Pre-requisites

12 December 2022 19:53

Lab Setup Requirement	Hardware CPU - Intel Core i3/i5/i7 processor, RAM - at least 8 GB HDD- 512 GB / 1 TB, OS - Windows 10 /8.1/11, MS Word/excel, PowerBI desktop (optional)

Pre-requisites
Software - 1. SQL Server 2016, 2017 or 2019 Enterprise/Developer edition, 2. SQL Server Management Studio/Azure Data Studio, 3. Visual Studio 2019/2022/VS code, (IDE) 4. A Valid Azure Subscription, Azure CLI, Storage explorer, Microsoft Azure Subscription, Git tools and GitHub account, Azure PowerShell, PowerShell ISE, Note: Linux environment VMs (Apache hadoop/big data) (Linux OS) Tool: Putty.exe Azure based Linux servers, Vmware player/ virtual box AWS Tools for VS code, GCP Tools for VS code.

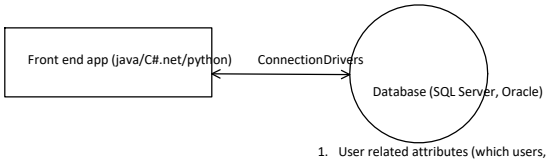
Azure Subscription (trial)
-- 12 months of free service + 30 days of 200 USD credit
-- enterprise subscription

Database Fundamentals and SQL Server BI

12 December 2022 19:54

Application metadata

MSSQL - 1433
MySQL - 3306



1. User related attributes (which users, port, MSSQL - 1433, encryption, protocol (TCP)
2. SQLConnection
3. ADOConnection (ADO.Net)

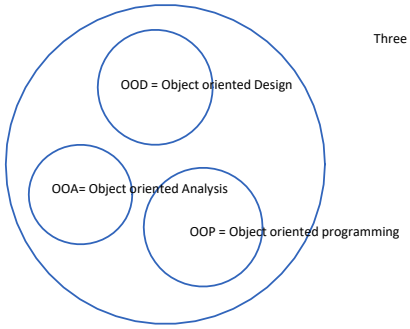
1960 (IBM) - developed the integrated Management system which is based on hierarchical database model.

1970 (IBM) - the relational database model was developed by E.F Codd.

1980 (IBM) - developed the Structured Query Language (SQL). It is declared as the standard language for the queries by ISO (International Standard Organization) and ANSI (American National Standards Institute).

OOPs principles

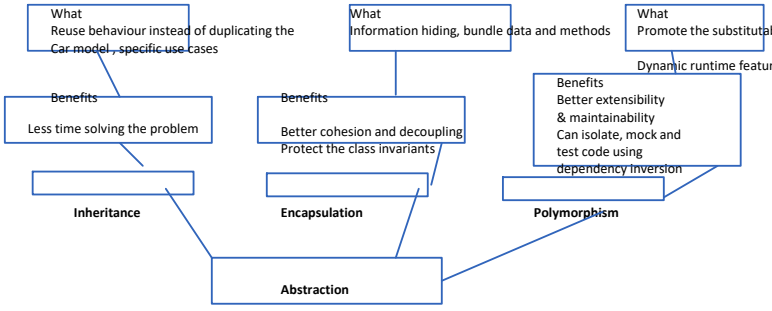
- Abstraction
- Encapsulation
- Inheritance
- Polymorphism



Three pillars of object design

Type Car = {
Types: 'small' | 'medium' | 'heavy'
Color: 'white' | 'black' | 'red'
Full_efficiency: 'efficient' | 'non-efficient'
Price: 'chaper' | 'moderate' | 'expensive'
}

Data Abstraction is the root principle of design.



1. Cohesion - represents a relationship within the module where a single class, has its well-defined purpose.

- It refers to what the class (a module) can do.

a) Low cohesion - low cohesion means a class which can perform a great variety of actions - broad, involves multiple operations.

Contracts - interface, abstract class, types

What
Data Modelling concepts using the contracts and concretions. (objects)

Benefits
Simply design API, simply data design, should
Separate how the data/ abstract can be designed
Declarative vs imperative

Staff Class
checkEmail()
sendEmail()
emailValidate()
PrintLetter()

b) High cohesion - means the class is to be focused on what it should be implementing. It includes only the methods related to the intension of the class.

Dependency Inversion - forms the pillar of object oriented programming to define as couple the software modules loosely so that one form of objects/ class while changing, doesn't affect others.

Staff Class
salary
emailaddr
setSalary (newsalary)
getSalary()
setEmailAddr(newEmail)
getEmailAddr()

Polymorphism - Compile time / Static polymorphism (method overloading, operator overloading)
Runtime / dynamic polymorphism (function overriding)

2. Coupling - refers to the principle of how related or dependent two classes / modules are toward each other. For low coupled classes, changing something major in one class should not affect other.

Coupling definition

In OOD, the coupling refers to the degree of the direct knowledge that one element/ object has of another element/object. How often the changes in class A forces the changes in class B.

1. Tight coupling - means when two classes often change together, when class A cant be changed directly because it's going to make changes in class B.

```
Class Subject {
Topic t = new Topic();        // subject class is tightly coupled the topic class
Public void letsRead()
{
 t.understand();
}
}


Class Topic {
 public void undertand()
{
 System.out.println("Tight coupling scenario");
}

}
```

2. Loose coupling - when two classes are just aware of each other , like class A and class B. Also, class B is expose through its interface, then class A and class B are loosely coupled.

e.g. Microservices application

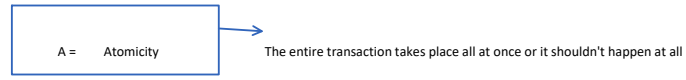
```
Public interface Topic
{
 void understand();
}

Class Topic1 implements Topic {
Public void undertand()
{
 System.out.println("This is loose coupling example");
}
}
Class Topic2  implements Topic {
Public void understand()
{
System.out.println("This is another loose coupling class");
}
}
Public class Subject{
Public static void main(String[] a)
{
 Topic a  = new Topic1();
t.understand();
}
}
```

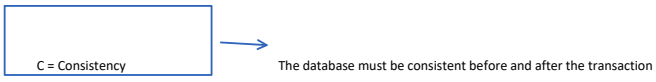
e.g. microservices application design

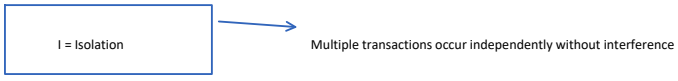
Empaddr and emp classes are separate classes where change of a simple address of a emp entity / object does not affect to the emp class and its related methods.

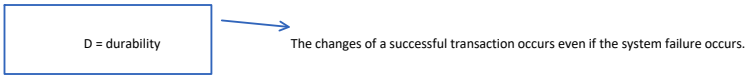
High coupling scenario - it would make difficult to change or maintain the code, since classes are closely knit together , making a change could require an entire system revamp.

Best practice - good software design should always has **high cohesion and low coupling**.









Different Types of Databases

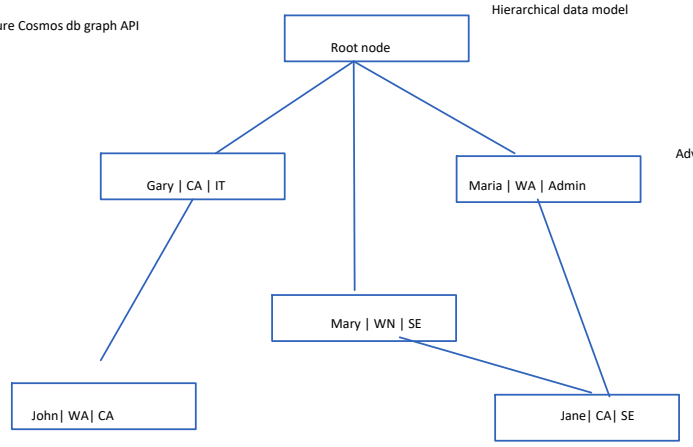
1. **Flat file database** -- kind of text database where each line of the plain text file holds only a single record. (e.g. MS Access, MS Excel)
2. **Hierarchical database** -- based on hierarchical data model, where the data is viewed as a collection of tables, data is designed into a tree like structure where each record consists of one parent record and many child record. (e.g. IBM DB2 - IBM Information Management System (IMS) , Windows Registry, XML data storage)
3. **Network model database** - can consists of multiple parent segments and this segments can be grouped together as levels but there's always exists a logical association between the segments belonging to any level.
4. **Relational database** - consists of set of tables with columns and rows
5. **Object-oriented database** - information can be represented in the form of object-oriented programming. Inclined towards the objects e.g. multimedia records in a relational database can be definable data object.
   Mongo db is a object-oriented database.
6. **Distributed Database** - Consists of two or more files located in different sites/ location.

e.g. SQL Server mirror databases,

7. **NoSQL databases** - non-relational db has support for unstructured, semi-structured data and it can include dynamic schema, flexible data model for faster data retrieval
   e.g. Mongo db, Cassandra, Couch db, Azure Cosmos db
8. **Graph database** - node - entity (rows in the table), attributes (relationship/columns)

e.g. Neo4j, Azure Cosmos db graph API

Hierarchical data model

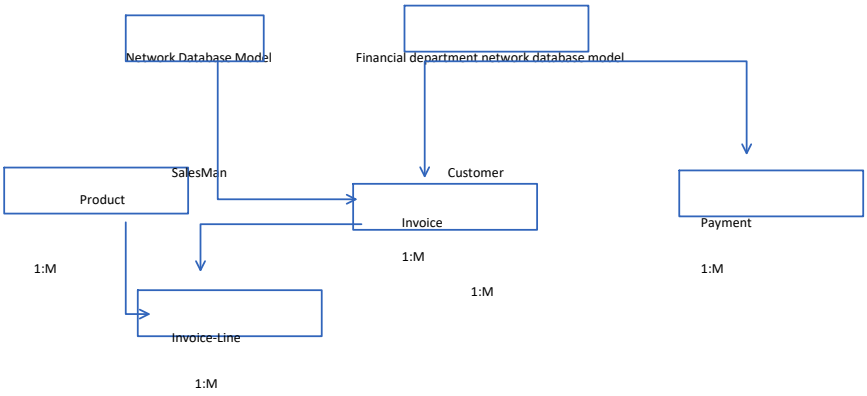


Advantages

- As the database is based on the hierarchical data models, the relationship between various layers are logically simple , it has a very simple hierarchical database structure.
- It has the data sharing facility since all data are held in a common database data and sharing of data becomes practically feasible
- It offers data security and integrity since it's based on parent-child relationship.

Disadvantages:

- Design is simple but implementation is complex
- This model also lacks flexibility as the changes to the new tables or segments often leads to very complex system management tasks.
- It has no standards as the implementation of the model does not provide any specific standard and limited to the relationship s which does not conform to 1:N format.



Advantages

- This model is simple and easy to design compared to hierarchical data model
- This model is capable of handling multiple types of data easily and we can access data easily, we can implement 1:1, 1:M, M:N relationships.

Disadvantages

- The schema of the network database mode is quite complex and records are maintained through pointers.
- The design of the network database mode is not user-friendly
- The model does not have any scope of automated query optimization

Data Dictionary or metadata in DBMS

A data dictionary is an integral part of a database. It holds the information about the database and the data which it stores named as metadata.

Characteristics of data dictionary

- A metadata is defined as the data about the data
- It's the self-describing in nature of databases
- It holds the information about each data elements in the database
- Such as names, types, ranges of values, access authorization, include the application details / program which uses the data
- Metadata is being used by programmers to develop the application, queries to manage and manipulate the data.

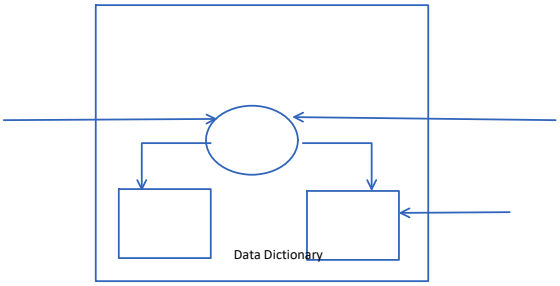
Types of Data Dictionary

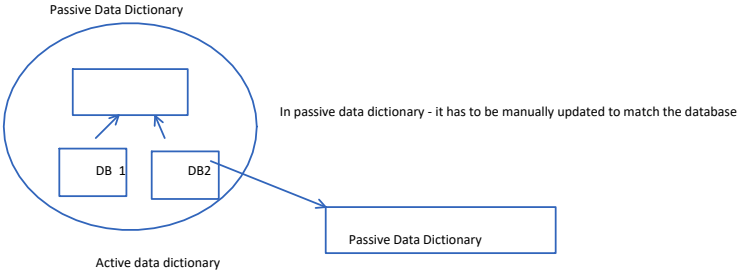
1. Active Data Dictionary -- self updating

a) managed automatically by the data management software
b) It's always consistent with the current structure of the database (e.g. RDBMS)

2. Passive Data Dictionary - mainly for documentation purpose
- Managed by user of the system and is modified manually by the user. (e.g. Dataedo by IBM IMS)

Active Data Dictionary





Data considered in DBMS for data Dictionary

Schema
- Tables

-Columns
- Constraints
- Foreign keys
- Indexes
- Sequences

Programs in SQL
- Views
- Stored Procedures
- User defined Functions (UDFs)
- Triggers

Storage
- Size of tables and indexes stored inside the db
- Number of rows in table

Benefits of data dictionary

1. Improves the data quality
2. Spot the data anomalies
3. Implement transparency and collaboration
4. Get access to the good data
5. Involve regulatory compliance
6. Enables fast and accurate data analysis

OLTP - Online Transaction Processing (SQL server database engine / MSSQL, mysql, oracle)
OLAP - Online Analytical Processing (SQL server datawarehouse)

Employee Table

Fields	Columns/Attributes Emp ID	Columns/Attributes Emp Name	Columns/Attributes Emp Hire date
Rows1 (records/ tuples)	1001	John	10/12/2021
Row 2	1002	Nash	11/12/2021

Foreign Key

Primary Key - Emp_id which uniquely defines the each record in the employee table & help to distinctly identify each employee.

Employee_Address table

Fields	Attributes (emp_id)	Emp_primary_address	Emp_secondary_address
Row1	1001		
Row2	1002		

Primary Key = emp_id

Primary Key definition

A primary key is a column or a set of columns in a table whose values uniquely identifies a row in the table. A relational database is designed to enforce the uniqueness of primary keys by allowing only one row with a given primary key value in a table.

Foreign Key definition

A foreign Key (FK) is a column or combination of columns which is used to establish and enforce a relationship between the data in two tables.

- A Foreign key is a column whose value corresponds to the values of the primary key in another table
- A foreign key is a column or a set of column in table whose values corresponds to the values of the primary key in another table.

- In order to add a row with a given foreign key value, there must exist a row in the related table with the same primary key value.

- Emp_id the foreign key column of employee_address table whose value corresponds to values of the primary key (emp_id) of employee table

Integrity Constraints in SQL

- Constraints are the set of rules which enforce on the data to be entered into the database table. Basically, constraints are used to restrict the type of data which can be inserted into a database table.

Feature of Integrity Constraints

- The integrity constraints are one of the protocols which should be followed by the table's data columns. The constraints are generally established to restrict multiple types of information which can be entered into a table to ensure the integrity of data.

- Achieving the complete configuration of the constraints ensures that the data in the db is accurate and reliable to be consumed in the application.
- We can apply the integrity constraints at the column level or table level.
- The table level integrity constraints are applied on the entire table
- While the column level integrity constraints apply to the only one column.

Constraints can be defined in two ways:

a) Column level: The constraints can be defined immediately after the column definition with the CREATE TABLE statement. So, these are called as the column-level constraints.
b) Table level: The constraints can be defined after all the columns specified, with the ALTER TABLE statement. This is referred as the table level constraints.

SQL Server has the support for six types of constraints

1. **Primary Key constraint -** The **primary key** is a set of one or more fields / columns of a table which helps to identify the records in the db table. It can not accept any null or duplicate values.

Features of Primary Key Constraint

- The primary key must have distinct values which means one of the columns of the table should have a unique value from the other.
- By default with the primary key, null values are not allowed in a primary key column of a table.
- Any table may have only a single primary key which can be made up of multiple fields.

Primary key constraint defined at the column level:

```
Create table table_name
(
Column1 datatype [CONSTRAINT constaints_name] PRIMARY KEY,
Column2 datatype
);
```

2. Unique Key Constraints - A unique key is a set of one or more columns/fields which uniquely identifies each row / record in a table.

Features of Unique Key Constraint

- A table can have more than one unique key unlike the primary key
- Unique key constraint can also accept null values for a column
- Unique constraints are also referenced by the foreign key of another table. It can be used when developer wants to enforce unique constraints on a column and a group of columns which is not a primary key.

Students table - 1

Roll_no	Student_Name	Batch	Phone_no	Citizen_ID	Country_of_origin
001	Alan	01	212-334-2234	AB01	US
002	Matt	02	202-440-2335	CD04	Canada
003	Hans	03	304-223-2337		

Roll_no is a primary key

Citizen_ID , Country_of_origin are the unique key for the Students table

Student table -2

Roll_no	Student_Name	Batch	Phone_no	Citizen_ID	Country_of_origin
001	Alan	01	212-334-2234	AB01	US
002	Matt	02	202-440-2335	CD04	US
003	Hans	03	304-223-2337		Holland

Primary Key - Roll_no
Unique Key - Citizen_ID

Difference between Primary Key and Unique Key

Features	Primary Key	Unique Key
Basic	Used to serve as a unique identifier for each row in a table	Uniquely identifies a row which is not a primary key
NULL value acceptance	Cannot accept any NULL values	Can accept NULL values
Number of Keys can be defined in the table	Only one primary Key in a table	More than one unique key
Auto-increment	A Primary Key has the support for auto-increment value.	A unique Key does not support the auto-increment value
Modification	We cannot change or delete values stored in primary key	We can change the unique key values

IDENTITY (2,1)
Seed - initialization value (2)
Increment - increment value (3, 4, 5)

Indexes in SQL Server

Definition - An index is a schema object. It's used by the db server to speed up the data retrieval or rows/records by using a pointer. It can reduce disk (I/O) by using a rapid path access method to locate data quickly.

Features of Indexes

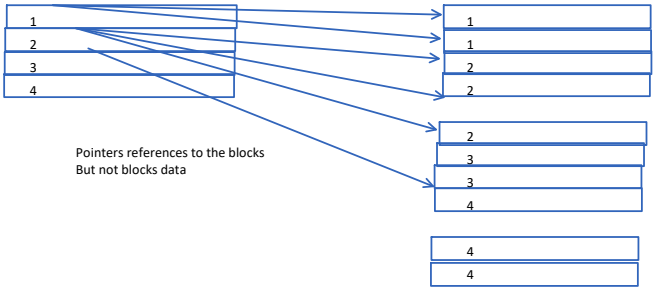
1. An Index can help to speed up the select queries and where clauses.
2. Indexes can be created or dropped with no effect on the data.
3. When an index is created, it includes a column contains a wide range of values.

Create index index_name on table_name column_name;

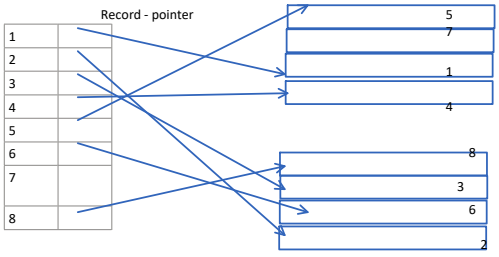
Table - table_name
Column - column_name

Create index index_name on table_name (col1, col2...)

a) Clustered Index - Clustered index is a kind of index which should satisfy the conditions
- The data or file, which moving to the secondary memory should be in sequential or sorted order,
- There should be a key value, means it cant have any repeated value.
- When applied clustered index on a table, it should perform sorting in that table only.
- We can create only one clustered index in a table like primary key
- Clustered index is as same as the dictionary where the data is arranged by alphabetical order.
- In clustered index, index contains pointer to block but not direct data gets blocked.
- We can have one clustered index on multiple columns and this kind of index is called composite clustered index.



b) Non-Clustered index - index contains the corresponding pointer to the data



Difference between Clustered and Non-clustered index

Clustered index	Non-Clustered Index
Faster	Is slower
Required less memory for operations	Required more memory for operations
In clustered index, index is the main data	Index is the copy of the data
A table can have only one clustered index	A table can have multiple non-clustered index
Clustered index store pointers to block not data	Non-clustered index store both the value and a pointer to actual row which holds the data
It has the ability to store data on disk	Non-clustered index does not have inherent ability to store data on disk
Primary Keys of the table by default is considered as clustered index	Composite key / used with Unique key of the table defines the non-clustered index
A clustered index is type of index in which table records are physically recorded to match the index	A non-clustered index is a special type of index in which logical order of the index doesn't match physical stored order of the rows on disk
Clustered index size is larger	Non-clustered index size is smaller.

Surrogate Keys

Surrogate key is called synthetic primary key which is generated when a new record is inserted into the table automatically by a database which can be declared as the primary key of that table.

Features of Surrogate Key

1. It's sequential number outside the database which is made available to the user and application or it just acts as an object which's present in the db but not visible to the user or app
2. It's automatically generated by the system
3. It holds an anonymous integer
4. It contains unique value for all records of the table
5. The value can never be modified by the user or app
6. Surrogate key is called the factless key as it is added just for the case of identifying unique values and contains no relevant fact(information) which is useful for the table.

Student_reg_A

Reg_no	Name	% obtained
210101	Harry	80

210102	Matt	70
210103	Chris	84
210104	Lee	85

Student_reg_B

Reg_no	Name	% obtained
CS101	Maria	70
CS102	Simon	60
CS203	Sam	90

Surr_no	Reg_no	Name	% obtained
1	210101	Harry	80
2	210102	Matt	70
3	210103	Chris	84
4	210104	Lee	85
5	CS101	Maria	70
6	CS102	Simon	60
7	CS203	Sam	90

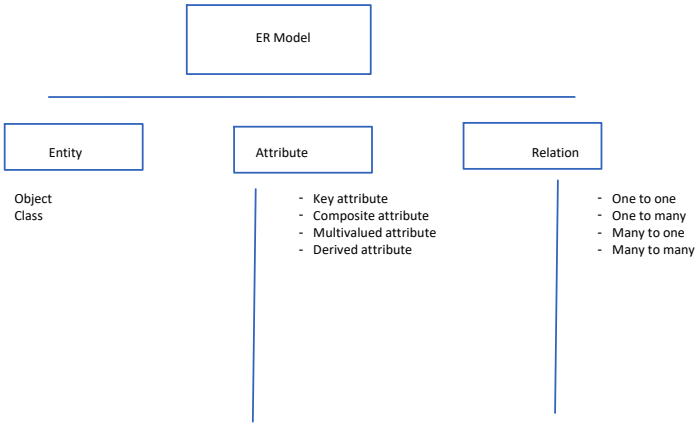
Item1
Item2
Item3
Item4
Item5
Item6
Item7
Item8

OFFSET 3 ROWS

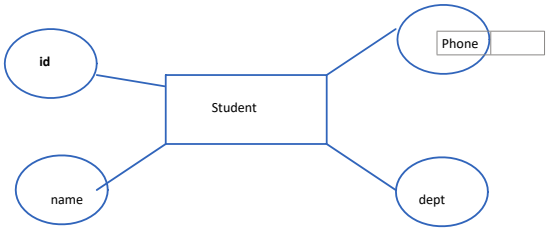
FETCH 5 ROWS

Benefits

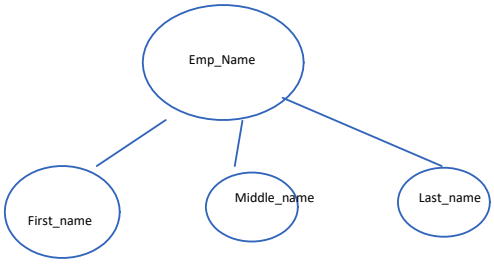
1. There's no direct information related with the table, so the changes are only based on the requirements of the application
2. Performance enhanced as the value of the key is relatively smaller
3. The key value is guaranteed to contain unique information
4. As it holds smaller constant values, the integration of the table easier
5. Enables to execute fast queries.



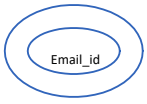
a) Key Attribute - key attribute is used to represent the main characteristics of an entity. It represents the primary key.



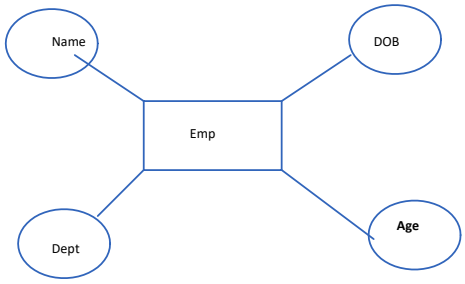
b) Composite attribute - composed of many other attributes altogether is called as composite attribute. The composite attribute is multiple simple attributes joined together.

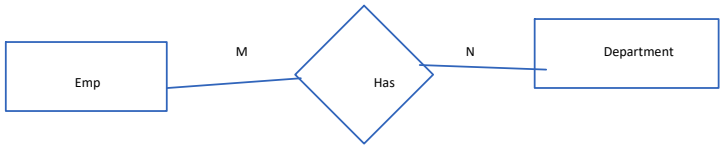


c) Multi-valued attribute - this attribute can have more than one value. These attributes are called as multivalued attributed.

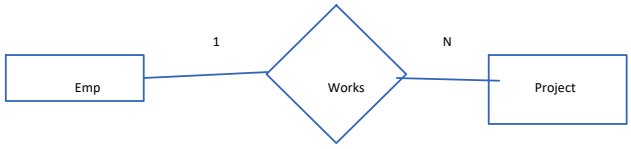


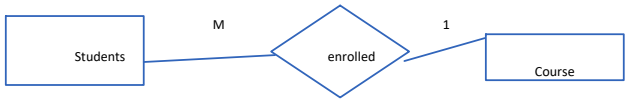
d) Derived attribute - An attribute can be derived from other attribute,





E1 : E2





Data types in SQL Server

Numeric data type in SQL server

Numeric	Storage space		
Tinyint	1 byte		
smallint	2 byte		
int	4 byte		
bigint	8 bytes		
Decimal(p,s) Numeric(p,s)	5-17 byte		
smallmoney	4 byte		
money	8 byte		
real	4 byte		
Float(n)	4-8 bytes		

P = precision (total number of digits can be stored both to the left and right of the decimal)
S = scale (max no of digits to the right of the decimal)

Decimal(8,3) allow the storage of total 8 digits and 3 of the digits to the right of the decimal or values.

Character data type

Char(n)	1 byte per character defined upto n
Varchar(n)	1 byte per character stored upto max 8000 bytes
text	1 byte per character upto 2 GB
Nchar(n)	2 bytes per character
Nvarchar(n)	2 bytes per character upto max 4000 bytes
ntext	2 bytes per character stored upto 2 gb

Nvarchar(max)

Date & time data type

datetime	0.00333 sec	8 bytes
Datetime2	100 nanosec	6-8 bytes
Date	1 day	3 bytes
time	00:00:00 to 23:59:59	3-5 bytes

Binary Data type

binary	Fixed with binary data	Upto 8000 bytes
Varbinary	Variable length binary data	Upto 8000 bytes

Column Property in SQL Server

IDENTITY property in SQL server

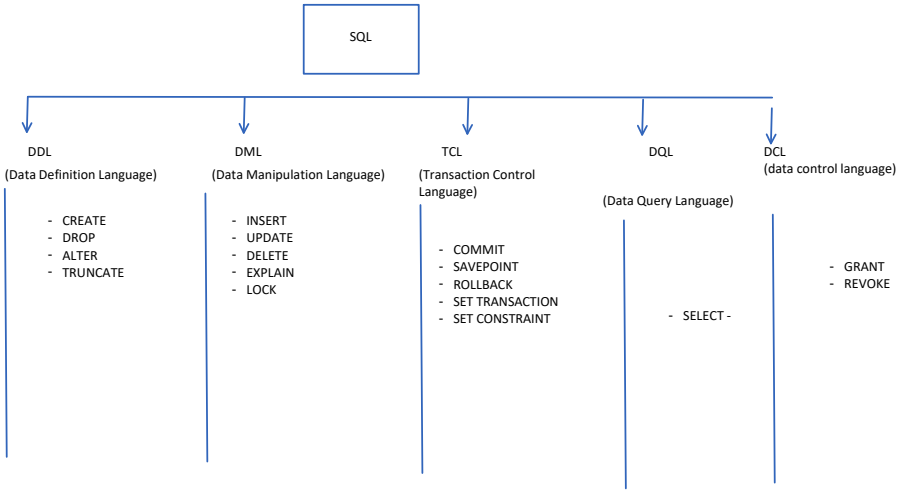
Identity property on columns can be defined to have a numeric data type. When the IDENTITY property is defined, SQL server manages the values in the columns on behalf of user.

IDENTITY consists of two parameters

- Seed - seed parameter defines the first number which will be assigned when the data is inserted into the table
- Increment - defines the number which will be added to the previous value for every subsequent row inserted into the table.

IDENTITY(1,2) - it'll start the value with 1 and will increment by 2 everytime when a new row is inserted into the table.

Nullability constraint in SQL server
Nullability is the most common property assigned to a column, whenever the column is defined as NOT NULL, we're required to assign a value to the column,
Whereas, when a column is defined as NULL, you don't have to assign a value to the column.



TCL

The transactions group a set of tasks into a single execution unit. Each transaction begins with a specific task and ends when all the tasks in the group successfully complete.

If any of the tasks fail, the transactions fail. Therefore, a transaction has only two results,

- Success
- Failure

MARS in SQL Server

The multiple active result sets is a feature with SQL server to allow the execution of multiple batches on a single connection. When MARS is enabled for use in SQL Server, each command object used adds a session to the connection.

- Applications can have multiple default result sets open and can interleave reading from them.
- MARS enables the execution of multiple db connection requests within a single connection. It allows batches to run. The database applications could not maintain multiple active statements in a connection.
- Applications can execute others statements (INSERT, UPDATE, DELETE, stored procedure calls) while the default result sets are open.

Benefit of schema

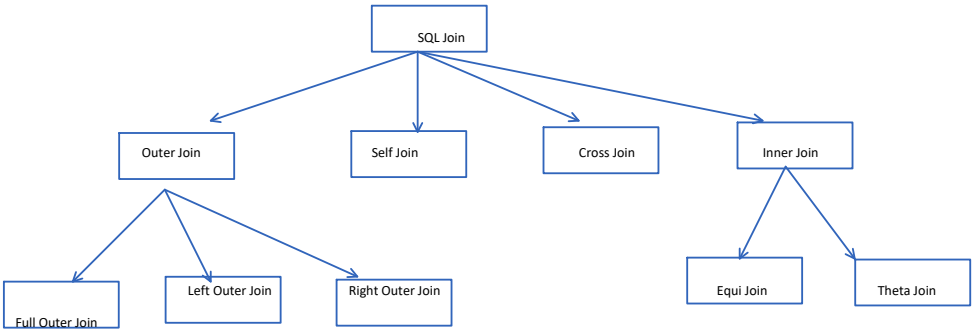
A database schema is considered the blueprint of a database object which describes how the data may relate to other tables or other data models.

The default schema for sql server is dbo.

But it's always recommended to create a customized schema to define how the different types of data (product, customer, customerAddress, employee) have been related to other tables / data models.

SQL Server Joins

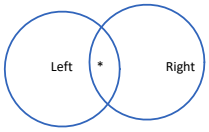
A SQL Join is a special form of generating a meaningful data by combining multiple tables relate to each other using a 'Key'. Typically, relational tables, must be designed with a unique column and this column is used to create relationships with one or more other tables. When you require a result-set that includes related rows from multiple tables, then the SQL joins are required on the column.



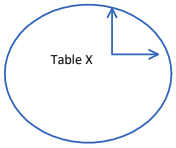
SQL Inner Join - The most simple and common form of join which is default join type.

- An Inner join combines two tables based on the criteria in the IN clause while also eliminating any rows from both tables which do not meet the criteria.

Cross - Join



Self join



Joining the table itself , is referred as self join

Normalization

- Benefits

1. Normalization is the process of organizing data in the database
2. It's used to eliminate undesirable characteristics from a set of tables. Used to eliminate the errors/ anomalies from insertion, update or delete.
3. Normalization divides the larger table into smaller tables linking them with relationships.
4. The normal form is used to reduce redundancy from the database table.

First Normal Form

- A relation will be 1NF if it contains an atomic value
- It can include a single value only in an attribute/field
- An attribute of a table cannot hold multiple values. It must hold single-valued attribute.

Second Normal Form

- All of the tables should be 1NF
- In the second normal form, all non-key attributes are fully functional dependent on the primary key

A B C - three variables

A -> B A is dependent on B
B -> C B is dependent on C

A -> C (A is dependent on C), (transitive dependency)

Third Normal Form

- A relation will be called in 3NF, if it is already in 2NF, and not contains transitive dependency
- 3NF is used to reduce the data duplication.
- It's also used to achieve data integrity
- If there's no transitive dependency exists for non-prime attributes for a table, then the relation/table must be in third normal form (3NF).

A relation is in third normal form if it holds at least one of the conditions for dependency X-> Y

1. X is super key
2. Y is prime attribute/ primary key, each element of Y is dependent on to some part of candidate/super key.

SQL Data Warehouse

12 December 2022 19:54

OLAP - Online Analytical Processing platform (Data Warehouse)

1. Datawarehouse definition
2. Concepts on Data Marts
3. What are the Data Lakes? Why we should design a Data Lake?
4. Examples of data warehouse
5. Examples of Data Lake & Data Mart
6. Data Warehouse Architecture
7. Tables design in Datawarehouse
8. Star schema design in SQL Server db through SSMS (SQL Server Management Studio)
9. - Fact table
   - Dimension table
   - Benefits of Data Warehouse
10. Define the concepts on Data Modelling
    - Star Schema (Demo/lab)
    - Snowflake Schema
    -
11. Definition of data integration
12. OLAP (Online Analytical Processing) , benefits, use cases
13. Difference between OLAP(SQL Data warehouse) and OLTP (SQL database) database
14. Data Mining concepts

Tools -

1. SQL Server Management Studio (SSMS)

A Data-warehouse is a process of collecting and managing data from varied sources to provide a meaning business insights.

e-Commerce use case

Customer sample dataset

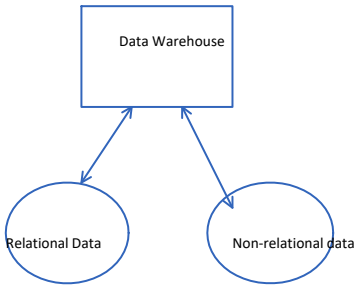
ID	Name	Location	Items Purchased	Time of purchase	IP address	Items kept in Cart
1	Matt	NJ	Laptop, Monitor	2022-9-6 17:00:00	120.34.23.5	Printer
2	Joe	NY	Headset	2021-12-01	192.168.1.1	Book

Problem Statement

a) Define which customers have made maximum purchases over the last three months
b) Define which customers have made minimum purchases (sales) over the last three months
c) Define which locations customers visited maximum to the ecommerce website?
d) Define how to make a recommendation of a product to customer which he/she can purchase next?
e) Define which of items has been kept in shopping cart of the customers but never made purchase?

- Sales Analytics
- Sentiment Analytics
- Weblog Analytics
- Recommendation Engine

Retail domain
Healthcare domain
Banking and Financial domain
Manufacturing domain



To ingest the relational / non-relational dataset into the SQL Data warehouse, the following analytical steps has to be performed.

Non-relational data - json, xml format
Real-time devices where data is extracted (IoT devices, weather sensors etc.)

ETL - Extract - Transform - Load model in Data Analytics

a) Data Extraction

Tasks-

- Data Cleaning (remove all duplicate data)
- Data parsing (sort the data as per the analytical requirement)
- Identify the structuredness/ formatting of the data (into proper format, in terms of rows/columns design)
- Impose the data into the tabular structure (table format with specific attributes and records)

b) Data Transformation

- Sql queries to start the analytical references (joins, functions, indexes, complex queries, views/stored procedures)

- Total no of customers who made the max purchase for products
- Total no of customers who made the min purchase of products
- What're the location(city/country) where customers visited max to the ecommerce website

c) Data Load

The processed data can be stored into various data warehouses for future analytical purposes. Load the data into the SQL datawarehouse. Design further the schema of the dataset with Star and snowflake schema.

<xml>

<node1>
<node2> Name </node2>
<node3> Address </node3>

A XML data for Customer

file: Customer.xml (Extensible Markup Language)

```
<Customer>
 <Name> Matt </Name>
<address> New York </address>
<phone> 202-122-1222 </phone>
</Customer>
```

Customer.json (javascript object notation)

```
Customer:
"name": "matt"
"address": "New York"
"phone": "212-122-1222"
```

file: Customer.json

Requirements of Data Warehouse

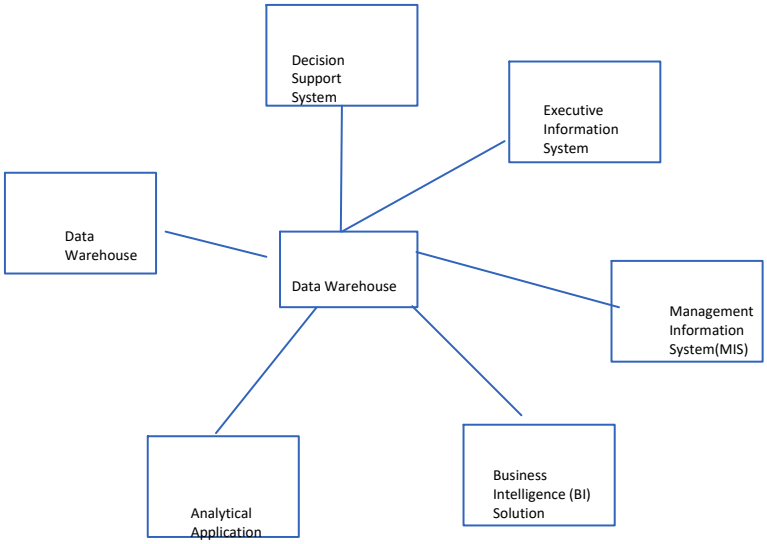
1. A Data warehouse (DW) is process of collecting and managing data from various sources to provide meaningful business insights.
2. A Data warehouse is typically used to connect and analyse business data from heterogenous sources. (data sources - traditional file system data, .csv, excel data, json/xml data, social media data, ERP/CRM dataset)
3. The data warehouse is the core of the Business Intelligence (BI) system which is built for data analysis and reporting.

Features of Data warehouse

- The data warehouse is maintained separately from the organization's operational database.
- We can call a data warehouse in some other names as well.

Examples of Data warehouse

- Informatica
- Vertica
- SQL Server Analysis Services (SSAS)
- Oracle Data warehouse
- Azure SQL Datawarehouse
- AWS Redshift
- GCP BigQuery



- How the data warehouse works

A data warehouse works as a central repository where the information arrives from one or more data sources. Data flows into the data warehouse from the transactional system and other relational databases.

Data may be

1. Structured
2. Semi-structured
3. Unstructured

- The data is processed, transformed and ingested so that users can access the processed data in the DW through the BI tools, SQL client tools and excel sheets.
- A DW also merges data coming from different sources into one comprehensive database.

- Types of Data Warehouse

1. Enterprise Data Warehouse (EDW)

Enterprise data warehouse (EDW) is a centralized warehouse. It provides decision support service across the enterprise. It offers a unified approach for organizing and representing data. It also provides the ability to classify the data according to the subject and gives access according to the division of the data.

e.g. SAP successfactor (employee organizational data)

2. Operational Data Store

Operational data store (ODS) is nothing but the data store required when neither the data warehouse nor the OLTP systems support organizations reporting requirements. In ODS, Data warehouse is refreshed in real time. It is widely preferred for routine activities like storing employee activities.

e.g. Salesforce CRM

3. Data Mart

A data mart is a subset of data warehouse. It specifically designed for a particular line of business such as Sales, Finance, Accounting, HR etc.

A data mart can collect data directly from various different data sources.

Limitation of DBMS

1. Data volume , there 's a limitation. In Azure SQL db, maximum volume/size of the db can be 5 TB.
2. Traditional DBMS cant fulfill the requirements of Big Data (>100 TB, 500 TB, 100 PB, 500 PB)
3. A data warehouse s separate from DBMS, it stores a huge amount of data which is typically collected from multiple heterogenous sources like files, DBMS etc.
4. The goal is to produce statistical results which can help in decision making

e.g. a college might see the exams results, student performance analysis results.

Need for Data warehouse

1. An ordinary RDBMS can store only in MB/GB amount of data and too specific purpose.
2. For storing in TB size, the data storage should be used Data warehouse,
3. A transactional database (e.g. RDBMS - SQL server db, mysql) doesn't offer to analytics.
4. To perform effective analytics, an organization needs to keep a centralized data warehouse to study its business by organizing , and using its historic data for taking strategies, decisions and analyse the trends.

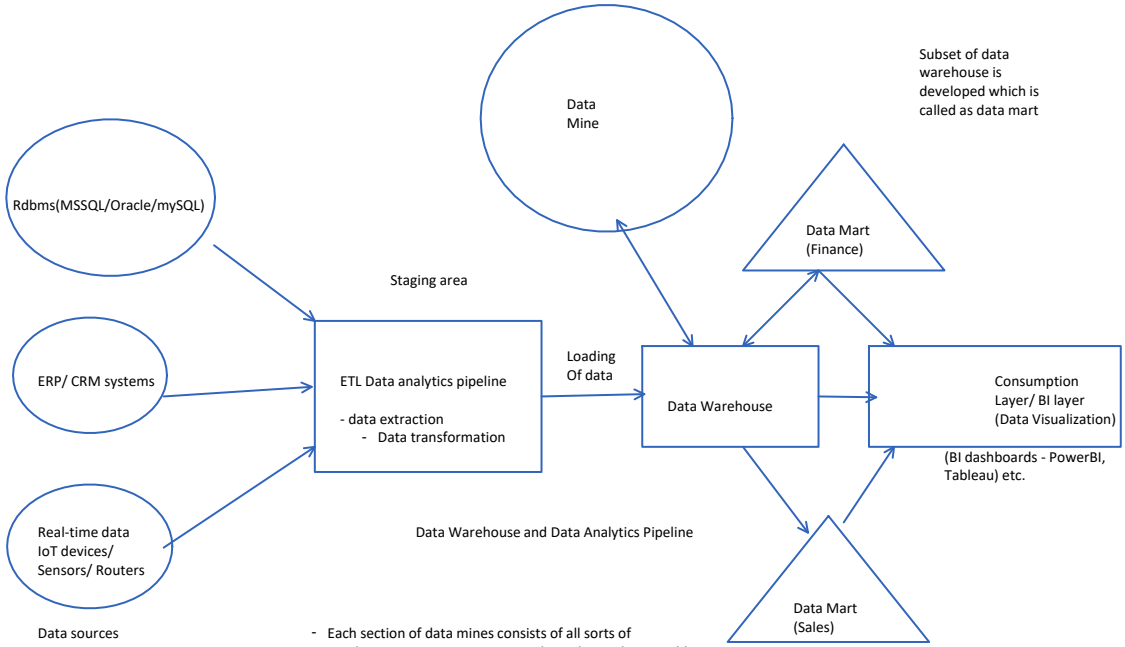
Difference Between RDBMS and Data Warehouse

Features	RDBMS (OLTP)	Data warehouse (OLAP)
Purpose	A common transactional database on operational or transactional processing. Each operation is an indivisible transaction.	A data warehouse is based on analytical processing of data. We can build data pipeline through ETL jobs. (Extract - Transform - Load)
analysis	The RDBMS system stores the current and upto-date data which is been used for daily operations.	A Data warehouse is integrated generally at the organization level by combining data from different databases. Example - A data warehouse integrates data from one or more databases. So that analysis can be done to get the results such as top sales of the month of an ecommerce website.
	A database is generally application specific. e.g. more data includes for app specific detail Customer database contains various tables customer, customerAddress, customerPurchase , customerOrders etc.	A data warehouse is integrated generally at the organization level , by combining data from various databases. e.g. A data warehouse integrates data from one or more multiple databases, so that the analysis can be implemented to get the results. "top product purchased by customer over last six months" in case of retail/ecommerce domain.
	Construction of the database is not so expensive.	Construction of data warehouse can be extensive.

Examples of Data warehousing

1. Social Media Websites - The social networking sites - Linkedin, Twitter are based on large data sets.
2. Banking - Credit card holders information, Spending/purchase pattern of customers, stored with the centralized DW of the banks.
3. Governments - data warehouse can be used to store and analyse tax payments which used to detect the tax liabilities.

e-commerce, Retail, healthcare, marketing.



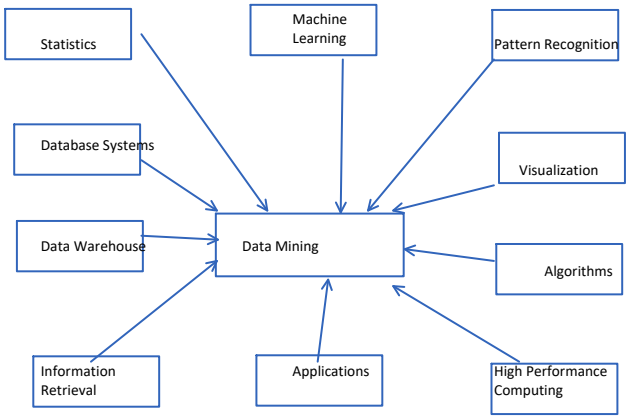
- Each section of data mines consists of all sorts of Employee, project, customer, products data, schema, tables, Attributes, relations etc.
- The data warehouse which is the centralized analytics data repository of an organization gets full set of data over extended period of time(> 10 yrs, > 20 yrs, > 30 yrs etc.) stored within the data mines.

Definition of Data Mining

The Data Mining refers to the knowledge mining from data, knowledge extraction, data / pattern analysis, data insights analysis. It's basically the process carried out for the extraction of useful information from the bulk of data or data warehouses.

The data mining is the result of extraction of the data patterns and knowledge after the data is processed through ETL jobs.

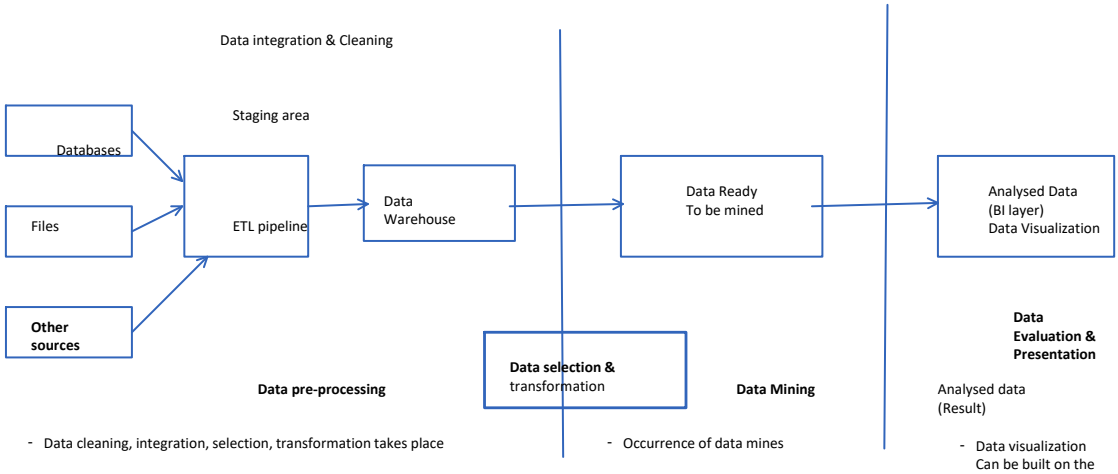
Technical data mining is the computation process of analysing data from different perspectives, dimensions, angles, categorizing/ summarizing it into meaningful information.



Data Mining can be applied to any type of data

- Data warehouses
- Transactional databases
- Relational databases
- Multimedia databases
- WWW (web apps)

Data mining is a while process involving into data pre-processing, data mining and data evaluation and presentation.

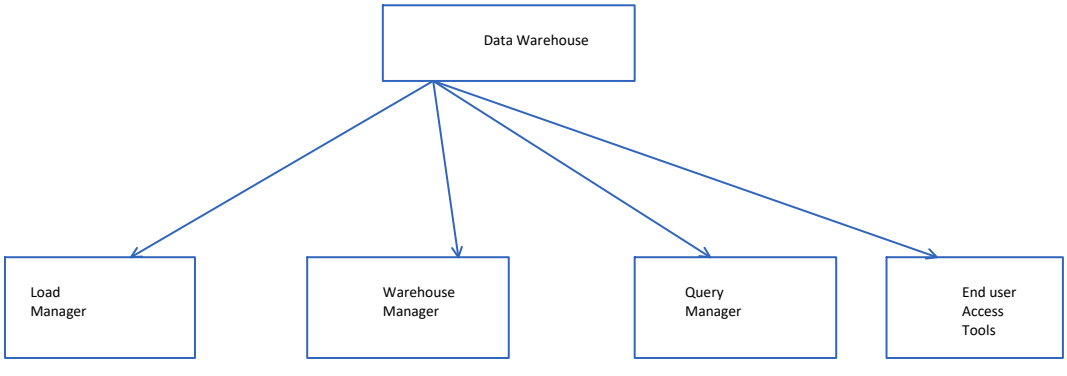


- Data cleaning, integration, selection, transformation takes place
- Occurrence of data mines
- Data visualization Can be built on the analysed data

Apps for data mining

- Financial Analysis
- Biological analysis
- Scientific analysis
- Fraud detection analysis
- Research analysis

- Components of Data Warehouse



- Load manager is called the front-end Component.
- It performs operations associated With the management of data into the data warehouse
- Query Manager is the backend
- Kinds of tools can be used

- Load manager is called the front-end Component.
- It performs all of the operations associated with the extraction and loading of the data Into the data warehouse.

- It performs operations associated With the management of data into the data warehouse
- It performs the analysis of the data to ensure consistency , creation

Of index, views,

- Generation of normalization, aggregation, transformation & merging

Of all source and backing up data.

- Query Manager is the backend

Component of SQL DW.

- Performs all the operations related to the management

Of user queries.

- Operations of the DW components are direct

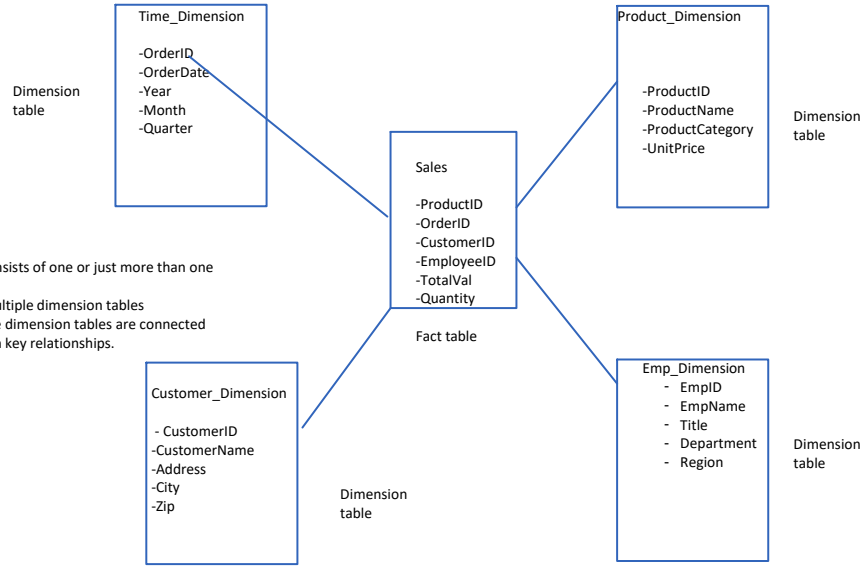
Queries to the appropriate tables for scheduling of the query execution.

- Kinds of tools can be used

a) Data Reporting tools (SSRS, Excel, PowerBI, Tableau etc.)
b) Query tools (SQL client tools)
c) App development tools (Visual studio)
d) OLAP tools / data mining tools (DB2, informatica)

Star Schema in Data Warehouse modelling

Star schema is the fundamental schema among the data mart and warehouse. It's simplest.
This schema is widely used to develop and build a data warehouse and dimensional data marts.
It includes one or more fact tables indexing any number of dimension tables.



- Star schema consists of one or just more than one Fact table
- It consists of multiple dimension tables
- The fact and the dimension tables are connected

on the basis of foreign key relationships.

Features of Star Schema on data modelling

- In star schema, the business process data, which holds the quantitative data about the a business which is distributed in fact and dimension tables.
- The fact table consists of more number of columns / attributes
- The dimension are smaller in size compared to fact tables, contains more number of rows/records.

Advantages of Star schema

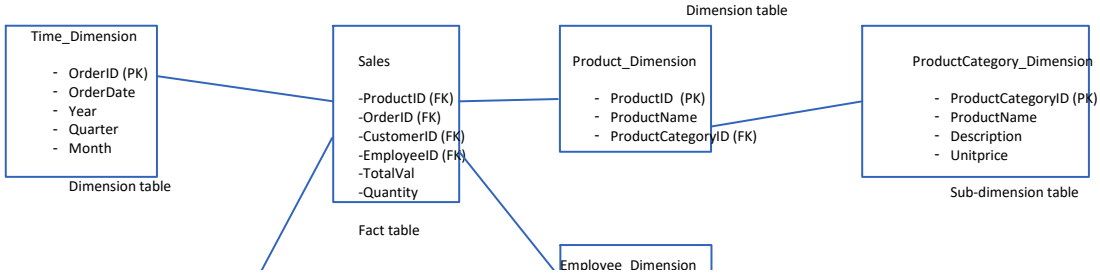
1. Simpler Queries - Join logic of star schema is quite simple and in comparison other SQL joining logic can be seen to fetch data from a transactional schema. The tables are highly normalized.
2. Simplified Business Reporting logic - In comparison to a transactional schema, which is highly normalized, the star schema makes simpler common business reporting logic like as reporting and ad-hoc analysis on data warehouse.
3. Build OLAP systems - Star schema is widely used in data warehouses (OLAP systems). In fact, the Star schema can help to deliver the major Relational OLAP model which can use star schema as a source of data.

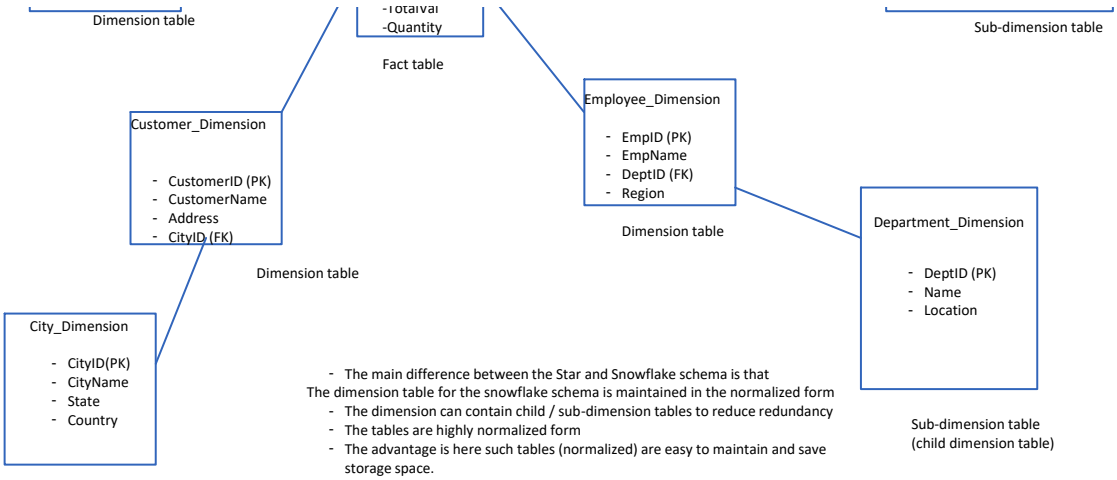
Snowflake Schema

The snowflake schema is a variant of the star schema. The centralized fact table is connected to multiple dimensions. In the snowflake schema, dimensions are presented in a normalized form in multiple tables.

The snowflake structure is materialized when the dimensions of a star schema are detailed and highly structured, also having several levels of relationship, the child tables have multiple parent tables.

The snowflake schema effect affects only the dimension tables and does not affect the fact tables.





- The main difference between the Star and Snowflake schema is that The dimension table for the snowflake schema is maintained in the normalized form
- The dimension can contain child / sub-dimension tables to reduce redundancy
- The tables are highly normalized form
- The advantage is here such tables (normalized) are easy to maintain and save storage space.
- It also means more joins will be required to execute the query.

Sub-dimension table
(child dimension table)

Sub-dimension table
(Child dimension table)

Characteristics of Snowflake schema

- The tables are highly normalized compared to star schema, they take less disk storage space .
- It's easy to implement dimension which is added to the schema
- They are multiple dimension tables, so performance is sometimes reduced, querying is bit more complex.
- Structured data which reduces the complexity around the problem of data integrity
- It uses small disk space because the data is highly structured.

Difference between Star and Snowflake schema

Star Schema	Snowflake schema
In Star schema, the fact and the dimension tables are contained.	While in snowflake schema, The fact tables, the dimension tables as the sub-dimension tables are contained.
Star schema is top-down approach	While it's a bottom-up approach
Star schema uses more space	While, it uses less space because all of the dimension tables are normalized upto one or more sub-dimension tables.
It takes less time for the execution of queries.	While for snowflake schema, it takes more time for the execution of queries.
In star schema, normalization is not used	In snowflake schema, both normalization and denormalization is used
It's quite simple to design	While, design is complex compared to star schema
The query execution & complexity for star schema is quite low.	Query execution and complexity of snowflake schema is higher than star schema
It has less number of foreign keys because of less number of dimension tables	While, snowflake schema has more of foreign keys because of higher number of dimension tables
Star schema has high data redundancy	While, it has less data redundancy because all of the dimension tables are normalized to sub-dimension tables.

Difference between Fact and Dimension table

Fact table	Dimension table
Fact table contains the measuring of the attributes Of a dimension table.	Dimension table contains the attributes on that truth table which calculates the metric.
In fact table, there's less attributes than dimension table. It means fact table contains more number of Rows/ records.	While in dimension table, there's more attributes than the fact table. More number of columns are available in the dimension tables.
In fact table, there' more records than dimension Table,	There's less records, more columns than fact table
Fact table forms a vertical table	While, dimension table forms a horizontal table.
The attribute format of fact table is in numerical format and text format.	While, the number of dimension is more than fact table in a schema.
It's used mainly for analysis purpose and decision Making systems	While the main task of the dimension table is to store the information about a business and its process.

Hands-on Lab

Create Star Schema with SQL Server Management Studio

Tasks

1. **Create a dedicated db for Star Schema**
2. **Create Database diagram support for the db**
3. **Create new database diagram**
4. **Create the data types for Fact and dimension tables (Under Database -> Programmability -> Types -> User-Defined-data-type)**
   1. **dbo.udt_surrogate_key (int, not null)**
   2. **dbo.udt_business_key (nvarchar(20), not null)**
   3. **dbo.udt_dollar_amount (money, not null)**
   4. **dbo.udt_quantity(int, not null)**

5. **Create Dimension tables**
6. **Create Fact tables**
7. **Create Build out relationships between Fact and dimension tables.**

1. Click on empty space & select "Select All"
2. Right click on any table & click "Table view" & choose the option "Name only"
3. Right click on the any of the table and click on option "Authorize selected tables"
4. Right Click on the empty space & click on "arrange tables"

Dimension tables

1.dim_date primary_key- (date_key)
2. dim_individual_customer PK - (individual_customer_key)
3. dim_territory PK - (territory_key)
4. dim_store PK - (store_key)
5. dim_employee PK - (employee_key)

Fact table (fact_sales_order)

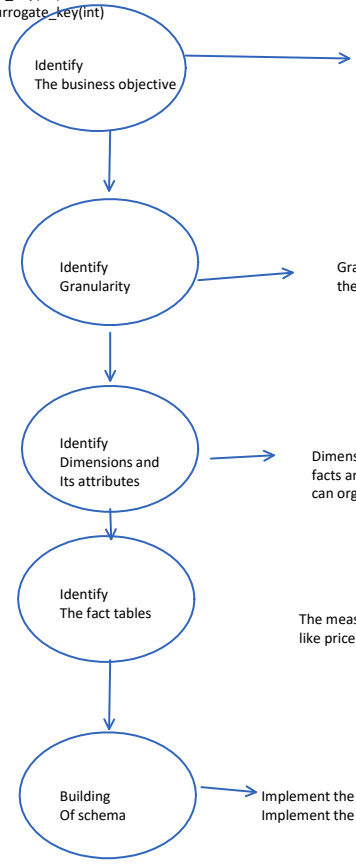
date_key udt_surrogate_key(int)
territory_key udt_surrogate_key(int)
employee_key udt_surrogate_key(int)
store_key udt_surrogate_key(int)
individual_customer_key udt_surrogate_key(int)

Dimension Data Modelling

Dimension data modelling is comprised of fact and dimension tables. The main objective of the dimension data modelling is to improve the data retrieval so it is optimized for SELECT operation.

Advantage - we can store the data in such a way so it's becomes easier to retrieve the data once stored in the data warehouse.
Dimensional model is the data model used by many OLAP systems.

Steps for creating the dimension data model



Sales, marketing, HR are some of the examples as per the requirement
Of the org. Since the most important step of data modelling is the selection of business
Objective which also depends on the quality of data available for that process

Granularity refers to the lowest level of information stored in the table. The level of detail for the business problem and its solution is described by the context.

Dimensions are the objects or things. Dimensions can categorize and describe the data warehouse facts and measures in a way which supports answers to the business problems. A data warehouse can organize the descriptive attributes as columns in dimension tables.

The measurable data is stored within the fact tables. Most of the fact table rows are numerical values like price, cost per unit etc.

Implement the relationship between the fact and dimension table through foreign key constraints.
Implement the dimensional model in this step.

Measures

Measures in Data warehouse are the set of aggregates which we can calculate, such as sum of total orders placed by the customer.

Measures example

Qualitative - productID
Quantative - like the price of the product

The source of all OLAP structures is a table within a data source. OLAP has two terms to refer to the

source tables.

- Fact table - is used to define the measures
- Dimension table - stores the data used to define dimensions

In data warehouse, the dimensions are pieces of data which allows us to understand and index measures in the data models. Dimensions are either characteristics of a measure or pieces of data which helps to contextualize the fact.

- Dimensions example
-

a) Product which was sold online
b) Product color
c) Product name
d) Name of the customer purchased the product
e) Name of the employee sold the product
f) Store location
g) Warehouse location

- **Dimensions**
- **Attributes**
- **Measures**
- **Hierarchies**

1. Dimensions define the basic business attributes like we want to analyse. It refers to customers, products, and time.

2. The attributes within a dimension define the columns within a dimension which are used for analysis like CustomerName, ProductName, City, PortalCode and OrderDate.

3. Measures are the set of aggregates which we can calculate such as sum of total orders, number of orders.

4. Hierarchies allows us to define a navigation structure within a dimension such as
   - Customers within cities within states within countries
   -Calendar months within the calendar quarter within the calendar years.
- Fiscal months within the fiscal quarter within fiscal year

Cube - A cube contains of the analysis objects you define with the highest level of security that can be assigned. Within SQL Server Analysis Services, we can multiple cubes for the users within the SQL Server analysis services database.

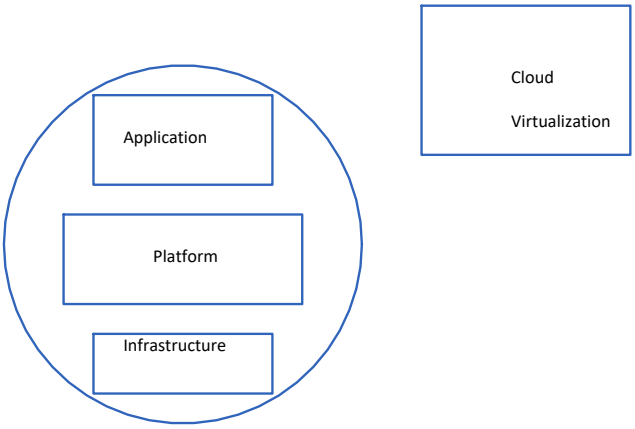
Cloud Fundamentals

12 December 2022 19:54

1. Cost efficiency
2. Security
3. Flexibility
4. Mobility
5. Insights
6. Increased collaboration
7. Quality Control
8. Disaster Recovery (infrastructure, platform, application, database, network)
9. Loss Prevention (any data deleted should be able to recover it)
10. Automated Software Updates
11. Sustainability

By definition, the Cloud computing refers to the internet based computing. Cloud computing refers to the components and sub-components required

- Front-end (thin client (web apps), thick client (desktop apps))
- Backend (servers, storage)
- Cloud based network (internet, intranet & intercloud)



Benefits of Cloud Computing

1. Cloud is easy to access for everyone. All the resources servers, disk, network, IP address, storage, databases, data warehouses etc. all accessible to everyone globally through internet.
2. Developing new applications, services, storage, databases are easier to build onto cloud.
3. It's useful to migrate the existing applications, databases, storage to the cloud as per supportability of the cloud vendor.
4. Software on demand.
5. Cost effectiveness - **Pay as you go**. (pay by hour)
6. Analysis of data
7. Streaming of high quality OTT resources (audio/video streaming)

Cloud Migration - Capex (capital expenditure) -> Opex (operational expenditure)
(on -prem env) (cloud env)

Features of Cloud

a) Scalability - increase the number of instances or sizes of server, storage or databases on cloud.
- **horizontal scalability (add more server instances)**
1.1 Scale out - add more server instances
1.2 Scale in - reduce the number of server instances
- **Vertical scalability (add more compute capability - more CPU core, memory, disk storage etc.)**
1.1 scale up - add more compute capacity to the server instances
1.2 scale down - reduce the compute capacity to the server instances

As per the Cloud design principles, **horizontal scalability (scale out) is recommended to go for in business.**

b) **SLA - Service Level Agreements -** 99.99% of the availability will be there all the time of the Azure SQL db instance.
- there will be only 0.001sec of downtime per sec/week/month basis will be there.

- Associated with each and every cloud provider (Amazon web services, Microsoft Azure, Google Cloud platform, IBM cloud, Oracle Cloud)
- Cloud provider can compensate for any downtime for any missed SLA / guaranteed uptime.
- This is the part of cloud based business model for the cloud provider to their customer

Different types of Cloud

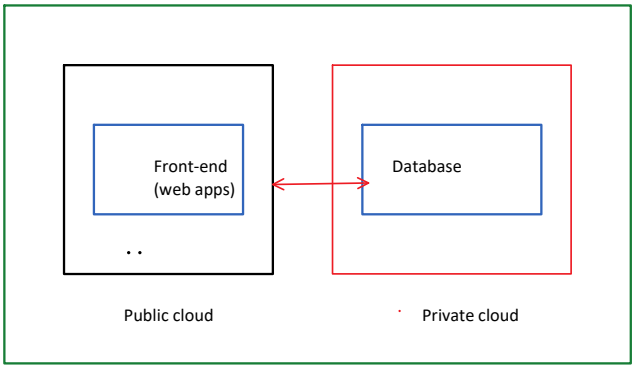
Public Cloud
Private Cloud
Hybrid Cloud

Cloud Models

IaaS (Infrastructure as Service)
PaaS (Platform as Service)
SaaS (Software as Service)

Different types of Cloud Computing Platform

	Public Cloud	Private Cloud	Hybrid Cloud
Definitions	Public cloud is open to all to store and access information via the internet, using pay-as-you-go model. In Public cloud, computing resources are managed and operated by the Cloud Service provider (i.e. Microsoft Azure, Amazon Web Services, Google Cloud platform etc.)	Private cloud is known the organization's internal cloud. The applications deployed over the private cloud are accessible only within the private network.	It's the combination of both public and private cloud. Hybrid cloud = public cloud + private cloud
Features	Public cloud is secure with the cloud controls (policy, Certificates, keys/passwords)	Private cloud is the most secure cloud platform since the applications are being deployed into the organization's own server and datacenters.	Hybrid cloud is partially secure because the services which are running on the public cloud can be accessed by anyone. While the services which are running on a private cloud can be accessed only by the org's users.
Benefits	Public cloud can be owned at a lower cost than the private and hybrid cloud	Private Cloud provides high level of security and privacy to the users	Hybrid cloud is suitable for organizations which require more security than the public cloud
Benefits	Public cloud is maintained by the cloud service provider (CSP) and users do not need to worry about the maintenance of the cloud	Private cloud offers better performance with improved speed and space capacity.	Hybrid cloud helps to deliver new products and services more quickly
Benefits	Public cloud is easier to integrate, offers a better flexibility approach to the customers, public cloud is delivered through internet, apps, databases, virtual machines are accessible over the internet.	Each organization has their own private cloud where they have full control over the cloud because it's managed by the organzation itself.	Hybrid cloud offers flexible resources because public cloud and secure resources because of private cloud.
Cons	Public cloud is less secure because resources are shared publicly	Private cloud is accessible only within the organization level, the area of operations in private cloud is limited.	Security feature is not good as private cloud. Managing a hybrid cloud is complex because it's difficult to manage more than one type of deployment model.
cons	Performance of the public cloud resources Depends upon the high-speed network linked to the cloud provider. The client no control to the data underlying to the infrastructure	Private cloud is not suitable for organizations which has a high user base and organizations that do not have the pre-built infra, sufficient developers and can manage the cloud platform	In the Hybrid Cloud, the reliability of the services depends on the cloud providers
examples	Microsoft Azure, AWS, GCP, IBM bluemix, Oracle Cloud	HP data centers, Azure Stack, Vmware private cloud, Ubuntu canonical private cloud	Openstack, Azure hybrid cloud



Hybrid Cloud Platform

Different types of Cloud Models

	Cloud Model	Benefits	Examples
IaaS (Infrastructure as Service)	Rent for respective infrastructure required for business. End users are responsible to manage the underlying infrastructure - Compute - Cpu	Less complex Less development cycle	Azure Virtual Machine Azure Virtual Network Azure Disk

	- Memory - Disk storage - Containers - network		
PaaS (Platform as a Service)	Managed services (Compute, storage, database, data warehouse, analysis services) -- end users are not responsible to manage infrastructure in PaaS To manage - Compute - Disk storage - Network - OS - OS update/patch management - Security of the infra managed by the cloud providers Only responsible to manage their application on Azure	Less burden on infra provisioning Developers can focus more into their application or business logic Development language agnostic (.net, java, springboot, react/angular, python, ruby etc.)	Azure App Service (Web apps) Azure Storage Azure SQL database Azure Hadoop Services Azure Analysis Services Azure Data Lake Services etc.
Software as Service (SaaS)	Managed services, End users/customers are not responsible for the application development and deployment.	Almost zero burden on the end user in terms of app development. No burden on app scalability, reliability, disaster recovery, app backup or restore	Office 365 Gsuite gmail, Google drive Salesforce

Hands-on Lab

Creation of Azure Storage Account

1. Login to Azure portal (https://portal.azure.com)

Azure Fundamentals

12 December 2022 19:55

Basics of PowerShell Scripting

12 December 2022 19:55

Intro to Big Data

12 December 2022 19:55

Apache Hadoop Overview

12 December 2022 19:55

Apache Hadoop (Deep Dive)

12 December 2022 19:56

Azure Data Factory

12 December 2022 19:56

Azure Data Lake Gen2

12 December 2022 19:56

Azure SQL Database

12 December 2022 19:57

Azure Blob Storage

12 December 2022 19:57

Azure Analysis Services

12 December 2022 19:57

Azure Synapse Analytics

12 December 2022 19:57

Case Study

12 December 2022 19:57

Apache Spark

12 December 2022 19:58

Python Programming

12 December 2022 19:58

Azure Databricks

12 December 2022 19:58

Overview of AWS

12 December 2022 19:58

Overview of GCP

12 December 2022 19:58

Case Study

12 December 2022 19:58

L1 Preparation

12 December 2022 19:59