Godavari Foundation's

Godavari College of Engineering, Jalgaon

Department of Computer

Lab Manual

Database System Laboratory

Practical No:- ____

Date:-_____

Name of Student:- _____

Class:-_____ **Roll No:-_____**

Title:

_____

_____

_____

_____

Aim: -

_____

_____

_____

_____

_____

_____

Software Requirement: _____

Hardware Requirement:- _____

Theory:-

SUB QUERIES

A Subquery or Inner query or a Nested query is a query within another SQL query and embedded within the WHERE clause.

Subqueries with the SELECT Statement

Syntax:-

```
SELECT column_name [, column_name ]
FROM   table1 [, table2 ]
WHERE  column_name OPERATOR
(SELECT column_name [, column_name ]
FROM table1 [, table2 ]
[WHERE])
```

Example:-

```
SQL> SELECT *
FROM CUSTOMERS
WHERE ID IN (SELECT ID
FROM CUSTOMERS
WHERE SALARY > 4500) ;
```

Subqueries with the INSERT Statement

Syntax:-

```
INSERT INTO table_name [ (column1 [, column2 ]) ]
SELECT [ *|column1 [, column2 ]
FROM table1 [, table2 ]
[ WHERE VALUE OPERATOR ]
```

Example:-

```
SQL> INSERT INTO CUSTOMERS_BKP
SELECT * FROM CUSTOMERS
WHERE ID IN (SELECT ID
FROM CUSTOMERS) ;
```

Subqueries with the UPDATE Statement

Syntax:-

```
UPDATE table
SET column_name = new_value
[ WHERE OPERATOR [ VALUE ]
(SELECT COLUMN_NAME
FROM TABLE_NAME)
[ WHERE) ]
```

Example:-

```
SQL> UPDATE CUSTOMERS
SET SALARY = SALARY * 0.25
WHERE AGE IN (SELECT AGE FROM CUSTOMERS_BKP
WHERE AGE >= 27 );
```

Subqueries with the DELETE Statement

Syntax:-

```
DELETE FROM TABLE_NAME
[ WHERE OPERATOR [ VALUE ]
(SELECT COLUMN_NAME
FROM TABLE_NAME)
[ WHERE) ]
```

Example:-

```
SQL> DELETE FROM CUSTOMERS
WHERE AGE IN (SELECT AGE FROM CUSTOMERS_BKP
WHERE AGE >= 27 );
```

SET OPERATORS

Set operators are used to join the results of two (or more) SELECT statements.

UNION

When multiple SELECT queries are joined using UNION operator, it displays the combined result from all the compounded SELECT queries,after removing all duplicates and in sorted order (ascending by default), without ignoring the NULL values.

Example:-

```
SELECT 1 NUM FROM DUAL
UNION
SELECT 5 FROM DUAL
UNION
SELECT 3 FROM DUAL
UNION
SELECT 6 FROM DUAL
UNION
SELECT 3 FROM DUAL;
```

Output:-

NUM
1
3
5
6

UNION ALL

UNION ALL gives the result set without removing duplication and sorting the data.

Example:-

```
SELECT 1 NUM FROM DUAL
UNION ALL
SELECT 5 FROM DUAL
UNION ALL
SELECT 3 FROM DUAL
UNION ALL
SELECT 6 FROM DUAL
UNION ALL
SELECT 3 FROM DUAL;
```

Output:-

NUM
1
5
3
6
3

INTERSECT

Using INTERSECT operator, Oracle displays the common rows from both the SELECT statements, with no duplicates and data arranged in sorted order (ascending by default).

Example:-

```
SELECT SALARY
FROM employees
WHERE DEPARTMENT_ID = 10
INTRESECT
SELECT SALARY
FROM employees WHERE DEPARTMENT_ID = 20
```

MINUS

Minus operator displays the rows which are present in the first query but absent in the second query, with no duplicates and data arranged in ascending order by default.

Example:-

```
SELECT SALARY
FROM employees
WHERE DEPARTMENT_ID = 10
MINUS
SELECT SALARY
FROM employees
WHERE DEPARTMENT_ID = 20
```

JOINS

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

Different Types of SQL JOINs:-

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table

Inner JOIN (Simple Join)

It is used to return all rows from multiple tables where the join condition is satisfied. It is the most common type of join.

Syntax:-

```
SELECT columns
FROM table1
INNER JOIN table2
ON table1.column = table2.column;
```

Example:-

```
SELECT officers.officer_name, officers.address, students.course_name
FROM officers
INNER JOIN students
ON officers.officer_id = students.student_id;
```

Left Outer Join

The LEFT OUTER JOIN returns all rows from the left hand table specified in the ON condition and only those rows from the other table where the join condition is fulfilled.

Syntax:-

```
SELECT columns
FROM table1
LEFT[OUTER] JOIN table2
ON table1.column = table2.column;
```

Example:-

```
SELECT officers.officer_name, officers.address, students.course_name
FROM officers
LEFT JOIN students
ON officers.officer_id = students.student_id;
```

Right Outer Join

The MySQL Right Outer Join returns all rows from the RIGHT-hand table specified in the ON condition and only those rows from the other table where the join condition is fulfilled.

Syntax:-

```
SELECT columns
FROM table1
RIGHT [OUTER] JOIN table2
ON table1.column = table2.column;
```

Example:-

```
SELECT officers.officer_name, officers.address, students.course_name, students.student_name
FROM officers
RIGHT JOIN students
ON officers.officer_id = students.student_id;
```

MYSQL QUERIES

SUB QUERIES:-

use student;

IN

```
select Pass_Name from Reservation
where Flightno IN
(select Flightno from Flight_Sch where Flight_day1=1);

select Pnr, Flightno, FlightDate, Reserv_Date, Total_Fare, Branch_Code
from Reservation
where Branch_Code IN
(select Branch_Code from Branch where City='Jalgaon');
```

NOT IN

```
select Pass_Name from Reservation
where Flightno NOT IN
(select Flightno from Flight_Sch where Flight_day1=1);

select Pnr, Flightno, FlightDate, Reserv_Date, Total_Fare, Branch_Code
from Reservation
where Branch_Code NOT IN
(select Branch_Code from Branch where City='Jalgaon');
```

EXISTS

```
select * from Flight
where EXISTS
(select * from Reservation
where Reservation.Flightno=Flight.Flightno);
select * from Flight;
select Flightno from Reservation;
```

NOT EXISTS

```
select * from Flight
where NOT EXISTS
( select * from Reservation
where Reservation.Flightno=Flight.Flightno);
```

ANY

```
select Pass_Name, Total_Fare
from Reservation
where Total_Fare > ANY
(select Total_Fare
from Reservation
where Branch_Code='SBI');
```

ALL

```
select Pass_Name, Total_Fare
from Reservation
where Total_Fare > ALL
(select Total_Fare
from Reservation
where Branch_Code='SBI');
```

SET OPERATIONS QUERIES:-

```
use student;
```

UNION

```
select Flightno, Flight_Date
from Flight
UNION
select Flightno, Flight_Date
from Reservation

select * from Flight;
select * from Reservation;
```

UNION ALL

```
select Flightno, Flight_Date
from Flight
UNION ALL
select Flightno, Flight_Date
from Reservation

select * from Flight;
select * from Reservation;
```

INTERSECT

```
select Flightno, Flight_Date
from Flight
INTERSECT
select Flightno, Flight_Date
from Reservation;

select * from Flight;
select * from Reservation;
```

ALIAS QUERY for INTERSECT in MYSQL:-

```
select Flight.Flightno
from Flight
where Flight.Flightno IN
(select Reservation.Flightno from Reservation);
```

MINUS

```
select Flightno, Flight_Date
from Flight
MINUS
select Flightno, Flight_Date
from Reservation;

select * from Flight;
select * from Reservation;
```

ALIAS QUERY for MINUS in MYSQL:-

```
select Flightno
from Flight
LEFT JOIN Reservation
USING(Flightno)
where Reservation.Flightno IS NULL;
```

JOINS QUERIES:-

```
use student;
```

INNER JOIN

```
select Flight.Flightno, Flight_Date, Airbusno, Deprt_time
from Flight
INNER JOIN Flight_Sch
```

```
ON Flight.Flightno=Flight_Sch.Flightno;

select * from Flight;
select * from Flight_Sch;
```

LEFT OUTER JOIN

```
select Flight.Flightno, Flight_Date, Airbusno, Deprt_time
from Flight
LEFT OUTER JOIN Flight_Sch
ON Flight.Flightno=Flight_Sch.Flightno;

select * from Flight;
select * from Flight_Sch;
```

LEFT JOIN

```
select Flight.Flightno, Flight_Date, Airbusno, Deprt_time
from Flight
LEFT JOIN Flight_Sch
ON Flight.Flightno=Flight_Sch.Flightno;

select * from Flight;
select * from Flight_Sch;
```

RIGHT OUTER JOIN

```
select Flight.Flightno, Flight_Date, Airbusno, Deprt_time
from Flight
RIGHT OUTER JOIN Flight_Sch
ON Flight.Flightno=Flight_Sch.Flightno;

select * from Flight;
select * from Flight_Sch;
```

RIGHT JOIN

```
select Flight.Flightno, Flight_Date, Airbusno, Deprt_time
from Flight
RIGHT JOIN Flight_Sch
ON Flight.Flightno=Flight_Sch.Flightno;

select * from Flight;
select * from Flight_Sch;
```

FULL OUTER JOIN

```
select Flight.Flightno, Flight_Date, Airbusno, Deprt_time
from Flight
FULL OUTER JOIN Flight_Sch
ON Flight.Flightno=Flight_Sch.Flightno;

select * from Flight;
select * from Flight_Sch;
```

FULL JOIN

```
select Flight.Flightno, Flight_Date, Airbusno, Deprt_time
from Flight
FULL JOIN Flight_Sch
ON Flight.Flightno=Flight_Sch.Flightno;

select * from Flight;
select * from Flight_Sch;
```

EQUI JOIN

```
select x.Route_Code, x.First_Fare, y.First_Fare
from Fare x, Fare y
where x.First_Fare=y.First_Fare;

select x.Route_Code, x.First_Fare, y.First_Fare
from Fare x, Fare y
where x.First_Fare < y.First_Fare and y.Route_Code='AUR-JAL';
```

CROSS JOIN

```
select x.Flightno, x.Flight_Date, y.Route_Code
from Flight x
CROSS JOIN Flight_Sch y;

select * from Flight;
select * from Flight_Sch;
```

Conclusion:-

_____
_____
_____
_____