CA NSM

Management Database Overview

r 11.2

CA Product References

This document references the following CA components and products:

- CA 7® Workload Automation
- CA Access Control
- CA ADS™ (CA ADS)
- CA Advanced Systems Management (CA ASM)
- CA Cohesion Application Configuration Manager (ACM)
- CA ASM2® Backup and Restore
- CA eHealth Performance Manager
- CA Jobtrac™ Job Management (CA Jobtrac JM Workstation)
- CA NSM
- CA NSM Job Mangement Option (CA NSM JMO)
- CA San Manager
- CA Scheduler® Job Management (CA Scheduler JM)
- CA Security Command Center (CA SCC)
- CA Service Desk
- CA Service Desk Knowledge Tools
- CA Software Delivery
- CA Spectrum® Infrastructure Manager
- CA Virtual Performance Management (CA VPM)

Contact CA

Contact Technical Support

For your convenience, CA provides one site where you can access the information you need for your Home Office, Small Business, and Enterprise CA products. At http://ca.com/support, you can access the following:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Provide Feedback

If you have comments or questions about CA product documentation, you can send a message to techpubs@ca.com.

If you would like to provide feedback about CA product documentation, complete our short customer survey, which is also available on the CA Support website, found at http://ca.com/docs.

Contents

Contents 5

Chapter 1: Introducing the Management Database 9

Managing Information Technology 9
- Reduce Costs 10
- Mitigate Risks 10
- Ensure Infrastructure Availability 11

The MDB Approach 11
- Integrated IT Management Data 12

The MDB Schema 12
Integrated Database Administration 13
The MDB and Relational Databases 13
What Does the MDB Contain? 14

Chapter 2: MDB Deployment 15

Single MDB 15
Multiple MDBs 16
Planning for Availability 17
- Hardware 17
- Failover 17
- Cluster Support 18
- Replication 18
- Database Backup 18

Chapter 3: MDB Administration 19

Security 19
Define New Users 20
Maintenance 20
Schema Upgrades 21
Extend the MDB 21

Chapter 4: Ingres MDB Considerations 23

Windows Paging File 23
Memory Pre-Allocation 23
NFC Unicode 23

MDB Owner Name 24
Set Configuration Parameters 24
Define Ingres User IDs 25
Create a New User to Administer the MDB 26
Enable an Existing User to Administer the MDB 27
Access to MDB Objects 27
Case Sensitivity and Collations 28
How To Maintain an Ingres MDB 28
Monitor File Size 29
Monitor Disk Space 30
Reorganize Tables 31
Collect Optimization Statistics 33
Reorganize the System Catalog 34
Back Up the Ingres MDB 35
Remove the Ingres MDB 36

Chapter 5: Microsoft SQL Server MDB Considerations 39

Connections 39
MDB Owner 39
Tempdb 39
Partitions 40
Security 40
Configuration 40
Environment 40
Case Sensitivity 41
Collations 41
Microsoft SQL Server Maintenance 43
Remove the SQL Server MDB 43

Chapter 6: Oracle MDB Installation and Maintenance 45

Storage Considerations 45
Stripe and Mirror Everything - The S.A.M.E. Method 45
RAID 46
Log Files 46
File Systems 47
Optimum Flexible Architecture (OFA) 47
Database Installation, Configuration, and Deployment 47
Database Block Size 48
Tablespaces 48
Database File Management 49
Global Database Name, System Identifier (SID) 49

Processes Parameter 50
Character Sets 50
Connection Mode 50
Redo Log Files 50
MDB Database Users 51
Security Considerations 51
Archive Logging 51
Undo and Database Backup 52
Monitoring and Administration 54
MDB Monitoring, Performance Management, and Analysis 54
Oracle MDB Defragmentation and Space Recovery 54
MDB Administration 55
Occasional Exporting of data from Oracle 55
MDB Backup and Recovery 55
Remove the Oracle MDB 56

Chapter 7: Unicenter NSM and the MDB 57

WorldView 57
Security 57
Data Capacity Estimation 57
Topology or Implementation Scenarios 57
Tuning and Performance Requirements 58
Backup, Administration, and Maintenance Requirements 58
Integration Considerations 58
Enterprise Management 59
Security 59
Data Capacity 61
Topology or Implementation Scenarios 62
Schema and Reporting Information 62
Tuning and Performance 72
Backup, Administration, and Maintenance Requirements 72
Integration Considerations 72
High Availability Considerations 73
Performance Management 74
Security 74
Data Capacity Estimation 76
Topology or Implementation Scenarios 78
Schema and Reporting Information 82
Tuning and Performance Requirements 89
Backup, Administration, and Maintenance Requirements 91
Integration Considerations 95
Unicenter Management Portal 98

Access to Unicenter MP Tables in the MDB 98
Access to Enterprise Management Data in the MDB 98
JDBC Usage 99
Security 99
Data Capacity Estimation 100
Topology or Implementation Scenarios 101
Asset Registration 101
Security 102

Index **103**

Chapter 1: Introducing the Management Database

CA Management Database (MDB) r1 is a common enterprise data repository that integrates CA product suites. The MDB provides a unified database schema for the management data stored by all CA products (distributed). Use of the MDB with CA products enables full integration for managing your IT infrastructure.

This Overview contains information you need to know about deploying single and multiple MDBs, administration, such as defining new users, and maintenance activities, such as database backup.

CA MDB r1 supports multiple database platforms, including Ingres, SQL Server, and Oracle.

This section contains the following topics:

Managing Information Technology (see page 9)
The MDB Approach (see page 11)
The MDB Schema (see page 12)
Integrated Database Administration (see page 13)
The MDB and Relational Databases (see page 13)
What Does the MDB Contain? (see page 14)

Managing Information Technology

CIOs are challenged to deliver a highly flexible infrastructure with a high degree of automation. This challenge is frequently referred to as *on-demand computing* or *utility computing*. Delivering on-demand computing is essentially an IT management problem. The CIO needs to have advanced management solutions that help to meet three key goals:

- Reduce costs
- Mitigate risks
- Ensure infrastructure availability

Reduce Costs

To become a fully aligned business partner for an enterprise, IT must reduce costs. Total implementation costs are reduced significantly when products do the following:

- Make deployment and administration easier
- Enable better decision making
- Provide the means to automate business processes

IT management software should provide simple and flexible options that can deploy products one module at a time or by functional suite. When data sources are shared, fewer systems need to be administered, managed and monitored; and fewer staff resources are required. Increased visibility and timely access to integrated IT management data help managers make the best possible decisions.

Business process automation may eliminate the costs of manual processes. However, full automation requires complete information, which may be provided best when products share a single data source. For example, a CIO weighing expensive hardware or software purchases may obtain utilization rates for existing hardware and software--and review current lease agreements and contracts--prior to making a purchasing decision.

Mitigate Risks

Enterprises today face increasing risks and exposures in the areas of platform and application management, event management, business processes, records retention, and security. A unified data strategy enables the CIO and IT management staff to ensure that the IT infrastructure is fully secured and protected, and adheres to proven compliance and governance guidelines. Some of the primary approaches to minimizing such risks include:

- Applying consistent security
- Ensuring availability
- Implementing open standards

Managing risks is more difficult when IT environments include multiple heterogeneous databases and proprietary file formats. A common data source and data definitions provide the trace and audit capabilities that are required in complex regulatory environments. Data storage should be based on a relational database management system that provides transaction support, recovery, clustering and high availability. Extensible data definitions provide a way to manage additional proprietary data.

Software products for IT management need to be secure, scalable, robust, and built on mature, proven technologies and architectures. Support for open standards and interfaces, and implementation of best practices such as the Information Technology Infrastructure Library (ITIL), helps mitigate risks.

Ensure Infrastructure Availability

IT downtime means money lost to the business. Keeping the IT infrastructure running and available requires complete information about critical business systems in real time, not from secondary data sources that are extracted and reintegrated later. At the same time, real-time information requirements must not limit the deployment options for IT management products. Organizations should be able to deploy IT management products using either a single data source or multiple data sources.

If multiple sources of data are used, a unified view of information is necessary to ensure an available infrastructure. Highly available systems take advantage of clustered servers so that any failure automatically transitions work to another server in the cluster. Redundancy ensures that business may continue even when some network nodes are lost. IT management data sources hold time-critical information that ensures the availability of the infrastructure. These data sources themselves must be highly available, redundant, and clustered to minimize the business impact of hardware or software failures.

The MDB Approach

The Management Database (MDB) is a common enterprise data repository that integrates CA product suites. The MDB provides a unified database schema for the management data stored by all CA products (mainframe and distributed). Use of the MDB with CA products enables full integration for managing your IT infrastructure.

The MDB integrates management data from all IT disciplines and CA products. Customers may extend the MDB Schema to include additional IT management data from non-CA software products and tools.

Note: The MDB Schema is available from http://ca.com/support. Search the Technical Support knowledge base for more information.

The primary advantages of the MDB approach are:

- Integrated IT management data
- Integrated database administration

Integrated IT Management Data

A unified database schema for IT management data provides increased visibility into the underlying IT organization, including storage, performance, hardware, and software information. This visibility improves IT decision-making and helps to reduce costs and increase utilization of the infrastructure. Using a unified database schema enables fully-informed decisions about hardware, software and storage purchases, as well as provisioning, scheduling, data protection, and more.

The unified MDB schema enables integration of products without additional programming effort. Integration methods that do not include a single database schema require point-to-point programming efforts to access data from disparate sources. Data integration enables rapid development of new product features, because there is no need for programming interfaces to make disparate data available. Without an MDB, data is stored in multiple locations and schemas, making it difficult to integrate and to create new features that take advantage of data relationships.

Example

Without an MDB, it is difficult to determine whether a server located through a discovery process has been backed up, since the data for discovery and storage are maintained by separate products, in separate databases, and on separate servers. With an MDB, the data is always in one integrated schema. Storage and inventory management (discovery) are more tightly integrated. It is also possible to provide this feature to customers.

The MDB Schema

The MDB schema provides a unified and extensible model for enterprise IT management. It contains both common tables and product-specific tables that were previously implemented in separate product databases. The MDB serves as the enterprise database for all CA products and acts as the primary reference point. This allows you to write single queries that retrieve data from tables across different products.

The MDB schema includes the database objects used by CA products and their components. These include tables, columns, views, and procedures. The MDB manages operational and transaction data, as well as the analytical data used for intelligence and data mining. Depending on your organization's business needs and structure, you may use a single MDB or multiple MDBs; both approaches utilize the same schema.

Integrated Database Administration

A unified MDB schema means only one database to administer rather than many which makes the management data easier and less expensive to manage. On demand reporting, data mining, and other (non-product) operations that use the MDB become easier. The MDB uses a relational database management system, so its data is available and accessible through the well-established, easy to use Structured Query Language (SQL). In addition to ease of data access, relational database management systems provide a high degree of reliability, availability, and scalability as well as good performance and cluster support. Since the MDB uses a relational database management system, it inherits these advantages.

The MDB and Relational Databases

To store critical operational and IT data, a management database requires a commercial-grade relational database management system (RDBMS). All CA products that use MDB r1 come pre-packaged with a full-featured open source Ingres RDBMS at no additional cost.

What Does the MDB Contain?

The MDB contains complete, consistent, and manageable definitions for all key IT management data. These definitions are shared by CA product solutions right out of the box, and they also may be extended to other products and third-party tools. MDB information supports CA solutions in the following categories of IT management:

- Assets and inventory
- Storage
- Security
- Operations
- Services
- Job scheduling
- Servers and desktops
- Databases
- Application life cycle

This complete and consistent support for the full spectrum of IT management data provides a foundation for the integration of CA management solutions.

Chapter 2: MDB Deployment

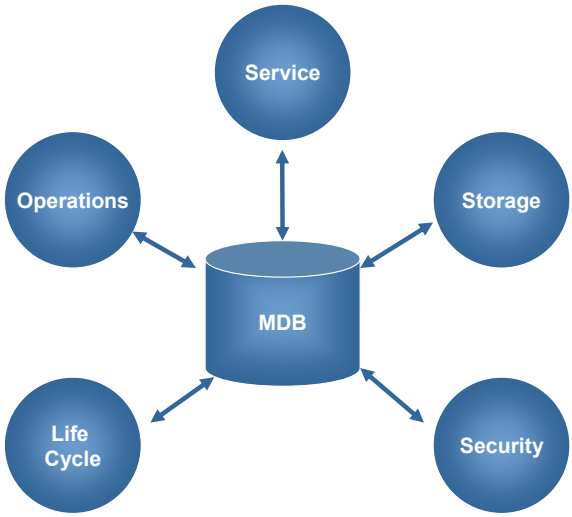
This section contains the following topics:

Single MDB (see page 15)
Multiple MDBs (see page 16)
Planning for Availability (see page 17)

Single MDB

Deployment of a single global MDB to store data for all products is the simplest to implement. This deployment option is shown in the following illustration. In general, this implementation costs the least and is the easiest to manage.

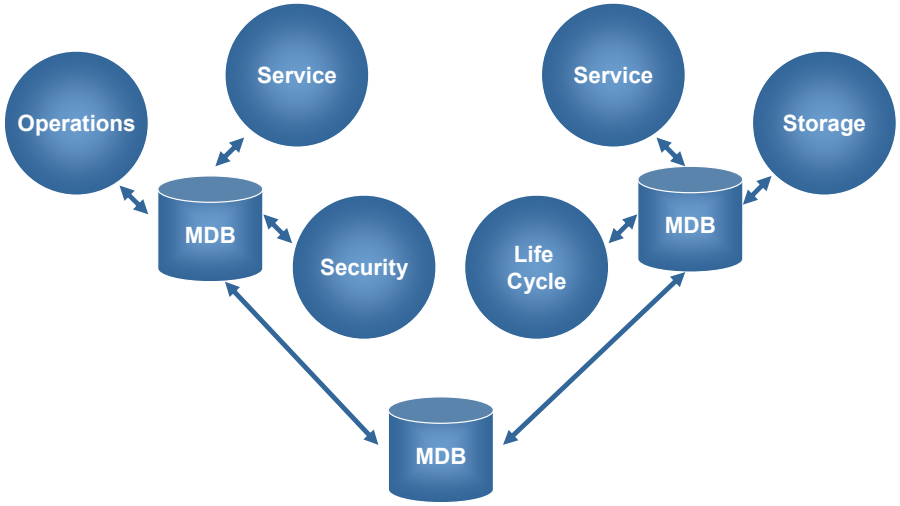
A single integrated view of the underlying IT infrastructure is available with no additional processing required. In this deployment, administration requirements, including establishing security and data availability, occur once for the single MDB.



Multiple MDBs

Multiple MDB instances may be needed for reasons of size, legal requirements, organizational structure, or geography. In addition, organizations may need to deploy one or more MDBs in IT environments that include non-CA software solutions.

The MDB enables a federated approach that uses multiple databases or data instances. To gain a single integrated view, data must be retrieved or accessed from multiple data instances as shown in the following illustration. Supporting multiple MDBs may require additional planning and administration.



In this scenario, one or more products access different MDBs. Each of the following approaches provides a single integrated view (for reporting purposes) of the underlying IT infrastructure:

- **Distributed query support**, provided by the relational database being used, enables a federated approach. With distributed query, multiple MDBs, each with their own management information, are segmented and implemented by function/geography and organization. Distributed queries allow data from different databases (MDBs) to appear as if it were from one database.
- **Replication**, available by the relational database being used and built into some of CA's management products directly, enables a federated approach. With replication, each function/geography or organization uses its own MDB for management information. Data from each MDB is replicated to a central database server with an MDB that provides a single, integrated view of the company's IT infrastructure. In a replicated approach, the central database server might also serve as an MDB supporting one or more CA products.

Planning for Availability

The data stored in the MDB is critical and is required for the operation of CA solutions that manage your IT infrastructure. It is important that you plan for availability in case of any eventuality that would put your IT infrastructure and the functionality of critical business processes at risk. Without such planning, the MDB could become a single point of failure. You may use several features of the underlying relational database management software to help protect against failures. These include the use of failover capabilities, cluster support, custom replication, and the performance of database backups on a regular basis.

Planning for availability must be an integral part of the implementation of your CA solutions.

Hardware

A planned and designed Management Database (MDB) environment provides a quality, high performing system with extendable growth capability. Lack of or incorrect architecture decisions may cause poor performance, lack of scaleability, poor reliability, or poor availability. To prevent such problems, the following key areas should be considered prior to MDB deployment:

- Processor Technology (in database servers a minimum of two is preferred)
- Storage and I/O subsystems (in database servers a minimum of two I/O channels is recommend)
- Network Bandwidth and Availability (in database servers utilize backbone networks for linking servers to each other, where possible separate out client/server traffic from server/server traffic)
- Dedicated or shared server MDB deployment

Failover

Failover support enables the relational database management system to operate on a cluster as a distributed application, providing transparent access to an MDB that resides on shared storage devices. If there is a database or hardware failure on one of the nodes, processing continues uninterrupted using the remaining nodes.

Cluster Support

Active cluster support allows a system's workload to be spread across the nodes of a cluster. If one node fails, the remaining nodes continue processing to avoid loss of service. Relational database management systems exploit this capability and automatically balance the load across the nodes in the cluster.

Replication

Replication capabilities enable you to move data from one database instance to another. By moving data to a secondary database instance on a regular basis, it is possible to recover from a disk crash, because processing may be resumed on a secondary database instance. The secondary database instance may be different from the original based on the intervals for which replication occurs.

Note: This is a custom solution and must be built for the specific CA products in use at your site.

Database Backup

The MDB must be backed up at regular intervals. In the event that disaster recovery is needed or a failure occurs, the MDB may be restored.

Chapter 3: MDB Administration

This section contains the following topics:

Security (see page 19)
Define New Users (see page 20)
Maintenance (see page 20)
Schema Upgrades (see page 21)
Extend the MDB (see page 21)

Security

All CA products share the database objects in the MDB. This provides a consistent data architecture within which management information uses a single standard for access control and administration to ensure integrity and availability. Enforcement of security rules for access to the MDB is provided by the underlying relational database management system (RDBMS).

The MDB contains the database objects for all CA products. Database objects include tables, views, procedures, and rules. Typically, a CA product utilizes a specific set of database objects (such as tables, indexes, views, constraints and so forth) to expose information to the user. While there may be many shared objects between products, the MDB security model ensures that the user of a CA product may only access the database objects for that product. For example, a customer using CA's Unicenter Service Desk product may only access tables used by that product and may not access tables for another product.

The MDB security model restricts access to database objects used by a product by using underlying RDBMS user groups. A user group is created for each product (or for each product application component). Users of a product, or users created for the product's component, are assigned to a user group as their default. Any number of users may be associated with a user group. Through grouping the database objects used by a particular product and associating database users with a user group, the product is restricted in the data, procedures, and rules that it may access.

The MDB is created with many user groups already defined. Products specify these groups when they connect to the database. If you choose to use external tools such as report writers to access MDB data, you may limit the report writer user's MDB access by using these user groups as part of the database connection information.

Define New Users

User definitions are required for access to the MDB. Some CA products create users automatically during installation; others require that you create users manually. Consult specific CA product documentation to determine what is required.

If you use an external tool such as a report writer for creating new applications to access MDB data, you may need to create new users for that purpose.

If you access database objects used by an existing CA product, any new database users you create must be added to an existing database access group for the CA product.

If you access new database objects, or database objects from several different products, you must create a new user group for the database and add the user to that group.

Maintenance

Regardless of database implementation, regular maintenance activities are required to ensure optimal performance and responsiveness for an MDB.

Maintenance activities such as database backup, system catalog reorganization, and file and disk space monitoring for optimization are essential for best performance. These activities must occur regardless of the number or variety of CA products that access an instance of the MDB. In addition, table reorganization and statistics are also useful for performance purposes.

Most databases provide for maintenance using command line utilities and SQL commands. These features allow sites to use command files and a job scheduler to automate the maintenance process.

More Information

For more information about specific maintenance utilities and commands, consult your database documentation.

Schema Upgrades

The MDB schema is upgraded when patches are applied or when an MDB is already installed and a newer MDB release is installed by another CA product.

Prior to any action that will upgrade the MDB schema, the database should be backed up by a database administrator. The backup provides the ability to restore in case an error occurs during the upgrade process.

Extend the MDB

Customers and third parties may extend the MDB schema to include additional IT management data. New database objects (for example, tables, views, and procedures) can be created that incorporate the IT management data from CA products and also provide additional data, features, and capabilities.

If a CA product already provides functionality for extending its data model, then that product's own functionality should be used. These CA products provide support for identifying MDB extensions and upgrading them for subsequent product releases.

The standard method for extending the MDB is to add new database objects (for example, tables, views, and procedures) as needed.

Important! To ensure the integrity of software products and data, customers must not change any standard database objects that were provided with the base MDB or add new objects to the "mdbadmin" domain as this may prohibit successful future upgrades if a duplicate object name is found.

For details about creating a schema, owner IDs, and database objects, consult your database documentation.

Chapter 4: Ingres MDB Considerations

This section contains the following topics:

Windows Paging File (see page 23)
Memory Pre-Allocation (see page 23)
NFC Unicode (see page 23)
MDB Owner Name (see page 24)
Set Configuration Parameters (see page 24)
Define Ingres User IDs (see page 25)
Access to MDB Objects (see page 27)
Case Sensitivity and Collations (see page 28)
How To Maintain an Ingres MDB (see page 28)

Windows Paging File

When an Ingres MDB is installed running under Microsoft Windows, you should have a minimum Windows paging file size of two (2) gigabytes or one and a half times the RAM in the computer, whichever is greater.

Memory Pre-Allocation

To create an MDB, Ingres pre-allocates most of the resources that it requires. It often uses hundreds of megabytes of virtual memory and several hundred thousand handles. The exact amount of resources depends upon the number and sizes of the page caches used. The default (medium) MDB configuration uses approximately one (1) gigabyte of virtual memory and under Windows 350,000 to 500,000 handles.

NFC Unicode

For Ingres, the MDB is created as a Unicode Normalization Form C (NFC) database. This allows Unicode data to be stored and manipulated by defining columns as Unicode data types (that is, nchar, nvarchar, and long nvarchar). If a Unicode collation name is not specified, the Unicode database is created with the default collation sequence "udefault." The Unicode database created in this manner (createdb -n) uses NFC for normalization of Unicode strings for processing and storage. NFC results from the canonical decomposition of a Unicode string, followed by the replacement of all decomposed sequences by primary composites, where possible.

For more information, see section A1.2 of http://www.unicode.org.

MDB Owner Name

The owner of any MDB database (the MDB DBA) is *mdbadmin*. When connected to an Ingres MDB, this user name is returned by the following Ingres SQL statement:

```
SELECT dbmsinfo('dba')
```

Set Configuration Parameters

To simplify Ingres configuration for the CA products that use the Management Database, use the *setupmdb* utility to set the II_MDB_SIZE parameter to one of the following values:

- small
- medium
- large

Important! The *setupmdb* utility works with Windows authentication only. It does not work with Microsoft SQL Server authentication.

When you install a CA product, one of these settings is established automatically. Do not make any changes unless instructed to do so by the CA product documentation or CA technical support.

If you install a second CA product that uses an existing Management Database, you may use the configuration parameter utility to increase the Ingres configuration setting. The utility only allows increasing the setting. For example, you may go from small to medium, but not from medium to small.

The table below provides guidelines for specifying the Ingres MDB requirements and the machine sizes required to support them.

Size	Connections	Number of CPUs (suggested minimum)	Memory (required)
Small	Up to 100	1	2 Gigabytes
Medium	Up to 500	1	2 Gigabytes
Large	Up to 1000	2	4 Gigabytes

Note: The Large data model is not supported on Microsoft Windows 2000 systems.

II_MDB_SIZE should be set to one of the values in the previous table. You must ensure that the server hosting the Management Database has sufficient memory and number of processors. The configuration utility must be run on the server where Ingres is installed and you must shut Ingres down before running the configuration utility. For Ingres shutdown instructions, see the *Ingres Getting Started* guide. The new setting takes effect the next time Ingres is started.

Example

The *setupmdb* utility may be found on Windows in the <II_SYSTEM>\ingres\MDB subdirectory or on Linux in $II_SYSTEM/ingres/mdb.

```
setupmdb -II_MDB_SIZE=medium -reinitcfg
```

In this example, the reinitcfg flag re-initializes the II_MDB_SIZE configuration setting to medium. Since the configuration size may only be increased, this example may only change a small to medium. If the previous setting was medium or larger, this example has no effect.

Ingres configuration settings are stored in the <II_SYSTEM>\ingres\files\config.dat (Windows) file or $II_SYSTEM/ingres/files/config.dat (Linux) file. A backup copy of the previous file is saved.

Define Ingres User IDs

To connect to an Ingres MDB, you must possess a valid Ingres user ID and an associated valid user ID on the operating system that hosts the Ingres database server.

The Management Database is created with all of the pre-defined Ingres user IDs required by CA products to access the database. A user ID is not accessible until an operating system user ID with the same user name is created on the database server with the MDB. The operating system user IDs are not created until the product is installed. If you have one CA product accessing the Management Database, the database server will have defined only the operating system user IDs required for that product. Ingres user IDs are defined without passwords, because Ingres authenticates the operating system user definition before checking whether the user is defined to Ingres.

Some product installations create operating system users when the product is installed. In such cases, product installation should prompt for the password to be used in creating the OS user ID. Whenever possible, it should refrain from creating the password automatically, or (if created automatically) it should be a strong password.

Important! The Ingres user ID mdbadmin is the administrative owner of all MDB database objects. This user name should not have a corresponding operating system user ID; if it did, it would expose the Management Database to administration and access from remote sources. This user name should not be used by any product to access objects in the Management Database, as it is for CA internal use only. If such an operating system user is found, it should be investigated and ultimately removed.

Create a New User to Administer the MDB

A new Ingres user may be created who is able to impersonate the MDB administrative user (mdbadmin). This impersonation is valid for all Ingres commands that support the specification of a user, typically those using a -u parameter. The Ingres SQL command is a good example of this.

Other commands such as the Ingres checkpoint command (ckpdb) also support the -u parameter.

To create an administrative user

1. Create an operating system user ID for the Ingres user to be defined. The OS user ID must conform to Ingres user naming conventions.
2. Create an Ingres user with the following default privileges (createdb, security, operator, maintain_locations, maintain_users). This may be done by using accessdb, the SQL command or Visual DBA.

   To create a user, use the following SQL commands:

   ```
   create user username with default
   privileges=(createdb,security,operator,maintain_locations,maintain_users) \g

   grant db_admin on database mdb to user username \g
   ```

   These statements need to be issued from the database server by the OS user that installed Ingres/MDB. In addition the command line for the SQL command is:

   ```
   sql iidbdb -u$ingres
   ```

   Note: The connection in this case is made to the Ingres installation database, not the MDB.

Enable an Existing User to Administer the MDB

An existing user may be given the ability to administer the MDB by impersonating the MDB administrative user (mdbadmin). This is accomplished by using the following SQL commands:

```
alter user username with
default_privileges=(createdb,security,operator,maintain_locations,maintain_users)
\g

grant db_admin on database mdb to user username \g
```

These statements need to be issued from the database server by the operating system user that installed the Ingres MDB. In addition, the command line for the SQL command is:

```
sql iidbdb –u$ingres
```

Note: The Ingres Visual DBA tool may be used to create users, alter users and add grants.

For information about creating users with administrative capabilities, see the *Ingres r3 Database Administrator Guide.*

For information about the create user, alter user, and grant commands, see the *Ingres r3 SQL Reference.*

Access to MDB Objects

When a new Ingres user ID is created, that user does not automatically have access to MDB objects. One method of granting access to a specific group of database objects is to assign a predefined user group. For example, a newly created Ingres user ID could be assigned to a Service Desk user group to limit access only to those database objects permitted to the Service Desk group and to no other objects.

If an Ingres user ID is created to access database objects for reporting purposes, the new user must be granted access to the appropriate database objects (tables and views); this is accomplished using Ingres grant statements.

Case Sensitivity and Collations

An Ingres MDB is created as a Unicode Normal Form C database and by default all columns are defined as case sensitive. Unicode Table columns that require case-insensitive sorting or searching may be assigned a case-insensitive collation sequence when the MDB is created.

How To Maintain an Ingres MDB

The following sections provide maintenance recommendations for an Ingres-based MDB.

The following maintenance activities are recommended:

- Monitoring file size
- Monitoring disk space
- Reorganizing tables
- Collecting statistics
- Reorganizing the system catalog
- Backing up the database

Monitoring file size and disk space, reorganizing system catalogs, and backing up the MDB database are generic maintenance activities. These activities need to occur regardless of the number or variety of CA products accessing an instance of the MDB. Table reorganization and statistics gathering are also necessary for performance purposes.

Command line utilities and SQL commands are used to perform these maintenance functions. The sections that follow describe the names of these utilities and SQL statements used in each function. Using a job scheduler, sites may create command files that automate the housekeeping processes.

Note: Ingres provides a Visual DBA utility for administering Ingres databases, including MDBs. Most functions that are available from command line utilities and SQL statements are also available from the Ingres Visual DBA utility. For more information, see your Ingres documentation.

More Information

Ingres r3 Database Administrator Guide, "Improving Database and Query Performance" (Database Maintenance).

Monitor File Size

Ingres creates an operating system file for each table and index. As the number of rows in a table increases, so does the file size on the disk, up to the maximum file size for the operating system. Two gigabytes is the maximum file size on most but not all Windows 32-bit operating systems. However, other operating systems and file systems may have different limits that should be considered when monitoring.

The need to monitor file size is required for operating systems where the maximum file size is 2 gigabytes per Ingres database file. This is because when reaching the maximum size, it is no longer possible to put more rows into the table until it is split (partitioned) across two or more locations (the maximum number of locations is 255). On operating systems that support larger file sizes, the need to monitor is reduced.

Periodically check the size of the files at the MDB database location(s). When the limit is near, you may either extend the tables across more locations, or partition the tables using the following commands:

Note: For information about how to determine the table name for an MDB database file, see the *Ingres r3 Database Administrator Guide*, Chapter 8, "Maintaining Databases" (Translating File Names into Table Names).

- **accessdb**

  The *accessdb* command is an Ingres command to create a location. You may specify the purpose of the location: databases, checkpoints, journals, dumps, or work.

- **extenddb**

  The *extenddb* command is an Ingres command that extends a database to use an already created location. In order to extend a table across more than one location, the new locations must exist for use as database locations. The SQL MODIFY statement may then be used to enable the table to increase in size by using the additional file location.

Table partitioning is a new feature of Ingres r3 that allows a single table to be partitioned across multiple locations. Partitioning occurs based on the pre-defined value(s) of column(s) in the table. Each partition of the table may be stored in a file in a different location. The maximum size of each file is limited by the underlying file system. Table partitioning may also occur when a table reaches the file size limit. Rather than partitioning tables based on what rows go where, you may partition based on size to allow potentially better querying. A table partitioned into *n* locations might be additionally partitioned into *n+1* locations.

All of the above may also be achieved in native SQL or with the Ingres Visual DBA tool.

More Information

Ingres r3 Database Administrator Guide, Chapter 3, "Alternate Locations" (Extending and Unextending a Database; Relocating Database Files)

Ingres r3 Command Reference, Chapter 3, "Commands" (extenddb)

Ingres r3 SQL Reference, Chapter 8, "SQL Statements" (Modify)

Monitor Disk Space

The disk space available to Ingres is the amount of disk space available on the storage media where the Ingres software, the transaction log file(s), data, checkpoints, journals, dumps, and work directories are located. It is important to have sufficient disk space available to accommodate the MDB. Ingres imposes no limits on how much disk space a location may use. However, the operating system or the disk (virtual or real) may impose space limitations.

To operate properly, a database also requires work location space. This space is required for sorting and for temporary and transient files within Ingres. For example, sorting, modifying a table, or using a "session temporary table". Therefore, there must be sufficient temporary space to service the needs of applications and maintenance for the Ingres instance.

Recommended Actions

Disk space should be monitored at least once per day. Operating system commands are available to monitor available disk space.

If available disk space becomes low, it is necessary to expand the available disk space by doing one of the following:

- Adding new disk space
- Adding a new location and extending or relocating tables into the new location

More Information

Ingres r3 Database Administrator Guide, Chapter 3, "Alternate Locations"

Ingres r3 Database Administrator Guide, Chapter 16, "Calculating Disk Space"

Reorganize Tables

Database tables must be re-organized to clean up indexes and other structures. Frequent updates and inserts to tables may create overflow pages and secondary indexes that contribute to inefficient search operations.

Available Functions

To reorganize indexes and tables, you can use the SQL *modify* statement or the Ingres *usermod* command. An SQL script may be created to reorganize tables using SQL modify, or usermod can be run from the command line. The usermod command modifies the user-defined tables of a database to their original storage structure and recreates any secondary indexes.

The SQL *modify* statement destroys indexes and does not recreate them unless the indexes were originally created "with persistence". For indexes that are not created "with persistence", it is necessary to create the index after modify has completed.

Tables within MDB have different characteristics, such as:

- Numbers of pages
- Numbers of indexes
- Rates of growth
- Rates of change

Each table will have a different modification cycle. Some must be modified once a year; others must be modified as frequently as once a day. Consult your CA product documentation to determine product specific requirements.

The following approaches can be used to reorganize the database:

- Run usermod once a week against all the tables in the database.
- For tables which have a Btree structure, running "modify to merge" on a short cycle will recognize the table's index pages, which is quicker than a complete modify, run on a longer cycle.
- If a table does not need to be modified but an index needs to be reorganized either the index may be dropped and recreated or the index can be modified; this may be quicker than reorganizing the table and all the indexes.
- Tables that require modification tend to be those that grow quickly or have frequent updates and inserts.

  Monitoring table sizes on a regular basis is one way to determine tables requiring modification. This may be done by monitoring the size of files that hold tables in the MDB.
- Another indication of tables that require modification is the number of pages holding overflow chains.

  Overflow chains slow performance. To determine that tables require modification due to a large number of overflow pages, check the number of overflow pages for the MDB tables and secondary indexes.

To monitor overflow in the Visual DBA tool (VDBA), select a table or secondary index in the Database Object Manager window, and click the Pages tab. Use the legend to interpret the information displayed. If the number of overflow pages is greater than 10-15% of the number of data pages, you should consider modifying the table.

More Information

Ingres r3 Database Administrator Guide, Chapter 4, "Managing Tables and Views" (Reorganizing a Table)

Ingres r3 Database Administrator Guide, Chapter 17, "Improving Database and Query Performance"

Ingres r3 SQL Reference, Chapter 8, "SQL Statements" (Modify)

Ingres r3 Command Reference Guide, Chapter 3, "Commands" (usermod)

Collect Optimization Statistics

For best performance, an MDB should be optimized to match the data distribution patterns of a product's actual database changes. For this purpose, Ingres provides a Query Optimizer Facility (within the OPF) that uses statistical and histogram data about the table columns that are used in queries. When you optimize a database, data distribution statistics are collected that help queries run more quickly and use fewer system resources. Collecting these statistics increases the likelihood of optimal query plans in the future.

optimizedb

The *optimizedb* command creates statistic and histogram information that the Ingres query optimizer uses.

The frequency of running optimizedb on a table column depends on the rate of change of:

- Histogram data within the column
- Range of values within the column

Within a multi-column table, each column may have a different frequency need for running optimizedb. There may be columns in a table on which it is unnecessary to run optimizedb, as these columns are never used in the WHERE clause of an SQL statement. The number of rows in the table may be so large that using the optimizedb option of sampling against a column is as effective as running optimizedb against every column of every row within the table.

optimizedb may run while data is changing in the table. It is only upon completion of gathering the statistics from the table that exclusive locks are held on the system catalogs (iihistogram and iistatistics). These exclusive locks are held for the time that it takes to update the system catalogs.

optimizedb adds new data to the system catalogs and therefore after it has been run, the system catalog should be reorganized. For more information, see How To Reorganize the System Catalog.

Running optimizedb is an important maintenance activity for the MDB. Frequency is dependent on each CA product. Consult your product's documentation for information about its MDB database optimization requirements.

More Information

Ingres r3 Database Administrator Guide, Chapter 13, "Using the Query Optimizer"

Ingres r3 Command Reference Guide, Chapter 3, "Commands" (optimizedb)

Reorganize the System Catalog

In an active database, users must create and modify tables, views and other database objects. To reduce data page overflow and locking contention, you can do regular system catalog modification.

sysmod

sysmod is an Ingres utility that modifies the system catalog in an Ingres database. This utility is fast and efficient.

sysmod should be run after database creation, after significant changes to database schema (for example, creating and dropping tables), and after running optimizedb.

sysmod permits a list of system catalogs to be specified (for example, iihistogram iistatistics). If this option is used, only the listed catalogs are reorganized.

To use sysmod, Ingres needs to be running; however, there must be no sessions connected to the database.

In summary, if table reorganization, collection of optimization statistics, and reorganization of the system catalog are done in a single session, you should perform the steps in the following order:

- usermod
- optimizedb
- sysmod

More Information

Ingres r3 Command Reference Guide, Chapter 3, "Commands" (sysmod)

Back Up the Ingres MDB

An Ingres MDB should be backed up using the Ingres checkpoint utility (ckpdb). The checkpoint utility has the following features:

- Runs in online (with active connected sessions) or offline (with no connected sessions) mode
- Runs against the database, a table, or a list of tables

The following approaches may be used for backing up the MDB:

- At a minimum, the MDB should be backed up at the end of a maintenance cycle (after all other maintenance activities are completed). This means that recovering the database requires applying the journals, but not reapplying maintenance.
- Perform a backup just before and then again immediately after maintenance. This provides a recovery point at the start of maintenance, in case maintenance fails for some reason.
- Perform a backup before maintenance, after maintenance, and at suitable points between maintenance cycles. The schedule should always allow a minimum time period for recovery based on the size of the checkpoint and the size of the journal files.

More Information

Ingres r3 Database Administrator Guide, Chapter 15, "Backup and Recovery"

Journaling

Journaling is the Ingres method of capturing changes that have been made to the database. You may switch journaling on or off for all tables in a database or for specific tables only. Having a checkpoint and journals for a database lets you recover the database from a catastrophic failure (for instance, a disk crash). The time required to recover a database depends on the following:

- Size of the database
- Size of the journal files
- Time spent replacing the database files onto disk

Specifying the journaling flag (+j/-j) forces ckpdb to run offline, and therefore no connections are allowed.

ckpdb should be run in offline mode against the database with the +j flag (journaling on) after the following processes have occurred:

- MDB is created
- Metadata is created
- Data is loaded

Re-running checkpoints does not require the inclusion of the +j/-j. The current setting applies to all future checkpoints until this value is changed. That is, journaling stays on until it is turned off, and it stays off until it is turned on.

Important! When the MDB is created, journaling is automatically turned on. It should **not** be turned off.

Remove the Ingres MDB

The MDB should not be removed except in extreme cases.

Before removing an MDB, make sure the data stored in the database is no longer needed and that all products that were accessing the database have been uninstalled or configured to work with another MDB. Once an MDB is removed the data is lost.

Note: You should make a backup of the MDB before removing it, especially if more than one product has shared it.

To remove an Ingres MDB

1. On the Ingres server, destroy the database using the following command-line command:

   ```
   destroydb mdb –umdbadmin
   ```

2. Issue the following commands to remove MDB information from the Ingres configuration file:

   ```
   iiremres -v ii.%HOST%.mdb.mdb_dbname
   iiremres -v ii.%HOST%.mdb.mdb.version.major
   iiremres -v ii.%HOST%.mdb.mdb.version.minor
   iiremres -v ii.%HOST%.mdb.mdb.version.build
   iiremres -v ii.%HOST%.mdb.mdb.description
   iiremres -v ii.%HOST%.mdb.mdb.size
   ```

3. Remove the signature file. This file is located in the mdb subdirectory of the Ingres files directory. It is named:

   ```
   mdb.signature.txt
   ```

4. Remove OS users from the database server that were created by CA products for use with the MDB.

Chapter 5: Microsoft SQL Server MDB Considerations

This section contains the following topics:

Connections (see page 39)
MDB Owner (see page 39)
Tempdb (see page 39)
Partitions (see page 40)
Security (see page 40)
Configuration (see page 40)
Environment (see page 40)
Case Sensitivity (see page 41)
Collations (see page 41)
Microsoft SQL Server Maintenance (see page 43)
Remove the SQL Server MDB (see page 43)

Connections

Microsoft SQL Server supports setting a limit on the number of concurrent user connections.

Install Microsoft SQL Server to use unlimited concurrent user connections.

For best performance with more than 255 concurrent connections, increase the "maximum worker threads" setting.

MDB Owner

The MDB is owned by the Microsoft Windows user who ran the MDB installation.

Tempdb

Microsoft SQL Server uses the tempdb database as a scratch area for MDB temporary tables, sorting, subqueries, and so forth.

The size of the tempdb database should be increased based on available disk space and expected usage. Microsoft SQL Server adjusts the size incrementally over time, but each adjustment causes a performance hit.

Partitions

Microsoft SQL Server supports partitioning of data files among multiple disk drives to ensure recovery and improved performance.

For best results:

- The Microsoft SQL Server MDB data and log directories should reside on separate disk drives for better recovery operations.
- The tempdb database should reside in a separate partition for improved performance.

Security

Microsoft SQL Server supports Windows and Microsoft SQL Server authentication.

Microsoft SQL Server must be installed using mixed mode (both Microsoft SQL Server and Windows authentication).

Configuration

Microsoft SQL Server supports various configuration options. The only configuration option set during the MDB installation is recursive triggers.

The Microsoft SQL Server configuration options should not be modified from their default values.

For More Information, see *Microsoft SQL Server Books Online* (**sp_configure**).

Environment

Microsoft SQL Server can be installed on the same server as other applications but consider the following:

- Installing Microsoft SQL Server on a dedicated computer may improve performance.
- Depending on the number of processors, memory, and disks on your computer, a dedicated computer may not be required. Network latency between the application and the SQL Server may in fact degrade application performance.

Case Sensitivity

The SQL Server instance that hosts the MDB can either be case sensitive or case insensitive. The MDB is created as a case insensitive database.

If the SQL Server instance is defined as case sensitive, all connection requests to the MDB must specify the database name (mdb) in lower case. If the database instance is defined as case insensitive it does not matter what case is used in the database name when connecting to the MDB.

Collations

SQL Server collations determine how character data is compared and sorted. Collations are composed of a language designator and a sorting style. The two different sorting styles are binary and composite.

Binary is a sorting style that sorts data by binary value. Each character, upper and lower case, has a different binary value. The binary value of these characters, however, may not match the dictionary order for that language. Collations that use the binary sorting style contain the term BIN.

Composite sorting style is denoted by a term to indicate sensitivity for case (CI/CS), accent (AI/AS), kana (KI/KS), and width (WI/WS). Minimally, a composite sorting style term contains a sensitivity designation for case and for accent. For example, a composite sorting style term of CI_AS signifies case insensitivity and accent sensitivity. A term of **CI_AI_KI_WI** signifies case insensitivity, accent insensitivity, kana insensitivity, and width insensitivity.

SQL Server **Installation and MDB Collations**

The collation sequence for the MDB is set based on the default collation sequence of the SQL Instance. The default collation sequence for the instance is specified when SQL Server is installed. There are many collations that can be selected including those provided by SQL Server and those that come with Windows.

Some of the available collation sequences do not support kana or width sensitivity. If such a collation is in effect when the MDB is created, case and accent sensitive values are set for columns but since kana and width sensitive characters are not available in the collation they are not set.

Setting the MDB Database Collation

The SQL Server MDB is created as a case insensitive database. This allows database object names (such as table names, view names, column names, and so forth) to be referred to in upper, lower or mixed case. The MDB is also created as kana insensitive and width insensitive. The accent sensitivity of the MDB is inherited from the SQL Server instance and therefore may vary by installation.

During MDB creation the language and accent sensitivity of the database server are determined by examining the database collation of the SQL Server instance. The collation established for the MDB is a combination of the language of the SQL Server instance, the accent sensitivity of the instance and a case insensitive setting. If the database server does not have an accent sensitivity setting (if it uses a binary sort), then the MDB will use accent sensitive as a part of its collation setting.

For example, if the SQL Server instance has a collation of Latin1_General_CS_AI, then the MDB will have a collation of Latin1_General_CI_AI. Similarly, if the instance has a collation of Korean_90_CS_AS_KS_WS, then the collation of the MDB will be Korean_90_CI_AS. Because MDB collations do not specify width or kana sensitivity, they are implicitly width and kana insensitive.

Setting MDB Column Collations

Data in MDB columns may have either case insensitive or case sensitive collations. Columns that are defined as case insensitive inherit the accent sensitivity of the SQL Server instance; they will also be kana and width insensitive. Columns that are defined as case sensitive are also accent sensitive.

Microsoft SQL Server Maintenance

Extended updates to the MDB will eventually cause reduced Microsoft SQL Server performance. Microsoft SQL Server should receive the following periodic maintenance:

- **Monitor indexes** - Fragmented indexes should be rebuilt using Microsoft SQL Server facilities.
- **Monitor data files** - Operating system data files should be de-fragmented as necessary.
- **Monitor the transaction log -** The size of the transaction log automatically expands and shrinks. For best performance:
  - Increase the initial size of the transaction log file based on estimated usage.
  - Use a specific growth increment amount such as 100 Megabytes, instead of a percentage increment.
  - Disable automatic transaction log file shrinkage and manually shrink the transaction log files as determined from your monitoring efforts.
  - De-fragment the transaction log file as necessary.
- **Monitoring disk space** - Disk space should be kept at least 20% free.
- **Backup and restore** - The following should be included in your backup and recovery plan:
  - Database and transaction log files should be scheduled for regular backups.
  - The master database should be included in the backup plan.
  - Multiple simultaneous backup devices may be used to improve performance.

For More Information, see *Microsoft SQL Server Books Online* (*Database Maintenance Plan Wizard, DBCC SHOWCONTIG*, and *DBCC_INDEXDEFRAG*).

Remove the SQL Server MDB

The MDB should not be removed except in extreme cases.

Before removing an MDB, make sure the data stored in the database is no longer needed and that all products that were accessing the database have been uninstalled or configured to work with another MDB. Once an MDB is removed the data is lost.

Note: You should make a backup of the MDB before removing it, especially if more than one product has shared it.

To remove a SQL Server MDB

1. Using the SQL Server Enterprise manager, remove the MDB database.
2. Remove the signature file located in the folder that was specified in the MDB_TARGET_DIR parameter of the installation.

   The file is named:

   `mdb.signature.txt`

Chapter 6: Oracle MDB Installation and Maintenance

This section contains the following topics:

Storage Considerations (see page 45)
Database Installation, Configuration, and Deployment (see page 47)
MDB Database Users (see page 51)
Security Considerations (see page 51)
Monitoring and Administration (see page 54)

Storage Considerations

Oracle includes features that can maximize the efficiency of your file storage system. Consideration of these features when implementing your Oracle MDB may be beneficial.

Stripe and Mirror Everything - The S.A.M.E. Method

Oracle's 10g's Automatic Storage Management (ASM) is based on the S.A.M.E. initiative. The main objective of S.A.M.E. offers maximum configuration flexibility and performance tuning by using large I/O and minimize random access by minimizing length of head movement.

For best I/O:

- Stripe using disk technology instead of the database
- Mirror data
- Use outer edge of disk for hot data
- Hot data on outer edge of disk and colder on inner

RAID

RAID 5 is powerful and inexpensive, but best avoided when configuring Oracle databases. An alternative to RAID 5 is RAID 0, commonly known as disk mirroring.

For optimal performance, the MDB should not be placed on RAID 5 storage subsystem. For best RAID performance, use hardware RAID 1+0. RAID 1+0 is preferred because of the heavy write penalty associated when using RAID 5 and RAID 1 (Mirroring) provide faster READ I/O. If RAID 1+0 (strip + mirror) is not available, then RAID 0+1 (Mirror of strip) would be another alternative.

Log Files

Place Redo Logs and Archive Logs separate from data files to assist with performance and recovery purposes. Keep key table datafiles on different disks (or separate arrays) and controllers than the corresponding index datafiles if not using ASM.

File Systems

An Oracle MDB is comprised of several files which store user data, database meta data, and even information to recover from a failure. As such you should take into consideration the type of storage sub-system you will be locating the files on. There are the following options:

- File System - Creates database files managed by your operating system's file system.

  Note: File System is the choice recommended for MDB deployment.

- Automatic Storage Management (ASM) - ASM allows automatic stripe-and-mirror everywhere approach for automatically load balancing the disk I/O sub-system. Automatic Storage Management requires a separate instance to configure and manage disks groups.
- Raw Devices - This method is primarily used in Oracle Real Application Clusters environments.

Optimum Flexible Architecture (OFA)

The MDB follows Oracle's Optimum Flexible Architecture (OFA) for layout. An OFA implementation is the standard that defines how to set up Oracle Databases across all platforms in a consistent manner.

Database Installation, Configuration, and Deployment

The Oracle user that is used to run the MDB installation must have database administrator privileges assigned.

Database Block Size

Larger block sizes result in more efficient I/O activity at potential expense of less efficient cache utilization and greater strain on the I/O system. An 8KB block size should be utilized in your MDB Oracle deployment. Selecting a block size other than 8KB requires advanced knowledge and should only be done when absolutely required.

Tablespaces

Installation allows you to use existing tablespaces or have new ones created for the MDB. If tablespaces are created, a tablespace for data and a tablespace for indexes will be created for optimal performance.

The system defined defaults for temporary and undo tablespaces are used.

Note: Since the MDB schema is predetermined, customers are advised not to alter or add additional tablespaces or tablespaces datafiles without consulting CA support or CA services.

Using Existing Tablespaces for the MDB

If you choose to use existing tablespaces for the MDB, then the tablespaces require a minimum of 200 megabytes of available disk space If this amount of space is not available the creation of the MDB will fail.

Existing tablespaces should have archive logging enabled so that online backups can be taken, archive audit log analysis may be performed, and data recovery options such as complete and point-in-time media are available.

Tablespaces Created by MDB Installation

When the MDB is installed, tablespaces are created with archive logging enabled. Tablespaces are created using the EXTENT MANAGEMENT LOCAL clause, enabling automate extent management. Tablespaces are also created with the SEGMENT SPACE MANAGEMENT AUTO clause enabling Automatic Segment Space Management.

System Temp Tablespace

The MDB uses a system temp tablespace to store temporary tables. At least 50 megabytes of space should be available for this purpose.

Database File Management

AUTOEXTEND is set to enable data files to grow. Proper monitoring of the operating system and disk space should be maintained to insure the space is always available for database growth.

Note: Products such as CA Unicenter Database Performance Management and CA Unicenter NSM may be utilized to automate and monitor such work.

Global Database Name, System Identifier (SID)

The Global Database Name is the full name of the database which uniquely identifies it from other Oracle databases. The global database name is of the form database_name.database_domain as in omdbprod1.us.ourcompany.com. The database name portion (omdbprod1) designates this as the first production Oracle MDB within the enterprise. The database domain portion (us.ourcompany.com) specifies the domain in which the MDB is located. Together database name and domain make up the Global Database Name.

Processes Parameter

The default value for this parameter is 150 which is acceptable for the majority of MDB deployments.

Character Sets

Choose from one of the following options:

- Choose Default if you only need the MDB to support the language currently used by the operating system.
- Choose Unicode (AL32UTF8) if your MDB needs to support multiple languages.

Connection Mode

When setting up connections, the Oracle MDB Dedicated Server Mode is recommended so there is a dedicated server process for each user process. Dedicated Server Mode is used when the number of total clients is small, or with persistent long-running requests to the database.

Redo Log Files

A database has a minimum of two redo log groups. A typical configuration has three redo log groups. Multiplexing of the redo log files is highly recommended.

MDB Database Users

When the MDB is created, a user named MDBADMIN is created. MDBADMIN is the user that owns the MDB database.

An Oracle user with SYSDBA privileges is required to create the MDB. This user's password should be specified when creating the MDB. If no user is specified when the MDB is created, the SYS user will be used and it's password will be required. However, use of the SYS and SYSTEM accounts should be avoided when creating or making changes to the MDB schema.

Security Considerations

To ensure that your MDB is secure:

- Change the passwords for all MDB administrative-level accounts routinely.
- Keep the MDB in a secure location. Make sure the database is in a directory owned by a secure user who has read-only permissions. Any users who have access to a saved database may potentially restore it on another system and view the data contained.

Archive Logging

Upon installation of the MDB, archive logging is enabled allowing you to:

- Perform online backups so you will have complete and point-in-time media data recovery options.
- Perform archive audit log analysis.

Note: You may utilize the Unicenter Database Analyzer for Oracle to comprehensively audit, recover, and analyze the Oracle log.

Undo and Database Backup

Changes made by transactions to the database are stored using undo data. When you install the database, undo tablespace and UNDO_MANAGEMENT = AUTO should be set. This enables undo functionality should the need arise in a recovery situation.

To control retention of undo records, Oracle maintains an undo retention period, which in turn affects the size of the undo tablespace. The longer the retention period, the bigger the tablespace. The undo retention period should be at least as long as your longest running query.

Logical-level backups, such as exporting database objects like tables or tablespaces, may be a useful supplement to physical backups for some purposes but may not protect your entire database.

To back up your database is to make backup copies of your datafiles, control file, and archived redo logs if any. Restoring a database from backup is simply copying the physical files that make up the database from some backup medium (disk or tape) to their locations during normal database operation. Recovery of your database is the process of updating database files restored from a backup with the changes made to the database since the backup, typically using redo log files.

To take advantage of Oracle's features for automatic management of backup and recovery, configure your database for

- Offline Backup - When there is the opportunity for downtime, shutdown the databases and capture a full backup. An offline backup should include all the datafiles, online redo logs and at least one control file.
- Online Backup - An online backup is done while the database is open and accessible by users and applications. You must have an offline backup taken earlier and be in archivelog mode to be a valid backup. You may backup some or all of the data files and you must be sure to backup the archived redo logs that are created between online backups.
- Logical Backup - This is an export and may be done for a full database, specific users, or particular tables.

With an offline backup or export you may only recover to the point in time at which the backup was done. With an online backup you may recover to an SCN (transaction), a specific point in time, or to a point in time AFTER the backup was made (using the archived redo logs).

Note: To ease complexity and automate backup and recovery activities, you may use a solution such as CA BrightStor ArcServe Backup for Oracle.

Monitoring and Administration

An Oracle MDB will not stop functioning if properly setup and managed. Events such as database errors, file systems filling up, and out of space conditions should be actively monitored and fixed to insure continuous operation.

You should regularly perform the following:

- MDB monitoring, performance management, and analysis
- Oracle defragmentation and space recovery
- MDB administration
- Occasional exporting of data from Oracle
- MDB backup and recovery precautions

MDB Monitoring, Performance Management, and Analysis

The MDB should be monitored with alarm and event notification in case of any problems.

Note: Customers may choose to perform these activities manually or may automate the work using CA's Unicenter Database Performance Management.

Oracle MDB Defragmentation and Space Recovery

Fragmentation wastes critical storage space and reduces performance over time. Fragmentation needs to be monitored and reduced as appropriate.

Note: Customers may choose to initiate defragmentation manually or have it run automatically while the MDB continues to run using CA’s Unicenter TSreorg for Oracle.

MDB Administration

Occasionally database objects within the MDB need to be added, removed or altered.

Note: Such administrative tasks may be done manually or alternatively they may be performed automatically using CA's Unicenter Database Administration Solution. Unicenter Database Management r11.1 includes the new Unicenter Database Command Center.

Occasional Exporting of data from Oracle

On occasion there may be the need to remove a portion of the MDB data to another application for backup, replication, or reporting purposes.

Note: Administrators may perform this operation manually or automatically with CA's Unicenter Fast Unload for Oracle.

MDB Backup and Recovery

For MDB protection and database recovery, you should perform backup and recovery on a regular schedule.

Note: You may do this manually or automate it with CA technology solutions. Unicenter Log Analyzer for Oracle helps analyze transaction logs, audit data updates, undo or redo changes, and analyze transaction-level detail to reduce data loss or troubleshoot application performance problems. See the *BrightStor Arcserve Backup for Oracle* guide.

Remove the Oracle MDB

The MDB should not be removed except in extreme cases.

Before removing an MDB, make sure the data stored in the database is no longer needed and that all products that were accessing the database have been uninstalled or configured to work with another MDB. Once an MDB is removed the data is lost.

Note: You should make a backup of the MDB before removing it, especially if more than one product has shared it.

To remove an Oracle MDB

1. Drop the mdb_data and mdb_index tablespaces.

   For example:

   ```
   DROP TABLESPACE MDB_DATA INCLUDING CONTENTS AND DATAFILES
   CASCADE CONSTRAINTS;
   DROP TABLESPACE MDB_INDEX INCLUDING CONTENTS AND DATAFILES
   CASCADE CONSTRAINTS;
   ```

2. Drop the MDBADMIN user. For example:

   ```
   DROP USER MDBADMIN CASCADE
   ```

3. Drop the users that were created by the CA products that were using the MDB.
4. Drop the roles that were created by the MDB installation.

Chapter 7: Unicenter NSM and the MDB

This chapter is a reference for the CA Management Database (MDB) as it is employed by Unicenter NSM components.

This section contains the following topics:

WorldView (see page 57)
Enterprise Management (see page 59)
Performance Management (see page 74)
Unicenter Management Portal (see page 98)
Asset Registration (see page 101)

WorldView

WorldView uses the MDB to store all of its component-specific data, such as ManagedObject definitions.

Security

For comprehensive information about security for WorldView and the MDB, see the chapter "Securing Unicenter NSM" in the *Administrator Guide*.

Data Capacity Estimation

The capacity needed to store WorldView data is based upon how large a network you have. WorldView will have at most two rows in the MDB to resolve one WorldView object with all of its property. For each network device that has Agent Technology agents, another WorldView object is created to show the state of each agent. Each network object can have multiple agents.

For example, in a network with ten full subnets (254 objects per subnet), there would be 2540 * 2 (5080) rows in the MDB to represent the network objects in WorldView. Each one of these can have an overage of four agents, which would add another (2540 * 4) * 2 (20320) rows. Each one of the WorldView objects that is in the topology has one row for each parent/child containment inclusion (12700).

Topology or Implementation Scenarios

See the appendix "Replicating Objects in the WorldView Repository Using Repository Bridge" in the *Administrator Guide*.

Tuning and Performance Requirements

If you find that Datascoping in WorldView is taking longer than normal, you should optimize the MDB. If you want to optimize only the portion of the MDB that deals directly with WorldView Datascoping, run the following commands:

```
optimizedb -zk -zv -zr249 -zu249 -zc i_13309065 -rtng_inclusion –auuid -
aparent_uuid -achild_uuid -rtng_managedobject -auuid -aasset_uuid

sysmod dbname
```

Backup, Administration, and Maintenance Requirements

See the *Management Database Overview* and the Microsoft SQL Server online books or *Ingres r3 Database Administrator Guide*, Chapter 15, "Backup and Recovery" for discussion of these topics.

Integration Considerations

Whenever a WorldView object of particular network classes is created, the object is registered to the CA Common Asset tables using the name of the object (DNS Name). A Common Asset UUID is returned and stored in the asset_uuid WorldView property. This value can be used to traverse the Common Asset tables. The classes that are registered to the Common Asset tables are classes and subclasses of the following:

- Router
- Switch
- Bridge
- Host
- Workstation
- Printers
- Hub

A class can be made to call the Common Asset Registration by setting the class level property regasset to true. Since the Common Assets from WorldView are tracked by DNS Name, the name of the object created must be a unique dnsname.

Enterprise Management

Enterprise Management uses the MDB to store its component-specific data, such as Calendar definitions, Message Records and Actions, Security Access Rules, and so forth. The data content can be migrated from an existing database during an upgrade or default data is created for you during a fresh installation.

Some Enterprise Management components only need access to the MDB at initialization time, while other components require ongoing and recurring access during execution. For example, the following components typically access the database as follows:

CAUTIL and the GUI

Access the database continuously as the user drives them.

Event Management and Calendar

Access the database during initialization and reload time.

DOC

Accesses the database only as part of other components that are accessed by the GUI.

CA Trap Daemon

Accesses the database during initialization.

WMS

Accesses the database during initialization and run time to update log entries.

Enterprise Management on Windows, Linux, and UNIX share a common schema in the MDB. This means that when these platforms are using the same database type, Enterprise Management data can be shared from a single MDB server no matter which platform hosts the database. If one platform is using a different database type (for example, one platform uses Ingres and the other uses Microsoft SQL), the MDB cannot be shared.

Security

Enterprise Management access is classified into two types, as follows:

- Direct access to the Enterprise Management tables in the MDB
- Access to WorldView tables in the MDB using the WV API

These two table sets can be in different MDBs.

All MDB connections use the administrative account defined during installation, which defaults to nsmadmin. The nsmadmin user is a member of the EMADMIN group and UNIADMIN user groups, and therefore has administrative access to all Unicenter NSM tables in the MDB. The administrative account is stored in the Enterprise Management settings CAI_DBUID and CAI_DBPWD. Enterprise Management can use any ID you choose, either during installation or afterwards by modifying the Enterprise Management settings.

If the ID you choose is to be used for any groups other than Enterprise Management such as WVADMIN, REGADMIN, and so forth, it also needs to be a member of those groups.

If the database server information changes, you must synchronize the connection credentials by resetting these settings. Any new user that you create must be given all the same database permissions, grants, and groups as nsmadmin.

For additional information about securing Enterprise Management, see the chapter "Securing Unicenter NSM Objects" in the *Administrator Guide*.

Ingres Databases

Security for the MDB is described by the underlying Ingres security model.

See the Microsoft SQL Server online books or the *Ingres Database Administrator Guide* for a complete discussion of security considerations.

Linux and UNIX Platforms

Administrative access to OPR, Calendar, and Security tables in the MDB requires that the user be a member of the Ingres user group EMADMIN. Read-only access to OPR, Calendar, and Security tables in the MDB may be granted by making the user a member of the Ingres user group EMUSER. On Linux and UNIX platforms, administrative access by the super user (root, for example) is implicit and does not require additional configuration.

Data Capacity

The capacity needed to store Enterprise Management data occurs in three areas, as follows:

Calendar

The number of calendars defined in the system determines the data capacity for calendar data in the MDB. Each calendar requires one row in the MDB. If you define 100 calendars, there are 100 rows in the MDB.

Event

The total number of defined message policies, the number of associated messages actions, and the number of console views defined determines the data capacity for event data in the MDB. Each defined message policy, message action, and console view requires one row in the MDB.

Therefore, if you have 100 message policies, 200 message actions, and 50 console views, there are 350 rows in the MDB.

Security

The number of asset types, asset groups, user groups, and permit rules determines the data capacity for security data in the MDB. There are approximately 200 predefined asset types and approximately 1000 predefined permit rules that are created during Unicenter NSM Security installation. Each defined user or user group, asset type or asset group, and permit rule requires one row in the MDB.

The number of permit rules can equal the product of the number of user groups and the number of asset types and asset groups.

Therefore, if you have six user groups and 200 asset types and asset groups, the total number of permit rules can be up to 200 * 6 rows, or 1200 rows.

As part of your Unicenter NSM deployment plan, you should form an opinion about the scale of management you will be doing across the Unicenter NSM components and how much data you expect to collect based on the information provided in this section. If you expect to use several of the Unicenter NSM components with a team of administrators to manage thousands of nodes with data collected over days and weeks from the same MDB, you should ensure there are several gigabytes available for growth.

Enterprise Management data in the MDB is mostly static after it is defined and loaded. It does not grow substantially over time without user interaction. For example, if you load twenty thousand message records and actions, the MDB data files will require more disk space than if you have only two hundred message records and actions.

Topology or Implementation Scenarios

Enterprise Management access to the MDB is supported for local and remote servers. However, a local MDB provides better performance. As most Enterprise Management components (except for the Unicenter NSM Job Management Option) connect once to the MDB and then cache data locally, remote MDB usage usually does not cause a noticeable performance problem. Managing a remote MDB with twenty thousand message records and actions will, however, be a slower process than managing a similar local one.

Linux and UNIX Platforms

Enterprise Management data can be stored in either a local or remote MDB on any supported Unicenter NSM r11 platform. However, all Enterprise Management components must share the same MDB. The Unicenter NSM MDB is configured during the Unicenter NSM installation process.

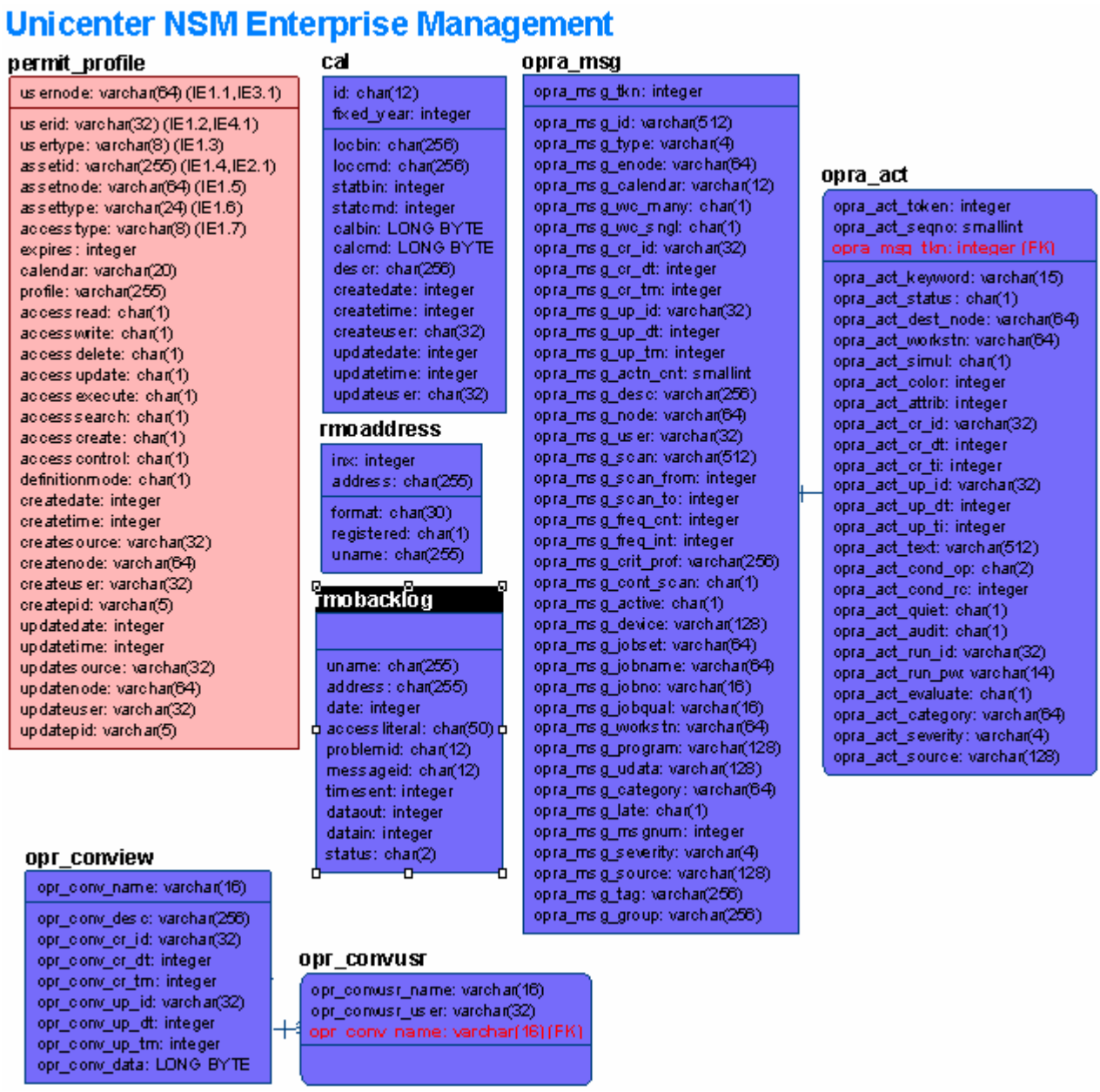
Enterprise Management daemons and utilities establish both local and remote database connections using Ingres VNODEs. These VNODEs can be managed and configured using standard Ingres utilities.

See the *Ingres Connectivity Guide* for more information about Ingres database access methods.

Schema and Reporting Information

Following is a simplified schema of the Enterprise Management model in the MDB.

Note: Installing Enterprise Management does not create tables for Enterprise Management in any existing MDB. The MDB installs with pre-existing, empty, Enterprise Management tables. Only when you install Enterprise Management does the product insert data into these tables.


Unicenter NSM Enterprise Management
permit_profile
usernode: varchar(64) (IE1.1,IE3.1)
userid: varchar(32) (IE1.2,IE4.1)
usertype: varchar(8) (IE1.3)
assetid: varchar(255) (IE1.4,IE2.1)
assetnode: varchar(64) (IE1.5)
assettype: varchar(24) (IE1.6)
accesstype: varchar(8) (IE1.7)
expires: integer
calendar: varchar(20)
profile: varchar(255)
accessread: char(1)
accesswrite: char(1)
accessdelete: char(1)
accessupdate: char(1)
accessexecute: char(1)
accesssearch: char(1)
accesscreate: char(1)
accesscontrol: char(1)
definitionmode: char(1)
createdate: integer
createtime: integer
createsource: varchar(32)
createnode: varchar(64)
createuser: varchar(32)
createpid: varchar(5)
updatedate: integer
updatetime: integer
updatesource: varchar(32)
updatenode: varchar(64)
updateuser: varchar(32)
updatepid: varchar(5)
cal
id: char(12)
fixed_year: integer
locbin: char(256)
loccmd: char(256)
statbin: integer
statcmd: integer
calbin: LONG BYTE
calcmd: LONG BYTE
descr: char(256)
createdate: integer
createtime: integer
createuser: char(32)
updatedate: integer
updatetime: integer
updateuser: char(32)
rmoaddress
inx: integer
address: char(255)
format: char(30)
registered: char(1)
uname: char(255)
rmobacklog
uname: char(255)
address: char(255)
date: integer
accessliteral: char(50)
problemid: char(12)
messageid: char(12)
timesent: integer
dataout: integer
datain: integer
status: char(2)
opra_msg
opra_msg_tkn: integer
opra_msg_id: varchar(512)
opra_msg_type: varchar(4)
opra_msg_enode: varchar(64)
opra_msg_calendar: varchar(12)
opra_msg_wc_many: char(1)
opra_msg_wc_sngl: char(1)
opra_msg_cr_id: varchar(32)
opra_msg_cr_dt: integer
opra_msg_cr_tm: integer
opra_msg_up_id: varchar(32)
opra_msg_up_dt: integer
opra_msg_up_tm: integer
opra_msg_actn_cnt: smallint
opra_msg_desc: varchar(256)
opra_msg_node: varchar(64)
opra_msg_user: varchar(32)
opra_msg_scan: varchar(512)
opra_msg_scan_from: integer
opra_msg_scan_to: integer
opra_msg_freq_cnt: integer
opra_msg_freq_int: integer
opra_msg_crit_prof: varchar(256)
opra_msg_cont_scan: char(1)
opra_msg_active: char(1)
opra_msg_device: varchar(128)
opra_msg_jobset: varchar(64)
opra_msg_jobname: varchar(64)
opra_msg_jobno: varchar(16)
opra_msg_jobqual: varchar(16)
opra_msg_workstn: varchar(64)
opra_msg_program: varchar(128)
opra_msg_udata: varchar(128)
opra_msg_category: varchar(64)
opra_msg_late: char(1)
opra_msg_msgnum: integer
opra_msg_severity: varchar(4)
opra_msg_source: varchar(128)
opra_msg_tag: varchar(256)
opra_msg_group: varchar(256)
opra_act
opra_act_token: integer
opra_act_seqno: smallint
opra_msg_tkn: integer (FK)
opra_act_keyword: varchar(15)
opra_act_status: char(1)
opra_act_dest_node: varchar(64)
opra_act_workstn: varchar(64)
opra_act_simul: char(1)
opra_act_color: integer
opra_act_attrib: integer
opra_act_cr_id: varchar(32)
opra_act_cr_dt: integer
opra_act_cr_ti: integer
opra_act_up_id: varchar(32)
opra_act_up_dt: integer
opra_act_up_ti: integer
opra_act_text: varchar(512)
opra_act_cond_op: char(2)
opra_act_cond_rc: integer
opra_act_quiet: char(1)
opra_act_audit: char(1)
opra_act_run_id: varchar(32)
opra_act_run_pw: varchar(14)
opra_act_evaluate: char(1)
opra_act_category: varchar(64)
opra_act_severity: varchar(4)
opra_act_source: varchar(128)
opr_conview
opr_conv_name: varchar(16)
opr_conv_desc: varchar(256)
opr_conv_cr_id: varchar(32)
opr_conv_cr_dt: integer
opr_conv_cr_tm: integer
opr_conv_up_id: varchar(32)
opr_conv_up_dt: integer
opr_conv_up_tm: integer
opr_conv_data: LONG BYTE
opr_convusr
opr_convusr_name: varchar(16)
opr_convusr_user: varchar(32)
opr_conv_name: varchar(16) (FK)

Tables in the Schema

The tables used by Enterprise Management are documented as follows. PK refers to Primary Index Key and SK refers to Secondary Index Key.

Table cal

Component: Calendar

Contains the data definition for calendars. Calendar is a common component used by Event, Alert, Workload, Security, and other components.

Column	Definition	Description	Keys
id	char(12)	calendar id	PK
fixed_year	integer	Fixed Year if non zero	PK
locbin	char(256)	binary file location	
loccmd	char(256)	command file location	
statbin	integer		
statcmd	integer		
calbin	long byte	calendar binary file	
calcmd	long byte	calendar command file	
descr	char(256)	description	
createdate	integer	create date	
createtime	integer	create time	
createuser	char(32)	create user	
updatedate	integer	update date	
updatetime	integer	update time	
updateuser	char(32)	update user	

Table opra_msg

Component: Event

Contains message record definitions of the Event policy.

Column	Definition	Description	Keys
opra_msg_tkn	integer	uniq message token	PK
opra_msg_id	varchar(512)	message id	PK
opra_msg_type	varchar(4)	message type(CMD or MSG)	

Column	Definition	Description	Keys
opra_msg_enode	varchar(64)	evaluation node	
opra_msg_calendar	varchar(12)	calendar to use	
opra_msg_wc_many	char(1)	mwc override	
opra_msg_wc_sngl	char(1)	swc override	
opra_msg_cr_id	varchar(32)	message creation user id	
opra_msg_cr_dt	integer	message creation date	
opra_msg_cr_tm	integer	message creation time	
opra_msg_up_id	varchar(32)	message update user id	
opra_msg_up_dt	integer	message update date	
opra_msg_up_tm	integer	message update time	
opra_msg_actn_cnt	smallint	count of actions –not in use	
opra_msg_desc	varchar(256)	description	
opra_msg_node	varchar(64)	node origin	
opra_msg_user	varchar(32)	user origin	
opra_msg_scan	varchar(512)	scan event text as one field	
opra_msg_scan_from	integer	start position for 'scan'	
opra_msg_scan_to	integer	end position for 'scan'	
opra_msg_freq_cnt	integer	select if 'cnt' msgs	
opra_msg_freq_int	integer	arrive in 'int'secs	
opra_msg_crit_prof	varchar(256)	additional matching flags: case, RE	
opra_msg_cont_scan	char(1)	continue to scan: y/n	
opra_msg_active	char(1)	msg record active: y/n	
opra_msg_device	varchar(128)	'device' matching filter	
opra_msg_jobset	varchar(64)	'jobset' matching filter	
opra_msg_jobname	varchar(64)	'jobname' matching filter	
opra_msg_jobno	varchar(16)	'jobno' matching filter	
opra_msg_jobqual	varchar(16)	jobqual' matching filter	
opra_msg_workstn	varchar(64)	'workstation' matching filter	
opra_msg_program	varchar(128)	'program' matching filter	

Column	Definition	Description	Keys
opra_msg_udata	varchar(128)	‘user data’ matching filter (CA_UDATA env.)	
opra_msg_category	varchar(64)	category' matching filter	
opra_msg_late	char(1)	process late event-not in use--	
opra_msg_msgnum	integer	'msgnum' matching filter	
opra_msg_severity	varchar(4)	'severity' matching filter	
opra_msg_source	varchar(128)	'source' matching filter	
opra_msg_tag	varchar(256)	'tag' matching filter	
opra_msg_group	varchar(256)	msg record group	

Table opra_act

Component: Event

Contains the message action definitions of the Event policy.

Column	Definition	Description	Keys
opra_act_token	integer	uniq action token	PK
opra_act_seqno	smallint	action sequence number	PK
opra_act_keyword	varchar(15)	action keyword	
opra_act_status	char(1)	Active or Inactive	
opra_act_dest_ node	varchar(64)	run act on node	
opra_act_workstn	varchar(64)	workstation id	
opra_act_simul	char(1)	simulate sw y|n	
opra_act_color	integer	display color	
opra_act_attrib	integer	display attib.	
opra_act_cr_id	varchar(32)	create user id	
opra_act_cr_dt	integer	create date	
opra_act_cr_ti	integer	create time	
opra_act_up_id	varchar(32)	update user id	
opra_act_up_dt	integer	update date	
opra_act_up_ti	integer	update time	

Column	Definition	Description	Keys
opra_act_text	varchar(512)	action text	
opra_act_cond_op	char(2)	conditional opcode eq|ne|...	
opra_act_cond_rc	integer	return code to check	
opra_act_quiet	char(1)	quiet switch y|n	
opra_act_audit	char(1)	audit switch y|n	
opra_act_run_id	varchar(32)	user id of action	
opra_act_run_pw	varchar(14)	password for user id	
opra_act_evaluate	char(1)	evaluate flag y|n	
opra_act_category	varchar(64)	override category	
opra_act_severity	varchar(4)	overrider serverity	
opra_act_source	varchar(128)	override source	

Table opra_ctl

Component: Event

Contains the next token number used for message action record.

Column	Definition	Description	Keys
opra_ctl_last_tkn	integer	last message token used	
opra_ctl_wcmany	char(1)	global system multi wc	
opra_ctl_wcsngl	char(1)	global system single wc	

Table opr_conview

Component: Event

Contains console view definitions.

Column	Definition	Description	Keys
opr_conv_name	varchar(16)	console view name	PK
opr_conv_desc	varchar(256)	console view description	
opr_conv_cr_id	varchar(32)	creation id	
opr_conv_cr_dt	integer	creation date	
opr_conv_cr_tm	integer	create time	
opr_conv_up_id	varchar(32)	update id	

Column	Definition	Description	Keys
opr_conv_up_dt	integer	update date	
opr_conv_up_tm	integer	update time	
opr_conv_data	long byte	binary filter data	

Table management_asset_group

Component: Security

Contains definitions for the asset group object.

Column	Definition	Description	Keys
assetnode	varchar(64)	node name for asset	PK
id	varchar(20)	asset group id	PK
name	varchar(255)	asset group name	PK
asset	varchar(255)	asset id	PK,SK
type	varchar(24)	type of asset	
createdate	integer	creation date	
createtime	integer	creation time	
createsource	varchar(64)	source for this record	
createnode	varchar(64)	creation node	
createuser	varchar(32)	creation user	
createpid	char(5)	creation process id	
updatedate	integer	update date	
updatetime	integer	update time	
updatesource	varchar(64)	source for update	
updatenode	varchar(64)	udpate node	
updateuser	varchar(32)	update user	
updatepid	char(5)	update process id	

Table user_group

Component: Security

Contains definitions for the user_group object.

Column	Definition	Description	Keys
usernode	varchar(64)	user node name	PK
id	varchar(32)	user group id	SK(1)
gid	integer	group id number	SK(1)
name	varchar(255)	user group name	SK(1)
userid	varchar(32)	user member id	SK(1)
type	varchar(8)	user or user group	
createdate	integer	creation date	
createtime	integer	creation time	
createsource	varchar(32)	creation source	
createnode	varchar(64)	node of creation	
createuser	varchar(32)	creation user	
createpid	varchar(5)	creation process id	
updatedate	integer	update date	
updatetime	integer	update time	
updatesource	varchar(32)	update source	
updatenode	varchar(64)	update node	
updateuser	varchar(32)	update user	
updatepid	varchar(5)	update process id	
globalgroup	varchar(5)	grobal group name	

Table permit_profile

Component: Security

Contains the security rules and attaches them to users, user groups, assets, or asset groups.

Column	Definition	Description	Keys
usernode	varchar(64)	user node	PK, SK(1)
userid	varchar(32)	user id for this rule	SK(1)

Column	Definition	Description	Keys
usertype	varchar(8)	user or user group	SK(1)
assetid	varchar(255)	asset identification	SK(1), SK(2)
assetnode	varchar(64)	asset node	SK(1)
assettype	varchar(24)	calendar or other object	SK(1)
accesstype	varchar(8)	permit or deny	SK(1)
expires	integer	Rule expiration date	
calendar	varchar(20)	rule calendar id	
profile	varchar(255)		
accessread	char(1)	access read	
accesswrite	char(1)	acceess write	
accessdelete	char(1)	access delete	
accessupdate	char(1)	access update	
accessexecute	char(1)	access execute	
accesssearch	char(1)	access search	
accesscreate	char(1)	access create	
accesscontrol	char(1)	access control	
definitionmode	char(1)		
createdate	integer	create date	
createtime	integer	create time	
createsource	varchar(32)	create source	
createnode	varchar(64)	create node	
createuser	varchar(32)	create user	
createpid	varchar(5)	create process id	
updatedate	integer	update date	
updatetime	integer	update time	
updatesource	varchar(32)	update source	
updatenode	varchar(64)	update node	
updateuser	varchar(32)	update user	
updatepid	varchar(5)	update process id	

Table assettyp_profile

Component: Security

Describes various asset types and access modes that are applicable to those asset types.

Column	Definition	Description	Keys
type	varchar(24)	type of asset	PK
violmode	varchar(8)	violation mode (quiet)	
accessread	char(1)	access read	
accesswrite	char(1)	access write	
accessdelete	char(1)	access delete	
accessupdate	char(1)	access update	
accessexecute	char(1)	access execute	
accesssearch	char(1)	access search	
accesscreate	char(1)	access create	
accesscontrol	char(1)	access control	
definitionmode	char(1)	definition mode	
protectionlevel	varchar(20)	level of protection (object)	
createdate	integer	create date	
createtime	integer	create time	
createsource	varchar(64)	create source	
createnode	varchar(64)	create node	
createuser	varchar(32)	create user	
createpid	char(5)	create process id	
updatedate	integer	update date	
updatetime	integer	update time	
updatesource	varchar(64)	update source	
updatenode	varchar(64)	update node	
updateuser	varchar(32)	update user	
updatepid	char(5)	update process id	

Tuning and Performance

Consider the following to ensure optimum performance:

- Accesses and updates to data in a local database are faster than doing the same with a remote database.

Ingres Databases

- Ensure the database is optimized for data load and access. For Ingres, this means ensuring the Enterprise Management tables are re-indexed after loading large amounts of data or after optimizing the tables (or the entire MDB) using the Ingres optimizedb command.

  Follow the standard Ingres recommendations to optimize tables as described in the *Ingres Database Administrator Guide*.

Backup, Administration, and Maintenance Requirements

See the *Management Database Overview* and the Microsoft SQL Server online books or *Ingres Database Administrator Guide* for information about this topic.

Note: Enterprise Management does not use Ingres virtual node connections and therefore no virtual node is listed in any Ingres utilities, such as netutil.

Integration Considerations

For custom or third party applications to integrate with Enterprise Management data, it is important to understand the schema described in the Schema and Reporting Information topic, and to have security access to the Enterprise Management tables.

Take great care in interacting with the database, as poorly constructed SQL queries, altering indexes or table structures, or locking rows and tables that are in use by software applications can cause critical errors, serious performance degradation, or both.

Linux and UNIX Platforms

Enterprise Management data held in the MDB must be available to consuming applications with appropriate access to the data. Enterprise Management permits the following:

- Read/write access to Enterprise Management data in the MDB for a user through the uni_db_addusr utility.
- Read only access to Enterprise Management data in the MDB for a user.

The uni_db_addusr utility lets you manage MDB access to Enterprise Management tables. The super user (root, for example) implicitly has full administrative access to all Enterprise Management tables and no additional configuration is required.

Use the following command to grant administrative (read/write) privileges to Enterprise Management tables for non-root users:

```
$CAIGLBL0000/db/scripts/uni_db_addusr -a username password
```

Use the following command to remove administrative privileges to Enterprise Management tables for non-root users:

```
$CAIGLBL0000/db/scripts/uni_db_addusr -r username
```

You can grant read only access by adding a non-root user to the Ingres user group EMUSERUse standard Ingres utilities (for example, netutil).

For a complete explanation of Enterprise Management security with respect to the MDB, see the section Access to EM Database Tables by Non-Root Users on UNIX/Linux in *Inside Event Management and Alert Management*.

See the *Ingres Database Administrator Guide* for more information about the Ingres security model.

High Availability Considerations

Enterprise Management's CA Unicenter service and the database service, when running in High Availability mode, must be launched by the Cluster Administrator and not by the Windows Service Control Manager, if running on the same node of a cluster. They will then switch together from one node to another during a failover event.

In highly available installations, the MDB data files, which include Enterprise Management data, reside on the cluster shared disk. When Enterprise Management and the database fail over from one node to the next, the shared disk containing the MDB data files becomes available to the new node and the Enterprise Management and database server on the new node pick up where they left off on the old node.

For additional information about high availability and cluster management with Unicenter NSM, see the appendix "Making Components Cluster Aware and Highly Available" in the *Administrator Guide*.

Performance Management

The Performance Domain Server enables uploading of performance data into the MDB using historical performance data provided by the Performance Data Grid. All this happens by default once the Performance Domain Server is installed. The data is published to the MDB automatically, but the content and granularity of data stored in the MDB is configurable. Additionally, the Performance Domain Server enables real time publishing of Asset and Configuration information into the MDB.

Performance Scope, Performance Trend, Performance Configuration, Performance Web Reporting, and cfgutil applications access the Worldview objects in the MDB.

Security

Systems Performance access is classified into two types, as follows:

- Direct access to the Systems Performance tables in the MDB
- Access to WorldView tables in the MDB using the WV API

Direct Access to Systems Performance Tables in the MDB

You can directly access Systems Performance tables in the MDB through either of the following two ways:

- Performance Domain Server
- SQL Access

Performance Domain Server

The Performance Domain Server accesses the MDB directly to store Performance, Configuration, and Asset information.

- Systems Performance does not create any predefined OS or database users to access the MDB.
- Out-of-the-box, the database user (created by the NSM installer) associated with database group UNIADMIN is used by Performance Domain Server to access the MDB. By default, this user has rights to access Systems Performance tables and the Asset registration tables.
- The database group PDADMIN provides read/write access to Performance Management tables in the MDB. Customers needing restricted access only to Performance Data in the MDB can associate a known user to this database group after installation. See the topic Integration Considerations for how to do this.

All MDB credentials collected by the Systems Performance installer are stored on the machine hosting the domain server using PKI encryption in the file %CASP_PATH%/appdata/domainserver/data/dbcred.dat ($CASP_PATH for non-windows).

The file dbcred.dat maintains the credentials needed to connect to the MDB and must be managed using cfgutil to handle credential changes on the MDB server. See Integration Considerations for more information about managing the MDB credentials using cfgutil.

Ingres Databases

Ingres allows connection to remote database servers using vnodes. vnodes are predefined Ingres names that store connection information for a remote database. An alternate mechanism supported by Ingres is called vnodeless or dynamic vnodes. With dynamic vnodes, information is not predefined; the information is used dynamically when needed during connection time.

The Performance Domain server uses vnodeless connections to connect to the MDB.

Microsoft SQL Server Databases

In this section, "database group" refers to the Microsoft SQL Server "database role."

SQL Access

- The database group PDUSER provides read only access to Performance Management tables. A user associated with this database group has read only access to Performance tables.
- Out-of-the-box, there are no database users associated with the PDUSER database group. To associate a custom user with the PDUSER database group, see Integration Considerations.

Microsoft SQL Server Databases

In this section, "database group" refers to the Microsoft SQL Server "database role."

Access to WorldView tables in the MDB using the WV API

Performance Configuration, Performance Scope, Performance Trend, and cfgutil are applications that access WorldView using the WorldView API. These applications only pass the WorldView repository name and do not pass any MDB user credentials to the WorldView API.

Performance Web Reporting also uses the WorldView API to access WorldView. However, this application passes a user name and a password to the WorldView API. The Systems Performance installer collects the required WorldView user name and password during installation. Performance Web Reporting passes this information to the WorldView API. You can change the WorldView user name and password after installation using the Web Reporting GUI interface.

See the section on Worldview to learn more about the user credentials needed for WorldView access and how to customize these user credentials after installation.

Data Capacity Estimation

The key tables in the model include the Machine, Resource, Day, Time and Data tables. To understand the rationale behind the calculations that follow, consider what causes the data table to grow. The data table effectively stores values that correspond to the four dimensions of Machine, Resource, Day and Time. For every combination of the four dimensions, there is a row in the table as follows:

The number of data rows = Product of number of rows from different dimensions

Dimensions	**Rows**
1 machine =	1 pd_machine row
1 resource =	1 pd_resource row
1 day =	1 pd_day row
1 time value =	1 pd_time row
number of data rows =	number of machines affected *
	number of resources affected *
	number of days affected *
	number of time intervals affected

The information in Performance Management Data Capacity Estimation later in this appendix provides an insight into this area, and also provides tips and sample usage scenarios on how to manage data capacity effectively.

Therefore, if you store one machine with 10 resources for 30 days with data stored at 1 minute granularity, applying the above calculations, we have:

Dimensions	Rows
1 machine =	1 pd_machine row
10 resources =	10 pd_resource rows
30 days =	30 pd_day rows
1 minute granularity =	1440 rows (1440 minutes in a day)
number of pd_val_1min row =	1 * 10 * 30 * 1440 = 432, 000 rows

Using this information, you can consider a few scenarios and measure the impact of the dimensions on the data table in terms of the number of data rows and the disk usage:

1. 10 machines, 500 resources, 5 minute granularity, 1 year data retention
2. 100 machines, 100 resources, 1 hour granularity, 1 year data retention
3. 1000 machines, 50 resources, 3 hour granularity, 1 year data retention
4. 1000 machines, 100 resources, 1 hour granularity, 1 month data retention
5. 500 machines, 150 resources, 1 hour granularity, 3 month data retention
6. 100 machines, 50 resources, 20 minute granularity, 1 month data retention

Scenario	No. of Machines A	No. of Resources B	Frequency in Minutes C	Retention in Days D	No. of Data Rows A*B*(1440 /C)*D
1	10	500	5	365	525600000
2	100	100	60	365	87600000
3	1000	50	180	365	146000000
4	1000	100	60	31	74400000
5	500	150	60	93	167400000
6	100	50	20	31	11160000

Topology or Implementation Scenarios

Direct access to the MDB by Systems Performance is supported for both local and remote MDB servers. However, when possible, use the local MDB server for better performance.

For data storage, the Systems Performance MDB implementation supports a virtually infinite number of implementation scenarios. The volume of data, along with the retention of that data is completely customizable. Overall, the general rule to follow for any implementation is to upload to the MDB only data that you are certain you need, and use, in a reporting or analytical capacity.

A Case Study

Atemp Corporation is a multinational company with operations around the world. Atemp uses Unicenter NSM to manage its enterprise-wide data. Atemp has defined their operational usage of Systems Performance data and wishes to implement a suitable Data Aging and Aggregation strategy for its Management Database.

Operational Usage of Performance Data

Atemp has determined the following usages of its Systems Performance data:

- Atemp has a policy of storing its System Performance Data for at least 3 years.
- Atemp collects Performance data for 1000 machines in the enterprise. Some of these machines (300) are test machines that are reconfigured frequently and the data collected on these machines are not useful by any means.
- The data is originally collected at 5 minute intervals. Although data is regularly collected for 2000 resources, only 45 unique resources are ever used.
- Atemp analyzes resource usage for data up to 1 week old at its finest granularity (that is, 5 minute intervals).
- Atemp occasionally has need to analyze historic data at the very fine level using raw data collected.
- Any data older than a month is aggregated in reports and used for historic analysis and future planning requirements.
- Annual usage reports are required for some key resources (approximately 10).
- Data older than 1 year is seldom used.

Analysis

The following two scenarios are presented:

- Scenario 1 allows the storing of raw Performance Data as collected in a database.
- Scenario 2 allows a planned storage of summarized Performance Data in the database.

To arrive at a good strategy, it is essential to ask some fundamental questions and evaluate the scenarios against them.

- Is it necessary to store all the data in a database? A machine may be configured to collect 2000 resources. Is it necessary to upload all the resources into the database? Would storing the top resources into the database suffice? The historic usage of resources should indicate what the top resources are.
  - Scenario 1: Store all 1000 machines with 2000 resources.
  - Scenario 2: Store data for all 1000 machines (could review this later). Store only the 45 key resources.
- For the data identified as useful, what is a suitable granularity? If data is collected at five minute intervals, is it necessary to store the data in the database at the same granularity? If the analysis is critical and averaging is not appropriate, how long is it necessary to maintain this level of granularity?
  - Scenario 1: Store 5 minute collection as it is in MDB forever.
  - Scenario 2: Store data at a higher granularity of 4 hours. Assume that users needing raw data would use Web Reporting Services to access data from the Performance Data Grid.
- How long is the data needed at the given granularity? Could the data be aggregated after a certain period? If data is aggregated, how could a user get back the original granular data if there is a need to do so in the future? If we have very fine data in the database, could we aggregate the finer data to a higher granularity after 1 week or 1 month?
  - Scenario 1: Data is needed for 3 years. Assume it is necessary to store raw data as it is for 3 years.
  - Scenario 2: Maintain the database for best performance based on use of data.

    Data older than one week could be aggregated to 1 week data. Similarly, data older than four weeks could be aggregated to one month and data older than 12 months could be aggregated to one year. Every time aggregation occurs, it is possible to get rid of the source data that is aggregated.

- What are the costs involved? Are there better alternatives? How many rows are there? If only raw data is of interest, is it better to obtain the raw data from the Grid using Web Reporting Services? The summary data could then be stored in the database for summary analysis and future planning.

  Total number of data rows in = Number of machines *
  Number of resources *
  Number of days *
  Number of time intervals

  Assumptions: Start of 1st year is 1st Jan 2004
  Databases backed up before any aggregation (as described in the topic Data Aging and Aggregation Strategy Implementation)

Scenario 1: Number of rows in pd_val_5min = 1000 * 2000 * (3 * 365) * 288 = 630,720,000,000 rows (630 billion rows)

Scenario 2: As data is aggregated, it is necessary to get the row count in steps. The following steps through the operations involved in aggregation and also gets the approximate row counts involved.

1. Before Week1 data aggregation, the number of rows in pd_val_4hr = 1000 * 45 * 7 * 6 = 1,890,000 rows.

   Aggregate Week 1 data using the following command:

   ```
   aggbyage local mdb 01-jan-2004 08-jan-2004 4hr 1week
   ```

   The command uses a local MDB database and converts four-hour data to one-week data starting (and including) 01-Jan-2004 and ending (and excluding) 08-Jan-2004, with the following results:

   - After Week1 data aggregation, number of rows in pd_val_4hr = 0.
   - After Week1 data aggregation, number of rows in pd_val_1week = 1000 * 45 * 1 * 6 = 270,000 rows.

2. Before Month1 data aggregation (after Week4 aggregation), the number of rows in pd_val_1week = 4 * 270,000 = 1,080,000 rows

   After Month1, if we apply the monthly aggregation using the following command:

   ```
   aggbyage local mdb 01-jan-2004 01-feb-2004 1week 1month
   ```

   The following results are achieved:

   - After Month1 data aggregation, number of rows in pd_val_1week = 0
   - After Month1 data aggregation, number of rows in pd_val_1month = 1000 * 45 * 1 * 6 = 270,000 rows

3. Extending the previous step to Year1 and before Year1 data aggregation (after Week 52 aggregation and Month12 aggregation), number of rows in pd_val_1month = 12 * 270,000 = 3,240,000 rows.

   After Year1, if we apply the yearly aggregation using the following command:

   ```
   aggbyage local mdb 01-jan-2004 02-jan-2005 1month 1year
   ```

   The following results are achieved:

   - After Year1 data aggregation, number of rows in pd_val_1month = 0
   - After Year1 data aggregation, number of rows in pd_val_1year = 1000 * 45 * 1 * 6 = 270,000 rows

4. Extending the previous step to 5 years, this would mean after 5 years results would be 5 * 270,000 = 1,350,000 rows in pd_val_5year.

   Scenario 2 strategy leads to 1,350,000 rows after 5 years.

Comparison of the scenarios

Scenario 1	Scenario 2
At the end of 5 years, there are 630,720,000,000 rows.	At the end of 5 years, there are 1,350,000 rows.
Provides the ability to look at raw data collected in the database.	Raw data is not available in the database. Raw data is still available using the Performance Grid.
Summary data could be inferred.	Summary data is readily obtained by storing weekly, monthly aggregates. It is possible to schedule SQL reports to get the data out before and after aggregation.
Storing raw data could limit the amount of data stored over the years.	Storing summary data allows for varying flexibility to store data for several years.
There is no manipulation of data using aggregation scripts. All operations done on raw data using SQL.	Uses a planned Data Aging and Aggregation strategy to aggregate data. It is necessary to schedule these aggregation scripts to run appropriately at end of week, end of month and end of year.
Database Administration tasks can take a long time.	Database Administration tasks are relatively simpler with manageable data.
Data upload and retrieval times can be affected by huge amounts of data.	Less data means faster load and retrieval times.

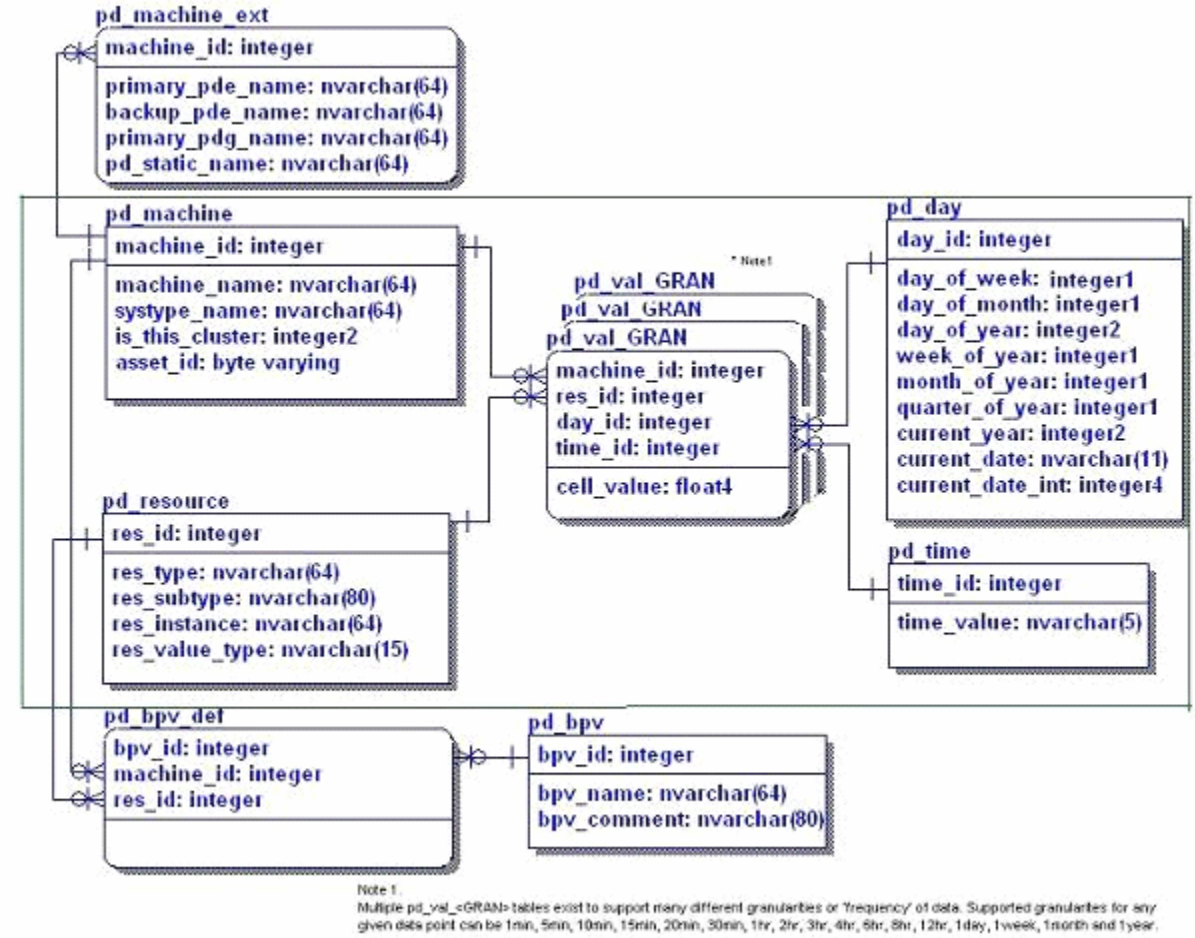
Conclusion

Although every enterprise has unique requirements, they all have the common need to store, analyze, and interpret useful data. There is no single strategy that works for all.

The Atemp corporation case highlights two extreme cases and shows what can be achieved (at what costs) by adopting different strategies. Consider the case as a guideline to work out a viable strategy for any given enterprise.

Schema and Reporting Information

Following is a simplified schema of the Systems Performance model in the MDB.



Microsoft SQL Server Databases

In this diagram, generic names represent the data types. Following are the mappings for the data type names that differ in Microsoft SQL Server.

- integer: int4
- integer2: smallint
- integer1: tinyint
- float: real
- byte varying: binary

Tables in the Schema

The tables used by Systems Performance are documented as follows. PK refers to Primary Index Key and SK refers to Secondary Index Key.

Microsoft SQL Server Databases

In these tables, generic names represent the data types. Following are the mappings for the data type names that differ in Microsoft SQL Server.

- integer: int4
- integer2: smallint
- integer1: tinyint
- float: real
- byte varying: binary

Table pd_machine

Contains a row for every machine.

Column	Definition	Description	Keys
machine_id	integer	Machine identifier	PK
machine_name	nvarchar(64)	Machine name	Unique constraint
systype_name	nvarchar(64)	System type name	
is_this_cluster	integer1	Cluster identifier. If this machine is a cluster parent, this flag is set to 1. The default is 0.	
asset_id	byte(16)	Common asset identifier	SK

Table pd_machine_ext

Contains information about machines available to the profile domain server. This table is populated using the Performance Domain Server. The table also provides information useful to Grid API users.

Any machine that exists in the pd_machine table with is_this_cluster=0, can have an entry in this table. It is not mandatory for every machine to be represented in this table.

Column	Definition	Description	Keys
machine_id	integer	Machine identifier	PK
primary_pd_name	nvarchar(64)	Primary PDE name	
backup_pde_name	nvarchar(64)	Backup PDE name	
primary_pdg_name	nvarchar(64)	Primary PDG name	
pd_static_name	nvarchar(64)	Static name	
last_dns_name	nvarchar(64)	Last DNS name	
last_dns_name_1	nvarchar(64)	Previous DNS name	
last_dns_name_2	nvarchar(64)	Previous DNS name	
last_dns_name_3	nvarchar(64)	Previous DNS name	
last_dns_name_4	nvarchar(64)	Previous DNS name	
last_dns_name_5	nvarchar(64)	Previous DNS name	
profile_name	nvarchar(50)	Profile name	
config_server	nvarchar(64)	Configuration server	
mac_address	nvarchar(18)	MAC address	
agent_version	integer	Agent version	
neugent_version	integer	Neugent version	
pp_version	integer	Profile server version reference	
pd_protocol	integer	Protocol used	
pd_date	nvarchar(26)	Profile server date reference	
pd_address	nvarchar(64)	Profile server address reference	
object_id_high	integer	Profile domain server high identifier	
object_id_low	integer	Profile domain server low identifier	

Table pd_cluster_ext

Allows extended properties for a cluster parent. Any machine that exists in the pd_machine table with is_this_cluster=1 can have an entry in this table. It is not mandatory for every cluster parent to be represented in this table.

Column	Definition	Description	Keys
machine_id	integer	Machine identifier	PK
cluster_alias	nvarchar(30)	Cluster alias name	
cluster_type	nvarchar(30)	Cluster type	

Table pd_cluster_def

Defines cluster parent-child relationships. Cluster parents are entries in the pd_machine table with is_this_cluster=1. Cluster children are entries in the pd_machine table with is_this_cluster-0.

Column	Definition	Description	Keys
parent_id	integer	Cluster parent ID. Should match pd_machine.machine_id with is_this_cluster=1.	PK10
child_id	integer	Cluster child ID. Should match pd_machine.machine_id with is_this_cluster=0.	PK11

Table pd_resource

Contains a row for every resource.

Column	Definition	Description	Keys
res_id	integer	Resource identifier	PK
res_type	nvarchar(64)	Resource type	Unique constraint (type, subtype, instance)
res_subtype	nvarchar(64)	Resource subtype	
res_instance	nvarchar(64)	Resource instance	
res_value_type	nvarchar(64)	Indicates what the resource value stands for. For example, AVERAGE, RATE, PERCENT, and so forth.	

Table pd_day

Contains a row for every day of the year.

Column	Definition	Description	Keys
day_id	integer	Day identifier	PK
day_of_week	integer1	Day of week. Sun = 1, Mon = 2, Tue = 3, Wed = 4, Thu = 5, Fri = 6, Sat = 7.	
day_of_month	integer1	Day of month. Range: 1 - 31 as in dd of dd/mm/yyyy.	
day_of_year	integer2	Day of year. Range: 1 - 366.	
week_of_year	integer1	Week of year. Range 0 - 52.	
month_of_year	integer1	Month of year. Range: 1 - 12 as in mm of dd/mm/yyyy. datepart(Mm, '31-Jan-2000') - SQL Server	
quarter_of_year	integer1	Quarter of year. Range: 1 - 4.	
current_year	integer2	Year as in yyyy.	
current_date	nvarchar(11)	Date without time as in 'yyyy-mm-dd'. Note this is not a date column.	SK1
current_date_int	integer	This is the same as current_date but stored as integer. The column stores time interval in seconds between 1 Jan 1970 and the current date. Use this for sorting dates sequentially or comparing dates.	SK2

Table pd_time

Contains 1440 rows representing every minute of a day starting from 0:00 up to 23:59. The contents of this table are fixed.

Column	Definition	Description	Keys
time_id	integer	Time identifier	PK

Column	Definition	Description	Keys
time_value	nvarchar(5)	Time stamp as in 'hh:mm'	

Table pd_val_GRAN

Contains an entry for every granular level data for every machine, resource and day. It is expected that this table will grow rapidly.

GRAN stands for 1min, 5min, 10min, 15min, 20min, 30min, 1hr, 2hr, 3hr, 4hr, 6hr, 8hr, 12 hr, 1day, 1week, 1month, and 1 year. All 17 tables have the same columns and data types as described.

Column	Definition	Description	Keys
machine_id	integer	Machine identifier	PK10
res_id	integer	Resource identifier	PK11
day_id	integer	Day identifier	PK12
time_id	integer	Time identifier	PK13
cell_value	float4	The point value. Nullable. If NULL, indicates no data collected for point.	

Table pd_bpv

Contains user-defined Business Process Views.

Column	Definition	Description	Keys
bpv_id	integer	BPV identifier	PK
bpv_name	nvarchar(64)	BPV name	Unique constraint

Table pd_bpv_def

Contains user-defined Business Process View definitions that link a Business Process View with machines and resources.

Column	Definition	Description	Keys
bpv_id	integer	BPV identifier	PK10
machine_id	integer	Machine identifier	PK11
res_id	integer	Resource identifier	PK12

Table pd_max_machine

Manages the maximum machine identifier. There is one row in this supporting table.

Column	Definition	Description
max_machine_id	integer	Maximum machine identifier used so far.

Table pd_max_resource

Manages the maximum resource identifier. There is one row in this supporting table.

Column	Definition	Description
max_res_id	integer	Maximum resource identifier used so far.

Table pd_max_day

Manages the maximum day identifier. There is one row in this supporting table.

Column	Definition	Description
max_day_id	integer	Maximum day identifier.

Table pd_max_bpv

Manages the maximum Business Process View identifier. There is one row in this supporting table.

Column	Definition	Description
max_bpv_id	integer	Maximum Business Process View identifier.

Table pd_global

Provides Performance Data specifics for the database. There is one row in the table.

Column	Definition	Description
major_version	integer	Major version number.
minor_version	integer	Minor version number.
build_version	integer	Build version number.

Column	Definition	Description
revision_version	integer	Revision version number.

Tuning and Performance Requirements

The Data Capacity Estimation and Topology or Implementation Scenarios topics have illustrated how the data in a database can grow quickly. This topic shows the performance issues that arise from high data growth. The number of rows in a data table can influence the following:

- Time it takes to load data. The more the data, the more time it takes to load.
- Time it takes to access data. The more the data, the time needed to access or retrieve data is relatively more (although this is negligible in queries that uses index properly).
- The disk space consumed. In addition to the tables, the indexes also consume disk space.

To ensure efficient and rapid access to the performance data you store in the MDB, it is important to manage your data. Consider the following best practices:

Only upload data to the MDB that you really need and will use in a reporting or analytical capacity. The example below illustrates the importance of this consideration.

Example 1

1000 machines — 1000 rows in pd_machine

250 resources — 250 rows in pd_resource

2 months of data — 60 rows in pd_day (approximately)

5 minute granularity — 288 rows in pd_time *

Total number of rows in pd_val_5min = 1000 * 250 * 60 * 288

= 43, 200, 000, 000 rows

= 43 billion rows

Example 2

1000 machines	1000 rows in pd_machine
20 resources	20 rows in pd_resource
1 year of data	365 rows in pd_day (approximately)
4 hour granularity	6 rows in pd_time *

Total number of rows in pd_val_4hr = 1000 * 20 * 365 * 6
= 43, 800, 000 rows
= 43 million rows

Even though Example 2 is able to store an entire year's worth of data (where Example 1 only stores two months), Example 2 only uses 0.1% of the rows used by Example1. By carefully selecting the performance metrics that are relevant to your reporting requirements and by choosing to load less granular data, the gains in terms of scalability and space usage can be immense.

In addition to an appropriate data load or storage policy, consider the following points to ensure optimum performance:

- Ensure the database is optimized for data load. See the section "Implement an Indexing Strategy" under the Systems Performance topic Backup, Administration, and Maintenance Requirements for more information.
- Loading data into a local database is faster than loading data into a remote database.
- Use higher granularities of data to populate the database. The Performance Domain Server populates the MDB with a default granularity of 4 hours.
- Store only data you really need. Use Data Aging scripts that come with Systems Performance to purge old data.

Backup, Administration, and Maintenance Requirements

The fundamental administrative considerations for Systems Performance data are as follows:

- Indexing strategy implementation
- Data aging and aggregation strategy implementation
  - Obsolete data deletion
  - Data aggregation
- Data backup and restoration
- How you keep the database in optimal state (Ingres)

Indexing Strategy Implementation

Keep the indexes optimal. Use the script **reindex.sql** available under %*CASP_PATH*%/bin/utilities/scripts ($*CASP_PATH* for non-Windows) directory to optimize the indexes. We recommend that you run the script whenever large amounts of data are added to the database. As best practice, run the script every night. Ensure no other process uses the tables involved when you run the scripts.

You can run the script in the following ways (use $*CASP_PATH* for non-Windows):

Ingres Databases

- On a local database:

  ```
  sql DBNAME-umdbadmin < %CASP_PATH%/bin/utilities/scripts/reindex.sql
  ```

- On a remote database:

  ```
  sql vnode::DBNAME -umdbadmin < %CASP_PATH%/bin/utilities/scripts/reindex.sql
  ```

Microsoft SQL Server Databases

- On a local database:

  ```
  osql -d DBNAME-E -e < %CASP_PATH%/bin/utilities/scripts/reindex.sql
  ```

- On a remote database:

  ```
  osql -s DBSERVER -d DBNAME -E -e <
  %CASP_PATH%/bin/utilities/scripts/reindex.sql
  ```

Note: These examples assume that the user running the command has permission to access the database.

In large installations, it is possible that there is not much downtime to run the script or there may be resource constraints because of the amount of data stored. In these cases, you can choose to split the SQL into two files and schedule them. For Ingres, you can run the script involving tables with hash index more frequently than the ones that use btree.

Data Aging and Aggregation Strategy Implementation

An appropriate data aging and aggregation strategy is critical to ensure the well-being and efficient operation of Performance Data in the MDB. To assist with the creation of data aging and aggregation policies, Unicenter NSM supplies a set of administration scripts with the Systems Performance tools. These can be found under the directory %*CASP_PATH*%bin/utilities/scripts *($CASP_PATH* on non-Windows).

The scripts and their usage are documented in the product documentation. In the topics that follow, you can review the scripts and learn what to watch for when using these scripts.

Obsolete Data Deletion

When you want to delete obsolete data, use the scripts located in %*CASP_PATH*%bin/utilities/scripts ($*CASP_PATH* on non-Windows) that start with delall. These scripts can support a single granularity, or if granularity is not provided, they support all granularities. The scripts provided to delete data include the following:

delallbyage

Deletes all data before a given date.

delallresbyage

Deletes all data before a given date for the requested Resources.

delallmachbyage

Deletes all data before a given date for the requested Machines.

dellallmachresbyage

Deletes all data before a given date for the requested Machines and Resources.

If the deletion scripts fail because of insufficient database or OS resources, you can just rerun the scripts after fixing the resource problem. Since the above scripts merely delete data, any failures in the log file can cause little damage.

Data Aggregation

Additional scripts located in %*CASP_PATH*%bin/utilities/scripts ($*CASP_PATH* on non-Windows) let you aggregate lower granular data to a higher granularity and at the same time let you delete the lower granularity data. These scripts include the following:

aggbyage

Aggregates data by age and then removes the lower granularity data.

aggmachbyage

Aggregates data by age for the requested Machines.

aggmachresbyage

Aggregates data by age for the requested Machines and Resources.

For the aggregation scripts to work, you must run a simple setup operation once for each database against which the scripts may be used. This ensures that certain helper tables are set up to support the various aggregation operations that you plan to use. Run the following command once for each database that needs the aggregation scripts:

Ingres Databases

```
sql mdb -umdbadmin < create_group_tables.sql
```

Microsoft SQL Server Databases

```
osql -s DBSERVER -d DBNAME -E -e <create_group_tables.sql
```

If you want to destroy the helper tables that are created when you run the script, use the following command instead:

Ingres Databases

```
sql mdb -umdbadmin < drop_group_tables.sql
```

Microsoft SQL Server Databases

```
osql -s DBSERVER -d DBNAME -E -e <drop_group_tables.sql
```

If the aggregation scripts fail because of insufficient database or OS resources, you may want to restore data from a backup and rerun the scripts after fixing the resource problem.

Data Backup and Restoration

Microsoft SQL Server Databases

Microsoft SQL Server provides the TSQL statements BACKUP and RESTORE to back up and restore your data.

See the Microsoft SQL Server online books for information about using the statements.

Ingres Databases

Ingres provides the utilities ckpdb and rollforwarddb to back up and restore your data.

Ingres also provides the utilities unloaddb and copydb to back up or archive and restore data. These utilities copy user data into flat files.

See the *Ingres Database Administrator Guide* for information about using the utilities.

How You Keep the Database in Optimal State (Ingres)

Ingres provides the optimizedb utility to optimize the data in the database and keep the statistics up-to-date. We recommend that you run the utility after large data uploads. For Systems Performance, consider the following sequence of actions:

1. After large Performance Data uploads, re-index the Systems Performance tables using the information provided in the Implement an Index Strategy topic.
2. Run the following command:

   ```
   optimizedb -umdbadmin -zk mdb
   ```

   This command attempts to optimize the data for the key columns in the whole of the MDB.

   To restrict the command to Systems Performance tables only, use the command with the *-rTableName* option.

   For example, if you know the Performance Data is only stored as 4-hour granularity (default), then you can run the following command to optimize only the key tables used by Systems Performance (a command spanning many lines should be typed as one line):

   ```
   optimizedb -umdbadmin -zk mdb -rpd_machine -rpd_resource -rpd_day -rpd_time -rpd_val_4hr
   ```

3. Run the following command:

   ```
   sysmod -umdbadmin mdb
   ```

Integration Considerations

Integration and configuration tasks are required to let users and applications access the performance data held in the MDB.

How You Allow Users Read/Write Access to Performance Data

The following example demonstrates how to set up an OS and database user with read/write access to just the Performance-related data in the MDB.

This setup is only needed if the default user setup for Systems Performance access needs to be changed after Systems Performance is installed. This setup applies to accessing Performance data using Performance Domain Server.

In this example, the user perfadmin needs to be given access to just the Systems Performance-related data in the MDB. This user does not need access to any other data in the MDB. Follow these steps:

Microsoft SQL Server Databases

1. Create a new user named perfadmin on the MDB server machine using Microsoft SQL Server Enterprise Manager.
2. Set the default database role for perfadmin as pdadmin.
3. Ensure the new user is added to the database role regadmin using the Microsoft SQL Server Enterprise Manager:

   ```
   sp addrolemember 'regadmin', 'perfadmin'
   ```

4. Since this user's credentials must be known to all Performance Domain Servers connecting to the MDB, run the cfgutil utility on all Performance Domain Server machines that would use this user and amend the credentials information as described in the topic How You Register Credentials for Systems Performance Use that follows.

Ingres Databases

1. Create a new OS user named perfadmin on the MDB server machine.
2. Create a new Ingres user named perfadmin on the MDB server machine using the Ingres utility accessdb.
3. Set the default Ingres group for perfadmin as pdadmin in accessdb.
4. Ensure this Ingres user is also added to the Ingres group regadmin as a valid user. This cannot be achieved from accessdb, but can be done by one of the following ways:
   - Using vdba - Connect DOM - Groups (select regadmin) - Alter (select perfadmin).
   - Running the following SQL statement on the MDB server against the iidbdb database:

     ```
     alter group regadmin add users (perfadmin)
     ```

5. Since this user's credentials must be known to all Performance Domain Servers connecting to the MDB, run the cfgutil utility on all Performance Domain Server machines that would use this user and amend the credentials information as described in the topic How You Register Credentials for Systems Performance Use that follows.

How You Allow Users Read Only Access to Performance Data

The following example demonstrates how to set up an OS and Ingres user to read just the Performance-related data in the MDB.

This setup strictly applies to SQL users only and cannot be used to access data using Systems Performance applications.

In this example, the user perfread needs to be permitted to read just the Systems Performance data in the MDB. Follow these steps:

Microsoft SQL Server Databases

1. Create a new user named perfread on the MDB server machine using Microsoft SQL Server Enterprise Manager.
2. Set the default database role for perfread as pduser in Microsoft SQL Server Enterprise Manager.

Ingres Databases

1. Create a new OS user named perfread on the MDB server machine.
2. Create a new Ingres user named perfread on the MDB server machine using the Ingres utility accessdb.
3. Set the default Ingres group for perfread as pduser in accessdb.

How You Register Credentials for Systems Performance Use

Performance Domain Server requires certain critical information to connect to the MDB. This information must be kept up-to-date if any changes occur after installation.

- MDB server name.
- Protocol needed to connect to the MDB. Use tcp_ip for Windows, Linux and UNIX clients needing access to MDB. (Ingres only).
- The installation ID on the mdb server.

  For Microsoft SQL Server databases, use the Microsoft SQL Server Enterprise Manager to retrieve the instance name.

  For Ingres databases, get this by running ingprenv II_INSTALLATION on the MDB server.

- The user name that has access to the Systems Performance Data on the MDB server. This user should have valid read/write access to Systems Performance-related data in the MDB as documented in this topic.
- The OS password for the user on the MDB server. (Ingres only).
- The database password for the user on the MDB server. (Microsoft SQL Server only).
- The database name (for example, mdb).

All the information is collected and stored once during the Systems Performance installation in the file %*CASP_PATH*%/appdata/domainserver/data/dbcred.dat ($*CASP_PATH* for non-Windows) using PKI encryption.

However, there are two circumstances when this information could become out-of-date:

- When a new or additional user needs read/write access to the MDB.
- When any of the information (password, for example) changes.

To allow easy management of the MDB credentials for Systems Performance, the utility cfgutil provides the option to manage the credentials file after installation.

For example, to connect to an MDB using the following credentials:

Ingres Databases

MDB Server name: sorcerer

Protocol: tcp_ip

Installation id : EI

User: perfadmin

Password: password

Database name: mdb

Following is an example command:

```
cfgutil -a DBType=Ingres Databasename=mdb PerfDBUserID=perfadmin
PerfDBPasswd=password ServerID=sorcerer ListenerID=EI Protocol=tcp_ip
```

Microsoft SQL Server Databases

MDB Server name: sorcerer

User: perfadmin

Password: password

Database name: mdb

Following is an example command:

```
cfgutil -a DBType=MS-SQL Databasename=mdb PerfDBUserID=perfadmin
PerfDBPasswd=password ServerID=sorcerer
```

Unicenter Management Portal

Unicenter MP uses the MDB in the following two ways:

- Stores and retrieves data from Portal tables defined in the MDB.
- Retrieves data stored by other Unicenter products and visualize this data in forms or reports, scoreboards, and other portlets.

Access to Unicenter MP Tables in the MDB

Unicenter MP uses JDBC to directly access Portal tables in the MDB. Examples of information that portal stores in the MDB include definitions of documents, channels published in the portal, workplaces, definitions of event and alert filters, and so forth.

Access to Enterprise Management Data in the MDB

When accessing enterprise management data stored in the MDB by other Unicenter products, Unicenter MP uses a set of credentials provided by the Portal Administrator when defining its connections to this Unicenter product or the product's sub-components. Defining connections to Unicenter products and components is one of the first post-installation administration tasks the Portal Administrator must complete before the portal can retrieve any Unicenter data.

Unicenter MP provides an Administrator wizard portlet to assist the Portal Administrator through the process of defining or modifying connections to Unicenter components. For more information about the Administration Wizard, see the Unicenter MP Administration online help. Since the portal supports a separate connection for each Unicenter product or component, the Portal Administrator can specify one set credentials for accessing WorldView data and another set to access Network Performance data. The same Administrator wizard can be used to change the credentials for accessing the MDB. For more information about the Administration wizard, see the Unicenter MP Administration online Help.

Unicenter MP uses one of the following methods to access data from Unicenter products stored in the MDB:

- Distributed Intelligent Architecture (DIA)
- JDBC

For example, WorldView data is always accessed using DIA and Network Performance Data is always accessed using JDBC. Both methods of accessing data require providing appropriate credentials. These credentials are verified by database security.

JDBC Usage

Since Unicenter MP uses JDBC, installing a database client on the Unicenter MP server is not required. The database JDBC driver is installed during the Unicenter MP installation.

Ingres Databases

If your MDB is running on Ingres, the Ingres Data Access server must be up and running on the MDB server for the JDBC connection to work.

Security

Unicenter MP lets you specify a different set of credentials when accessing Portal tables and tables owned by other Unicenter products.

Access to Portal Data in the MDB

To function properly, Unicenter MP must be able to access Portal tables with credentials that have read/write access to Portal tables in MDB. The user ID and password that Unicenter MP uses to access Portal tables in the MDB are specified by the user during the Unicenter MP installation.

We recommend that to grant a databases user the required access rights to portal tables is to make the user a part of the predefined database groups or roles that are defined as part of MDB:

- Any user associated with the database group or role UNIADMIN will have read/write rights to access portal tables and other NSM tables.
- Any user associated with database group or role UMPADMIN will have read/write rights to access portal tables only.

If the MDB is installed on the same server as Unicenter MP, the Unicenter MP installation handles the creation of the database user specified by the installer and ensures that this user is part of the UNIADMIN group or role. If the MDB is installed on a remote server, you must specify the correct database credentials; Unicenter MP does not create or modify OS or database users on the remote system. For more details on the Unicenter MP installation process, see the *Unicenter Management Portal Getting Started*.

Customers that need read only access for Portal data only in the MDB can associate a known user to this Ingres group after installation.

Any user associated with the database group or role UMPUSER has read only rights to Unicenter MP tables. Users that need read only access for Portal data only in the MDB can associate a known user to this database group or role after installation.

Access to Enterprise Management Data in the MDB

When accessing data stored in the MDB by Unicenter products, Unicenter MP uses credentials provided by the Portal Administrator when defining connections to a particular Unicenter product or product component. This set of credentials is used to authenticate and authorize the user on the MDB server. Consult your product-specific guide to determine what credentials are required by the different Unicenter NSM components to access its component-specific data.

Data Capacity Estimation

The growth of Unicenter MP tables depend on the new content that you add to Unicenter MP. The following table provides examples of how some of the key Portal tables can grow.

New Content	is equal to:
1 portal document	1 por_objectrepos row
1 portal channel	1 por_objectrepos row
1 user	1 por_user row
1 group	1 por_group row
1 ump notification	1 ump_message row
1 security profile	1 ump_profile row

Topology or Implementation Scenarios

Unicenter MP access to the MDB can be supported for both local and remote MDB servers.

In a typical installation, the MDB is not shared between multiple installations of Unicenter MP. However, it is possible to configure multiple installations of Unicenter MP to use the same MDB and share a common database and a common file system. When using this configuration, each portal is independent. If a portal fails for any reason, the content is still accessible from any of the remaining portals. This setup is used primarily when you need high-availability portals and requires some post-installation configuration steps. For more information on how set up multiple installations of Unicenter MP with a shared non-distributed portal repository, see the *Unicenter Management Portal Getting Started*.

Asset Registration

Asset Registration is a common component for CA products that provides an interface to register assets in the MDB. It hides the complexity of the asset registration process and guarantees its integrity and consistency.

The primary function of asset registration is the reconciliation of IT asset information coming from several data sources into distinct physical hardware assets.

The benefits that Asset Registration provides as a common component can be summed up as follows:

- Asset reconciliation contributes to the reduction of complexity in the IT enterprise infrastructure by preventing duplicate asset data.
- The rules for asset reconciliation are hidden from higher-level applications and are guaranteed to be consistent across all consuming CA applications or customized extensions that are deployed in the enterprise because asset registration is a cross-platform, embedded component. This gives a common view of assets available in the MDB in terms of physical or logical computers and licensed or used software.
- The value of CA product integration in the enterprise increases because the asset reconciliation process becomes more accurate as more products participate in asset registration.
- The referential integrity of the asset registration tables in the MDB is enforced.

Security

The Asset Registration component uses the REGADMIN group defined in the MDB to restrict the scope of its access to the database. The scope of its access include the CA common tables—most significantly the asset registration tables.

In general, the default table and procedure level grants given to the REGADMIN group should not be modified. Use default groups and permissions.

Index

A

accessing portal data • 99
Administrator wizard portlet • 98
aging and aggregation strategy • 92
allow read only access • 96
allow read/write access • 95
API • 75
Archive Logging • 51
archive logs • 46
Asset Registration • 101
AUTOEXTEND, use of • 49
Automatic Storage Management (ASM) • 46

B

back up and restore data • 94
backup and recovery plan • 43
block size, database • 48

C

CA MDB, description of • 9
case sensitivity • 41
character sets, use of • 50
ckpdb • 94
cluster awareness • 73
cluster support, active • 18
command line utilities, using • 20
Common Asset tables • 58
configuration options, Microsoft SQL Server • 40
connections, number of concurrent • 39
copydb utility • 94

D

data capacity • 61
database file management • 49
database objects • 21
dedicated server mode • 50
deployment • 15
disk mirroring • 46
disk space, monitoring • 43
dynamic vnodes • 74

E

extending the MDB schema • 21

F

fragmented indexes • 43

G

global database name, description of • 49

H

high availability • 73
high data growth, effects • 89

I

implementation example • 78
indexing strategy, MDB • 91
Ingres support • 9
integrated view • 16
integration considerations • 58

J

JDBC • 98
job scheduler, using a • 20

K

key table datafiles • 46

M

maintenance activities • 20
maintenance process, automating the • 20
managing data capacity • 76
master database • 43
maximum worker threads setting, increasing the • 39

N

new users, defining • 20

O

object names, duplicate • 21
optimizedb command • 58, 94

P

Performance Domain Server • 74
performance requirements • 58
place redo log files • 46

planning for availability • 17
portal table growth • 100
processes parameter • 50

R

RAID • 46
redo log files • 50
REGADMIN group • 102
register assets • 101
register credentials • 96
reindex script • 91
removing an Oracle MDB • 56
replication, description of • 18
report writer • 20
rollforwarddb utility • 94

S

schema upgrades • 21
scripts to aggregate data • 93
scripts to delete data • 92
security • 59
SQL commands • 20
storage considerations, Oracle • 45
Systems Performance tables • 83

T

tablespaces, use of • 48
tempdb database, use of the • 39

U

Unicenter MP in the MDB • 98
unloaddb utility • 94

V

vnodes • 74

W

Worldview tables, access • 75