Avaya Call Management System

Open Database Connectivity

Version 5.2

07-601580
March 2007

Notice

While reasonable efforts were made to ensure that the information in this document was complete and accurate at the time of printing, Avaya Inc. can assume no liability for any errors. Changes and corrections to the information in this document might be incorporated in future releases.

Documentation disclaimer

Avaya Inc. is not responsible for any modifications, additions, or deletions to the original published version of this documentation unless such modifications, additions, or deletions were performed by Avaya. Customer and/or End User agree to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation to the extent made by the Customer or End User.

Link disclaimer

Avaya Inc. is not responsible for the contents or reliability of any linked Web sites referenced elsewhere within this documentation, and Avaya does not necessarily endorse the products, services, or information described or offered within them. We cannot guarantee that these links will work all the time and we have no control over the availability of the linked pages.

Warranty

Avaya Inc. provides a limited warranty on this product. Refer to your sales agreement to establish the terms of the limited warranty. In addition, Avaya's standard warranty language, as well as information regarding support for this product, while under warranty, is available through the Avaya Support Web site:

http://www.avaya.com/support

License

USE OR INSTALLATION OF THE PRODUCT INDICATES THE END USER'S ACCEPTANCE OF THE TERMS SET FORTH HEREIN AND THE GENERAL LICENSE TERMS AVAILABLE ON THE AVAYA WEB SITE http://support.avaya.com/LicenseInfo/ ("GENERAL LICENSE TERMS"). IF YOU DO NOT WISH TO BE BOUND BY THESE TERMS, YOU MUST RETURN THE PRODUCT(S) TO THE POINT OF PURCHASE WITHIN TEN (10) DAYS OF DELIVERY FOR A REFUND OR CREDIT.

Avaya grants End User a license within the scope of the license types described below. The applicable number of licenses and units of capacity for which the license is granted will be one (1), unless a different number of licenses or units of capacity is specified in the Documentation or other materials available to End User. "Designated Processor" means a single stand-alone computing device. "Server" means a Designated Processor that hosts a software application to be accessed by multiple users. "Software" means the computer programs in object code, originally licensed by Avaya and ultimately utilized by End User, whether as stand-alone Products or pre-installed on Hardware. "Hardware" means the standard hardware Products, originally sold by Avaya and ultimately utilized by End User.

License type(s)

Designated System(s) License (DS). End User may install and use each copy of the Software on only one Designated Processor, unless a different number of Designated Processors is indicated in the Documentation or other materials available to End User. Avaya may require the Designated Processor(s) to be identified by type, serial number, feature key, location or other specific designation, or to be provided by End User to Avaya through electronic means established by Avaya specifically for this purpose.

Concurrent User License (CU). End User may install and use the Software on multiple Designated Processors or one or more Servers, so long as only the licensed number of Units are accessing and using the Software at any given time. A "Unit" means the unit on which Avaya, at its sole discretion, bases the pricing of its licenses and can be, without limitation, an agent, port or user, an e-mail or voice mail account in the name of a person or corporate function (e.g., webmaster or helpdesk), or a directory entry in the administrative database utilized by the Product that permits one user to interface with the Software. Units may be linked to a specific, identified Server.

Copyright

Except where expressly stated otherwise, the Product is protected by copyright and other laws respecting proprietary rights. Unauthorized reproduction, transfer, and or use can be a criminal, as well as a civil, offense under the applicable law.

Third-party components

Certain software programs or portions thereof included in the Product may contain software distributed under third party agreements ("Third Party Components"), which may contain terms that expand or limit rights to use certain portions of the Product ("Third Party Terms"). Information identifying Third Party Components and the Third Party Terms that apply to them is available on the Avaya Support Web site:

http://support.avaya.com/ThirdPartyLicense/

Preventing toll fraud

"Toll fraud" is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or is not working on your company's behalf). Be aware that there can be a risk of toll fraud associated with your system and that, if toll fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya fraud intervention

If you suspect that you are being victimized by toll fraud and you need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1-800-643-2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Support Web site:

http://www.avaya.com/support

Trademarks

Avaya and the Avaya logo are either registered trademarks or trademarks of Avaya Inc. in the United States of America and/or other jurisdictions.

MultiVantage is a trademark of Avaya Inc.

All other trademarks are the property of their respective owners.

Downloading documents

For the most current versions of documentation, see the Avaya Support Web site:

http://www.avaya.com/support

Avaya support

Avaya provides a telephone number for you to use to report problems or to ask questions about your product. The support telephone number is 1-800-242-2121 in the United States. For additional support telephone numbers, see the Avaya Support Web site:

http://www.avaya.com/support

Contents

Preface

Avaya Call Management System (CMS) is an application for businesses and organizations that use Avaya communication servers to process large volumes of telephone calls using the Automatic Call Distribution (ACD) feature. Avaya CMS supports solutions for routing and agent selection, multi-site contact centers, remote agents, reporting, interfaces to other systems, workforce management, desktop applications, system recovery, and quality monitoring.

Open Database Connectivity (ODBC) is an optional Avaya Call Management System (CMS) feature that allows you to access data in the CMS database for use in other software applications, such as spreadsheet programs. With ODBC, you can access the CMS data directly from your application without the need to understand database connectivity or format.

Avaya CMS is part of the Operational Effectiveness solution of the Avaya Customer Interaction Suite.

This section includes the following topics:

Purpose

The purpose of this document is to describe how to use ODBC. This document will help users decide which database items to use in custom reports, spreadsheets, and other user applications that require external data access.

Intended users

This document is written for:

- Avaya support personnel
- Contact center administrators

Users of this document must be familiar with Avaya CMS, have a basic understanding of SQL, and have a basic understanding of database logic and purpose.

Conventions and terminology

If you see any of the following safety labels in this document, take careful note of the information presented.

CAUTION:
Caution statements call attention to situations that can result in harm to software, loss of data, or an interruption in service.

WARNING:
Warning statements call attention to situations that can result in harm to hardware or equipment.

DANGER:
Danger statements call attention to situations that can result in harm to personnel.

SECURITY ALERT:
Security alert statements call attention to situations that can increase the potential for unauthorized use of a telecommunications system.

Reasons for reissue

This is the first issue of this document.

Availability

Copies of this document *are* available from one or both of the following sources:

Note:
Although there is no charge to download documents through the Avaya Web site, documents ordered from the Avaya Publications Center must be purchased.

- The Avaya online support Web site, http://www.avayadocs.com
- The Avaya Publications Center, which you can contact by:

  Voice:

  +1-207-866-6701
  +1-800-457-1764 (Toll-free, U.S. and Canada only)

  Fax:

  +1-207-626-7269
  +1-800-457-1764 (Toll-free, U.S. and Canada only)

  Mail:

  GlobalWare Solutions
  200 Ward Hill Avenue
  Haverhill, MA 01835 USA
  Attention: Avaya Account Manager

  E-mail:

  totalware@gwsmail.com

Related documentation

You might find the following Avaya CMS documentation useful. This section includes the following topics:

- Avaya CMS upgrade documents on page 11
- Documentation Web sites on page 12

Change description

For information about recent changes made in Avaya CMS, see:

- *Avaya Call Management System Release 14 Change Description*, 07-601579

Administration documents

For more information about Avaya CMS administration, see:

- *Avaya Call Management System Release 14 Administration,* 07-601585
- *Avaya Call Management System (CMS) Release 14 Database Items and Calculations,* 07-601591
- *Avaya Call Management System Supervisor Release 14 Reports*, 07-601590
- *Avaya Call Management System (CMS) Supervisor Release 14 Installation and Getting Started*, 07-601587
- *Avaya Call Management System High Availability User Guide*, 07-300066
- *Avaya Call Management System High Availability Connectivity, Upgrade and Administration*, 07-600957

Software documents

For more information about Avaya CMS software, see:

- *Avaya Call Management System Release 14 Software Installation, Maintenance, and Troubleshooting Guide,* 07-601578
- *Avaya CMS Open Database Connectivity Version 5.2,* 07-601580
- *Avaya Call Management System Release 14 LAN Backup User Guide,* 07-601589
- *Avaya Call Management System Release 14 External Call History Interface,* 07-601586
- *Avaya CMS Custom Reports,* 585-215-822
- *Avaya CMS Forecast User Guide,* 585-215-825
- *Avaya Call Management System (CMS) Supervisor Release 14 Report Designer*, 07-601588
- *Avaya Business Advocate Reports*, 07-601618

Hardware documents

For more information about Avaya CMS hardware, see:

- *Avaya Call Management System Sun Netra 210 Computer Hardware Installation, Maintenance, and Troubleshooting,* 07-600963
- *Avaya Call Management System Sun Fire V880/V890 Computer Hardware Installation, Maintenance, and Troubleshooting,* 07-600965
- *Avaya Call Management System Sun Blade 100/150 Workstation Hardware Installation, Maintenance, and Troubleshooting,* 07-600964
- *Avaya Call Management System Terminals, Printers, and Modems,* 585-215-874

Call Center documents

For more information about Avaya Call Center documents, see:

- *Avaya Call Management System Switch Connections, Administration*, and *Troubleshooting*, 07-601582

Avaya CMS upgrade documents

There are several upgrade paths supported with Avaya CMS. There is a document designed to support each upgrade.

This section includes the following topics:

Base load upgrades

Use a base load upgrade when upgrading CMS to the latest load of the same version (for example, r14ak.g to r14al.k). A specific set of instructions is included with the upgrade. The *Avaya Call Management System Release 14 Base Load Upgrade* document is shipped to the customer site with the CMS software CD-ROM as part of a Product Correction Notice (PCN).

Platform upgrades and data migration

Use a platform upgrade when upgrading to a new hardware platform (for example, upgrading from a Ultra 5 to a Sun Netra 210). The new hardware platform is shipped from the Avaya factory with the latest CMS load. Therefore, as part of the upgrade you will have the latest CMS load (for example, R3V11 to R14).

For more information about platform upgrades and data migration, see:

- *Avaya Call Management System Release 14 Platform Upgrade and Data Migration*, 07-601581

Avaya Call Management System Upgrade Express (CUE)

Use CUE when CMS is being upgraded from an earlier version (for example, R3V11) to the latest version (for example, R14).

A specific set of upgrade instructions is included with the upgrade. The *Avaya Call Management System Release 14 CMS Upgrade Express (CUE) for Sun Computers* document is included on the CUE software CD-ROM that is shipped to the customer site with the CUE kit.

For information about customer requirements for CUE upgrades, see:

- *Avaya Call Management System Release 14 CMS Upgrade Express (CUE) Customer Requirements, 700419930*

Documentation Web sites

For Avaya product documentation, go to http://www.avayadocs.com. Additional information about new software or hardware updates will be contained in future issues of this book. New issues of this book will be placed on the Web site when available.

Use the following Web sites to view related support documentation:

- Information about Avaya products and service

  http://www.avaya.com

- Sun hardware documentation

  http://docs.sun.com

Support

Contacting Avaya technical support

Avaya provides support telephone numbers for you to report problems or ask questions about your product.

For United States support:

1- 800- 242-2121

For international support:

See the 1-800 Support Directory listings on the Avaya Web site.

Escalating a technical support issue

Avaya Global Services Escalation Management provides the means to escalate urgent service issues. For more information, see the Escalation Management listings on the Avaya Web site.

About Open Database Connectivity

This section presents an overview of how Open Database Connectivity (ODBC) works and how it interacts with the Avaya Call Management System (CMS).

This section contains the following topics:

ODBC background and functionality

The ODBC feature is a client/server feature. To access the server the clients must be connected to a network that is fully functional and able to access the server. The *clients* are the computers that are accessing data through ODBC. The *server* is the CMS machine where the CMS database is located.

The ODBC feature is especially useful for call centers with multiple sites. ODBC allows access to data at multiple sites. You can use this data to produce reports. ODBC uses Structured Query Language (SQL) to access data.

ODBC is an Application Programming Interface (API) that allows you to access one or many Database Management Systems (DBMSs). You can use queries to access data in the database for use in reports and other outside applications.

This section contains the following topics:

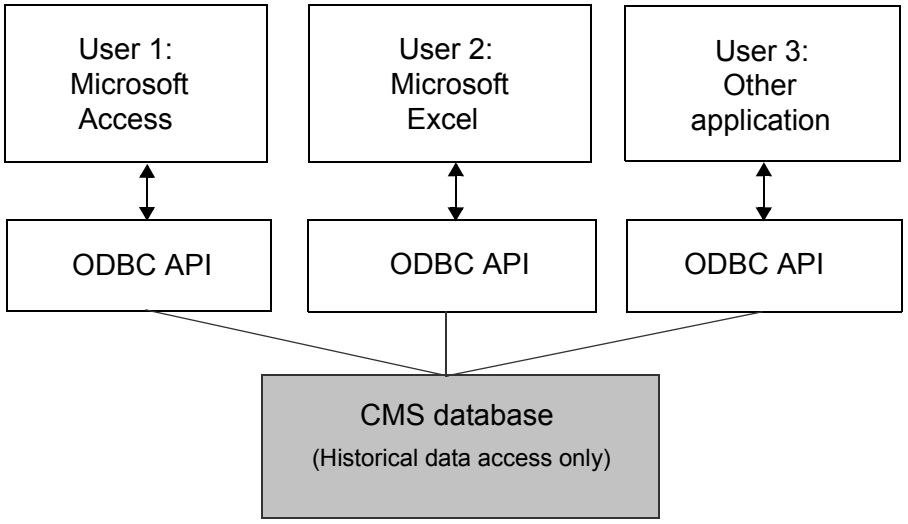
Data access through ODBC

ODBC was developed as a way to access different types of data. A single call center may be working with different applications that must access call center data. For example, a call center could be using Microsoft Access.

ODBC provides a standard method of database access without users having to be concerned with the underlying functionality of network software, naming conventions, and other complexities involved in accessing data through a DBMS. The data must be queried through the embedded SQL query function in the application that you are using. Refer to your specific application documentation for further information on the embedded SQL function for that particular application.

Data access diagram

The following figure illustrates user data access through ODBC.



Important:
Avaya will support only ODBC connectivity. Avaya will not support third-party client applications, such as Microsoft Access, or Windows.

Structured query language

ODBC uses Structured Query Language (SQL) to query and access data. Because SQL is a language, queries written in SQL can be used to access data with different formats. SQL is the basis for relational database access.

A relational database model is a table that stores data in rows and columns. Relationships between tables are established through data items that match data item values in another table.

SQL queries access the data stored in the relational database tables and extracts it for use in other applications. You compose an SQL query in the Windows application for which you need the data.

You can also use SQL to construct data calculations. You can use data calculations to see a sum of the data. For example, you can view the total number of calls routed to a particular split or skill.

CMS support of ODBC

The Informix database management system (DBMS) used by CMS is supported by the OpenLink Multi-Tier ODBC driver. The driver is an implementation of an ODBC application programming interface (API) that supports a particular database management system.

A driver is a dynamic link library (DLL) that is specific to a type of DBMS. The driver manager provides the link between the user's applications and the DBMS itself. When you run a query, the DBMS makes the link by selecting the DLL with the appropriate data format.

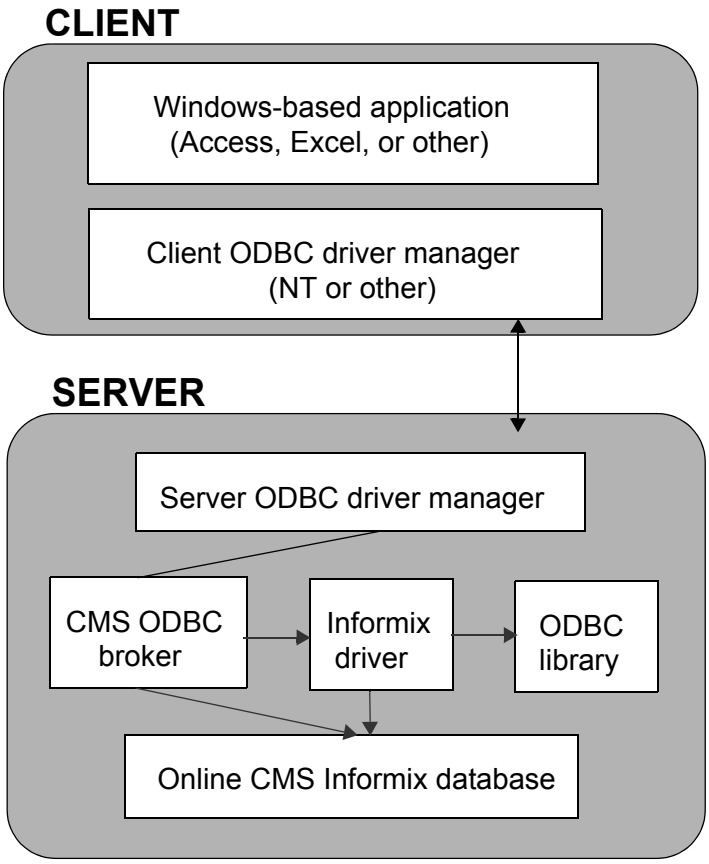
Important:
If you choose to develop an application for the ODBC driver, Avaya cannot provide support for that application or for any other third-party software or related mapping.

For more information about the OpenLink driver, see About the ODBC driver on page 19.

Note:
Avaya tests ODBC only in English. ODBC version 5.2 will support other double-byte languages, but if you use a language other than English, Avaya cannot provide ODBC support.

ODBC driver functionality

The following figure illustrates how the driver provides the interface between client applications and the data being accessed in the server.



Uses for ODBC data

Data extracted and stored by an ODBC application can be used by ODBC-enabled programs, such as workforce management packages, network routers, and blended inbound outbound applications. You can use an ODBC data application to generate reports using data from multiple call center sites and their databases.

Requesting data using ODBC

All queries in ODBC must be composed so that they ensure continued CMS performance. The query is invoked differently in each Windows application.

For more information on how to compose efficient database queries, see Performance impact on page 20 and the chapter on editing queries in the appropriate Avaya CMS Supervisor report designer user guide.

For more information on how your application interfaces with ODBC, refer to the user documentation for your application.

About the ODBC driver

The OpenLink ODBC driver has two main components:

- A generic ODBC driver
- OpenLink Request Agent

The OpenLink ODBC driver is a multi tier driver. The driver controls both ODBC calls and the manner in which these calls are transferred over the communications infrastructure to the relevant database management system.

The OpenLink Request Broker provides the link between the user's applications and the database management system. After ODBC is installed and administered, the ODBC driver and its components are transparent to the client. The CMS ODBC feature allows multiple, synchronous accesses from clients, users, and applications.

The OpenLink Request Agent resides on the client and interfaces with the ODBC driver through proprietary protocol. ODBC drivers are controlled by an ODBC driver manager, which is the OpenLink Request Broker. The OpenLink Request Broker resides on the server. ODBC uses data source names as the link between the ODBC Request Broker and the relevant ODBC driver for a particular database.

For additional information on the ODBC driver, see the online Help file that is included on the CMS OpenLink ODBC driver CD-ROM.

ODBC features

An OpenLink ODBC driver provides the ODBC communication and connectivity that allows external data access to the CMS database. There are individual drivers that support Solaris, Windows 98, Windows 2000, Windows XP, and Windows NT 4.0 clients. All historical CMS database tables, dictionary tables, and customer-provided tables can be accessed by ODBC clients through standard ODBC-enabled software applications.

All standard Structured Query Language (SQL) queries generated by the user applications are supported by the ODBC driver, as limited by permissions. Table-level, read-only permissions restrict access to certain database tables.

This section contains the following topics:

Languages

Avaya tests ODBC only in English for use with CMS. ODBC version 5.2 will support other double-byte languages, but if you use a language other than English, Avaya does *not* provide ODBC support for that language.

Supported logins

Depending on your licensing agreement, the CMS ODBC feature supports either five or ten simultaneous logins. Additional logins can be added in increments of five.

Queries

You can generate queries from client to server and from user applications. The ODBC driver is installed on the server, and the accompanying software is installed on the clients. The extracted data can be used in workforce management packages, network routers, and blended inbound or outbound applications. Any application that supports ODBC functionality can access tables with the proper table permissions. For example, one of the tables that could be accessed is the CMS Dictionary.

For more information on how to compose efficient database queries, see the chapter on editing queries in the appropriate Avaya Supervisor report designer user guide.

Performance impact

The number, size, and types of queries received by the CMS may impact performance. The recommendations for composing queries in the CMS custom report editor also apply to ODBC queries. Some of the activities that may impact on CMS performance are:

Tables : Use the exact table and database item names when querying the database. You can maximize the performance of the system by running queries that access large tables or that perform table joins during a period of low agent activity and low real-time report activity. Accessing large tables, such as the split/skill or agent tables, or joining tables in queries may have a negative impact on CMS performance.

Calculations : Review calculations before sending them to the database. This ensures that the correct operation is performed. Arithmetic operations are performed with the rules of operator precedence, in order from left to right. The operator precedence is shown in the following table:

Precedence	Arithmetic operations
1	( ) [ ]
2	* / %
3	+ -

Queries: Prioritize resource intensive queries the same way you prioritize reports during high business activity. Running complex or multiple queries on the database impact system performance similar to running multiple reports.

Number of simultaneous database accesses: Minimize the number of database connects and disconnects from an application, and spread your ODBC activities throughout the day.

Synonyms: Download the synonyms to your client application or database and then perform the join at the client.

CMS maintenance: Be aware that during off-peak hours, CMS runs it's own activities, such as archiving and making backups. This can use a significant amount of resources and time when working with a large database.

Table permissions, security and port allocation

ODBC users log into the CMS server with password protection. Users have SQL access to Informix tables, as limited by the table permissions. The database tables that are accessible in a particular version of CMS are described in detail in Database tables on page 51.

All historical and dictionary database tables have read-only access permission. The customer-created tables (any table name that begins with "c_") on the host have read and write permissions. No other tables are accessible through ODBC.

Important:
If your network uses a fire wall, it is common for unused ports to be locked. ODBC uses UDP port 60001 and the TCP ports between PortLow and PortHigh. If these ports are locked, you will not be able to connect to the CMS database with ODBC.

Depending on the ODBC version, the ports will be defined under [`Protocol TCP`] in one of the following files:

CMS version	Openlink version	File location
R14 and later	5.2	**/cms/dc/odbc/cmsrqb5.2_init**

CMS feature interactions that require client support

Some CMS features require specific client administration and interaction. The data received by the client application is in raw form. You must format raw data for use within your application. Dictionary names and certain time fields are most affected by this formatting.

Dictionary names: Clients can access CMS dictionary names. The client must map the synonym to the report. Underlying data is numeric. For example, different splits are stored as numbers and not by their names.

Permissions: Applications that access Informix externally, such as database access scripts, may not work if the table permission script tries to access a table to which permission is denied. CMS-like permissions to individual entities, such as splits and vectors, are not available through the ODBC interface. It is up to the client application to create and enforce permissions at this level if this is required.

Field Display: The time and date data received from the database may not be formatted. The time is usually shown in seconds or in military format. Review data for formatting when importing it into your software application. See your software's documentation for further information on formatting data.

For more information on data format and values, see Database tables on page 51.

Installing ODBC 5.2 on the CMS server

This section describes system requirements, software installation procedures, and configuration procedures for the OpenLink Multi-Tier ODBC driver on the CMS server.

This section contains the following topics:

System requirements

The following system requirements must be met before you install the software:

- The ODBC version 5.2 server must be installed on a supported CMS hardware platform.
- You must use the corresponding ODBC driver versions for the client and server to ensure connectivity.
- The server must have CMS R14 or later installed.
- The network communication software must be correctly installed and configured, and the network must be fully functional so that the server can communicate with the clients.

Determining the CMS version

Before installing the OpenLink ODBC driver on the server, determine which version of CMS is installed.

To determine the CMS version:

1. On the CMS system, enter:

   ```
   pkginfo  -x cms
   ```

   The system displays the CMS version information.
2. Record the CMS version so you can select the correct installation options.

Openlink ODBC compatibility table

Use the following table to determine CMS and ODBC compatibility.

	CMS R13 or later
Openlink ODBC version	5.2
Solaris OS version	5.9
Solaris ODBC client version	2.9
Informix provider type or Informix database domain name (Used by Windows client)	2000
CMS database path	cms
Informix server type (Used by Solaris client)	2000
Openlink data source driver	ODBC Generic Driver 32 Bit v5.0

Installing ODBC on the CMS server

To install ODBC on the CMS server:

1. Verify that you are installing the correct software for your CMS load.

   For more information, see Determining the CMS version on page 24 and the Openlink ODBC compatibility table on page 24.

2. Load the CD-ROM, *Avaya CMS OPENLINK ODBC Driver* into the CD-ROM drive.
3. Choose one of the following procedures:
   - If this is *not* an upgrade of the ODBC software, go to Step 4.
   - If this is an upgrade of the ODBC software, perform the following steps:
     a. Shut down the request broker by entering:

        `/cms/dc/odbc/odbc_init -r 0`

     b. Remove the old **openlink** directory by entering:

        `rm -fr /usr/openlink`

4. Create the OpenLink directory and change to the new directory by entering the following commands:

   `mkdir /usr/openlink`

   `cd /usr/openlink`

5. Enter:

   `cp /cdrom/cdrom0/server/cmsr14/* /usr/openlink`

6. Install the server components on the system by entering:

   `./install.sh`

   The system displays the following message:

```
Extracting (srami9zz.taz) ...
.........
.........
.........
Enter the port number the the broker will listen on for
client connections [Enter=Default] :
```

7. Accept the default setting by pressing **Enter**.

   The system displays the following message:

```
Welcome to the OpenLink Admin Assistant Setup.

This program will install the HTTP based OpenLink Admin Assistant,
thereby enabling remote configuration for all OpenLink Server
Components (Rule Book,
Service and Database Agents) from any Web Browser.

TCP/IP Port to use? [ENTER=8000] :
```

8. Accept the default setting by pressing **Enter**.

   The system displays the following message:

```
Log File? [ENTER=www_sv.log]
```

9. Accept the default setting by pressing **Enter**.

   The system displays the following message:

```
Log all requests (y/n)? [ENTER=n]
```

10. Accept the default setting by pressing **Enter**.

    The system displays the following message:

```
Administrator account? [ENTER=admin]
```

11. Accept the default setting by pressing **Enter**.

    The system displays the following message:

```
Administrator's password? [ENTER=admin]
```

12. Accept the default setting by pressing **Enter**.

    The system displays the following message:

```
The OpenLink Admin Assistant is now ready for use.
..........
..........
..........
End of installation
```

13. Enter the following commands:

    cd /usr/openlink/bin

    ./oplrqb -v

    The system displays licence information.

14. Choose one of the following commands to configure and initiate the ODBC software:

    - If this is a new installation, enter:

      /cms/dc/odbc/odbc_init

      The system displays the following message:

```
ODBC driver initialization complete.
```

    - If this is an upgrade or reinstallation, enter:

      /cms/dc/odbc/odbc_init -r 1

      The system displays the following message:

```
oplrqb has been activated
```

15. Verify that the ODBC Request Broker is active on the server by entering:

    ps -ef | grep oplrqb

    The displayed message should show the oplrqb process running from the **/usr/openlink/bin** directory similar to the following:

```
root  1462  1459  0 14:41:38 ?
0:00 /usr/openlink/bin/oplrqb -f +configfile /cms/dc/odbc/cmsrqb5.2_init +loglevel 5
+l
```

    If you do not see an *oplrqb* process running after completing Step 15, repeat the installation. If after reinstalling the software the process still does not start, call Avaya support.

Setting debug levels

To set the server ODBC debug level:

1. Enter:

   cd /cms/dc/odbc

2. Enter:

   ./odbc_init -d *x*

   where ***x*** is one of the following parameters:

   - To turn off the debug utility, enter: **0**
   - To turn on the debug utility, enter: **1**

Setting log levels

To set the CMS server log level for ODBC:

1. Enter:

   cd /cms/dc/odbc

2. Enter:

   ./odbc_init -l x

   where **l** is the lowercase letter l, and

   where ***x*** is one of the following log levels:

   - **0** turns off the CMS server logging
   - **1** through **7** sets the log level, with 1 as the lowest level and 7 as the highest. The default and recommended setting is 5.

Note:
Avaya recommends that users not set the log level to a value greater than 5. Doing so could reduce performance. If you use the log level 7 (troubleshooting), make sure that the value is changed back to the default log level (5) after troubleshooting.

Turning ODBC on or off

To turn the ODBC feature on or off on the server:

1. Enter:

   cd /cms/dc/odbc

2. Enter:

   ./odbc_init -r *x*

   where ***x*** is one of the following parameters:

   - To turn off the ODBC feature**, enter:** **0**
   - To turn on the ODBC feature, enter: **1**

Configuring ODBC for secure connections

To configure ODBC for secure connections:

Important:
You must have **root** permissions on the CMS server to perform this procedure.

1. Enter:

   cd /usr/openlink

2. Enter the following command to set the ODBC path:

   . ./openlink.sh

3. Enter the following command to verify the path:

   echo $PATH

   The path is `/usr/openlink/bin:/usr/openlink/samples/ODBC:/usr/sbin:/usr/bin:/usr/dt/bin:/usr/openwin/bin`

4. Enter:

   cd /usr/openlink/bin

5. Enter the following command to create your security key file:

   ./mkcert.sh *mykey*

   where ***mykey*** is the name of your security key file.

6. Enter the following information when prompted:

- Host name
- Organization
- Organizational Unit
- State or Province Name
- Locality Name
- Country Name

It might take a long time to create the security key file.

7. Enter:

   ls -alrt

   Locate the security key file you just created.

8. Enter the following command to change the permissions on the security key file:

   chmod 644 *mykey*.*

   where ***mykey*** is the name of your security key file.

9. Enter:

   ls -arlt

10. Verify that the permissions on the security key file are `-rw-r--r--`.
11. Enter:

    cd /cms/dc/odbc

12. Enter the following command to save the original ODBC configuration file.

    cp -p cmsrqb5.2_init cmsrqb5.2_init.ORIG

13. Enter:

    vi /cms/dc/odbc

14. Locate the `Request Broker` section in the file, and add the following lines:

    - `SSLKeyFile = mykey.key`
    - `SSLRequired = No`

    For example:

```
[Request Broker]
BinaryDirectory  = /usr/openlink/bin
CommandLine      = ;+logfile /tmp/oplrqb.log +debug
Protocols        = tcp
SSLKeyFile       = mykey.key
SSLRequired      = No
```

Note:
Entering `SSLRequired = No` will allow both secure and unsecure connections to the ODBC server. You can enter `SSLRequired = Yes` if all connections to the ODBC server will be secure.

15. Save and close the file.
16. Enter the following command to stop the request broker:

    `/cms/dc/odbc/odbc_init -r 0`
17. Enter: **`y`**
18. Enter the following command to start the request broker:

    `/cms/dc/odbc/odbc_init -r 1`
19. Verify that the ODBC request broker is running, by entering:

    `ps -ef | grep opl`

    The system should display a message similar to the following.

```
root 16425 16423 10 10:24:53 ?        0:07 /usr/openlink/bin/oplrqb -fd +configfile /
cms/dc/odbc/cmsrqb5.2_init +loglevel
    root 16451 16223  0 10:25:21 pts/7    0:00 grep opl
```

20. Configure an ODBC data source. For more information about configuring a data source, see Installing ODBC 5.2 on a Windows client on page 33.

Installing ODBC 5.2 on a Windows client

The OpenLink ODBC Request Agent software can be installed on your desktop computer or on your network for each client to access.

This section contains the following topics:

Requirements

Before installing the OpenLink ODBC driver software on your computer, verify that:

- The client network software is installed.

  Verify that you have a Winsock-compliant desktop TCP/IP product installed. Check for the existence of the **winsock.dll** file with Windows Explorer or through your file manager.

- Your computer is communicating with the CMS server over the network.

  Use your desktop TCP/IP product's Packet Internet Groper (PING) utility to ensure that communication between your computer and the CMS server is functional. Do not proceed if basic communications between your computer and the server cannot be established.

- The desktop computer is running Windows 98, Windows 2000, Windows NT 4.0, or Windows XP.

- The network communication software must be correctly installed and configured, and the network must be fully functional, so that the server and client can communicate.

Installing ODBC on a Windows client

To install OpenLink ODBC software on a Windows client:

1. Start Microsoft Windows.

   Note:
   The Windows XP interface is completely customizable. You may notice some differences based on your system configuration.

2. Insert the CD-ROM, *Avaya CMS OPENLINK ODBC Driver* into the CD-ROM drive.
3. Open Windows Explorer, and then select the CD-ROM drive with the ODBC CD-ROM disc.
4. Select the **Client** folder.
5. Select the **Win32** folder.
6. Double click the application, **SETUP.EXE**

   Note:
   Avaya recommends that you accept the default installation configuration.

   The system displays the **Welcome** window.

7. Select **Next**.

   The system displays the **Software License Agreement** window.

8. Accept the license agreement and select **Yes**.

   The system displays the **Choose Destination Location** window.

9. Select **Next**.

   The system displays the **Select Components** window.

10. *Deselect* **JDBC Client**.
11. Select **Next**.

    The system displays the **Select Program Folder** window.

12. Select **Next**.

    The system displays the **Start Copying Files** window.

13. Select **Next**.

    The system copies the files, and then displays the **Setup Complete** window.

14. Select **Finish**.
15. Go to Configuring an ODBC data source on page 35.

Installing clients over the network

To install ODBC on a desktop computer client that does not have a CD-ROM drive, or to install ODBC over the network:

1. Install the ODBC driver on a networked machine that has a CD-ROM drive. For more information, see Installing ODBC on a Windows client on page 34.
2. Go to the system where you want to install the client.
3. Open the computer's Program Manager or Windows Explorer and go to the directory for the networked machine.
4. Copy the **OpenLink** folder from the networked machine.
5. Change directories to your client computer's hard drive.
6. Copy the **OpenLink** folder to your hard drive.
7. Run the application, **SETUP.EXE**.
8. Set up the software using the **Normal** option.
9. Go to Installing ODBC on a Windows client on page 34.

Configuring an ODBC data source

After associating the CMS data source with the OpenLink Generic 32 bit v5.0 driver, you must configure database access to a specific server.

Note:
The Windows XP interface is completely customizable. You may notice some differences based on your system configuration.

To configure your ODBC driver software to access CMS data:

1. Select **Start > Programs > Openlink Data Access Drivers > C++ Demo 32 Bit**

   The system displays the **ODBC SDK 2.0 C++ Demo** window.
2. Select **Environment > Add Data Source**

   The system displays the **Create New Data Source** window.
3. Select one of the following options:
   - **System Data Source (Applies to this machine only)**

     Choose the System Data Source option if you want the data source to be available to all users.

- **User Data Source (Applies to this machine only)**

  Choose the User Data Source option if you want the data source to be available to the current user. This setting is useful if you must provide access for a specific user. Avaya does not recommend that you administer data sources on a per-user login ID basis.

4. Select **Next**.
5. The **Create New Data Source** window displays a list of data source drivers.
6. Select **Openlink Generic ODBC Driver**.
7. Select **Next**, and then select **Finish**.

   The system displays the Openlink Multi Tier DSN Configuration wizard.
8. Perform the following tasks:
   a. Enter a representative name in the **Name** field for the server or database to which you are connecting to. An example entry for this field is ODBC5.2.
   b. Enter description of the data source to which you are connecting in the **Description** field. An example entry for this field is ODBC.
   c. Enter the host name or IP address, and the default port of your database host machine in the **Server** field.

      The format should be ***data_source:port***. The default port for CMS is 5000. An example entry for this field is 135.9.82.31:5000.
9. Select **Next**.
10. Perform the following tasks:
    a. Enter **`Informix 2000`** in the **Domain** field.
    b. Enter an ODBC data source name in the **Database** field.

       In a multi-site call center situation, you can use this field to differentiate between call center locations. An example entry for this field is CMS.
    c. Enter the network alias or IP address of the machine to which you want to make ODBC connections in the **Informix Server** field. An example entry for this field is 135.9.82.31.
    d. Choose one of the following options:
       - If you want to connect with a secure connection, select the **Secure connection** box. An example of a type of secure connection is SSH.
       - If you *do not* want to connect with a secure connection, do not select the **Secure connection** box.
    e. Select the **Connect now to verify that all settings are correct** option.

       This option creates a test connection to verify the ODBC data source connection.

f. Enter a valid CMS user ID in the **Login ID** field.

   If you enter a CMS user login ID, the system displays a default user name for each login to the data source.

g. Enter the password for the database user ID in the **Password** field.

11. Select **Next**.
12. Enter `60` in the **Row buffer size** field.

    The Row buffer size field specifies the number of records to be transported over the network in a single network hop.

13. Select **Next**.
14. Select **Finish**.

    At this point, the ODBC driver software is installed on your computer.

15. Continue with Configuring ODBC tracing options on page 39 to specify additional configuration settings.

Note:

Once the driver is configured, the OpenLink ODBC driver is accessible to ODBC-enabled applications on your computer. Any queries that you send to the CMS database from client Windows applications, such as Microsoft Access, will use the ODBC feature to access data and copy it to your applications.

You need to format the data within your application. The data returned from your SQL queries is formatted in the manner described in Database tables on page 51.

Accessing the ODBC Data Source Administrator window

The ODBC Driver Administration utility resides within your desktop environment's control panel. This utility is an optional method for adding and removing ODBC drivers. To access the ODBC Driver Administration utility after the OpenLink ODBC driver software has been installed, choose one of the following procedures, depending upon your version of Windows:

- Accessing the ODBC Data Source Administrator window with Windows 98, or Windows NT 4.0 on page 38
- Accessing the ODBC Data Source Administrator window with Windows XP, or Windows 2000 on page 38

Accessing the ODBC Data Source Administrator window with Windows 98, or Windows NT 4.0

To access the ODBC Data Source Administrator window on a Windows 98, Windows 2000, or Windows NT 4.0 system:

1. In the Windows task bar, select **Start > Settings > Control Panel**.

   The system displays the **Control Panel** window.

2. Double click **ODBC Data Sources**.

   The system displays the **ODBC Data Source Administrator** window.

Accessing the ODBC Data Source Administrator window with Windows XP, or Windows 2000

To access the ODBC Data Source Administrator window on a Windows XP or Windows 2000 system:

1. In the Windows task bar, select **Start > Control Panel**.

   Note:
   The Windows XP interface is completely customizable. You may notice some differences based on your system configuration.

2. Select **Administrative Tools**.
3. Double click **Data Sources (ODBC)**.

   The system displays the **ODBC Data Source Administrator** window.

Removing a data source

To remove any data source, perform the following:

1. Select the System DSN tab in the **ODBC Data Source Administrator** window.

   For information on how to access the **ODBC Data Source Administrator** window, go to Accessing the ODBC Data Source Administrator window on page 37.

2. Select the appropriate ODBC data source.
3. Select the **Remove** button in the **ODBC Data Source** window and follow the prompts.

Configuring ODBC tracing options

You may specify how the ODBC driver traces ODBC function calls. If tracing is activated, the system generates a file that contains the actual ODBC function calls.

To set the ODBC tracing options:

1. In the **ODBC Data Source Administrator** window, select the **Tracing** tab.

   For information on how to access the **ODBC Data Source Administrator** window, go to Accessing the ODBC Data Source Administrator window on page 37.

2. Choose one of the following options:
   - Trace ODBC calls or observe ODBC activity by selecting the **Start Tracing Now** button.
   - Stop tracing ODBC function calls automatically by selecting the **Stop Tracing Now** button. This will terminate the ODBC tracing upon completion of the ODBC session.
   - Select or change the file to which the OpenLink Request Broker writes tracing information, by performing one of the following steps:
     - Enter a file name and path in the **Log file Path** field.
     - Use the **Browse...** button to select the appropriate file from the **Select ODBC Log File** window.

Note:
The default log file is **\SQL.LOG**.

CAUTION:
Do not change the default entry in the **Custom Trace DLL** field.

Viewing installed ODBC data source drivers

Use the **Drivers** window to verify installation of the OpenLink data source driver.

To view a list of installed ODBC drivers:

1. In the **ODBC Data Source Administrator** window, select the **Drivers** tab.

   For information on how to access the **ODBC Data Source Administrator** window, go to Accessing the ODBC Data Source Administrator window on page 37.

2. View detailed information about an installed driver by selecting the driver from the list, and then selecting the **About** tab.

Note:
If the appropriate OpenLink data source driver is not displayed in the **Drivers** window, return to Installing ODBC on a Windows client on page 34 and reinstall the driver.

Testing ODBC connectivity on a Windows client

After you have installed the OpenLink ODBC driver software on both the client and the server, you can open a demonstration connection to a data source on the server to show connectivity and test SQL access.

This section contains the following procedures:

Connecting to and accessing data

To connect to a data source from the client, complete the following steps:

Note:
The Windows XP interface is completely customizable. You may notice some differences based on your system configuration.

1. Go to the **Start** button and select **Programs > OpenLink Data Access Drivers > C++ Demo 32 bit**

   The system displays the **ODBC SDK 2.0 C++ Demo** window.

2. Select **Environment > Open Connection**.

   The system displays the **Select Data Source** window.

Note:
Depending on the ODBC driver version you are using, the **Select Data Source** window might display and you will be prompted to select a file or machine data source.

3. Select the **Machine Data Source** tab.
4. Select the data source that you want to use from the list, and then select **OK**.

Note:
You should have already configured the CMS data source. If you do not see a data source in the **SQL Data Sources** window, see Configuring an ODBC data source on page 35.

The system displays the **OpenLink ODBC Login** window.

Note:
The system will not display the **OpenLink ODBC Login** window if you selected the **No Login Dialog Box** option during the software configuration.

5. Enter your CMS server user name and password (CMS server login).
6. Select **Ok**.

   The system displays the **ODBC SDK 2.0 C++ Demo** window for the connection.
7. Select **SQL > Execute SQL**.

   The system displays the **ODBC SDK 2.0 C++ Demo** window.
8. Enter the following SQL query in the text box:

   select count(*) from hsplit

   Note:
   Any valid SQL query may be entered.
9. Select **OK**.

   The system displays the queried data in the **ODBC SDK 2.0 C++ Demo** window.

   You should see a count column on the **ODBC SDK 2.0 C++ Demo** window. The value in this column is the number of records in the CMS `hsplit` table. This result confirms that you have successfully accessed the database from the client. Use this test as a troubleshooting tool in the future to verify connectivity and data access from the client.

Disconnecting from a data source

After you have completed test querying the database, you can disconnect from the data source through the ODBC driver software.

To disconnect from the data source:

1. Select **Environment > Close Connection**.

   The system displays the **ODBC SDK 2.0 C++ Demo Close Current Connection** window.
2. Select **OK** to close the connection.
3. Select **File > Exit**

   The system disconnects from the data source.

Installing ODBC 5.2 on a Solaris client

The OpenLink ODBC driver software can be installed on a Solaris client. The client software must be configured and tested for connectivity after it is installed.

This section contains the following topics:

Requirements

Before installing the OpenLink ODBC driver software on a Solaris client, you must:

- Verify that the client is communicating with the CMS server over the network.

  Use the network protocol's ping utility to verify that communication between the client and the CMS server is functional. Do not proceed if basic communications between the client and the server cannot be established. If the client cannot recognize the network, the ODBC driver will not function properly.
- Verify that the ODBC clients are on supported computers with the appropriate software installed. See the Openlink ODBC compatibility table on page 24 for more information.
- Develop an application using the ODBC API.

Installing ODBC on the Solaris client

To install the OpenLink ODBC 5.2 software on the Solaris client:

1. Insert the CD-ROM, *Avaya CMS OPENLINK ODBC Driver* into the CD-ROM drive.
2. Log into the system as **root**.
3. Enter:

   ```
   cd /usr
   ```

4. Enter:

   ls

5. Verify that the server drivers are installed on the system. If the server drivers are installed, there will be an **openlink** directory.
6. Choose one of the following:
   - If the server drivers are installed, continue with Step 7.
   - If the server drivers are not installed, enter:

     mkdir /usr/openlink

     The system creates the **/usr/openlink** directory.
7. Enter:

   cd openlink

   Note:
   If you are installing the client on the same system as the server, you must remove any existing ODBC taz files. Enter: **rm /usr/openlink/*taz**
8. Enter:

   uname -a

   The system displays the current Solaris version.

```
SunOS CMS1 5.9 Generic_108528-08 sun4u sparc...
```

9. Record the Solaris version for use later in this procedure.

   Example:

   In the previous example, the Solaris version is 5.9.
10. Copy the client components to **/usr/openlink**, enter:

    cp /cdrom/cdrom0/client/solaris2.8/* /usr/openlink
11. Enter:

    ./install.sh

    Note:
    If the required broker is running, you might see a message that asks you to shut down the broker. At the prompt, press **Enter** to choose the default.

    The system displays the following message:

```
Extracting (srami9zz.taz) ...
.........
.........
.........
Enter the port number the the broker will listen on for
client connections [Enter=Default] :
```

12. Accept the default setting by pressing **Enter**.

    The system displays the following message:

```
Welcome to the OpenLink Admin Assistant Setup.

This program will install the HTTP based OpenLink Admin Assistant,
thereby enabling remote configuration for all OpenLink Server
Components (Rule Book,
Service and Database Agents) from any Web Browser.

TCP/IP Port to use? [ENTER=8000] :
```

13. Accept the default setting by pressing **Enter**.

    The system displays the following message:

```
Log File? [ENTER=www_sv.log]
```

14. Accept the default setting by pressing **Enter**.

    The system displays the following message:

```
Log all requests (y/n)? [ENTER=n]
```

15. Accept the default setting by pressing **Enter**.

    The system displays the following message:

```
The OpenLink Admin Assistant is now ready for use.
..........
..........
..........
Enter the name of the user that will own the programs [ENTER=Use
Current User Settings] :
```

16. Enter:

    root

    The system displays the following message.

```
Enter the name of the group that will own the programs [ENTER=Use
Current Group Settings] :
```

17. Enter:

    root

    The system displays the following message:

```
Changing ownership ...

End of installation
```

18. Go to Configuring ODBC on the Solaris client on page 46.

Configuring ODBC on the Solaris client

You must configure the correct drivers for your system to work properly and you must also modify the **/usr/openlink/cms_odbc.ini** file. You must place the modified **/usr/openlink/cms_odbc.ini** file in the **$HOME** directory as **.odbc.ini** for every user that will initiate the client application.

CAUTION:
The client application will fail if it is initiated by a user who does not have an **.odbc.ini** file.

To configure the ODBC 5.2 drivers on the Solaris client:

1. Enter:

   cp /usr/openlink/cms_odbc.ini $HOME/.odbc.ini

   The system copies the **cms_odbc.ini** file to the **$HOME/** directory and renames the **cms_odbc.ini** file to **.odbc.ini**

2. Enter:

   cd $HOME

3. Open the **.odbc.ini** file. Enter:

   vi .odbc.ini

4. Edit the `Host` parameter in the Informix section. Enter the host name of the system you will be connecting to.

   Example:

   The modified file will look similar to the following:

```
[Informix2000]
Driver= /usr/openlink/lib/oplodbc.so.1
Host = CMS1
ServerType= Informix 2000
ServerOptions=
Database= cms
Options=
ReadOnly=Yes
FetchBufferSize= 60
```

5. Save and close the file by pressing **Esc**. Then enter:

   :wq!

6. Edit your **.profile** file by entering:

   vi .profile

7. Enter the following lines at the end of the **.profile** file:

   LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/openlink/lib

   export LD_LIBRARY_PATH

   UDBCINI=$HOME/.odbc.ini

   export UDBCINI

   Example:

   The modified **.profile** file should look similar to the following:

```
# PROFILEVERSION: r14aa.x
# Warning: this file has been modified by CMS
# Be very careful when making changes.
# NOTE 1: do not remove the first line of the file
# NOTE 2: PROFILEVERSION indicates the load inwhich the
#         file is changed and delivered.  Given that changes
#         to the file are not delivered every load,
#         the PROFILEVERSION does not generally equal
#         the load being built or installed.  The PROFILEVERSION
#         should not be modified on a customer machine.
. . .
. . .
. . .
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/openlink/lib
export LD_LIBRARY_PATH
UDBCINI=$HOME/.odbc.ini
export UDBCINI
```

8. Save and close the file by pressing **Esc**. Then enter:

   :wq!

9. Enter:

   . .profile

   The system executes the **.profile** file.

10. Turn on the Openlink request broker by entering:

    /cms/dc/odbc/odbc_init -r 1

11. Go to Testing ODBC connectivity on a Solaris client on page 48.

Testing ODBC connectivity on a Solaris client

On the Solaris client, initiate the test application to verify connectivity with the following procedure:

1. Enter:
   - If the client and server software are on the same system, enter:

     cd /usr/openlink/samples/ODBC

   - If the client and server software are on different systems, enter:

     cd /usr/openlink/odbcsdk/examples

2. Enter:

   ./odbctest

   The system displays the following message:

```
DSN = InformixX;(the valid value for your CMS)
UID = Valid _UNIX_User_on_Server;
PWD = User_password
```

3. Enter the ODBC connect string:

   Example:

   `DSN=Informix2000;UID=cms;PWD=cmspasswd`

   where `DSN` = data source name, `UID` = cms client ID, and `PWD` = cms client password.

4. At the `SQL >` prompt, enter a valid SQL query.

   Example:

   `select count(*) from hsplit`

Database tables

This section describes the CMS historical database tables that can be accessed through ODBC. For detailed information about specific database tables, see the appropriate database items and calculations document for your CMS release.

This section includes the following topics:

Things to consider when using ODBC

Some things to consider when using ODBC with CMS are:

Dictionary names: Clients can access CMS Dictionary names. You must map the synonym to the report from the client.

Permissions: Applications that access Informix externally, such as database access scripts, may not work if the table permission script tries to access a table to which permission is denied.

Field display: The time and date data you receive from the database may not be formatted. Generally, times may be shown in seconds or in 24-hour (military) format. You will need to review data for formatting when you import it into your software application. The data returned from your SQL queries will be formatted in the manner described in your database interface specifications. See your software's documentation for further information on formatting data.

Switch features and capabilities: Some switch features and capabilities have an impact on CMS open database items. For more information on these features and capabilities, see the appropriate database items and calculations document for your CMS release.

CMS database logic structure

CMS historical tables store information in one record per row format. This formatting affects the way data can be accessed through ODBC. When accessing data in the historical tables, you may need to sum the information to retrieve complete data.

For example, a record will be created for each split/skill that an agent is logged into in the agent tables. If an agent is logged into four splits/skills, there will be four records for that agent. Similarly, if an agent starts the day with four splits/skills, and is added to a fifth split/skill before the end of the day, the agent's fifth record will be generated only from the point at which the additional split/skill was added. The other four records will reflect the total logon time.

This section contains the following topics:

Agent tables

If an agent logs off and logs on more than once in a specified interval, another complete set of records is created for that agent for each logon in the agent tables.

For example, if an agent logs into four split/skills, logs out, and then logs back on during a set interval, there will be two sets of four records for that agent, one set per logon.

VDN tables

The VDN tables store one record per vector on which a VDN terminates. Therefore, if the terminating vector for a specific VDN changes in a set interval, there are two records for that VDN - one per terminating vector. This logic also applies to the Vector, Trunk, Trunk Group, and Split/Skill tables. If information is required from these tables, a sum structured query language (SQL) query may be necessary to access complete data from each table.

Circular structure tables

The Exceptions, Call Record, and Agent Trace tables are circular files. These tables populate continuously, until the table capacity plus ten percent has been reached. At that point, the oldest ten percent of the records are deleted.

For example, if an agent trace table has a capacity of 100 rows, and the total rows populated equals 110, the oldest ten rows will automatically be deleted. Therefore, the data in that table will change continuously as the table is updated.

CMS database table names

To select data for external use, you must use the names listed in the following table in your queries. The following table contains all the tables that are ODBC-accessible in CMS and a brief description of the data in each. If a table is not listed here, it is not accessible through ODBC. The table will still be accessible by **root**. All tables included in this chapter have read-only external user access.

ODBC accessible tables

Name	Data stored
hsplit	Split/Skill data for each intrahour interval
dsplit	Split/Skill data summarized by day
wsplit	Split/Skill data summarized by week
msplit	Split/Skill data summarized by month
hagent	Agent data for each intrahour interval
dagent	Agent data summarized by day
wagent	Agent data summarized by week
magent	Agent data summarized by month
htkgrp	Trunk group data for each intrahour interval
dtkgrp	Trunk group data summarized by day
wtkgrp	Trunk group data summarized by week
mtkgrp	Trunk group data summarized by month
htrunk	Trunk data for intrahour interval
dtrunk	Trunk data summarized by day

ODBC accessible tables (continued)

Name	Data stored
wtrunk	Trunk data summarized by week
mtrunk	Trunk data summarized by month
hvector	Vector data for each intrahour interval
dvector	Vector data summarized by day
wvector	Vector data summarized by week
mvector	Vector data summarized by month
hvdn	VDN data for each intrahour interval
dvdn	VDN data summarized by day
wvdn	VDN data summarized by week
mvdn	VDN data summarized by month
hcwc	Call work code data for each intrahour interval
dcwc	Call work code data summarized by day
wcwc	Call work code data summarized by week
mcwc	Call work code data summarized by month
call_rec	Call record data
d_secs	The number of seconds in the daily data collection period
m_secs	The number of seconds in the monthly data collection period
w_secs	The number of seconds in the weekly data collection period
arch_stat	The status of archiver executions
customer_log	The customer error log data
agroups	Agent group definitions
synonyms	Dictionary synonyms

ODBC accessible tables (continued)

Name	Data stored
acd_shifts	Shift times and maximum agents logged in for each shift
dbitems	Dictionary standard and custom database items, constants, and calculations
f_cday	Forecast current day configuration data by split/skill
f_cdayrep	Current day forecast data by split/skill
haglog	Agent login and logout information
ag_actv	Agent activity trace data
agex	Agent exceptions
fullex	Disk full exceptions
spex	Split/skill exceptions
tgex	Trunk group exceptions
vecex	Vector exceptions
vdnex	VDN exceptions
linkex	Link down exceptions
mctex	Malicious call trace exceptions

Description of the CMS database tables

This section provides an overview of the types of data collected by CMS and definitions for the data presented in CMS database table items on page 58.

This section contains the following information:

About the Database item column

The Database item column contains the name of a particular database item. Many database items are contained in more than one database table.

Index database items

The index database items in each table are marked (index). Indexes add structure to table rows so that CMS can retrieve data faster. The row search criteria you define for custom reports should be based on indexes whenever possible. For historical custom reports, always include a "where" clause based on the ROW_DATE database item.

Call-based data and interval-based data

Items in the CMS database can be either call-based or interval-based. Most CMS database items are call-based. **Call-based data is** committed to the database after a call completes. Therefore, if a call starts and ends in different collection intervals, all of the data are recorded in the interval in which the call and any After Call Work (ACW) are completed.

Interval-based data represents the amount of time during a collection interval that is used for a particular activity. Interval-based items are updated throughout the collection interval and timing is restarted at the end of the interval. Most interval-based items start with I_ or TI_. The database items ALLINUSETIME (trunk-group tables) and MBUSYTIME (trunk and trunk-group tables) are also interval-based. Each database item has a defined column type.

About the Data types column

The Data type column contains a letter that represents a specific data type. The following table describes the data types:

Data type	Description
A = Administrative data	Administered on the ECS or on CMS. For example, the database item INTRVL in the split/skill table contains the number of minutes in the intrahour interval (15, 30, 60) assigned for the specified ACD on CMS.
B = Busy Hour data	Gives data that is only meaningful for the busy hour.
C = Cumulative data	Accumulates throughout the collection interval. Most real-time database items contain cumulative data.
I = Row Identifier data	Gives data that is common to all tables, such as time, date, split in the split/skill tables, and so on.

Data type	Description
M = Maximum Interval Value data	Gives data that is the maximum reached for any value in the specified interval.
N = Special Table data	Belongs only to a specific table, such as the Historical Agent Login/Logout table or Current Day Forecast table.
S = Status data	Shows the current status (a snapshot of a particular ACD element). For example, the database item INQUEUE in the split/skill real-time table contains the number of split/skill calls currently waiting in queue.

Note:
Administrative, Cumulative, Maximum Interval Value, Row Identifier and Busy Hour data items apply to historical and real-time database items.

Special Table data items apply only to historical database items.

About the Column type and Length columns

The Column type and Length columns describe the format of a database item. These columns include either the length in bits of the database item or the Informix data type.

Column type and length information is included for the user's reference. Because data gathered through ODBC can be used in a variety of applications, it is helpful to know what type of data you are accessing and how long it is in bytes. Any difference or exception in the column type or length between tables in a table group is indicated in the Column type and Length columns.

Informix data column types table

The following table defines the data column types.

Column type	Definition
char(*n*)	An ASCII string of *n* characters, 1 byte per character.
date	Informix date type, 4 bytes long. The Informix date format is yyyy-mm-dd. For example, May 19, 1998 would display as: 1998-05-19.
integer	4 byte integer

smallint	2 byte integer
smallfloat	Informix floating point numerical type, 4 bytes long. The Informix smallfloat format is a decimal type used for percentages, and includes a comma and a plus/ minus sign.

CMS database table items

This section provides a list of the database items that can be accessed through ODBC. Not all database items will be available with every CMS or switch release.

For additional information about specific database items, see the appropriate database items and calculations document for your CMS release. Descriptions are provided for any items that are not included in the database items and calculations document.

This section contains the following topics:

Agent database items

The Agent database item descriptions apply only to historical items. Historical agent database items apply to the Intrahour Agent (**hagent**), Daily Agent (**dagent**), Weekly Agent (**wagent**), and Monthly Agent (**magent**) tables. Unless indicated otherwise, all items listed in the Agent database item table on page 59 are included in all four tables.

Agent database item table

Database item	Data type	Column type	Length
ABNCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
ABNTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
ACD (index)	I	smallint	2 bytes
ACD_RELEASE	C	integer (*not* in hagent table)	4 bytes
ACDAUXOUTCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
ACDCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
ACDCALLS_R1	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
ACDCALLS_R2	C	integer	4 bytes
		smallint (hagent)	2 bytes
ACDTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
ACWINCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
ACWINTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
ACWOUTADJCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
ACWOUTCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
ACWOUTOFFCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
ACWOUTOFFTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
ACWOUTTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
ACWTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
ANSRINGTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
ASSISTS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
AUXINCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
AUXINTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
AUXOUTADJCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
AUXOUTCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
AUXOUTOFFCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
AUXOUTOFFTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
AUXOUTTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
CONFERENCE	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_ABNCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_ABNTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
DA_ACDCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_ACDTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
DA_ACWINCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_ACWINTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
DA_ACWOADJCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_ACWOCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_ACWOOFFCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_ACWOOFFTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
DA_ACWOTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
DA_ACWTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
DA_ANSTIME	C	integer	4 bytes
		smallint (hagent table)	2 bytes
DA_OTHERCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
DA_OTHERTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
DA_RELEASE	C	integer	4 bytes
EVENT1-9	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
EXTENSION	A	char(6)	6 byte ASCII text string
HOLDABNCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
HOLDACDTIME	C	integer	4 bytes
HOLDCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
HOLDTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
I_ACDAUXINTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_ACDAUX_OUTTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_ACDOTHERTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_ACDTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_ACWINTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_ACWOUTTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_ACWTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_AUXINTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_AUXOUTTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
I_AUXSTBYTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_AUXTIME	C	integer	4 bytes
I_AUXTIME0 - 99 (I_AUXTIME0 - 99 is only available with a CMS load that has been upgraded to use 100 AUX reason codes.)	C	integer	4 bytes
I_AUXTIME_R1	C	integer	4 bytes
I_AUXTIME_R2	C	integer	4 bytes
I_AVAILTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_DA_ACDTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_DA_ACWTIME	C	integer	4 bytes
		smallint (hagent table	2 bytes
I_OTHERSTBYTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_OTHERSTBYTIME_R1	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_OTHERSTBYTIME_R2	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
I_OTHERTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_RINGTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
I_STAFFTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
INCOMPLETE	C	smallint	2 bytes
INTRVL	A	smallint (only in hagent table)	2 bytes
LOC_ID	C	integer	4 bytes
LOGID (index)	A	char(10)	10 byte ASCII text string
NOANSREDIR	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
O_ACDCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
O_ACDTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
O_ACWTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
PHANTOMABNS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
RINGCALLS	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes
RINGTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
ROW_DATE (index)	I	date	4 byte Informix date
SPLIT (index)	I	smallint	2 bytes
STARTTIME	I	smallint (only in hagent table)	2 bytes
TI_AUXTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
TI_AUXTIME0 - 9 or TI_AUXTIME0 - 99 (TI_AUXTIME0 - 99 is only available with a CMS load that has been upgraded to use 100 AUX reason codes.)	C	integer	4 bytes

Agent database item table (continued)

Database item	Data type	Column type	Length
TI_AVAILTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
TI_OTHERTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
TI_STAFFTIME	C	integer	4 bytes
		smallint (hagent)	2 bytes
TRANSFERRED	C	smallint (dagent, hagent)	2 bytes
		integer (magent, wagent)	4 bytes

Agent Login/Logout database items

The Agent Login/Logout database items are historical items that apply to the Agent Login/Logout (**haglog**) table. The Agent Login/Logout database item table on page 69 describes the data items in the CMS Agent Login/Logout database.

Agent Login/Logout database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
EXTN	char(6)	6 byte ASCII text string
INFLAG	char(1)	1 byte ASCII text string
LOC_ID	integer	4 bytes

Agent Login/Logout database item table (continued)

Database item	Column type	Length
LOGID	char(10)	10 byte ASCII text string
LOGIN	integer	4 bytes
LOGONSKILL2-20	smallint	2 bytes
LOGOUT	integer	4 bytes
LOGOUT_DATE	date	4 byte Informix date
LOGOUT_REASON	smallint	2 bytes
OUTFLAG	char(1)	1 byte ASCII text string
PREFERENCE	integer	4 bytes
SKLEVEL	integer	4 bytes
SKLEVEL2-20	integer	4 bytes
SKPERCENT	integer	4 bytes
SKPERCENT2-20	integer	4 bytes
ROW_DATE (index)	date	4 byte Informix date
SKILLTYPE	char(1)	1 byte ASCII text string
SKILLTYPE2-4	char(1)	1 byte ASCII text string
SKLEVEL	smallint	2 bytes
SKLEVEL2-20	smallint	2 bytes
SPLIT (index)	smallint	2 bytes

Agent Trace database items

The Agent Trace database item descriptions are historical items that apply specifically to the Agent Trace (**ag_actv**) table. The Agent Trace database item table on page 71 describes the data items in the CMS Agent Trace database.

Agent Trace database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
AGT_RELEASED	char(1)	1 byte ASCII text string
ASSIST_ACTV	char(1)	1 byte ASCII text string
AUXREASON	smallint	2 bytes
CALLER_HOLD	char(1)	1 byte ASCII text string
CALLING_II	char(2)	2 byte ASCII text string
CALLING_PTY	char(12)	12 byte ASCII text string
CONFERENCE	char(1)	1 byte ASCII text string
DIGITS_DIALED	char(16)	16 byte ASCII text string
DIRECTION	smallint	2 bytes

Agent Trace database item table (continued)

Database item	Column type	Length
DURATION	integer	4 bytes
EVENT_TIME	integer	4 bytes
EXT_CALL_ORIG	char(1)	1 byte ASCII text string
KEYBD_DIALED	char(1)	1 byte ASCII text string
LOC_ID	integer	4 bytes
LOGID (index)	char(10)	10 byte ASCII text string
LOGOUTREASON	smallint	2 bytes
MCT	char(1)	1 byte ASCII text string
RECONNECT	char(1)	1 byte ASCII text string
ROW_DATE (index)	date	4 byte Informix date
SPLIT	smallint	2 bytes
STARTTIME	smallint	2 bytes
TRANSFERRED	char(1)	1 byte ASCII text string
UCID	char(20)	20 byte ASCII text string
WMODE_SEQ	smallint	2 bytes

Agent Trace database item table (continued)

Database item	Column type	Length
WORKCODE	char(16)	16 byte ASCII text string
WORKMODE	smallint	2 bytes

Call Record database items

The Call Record database item descriptions are historical items that apply specifically to the Call Record (**call_rec**) table. The Call Record database item table on page 73 describes the database items in the CMS Call Record database.

Call Record database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
ACWTIME	integer	4 bytes
AGT_RELEASED	smallint	2 bytes
ANSHOLDTIME	integer	4 bytes
ANSLOGIN	char(9)	9 byte ASCII text string
ANSREASON	smallint	2 bytes
ASSIST	smallint	2 bytes
AUDIO	smallint	2 bytes
CALLID	integer	4 bytes
CALLING_II	char(2)	2 byte ASCII text string

Call Record database item table (continued)

Database item	Column type	Length
CALLING_PTY	char(12)	12 byte ASCII string
CONFERENCE	smallint	2 bytes
CONSULTTIME	integer	4 bytes
DA_QUEUED	smallint	2 bytes
DIALED_NUM	char(24)	24 byte ASCII text string
DISPIVECTOR	smallint	2 bytes
DISPOSITION	smallint	2 bytes
DISPPRIORITY	smallint	2 bytes
DISPSKLEVEL	smallint	2 bytes
DISPSPLIT	smallint	2 bytes
DISPTIME	integer	4 bytes
DISPVDN	char(5)	5 byte ASCII text string
DURATION	integer	4 bytes
EQLOC	char(8)	8 byte ASCII text string
EVENT1-9	smallint	2 bytes
FIRSTVECTOR	smallint	2 bytes
FIRSTVDN	char(5)	5 byte ASCII text string
HELD	smallint	2 bytes
HOLDABN	smallint	2 bytes

Call Record database item table (continued)

Database item	Column type	Length
LASTCWC	char(16)	16 byte ASCII text string
LASTDIGITS	char(16)	16 byte ASCII text string
LASTOBSERVER	char(9)	9 byte ASCII text string
MALICIOUS	smallint	2 bytes
NETINTIME	integer	4 bytes
OBSERVINGCALL	smallint	2 bytes
ORIGHOLDTIME	integer	4 bytes
ORIGLOGIN	char(9)	9 byte ASCII text string
ORIGREASON	smallint	2 bytes
ROW_DATE	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEGMENT	smallint	2 bytes
SEGSTART	integer	4 bytes
SEGSTOP	integer	4 bytes
SPLIT1	smallint	2 bytes
SPLIT2	smallint	2 bytes
SPLIT3	smallint	2 bytes
TALKTIME	integer	4 bytes
TKGRP	smallint	2 bytes

Call Record database item table (continued)

Database item	Column type	Length
TRANSFERRED	smallint	2 bytes
UCID	char(20)	20-byte ASCII text string

Call Work Codes database items

The Call Work Codes database item descriptions apply to historical items. Historical call work codes database items apply to the Intrahour Call Work Codes (**hcwc**), Daily Call Work Codes (**dcwc**), Weekly Call Work Codes (**wcwc**), and Monthly Call Work Codes (**mcwc**) tables. Unless indicated otherwise, all items listed in the Call Work Codes database item table on page 76 are included in all four tables.

Call Work Codes database item table

Database item	Data type	Column type	Length
ACD (index)	I	smallint	2 bytes
ACDCALLS	C	integer	4 bytes
		smallint (hcwc)	2 bytes
ACDTIME	C	integer	4 bytes
ACWTIME	C	integer	4 bytes
CWC (index)	I	char(16)	16 byte ASCII text string
INCOMPLETE	C	smallint	2 bytes
INTRVL	A	smallint (hcwc only)	2 bytes

Call Work Codes database item table (continued)

Database item	Data type	Column type	Length
ROW_DATE (index)	I	date	4 byte Informix date
STARTTIME	I	smallint (hcwc)	2 bytes

Exceptions historical database items

CMS stores exception types using the numerical values, then translates the numbers into the text you see in standard exception reports. For detailed information about specific database items, see *Avaya CMS Database Items and Calculations,* 585-780-702.

This section contains the following exception tables:

Agent Exceptions database items

The Agent Exceptions database items apply to the Agent Exceptions (**agex**) table.

Agent Exceptions database item table

Database item	Column type	Length
ACD	smallint	2 bytes
EXTYPE	smallint	2 bytes

Agent Exceptions database item table (continued)

Database item	Column type	Length
LOGID	char(10)	10 byte ASCII text string
REASON_CODE	smallint	2 bytes
ROW_DATE (index)	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEQNO	integer	4 bytes
SPLIT	smallint	2 bytes
THRESHOLD	smallint	2 bytes
TIME	smallint	2 bytes
AGLOCID	smallint	2 bytes

Split/Skill Exceptions database items

The Split/Skill Exceptions database items apply to the Split/Skill Exceptions (**spex**) table.

Split/Skill Exceptions database item table

Database item	Column type	Length
ACD	smallint	2 bytes
EXTYPE	smallint	2 bytes
ROW_DATE (index)	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEQNO	integer	4 bytes
SPLIT	smallint	2 bytes
THRESHOLD	smallint	2 bytes
TIME	smallint	2 bytes

Trunk Group Exceptions database items

The Trunk Group Exceptions database items apply to the Trunk Group Exceptions (**tgex**) table.

Trunk Group Exceptions database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
EXTYPE	smallint	2 bytes
EQLOC	char(9)	9 byte ASCII text string
LOGID	char(10)	10 byte ASCII text string
ROW_DATE	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEQNO	integer	4 bytes
THRESHOLD	smallint	2 bytes
TIME	smallint	2 bytes
TKGRP	smallint	2 bytes

VDN Exceptions database items

The VDN Exceptions database items apply to the VDN Exceptions (**vdnex**) table. VDN exceptions are only available with the Vectoring feature.

VDN Exceptions database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
EXTYPE	smallint	2 bytes

VDN Exceptions database item table (continued)

Database item	Column type	Length
ROW_DATE	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEQNO	integer	4 bytes
THRESHOLD	smallint	2 bytes
TIME	smallint	2 bytes
VDN	char(6)	6 byte ASCII text string
VECTOR	smallint	2 bytes

Vector Exceptions database items

The Vector Exceptions database items apply to the Vector Exceptions (**vecex**) table. Vector exceptions are only available with the Vectoring feature.

Vector Exceptions database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
EXTYPE	smallint	2 bytes
ROW_DATE	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEQNO	integer	4 bytes
THRESHOLD	smallint	2 bytes
TIME	smallint	2 bytes
VECTOR	smallint	2 bytes

Malicious Call Trace Exceptions database items

The Malicious Call Trace Exceptions database items apply to the Malicious Call Trace Exceptions (**mctex**) table.

Malicious Call Trace Exceptions database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
ANI_SID	char(8)	8 byte ASCII text string
EQLOC	char(9)	9 byte ASCII text string
II_DIGITS	smallint	2 bytes
LOGID	char(10)	10 byte ASCII text string
ROW_DATE	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEQNO	integer	4 bytes
SPLIT	smallint	2 bytes
TKGRP	smallint	2 bytes
VDN	char(6)	6 byte ASCII text string

Data Collection Exceptions database items

The Data Collection Exceptions database items apply to the Data Collection Exceptions (**linkex**) table.

Data Collection Exceptions database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
DURATION	integer	4 bytes
REASON	smallint	2 bytes
ROW_DATE	date	4 byte Informix date
ROW_TIME	smallint	2 bytes
SEQNO	integer	4 bytes
THRESHOLD	smallint	2 bytes

Disk Full Exceptions database items

The Disk Full Exceptions database items apply to the Disk Full Exceptions (**fullex**) table.

Disk Full Exceptions database item table

Database item	Description	Column type	Length
PROC_NAME	The name of the process that failed because the disk was full.	char(30)	30 byte ASCII text string
ROW_DATE	The date at which the disk was full.	date	4 byte Informix date
ROW_TIME	The time at which the disk was full.	smallint	2 bytes

Disk Full Exceptions database item table (continued)

Database item	Description	Column type	Length
SEQNO	The sequence number of this record.	smallint	2 bytes
TASK_GRP	The activity that failed because the disk was full.	char(2)	2 byte ASCII text string

Split/Skill database items

The Split/Skill database item descriptions apply to historical items. Historical split/skill database items apply to the following tables:

- Intrahour Split/Skill (**hsplit**)
- Daily Split/Skill (**dsplit**)
- Weekly Split/Skill (**wsplit**)
- Monthly Split/Skill (**msplit**)

Unless indicated otherwise, all of the database items listed in the Split/Skill database item table on page 83 are included in all four tables.

Split/Skill database item table

Database item	Data type	Column type	Length
ABNCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ABNCALLS1-10	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ABNRINGCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ABNTIME	C	integer	4 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
ACCEPTABLE	C	integer	4 bytes
ACD (index)	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACDAUXOUTCALLS	A	smallint	2 bytes
ACDCALLS	C	integer	4 bytes
		smallint in hsplit table	2 bytes in hsplit table
ACDCALLS1-10	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACDCALLS_R1	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACDCALLS_R2	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACDTIME	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACWINCALLS	C	integer	4 bytes
ACWINTIME	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACWOUTADJCALLS	C	integer	4 bytes
ACWOUTCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
ACWOUTOFFCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACWOUTOFFTIME	C	integer	4 bytes
		smallint (hsplit)	2 bytes
ACWOUTTIME	C	integer	4 bytes
ACWTIME	C	integer	4 bytes
ANSTIME	C	integer	4 bytes
ASSISTS ASSISTS	C C	integer	4 bytes
		smallint (hsplit)	2 bytes
AUXINCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
AUXINTIME	C	integer	4 bytes
AUXOUTADJCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
AUXOUTCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
AUXOUTOFFCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
AUXOUTOFFTIME	C	integer	4 bytes
AUXOUTTIME	C	integer	4 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
BACKUPCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
BUSYCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
BUSYTIME	C	integer	4 bytes
CALLSOFFERED	C	integer	4 bytes
		smallint (hsplit)	2 bytes
CONFERENCE	C	integer	4 bytes
		smallint (hsplit)	2 bytes
DA_ACWINCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
DA_ACWINTIME	C	integer	4 bytes
DA_ACWOCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
DA_ACWOTIME	C	integer	4 bytes
DEQUECALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
DEQUETIME	C	integer	4 bytes
DISCCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
DISCTIME	C	integer	4 bytes
EVENT1-9	C	integer	4 bytes
		smallint (hsplit)	2 bytes
HIGHCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
HOLDABNCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
HOLDCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
HOLDTIME	C	integer	4 bytes
I_ACDAUXINTIME	C	integer	4 bytes
I_ACDAUX_OUTTIME	C	integer	4 bytes
I_ACDOTHERTIME	C	integer	4 bytes
I_ACDTIME	C	integer	4 bytes
I_ACDTIME_R1	C	integer	4 bytes
I_ACDTIME_R2	C	integer	4 bytes
I_ACWINTIME	C	integer	4 bytes
I_ACWOUTTIME	C	integer	4 bytes
I_ACWTIME	C	integer	4 bytes
I_ACWTIME_R1	C	integer	4 bytes
I_ACWTIME_R2	C	integer	4 bytes
I_ARRIVED	C	integer	4 bytes
I_AUXINTIME	C	integer	4 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
I_AUXOUTTIME	C	integer	4 bytes
I_AUXTIME	C	integer	4 bytes
I_AUXTIME0	C	integer	4 bytes
I_AUXTIME1-9	C	integer	4 bytes
I_AVAILTIME	C	integer	4 bytes
I_DA_ACDTIME	C	integer	4 bytes
I_DA_ACWTIME	C	integer	4 bytes
I_NORMTIME	C	integer	4 bytes
I_OL1TIME	C	integer	4 bytes
I_OL2TIME	C	integer	4 bytes
I_OTHERTIME	C	integer	4 bytes
I_OTHERTIME_R1	C	integer	4 bytes
I_OTHERTIME_R2	C	integer	4 bytes
I_RINGTIME	C	integer	4 bytes
I_RINGTIME_R1	C	integer	4 bytes
I_RINGTIME_R2	C	integer	4 bytes
I_STAFFTIME	C	integer	4 bytes
I_TAUXTIME	C	integer	4 bytes
I_TAVAILTIME	C	integer	4 bytes
INCOMPLETE	C	smallint	2 bytes
INFLOWCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
INTERFLOWCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
INTRVL	A	smallint (only in hsplit)	2 bytes
LOWCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
MAXINQUEUE	M	integer	4 bytes
		smallint (hsplit)	2 bytes
MAXOCWTIME	M	integer	4 bytes
MAXSTAFFED	M	integer	4 bytes
		smallint (hsplit)	2 bytes
MAXTOP	M	integer	4 bytes
MEDCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
NOANSREDIR	C	integer	4 bytes
		smallint (hsplit)	2 bytes
O_ABNCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
O_ACDCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
O_ACDTIME	C	integer	4 bytes
O_ACWTIME	C	integer	4 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
O_OTHERCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
OTHERCALLS	C	integer	4 bytes
OTHERTIME	C	integer	4 bytes
OUTFLOWCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
OUTFLOWTIME	C	integer	4 bytes
PERIOD1-9	A	smallint	2 bytes
PERIODCHG	A	integer	4 bytes
		smallint (hsplit)	2 bytes
PHANTOMABNS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
RINGCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
RINGTIME	C	integer	4 bytes
ROW_DATE (index)	I	date	4 bytes Informix date
RSV_LEVEL		smallint	2 bytes
SERVICELEVEL	A	integer	4 bytes
		smallint (hsplit)	2 bytes
SLVLABNS	C	integer	4 bytes
SLVLOUTFLOWS	C	integer	4 bytes

Split/Skill database item table (continued)

Database item	Data type	Column type	Length
SPLIT (index)	I	smallint	2 bytes
STARTTIME	I	smallint (only in hsplit table)	2 bytes
SVCLEVELCHG	A	integer	4 bytes
		smallint (hsplit)	2 bytes
TOPCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
TRANSFERRED	C	integer	4 bytes
		smallint (hsplit)	2 bytes

Trunk Group database items

The Trunk Group database item descriptions apply to historical items. Historical trunk group database items apply to the Intrahour Trunk Group (**htkgrp**), Daily Trunk Group (**dtkgrp**), Weekly Trunk Group (**wtkgrp**), and Monthly Trunk Group (**mtkgrp**) tables. Unless indicated otherwise, items listed in the Trunk Group database item table on page 91 are included in all four tables.

Trunk Group database item table

Database item	Data type	Column type	Length
ABNCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes

Trunk Group database item table (continued)

Database item	Data type	Column type	Length
ABNQUECALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
ABNRINGCALLS	C	integer	4 bytes
ABNVECCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
ACD (index)	I	smallint	2 bytes
ACDCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
ACDCALLS_R1	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
ACDCALLS_R2	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
ALLINUSETIME	C	integer	4 bytes
AUDIO	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
BACKUPCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_ABNCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes

Trunk Group database item table (continued)

Database item	Data type	Column type	Length
BH_ACDCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_ALLINUSETIME	B	integer	4 bytes
BH_BUSYCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_DISCCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_INCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_INTERVAL	B	integer (only in dtkgrp table)	4 bytes
BH_INTIME	B	integer	4 bytes
BH_OABNCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_OACDCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_OOTHERCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_OTHERCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes

Trunk Group database item table (continued)

Database item	Data type	Column type	Length
BH_OUTCALLS	B	integer	4 bytes
		smallint (htkgrp)	2 bytes
BH_OUTTIME	B	integer	4 bytes
BH_STARTTIME	B	integer	4 bytes
BLOCKAGE	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
BUSYCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
COMPLETED	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
CONNECTCALLS	C	integer	4 bytes
DISCCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
FAILURES	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
I_INOCC	C	integer	4 bytes
I_OUTOCC	C	integer	4 bytes
INCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
INCOMPLETE	C	smallint	2 bytes
INTIME	C	integer	4 bytes

Trunk Group database item table (continued)

Database item	Data type	Column type	Length
INTRVL	A	smallint (only in htkgrp table)	2 bytes
MBUSYTIME	C	integer	4 bytes
O_ABNCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
O_ACDCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
O_OTHERCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
OTHERCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
OUTCALLS	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
OUTTIME	C	integer	4 bytes
ROW_DATE (index)	I	date	4 byte Informix date
SETUPTIME	C	integer	4 bytes
SHORTCALLS	C	integer	4 bytes
SPLIT	A	smallint	2 bytes
TKGRP (index)	I	smallint	2 bytes

Trunk Group database item table (continued)

Database item	Data type	Column type	Length
TRANSFERRED	C	integer	4 bytes
		smallint (htkgrp)	2 bytes
TRUNKS	A	smallint	2 bytes
VDN	A	char(6)	6 byte ASCII text string
VECTOR	A	smallint	2 bytes

Trunk database items

The Trunk database item descriptions apply to historical items. Historical trunk database items apply to the Intrahour Trunk (**htrunk**), Daily Trunk (**dtrunk**), Weekly Trunk Group (**wtrunk**), and Monthly Trunk (**mtrunk**) tables. Unless indicated otherwise, all items listed in the Trunk database item table on page 96 are included in all four tables.

Trunk database item table

Database item	Data type	Column type	Length
ABNCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
ACD (index)	I	smallint	2 bytes
ACDCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
ACDCALLS_R1	C	integer	4 bytes
		smallint (htrunk)	2 bytes

Trunk database item table (continued)

Database item	Data type	Column type	Length
ACDCALLS_R2	C	integer	4 bytes
		smallint (htrunk)	2 bytes
AUDIO	C	integer	4 bytes
		smallint (htrunk)	2 bytes
EQLOC (index)	A	char(8)	8 byte ASCII text string
FAILURES	C	integer	4 bytes
		smallint (htrunk)	2 bytes
I_INOCC	C	integer	4 bytes
		smallint (htrunk)	2 bytes
I_OUTOCC	C	integer	4 bytes
		smallint (htrunk)	2 bytes
INCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
INCOMPLETE	C	smallint	2 bytes
INTIME	C	integer	4 bytes
		smallint (htrunk)	2 bytes
INTRVL	A	smallint (only in htrunk table)	2 bytes
LOC_ID	C	integer	4 bytes

Trunk database item table (continued)

Database item	Data type	Column type	Length
MBUSYTIME	C	integer	4 bytes
		smallint (htrunk)	2 bytes
O_ABNCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
O_ACDCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
O_OTHERCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
OUTCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
OTHERCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
OUTTIME	C	integer	4 bytes
		smallint (htrunk)	2 bytes
ROW_DATE (index)	I	date	4 byte Informix date
SHORTCALLS	C	integer	4 bytes
		smallint (htrunk)	2 bytes
TKGRP (index)	A	smallint	2 bytes

Vector database items

The Vector database item descriptions apply to historical items. Vector database items are only available if you purchased the optional Vectoring feature.

Historical vector database items apply to the Intrahour Vector (**hvector**), Daily Vector (**dvector**), Weekly Vector (**wvector**), and Monthly Vector (**mvector**) tables. Unless indicated otherwise, all items listed in the Vector database item table on page 99 are included in all four tables.

Vector database item table

Database item	Data type	Column type	Length
ABNCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
ABNQUECALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
ABNRINGCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
ABNTIME	C	integer	4 bytes
ACD (index)	I	smallint	2 bytes
ACDCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
ACDCALLS_R1	C	integer	4 bytes
		smallint (hvector)	2 bytes
ACDCALLS_R2	C	integer	4 bytes
		smallint (hvector)	2 bytes

Vector database item table (continued)

Database item	Data type	Column type	Length
ADJATTEMPTS	C	integer	4 bytes
		smallint (hvector)	2 bytes
ADJROUTED	C	integer	4 bytes
		smallint (hvector)	2 bytes
ANSTIME	C	integer	4 bytes
BACKUPCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
BUSYCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
BUSYTIME	C	integer	4 bytes
DEFLECTCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
DISCCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
DISCTIME	C	integer	4 bytes
GOTOCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
GOTOTIME	C	integer	4 bytes
INCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes

Vector database item table (continued)

Database item	Data type	Column type	Length
INCOMPLETE	C	smallint	2 bytes
INFLOWCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
INTERFLOWCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
INTIME	C	integer	4 bytes
INTRVL	A	smallint (only in hvdn table)	2 bytes
LOOKATTEMPTS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
LOOKFLOWCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
NETDISCCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
NETPOLLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
OTHERCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
OTHERTIME	C	integer	4 bytes
OUTFLOWCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes

Vector database item table (continued)

Database item	Data type	Column type	Length
OUTFLOWTIME	C	integer	4 bytes
PHANTOMABNS	C	integer	4 bytes
RINGCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
RINGTIME	C	integer	4 bytes
ROW_DATE (index)	I	date	4 byte Informix date
VDISCCALLS	C	integer	4 bytes
		smallint (hvector)	2 bytes
VECTOR (index)	I	smallint	2 bytes

VDN database items

The VDN Database item descriptions apply to historical items. VDN database items are only available if you purchased the optional Vectoring feature.

Historical VDN database items apply to the Intrahour VDN (**hvdn**), Daily VDN (**dvdn**), Weekly VDN (**wvdn**), and Monthly VDN (**mvdn**) tables. Unless indicated otherwise, all items listed in the VDN database item table on page 102 are included in all four tables.

VDN database item table

Database item	Data type	Column type	Length
ABNCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes

VDN database item table (continued)

Database item	Data type	Column type	Length
ABNCALLS1-10	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ABNQUECALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ABNRINGCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ABNTIME	C	integer	4 bytes
ACCEPTABLE	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ACD (index)	I	smallint	2 bytes
ACDCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ACDCALLS _R1	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ACDCALLS _R2	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ACDTIME	C	integer	4 bytes
ACWTIME	C	integer	4 bytes
ADJATTEMPTS	C	integer	4 bytes
		smallint (hvdn)	2 bytes

VDN database item table (continued)

Database item	Data type	Column type	Length
ADJROUTED	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ANSCONNCALLS 1-10	C	integer	4 bytes
		smallint (hvdn)	2 bytes
ANSTIME	C	integer	4 bytes
BACKUPCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_ABNCALLS	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_ACDCALLS	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_ACDTIME	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_BUSYCALLS	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_DISCCALLS	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_INTERVAL	B	integer (only in dvdn table)	4 bytes

VDN database item table (continued)

Database item	Data type	Column type	Length
BH_OTHERCALLS	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_STARTTIME	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BH_VDNCALLS	B	integer	4 bytes
		smallint (hvdn)	2 bytes
BSRPLAN	A	smallint	2 bytes
BUSYCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
BUSYTIME	C	integer	4 bytes
CONNECTCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
CONNECTTIME	C	integer	4 bytes
CONNTALKTIME	C	integer	4 bytes
		smallint (hvdn)	2 bytes
DEFLECTCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
DISCCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
DISCTIME	C	integer	4 bytes

VDN database item table (continued)

Database item	Data type	Column type	Length
HOLDABNCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
HOLDACDCALLS	C	integer	4 bytes
HOLDACDTIME	C	integer	4 bytes
HOLDCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
HOLDTIME	C	integer	4 bytes
I_ARRIVED	C	integer	4 bytes
INCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
INCOMPLETE	C	smallint	2 bytes
INFLOWCALLS	C	integer	4 bytes
		smallint (hsplit)	2 bytes
INTERFLOWCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
INTIME	C	integer	4 bytes
INTRVL	A	smallint (only in hvdn table)	2 bytes
LOOKATTEMPTS	C	integer	4 bytes
		smallint (hvdn)	2 bytes

VDN database item table (continued)

Database item	Data type	Column type	Length
LOOKFLOWCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
MAXOCWTIME	M	integer	4 bytes
		smallint (hvdn)	2 bytes
MAXWAITING	M	integer	4 bytes
		smallint (hvdn)	2 bytes
NETDISCCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
NETINCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
NETINTIME	C	integer	4 bytes
NETPOLLS	C	integer	4 bytes
NOANSREDIR	C	integer	4 bytes
		smallint (hvdn)	2 bytes
NUMTGS	A	integer	4 bytes
OTHERCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
OTHERTIME	C	integer	4 bytes
OUTFLOWCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes

VDN database item table (continued)

Database item	Data type	Column type	Length
OUTFLOWTIME	C	integer	4 bytes
PERIOD1-9	A	smallint	2 bytes
PERIODCHG	A	integer	4 bytes
		smallint (hvdn)	2 bytes
PHANTOMABNS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
RETURNCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
RINGCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
RINGTIME	C	integer	4 bytes
ROW_DATE (index)	I	date	4 byte Informix date
SERVICELEVEL	A	integer	4 bytes
		smallint (hvdn)	2 bytes
SKILLACWTIME1-3	C	integer	4 bytes
SKILLCALLS1-3	C	integer	4 bytes
		smallint (hvdn)	2 bytes
SKILLTIME1-3	C	integer	4 bytes
SKILL1-3	A	smallint	2 bytes
SLVLABNS	C	integer	4 bytes
SLVLOUTFLOWS	C	integer	4 bytes

VDN database item table (continued)

Database item	Data type	Column type	Length
SVCLEVELCHG	A	integer	4 bytes
		smallint (hvdn)	2 bytes
TRANSFERRED	C	integer	4 bytes
		smallint (hvdn)	2 bytes
VDISCCALLS	C	integer	4 bytes
		smallint (hvdn)	2 bytes
VDN (index)	I	char(6)	6 byte ASCII text string
VECTOR (index)	A	smallint	2 bytes

Forecasting database tables

The Forecasting database tables are only available if you purchased the optional Avaya CMS Forecast package.

This section contains the following topics:

Current Day Configuration database items

The Current Day Configuration database items apply to the Current Day (**f_cday**) table.

Current Day Configuration database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
CHANGE	smallfloat	4 byte Informix floating point type
CHPROF	smallint	2 bytes
FMETHOD	smallint	2 bytes
HDATE1-4	date	4 byte Informix date
ROW_DATE (index)	date	4 byte Informix date
SPLIT (index)	smallint	2 bytes
TRENDBASE	date	4 byte Informix date
WT1-4	smallint	2 bytes

Current Day Report database items

The Current Day Report database items apply to the Current Day Report (**f_cdayrep**) table.

Current Day Forecast Report database item table

Database item	Column type	Length
ACD (index)	smallint	2 bytes
AGOCC	smallfloat	4 byte Informix floating point type
AVGAGSERV	smallint	2 bytes
AVGSPEEDANS	smallint	2 bytes
FCALLS	integer	4 bytes
INTRVL	smallint	2 bytes
NUMAGREQ	smallint	2 bytes
RAGOCC	smallfloat	4 byte Informix floating point type
RAVGSPEEDANS	smallint	2 bytes
ROW_DATE (index)	date	4 byte Informix date
RSERVLEVELP	smallfloat	4 byte Informix floating point type
SERVLEVELP	smallfloat	4 byte Informix floating point type
SERVLEVELT	smallint	2 bytes
SPLIT (index)	smallint	2 bytes
STARTTIME	smallint	2 bytes

Administrative database tables

Administrative database tables require read permission from the Maintenance sub-menu. Most of these database items require you to enter additional information in order to become functional. Administrative database tables are available with any CMS or switch release.

This section contains the following topics:

Data Collection Period database items

The Administrative data collection period database items apply to the Daily Data Collection Period (**d_secs**), Weekly Data Collection Period (**w_secs**), and Monthly Data Collection Period (**m_secs**) tables. The tables indicate the number of seconds in the data collection period (daily, weekly, monthly).

Unless indicated otherwise, all items listed in the Data Collection Period database item table on page 112 are included in all three tables.

Data Collection Period database item table

Database item	Description	Data type	Column type	Length
ACD (index)	The ACD number for which data was collected.	A	smallint	2 bytes
ROW_DATE	The day for which data was collected or the exception occurred.	A	date	4 byte Informix date
SECSPERDAY	The number of seconds in the daily data collection period	A	integer (only in d_secs table)	4 bytes

Data Collection Period database item table (continued)

Database item	Description	Data type	Column type	Length
SECSPERMN	The number of seconds in the monthly data collection period.	A	integer (only in m_secs table)	4 bytes
SECSPERWK	The number of seconds in the weekly data collection period.	A	integer (only in w_secs table)	4 bytes

Archiver Execution Status database items

The Archiver Execution Status database item descriptions apply specifically to items in the Archiver Execution Status (**arch_stat**) table. The table contains status information on recent archiver executions, and displays status and the next run scheduled.

Archiver Execution Status database item table

Database item	Description	Data type	Column type	Length
ACD (index)	The ACD number for which data was collected.	A	smallint	2 bytes
ARCH_TYPE (index)	The type of archiver executions being run. Values are: 1 = interval, 2 = daily, 3 = weekly, 4 = monthly.	A	char(20)	20 byte ASCII text string
LAST_TIME	The last time the archiver execution was run.	A	char(20)	20 byte ASCII text string
STATUS	The status of the archiver execution. This field indicates if the execution was not run, is currently running, or has finished. Values are: 1 = not run 2 = finished, the archive was successful 3 = finished, the archive had a failure 4 = running If the status field displays 3, for finished, but had a failure, you should consult the error log (customer_log table) and the archive log for the reason prior to troubleshooting.	A	char(9)	9 byte ASCII text string

Customer Log database items

The Customer Log database items apply to the Customer Log (**customer_log**) table. The table contains customer error log information on recent archiver executions. The information includes the error code, the date the error occurred, the severity, the associated event, and a description of the error. The Customer Log table is not backed up by the CMS Maintenance backup.

Customer Log database item table

Database item	Description	Data type	Column type	Length
ACD_ID	The ACD number for which data was collected.	A	integer	4 bytes
COUNTS	The number of occurrences of the error.	A	integer	4 bytes
DATE_OCCURRED	The date that the error occurred.	A	date	4 byte Informix date
DESCRIPTION	A text description of the error.	A	char(256)	256 byte ASCII text string
ERROR_CODE	The error code number.	A	integer	4 bytes
LAST_TIME	The last time the error occurred.	A	integer	4 bytes
SEVERITY	The level of severity of the error.	A	char(10)	10 byte ASCII text string

Agent Group database items

The Agent Group database items apply to the Agent Group (**agroups**) table. The table contains dictionary information on agent groups.

Agent Group database item table

Database item	Description	Data type	Column type	Length
ACD_NO (index)	The ACD number for which data was collected.	A	smallint	2 bytes
ITEM_TYPE (index)	The type "agent group."	A	char(20)	20 byte ASCII text string
ITEM_NAME (index)	The name of the agent group.	A	char(20)	20 byte ASCII text string
VALUE (index)	An agent login ID belonging to the agent group.	A	char(9)	9 byte ASCII text string

Synonyms database items

The Synonyms database items apply to items in the Synonyms (**synonyms**) table. The table contains dictionary synonyms.

Synonyms database item table

Database item	Description	Data type	Column type	Length
ACD_NO (index)	The ACD number for which data was collected.	A	smallint	2 bytes
DESCR	The description, or definition, of the dictionary synonym.	A	char(150)	5 byte ASCII text string
ITEM_TYPE (index)	The type of synonym.	A	char(20)	20 byte ASCII text string
ITEM_NAME (index)	The name of the synonym. There can be many ITEM_NAMEs for a specific ITEM_TYPE.	A	char(60) (Only first 20 are significant)	60 byte ASCII text string

Synonyms database item table (continued)

Database item	Description	Data type	Column type	Length
STANDARD	This item indicates if the item is a standard or custom synonym. Values are: 1 = standard, not 1 = custom.	A	smallint	2 bytes
VALUE (index)	The item name's corresponding value. Because each ITEM_TYPE can have many different ITEM_NAMEs, a discrete value is assigned to each synonym ITEM_NAME.	A	char(40) (Only first 9 are significant)	40 byte ASCII text string

ACD Shifts database items

The ACD Shifts database items apply to the ACD Shifts (**acd_shifts**) table. The table contains information on ACD shift times and the maximum number of agents logged in for each shift.

ACD Shifts database item table

Database item	Description	Data type	Column type	Length
ACD (index)	The ACD number for which data was collected.	A	smallint	2 bytes
SHIFT_ID	The identification number of the ACD shift. Values are 1 - 4.	A	smallint	2 bytes
START_TIME	The ACD shift start time.	A	smallint	2 bytes
STOP_TIME	The ACD shift stop time.	A	smallint	2 bytes
MAX_AGENTS	The maximum number of agents logged in per shift.	A	smallint	2 bytes

Database Items

Database Items applies to items in the Database Items (**dbitems**) table. The table contains definitions for:

- Dictionary standard and custom database items
- Constants

- Calculations

Database Items table

Database item	Description	Data type	Column type	Length
ITEM_TYPE (index)	The type of data for the row. Valid values are: dbase = database item calc = calculation constant = constant cust_def = customer-defined database item	A	char(8)	8 byte ASCII text string
ITEM_NAME (index)	The name of the data item. There can be many ITEM_NAMEs for a specific ITEM_TYPE.	A	char(20)	20 byte ASCII text string
FORMULA	The formula for the database constant or calculation.	A	char(70)	70 byte ASCII text string
STANDARD	This item indicates if the item is a standard or custom database item. Values are: 1 = standard, not 1 (null) = custom.	A	smallint	2 bytes
DESCR	The description of the database calculation, constant, or standard/custom database item.	A	char(50)	50 byte ASCII text string

Generating a CMS database schema

This section explains how to generate the schema definition information of the CMS database.

This section contains the following topics:

Generating the schema for the entire CMS database

To generate the CMS database schema:

1. Log into the system as **root**.

2. Enter the following command to set the Informix environment:

   ```
   . /opt/informix/bin/setenv
   ```

3. Enter:

   ```
   cd /tmp
   ```

   Note:
   You can use a different directory to contain your database schema with one exception. Never use the **root** (**/**) directory to store a database schema. A database schema can be very large and will use up all of the disk space allocated to the **root** directory. If the **root** directory does not have enough disk space, the CMS system will not function correctly.

4. Enter:

   ```
   dbschema -d cms cms.sql
   ```

   The system saves the CMS database schema in the **/tmp** directory as a file named **cms.sql**.

   Note:
   You can use different options to modify the **dbschema** command used in Step 4. For more information, see About the dbschema command on page 118.

About the dbschema command

You can modify the **dbschema** command using additional command options.

For more information about the **dbschema** command options:

1. Log into the system as **root**.
2. Enter the following command to set the Informix environment:

   ```
   . /opt/informix/bin/setenv
   ```

3. Enter:

   ```
   dbschema
   ```

   The system displays all of the options that you can use to modify the **dbschema** command.

Troubleshooting

This section presents general troubleshooting procedures and error messages for ODBC. For more detailed information, see the on-line Help file that is included on the Avaya CMS OPENLINK ODBC driver CD-ROM.

> **Important:**
> If you choose to develop an application for the ODBC driver, Avaya cannot provide support for that application or for any other third-party software or related mapping.

This section contains the following topics:

Clients cannot connect to the ODBC server

Verify that the ODBC Request Broker is active on the server by entering:

ps -ef | grep oplrqb

The displayed message should show the oplrqb process running from the **/usr/openlink/bin** directory.

The system displays a message similar to the following:

```
root  1462  1459  0 14:41:38 ?
0:00 /usr/openlink/bin/oplrqb -f +configfile /cms/dc/odbc/cmsrqb5.2_init +loglevel 5
+l
```

Multiple connections with the same username

To make multiple connections with the same username:

1. Modify the **cmsrqb5.2_init** file under **/cms/dc/odbc** so that the `ReUse` field under `[generic_inf2000]` is set to `never`.

   Note:
   The default setting is `ifsame database`, `ifsame user`.

2. Enter:

   `cd /cms/dc/odbc`

3. Turn off ODBC by entering:

   `./odbc_init -r 0`

4. Turn on ODBC by entering:

   `./odbc_init -r 1`

Network support

Avaya does not control customer network configuration or ODBC-enabled client applications. Installation and ongoing maintenance support is limited to determining if data is being transferred correctly in the most basic client/server relationship. This is defined as a CMS system running ODBC on the same network hub as the client PC.

Verify that the trouble occurs on the same network subnet. Then continue with troubleshooting procedures. If the trouble does not occur on the same network subnet, contact the Avaya help line.

Server log file

Once the ODBC driver is installed and initiated, the server log file, **odbc.log** records the logging levels of all ODBC activities. The default log level is **5**. Avaya recommends that users do not set the log level to a value greater than 5.

Review the **odbc.log** file for information about ODBC sessions. Archived records of past ODBC sessions are maintained in the **odbc.log.01** and **odbc.log.02** files. These logging levels are set by running **`./odbc_init -l`** located in the **/cms/dc/odbc** directory.

For more information, see Setting log levels on page 28.

Client trace

The OpenLink client component configuration utility, located in the OpenLink group on your Windows desktop, allows you to enable or disable ODBC trace logging. Trace logging provides you with:

- Records of your entire ODBC session, including all ODBC calls made by the ODBC-compliant application you are using
- Native database error messages that might not have been replaced by the ODBC-compliant application you were using.

See Configuring ODBC tracing options on page 39 for information on configuring this utility.

Possible failure causes for a test application error

A test application error could result from one of the following causes:

Problem	Solution
The driver is not active on the CMS server.	Verify that the ODBC driver is active on the server by entering: **`ps -ef | grep oplrqb`** You should see an **`oplrqb`** process running.
The **.odbc.ini** file is not in the initiator's **$HOME** directory.	Place the **.odbc.ini** file in the initiator's **$HOME** directory.
The **.odbc.ini** file is incorrect.	Verify that the entered host is correct and the database path is correct. For example, **cms**.
The LD_LIBRARY_PATH is not set.	Enter **`echo $LD_LIBRARY_PATH`** to verify that LD_LIBRARY_PATH is included in the display. See Testing ODBC connectivity on a Solaris client on page 48 for more information.
The UDBCINI is not set.	Enter **`echo $UDBCINI`** to verify that **$HOME/.odbc.ini** is included in the display. See Configuring ODBC on the Solaris client on page 46 for more information.
The data source is not consistent with the cmsrqb5.2_init entry.	Verify that the data source you entered is correct. For example, **`Informix2000`**.

For more information about configuration settings, see the Openlink ODBC compatibility table on page 24.

OpenLink error messages

The error messages that you might receive from the OpenLink ODBC driver are displayed on the client and in the server log file. Some common error messages that you might receive and possible solutions are shown in the following table.

Message	Possible solution
`Invalid Username/ Password`	This message is displayed when: • The operating system-level username and password verification is in use, and • When you enter an invalid operating system-level username and password combination when connecting to your OpenLink ODBC Data Source. This occurs even when the username and password combination entered is valid at the database level.
`Unable to Locate Requested Service`	This message is displayed as a result of the OpenLink Session Rules Book being incorrectly configured. This error can be caused by an invalid reference to the OpenLink database agent executable program responsible for providing database access. This error will not occur with the default settings.
`Unable to Load OpenLink Request Agent`	This message indicates a client machine problem at the network transport level. Verify that your client machine's network software is correctly configured.
`Remote Procedure Call (RPC) Unable to Send`	This message indicates a corruption of the communications channel being used by the OpenLink Request Agent. This error typically occurs when the server Request Broker has been shut down. Verify the status of the Request Broker. Try to re-initiate the ODBC session.

`RPC Timed Out`	This message is displayed when timeout settings in either the client side or server side Session Rules Books have been exceeded. This message typically occurs when communication cannot be established from client to server or server to client. Verify that the oplrqb is running on the server side by entering: **`ps -ef | grep oplrqb`** You should see an oplrqb process running.
`RPC Unable to Receive`	This message indicates that the server Request Broker is no longer communicating with the client. This error occurs when the server Request Broker has been shut down or re-initiated during a session. Verify that the Request Broker is active on the server.
`RPC Host Unknown`	This message results from a network failure or invalid host entry in the **Data Source Setup** window. Check the **Hostname:** field entry in that window; if you suspect a network problem, verify this and correct it if necessary; or retry your ODBC session.
`Unknown Database Agent Requested`	This message results when the client administration associated with the ODBC session does not resolve to the oplrqb rules in cmsrqb5.2_init. To resolve this error, the **Data Source Setup** window **Type:** field should be set to the appropriate Informix provider type or domain name. For more information see Openlink ODBC compatibility table on page 24. Retry your ODBC session.
`Unable to Start the Requested Database Agent`	This message results when the oplrqb is able to resolve to a database agent, but is unable to execute the program. To resolve this, set the **Type:** field on the **Data Source Setup** window to the appropriate Informix provider type or domain name. For more information see Openlink ODBC compatibility table on page 24. Alternately, set the generic_inf9 to inf9_mv. Finally, the inf9_mv should exist in **/usr/openlink/bin** as an executable.
`Database Errors`	Database errors are displayed after failed database accesses. To resolve this problem, correct the database query and resubmit it.

Cannot make a secure connection

Perform the following steps before calling you CMS support representative:

1. Verify that the name of your security key file, matches the file name in the `Request Broker` section of the **/cms/dc/odbc** file.
2. Regenerate your security key file. For more information, see Configuring ODBC for secure connections on page 29.

Glossary

Abandoned call	A call in which a caller hangs up before receiving an answer from an agent. The call could be queued to a split/skill or in a vector/vector directory number (VDN) or ringing at an agent before it is abandoned.
Access permissions	Permissions assigned to a Call Management System (CMS) user so that the user can access different subsystems in CMS or administer specific elements (splits/skills, trunks, vectors, and so on) of the ACD. Access permissions are specified as read or write permission. Read permission means the CMS user can access and view data (for example, run reports or view the Dictionary subsystem). Write permission means the CMS user can add, modify, or delete data and execute processes.
ACD	See Automatic Call Distribution.
ACD call	A call that queued to a split/skill and was answered by an agent in that split/skill, or a call that queued as a direct agent call and was answered by the agent for whom it was queued.
ACW	See After Call Work.
Adjunct/Switch Applications Interface (ASAI)	An open application interface through which processors and switches can jointly provide services that require applications to initiate, receive, and control calls or make use of switch features. (See Open Application Interface.)
After Call Work (ACW)	An agent state generally representing work related to the preceding ACD call.
API	See Application Programming Interface.
Application Programming Interface (API)	A set of related functions that a computer programmer uses to obtain some kind of service from another piece of software. Programmers of Windows based applications use the Windows API to create windows, draw text on the screen, access files, and perform all other services provided by Windows. Despite the use of the word application in this term, applications might not be the only software using an API; lower-level software components such as network drivers also have APIs, but these components are not “applications” and are not used directly by applications.
ASAI	See Adjunct/Switch Applications Interface.
Automatic Call Distribution (ACD)	A switch feature using software that channels high-volume incoming and outgoing call traffic to agent groups (splits or skills). Also an agent state where the extension is engaged on an ACD call.
Backup	The process of protecting data by writing the contents of the disk to an archive (or tape) that can be removed from the computer environment and stored safely.

Calculation — The abbreviated name (calculation name) for the formula calculation that generates the data for a field in a report.

Call Management System (CMS) — A software product used by business customers that have Avaya telecommunications switches and receive a large volume of telephone calls that are processed through the Automatic Call Distribution (ACD) feature of the switch. The CMS collects call-traffic data, formats management reports, and provides an administrative interface to the ACD feature in the switch.

Call Management System Query Language (CMSQL) — A tool that allows direct queries of the historical database. This tool is the interactive interface typically used to view the Informix database. For CMS purposes, CMSQL is used instead of Informix SQL.

Call Vectoring — A switch feature that provides a highly flexible method for processing ACD calls.

A call vector is a set of instructions that controls the routing of incoming and outgoing calls based on current conditions. Examples of call vector conditions include time of day and the number of calls in queue.

Call Work Code (CWC) — An ACD capability that allows the agent to enter a string of digits during or after the call and send them to CMS for management reporting.

Avaya Supervisor — The Call Management System application for the Microsoft Windows operating environment.

CMS — See Call Management System.

Current interval — Represents the current intrahour interval, which can be 15, 30, or 60 minutes. The current interval is part of the real-time database. CMS starts collecting ACD cumulative data at the beginning of the interval (on the hour, half-hour, or quarter hour) and continues collecting ACD cumulative data until the end of the interval. When the current interval changes, all cumulative data is cleared and CMS begins counting cumulative data again starting from zero. The length of the interval is set in the System Setup: Storage Intervals window and is called the *intrahour interval*.

CWC — See Call Work Code.

Daily data — Interval data that has been converted to a 1-day summary.

CMS database — A group of files that store ACD data according to a specific time frame: current and previous intrahour real-time data and intrahour, daily, weekly, and monthly historical data.

Database item — A name for a specific type of data stored in one of the CMS databases. A database item may store ACD identifiers (split numbers or names, login IDs, VDNs, and so on) or statistical data on ACD performance (number of ACD calls, wait time for calls in queue, current states of individual agents and so on).

Database Management System (DBMS) — The software that manages access to structured data. For example, the Microsoft SQL Server is a database management system. Database management system can also be used generally to include PC database products such as Microsoft Access, as well as any other software that can provide data access services.

CMS database tables	CMS uses these tables to collect, store, and retrieve ACD data. Standard CMS items (database items) are names of columns in the CMS database tables.
DBMS	See Database Management System.
Dictionary	A CMS subsystem that can be used to assign names to various call center elements such as login IDs, splits/skills, trunk groups, VDNs and vectors. These names are displayed on reports, making them easier to interpret. Dictionary also allows customized calculations to be created for use in reports.
Driver manager	A dynamic link library that loads drivers on behalf of an application.
Dynamic link library	A dynamic link library is another name for a driver or a driver manager. A dynamic link library is specific to the DBMS of the data being accessed. For example, an Informix specific dynamic link library will be used to access data in an Informix database, such as the CMS database.
Entity	A generic term that refers to one of the following: Agent, Split/Skill, Trunk, Trunk Group, VDN, or Vector.
Exception	A type of activity in the ACD which falls outside the limits you have defined. An exceptional condition is defined in the CMS Exceptions subsystem, and usually indicates abnormal or unacceptable performance of the ACD (by agents, splits/skills, VDNs, vectors, trunks, or trunk groups).
Historical database	A database that contains intrahour records for up to 62 days, daily records for up to 5 years, and weekly/monthly records for up to 10 years for each CMS table.
Historical reports	Reports that display past ACD data for various CMS tables.
Informix	A relational database management system used to organize CMS historical data.
Informix SQL	A query language tool that is used to extract data from an Informix database.
Intrahour interval	A 15-, 30-, or 60-minute segment of time starting on the hour. An intrahour interval is the basic unit of CMS report time.
LAN	See *Local Area Network*.
Local area network (LAN)	A private interactive communication network that allows computers to communicate over short distances, usually less than one mile, at high data transfer rates from 1 Mbps to as high as 100 Mbps.
Monthly data	Daily data that has been converted to a monthly summary.
Multi-tier driver	OpenLink Multi-Tier drivers feature a Generic Driver installed on the client, a Request Broker and Database Agent installed on the server, and use OpenLink Database Independent Networking. The Multi-Tier drivers are more sophisticated than the Single-Tier drivers as they have been developed with enterprise-wide deployment in mind and feature enhancements in areas such as performance, security and configuration management.
ODBC	See Open Database Connectivity.

Open Database Connectivity (ODBC)	Open Database Connectivity is a standard application programming interface (API) for accessing data in both relational and non-relational databases.
Previous interval	Represents one intrahour interval and is part of the real-time database. At the end of each intrahour interval, the contents of the current intrahour interval are copied to the previous intrahour interval portion of the real-time database.
Read permission	The CMS user with read permission can access and view data (for example, run reports or view the Dictionary subsystem). Read permission is granted from the User Permissions subsystem.
Real-time database	Consists of the current and previous intrahour data on each CMS-measured agent, split/skill, trunk, trunk group, vector, and VDN.
Single-user mode	Only one person can log into CMS. Data continues to be collected if data collection is “on.” This mode is required to change some CMS administration.
SQL	See Structured query language.
Structured query language (SQL)	A language used to interrogate and process data in a relational database (such as Informix).
Switch	A private switching system providing voice-only or voice and data communications services (including access to public and private networks) for a group of terminals within a customer's premises.
Trunk	A telephone line that carries calls between two switches, between a Central Office (CO) and a switch, or between a CO and a phone.
Trunk group	A group of trunks that are assigned the same dialing digits - either a phone number or a Direct Inward Dialed (DID) prefix.
VDN	See Vector Directory Number.
Vector	A list of steps that process calls in a user-defined manner. The steps in a vector can send calls to splits/skills, play announcements and music, disconnect calls, give calls a busy signal, or route calls to other destinations. Calls enter vector processing via VDNs, which may have received calls from assigned trunk groups, from other vectors, or from extensions connected to the switch.
Vector directory number (VDN)	An extension number that enables calls to connect to a vector for processing. A VDN is not assigned an equipment location. It is assigned to a vector. A VDN can connect calls to a vector when the calls arrive over an assigned automatic-in trunk group, dial-repeating (DID) trunk group, or ISDN trunk group. The VDN by itself may be dialed to access the vector from any extension connected to the switch.
Weekly data	Daily data that has been converted to a weekly summary.
Write permission	The CMS user can add, modify, or delete data and execute processes. Write permission is granted from the User Permissions subsystem.

Index

F

G

H

I

L

M

N

O

P

Q

R

S

T

V