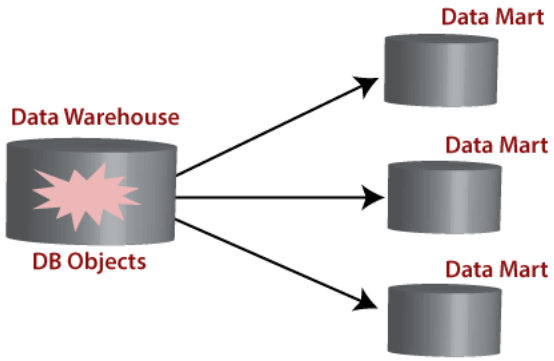
# UNIT 5

## What is Data Mart?

A **Data Mart** is a subset of a directorial information store, generally oriented to a specific purpose or primary data subject which may be distributed to provide business needs. Data Marts are analytical record stores designed to focus on particular business functions for a specific community within an organization. Data marts are derived from subsets of data in a data warehouse, though in the bottom-up data warehouse design methodology, the data warehouse is created from the union of organizational data marts.

The fundamental use of a data mart is **Business Intelligence (BI)** applications. **BI** is used to gather, store, access, and analyze record. It can be used by smaller businesses to utilize the data they have accumulated since it is less expensive than implementing a data warehouse.




## Reasons for creating a data mart

- Creates collective data by a group of users
- Easy access to frequently needed data
- Ease of creation
- Improves end-user response time
- Lower cost than implementing a complete data warehouses
- Potential clients are more clearly defined than in a comprehensive data warehouse
- It contains only essential business data and is less cluttered.

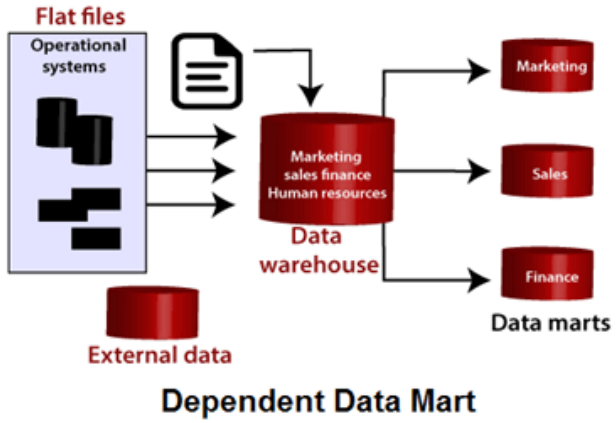
## Types of Data Marts

There are mainly two approaches to designing data marts. These approaches are

- Dependent Data Marts
- Independent Data Marts

## Dependent Data Marts

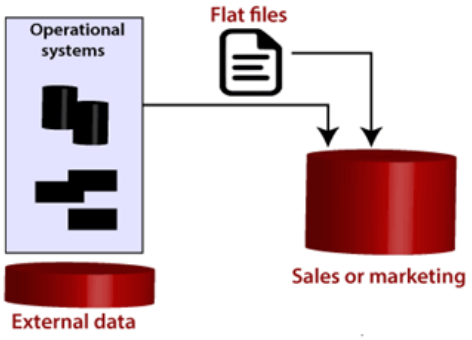
A dependent data marts is a logical subset of a physical subset of a higher data warehouse. According to this technique, the data marts are treated as the subsets of a data warehouse. In this technique, firstly a data warehouse is created from which further various data marts can be created. These data mart are dependent on the data warehouse and extract the essential record from it. In this technique, as the data warehouse creates the data mart; therefore, there is no need for data mart integration. It is also known as a **top-down approach**.



Dependent Data Mart

**Independent Data Marts**

The second approach is Independent data marts (IDM) Here, firstly independent data marts are created, and then a data warehouse is designed using these independent multiple data marts. In this approach, as all the data marts are designed independently; therefore, the integration of data marts is required. It is also termed as a **bottom-up approach** as the data marts are integrated to develop a data warehouse.



Independent Data Mart

Other than these two categories, one more type exists that is called "**Hybrid Data Marts**."

**Hybrid Data Marts**

It allows us to combine input from sources other than a data warehouse. This could be helpful for many situations; especially when Adhoc integrations are needed, such as after a new group or product is added to the organizations.

**Steps in Implementing a Data Mart**

The significant steps in implementing a data mart are to design the schema, construct the physical storage, populate the data mart with data from source systems, access it to make informed decisions and manage it over time. So, the steps are:

**Designing**

The design step is the first in the data mart process. This phase covers all of the functions from initiating the request for a data mart through gathering data about the requirements and developing the logical and physical design of the data mart.

It involves the following tasks:

1. Gathering the business and technical requirements
2. Identifying data sources
3. Selecting the appropriate subset of data
4. Designing the logical and physical architecture of the data mart.

**Constructing**

This step contains creating the physical database and logical structures associated with the data mart to provide fast and efficient access to the data.

It involves the following tasks:

1. Creating the physical database and logical structures such as tablespaces associated with the data mart.
2. creating the schema objects such as tables and indexes describe in the design step.
3. Determining how best to set up the tables and access structures.

**Populating**

This step includes all of the tasks related to the getting data from the source, cleaning it up, modifying it to the right format and level of detail, and moving it into the data mart.

It involves the following tasks:

1. Mapping data sources to target data sources
2. Extracting data
3. Cleansing and transforming the information.
4. Loading data into the data mart
5. Creating and storing metadata

**Accessing**

This step involves putting the data to use: querying the data, analyzing it, creating reports, charts and graphs and publishing them.

It involves the following tasks:

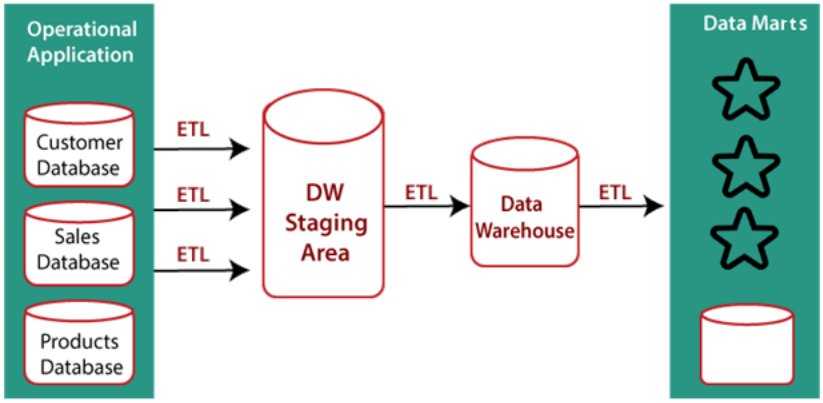
1. Set up and intermediate layer (Meta Layer) for the front-end tool to use. This layer translates database operations and objects names into business conditions so that the end-clients can interact with the data mart using words which relates to the business functions.
2. Set up and manage database architectures like summarized tables which help queries agree through the front-end tools execute rapidly and efficiently.

**Managing**

This step contains managing the data mart over its lifetime. In this step, management functions are performed as:

1. Providing secure access to the data.
2. Managing the growth of the data.
3. Optimizing the system for better performance.
4. Ensuring the availability of data event with system failures.

Difference between Data Warehouse and Data Mart




| | |
|---|---|
| A Data Warehouse is a vast repository of information collected from various organizations or departments within a corporation. | A data mart is an only subtype of a Data Warehouses. It is architecture to meet the requirement of a specific user group. |

| | |
|---|---|
| It may hold multiple subject areas. | It holds only one subject area. For example, Finance or Sales. |
| It holds very detailed information. | It may hold more summarized data. |
| Works to integrate all data sources | It concentrates on integrating data from a given subject area or set of source systems. |
| In data warehousing, Fact constellation is used. | In Data Mart, Star Schema and Snowflake Schema are used. |
| It is a Centralized System. It is a Decentralized System. | |
| Data Warehousing is the data-oriented. | Data Marts is a project-oriented. |

**DESIGN COST ESTIMATION**

There are several components to consider.

- Initial set up cost
  - Cloud data warehouse service
  - Implementation cost
  - Data warehouse administration, operation and optimization training
- On-going operational costs
  - Storage
  - Compute
  - Network charges
  - Premium technical support
- Miscellaneous costs
  - Data ingestion tool or service
  - Data warehouse automation tools/service
  - Business intelligence tool or service
  - Other data applications using the warehouse data

**Top Cloud Data Warehouse Pricing Structures**

Although there are dozens of well-known cloud-based data warehouse vendors, the following is a brief overview of the pricing structures of the top vendors.

***Amazon Redshift***
Amazon advises that you first choose a cluster and node type configuration to suit your needs. Don't worry if you want to change, as you can easily scale your nodes or switch between node types with a single API call or a few clicks in the Amazon Redshift console. Amazon offers two types of pricing model: on-demand and reserved instances. Reserved instances typically have a larger discount; however, they require a bigger up-front investment. Amazon offers three types of Redshift nodes: RA3, DC2, DS2 – each has an optimal use case.

If your data warehouse is likely to be less than 1 TB, Amazon suggests you choose DC2 nodes, which run from $0.25/hour to $4.80/hour. If you require lots of storage, then they suggest using RA3 nodes with managed storage. RA3 nodes cost from $0.85/hour to $6.80/hour, with an additional charge of $0.024 per GB/month for managed storage.

***Azure Synapse***

Azure Synapse is a limitless analytics service that brings together enterprise data warehousing and Big Data analytics. It gives you the freedom to query data, using either serverless or provisioned resources. Data storage is charged at the rate of $122.88 per TB of data processed ($0.17/1 TB/hour). Data storage includes the size of your data warehouse and seven days of incremental snapshot storage.

Azure Synapse SQL has its own method for calculating compute resources called Data Warehouse Units (DWU). Azure has a sliding scale from 100 DWUs costing $1.20/hour to 30,000 DWUs costing $360/hour.

***Google BigQuery***
The two main components of Google BigQuery pricing are storage and compute. Storage has 2-tiered pricing.

- **Active** – A monthly charge for data stored in tables that have been modified in the prior 90 days.
- **Long-term** – A lower monthly charge for tables not accessed in the prior 90 days.

Active storage starts $0.02 per GB/month. The first 10GB is free each month. Any data that isn't accessed for 90 days is automatically moved to long-term storage, which costs $0.01/GB/month.

Compute usage is called Query pricing and refers to the cost of running SQL commands, user-defined functions, and qualifying Data Manipulation Language and Data Definition Language statements. Query pricing also has two pricing models

- **On-demand** – You only pay for the queries you run.
- **Flat-rate pricing** – Offered in per-second, monthly or annual commitments.

On-demand pricing charges for the number of bytes processed. Pricing starts at $5 per TB/month; the first TB is free.

***Snowflake***
The Snowflake architecture separates data warehousing into three distinct layers: storage, virtual warehouses (compute) and cloud services. Snowflake pricing is based on the actual usage of these layers and of serverless features.

All customers are charged a monthly fee for the data they store in Snowflake, and it's based on the average amount of storage used per month. A virtual warehouse is one or more compute clusters that enable customers to load data and perform queries. Customers pay for virtual warehouses using Snowflake credits.

How much a virtual warehouse costs depends on size. The smallest is XS and costs one credit per hour. A 4XL virtual warehouse costs 128 credits per hour. The cloud services layer provides all permanent state management and overall coordination of Snowflake. You also pay for cloud services using Snowflake credits at a 10 percent discount rate.

Snowflake is different from the other vendors since it doesn't run on its own infrastructure. You have the choice to run on Amazon, Azure or Google. As a result, you should be aware that there's some slight variation in pricing between underlying cloud infrastructure vendors. Qlik offers a free QlikSense app to help Snowflake users understand their usage costs. For more details, follow this link.

**Cloud Data Warehouse Pricing Summary**

| Data Warehouse | Price Structure | Pricing Web Page |
|---|---|---|
| **Amazon Redshift** | • Number of Nodes<br>• Cluster Price<br>• Storage Volume | https://aws.amazon.com/redshift/pricing/ |
| **Azure Synapse** | • Storage Volume<br>• Data Warehouse "Units" | https://azure.microsoft.com/en-us/pricing/details/synapse-analytics/ |
| **Google Big Query** | • Storage Volume<br>• Query Volume (# of bytes processed) | https://cloud.google.com/bigquery/pricing |
| **Snowflake** | • Storage Volume<br>• Compute - Credits<br>• Cloud Services - Credits | https://www.snowflake.com/pricing/ |

**META DATA**

**What is Metadata?**

Metadata is simply defined as data about data. The data that is used to represent other data is known as metadata. For example, the index of a book serves as a metadata for the contents in the book. In other words, we can say that metadata is the summarized data that leads us to detailed data. In terms of data warehouse, we can define metadata as follows.

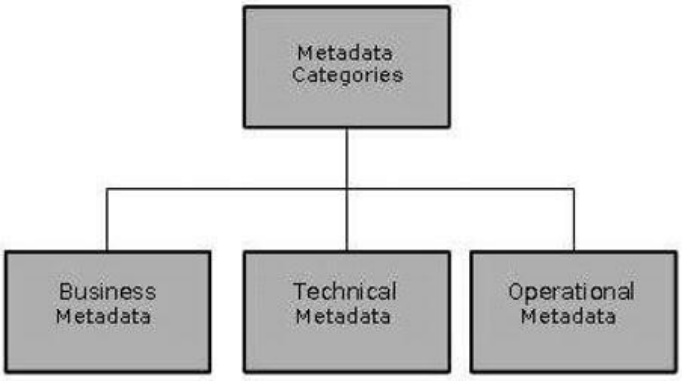
- Metadata is the road-map to a data warehouse.
- Metadata in a data warehouse defines the warehouse objects.
- Metadata acts as a directory. This directory helps the decision support system to locate the contents of a data warehouse.

**Note** − In a data warehouse, we create metadata for the data names and definitions of a given data warehouse. Along with this metadata, additional metadata is also created for time-stamping any extracted data, the source of extracted data.

**Categories of Metadata**

Metadata can be broadly categorized into three categories −

- **Business Metadata** − It has the data ownership information, business definition, and changing policies.
- **Technical Metadata** − It includes database system names, table and column names and sizes, data types and allowed values. Technical metadata also includes structural information such as primary and foreign key attributes and indices.
- **Operational Metadata** − It includes currency of data and data lineage. Currency of data means whether the data is active, archived, or purged. Lineage of data means the history of data migrated and transformation applied on it.

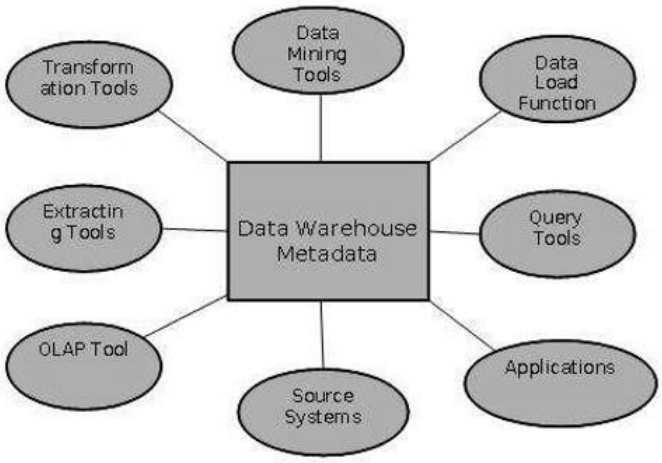



**Role of Metadata**

Metadata has a very important role in a data warehouse. The role of metadata in a warehouse is different from the warehouse data, yet it plays an important role. The various roles of metadata are explained below.

- Metadata acts as a directory.
- This directory helps the decision support system to locate the contents of the data warehouse.
- Metadata helps in decision support system for mapping of data when data is transformed from operational environment to data warehouse environment.
- Metadata helps in summarization between current detailed data and highly summarized data.
- Metadata also helps in summarization between lightly detailed data and highly summarized data.
- Metadata is used for query tools.
- Metadata is used in extraction and cleansing tools.
- Metadata is used in reporting tools.
- Metadata is used in transformation tools.
- Metadata plays an important role in loading functions.

The following diagram shows the roles of metadata.




**Metadata Repository**

Metadata repository is an integral part of a data warehouse system. It has the following metadata −

- **Definition of data warehouse** − It includes the description of structure of data warehouse. The description is defined by schema, view, hierarchies, derived data definitions, and data mart locations and contents.
- **Business metadata** − It contains has the data ownership information, business definition, and changing policies.
- **Operational Metadata** − It includes currency of data and data lineage. Currency of data means whether the data is active, archived, or purged. Lineage of data means the history of data migrated and transformation applied on it.
- **Data for mapping from operational environment to data warehouse** − It includes the source databases and their contents, data extraction, data partition cleaning, transformation rules, data refresh and purging rules.
- **Algorithms for summarization** − It includes dimension algorithms, data on granularity, aggregation, summarizing, etc.

**Challenges for Metadata Management**

The importance of metadata cannot be overstated. Metadata helps in driving the accuracy of reports, validates data transformation, and ensures the accuracy of calculations. Metadata also enforces the definition of business terms to business end-users. With all these uses of metadata, it also has its challenges. Some of the challenges are discussed below.

- Metadata in a big organization is scattered across the organization. This metadata is spread in spreadsheets, databases, and applications.
- Metadata could be present in text files or multimedia files. To use this data for information management solutions, it has to be correctly defined.
- There are no industry-wide accepted standards. Data management solution vendors have narrow focus.
- There are no easy and accepted methods of passing metadata.

**Benefits of Metadata Repository**

1. It provides a set of tools for enterprise-wide metadata management.
2. It eliminates and reduces inconsistency, redundancy, and underutilization.
3. It improves organization control, simplifies management, and accounting of information assets.
4. It increases coordination, understanding, identification, and utilization of information assets.
5. It enforces CASE development standards with the ability to share and reuse metadata.
6. It leverages investment in legacy systems and utilizes existing applications.
7. It provides a relational model for heterogeneous RDBMS to share information.
8. It gives useful data administration tool to manage corporate information assets with the data dictionary.
9. It increases reliability, control, and flexibility of the application development process.

**BACKUP**

A data warehouse is a complex system and it contains a huge volume of data. Therefore it is important to back up all the data so that it becomes available for recovery in future as per requirement. In this chapter, we will discuss the issues in designing the backup strategy.

**Backup Terminologies**

Before proceeding further, you should know some of the backup terminologies discussed below.

- **Complete backup** − It backs up the entire database at the same time. This backup includes all the database files, control files, and journal files.
- **Partial backup** − As the name suggests, it does not create a complete backup of the database. Partial backup is very useful in large databases because they allow a strategy whereby various parts of the database are backed up in a round-robin fashion on a day-to-day basis, so that the whole database is backed up effectively once a week.
- **Cold backup** − Cold backup is taken while the database is completely shut down. In multi-instance environment, all the instances should be shut down.
- **Hot backup** − Hot backup is taken when the database engine is up and running. The requirements of hot backup varies from RDBMS to RDBMS.
- **Online backup** − It is quite similar to hot backup.

**Hardware Backup**

It is important to decide which hardware to use for the backup. The speed of processing the backup and restore depends on the hardware being used, how the hardware is connected, bandwidth of the network, backup software, and the speed of server's I/O system. Here we will discuss some of the hardware choices that are available and their pros and cons. These choices are as follows −

- Tape Technology
- Disk Backups

**Tape Technology**

The tape choice can be categorized as follows −

- Tape media
- Standalone tape drives
- Tape stackers
- Tape silos

**Tape Media**

There exists several varieties of tape media. Some tape media standards are listed in the table below −

| Tape Media | Capacity | I/O rates |
|---|---|---|
| DLT | 40 GB | 3 MB/s |
| 3490e | 1.6 GB | 3 MB/s |
| 8 mm | 14 GB | 1 MB/s |

Other factors that need to be considered are as follows −

- Reliability of the tape medium
- Cost of tape medium per unit
- Scalability
- Cost of upgrades to tape system
- Cost of tape medium per unit
- Shelf life of tape medium

**Standalone Tape Drives**

The tape drives can be connected in the following ways −

- Direct to the server
- As network available devices
- Remotely to other machine

There could be issues in connecting the tape drives to a data warehouse.

- Consider the server is a 48node MPP machine. We do not know the node to connect the tape drive and we do not know how to spread them over the server nodes to get the optimal performance with least disruption of the server and least internal I/O latency.
- Connecting the tape drive as a network available device requires the network to be up to the job of the huge data transfer rates. Make sure that sufficient bandwidth is available during the time you require it.
- Connecting the tape drives remotely also require high bandwidth.

**Tape Stackers**

The method of loading multiple tapes into a single tape drive is known as tape stackers. The stacker dismounts the current tape when it has finished with it and loads the next tape, hence only one tape is available at a time to be accessed. The price and the capabilities may vary, but the common ability is that they can perform unattended backups.

**Tape Silos**

Tape silos provide large store capacities. Tape silos can store and manage thousands of tapes. They can integrate multiple tape drives. They have the software and hardware to label and store the tapes they store. It is very common for the silo to be connected remotely over a network or a dedicated link. We should ensure that the bandwidth of the connection is up to the job.

**Disk Backups**

Methods of disk backups are −

- Disk-to-disk backups
- Mirror breaking

These methods are used in the OLTP system. These methods minimize the database downtime and maximize the availability.

**Disk-to-Disk Backups**

Here backup is taken on the disk rather on the tape. Disk-to-disk backups are done for the following reasons –

- Speed of initial backups
- Speed of restore

Backing up the data from disk to disk is much faster than to the tape. However it is the intermediate step of backup. Later the data is backed up on the tape. The other advantage of disk-to-disk backups is that it gives you an online copy of the latest backup.

**Mirror Breaking**

The idea is to have disks mirrored for resilience during the working day. When backup is required, one of the mirror sets can be broken out. This technique is a variant of disk-to-disk backups.

**Note** − The database may need to be shutdown to guarantee consistency of the backup.

**Cold Backup**

- Cold backup needs to shut down your system which makes your system unavailable
- Therefore It is recommended to schedule cold backups during off-peak hours.
- All the parties affected from cold backup should get communication of the timeframe in which the system will be unavailable, as it will avoid potential loss of productivity with the system.

**Hot Backup**

- Hot backup is highly preferred method in global or highly available environments where downtime of window is not possible, as Hot backup can also occur while system is running, therefore it does not affect system usage.
- Hot backup narrows the gap between the time period when the last backup was run and when the system experiences failure because the backup can occur more frequently. Therefore reducing the amount of work lost in the failure event.
- Generally, due to global nature of their business, many organization prefer to use hot backup strategy. As their system accessed globally, which leaves minimum or no window for downtime, and thus a hot backup strategy is required.

| HOT BACKUP | COLD BACKUP |
|---|---|
| Also, known as dynamic backup. | Also, known as static backup. |
| Hot backups are resource intensive. | Cold backups consume fewer resources in comparison. |
| It can be performed when the systems are up and running. | Database operations need to be stopped when the backup is performed. |
| Database is available at all times. | Database can't be even accessed when the backup is in progress. |

**SURE WEST BROADBAND ONLINE BACKUP**

Refer the test book

**DISASTER RECOVERY PLAN**

Preparing for disaster recovery starts long before an emergency hits. The process begins with a business impact analysis and risk assessment. These two steps are essential parts of the preparation process, since they help the business quantify financial and operational costs that are likely to be incurred should a disaster strike.

It's best to conduct these evaluations during times of stability, when various stakeholders can devote time to thoroughly assessing how safety, security, compliance, and other key components may be impacted by various events.

When conducting a business impact analysis, stakeholders come together to detail a series of different disaster scenarios—and then predict the level of data loss and downtime that is most likely to ensue.

For example, the disaster recovery testing team might start by answering questions such as:

- What will happen if a natural disaster causes the destruction of an entire physical facility?
- Which teams will be prevented from doing their jobs if an outage occurs?
- How will operations be impacted if a major storm were to hit headquarters?
- Who will need to work from home if there's a global pandemic—and how?

Addressing these and other "what if" scenarios allows the organization to identify critical business functions, calculate potential losses, and determine how much downtime could be tolerated before a major disruption ensues. This business impact analysis can then be used to determine the full scope of hardware, equipment, and IT resources that would be needed before the threshold is breached.

A second essential element of this analysis is conducting a risk assessment. By further evaluating the potential ramifications of an unplanned event, the business can identify specific hazards and network infrastructure vulnerabilities—and then prepare procedures to minimize any long-term damage.

Together, the results of both the business impact analysis and risk assessment can be used to inform a robust disaster recovery strategy. Since the goal is to recover business functions as quickly as possible, these preparation steps are key: They'll help ensure the recovery process can be initiated without delay when necessary.

## RECOVERY MODELS

Recovery is the phase of reconstructing a database after some element of a database has been hidden. The recovery model of a current database is inherited from the model database when the new database is generated. The model for a database can be changed after the database has been created.

- **Full recovery model** − It provides the most flexibility for recovering the database to an earlier point of time.
- **Bulk-logged recovery model** − Bulk-logged recovery provides higher performance and lowers log space consumption for certain large-scale operations.
- **Simple recovery model** − Simple recovery provides the highest performance and lower log space consumption but with significant exposure to data loss in the event of a system failure. The amount of exposure to data loss varies with the model chosen. Each recovery model addresses a different need.

### Simple Recovery Model

In the Simple recovery model, database transaction logs are cleared along with the Checkpoint or Backup operation to minimize transaction logs.

### Principles

I think the name "Simple" does not accurately indicate how a database works under this model. A more accurate name is "Checkpoint with truncate log." In detail, all committed transactions are cleared upon completion of the Checkpoint or Backup operation, with only a few logs kept, necessary for recovery when an instance restarts. This model can minimize database transaction logs and storage usage, reduce storage overhead, and eliminate the need for special DBAs to maintain and back up database logs.

However, this model has obvious disadvantages. For example:

1. Database log backups cannot be implemented.
2. Databases based on the Simple model cannot implement point-in-time recovery.
3. Data can at most be recovered to the last backup file (either full backup or differential backup) and cannot be recovered to the latest availability status.

### Application Scenarios

According to the aforementioned principles of the Simple database recovery model, we can easily find applicable scenarios for the Simple model, including:

1. Non-crucial data (for example, log information) is stored in databases.
2. Databases do not require point-in-time recovery at any time and in any cases.
3. Loss of partial databases is tolerable in the event of database disasters.
4. Data in a database has a very low change frequency.
5. Databases do not require high availability (HA) in a foreseeable period (such as Database Mirroring, AlwaysOn, and Log Shipping).

## Full Recovery Model

The Full model in SQL Server is quite the opposite of the Simple recovery model. This section shows the following four aspects about the Full model: working principles, application scenarios, setting, and example scenarios.

### Principles

In contrast to Simple, we can consider the Full model as "Checkpoint without truncate log," that is, the SQL Server database engine does not truncate transaction logs. Therefore, compared with databases using the Simple model, databases using the Full model have transaction log files that increase faster and are much larger. These database log files contain all recently committed transactions until a transaction log backup occurs and finishes successfully.

Therefore, databases using the Full model have the following features:

1. Database logs can be backed up.
2. Point-in-time recovery can be implemented.
3. Data can be recovered to a point in time very close to a disaster occurrence time point to minimize data loss.

### Application Scenarios

Now that we have described the Full model, let us take a look at applicable scenarios for the Full model, including:

1. Critical business data stored in databases (such as order information and payment information).
2. Data with very high security requirements, which, if lost, must be retrieved to the greatest extent possible at any time and in all cases.
3. Very little data loss is acceptable in the event of disasters.
4. Very high database HA is required (for example, high requirements on Database Mirroring or Alwayson).
5. Point-in-time recovery of databases is required.
6. Database recovery per page is required.

Of course, compared with the Simple model, the transaction log files in the Full model have a higher growth speed and range. Therefore, DBAs need to maintain, monitor, and back up database transaction logs.

### Bulk-Logged Recovery Model

As a mix of the Simple and Full recovery models, the Bulk-logged model adapts and improves the Bulk Imports operation under the Full model.

#### Principles

In a SQL Server database system, a method called Bulk Imports is available for quickly importing data, such as BCP, Bulk INSERT, and INSERT INTO... SELECT. If these Bulk operations are performed in a database under the Full model, massive amounts of log information are generated, significantly influencing SQL Server performance. The Bulk-logged model is designed to solve this problem. When a Bulk Imports operation is performed in a database running under the Bulk-logged model, very few logs are recorded to prevent the sharp increase in transaction logs and guarantee stable and efficient SQL Server performance. Simply, when no Bulk Imports operations are performed, the Bulk-logged model is equivalent to the Full model; when a Bulk Imports operation is performed, it is equivalent to the Simple model. Therefore, databases using the Bulk-logged model cannot implement point-in-time recovery. This is also a disadvantage in the Simple model.

#### Application Scenarios

Based on Bulk-logged model principles, applicable scenarios include:

1. Bulk Imports operations, such as BCP, Bulk INSERT and INSERT INTO... SELECT
2. SELECT INTO operations
3. Index-related operations: CREATE/DROP INDEX, ALTER INDEX REBUILD or DBCC DBREINDEX

4. The most common application scenario for the Bulk-logged model is switching to Bulk-logged before a Bulk operation and then switching back to Full after the Bulk operation.

**TESTING**

Testing is very important for data warehouse systems to make them work correctly and efficiently. There are three basic levels of testing performed on a data warehouse −

- Unit testing
- Integration testing
- System testing

**Unit Testing**

- In unit testing, each component is separately tested.
- Each module, i.e., procedure, program, SQL Script, Unix shell is tested.
- This test is performed by the developer.

**Integration Testing**

- In integration testing, the various modules of the application are brought together and then tested against the number of inputs.
- It is performed to test whether the various components do well after integration.

**System Testing**

- In system testing, the whole data warehouse application is tested together.
- The purpose of system testing is to check whether the entire system works correctly together or not.
- System testing is performed by the testing team.
- Since the size of the whole data warehouse is very large, it is usually possible to perform minimal system testing before the test plan can be enacted.

**Test Schedule**

First of all, the test schedule is created in the process of developing the test plan. In this schedule, we predict the estimated time required for the testing of the entire data warehouse system.

There are different methodologies available to create a test schedule, but none of them are perfect because the data warehouse is very complex and large. Also the data warehouse system is evolving in nature. One may face the following issues while creating a test schedule −

- A simple problem may have a large size of query that can take a day or more to complete, i.e., the query does not complete in a desired time scale.
- There may be hardware failures such as losing a disk or human errors such as accidentally deleting a table or overwriting a large table.

**Note** − Due to the above-mentioned difficulties, it is recommended to always double the amount of time you would normally allow for testing.

**Testing Backup Recovery**

Testing the backup recovery strategy is extremely important. Here is the list of scenarios for which this testing is needed −

- Media failure
- Loss or damage of table space or data file
- Loss or damage of redo log file
- Loss or damage of control file
- Instance failure
- Loss or damage of archive file
- Loss or damage of table
- Failure during data failure

**Testing Operational Environment**

There are a number of aspects that need to be tested. These aspects are listed below.

- **Security** − A separate security document is required for security testing. This document contains a list of disallowed operations and devising tests for each.
- **Scheduler** − Scheduling software is required to control the daily operations of a data warehouse. It needs to be tested during system testing. The scheduling software requires an interface with the data warehouse, which will need the scheduler to control overnight processing and the management of aggregations.
- **Disk Configuration.** − Disk configuration also needs to be tested to identify I/O bottlenecks. The test should be performed with multiple times with different settings.
- **Management Tools.** − It is required to test all the management tools during system testing. Here is the list of tools that need to be tested.
    - Event manager
    - System manager
    - Database manager
    - Configuration manager
    - Backup recovery manager

**Testing the Database**

The database is tested in the following three ways −

- **Testing the database manager and monitoring tools** − To test the database manager and the monitoring tools, they should be used in the creation, running, and management of test database.
- **Testing database features** − Here is the list of features that we have to test −
    - Querying in parallel
    - Create index in parallel
    - Data load in parallel
- **Testing database performance** − Query execution plays a very important role in data warehouse performance measures. There are sets of fixed queries that need to be run regularly and they should be tested. To test ad hoc queries, one should go through the user requirement document and

understand the business completely. Take time to test the most awkward queries that the business is likely to ask against different index and aggregation strategies.

**Testing the Application**

- All the managers should be integrated correctly and work in order to ensure that the end-to-end load, index, aggregate and queries work as per the expectations.
- Each function of each manager should work correctly
- It is also necessary to test the application over a period of time.
- Week end and month-end tasks should also be tested.

**Logistic of the Test**

The aim of system test is to test all of the following areas −

- Scheduling software
- Day-to-day operational procedures
- Backup recovery strategy
- Management and scheduling tools
- Overnight processing
- Query performance

**Note** − The most important point is to test the scalability. Failure to do so will leave us a system design that does not work when the system grows.