Task 5 - writing join queries, recursive queries, and views

RECURSIVE QUERIES:

Title: Implementation of different types of joins & recursive query

- A SQL join combines records from two tables.
- A join locates related column values in the two tables.
- A query can contain zero, one & multiple join operations.
- INNER join is the same as join, the keyword

Objective

To implement different types of joins & recursive queries.

Theory:

These joins clause is used to combine records from two or more tables in a database. The join is actually performed by the where clause where combines specified row of tables

Syntax:

SELECT column1, column2, column3 ... from table_name1, table_name2 WHERE table_name1.column_name = table_name1.column_name.

Types of joins:

1. Simple join.
2. Self join.
3. outer join

Simple join:

It is the most common type of join. it retrieves the row from 2 tables having a common column & is further classified into.

In the above statement, item-id = cust.id perform for the join statement. It combines the matched row of tables.

It can be used as follows:

- To Insert records in the target table.
- To update records in the target table.
- To create views.

Non Equi - join:

It specifies the relationship btw colums belonging to different table by making use of relation operators the query.

Example

select * from emp x, empy where x.salary >= (select avg(salary) from x.emp where x.dept no = y.deptno);

Outer join:

It extends the result of a simple join. An outer join returns all the rows returned by simple join as well as those rows from one table. The symbol (+) represents outer join.

Different type of SQ2 JOINS

Here are the different types of the joins in SQL:

(INNER JOIN): Return records that have match

SELECT column_name(s) from table1 INNER JOIN table2 ON table1.column_name = table2.column_name.

(left outer JOIN): Return all records from the left table, & the matched records from the right table

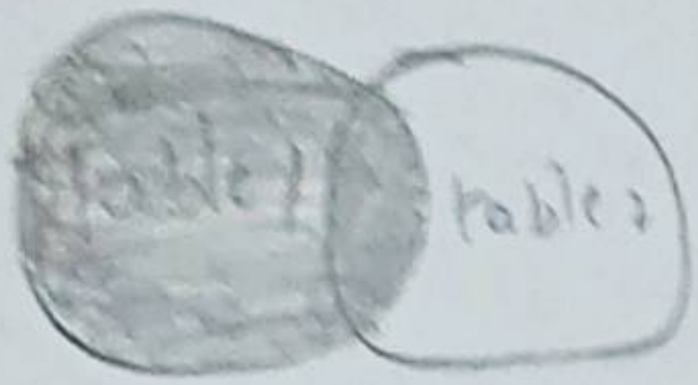
table1.column-name = table2.column-name;

or (outer) JOIN: Return all record from the
ht table, & the matched records from ad
ft table.

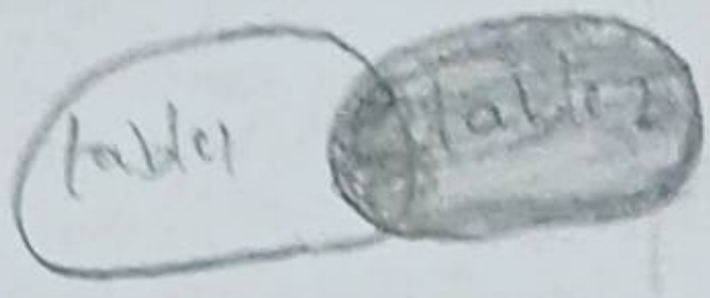
ll (outer) JOIN: Return all records where
re is a match in either left or right table
t column-name(s) from table1

ll outer JOIN table2 ON table1.column-name =
able2.column-name;

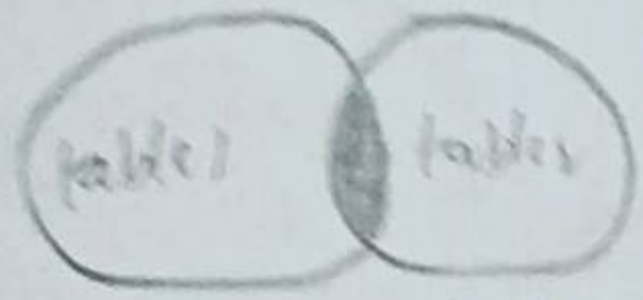
LEFT JOIN



RIGHT JOIN



INNER JOIN



FULL OUTER JOIN

Consider the following two tables - member &
borrowed.

INNER Join Query

Select borrowed.membno, member.name from member
Inner join borrowed.
on member.membno = borrowed.membno;

LEFT Join Query

SELECT member.name, borrowed.membno from
member LEFT join borrowed ON borrowed.
mem.bno = member.membno;

SQL RIGHT join keyword

Select number_name, borrowed.number_name from number
Right join borrowed on borrowed.number_name = number.number_name;

SQL FULL OUTER JOIN keyword

SELECT number_name, borrowed.
FULL OUTER JOIN/FULL JOIN on
Select number_name, borrowed.number_name from number
full join borrowed on borrowed.number_name =
number.number_name;

Recursive Queries

Syntax
WITH RECURSIVE [cte_name)(column,...)
AS ([non-recursive term]
UNION ALL
[recursive term])
SELECT... FROM [cte_name];

write a recursive query to create a multiplication table for 2

WITH RECURSIVE x2(result) AS (
SELECT 1
UNION ALL
SELECT result+2 from x2
where result < from x2 limit 10;
)
SELECT * from x2;

output

result
1
2
4
8
16
32

key
128
256
512

(10 rows)

fibonacci sequence.

```
with Recursive fb(f1, f2) AS (
    select 0, 1
union All
select f2, (f1+f2) from fib)
select f2, (f1+f2) from fb)
select f1 from fib limit 10;
```

f1
0
1
2
3
5
8
13
21
34
16 rows)

28/8/25

Result: The implementation of SQL commands using joins & recursive querying are executed successfully.

Task-5 Writing Joins Queries, Equivalent Avi/Or Recursive Queries.

```
&plus: Release 11.2.0.2.0 Production on Thu Sep 18 14:29:37 2025

pyright (c) 1982, 2014, Oracle.  All rights reserved.

&> connect
ter user-name: system
ter password:
nnected.
&> create table member78 (memno number(5), name varchar(10));

ble created.

&> desc member78;
ame                                       Null?    Type
----------------------------------------- -------- ----------------------------
EMNO                                      NUMBER(5)
AME                                       VARCHAR2(10)

&> insert into member78 values (1,'alice');

w created.

&> insert into member78 values (2,'bob');

w created.

&> insert into member78 values (3,'charlie');

w created.

&> insert into member78 values (4,'david');

w created.

&> select* from member78;

    MEMNO NAME
---------- ----------
         1 alice
         2 bob
         3 charlie
         4 david

> create table borrowed78 (memno number(5), book_id number(5));

e created.
```

Name	Null?	Type
MEMNO		NUMBER(5)
BOOK_ID		NUMBER(5)

```
SQL> insert into borrowed78 values (2, B101);
insert into borrowed78 values (2, B101)
                               *
ERROR at line 1:
ORA-00984: column not allowed here


SQL> insert into borrowed78 values (2, 101);

1 row created.

SQL> insert into borrowed78 values (3, 102);

1 row created.

SQL> insert into borrowed78 values (4, 103);

1 row created.

SQL> select* from borrowed78;

    MEMNO    BOOK_ID
--------- ----------
        2        101
        3        102
        4        103
```

```
SQL> select borrowed78.memno, member78.NAME FROM member78 INNER JOIN borrowed78 ON member78.memno = borrowed78.memno

     MEMNO NAME
---------- ----------
         2 bob
         3 charlie
         4 david

SQL> select member78.NAME, borrowed78.memno FROM member78 LEFT JOIN borrowed78 ON borrowed78.memno = member78.memno

NAME            MEMNO
---------- ----------
bob                 2
charlie             3
david               4
alice

SQL> select member78.NAME, borrowed78.memno FROM member78 RIGHT JOIN borrowed78 ON borrowed78.memno = member78.memno;

NAME            MEMNO
---------- ----------
bob                 2
charlie             3
david               4

SQL> select member78.NAME, borrowed78.memno FROM member78 FULL JOIN borrowed78 ON borrowed78.memno = member78.memno;

NAME            MEMNO
---------- ----------
alice
bob                 2
charlie             3
david               4

SQL>
```

EX NO.	5
PERFORMANCE (5)	5
RESULT AND ANALYS'S (5)	5
VIVA VOCE (5)	5
RECORD (5)	
TOTAL (25)	

Result: Thus, the writting join Queries, Equivalent AND(OR) Recursive Queries.