MySQL Natural Join

When we combine rows of two or more tables based on a common column between them, this operation is called joining. **A natural join is a type of join operation that creates an implicit join by combining tables based on columns with the same name and data type**. It is similar to the [INNER](https://www.javatpoint.com/mysql-inner-join) or [LEFT JOIN](https://www.javatpoint.com/mysql-left-join), but we cannot use the ON or USING clause with natural join as we used in them.

Points to remember:

* There is no need to specify the column names to join.
* The resultant table always contains unique columns.
* It is possible to perform a natural join on more than two tables.
* Do not use the ON clause.

Syntax:

The following is a basic syntax to illustrate the natural join:

1. **SELECT** [column\_names | \*]
2. **FROM** table\_name1
3. NATURAL JOIN table\_name2;

In this syntax, we need to specify the **column names** to be included in the result set after the SELECT keyword. If we want to select all columns from both tables, the **\*** operator will be used. Next, we will specify the **table names** for joining after the FROM keyword and write the **NATURAL JOIN** clause between them.

Natural Join Example

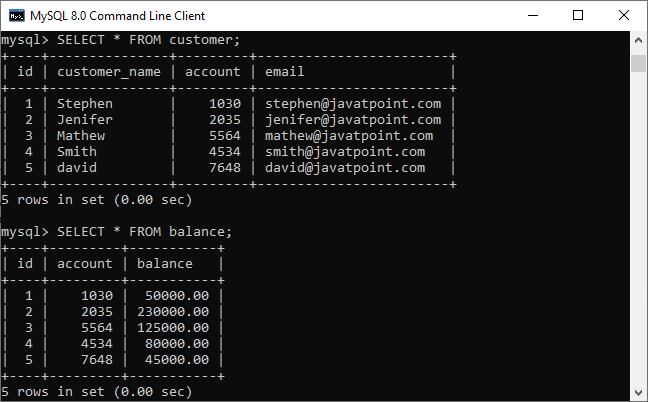
Let us understand how natural join works in [MySQL](https://www.javatpoint.com/mysql-tutorial) through examples. First, we will create two tables named **customer** and **balance** using the below statements:

1. /\* -- Table name: customer -\*/
2. **CREATE** **TABLE** customer (
3. id **INT** AUTO\_INCREMENT **PRIMARY** **KEY**,
4. customer\_name **VARCHAR**(55),
5. account **int**,
6. email **VARCHAR**(55)
7. );
9. /\* -- Table name: balance -\*/
10. **CREATE** **TABLE** balance (
11. id **INT** AUTO\_INCREMENT **PRIMARY** **KEY**,
12. account **int**,
13. balance **FLOAT**(10, 2)
14. );

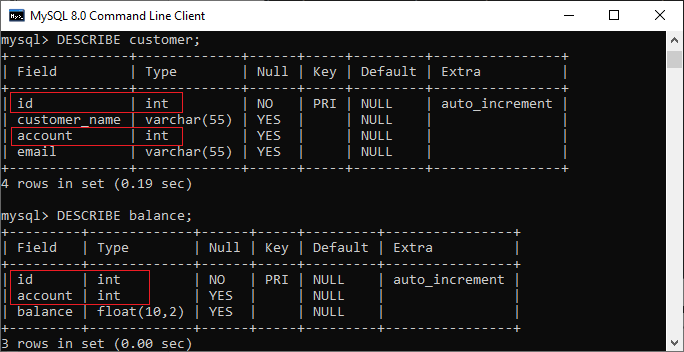
Next, we will fill some records into both tables using the below statements:

1. /\* -- Data for customer table -\*/
2. **INSERT** **INTO** customer(customer\_name, account, email)
3. **VALUES**('Stephen', 1030, 'stephen@javatpoint.com'),
4. ('Jenifer', 2035, 'jenifer@javatpoint.com'),
5. ('Mathew', 5564, 'mathew@javatpoint.com'),
6. ('Smith', 4534, 'smith@javatpoint.com'),
7. ('David', 7648, 'david@javatpoint.com');
9. /\* -- Data for balance table -\*/
10. **INSERT** **INTO** balance(account, balance)
11. **VALUES**(1030, 50000.00),
12. (2035, 230000.00),
13. (5564, 125000.00),
14. (4534, 80000.00),
15. (7648, 45000.00);

Next, we will execute the [SELECT statement](https://www.javatpoint.com/mysql-select) to verify the table data:



Now, we will see the condition that fulfills the criteria for natural join. We can do this by examining the table structure using the **DESCRIBE** statement. See the below image:

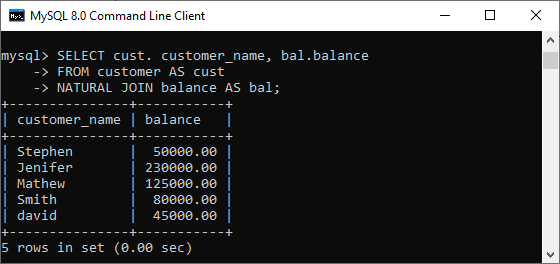


In this image, we can see that **column names id and account and its data types** are the same that fulfill the natural join criteria. Hence we can use natural join on them.

Execute the below statement for joining tables using natural join:

1. mysql> **SELECT** cust. customer\_name, bal.balance
2. **FROM** customer **AS** cust
3. NATURAL JOIN balance **AS** bal;

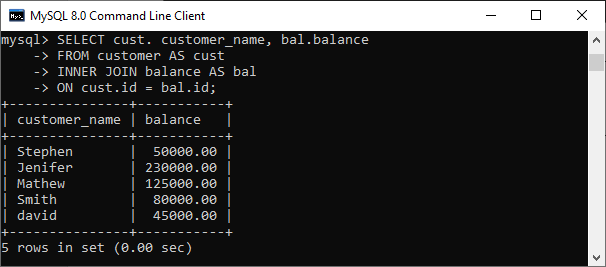
We will get the following result:



We can do the same job with the help of **INNER JOIN using the ON clause**. Here is the query to explain this join:

1. mysql> **SELECT** cust. customer\_name, bal.balance
2. **FROM** customer **AS** cust
3. **INNER** JOIN balance **AS** bal
4. **ON** cust.id = bal.id;

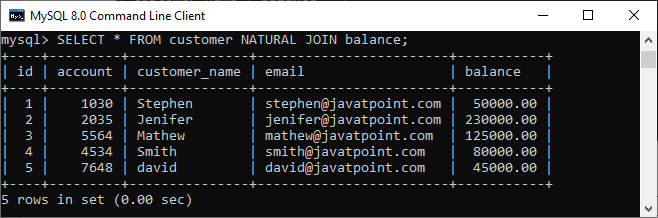
After successful execution, we will get the same result as the natural join:



Now, we will use (\*) in the place of column names as follows:

1. mysql> **SELECT** \* **FROM** customer NATURAL JOIN balance;

Suppose we use the asterisk (\*) in the place of column names, then the natural join automatically searches the same column names and their data types and join them internally. Also, it does not display the repeated columns in the output. Hence, we should get the below output after executing the above statement:

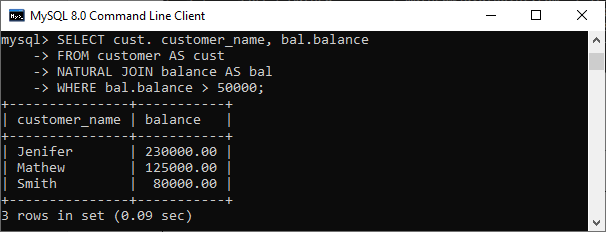


Natural Join with WHERE Clause

The [WHERE clause](https://www.javatpoint.com/mysql-where) is used to return the filter result from the table. The following example illustrates this with the natural join clause:

1. mysql> **SELECT** cust. customer\_name, bal.balance
2. **FROM** customer **AS** cust
3. NATURAL JOIN balance **AS** bal
4. **WHERE** bal.balance > 50000;

We will get the following result where customer information is displayed whose **account balance is greater than 50000**.



Natural Join Using Three Tables

We know that natural join can also perform a join operation on more than two tables. To understand this, we will use the syntax as follows:

1. **SELECT** [column\_names | \*]
2. **FROM** table\_name1
3. NATURAL JOIN table\_name2
4. NATURAL JOIN table\_name3;

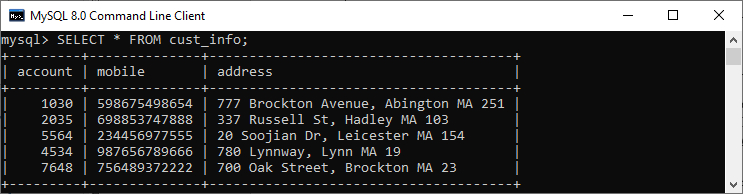
Let us create another table named **cust\_info** using the below statement:

1. **CREATE** **TABLE** cust\_info (
2. account **int**,
3. mobile **VARCHAR**(15),
4. address **VARCHAR**(65)
5. );

Then, we will fill records into this table:

1. **INSERT** **INTO** cust\_info(account, mobile, address)
2. **VALUES**(1030, '598675498654', '777 Brockton Avenue, Abington MA 251'),
3. (2035, '698853747888', '337 Russell St, Hadley MA 103'),
4. (5564, '234456977555', '20 Soojian Dr, Leicester MA 154'),
5. (4534, '987656789666', '780 Lynnway, Lynn MA 19'),
6. (7648, '756489372222', '700 Oak Street, Brockton MA 23');

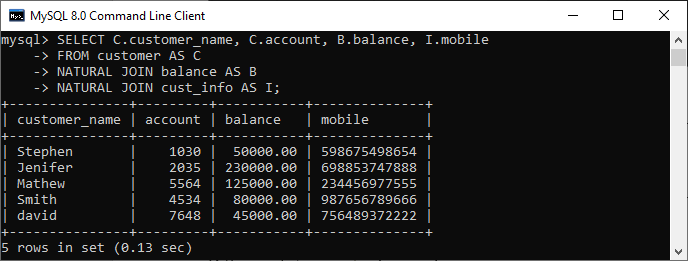
We can verify the data using the SELECT statement. See the below image:



To join three tables using natural join, we need to execute the statement as follows:

1. mysql> **SELECT** C.customer\_name, C.account, B.balance, I.mobile
2. **FROM** customer **AS** C
3. NATURAL JOIN balance **AS** B
4. NATURAL JOIN cust\_info **AS** I;

It will give the below result. Here we can see that the account number is present in all three columns but arrived only once in the output that fulfills the natural join criteria.



Difference between Natural Join and Inner Join

|  |  |  |
| --- | --- | --- |
| **SN** | **Natural Join** | **Inner Join** |
| **1.** | It joins the tables based on the same column names and their data types. | It joins the tables based on the column name specified in the ON clause explicitly. |
| **2.** | It always returns unique columns in the result set. | It returns all the attributes of both tables along with duplicate columns that match the ON clause condition. |
| **3.** | If we have not specified any condition in this join, it returns the records based on the common columns. | It returns only those rows that exist in both tables. |
| **4.** | The syntax of natural join is given below: SELECT [column\_names | \*] FROM table\_name1 NATURAL JOIN table\_name2; | The syntax of inner join is given below: SELECT [column\_names | \*] FROM table\_name1 INNER JOIN table\_name2 ON table\_name1.column\_name = table\_name2.column\_name; |