Symposium OPEN IVR

SQL Server User Guide

Product release:	4.0
Document release:	Standard 1.0
Date:	July 1998

Publication history

July 1998	This is the Standard 1.0 issue of the *Symposium OPEN IVR 4.0 SQL Server User Guide*.

Contents

About this guide

Purpose of this guide

The Symposium OPEN IVR *SQL Server User Guide* is prepared for application developers who want to interface IVR applications with a Structure Query Language (SQL)-based, ANSI-compliant database management system (DBMS).

Note: Although use of the SQL Server does not require an in-depth knowledge of a database management system, you should be familiar with SQL command syntax and the database that will be accessed; for example, the names of the tables and columns. Consult your database administrator for assistance if you are unfamiliar with your database environment.

Document organization

This guide is organized as follows:

- Chapter 1 provides an introduction to the SQL Server.
- Chapter 2 provides information about accessing remote databases.
- Chapter 3 provides information about building an IVR application that interfaces with an SQL database.

Supporting Release 4.0 documentation

Besides this guide, you may find it useful to have these books available.

Symposium OPEN IVR Application Development Guide

Symposium OPEN IVR Cell Catalog

Symposium OPEN IVR Error Messages Guide

Symposium OPEN IVR Product Guide

Symposium OPEN IVR System Administration Guide

Conventions used in this guide

Throughout this guide, several typographic conventions have been used to highlight certain types of information:

- Items that appear on the Symposium OPEN IVR screens are identified in quotes, for example:

  the “Timeout” part of the parameters window

- Prompts, File and Menu names, Directories, Accounts, and Options are shown in a different typeface, for example:

  `/sci>`

  `sci/install` menu

  `/u/nortel/3270/getbalance.act` file

- Symposium OPEN IVR buffer names are shown in all uppercase characters, for example:

  the CURRENT MESSAGE buffer

- Field names are shown in italics, for example:

  the *Device* field

- Items you must type are shown in bold in a different typeface, for example:

  type **`tn5250`** at the prompt

- Variables shown in command lines appear in italics, for example:

  the `host_cfg`*n* file, where *n* is a variable representing a board number

- Key names you press are shown in angled brackets, for example:

  the <F1> key

- Keyboards usually have keys named <Return> or <Enter> that perform the same function. For convenience, this guide uses the keyname <Enter> to represent both keynames.

Chapter 1: Introduction

The Symposium OPEN IVR SQL Server interfaces with several SQL-based ANSI-compliant relational and non-relational database management systems (DBMS) that enable your applications to retrieve, insert, update, and delete information stored in the supported databases.

This chapter provides information about the following topics:

- SQL server overview
- getting started
- designing an application

SQL Server overview

AP software and IVR Developer support application access to a message database and provide a method for building a simple database with the Information Database Editor. You can also develop applications that access and update SQL-based databases, using the cells in Table 1-1.

Table 1-1: SQL cells

Cell	Description
QCMD	SQL command cell
QCNT	SQL select count cell
QDEL	SQL delete cell
QINS	SQL insert cell
QSEL	SQL select cell
QROW	SQL retrieve row
QUPD	SQL update cell

With the exception of the QCMD cell, these cells execute one standard SQL SELECT, INSERT, UPDATE, or DELETE statement. You can define access to database tables or views from within your applications. To make more complex queries, you can issue multistatement transactions either in the QCMD cell or in embedded SQL for C (ESQL/C) in a USER cell.

The Symposium OPEN IVR SQL server is compatible with the following RDMS that support ANSI SQL:

- Informix-SE Version 7.2
- Ingres 6.4/06; Open INGRES 1.2/01
- Oracle 7.3
- Sybase 10.0.3 Client Open is supported to work with Sybase Open System 10 and 11 Server
- Microsoft SQL Server 6.5 on Windows NT through OpenLink

To manipulate information in a database, you normally type one or more SQL statements at your keyboard or include embedded SQL commands in a C program. The Symposium OPEN IVR SQL Server cells are designed to execute SQL statements as well, but from within an AP software application.

Note: One exception is the QCMD cell.

When you include an SQL cell in an application, you specify the same information as contained in a simple SQL statement. You do not need to know the correct SQL syntax in order to use SQL cells. You only need to reference the correct table and column names. The SQL Server cells execute the correct SQL commands based on the cell parameters that you specify.

The differences between SQL statements and the SQL Server cells follow.

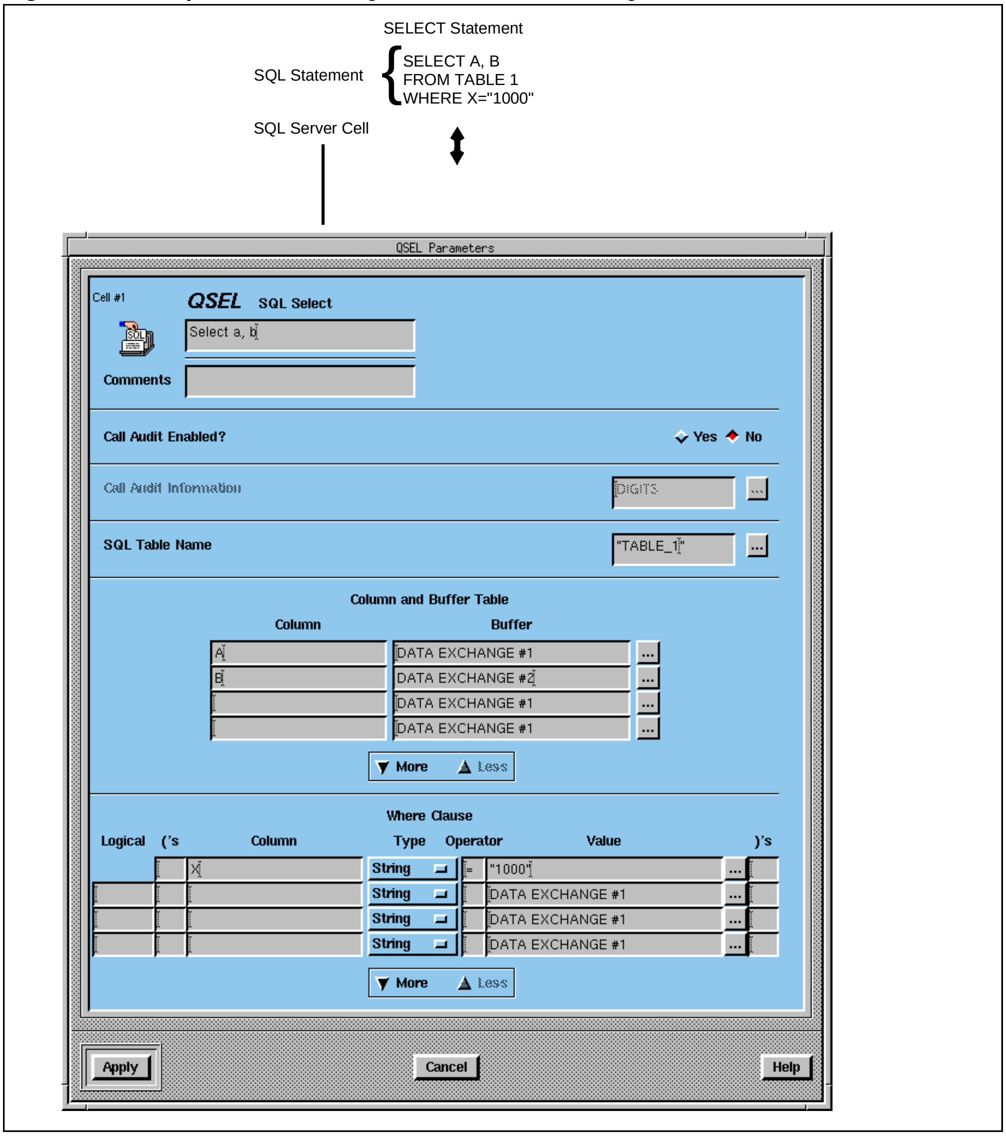
- Only one table for each SQL cell is allowed. Consequently, to perform a join, you need to build a view and then reference the name of the view in the table name parameter of the appropriate cell table. Finally, you build a view using your DBMS.

  Note: One exception is the QCMD cell, which can access two tables.

- With the exception of the QCMD cell and when you are building a view, complex queries, such as subqueries, are not supported in the SQL server files.
- With the exception of the QCMD cell, SQL expression syntax is not supported.
- With the exception of the QCMD cell, each cell represents one SQL transaction. Each SQL statement is committed if it is successful, and rolled back if it fails.You cannot roll back the cell transaction once the cell completes processing.

  Figure 1-1 on page 1-4 illustrates a simple SQL statement as well as the equivalent information in the QSEL cell.

Figure 1-1: Comparison of an SQL statement and an SQL cell



```
SELECT A, B
FROM TABLE 1
WHERE X="1000"
```

QSEL Parameters

Cell #1 **QSEL** SQL Select

Select a, b

Comments

Call Audit Enabled? Yes [✓ No]

Call Audit Information DIGITS

SQL Table Name "TABLE_1"

Column and Buffer Table

Column	Buffer
A	DATA EXCHANGE #1
B	DATA EXCHANGE #2
	DATA EXCHANGE #1
	DATA EXCHANGE #1

More Less

Where Clause

Logical	('s	Column	Type	Operator	Value	)'s
		X	String	=	"1000"	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	

More Less

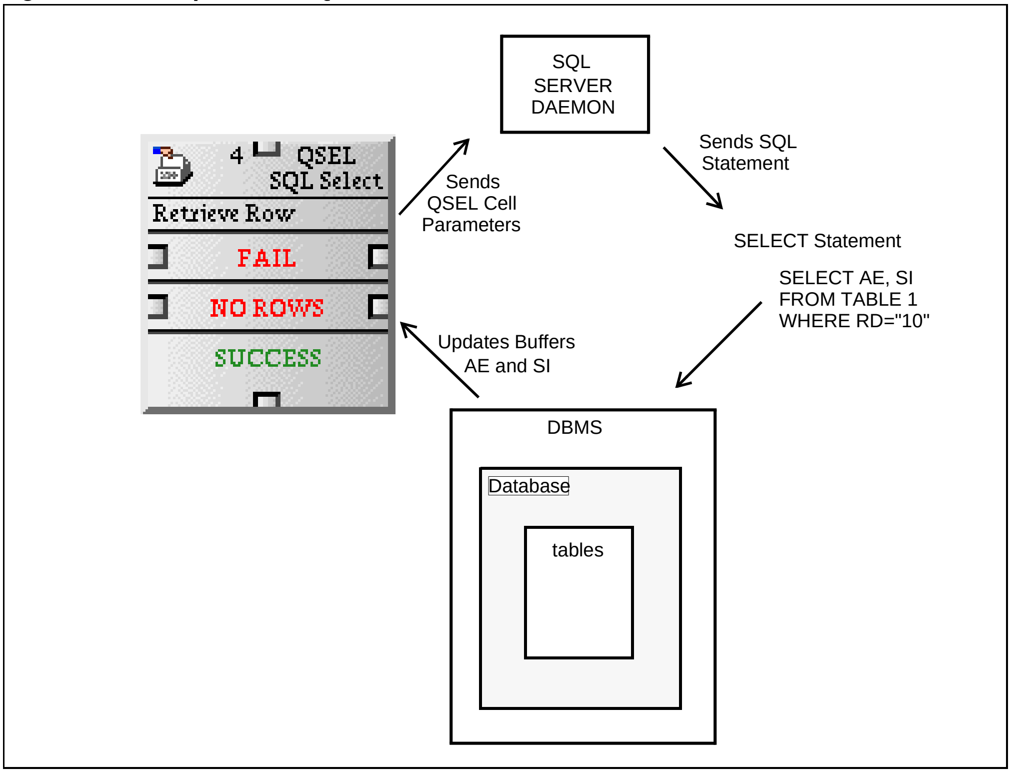
Apply Cancel Help

SQL Server background processes

When you boot your system, the AP software automatically loads the SQL server as a daemon process, or as middleware if you are using a remote server. The server manages all requests by callers for information stored in the database and executes the corresponding SQL statements against the DBMS. The data that the caller can manipulate is determined by how you define the SQL cells in your application and the privileges you grant to the database user.

Figure 1-2 displays the flow of an SQL cell accessing data from an SQL-based database.

Figure 1-2: Example of an SQL transaction



In this example, when the QSEL cell is reached during execution of the application, the cell parameters are passed to the SQL server daemon. The daemon process submits the corresponding SQL SELECT statement to the DBMS and retrieves the first row that matches the statement's selection criteria. The daemon process updates the output buffers identified by the QSEL cell, and the application continues processing with the next cell.

Getting started

Perform the following procedures before you run an application that accesses an SQL-based database:

- install the DBMS client software on the AP
- configure the DBMS
- configure an ODBC connection, if appropriate
- create or updating the database
- set up the user account
- grant access privileges

Installing the DBMS client software on the AP

The Symposium OPEN IVR SQL server is compatible with the following RDMS that support ANSI SQL:

- Informix-SE Version 7.2
- Ingres 6.4/06; Open INGRES 1.2/01
- Oracle 7.3
- Sybase 10.0.3 Client Open is supported to work with Sybase Open System 10 and 11 Server
- Microsoft SQL Server 6.5 on Windows NT through OpenLink

  Note: You purchase and install the DBMS separately from the AP and IVR Developer software. Refer to the documentation that accompanies your specific database or ODBC driver software for additional information about installation procedures.

Note: Locally installed server software should be installed in a directory under the `/dev/u/` division. Otherwise, you can install the SQL database on a remote database server. An SQL database should never be installed locally into the root division. Doing so could result in filling up the root division and crashing the system. In all cases, you should install the client software under the `/dev/u` division.

Configuring the DBMS

You need to edit the corresponding configuration file for each DBMS in the AP software directory `sys_files`. Refer to the appropriate information below about editing the configuration files in the directory `sys_files`.

Informix

Edit the file `sys_files/informix.conf` and set the environment variables `DBPATH`, `INFORMIXSERVER`, and `INFORMIXDIR` to the values recommended in your Informix installation notes.

Ingres

Edit the file `sys_files/ingres.conf` and set the environment variable `II_SYSTEM` to the value recommended in your Ingres installation notes.

Oracle

Edit the file `sys_files/oracle.conf` and set the environment variables `ORACLE_HOME` and `ORACLE_SID` to the values recommended in your Oracle installation notes.

Sybase

Edit the file `sys_files/sybase.conf` and set the environment variables `DSQUERY`, `SYBASE,` and `SYBPLATFORM` to the values recommended in your Sybase installation notes.

MS SQL Server

Edit the file `sys_files/odbc.conf` and set the environment variables as follows:

ODBCDIR: Set the environment variable to the path where the ODBC driver is installed.

LD_LIBRARY_PATH

PATH: Append ODBCDIR and ODBCDIR/lib to the environment variable PATH.

Note: Do not overwrite existing paths.

Configuring an ANSI-compliant database management connection

Open Database Connectivity (ODBC) is a standards-based Application Programming Interface (API) that allows database clients to be configured to access a wide array of ANSI-compliant database management servers.

Note: This API is used to access the MS SQL Server from your AP software. It is not used to access Informix, Ingres, Oracle, or Sybase.

Perform the following steps to allow your IVR applications to access remote ANSI-compliant databases. Refer to the documentation that came with your specific ODBC driver software for more information about configuring your software.

Before you start this procedure, you need to know the following information:

• user name	• name of the remote database
• datasource name	• type of network
• local path of the driver software	• remote database path name
• description of the driver software	• DBMS server node name
• user identification	

The ODBC driver software configuration file`odbc.ini` is an index to ODBC data sources that are available. The file`odbc.ini` contains information about the DBMS type, the driver name, and the network type.

The following example displays a typical Visigenic `.odbc.ini` file:

```
[ODBC Data Sources]
Username=Visigenic MS SQL Server

[Datasource_Name]
Driver=Driver_Pathname
Description=Microsoft SQL Server ODBC Driver
UID=User_ID
Database=Remote_Database_Name
Network=Network_Type
Address=Hostname
UserProcForPrepare=yes

[ODBC]
Trace=0
TraceFile=odbctrace.out
InstallDir=Directory
```

Creating or updating databases

When the DBMS is installed on the AP, you can use the tools associated with your DBMS to create or update the appropriate tables and views. Refer to the documentation that accompanies your particular DBMS for more information about creating or updating the database.

Setting up user accounts

Only users with authorized user accounts can access your database. Refer to the documentation that came with your particular DBMS for more information about setting up user accounts.

Granting access privileges

To ensure that callers can access specific data while preserving the security of the database, the database administrator can grant the appropriate access privileges to those tables and views in the database that may be accessed by the user and the user account running AP software. Consult your database administrator for more information about granting accessing privileges.

Planning an application

You need to consider the following questions before you build an application that involves SQL cells:

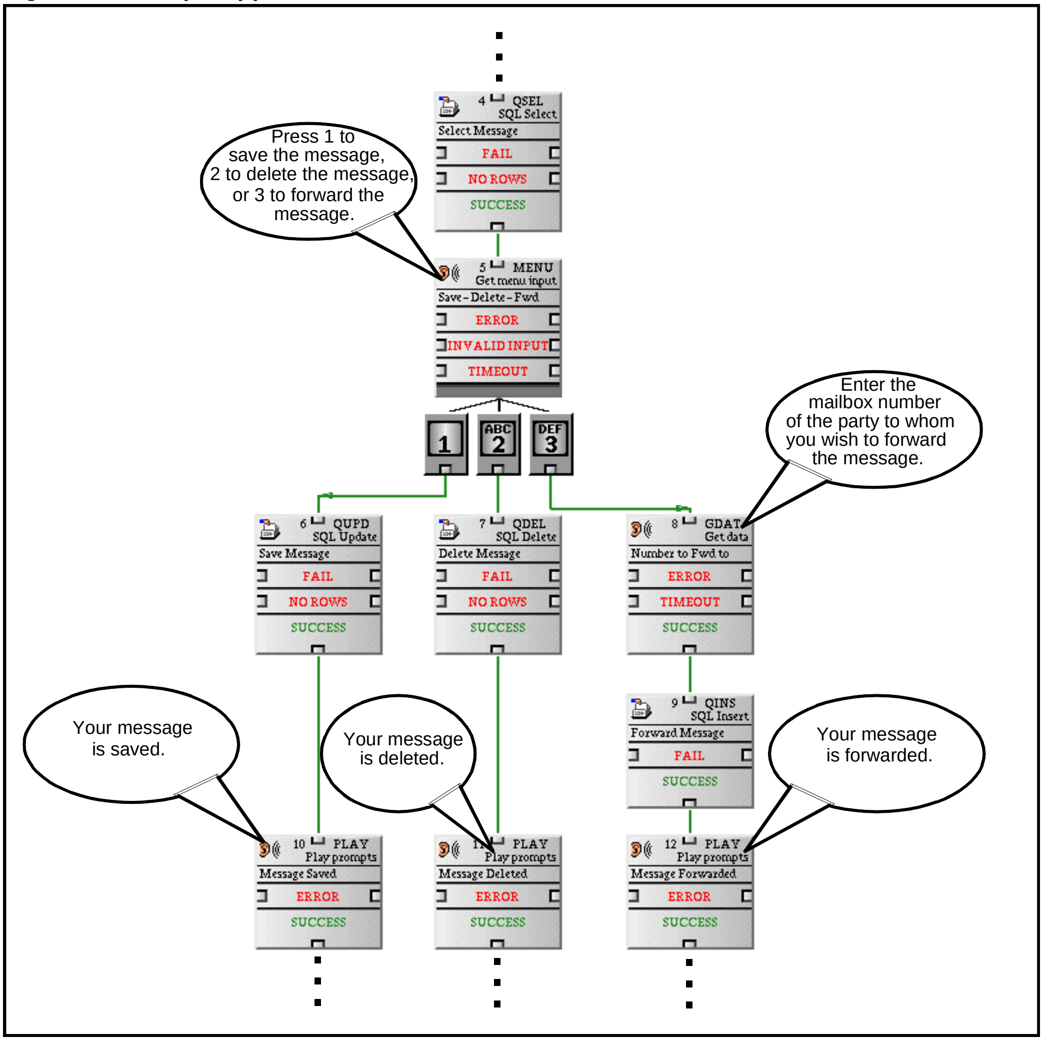
- What information do you want callers to be able to manipulate?
- Where is the information stored? For example, what is the name of the database and the specific tables?
- Should callers be able to retrieve, insert, update, and delete data?

Once you know the names of the databases, tables, and columns where the data resides and establish the appropriate privileges to the AP software accounts, you can build an application that accesses this data.

> ***Note:*** The SQL database does not need to reside on the AP. See Chapter 2, “Accessing remote databases,” for more information about accessing remote databases.

Figure 1-3 on page 1-11 shows an example of an application that uses SQL cells to insert, delete, and update information in a database based on a comparison made between the input of the caller and the contents of several buffers.

Figure 1-3: Sample application call flow



Chapter 2: Accessing remote databases

This chapter provides information about the following topics:

- establishing a direct connection
- establishing an ANSI-compliant connection

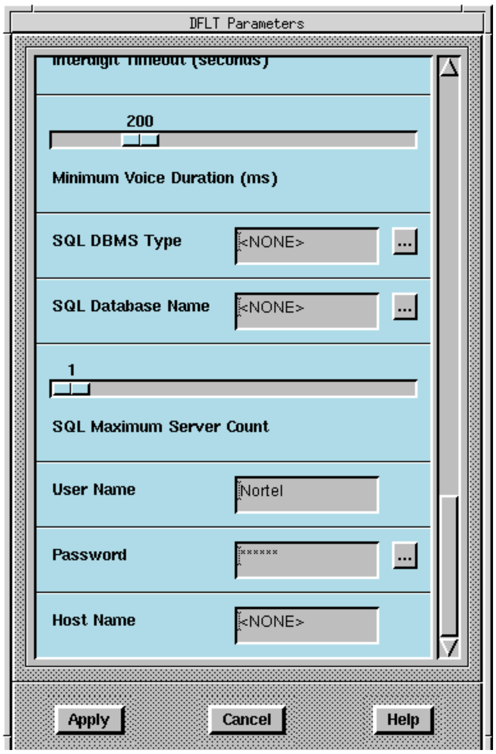
Establishing a direct connection

If an application uses SQL cells that access a directly connected SQL database, such as an Informix database that resides on the LAN, then you need to specify the following values in the START cell:

- SQL DBMS type
- SQL database name
- SQL maximum server count

Figure 2-1 on page 2-2 displays the START cell parameter page.

Figure 2-1: START cell parameter page


DFLT Parameters

Interdigit Timeout (seconds)

200

Minimum Voice Duration (ms)

SQL DBMS Type <NONE>

SQL Database Name <NONE>

1

SQL Maximum Server Count

User Name Nortel

Password ******

Host Name <NONE>

This section provides information about the following START cell parameters:

- SQL DBMS Type
- SQL Database Name
- SQL Maximum Server Count

Note: Refer to the *Symposium OPEN IVR Cell Catalog* for more information about the START cell parameters.

SQL DBMS type

Specify the DBMS type that you use for your application.

SQL database name

Specify the database name. The database must exist when the application is loaded. Although the buffer allows a maximum of 31 characters in length for the database name, Informix supports a maximum of 8 characters, and Sybase supports a maximum of 12.

Note: In order to use ANSI-compliant database management functionality with the Microsoft SQL Server, specify the SQL datasource name, not the database name, in the parameter `SQL Database Name`.

SQL maximum server count

Specify the number of servers, from 1 to 99. The number of servers you specify depends on the number of queued requests. The recommended number of servers at this time is one.

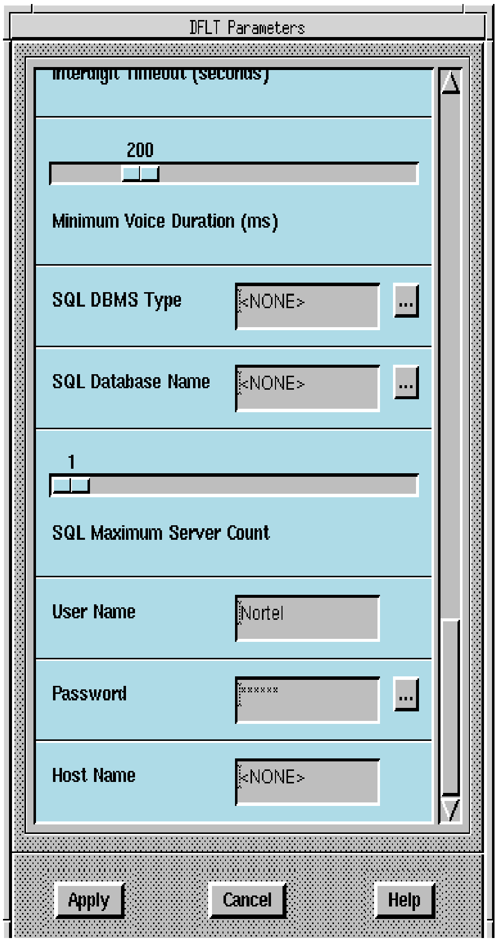
ATTENTION

System performance is significantly affected by increasing the number of IVR data server processes. Increase the number of IVR data server processes only when necessary.

Establishing an ANSI-compliant database management connection to the Microsoft SQL Server

Using the Open Database Connectivity (ODBC) feature, the application developer can access databases that are not supported by UNIX systems by defining the parameter `SQL DBMS Type` in the START cell as `odbc`.

Figure 2-2: DFLT cell parameters



DFLT Parameters

Interdigit Timeout (seconds)

200

Minimum Voice Duration (ms)

SQL DBMS Type <NONE>

SQL Database Name <NONE>

1

SQL Maximum Server Count

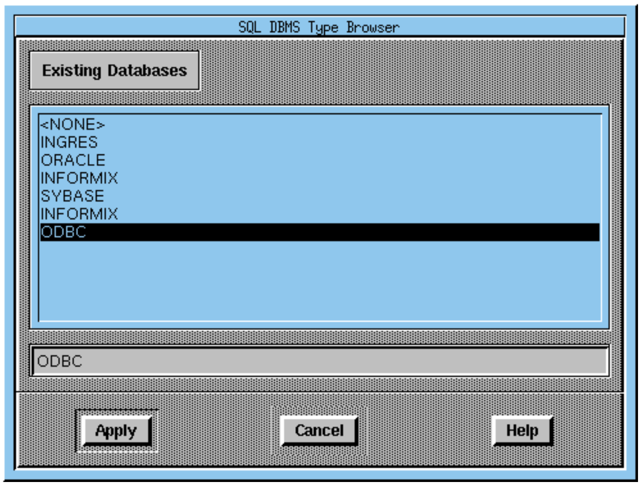
User Name Nortel

Password ******

Host Name <NONE>

If you click the button for the parameter `SQL DBMS type`, the system displays the SQL DBMS Type Browser window as shown in Figure 2-3.

Figure 2-3: SQL DBMS Type Browser window



<NONE>
INGRES
ORACLE
INFORMIX
SYBASE
INFORMIX
ODBC

ODBC

You need to install the optional ODBC driver software and copy the vendor-specific initialization file to the directory in which AP software and IVR Developer are installed before you enable the `obdc` value in the START cell. For more information about the ANSI-compliant databases, refer to the *Symposium OPEN IVR Product Guide.*

If an application uses SQL cells that access an ANSI-compliant database, then you need to specify the following values in the START cell:

- DBMS type
- database name
- server count

SQL DBMS type

Specify the option `ODBC` as the `DBMS` type.

SQL database name

You need to specify the SQL datasource name, and not the database name, in order to use ODBC functionality. The datasource specifies a particular configuration that resides in the file `odbc.ini.`

SQL maximum server count

Specify the number of servers to use, from 1 to 99. The number of servers you specify depends on the number of queued requests. For a development system, the recommended number is 1. For other systems, the number of servers depends on the processes you are running concurrently.

ATTENTION

Overall system performance is significantly affected by increasing the number of IVR data server processes. Increase the number of IVR data server processes only when necessary.

Controlling SQL application execution for server startup and shutdown

Whenever you boot or reboot your AP, the SQL applications will need to be unloaded, stopped, reloaded, and restarted. This can be accomplished with the LOAD, UNLOAD, START, and STOP commands available from the Application Management screen, which is accessible from the Application Management option under the Administration tools.

For more information about these commands and their associated procedures, refer to the *Symposium OPEN IVR System Administration Guide*.

Note: If you have an error in any of your connection parameters, your file will not load.

Chapter 3: Building an SQL application

The Symposium OPEN IVR SQL Server includes cells that you can use in your application to allow callers to manipulate data in your database. This chapter provides information about the following topics:

- performing multistatement transactions with the QCMD cell
- counting selected rows with the QCNT cell
- deleting a row with the QDEL cell
- inserting a row with the QINS cell
- retrieving information with the QSEL cell
- retrieving a row with the QROW cell
- updating a row with the QUPD cell

If you are familiar with standard and embedded SQL statements, you can use the SQL cells to design applications that access databases.

For more information about creating applications for your system, refer to the *Symposium OPEN IVR Application Development Guide* and the *Cell Catalog*.

Performing multistatement transactions with the QCMD cell

The QCMD cell is an advanced SQL cell capable of performing functions which would otherwise require using C language and embedded SQL programming through a USER cell. You can also use the QCMD cell to perform multistatement SQL transactions within a particular application. The parameter `Commit?` in the QCMD cell allows you to specify whether the cell commits after performing a transaction or waits for another command to be issued.

Other SQL cells perform an automatic commit after performing a single transaction. For example, if you use the QUPD cell to update a specified database, the cell commits after performing the transaction. This behavior limits the ability of the application developer to string a series of SQL cells together to form a single, logical database transaction.

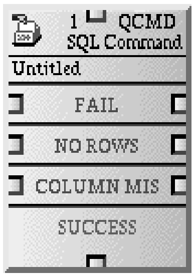
CAUTION

Risk of system corruption

The QCMD cell is intended to be used by application developers who are proficient in SQL.

Figure 3-1 displays the QCMD cell face.

Figure 3-1: QCMD cell



QCMD cell branches

The QCMD cell includes the following branches:

- Fail
- No Rows
- Column Mismatch
- Success

Fail

The QCMD cell cannot perform the SQL transaction specified in the parameter `SQL Statement`. Inspect the buffer SQL Error Code for details about the error condition. You can also check the IVR Transaction Log and the SQL error log for additional information.

For more information, refer to the *Symposium OPEN IVR Error Messages Guide* and to the documentation supplied with your SQL server.

No Rows

No rows are affected or retrieved.

Column Mismatch

If you supply less buffers than columns in the specified select query, then the cell takes the branch Column Mismatch.

Success

The cell successfully performs the SQL statement specified in the parameter `SQL Statement`.

QCMD cell parameters

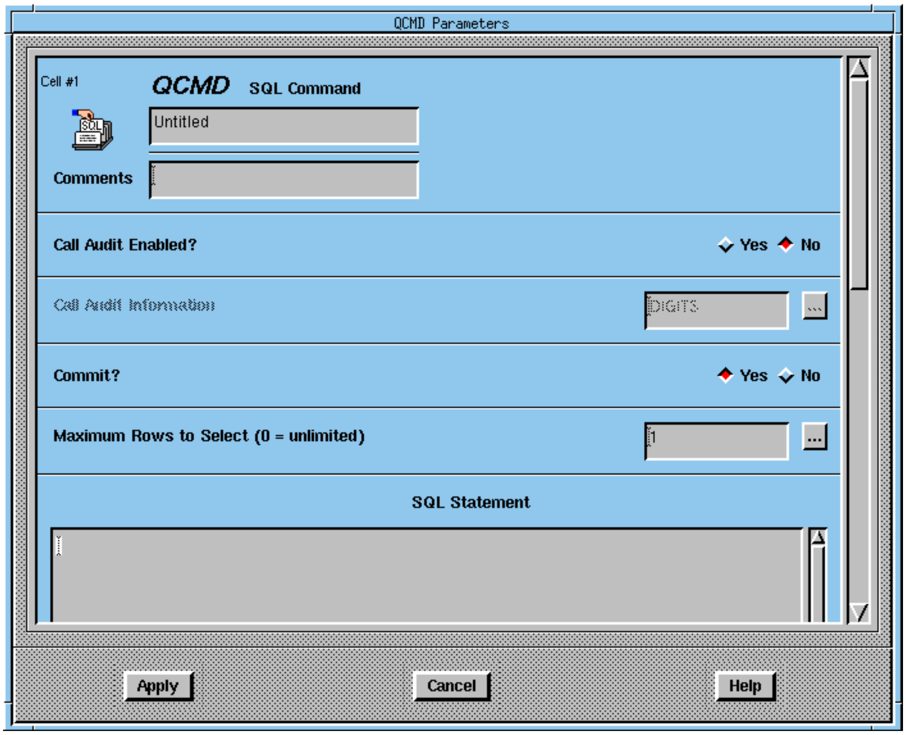
The QCMD cell contains the following parameters:

- `Call Audit Enabled?`
- `Call Audit Information`
- `Commit?`
- `Maximum Rows to Select`
- `SQL Statement`

- `Parameter Buffers`
- `Return Buffers`

Figures 3-2 and 3-3 show the QCMD cell parameter pages.

Figure 3-2: QCMD cell (part 1)



QCMD Parameters

Cell #1

QCMD SQL Command: Untitled

Comments:

Call Audit Enabled? Yes [✓ No]

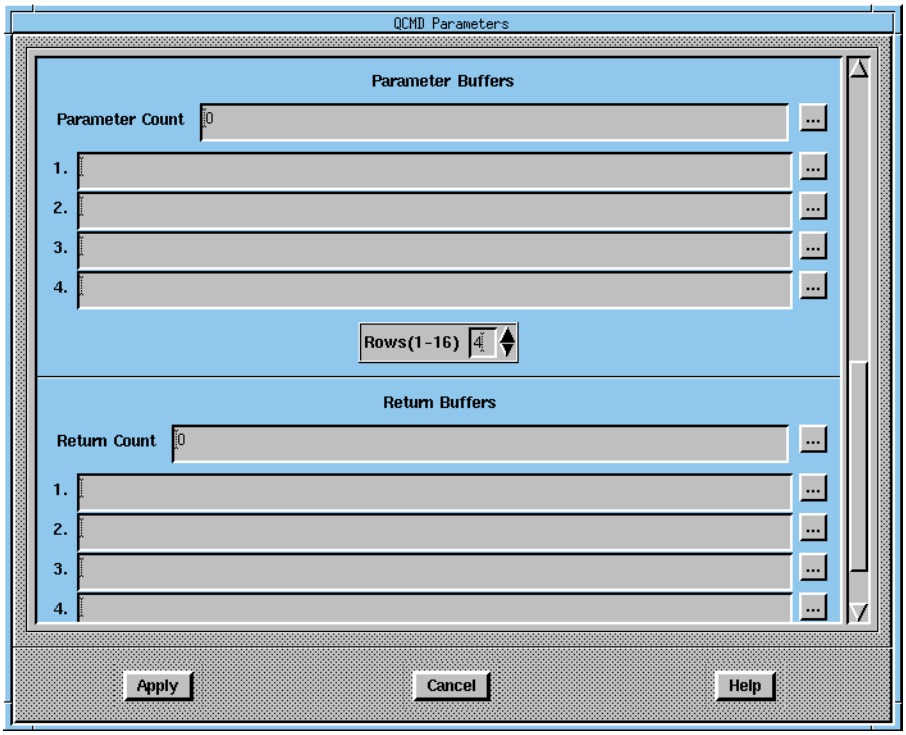
Call Audit Information: DIGITS

Commit? [✓ Yes] No

Maximum Rows to Select (0 = unlimited): 1

SQL Statement

Figure 3-3: QCMD cell (part 2)



QCMD Parameters

Parameter Buffers

Parameter Count: 0

1.
2.
3.
4.

Rows(1-16): 4

Return Buffers

Return Count: 0

1.
2.
3.
4.

Call Audit Enabled?

The default setting of the parameter `Call Audit Enabled?` is No. If you set this parameter to Yes, the QCMD cell logs the following information to the call audit statistics file, `audit_stat.d`:

- application name
- cell name
- cell number
- date and time of cell execution
- contents of the cell comment field
- contents of the Call Audit Information buffer

Call Audit information

If you set the parameter `Call Audit Enabled` to Yes, the contents of the buffer specified in the parameter `Call Audit Information` are logged to the file `audit_stat.d`.

Commit?

The default setting is Yes. When set to No, this parameter allows a series of SQL statements in QCMD cells to be strung together to form a multistatement transaction. If, however, you set the parameter `Commit` to `No`, and follow the QCMD cell with another type of SQL cell such as QSEL, the transaction is automatically committed after you exit from the other SQL cell.

Maximum Rows to Select

The parameter `Maximum Rows to Select` allows you to specify the maximum number of rows you want the QCMD cell to return. For no limit to the number of rows, select `0`. The default value is `1`. The first row is returned by the QCMD cell. Subsequent rows are returned by the QROW cell, typically in a loop.

SQL Statement

Use this text field to enter in your SQL statement(s). Parameters that are supplied at runtime are marked with question marks. Backslashes quote question marks and allow for their literal use. Values marked by question marks are received by the parameter `Parameter Buffers` in the same order in which they are entered. For example, the first question mark corresponds to the first value listed in the parameter `Parameter Buffers`, and the second question mark corresponds to the second value.

Parameter Buffers

Enter the values of your specific query in this parameter. Use these parameters for your input. For example, if you need to query the database for the names of all the employees of a certain department, then you need to specify the department in one of the fields in the parameter `Parameter Buffers`.

Return Buffers

The parameter `Return Buffers` specifies the database location of the retrieved values. Use this parameter for the output from the database.

Using multistatement transactions in the QCMD cell

If the parameter `Commit?` is set to No, you can string together a series of SQL commands for concurrency.

> ***Note:*** If you are not familiar with the required considerations for transactional concurrency in a relational DBMS, it is recommended that you leave the parameter `Commit?` set to Yes.

Before you create an application that involves the QCMD cell, consider the following criteria:

- Determine the number of IVR data server processes that you need. AP software allows a number of processes between 0 and 99 for each specific type.
- Determine the number of multistatements in all applications.
- Determine the total elapsed time that the SQL transactions are active.
- Determine the amount of transactional concurrency that is already in use. For example, account for a database that is accessed by an application other than AP software.

Using SQL statements in the QCMD cell

You can specify any standard SQL statement that your particular DBMS supports. You can specify standard SQL statements to perform one of the three SQL tasks:

- data retrieval
- data manipulation
- database management
- execution of stored procedures

> ***Note:*** The QCMD cell does not validate SQL statements. When the QCMD cell is executed, the contents of the parameter `SQL Statement` are passed to the DBMS exactly as they are typed.

You can specify the available SQL commands in the text field `SQL Statement` to create a multistatement transaction. Table 3-1 lists examples of standard SQL statements.

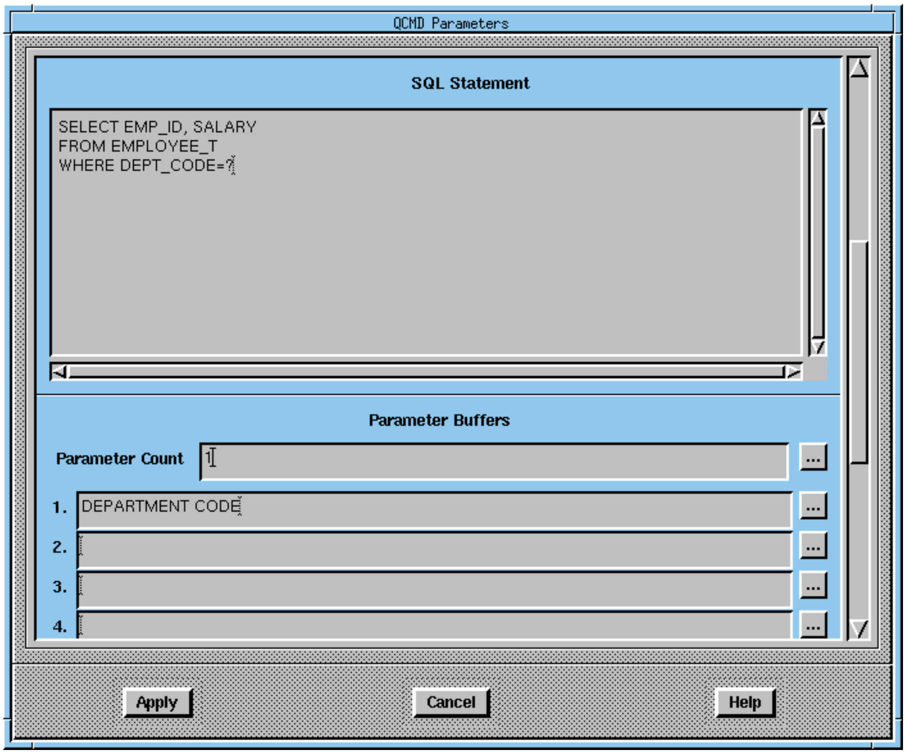
Table 3-1: Primary types of QCMD cell SQL statements

Type	Example	Description
Data Retrieval	Select	Retrieves a relational set from zero or more SQL tables. The first row is retrieved into specified buffers. Subsequent rows are retrieved using the existing QROW cell.
Data Manipulation	Update/Insert	Alters values from within SQL tables. Usually requires values supplied by the application in buffers.
Database Management	Delete/ Misc.	Deletes rows from within SQL tables, or performs other database specific tasks that neither require nor return values.
Data Retrieval and Manipulation	Select	Requires values supplied by the application in the where clause and retrieves values from the DBMS into buffers.

QCMD cell example transaction

You can use a complex SQL statement in the parameter `SQL Statement` to retrieve information. For example, if you need to provide a list of employees and salary amounts from a department, which is specified at runtime from within a specific table, use the SQL command SELECT to specify the information you want to retrieve. You also need to use the SQL command FROM to specify the name of the table from which this information is accessed. Finally, you need to provide the SQL command WHERE in order to accommodate the Department Code, which is specified during runtime by the caller. Figure 3-4 on page 3-9 shows an example of the QCMD cell with a simple SQL statement in the parameter `SQL Statement` to accomplish the criteria listed previously in this paragraph.

Figure 3-4: QCMD example (part 1)



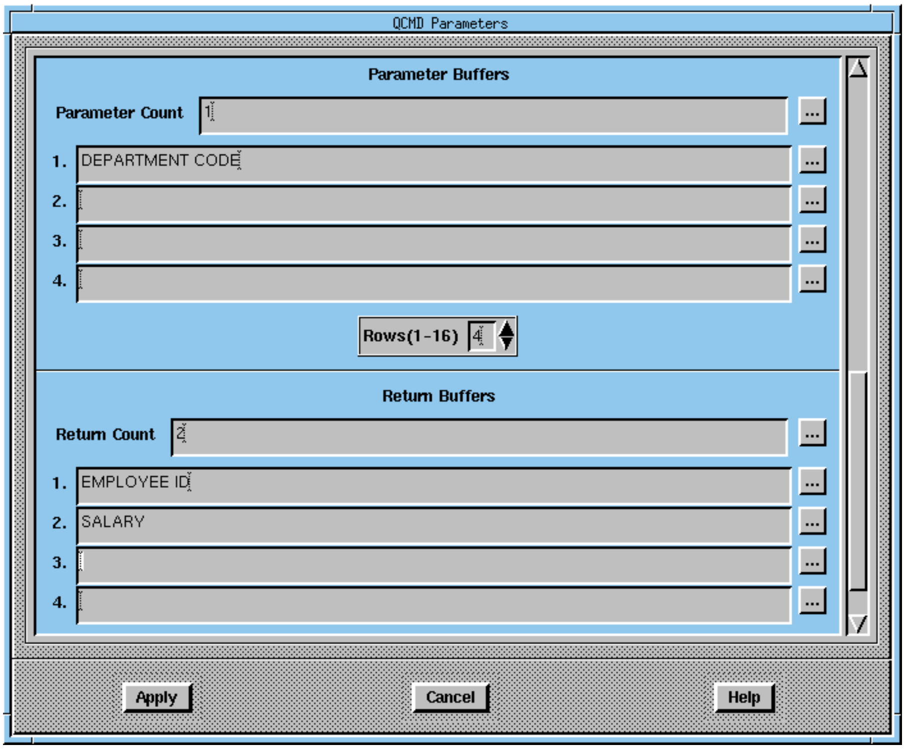
```
SELECT EMP_ID, SALARY
FROM EMPLOYEE_T
WHERE DEPT_CODE=?
```

Parameter Count: 1

1. DEPARTMENT CODE
2.
3.
4.

The character `?` enables access to this information during runtime. The entry DEPARTMENT CODE is a variable used only for this example. Figure 3-5 on page 3-10 shows the parameter `Return Buffers`.

Figure 3-5: QCMD example (part 2)



Parameter Count: 1

1. DEPARTMENT CODE
2.
3.
4.

Rows(1-16): 4

Return Count: 2

1. EMPLOYEE ID
2. SALARY
3.
4.

The parameter `Return Buffers` lists the values specified by the command SELECT in the parameter `SQL Statement`. The order of the returned values matches that of the command. In this example, the first value returned from the specified database is the EMPLOYEE ID, and the second value is SALARY.

SQL commands to avoid

Avoid any command that directly interacts with the database to change concurrency, locking, and transaction control.

Sybase SQL commands to avoid

Depending on the particular DBMS, certain statements are supported while others may cause unexpected behavior in the IVR system. The known statements that can cause problems are listed in this section. Table 3-2 lists the commands to avoid for the Sybase DBMS.

Note: Do not execute any command that affects metadata.

Table 3-2: Sybase commands to avoid

Command	Description
transaction isolation level	ANSI compliance.
chained mode	Integrity of individual commands and commit/rollback mechanics.
disconnect	Disconnects from the database.
connect	Connects to a database.
dynamic SQL	Cannot be used from within a IVR application.
use	Changes the default database to which the session is connected.
Metadata commands	Create and delete tables, databases, users, table spaces, and so on.

Ingres SQL commands to avoid

Table 3-3 lists the commands to avoid for the Ingres DBMS.

> ***Note:*** Do not execute any command that affects metadata.

Table 3-3: Ingres commands to avoid

Command	Description
set lockmode	Changes the default locking strategy.
disconnect	Disconnects from database.
connect	Connects to a database.
dynamic SQL	Cannot be used from within an IVR application.
Metadata commands	Create and delete tables, databases, users, table spaces, and so on.

Informix-SE SQL commands to avoid

Table 3-4 lists the commands to avoid for the Informix-SE DBMS.

> ***Note:*** Do not execute any command that affects metadata.

Table 3-4: Informix SE commands to avoid

Command	Description
database	Attaches the session to a database.
close database	Disconnects the session from the database.
dynamic SQL	Cannot be used from within an IVR application.
Metadata commands	Create and delete tables, databases, users, table spaces, and so on.

Oracle SQL commands to avoid

Table 3-5 lists the commands to avoid for the Oracle DBMS.

> ***Note:*** Do not execute any command that affects metadata.

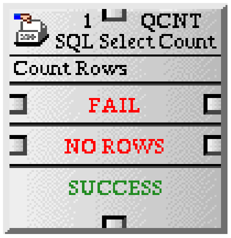
Table 3-5: Oracle commands to avoid

Command	Description
commit	Commits transactions.
release	Disconnects the session from the database.
connect	Connects the session to a database.
dynamic SQL	Cannot be used from within an IVR application.
Metadata commands	Create and delete tables, databases, users, double-spaces, and so on.

Counting selected rows with the QCNT cell

You can use the QCNT (SQL SELECT COUNT) cell to count the number of rows in a table that match specific criteria. The QCNT cell parameters identify the table to be accessed and the WHERE clause selection criteria. Figure 3-6 shows the QCNT cell.

Figure 3-6: QCNT cell



QCNT cell branches

The QCNT cell includes the following cell branches:

- Fail
- No Rows
- Success

Fail

The QCNT cell cannot perform the SQL transaction. Inspect the buffer SQL Error Code for details about the error condition. You can also read the IVR transaction log and the DBMS error log for additional information.

No Rows

No rows are affected or retrieved.

Success

The cell successfully performs the SQL transaction. The system buffer SQL ROWS contains the number of rows that matched the specified criteria.

QCNT cell parameters

The QCNT cell contains the following parameters:

- `Call Audit Enabled?`
- `Call Audit Information`
- `SQL Table Name`
- `Where Clause`

Call Audit Enabled?

The default setting of the parameter `Call Audit Enabled?` is No. If you set this parameter to Yes, the QCNT cell logs the following information to the call audit statistics file, `audit_stat.d`:

- application name
- cell name

- cell number
- date and time of cell execution
- contents of the cell comment field
- contents of the Call Audit Information buffer

Call Audit Information

If you set the parameter `Call Audit Enabled` to Yes, the contents of the buffer specified in the parameter `Call Audit Information` are logged to the file `audit_stat.d.`

SQL table name

This is the name of the table containing the rows that you wish to count.

WHERE Clause

You can enter one or more WHERE clauses. You can select a value by either typing the value in the text field or by selecting the field and pressing mouse button three. The system displays a menu that contains the valid values for that particular field. Move the cursor to the value you want and release mouse button three. The value is displayed in the field. You can specify values for the following fields:

- Logical
- (‘s
- Column
- Type
- Operator
- Value
-)’s

Logical

You can enter a logical (Boolean) operator (`AND, NOT, ANDNOT, ORNOT`, or `OR`) when specifying a compound expression.

(‘s

You can enter the left parenthesis as single— **(** —, double— **((** —, or triple— **(((**.

Column

You can enter the name of a column, up to a maximum of 31 characters.

Operator

You can enter the relational operator, such as **`=, !=, >, <, >=, <=`** (equal, not equal, greater than, less than, greater than or equal to, less than or equal to).

Value

You can enter a value against which the query is matched. The value can be a system buffer, an application buffer, a number, or a hard-coded value enclosed in double quotes.

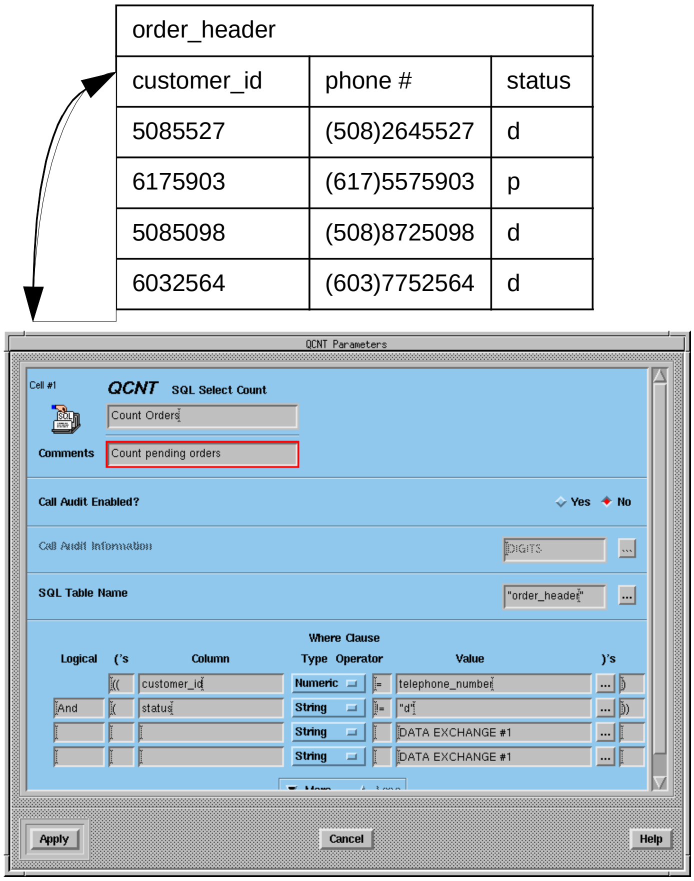
)’s

You can enter closing parenthesis as ingle— **)** —, double— **))** —, or triple— **)))**.

QCNT cell example transaction

You can use the QCNT cell within an application to count the number of outstanding orders in an existing customer account. For example, you can use the QCNT cell to count the number of rows in the table “order_header” where the `customer_id` field is the caller’s identification number. In this example, the information is derived from the telephone number. The `status` field is not set to “`d`” for “delivered.” Figure 3-7 on page 3-17 displays the QCNT cell used in this example, as well as the table “order_header” that is accessed.

Figure 3-7: Example QCNT cell transaction



order_header		
customer_id	phone #	status
5085527	(508)2645527	d
6175903	(617)5575903	p
5085098	(508)8725098	d
6032564	(603)7752564	d

QCNT Parameters

Cell #1 **QCNT** SQL Select Count

Count Orders

Comments: Count pending orders

Call Audit Enabled? Yes [✓ No]

Call Audit Information: DIGITS

SQL Table Name: "order_header"

Where Clause

Logical	('s	Column	Type	Operator	Value	)'s
	((	customer_id	Numeric	=	telephone_number	)
And	(	status	String	!=	"d"	))
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	

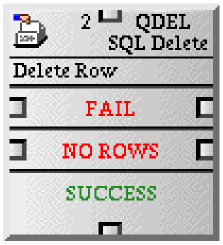
In this example, the QCNT cell performs the following SQL statement:

```
SELECT COUNT
    FROM order_header
    WHERE ((customer_id = :telephone_number)
        AND (status! = “d”)
```

Deleting a row with the QDEL cell

The QDEL (SQL DELETE) cell allows you to delete a row in a database table. You can specify the rows to delete in a table. Figure 3-8 shows the QDEL cell.

Figure 3-8: QDEL cell



QDEL cell branches

This section provides information about the following QDEL cell branches:

- Fail
- No Rows
- Success

Fail

The QDEL cell cannot perform the SQL transaction. Inspect the buffer SQL error code for details about the error condition. You can also read the IVR transaction log and the DBMS error log for additional information.

No Rows

No rows are affected or deleted.

Success

The cell successfully performs the SQL transaction. The system buffer SQL ROWS contains the number of rows that matched the specified criteria.

QDEL cell parameters

The QDEL cell contains the following cell parameters:

- `Call Audit Enabled?`
- `Call Audit Information`
- `SQL Table Name`
- `Where Clause`

Call Audit Enabled?

The default setting of the parameter `Call Audit Enabled?` is No. If you set this parameter to Yes, the QDEL cell logs the following information to the call audit statistics file, `audit_stat.d:`

- application name
- cell name
- cell number
- date and time of cell execution
- contents of the cell comment field
- contents of the Call Audit Information buffer

Call Audit Information

If you set the parameter `Call Audit Enabled` to Yes, the contents of the buffer specified in the parameter `Call Audit Information` are logged to the file `audit_stat.d`.

SQL table name

This is the name of the table containing the rows that you wish to delete.

WHERE clause

You can enter one or more WHERE clauses. You can select a value by either typing the value in the text field or by selecting the field and pressing mouse button three. The system displays a menu that contains the valid values for that particular field. Move the cursor to the value you want and release mouse button three. The value appears in the field.

> ***Note:*** If you do not specify any rows in the parameter `Where Clause`, then the QDEL cell deletes all the rows in the specified table.

You can specify values for the following fields:

- Logical
- (‘s
- Column
- Type
- Operator
- Value
-)’s

Logical

You can enter a logical (Boolean) operator (`AND, NOT, ANDNOT, ORNOT`, or `OR`) when specifying a compound expression.

(‘s

You can enter the left parenthesis as single, double, or triple [**(**, **((**, or **(((**].

Column

You can enter the name of a column, up to a maximum of 31 characters.

Operator

You can enter the relational operator, such as **`=, !=, >, <, >=, <=`** (equal, not equal, greater than, less than, greater than or equal to, less than or equal to).

Value

You can enter a value against which the query is matched. The value can be a system buffer, an application buffer, a number, or a hard-coded value surrounded by double quotes.

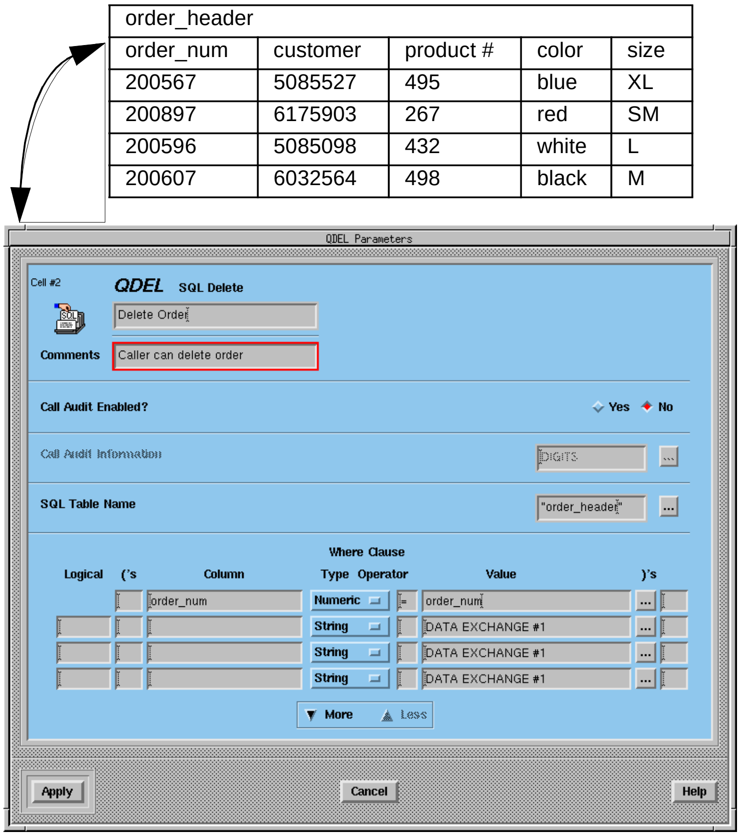
)’s

You can enter the closing parenthesis as single, double, or triple [**)**, **))**, or **)))**].

QDEL cell example transaction

You can use the QDEL cell within an application to remove the specified row from the table “order_header” where “order_num” is equal to “order_num.” Figure 3-9 displays the QDEL cell used in this example as well as the table “order_header” that it accesses.

Figure 3-9: QDEL transaction example



order_header				
order_num	customer	product #	color	size
200567	5085527	495	blue	XL
200897	6175903	267	red	SM
200596	5085098	432	white	L
200607	6032564	498	black	M

QDEL Parameters

Cell #2 **QDEL** SQL Delete

Delete Order

Comments: Caller can delete order

Call Audit Enabled? Yes [✓ No]

Call Audit Information: DIGITS

SQL Table Name: "order_header"

Where Clause

Logical	('s	Column	Type	Operator	Value	)'s
		order_num	Numeric	=	order_num	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	

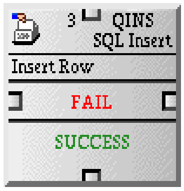
In this example, the QDEL cell performs the following SQL statement:

```
DELETE FROM order_header
    WHERE order_num = :order_num
```

Inserting a row with the QINS cell

The QINS (SQL INSERT) cell allows you to insert a row of data into a database table. The QINS parameter identifies the table to be accessed. Figure 3-10 displays the QINS cell.

Figure 3-10: QINS cell



QINS cell branches

The QINS cell contains the following cell branches:

- Fail
- Success

Fail

The QINS cell cannot perform the SQL transaction. Inspect the buffer SQL Error Code for details about the error condition. You can also read the IVR Generator transaction log and the DBMS error log for additional information.

Success

The cell successfully performs the SQL transaction. A row is added to the specified table with the values listed in the parameter `Column and Value Table`. The system buffers ROWS contains the number of rows that matched the specified criteria.

QINS cell parameters

The QINS cell contains the following cell parameters:

- `Call Audit Enabled?`
- `Call Audit Information`
- `SQL Table Name`
- `Column and Value Table`

Call Audit Enabled?

The default setting of the parameter `Call Audit Enabled?` is No. If you set this parameter to Yes, the QINS cell logs the following information to the call audit statistics file, `audit_stat.d:`

- application name
- cell name
- cell number
- date and time of cell execution
- contents of the cell comment field
- contents of the Call Audit Information buffer

Call Audit Information

If you set the parameter `Call Audit Enabled` to Yes, the contents of the buffer specified in the parameter `Call Audit Information` are logged to the file `audit_stat.d`.

SQL Table Name

This is the name of the table containing the rows that you wish to insert.

Column and Value Table

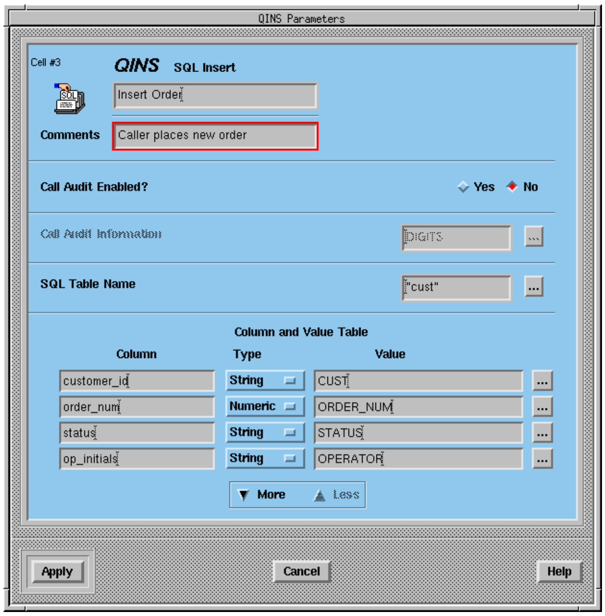
You can use the parameter `Column and Value Table` to specify each column in a new row, the column type, and the corresponding value to be inserted. You can set the field *Type* to one of the following options: `numeric, string, money, date`, or `float`. You can set the field *Value* to specify a specific value or a buffer containing the value.

QINS cell example transaction

You can use the QINS cell within an application to allow a caller to insert data into the table “cust” when a caller creates a new order loading different values in the columns “order_num,” “customer_id,” “status,” and “op_initials.” Figure 3-11 on page 3-24 displays the QINS cell used in this example as well as the table “order_header” that it accesses.

Figure 3-11: QINS transaction example

cust			
order_num	customer_id	status	op_initials
200567	5085527	d	jr
200897	6175903	p	jb
200596	5085098	d	pg
200607	6032564	d	jr



QINS Parameters

Cell #3 **QINS** SQL Insert

Insert Order

Comments: Caller places new order

Call Audit Enabled? Yes [✓ No]

Call Audit Information: DIGITS

SQL Table Name: "cust"

Column and Value Table

Column	Type	Value
customer_id	String	CUST
order_num	Numeric	ORDER_NUM
status	String	STATUS
op_initials	String	OPERATOR

In this example, the QINS cell performs the following SQL statement:

```
INSERT INTO cust
(order_num,customer_id,status,op_initials)
```

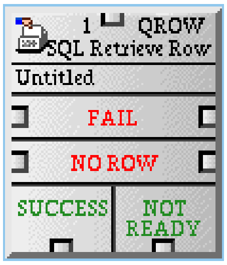
```
INSERT into = (:cust,:order_num,:status,:operator)
```

Retrieving a row with the QROW cell

You can use the QROW (SQL RETRIEVE A ROW) cell to sort through the multiple rows returned when you use the QSEL cell or the QCMD cell to select multiple rows. The QROW cell retrieves up to one complete row from these returned rows. The data that is retrieved by the QROW cell is valid until another QSEL cell is executed or a SELECT statement is issued in the QCMD cell.

Figure 3-12 displays the QROW cell.

Figure 3-12: QROW cell



This section provides information about the following topics:

- QROW cell branches
- QROW cell parameters

QROW cell branches

This section provides information about the following QROW cell branches:

- Fail
- No Rows
- Success
- Not Ready

Fail

The QROW cell cannot perform the SQL transaction. Inspect the buffer SQL ERROR CODE for details about the error condition. You can also read the IVR transaction log and the DBMS error log for additional information.

No Rows

No rows are affected or retrieved.

Success

The cell successfully performs the SQL transaction.

Not Ready

The row exists but is not retrieved yet.

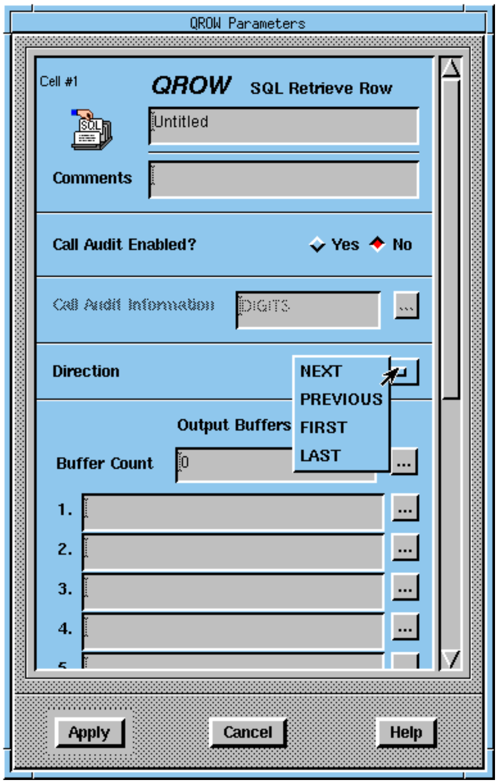
QROW cell parameters

This section provides information about the following QROW cell parameters:

- `Call Audit Enabled?`
- `Call Audit Information`
- `Direction`
- `Output Buffer`

Figure 3-13 on page 3-27 displays the QROW parameters page.

Figure 3-13: QROW parameters



QROW Parameters

Cell #1

QROW SQL Retrieve Row

Untitled

Comments

Call Audit Enabled? Yes [✓ No]

Call Audit Information DIGITS

Direction NEXT

NEXT
PREVIOUS
FIRST
LAST

Output Buffers

Buffer Count 0

1.
2.
3.
4.

Call Audit Enabled?

The default setting of the parameter `Call Audit Enabled?` is No. If you set this parameter to Yes, the QROW cell logs the following information to the call audit statistics file, `audit_stat.d`:

- application name
- cell name
- cell number
- date and time of cell execution
- contents of the cell comment field

- contents of the Call Audit Information buffer

Call Audit Information

If you set the parameter `Call Audit Enabled` to Yes, the contents of the buffer specified in the parameter `Call Audit Information` are logged to the file `audit_stat.d`.

Direction

Set the parameter `Direction` to indicate the row that you want to retrieve. You can choose from the following options: `Next, Previous, First`, and `Last`.

- If you select the option `Next`, then the QROW cell retrieves the next matching row from the group returned by the QSEL cell.
- If you select the option `Previous`, then the QROW cell retrieves the previous matching row from the group returned by the QSEL cell.
- If you select the option `First`, then the QROW cell retrieves the first matching row from the group returned by the QSEL cell.
- If you select the option `Last`, then the QROW cell retrieves the last matching row from the group returned by the QSEL cell.

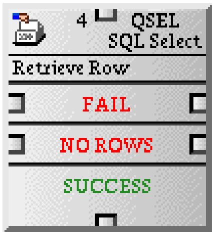
Output buffers

Set the number of output buffers equal to the number of units of information you are expecting the row to contain.

Retrieving information with the QSEL cell

You can use the QSEL (SQL SELECT) cell to retrieve a row or rows of data from a database. The QSEL cell returns all rows that match the criteria you specify on the QSEL parameter page, up to the maximum number specified in the parameter `Maximum Rows to Select`. The QSEL parameters identify the table or view to be accessed, the columns containing values to be returned, and the WHERE clause selection criteria. Figure 3-14 displays the QSEL cell.

Figure 3-14: QSEL cell



QSEL cell branches

The QSEL cell contains the following cell branches:

- Fail
- No Rows
- Success

Fail

The QSEL cell cannot perform the SQL transaction. Inspect the buffer SQL Error Code for details about the error condition. You can also read the IVR transaction log and the DBMS error log for additional information.

No Rows

No rows are affected or retrieved.

Success

The cell successfully performs the SQL transaction. The system buffer SQL ROWS is set to 1.

> ***Note:*** After the QSEL cell returns multiple rows, use the QROW cell to distill specific information from the rows returned.

QSEL cell parameters

The QSEL contains the following cell parameters:

- `Call Audit Enabled?`
- `Call Audit Information`
- `SQL Table Name`
- `Maximum Rows to Select`
- `Column and Buffer Table`
- `Where Clause`

Call Audit Enabled?

The default setting of the parameter `Call Audit Enabled?` is No. If you set this parameter to Yes, the QSEL cell logs the following information to the call audit statistics file, `audit_stat.d`:

- application name
- cell name
- cell number
- date and time of cell execution
- contents of the cell comment field
- contents of the Call Audit Information buffer

Call Audit Information

If you set the parameter `Call Audit Enabled` to Yes, the contents of the buffer specified in the parameter `Call Audit Information` are logged to the file `audit_stat.d`.

SQL Table Name

This is the name of the table containing the rows that you wish to select.

Maximum Rows to Select

The parameter `Maximum Rows to Select` allows you to specify the maximum number of rows you want the QSEL cell to return. For no limit to the number of rows, select `0`. The default value is `1`. The first row is returned by the QSEL cell.

Column and Buffer Table

SQL databases return rows of data in different orders. This creates problems for applications that depend on retrieving rows in a certain order. You can specify the order of the rows returned by the QSEL Cell in the `Column and Buffer Table` parameter field *Order By* by clicking on the button to select a sorting option. You can choose from the following sorting options: `Ascending,Descending`, and `None`. The default setting is `None`.

WHERE clause

You can enter one or more WHERE clauses. You can select a value by either typing the value in the text field or by selecting the field and pressing mouse button three. The system displays a menu that contains the valid values for that particular field. Move the cursor to the value you want and release mouse button three. The value appears in the field. You can specify values for the following fields:

- Logical
- (‘s
- Column
- Type
- Operator
- Value
-)’s

Logical

You can enter a logical (Boolean) operator (`AND`, `NOT`, `ANDNOT`, `ORNOT`, or `OR`) when specifying a compound expression.

(‘s

You can enter the left parenthesis as single, double, triple {**(**, **((**, or **(((**}.

Column

You can enter the name of a column, up to a maximum of 31 characters.

Operator

You can enter the relational operator, such as **`=, !=, >, <, >=, <=`** (equal, not equal, greater than, less than, greater than or equal to, less than or equal to).

Value

You can enter a value against which the query is matched. The value can be a system buffer, an application buffer, a number, or a hard-coded value surrounded by double quotes.

)’s

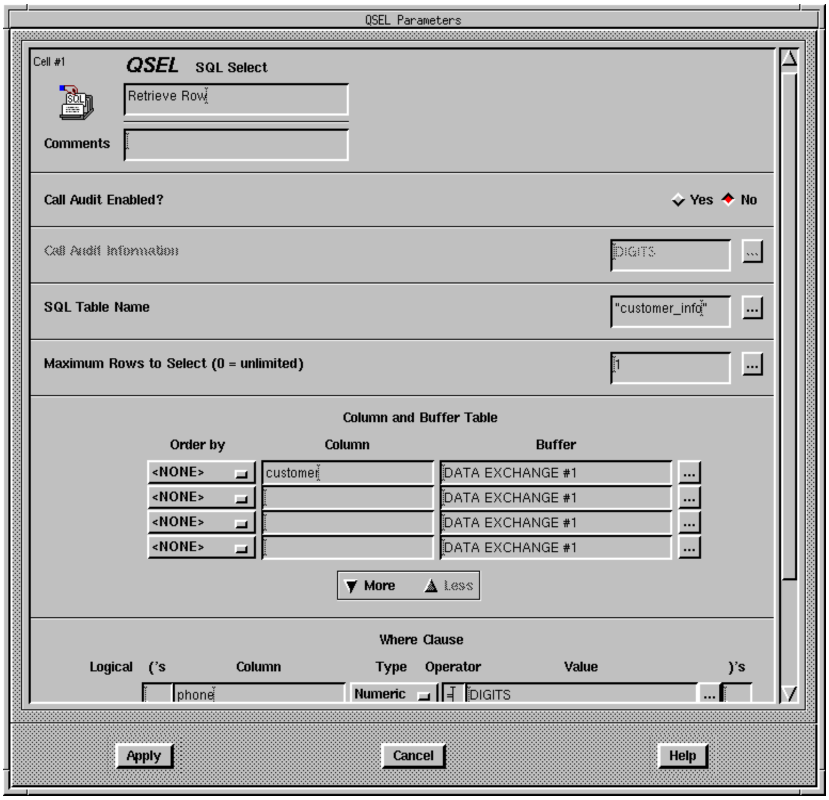
You can enter the closing parenthesis as single, double, or triple [**)**, **))**, or **)))**].

QSEL cell example transaction

You can use the QSEL cell to query the database to see if the caller is an existing customer. You can set the QSEL cell parameter to search the table “customer_info” for all entries where the value of the *phone* field is equal to the telephone number that the caller enters in a GDAT cell earlier in an application. If rows are not selected by the QSEL cell, then the caller is identified as a new customer. Figure 3-15 displays the QSEL cell used in this example as well as the table “customer_info” that it accesses.

Figure 3-15: QSEL transaction example

customer_info			
customer	phone	credit card #	cc type
Janet Calert	(508)4565527	543 7589 3445	VISA
Paul Abram	(617)6395903	267 8954 2334	MC
Mary Jones	(508)9845098	432 9087 5667	AMEX
Jerry Johns	(603)6342564	498 9765 4889	VISA


QSEL Parameters

Cell #1 **QSEL** SQL Select

Retrieve Row

Comments

Call Audit Enabled? Yes [✓ No]

Call Audit Information DIGITS

SQL Table Name "customer_info"

Maximum Rows to Select (0 = unlimited) 1

Column and Buffer Table

Order by	Column	Buffer
<NONE>	customer	DATA EXCHANGE #1
<NONE>		DATA EXCHANGE #1
<NONE>		DATA EXCHANGE #1
<NONE>		DATA EXCHANGE #1

Where Clause

Logical	('s	Column	Type	Operator	Value	)'s
		phone	Numeric	=	DIGITS	

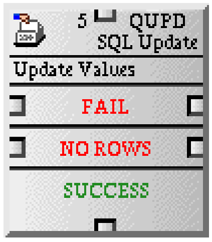
In this example, the QSEL cell performs the following SQL statement:

```
SELECT customer FROM customer_info where phone=:DIGITS
```

Updating a row with the QUPD cell

You can use the QUPD (SQL UPDATE) cell to modify values in an existing row in a database table. The QUPD cell parameters identify the table to be accessed; the WHERE clause specifies the rows to modify. Figure 3-16 displays the QUPD cell.

Figure 3-16: QUPD cell



QUPD cell branches

The QUPD cell contains the following cell branches:

- Fail
- No Rows
- Success

Fail

The QUPD cell cannot perform the SQL transaction. Inspect the buffer SQL ERROR CODE for details about the error condition. You can also read the IVR transaction log and the DBMS error log for additional information.

No Rows

No rows are affected. The system buffer SQL ROWS is set to `0`.

Success

The cell successfully performs the SQL transaction. The system buffer SQL ROWS contains the number of rows that are modified.

QUPD cell parameters

The QUPD cell contains the following cell parameters:

- `Call Audit Enabled?`
- `Call Audit Information`
- `SQL Table Name`
- `Maximum Rows to Select`
- `Column and Buffer Table`
- `Where Clause`

Call Audit Enabled?

The default setting of the parameter `Call Audit Enabled?` is No. If you set this parameter to Yes, the QUPD cell logs the following information to the call audit statistics file, `audit_stat.d`:

- application name
- cell name
- cell number
- date and time of cell execution
- contents of the cell comment field
- contents of the Call Audit Information buffer

Call Audit Information

If you set the parameter `Call Audit Enabled` to Yes, the contents of the buffer specified in the parameter `Call Audit Information` are logged to the file `audit_stat.d`.

SQL Table Name

This is the name of the table containing the rows that you wish to select.

Column and Value Table

You can use the parameter `Column and Value Table` to specify each column in a new row, the column type, and the corresponding value to be inserted. You can set the field *Type* to one of the following options: `numeric, string, money, date`, or `float`. You can set the field *Value* to specify a specific value or a buffer containing the value.

WHERE clause

You can enter one or more WHERE clauses. You can select a value by either typing the value in the text field or by selecting the field and pressing mouse button three. The system displays a menu that contains the valid values for that particular field. Move the cursor to the value you want and release mouse button three. The value appears in the field. You can specify values for the following fields:

- Logical
- (‘s
- Column
- Type
- Operator
- Value
-)’s

Logical

You can enter a logical (Boolean) operator (`AND,NOT,ANDNOT,ORNOT`, or `OR`) when specifying a compound expression.

(’s

You can enter the left parenthesis as single, double, or triple [**(**, **((**, or **(((**].

Column

You can enter the name of a column, up to a maximum of 31 characters.

Operator

You can enter the relational operator, such as **`=, !=, >, <, >=, <=`** (equal, not equal, greater than, less than, greater than or equal to, less than or equal to).

Value

You can enter a value against which the query is matched. The value can be a system buffer, an application buffer, a number, or a hard-coded value surrounded by double quotes.

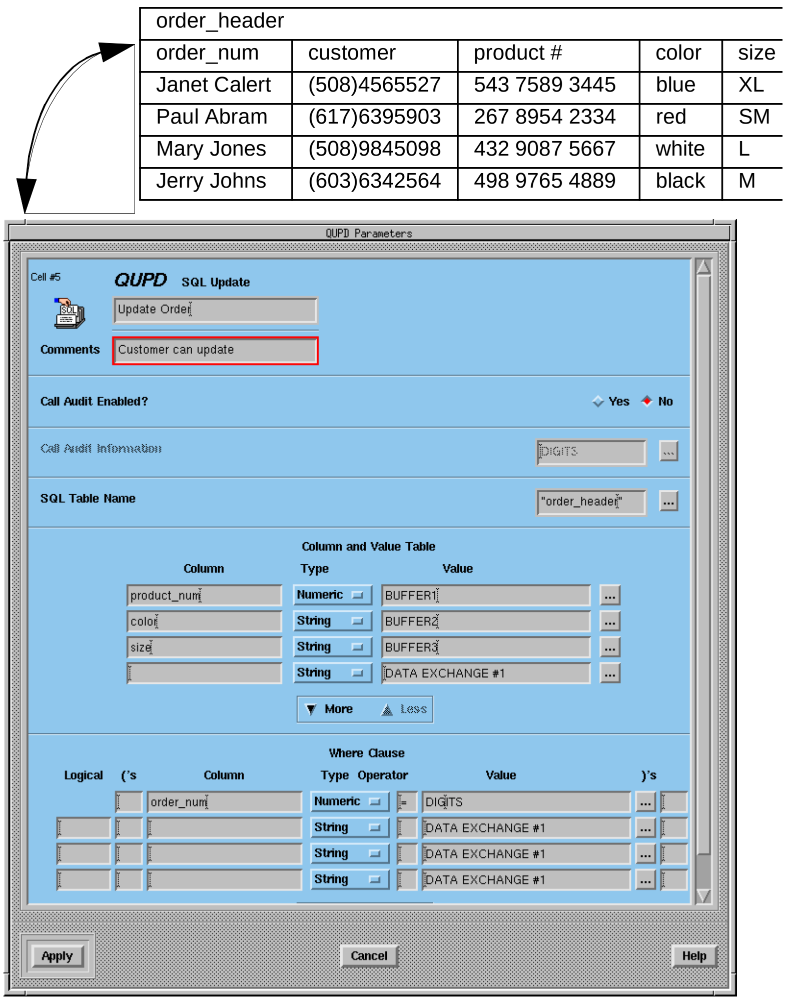
)'s

You can enter the closing parenthesis as single, double, or triple [**)**, **))**, or **)))**].

QUPD cell example transaction

You can use the QUPD cell to allow a caller to update an existing order. Figure 3-17 on page 3-38 displays the QUPD cell and the table "order-header."

Figure 3-17: Example QUPD transaction



order_header				
order_num	customer	product #	color	size
Janet Calert	(508)4565527	543 7589 3445	blue	XL
Paul Abram	(617)6395903	267 8954 2334	red	SM
Mary Jones	(508)9845098	432 9087 5667	white	L
Jerry Johns	(603)6342564	498 9765 4889	black	M

QUPD Parameters

Cell #5 **QUPD** SQL Update

Update Order

Comments: Customer can update

Call Audit Enabled? Yes [✓ No]

Call Audit Information: DIGITS

SQL Table Name: "order_header"

Column and Value Table

Column	Type	Value
product_num	Numeric	BUFFER1
color	String	BUFFER2
size	String	BUFFER3
	String	DATA EXCHANGE #1

More Less

Where Clause

Logical	('s	Column	Type	Operator	Value	)'s
		order_num	Numeric	=	DIGITS	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	
			String		DATA EXCHANGE #1	

Apply Cancel Help

In this example, the QUPD cell performs the following SQL statement:

```
UPDATE order_header
SET product_num=:BUFFER1,color=:BUFFER2,size=:BUFFER3
WHERE order_num=:digits
```

Glossary

This section lists brief definitions of the terms appearing in this guide.

μ-law

Aso called Mu-law. A form of Pulse Code Modulation (PCM), alternative to A-law. μ-law is used for the B channel of T1 streams. μ-law is the commonly used North American standard.

acoustic coupler

A device (such as the Audio Interface Unit) used to interface between sound equipment and a port on the MRS/AP.

Administration Tools

A set of tools, accessible from the Symposium OPEN IVR Control Panel, used to configure and control the Symposium OPEN IVR system.

ADSI

Analog Display Services Interface. The Bellcore standard defining the protocol used for the flow of information between a piece of telephone equipment (such as a switch, server, or voice mail) and a subscriber's telephone, PC, data terminal, or other device with a screen.

agent

A person whose primary role is to handle customer calls that are transferred from a queue within a call center.

A-law

A form of Pulse Code Modulation (PCM), in which the code word maps nonlinearly to the signal value. It is the standard compression algorithm used in digital communications systems of the European digital hierarchy. A-law code words are 8 bits wide. A-law is used for the B channel in ISDN.

Analog Display Services Interface

See ADSI.

ANI

Automatic Number Identification. A protocol implemented in some trunk cards that records the calling number.

ANSI

American National Standards Institute.

AP

Application Processor. A computer or workstation running the AP software in a distributed configuration. Symposium OPEN IVR combines one or more Multimedia Resource Servers (MRSs) with a general-purpose Application Processor (AP) to provide a versatile application development and run-time environment. The AP provides the environment for developing, managing, and controlling the IVR applications.

AP software

Nortel’s premiere voice application development and management software package that runs on the Application Processor (AP).

API

Application Programming Interface. Software libraries that enable the development of applications in a new environment, which communicate with applications in the legacy system. Software libraries that provide a standard access to services provided by a software package.

application

1 In Symposium OPEN IVR, a customized program that controls activity on one or more telephone trunks connected to an MRS voice system.

2 On a host computer, any type of program that carries out a task.

application developer

A person who creates Symposium OPEN IVR applications.

Application Editor

A utility used to graphically create or edit interactive voice response applications.

application processor

See AP.

Automatic Number Identification

See ANI.

branch

A pathway from one cell in an application to the next cell.

buffers

A part of computer memory used to store information from one cell and pass it to another cell.

caller

A person whose phone call is received or originated by a Symposium OPEN IVR application.

call flow

A diagram of a Symposium OPEN IVR application.

Call Logic Interpreter

See CLI.

cell

The basic element of an application. Each cell causes some action to be performed—for example, playing a prompt to a caller. Each cell has a set of branches to other cells. After the cell performs its action, it determines which branch the application should follow to the next cell.

channel

1 A telephone trunk within a cluster of MRSs.

2 A path of communication between the AP and a communications or transmission facility. Channel sequences can span multiple MRSs, and their numbering starts at zero (0). Also referred to as a **port**.

CLI

Call Logic Interpreter. Acts as a virtual machine which manages memory for IVR applications.

CO

Central Office.

Control Panel

The Symposium OPEN IVR panel containing the buttons from which you can access the Administration, Information Database Editor, Report, Application, Prompt Management, and SmartHelp tools.

CTI

Computer-telephone integration.

database

A collection of related information arranged for ease and speed of retrieval.

Database Management System

See DBMS.

DBMS

Database Management System. Software that allows you to logically organize data in a table format consisting of columns and rows.

Dialed Number Identification Service

See DNIS.

DID

Direct inward dialing. A protocol used by some telephone equipment that allows callers to dial directly to an extension within a telephone system without going through a switchboard.

direct inward dialing

See DID.

DNIS

Dialed Number Identification Service. A feature of 800 and 900 numbers, in which the number dialed or a portion of it is sent to the MRS and stored in the DIGITS buffer in OPEN IVR.

DTMF

Dual Tone Multi-Frequency. Touchtone (push-button) dialing. DTMF is used to encode digits over analog telephone lines. Applications can collect information from callers by having them press telephone keys to create DTMF tones.

Dual Tone Multi-Frequency

See DTMF.

EXEC

The cell that executes an OPEN IVR application on the current channel.

falsing

A condition in which certain frequencies in a voice recording are incorrectly interpreted as DTMF tones. In some applications, false DTMF tones may cause the system to perform unwanted activities, such as selecting menu choices, terminating playback, or hanging up prematurely.

field

The specific location of information (data) within a record in a relational database.

gain

The increase in signal power, measured in decibels, when the signal is boosted by an electronic device.

GDAT

Play Prompts and Get Data. A cell that allows you to play one or more prompts and collect up to 16 digits of input from the caller.

ground start

A method of signaling on the trunk in which one side of the two wires (typically the ring wire) is momentarily grounded to obtain a dial tone. This type of signal is normally used with PBXs.

hook-flash

Depressing and releasing the plunger or handset-cradle on a telephone.

host link

The ability of Symposium OPEN IVR to communicate with other host computers supported by one of the OPEN IVR gateway products.

information database

See database.

Information Database Editor

A utility used to create a message or information database, and to edit an information database.

IVR

Interactive Voice Response.

IVR Developer

A licensed component of AP software that provides application development tools to graphically create or edit interactive voice response applications.

loopstart

A method of signalling on the trunk in which a supervisory signal is given (by taking the phone off the hook). Loopstart seizes the line by bridging the resistance of the tip and ring wires of the telephone line to obtain a dial tone. Loopstart signalling is normally used for single lines or key systems.

MF tones

Multifrequency tones. These are tones used by ANI. Symposium OPEN IVR applications can receive and store MF digits when the ANI option is in use.

MRS

Multimedia Resource Server. Symposium OPEN IVR combines one or more Multimedia Resource Servers (MRSs) with a general-purpose Application Processor (AP) to provide a versatile application development and/or run-time environment.

MRS/AP

A single unit that runs a “folded” configuration; that is, a configuration that includes both the Multimedia Resource Server and the Application Processor. The MRS/AP contains the functionality of the other platforms, but in a single hardware unit.

MRS/Rack

A unit comprising a rack-mountable Application Processor (AP) and a rack-mountable Multimedia Resource Server (MRS). Multiple MRS/Racks can be installed in cabinets and stacked to take up less room.

MRS/Tower

The MRS/Tower platform is comprised of an Application Processor (AP) and a Tower Multimedia Resource Server (MRS).

Mu-law

See μ-law.

multifrequency tones

See MF tones.

Multimedia Resource Server

See MRS.

node

A grouping composed of an AP (running Symposium OPEN IVR) connected to one or more MRSs.

ODBC

Open Database Connectivity. A standards-based API that allows database clients to be configured to access a wide array of ODBC-compliant servers.

Open Database Connectivity

See ODBC.

parameter

A variable used to define the function of Symposium OPEN IVR cells. You set parameters in the parameter window for each cell (when using the Application Editor), or in the cell window (when using the character-based Application Editor).

PBX

Private Branch Exchange. A private phone system that allows communication within a business and between the business and the public telephone system.

PCM

Pulse Code Modulation. A method that uses a sequence of discrete integers as code words to describe a time-varying signal. Each integer code word describes the signal level at a point in time.

PDAT

Play Prompts with Data. A cell that allows you to play prompts to announce the contents of a buffer. The buffer content is read according to the specified type.

PRGS

Call Progress Detection.

port

A path of communication between the AP and a communications or transmission facility. Port sequences can span multiple MRSs, and their numbering starts at zero (0). Also referred to as a **channel**. *See also* trunk.

private branch exchange

See PBX.

prompt

A voice recording that helps lead a caller through a Symposium OPEN IVR application.

Prompt Management tools

A set of tools, accessed from the Symposium OPEN IVR Control Panel, that is used to record, load, and update prompts.

pulse code modulation

See PCM.

QCMD

An advanced SQL cell capable of performing functions that would otherwise require using C language and embedded SQL programming through a USER cell. You can also use the QCMD cell to perform multistatement SQL transactions within a particular application.

QCNT

A cell that allows you to execute an SQL SELECT COUNT statement to return the number of rows in a table that match specific criteria.

QDEL

SQL DELETE. A cell allows you to delete selected rows in a database table.

QDS

SQL Data Server Manager. A process that distributes the SQL database access workload among data server interface processes.

QINS

SQL INSERT. A cell that allows you to insert a row of data into a database table.

QROW

SQL RETRIEVE A ROW. A cell that allows you to sort through the multiple rows returned when you use the QSEL cell or the QCMD cell to select multiple rows. The QROW cell retrieves up to one complete row from these returned rows.

QSEL

SQL SELECT. A cell that allows you to retrieve rows of data from a database that match the criteria you specify on the QSEL parameter page.

QUPD

SQL UPDATE. A cell that allows you to modify values in an existing row in a database table.

record

1 To store sound on tape or disk for later playback.

2 In a database, a group of related data items (fields) treated as one item.

Reorder tone

A reorder tone is a fast busy or channel busy tone, which is applied 120 times per minute. The tone means that all switching paths are busy, all toll trunks are busy, there are equipment blockages, the caller dialed an unassigned code, or the digits dialed were scrambled.

Reports tools

A set of tools, accessed from the Symposium OPEN IVR Control Panel, that is used to generate reports about the system's activities.

Send Fax

See SFAX.

SFAX

Send Fax. Cell that sends one or more faxes to a destination fax machine.

SQL

Structured Query Language. A type of relational database.SQL statements and commands are used from an interface with SQL-based ANSI-compliant relational and non-relational database management systems (DBMS) to allow applications to retrieve, insert, update, and delete information stored in the supported databases.

Structured Query Language

See SQL.

subscriber

A person serviced by a voice processing application (such as voice mail).

system administrator

A person responsible for configuring MRSs, installing and running Symposium OPEN IVR applications, managing prompts, and running reports.

Telco

Telephone company.

Terminal Resource Server

See TRS.

TRS

trunk

A communication line between two switching systems, frequently referred to as a port. Trunk numbering starts at 1.

voice storage number

See VSN.

VSN

Voice storage number. Internal numbering system used by the MRS for prompts, messages, and faxes (if the fax option is installed); Symposium OPEN IVR uses its own numbering system.

wink

A momentary interruption in single frequency tone that indicates the distant central office (CO) is ready to receive the digits that were just dialed.

X-Window Application Editor

See XAE.

XAE

X-Window Application Editor. Another term for the Application Editor.

Index

Symposium OPEN IVR

SQL Server User Guide

Toronto Information Products
Nortel
522 University Avenue, 14th Floor
Toronto, Ontario Canada
M5G 1W7

Publication number:	P0881282
Product release:	4.0
Document release:	Standard 1.0
Date:	July 1998

Printed in the United States of America

NORTEL
NORTHERN TELECOM