MySQL - RDBMS

Agenda

- Joins

Case sensitive string comparision

```
SELECT 'SunBeam' = 'SUNBEAM';

SELECT BINARY 'SunBeam' = BINARY 'SUNBEAM';

SELECT BINARY 'SUNBEAM' = BINARY 'SUNBEAM';

SELECT * FROM emp WHERE ename = 'King';
-- case insensitive search

SELECT * FROM emp WHERE BINARY ename = BINARY 'King';
-- case sensitive search -- do not match

SELECT * FROM emp WHERE BINARY ename = BINARY 'KING';
-- case sensitive search -- match
```

Joins

```
USE classwork;

SELECT USER(), DATABASE();
-- sunbeam@localhost, classwork

SOURCE D:/sep21/DAC/dbt/db/joins.sql

SHOW TABLES;

SELECT * FROM emps;

SELECT * FROM depts;

SELECT * FROM addr;

SELECT * FROM meeting;

SELECT * FROM emp_meeting;
```

Cross Joins

```
// for loop
for(int i=0; i<emp.length; i++) {
    Emp e = emp[i];
    for(int j=0; j<dept.length; j++) {
        Dept d = dept[j];
        System.out.println(e.ename + " -- " + d.dname);
    }
}
```

```
// for-each loop
for(Emp e:emp) {
    for(Dept d:dept) {
        System.out.println(e.ename + " -- " + d.dname);
    }
}
```

```
SELECT e.ename, d.dname FROM emps e
CROSS JOIN depts d;

SELECT e.ename, d.dname FROM depts d
CROSS JOIN emps e;
```

```
SELECT e.ename, d.dname FROM emps AS e
CROSS JOIN depts AS d;
-- can use AS keyword for table alias

SELECT emps.ename, depts.dname FROM emps
CROSS JOIN depts;
-- using alias is not mandetory, we can directly use table name

SELECT ename, dname FROM emps
CROSS JOIN depts;
-- if column names are different in both tables, writing alias/tablename is
optional

SELECT ename, dname, deptno FROM emps
CROSS JOIN depts;
-- ERROR: Column 'deptno' in field list is ambiguous.

SELECT ename, dname, depts.deptno FROM emps
CROSS JOIN depts;
```

Inner Join

- Joining two tables (Getting data from two tables based on some condition).

- Obviously the table must be related by some way (some column).
    - One DEPT can have Many EMP.
    - Many EMP can be in One DEPT.
    - This relation is established with "deptno" column in depts and in emps.

```
-- display ename and his dname.
SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno;

-- display ename and names of dept in which he is not working
SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno != d.deptno;
```

Equi-join

- When in any join query condition is of equality, it is referred as "equi-join".

Non-equi-join

- When in any join query condition is of non-equality(<, >, <=, >=, !=), it is referred as "non-equi-join".

Left Outer Join

- Left Join = Intersection + Extra rows from Left table

```
SELECT e.ename, d.dname FROM depts d
LEFT OUTER JOIN emps e ON e.deptno = d.deptno;

SELECT e.ename, d.dname FROM depts d
LEFT JOIN emps e ON e.deptno = d.deptno;
-- OUTER keyword is optional

SELECT e.ename, d.dname FROM  emps e
RIGHT OUTER JOIN depts d ON e.deptno = d.deptno;
```

Right Outer Join

- Right Join = Intersection + Extra rows from Right table

```
SELECT e.ename, d.dname FROM depts d
RIGHT OUTER JOIN emps e ON e.deptno = d.deptno;

SELECT e.ename, d.dname FROM depts d
RIGHT JOIN emps e ON e.deptno = d.deptno;
-- OUTER keyword is optional
```

```
SELECT e.ename, d.dname FROM emps e
LEFT JOIN depts d ON e.deptno = d.deptno;
```

Full Outer Join

- Full Join = Intersection + Extra rows from Left table + Extra rows from Right table
- Full Outer Join is not supported in MySQL. It can work well in Oracle, MS-SQL, ...

```
SELECT e.ename, d.dname FROM emps e
FULL OUTER JOIN depts d ON e.deptno = d.deptno;
```

Set Operators

- Used to combine results of two queries (if output contains same number of columns).

```
(SELECT dname AS name FROM depts)
UNION ALL
(SELECT ename FROM emps);
```

```
(SELECT sal FROM emp)
UNION ALL
(SELECT price FROM books);
```

```
(SELECT e.ename, d.dname FROM emps e
LEFT OUTER JOIN depts d ON e.deptno = d.deptno)
UNION ALL
(SELECT e.ename, d.dname FROM emps e
RIGHT OUTER JOIN depts d ON e.deptno = d.deptno);
```

```
(SELECT e.ename, d.dname FROM emps e
LEFT OUTER JOIN depts d ON e.deptno = d.deptno)
UNION
(SELECT e.ename, d.dname FROM emps e
RIGHT OUTER JOIN depts d ON e.deptno = d.deptno);
-- simulation of full outer join in MySQL
```

Self Join

```
-- print ename and his manager name.
SELECT e.ename, m.ename AS mname FROM emps e
```

```
INNER JOIN emps m ON e.mgr = m.empno;

-- print ename and his manager name.
SELECT e.ename, m.ename AS mname FROM emps e
LEFT JOIN emps m ON e.mgr = m.empno;
```

Joins Practice

```
-- display ename, emp's dname and emp's dist.
SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno;

SELECT e.ename, d.dname, a.dist FROM emps e
LEFT JOIN depts d ON e.deptno = d.deptno
INNER JOIN addr a ON e.empno = a.empno;
```

```
-- display ename and his meeting topics.
SELECT * FROM emps;

SELECT * FROM meeting;

SELECT * FROM emp_meeting;

SELECT e.ename, m.topic FROM emp_meeting em
INNER JOIN emps e ON em.empno = e.empno
INNER JOIN meeting m ON em.meetno = m.meetno;
```

```
-- emps are travelling from their home town to attend few meetings. Display name
of emp and meeting topic and from where he is travelling.

SELECT e.ename, m.topic, a.dist, a.tal
FROM emp_meeting em
INNER JOIN emps e ON em.empno = e.empno
INNER JOIN meeting m ON em.meetno = m.meetno
INNER JOIN addr a ON e.empno = a.empno;

-- emps are representing their depts in few meetings. Display name of emp and
meeting topic and their dept.
SELECT e.ename, m.topic, d.dname
FROM emp_meeting em
INNER JOIN emps e ON em.empno = e.empno
INNER JOIN meeting m ON em.meetno = m.meetno
LEFT JOIN depts d ON e.deptno = d.deptno;
```

```
-- print dname and number (count) of emps in that dept.
SELECT deptno, COUNT(empno) FROM emps
GROUP BY deptno;

SELECT d.dname, COUNT(e.empno) FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno
GROUP BY d.dname;

SELECT d.dname, COUNT(e.empno) FROM emps e
RIGHT JOIN depts d ON e.deptno = d.deptno
GROUP BY d.dname;
```

```
-- display emps and their number of meetings in desc order of meeting count.
SELECT e.ename, m.topic FROM emp_meeting em
INNER JOIN emps e ON em.empno = e.empno
INNER JOIN meeting m ON em.meetno = m.meetno;

SELECT em.empno, COUNT(em.meetno)
FROM emp_meeting em
GROUP BY em.empno;

SELECT e.ename, COUNT(em.meetno)
FROM emp_meeting em
INNER JOIN emps e ON e.empno = em.empno
GROUP BY e.ename;

SELECT e.ename, COUNT(em.meetno)
FROM emp_meeting em
INNER JOIN emps e ON e.empno = em.empno
GROUP BY e.ename
ORDER BY COUNT(em.meetno) DESC;
```

```
-- display all emps in DEV dept.
SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno;

SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno
WHERE d.dname = 'DEV';
```

```
SELECT columns FROM table1
xxx JOIN table2 ON condition
xxx JOIN table3 ON condition ...
WHERE condition
GROUP BY column
HAVING condition
```

```
ORDER BY column
LIMIT n;
```

Non-standard joins

```
-- display ename and dname.
SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d ON e.deptno = d.deptno;

SELECT e.ename, d.dname FROM emps e
JOIN depts d ON e.deptno = d.deptno;
-- by default join is INNER (in MySQL).

SELECT e.ename, d.dname FROM emps e
CROSS JOIN depts d ON e.deptno = d.deptno;
-- In MySQL, we can apply condition on CROSS JOIN

SELECT e.ename, d.dname FROM emps e
CROSS JOIN depts d WHERE e.deptno = d.deptno;
-- You may use WHERE clause with CROSS JOIN
-- However choose INNER JOIN if applicable.

SELECT e.ename, d.dname FROM emps e, depts d
WHERE e.deptno = d.deptno;
-- Join without JOIN keyword is old-style join.

SELECT e.ename, d.dname FROM emps e
INNER JOIN depts d USING (deptno);
-- joined columns from both tables have SAME name
-- the condition can be given using USING keyword
-- USING (colname) --> t1.colname = t2.colname;
-- This is always eqi-join.
-- This works only in MySQL.

SELECT e.ename, d.dname FROM emps e
NATURAL JOIN depts d;
-- num of joined columns = 1 (same name)
-- NATURAL JOIN = Implicit Join Condition
-- Equality Check of Columns with Same Name in Both tables.
-- In this example
--  NATURAL JOIN: ON e.deptno = d.deptno.

-- display all possible depts for Amit & Nilesh.
SELECT e.ename, d.dname FROM emps e
CROSS JOIN depts d WHERE e.ename IN ('AMIT', 'NILESH');
```

Natural Join

- table1: a, b, c, d
- table2: a, b, x, y

- table1 NATURAL JOIN table2 --> ON t1.a = t2.a AND t1.b = t2.b;