**TASK 3 STRUCTURED QUERY LANGUAGE 21-03-23**

**WHAT IS SQL?**

SQL (pronounced “ess-que-el”) stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc.

Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as “Select”, “Insert”, “Update”, “Delete”, “Create”, and “Drop” can be used to accomplish almost everything that one needs to do with a database.

**WHAT CAN SQL DO?**

* SQL can execute queries against a database
* SQL can retrieve data from a database
* SQL can insert records in a database
* SQL can update records in a database
* SQL can delete records from a database
* SQL can create new databases
* SQL can create new tables in a database
* SQL can create stored procedures in a database
* SQL can create views in a database
* SQL can set permissions on tables, procedures, and views

**WHY IS SQL IMPORTANT?**

Structured query language (SQL) is a popular query language that is frequently used in all types of applications. Data analysts and developers learn and use SQL because it integrates well with different programming languages. For example, they can embed SQL queries with the Java programming language to build high-performing data processing applications with major SQL database systems such as Oracle or MS SQL Server. SQL is also fairly easy to learn as it uses common English keywords in its statements.

**HISTORY OF SQL?**

SQL was invented in the 1970s based on the relational data model. It was initially known as the structured English query language (SEQUEL). The term was later shortened to SQL. Oracle, formerly known as Relational Software, became the first vendor to offer a commercial SQL relational database management system.

**WHAT ARE SQL COMMANDS?**

Structured query language (SQL) commands are specific keywords or SQL statements that developers use to manipulate the data stored in a relational database. You can categorize SQL commands as follows.

**DATA DEFINITION LANGUAGE (DDL)**

Data definition language (DDL) refers to SQL commands that design the database structure. Database engineers use DDL to create and modify database objects based on the business requirements. For example, the database engineer uses the CREATE command to create database objects such as tables, views, and indexes.

**DATA QUERY LANGUAGE (DQL)**

Data query language (DQL) consists of instructions for retrieving data stored in relational databases. Software applications use the SELECT command to filter and return specific results from a SQL table.

**DATA MANIPULATION LANGUAGE (DML)**

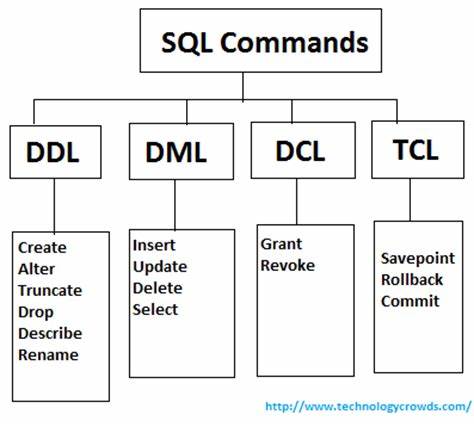
Data manipulation language (DML) statements write new information or modify existing records in a relational database. For example, an application uses the INSERT command to store a new record in the database.

**DATA CONTROL LANGUAGE (DCL)**

Database administrators use data control language (DCL) to manage or authorize database access for other users. For example, they can use the GRANT command to permit certain applications to manipulate one or more tables.

**TRANSACTION CONTROL LANGUAGE (TCL)**

The relational engine uses transaction control language (TCL) to automatically make database changes. For example, the database uses the ROLLBACK command to undo an erroneous transaction.



**EXAMPLES OF SQL COMMANDS**

**DDL COMMANDS**

* DDL in DBMS is a language that allows user to define the database components and their relationship with each other.
* These commands work with the structure of tables like creating a table, deleting a table, and altering a table.
* DDL commands allow us to define and change the data types of table columns, and integrity constraints while defining the table structure. Integrity constraints are rules set for the data being stored in the table. This is explained in the next section of this article.
* All DDL commands are auto-committed which means the changes done using them are permanently saved in the Database but they can be rolled back.

The DDL commands in DBMS are as follows :

1. CREATE
2. ALTER
3. TRUNCATE
4. DROP

**CREATE**

CREATE is a DDL command in DBMS that is used to create tables or databases. While creating a table, we specify table names, column names, datatypes of each column, and column sizes. If there are any integrity constraints or key constraints of the table like PRIMARY KEY, UNIQUE, NOT NULL, etc. we must specify them while creating it along with the column names and details.

Notes :

1. Some of the essential datatypes in SQL are as follows :

* **int/number :**

Stores integer numbers

* **varchar :**

Stores string

* **date :**

Stores date

2. Integrity constraints are the rules that a table's data columns must follow.

Some of the integrity constraints in SQL are as follows:

**PRIMARY KEY:**

Primary key is a column that is unique in the table and is used to uniquely identify each record in the table.

**FOREIGN KEY:**

Foreign key is a column used to define a relationship between two tables by referencing to Primary key of another table.

**NOT NULL:**

Not null integrity constraint is used to say that this column cannot be null. It has to have some data.

3. SQL commands are not case-sensitive.

**SYNTAX:**

CREATE TABLE table\_name

(

column1 datatype,

column2 datatype,

column3 datatype,

.....

)

**EXAMPLE:**

Let's create a table that store data of school students using CREATE DDL command.

*--Defining a new table using CREATE*

CREATE TABLE Students

(

*--Giving the column names and data type details of the table.*

Roll\_no INT PRIMARY KEY,

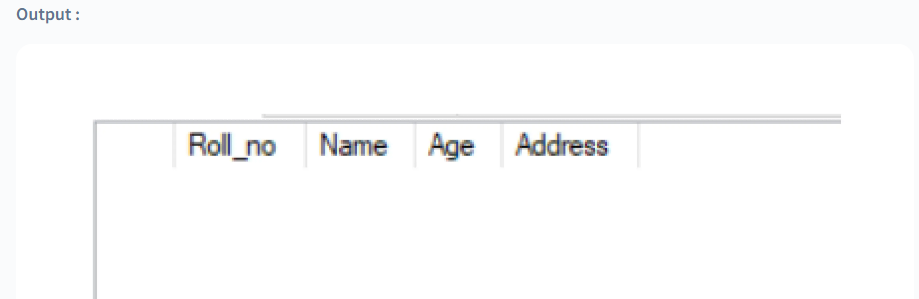
Name VARCHAR(20),

Age INT,

Address VARCHAR(30)

);

**OUTPUT:**



We have created a new table called, Students, using the DDL command CREATE. There are 4 columns in the table, which are specified in the command with their corresponding data types and sizes. The PRIMARY KEY of this table is Roll\_no, and it has been specified when defining the table with the PRIMARY KEY integrity constraint. In the output section, a blank table Students is displayed that was created using the CREATE command with the specified columns. The table is empty as we have not added any data to it yet.

**ALTER**

ALTER is a DDL command used to change the structure of an existing database table. Using this command we can perform operations like adding a new column, removing any column, adding or removing integrity constraints, or changing the data type of the existing column on the existing table. ALTER command is used to modify the existing structure of the table.

**SYNTAX:**

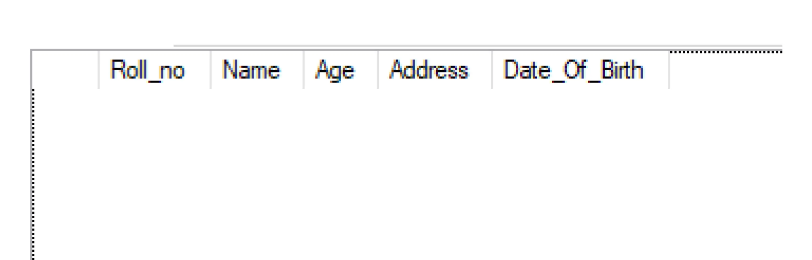
ALTER TABLE table\_name ADD column\_name column-definition;

**EXAMPLE - 1 :**

Let's add a new column in the **Students** table using the ALTER DDL command.

ALTER TABLE Students ADD Date\_Of\_Birth DATE;

**OUTPUT**

****

In this section, we have modified the existing table Students using the ALTER command in DBMS. We have added a new column named Date\_of\_Birth with data type DATE using the ALTER DDL command in DBMS. In the output section, we can see the column has been added to the existing table.

**EXAMPLE - 2 :**

Let's see one more example illustrating how to modify an existing column in the table**.**

ALTER TABLE Students MODIFY Age VARCHAR;

ALTER TABLE Students MODIFY Age VARCHAR;

In this command ALTER statement is used to change the data type of the existing column Age in the Students table from int to varchar.

**TRUNCATE**

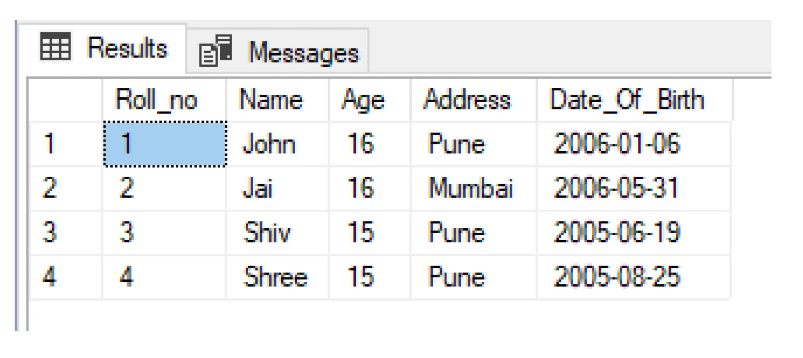
TRUNCATE is a DDL command used to delete all records from the table while keeping its structure intact. This command is used to get rid of all the records and free up the space being used by the table. However, the table's structure remains unchanged, including the column names and details. Thus, the table can be used to store new data in the future.

**SYNTAX:**

TRUNCATE TABLE table\_name;

**EXAMPLE:**

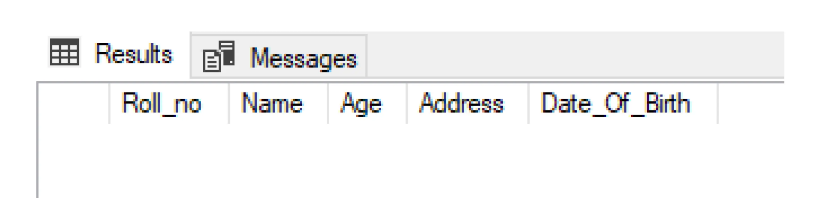
As shown in the image below, the Students table stores records of some students currently.



Consider, that the school has started a new batch of students and wants to store their details in the same structure. They can use the same table by removing the records from the existing table and keeping its structure as it is. Let's achieve the same using the TRUNCATE DDL commands in DBMS.

TRUNCATE TABLE Students;

**OUTPUT:**



As we can see in the output, after using the TRUNCATE command all the records from the Students table are deleted and the table structure is as it is. The same empty Students can be used again.

**DROP**

DROP is a DDL command used to delete an existing table completely from the database. This command deletes the table records along with their structure. The table and all its components are completely deleted and cannot be found again in the database.

**SYNTAX:**

DROP TABLE table\_name;

**EXAMPLE:**

Let's delete the Students table completely from the database.

DROP TABLE Students;

The above DROP command deletes the Students table from the database.

**CONCLUSION:**

Data Definition Language is a subset of SQL used to define the structure of the database.

All DDL commands are auto-committed i.e. all the changes are saved permanently in the database.

The structure of the database is defined using these commands hence they are known as Data Definition Language in DBMS.

CREATE, ALTER, TRUNCATE, and DROP are some of the common DDL commands in DBMS.

**EXAMPLES OF SQL COMMANDS**

**DML COMMANDS**

Let's see various dml commands in dbms, along with their examples:

**SELECT:**

SELECT is the most widely used data manipulation command in Structured Query Language for retrieving data from the table.

It is used to fetch the data in a set of records from a table, view or a group of tables, views by making the use of SQL joins.

The returned data from the SELECT command is stored in a new table called result set.

**SYNTAX OF SELECT COMMAND:**

SELECT \* FROM table\_name;

The SELECT query is used when we want to retrieve the data from all the columns of the table.

SELECT column\_Name\_1, column\_Name\_2, … FROM table\_name;

Here, columnName1, columnName2 are the names of those columns whose data we want to show.

This syntax for the SELECT command is used when we want to retrieve the data from some specific columns of the table.

**EXAMPLES:**

Let's say we have a table student\_details, having ROLL\_NO, FIRST\_NAME, LAST\_NAME and MARKS as the fields, and some data has been stored in these fields.

Now, if we want to access all the data stored in the table, we will make use of the SELECT command.

SELECT \* FROM student\_details

NOTE: Here, \* represents all the columns, which means we are selecting or retrieving the data from all the columns of a given table.

**OUTPUT:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARKS** |
| --- | --- | --- | --- |
| 1 | Rita | Goyal | 67 |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |

Now, if we want to only show the ROLL\_NO and the marks of a student. Then we can write:

SELECT ROLL\_NO, MARKS FROM student\_details

**OUTPUT:**

| **ROLL\_NO** | **MARKS** |
| --- | --- |
| 1 | 67 |
| 2 | 91 |
| 3 | 100 |

Here, we have specifically mentioned the column names separated by comma, for which we want to access the data.

### INSERT

INSERT is another important data manipulation command in Structured Query Language used to insert data into the rows of a table.

**Syntax of INSERT command:**

INSERT INTO table\_name

(col\_1, col\_2,.... col\_n)

VALUES (value\_1, value\_2, .... value\_n);

Another way is to only specify the value of data to be inserted without the column names.

INSERT INTO table\_name

VALUES (value\_1, value\_2, .... value\_n);

Here, col\_1, col\_2 .. col\_n represents the name of the columns and value\_1, value\_2, ... value\_n represents the value to be inserted in the corresponding columns.

**Note:** Using the syntax of col1 col2 ... along with value\_1, value\_2, we can skip any column data and the default value to that column will be assigned, but in case of the syntax in which only values are assigned without the column names, we need to provide the data in a sequential manner.

**Examples:**

Taking our previous example of the student\_details table, let's say we want to add the 4th entry of a student. We can write a query:

INSERT INTO student\_details (ROLL\_NO, FIRST\_NAME, LAST\_NAME, MARKS) VALUES (4, 'Arpit', 'Garg', 82);

**OUTPUT:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARKS** |
| --- | --- | --- | --- |
| 1 | Rita | Goyal | 67 |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |
| 4 | Arpit | Garg | 82 |

As we can see from the output, the 4th row for the newly added student has been created.

Let's say we want to add the 5th record in our table, using the second syntax in which we only specify the value of data to be inserted without the column names.

We can write a query;

INSERT INTO student\_details

VALUES (5, 'Kevin', 'Mehta', 23);

**OUTPUT:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARKS** |
| --- | --- | --- | --- |
| 1 | Rita | Goyal | 67 |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |
| 4 | Arpit | Garg 82 |  |
| 5 | Kevin | Mehta | 23 |

As we can see from the output, the 5th row for the newly added student has been created.

### UPDATE:

UPDATE command is another important data manipulation command in Structured Query Language, it allows us to update or modify the existing data in tables. **It changes the data from one or more records in a table.**

**Syntax of UPDATE command:**

UPDATE table\_name

SET col\_1 = value\_1, col\_2 = value\_2,...

WHERE condition;

Here, the specified condition helps us to select the rows for which the value of columns needs to be updated.

**Examples:**

Again, coming to our previous example of the student\_details table, let's say, we want to change the marks of a student whose roll\_no is given to us.

We can write the query to update the marks, by specifying the roll\_no for which we are updating the marks.

UPDATE student\_details

SET MARKS = 92

WHERE ROLL\_NO = 4;

Now, the updated table looks like this.

**OUTPUT:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARKS** |
| --- | --- | --- | --- |
| 1 | Rita | Goyal | 67 |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |
| 4 | Arpit | Garg | 92 |

We can also update the value of multiple fields in a table by setting the values separated by commas.

UPDATE student\_details

SET MARKS = 92, LAST\_NAME = 'Gupta'

WHERE ROLL\_NO = 4

Now, the updated table looks like this.

**Note:** We have specified the condition using the WHERE clause, which states that the ROLL\_NO should be equal to 4 for the student whose data we want to update.

**OUTPUT:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARKS** |
| --- | --- | --- | --- |
| 1 | Rita | Goyal | 67 |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |
| 4 | Arpit | Gupta | 92 |

### DELETE:

**DELETE** is another useful data manipulation command in Structured Query Language, which allows us to remove single or multiple records from the database tables depending upon the condition we specify in the WHERE clause.

If **no condition** is specified in the **WHERE clause**, then it will **delete all the rows of a table**.

Delete command keeps the record for each deleted row and so it does not remove the stored data permanently from the database and we can always roll back to the previous stage using the ROLLBACK command as the log for that last deleted entry has already been stored in the DELETE command logs.

**Syntax of DELETE command:**

DELETE FROM table\_name;

As we can see here no condition has been specified so it will delete all the rows from the table.

OR

DELETE FROM table\_name WHERE condition;

Here, the condition is specified using the **WHERE** clause and only those records that fulfill the condition are deleted.

**Examples:**

Coming to our example of the student\_details table, let's say we want to delete all the records of the students whose marks are less than 80. Here, only one record for ROLL\_NO 1 will be deleted

**Original table:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARKS** |
| --- | --- | --- | --- |
| 1 | Rita | Goyal | 67 |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |
| 4 | Arpit | Garg | 92 |

DELETE FROM student\_details WHERE MARKS < 80;

**OUTPUT:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARKS** |
| --- | --- | --- | --- |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |
| 4 | Arpit | Garg | 92 |

Now, let's say we want to delete all the records from the table, so we can simply write.

DELETE FROM student\_details;

Its output will be an empty table with no data.

* As we discussed above the **DELETE** command in SQL maintains transaction logs for each deleted record and so it allows us to restore the deleted data by using the **ROLLBACK** command.

Let's discuss the example for the same.

We will first delete all the rows in the **student\_details table**.

DELETE FROM student\_details

Let's try to display the data from the table.

SELECT \* FROM student\_details;

**OUTPUT**: It's an empty table.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Now, let's use the Rollback command to restore the data.

ROLLBACK;

Now, let's try to retrieve the data from the table again.

SELECT \* FROM student\_details;

**OUTPUT:**

| **ROLL\_NO** | **FIRST\_NAME** | **LAST\_NAME** | **MARK** |
| --- | --- | --- | --- |
| 1 | Rita | Goyal | 67 |
| 2 | Kanika | Mittal | 91 |
| 3 | Reshma | Kumari | 100 |
| 4 | Arpit | Garg | 92 |

## **CONCLUSION**

* DML is an abbreviation of Data Manipulation Language.
* The DML in DBMS is responsible for performing **CRUD** (Create Read Update and Delete) operations on the data of the database table.
* Data Manipulation Language (DML) allows you to modify the database tables by inserting, modifying, and deleting their data.
* DML commands are not auto-commited and hence to store the data change permanently we need to commit our change.
* **SELECT** command is used for retrieving the data from the database.
* A **SELECT** statement retrieves zero or more rows from one or more database tables.
* **INSERT** command is used to insert data into a table.
* Using **INSERT** command, you may insert one row at a time, or select several rows from an existing table and insert them all at once.
* **UPDATE** command is used to update existing records within a table.
* If we want to update a particular record value, we use the **WHERE** clause along with the **UPDATE** command. If you do not use the **WHERE** clause, all the rows will be affected by the **UPDATE** command.
* The **DELETE** command is used to delete existing records from a database table. The **WHERE** clause is used with the **DELETE** command which specifies the record(s) that should be deleted. If you omit the **WHERE** clause, it will delete the whole table.