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.

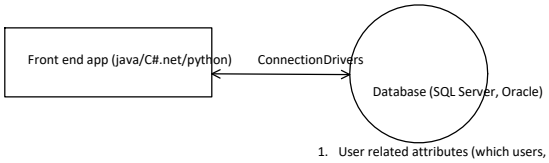
Azure Subscription (trial)
-- 12 months of free service + 30 days of 200 USD credit
-- enterprise subscription

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)

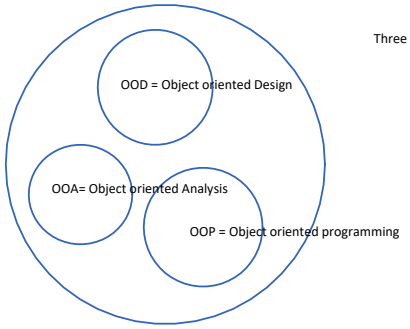
1960 (IBM) - developed the integrated Management system which is based on hierarchical database model.

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

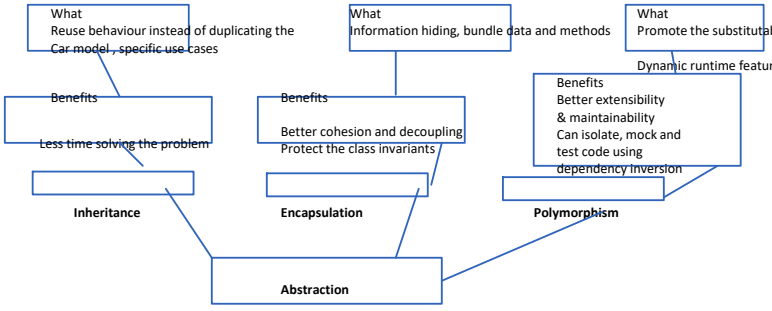
- Abstraction
- Encapsulation
- Inheritance
- Polymorphism



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

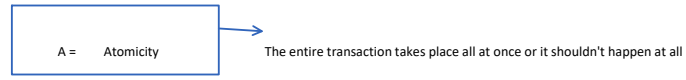
```
Public interface Topic
{
 void understand();
}

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();
}
}
```

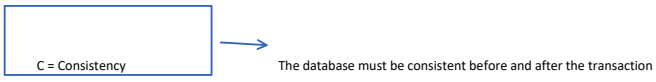
e.g. microservices application design

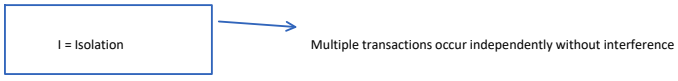
Empaddr and emp classes are separate classes where change of a simple address of a emp entity / object does not affect to the emp class and its related methods.

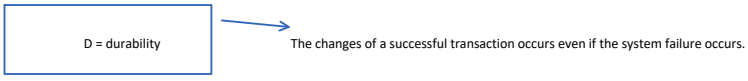
High coupling scenario - it would make difficult to change or maintain the code, since classes are closely knit together , making a change could require an entire system revamp.

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









Different Types of Databases

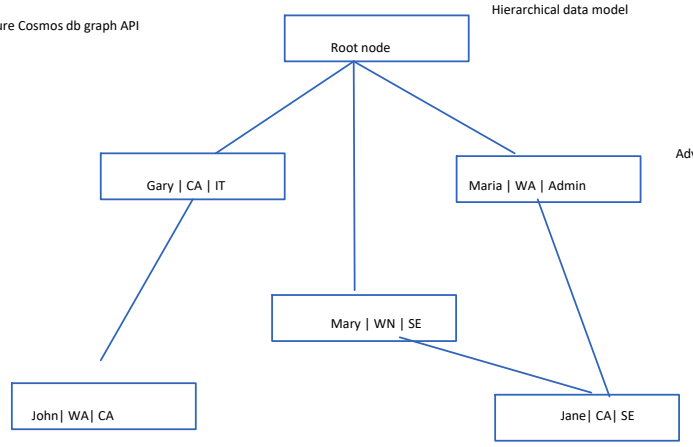
1. **Flat file database** -- kind of text database where each line of the plain text file holds only a single record. (e.g. MS Access, MS Excel)
2. **Hierarchical database** -- based on hierarchical data model, where the data is viewed as a collection of tables, data is designed into a tree like structure 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

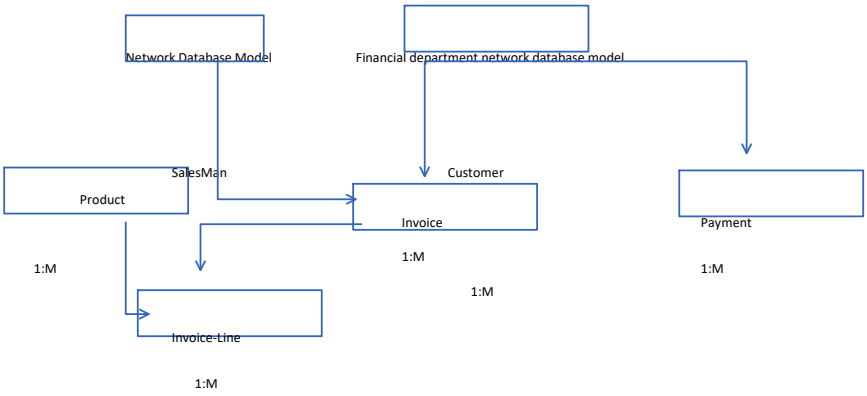


Advantages

- 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.

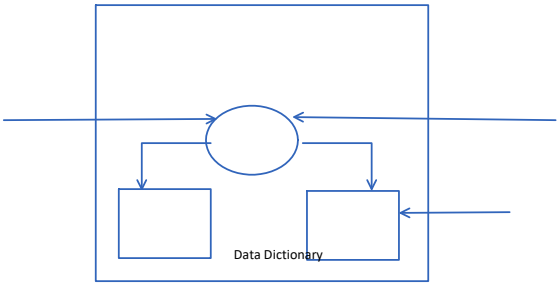
Types of Data Dictionary

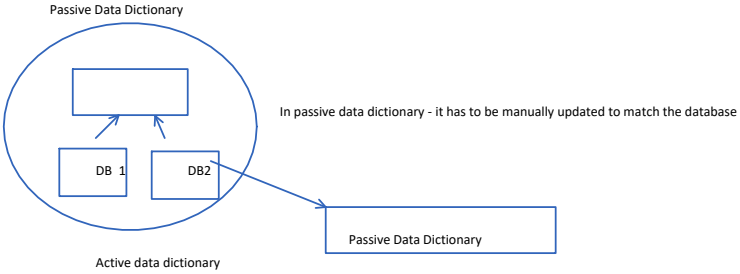
1. Active Data Dictionary -- self updating

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

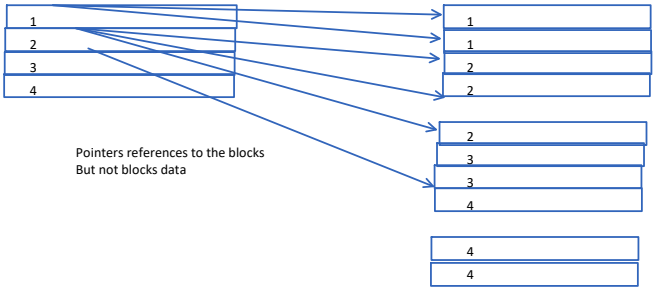
1. An Index can help to speed up the select queries and where clauses.
2. Indexes can be created or dropped with no effect on the data.
3. When an index is created, it includes a column contains a wide range of values.

Create index index_name on table_name column_name;

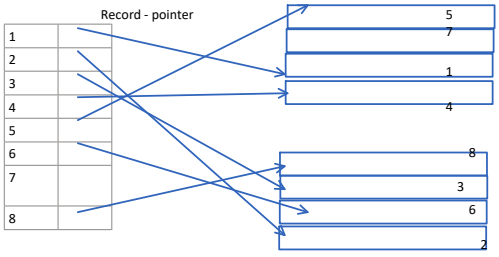
Table - table_name
Column - column_name

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

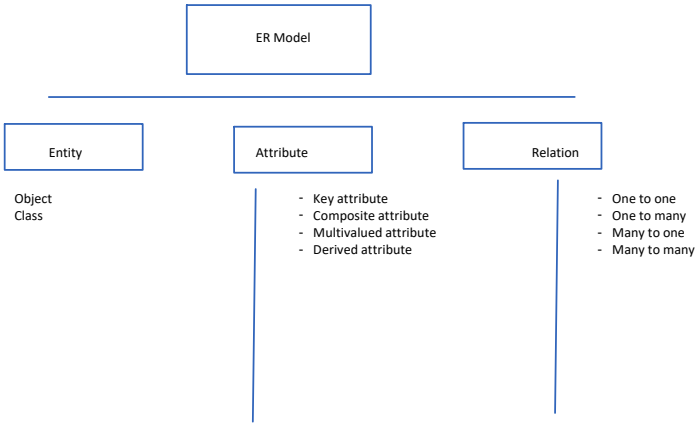
Item1
Item2
Item3
Item4
Item5
Item6
Item7
Item8

OFFSET 3 ROWS

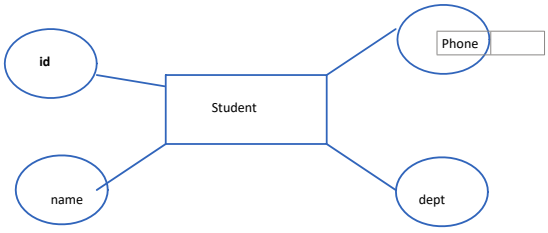
FETCH 5 ROWS

Benefits

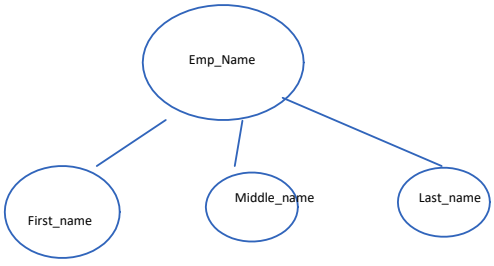
1. There's no direct information related with the table, so the changes are only based on the requirements of the application
2. Performance enhanced as the value of the key is relatively smaller
3. The key value is guaranteed to contain unique information
4. As it holds smaller constant values, the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



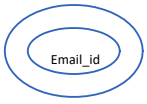
a) Key Attribute - key attribute is used to represent the main characteristics of an entity. It represents the primary key.



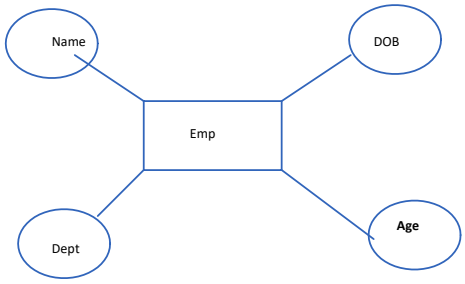
b) Composite attribute - composed of many other attributes altogether is called as composite attribute. The composite attribute is multiple simple attributes joined together.

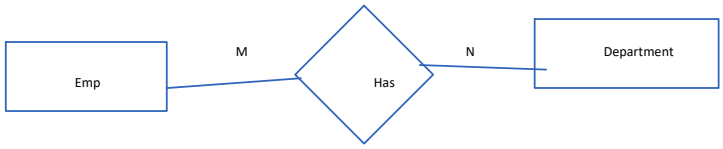


c) Multi-valued attribute - this attribute can have more than one value. These attributes are called as multivalued attributed.

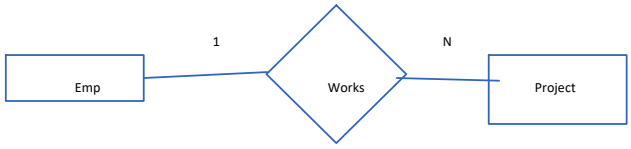


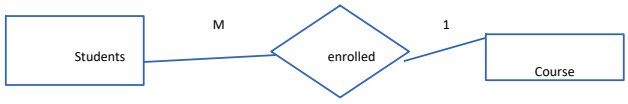
d) Derived attribute - An attribute can be derived from other attribute,





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

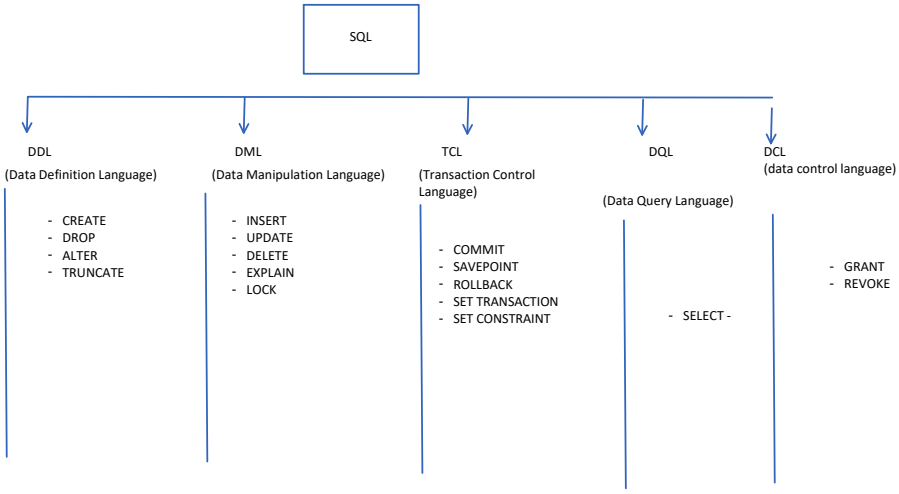
Identity property on columns can be defined to have a numeric data type. When the IDENTITY property is defined, SQL server manages the values in the columns on behalf of user.

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

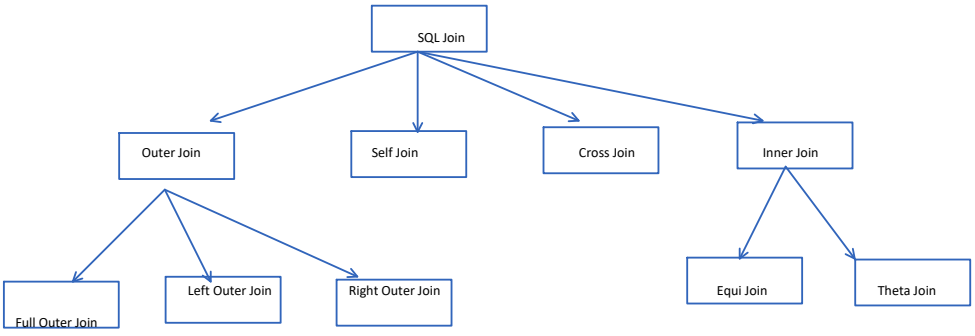
A database schema is considered the blueprint of a database object which describes how the data may relate to other tables or other data models.

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

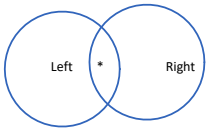
A SQL Join is a special form of generating a meaningful data by combining multiple tables relate to each other using a 'Key'. Typically, relational tables, must be designed with a unique column and this column is used to create relationships with one or more other tables. When you require a result-set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



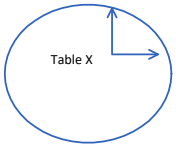
SQL Inner Join - The most simple and common form of join which is default join type.

- 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

SQL Data Warehouse

12 December 2022 19:54

Cloud Fundamentals

12 December 2022 19:54

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