5. Querying (using ANY, ALL, UNION, INTERSECT, JOIN Constraints etc.)

UNION

Union is used to combine the results of two queries into a single result set of all matching rows. Both the queries must result in the same number of columns and compatible data types in order to unite. All duplicate records are removed automatically unless UNION ALL is used.

Syntax

{ <query_specification> | (<query_expression>) }
{ UNION | UNION ALL}
{ <query_specification> | (<query_expression>) }

mysql> create table authors(name varchar(30));
Query OK,
mysql> insert into authors values('Rohith'),('kavya'),('Rahul');
Query OK, 3 rows affected
mysql> select * from authors;

```
+--------+
| name   |
+--------+
| Rohith |
| kavya  |
| Rahul  |
+--------+
```

3 rows in set
mysql> create table speakers(name varchar(30));
Query OK**,**
mysql> insert into speakers values('Rani'),('Raju'),('Shilpa'),('sony');
Query OK,
mysql> select * from speakers;

```
+--------+
| name   |
+--------+
| Rani   |
| Raju   |
| Shilpa |
| sony   |
+--------+
```

mysql> select name from Speakers union select name from Authors order by name;

+--------+
| name |
+--------+
| kavya |
| Rahul |
| Raju |
| Rani |
| Rohith |
| Shilpa |
| sony |
+--------+

INTERSECT

It is used to take the result of two queries and returns only those rows which are common in both result sets. It removes duplicate records from the final result set.

Syntax

{ <query_specification> | (<query_expression>) }
{ INTERSECT }
{ <query_specification> | (<query_expression>) }

You want the list of people who are Speakers and they are also Authors. Hence, how will you prepare such a list?

mysql> select name from Speakers intersect select name from Authors order by name;
Empty set

EXCEPT / MINUS

It is used to take the distinct records of two one query and returns only those rows which do not appear in the second result set.

Syntax

{ <query_specification> | (<query_expression>) }
{ EXCEPT | INTERSECT }
{ <query_specification> | (<query_expression>) }

You want the list of people who are only Speakers and they are not Authors. Hence, how will you prepare such a list?

mysql> select name from Speakers except select name from Authors order by name;

+--------+

| name |
+--------+
| Raju |
| Rani |
| Shilpa |
| sony |
+--------+

You want the list of people who are only Authors and they are not Speakers. Hence, how will you prepare such a list?

```
mysql> select name from Authors except select name from Speakers order by name;
+--------+
| name |
+--------+
| kavya |
| Rahul |
| Rohith |
+--------+
3 rows in set
```

The SQL ANY Operator

The ANY operator:

- returns a boolean value as a result
- returns TRUE if ANY of the subquery values meet the condition

ANY means that the condition will be true if the operation is true for any of the values in the range.

ANY Syntax

```
SELECT column_name(s)
 FROM table_name
WHERE column_name operator ANY
 (SELECT column_name
 FROM table_name
 WHERE condition);
```

Note: The *operator* must be a standard comparison operator (=, <>, !=, >, >=, <, or <=).

Create Table

Consider the following Products Table and OrderDetails Table,**Products Table**

ProductID	ProductName	SupplierID	CategoryID	Price
1	Chais	1	1	18
2	Chang	1	1	19
3	Aniseed Syrup	1	2	10
4	Chef Anton's Cajun Seasoning	2	2	22
5	Chef Anton's Gumbo Mix	2	2	21
6	Boysenberry Spread	3	2	25
7	Organic Dried Pears	3	7	30
8	Northwoods Cranberry Sauce	3	2	40
9	Mishi Kobe Niku	4	6	97

OrderDetails Table

OrderDetailsID	OrderID	ProductID	Quantity
1	10248	1	12
2	10248	2	10
3	10248	3	15
4	10249	1	8
5	10249	4	4
6	10249	5	6
7	10250	3	5
8	10250	4	18
9	10251	5	2
10	10251	6	8
11	10252	7	9
12	10252	8	9
13	10250	9	20
14	10249	9	4

SQL ANY Examples

1) The following SQL statement lists the ProductName if it finds ANY records in the OrderDetails table has Quantity equal to 10 (this will return TRUE because the Quantity column has some values of 10):

```
SELECT ProductName
FROM Products
WHERE ProductID = ANY
(SELECT ProductID
FROM OrderDetails
WHERE QuantityID = 10);
```

```
+-------------+
| ProductName |
+-------------+
| chang       |
+-------------+
1 row in set
```

2) The following SQL statement lists the ProductName if it finds ANY records in the OrderDetails table has Quantity larger than 99 (this will return TRUE because the Quantity column has some values larger than 99):

mysql> SELECT ProductName FROM Products WHERE ProductID = ANY (SELECT ProductID FROM OrderDetails WHERE QuantityID > 10);

```
+-----------------------------+
| ProductName                 |
+-----------------------------+
| chais                       |
| Aniseed syrup               |
| chef antnons cajun seasoning |
| mishikobeniku               |
+-----------------------------+
4 rows in set
```

The SQL ALL Operator

The ALL operator:

- returns a boolean value as a result
- returns TRUE if ALL of the subquery values meet the condition
- is used with SELECT, WHERE and HAVING statements

ALL means that the condition will be true only if the operation is true for all values in the range.

ALL Syntax With SELECT

```
SELECT ALL column_name(s)
FROM table_name
WHERE condition;
```

ALL Syntax With WHERE or HAVING

SELECT column_name(s)
FROM table_name
WHERE column_name operator ALL (SELECT column_name FROM table_name WHERE condition);

Note: The *operator* must be a standard comparison operator (=, <>, !=, >, >=, <, or <=).

1) **The following SQL statement lists ALL the product names:**

mysql> SELECT ALL ProductName FROM Products WHERE TRUE;

```
+-----------------------------+
| ProductName                 |
+-----------------------------+
| chais                       |
| chang                       |
| Aniseed syrup               |
| chef antnons cajun seasoning |
| chefantonsgumbo mix         |
| boysenberry spread          |
| orgnicdriedpears            |
| northwoodscranberrysauce    |
| mishikobeniku               |
+-----------------------------+
```

9 rows in set

2) **The following SQL statement lists the ProductName if ALL the records in the OrderDetails table has Quantity equal to 10. This will of course return FALSE because the Quantity column has many different values (not only the value of 10):**

mysql> SELECT ProductName FROM Products WHERE ProductID = ALL (SELECT ProductID FROM OrderDetails WHERE QuantityID = 10);

```
+-------------+
| ProductName |
+-------------+
| chang       |
+-------------+
```

1 row in set