MYSQL JOINS

by:-

ANKIT BANSAL

AUG 7, 2013.

- JOINS ARE USED TO COMBINE ROWS OF TWO OR MORE DIFFERENT TABLES IN A TEMPORARY TABLES.
- IT IS MORE OPTIMISED THAN THE SUBQUERIES METHOD.

- BASIC SYNTAX OF JOIN.

```
SELECT<column name>
FROM table1
join_type table2
[ON(join_condition)]
```

PERFORMANE COMPARISION OF JOIN AND SUQUERIES USING EXPLAIN

*Untitled Document 1 - gedit

File Edit View Search Tools Documents Help

Open Save Undo

explain.txt *Untitled Document 1

```
mysql> EXPLAIN SELECT DISTINCT branch FROM holdings NATURAL JOIN titles WHERE author = 'ann brown';
+----+-------------+----------+------+---------------+-------+---------+--------------------+------+-----------------------------+
| id | select_type | table    | type | possible_keys | key   | key_len | ref                | rows | Extra                       |
+----+-------------+----------+------+---------------+-------+---------+--------------------+------+-----------------------------+
|  1 | SIMPLE      | titles   | ALL  | PRIMARY       | NULL  | NULL    | NULL               |    5 | Using where; Using temporary |
|  1 | SIMPLE      | holdings | ref  | title         | title | 22      | sample.titles.title |    1 | Using index                 |
+----+-------------+----------+------+---------------+-------+---------+--------------------+------+-----------------------------+
2 rows in set (0.00 sec)

EXPLAIN SELECT DISTINCT h.branch FROM holdings AS h WHERE title IN ( SELECT t.title FROM titles AS t WHERE author = 'ann brown');
+----+--------------------+-------+----------------+---------------+---------+---------+------+------+--------------------------+
| id | select_type        | table | type           | possible_keys | key     | key_len | ref  | rows | Extra                    |
+----+--------------------+-------+----------------+---------------+---------+---------+------+------+--------------------------+
|  1 | PRIMARY            | h     | index          | NULL          | PRIMARY | 26      | NULL |   10 | Using where; Using index |
|  2 | DEPENDENT SUBQUERY | t     | unique_subquery | PRIMARY       | PRIMARY | 22      | func |    1 | Using where              |
+----+--------------------+-------+----------------+---------------+---------+---------+------+------+--------------------------+
2 rows in set (0.00 sec)
```

Plain Text Tab Width: 8 Ln 18, Col 1 INS

Menu Wed Aug 7, 10:58

TYPES OF JOINS

- INNER JOIN
- NATURAL JOIN
- OUTER JOIN
  - LEFT JOIN
  - RIGHT JOIN
- SELF JOIN
- MULTIPLE TABLE JOIN

INNER JOIN

- SYNTAX

  SELECT<column name> FROM table1 JOIN/INNERJOIN table2

  [ON(join_condition)] /

  [WHERE(CONDITION)] /

  [USING(column_name)];

- WHEN using USING TYPE THE COLUMN NAME SHOULD BE SAME IN BOTH THE TABLES.
- IF CONDITION IS NOT DEFINED IT WORKS LIKE A CARTESIAN PRODUCT OF THE TWO TABLE.

NATURAL JOIN

- SYNTAX

  SELECT<column name> FROM table1 NATURAL JOIN table2;

- ONE COLUMN NAME SHOULD BE SAME IN BOTH THE TABLES UPON WHICH JOIN WILL BE PERFORMED.

LEFT JOIN

- LEFT JOIN IS USED TO PRESERVE ALL THE ROWS MATCHING OR NON-MATCHING OF THE TABLE WRITTEN LEFT OF THE JOIN.
- SYNTAX

```
SELECT<column name> FROM table1  LEFT JOIN table2
        [ON(join_condition)] /
        [USING(column_name)];
```

- WHEN using USING TYPE THE COLUMN NAME SHOULD BE SAME IN BOTH THE TABLES.

RIGHT JOIN

- RIGHT JOIN IS USED TO PRESERVE ALL THE ROWS MATCHING OR NON-MATCHING OF THE TABLE WRITTEN RIGHT OF THE JOIN.
- SYNTAX

```
SELECT<column name> FROM table1  RIGHT JOIN table2
        [ON(join_condition)] /
        [USING(column_name)];
```

- WHEN using USING TYPE THE COLUMN NAME SHOULD BE SAME IN BOTH THE TABLES.

SELF JOIN

- SELF JOIN IS JOINING TABLE TO ITSELF MAKING COPY OF IT TEMPORARY.
- SYNTAX

  SELECT<column name> FROM table1 AS a JOIN table1 AS b

  [ON(join_condition)] /

  [WHERE(CONDITION)] /

  [USING(column_name)];

- WHEN using USING TYPE THE COLUMN NAME SHOULD BE SAME IN BOTH THE TABLES.

MULTIPLE TABLE JOIN

- WE CAN ALSO JOIN MORE THAN TWO TABLES.

  LIKE: SELECT * FROM tab1,tab2,tab3;
- THIS STATEMENT WILL PRODUCE THE CARTESIAN PRODUCT OF ALL THE 3 TABLES.
- EXAMPLE:

  SELECT pp.emp_id FROM proj_tech AS pt JOIN technologies AS t

  ON pt.tech_id = t.id AND t.name = 'android' JOIN past_proj pp

  ON pp.p_id = pt.p_id GROUP BY pp.emp_id;