Unit-3: Database Management Systems and SQL

Chapter-1: Database Concepts and SQL

Learning Objectives

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

- What is DBMS?
- What is relational database model?
- Relation
- Tuples
- SQL
- DDL
- DML
- Relational Algebra
- Selection
- Projection
- Union
- Cartesian Product

Introduction

Database is a collection of related information that is organized in such a way that supports for easy access, modify and maintain data. The contents of a database are obtained by combining data from all the different sources in an organization. Generally, the database is managed by some special software packages known as Database Management Systems (DBMSs). DBMSs are specially designed applications to create connection between user and program, and to store data in an organized manner. The purpose of DBMSs software is to allow the user to create, modify and administration of database. Examples of database management systems are: Ms-Access, MySQL, PostgreSQL, SQLite, Microsoft SQL Server, Oracle, SAP, dBASE, FoxPro, etc.

Relational data model

The relational data model is a database model based on first-order predicate logic(First Order Predicate Logic is one where the quantification is over simple variables), formulated and proposed by Edgar F. Codd. in 1969. The first-order predicate logic is a symbolised reasoning, in which statement is broken down into a subject and a predicate. The predicate modifies the properties of the subject, while in the first-order logic, a predicate can only refer to a single subject. In the relational data model, database is represented as collection of related tables. Each table is termed as relation and has its unique name in the relational data model. Tables are formed by using rows and columns. A row (horizontal subset) of a table represents a tuple or record, while column (vertical subset) of a table represents an attribute.

Relation

In database, a relation means a 'table', in which data are organized in the form of rows and columns. Therefore in database, relations are equivalent to tables.

For example

Relation: Student

Ad No	Name	Class	Section	Average
101	Anu	12	A	85
105	Balu	12	D	65
203	Leena	11	B	95
205	Madhu	10	B	75
305	Surpreeth	9	C	70
483	Usha	6	A	60

Domain

A domain is the original sets of atomic values used to model data. In data base management and database, a domain refers to all the possible unique values of a particular column.

For example:

i) The domain of gender column has a set of two possible values i.e, Male or Female.

ii) The domain of marital status has a set of four possible values i.e, Married, Unmarried, Widows and Divorced.

Therefore, a domain is a set of acceptable values of a particular column, which is based on various properties and data types. We will discuss data types later in this chapter.

Tuple

Horizontal subset/information in a table is called tuple. The tuple is also known as a 'record', which gives particular information of the relation (table).

For example:

i) In customer table, one row gives information about one customer only.

ii) In student table, one row gives information about one student only.

Key

Keys are an important part of a relational database and a vital part of the structure of a table. They help enforce integrity and help identify the relationship between tables. There are three main types of keys -

candidate keys, primary keys, foreign keys and alternate keys.

Primary Key: A column or set of columns that uniquely identifies a row within a table is called primary key.

Candidate Key: Candidate keys are set of fields (columns with unique values) in the relation that are eligible to act as a primary key.

Alternate Key: Out of the candidate keys, after selecting a key as primary key, the remaining keys are called alternate key.

Foreign Key: A foreign key is a field (or collection of fields) in one table that uniquely identifies a row of another table. In other words, a foreign key is a column or a combination of columns that is used to establish a link between two tables.

Degree

The degree is the number of attributes (columns) in a table.

Cardinality

Cardinality is number of rows (tuples) in a table.

Example:

Relation: Student

Ad No	Name	Class	Section	Average
101	Anu	12	A	85
105	Balu	12	D	65
203	Leena	11	B	95
205	Madhu	10	B	75
305	Surpreeth	9	C	70
483	Usha	6	A	60

Fields (Attributes/Columns):- AdNo, Name, Class, Section and Average.

Tuples (Rows/Records):

101	Anu	12	A	85

Domain: Possible values of section are ('A','B','C','D')

Degree: 5 (Number of columns).

Cardinality: 6 (Number of rows).

Candidate Key: In the above table, AdNo and Name has unique values. Therefore, AdNo and Name are candidate keys.

Primary Key: Out of the AdNo and Name, AdNo is the primary key.

Alternate Key: In the candidate key, AdNo is the primary key and the Name is the Alternate key.

Structured Query Language (SQL)

Structured Query Language (SQL) is a standard language used for accessing databases. This is a special purpose programming language used to create a table, manage data and mainipulate data.

SQL provides statements for a variety of tasks, including:

i) Querying data
ii) Inserting, updating, and deleting rows in a table
iii) Creating, replacing, altering, and dropping objects (tables)
iv) Controlling access to the database and its objects (tables)
v) Guaranteeing database consistency and integrity

SQL unifies all of the preceeding tasks in one consistent language.

Advantages of using SQL:

1) **SQL is portable:** SQL is running in all servers, mainframes, PCs, laptops, and even mobile phones.
2) **High speed:** SQL queries can be used to retrieve large amounts of records from a database quickly and efficiently.
3) **Easy to learn and understand:** SQL generally consists of English statements and as such, it is very easy to learn and understand. Besides, it does not require much coding unlike in programming languages.
4) **SQL is used with any DBMS system with any vendor:** SQL is used by all the vendors who develop DBMS. It is also used to create databases, manage security for a database, etc. It can also be used for updating, retrieving and sharing data with users.
5) **SQL is used for relational databases:** SQL is widely used for relational databases.
6) **SQL acts as both programming language and interactive language:** SQL can do both the jobs of being a programming language as well as an interactive language at the same time.
7) **Client/Server language:** SQL is used for linking front end computers and back end databases. It provides client server architecture (Email, and the World Wide Web - all apply the client-server architecture).
8) **Supports object based programming:** SQL supports the latest object based programming and is highly flexible.

Types of SQL Statements

The SQL statements are categorized into different categories based upon the purpose. They are;

i) Data Definition Language (DDL) statement

ii) Data Manipulation Language (DML) statement

iii) Transaction Control Statement

iv) Session Control Statement

v) System Control Statement

vi) Embedded SQL Statement

Out of these six, we will be studying only the first two types in this course.

Data Definition Language (DDL) Statements

Data Definition Language (DDL) or Data Description Language (DDL) is a standard for commands that defines the different structures in a database. DDL statements are used to create structure of a table, modify the existing structure of the table and remove the existing table. Some of the DDL statements are CREATE TABLE, ALTER TABLE and DROP TABLE.

Data Manipulation Language (DML) Statements

Data Manipulation Language (DML) statements are used to access and manipulate data in existing tables. The manipulation includes inserting data into tables, deleting data from the tables, retrieving data and modifying the existing data. The common DML statements are SELECT, UPDATE, DELETE and INSERT.

Data Types

Each value manipulated by SQL Database has a data type. The data type of a value associates a fixed set of properties with the value. In SQL there are three main data types: Character, Number, and Date types.

Character

Character data types stores character (alphanumeric) data, which are words and free-form text. They are less restrictive than other data types and consequently have fewer properties. For example, character columns can store all alphanumeric values, but number columns can store only numeric values. Character data types are;

i) CHAR

ii) VARCHAR

iii) VARCHAR2

CHAR: CHAR should be used for storing fix length character strings. String values will be space/blank padded (The adding of meaningless data [usually blanks] to a unit of data to bring it up to some fixed size) before they are stored on the disk. If this type is used to store variable length strings, it will waste a lot of disk space (always allocate fixed memory) . If we declare data type as CHAR, then it will occupy space for

NULL values.

Format: CHAR(n)

Fixed-length character string having maximum length n.

VARCHAR: Varchar is a variable character string. If we declare data type as VARCHAR, then it will occupy space for NULL values. It can have maximum of 2000 characters.

Format: VARCHAR (n)

Variable-length character string having maximum length n.

VARCHAR2: VARCHAR2 is used to store variable length character strings. The string value's length will be stored on disk with the value itself. VARCHAR2 can store up to 4000 bytes of characters. Thus, the difference between VARCHAR and VARCHAR2 is that VARCHAR is ANSI standard but takes up space for variables, whereas the VARCHAR2 is used only in Oracle but makes more efficient use of space.

Format: VARCHAR2 (n)

Example:

CHAR(10) has fixed length, right padded with spaces.

VARCHAR(10) has fixed length, right padded with NULL

VARCHAR2(10) has variable length.

Name char (10): Suppose if we store Name is as "Ravi", then first four places of the ten characters are filled with Ravi and the remaining 6 spaces are also allocated to Name. Thus, the size of name is always ten.

Name varchar (10): Suppose if we store Name is as "Ravi", then first four places of the ten characters are filled with Ravi and the remaining 6 spaces are filled with NULL.

Name varchar2 (10): Suppose if we store Name is as "Ravi", then only first four places are filled with Ravi.

The following table gives possible string data types used in different DBMS

Data type	Access	SQL Server	Oracle	My SQL	Postgre SQL
string (fixed)	N/A	Char	Char	Char	Char
string (variable)	Text (<256) Memo (65k+)	Varchar	Varchar Varchar2	Varchar	Varchar

Numeric data type: Numeric data types are mainly used to store number with or without fraction part. The numeric data types are:

1. NUMBER
2. DECIMAL
3. NUMERIC

4. INT
5. FLOAT

NUMBER: The Number data type stores fixed and floating-point numbers. The Number data type is used to store integers (negative, positive, floating) of up to 38 digits of precision.

The following numbers can be stored in a Number data type column:

- Positive numbers in the range 1 x 10-130 to 9.99...9 x 10125 with up to 38 significant digits.
- Negative numbers from -1 x 10-130 to 9.99...99 x 10125 with up to 38 significant digits.
- Zero

Format:

NUMBER (p, s)

Where;

- 'p' is the precision or the total number of significant decimal digits, where the most significant digit is the left-most nonzero digit and the least significant digit is the right-most known digit.
- 's' is the scale or the number of digits from the decimal point to the least significant digit.

DECIMAL and NUMERIC: Decimal and numeric data types have fixed precision and scale.

Format:

DECIMAL[(p[, s])] and NUMERIC[(p[, s])]

Square brackets ([ ]) are option.

where;

- 'p' is the precision or the total number of significant decimal digits, where the most significant digit is the left-most nonzero digit and the least significant digit is the right-most known digit.
- 's' is the scale or the number of digits from the decimal point to the least significant digit.

INT/INTEGER: The int data type is the integer data type in SQL. This used to store integer number (without any fraction part).

FLOAT: This data type is used to store number with fraction part(real numbers).

The following table gives possible numeric data types used in difference DBMS

Data type	Access	SQL Server	Oracle	My SQL	Postgre SQL
Integer	Number	Int	Number (integer)	Int Integer Numeric	Int Integer
Float	Number (single)	Float Real	Number	Float Decimal Numeric	Numeric

DATE

Date is used to store valid date values, which is ranging from January 1, 4712 BC to December 31, 9999 AD. The date formats are: YYYY-MM-DD or DD/MON/YY or YYYY/MM/DD or MM/DD/YY or DD-MON-YYYY.

Format: DATE

Relational Algebra

An algebra is a combination of set of operands and a set of operators. We can form algebraic expressions by applying operators to operands. Relational algebra consists of a set of operations that take one or two relations as input and produces a new relation as output. Operators map values are taken from the domain and put it into other domain values. If domain is produced from more than one relation, then we get relational algebra.

Operation in relational algebra:

1. Selection
2. Projection
3. Union
4. Cartesian Product

Selection

Selection in relational algebra returns those tuples(records) in a relation that fulfil a condition(Produce table containing subset of rows).

Syntax:

? condition (relation)

Example: The table S (for STUDENT).

Relation: Student

AdNo	Name	Class	Section	Average
101	Anu	12	A	85
105	Balu	12	D	65
203	Leena	11	B	95
205	Madhu	10	B	75
305	Surpreeth	9	C	70
483	Usha	6	A	60

? class=12 (S)

Output:

AdNo	Name	Class	Section	Average
101	Anu	12	A	85
105	Balu	12	D	65

Projection

Projection in relational algebra returns those columns in a relation that given in the attribute list (Produce table containing subset of columns).

Syntax:

π attribute list(relation)

Example:

π Adno,Name (S)

Output:

AdNo	Name
101	Anu
105	Balu
203	Leena
205	Madhu
305	Surpreeth
483	Usha

Union

The union operator is used to combine two or more tables. Each table within the UNION should have the same number of columns, similar data types and also the columns must be in the same order. In the union operation, duplicate records will be automatically removed from the resultant table.

For example:

Table: Student 1

Roll no	Name
11	Kumar
22	Mohan
33	Rohit

Table: Student2

Roll no	Name
22	Mohan
11	Rahul
77	Kavita

Query is

σ (Students 1)

Union

σ (Students 2)

Or

π rollno, Name (Students 1)

Union

π rollno, Name (Students 2)

Resultant table is:

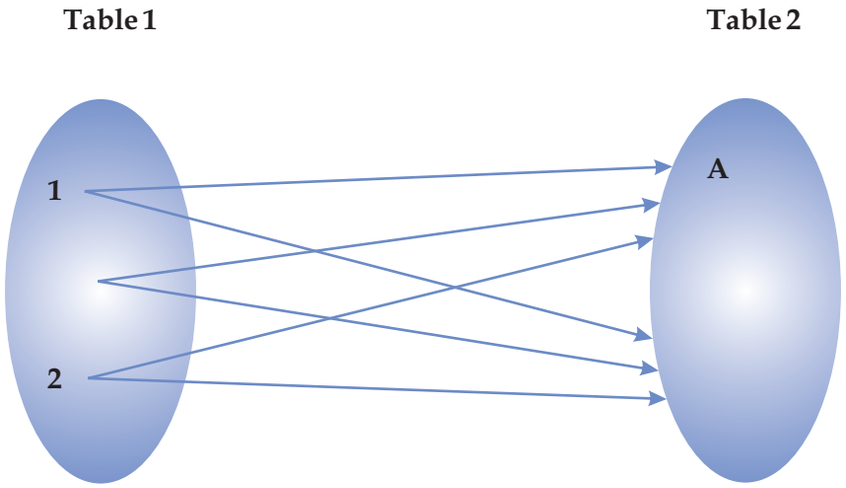
Roll no	Name
11	Kumar
22	Mohan
33	Rohit
11	Rahul
77	Kavita

In the above resultant table, student1 is copied as it is, but in student2, roll no 22 Mohan's information is same as student1. So, that is not copied in the resultant table again. Roll no 11 is same as student1, but name is different. So, that is copied in the resultant table. Roll no 77 is not in student1 table, so that is also copied in the resultant table.

Cartesian product

SQL joins are used to relate information in different tables. It combines fields from two or more tables by comparing values of common columns (join condition). A join condition is a part of the SQL query that retrieves rows from two or more tables. If join condition is ommitted or if it is invalid, then join operation will result in a Cartesian product. Cartesian product is a binary operation and is denoted by (x) Cartesian product returns a number of rows equal to number of rows in the first table multiply by number of rows in

the second table. At the same time, number of columns equal to number of columns in the first table added by number of columns in the second table.



For example:

Table 1: Product

Product_no	Product_name	Price
111	Computer	50000
222	Printer	10000
333	Scanner	12000
444	Modem	500

Table 2: Customer

Cust_no	Cust_name	City	Product_no
101	Kavitha	Delhi	333
201	Mohan	Mumbai	222
301	Rohan	Bangalore	111
401	Sahil	Mumbai	333
501	Rohita	Delhi	444

Query is

σ (Product, customer)

Product_no	Product_name	Price	Cust_no	Cust_name	City	Product_no
111	Computer	50000	101	Kavitha	Delhi	333
111	Computer	50000	201	Mohan	Mumbai	222
111	Computer	50000	301	Rohan	Bangalore	111
111	Computer	50000	401	Sahil	Mumbai	333
111	Computer	50000	501	Rohita	Delhi	444
222	Printer	10000	101	Kavitha	Delhi	333
222	Printer	10000	201	Mohan	Mumbai	222
222	Printer	10000	301	Rohan	Bangalore	111
222	Printer	10000	401	Sahil	Mumbai	333
222	Printer	10000	501	Rohita	Delhi	444
333	Scanner	12000	101	Kavitha	Delhi	333
333	Scanner	12000	201	Mohan	Mumbai	222
333	Scanner	12000	301	Rohan	Bangalore	111
333	Scanner	12000	401	Sahil	Mumbai	333
333	Scanner	12000	501	Rohita	Delhi	444
444	Modem	500	101	Kavitha	Delhi	333
444	Modem	500	201	Mohan	Mumbai	222
444	Modem	500	301	Rohan	Bangalore	111
444	Modem	500	401	Sahil	Mumbai	333
444	Modem	5003	501	Rohita	Delhi	444

Table 1:

Number of rows (cardinality) = 4

Number of columns (degree) = 3

Table 2:

Number of rows (cardinality) = 5

Number of columns (degree) = 4

Cartesian product:

Number of rows (cardinality) = 4*5 = 20

Number of columns (degree) = 3+4 = 7

LET'S REVISE

- **Relation:** In database, a relation means a 'table' (form of rows and columns).
- **Domain:** A domain is the original sets of atomic values used to model data.
- **Tuple:** A row in a table.
- **Attribute:** A column in a table.
- **Primary Key:** A column or set of columns that uniquely identifies a row within a table is called primary key.
- **Candidate Key:** Candidate keys are set of fields (columns with unique values) in the relation that are eligible to act as a primary key.
- **Alternate Key:** Out of the candidate keys, after selecting a key as primary key, the remaining keys are called alternate key.
- **DDL:** Data Definition Language (DDL) or Data Description Language (DDL).
- **DML:** Data Manipulation Language (DML).
- **String datatypes:** CHAR, VARCHAR, VARCHAR2
- **Numeric datatype:** NUMBER, NUMERIC, INT, FLOAT, DECIMAL
- **Date:** DATE
- **Selection:** Selection in relational algebra returns those tuples (records) in a relation that fulfil a condition(Produce table containing subset of rows).
- **Projection:** Projection in relational algebra returns those columns in a relation that given in the attribute list (Produce table containing subset of columns).
- **Union:** The union operator is used to combine two or more tables. In the union operation, duplicate records will be automatically removed from the resultant table.
- **Cartesian product:** SQL joins are used to relate information in different tables. Cartesian product returns a number of rows equal to number of rows in the first table multiply by number of rows in the second table. At the same time, number of columns equal to number of columns in the first table added by number of columns in the second table.

EXERCISE

1. Expand the following:
   (i) SQL (ii) DBMS
2. What is relational database model?
3. What is relation?
4. Define the following:
   a) Cardinality
   b) Degree
   c) Tuple
   d) Field
5. Define the following keys.
   a) Primary key
   b) Candidate key
   c) Alternate key
6. What is DDL?
7. What all character types are possible in sql?
8. What all numeric data types are possible in sql?
9. Write all character data types in SQL.
10. Write all number data types. In SQL
11. Define the following:
    a) Projection
    b) Selection
    c) Union
    d) Cartesian product
12. Differentiate between char and varchar.
13. Write the similarity between decimal and numeric data types.
14. What is the importance of primary key in a table? Explain with suitable example.

15. Differentiate between primary key and candidate key.
16. Differentiate between candidate key and alternate key.
17. What all are domain name possible in gender?
18. Write any four advantages of SQL.
19. In which situation one can apply union operation of two tables.
20. Differentiate between union and Cartesian product.
21. A table 'customer' has 10 columns but no row. Later, 10 new rows are inserted and 3 rows are deleted in the table. What is the degree and cardinality of the table 'customer?
22. A table 'game1' has 3 columns and 20 rows and another table 'game2' has the same column as game1 (ie 3) and 15 rows. 5 rows are common in both the table. If we take union, what is the degree and cardinality of the resultant table?
23. A table 'student1' has 4 columns and 10 rows and 'student2' has 5 columns and 7 rows. If we take Cartesian product of these two tables, what is the degree and cardinality of the resultant table?
24. From the following two tables, write the output of the Cartesian product. Also write primary key of customer and product table.

Customer

Cust. No.	Name	Address	Phone No.
111	Rohan Aggarwal	Delhi	28756389
222	Kanika Jain	Delhi	29807654
333	Keshav Gupta	Mumbai	25678945
444	Dharna	Bambay	24675678

Product

P. No.	P. Name	Price	Qty
101	Computer	35000	3
103	Scanner	20000	2
105	Printer	15000	1

25. In the following two tables, find the union value of employee and emp.

EMPLOYEE

Emp. No.	Name	Salary
1000	Abishek Garg	25000
222	Prachi Goal	30000
1002	Simran Dua	25000
1003	Rishika Pal	40000
1004	Mohit Batra	23000

EMP

Emp. No.	Name	Salary
1002	Simran Dua	25000
1004	Mohit Batra	23000
1007	Sonal Gupta	26000
1009	Rohit Batia	50000

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 BY ….HAVING
- ORDER BY
- UPDATE AND DELETE Command
- ALTER TABLE AND DROP TABLE Command
- EQUI JOIN

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:

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

Constrains

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));
```

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:

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

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

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:

```
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;

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

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

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

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

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.

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 phonenumber. 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?";

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) Equi Join

    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	

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.

a) Display all items information.
b) Display item name and price value.
c) Display soap information.
d) Display the item information whose name starts with letter 's'.
e) Display a report with item number, item name and total price. (total price = price * quantity).
f) Display item table information in ascending order based upon item name.
g) Display item name and price in descending order based upon price.
h) Display item name, whose price is in between 50 to 100.
i) Add new column totalprice with number (10, 2).
j) Increase price value by 100.
k) Fill up totalprice = price * quantity.
l) Remove powder information.
m) Remove totalprice column.
n) Remove whole item structure.

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

a) select sum(price) from item;
b) select avg(price) from item;
c) select min(price) from item;
d) select max(price) from item;
e) select count(price) from item;
f) select distinct price from item;
g) select count(distinct price) from item;

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:

a) To display Iname, price and corresponding Brand name (Bname) of those items, whose price is between 25000 and 30000 both values inclusive).

b) To display ICode, Price and BName of the item, which has IName as "Television".

c) 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	

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.

(i) Display all students' information.

(ii) Display Rohan Saini's information.

(iii) Display number of students in the table.

(iv) Display number of students in each sex.

(v) Display students' information in ascending order using name.

(vi) Display students' information in descending order using average marks.

(vii) Display students' name starting with letter "K".

(viii) Display students' information, whose name ends with "l".

(ix) Display a report with adno,name,average*5 as total marks from student table.

(x) Display students' information, whose average marks are in between 80 to 90.

(xi) Display students' info., who are getting average marks of more than 80 and scode 333.

(xii) Display students' name and average marks, whose scode is 222 and 333.

(xiii) Display sum of average marks.

(xiv) Display maximum average marks

(xv) 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';

(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.