



Chapter-2: Structure Query Language

Learning Objectives

At the end of this chapter the students will be able to understand:

- ❖ What is SQL?
- ❖ Need for SQL
- ❖ How to create tables in SQL?
- ❖ How to add information to tables?
- ❖ SELECT ... FROM...WHERE (with aggregate functions)
- ❖ GROUP BYHAVING
- ❖ ORDER BY
- ❖ UPDATE AND DELETE Command
- ❖ ALTER TABLE AND DROP TABLE Command
- ❖ EQUIJOIN

Introduction

SQL (Structured Query Language) is a standard language for accessing and manipulating databases. SQL commands are used to create, transform and retrieve information from Relational Database Management Systems and also used to create interface between user and database. By using SQL commands, one can search any data in the database and perform other functions like, create tables, add records, modify data, remove rows, drop table etc. SQL commands are used to implement the following;

- ❖ 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 views in a database

CREATE TABLE Command

CREATE TABLE command is used to create table structure. In this command, we need to give full information about table such as number of columns, type of each column and constraints (primary key).

The CREATE TABLE command requires:



Computer Science



- ❖ Name of the table,
- ❖ Names of fields,
- ❖ Definitions and constraints for each field.

Constraints

In SQL, we have the following constraints:

- ❖ NOT NULL - To check a column cannot store NULL value.
- ❖ PRIMARY KEY - To check that a column have an unique identity which helps to find a particular record in a table.

Syntax:

```
CREATE TABLE <table name>
(<column name1> <data type>[size][constraints],
<column name2> <data type>[size][constraints],
.
.
.
<column name n> <data type>[size][constraints]);
```

Example:

Create the following table:

Table: Student

Column Name	Data Type	Size	Constraints
Adno	Numeric	3	Primary key
Name	Varchar	20	NOT NULL
Class	Numeric	2	
Section	Char	1	
Fees	Numeric	10, 2	

Command:

```
CREATE TABLE student
(Adno Numeric (3) Primary Key,
Name varchar (20) not null,
Class Numeric (2),
Section char (1),
Fees numeric (10, 2));
```



Computer Science



INSERT INTO Command:

This command is used to add rows in the table, but can add only one row at a time.

Syntax:

```
INSERT INTO <table name> [Column_name1, Column_name2, .....Column_name n] VALUES (value1,value2,value3,....,value n);
```

OR

```
INSERT INTO <table name> VALUES (value1,value2,value3,....,value n);
```

Note: [] Option

Example:

Insert the following information to the table student:

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	2500
222	Mohit Sharma	11	B	4500
333	K.P.Gupta	12	B	3000
444	Ajit Kumar	10	A	2000
555	Nandini	12	C	3000
666	Rohan Sharma	11	B	2500

```
INSERT INTO student VALUES (111,"Anu Jain",12,"A",2500);
```

```
INSERT INTO student VALUES (222,"Mohit Sharma",11,"B",4500);
```

[**Note:** If we want to insert values from the selective columns then we have to use this method INSERT INTO student (ADNO, Name, CLASS) VALUES (777,'LEENA', 'B');]

SELECT Command

This command is used to view table information from SQL database. By using SELECT command, we can get one or more fields information, while using *, one can get all fields information.

Syntax:

```
SELECT (* / field list)
```

```
FROM <table name>
```

```
[WHERE <condition>;]
```

We can specify any condition using where clause. Where clause is optional.



Example:

1. Display student table information.

```
SELECT *
```

```
FROM student;
```

This will display all information of the particular table (student) in the database.

2. To display name and class of student table information.

```
SELECT name, class
```

```
FROM student;
```

3. To display name of 10th class student information.

```
SELECT name
```

```
FROM student
```

```
WHERE class = 10;
```

Operators used in SQL commands:

Arithmetic operators:

Arithmetic operator takes two operands and performs a mathematical calculation on them. However, they can be used only in SELECT command. The arithmetic operators used in SQL are:

+ Addition

- Subtraction

* Multiplication

/ Division

Example (string join)

- 1) **Table:** Name

First Name	Second Name
Anu	Jain
Madhu	Bhattia

Display first name with second name.

```
SELECT FirstName + SecondName
```

```
FROM Name;
```

Output:



Computer Science



FirstName + SecondName
Anu Jain
Madhu Bhattia

2) Table: Salary

Basic	DA
25000	5000
35000	7000

SELECT Basic + DA

FROM Salary;

Output:

Basic + DA
30000
42000

SELECT Basic + DA as "NET PAY"

FROM Salary;

[Note: If we want to give new name of the column then we have to use above format]

Output:

NET PAY
30000
42000

Select DA-100

From salary;

Output:

DA-100
4900
6900

Select DA*100

From salary;



Output:

DA*100
500000
700000

Select DA/100

From salary;

Output:

DA/100
50
70

Relational operators:

Relational operators are used to implement comparison between two operands. These operators can be used only in 'where clause'. Relational operators are -

< less than

> greater than

<= less than or equal to

>= greater than or equal to

= equal to

!= not equal to

Example:

Table: Student

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	2500
222	Mohit Sharma	11	B	4500
333	K.P.Gupta	12	B	3000
444	Ajit Kumar	10	A	2000
555	Nandini	12	C	3000
666	Rohan Sharma	11	B	2500



Computer Science



1. Display students' name, who are paying below 3000 fees.

```
SELECT name  
FROM student  
WHERE fees < 3000;
```

Output:

Name
Anu Jain
Ajit Kumar
Rohan Sharma

2. Display students' name, who are paying above or equal to 3000 fees.

```
SELECT name  
FROM student  
WHERE fees >= 3000;
```

Output:

Name
Mohit Sharma
Nandini

3. Display students' information, who are not in class 10

```
SELECT *  
FROM student  
WHERE class != 10;
```

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	2500
222	Mohit Sharma	11	B	4500
333	K.P.Gupta	12	B	3000
555	Nandini	12	C	3000
666	Rohan Sharma	11	B	2500

Logical operators:

Logical operators are also possible only in 'where clause' and are used to merge more than one condition. Logical operators are:



AND

OR

NOT

Example:

1. Display information of students in class 11B.

SELECT *

FROM student

WHERE class = 11 AND section = 'B';

Adno	Name	Class	Section	Fees
222	Mohit Sharma	11	B	4500
666	Rohan Sharma	11	B	2500

2. Display 11th and 12th class students' information.

SELECT *

FROM student

WHERE class = 11 OR class = 12;

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	2500
222	Mohit Sharma	11	B	4500
333	K.P.Gupta	12	B	3000
555	Nandini	12	C	3000
666	Rohan Sharma	11	B	2500

3. Display students' information, who are not in 10th class.

SELECT *

FROM student

WHERE NOT class = 10;

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	2500
222	Mohit Sharma	11	B	4500
333	K.P.Gupta	12	B	3000
555	Nandini	12	C	3000
666	Rohan Sharma	11	B	2500



Computer Science



LIKE OPERATOR

LIKE OPERATOR is used to search a value similar to specific pattern in a column using wildcard operator. There are two wildcard operators - percentage sign (%) and underscore (_). The percentage sign represents zero, one, or multiple characters, while the underscore represents a single number or character. The symbols can be used in combinations.

For example:

1. Display the names that start with letter "A".

```
SELECT name
```

```
FROM student
```

```
WHERE name LIKE "A%";
```

Here, % replaces one or more characters.

Name
Anu Jain

2. Display names, whose name's second letter is 'o'.

```
SELECT name
```

```
FROM student
```

```
WHERE name LIKE "_o%";
```

Here, % replaces one or more than one character and _ replaces only one character.

Name
Mohit Sharma
Rohan Sharma

3. Display names, whose name has 7 characters.

```
SELECT name
```

```
FROM student
```

```
WHERE name LIKE "_____";
```

Here, _ replaces only one character. As such, 7 underscores replace 7 characters.

Name
Nandini

IN Operator

The IN operator allows us to specify multiple values in a WHERE clause



For example:

Display students' information, who are in section A and B.

```
SELECT *
```

```
FROM student
```

```
WHERE class IN ("A","B");
```

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	2500
222	Mohit Sharma	11	B	4500
333	K.P.Gupta	12	B	3000
444	Ajit Kumar	10	A	2000
666	Rohan Sharma	11	B	2500

BETWEEN Operator

The BETWEEN operator is used to test whether or not a value (stated before the keyword BETWEEN) is "between" the two values stated after the keyword BETWEEN.

For example:

Display students' information, who are paying fees between 2500 and 3500.

```
SELECT *
```

```
FROM student
```

```
WHERE fees BETWEEN 2500 AND 3500;
```

[Note: In the above Query 2500 and 3500 is also included]

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	2500
333	K.P.Gupta	12	B	3000
444	Ajit Kumar	10	A	2000
555	Nandini	12	C	3000
666	Rohan Sharma	11	B	2500

ORDER BY

This command is used to arrange values in ascending or descending order.

For example:



Computer Science



SELECT *

FROM student

ORDER BY fees ASC;

'asc' for ascending order. Without asc also the list is displayed with ascending order only.

Adno	Name	Class	Section	Fees
444	Ajit Kumar	10	A	2000
111	Anu Jain	12	A	2500
666	Rohan Sharma	11	B	2500
333	K.P.Gupta	12	B	3000
555	Nandini	12	C	3000
222	Mohit Sharma	11	B	4500

SELECT *

FROM student

ORDER BY fees DESC;

'desc' for descending order. If the 'desc' is not given, the list will be displayed with ascending order.

Adno	Name	Class	Section	Fees
222	Mohit Sharma	11	B	4500
555	Nandini	12	C	3000
333	K.P.Gupta	12	B	3000
666	Rohan Sharma	11	B	2500
111	Anu Jain	12	A	2500
444	Ajit Kumar	10	A	2000

Aggregate functions

Aggregate functions are used to implement calculation based upon a particular column. These functions always return a single value.

Aggregate functions are:

1. SUM()
2. AVG()
3. MAX()



4. MIN()
5. COUNT()

SUM()

This function is used to find the total value of a particular column.

Example:

```
SELECT SUM (fees)
FROM student;
```

SUM (fees)
17500

AVG()

This function is used to find the average value of a particular column.

Example:

```
SELECT AVG (fees)
FROM student;
```

AVG (fees)
2916.6666

MAX()

This function is used to find the maximum value of a particular column.

Example:

```
SELECT MAX (fees)
FROM student;
```

MAX (fees)
4500

MIN()

This function is used to find the minimum value of a particular column.

Example:

```
SELECT MIN (fees)
FROM student;
```



Computer Science



MIN(fees)

2000

COUNT()

This function is used to find the number of values (ie. number of rows) of a particular column.

Example:

```
SELECT COUNT (fees)
```

```
FROM student;
```

(or)

```
SELECT COUNT (*)
```

```
FROM student;
```

COUNT (fees)

6

(or)

COUNT (*)

6

GROUP BY

The SQL GROUP BY is a clause that enables SQL aggregate functions for grouping of information. (ie. GROUP BY clause is used in collaboration with the SELECT statement to arrange identical data into groups.). This clause is used whenever aggregate functions by group are required.

For example:

1. Display number of students in each class.

```
SELECT count (*), class
```

```
FROM student
```

```
GROUP BY class;
```

Count (*)	Class
2	11
3	12
1	10



2. Display sum of fees for each class.

```
SELECT class, sum (fees)
```

```
FROM student
```

```
GROUP BY class;
```

Class	Sum (fees)
11	7000
12	8500
10	2000

Having clause

As mentioned earlier, the 'where' clause is used only to place condition on the selected columns, whereas the 'HAVING' clause is used to place condition on groups created by 'group by' clause, because here the 'WHERE' clause is not useable.

Example:

Display sum of fees which is more than 5000 for each class

```
SELECT class, sum (fees)
```

```
FROM student
```

```
GROUP BY class
```

```
HAVING sum (fees)>5000;
```

Class	Sum (fees)
11	7000
12	8500

DISTINCT

The DISTINCT keyword is used to remove duplicate values in a particular column.

For example:

Display class in student table.

```
SELECT class
```

```
FROM student;
```

Class
12
11



Computer Science



12
10
12
11

Display different classes from student table.

```
SELECT DISTINCT class
```

```
FROM student;
```

Class
12
11
10

UPDATE Command

This command is used to implement modification of the data values.

Syntax:

```
UPDATE <table name>
```

```
SET <column name1>=new value, <column name>=new value etc
```

```
[WHERE <condition>];
```

Example:

1. Increase fees value by 500.

```
UPDATE student
```

```
SET fees = fees + 500;
```

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	3000
222	Mohit Sharma	11	B	5000
333	K.P.Gupta	12	B	3500
444	Ajit Kumar	10	A	2500
555	Nandini	12	C	3500
666	Rohan Sharma	11	B	3000



2. Increase the fees value by 100 for adno 222.

UPDATE student

SET fees = fees+100

WHERE adno = 222;

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	3000
222	Mohit Sharma	11	B	5100
333	K.P. Gupta	12	B	3500
444	Ajit Kumar	10	A	2500
555	Nandini	12	C	3500
666	Rohan Sharma	11	B	3000

DELETE Command

This command is used to remove information from a particular row or rows. Please remember that this command will delete only row information but not the structure of the table.

Syntax:

DELETE

FROM <table name>

[WHERE <condition>;]

For example:

1. Remove adno 444 information.

DELETE

FROM student

WHERE adno = 444;

Adno	Name	Class	Section	Fees
111	Anu Jain	12	A	3000
222	Mohit Sharma	11	B	5100
333	K.P.Gupta	12	B	3500
555	Nandini	12	C	3500
666	Rohan Sharma	11	B	3000



Computer Science



2. Remove all records.

DELETE

FROM student;

Adno	Name	Class	Section	Fees

ALTER TABLE command

This command is used to implement modification of the structure of the table. This is a DDL command. Using this command, we can add a new column, remove the existing column and modify data type of existing column.

Syntax:

ALTER TABLE <table name>

[ADD/MODIFY/DROP] <column name>;

For example:

1. Add one new column totalfees with number (10,2).

ALTER TABLE student

ADD totalfees number(10,2);

2. Change totalfees datatype as number(12,2).

ALTER TABLE student

MODIFY totalfees number(12,2);

3. Remove totalfees column.

ALTER TABLE student

DROP totalfees;

DROP TABLE Command

This command is used to remove the entire structure of the table and information. This is also from the DDL command.

Syntax:

DROP TABLE <table name>;

For example:

Remove the whole structure of student table.

DROP TABLE student;



Equi Join

Equi Joins are used to give information in different tables. It is a special type of join in which we use only equality operator.

For example

```
SELECT *
FROM product, customer
WHERE product.product_no = customer.product_no;
(or)
SELECT *
FROM product p, customer c
WHERE p.product_no=c.product_no;
```

Product_no	Product_name	Price	Cust_no	Cust_name	City	Product_no
111	Computer	50000	301	Rohan	Bangalore	111
222	Printer	10000	201	Mohan	Mumbai	222
333	Scanner	12000	101	Kavitha	Delhi	333
333	Scanner	12000	401	Sahil	Mumbai	333
444	Modem	500	501	Rohita	Delhi	444

SQL Non-equi joins

The non-equi join is a type of join in which, we use only relational operators except equal operator. These include >, <, >=, <= and !=.

For example

```
SELECT *
FROM product, customer
WHERE product.product_no != customer.product_no;
```

Product_no	Product_name	Price	Cust_no	Cust_name	City	Product_no
111	Computer	50000	201	Mohan	Mumbai	222
111	Computer	50000	101	Kavitha	Delhi	333
111	Computer	50000	401	Sahil	Mumbai	333
111	Computer	50000	501	Rohita	Delhi	444



Computer Science



222	Printer	10000	301	Rohan	Bangalore	111
222	Printer	10000	101	Kavitha	Delhi	333
222	Printer	10000	401	Sahil	Mumbai	333
222	Printer	10000	501	Rohita	Delhi	444
333	Scanner	12000	301	Rohan	Bangalore	111
333	Scanner	12000	201	Mohan	Mumbai	222
333	Scanner	12000	501	Rohita	Delhi	444
444	Modem	500	301	Rohan	Bangalore	111
444	Modem	500	201	Mohan	Mumbai	222
444	Modem	500	101	Kavitha	Delhi	333
444	Modem	500	401	Sahil	Mumbai	333



Computer Science



LET'S REVISE

- ❖ **CREATE TABLE:** Used to create structure of the table.
- ❖ **ALTER TABLE:** Used to implement structure modification.
- ❖ **DROP TABLE:** To remove full structure.
- ❖ **INSERT INTO:** To add row values.
- ❖ **SELECT:** To display row or column information.
- ❖ **DISTINCT:** To select different information.
- ❖ **MAX():** To select maximum value of a particular column.
- ❖ **MIN():** To select minimum value of a particular column.
- ❖ **SUM():** To find total value of a particular column.
- ❖ **AVG():** To find average value of a particular column.
- ❖ **COUNT():** Number of records in the table.



Computer Science



EXERCISE

1. Mr. Rohan has created a table 'student' with rollno., name, class and section. Now he is confused to set the primary key. So identify the primary key column.

2. Ms. Ravina is using a table 'customer' with custno, name, address and phonenumner. She needs to display name of the customers, whose name start with letter 'S'. She wrote the following command, which did not give the result.

Select * from customer where name="S%";

Help Ms. Ravina to run the query by removing the errors and write the correct query.

3. Write SQL query to add a column totalprice with data type numeric and size 10, 2 in a table product.
4. The name column of a table 'student' is given below.

Name
Anu Sharma
Rohan Saini
Deepak Sing
Kannika Goal
Kunal Sharma

Based on the information, find the output of the following queries:

❖ Select name from student where name like "%a";

❖ Select name from student where name like "%h%";

5. A customer table is created with cno,cname, and address columns. Evaluate the following statement whether it is correct or not?

Delete cname from customer;

6. Shopkeeper needs to change the first name of one of his customers in table 'customer'. Which command should he use for the same?

7. Sonal needs to display name of teachers, who have "o" as the third character in their name. She wrote the following query.

Select name

From teacher

Where name = "\$\$o?";



Computer Science



But the query is not producing the result. Identify the problems.

8. Pooja created a table 'bank' in SQL. Later on, she found that there should have been another column in the table. Which command is used to add column to the table?
9. Surpreeth wants to add two more records of customer in customer table. Which command is used to implement this?
10. Deepak wants to remove all rows from the tableBank. But he needs to maintain the structure of the table. Which command is used to implement the same?
11. While creating table 'customer', Rahul forgot to add column 'price'. Which command is used to add new column in the table. Write the command to implement the same.
12. Write the syntax of creating table command.
13. Write the syntax of dropping table command.
14. What all are the clause possible in select statement.
15. What is the default value of order by command.
16. Differentiate between delete and drop table command.
17. Differentiate between update and alter table command.
18. Differentiate between order by and group by command.
19. Define the following.
 - a) Union
 - b) Cartesian product
 - c) EquiJoin
 - d) Non equi join.
20. What is the use of wildcard?
21. Create the following table items.

Column name	Data type	Size	Constrain
Itemno	Number	3	Primary key
Iname	Varchar	15	
Price	Number	10,2	
Quantity	Number	3	



Computer Science



22. Insert the following information:

Table: Item

Itemno	Iname	Price	Quantity
101	Soap	50	100
102	Powder	100	50
103	Face cream	150	25
104	Pen	50	200
105	Soap box	20	100

23. Write queries based upon item table given in q. no 22.

- Display all items information.
- Display item name and price value.
- Display soap information.
- Display the item information whose name starts with letter 's'.
- Display a report with item number, item name and total price. (total price = price * quantity).
- Display item table information in ascending order based upon item name.
- Display item name and price in descending order based upon price.
- Display item name, whose price is in between 50 to 100.
- Add new column totalprice with number (10, 2).
- Increase price value by 100.
- Fill up totalprice = price * quantity.
- Remove powder information.
- Remove totalprice column.
- Remove whole item structure.

24. Write outputs based upon item table given in q. no 22.

- select sum(price) from item;
- select avg(price) from item;
- select min(price) from item;
- select max(price) from item;
- select count(price) from item;
- select distinct price from item;
- select count(distinct price) from item;



Computer Science



h) `select iname,price*quantity from item;`

25. In a database there are two tables - 'Brand' and 'Item' as shown below:

BRAND:

ICODE	BNAME
100	SONY
200	HP
300	LG
400	SAMSUNG

ITEM:

ICODE	INAME	PRICE
100	TELEVISION	25000
200	COMPUTER	30000
300	REFRIGERATOR	23000
400	CELL PHONE	40000

Write MYSQL queries for the following:

- To display Iname, price and corresponding Brand name (Bname) of those items, whose price is between 25000 and 30000 both values inclusive).
- To display ICode, Price and BName of the item, which has IName as "Television".
- To increase the Prices of all items by Rs. 10%.

26. Create the following table

Students

Column name	Data type	Size	Constraints
Adno	Integer	3	Primary key
Name	Varchar	20	
Average	Integer	3	
Sex	Char	1	
Scode	Integer	4	



Computer Science



27. Insert the following information:

Students

Adno	Name	Average	Sex	Scode
501	R.Jain	98	M	111
545	Kavita	73	F	333
705	K.Rashika	85	F	111
754	Rahul Goel	60	M	444
892	Sahil Jain	78	M	333
935	Rohan Saini	85	M	222
955	Anjali	64	F	444
983	Sneha Aggarwal	80	F	222

28. Write queries based upon item table given in q. no 27.

- Display all students' information.
- Display Rohan Saini's information.
- Display number of students in the table.
- Display number of students in each sex.
- Display students' information in ascending order using name.
- Display students' information in descending order using average marks.
- Display students' name starting with letter "K".
- Display students' information, whose name ends with "I".
- Display a report with adno,name,average*5 as total marks from student table.
- Display students' information, whose average marks are in between 80 to 90.
- Display students' info., who are getting average marks of more than 80 and scode 333.
- Display students' name and average marks, whose scode is 222 and 333.
- Display sum of average marks.
- Display maximum average marks
- Display minimum average marks.



- (xvi) Display average value of average marks.
- (xvii) Display maximum, minimum and sum of average marks in each scode.
- (xviii) Display number of students in each scode.

29. Create the following table.

Column name	Data type	Size	Constraints
Scode	Integer	3	Primary key
Sname	Varchar	20	
Place	Varchar	10	

30. Insert the following information.

Streams

Scode	Sname	Place
111	Science	SBlock
222	Commerce	CBlock
333	Humanity	HBlock
444	Art	ABlock

31. Write queries based upon item table given in q. no 27& 30

- (i) To display Adno, Name, Sex and Average from Student's table and Stream name (Sname) and place from Stream table with respect to Scode.
- (ii) Add the following information into the student table.
999 Deepak Sharma 83 M 2222
- (iii) Display science stream students' information.

32. Give the output of the following SQL queries.

- (i) Select sum(Average)
From students
Where sex='M';



Computer Science



(ii) Select distinct (Scode)

From students;

33. Remove 111 scode information.
34. Add new column state with varchar(10).
35. Increment 2 marks for 444 scode info.
36. Remove column state.
37. Remove the whole table stream.