Agenda

- SubQuery
- Views

SubQuery

- A query inside another query is called as subquery
- we have two types of subquery
    - 1. single row subquery
        - In this the subquery/inner query returns single row
    - 2. multi rows subquery
        - In this the subquery return multiple rows
        - the output of inner query can be compared using ANY, ALL or IN operator.

Single row Subquery

```
--- display emp with highest salary
SELECT * FROM emp ORDER BY sal DESC LIMIT 1;

SELECT * FROM emp WHERE sal = MAX(sal);
-- cannot use group functions in where clause

SET @maxsal = (SELECT MAX(sal) FROM emp);
SELECT * FROM emp WHERE sal = @maxsal;

SELECT * FROM emp WHERE sal = (SELECT MAX(sal) FROM emp);

-- display emp with second highest sal
SELECT * FROM emp ORDER BY sal DESC LIMIT 1, 1;

SELECT DISTINCT sal FROM emp ORDER BY sal DESC LIMIT 1,1;

SELECT * FROM emp WHERE sal=(SELECT DISTINCT sal FROM emp ORDER BY sal DESC LIMIT
1,1);

-- display emp with second highest sal
SELECT * FROM emp WHERE sal=(SELECT DISTINCT sal FROM emp ORDER BY sal DESC LIMIT
2,1);

-- display emps working in same dept of 'KING'
SELECT * FROM emp WHERE deptno = (SELECT deptno FROM emp WHERE ename = 'KING');

SELECT * FROM emp WHERE deptno = (SELECT deptno FROM emp WHERE ename = 'KING') AND
ename != "KING";
```

MultiRow SubQuery

```
-- find all emps having sal more than all salesman.
SELECT * FROM emp WHERE sal > ALL(SELECT sal FROM emp WHERE job = "SALESMAN");

-- convert the above query to single row subquery
SELECT * FROM emp WHERE sal > (SELECT MAX(sal) FROM emp WHERE job = "SALESMAN");

-- find emp with sal less than sal of "any" emp in deptno=20
SELECT sal FROM emp WHERE deptno = 20;

SELECT * FROM emp WHERE sal < ANY(SELECT sal FROM emp WHERE deptno = 20);

-- convert the above query to single row subquery
SELECT * FROM emp WHERE sal < (SELECT MAX(sal) FROM emp WHERE deptno = 20);

-- display the depts which have emps.
SELECT * FROM depts WHERE deptno = ANY (SELECT deptno FROM emps);
SELECT * FROM depts WHERE deptno IN (SELECT deptno FROM emps);

-- display depts which do not have any emp.
SELECT * FROM depts WHERE deptno != ALL(SELECT deptno FROM emps);
SELECT * FROM depts WHERE deptno NOT IN (SELECT deptno FROM emps);
```

ANY vs IN operator

- ANY can be used in sub-queries only, while IN can be used with/without sub-queries.
- ANY can be used for comparision (<, >, <=, >=, =, or !=), while IN can be used only for equality comparision (=).
- Both operators are logically similar to OR operator.

ANY vs ALL operator

- Both can be used for comparision (<, >, <=, >=, =, or !=).
- Both are usable only with sub-queries.
- ANY is similar to logical OR, while ALL is similar to logical AND.

Corelated SubQuery

- By default inner query is executed for each row of the outer query.
- If no optimization settings are enabled, sub-queries are slower than joins.

```
-- display the depts in which emps exists
SELECT * FROM dept WHERE deptno IN (SELECT deptno FROM emp);
-- 10, ACC --> SELECT deptno FROM emp - 14 rows
-- 20, RES --> SELECT deptno FROM emp - 14 rows
-- 30, SAL --> SELECT deptno FROM emp - 14 rows
-- 40, OPS --> SELECT deptno FROM emp - 14 rows
```

- The sub-query execution can be speed-up if inner queries return/process less number rows.
- This is typically done by using WHERE clause in inner query.
- The WHERE clause in inner query may depend on current row of the outer query. This kind of query is called as "co-related sub-query".

```
SELECT * FROM dept WHERE deptno IN (SELECT DISTINCT deptno FROM emp);
-- 10, ACC --> SELECT DISTINCT deptno FROM emp - 3 rows
-- 20, RES --> SELECT DISTINCT deptno FROM emp - 3 rows
-- 30, SAL --> SELECT DISTINCT deptno FROM emp - 3 rows
-- 40, OPS --> SELECT DISTINCT deptno FROM emp - 3 rows

SELECT * FROM dept d WHERE d.deptno IN (SELECT e.deptno FROM emp e WHERE e.deptno
= d.deptno);
-- 10, ACC --> SELECT deptno FROM emp WHERE deptno=10 - 3 rows
-- 20, RES --> SELECT deptno FROM emp WHERE deptno=20 - 5 rows
-- 30, SAL --> SELECT deptno FROM emp WHERE deptno=30 - 6 rows
-- 40, OPS --> SELECT deptno FROM emp WHERE deptno=40 - 0 rows

SELECT * FROM dept d WHERE d.deptno = (SELECT DISTINCT e.deptno FROM emp e WHERE
e.deptno = d.deptno);
-- 10, ACC --> SELECT DISTINCT deptno FROM emp WHERE deptno=10 - 1 rows
-- 20, RES --> SELECT DISTINCT deptno FROM emp WHERE deptno=20 - 1 rows
-- 30, SAL --> SELECT DISTINCT deptno FROM emp WHERE deptno=30 - 1 rows
-- 40, OPS --> SELECT DISTINCT deptno FROM emp WHERE deptno=40 - 0 rows

SELECT * FROM dept d WHERE EXISTS (SELECT e.deptno FROM emp e WHERE e.deptno =
d.deptno);
-- -- EXISTS check if sub-query return row(s).

-- display depts which do not have any emp.
SELECT * FROM dept d WHERE NOT EXISTS (SELECT e.deptno FROM emp e WHERE e.deptno =
d.deptno);
```

Sub-query in projection

```
-- display number of emps in each dept along with total number of emps.
-- eg -> 10 - 3 - 14
SELECT deptno, COUNT(empno) FROM emp GROUP BY deptno;

SELECT deptno,COUNT(empno) empcount,(SELECT COUNT(empno) FROM emp) totalemp FROM
emp GROUP BY deptno;

SELECT deptno, CONCAT(COUNT(empno),"/",(SELECT COUNT(empno) FROM emp)) count FROM
emp GROUP BY deptno;
```

Subquery in FROM Clause

- Inner-query can be written in FROM clause of SELECT statement.
- The output of the inner query is treated as a table (MUST give table alias) and outer query execute on that table.
- This is called as "Derived table" or "Inline view".

```
-- display empno, ename, sal and category of emp.
-- sal<= 2000 -> POOR ,sal > 2000 -> RICH

SELECT empno, ename, sal, IF(sal<=2000,"POOR","RICH") AS category FROM emp;

-- display no of emps in each category
SELECT category,COUNT(ename) FROM (SELECT empno, ename, sal,
IF(sal<=2000,"POOR","RICH") AS category FROM emp)AS empcategory GROUP BY category;
```

Sub-query in DML

```
-- INSERT new emp with name 'JOHN' and sal=2000 in dept 'OPERATIONS'.
SELECT deptno FROM dept WHERE dname = 'OPERATIONS';

INSERT INTO emp(ename, sal, deptno) VALUES ('JOHN', 2000, (SELECT deptno FROM dept
WHERE dname = 'OPERATIONS'));

-- give comm 100 to all emps in OPERATIONS dept.
UPDATE emp SET comm = 100 WHERE deptno =
(SELECT deptno FROM dept WHERE dname = 'OPERATIONS');

-- delete emps from OPERATIONS dept.
DELETE FROM emp WHERE deptno =
(SELECT deptno FROM dept WHERE dname = 'OPERATIONS');
```

- In MySQL, DML cannot be performed on the table from which inner query is selecting.

```
INSERT INTO emp(empno,ename,sal) VALUES(1000, 'JACK', 6000);
-- delete emp with max sal.
DELETE FROM emp WHERE sal = (SELECT MAX(sal) FROM emp);
```

SQL Performance

- In MySQL, query Performance is measured in terms of query cost. Lower the cost, better is performance.
- The cost of query depends on data in the table(s), MySQL version, server machine config, optimizer settings, etc.

```
SELECT @@optimizer_switch;
EXPLAIN FORMAT = JSON SELECT * FROM dept WHERE deptno IN (SELECT deptno FROM emp);

EXPLAIN FORMAT = JSON SELECT * FROM dept d WHERE d.deptno = (SELECT DISTINCT
e.deptno FROM emp e WHERE e.deptno = d.deptno);
```

Views

- It is just a projection of data.
- CREATE VIEW viewname AS SELECT ...;
- In MySQL views are non-materialized i.e. output of SELECT statement (of view) is not saved in server disk.
- Only SELECT query is saved in server disk.

```
SELECT empno, ename, sal, IF(sal<=2000,"POOR","RICH") AS category FROM emp;

CREATE VIEW v_empcategory AS SELECT empno, ename, sal, IF(sal<=2000,"POOR","RICH")
AS category FROM emp;

SHOW TABLES;
SHOW FULL TABLES;
SELECT * FROM v_empcategory;

SELECT category, COUNT(empno) FROM v_empcategory GROUP BY category;
```

- Their are two types of views
    - 1. Simple
        - DQL + DML
    - 2. Complex
        - DQL

```
CREATE VIEW v_empsal AS SELECT empno, ename, sal FROM emp;
SELECT * FROM v_empsal;
INSERT INTO v_empsal VALUES(1001,"e1",1000);

CREATE VIEW v_empjobsummary AS
SELECT job, SUM(sal) salsum, AVG(sal) salavg, MAX(sal) salmax, MIN(sal) salmin
FROM emp GROUP BY job;

SELECT * FROM v_empjobsummary;

-- we cannot perform DML Operation on v_empjobsummary


CREATE VIEW v_richemp AS SELECT * FROM emp WHERE sal > 2500;
SELECT * FROM v_richemp;
```

```
INSERT INTO v_richemp(empno,ename,sal) VALUES(1002,"e2",2600);
SELECT * FROM v_richemp;

INSERT INTO v_richemp(empno,ename,sal) VALUES(1003,"e3",2200);
SELECT * FROM v_richemp;

SELECT * FROM emp;

ALTER VIEW v_richemp AS SELECT * FROM emp WHERE sal > 2500 WITH CHECK OPTION;

INSERT INTO v_richemp(empno, ename, sal) VALUES(1004, "e4", 2100);
--error check option failed

CREATE VIEW v_richemp2 AS SELECT empno,ename,sal,comm FROM v_richemp;

SELECT * FROM v_richemp2;

SHOW CREATE VIEW v_richemp2;

DROP VIEW v_richemp;

SELECT * FROM v_richemp2;
-- error: invalid table/view.

DROP VIEW v_richemp2;

-- display empno,empname,deptno,deptname frmo emp and dept table
SELECT e.empno, e.ename, d.deptno, d.dname FROM emp e INNER JOIN dept d ON
e.deptno = d.deptno;

-- cretae view v_empdept for above query
CREATE VIEW v_empdept AS
SELECT e.empno, e.ename, d.deptno, d.dname FROM emp e INNER JOIN dept d ON
e.deptno = d.deptno;

SELECT * FROM v_empdept;

-- display all emps in ACCOUNTING dept.
SELECT e.empno, e.ename, d.deptno, d.dname FROM emp e INNER JOIN dept d ON
e.deptno = d.deptno WHERE d.dname = 'ACCOUNTING';

SELECT * FROM v_empdept WHERE dname = 'ACCOUNTING';
```