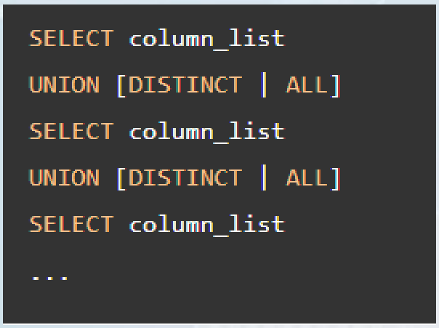
Summary – Day 13

Union:

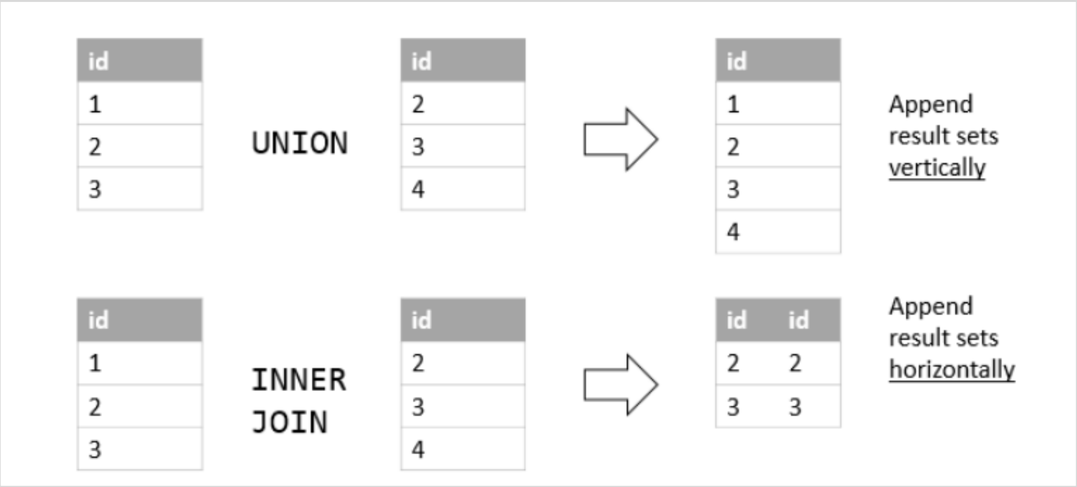
- MySQL UNION operator allows you to combine two or more result sets of queries into a single result set. The following illustrates the syntax of the UNION operator.



```
SELECT column_list
UNION [DISTINCT | ALL]
SELECT column_list
UNION [DISTINCT | ALL]
SELECT column_list
...
```

- To combine result set of two or more queries using the UNION operator, these are the basic rules that you must follow:
  1. First, the number and the orders of columns that appear in all SELECT statements must be the same.
  2. Second, the data types of columns must be the same or compatible.

- By default, the UNION operator removes duplicate rows even if you don’t specify the DISTINCT operator explicitly.


id
1
2
3
UNION
id
2
3
4
id
1
2
3
4
Append result sets vertically
id
1
2
3
INNER JOIN
id
2
3
4
id id
2 2
3 3
Append result sets horizontally

Union All:

- MySQL UNION ALL operator is a union query command which syndicates multiple SELECT statements' results into a single result row.

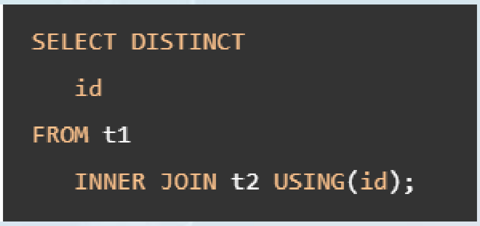
- Like, MySQL UNION operator it is also a useful command in MySQL database to combine more than two of the output set provided by the use of SELECT queries. But there is slightly a difference in their work which makes them two different commands used in MySQL for their respective purposes.

- The MySQL UNION operator performs to give the distinctive values of set in the result after the union of the result rows of SELECT statements whereas the MySQL UNION ALL operator allows the union producing the result set from SELECT statements having replica values in the records fetched from the database tables where it is applied.

```
SELECT Col_expr1, Col_expr2,…,Col_exprN FROM TableName_A
[WHERE option condition] UNION ALL
SELECT Col_expr1, Col_expr2,…,Col_exprN FROM TableName_B
[WHERE option condition];
```

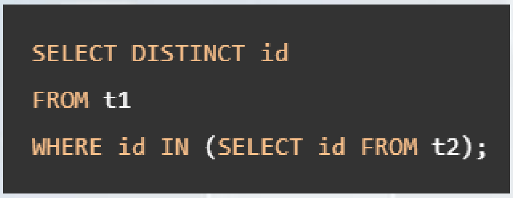
Intersect:

- The INTERSECT operator is a set operator that returns only distinct rows of two queries or more queries.
- The INTERSECT operator compares the result sets of two queries and returns the distinct rows that are output by both queries
- To use the INTERSECT operator for two queries, you follow these rules.
    - The order and the number of columns in the select list of the queries must be the same.
    - The data types of the corresponding columns must be compatible.
- We can emulate intersect using two ways.
    - Using inner join:
        - The following statement uses DISTINCT operator and INNER JOIN clause to return the distinct rows in both tables.



```
SELECT DISTINCT
   id
FROM t1
   INNER JOIN t2 USING(id);
```

- Using IN and Subquery:
    - The following statement uses the IN operator and a subquery to return the intersection of the two result sets.



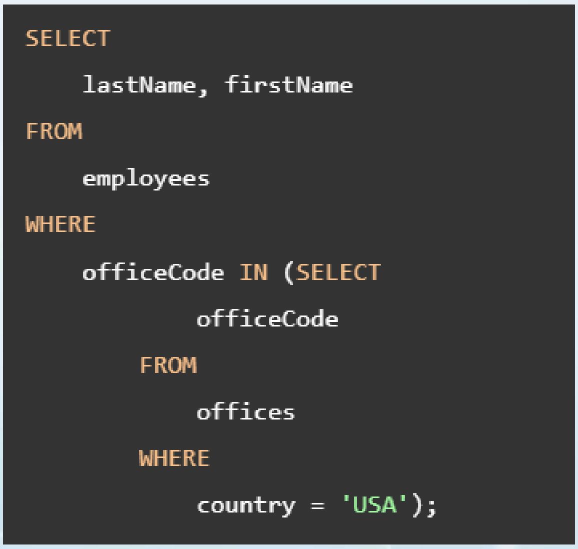
```
SELECT DISTINCT id
FROM t1
WHERE id IN (SELECT id FROM t2);
```

Subqueries:

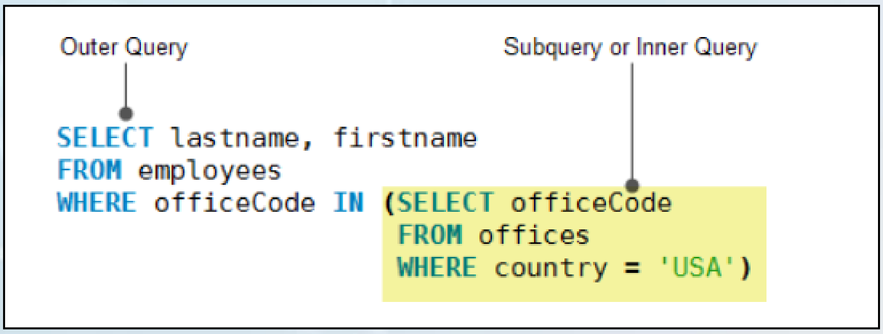
- A MySQL subquery is a query nested within another query such as SELECT, INSERT, UPDATE or DELETE. Also, a subquery can be nested within another subquery.

- A MySQL subquery is called an inner query while the query that contains the subquery is called an outer query. A subquery can be used anywhere that expression is used and must be closed in parentheses.

- For example, the following query uses a subquery to return the employees who work in the offices located in the USA.



```
SELECT
    lastName, firstName
FROM
    employees
WHERE
    officeCode IN (SELECT
            officeCode
        FROM
            offices
        WHERE
            country = 'USA');
```

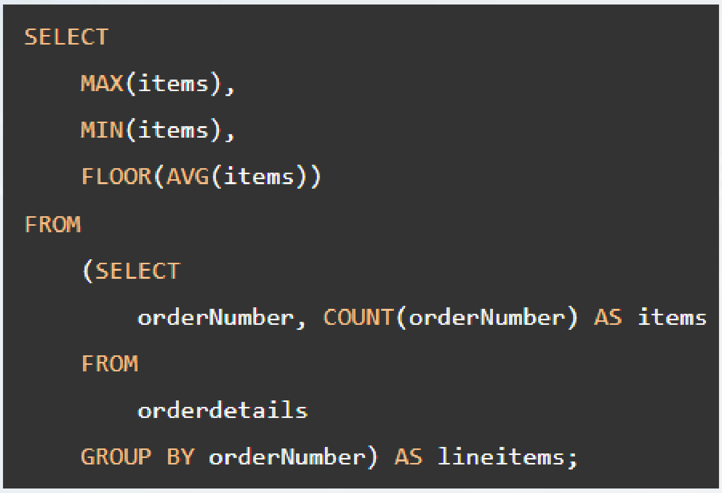


```
SELECT lastname, firstname
FROM employees
WHERE officeCode IN (SELECT officeCode
                     FROM offices
                     WHERE country = 'USA')
```

- We can also use the subqueries after FROM clause.

- When you use a subquery in the FROM clause, the result set returned from a subquery is used as a temporary table. This table is referred to as a derived table or materialized subquery.

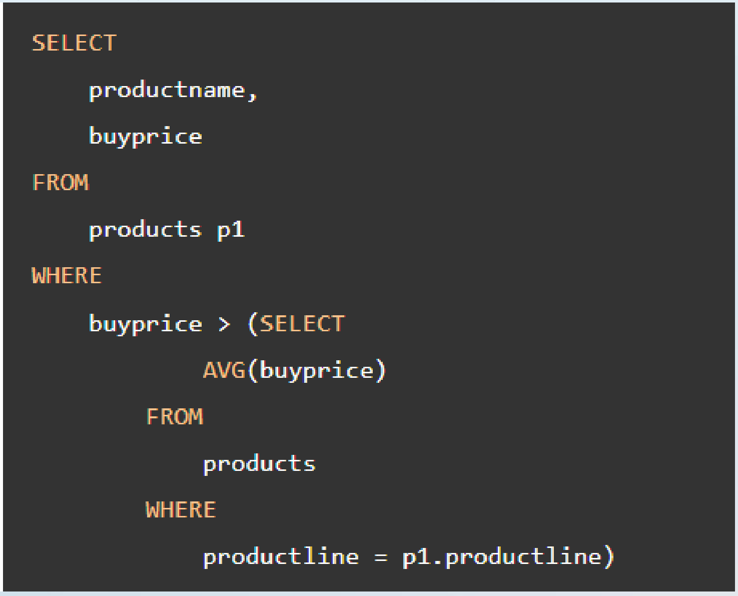
- The following subquery finds the maximum, minimum, and average number of items in sale orders.



```
SELECT
    MAX(items),
    MIN(items),
    FLOOR(AVG(items))
FROM
    (SELECT
        orderNumber, COUNT(orderNumber) AS items
    FROM
        orderdetails
    GROUP BY orderNumber) AS lineitems;
```

Correlated Subquery:

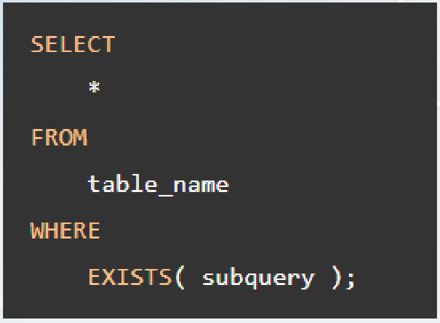
- A subquery is independent. It means that you can execute the subquery as a standalone query.
- Unlike a standalone subquery, a correlated subquery is a subquery that uses the data from the outer query. In other words, a correlated subquery depends on the outer query. A correlated subquery is evaluated once for each row in the outer query.
- The following example uses a correlated subquery to select products whose buy prices are greater than the average buy price of all products in each product line.



```
SELECT
    productname,
    buyprice
FROM
    products p1
WHERE
    buyprice > (SELECT
            AVG(buyprice)
        FROM
            products
        WHERE
            productline = p1.productline)
```

Subqueries with EXISTS and NOT EXISTS:

- When a subquery is used with the EXISTS or NOT EXISTS operator, a subquery returns a Boolean value of TRUE or FALSE. The following query illustrates a subquery used with the EXISTS operator.



```
SELECT
    *
FROM
    table_name
WHERE
    EXISTS( subquery );
```