SQL JOIN's

What is SQL JOIN’s?

In the expert level JOIN’S are more common SQL commands used in day to day life. JOIN’s are used to retrieving the records from multiple tables. SQL allows you to take JOIN’s on multiple tables from same database as well as different databases from same server.

Basically the tables are interrelated with each other using Primary & foreign key. So these keys are used in the JOIN’s to interlink two tables.

Why should we use JOIN’s?

Many times you are thinking “Why use SQL JOIN’s” as same task can be done using different queries. In the database queries are executed one by one & result of successive query can be use for next query. If we use the JOIN’s queries then instead of processing multiple queries SQL server process only single query which reduce the SQL server overhead. The main advantage of SQL JOIN’s is the improved performance. Also using multiple queries lead more data transfer from SQL server to application which reduces the performance.

Types of SQL JOINs:

Before we start learning SQL JOIN’s, we will see the list of different types of SQL JOINs:

- **INNER JOIN**: The INNER JOIN keyword selects all rows from both tables as long as there is a match between the columns in both tables (Table1 and Table2).
- **LEFT JOIN**: The LEFT JOIN keyword returns all rows from the left table (Table1), with the matching rows in the right table (Table2). The result is NULL in the right side when there is no match.
- **RIGHT JOIN**: The RIGHT JOIN keyword returns all rows from the right table (Table2), with the matching rows in the left table (Table1). The result is NULL in the left side when there is no match.
- **FULL JOIN**: The FULL OUTER JOIN keyword returns all rows from the left table (Table1) and from the right table (Table2). The FULL OUTER JOIN keyword combines the result of both LEFT and RIGHT joins.
Let's go ahead & learn different types of JOIN queries with practical example. Following are the three tables, Customers Product & Order.

<table>
<thead>
<tr>
<th>CustID</th>
<th>Name</th>
<th>Address</th>
<th>ContactNo</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sam</td>
<td>New Delhi</td>
<td>95555555555</td>
</tr>
<tr>
<td>2</td>
<td>Rahul</td>
<td>Gurgaon</td>
<td>96666666666</td>
</tr>
<tr>
<td>3</td>
<td>Hano</td>
<td>Noida</td>
<td>94444444444</td>
</tr>
<tr>
<td>4</td>
<td>Jeetu</td>
<td>Delhi</td>
<td>93333333333</td>
</tr>
<tr>
<td>5</td>
<td>Ankit</td>
<td>Noida</td>
<td>92222222222</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ProductID</th>
<th>Name</th>
<th>UnitPrice</th>
<th>CatID</th>
<th>EntryDate</th>
<th>ExpiryDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Dell Computer</td>
<td>25000</td>
<td>1</td>
<td>2012-10-16 23:05:05.550</td>
<td>2012-10-16 23:05:05.550</td>
</tr>
<tr>
<td>2</td>
<td>HCL Computer</td>
<td>20000</td>
<td>1</td>
<td>2012-10-16 23:05:46.990</td>
<td>2012-10-16 23:05:46.990</td>
</tr>
<tr>
<td>3</td>
<td>Apple Mobile</td>
<td>40000</td>
<td>2</td>
<td>2012-10-16 23:06:11.283</td>
<td>2012-10-16 23:06:11.283</td>
</tr>
<tr>
<td>5</td>
<td>Sony Laptop</td>
<td>35000</td>
<td>2</td>
<td>2012-10-16 23:06:52.143</td>
<td>2012-10-16 23:06:52.143</td>
</tr>
<tr>
<td>7</td>
<td>HP Printer</td>
<td>12000</td>
<td>4</td>
<td>2012-10-16 23:07:35.010</td>
<td>2012-10-16 23:07:35.010</td>
</tr>
<tr>
<td>8</td>
<td>Canon Printer</td>
<td>10000</td>
<td>4</td>
<td>2012-10-16 23:07:54.213</td>
<td>2012-10-16 23:07:54.213</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OrderID</th>
<th>ProductID</th>
<th>Quantity</th>
<th>Price</th>
<th>CustomerID</th>
<th>ContactNo</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>150000</td>
<td>1</td>
<td>95555555555</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4</td>
<td>80000</td>
<td>2</td>
<td>NULL</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>40000</td>
<td>3</td>
<td>94444444444</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>5</td>
<td>20000</td>
<td>4</td>
<td>93333333333</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1</td>
<td>35000</td>
<td>5</td>
<td>96666666666</td>
</tr>
</tbody>
</table>

**SQL INNER JOIN Keyword:**

The INNER JOIN is selects all rows from both tables as sql query match the specified condition.

**SQL INNER JOIN Syntax:**

```sql
1 SELECT column_name(s) 
2 FROM Table1 
3 JOIN Table2 
4 ON Table1.column_name=Table2.column_name;
```
or

1 SELECT column_name(s)

2 FROM Table1

3 INNER JOIN Table2

4 ON Table1.column_name=Table2.column_name;

In the query you can use JOIN or INNER JOIN, both are same.

**INNER JOIN Query Example:**

Suppose, the dealer want the list of order details like Product name, Unit Price, Quantity & Price. So in this case you can use INNER JOIN to get the records from both Product & Order tables. The ProductId is common in both tables. So in the INNER JOIN condition is added on ProductId & only matching records from Product & Order tables are returned.

```
2 FROM Product AS T1
```
INNER JOIN Order AS T2 ON T1.ProductID = T2.ProductID

ORDER BY T2.OrderID

Following is the result upon executing the above SQL INNER JOIN query:

All matching rows from both tables

<table>
<thead>
<tr>
<th>OrderID</th>
<th>ProductID</th>
<th>Name</th>
<th>UnitPrice</th>
<th>Quantity</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Dell Computer</td>
<td>25000</td>
<td>6</td>
<td>150000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>HCL Computer</td>
<td>20000</td>
<td>4</td>
<td>80000</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>HCL Computer</td>
<td>20000</td>
<td>2</td>
<td>40000</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>Apple Mobile</td>
<td>40000</td>
<td>5</td>
<td>200000</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>Sony Laptop</td>
<td>35000</td>
<td>1</td>
<td>35000</td>
</tr>
</tbody>
</table>

**SQL LEFT JOIN Keyword:**

**LEFT JOIN** returns all records/rows from left table and from right table returns only matched records. *Where no matches have been found in the table on the right, NULL is returned.*

**SQL LEFT JOIN Syntax:**

```sql
1 SELECT column_name(s) 
2 FROM Table1 
3 LEFT JOIN Table2 
4 ON Table1.column_name=Table2.column_name; 
```

or
LEFT JOIN query Example:

```
SELECT T2.OrderID AS OrderID, T1.ProductID, T1.Name, T1.UnitPrice,
       T2.Quantity AS Quantity, T2.Price AS Price
FROM Product AS T1
LEFT OUTER JOIN Order AS T2 ON T1.ProductID = T2.ProductID
ORDER BY T1.ProductID
```

Following is the result upon executing the above SQL LEFT OUTER JOIN query. You can see the all records from the Product table & NULL values are displayed for all records where no matching records found in Order table.
SQL RIGHT JOIN Keyword:

**RIGHT JOIN** is exact opposite to **LEFT JOIN**, it returns all records/rows from right table and from left table returns only matched records. Where no matches have been found in the table on the left, **NULL** is returned.

SQL RIGHT JOIN Syntax:

1. `SELECT column_name(s)`
2. `FROM Table1`
3. `RIGHT JOIN Table2`
4. `ON Table1.column_name=Table2.column_name;`

or

1. `SELECT column_name(s)`
2. `FROM Table1`
3. `RIGHT OUTER JOIN Table2`
4. `ON Table1.column_name=Table2.column_name;`
RIGHT JOIN query Example:

```sql
2 FROM Product AS T1
3 RIGHT OUTER JOIN Order AS T2 ON T1.ProductID = T2.ProductID
4 ORDER BY T1.ProductID
```

Following is the result upon executing the above SQL RIGHT JOIN query. You can see the all records from the Order table & NULL values are displayed for all records where no matching records found in Product table. **But if all records are matched then it will left table then no NULL records are returned in result set.**
**SQL FULL OUTER JOIN Keyword:**

**FULL OUTER JOIN** combines left outer join and right outer join. This join returns all records/rows from both the tables. If there are no columns matching in the both tables, it returns NULL values.

**SQL FULL OUTER JOIN Syntax:**

1. `SELECT column_name(s)`
2. `FROM Table1`  
3. `FULL OUTER JOIN Table2`  
4. `ON Table1.column_name=Table2.column_name;`

**FULL OUTER JOIN query Example:**

Suppose we want to get all Order records against all the Product records, then we can use the FULL OUTER JOIN script shown below to get all records from both tables.

```
```
FROM Product AS T1

FULL OUTER JOIN Order AS T2 ON T1.ProductID = T2.ProductID

ORDER BY T1.ProductID

Following is the result upon executing the above SQL FULL OUTER JOIN query:

<table>
<thead>
<tr>
<th>OrderID</th>
<th>ProductID</th>
<th>Name</th>
<th>UnitPrice</th>
<th>Quantity</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Dell Computer</td>
<td>25000</td>
<td>6</td>
<td>150000</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>HCL Computer</td>
<td>20000</td>
<td>4</td>
<td>80000</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>HCL Computer</td>
<td>20000</td>
<td>2</td>
<td>40000</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>Apple Mobile</td>
<td>40000</td>
<td>5</td>
<td>200000</td>
</tr>
<tr>
<td>5</td>
<td>NULL</td>
<td>Samsung Mobile</td>
<td>25000</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>Sony Laptop</td>
<td>35000</td>
<td>1</td>
<td>35000</td>
</tr>
<tr>
<td>7</td>
<td>NULL</td>
<td>Dell Laptop</td>
<td>36000</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>8</td>
<td>NULL</td>
<td>HP Printer</td>
<td>12000</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>9</td>
<td>NULL</td>
<td>Canon Printer</td>
<td>10000</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>

Conclusion on SQL JOINS:

- JOIN’s are used to combine & get the data from different tables.
- INNER JOIN returns rows when there is a match in both tables.
- LEFT JOIN returns all rows from the left table, even if there are no matches in the right table.
- RIGHT JOIN returns all rows from the right table, even if there are no matches in the left table.
- FULL JOIN returns rows when there is a match in one of the tables.
- The performances of JOIN's are better than sub queries.
- In the OUTER JOIN'S when no records are returned then NULL values are returned in the result set.
- JOIN queries can be used with the conjunction of SELECT, INSERT, UPDATE, DELETE commands.
- Also we can use different clauses along with JOIN's like GROUP BY, WHERE, SUB QUERIES, AGGREGATE FUNCTIONS etc.
Source:

http://www.softwaretestingclass.com/sql-joins/