**SQL SERVER**

Database : It is like a store. Contains various types of data/information, in the form of tables or in a predefined format so that it can be easily retrieved.

Data : Raw facts/ figures. Never provides proper meaning.

Consider the numbers "3," "7," and "5." These individual numbers are data. Without context, they are just isolated values with no inherent meaning.

Information : processed data. Gives meaning to us.

If we take the numbers "3," "7," and "5" and interpret them as the scores of three students in a class, we can derive information. For example, "The average score is 5, the highest score is 7, and the lowest score is 3." This organized and contextual representation is information because it provides meaningful insights about the scores.

Data becomes information when it is interpreted, analyzed and presented in a way that adds value and makes it understandable.

DBMS : Database Management System

DBMS is used to store and manipulate data/information in a database. It is a type of software present in database to maintain and manage the data present in database.

|  |  |
| --- | --- |
| File system | DBMS |
| Stores files in storage medium within a computer | A software to manage the database. |
| Redundant data can be there | No redundant data |
| No inbuilt mechanism for backup and recovery. | Backup and recovery is provided. |
| No efficient query processing | Query processing is there. |
| Less data consistency ( no accuracy, no reliability and no integrity) | Consistency is maintained as normalisation is there(accurate, reliable and maintainable.) |
| Less complex | More complex |
| Provide less security | Has more security mechanisms compared to file system. |
| Data sharing is not easy as data is present in multiple files | Due to its centralised nature, data sharing is easy |
| Gives the details of storage and representation. | Hides the internal details. |
| Cobol, C++ | Oracle, SQL Server |
| Suitable for small data. | Useful for large amount of data. |

Types of DBMS

1. File management system(FMS)/File database Management System(FDMS)

First model

Released in 1950’s

Data is always arranged in a continuos stream of characters or in sequential manner.

Disadvantage : Whenever we need to perform a search, we have to start the searching from the beginning of the file, which increases the searching time.

1. Hierarchy Management system(HMS) / Hierarchial Database management system(HDMS):

Developed by IBM in 1960’s.

They used this in Information management system (IMS) project developed by IBM.

Top level called root.

Advantage : easily retrieve values without wasting much time.

Disadvantages: Only one person can use the database at a time.

If want to change or add a new level in between existing levels, then user has to reconstruct the entire tree structure, but it is tedious and time consuming.

A diagram of a family tree

Description automatically generated

1. Network database Management system(NDMS)

Developed by IBM in 1969 also while developing IMS project.

Developed based on a operating system called MULTICS.

Has a similar structure as hierarchical model, but it is easier to add new level or new relationship in network model.

Each node can act as both parent and child, and allows for complex relationships and supports many-to-many relationships.

Has a centralised system of database, and works based on time sharing by allowing multiple users to work on same database concurrently.

It involves the fair allocation of processor time, task scheduling, transaction isolation, concurrency control, and resource management to ensure efficient and reliable operation in a multi-user environment.

MULTICS :

Multiplex Information Communication System

Uses Hierarchial Operating System.

Security was fundamental principle of MULTICS

Time sharing OS ( allows multiple users to use the same system simultaneously.)

Dynamic linking and shared libraries ( allowed programs to share code at runtime.)

Virtual memory, hierarchical file system and security features has a lasting impact on other upcoming OS.

Advantage : more than one person can use this database simultaneously.

Disadvantages: No proper security to centralised database

Redundancy is increased.

Occupies more memory and leads to decrease in system performance and increases the inconsistency.

Why there is redundancy issue in NDMS:

In the network model, records can act as both owners and members, creating a web of relationships. This flexibility can lead to situations where the same data is owned by multiple records, contributing to redundancy.

1. Relational database management system ( RDBMS)

Developed by german scientist E.F.Codd in 1970.

E.F codd also specified some rules and criteria with which a DBMS can be considered as a RDBMS.

He introduced these rules to maintain the data in the database to ensure the consistency, integrity and reliability for the data in the RDBMS.

He defined 12 rules named as Codd rules.

They are :

1. Information rule

All the information should be stored in one and only way.

Ex : values in table

1. Guaranteed Access rule.

Each piece of data must be accessed by without any ambiguity.

1. Systemic treatment of null values.

The DBMS must allow each field to be null or undefined.

Null values means missing or undefined information and that also needs to be treated systematically with support of operations involving null values.

1. Dynamic online catalog data based on relational model

The structure and metadata of the database must also be maintained same as the relational data in the form of tables and defnitions. This data also must be accessed with the same relational data.

1. Comprehensive data sublanguage rule

Must support a relational data sublanguage.

Example : SQL

1. View Updating Rule

Users should be able to update the view as if they were updating the underlying base tables.

1. High level Insert, update and delete

Users can insert, update or delete data from the DBMS, without specifying the physical implementation details.

1. Physical data independence

The logical structure of data shown to the user should not be updated, though we change data in physical storage.

1. Logical data independence

Changes to logical data(table structure, domains etc) should not affect the application

1. Integrity independence

Integrity constraints should be defined in a catalog(systematic list). So, that change to integrity constraints should not affect the application programs.

1. Distributed Independence

Should allow data to be distributed among multiple locations. Users should be able to access the distributed data with the same relational operations.

1. Non-subversion rule

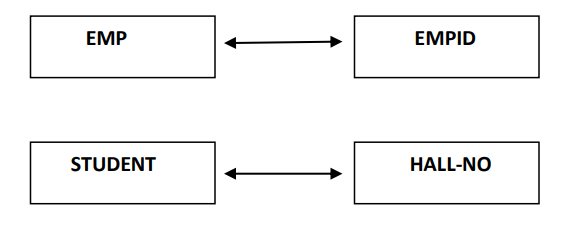
users should not be able to use the low-level language to perform operations that would violate the integrity or security rules established by the high-level language.

Relation : commonness between objects

Types of relations: ( degree of relationships)

1. One-One relationship

One object with other object



1. One-Many relationship

One object with multiple objects

A group of rectangular objects with black text

Description automatically generated

1. Many-Many relationship

Multiple objects have relationship with multiple objects.

A diagram of a network

Description automatically generated

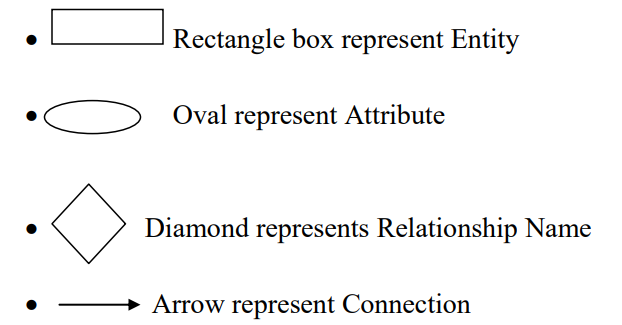
Properties of RDBMS:

1. Data is defined as tables
2. Tables are represented as rows and columns
3. Horizontal lines : rows/tuples/records
4. Vertical lines : columns/attributes/ fields
5. Cell : intersection of rows and columns
6. Actual data stored in cell
7. Table -> entity
8. Table should not contain any duplicate columns
9. No specific order is to be followed to insert records or define the column.
10. Should not contain any duplicate values.

ER diagram ( entity and relationship diagram)

Pictorial representation of database

Developed by Mr.chen



**SQL Server**

RDBMS product

Developed by Microsoft.

Provide GUI interface to interact with the database without remembering commands.

Run only in windows( platform dependent)

SQL server : collection of databases

Databases : collection of tables, procedures and functions

We use SQL Server Management Studio to connect to SQL Server.

To connect to SQL Server we see 4 options:

1. Server Type
2. Database engine : store, manage and access data from database.
3. Analysis Services : used for data warehouse it will show the data in

three dimensions (Rows, Columns and New dimension)

1. SQL Server Compact edition : develop mobile application

or mobile software.

1. Reporting Services : a reporting tool used to generate reports. Reports can include rich data visualization, including charts, maps etc
2. Integration services : to convert tables from relational database to another relational database for e.g. If we want to convert SQL Server tables to ORACLE tables or My SQL tables then will be used.
3. Server Name
4. Server Authentication
5. Windows Authentication: no required user name and password because operating system will generate User Id and Password by default.
6. SQL Server Authentication : work on the current user and when we work on SQL Server authentication then user should enter User Id and Password (These User ID and Password will give at the time of SQL Server installation)
7. Username and password.

Two types of Databases in SQL Server:

1. System database
2. Master : to manage system level information of SQL Server
3. Model : used as template for newly created databases.
4. Msdb : contains the SQL commands which are executed by user.
5. Tempdb: When ever SQL server is started tempdb will be created in SQL server. It is used to store temporary tables once we restart the server the tempdb database is destroyed.
6. User Database

created and manage by the user for storing their objects like tables, views, procedure etc.

When we create a database on SQL server, it will generate two database files

1. Primary database files

Contains startup information of database

To store database objects like tables and views.

Extension : .mdf file

1. Log file

Transaction query information

Extension : .ldf file

Root location :

C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\DATA

**Data types of SQL Server**

Attribute that specifies what data type.

1. Integer data type
2. TinyInt : 1byte(8 bits) [0 to 2^8-1]
3. SmallInt : 2bytes(16bits) [-2^15 to 2^15-1]
4. Int : 4 bytes ( 32 bits) [-2^31 to 2^31-1]
5. Bigint : 8 bytes(64 bits) [-2^63 to 2^63-1]
6. Decimal data type

To store fixed point decimal values

Commonly used when we need to represent values with a precission an scale

Precission : total number of digits in the number.

Scale : number of digits to the right of the decimal point.

Default precission : 18

Maximum precission : 38

Precission - Stored memory

1-9 - 5bytes

10-19 - 9bytes

20-28 - 13bytes

29-38 - 17bytes

1. Money Data type

To store currency values

Have fixed precission and scale.

Precission : 19

Scale : 4

1. Small money : 4bytes ( 32 bits) -> -2^31 to 2^31-1
2. Money : 8 bytes ( 64 bits) -> -2^63 to 2^63-1
3. Date and time datatype

Date : yy/mm/dd

Time: hh/mm/ss.ms

Date and time : both date and time of the day

Yy/mm/dd hh/mm/ss.ms

1. Character datatype:

To enter character values

1. Char(n)

Fixed length data type.

Store string values in non-unicode manner. (1 char in 1 byte)

Max length : 1 to 8000bytes.

1. Varchar(n)

Variable length data type.

Store string values in non-unicode manner. (1 char in 1 byte)

Max length : 1 to 8000bytes.

Varchar(MAX) : max=2^31-1 (can give when you don’t have the length of characters)

1. Text

Same as Varchar(MAX)

1. Nchar(n)

Fixed length data type

Store string in Unicode manner ( 1 char in 2 bytes)

Max length : 1 to 4000 bytes

1. Nvarchar(N)

Variable length data type

Store string in Unicode manner ( 1 char in 2 bytes)

Max length : 1 to 4000bytes

Varchar(MAX) : max=2^31-1 (can give when you don’t have the length of characters)

1. NText : Nvarchar (max) data type
2. Binary data type

To store images, videos and audio data.

1. Binary(n)

Fixed length data type.

1-8000 bytes

1. Varbinary(n/max)

Variable length data type

1-8000 bytes

1. Image

Same as varbinary(max)

**STRUCTURE QUERY LANGUAGE**

Non procedural language

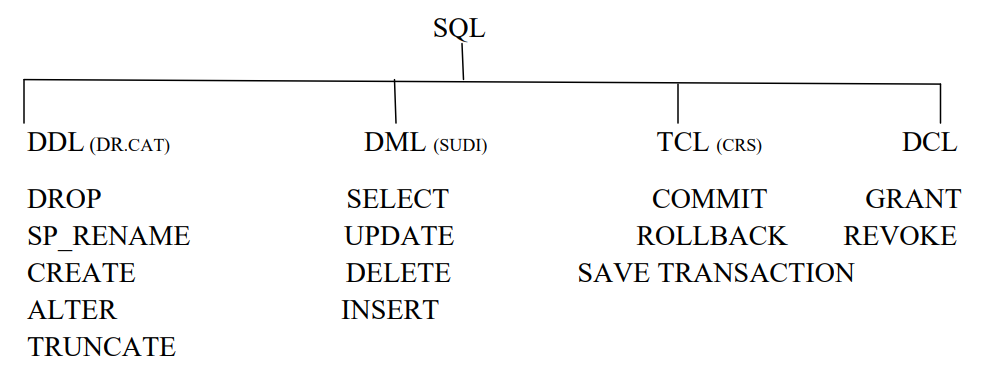
Used to communicate with oracle, sqlserver, etc.

Developed by Mr.E.F.Codd in 1968

Released by ANSI in 1972

Features

1. Not case sensitive
2. Every command of sql must end with semicolon. ( exemption for sql server)
3. Sequel ( structured query language)
4. Common language Interface, use dto communicate with any type of database.
5. Similar to normal English language.
6. Divided into 4 sublanguages
7. DDL (data definition language)
8. DML (data manipulation language)
9. TCL (Transaction control language)
10. DCL ( data control language)



Data definition language

1. Create : create database

Syntax : CREATE TABLE <TABLE NAME>

(COL 1 DATA TYPE (size),

COL2 DATA TYPE (size),

:

:

:

:

COLN DATA TYPE (size));

Example : CREATE TABLE EMP (EID Int, ENAME Varchar (15), SAL DECIMAL (6, 2));

1. Alter : modify the structure of a table
2. Alter-alter :

Syntax : ALTER TABLE <TABLE NAME> ALTER COLUMN <COLUMN NAME> DATA TYPE (SIZE)

Example : ALTER TABLE EMP ALTER COLUMN ENAME char (25);

1. Alter-add

Syntax : ALTER TABLE <TABLE NAME> ADD <COLUMNNAME>DATA TYPE(size);

Example : ALTER TABLE EMP ADD EADDRESS VARCHAR(MAX);

1. Alter-drop

Syntax : ALTER TABLE <TABLE NAME> DROP COLUMN <COLUMNNAME>

Example : ALTER TABLE EMP DROP COLUMN SAL;

1. Rename
2. Change the column name

Syntax : SP\_RENAME ‘TABLENAME.OLDCOLUMNNAME’, ‘NEW\_COLUMNNAME’, ‘OBJECT’;

Object : usually COLUMN(for renaming column.

Example : SP\_RENAME ‘EMP.SAL’,’EMP\_SALARY’,’COLUMN’;

1. Changing the table name

Syntax : SP\_RENAME ‘OLD TABLE NAME’,’NEW TABLE NAME’;

Example : SP\_RENAME ‘EMP’,’EMPLOYEE’

1. Truncate

delete all the records from existing table permanently.

Syntax : TRUNCATE TABLE TABLE\_NAME

Example : TRUNCATE TABLE EMPLOYEE

1. Drop

remove the table permanently from the database

Syntax : DROP TABLE <TABLE NAME>

Example : DROP TABLE EMP;

NOTE :

SP\_help: This command is used to see the structure of table

Syntax: SP\_help <table name>

Ex: SP\_help EMP

Data Manipulation Language

To manipulate data in database

There are four commands:

1. INSERT

To insert records to the existing table

Can insert records by two methods:

1. Implicit method

can enter the values into the required columns in the table,

so that user can omit (or) left some columns data while he enters the records

into the table

Automatically takes Null when user omits any data.

Syntax: INSERT INTO <TABLE NAME> (COL1, COL2….COLN) VALUES (VAL1, VAL2,… VALN);Ex : INSERT INTO EMP (EID, SAL) VALUES (106,9999);

1. Explicit method

Enter all the values into the columns without omitting any column data.

Syntax: INSERT INTO <TABLE NAME> VALUES <VAL1, VAL2, ….VALN>;

Ex: INSERT INTO EMP VALUES (101,’RAJ’,9500);

1. UPDATE

To modify the existing data.

Can modify all the records and also specific records using where clause.

Syntax : UPDATE <TABLE NAME> SET COL=VALUE

Ex : UPDATE EMP SET SAL=10000;

Syntax : UPDATE <TABLE NAME> SET COL1=VALUE,COL2=VALUE… WHERE CONDITION

Example : UPDATE EMP SET SAL=1000,NAME=”Ramesh”

1. DELETE

Delete all records

Syntax : DELETE FROM <TABLE NAME>

Delete a specific record

Syntax: DELETE FROM <TABLE NAME> WHERE CONDITION

Ex : DELETE FROM EMP WHERE NAME=’RAMESH’;

|  |  |
| --- | --- |
| DELETE | TRUNCATE |
| DML command | DDL command |
| Temporary deletion can use ROLLBACK, if not committed. | Permenant deletion, the changes cant be reverted back |
| Can delete a particular row | All the rows can only be deleted |

1. SELECT

Retrieve data from table.

Can retrieve all the rows or specific data based on condition.

whenever we need to check a condition, we need to use a special clause called ‘where’

Data can be retrieved by 3 ways:

1. Projection

Getting data from specific columns

Syntax: SELECT COL1,COL2……..COLN FROM <TABLE NAME>

Ex: SELECT EID,ENAME FROM EMP;

1. Selection

Getting data based on condition.

Syntax: SELECT \* FROM <TABLENAME> WHERE (CONDITION);

Ex: SELECT \* FROM EMP WHERE EID=101;

1. Joins

SELECT Customers.CustomerID, Customers.CustomerName, Orders.OrderID, Orders.OrderDate

FROM Customers

INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID;

Where Clause

To specify a condition to retrieve, update or delete records from the table.

We can apply where is SELECT, UPDATE, DELETE

SELECT \* FROM <TABLE NAME> WHERE <CONDITION>;

UPDATE <TABLE NAME> SET <COLUMN\_NAME>=COLUMN\_VALUE WHERE <CONDITION>;

DELETE FROM <TABLE NAME> WHERE <CONDITION>;

ALIAS

A duplicate name or alternate name for the original column name or table name or expression name.

SELECT EID AS “EMPLOYEE ID”, ENAME AS “EMPLOYEE

NAME”, SAL AS “SALARY” FROM EMP;

NOTE: In the above example the keyword ‘as’ is optional

EX: SELECT EID “EMPLOYEE ID”, ENAME “EMPLOYEE NAME”,

SAL “SALARY” FROM EMP;

NOTE: In the above example quotations is also optional but there should

not be space between column name

EX: SELECT EID EMPLOYEEID, ENAME EMPLOYEENAME, SAL

SALARY FROM EMP;

Note : We cannot check conditions on the alias name.

IDENTITY

Generate unique values in sequential interaction without user interaction.

Default value is IDENTITY(1,1)

Syntax : IDENTITY(seed,increment)

Example:

CREATE TABLE EMP (EID int IDENTITY(100,1),ENAME varchar(100));

System functions in SQL

Select function\_name <Expressions>;

1. Mathematical Functions

Perform operations based on input values and return a numerical value.

1. select ABS(-15)---- 15

select ABS(45)----- 45

1. select ceiling(15.000)----15

select ceiling(15.0001)----16

select ceiling(-12.34)-----(-12)

1. select floor(15.000)---15

select floor(15.0001)----15

select floor(-12.34)----(-13)

1. select SQUARE(5)---25
2. select SQRT(25)-----5
3. select SIGN(42)------------1

select SIGN(0)-------------0

select SIGN(-42)-----------(-1)

1. select PI()---------3.14159265358979
2. select LOG(2)------ 0.693147180559945
3. select LOG10(10)------1
4. select SIN(0)------0
5. select COS(0)-----1
6. select TAN(0)------0
7. String functions
8. Select ASCII (‘Z’) -----90
9. Select char(90)----Z
10. Select CHARINDEX (‘S’,’SUDHAKAR’) -------1
11. Select LEFT (‘SUDHAKAR’, 5) ----SUDHA
12. Select RIGHT(‘SUDHAKAR’,2)----AR
13. Select LENGTH(‘RAM’)-----3
14. Select LOWER(‘RAM’)-----ram
15. Select UPPER(‘ram’)----RAM
16. Select LTRIM(‘ ram’)-----ram
17. Select RTRIM(‘ram ’)------ram
18. Select REPLACE(‘JACK’,’J’,’BL’)----BLACK
19. Select REPLICATE(‘SAI’,3)-----SAISAISAI
20. Select REVERSE(‘HELLO’)-----OLLEH
21. Date and time functions
22. Select GETDATE () ------- 2024-02-01 15:35:22.670
23. Select day(getdate())-----01
24. Select month(getdate())-----02
25. Select year(getdate())-----2024
26. Select GETUTCDATE() ---- Returns the date time value in UTC format
27. Conversion Functions
28. Cast

Select CAST (10.2587 as Int) 🡪 10

1. Convert

Select Convert (Varchar (24), get date (), 113)

1. Aggregate Functions / Group functions

Perform calculations on a set of values and return a single value.

1. Select sum(salary) from Emp;
2. Select avg(salary) from emp;
3. Select max(salary) from emp;
4. Select min(salary) from emp;
5. Select count(\*) from emp; (number of records in table)

Select count(ename) from emp; (total number of rows in column)

Select count(distinct ename) from emp; (returns number of records without null and duplicate values).

Operators in SQL

Performs specific operations on expressions and operands.

Types:

1. Assignment Operator ( = )

Select ename from employee where salary=1000;

Update emp set ename=’Ram’ where salary=1000;

Delete from emp where eid=12;

1. Arthimetic Operator

Select 100+250

Select 245-400

Select 20\*20

Select 25/5

Select 37%6

Select 20/5+20/5

Select 35.50+20

update Employee set class=

Case

when esalary>50000 then 'Manager'

when esalary<50000 then 'Employee'

Else'Fail'

End

1. Comparison operator

are > (Greater Than)

< (Less Than)

>= (Greater Than or Equal To)

<= (Less Than or Equal To)

!= (Not Equal To)

!< (Not Less Than)

!> (Not Greater Than)

Ex : Select ename from EMP where salary

1. Logical Operator

Return Boolean data type

AND  
OR  
NOT  
BETWEEN

NOT BETWEEN  
LIKE  
NOT LIKE  
IN  
NOT IN  
EXISTS  
NOT EXISTS

ANY

ALL

SOME

1. Set operator

Combines results of two or more queries into a single query.

Rules : The column names and order should match and the data types has to be compatible.

1. UNION: it combines the result of two or more select statements into a single result set that includes all the records belongs to all queries except duplicate values.
2. UNION ALL: it is same as union but returns duplicate values
3. INTERSECT: INTERSECT returns any distinct values that are common in left

and right tables.

1. EXCEPT: EXCEPT returns any distinct values from the left query that are not

found on the right query

**Clauses in SQL**

1. Where

SELECT \* FROM EMP WHERE SAL=1000

1. Order by

To sort or arrange data in table

Descending order : desc

Ascending order : no need to mention explicitly

SELECT \* FROM EMP ORDER BY EID (For Ascending Order)

SELECT \* FROM EMP ORDER BY ENAME DESC (For Descending Order)

1. TOP N

Select top 3 \* from Employee;

1. Group by

Used when we need to group similar kind of data.

When we use group by we use group by functions like count(),sum(),max(),min(),avg().

If we use group by clause in the query, first the data in the table will be divided into different groups based on the columns and then execute the group function on each group to get the result.

1. Having clause

filtering and restricting the records in a table just like where clause

SELECT DEPT, COUNT=COUNT (\*) FROM EMP GROUP BY DEPT

HAVING COUNT (\*) >3

|  |  |
| --- | --- |
| Where Clause | Having clause |
| Filter and restrict records before grouping | Filter and restrict records after grouping |
| If we want to apply condition on a column with aggregate function where cant be used | If we want to apply condition on a column with aggregate function where can be used |
| Can apply without Groupby | With groupby |
| Applied on rows | Applied on groups and restrict groups |
| Cant support group functions | Can support group functions |

**SYNONYM**

A database object which can be created as an alias for table, view, procedure, etc.

Any DML operations performed on the synonym will directly affect the base table.

A synonym is created on entire table, not on a partial table.

When we create a synonym on another synonym then DML operations don’t affect the base table as synonym chaining is not allowed.