Unit 1 - Chapter 4,5

```
CREATE DATABASE DatabaseName;
SHOW DATABASES;
USE DatabaseName;
DROP DATABASE DatabaseName;
```

```
CREATE TABLE table_name(
   column1 datatype,
   column2 datatype,
   column3 datatype,
   .....
   columnN datatype,
   PRIMARY KEY( one or more columns )
);
```

Example:

```
SQL> CREATE TABLE CUSTOMERS(
   ID   INT              NOT NULL,
   NAME VARCHAR (20)     NOT NULL,
   AGE  INT              NOT NULL,
   ADDRESS  CHAR (25) ,
   SALARY   DECIMAL (18, 2),
   PRIMARY KEY (ID)
);
```

```
DESC CUSTOMERS;
+---------+---------------+------+-----+---------+-------+
| Field   | Type          | Null | Key | Default | Extra |
+---------+---------------+------+-----+---------+-------+
| ID      | int(11)       | NO   | PRI |         |       |
| NAME    | varchar(20)   | NO   |     |         |       |
| AGE     | int(11)       | NO   |     |         |       |
| ADDRESS | char(25)      | YES  |     | NULL    |       |
| SALARY  | decimal(18,2) | YES  |     | NULL    |       |
+---------+---------------+------+-----+---------+-------+
```

Basic syntax of DROP TABLE statement is as follows:

```
DROP TABLE table_name;
```

The SQL INSERT INTO syntax would be as follows:

```
INSERT INTO TABLE_NAME VALUES (value1,value2,value3,...valueN);
```

Example:

```
INSERT INTO CUSTOMERS VALUES (7, 'Muffy', 24, 'Indore', 10000.00 );
```

If you want to fetch all the fields available in the field, then you can use the following syntax:

```
SELECT * FROM table_name;
```

Following is an example, which would fetch ID, Name and Salary fields of the customers available in CUSTOMERS table:

```
SQL> SELECT ID, NAME, SALARY FROM CUSTOMERS;
```

SQL - WHERE Clause

```
SQL> SELECT ID, NAME, SALARY
FROM CUSTOMERS
WHERE SALARY > 2000;
SQL> SELECT ID, NAME, SALARY
FROM CUSTOMERS
WHERE NAME = 'Hardik';
```

SQL - AND and OR Operators

```
SQL> SELECT ID, NAME, SALARY
FROM CUSTOMERS
WHERE SALARY > 2000 AND age < 25;
```

```
SQL> SELECT ID, NAME, SALARY
FROM CUSTOMERS
WHERE SALARY > 2000 OR age < 25;
```

SQL - UPDATE Query

```
SQL> UPDATE CUSTOMERS
SET ADDRESS = 'Pune'
WHERE ID = 6;
```

```
SQL> UPDATE CUSTOMERS
```

```
SET ADDRESS = 'Pune', SALARY = 1000.00;
```

SQL - DELETE Query

```
SQL> DELETE FROM CUSTOMERS
WHERE ID = 6;
```

If you want to DELETE all the records from CUSTOMERS table, you do not need to use WHERE clause and DELETE query would be as follows:

```
SQL> DELETE FROM CUSTOMERS;
```

SQL - LIKE Clause

Following is an example, which would display all the records from CUSTOMERS table where SALARY starts with 200:

```
SQL> SELECT * FROM CUSTOMERS
WHERE SALARY LIKE '200%';
```

WHERE SALARY LIKE '%2'	Finds any values that end with 2
WHERE SALARY LIKE '%200%'	Finds any values that have 200 in any position

SQL - ORDER BY Clause

Following is an example, which would sort the result in ascending order by NAME and SALARY:

```
SQL> SELECT * FROM CUSTOMERS
```

```
    ORDER BY NAME, SALARY;
```

```
SQL> SELECT * FROM CUSTOMERS
    ORDER BY NAME DESC;
```

```
SQL> SELECT * FROM CUSTOMERS
    ORDER BY NAME, SALARY;
```

```
SQL> SELECT * FROM CUSTOMERS
    ORDER BY NAME DESC;
```

SQL - Distinct Keyword

```
SQL> SELECT DISTINCT SALARY FROM CUSTOMERS
    ORDER BY SALARY;
```

SQL - ALTER TABLE Command

The basic syntax of **ALTER TABLE** to add a new column in an existing table is as follows:

```
ALTER TABLE table_name ADD column_name datatype;
```

The basic syntax of ALTER TABLE to **DROP COLUMN** in an existing table is as follows:

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

The basic syntax of ALTER TABLE to change the **DATA TYPE** of a column in a table is as follows:

```
ALTER TABLE table_name MODIFY COLUMN column_name datatype;
```

The basic syntax of ALTER TABLE to add a **NOT NULL** constraint to a column in a table is as follows:

```
ALTER TABLE table_name MODIFY column_name datatype NOT NULL;
```

The basic syntax of ALTER TABLE to **ADD UNIQUE CONSTRAINT** to a table is as follows:

```
ALTER TABLE table_name

ADD CONSTRAINT MyUniqueConstraint UNIQUE(column1, column2...);
```

The basic syntax of ALTER TABLE to **ADD CHECK CONSTRAINT** to a table is as follows:

```
ALTER TABLE table_name

ADD CONSTRAINT MyUniqueConstraint CHECK (CONDITION);
```

The basic syntax of ALTER TABLE to **ADD PRIMARY KEY** constraint to a table is as follows:

```
ALTER TABLE table_name

ADD CONSTRAINT MyPrimaryKey PRIMARY KEY (column1, column2...);
```

The basic syntax of ALTER TABLE to **DROP CONSTRAINT** from a table is as follows:

```
ALTER TABLE table_name

DROP CONSTRAINT MyUniqueConstraint;
```

The basic syntax of ALTER TABLE to **DROP PRIMARY KEY** constraint from a table is as follows:

```
ALTER TABLE table_name

DROP CONSTRAINT MyPrimaryKey;
```

SQL - TRUNCATE TABLE Command

The basic syntax of **TRUNCATE TABLE** is as follows:

```
TRUNCATE TABLE  table_name;
```

Rename Table

```
alter table
  table_name
rename to
  new_table_name;
```

Rename Column Name

Syntax

RENAME COLUMN <u>table-Name</u>.<u>simple-Column-Name</u> TO <u>simple-Column-Name</u>

Examples

To rename the manager column in table employee to supervisor, use the following syntax:

RENAME COLUMN EMPLOYEE.MANAGER TO SUPERVISOR