ALL ABOUT SQL JOINS – BREAD & BUTTER FOR DATA ANALYSTS

```
create database joins_practice;
use joins_practice;

create table test1(
 a int,
 b int
 );

insert into test1 values(10, 1);
insert into test1 values(10, 2);
insert into test1 values(20, 3);
insert into test1 values(30, 5);
insert into test1 values(30, 6);
insert into test1 values(40,41);
insert into test1 values(50, 7);


create table test2( c int, d int);

insert into test2 values(1000, 100);
insert into test2 values(50, 101);
insert into test2 values(30, 102);
insert into test2 values(40, 103);
insert into test2 values(40, 104);
insert into test2 values(30, 105);

select * from test1;
select * from test2;
```

ALL ABOUT SQL JOINS – BREAD & BUTTER FOR DATA ANALYSTS

```
-- (1)inner join / no of record should come 7

select x.a, x.b,y.c,y.d

from test1 as x

inner join test2 as y on x.a = y.c;


-- (2)left join / The LEFT JOIN keyword returns all records from the left table (table1), and the matching records (if any) from the right table (table2). = 10


select x.a, x.b,y.c,y.d

from test1 as x

left join test2 as y on x.a = y.c;


-- (3)right join / The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records (if any) from the left table = 8

select x.a, x.b,y.c,y.d

from test1 as x

right join test2 as y on x.a = y.c;


-- (4) cross join

select x.a, x.b,y.c,y.d

from test1 as x

cross join test2 as y ;


-- (5)full join or full outer join

select x.a, x.b, y.c, y.d

from test1 as x

FULL OUTER JOIN test2 as y on x.a = y.c


## It's important to note that MySQL does not support FULL OUTER JOIN directly.
```

## You can achieve the same result using a combination of a LEFT JOIN and a UNION with a RIGHT JOIN, like this:

```
SELECT x.a, x.b,y.c,y.d
from test1 as x
LEFT JOIN test2 as y on x.a = y.c
UNION
SELECT x.a, x.b,y.c,y.d
from test1 as x
RIGHT JOIN test2 as y on x.a = y.c
```

-- WITH BLANK & NULL--

```
drop table test3;
create table test3(a varchar(10));

insert into test3 values(1);
insert into test3 values(1);
insert into test3 values(2);
insert into test3 values(3);
insert into test3 values("Anand");
insert into test3 values(NULL);
insert into test3 values(NULL);
insert into test3 values(" ");
insert into test3 values(" ");
insert into test3 values("Anand");
insert into test3 values("Satyam");
insert into test3 values("Satyam");
```

```
select a , count(*) as tot_how_many
FROM test3
group by a
order by a ;

SELECT COUNT(a) AS tot_records,COUNT(DISTINCT a) tot_dist_records FROM test3;
select *,ROW_NUMBER() OVER() AS ROW_NUM from test3;
SELECT DISTINCT *  FROM test3;
SELECT DISTINCT a  FROM test3;

## check for uniqueness of the values
SELECT count(a) as tot_records ,
    count(distinct a) as tot_dist_records,
    CASE WHEN count(distinct a)= count(a) THEN 'column values are unique' ELSE 'column values are NOT unique' END as uniqueness
FROM test3;

create table test4(c varchar(10));

insert into test4 values(1);
insert into test4 values(3);
insert into test4 values(5);
insert into test4 values(7);
insert into test4 values("Anand");
insert into test4 values(NULL);
insert into test4 values(" ");
insert into test4 values("Satyam");
insert into test4 values("Satyam");
```

```
select * from test3;

select * from test4;


-- (1)inner join / no of record should come 7

select x.*,y.*

from test3 as x

inner join test4 as y on x.a = y.c;


-- (2)left join / The LEFT JOIN keyword returns all records from the left table (table1), and the matching records (if any) from the right table (table2). = 10


select x.*,y.*

from test3 as x

left outer join test4 as y on x.a = y.c;


-- (3)right join / The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records (if any) from the left table = 8

select x.*,y.*

from test3 as x

right outer join test4 as y on x.a = y.c;


-- (4) cross join

select x.*,y.*

from test3 as x

cross join test4 as y on x.a = y.c;


-- (5)full join or full outer join

select x.*,y.*

from test3 as x
```

FULL OUTER JOIN test4 as y on x.a = y.c;

## It's important to note that MySQL does not support FULL OUTER JOIN directly.

## You can achieve the same result using a combination of a LEFT JOIN and a UNION with a RIGHT JOIN, like this:

```
SELECT x.*,y.*

from test3 as x

LEFT JOIN test4 as y on x.a = y.c

UNION

select x.*,y.*

from test3 as x

RIGHT JOIN test4 as y on x.a = y.c ;
```

--- REAL USE CASE--

```
DROP TABLE emp;

create table emp(

 emp_id int PRIMARY KEY,

 age int,

 emp_name VARCHAR(20),

 Designation VARCHAR(30),

 dept_id INT

 );


insert into emp values(1, 33,"Anand","Analdst",101);

insert into emp values(2, 30,"Naman","Data Scientist",101);

insert into emp values(3,27,"Teja"," ",103);

insert into emp values(4, 26,"Surbhi","Analdst",101);
insert into emp values(5,29,"Akshad",NULL,105);

insert into emp values(6,24,"Prachi"," ",NULL);

insert into emp values(7,37,"Hema","Supervisor",104);

insert into emp values(8, 30,"Sachin"," ",102);


select * from emp;


DROP TABLE dept;

create table dept(

 deptid int PRIMARY KEY,

 dept_name VARCHAR(30),

 dept_head VARCHAR(30)

 );


insert into dept values(101,"Analytics","Vishwanath Patil");

insert into dept values(102,"HR","Suman Mathur");

insert into dept values(103,"Admin","Prakash");

insert into dept values(104,"Testing"," ");

insert into dept values(105,"IT",NULL);


select * from dept;


-- (1)inner join / no of record should come

select e.*, d.*

from emp as e

INNER join dept as d on e.dept_id = d.deptid;


-- (2)left join / The LEFT JOIN kedword returns all records from the left table (table1), and the
matching records (if and) from the right table (table2). = 10
```

select e.*, d.*

from emp as e

LEFT OUTER join dept as d on e.dept_id = d.deptid;

-- (3)right join / The RIGHT JOIN kedword returns all records from the right table (table2), and the matching records (if and) from the left table = 8

select e.*, d.*

from emp as e

RIGHT OUTER join dept as d on e.dept_id = d.deptid;

-- (4) cross join

select e.*, d.*

from emp as e

CROSS join dept as d on e.dept_id = d.deptid;

-- (5)full join or full outer join

select e.a, e.b, d.c, d.d

from emp as e

FULL OUTER JOIN dept as d on e.a = d.c

## It's important to note that MdSQL does not support FULL OUTER JOIN directld.

## So u can achieve the same result using a combination of a LEFT JOIN and a UNION with a RIGHT JOIN, like this:

SELECT e.*, d.*

from emp as e

LEFT JOIN dept as d on e.dept_id = d.deptid

UNION

SELECT e.*, d.*

from emp as e

```
RIGHT JOIN dept as d on  e.dept_id = d.deptid;


-- Create the 'departments' table

CREATE TABLE departments (

   dep_id INT PRIMARY KEY,

   dep_name VARCHAR(50)

);


-- Insert values into the 'departments' table

INSERT INTO departments (dep_id, dep_name) VALUES

   (101, 'HR'),

   (102, 'Marketing'),

   (103, 'Sales'),

   (104, 'IT'),

   (105, '');  -- Insert a blank value



-- Create the 'employees' table with dep_id as VARCHAR

CREATE TABLE employees (

   emp_id INT PRIMARY KEY,

   emp_name VARCHAR(50),

   dep_id VARCHAR(50)

);


-- Insert values into the 'employees' table, including a blank value

INSERT INTO employees (emp_id, emp_name, dep_id) VALUES

   (1, 'Alice', '101'),

   (2, 'Bob', '102'),

   (3, 'Carol', '103'),
   (4, 'Dave', NULL),  -- Insert a blank value

   (5, 'Eve', ''),

   (6, 'Frank', '105');

INSERT INTO employees (emp_id, emp_name, dep_id) VALUES   (7, 'Frank', '101');


SELECT * FROM departments;

SELECT * FROM employees;




-- INNER JOIN:

-- An INNER JOIN returns only the rows where there is a match in both tables. Rows with NULL values or blank values in the joined column are excluded.

SELECT employees.dep_id,emp_id, emp_name, dep_name

FROM employees

INNER JOIN departments

ON employees.dep_id = departments.dep_id;


--LEFT JOIN (or LEFT OUTER JOIN):

--A LEFT JOIN returns all rows from the left table (employees) and the matching rows from the right table (departments). If there is no match in the right table, NULL values are used.

SELECT emp_id, emp_name, dep_name

FROM employees

LEFT JOIN departments

ON employees.dep_id = departments.dep_id;




-- RIGHT JOIN (or RIGHT OUTER JOIN):

-- A RIGHT JOIN returns all rows from the right table (departments) and the matching rows from the left table (employees). If there is no match in the left table, NULL values are used.

SELECT emp_id, emp_name, dep_name
```

```
FROM employees

RIGHT JOIN departments

ON employees.dep_id = departments.dep_id;



-- Cross join

SELECT e.emp_id, e.emp_name, d.dep_id, d.dep_name

FROM employees e

CROSS JOIN departments d;



-- Perform a full outer join between the 'employees' and 'departments' tables

-- It's important to note that MySQL does not support FULL OUTER JOIN directly.

-- You can achieve the same result using a combination of a LEFT JOIN and a UNION with a RIGHT JOIN, like this:


SELECT e.emp_id, e.emp_name, d.dep_id, d.dep_name

FROM employees e

LEFT JOIN departments d

ON e.dep_id = d.dep_id


UNION


SELECT e.emp_id, e.emp_name, d.dep_id, d.dep_name

FROM employees e

RIGHT JOIN departments d

ON e.dep_id = d.dep_id;
```