SQL Statements

1. DDL (Data Definition Language)

> CREATE

> ALTER

> DROP

> TRUNCATE

2. DML (Data Manipulation Language)

> INSERT

> UPDATE

> DELETE

3. TCL (Transaction Control Language)

> COMMIT

> SAVEPOINT

> ROLLBACK

4. DCL (Data Control Language)

> GRANT
> REVOKE

5. DQL (Data Query Language)

> SELECT
> PROJECTION
> SELECTION
> JOINS

To create a database in MySQL

```
CREATE DATABASE database_name;

CREATE DATABASE DEMO;
```

To access the database

```
USE database_name;

USE DEMO;
```

To display the databases available in MySQL

```
SHOW DATABASES;
```

To display the tables that are available in a Database

```
SHOW TABLES;
```

DDL (Data Definition Language)

CREATE:
This command is used to create the database objects such

as tables, relations, procedures, triggers, views etc...

Syntax:

```
CREATE TABLE table_name
(
 column_name_1 DATATYPE CONSTRAINT,
 column_name_2 DATATYPE CONSTRAINT,
 :
 column_name_n DATATYPE CONSTRAINT
);
```

```
CREATE TABLE ACCOUNTS
(
 ACCNO BIGINT PRIMARY KEY,
 ACC_HOLDER_NAME VARCHAR(20) NOT NULL,
 PHONE BIGINT UNIQUE NOT NULL CHECK(LENGTH(PHONE)=10),
 PINCODE INT
);
```

To display the table description

```
DESC table_name;

CREATE TABLE BRANCH
(
 BID INT PRIMARY KEY,
 B_NAME VARCHAR(20) UNIQUE,
 PINCODE INT
);

CREATE TABLE LOCATION
(
 PINCODE INT PRIMARY KEY,
 AREA VARCHAR(20) UNIQUE,
 CITY VARCHAR(20) NOT NULL,
 STATE VARCHAR(20) DEFAULT 'KARNATAKA'
);
```

ALTER

This command is used to modify the structure of the table.

1. To add a new column

```
ALTER TABLE table_name
ADD column_name DATATYPE CONSTRAINT;
```

To add a column after a certain column,

```
ALTER TABLE table_name
ADD column_name DATATYPE CONSTRAINT AFTER
existing_column_name;
```

>> Requirement: To add the mail_ID column inside ACCOUNTS Table.

```
ALTER TABLE ACCOUNTS
ADD MAIL_ID VARCHAR(30) UNIQUE;
```

>> Requirement 2: To add the IFSC_Code column in BRANCH table

```
ALTER TABLE BRANCH
ADD IFSC_Code VARCHAR(15) UNIQUE NOT NULL;
```

2. To remove a column

```
ALTER TABLE table_name
DROP column_name;
```

Requirement : To remove the MAIL_ID column from ACCCOUNTS table

───

```
ALTER TABLE ACCOUNTS
DROP MAIL_ID;
```

Requirement 2: To remove the B_NAME column from BRANCH table

───

```
ALTER TABLE BRANCH
DROP B_NAME;
```

3. To modify the datatype of a column

══════════════════════════════════════

```
ALTER TABLE table_name
MODIFY column_name new_datatype constraint;
```

Requirement : To modify Acc_holder_name column datatype to CHAR datatype

```
ALTER TABLE ACCOUNTS
MODIFY acc_holder_name CHAR(20) NOT NULL;
```

Requirement 2: To modify the IFSC_Code from VARCHAR to BIGINT

```
ALTER TABLE BRANCH
MODIFY IFSC_Code BIGINT;
```

4. To modify the NULL/NOT NULL

```
ALTER TABLE table_name
MODIFY column_name existing_datatype NULL/NOT NULL;
```

Requirement : Changing Phone to NULL

```
ALTER TABLE ACCOUNTS
MODIFY PHONE BIGINT NULL;
```

Requirement 2: Changing Area to NOT NULL from LOCATION Table

```
ALTER TABLE LOCATION
MODIFY AREA VARCHAR(20) NOT NULL;
```

5. To change the table name

```
ALTER TABLE table_name
RENAME new_table_name;
```

Requirement: Change the table name of Accounts to ACC

```
ALTER TABLE ACCOUNTS
RENAME ACC;
```

6. To change the column name

```
ALTER TABLE table_name
CHANGE old_col_name new_column_name existing_datatype;

ALTER TABLE ACC
CHANGE ACC_HOLDER_NAME NAME CHAR(20);
```

7. To add the constraints

```
ALTER TABLE table_name
ADD CONSTRAINT PRIMARY KEY(column_name);
ADD CONSTRAINT UNIQUE(column_name);
ADD CONSTRAINT CHECK(condition);
ADD CONSTRAINT FOREIGN KEY(column_name) REFERENCES
parent_table(column_name);

ALTER TABLE ACC
```

```
ADD CONSTRAINT FOREIGN KEY(BID) REFERENCES BRANCH(BID);
```

8. To remove Primary Key

```
ALTER TABLE table_name
DROP PRIMARY KEY;

ALTER TABLE ACC
DROP PRIMARY KEY;
```

9. To remove UNIQUE constraint

```
ALTER TABLE table_name
DROP INDEX column_name;

ALTER TABLE ACC
DROP INDEX PHONE;
```

10. To remove CHECK and FOREIGN KEY constraint

```
ALTER TABLE table_name
DROP CONSTRAINT constraint_name;
```

Step 1: Accessing Information_schema database

```
USE INFORMATION_SCHEMA;
```

Step 2: Displaying the tables that are present in information_schema

```
SHOW TABLES;
```

One among the table is known as TABLE_CONSTRAINTS;

Step 3: Describe table_constraints;

```
DESC table_constraints;
```

There is a column known as Constraint_name and that is where all our constraint names will be stored.

Step 4: Execute a DQL query to retrieve the data or constraint_name.

```
SELECT *
FROM TABLE_CONSTRAINTS
WHERE TABLE_NAME='provide your table name';
```

Note: TABLE_NAME is a column present in TABLE_CONSTRAINTS.

Step 5: The constraint names will be obtained. Copy the constraint name and come back to DEMO database

```
USE DEMO;
```

Step 6: Execute the ALTER command by providing the copied constraint_name

```
ALTER TABLE ACC
DROP CONSTRAINT  acc_ibfk_1;
```