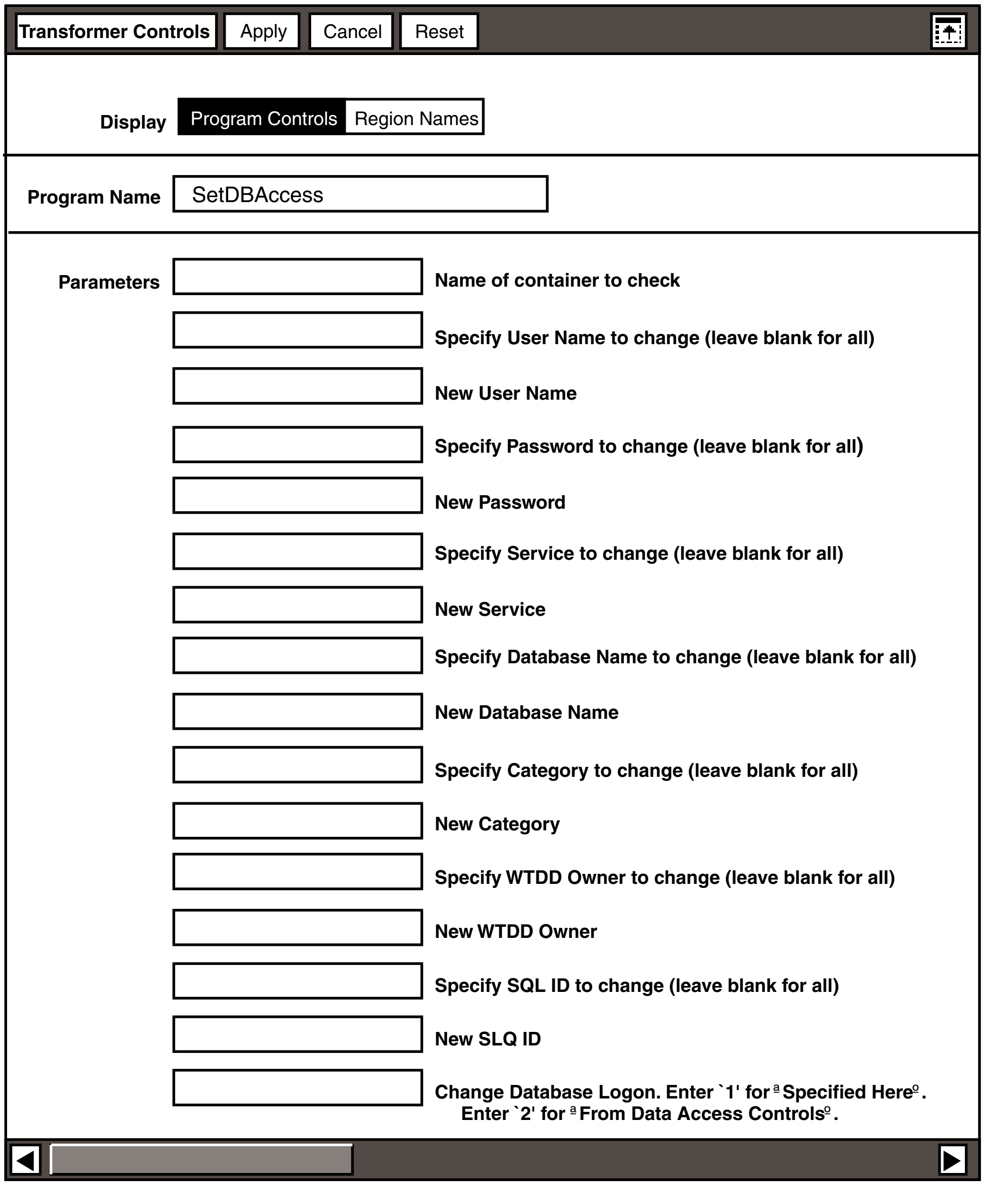
Except for the first and last fields, the fields in the Transformer Controls window, as shown in Figure 31 on page 82, correspond in pairs with the `Other Data` parameter options in the Icon Options window for data access tools.

For example, if you want to change the user ID to Marktng, type the current user name in the `Specify User Name to change` field and then type `Marktng` in the `New User Name` field. If you do not want to change a parameter, leave the appropriate field blank. The SetDBAccess transformer changes only the options for which you specify new values.

In the following situations, the SetDBAccess transformer program cannot set or change the options of data access icons:

- If you do not have permission to change the container
- If an icon is inside a file drawer that is nested inside another file drawer

If you want to change the icons inside a file drawer, the file drawer must be on your desktop. Also, you must enter the name of the file drawer in the `Name of container to check` field in the Transformer Controls window.



Display: Program Controls | Region Names

Program Name: SetDBAccess

Parameters:

- Name of container to check
- Specify User Name to change (leave blank for all)
- New User Name
- Specify Password to change (leave blank for all)
- New Password
- Specify Service to change (leave blank for all)
- New Service
- Specify Database Name to change (leave blank for all)
- New Database Name
- Specify Category to change (leave blank for all)
- New Category
- Specify WTDD Owner to change (leave blank for all)
- New WTDD Owner
- Specify SQL ID to change (leave blank for all)
- New SLQ ID
- Change Database Logon. Enter `1' for ªSpecified Hereº. Enter `2' for ªFrom Data Access Controlsº.

Figure 31. Transformers Controls Window

To specify a container and set or change data access options:

1. In the `Name of container to check` field, enter the name of a folder, file drawer, capsule, or envelope that resides on your desktop.

2. In the remaining fields, enter values to which you want data access options set or changed.

   If you do not want to change a parameter, leave the field blank.
3. Click on the **Run** button in the window header.

The transformer program sets or changes the options of each data access icon that is in the specified container.

To change an option, regardless of its current value:

1. Leave the `Specify parameter` field blank.
2. Type a new value in the `New parameter` field only, where *parameter* is a configuration parameter for data access icons.

Changing Icons on a Desktop

The SetDBAccess transformer lets you change options for:

- All data access icons on a desktop
- All the data access icons on all desktops on a specified disk

To perform either task, you must run the SetDBAccess transformer from the system desktop.

To change all data access icon options for a particular desktop:

1. Open the transformer on the system desktop and click on the **Show controls** button.
2. Type the following path in the `Name of container to check` field:

   `disk|Desktops|desktop`

   where `disk` is the name of the disk containing the desktop to be changed and `desktop` is the name of the desktop to be changed.

   For example, to change all icons on Smith's desktop, where that desktop is on the system disk, type the following path in the `Name of container to check` field:

   `System|Desktops|Smith`

   All icons on Smith's desktop would change except for the icons that are inside file drawers.
3. Run the transformer.

To change all data access icon options on all desktops on a specified disk:

1. Open the transformer on the system desktop and click on the **Show controls** button.
2. Type the following path in the `Name of container to check` field:

   `disk|Desktops`

   where `disk` is the name of the disk containing the desktops to be changed.

For example, to change all icons on the data1 disk, type the following path in the `Name of container to check` field:

```
data1|Desktops
```

Except icons inside file drawers, all icons on desktops that are stored on the data1 disk are changed.

3. Run the transformer.

Changing Icons in File Drawers

Use the SetDBAccess transformer to change options for:

- All data access icons inside file drawers
- All the data access icons in all file drawers on a specified disk

You must run the SetDBAccess transformer from the system desktop.

To change all data access icon options inside file drawers:

1. Open the transformer on the system desktop and click on the **Show controls** button.
2. Type the following path in the `Name of container to check` field:

   ```
   disk|File Drawers|drawer
   ```

   where `disk` is the name of the disk containing the desktops that are to be changed and `drawer` is the name of the file drawer whose desktops are to be changed.
3. Run the transformer.

For example, to change all database access icons in the *Query Icons file drawer on the system disk, type the following path in the `Name of container to check` field:

```
System|File Drawers|*Query Icons
```

All icons that are stored in the *Query Icons file drawer are changed.

To change all data access icon options in all file drawers on a specified disk:

1. Open the transformer on the system desktop and click on the `Show controls` button.
2. Type the following path in the `Name of container to check` field:

   ```
   disk|File Drawers
   ```

   where `disk` is the name of the disk containing the desktops that are to be changed.
3. Run the transformer.

For example, to change all data access icons in all file drawers that are stored on the data1 disk, type:

```
data1|File Drawers
```

All icons that are stored in file drawers on the data1 disk are changed.

Appendix A. Reserved Words for DB2 Databases

The words listed in Table 11 are reserved by the DB2 family of databases and cannot be used for the names of database objects such as databases, tables, rules, defaults, and so on. An X indicates a reserved word in a particular database. Reserved words that are marked with an asterisk are IBM SQL statements, also known as extended SQL statements.

See "Appendix B. Reserved Words for Non-IBM UNIX Databases," on page 93 for reserved words in non-IBM UNIX databases.

Table 11. Reserved words for DB2 family of databases

Reserved Word	DB2 UDB for MVS	DB2 UDB for VM	DB2 UDB for AIX	DB2 UDB for NT
ABORT *	X	X		X
ACQUIRE		X		
ADD	X	X	X	X
ALL	X	X	X	X
ALTER	X	X	X	X
AND	X	X	X	X
ANY	X	X	X	X
AS	X	X	X	X
ASC		X	X	X
AUDIT	X			
AVG		X	X	X
BEGIN *	X	X		X
BETWEEN	X	X	X	X
BIND			X	X
BINDADD			X	X
BIT			X	X
BUFFERPOOL	X			
BY	X	X	X	X

Table 11. Reserved words for DB2 family of databases

Reserved Word	DB2 UDB for MVS	DB2 UDB for VM	DB2 UDB for AIX	DB2 UDB for NT
CASCADE			X	X
CHAR			X	X
CHARACTER			X	X
CLUSTER	X			
COLLECTION				
COLUMN	X	X	X	X
COMMENT		X	X	X
COMMIT		X	X	X
CONNECT		X	X	X
CONSTRAINT			X	X
CONTROL			X	X
COUNT	X	X	X	X
CREATE		X	X	X
CREATETAB			X	X
CURRENT	X	X	X	X
CURSOR	X		X	X
DATA			X	X
DATABASE	X		X	X
DATE			X	X
DBA *		X		
DBADM			X	X
DBSPACE		X		
DEC			X	X
DECIMAL			X	X
DELETE	X	X	X	X
DESC		X	X	X
DESCRIPTOR	X			
DISTINCT	X	X	X	X
DOUBLE			X	X
DROP	X	X	X	X

Table 11. Reserved words for DB2 family of databases

Reserved Word	DB2 UDB for MVS	DB2 UDB for VM	DB2 UDB for AIX	DB2 UDB for NT
EDITPROC	X			
END	X			
ERASE	X			
EXCEPT			X	X
EXCLUSIVE		X	X	X
EXEC	X			
EXECUTE	X		X	X
EXISTS	X	X	X	X
EXPLAIN		X		
FETCH			X	X
FIELDPROC	X			
FLOAT			X	X
FOR	X	X	X	X
FOREIGN			X	X
FROM		X	X	X
GO	X			
GOTO	X			
GRANT	X	X	X	X
GRAPHIC			X	X
GROUP	X	X	X	X
HAVING	X	X	X	X
I		X		
IDENTIFIED		X		
IMMEDIATE	X			
IN	X	X	X	X
INDEX	X	X	X	X

Table 11. *Reserved words for DB2 family of databases*

Reserved Word	DB2 UDB for MVS	DB2 UDB for VM	DB2 UDB for AIX	DB2 UDB for NT
INSERT	X	X	X	X
INT			X	X
INTEGER			X	X
INTERSECT			X	X
INTO	X	X	X	X
IS	X	X	X	X
KEY	X		X	X
LIKE	X	X	X	X
LOCK		X	X	X
LOCKSIZE	X			
LONG			X	X
MAX		X	X	X
MIN		X	X	X
MIXED			X	X
MODE		X	X	X
NAMED		X		
NHEADER		X		
NOT	X	X	X	X
NULL	X	X	X	X
NUMERIC			X	X
NUMPARTS	X		X	
OF	X	X	X	X
ON	X	X	X	X
ONLY			X	X
OPTIONS		X	X	X
OR	X	X	X	X

Table 11. Reserved words for DB2 family of databases

Reserved Word	DB2 UDB for MVS	DB2 UDB for VM	DB2 UDB for AIX	DB2 UDB for NT
ORDER	X	X	X	X
PACKAGE			X	X
PAGE		X		
PAGES		X		
PART	X			
PCTFREE		X		
PCTINDEX		X		
PLAN	X			
PRECISION			X	X
PRIMARY			X	X
PRIQTY	X			
PRIVATE		X		
PRIVILEGES	X	X	X	X
PROGRAM		X	X	X
PUBLIC		X	X	X
REAL			X	X
REFERENCES			X	X
RESOURCE		X		
RESTRICT			X	X
REVOKE		X	X	X
ROLLBACK		X	X	X
ROW		X		
RUN		X		
SBCS			X	
SCHEDULE		X		
SECQTY	X			
SELECT	X	X	X	
SET	X	X	X	

Table 11. Reserved words for DB2 family of databases

Reserved Word	DB2 UDB for MVS	DB2 UDB for VM	DB2 UDB for AIX	DB2 UDB for NT
SHARE		X	X	
SMALLINT			X	
SOME	X			
SQLMACRO *	X	X		X
STATISTICS		X		
STOGROUP	X			
STORPOOL		X		
SUM		X	X	
SYNONYM	X	X	X	
TABLE	X	X	X	X
TABLESPACE	X			
TIME			X	X
TIMESTAMP			X	X
TO	X	X	X	X
TRANSLATE			X	X
UNION	X	X	X	X
UNIQUE			X	X
UPDATE			X	X
USER			X	X
USING				
VALUES			X	X
VARCHAR			X	X
VARGRAPHIC			X	X
VIEW			X	X
WHERE			X	X
WITH			X	X
WORK			X	X

Appendix B.   Reserved Words for Non-IBM UNIX Databases

The words listed in “Appendix B. Reserved Words for Non-IBM UNIX Databases” are reserved by UNIX databases and cannot be used for the names of database objects such as databases, tables, rules, defaults, and so on. An X indicates a reserved word in a particular database. Reserved words that are marked with an asterisk are IBM SQL statements, also known as extended SQL statements.

See “Appendix A. Reserved Words for DB2 Databases,” on page 87 for reserved words in DB2 family of databases.

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
? *	X	
ABORT		
ACCESS		X
ACTION		
ADD	X	X
ADDRESS	X	
ALL	X	X
ALTER	X	X
AND	X	X
ANY	X	X
AS	X	X
ASC	X	X
AUDIT		X
AUTHORIZATION		
AVG	X	
BEGIN	X	
BETWEEN	X	X

Table 12. Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE
BREAK	X	
BROWSE	X	
BULK	X	
BY	X	X
CASCADE		
CHANGE		
CHANGEPATH		
CHAR		X
CHARACTER		
CHECK		X
CHECKPOINT	X	
CLOSE		
CLUSTER		X
CLUSTERED	X	
COBOL		
COLUMN		X
COMMENT		X
COMMIT	X	
COMPRESS		X
COMPUTE	X	
CONFIRM	X	
CONNECT		X
CONSTRAINTS		
CONTINUE	X	
CONTROLROW	X	
CONVERT	X	
COPY IN FILE	X	
COUNT	X	
CREATE	X	X
CUME		

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
CURRENT		X
CURRENT_DATE		
CURRENT_TIME		
CURRENT_TIMESTAMP		
CURSOR		
DATABASE	X	
DATE		X
DATEFORMAT		
DBA	X	
DBCC	X	
DEC		
DECIMAL		X
DECLARE	X	
DEFAULT	X	X
DELETE	X	X
DESC	X	X
DESCRIBE		
DIRECTORY		
DISK	X	
DISTINCT	X	X
DISTRIBUTED		
DOUBLE		
DOUBLEPRECISION		
DROP	X	X
DUMMY	X	
DUMP	X	
ELSE	X	X
END	X	
ERRLVL	X	

Table 12. Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE
ERROR		
ERROREXIT	X	
ESCAPE		
EXCEPT	X	
EXCLUSIVE		X
EXEC	X	
EXECUTE	X	
EXISTS	X	X
EXIT	X	
EXPAND		
EXTRACT		
FETCH		
FILE		X
FILENAME		
FILLFACTOR	X	
FIRST		
FLOAT		X
FOR	X	X
FOREIGN		
FORTRAN		
FROM	X	X
FULL		
GOTO	X	
GRANT	X	X
GROUP	X	X
HAVING	X	X
HOLDLOCK	X	

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
IDENTIFIED		X
IF	X	
IMMEDIATE		X
IN	X	X
INCREMENT		X
INDEX	X	X
INDICATOR		
INHERITING		
INITIAL		X
INNER		
INSERT	X	X
INT		
INTEGER		X
INTERSECT		X
INTO	X	X
IS	X	X
JOIN		
KEY		
KILL	X	
LANGUAGE		
LAST		
LEFT		
LEVEL		X
LIKE	X	X
LINENO	X	
LOAD	X	
LOCK		X
LOGOFF *	X	

Table 12. Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE
LONG		X
MACRO		
MAJOR		
MAX	X	
MAXEXTENTS		X
MAXFILESIZE		
MIN	X	
MINOR		
MINUS		X
MIRROREXIT	X	
MODE		X
MODIFY		X
MODULE		
MOVINGAVG		
MOVINGSUM		
NAME		
NO		
NOAUDIT		X
NOCOMPRESS		X
NONCLUSTERED	X	
NOT	X	X
NULL	X	X
NUMBER		X
NUMERIC		
OF		X
OFF	X	
OFFLINE	X	X
OFFSETS	X	

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
ON	X	X
ONCE	X	
ONLINE	X	X
OPEN		
OPTION		X
OR	X	X
ORDER	X	X
OUTER		
OVER	X	
OVERRIDE		
PASCAL		
PASSWORD	X	
PATTERN		
PCTFREE		X
PERM	X	
PERMANENT	X	
PLAN	X	
PLI		
PRECISION		
PREPARE	X	
PRIMARY		
PRINT	X	
PRIOR		X
PRIVILEGES		X
PROC	X	
PROCEDURE	X	
PROCESSEXIT	X	
PUBLIC	X	X
QUIT		

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
RAISERROR	X	
RANK		
RATIOTOREPORT		
RAW		X
READTEXT	X	
REAL		
RECONFIGURE	X	
REFCHECK		
REFERENCES		
RENAME		X
RESET		
RESOURCE		X
RESTRICT		
RESUME		
RETURN	X	
REVOKE	X	X
RIGHT		
ROLLBACK	X	
ROW		X
ROWCOUNT	X	
ROWID		X
ROWLABEL		X
ROWNUM		X
ROWS		X
RULE	X	
SAVE *	X	
SCHEMA		
SELECT	X	X
SESSION		X

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
SET	X	X
SETUSER	X	
SHARE		X
SHUTDOWN	X	
SIZE		X
SMALLINT		X
SOME		
SPILLAREA		
SQLCODE		
START		X
STARTLOG	X	
STATISTICS	X	
STATUS	X	
STOPLOG	X	
SUBMIT	X	
SUCCESSFUL		X
SUM	X	
SUMMING		
SUPPRESS		
SYB_TERMINATE	X	
SYNONYM		X
SYSDATE		X
TABLE	X	X
TABLESPACE		
TAPE	X	
TEMP	X	
TEMPORARY	X	
TERTILE		
TEXTSIZE	X	
THEN		X

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
THRESHOLD		
TIME	X	
TINYINT		
TO	X	X
TRAN	X	
TRANSACTION	X	
TRIGGER	X	X
TRUNCATE	X	
TSEQUAL	X	
UID		X
UNION	X	X
UNIQUE	X	X
UNLOCK		
UPDATE	X	X
USE	X	
USER		X
VALIDATE		X
VALUES		X
VARCHAR		X
VARCHAR2		X
VIEW		
WAIT		
WAITFOR	X	
WHEN		
WHENEVER		X
WHERE	X	X
WHILE	X	
WITH	X	X

Table 12. Reserved words for UNIX databases

Reserved Word	**SYBASE**	**ORACLE**
WORK		
WRITETEXT	X	

Appendix C. Using Macros to Manage the WTDD

In most cases, you use the DBA tool to interactively work with the WTDD, as described in “Chapter 4. Getting Started With The DBA Tool,” on page 35. However, if you prefer to update tables in batch mode, or if you want to check the updates before they are applied to the WTDD, use the macros available in the DBA file drawer. This appendix lists macros for DB2 and UNIX databases that Meta5 supports.

In some instances, when using a DB2 database, you can use either the macros or the host-based utilities for performing these tasks. For more information, see *Administering Host Services for MVS: DB2 and Cooperative Application Services*.

Table 13 on page 105 lists which macros are available for the different DB2 databases.

Table 13. Macros for IBM databases

Macro name	**DB2 UDB for MVS**	**DB2 UDB for VM**	**DB2 UDB for NT**	**Description**
converttodd3.0	X	X	X	Converts a WTDD that was created by an earlier version of Meta5 into a WTDD Version 3.0.
createcollection				Creates a collection for the specified user.
createsyn	X	X	X	Creates a synonym or view for the named table.
createsynall	X	X	X	Calls other macros to create user synonyms or views.
createsyndbmscat	X	X	X	Creates synonyms for the DB2 for MVS catalog tables for the connected user.
createsynexplain	X	X		Creates a synonym for tables that decode values in a DB2 for MVS PLAN_TABLE table or SQL explain table.
createsynhelp		X		Creates synonyms for the DB2 for VM help tables. Help owner is usually the SQL DBA.
createsynwtdd	X	X	X	Drops and creates synonyms or views for the specified WTDD.

Table 13. Macros for IBM databases

Macro name	DB2 UDB for MVS	DB2 UDB for VM	DB2 UDB for NT	Description
createuserexplain	X	X		Creates a DB2 for MVS PLAN_TABLE table or DB2 for VM explain table for the specified user.
createwtdd	X	X	X	Creates a new WTDD in a Meta5 database.
dropcollection				Removes a collection for the specified user.
dropwtdd	X	X	X	Drops a WTDD for the connected user.
dropwtddspace	X			Drops the specified table space.
install	X	X	X	Creates a WTDD and loads it with descriptions of the DB2 for MVS or DB2 for VM catalog tables.
loaddbmscat	X	X	X	Loads a WTDD with information about the DB2 for MVS or DB2 for VM catalog.
loadexplain	X	X		Loads a WTDD with information about a DB2 for MVS PLAN_TABLE table or DB2 for VM explain table.
loadsample	X		X	Loads a WTDD with information about the DB2 for MVS sample tables.
newuserc		X		Grants connect privileges to an DB2 for VM user with the specified password.
newusercr		X		Grants connect and resource privileges to an DB2 for VM user with the specified password.
wtupdate	X	X	X	Updates the version number of a WTDD.

Macros that create synonyms do so in DB2 for MVS and DB2 for VM only if you are not the owner of the table. WTDD synonyms are necessary only if you do not specify the WTDD owner name in the data access icon's Icon Options window and if you are not the owner of the WTDD. You can also create a WTDD synonym if you write your own SQL statement that accesses the WTDD and if you want to use unqualified WTDD names in those queries. DB2 for AIX synonym macros create views instead of synonyms.

Table 14 on page 107 lists macros for UNIX databases. An X indicates that a macro exists for that particular database. All macro files that are found in the PROTOTYPE folder (for ORACLE database gateways) must be copied to the DBA file drawer and changed.

Table 14. Macros for UNIX databases

Macro name	**SYBASE**	**Red Brick**	**ORACLE**	**DB2 UDB for AIX**	**DataJoiner**	**Description**
converttodd3.0	X	X	X			Converts a WTDD that was created by an earlier version of Meta5 into a 2.1.0 WTDD version.
newdict	X	X	X			Creates WTDD tables and their descriptions.
newvar	X	X	X			Sets a variable value in WT_VARS.
wtddgrant			X			Grants SELECT privileges on the WTDD tables to PUBLIC.
wtddsyn			X			Recreates PUBLIC synonyms for the WTDD tables.
wtgrantall	X	X				Grants INSERT, UPDATE, and DELETE privileges on the WTDD tables to a specified user.
wtgrantread	X	X				Grants SELECT privileges on the WTDD tables to a specified user.
wtgrantwrite	X	X				Grants INSERT, UPDATE, and DELETE privileges on the WTDD tables to a specified user.
wtupdate	X	X	X	X	X	Updates the version number of each WTDD table.

Appendix D. Contents of WTDD Tables

This appendix describes the structure of WTDD tables. The columns in each WTDD table are listed by internal name in the sequence that the DBA tool displays them.

The tables in this appendix also list the print names of columns, as defined in the WTDD. The numbers in parentheses under the data type column are the maximum number of characters or digits permitted for that data type.

WT_CATALOG

The WT_CATALOG table contains information about database categories. This table has one row for each category. Table 15 lists and describes each of the columns in WT_CATALOG.

Table 15. WT_CATALOG columns

Internal name of column	Print name of column	Data type	Description
CATEGORY	Internal Category Name	Char (18)	Internal name of the category.
PRINTNAME	Category Print Name	Char (30)	Print name of the category as it is displayed in data access tools.
DORDER	Display Order	Integer (4)	The order within the logical database in which the category is displayed.
DESCRIPTION	Description of Category	Char (200)	Text description of the category.

WT_COLUMNS

The WT_COLUMNS table contains information about the columns in each of the tables and views in the logical database. This table has one row for each column in each table and view that are defined in the WTDD. Table 16 on page 110 lists and describes each of the columns in WT_COLUMNS. The physical order of the columns in WT_COLUMNS might not match the logical order that is shown in Table 16 on page 110.

Table 16. WT_COLUMNS columns

Internal Name of Column	Print Name of Column	Data type	Description
TABLENAME	Internal Table Name	Char (80)	Internal name of the table containing this column.
COLUMNNAME	Internal Column Name	Char (18)	Internal name of the column. The Browser tool supports up to 15 characters.
SQLNAME	SQL Name	Char (80)	Internal name of the column or view. Applications reference SQLNAME instead of COLUMNNAME for portability to other user environments and databases.
PRINTNAME	Column Print Name	Char (30)	Print name of the column as it is displayed in depictions in the workstation tools.
DATATYPE	Data Type	Integer (4)	Type of data in this column: **0**: Integer **1**: Real (floating point) **2**: Character **3**: Date
PRECISION	Field Width	Integer (4)	If DATATYPE is Integer, Real, or Character, this field contains the maximum number of digits or characters to display. If DATATYPE is date, this field contains 0.
FRACTION	Fraction Size	Integer (4)	For real (floating-point) numbers, the number of digits to the right of the decimal point.

Table 16. WT_COLUMNS columns

Internal Name of Column	Print Name of Column	Data type	Description
RESOLUTION	Date Interpretation	Integer (4)	If DATATYPE is date, the date is displayed to the nearest unit specified by the following code: **0**: Day **1**: Week **2**: Quad week **3**: Month **4**: Odd bimonth **5**: Quarter **6**: Year **7**: Even bimonth
KEYUSAGE	Key?	Integer (2)	Specifies whether this column is part of the primary key for this table: **0**: Not part of primary key **1**: Part of primary key
DOMAIN	Domain	Integer (4)	Number of the domain to which the column belongs. Domain numbers must be unique. Domain 0 (the universal domain) is the default domain number. Domain numbers 1 through 99 are reserved for system use.
NAVALUE	N/A Value	Char (30)	If an N/A value is used, this column has the N/A value, and the workstation tool displays N/A instead of the value.

Table 16. WT_COLUMNS columns

Internal Name of Column	Print Name of Column	Data type	Description
NAUSED	N/A Used?	Integer (2)	Specifies whether an N/A value is used with this column: **0**: N/A value is not used **1**: N/A value is used
DORDER	Display Order	Integer (4)	The order within the table in which the column is displayed.
DESCRIPTION	Description of Column	Char (200)	Text description of the column.

WT_CONTENTS

The WT_CONTENTS table lists which tables and views belong to each database category. WT_CONTENTS has one row for each table or view in a database category. Table 17 on page 112 lists and describes each of the columns in WT_CONTENTS.

Table 17. WT_CONTENT or WT_CONTENTS columns

Internal name of column	Print name of column	Data type	Description
TABLENAME	Internal Table Name	Char (80)	Internal name of the table or view (the WTDD does not distinguish between tables and views).
CATEGORY	Internal Category Name	Char (18)	Internal name of the category.
DORDER	Display Order	Integer (4)	The display order of this table within the category.

WT_DCONSTR

The WT_DCONSTR table contains information about default constraints on columns. This table has one row for each column with a default constraint. Only the Reporter tool uses this information. Table 18 on page 113 lists and describes each of the columns in WT_DCONSTR.

Table 18. WT_DCONSTR columns

Internal name of column	Print name of column	Data type	Description
TABLENAME	Internal Table Name	Char (80)	Internal name of the table or view (the WTDD does not distinguish between tables and views).
COLUMNNAME	Internal Column Name	Char (18)	Internal name of the column. The Browser tool supports up to 15 characters.
CONSTROP	Constraint Operator	Integer (2)	Specifies the operator used in the constraint: **0**: Equal **1**: Not equal **2**: Less than **3**: Greater than **4**: Less than or equal to **5**: Greater than or equal to
CONSTRVAL	Constraint Value	Char (50)	Specifies the actual value to use for the default constraint. It is stored as a character string, regardless of the data type of the column.

WT_DJOINS

The WT_DJOINS table contains information about how the tables and views are joined. This table has one row for each default join. Table 19 on page 113 lists and describes each of the columns in WT_DJOINS.

Table 19. WT_DJOINS columns

Internal name of column	Print name of column	Data type	Description
TABLE1	First Table	Char (80)	Internal name of the table or view that contains the first column.
COLUMN1	Internal Column Name 1	Char (18)	Internal name of the first column.

Table 19. WT_DJOINS columns

Internal name of column	Print name of column	Data type	Description
TABLE2	Second Table	Char (80)	Internal name of the table or view that contains the second column.
COLUMN2	Internal Column Name 2	Char (18)	Internal name of the second column.
JOINTYPE	Join Type	Integer (2)	The type of join: **0**: Not used by the Reporter tool **1**: Used by the Reporter tool
JOINOP	Join Operator	Integer (2)	Specifies the operator used in the join constraint: **0**: Equal **1**: Not equal **2**: Less than **3**: Greater than **4**: Less than or equal to **5**: Greater than or equal to
INOUT	Inner/Outer	Integer (2)	Specifies whether the join is an inner or outer join: **0**: Inner join (default) **1**: Currently not supported **2**: Currently not supported **3**: Currently not supported Outer join is not supported.

WT_DOMAINS

The WT_DOMAINS table contains text descriptions for each domain number that is identified by the database administrator. This table has one row for each domain description. Table 20 on page 115 lists and describes each of the columns in WT_DOMAINS.

Table 20. WT_DOMAINS columns

Internal name of column	Print name of column	Data type	Description
DOMAIN	Internal Domain Number	Integer (4)	Number of the domain.
PRINTNAME	Print Name	Char (30)	Display name of the domain as it is displayed in workstation tools.
DOMAINID	External Domain ID	Integer (4)	External identifier for the internal domain number.
DESCRIPTION	Description of Domain	Char (200)	Text description of the significance of the domain.

WT_FUNDEP

The WT_FUNDEP table contains the functional dependencies between columns within a database. Table 21 lists and describes each of the columns in WT_FUNDEP for your information only.

Table 21. WT_FUNDEP columns

Internal name of column	Print name of column	Data type	Description
IDCOL	Identifying column	Char (18)	Internal name of identifying column.
IDTBL	Identifying table	Char (30)	Internal name of the table containing the named column.
DEPCOL	Dependent Column	Char (18)	Internal name of the functionally dependent column.
DEPTBL	Dependent table	Char (80)	Internal name of the table containing the functionally dependent column.

WT_TABLES

For a user to access a table or view in one of the workstation tools, the WTDD must include both the print and internal names of the table or view. The WTDD does not make a distinction between tables and views. You define both to the WTDD as if they were tables. If you want the Reporter tool to use the table or view, you must also indicate in the WTDD whether the table or view contains fact or dimension information.

The WT_TABLES table has one row for each table or view in the logical database. Table 22 lists and describes each of the columns in WT_TABLES. The physical order of the columns in WT_TABLES might not follow the logical order that is shown in Table 22.

Table 22. WT_TABLES columns

Internal name of column	Print name of column	Data type	Description
TABLENAME	Internal Table Name	Char (80)	Internal name of the table or view (the WTDD does not distinguish between tables and views).
SQLNAME	SQL Name	Char (80)	Internal name of the table or view. Applications reference SQLNAME instead of TABLENAME for portability to other user environments and databases.
TABLETYPE	Table Type	Integer (2)	How the table or view will be used: **0**: Table or view cannot be used by the Reporter tool **1**: Table or view is a reporter fact table **2**: Table or view is a Reporter dimension table
VERSION	Version	Integer (4)	The version number of the table or view.
PRINTNAME	Table Printname	Char (30)	Print name of the table or view as it is shown in depictions in workstation tools.
DESCRIPTION	Description of Table	Char (200)	Text description of the table.

WT_TTBLNAMES

The WT_TTBLNAM and WT_TTBLNAMES tables determine which roots for temporary tables are checked out. Table 23 lists and describes each of the columns in this table.

Table 23. WT_TTBLNAM or WT_TTBLNAMES columns

Internal name of column	Print name of column	Data type	Description
ROOTNAME	Root Name	Char (15)	First 10 characters of the temporary table name.
USERID	User Id	Integer (9)	Workstation user name that checked out the root.
CREATETIME	Create Time	Date (Day)	Date when the root was checked out.
NETADDRESSA and NETADDRESSB	Net Address Part A and Net Address Part B	Integer (9)	Net address of the workstation that checked out the root (16 bits and 32 bits).
CREATEDATE	Create Date	Char (80)	Date when the root was checked out, in string format.
DESKTOPNAME	Desk Top Name	Char (80)	Workstation user name that checked out the root.
DBUSERNAME	Data Base User Name	Char (30)	Database user name that checked out the root.

WT_VARS

The WT_VARS table is used to assign values to special Meta5 system variables. The variables are designed to support RDBMS-specific features, or to customize data handling for a particular database. Only a few table entries are required for most gateways or databases to work properly with Meta5.

Currently, several gateways can determine the type of RDBMS they are connected to by checking the value of the variable DBTYPE. The installation macros for these gateways insert the necessary row into WT_VARS. Table 24 lists and describes each of the columns in this table.

Table 24. WT_VARS columns

Internal name of column	Print name of column	Data type	Description
NAME	Variable Name	Char (32)	The variable name. All variable names must consist of uppercase roman letters and numbers. The first character must be alphabetic.
Type	Variable Type	Char (32)	The data type of the variable. A value of M4INT4 indicates the variable has a numeric value. A value of MSTRING indicates the value of the variable is a string.
LONGVAL	Numeric Value	Integer (4)	If the value of TYPE is M4INT4, this column has the value of the variable named in the NAME column. Otherwise, this column is not referenced and should be Null or blank.
CHARVAL	Character Value	Char (32000)	If the value of TYPE is MSTRING, this column has the value of the variable named in the NAME column. Otherwise, this column is not referenced and should be Null or blank.

WT_VERSIONS

The table WT_VERSIONS contains the version number of every WTDD table. WT_VERSIONS should have only one row. Table 25 on page 118 lists and describes each of the columns in this table.

Table 25. WT_VERSION or WT_VERSIONS columns

Internal name of column	Print name of column	Data type	Description
WT_CATALOG	Catalog Version	Integer (9)	Version number of information in WT_CATALOG.
WT_CONTENTS	Contents Version	Integer (9)	Version number of information in WT_CONTENT or WT_CONTENTS.

Table 25. WT_VERSION or WT_VERSIONS columns

Internal name of column	Print name of column	Data type	Description
WT_TABLES	Tables Version	Integer (9)	Version number of information in WT_TABLES.
WT_COLUMNS	Columns Version	Integer (9)	Version number of information in WT_COLUMNS.
WT_DOMAINS	Domains Version	Integer (9)	Version number of information in WT_DOMAINS.
WT_DJOINS	Djoins Version	Integer (9)	Version number of information in WT_DJOINS.
WT_DCONSTR	Dconstr Version	Integer (9)	Version number of information in WT_DCONSTR.
WT_VERSIONS	Versions Version	Integer (9)	Version number of information in WT_VERSIONS.
WT_TTBLNAMES	Temporary Tables Version	Integer (9)	Version number of information in WT_TTBLNAM or WT_TTBLNAMES.
WT_FUNDEP	Functional Dependencies Version	Integer (9)	Version number of information in WT_FUNDEP.
WT_VARS	Variable Version	Integer (9)	Version number of WT_VARS table.
DD_VERSION	WTDD Version	Char (8)	Version number of the WTDD.

Important:

If WT_VERSIONS has more than one row of data, the `Ambiguous version number` error message is displayed in the Important Message window when any data access icon is opened. If the rows are identical, all of the identical rows are deleted. Therefore, before you delete excess rows, note the column values so you can re-enter a single row with the same values.

Appendix E. Using the SQLMACRO Facility

The SQLMACRO facility permits conditional processing of SQL statements (if...then...else). It also lets you check SQL return codes and create WTDD tables. Gateway macros do not provide this level of flexibility.

If an SQL statement is generated by a gateway macro, its result must be sent over the network. Gateway macros with SQL statements, such as those that are required for building a WTDD, can be inefficient. The SQL macros that are controlled by the SQLMACRO facility reside in a file on the host system. The location of the file depends on the type of system. For MVS, the file is a member of the met.SQLMACRO library. You can also view the SQLMACRO files by looking in the SQLMACRO folder that is configured in the DBS-NT-UDB folder in the DBA file drawer.

While you must run gateway macros from the Meta5 SA tool, you can use the SQLMACRO facility to run macros from various tools. You can enter the SQLMACRO command from any Meta5 tool where you can enter SQL statements. You can also run the macros that are processed by the SQLMACRO command in batch mode on the MVS host system. For a detailed description of how to do this, see *Administering Host Services for MVS: DB2 and Cooperative Application Services*.

The syntax of the SQLMACRO command that Meta5 issues is:

```
SQLMACRO macro_name parm_1 = value_1 ...parm_n = value_n
```

`macro_name`	The file name of a macro.
`parm_1`	The name of the first parameter.
`value_1`	The value of the first parameter.
`parm_n`	The name of the *n*th parameter.
`value_n`	The value of the *n*th parameter.

Some macro names begin with DSN, SQL, ARI, or QSQ. Macros with these names are run on DB2 for MVS, DB2 for Windows, DB2 for VM, and DB2 for AIX systems, respectively. Macros that exist for systems other than the one you are running allow you to run SQL on those systems through DRDA connections.

Glossary

This glossary defines terms that are used in this book and throughout the Meta5 library. If you do not find the term you are looking for, see the index of this book.

bimonth. A two-month period.

category. The highest-level division of a logical database. A category contains tables, and tables contain columns and rows. Logical joins between tables are defined at the category level.

column. A vertical field in a table. Columns can be identified by letters or text strings at the top of the display area.

database administrator (DBA). The person who issues administrative instructions to a database service. The database administrator defines the data tables used by data access tools, monitors database performance, and assigns user access.

data table. A table containing the internal or physical table names. The database administrator creates and maintains these tables. Also called user tables.

data type. The type of information represented by data in a database column. Data types generally available in the Meta5 logical tables are integer, real, character, and date. Other data types available in some Meta5 logical tables include null, long varchar, varchar, and time.

DBA. Database administrator.

DBA catalog. Database administration catalog. A database listing in the Query tool of all categories and tables available on a given relational database management system (RDBMS) through Meta5.

DB2 for AIX. DATABASE 2 for AIX, an IBM relational database management system that runs on an AIX system.

DB2 for MVS. DATABASE 2 for MVS, an IBM relational database management system that runs on an MVS system.

DB2 for OS/2. DATABASE 2 for OS/2, an IBM relational database management system that runs on OS/2 systems.

DB2 for VM. DATABASE 2 for VM, an IBM relational database management system that runs on VM systems.

default constraint. A constraint that is usually applied to a column. A default constraint is automatically applied in the Reporter tool unless a user changes it.

default join. A join that the database administrator has defined as part of the database design process and defined in the WTDD.

dimension. Data (such as periods of time, products, and locations) defining the context of facts. Dimensions typically make up the column or row headings in a report.

Distributed Relational Database Architecture (DRDA). An IBM architecture that allows SQL data requests between relational data sources within a standard protocol. The database becomes a coordinator to SQL requests to other DRDA data sources.

domain. In a database management system, a set of all acceptable values that can be assigned to a column. For example, a domain for a column called Age in a table called Employee might consist of integers between 16 and 70.

domain number. An integer assigned to each member of a domain.

DRDA. Distributed Relational Database Architecture.

element. Components, such as categories, columns, and tables, that make up a database.

even bimonth. A division of the calendar year into two-month periods ending in an even-numbered month: JF, MA, MJ, JA, SO, ND.

fact. Data consisting of measures such as unit price or volume. Facts typically make up the returned data in a report.

Icon Options window. A window in which the user can reset values or view information about an icon. Each icon has its own Icon Options window. The content of this window varies depending on the icon.

index. A list of pointers in a file used to locate records in a database.

inner join. A join, or intersection, between two tables in which rows with equal or identical values for the joined columns from both tables are listed in the results.

internal data dictionary. The system tables in the WTDD, whose primary function is management of data at the physical level. Also called the physical data dictionary.

internal name. In the Meta5 environment, the name by which a database item is stored in the WTDD. Each database category, table, and column has an internal name based on naming conventions required by the database system. To make these names more useful to users, each internal name has a corresponding external name, called a print name, that the Meta5 tool user sees on the screen. For example, a table with the internal name DBA_EMP might be known to the user as the Employee table.

join. The operation of retrieving data from two or more tables with matching joined columns in a relational database. In Meta5 windows, a join between tables is usually represented by a line with an equal sign (=) in the middle that joins two tables through a common field.

key column. A component of a *primary key.* In the DBA tool, a column can be designated as a primary key or as part of a primary key.

local database. The RDBMS that is running on the same system as Host Services or the DB2 for OS/2 gateway.

local network. The network to which a specified user or resource is directly connected.

logical database. A database representation that includes print names for tables and columns, and that groups the tables into useful categories. The logical database also contains definitions for default joins between tables. The WTDD stores the description of the logical database, the mapping of print names to internal names in the physical database, and other logical attributes, such as the type and length of data.

macro. An Meta5 text document that contains commands or calls to other macros and that can contain substitutable values that are passed during macro execution.

nonsense join. Two columns that are joined when it makes no sense to do so other than because they have a common characteristic, such as the same data type.

odd bimonth. A division of the calendar year into two-month periods ending in an odd-numbered month: FM, AM, JJ, AS, ON, DJ.

phantom. A catalog, table, or column that has been deleted from or changed in the logical database, but whose name remains in the user catalog or depiction area of the Query tool with an overstrike through it to alert users that the item no longer exists.

physical database. The physical set of stored data tables that underlies a logical database.

physical data dictionary. A data dictionary that the RDBMS uses to manage relational data. It contains definitions of the names and locations on disk of all the tables in its database, the names of all columns in the tables, the indexes that are defined on the tables, and any other information necessary to manage the relational data. Each RDBMS uses its own data dictionary to define the physical data.

pointer. A character or group of characters that serves as a link or indicator to the storage location of a data item.

primary key. In Meta5, the column or columns chosen through the DBA tool to uniquely identify each row in a database table. A primary key can be composed of one or more key columns needed to make the primary key unique.

print name. The name of a database item (such as a column or table) that is used in Meta5 windows instead of the more cryptic name by which the data item is identified in the database. Print names are identified in the WTDD and are converted during data retrieval to internal names that can be understood by the database.

quad week. A period of time lasting exactly four weeks, starting on a Sunday.

RDBMS. Relational database management system.

relational database. A database that is organized and accessed according to relationships between data items. These relationships are expressed as tables related by data values, rather than by pointers or memory locations.

relational database management system (RDBMS). The software that makes it possible to create, use, and maintain relational data in the form of tables consisting of rows and columns of data.

remote database. An RDBMS that is accessed through a local database through DRDA.

remote network. A network other than the local network, but which is accessible to the local network by some means.

row. (1) The horizontal component of a table. A row contains exactly one value for each column of the table. (2) A record, or single instance, in a database table.

selection level. The level of the database that is currently being displayed. For example, categories, tables, or columns.

server. The computer on which a service runs.

service. A software process that provides facilities to other parts of Meta5, such as a file service, a mail service, a mail gateway service, and a data access service.

Special menu. A menu that is displayed when the user points to the **Special** button. In tool windows, the menu choices are usually reserved for special functions or controls associated with the icon.

SQL. Structured Query Language.

SQL name. The table or column name that Meta5 data access tools use for generating SQL statements. The SQL name is used to avoid phantoms in database queries. If an SQL name is not provided, then data access tools use the internal name.

Structured Query Language (SQL). A command language for use with relational databases. The language consists of statements to insert, update, delete, query, and protect data. Meta5 tools use SQL statements to communicate with database servers.

system tables. Tables that are generated by the system and contain information that describes the organization of the physical database. The RDBMS maintains the system tables.

table. In a database, a single file, consisting of data organized in columns and rows.

two-way outer join. A join between two tables in which all rows that do not have a common value for the joined columns are listed in the results.

transformer. A tool that can be used to perform various types of data manipulation either interactively or in a capsule.

unique index. Within the DBA tool, a type of index in which pointers are used to locate records in a database by key field where the field has a distinct value for each record.

user table. Data table.

view. A view is a subset of a table or joined tables and is treated by the DBA tool like a standard table. Views are created to present data from two or more tables as if the data is in one table; views can also display part of the data in one table, allowing you to omit fields that contain sensitive information.

Workstation Tools Data Dictionary (WTDD). A logical data dictionary that standardizes the Meta5 desktop view of data across many relational database management systems. The WTDD stores the print names of database items, defines how the items are to be displayed, and includes other logical attributes of data. The WTDD is implemented as a set of standard relational tables that can be managed through the DBA tool.

WTDD. Workstation Tools Data Dictionary.

Meta5 Publications

This section lists Meta5 publications. To order copies of the books listed here, or to get more information about a book, see your Meta5, Inc. representative.

- **Meta5 Volume 1**
  - *Getting Started with the Meta5 Developer's Desktop*
  - *PC Integration Tools*
- **Meta5 Volume 2**
  - *Spreadsheet User’s Guide*
  - *Plot User’s Guide*
- **Meta5 Volume 3**
  - *Text User’s Guide*
  - *Layout User’s Guide*
- **Meta5 Volume 4**
  - *Capsule User's Guide*
  - *BASIC Tool User’s Guide*
- **Meta5 Volume 5**
  - *Data Access Tools User’s Guide*
- **Meta5 Volume 6**
  - *Forms User’s Guide*
- **Meta5 Volume 7**
  - *Transformers Guide*
- **Meta5 Volume 8**
  - *Error Messages and Codes*
- **Meta5 Volume 9**
  - *Installing Meta5 LAN Components*
  - *System Administration Guide and Reference*
- **Meta5 Volume 10**
  - *Database Gateway Services Guide*
  - *Administering Databases with Meta5*
- **Meta5 Volume 11**
  - *Installing and Configuring Meta5 Open Clients*
  - *Developing Applications with Open Data Access Service*
- **Meta5 Volume 12**

Administering Host Services for MVS:DB2 and Cooperative Application Services

Planning and Installing Host-to-LAN Communications for MVS

Index