DATA BASE MANAGEMENT SYSTEM

UNIT-4

SQL:

SQL is a short-form of the structured query language, and it is pronounced as S-Q-L or sometimes as See-Quell.

This database language is mainly designed for maintaining the data in relational database management systems. It is a special tool used by data professionals for handling structured data (data which is stored in the form of tables). It is also designed for stream processing in RDSMS.

You can easily create and manipulate the database, access and modify the table rows and columns, etc. This query language became the standard of ANSI in the year of 1986 and ISO in the year of 1987.

Uses of SQL:

- The basic use of SQL for data professionals and SQL users is to insert, update, and delete the data from the relational database.
- SQL allows the data professionals and users to retrieve the data from the relational database management systems.
- It also helps them to describe the structured data.
- It allows SQL users to create, drop, and manipulate the database and its tables.
- It also helps in creating the view, stored procedure, and functions in the relational database.
- It allows you to define the data and modify that stored data in the relational database.
- It also allows SQL users to set the permissions or constraints on table columns, views, and stored procedures.

Advantages/Characteristics of SQL:

1. No programming needed

SQL does not require a large number of coding lines for managing the database systems. We can easily access and maintain the database by using simple SQL syntactical rules. These simple rules make the SQL user-friendly.

2. High-Speed Query Processing

A large amount of data is accessed quickly and efficiently from the database by using SQL queries. Insertion, deletion, and updation operations on data are also performed in less time.

3. Standardized Language

SQL follows the long-established standards of ISO and ANSI, which offer a uniform platform across the globe to all its users.

4. Portability

The structured query language can be easily used in desktop computers, laptops, tablets, and even smartphones. It can also be used with other applications according to the user's requirements.

5. Interactive language

We can easily learn and understand the SQL language. We can also use this language for communicating with the database because it is a simple query language. This language is also used for receiving the answers to complex queries in a few seconds.

6. More than one Data View

The SQL language also helps in making the multiple views of the database structure for the different database users.

Disadvantages of SQL:

1. Cost

The operation cost of some SQL versions is high. That's why some programmers cannot use the Structured Query Language.

2. Interface is Complex

Another big disadvantage is that the interface of Structured query language is difficult, which makes it difficult for SQL users to use and manage it.

3. Partial Database control

The business rules are hidden. So, the data professionals and users who are using this query language cannot have full database control.

SQL Constraints:

Constraints are the rules that we can apply on the type of data in a table, to maintain the accuracy and integrity of the data inside table.

Constraints can be divided into following two types:

Column level constraints: limits only column data

Table level constraints: limits whole table data

The following constraints are commonly used in SQL:

NOT NULL - Ensures that a column cannot have a NULL value

UNIQUE - Ensures that all values in a column are different

PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table

FOREIGN KEY - Uniquely identifies a row/record in another table

CHECK - Ensures that all values in a column satisfies a specific condition

DEFAULT - Sets a default value for a column when no value is specified

INDEX - Used to create and retrieve data from the database very quickly

SQL Data Types

A data type defines what kind of value a column can hold.

These are the different data types into the following categories:

Numeric: This type of data stores numerical values. Data types that fall in this category include Integer, Float, Real, Numeric, or Decimal.

Character String: This type of data stores character values. The two common types are CHAR(n) and VARCHAR(n).

Date/Datetime: This type of data allows us to store data or datetime in a database table.

Binary: This type of data allows us to store binary objects in a database table.

Basic Types

char(n): A fixed-length character string with user-specified length.

varchar(n): A variable-length character string with user-specified maximum length n.

int: An integer (a finite subset of the integers of the integers that is machine).

smallint: A small integer (a machine-dependent subset of the integer type).

numeric(p,d): A fixed-point number with user-specified precision.

real, double precision: Floating-point and double-position floating-point numbers with machine-dependent precision.

float(n): A floating-point number, with precision of at least n digits.

Date and Time Types in SQL

date: A calendar date containing a (four-digit) year, month, and day of the month.

time: The time of day, in hours, minutes, and seconds.

timestamp: A combination of date and time.

Literals

In SQL, a literal is the same as a constant. A literal is an explicit numeric, character, string, or Boolean value not represented by an identifier.

Some basic literals are given below:

Integer Literals: Integer literals can be either positive numbers or negative numbers, but do not contain decimals. Example: 536, +536, -536, etc.

Decimal Literals: Decimal literals can be either positive numbers or negative numbers and contain decimals. Example: 24.7, +24.7, -24.7, etc.

Character Literals: Character literals are always surrounded by single quotes (') and contain a single character. Example: ‘A’, ‘%’, ‘9’, ‘)’, etc.

String Literals: String literals are also surrounded by single quotes but it contain group of characters. Example: 'Hello world!', 'XYZ', '123', etc.

BOOLEAN Literals: Boolean literals hold values ‘TRUE’, ‘FALSE’, or ‘NULL’.

Date and Time Literals: Datetime literals are character representations of datetime values that are enclosed in single quotes. Example: '2015/04/30', '2015/04/30 08:34:25'

SQL Operators:

Operators are the foundation of any programming language. We can define operators as symbols that help us to perform specific mathematical and logical computations on operands. In other words, we can say that an operator operates the operands. SQL operators have three different categories.

1. Arithmetic operator
2. Comparison operator
3. Logical operator

1. Arithmetic operators:

We can use various arithmetic operators on the data stored in the tables. Arithmetic Operators are:

Operator	Description
+	The addition is used to perform an addition operation on the data values.
–	This operator is used for the subtraction of the data values.
/	This operator works with the 'ALL' keyword and it calculates division operations.
*	This operator is used for multiply data values.
%	Modulus is used to get the remainder when data is divided by another.

2. Comparison operators:

Another important operator in SQL is a comparison operator, which is used to compare one expression's value to other expressions. SQL supports different types of the comparison operator, which is described below:

Operator	Description
=	Equal to.
>	Greater than.
<	Less than.
>=	Greater than equal to.
<=	Less than equal to.
<>	Not equal to.

3. Logical operators:

The Logical operators are those that are true or false. They return true or false values to combine one or more true or false values.

Operator	Description
AND	Logical AND compares between two Booleans as expressions and returns true when both expressions are true.
OR	Logical OR compares between two Booleans as expressions and returns true when one of the expressions is true.
NOT	Not takes a single Boolean as an argument and changes its value from false to true or from true to false.

Special operators:

Operator	Description
ALL	ALL is used to select all records of a SELECT STATEMENT. It compares a value to every value in a list of results from a query. The ALL must be preceded by the comparison operators and evaluates to TRUE if the query returns no rows.

ANY	ANY compares a value to each value in a list of results from a query and evaluates to true if the result of an inner query contains at least one row.
BETWEEN	The SQL BETWEEN operator tests an expression against a range. The range consists of a beginning, followed by an AND keyword and an end expression.
IN	The IN operator checks a value within a set of values separated by commas and retrieves the rows from the table which are matching.
EXISTS	The EXISTS checks the existence of a result of a subquery. The EXISTS subquery tests whether a subquery fetches at least one row. When no data is returned then this operator returns 'FALSE'.
SOME	SOME operator evaluates the condition between the outer and inner tables and evaluates to true if the final result returns any one row. If not, then it evaluates to false.
UNIQUE	UNIQUE operator searches every unique row of a specified table.

SQL Commands:

SQL commands are instructions. It is used to communicate with the database. It is also used to perform specific tasks, functions, and queries of data.

SQL can perform various tasks like create a table, add data to tables, drop the table, modify the table, set permission for users.

Types of SQL Commands

There are five types of SQL commands: DDL, DML, DCL & TCL

DDL (Data Definition Language)

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

All the command of DDL is auto-committed that means it permanently save all the changes in the database.

Here are some commands that come under DDL:

1. CREATE: It is used to create a new table in the database.

Syntax:

CREATE TABLE TABLE_NAME (COLUMN_NAME DATATYPES[,....]);

2. DROP: It is used to delete both the structure and record stored in the table.

Syntax

DROP TABLE table_name;

3. ALTER: It is used to alter the structure of the database. This change could be either to modify the characteristics of an existing attribute or probably to add a new attribute.

Syntax:

To add a new column in the table

ALTER TABLE table_name ADD column_name COLUMN-definition;

To modify existing column in the table:

ALTER TABLE table_name MODIFY(column_definitions....);

4. TRUNCATE: It is used to delete all the rows from the table and free the space containing the table.

Syntax:

TRUNCATE TABLE table_name;

5. 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.

Syntax:

SELECT expressions

FROM TABLES

WHERE conditions;

DML (Data Manipulation Language)

DML commands are used to modify the database. It is responsible for all form of changes in the database.

The command of DML is not auto-committed that means it can't permanently save all the changes in the database. They can be rollback.

Here are some commands that come under DML:

1. INSERT: The INSERT statement is a SQL query. It is used to insert data into the row of a table.

Syntax:

INSERT INTO TABLE_NAME

VALUES (value1, value2, value3, ... valueN);

2. UPDATE: This command is used to update or modify the value of a column in the table.

Syntax:

UPDATE table_name SET [column_name1= value1,...column_nameN = valueN] [WHERE CONDITION]

3. DELETE: It is used to remove one or more row from a table.

Syntax:

DELETE FROM table_name [WHERE condition];

DCL (Data Control Language)

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

Here are some commands that come under DCL:

1. Grant: It is used to give user access privileges to a database.

Example

GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER, ANOTHER_USER;

2. Revoke: It is used to take back permissions from the user.

Example

REVOKE SELECT, UPDATE ON MY_TABLE FROM USER1, USER2;

TCL (Transaction Control Language)

TCL commands can only use with DML commands like INSERT, DELETE and UPDATE only.

These operations are automatically committed in the database that's why they cannot be used while creating tables or dropping them.

Here are some commands that come under TCL:

1. Commit: Commit command is used to save all the transactions to the database.

Syntax:

COMMIT;

2. Rollback: Rollback command is used to undo transactions that have not already been saved to the database.

Syntax:

ROLLBACK;

SQL Index:

An index is a schema object. It is used by the server to speed up the retrieval of rows by using a pointer. It can reduce disk I/O(input/output) by using a rapid path access method to locate data quickly. An index helps to speed up select queries and where clauses, but it slows down data input, with the update and the insert statements. Indexes can be created or dropped with no effect on the data. In this article, we will see how to create, delete, and uses the INDEX in the database.

For example, if you want to reference all pages in a book that discusses a certain topic, you first refer to the index, which lists all the topics alphabetically and is then referred to one or more specific page numbers.

Creating an Index:

Syntax:

```
CREATE INDEX index
 ON TABLE column;
```

For multiple columns:

Syntax:

```
CREATE INDEX index
 ON TABLE (column1, column2,.....);
```


Unique Indexes:

Unique indexes are used for the maintenance of the integrity of the data present in the table as well as for the fast performance, it does not allow multiple values to enter into the table.

Syntax:

```
CREATE UNIQUE INDEX index
 ON TABLE column;
```

When should indexes be created:

- A column contains a wide range of values.
- A column does not contain a large number of null values.
- One or more columns are frequently used together in a where clause or a join condition.

When should indexes be avoided:

- The table is small

- The columns are not often used as a condition in the query
- The column is updated frequently

Removing an Index:

To remove an index from the data dictionary by using the DROP INDEX command.

Syntax:

DROP INDEX index;

To drop an index, you must be the owner of the index or have the DROP ANY INDEX privilege.

Altering an Index:

To modify an existing table's index by rebuilding, or reorganizing the index.

Syntax

ALTER INDEX IndexName
ON TableName REBUILD;

Confirming Indexes:

You can check the different indexes present in a particular table given by the user or the server itself and their uniqueness.

Syntax:

select * from USER_INDEXES;

It will show you all the indexes present in the server, in which you can locate your own tables too.

Renaming an index:

You can use the system stored procedure sp_rename to rename any index in the database.

Syntax:

```
EXEC sp_rename
  index_name,
  new_index_name,
  N'INDEX';
```

Views in SQL:

A view is a database object that has no values. **It is a virtual table, which is created according to the result set of an SQL query**. However, it looks similar to an actual table containing rows

and columns. Therefore, we can say that its contents are based on the base table. It is operated similarly to the base table but does not contain any data of its own. **Its name is always unique, like tables**. The views differ from tables as they are definitions that are created on top of other tables (or views).

Uses of views:

The primary use of view in SQL Server is to implement the security mechanism. It prevents users from seeing specific columns and rows from tables. It only shows the data returned by the query that was declared when the view was created. The rest of the information is completely hidden from the end-user.

Types of views:

The SQL Server categories the views into two types:

1. User-Defined Views

Users define these views to meet their specific requirements. It can also divide into two types one is the simple view, and another is the complex view. The simple view is based on the single base table without using any complex queries. The complex view is based on more than one table along with group by clause, order by clause, and join conditions.

2. System-Defined Views

System-defined views are predefined and existing views stored in SQL Server, such as Tempdb, Master, and temp. Each system views have its own properties and functions. They can automatically attach to the user-defined databases. We can divide the System-defined views in SQL Server into three types: Information Schema, Catalog View, and Dynamic Management View.

SQL Table:

Table is a collection of data, organized in terms of rows and columns. In DBMS term, table is known as relation and row as tuple.

SQL TABLE Variable:

- The SQL Table variable is used to create, modify, rename, copy and delete tables. Table variable was introduced by Microsoft.
- It was introduced with SQL server 2000 to be an alternative of temporary tables.
- It is a variable where we temporary store records and results. This is same like temp table but in the case of temp table we need to explicitly drop it.

SQL CREATE TABLE:

SQL CREATE TABLE statement is used to create table in a database.

Syntax:

create table "tablename"

("column1" "data type",

"column2" "data type",

"column3" "data type",

"columnN" "data type");

SQL DROP TABLE:

A SQL DROP TABLE statement is used to delete a table definition and all data from a table.

Syntax:

DROP TABLE "table_name";

SQL DELETE TABLE:

The DELETE statement is used to delete rows from a table. If you want to remove a specific row from a table you should use WHERE condition.

Syntax:

DELETE FROM table_name [WHERE condition];

But if you do not specify the WHERE condition it will remove all the rows from the table.

DELETE FROM table_name;

SQL RENAME TABLE:

In some situations, database administrators and users want to change the name of the table in the SQL database because they want to give a more relevant name to the table.

Syntax

RENAME old_table _name To new_table_name ;

SQL TRUNCATE TABLE:

A truncate SQL statement is used to remove all rows (complete data) from a table. It is similar to the DELETE statement with no WHERE clause.

Syntax:

TRUNCATE TABLE table_name;

SQL ALTER TABLE:

The ALTER TABLE statement in Structured Query Language allows you to add, modify, and delete columns of an existing table. This statement also allows database users to add and remove various SQL constraints on the existing tables.

Syntax:

ALTER TABLE table_name ADD column_name column-definition;

SQL Aggregate Functions:

Aggregate functions in SQL Server are used to perform calculations on one or more values and return the result in a single value. In SQL Server, all aggregate functions are built-in functions that avoid NULL values except for COUNT (*). We mainly use these functions with the GROUP BY and HAVING clauses of the SELECT statements in the database query languages.

DISTINCT:

The DISTINCT modifier is used when we want to consider the distinct values in the calculation. The ALL modifiers are used when we want to calculate all values, including duplicates. If we do specify any modifier, all aggregate functions use the ALL modifier by default.

Syntax:

aggregate_function_name(DISTINCT | ALL exp)

Aggregate Function	Descriptions
COUNT()	This function counts the number of elements or rows, including NULL values in the defined set.
SUM()	This function calculates the total sum of all NOT-NULL values in the given set.
AVG()	This function performs a calculation on NOT-NULL values to get the average of them in a defined set.
MIN()	This function returns the minimum (lowest) value in a set.
MAX()	This function returns the maximum (highest) value in a set.

SQL SELECT Query:

Select is the most commonly used statement in SQL. The SELECT Statement in SQL is used to retrieve or fetch data from a database. We can fetch either the entire table or according to some specified rules. The data returned is stored in a result table. This result table is also called result-set.

With the SELECT clause of a SELECT command statement, we specify the columns that we want to be displayed in the query result and, optionally, which column headings we prefer to see above the result table.

The select clause is the first clause and is one of the last clauses of the select statement that the database server evaluates. The reason for this is that before we can determine what to include in the final result set, we need to know all of the possible columns that could be included in the final result set.

Sample Table:

Student

ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18

Basic Syntax:

`SELECT column1,column2 FROM table_name`
`column1 , column2`: names of the fields of the table
`table_name`: from where we want to fetch

This query will return all the rows in the table with fields column1 , column2.

To fetch the entire table or all the fields in the table: `SELECT * FROM table_name;`

Query to fetch the fields ROLL_NO, NAME, AGE from the table Student:`SELECT ROLL_NO, NAME, AGE FROM Student;`

Output:

ROLL_NO	NAME	Age
1	Ram	18
2	RAMESH	18
3	SUJIT	20
4	SURESH	18

To fetch all the fields from the table Student: `SELECT * FROM Student;`

Output:

ROLL_NO	NAME	ADDRESS	PHONE	Age
1	Ram	Delhi	XXXXXXXXXX	18
2	RAMESH	GURGAON	XXXXXXXXXX	18
3	SUJIT	ROHTAK	XXXXXXXXXX	20
4	SURESH	Delhi	XXXXXXXXXX	18

SQL subqueries:

In SQL a Subquery can be simply defined as a query within another query. In other words, we can say that a Subquery is a query that is embedded in WHERE clause of another SQL query. Important rules for Subqueries:

- You can place the Subquery in a number of SQL clauses: WHERE clause, HAVING clause, FROM clause. Subqueries can be used with SELECT, UPDATE, INSERT, DELETE statements along with expression operator. It could be equality operator or comparison operator such as =, >, =, <= and Like operator.
- A subquery is a query within another query. The outer query is called as **main query** and inner query is called as **subquery**.
- The subquery generally executes first when the subquery doesn't have any **co-relation** with the **main query**, when there is a co-relation the parser takes the decision **on the fly** on which query to execute on **precedence** and uses the output of the subquery accordingly.
- Subquery must be enclosed in parentheses.
- Subqueries are on the right side of the comparison operator.
- ORDER BY command **cannot** be used in a Subquery. GROUPBY command can be used to perform same function as ORDER BY command.
- Use single-row operators with singlerow Subqueries. Use multiple-row operators with multiple-row Subqueries.

Syntax: There is not any general syntax for Subqueries. However, Subqueries are seen to be used most frequently with SELECT statement as shown below:

```
SELECT column_name
FROM table_name
WHERE column_name expression operator
    (SELECT COLUMN_NAME from TABLE_NAME   WHERE ...);
```

Basic Commands

1) INSERT command: Insert command is used to insert data into a table.

Syntax: INSERT into table-name values (data1, data2,..)

2) UPDATE command: Update command is used to update a row of a table.

Syntax: UPDATE table-name set column-name = value where condition;

3) Delete command: Delete command is used to delete data from a table. Delete command can also be used with condition to delete a particular row.

Syntax: DELETE from table-name;

SQL Joins:

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

General Syntax:

SELECT column-names

FROM table-name1 JOIN table-name2

ON column-name1 = column-name2

WHERE condition

Types of Join

Here are the different types of the JOINs in SQL:

(INNER) JOIN: It returns records that have matching values in both tables.

Syntax:

SELECT column-name-list

from table-name1

INNER JOIN

table-name2

WHERE table-name1.column-name = table-name2.column-name;

LEFT (OUTER) JOIN: It returns all records from the left table, and the matched records from the right table.

Syntax:

SELECT column-name-list

from table-name1

LEFT OUTER JOIN

table-name2

on table-name1.column-name = table-name2.column-name;

RIGHT (OUTER) JOIN: It returns all records from the right table, and the matched records from the left table.

Syntax:

select column-name-list

from table-name1

RIGHT OUTER JOIN

table-name2

on table-name1.column-name = table-name2.column-name;

FULL (OUTER) JOIN: It returns all records when there is a match in either left or right table.

Syntax:

select column-name-list

from table-name1

FULL OUTER JOIN

table-name2

on table-name1.column-name = table-name2.column-name;

Union

The SQL UNION clause/operator is used to combine the results of two or more SELECT statements without returning any duplicate rows.

Syntax:

SELECT column_name(s) FROM table1

UNION

SELECT column_name(s) FROM table2;

Union ALL

The UNION ALL operator is used to combine the results of two SELECT statements including duplicate rows.

Syntax:

SELECT column_name(s) FROM table1

UNION ALL

SELECT column_name(s) FROM table2;

difference between UNION and UNION ALL:

- UNION removes duplicate rows.
- UNION ALL does not remove duplicate rows.

Intersect

Intersect operation is used to combine two SELECT statements, but it only retuns the records which are common from both SELECT statements.

Syntax:

SELECT column_name(s) FROM table1

INTERSECT SELECT

column_name(s) FROM table2;

Minus

It takes all the results from the first SQL statement, and then subtract out the ones that are present in the second SQL statement to get the final result set.

Syntax:

[SQL Statement 1]

MINUS

[SQL Statement 2];