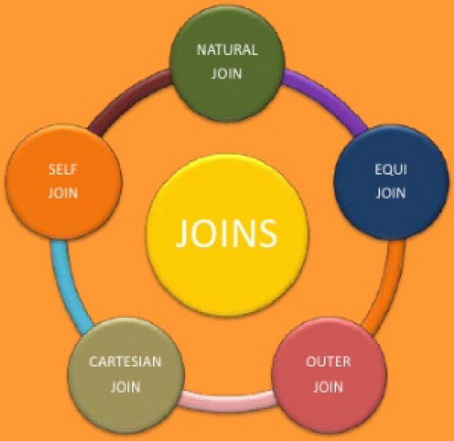
Using ORACLE®

Retrieving data from multiple tables(joins)
Sub-queries and Set operators

datamining tools

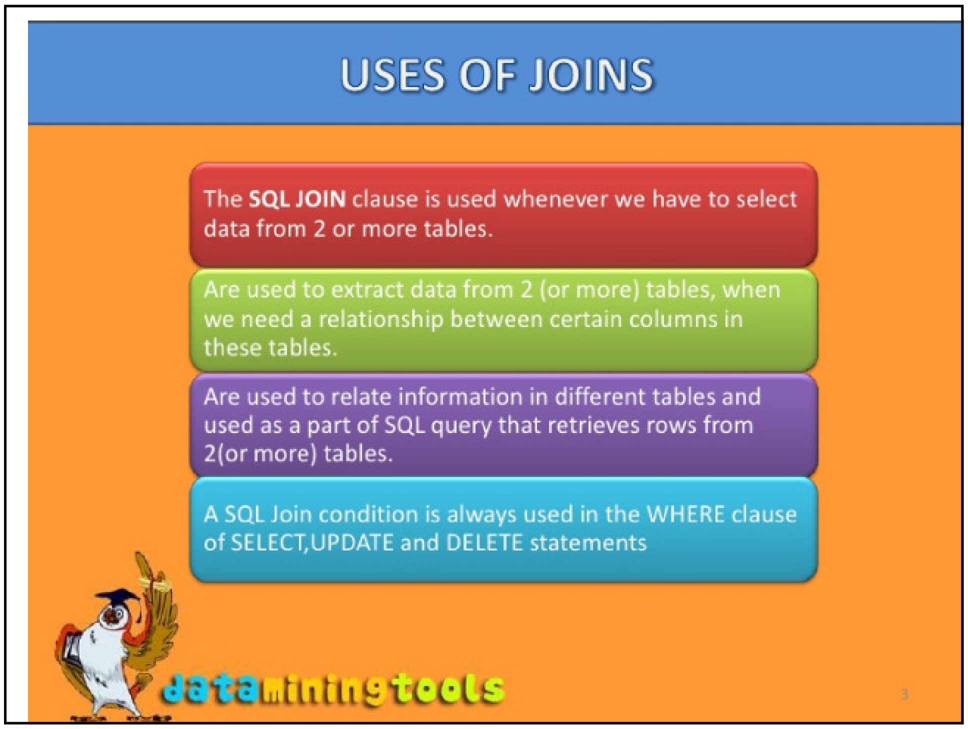
1

JOINS



datamining tools

2



USES OF JOINS

The **SQL JOIN** clause is used whenever we have to select data from 2 or more tables.

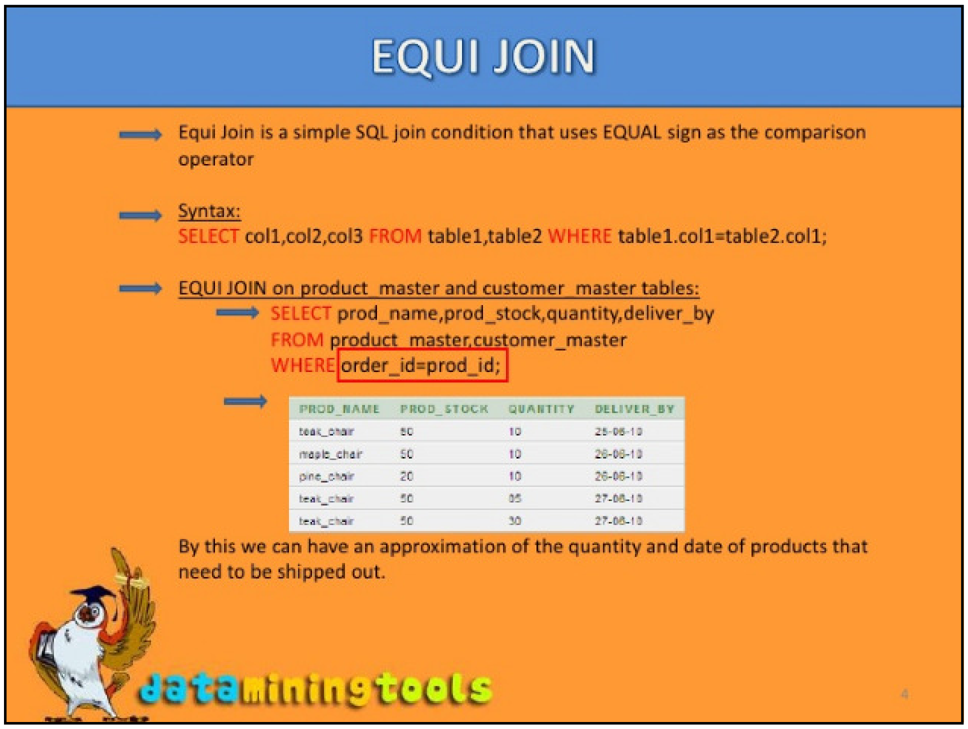
Are used to extract data from 2 (or more) tables, when we need a relationship between certain columns in these tables.

Are used to relate information in different tables and used as a part of SQL query that retrieves rows from 2(or more) tables.

A SQL Join condition is always used in the WHERE clause of SELECT,UPDATE and DELETE statements

dataminingtools

3



EQUI JOIN

- Equi Join is a simple SQL join condition that uses EQUAL sign as the comparison operator
- Syntax:
  SELECT col1,col2,col3 FROM table1,table2 WHERE table1.col1=table2.col1;
- EQUI JOIN on product_master and customer_master tables:
  - SELECT prod_name,prod_stock,quantity,deliver_by
    FROM product_master,customer_master
    WHERE $\boxed{\text{order\_id=prod\_id;}}$
  -

PROD_NAME	PROD_STOCK	QUANTITY	DELIVER_BY
teak_chair	80	10	25-08-13
maple_chair	50	10	26-08-13
pine_chair	20	10	26-08-13
teak_chair	50	05	27-08-13
teak_chair	50	30	27-08-13

By this we can have an approximation of the quantity and date of products that need to be shipped out.

dataminingtools

4

OUTER JOINS

- OUTER join condition returns all rows from both tables which satisfy the join condition along with rows which do not satisfy the join condition from one of the tables. The SQL outer join operator in Oracle is (+) and is used on one side of the join condition only.

- Syntax:
  SELECT col1,col2 FROM table1,table2 WHERE table1.col1 (+) = table2.col1;

- OUTER JOIN on product_master table and customer_master:
  SELECT p.prod_id, p.prod_name, o.order_id, o.quantity
  FROM customer_master o, product_master p

  WHERE p.prod_id (+)= o.order_id ;

PROD_ID	PROD_NAME	ORDER_ID	QUANTITY
VF001	teak_chair	VF001	10
VF002	maple_chair	VF002	10
VF003	pine_chair	VF003	10
VF001	teak_chair	VF001	05
VF001	teak_chair	VF001	30

dataminingtools

5

CARTESIAN JOINS

- If a SQL join condition is omitted or if it is invalid the join operation will result in a Cartesian product. The Cartesian product returns a number of rows equal to the product of all rows in all the tables being joined. For example, if the first table has 20 rows and the second table has 10 rows, the result will be 20 * 10, or 200 rows. This query takes a long time to execute.

- SYNTAX:
  SELECT col1,col2 FROM table1,table2;

- CARTESIAN JOIN on product_master and customer_master:
  SELECT order_id,prod_name
  FROM customer_master,product_master;

- Here each row from customer_master will be Mapped to each row of product_master.Here the This query contains 50 rows only 10 rows are Shown in the figure

ORDER_ID	PROD_NAME
VF001	teak_chair
VF001	maple_chair
VF001	pine_chair
VF001	teak_sofa
VF001	maple_sofa
VF001	pine_sofa
VF001	dining_table
VF001	single_bed
VF001	double_bed
VF001	teak_cupboard

More than 10 rows available. Increase rows selector to view more rows.

dataminingtools

6

SELF JOINS

A Self join is the type of SQL join where we join a particular table to itself. Here it is necessary to ensure that the join statement defines an ALIAS name for both the copies of the tables to avoid column ambiguity

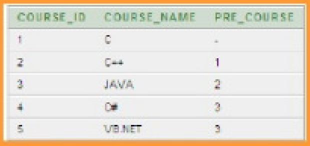
Syntax for Table Alias:

```
SELECT s.first_name FROM student_details s;
```

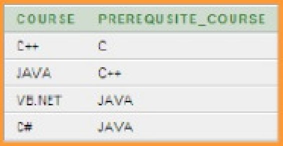
In this query alias s is defined for the table student_details and the column first_name is selected from the table.

Self Join on Course table:

```
SELECT a.course_name AS COURSE,b.course_name AS PREREQUSITE_COURSE
FROM course_m a,course_m b
WHERE a.pre_course=b.course_id;
```



COURSE_ID	COURSE_NAME	PRE_COURSE
1	C	-
2	C++	1
3	JAVA	2
4	C#	3
5	VB.NET	3



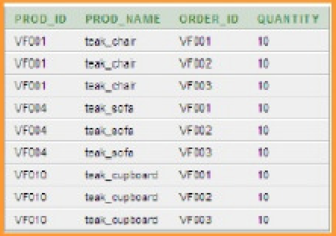
COURSE	PREREQUSITE_COURSE
C++	C
JAVA	C++
VB.NET	JAVA
C#	JAVA

7

NATURAL/CARTESIAN JOINS

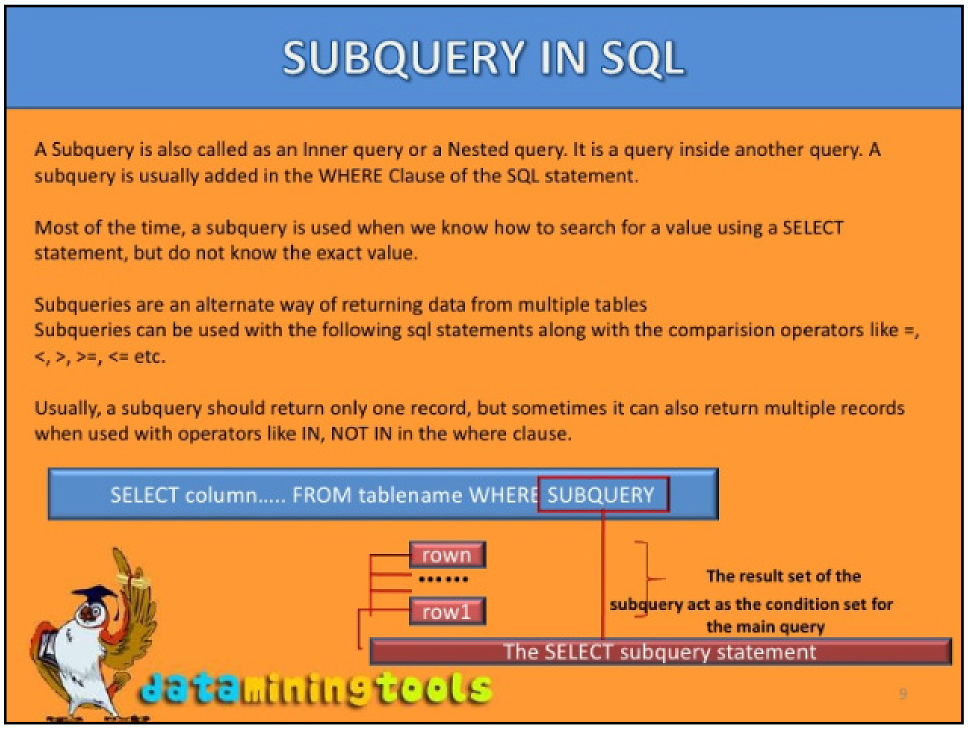
CARTESIAN JOIN is also known as NATURAL JOIN .The output of this join can be filtered using the WHERE clause.

```
SELECT prod_ID,prod_name ,order_id,quantity
FROM product_master
NATURAL JOIN customer_master
WHERE prod_name LIKE ('teak%') AND quantity=10;
```



PROD_ID	PROD_NAME	ORDER_ID	QUANTITY
VF001	teak_chair	VF001	10
VF001	teak_chair	VF002	10
VF001	teak_chair	VF003	10
VF004	teak_sofa	VF001	10
VF004	teak_sofa	VF002	10
VF004	teak_sofa	VF003	10
VF010	teak_cupboard	VF001	10
VF010	teak_cupboard	VF002	10
VF010	teak_cupboard	VF003	10

8



SUBQUERY IN SQL

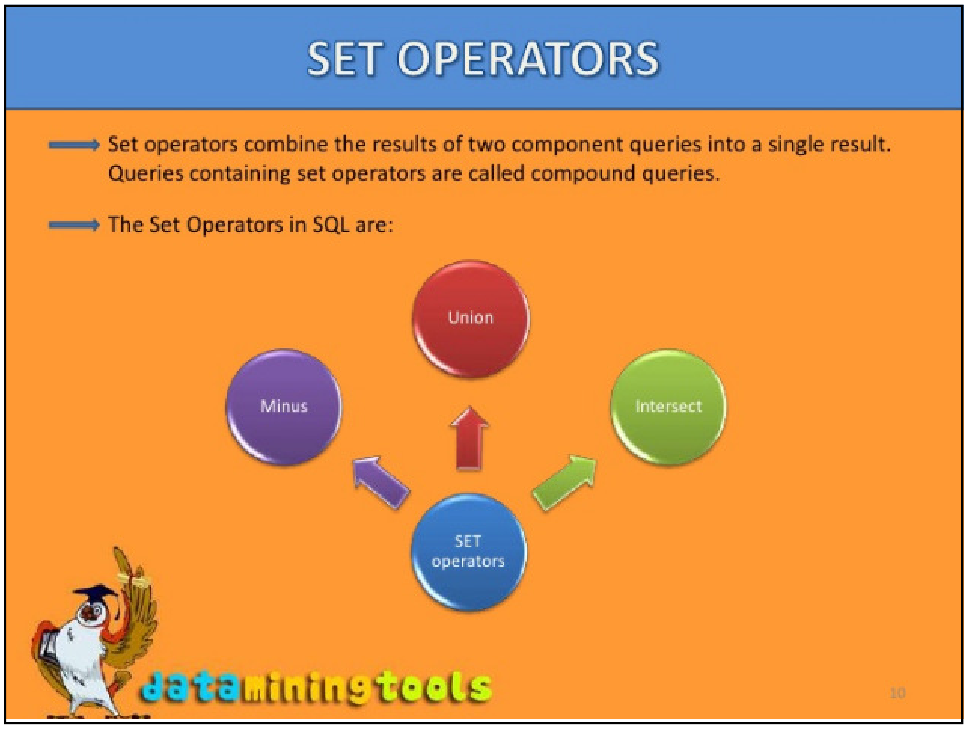
A Subquery is also called as an Inner query or a Nested query. It is a query inside another query. A subquery is usually added in the WHERE Clause of the SQL statement.

Most of the time, a subquery is used when we know how to search for a value using a SELECT statement, but do not know the exact value.

Subqueries are an alternate way of returning data from multiple tables
Subqueries can be used with the following sql statements along with the comparision operators like =, <, >, >=, <= etc.

Usually, a subquery should return only one record, but sometimes it can also return multiple records when used with operators like IN, NOT IN in the where clause.

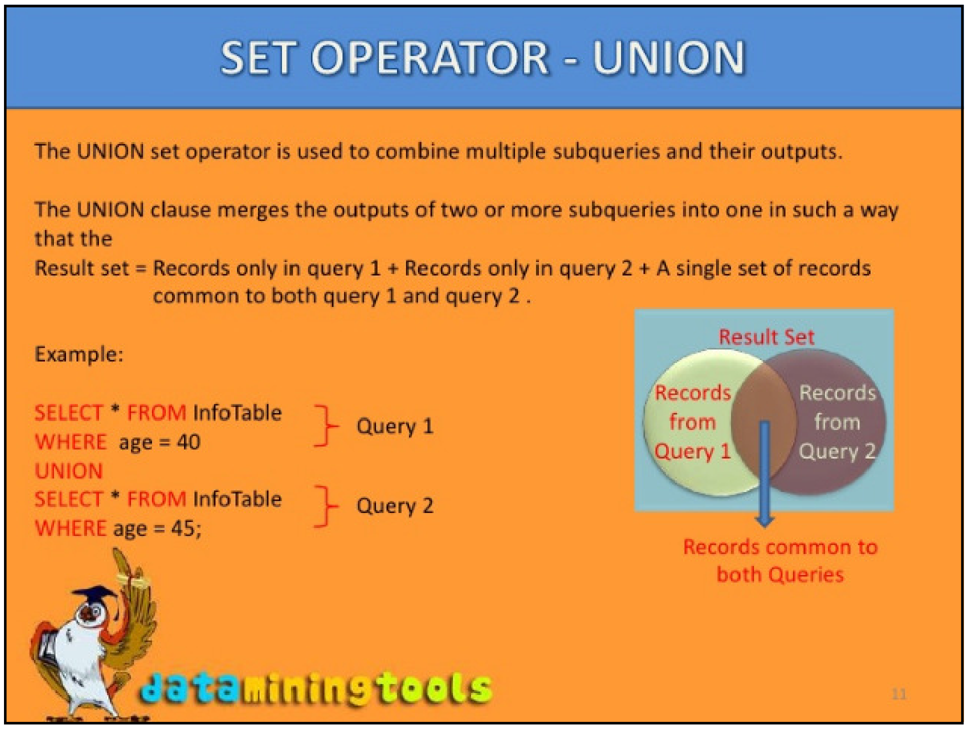
SELECT column….. FROM tablename WHERE SUBQUERY
rown
••••••
row1
The result set of the subquery act as the condition set for the main query
The SELECT subquery statement
dataminingtools
9



SET OPERATORS

- Set operators combine the results of two component queries into a single result. Queries containing set operators are called compound queries.
- The Set Operators in SQL are:

Union
Minus
Intersect
SET operators
dataminingtools
10



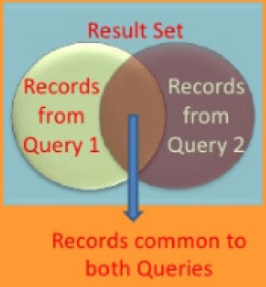
SET OPERATOR - UNION

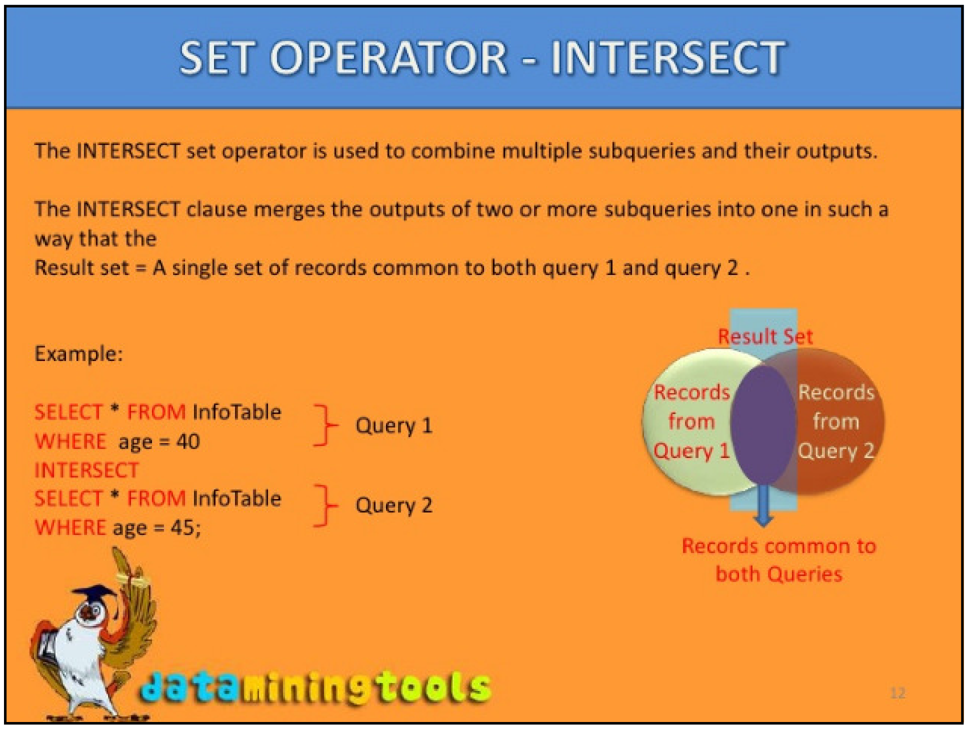
The UNION set operator is used to combine multiple subqueries and their outputs.

The UNION clause merges the outputs of two or more subqueries into one in such a way that the
Result set = Records only in query 1 + Records only in query 2 + A single set of records common to both query 1 and query 2 .

Example:

```
SELECT * FROM InfoTable      } Query 1
WHERE  age = 40
UNION
SELECT * FROM InfoTable      } Query 2
WHERE age = 45;
```





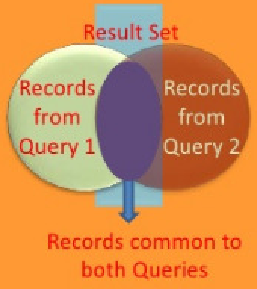
SET OPERATOR - INTERSECT

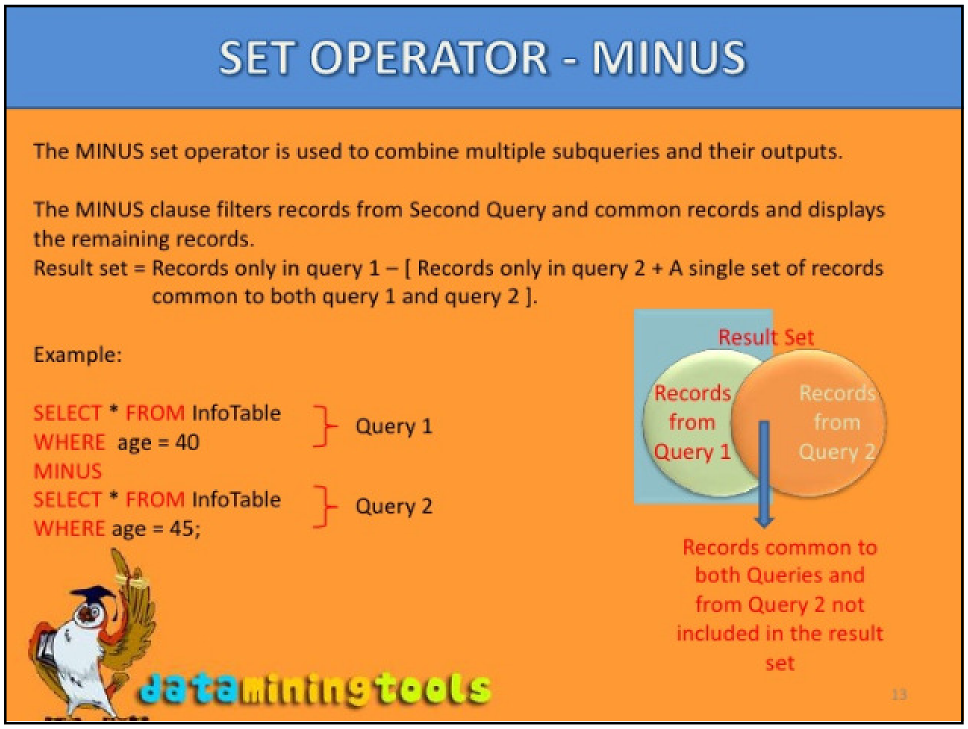
The INTERSECT set operator is used to combine multiple subqueries and their outputs.

The INTERSECT clause merges the outputs of two or more subqueries into one in such a way that the
Result set = A single set of records common to both query 1 and query 2 .

Example:

```
SELECT * FROM InfoTable      } Query 1
WHERE  age = 40
INTERSECT
SELECT * FROM InfoTable      } Query 2
WHERE age = 45;
```





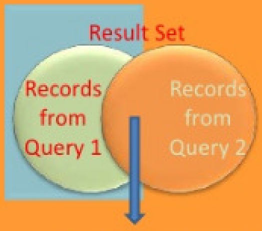
SET OPERATOR - MINUS

The MINUS set operator is used to combine multiple subqueries and their outputs.

The MINUS clause filters records from Second Query and common records and displays the remaining records.
Result set = Records only in query 1 – [Records only in query 2 + A single set of records common to both query 1 and query 2].

Example:

```
SELECT * FROM InfoTable
WHERE  age = 40          } Query 1
MINUS
SELECT * FROM InfoTable
WHERE age = 45;          } Query 2
```


Result Set
Records from Query 1
Records from Query 2
Records common to both Queries and from Query 2 not included in the result set

dataminingtools

13