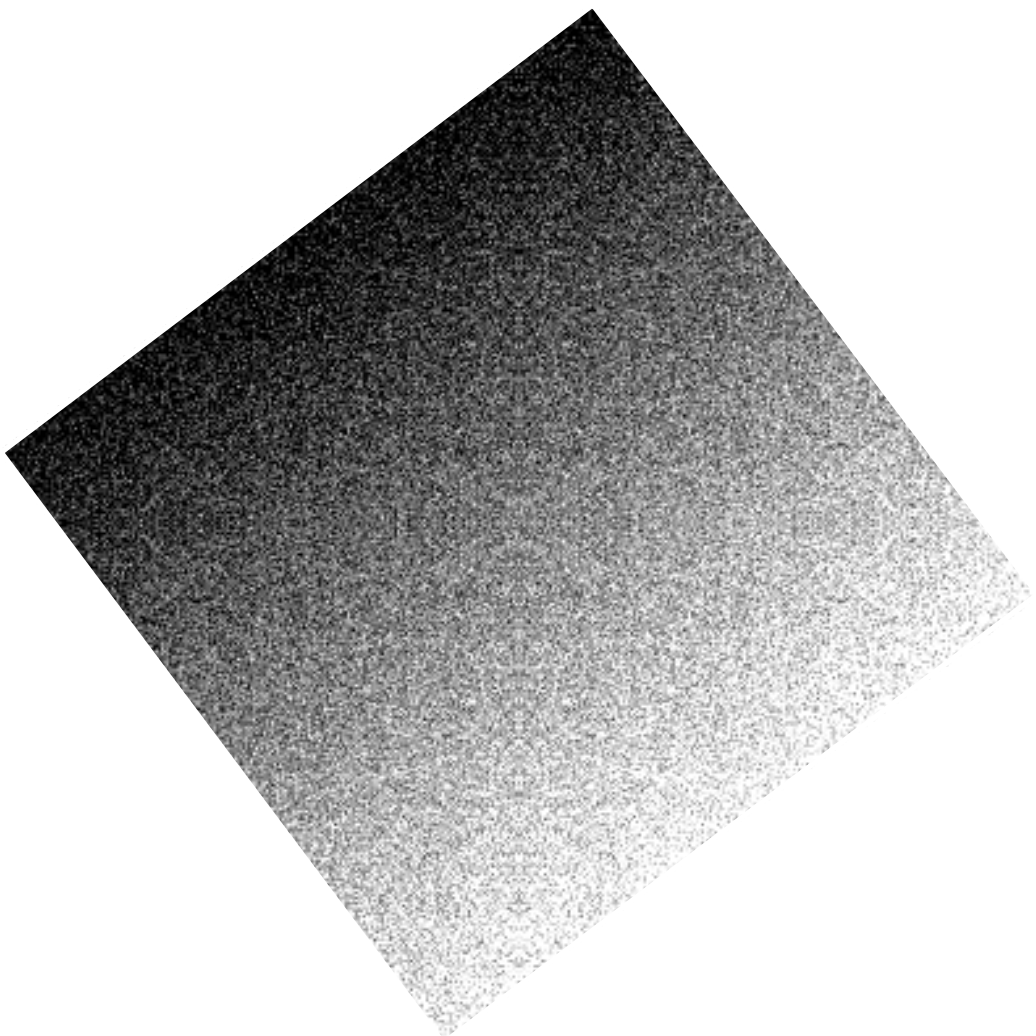
IBM Intelligent Decision Server

IBM

Administering Databases with IDS

Version 1



SC26-8891-00

IBM Intelligent Decision Server

IBM

Administering Databases with IDS

Version 1

SC26-8891-00

Note

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

First Edition (October 1996)

This edition applies to Version 1 of Intelligent Decision Server, 5639–IDS, 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 IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1996. All rights reserved.**
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Notices vi
Trademarks vi

About This Book viii
Who Should Read This Book viii
Prerequisites viii
We'd Like Your Comments viii

Chapter 1. Introducing IDS Database Administration 1
Database Administration Tasks 1
Database Administration Tools 1
DBA Tool 1
SA Tool 2
Query Tool 2
Spreadsheet Tool 2
Forms Design and Data Entry Tools 2
Characteristics of an IDS Logical Database 2
How Data Access Tools Retrieve Data 3

Chapter 2. Working with the WTDD 4
The WTDD 4
Logical and Physical Database Representations in the DBA Tool 4
WTDD Tables 6
WTDD Tables That Contain Category and Column Information 6
Creating WTDD Tables 7
DB2 for MVS 7
SQL/DS 8
DB2 for OS/2, DB2 for AIX, and DataJoiner 8
SYBASE, Red Brick, and ORACLE 8
WTDD Database Elements 9
Categories, Facts, and Dimensions 9
Keyed Columns 11
Default Joins 11
N/A Values 12
Default Constraints 12
Version Numbers 12
Handling Views 12
Modifying the WTDD 13
Effect on Reporter Icons 13
Effect on Query Icons 13
Backing Up and Restoring WTDD Tables 14

Chapter 3. Transferring Data 16
Transferring Data into Tables Using the SAVE FOR AT Command 16
Loading Data into Tables Using IDS Tools 17

© Copyright IBM Corp. 1996

Mapping Data Types between Supported Databases and IDS 18

Chapter 4. Authorizing Users 20
Configuring a Data Access Tool 20
Setting Parameters for Data Access Tools 21
Specifying a Gateway and a Database 21
Accessing a DB2 for AIX Database 24
Accessing a DataJoiner Gateway 25
Specifying a Category 25
Specifying a WTDD Owner 25
Specifying an SQL ID 26
Specifying a User Name and Password 26
Using a Single ID or Multiple (Group) IDs 26
IDS Security Features for Databases 26
The Data Access Controls Window 27
Securing a Data Access Tool Icon 27

Chapter 5. Accessing Databases with the SA Tool 29
Connecting to a Database 29
Running SQL Statements 30
Running Macros 32
Notes on Using Macros 33

Chapter 6. Desktop Transformers 34
The SetDBAccess Transformer 34
Changing Icons on a Desktop 35
Changing Icons in File Drawers 36
The Metadata Exchange Transformers 37
Configuring MDExport 38
Configuring MDImport 38
Metadata Transformers within a Capsule Environment 38

Appendix A. Reserved Words for DB2 Databases 40

Appendix B. Reserved Words for Unix Databases 45

Appendix C. Using Macros to Manage the WTDD 54

Appendix D. Contents of WTDD Tables 57
WT_CATALOG 57
WT_COLUMNS 57
WT_CONTENTS 59
WT_DCONSTR 60
WT_DJOINS 61
WT_DOMAINS 62
WT_FUNDEP 62
WT_TABLES 62
WT_TTBLNAMES 63

WT_VARS 64
WT_VERSIONS 65

Appendix E. Using the SQLMACRO Facility 66

Glossary 67

IDS Publications 69

Index 70

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

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

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling (1) the exchange of information between independently created programs and other programs (including this one) and (2) the mutual use of the information that has been exchanged, should contact:

IBM Corporation
555 Bailey Avenue, W92/H3
P.O. Box 49023
San Jose, CA 95161–9023

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

Trademarks

The following terms are trademarks of IBM Corporation in the United States or other countries or both:

ACF/VTAM	DXT	PROFS
AIX	Enterprise System/9000	PS/2
Application System/400	ES/9000	QMF
APPN	IBM	RACF
AS/400	MVS/ESA	RISC System/6000
Capsule	MVS/XA	RS/6000
DATABASE 2	OfficeVision	S/390

© Copyright IBM Corp. 1996

DataJoiner	OfficeVision/VM	SQL/DS
DB2	Operating System/2	System/360
DB2 for OS/2	Operating System/400	System/370
DB2/6000	OS/2	Virtual Machine/Enterprise Systems Architecture
DFSORT	OS/400	VM/ESA
Distributed Relational Database Architecture	Personal System/2	VTAM
DRDA	Presentation Manager	Win-OS/2

C-bus is a trademark of Corollary, Inc.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Microsoft, Windows, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

Other company, product, and service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

About This Book

This book describes database administration tasks and concepts with respect to the Intelligent Decision Server (IDS).

Who Should Read This Book

This book is intended for the use of IDS 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.
- Configuring Data Access tools for users.
- Using the 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:

- IDS desktop functions and capabilities, as described in *Working with the Intelligent Decision Server*.
- Structured Query Language (SQL).

You must also be familiar with the DBA, SA, Query, Reporter, Forms Design, Data Entry, SQL Entry, and Spreadsheet tools. Detailed documentation for most of these tools can be found in their respective user's guides.

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 IDS documentation, please visit our Website at:

http://www.dss.ibm.com

There you'll find the Feedback Connection page where you can enter comments and send them to us.

© Copyright IBM Corp. 1996

Chapter 1. Introducing IDS Database Administration

As database administrator, you are responsible for defining data, loading it into a database, and ensuring that applications using the data perform satisfactorily. This chapter describes database administrative tasks and tools that are unique to the IBM Intelligent Decision Server (IDS) environment and contains the following topics:

Database Administration Tasks

"Database Administration Tools"

"Characteristics of an IDS Logical Database" on page 2

"How Data Access Tools Retrieve Data" on page 3

Database Administration Tasks

IDS enables you to perform a number of database administration tasks. With IDS, you can:

Design the database.

Define categories, tables, columns, default joins, and default constraints in the logical database.

Create tables.

Add data to tables.

Create user accounts.

Grant table privileges to users.

Extract 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).

Database Administration Tools

IDS provides the following workstation tools to help you administer and access data. Certain functions that these tools can perform are described within this book. For more details on each database administration tool, see their respective tool guides.

DBA Tool

The DBA tool lets you define data as the user sees it by providing DBA access to the *Workstation Tools Data Dictionary (WTDD)*. The WTDD, described in detail in Chapter 2, "Working with the WTDD" on page 4, is the logical data dictionary that IDS tools use. The DBA tool also lets you run Structured Query Language (SQL) state-

© Copyright IBM Corp. 1996

ments. To run macros, however, you must use the SA tool. See the *DBA Tool User's Guide* for information on how to use the DBA tool.

SA Tool

The SA tool provides a graphical interface to administrative functions for several IDS features, including database access. You can enter or copy SQL statements in the SA tool window using the SQL Input Service. The DBA and SQL Entry tools also let you run SQL statements. The SA tool is the only IDS tool from which you can run macros and use the SAVE FOR command. See Chapter 5, “Accessing Databases with the SA Tool” on page 29 for instructions on how to use the SA tool to connect to a database and run macros and SQL statements.

Query Tool

The Query tool is used to retrieve data from a database. You can also update data or insert data into a database using the Query tool with a spreadsheet. See Chapter 3, “Transferring Data” on page 16 for instructions on how to load tables using the Query tool.

Spreadsheet Tool

The Spreadsheet tool performs calculations and analyses. Database administrators can also use it with the Query tool as a template to define data types that match the data types of columns in a table to be modified. See Chapter 3, “Transferring Data” on page 16 for instructions on how to load tables using the Spreadsheet tool.

Forms Design and Data Entry Tools

The Forms Design and Data Entry tools 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 Data Entry icons are created 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 IDS *Forms Design User's Guide* and the *Data Entry User's Guide*.

Characteristics of an IDS Logical Database

These database structures are unique to IDS:

- The Workstation Tools Data Dictionary (WTDD)
- The logical database
- Categories, fact tables, and dimension tables
- Keyed columns
- Default joins

The logical database, categories, keyed columns, and default joins are all defined in the WTDD. Chapter 2, “Working with the WTDD” on page 4 describes these database structures.

How Data Access Tools Retrieve Data

Figure 1 shows how data is retrieved from a database using the WTDD. The circled numbers correspond to the following numbered steps:

1. The user uses the Query tool to build 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 Query tool sends the SQL statement to the database, which uses its physical data dictionary to determine the actual location of the data.
4. The RDBMS then retrieves the data and sends it to the requesting workstation.

Figure 1. 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 2. To learn more about this feature, see the *Query User's Guide*.

Figure 2. DBA catalog in the Query Control window

Chapter 2. Working with the WTDD

The Workstation Tools Data Dictionary (WTDD) is a data dictionary that provides a logical view of a database. It is used exclusively by IDS tools. 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 key data access tools

The WTDD

The WTDD provides a logical view of databases that is standard across IDS-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 IDS 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 might have a physical column named PRODNUM but 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 keyed columns, joins between tables, constraints, and domains. These are described later in this chapter.

Each IDS logical database has a WTDD in the form of a set of tables. To maintain the WTDD, use the DBA tool. See the *DBA Tool User's Guide* for instructions on using the DBA tool.

Logical and Physical Database Representations in the DBA Tool

The logical database is a user-oriented representation of the physical database. Using the DBA tool, you define in the logical database how data is presented to users. The logical definitions you create are stored in the WTDD, which IDS data access tools use to create queries and reports. You can also use the DBA tool to create physical tables to contain the actual data.

© Copyright IBM Corp. 1996

The logical database puts useful groupings of tables into categories based on specific information needs. 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.

By contrast, the physical representation of the database in the DBA tool is used by the host's database management system. It is a set of physically stored data tables that the DBA tool represents as grouped by user. Table 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 executed 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 3 shows how logical and physical data is presented in the DBA tool. See the *DBA Tool User's Guide* for information on the features and functions of the tool.

Figure 3. 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, which you can view and manage using the DBA tool.

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
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

WTDD Tables That Contain Category and Column Information

Categories and columns are central to the logical database. Figure 4 shows the associations between tables that contain category information. The internal names are used in SQL statements. The print names are displayed in the Query tool and in the logical database that is depicted in the DBA tool.

Figure 4. Associations between tables that contain category information

Figure 5 shows the associations between tables that contain column information. The table and column names in this figure are the internal names that you use in SQL statements.

Figure 5. Associations between tables that contain column information

Creating WTDD Tables

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

createwtdd (for DB2, DB2 for OS/2, SQL/DS, DB2 for AIX, and DataJoiner databases).

newdict (for SYBASE**, Red Brick**, and ORACLE** databases).

A third macro, called **install**, is also located in the DBA file drawer for IBM databases. 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.

You must be connected to the appropriate database before calling these macros.

The **createwtdd** macro invokes an SQLMACRO facility supported by the IBM database gateways to send SQL statements to the database management system. You can enter the SQLMACRO command from any IDS tool that allows you to directly enter SQL statements. See Appendix E, “Using the SQLMACRO Facility” on page 66 for more information about the SQLMACRO facility.

The **createwtdd** and **newdict** macros also grant read and select privileges on the WTDD tables to PUBLIC, define categories, and create indexes on the tables.

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

DB2 for MVS

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

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

where:

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

SQL/DS

To create a WTDD in the SQL/DS for VM environment, run the **createwtdd** macro and provide the required parameters, as follows:

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

where:

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

DB2 for OS/2, DB2 for AIX, and DataJoiner

To create a WTDD in the DB2 for OS/2 and DB2 for AIX environments, run the **createwtdd** macro and provide the required parameters, as follows:

```
CALL createwtdd wto, wtgrant
```

where:

wto	Is the WTDD owner ID.
wtgrant	Is PUBLIC or is the ID to be granted access to the WTDD.

SYBASE, Red Brick, and ORACLE

To create a WTDD in the SYBASE, Red Brick, and ORACLE environments, run the **newdict** macro as follows (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. These elements help data access tools to design database tables, create views of the database, search for and retrieve information, and apply constraints to database tables.

Categories, Facts, and Dimensions

In the IDS environment, data is based on a multi-dimensional model that consists of fact and dimension table associations.

In the Reporter tool, facts typically make up the returned data in a report. Dimensions typically make up the column or row headings in a report.

A *category* is a group of tables that are used together. Tables contain either facts or dimensions.

A *fact* is data consisting of measures such as unit price or volume.

A *dimension* is data (such as periods of time, products, and locations) defining the context of facts.

Required: If you want to use the Reporter tool to retrieve data, define the tables as fact or dimension tables within categories.

Categories in the Logical Database

In the IDS environment, you create categories within an existing database system. Categories are strictly logical entities and do not define how or where data is stored. Categories are determined by the way the tables are used in applications. For example, you might specify that the tables containing employee, department, and job-related information are used by a personnel application. You might also specify that the table containing the employee information is used with a table containing salary information in a payroll application.

In the Query tool, all database categories are listed in the DBA catalog. The user selects a category from the catalog and then returns to the user catalog to query the data in those tables. See the *Query User's Guide* for details on specifying categories.

Required: For certain data access tools that use categories (Reporter, Forms Design, and Browser), you must configure the icons in the **Other Data** parameters in the Icon Options window to use a specific category at any one time. When a user opens a configured data access icon, the icon displays only the tables defined for the specified category. These tools will not open if a category is not specified.

The DBA tool represents categories as follows in the logical view of the database:

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 the database categories. The WT_CONTENTS table stores information about which tables are associated with each category. The *DBA Tool User's Guide* provides instructions on how to create categories.

Fact and Dimension Table Associations in the Reporter Tool

The multi-dimensional model in the IDS environment consists of facts and dimensions. Sets of facts and sets of dimensions are stored in separate tables. Fact tables are stored within a category. Figure 6 shows an example of fact tables within a category.

Figure 6. Fact tables within a category

Dimension tables consist of data that define facts. Figure 7 shows an example of dimension tables that correspond to the fact tables in figure 6.

Figure 7. Dimension tables that correspond to the fact tables in Figure 6.

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 the report.

There are two types of tables within the database category, the fact table and the dimension table.

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 the information stored in dimension tables.

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

Required: For the Reporter tool to function, you must follow these rules:

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 keyed column in a fact table, there must be a corresponding keyed column in a dimension table.

A period dimension table must exist.

When a user opens a Reporter icon, 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 joins.

Keyed Columns

Most tables have a set of columns, known as *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 8 has a primary key that consists of PRODUCT_ID, MARKET_DESC, and PERIOD. Three dimension tables are joined to the fact table, each with its own keyed column that corresponds with one of the names in the primary key.

Figure 8. One or more keyed columns define a primary key

Some IDS 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.

Default Joins

The Query tool uses joins between tables. You can specify in the DBA tool how tables are joined, and these joins become default joins. You can specify up to 400 joins per category. Query tool users can also create joins between any two columns that have like data types. If the Query tool users do not specify any joins, then the default joins are used. The WTDD table named WT_DJOINS stores information about joins.

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. However, the Query tool does not prevent the user from joining 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.

Required: To prevent users from creating nonsense joins, you must build data *domains* that you assign as a domain number 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 9 on page 12 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 9. Domain numbers are used to prevent nonsense joins

The WT_COLUMNS table stores information about the domain number given to each column. The WT_DOMAINS table stores information that describes each of the domain numbers.

N/A Values

IDS supports nulls, so it is unnecessary 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 workstation tools to read the WTDD until after you test your changes. The WTDD has a set of version numbers associated with it, to 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 tools.

Required: When you want data access tools to use the new version of the WTDD, run the **wtupdate** macro. This macro places the WTDD version number on the database server.

When you change the version number, all data access icons that are connected to the affected database will update their own copy of the WTDD tables the next time they are opened. Any new tools that are opened will 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 in 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 standard database table. You define views in the WTDD as logical tables.

Required: 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 icon is first opened, it reads a copy of the WTDD tables from the database server and records the current version number. Each time the icon 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 icon'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 Icons

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. Update Category 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 icon only, leaving other Reporter icons on your desktop (and throughout the system) running on the previous version of the WTDD.

If the WTDD is successfully updated, the Reporter tool generates a confirmation message. If the WTDD cannot be updated, 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 Icons

Whenever a Query icon reads a new copy of the WTDD from the database server, the icon's original query 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 retrieved by a query or report and those in the newly changed database. The Query tool indicates such inconsistencies by displaying deleted or changed database items as *phantoms*. Phantoms are indicated by an overstrike through the print name of a database item. In

addition, if the Query icon detects a phantom, a message appears 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. 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. IDS 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. The next time a data access tool is run, it will detect the change, update its copy of the WTDD, and use the new name provided in the SQLNAME column to generate the SQL.

The WT_COLUMNS and WT_TABLES tables store information about SQL names.

Backing Up and Restoring WTDD Tables

You can back up and restore WTDD tables 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.

The WTDD Backup capsule icon is located 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. In the User Input Control window that appears, 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 10.

Figure 10. WTDD Backup Capsule

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

   As shown in Figure 11 on page 15, 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 icon. The Query icon 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 11. 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. Arrow options are specified in the Arrow Options window as follows for backing up WTDD table contents:

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

a. Select the arrow between two spreadsheets that indicates stopped transfer of WTDD contents, and open the Arrow Options window.

b. Specify **No** for the **Stop Before Transfer** option.

c. 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 a and b for each table that you want to restore. When all arrows are unblocked, click on the *Run* button.

Chapter 3. Transferring Data

To transfer data between databases, you can use a site-specific database load utility (if you have one available), or you can use the SAVE FOR command in the SQL Input service in the SA tool.

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

Transfer database tables using the SAVE FOR command
Load data into tables using IDS tools

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

Transferring Data into Tables Using the SAVE FOR AT Command

The SAVE FOR command lets you transfer tables between databases. Tables must be no larger than a few megabytes, because lengthy transfers block the communications line. You can include the SAVE FOR command in a macro file. To use the SAVE FOR command, you must be connected to a database.

The SAVE FOR AT command copies selected data to a different database using two database gateway sessions.

Attention: The SAVE FOR AT command automatically converts data types used in various database management systems to supported IDS data types. To avoid incorrect or unsatisfactory results, you should always create the physical table before using the SAVE FOR AT command.

When you use the SAVE FOR AT command to transfer a table, two data–access sessions are used—one to select the data and one to insert it. If you have trouble making this kind of transfer, check the status window on your desktop to ensure that there is an extra data-access session available to handle the transfer.

© Copyright IBM Corp. 1996

The syntax of the SAVE FOR AT command is:

```
SAVE FOR user_name/password DATABASE database_name AS table_name -
AT gateway_name :network_name SELECT_statement
```

where:

user_name/password	Establishes access rights to the specified database for loading data. The privileges associated with *user_name* must include the authority to create tables in the destination database.
database_name	Specifies the destination database. For a DB2 database, specify the destination subsystem name. When *database_name* is unspecified, the default database or subsystem is used. You can also specify [.drda_server _name.]
table_name	Identifies the table in the destination database. The table is created if a physical table does not already exist. Otherwise, if the definition is compatible (that is, the number of columns and their data types match), the information is appended to the existing table.
gateway_name	Identifies the database gateway name as specified during its configuration.
network_name	Is the name of the network where the specified gateway resides. This is optional for gateways that reside on the local network.
SELECT_statement	Is a valid SELECT statement. Table and column names cannot be correlated references. This statement retrieves the specified data.

If your SELECT statement references a table belonging to DB2, DB2 for AIX, or DB2 for OS/2, you can add the FOR FETCH ONLY clause to the end of the SELECT statement. This may improve performance by as much as 60 percent.

Loading Data into Tables Using IDS Tools

You can use the Query and Spreadsheet tools to load data into tables. Use the Query tool to retrieve data from a table that you want to update or to one to which you want to add data. Then, copy a sample of one or two rows of the data to a spreadsheet. This step makes it easy to define data types that match the data types of columns in the table being modified. In effect, the spreadsheet serves as a template for data type settings. When you enter data into the spreadsheet, the data types are already defined for you. Finally, you can load the new or updated data back into the table.

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 that you selected are displayed in the user catalog, with their columns, as shown in Figure 12.

Figure 12. User Catalog in the Query Catalog window

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

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

6. Open a Spreadsheet icon.
   a. Click on the **Show Data** button in the Query results window 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 *Query Tool User's Guide*.

   b. Click on the **Select All** button in the Query results window, 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 will be considered as data.

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

7. Select **Enable Edits** from the Special menu in the Query results window.
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 IDS

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

Integer
Real
Character
Date

Tools that do not use the WTDD might not display data the same way as tools that use the WTDD. 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 2 lists the data types that the DBA tool defines for logical tables when the physical table already exists for each of the listed database.

Table 2. Data conversion from RDBMS to IDS format

IBM	SYBASE	Red Brick	ORACLE	DBA tool (IDS type)
integer, int, smallint	int, smallint, tinyint	integer or int, smallint, tinyint	number (n, 0), where n ú 9	Integer
float packed decimal, double zoned decimal	float, real, double precision, numeric, decimal, money, smallmoney	float, real, double precision, numeric or num, decimal or dec	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	character or char, time, timestamp	char, long varchar2, rowid, date	Character
date	datetime, smalldatetime	date		Date

Listed in Table 3 are the data types that the DBA tool ignores.

Table 3. Data types ignored by the DBA tool

IBM	SYBASE	Red Brick	ORACLE
fixed graphic, vargraphic, long vargraphic, large object, very large object	none	none	raw long, raw mlslabel

Chapter 4. Authorizing Users

IDS data access tools (such as Query and Reporter) 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 will be able to switch easily from one WTDD to another without specifying synonyms for the tables. These settings are made in the Icon Options window of the data access tools.

This chapter describes the information that is provided in each of the fields of the Icon Options window.

Configuring a Data Access Tool

To connect to a database, you must first configure a data access icon by providing information in the Icon Options window.

To configure a data access tool:

1. Select the icon and open the Icon Options window.
2. In the **Name** field, enter a name for the icon.

   It is good practice to 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.
3. Select **Other Data** in the **Display** field.

   Only the Reporter, Forms Design, and Browser tools display the **Category** field in the Icon Options window.

   Attention: 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.
4. Using the guidelines provided in the following sections, enter the required information into the corresponding text fields and select the appropriate choices in the other fields.

© Copyright IBM Corp. 1996

Setting Parameters for Data Access Tools

This section describes input for each field in the Other Data page of the Icon Options window for data access tools.

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 IBM database.

A database in the SYBASE, Red Brick, and ORACLE environments.

Accessing a Local IBM Database

In the IBM environment, the term *local database* refers to the RDBMS that is running on the same system as Host Services or the DB2 for OS/2 gateway or the DB2 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:

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
```

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. 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, as with the IBM AS/400, it is unnecessary to supply a local database name. For DB2 for MVS, you specify the name of a DB2 subsystem. For SQL/DS, you specify the name of the virtual machine running SQL/DS.

Accessing a Remote IBM Database Using DRDA

This section describes how data access icons 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 via Distributed Relational Database Architecture (DRDA).

DRDA connections must be set up between the hosts before IDS 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, 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, enter:

   ```
   .drda_server
   ```

   Be sure to include the period preceding 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 IDS CONNECT command. The IDS tools that let you enter SQL statements directly are the DBA, SA, and SQL Entry tools.

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

```
CONNECT TO drda_server
```

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

Using the IDS CONNECT Command: To use the IDS CONNECT command, enter:

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

where:

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

For example, enter:

```
CONNECT guest/demo DATABASE mydb.drda4
```

The IDS 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 V3.1 server:

```
Application server is DSN 3 1
```

Accessing a Default Database for SYBASE or Red Brick

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 and Red Brick 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

where:

servername	Is the name of the non-default SQL server.
databasename	Is 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 5, “Accessing Databases with the SA Tool” on page 29.
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 Icon Options window.

Accessing a Non-Default Oracle Database

When the IDS UNIX gateway is running on a different workstation than your Oracle server, the IDS 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 format:

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 as follows:

   T:host1:oracledb

   where:

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 Is the name of the target host defined in the UNIX /etc/hosts file.

oracledb Is 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: In SQL*NET version 2.0, you must create the SQLNET.ORA file and the TNSNAMES.ORA file before attempting to connect to an IDS 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 IDS gateway user's login file. This specifies the directory included in the SQLNET.ORA file and the TNSNAMES.ORA file.

To access a database other than that defined by the Default System ID (SID) 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 as follows:

   oracledb

   where:

oracledb 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 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.

To access a remote DB2 for AIX database, use either the DB2 CAE for AIX or the DB2 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 IDS is identical to accessing a DB2 for AIX database from IDS. See the *DataJoiner Administration Guide* for directions on how to configure different sources to DataJoiner.

If you currently use a SYBASE or Oracle Database Gateway, use the **nnsyb** and **nnora** macros to create nicknames for your SYBASE and Oracle WTDDs, respectively. See Appendix C, "Using Macros to Manage the WTDD" on page 54 for a list of all the available macros.

Specifying a Category

Only the Reporter, Forms Design, and Browser tools display the **Category** field:

The user specifies which tables the icon should 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. Tools that use categories will not open if you do not specify a category in the Icon Options window.

Specifying a WTDD Owner

The **WTDD Owner** field is for use with databases that support qualified table names and that handle multiple WTDDs.

This field makes it unnecessary for you to create synonyms. Without the information in this field, 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.

By using the **WTDD Owner** field, you allow users to easily 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 specify Marketing in the **SQL ID** field.

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

Specifying a User Name and Password

When an IDS tool connects to the database system, the system checks the **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 OS/2 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 fewer access privileges to users and create fewer synonyms or views because users share all the data. The major disadvantage of a single user ID is the lack of database security it offers. In addition, cost information for database usage is virtually nonexistent.

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 they require—you must grant privileges to many users and create multiple synonyms or views according to user requirements.

IDS Security Features for Databases

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

Through user ID setups

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 *Working with the Intelligent Decision Server*.

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 then appears, informing the user that this is a permanent change. The user's desktop name appears 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.

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 granted access to the database.

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.

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 whenever the icon is opened and a connection is made to the database gateway or server. It does this 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 networks that are linked via internetwork routing (INR). 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 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 6, "Desktop Transformers."

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.

Chapter 5. Accessing Databases with the SA Tool

You can use three IDS tools to run SQL: the SA tool, the DBA tool, and the SQL Entry tool. In addition to running SQL, the SA tool provides several functions for performing maintenance tasks on the database. With it, you can start and stop IDS services on a host system, obtain status information, control resources, and run macros and the SAVE FOR command.

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

Connect to a database to run database gateway commands
Run SQL statements
Run macros
Log your session

Brief descriptions of the DBA and the SQL Entry tools are also included. These tools also provide the capability to run SQL statements, but they do not let you run macros or the SAVE FOR command.

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

Connecting to a Database

To connect to a database:

1. Open an SA tool window.
2. Select a realm from the pull-down menu in the **Realm** field.
3. Select **Database Gateway Services** from the pull-down menu in the **Service Type** field.

   If the bottom edge of the SA tool window is close to the bottom of your desktop, all of the available service types might not appear in the Service Type menu. You can scroll through the menu by dragging the pointer to the bottom of the menu.
4. Select a service from the menu in the **Service** field, or type the name if you know it.
5. Click on the **Connect to Service** button to connect to the specified gateway service.

   The Database Gateway Services Password window opens.
6. Type the database gateway service password in the **Password** field.

   When you are connected, the Database Gateway Services window is displayed.

© Copyright IBM Corp. 1996

Running SQL Statements

You can use three different tools to run SQL statements:

The SQL input service of the SA tool
The SQL Control window of the DBA tool
The SQL Entry tool

Only the SA tool allows you to run macros and the SAVE FOR command. This section provides instructions on running SQL statements with the SA tool. For information on how to use the DBA and SQL Entry tools to run SQL statements, see the *DBA Tool User's Guide* and the *SQL Entry User's Guide*.

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 from the pull-down menu in the **Realm** field.
3. Select **SQL Input** from the pull-down menu in the **Service Type** field.
4. In the **Select Service** field, select a service from the menu 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.

   In this window, provide the following information for the corresponding fields:

Gateway Name	The name of the gateway to which you want to connect.
Database Name	The name of the database to which you want to connect.
User Name	The database user ID.
Password	The database user password.

6. Click on the **Connect to Service** button to start an SQL input session.

   If you typed valid information, an SQL Input window and an SQL Input Log window appear simultaneously.
7. Type your statement in the SQL Input window, and click on the **Apply** button.

   The command results will appear in the SQL Input Log window.
8. Repeat step 7 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.

To use the DBA tool to run SQL statements:

1. Open the SQL Control window.
2. Type or copy an SQL statement into the **SQL statement** field.
3. Click on the **Run** button in the window header.

See the *DBA Tool User's Guide* for detailed procedures on using this tool.

To use the SQL Entry tool to run SQL statements:

1. Create a source document to contain SQL statements.
2. Specify the name of the source document in the SQL Controls window.
3. Click on the **Run** button in the SQL Entry 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 IDS SQL extended statement as the name of a database object. See Appendix A, “Reserved Words for DB2 Databases” on page 40 and Appendix B, “Reserved Words for Unix Databases” on page 45 for a list of the reserved words for IBM 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 the SQL Input window header.

To enter multiple-line statements, use 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 are concatenated from the end of the first line to the beginning of the next line.

To stop a currently executing SQL statement, press the Stop function key. You might have to press the Stop key more than once. The connection to the SQL server and the gateway is dropped. Close the SQL Input window. Then reconnect to the SQL server using 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 database management and table manipulation. In IDS, macros must be called from the SQL Input window in the SA tool. To display the SQL Input window, follow the instructions in “Running SQL Statements” on page 30.

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

```
CALL macro_name parameters
```

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

The default macro folder 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.

IDS-supplied macros for each database are stored in a macro folder inside the DBA file drawer. Each macro folder in the DBA file drawer contains a help document that describes the macros and provides syntax examples. Macro parameters are listed and briefly described in each macro file. You will find the following macro folders in the DBA file drawer:

DBS - DB2
DBS - ORACLE7
DBS - OS2
DBS - OS400
DBS - REDBRICK
DBS - SQLDS
DBS - SYBASE
DBS - TERADATA
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 any error messages or other responses from the RDBMS.

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

Notes on Using Macros

Macros and SQL can be included as commands in macro files. IDS supports nested macros to a depth of 8.

The file server is case sensitive. Case must be preserved 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.

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 key. You might have to press the Stop key more than once. The connection to the SQL server and the gateway is dropped. Close the SQL Input window, and reconnect to the SQL server using the SQL Input service selection window.

Chapter 6. Desktop Transformers

This chapter describes transformers that allow users to configure IDS desktop tools and to display and manipulate WTDD table data from the desktop. For information on other transformers, see the *Transformers Guide and Reference*.

The SetDBAccess Transformer

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 searches a container icon for all icons 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.

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 13 on page 35, 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 **New parameter** field blank, where *parameter* is a data access option. 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.

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

© Copyright IBM Corp. 1996

Figure 13. Transformers Control Window

1. In the **Name of container to check** field, type the name of a folder, file drawer, capsule, or envelope that resides on your desktop.
2. In the remaining fields, type 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 information 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.
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 information 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 information 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, you would type the following information in the **Name of container to check** field:

   data1|Desktops

   Except icons inside file drawers, all icons on desktops stored on *data1* will be changed.
3. Run the transformer.

Changing Icons in File Drawers

The SetDBAccess transformer lets you 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 information 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.
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 information in the **Name of container to check** field:

System|File Drawers| Query Icons

All icons stored in the * Query Icons file drawer change.

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 information 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 stored on the *data1* disk, enter:

data1|File Drawers

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

The Metadata Exchange Transformers

The contents of many of the WTDD tables consist of *metadata*, or data about data. Generally, metadata describes three types of data:

Naming, which includes business terms and descriptions. In the IDS WTDD tables, you can find naming metadata in columns such as the PRINTNAME column (located in the WT_COLUMNS table) and the DESCRIPTION column (located in the WT_CATALOG table).

Navigation, which includes table joins. In the WTDD, you can find navigation metadata in tables such as the WT_JOINS table and the WT_DOMAINS table.

Calculation, which includes derived columns and constraints. In the WTDD, you can find calculation metadata in tables such as the WT_DCONSTR table.

The metadata exchange transformers describe these types of metadata in IDS by reading and writing information in a tag language between WTDD tables and a text region. There are two metadata exchange transformers, MDExport and MDImport.

MDExport can extract metadata from the WTDD tables and make it available to other applications.

MDImport can update the WTDD tables using the metadata description supplied by another application.

The metadata exchange transformers also allow IDS metadata to be exchanged as part of a data warehouse or information catalog. Applications such as IBM Visual Warehouse and IBM Dataguide can write and interpret the tag language that the transformers use.

The transformers work with the following WTDD tables:

WT_CATALOG
WT_COLUMNS
WT_CONTENTS

WT_DCONSTR
WT_DJOINS
WT_DOMAINS
WT_TABLES

For more information on WTDD tables see chapter 2, *Working with the WTDD*, and Appendix C.

Configuring MDExport

MDExport is a transformer that extracts metadata from the WTDD tables and writes it in a tagged description to a text region.

To configure MDExport:

1. Open the Administrator Icons file drawer and copy the MDExport transformer into an empty capsule.
2. Press and hold down both mouse buttons simultaneously on the transformer and open the Icon Options window.
3. Select **Other Data**.
4. Configure the fields for the database that you want to extract metadata from.
5. Close the Other Data window and the Icon Options window.

Configuring MDImport

MDImport reads metadata information in a tag language from an input text region and uses it to update the WTDD tables.

To configure MDImport:

1. Open the Administrator Icons file drawer and copy the MDImport transformer into an empty capsule.
2. Press and hold down both mouse buttons simultaneously on the transformer and open the Icon Options window.
3. Select **Other Data**.
4. Configure the fields for the database that you want to send updated metadata to.
5. Close the Other Data window and the Icon Options window.

Metadata Transformers within a Capsule Environment

You can expedite the processing of metadata tag files by using the metadata exchange transformers with other transformers within a capsule.

For example, in a data warehousing environment, such capsules can extract metadata from the WTDD tables into a workstation file for distribution. Other capsules can be configured to update the WTDD tables using the metadata tag files received from a central warehouse.

Similarly, these capsules will read and write the workstation files of an information catalog product such as Dataguide.

Appendix A. Reserved Words for DB2 Databases

The words listed in Table 4 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. See Appendix B, "Reserved Words for Unix Databases" on page 45 for reserved words in UNIX databases. Reserved words marked with an asterisk are IBM SQL statements, also referred to as extended SQL statements.

Table 4 (Page 1 of 5). Reserved words for DB2 family of databases

Reserved Word	DB2	SQL/DS	DB2 for AIX	DB2 for OS/2
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
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

© Copyright IBM Corp. 1996

Table 4 (Page 2 of 5). Reserved words for DB2 family of databases

Reserved Word	DB2	SQL/DS	DB2 for AIX	DB2 for OS/2
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
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

Table 4 (Page 3 of 5). Reserved words for DB2 family of databases

Reserved Word	DB2	SQL/DS	DB2 for AIX	DB2 for OS/2
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
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

Table 4 (Page 4 of 5). Reserved words for DB2 family of databases

Reserved Word	DB2	SQL/DS	DB2 for AIX	DB2 for OS/2
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
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		

Table 4 (Page 5 of 5). Reserved words for DB2 family of databases

Reserved Word	DB2	SQL/DS	DB2 for AIX	DB2 for OS/2
SECQTY	X			
SELECT	X	X	X	
SET	X	X	X	
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 Unix Databases

The words listed in Appendix B, “Reserved Words for 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. See Appendix A, “Reserved Words for DB2 Databases” on page 40 for reserved words in DB2 databases.

Table 5 (Page 1 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
? *	X		
ABORT			X
ACCESS		X	
ACTION			X
ADD	X	X	X
ADDRESS	X		
ALL	X	X	X
ALTER	X	X	X
AND	X	X	X
ANY	X	X	X
AS	X	X	X
ASC	X	X	X
AUDIT		X	
AUTHORIZATION			X
AVG	X		X
BEGIN	X		
BETWEEN	X	X	X
BREAK	X		X
BROWSE	X		
BULK	X		
BY	X	X	X
CASCADE			X
CHANGE			X
CHANGEPATH			X
CHAR		X	X
CHARACTER			X
CHECK		X	X

© Copyright IBM Corp. 1996

Table 5 (Page 2 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
CHECKPOINT	X		
CLOSE			X
CLUSTER		X	
CLUSTERED	X		
COBOL			X
COLUMN		X	X
COMMENT		X	
COMMIT	X		X
COMPRESS		X	
COMPUTE	X		
CONFIRM	X		
CONNECT		X	X
CONSTRAINTS			X
CONTINUE	X		X
CONTROLROW	X		
CONVERT	X		
COPY IN FILE	X		
COUNT	X		X
CREATE	X	X	X
CUME			X
CURRENT		X	X
CURRENT_DATE			X
CURRENT_TIME			X
CURRENT_TIMESTAMP			X
CURSOR			X
DATABASE	X		X
DATE		X	
DATEFORMAT			X
DBA	X		X
DBCC	X		
DEC			X
DECIMAL		X	X
DECLARE	X		X
DEFAULT	X	X	X

Table 5 (Page 3 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
DELETE	X	X	X
DESC	X	X	X
DESCRIBE			X
DIRECTORY			X
DISK	X		
DISTINCT	X	X	X
DISTRIBUTED			X
DOUBLE			X
DOUBLEPRECISION			X
DROP	X	X	X
DUMMY	X		
DUMP	X		
ELSE	X	X	
END	X		
ERRLVL	X		
ERROR			X
ERROREXIT	X		
ESCAPE			X
EXCEPT	X		
EXCLUSIVE		X	X
EXEC	X		
EXECUTE	X		
EXISTS	X	X	X
EXIT	X		
EXPAND			X
EXTRACT			X
FETCH			X
FILE		X	
FILENAME			X
FILLFACTOR	X		
FIRST			X
FLOAT		X	X
FOR	X	X	X
FOREIGN			X

Table 5 (Page 4 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
FORTRAN			X
FROM	X	X	X
FULL			X
GOTO	X		
GRANT	X	X	X
GROUP	X	X	X
HAVING	X	X	X
HOLDLOCK	X		
IDENTIFIED		X	
IF	X		
IMMEDIATE		X	
IN	X	X	X
INCREMENT		X	
INDEX	X	X	X
INDICATOR			X
INHERITING			X
INITIAL		X	
INNER			X
INSERT	X	X	X
INT			X
INTEGER		X	X
INTERSECT		X	
INTO	X	X	X
IS	X	X	X
JOIN			X
KEY			X
KILL	X		
LANGUAGE			X
LAST			X
LEFT			X
LEVEL		X	
LIKE	X	X	X
LINENO	X		

Table 5 (Page 5 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
LOAD	X		
LOCK		X	X
LOGOFF *	X		
LONG		X	
MACRO			X
MAJOR			X
MAX	X		X
MAXEXTENTS		X	
MAXFILESIZE			X
MIN	X		X
MINOR			X
MINUS		X	
MIRROREXIT	X		
MODE		X	
MODIFY		X	
MODULE			X
MOVINGAVG			X
MOVINGSUM			X
NAME			X
NO			X
NOAUDIT		X	
NOCOMPRESS		X	
NONCLUSTERED	X		
NOT	X	X	X
NULL	X	X	X
NUMBER		X	
NUMERIC			X
OF		X	X
OFF	X		X
OFFLINE	X	X	
OFFSETS	X		
ON	X	X	X
ONCE	X		
ONLINE	X	X	

Table 5 (Page 6 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
OPEN			X
OPTION		X	X
OR	X	X	X
ORDER	X	X	X
OUTER			X
OVER	X		
OVERRIDE			X
PASCAL			X
PASSWORD	X		
PATTERN			X
PCTFREE		X	
PERM	X		
PERMANENT	X		X
PLAN	X		
PLI			X
PRECISION			X
PREPARE	X		
PRIMARY			X
PRINT	X		
PRIOR		X	
PRIVILEGES		X	X
PROC	X		
PROCEDURE	X		X
PROCESSEXIT	X		
PUBLIC	X	X	X
QUIT			X
RAISERROR	X		
RANK			X
RATIOTOREPORT			X
RAW		X	
READTEXT	X		
REAL			X
RECONFIGURE	X		
REFCHECK			X

Table 5 (Page 7 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
REFERENCES			X
RENAME		X	
RESET			X
RESOURCE		X	X
RESTRICT			X
RESUME			X
RETURN	X		
REVOKE	X	X	X
RIGHT			X
ROLLBACK	X		X
ROW		X	
ROWCOUNT	X		
ROWID		X	
ROWLABEL		X	
ROWNUM		X	
ROWS		X	
RULE	X		
SAVE *	X		
SCHEMA			X
SELECT	X	X	X
SESSION		X	
SET	X	X	X
SETUSER	X		
SHARE		X	
SHUTDOWN	X		
SIZE		X	
SMALLINT		X	X
SOME			X
SPILLAREA			X
SQLCODE			X
START		X	
STARTLOG	X		
STATISTICS	X		
STATUS	X		

Table 5 (Page 8 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
STOPLOG	X		
SUBMIT	X		
SUCCESSFUL		X	
SUM	X		X
SUMMING			X
SUPPRESS			X
SYB_TERMINATE	X		
SYNONYM		X	X
SYSDATE		X	
TABLE	X	X	X
TABLESPACE			X
TAPE	X		
TEMP	X		
TEMPORARY	X		X
TERTILE			X
TEXTSIZE	X		X
THEN		X	
THRESHOLD			X
TIME	X		
TINYINT			X
TO	X	X	X
TRAN	X		
TRANSACTION	X		
TRIGGER	X	X	
TRUNCATE	X		
TSEQUAL	X		
UID		X	
UNION	X	X	X
UNIQUE	X	X	X
UNLOCK			X
UPDATE	X	X	X
USE	X		
USER		X	X
VALIDATE		X	

Table 5 (Page 9 of 9). Reserved words for UNIX databases

Reserved Word	SYBASE	ORACLE	Red Brick Warehouse
VALUES		X	X
VARCHAR		X	
VARCHAR2		X	
VIEW			X
WAIT			X
WAITFOR	X		
WHEN			X
WHENEVER		X	
WHERE	X	X	X
WHILE	X		
WITH	X	X	X
WORK			X
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 the *DBA Tool User's Guide*. 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 IBM and UNIX databases that IDS supports. A syntax example is shown for one macro used with IBM databases and one macro used with UNIX databases.

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, refer to *Administering Host Services for MVS: DB2 and Cooperative Application Services* for details.

Table 6 lists macros for IBM databases.

Table 6 (Page 1 of 2). Macros for IBM databases

Macro name	DB2	SQL/DS	DB2 for OS/2	Description
converttodd3.0	X	X	X	Converts a WTDD that was created by an earlier version of IDS 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 catalog tables for the connected user.
createsynexplain	X	X		Creates a synonym for tables that decode values in a DB2 PLAN_TABLE table or SQL explain table.
createsynhelp		X		Creates synonyms for the SQL/DS help tables. Help owner is usually the SQL DBA.
createsynwtdd	X	X	X	Drops and creates synonyms or views for the specified WTDD.
createuserexplain	X	X		Creates a DB2 PLAN_TABLE table or SQL/DS explain table for the specified user.
createwtdd	X	X	X	Creates a new WTDD in a IDS database.
dropcollection				Removes a collection for the specified user.
dropwtdd	X	X	X	Drops a WTDD for the connected user.

© Copyright IBM Corp. 1996

Table 6 (Page 2 of 2). Macros for IBM databases

Macro name	DB2	SQL/DS	DB2 for OS/2	Description
dropwtddspace	X			Drops the specified table space.
install	X	X	X	Creates a WTDD and loads it with descriptions of the DB2 or SQL/DS catalog tables.
loaddbmscat	X	X	X	Loads a WTDD with information about the DB2 or SQL/DS catalog.
loadexplain	X	X		Loads a WTDD with information about a DB2 PLAN_TABLE table or SQL/DS explain table.
loadsample	X		X	Loads a WTDD with information about the DB2 sample tables.
newuserc		X		Grants connect privileges to an SQL/DS user with the specified password.
newusercr		X		Grants connect and resource privileges to an SQL/DS user with the specified password.
wtupdate	X	X	X	Updates the version number of a WTDD.

Macros that create synonyms do so in DB2 and SQL/DS 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 that accesses the WTDD and want to use unqualified WTDD names in those queries. DB2 for AIX synonym macros create views instead of synonyms.

Table 7 lists macros for UNIX databases. A check mark in a column indicates a macro exists for that particular database.

Table 7 (Page 1 of 2). Macros for UNIX databases

Macro name	SYBASE	Red Brick Ware-house	ORACLE	DB2 for AIX	DataJoiner	Description
converttodd3.0	X	X	X			Converts a WTDD that was created by an earlier version of IDS 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.

Table 7 (Page 2 of 2). Macros for UNIX databases

Macro name	SYBASE	Red Brick Ware-house	ORACLE	DB2 for AIX	DataJoiner	Description
wtddsyn			X			Re-creates PUBLIC syno-nyms for the WTDD tables.
wtgrantall	X	X				Grants INSERT, UPDATE, and DELETE privileges on the WTDD tables to a speci-fied user.
wtgrantread	X	X				Grants SELECT privileges on the WTDD tables to a speci-fied user.
wtgrantwrite	X	X				Grants INSERT, UPDATE, and DELETE privileges on the WTDD tables to a speci-fied user.
wtupdate	X	X	X	X	X	Updates the version number of each WTDD table.

All macro files found in the PROTOTYPE folder (for ORACLE database gateways) must be copied to the DBA file drawer and modified.

Appendix D. Contents of WTDD Tables

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

The tables in this chapter 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 8 lists and describes each of the columns in WT_CATALOG.

Table 8. WT_CATALOG columns

Internal name of column	Print name of column	Data type (max. length)	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 appears 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 defined in the WTDD. Table 9 lists and describes each of the columns in WT_COLUMNS. The physical order of the columns in WT_COLUMNS may not match the logical order shown in Table 9.

Table 9 (Page 1 of 3). WT_COLUMNS columns

Internal Name of Column	Print Name of Column	Data type (Max. Length)	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 only 15 characters.

© Copyright IBM Corp. 1996

Table 9 (Page 2 of 3). WT_COLUMNS columns

Internal Name of Column	Print Name of Column	Data type (Max. Length)	Description
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 appears 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.
RESOLUTION	Date Interpretation	Integer (4)	If DATATYPE is date, the date will be 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**: is not part of primary key **1**: is part of primary key

Table 9 (Page 3 of 3). WT_COLUMNS columns

Internal Name of Column	Print Name of Column	Data type (Max. Length)	Description
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.
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 10 lists and describes each of the columns in WT_CONTENTS.

Table 10. WT_CONTENT or WT_CONTENTS columns

Internal name of column	Print name of column	Data type (max. length)	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. This information is used by the Reporter tool only. Table 11 lists and describes each of the columns in *WT_DCONSTR*.

Table 11. 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 only 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 12 lists and describes each of the columns in WT_DJOINS.

Table 12. WT_DJOINS columns

Internal name of column	Print name of column	Data type (max. length)	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.
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 identified by the database administrator. This table has one row for each domain description. Table 13 lists and describes each of the columns in WT_DOMAINS.

Table 13. WT_DOMAINS columns

Internal name of column	Print name of column	Data type (max. length)	Description
DOMAIN	Internal Domain Number	Integer (4)	Number of the domain.
PRINTNAME	Print Name	Char (30)	Display name of the domain as it appears 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 14 lists and describes each of the columns in WT_FUNDEP for your information only.

Table 14. WT_FUNDEP columns

Internal name of column	Print name of column	Data type (max. length)	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 be able to use the table or view, you must also indicate in the WTDD whether the table or view contains fact or

dimension information. The *Reporter User's Guide* describes in detail how fact and dimension tables are used.

The WT_TABLES table has one row for each table or view in the logical database. Table 15 lists and describes each of the columns in WT_TABLES. The physical order of the columns in WT_TABLES may not follow the logical order shown in Table 15.

Table 15. WT_TABLES columns

Internal name of column	Print name of column	Data type (max. length)	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 appears 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 have been checked out. Table 16 lists and describes each of the columns in this table.

Table 16 (Page 1 of 2). WT_TTBLNAM or WT_TTBLNAMES columns

Internal name of column	Print name of column	Data type (max. length)	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.

Table 16 (Page 2 of 2). WT_TTBLNAM or WT_TTBLNAMES columns

Internal name of column	Print name of column	Data type (max. length)	Description
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 IDS 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 IDS.

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 17 lists and describes each of the columns in this table.

Table 17. WT_VARS columns

Internal name of column	Print name of column	Data type (max. length)	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 18 on page 65 lists and describes each of the columns in this table.

Table 18. WT_VERSION or WT_VERSIONS columns

Internal name of column	Print name of column	Data type (max. length)	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.
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, an *Ambiguous version number* error message displays in the Important Message window when any data access icon is opened. If the rows are identical, SQL will delete all of them at once. 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 execution 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 generated by a gateway macro, each SQL statement and its result must be sent over the network. Gateway macros with many SQL statements, such as those required for building a WTDD, can be inefficient. The SQL macros 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 on the SQLMACRO folder configured in the DBS-OS2 folder in the DBA file drawer.

While gateway macros must be executed from the IDS SA tool, the SQLMACRO facility makes it possible to execute macros from various tools. The SQLMACRO command can be entered from any IDS tool that lets you enter SQL statements. In addition, the macros run by the SQLMACRO command can also be executed in batch mode on the MVS host system. For a detailed description of how to do this, refer to *Administering Host Services for MVS: DB2 and Cooperative Application Services*.

The syntax of the SQLMACRO command issued from IDS is:

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

where:

macro_name	Is the file name of a macro.
parm_1	Is the name of the first parameter.
value_1	Is the value of the first parameter.
parm_n	Is the name of the *n*th parameter.
value_n	Is 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, DB2 for OS/2, SQL/DS, and DB2 for AIX systems, respectively. Macros that exist for systems other than the one you are running allow you to execute SQL on those systems through DRDA connections.

© Copyright IBM Corp. 1996

Glossary

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.

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 IDS logical tables are integer, real, character, and date. Other data types available in some IDS logical tables include null, long varchar, varchar, time, and timestamp.

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 IDS.

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 between tables that the database administrator creates for use by data access tools when users do not specify any joins.

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. *Contrast with* fact.

(DRDA)

Distributed Relational Database Architecture . 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 may 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.

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

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 IDS environment, the name by which a database item is stored in the WTDD. Each database element 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 IDS 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 columns in a relational database. In IDS windows, a join between tables is usually represented by a line with an equal sign (=) in the middle.

keyed 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 DB2/2 gateway.

local network. The network to which a specified user or resource is directly connected. *See also* remote network.

logical database. A database representation that includes print names for tables and columns. It 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 IDS text document in the DBA file drawer that contains SQL commands and that can contain substitutable values that are passed when the macro is executed.

© Copyright IBM Corp. 1996

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.

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 database as seen by the host's database management system. It is a set of physically stored data tables, grouped by user, that is represented to IDS data access tool users by the 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.

primary key. In IDS, 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 keyed 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 IDS 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 SQL names that can be understood by the database.

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 via *DRDA*.

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

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

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

SQL name. The table or column name that IDS data access tools use for generating SQL. 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. *Compare with* internal name.

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.

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

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 IDS workstation 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.

IDS Publications

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

General Publications

Getting Started with the Intelligent Decision Server (SC26-8892)

Working with the Intelligent Decision Server (SC26-8893)

Installation, Configuration, and Administration Publications

Installing IDS LAN Components (GC26-8877)

System Administration Guide and Reference (SC26-8890)

Messages and Codes (GC26-8878)

Database Gateway Services Publications

Installing Database Gateway Services for UNIX Environments (GC26-8884)

Configuring and Administering Database Gateway Services (GC26-8885)

Administering Databases with IDS (SC26-8891)

Host Services Publications

Planning MVS Host-to-LAN Connectivity (GC26-8880)

Installing Host Services and Configuring Host-to-LAN Communications for MVS (GC26-8881)

Administering Host Services for MVS: DB2 and Cooperative Application Services (GC26-8882)

Developing Applications with Cooperative Application Services (GC26-8883)

Configuring and Administering Mail Gateway Services (GC26-8923)

Open Data Access Service Publications

Developing Applications with Open Data Access Service (SC26-8888)

Data Access Tools Publications

Browser User's Guide (SC26-8900)

Data Entry User's Guide (SC26-8897)

DBA Tool User's Guide (SC26-8894)

SQL Entry User's Guide (SC26-8898)

Data Analysis Tools Publications

BASIC Tool User's Guide (SC26-8908)

Query User's Guide (SC26-8901)

Spreadsheet User's Guide (SC26-8904)

Presentation Tools Publications

Layout User's Guide (SC26-8902)

Plot User's Guide (SC26-8905)

Reporter User's Guide (SC26-8903)

Text User's Guide (SC26-8896)

Application Development Tools Publications

Transformers Guide (SC26-8907)

Capsule User's Guide (SC26-8906)

Forms Design User's Guide (SC26-8899)

Interacting with PC Applications (SC26-8895)

Installing and Configuring IDS Open Clients (GC26-8889)

© Copyright IBM Corp. 1996

Index

Special Characters

@-variable 26

A

access privileges 26
Accessing a DataJoiner Gateway 25
Accessing a DB2 for AIX Database 24
Accessing a Local IBM Database 21
Accessing a Remote IBM Database Using DRDA 21
Accessing an Oracle Database 23
Accessing SYBASE and Red Brick Databases 23
Authorizing Users 20

B

backing up WTDD tables 14
Backup capsule 14
binary 19
Browser tool 20

C

Calculation metadata 37
Categories 6, 9
Category field 25
Changing icons in file drawers 36
Changing icons on a desktop 35
char 19
Character 18
columns 6
concatenated words 31
Configuring a Data Access Tool 20
Configuring MDExport 38
Configuring MDImport 38
constraints 12
Contents of WTDD Tables 57
createwtdd 7
Creating WTDD Tables 7

D

data access icons 34
data access paths 3
Data Entry tools 2
data transfer 15
data warehousing environment 38
Database Administration Tasks 1
database administration tools 1
database categories 10
Database Name field 21, 23
Dataguide 37, 38
Date 18
datetime 19
DBA catalog 9
DBA tool 1, 5, 14, 29
DBS - DataJoiner 32
DBS - DB2 32
DBS - DB2/6000 32
DBS - DB2/6000 PE 32
DBS - ORACLE7 32
DBS - OS2 32
DBS - OS400 32
DBS - REDBRICK 32
DBS - SQLDS 32
DBS - SYBASE 32
DBS - TERADATA 32
dec 19
decimal
  double zoned 19
  float packed 19
default constraints 12
default joins 11
Default SID 23
desktop names 27, 28
dimension table 10
dimensions 9
domain number 11
domains 11
double precision 19
double zoned decimal 19
DRDA connections 21

E

Enable Edits 18
entering multiple-line statements 31

© Copyright IBM Corp. 1996

F

fact table 10
facts 9
field
  user name 27
fields
  Category 25
  Database Name 21, 23
  Gateway 21
  Interface File Directory 23
  Name 20
  Name of container to check 34
  New parameter 34
  New User Name 34
  Password 26, 29
  Realm 29
  Service 29
  Service Type 29
  Specify User Name to change 34
  SQL ID 26
  SQL Name 14
  User Name 26
  WTDD Owner 25
fixed graphic 19
float 19
float packed decimal 19
FOR FETCH ONLY clause 17
Forms Design and Data Entry tools 2
Forms Design tool 20
Forms Design tools 2

G

Gateway field 21
gateway macro 66
group IDs 26
Grouping tables 9

I

Icon Options window 9
IDS CONNECT Command 22
image bit 19
install macro 7
int 19
Integer 18
Interface File Directory field 23
internal name 14
internal names 4

K

keyed columns 10
keyword 31

L

large object 19
loading data 17
logical database 4
long varchar 19
long varchar2 19
long vargraphic 19

M

macro
  default folder 32
  user-written 32
macros 7
  DBS - DataJoiner 32
  DBS - DB2 32
  DBS - DB2/6000 32
  DBS - DB2/6000 PE 32
  DBS - ORACLE7 32
  DBS - OS2 32
  DBS - OS400 32
  DBS - REDBRICK 32
  DBS - SQLDS 32
  DBS - SYBASE 32
  DBS - TERADATA 32
  list of 54
  running 32
MakeSecure transformer 28
Mapping Data Types 18
MDExport 37
MDImport 37
metadata 37
Metadata Exchange Transformers 37
METWTBLD program 7
money 19
multiple IDs 26
multiple-line statements 31

N

N/A values 12
Name field 20
Name of container to check field 34
Naming metadata 37
Navigation metadata 37
nchar 19
New parameter field 34
New User Name field 34
newdict 7
nnora macro 25
nnsyb macro 25
nonsense join 11
nulls 12
num 19
number 19
numeric 19
nvarchar 19

O

Other Data parameters 9
overstrike 13

P

password 22
Password field 26, 29
Phantoms 13
physical database 5
Populate WTDD Tables capsule 15
primary key 11
print names 4

Q

qualifying the WTDD table name 25
Query icon 13, 15
Query tool 2, 11, 13, 17

R

Real 18, 19
Realm field 29
Reporter icon 13
Reporter tool 9, 10, 12, 20
reserved word 31
reserved words 40
restoring WTDD tables 14
rowid 19
running macros 32
Running SQL Statements 30

S

SA tool 2, 16, 23, 29, 66
SAVE FOR AT command 16
SAVE FOR command 16, 29, 30
secured icons 27
securing an icon 27
security features 26
SELECT statement 17
Service field 29
Service Type field 29
SID 23
single IDs 26
smalldatetime 19
smallint 19
smallmoney 19
special characters desktop names 28
Specify User Name to change field 34
Specifying a Gateway and a Database 21
Specifying a User Name 26
Specifying a WTDD Owner 25
Specifying an SQL ID 26
Spreadsheet tool 2, 17, 20
SQL CONNECT TO command 22
SQL DROP command 25
SQL Entry tool 29, 30
SQL ID 26
SQL ID field 26
SQL Input Log 30
SQL Input service 16, 30
SQL interpreter 31
SQL macros 66
SQL name 14
SQL Name field 14
SQL Statements 30, 66
SQL/DS 8
SQLMACRO facility 7, 66
Stop key 31, 33
stopping data transfer 15
synonyms 25, 26

T

text 19

The SetDBAccess Transformer 34
timestamp 19
tinyint 19
transferring data 16

U

Update Category item 13
User Name field 26, 27
user-written macro 32
using group IDs 26
using multiple IDs 26
using single IDs 26
Using the SQLMACRO Facility 66

V

varbinary 19
varchar 19
vargraphic 19
version number 13
version numbers 12
very large object 19
views 12, 26
Visual Warehouse 37

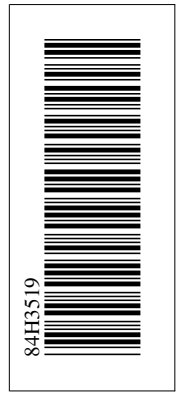
W

WT_CATALOG 9, 38, 57
WT_COLUMNS 11, 12, 14, 38, 57
WT_CONTENTS 9, 38, 59
WT_DCONSTR 12, 13, 38, 60
WT_DJOINS 11, 38, 61
WT_DOMAINS 12, 38, 62
WT_FUNDEP 62
WT_TABLES 13, 14, 38, 62
WT_TTBLNAMES 63
WT_VARS 64
WT_VERSIONS 65
WTDD 4
WTDD Backup capsule 14
WTDD database elements 9
WTDD Owner field 25
WTDD Owners 25
WTDD tables 6
WTDD version number 12
wtupdate macro 12, 13

IBM®

Part Number: 84H3519

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.



SC26-8891-

DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 664 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 677 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 747 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 860 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 868 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1122 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1141 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1149 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1200 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1249 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1405 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1419 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1424 OF 'IDS91MST'

DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1577 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2135 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2603 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2642 OF 'IDS91MST'
DSMBEG323I STARTING PASS 2 OF 4.
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 664 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 677 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 747 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 860 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 868 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1122 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1141 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1149 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'

DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1200 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1249 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1405 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1419 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1424 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1577 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2135 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2603 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2642 OF 'IDS91MST'
DSMBEG323I STARTING PASS 3 OF 4.
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 664 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 677 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 747 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 860 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'

DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 868 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1122 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1141 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1149 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1200 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1249 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1405 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1419 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1424 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1577 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2135 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2603 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2642 OF 'IDS91MST'
DSMBEG323I STARTING PASS 4 OF 4.
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S

DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 664 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 677 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 747 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 860 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 868 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1122 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1141 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1149 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1200 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1249 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1405 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1419 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1424 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S

DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 1577 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2135 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2603 OF 'IDS91MST'
DSMKPO654E INVALID POSTSCRIPT FILE.
DSMMOM395I '.EDFPO' LINE 70: .po @E@P@S
DSMMOM397I '.EDFPO' WAS IMBEDDED AT LINE 910 OF '.EDFAWRK'
DSMMOM397I '.EDFAWRK' WAS IMBEDDED AT LINE 130 OF '.EDFARTA'
DSMMOM397I '.EDFARTA' WAS IMBEDDED AT LINE 2642 OF 'IDS91MST'