Database: CREATE

To create database in the SQL, we use **"CREATE DATABASE"** statement.

Query Syntax:

```
CREATE DATABASE database_name;
```

Example 1: Create database named "test" in SQL.

Query:

```
CREATE DATABASE test;
```

Result:

Now your database is created in the memory.

Database: USE

To use database in the SQL, we use **"USE"** statement.

Query Syntax:

```
USE database_name;
```

Example 2: Use database created in Example 1

Query:

```
USE test;
```

Result:

Now you can use your database and create tables in the database.

Data Types in SQL

Some frequently used data types in SQL and their associated syntax, use and limits are given below.

Data Types	Use	Limit
`CHAR(SIZE)`	Holds a **fixed length String** (can contain letters, numbers and special characters). The fixed size is specified in parenthesis.	Can store up to 255 characters.
`VARCHAR(SIZE)`	Hold a **variable length String** (can contain letters, numbers and special characters). The maximum size is specified in parenthesis.	Can store up to 255 characters.
`INTEGER`	Holds numeric value	0 – 65536

Table: CREATE

To create table in a database we use following syntax.

Note: All table queries require selecting a database (USE command).

Syntax:

```
CREATE TABLE table_name
(
        column_name1 data_type(size),
        column_name2 data_type(size),
        column_name3 data_type(size),
        ....
);
```

Example 3: Create a table named "employee" that contains following three columns:

- id: an integer
- name: a variable length string with maximum 20 characters
- city: a variable length string with maximum 15 characters

Query:

```
CREATE TABLE employee
(
    id  INTEGER,
    name VARCHAR(20),
    city VARCHAR(15)
);
```

Result: Above query will create the table employee, but there is no data in the table (relation).

Table employee

id	name	city
Empty		

Table: INSERT

Inserting values into table requires usage of INSERT command and knowledge of column names and their associated data types. Values for string data type columns should be given inside single-quotes.

Syntax:

```
INSERT INTO table_name (column_name1, column_name2, ... column_nameN)
VALUES (value_1, value_2, ... value_N);
```

Example 3: Insert value into table named "employee" with following information:

- id: 1
- name: 'Ravi'
- city: 'Delhi'

Query:

```
INSERT INTO employee (id, name, city)
VALUES (1, 'Ravi', 'Delhi');
```

Result: This query will create 1 row in our table.

```
+------+-----+-------+
| id   | name | city  |
+------+-----+-------+
|    1 | Ravi | Delhi |
+------+-----+-------+
1 row in set (0.00 sec)
```

Example 4: Add more rows to table employee

Note: If you want to add more rows in the table, you have to run same query for each row.

Query:

```
INSERT INTO employee (name, id, city)
VALUES (2, 'Priyanka', 'Mumbai');

INSERT INTO employee (name, id, city)
VALUES (3, Rohan, 'Bangalore');
```

Result:

```
+------+----------+-----------+
| id   | name     | city      |
+------+----------+-----------+
|    1 | Ravi     | Delhi     |
|    2 | Priyanka | Mumbai    |
|    3 | Rohan    | Bangalore |
+------+----------+-----------+
3 rows in set (0.00 sec)
```

Table: SELECT

To fetch and display data from table we use SELECT command. We can display data for all the columns or any specific columns.

Syntax to display data for All columns:

```
SELECT * FROM table_name;
```

Syntax to display data for Specific columns:

```
SELECT column_name1, column_name2, ... column_nameN FROM table_name;
```

Example 5: Fetch all the data from employee table

Query:

```
SELECT * FROM employee;
```

Result:

```
+------+----------+-----------+
| id   | name     | city      |
+------+----------+-----------+
|    1 | Ravi     | Delhi     |
|    2 | Priyanka | Mumbai    |
|    3 | Rohan    | Bangalore |
+------+----------+-----------+
3 rows in set (0.00 sec)
```

Example 6: Fetch only name and city from employee table

Query:

```
SELECT name, city FROM employee;
```

Result:

```
+----------+-----------+
| name     | city      |
+----------+-----------+
| Ravi     | Delhi     |
| Priyanka | Mumbai    |
| Rohan    | Bangalore |
+----------+-----------+
3 rows in set (0.00 sec)
```

Example 7: Fetch only id and name from employee table

Query:

```
SELECT name, id FROM employee;
```

Result:

```
+------+----------+
| id   | name     |
+------+----------+
|    1 | Ravi     |
|    2 | Priyanka |
|    3 | Rohan    |
+------+----------+
3 rows in set (0.00 sec)
```

Table: DELETE

To delete data from the table, we use DELETE statement. DELETE statement delete the selected rows or all row from the table, but the table remains there. You can still add new rows, if you want.

Syntax to delete all rows:

```
DELETE FROM table_name;
```

Syntax to delete specific rows:

```
DELETE FROM table_name
WHERE some_column = some_value;
```

Table: TRUNCATE

We can also use TRUNCATE to delete data from the table however unlike DELETE, here we cannot delete specific rows. TRUNCATE is faster than DELETE. Just like DELETE command the table still remains in database, so we can add rows after TRUNCATE command too.

Syntax to delete all rows:

```
TRUNCATE TABLE table_name;
```

Table: DROP

DROP statement is used to delete and remove complete table from the database. You would need to re-create this table once again if you wish to store some data in future. If you try to fetch data from dropped table you'll get an error.

Syntax remove table from database:

```
DROP TABLE table_name;
```

Example 8: Fetch only name and city from employee table

Query:

```
DROP TABLE employee;
SELECT * FROM employee;
```

Result:

```
Error: Table test.employee doesn't exist.
```

Problem 1.1	**Create a table named student with following columns:** ➢ **student_id: integer** ➢ **first_name: string with maximum 15 characters** ➢ **last_name: string with maximum 15 characters** ➢ **age: integer** ➢ **city: string with maximum 20 characters** ➢ **course_code: string with fixed length of 3 characters**
Problem 1.2	**Insert following values in table student created in Problem 1.1.** ➢ **1,GIRISH,SHARMA,16,GANGANAGAR,PHY** ➢ **2,SATYAM,KUMAR,15,BHOPAL,BIO** ➢ **3,NEHA,JAIN,16,ALWAR,BIO**
Problem 1.3	**Write a query for fetching all details of table student.**
Problem 1.4	**Write a query for fetching only first_name and last_name along with their age.**
Problem 1.5	**Write a query to fetch all the course_codes present.**

Problem 2 **What is the difference between DELETE, TRUNCATE and DROP command?**

Problem 3 **Give output for each of the queries given below.**

Query 3.1

```
TRUNCATE TABLE student;
SELECT * FROM student;
```

Query 3.2

```
DROP TABLE student;
SELECT * FROM student;
```