Practical - 4

Aim :- Perform the following:
- Simple Queries
- Simple Queries with Aggregate functions

Description :-
Write and explain the syntax of the following:

1) select statement

Syntax: SELECT column1, column2 FROM table1, table2 WHERE column2 = 'value';

In the above statement; The SELECT clause specifies one or more columns to be retrieved; to specify multiple columns, use a comma and a space between column names.

2) select distinct statement

The SELECT DISTINCT statement is used to return only distinct (different) values.

3) min()

The min() function returns the smallest value of the selected column.

max()

The max() function returns the largest value of the selected column.

sum()
The sum() function sums up all the values in a given column or the values returned by an expression (which could be made up of numbers, column values, or both).

avg()
The avg() function provides the average value of a numeric column.

count()
The count() function uses All keyword. It means that SQL server counts all records in a table.

The syntax:

```
SELECT AVG(column_name) FROM table_name WHERE
                                    condition;

COUNT([ALL | DISTINCT] expression);

SELECT MIN( column_name)
FROM table_name
WHERE condition;

SELECT MAX ( column_name)
FROM table_name
WHERE condition;
```

SELECT SUM(salary) AS "Total Salary" FROM employees WHERE salary > 25000;

Here, we've alised the SUM(salary) expression as "Total Salary". As a result, "Total Salary" will display as the field name when the result set is returned.

Queries:

> create database customer;

> show databases;
Database
information_schema
demo
customer
tycs
mysql
sycs
test

> use customer;
Database changed

> create table customer(CustID, int, Custname varchar(20), contact int, city varchar(20), country varchar(20));

> insert into customer values(1, 'Rohan', 12345 'Mumbai', 'India');

> insert into customer values (2, 'Ankita', 12345, 'Mumbai', 'India');

> insert into customer values (3, 'Preeti', 12345, 'Kolkata', 'India');

> insert into customer values (4, 'Siya', 12345, 'Delhi', 'India');

> insert into customer values (5, 'Nisha', 12345, 'Hyderabad', 'India');

> select * from customer;

CustID	CustName	contact	city	country
1	Rohan	12345	Mumbai	India
2	Ankita	12345	Mumbai	India
3	Preeti	12345	Kolkata	India
4	Siya	12345	Delhi	India
5	Nisha	12345	Hyderabad	India

* select all values of 'country' in customer Table

> SELECT country FROM customer;

country
India
India
India
India
India

> SELECT DISTINCT country FROM customer;

country
India

> SELECT * from customer WHERE country = 'India';

custID	CustName	contact	city	country
1	Rohan	12345	Mumbai	India
2	Ankita	12345	Mumbai	India
3	Preeti	12345	Kolkata	India
4	Siya	12345	Delhi	India
5	Nisha	12345	Hyderabad	India

> create table Product(ProductID, int, PName varchar(20), UnitPrice decimal);

> insert into product values(1, 'Eraser', 10.00);
> insert into Product values(2, 'Book', 40.00);
> insert into Product values(3, 'Pen', 20.00);
> insert into Product values(4, 'Pencil', 20.00);
> insert into Product values(5, 'Sketch Pen', 50.00);

> select * From Product;

ProductID	PName	UnitPrice
1	Eraser	10.00
2	Book	40.00
3	Pen	20.00
4	Pencil	20.00
5	Sketch Pen	50.00

> SELECT * from Product WHERE UnitPrice = '20';

ProductID	PName	UnitPrice
1	Eraser	10.00

> SELECT * ProductID FROM WHERE ProductID = ANY (SELECT ProductID FROM Product WHERE UnitPrice > 30.00);

ProductID
2
5

> SELECT * ProductID FROM Product WHERE ProductID = ANY (SELECT ProductID FROM Product WHERE UnitPrice < 40);

ProductID
1
3
4

> SELECT * FROM Product WHERE UnitPrice BETWEEN 10 AND 50;

ProductID	PName	UnitPrice
1	Eraser	10.00
2	Book	40.00
3	Pen	20.00
4	Pencil	20.00
5	Sketch Pen	50.00

> SELECT * FROM Product WHERE PName LIKE 'P%';

ProductID	PName	UnitPrice
3	Pen	20.00
4	Pencil	20.00

```
> SELECT MAIN(UnitPrice)As smallestPrice FROM Product;
Smallest Price
 10.00

> SELECT MAX(UnitPrice)As ExpensivePrice FROM Product;
ExpensivePrice
 50.00

> SELECT AVG(UnitPrice) FROM Product;
AVG(UnitPrice)
  28

> create database orderDetails;
> use orderDetails;
Database changed

> create table orderDetails(orderDetailsID int,
ORDERID varchar(20), PName varchar(20),
Quantity int);
Query OK, 0 rows affected

> insert into orderDetails values(12345, 'PQRST',
                                  'Bag', 4);
```

```
> insert into orderDetails values(45618, 'STRWP',
                                  'Sofa', 1);

> insert into orderDetails values(56181, 'UVWXZ',
                                  'Bed', 1);

> insert into orderDetails values(78123, 'BQRSD',
                                  'Chair', 1);

> insert into orderDetails values(86123, 'PQRWD',
                                  'Helmet', 1);

> SELECT SUM(QUANTITY) AS sum_quantity
  FROM orderDetails;
  sum_quantity
     8
```

Practical - 5

Aim :-
Queries involving
- Date Functions (now(), curdate(), curtime(), date(), adddate(), datediff(), day(), month(), year(), hour(), min(), sec(), sysdate())
- String Functions (concat(), left(), right(), length(), lcase()/lower(), ucase()/upper(), replace(), trim(), ltrim(), rtrim(), instr(), mid(), strcmp())
- Math Functions (abs(), ceil(), floor(), mod(), pow(), sqrt(), round(), truncate())

Description :-
- Date Functions

1) now() - Return the current date and time
Syntax :- SELECT NOW();
2) curdate() - Returns the current date
Syntax - SELECT CURDATE();
3) curtime() - Returns the current time
Syntax - SELECT CURTIME();
4) date() - Extracts the data part of a date at date / time
Syntax - SELECT Name, DATE (BirthTime) As BirthDate FROM Test
5) adddate() - Adds a time / date interval to a date and then returns the date
Syntax - ADDDATE (date, days)
6) datediff() - The DATEDIFF() returns no. of days between two dates.

- String Functions

1) concat() - It concatenates two or more strings
Syntax :- SELECT CONCAT (string1);
2) LEFT() - Returns specified number of character FROM str
Syntax :- SELECT (string, length);
3) LENGTH() - Returns a length of a string
Syntax :- LEN (string);
4) rtrim() - Removes spaces from a string
Syntax :- RTRIM (string);
5) strcmp() - Function compares two strings
Syntax :- STRCMP (string1, string2);

- Math Functions

1) abs() - Returns the absolute value of a number
Syntax :- ABS (number);
2) Floor() - Returns the largest integer value that is smallest than or equal to a numbers.
Syntax :- FLOOR (number);
3) mod() - Function returns the remainder of n divided by another number.
Syntax :- MOD (x, y);
4) sqrt() - Function returns the square root of a number.
Syntax :- sqrt (x);
5) pow() - Function returns the value of a number raised to the power of another number.

```
> SELECT NOW();
2023-02-05 04:07:24
> SELECT CURDATE();
2023-02-15
> SELECT CURTIME();
04:12:46
> SELECT GETDATE();
2023-02-15 04:15:29.773
> SELECT ADDDATE("2017-06-15", INTEVAL 10DAY);
2017-06-25
> SELECT DATEDIFF(year, '2017/08/25', '2011/08/25')
                        AS DA Datediff;
-6
> SELECT DAY('2017/08/25') AS Day of Month;
25
> SELECT MONTH('2017/08/25') AS Month;
 Month
   8
> SELECT YEAR('2017/08/25') AS Year;
 Year
 2017
> SELECT HOUR("2017/06/20 09:34:00");
HOUR("2017/06/20 09:34:00")
 9
> SELECT MINUTE("2017-06-20 09:34:00");
MINUTE("2017-06-20 09:34:00")
 34
> SELECT SECOND("2017-06-20 09:34:00.00023");
```

```
SECOND("2017-06-20 09:34:000023")
0
> SELECT SYSDATE();
SYSDATE()
2023-02-16 03:34:08
> SELECT CONCAT('Yahoo.com');
Yahoo.com
> SELECT LEFT('SQL Tutorial', 3) AS ExtractString;
ExtractString
SQL
> SELECT RIGHT('SQL Tutorial', 3) AS ExtractString;
ExtractString

> SELECT LEN('W3Schools.com');
13
> SELECT LOWER('SQL Tutorial is Fun!');
sql tutorial is fun
> SELECT UCASE('SQL Tutorial is Fun!') AS
                    UppercaseText;
UppercasText
SQL TUTORIAL IS FUN!
> SELECT UPPER('SQL Tutorial is Fun!');
SQL TUTORIAL IS FUN!
> SELECT REPLACE('SQL Tutorial is Fun!');
> SELECT REPLACE('SQL Tutorial', 'Tutorial', 'command');
SQL command
> SELECT TRIM('SQL Tutorial') AS Trimmed String;
Trimmed String
SQL Tutorial
```

```
> SELECT TRIM ('SQL Tutorial') AS Trimmed String;
Trimmed String
 SQL Tutorial
> SELECT RTRIM ('SQL Tutorial') AS Right TrimmedString;
Right Trimmed String;
SQL Tutorial
> SELECT LTRIM ('SQL Tutorial') AS Left TrimmedString;
Left Trimmed String;
SQL Tutorial
> SELECT INSTR("W3Schools.com", "3") AS MatchPosition;
MatchPosition
 2
> SELECT MID ("SQL Tutorial", 5, 3) AS ExtractString;
ExtractString
 Tut
> SELECT STRCMP("SQL Tutorial", "SQL Tutorial");
 0
> SELECT Abs (-243.5) AS AbsNum;
AbsNum
243.5
> SELECT CEIL (25.75);
CEIL (25.75)
26
> SELECT FLOOR(25.75) AS FLOORValue;
FLOORValue
 25
> SELECT MOD (18, 4);
MOD (18, 4)
 2
```

```
> SELECT POW (8, 3);
POW(8, 2)
512
> SELECT SQRT (64);
8.0
> SELECT ROUND (235.415, 2) AS RoundValue;
RoundValue
235.420
> SELECT TRUNCATE (135.375, 2);
TRUNCATE (135.375, 2)
135.37
```

Practical - 6

Aim :- Join Queries
- Inner join
- Outer join

Description :-
Write and explain all the syntax of queries used in the practical.

1. Inner Join

→ Returns records that have matching values in both tables.

Syntax :-
```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name =
table2.column_name;
```

2. Left Join

→ Returns all records from the left table and the matched records from the right table.

Syntax :-
```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name;
```

3. Right Join

→ Returns all records from the right table and the matched records from the table.

Syntax :- SELECT column_name(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name =
table2.column_name;

4. Cross Join

→ CROSS JOINs are used to combine each row of one table with each row of another table, and return the Cartesian product of the sets of rows from the tables that are joined.

Syntax :- SELECT column_name(s)
FROM table1
CROSS JOIN table2;

Queries:

> select * from Employee;

EmpID	EmpName	Age	EmailID	PhoneNo	Address
1	Vardhan	22	vardy@abc.com	12345	Delhi
2	Himani	32	himani@abc.com	12345	Mumbai
3	Aayushi	24	Aayushi@abc.com	12345	Kolkata
4	Hemanth	25	Hemanth@abc.com	12345	Bengaluru
5	Swatee	26	Swatee@abc.com	12345	Hyderabad

> select * from Project;

ProjectID	EmpID	ClientID	ProjectName	ProjectStartDate
111	1	3	Project1	2019-04-21
222	2	1	Project2	2019-02-12
333	3	5	Project3	2019-01-10
444	4	2	Project4	2019-04-16
555	5	4	Project5	2019-05-23

> select * from Client;

CliID	CliName	Age	CliEmailID	PhoneNo	Address	EmpID
1	Susan	30	susan@abc.com	61890	Kolkata	1
2	Mois	21	mois@abc.com	61890	Kolkata	2
3	Soma	22	soma@abc.com	61890	Delhi	3
4	Zainab	40	zainab@abc.com	61890	Hyderabad	4
5	Bhaskar	32	bhaskar@abc.com	61890	Mumbai	5

INNER JOIN

Query:

```
SELECT Employee.EmpID, Employee.EmpName, Project.ProjectID
Project.ProjectName
FROM Employee
INNER JOIN Project
ON Employee.EmpID = Project.EmpID;
```

EmpID	EmpName	ProjectID	ProjectName
1	Vardhan	111	Project1
2	Himani	222	Project2
3	Aayushi	333	Project3
4	Hemanth	444	Project4
5	Swatee	555	Project5

Join Three Tables:

```
SELECT Employee.EmpID, Project.ProjectName, Client.CliName
FROM ((Employee
INNER JOIN Project ON Employee.EmpID = Project.EmpID)
INNER JOIN Client ON Employee.EmpID = Client.EmpID);
```

EmpID	ProjectName	CliName
1	Project1	Susan
2	Project2	Mois
3	Project3	Soma
4	Project4	Zainab
5	Project5	Bhaskar

LEFT JOIN

Queries:

```
SELECT e.EmpName, p.ProjectID, p.ProjectName
FROM Employee AS e
LEFT JOIN Project AS p
ON p.EmpID = Employee.EmpID;
```

EmpName	ProjectID	ProjectName
Vardhan	111	Project1
Himani	222	Project2
Aayushi	333	Project3
Hemanth	444	Project4
Swatee	555	Project5

RIGHT JOIN

Queries:

```
SELECT e.EmpName, p.ProjectID, p.ProjectName
FROM Employee AS e
RIGHT JOIN Project AS p
ON e.EmpID = p.EmpID;
```

EmpName	ProjectID	ProjectName
Vardhan	111	Project1
Himani	222	Project2
Aayushi	333	Project3
Hemanth	444	Project4
Swatee	555	Project5

```
SELECT Employee.EmpName, Project.ProjectID, Project.
                                        ProjectName
FROM Employee.EmpName, Project.ProjectIDProjectID
RIGHT JOIN Project
ON Project.EmpID = Employee.EmpID;
```

EmpName	ProjectID	ProjectName
Vardhan	111	Project1
Himani	222	Project2
Aayushi	333	Project3
Hemanth	444	Project4
Swatee	555	Project5

CROSS JOIN

Queries:

```
select Employee.EmpID, Employee.EmpName, Project.
ProjectID, Project.ProjectName
from Employee
CROSS JOIN Project;
```

EmpID	EmpName	ProjectID	ProjectName
5	Swatee	111	Project1
4	Hemanth	111	Project1
3	Aayushi	111	Project1
2	Himani	111	Project1
1	Vardhan	111	Project1
5	Swatee	222	Project2
4	Hemanth	222	Project2
3	Aayushi	222	Project2
2	Himani	222	Project2
1	Vardhan	222	Project2
5	Swatee	333	Project3
4	Hemanth	333	Project3
3	Aayushi	333	Project3
2	Himani	333	Project3
1	Vardhan	333	Project3

5	Swatee	444	Project4
4	Hemanth	444	Project4
3	Aayushi	444	Project4
2	Himani	444	Project4
1	Vardhan	444	Project4
5	Swatee	555	Project5
4	Hemanth	555	Project5
3	Aayushi	555	Project5
2	Himani	555	Project5
1	Vardhan	555	Project5

Practical - 9

Aim:- Views
- Creating Views (with and without check option)
- Dropping views
- Selecting from a view

Description:-

1. What is the view?

→ A database view is a subset of a database and is based on a query that runs on one or more database tables.

2. Write and explain all the syntax used in the practical.

Explanation:-

View: A database view is a virtual table that is derived from a physical/real table.

Advantages of view:

1. Simplifies the complex queries.
2. Limits Data Access to specific users
3. Provides extra layer of security
4. It enables computed columns

Disadvantages of view:

1. Performance decreases
2. Dependency on table