**SQL**

* SQL: -- Structed Query Language
* Sql is standard language for storing, manipulating and retrieving data in database.
* Sql became a standard of the American National Standards Institute(ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987.
* Sql is the standard language for Relation Database System

1) What is Data?

SQL Data type is an attribute that specifies the type of data of any object

2) What is Field?

The Field function returns the index position of a value in a list of values. This function performs a case-insensitive search.

3) What is Record?

A record is a data structure that can hold data items of different kinds. Records consist of different fields, similar to a row of a database table.

**Database**

A database is systematically organized storage of information, and it allows easy insertion, updating, analysis, and retrieval of data.

A database can be defined as a collection of a coherent, meaningful set of data. The term – ‘collection of coherent data needs to have a point of reference to be understood. A simple point of reference would be the example of postal address, which usually contains:

1. Building name
2. Flat number
3. Road name
4. State name
5. Pin code
6. Country code

SQL is used by many databases like My SQL, Oracle, SQL Server, PostgreSQL, etc.

There are lots of different database systems such as:

1. Bibliographic
2. Full-text
3. Numeric
4. Images

Basic Concepts and Advantages of DBMS:--

* A DBMS is system software for creating and managing databases.
* The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data.

A DBMS manages data and has many benefits. These are:

1. DATA INDEPENDENCE:

Application programs should be as free or independent as possible from details of data representation and storage.

DBMS can supply an abstract view of the data for insulating application code from such facts.

Two types are:

A) Physical data independence

B) Logical data independence

1. EFFICENT DATA ACCESS:

DBMS utilizes a mixture of sophisticated concepts and techniques for storing and retrieving data competently.

This feature becomes important in cases where the data is stored on external storage devices

1. DATA INTEGRITY AND SECURITY:

The data integrity provides protection for the existence of the database and maintaining the quality of the database.

If data is accessed through the DBMS, the DBMS can enforce integrity constraints on the data.

1. DATA ADMINISTRATION:

When several users share the data, integrating the administration of data can offer significant improvements.

1. DATA AVAILABILITY:

The data availability is responsible for the cost performance and the query update.

Availability functions make the database and available to users helps in defining and creating a database and getting the data in and out of a database.

The DBMS is used in the following:

1. Airlines : reservations, schedules, etc
2. Telecom : calls made, customer details, etc
3. Universities : registration, results, grades, etc
4. Sales : products, purchases, customers details, etc
5. Banking : all transactions etc

COMPONENTS OF DBMS:

* Users

Users may be of any kind such as DB administrator, system developer, or database users.

* Database application

Database application may be personal, organization’s, departmental.

* DBMS

Software that allows users to create and manipulate database access.

* Database

Collection of logical data as a single unit.

Relational Database Management System (RDBMS)

A RDBMS is a collection programs and capabilities that enable IT teams and others to create, update, administer and otherwise interact with a relational database.

* RDBMS store data in the form of tables, with most commercial rdbms using SQL to access the database.
* The RDBMS is the most popular database system among organizations across the world.
* It provides a dependable method of storing and retrieving large amounts of data while offering a combination of system performance and ease of implementation.
* An RDBMS is a type of DBMS that stores data in a row-based table structure which connects related data elements.

|  |  |  |
| --- | --- | --- |
|  | DBMS | RDBMS |
| Number of allowed users | It can only accept one user at a time | It can operate with multiple users |
| Hardware and Software requirements | It needs less software and hardware than RDBMS | It needs more software and hardware than DBMS |
| Database structure | Data is kept in a hierarchical form | It utilizes a table where the headers are used as column and rows contain the corresponding values. |
| Distributed database | It will not support | It will support |
| Database Normalization | It cannot be normalized | It will be normalized |
| Amount of data | It can only manage small amounts | It can handle any amount of data, from small to large. |
| ACID implementation | It do not use the ACID model for storing data | It base the structure of their data on the ACID model to ensure consistency |

And in the above table the ACID is equal to:

A—Atomicity

C—Consistency

I –Isolation

D—Durability

Features of RDBMS:

* The most basic RDBMS functions are related to create, read, update and delete operations.
* These form the foundation of a well-organized system that promotes consistent treatment of data.
* It is typically provides data dictionaries and metadata collection that are useful in data handling.
* These programmatically support well-defined data structures and relationships.
* The most common means of data access for the RDBMS is SQL
* Its main language components comprise data manipulation language and data definition language statements.
* It uses complex algorithms that support multiple concurrent user access to the database while maintaining data integrity
* It supports the work of database administrators who must manage and monitor database activity.
* Utilities help automate data loading and database backup.
* It manages log files that track system performance based on selected operational parameters.
* It provides graphical interfaces that help DBAs visualize database activity.
* ACID will particularly suited RDBMSes for handling business transactions.
* It is an intrinsic to operations of a variety of enterprise applications and are at the center of most master data management systems.

Uses of RDBMS:

* It is frequently used in disciplines such as manufacturing, human resources and banking.
* The system is also used for

Airlines, university, etc…

* Examples :

Oracle, MySQL, Microsoft SQL Server and PostgreSQL

Advantages of RDBMS:

* The use of an RDBMS can be beneficial to most organizations.
* The systematic view of raw data helps companies better understand and execute the information while enhancing the decision-making process.
* The use of tables to store data also improves the security of information stored in the databases.
* Users are able to customize access and set barriers to limit the content that is made available.
* It make it easy to add new data to the system or alter existing tables while ensuring consistency with the previously available content.

Other advantages

* Flexibility

Upload data is more efficient since the changes only need to be made in one place.

* Maintenance

Database administrators can easily maintain, control and update data in the database.

* Data structure

The table format used in RDBMSes is easy to understand and provides an organized and structural manner through which entries are matched by firing queries.

Drawback:

* The character limit placed on certain fields in the tables and the inability to fully understand new forms data such as complex numbers, designs and images.

Works:

* Each system will have varying numbers of tables with each table possessing its own unique primary key.
* The primary key is then used to identify each table.
* The rows are known as records or horizontal entities
* The columns are known as vertical entities and possess information about the specific field.

Assuring the integrity of data includes several specific tests, including entity, domain, referential and user-defined integrity.

* Entity integrity confirms that the rows are not duplicated in the table.
* Domain integrity makes sure that data is entered into the table based on the conditions.
* Referential integrity ensures that any row that is re-linked to different table cannot be deleted.
* User-defined integrity confirms that the table will satisfy all user-defined conditions.

Before creating these tables, the RDBMS must check the following Constraints:

1. Primary key

This identifies each row in the table.

One table can only contain one primary key.

The key must be unique and without null values.

1. Foreign key

This is used to link two tables.

The foreign key is kept in one table and refers to the primary key associated with another table.

1. Not null

This ensures that every column does not have null value, such as an empty cell.

1. Check

This confirms that each entry in a column or row satisfies a precise condition and that every column holds unique key.

1. Data integrity

This integrity of the data must be confirmed before the data is created.

Normalization

* The word normalization and normal form refer to the structure of a database
* Normalization was developed by IBM researcher **E.F.Codd** in the 1970.
* Normalization increases clarity in organizing data in Databases.
* Normalization of a database is achieved by following a set of rules called “Forms” in creating the database.

Normalization is the process of organizing the data in the database.

Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like insertion, update and deletion.

Normalization divides the larger table in smaller table and links them using relationship.

The normal form is used to reduce redundancy from the database tables.

There are different types of normal forms:

1. 1NF

A relation is in 1nfnif it contains an atomic value.

It must hold single-valued attribute.

First normal form disallows the multi-valued attribute.

1. 2NF

A relation will be in 2nf if it in 1nf and all non-key attributes are fully functional dependent on the primary key.

1. 3NF

A relation will be in 3nf if it is in 2nf and no transition dependency exists.

3nf is used to reduce the data duplication.

1. Boyce Codd Normal Form(BCNF)

BCNF is the advance version of 3nf.

It is stricter than 3nf.

A table is in BCNF if every functional dependency X🡪Y, X is the super key.

1. 4NF

A relation will be in 4nf if it is in Boyce Codd normal form and has no multi-valued dependency.

For a dependency A🡪B, if for a single value of A, multiple values of B exists, then the relation will be multi-valued dependency.

1. 5NF

A relation will be in 5nf if it is in 4nf and not contains any join dependency and joining should be lossless.

5nf is satisfied when all the tables are broken into as many tables as possible.

5nf is also known as Project-join normal form (PF/NF).

SQL SERVER

SQL Server is software developed by Microsoft.

* It is also called as MS SQL Server.
* It is implemented from the specification of RDBMS.

Features of MS SQL Server:

1. MS SQL Server is highly scalable
2. MS SQL Server is platform independent
3. MS SQL Server can be run on a single laptop or a network of cloud servers.
4. MS SQL Server is both a GUI and command based software
5. MS SQL Server supports SQL language which is an IBM product, non-procedural, common database and case insensitive language.

Basics of SQL Server:

1. Querying data

* Select 🡪 it used to select data from a database.

1. Sorting data

* Order by 🡪 sort the result set in ascending or descending order

Ascending ----- default

Descending ----- use the key DESC

1. Limiting rows

* Offset fetch 🡪 limit the number of rows returned by a query
* Select top 🡪 limit the number of rows or percentage of rows returned in a query’s result set

1. Filtering data

* Distinct 🡪 select distinct values in one or more columns of a table
* Where 🡪 filter rows in the output of a query based on one or more conditions
* And 🡪 combine two Boolean expressions and return true if all expressions are true
* Or 🡪 combine two Boolean expressions and return true if either of conditions is true
* In 🡪 check whether a value matches any value in a list or a subquery
* Between 🡪test if a value is between a range of values
* Like 🡪 check if a character string matches a specified pattern
* Column 🡪show you how to use column aliases to change the heading of the query output
* Table aliases 🡪 table alias to improve the readability of a query

1. Joining data

* Joins 🡪 a join clause is used to combine rows from two or more tables, based on a related column between them.
* Inner join 🡪select rows from a table that have matching rows in another table
* Left join 🡪return all rows from the left table and matching rows from the right table.

The result is null from the right side, if there is no match

* Right join 🡪 return all rows from the right table and matching rows from the left table.

The result is null from the left side, if there is no match

* Full outer join 🡪return matching rows from both left and right tables, and rows from each side if no matching rows exist
* Self join 🡪 a self join is a regular join, but the table is joined with itself
* Cross join 🡪join multiple unrelated tables and create Cartesian product of rows in the joined tables.

1. Grouping data

* Group by 🡪group the query result based on the values in a specified list of column expressions
* Having 🡪 specify a search condition for a group or an aggregate function.
* Grouping sets 🡪generates multiple grouping sets
* Cube 🡪generate grouping sets with all combinations of the dimension columns
* Rollup 🡪generate grouping sets with an assumption of the hierarchy input columns.

1. Subquery

This section deals with Subquery which is a query nested within another statement such as SELECT, INSERT, UPDATE or DELETE statement.

* Subquery 🡪it explains the Subquery concept and show you how to use various Subquery type to select data
* Exists 🡪test for the existence of rows returned by a Subquery
* Any 🡪returns true if any of the Subquery values meet the condition
* All 🡪returns true if all of the Subquery values meet the condition

1. Set operations

* Union 🡪combine the result sets of two or more queries into a single result set.
* Intersect 🡪return the intersection of the result sets of two or more queries.
* Except 🡪find the difference between the two result set of two input queries

1. Common table expression(CTE)

* CTE 🡪use common table expressions to make complex queries more readable
* Recursive CTE 🡪query hierarchical data using recursive cte.

1. Pivot

* Pivot 🡪coverts rows to columns.

1. Modifying data

* Insert 🡪insert a row into a table
* Insert multiple rows 🡪insert multiple rows into a table using a single INSERT statement
* Insert into select 🡪insert data into a table from the result of a query
* Update 🡪change the existing values in a table
* Update join 🡪update values in a table based on values another using joins clause.
* Delete 🡪delete one or more rows of a table
* Merge 🡪walk you through the steps of performing a mixture of insertion, update and deletion using a single statement

1. Data definition

* Create database 🡪show you how to create a new database in a SQL server instance using the CREATE database statement and SQL server management studio
* Drop database 🡪learn how to delete existing databases
* Create schema 🡪 describe how to create a new schema in a database
* Alter schema 🡪 show how to transfer a securable from a schema to another within the same database
* Drop schema 🡪learn how to delete a schema from a database
* Create table 🡪walk through the steps of creating a new table in a specific schema of a database
* Identity column 🡪learn how to use IDENTITY property to create the identity column for a table
* Sequence 🡪describe how to generate a sequence of numeric values based on a specification
* Alter table add column 🡪show you how to add one or more columns to an existing table
* Alter table alter column 🡪show you how to change the definition of existing columns in a table
* Alter table drop column 🡪learn how to drop one or more columns from a table.
* Computed columns 🡪to reuse the calculation logic in multiple queries
* Drop table 🡪show you how to delete tables from the database
* Truncate table 🡪delete all data from a table faster and more efficiently
* Select into 🡪learn how to create a table and insert from a query into it
* Rename a table 🡪walk through the process of renaming a table to a new one
* Temporary table 🡪for storing temporarily immediate data in stored procedures or database session

1. Expressions

* Case 🡪add if-else logic to SQL queries by using simple and searched case expressions
* Coalesce 🡪handle null values effectively using the coalesce expression
* Null if 🡪return null if the two arguments are equal; otherwise, return the first argument

Data Types of SQL SERVER:

* Bit 🡪 store bit data i.e., 0,1, or NULL in the database with the BIT data type
* Int 🡪integer types in SQL Server including BIGINT, INT, SMALLINT, and TINYINT
* Decimal 🡪how to store exact numeric values in the database by using DECIMAL or NUMERIC data type
* Char 🡪how to store fixed-length, non-Unicode character string in the database
* NChar 🡪 how to store fixed-length, Unicode character strings in the database
* Varchar 🡪 store variable-length, non-Unicode string in the database
* Nvarchar 🡪 store variable-length, Unicode string data in the table
* Datatime2 🡪illustrate to store both date and time data in a database
* Date 🡪the date data type store dates in the tables
* Time 🡪to store time data in the database
* Datetimeoffset 🡪manipulates the datetime with the time zone

Types of SQL Statements:

SQL commands are instructions

It is used to communicate with the database

It is also used to perform various specific tasks, functions, and queries of data

1. DDL
2. DML
3. DCL
4. TCL
5. DQL

**Data Definition Language (DDL):**

DDL changes the structure of the table like creating a table, deleting a table, altering a table etc….

All the commands of DDL are auto-committed.

There are some commands under DDL:

1. Create

It is used to create a new table in the database

1. Alter

It is used to alter the structure of the database

1. Drop

It is used to delete both the structure and record stored in the table

1. Truncate

It is used to delete all the rows from the table and free the space containing the table

**Data Manipulation Language (DML):**

DML commands are used to modify the database

It is responsible for all form of changes in the database

It is not auto-committed.

There are some commands under DML:

1. Insert

It is used to insert data into the row of the table

1. Update

It is used to update or modify the value of a column in the table

1. Delete

It is used to remove one or more row from a table

**Data Control Language (DCL):**

DCL commands are used to grant and take back authority from any database user

These are some commands under DCL:

1. Grant

It is used to give user access privileges to a database

1. Revoke

It is used to take back permissions from the user

**Transaction Control Language (TCL):**

TCL commands can only use with DML commands

It automatically committed in the database

There are some commands under TCL:

1. Commit

It is used to save all the transactions to the database

1. Rollback

It is used to undo transactions that have not already been saved to the database

1. Savepoint

It is used to roll the transaction back to a certain point without rolling back the entire transaction

**Data Query Language (DQL):**

DQL is used to fetch the data from the database

1. Select

This is the same as the projection operation of relational algebra

It is used to select the attribute based on the condition described by WHERE clause.

SQL Aggregate Functions:

SQL aggregation function is used to perform the calculations on multiple rows of a single column of a table

It returns a single value

It is also used to summarize the data

Types of SQL Aggregation Function:

1. Count function

* It is used to count the number of rows in a database table
* It can work both numeric and non-numeric data types
* Count considers duplicate and NULL
* COUNT(\*)

1. Sum function

* It is used to calculate the sum of all selected columns
* It works on numeric fields only
* SUM()

1. Avg function

* It is used to calculate the average value of the numeric type
* AVG function returns the average of all non-Null values
* AVG()

1. Max function

* It is used to find the maximum value of certain column
* It determines the largest value of all selected values of a column
* MAX()

1. Min function

* It is used to find the minimum value of certain column
* It determines the smallest value of all selected values of a column
* MIN()

Different Constraints:

1. Super key

Super key is a set of one or more than one keys that can be used to identify a record uniquely in a table.

1. Candidate key

Candidate key is a set of one or more columns that can identify a record uniquely in a table.

There can be multiple candidate keys in one table.

1. Alternate key

Alternate key is a key that can work as a primary key.

Basically, it is a candidate key that currently is not a primary key.

1. Composite/compound key

It is combination of more than one columns of a table.

It can be candidate key, primary key.

1. Unique key

It is a set of one or more columns of a table that uniquely identify a record in a database table.

It is like primary key but it can accept only one null value and it cannot have duplicate values.

1. Surrogate key

An artificial key which aims to uniquely identify each record.

1. Primary key

This identifies each row in the table.

One table can only contain one primary key.

The key must be unique and without null values.

1. Foreign key

This is used to link two tables.

The foreign key is kept in one table and refers to the primary key associated with another table.

1. Not null key

To ensure a column not to accept null

1. Check key

The process of adding logic for checking data before storing them in tables.

JOINS

* Joins 🡪 a join clause is used to combine rows from two or more tables, based on a related column between them.
* Inner join 🡪select rows from a table that have matching rows in another table
* Left join 🡪return all rows from the left table and matching rows from the right table.

The result is null from the right side, if there is no match

* Right join 🡪 return all rows from the right table and matching rows from the left table.

The result is null from the left side, if there is no match

* Full outer join 🡪return matching rows from both left and right tables, and rows from each side if no matching rows exist
* Self join 🡪 a self join is a regular join, but the table is joined with itself
* Cross join 🡪join multiple unrelated tables and create Cartesian product of rows in the joined tables.