**Week 4 - Database Technologies Assignment**

**1)List the type of joins. Explain every join type with an equivalent SQL statement.**

**1A)Types of SQL JOINS explained with examples: JOINS fundamentals**

In relational databases, such as SQL Server, Oracle, MySQL, and others, data is stored in multiple tables that are related to each other with a common key value. Accordingly, there is a constant need to extract records from two or more tables into a results table based on some condition. In SQL Server, this can be easily accomplished with the SQL JOIN clause

JOIN is an SQL clause used to query and access data from multiple tables, based on logical relationships between those tables.

In other words, JOINS indicate how SQL Server should use data from one table to select the rows from another table

**Different types of JOINS in SQL Server**

● INNER JOIN

● LEFT OUTER JOIN

● RIGHT OUTER JOIN

● SELF JOIN

● CROSS JOIN

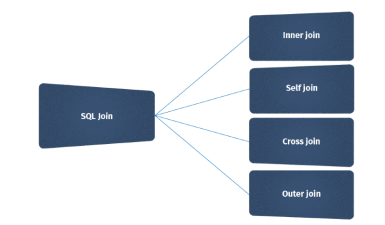
● **SQL INNER JOIN** creates a result table by combining rows that have matching values in two or more tables.

● **SQL LEFT OUTER JOIN** includes in a result table unmatched rows from the table that is specified before the LEFT OUTER JOIN clause.

● **SQL RIGHT OUTER JOIN** creates a result table and includes into it all the records from the right table and only matching rows from the left table.

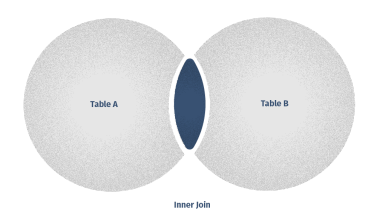
● **SQL SELF JOIN** joins the table to itself and allows comparing rows within the same table.

● **SQL CROSS JOIN** creates a result table containing paired combination of each row of the first table with each row of the second table.



**INNER JOIN**

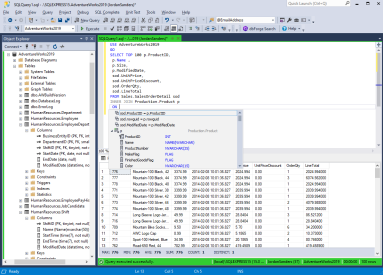
INNER JOIN statement returns only those records or rows that have matching values and is used to retrieve data that appears in both tables



In our example, we want to extract data from the Sales.SalesOrderDetail and Production.Product tables that are aliased with SOD for

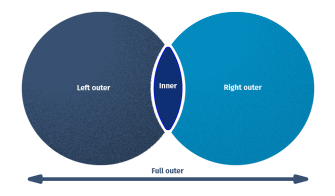
Sales.SalesOrderDetail and P for

Production.Product. In the JOIN statement, we match records in those columns. Make notice, how code suggestions work in SQL Complete

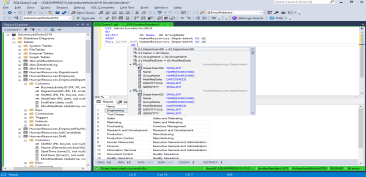


**OUTER JOIN**

When applying an SQL INNER JOIN, the output returns only matching rows from the stated tables. In contrast, if you use an SQL OUTER JOIN, it will retrieve not only the matching rows but also the unmatched rows as well.

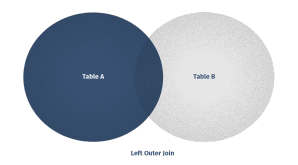


The FULL OUTER JOIN returns a result that includes rows from both left and right tables. In case, no matching rows exist for the row in the left table, the columns of the right table will have nulls. Correspondingly, the column of the left table will have nulls if there are no matching rows for the row in the right table.



**LEFT OUTER JOIN**

The LEFT OUTER JOIN gives the output of the matching rows between both tables. In case, no records match from the left table, it shows those records with null values.

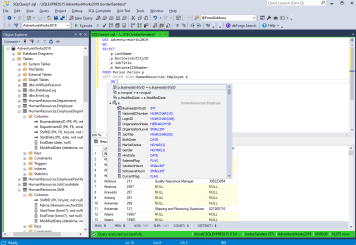


In our example, we want to join the tables Person.Person and HumanResources.Employee to

retrieve a list of all Person LastNames, but also show JobTitle if the Person is an Employee.

In the output, in case, there are no employees matching BusinessEntityID, NULL values will be listed in the corresponding rows for

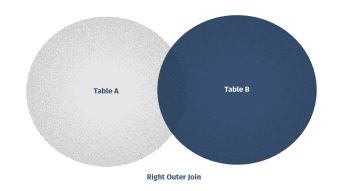
NationalIDNumber and JobTitle.

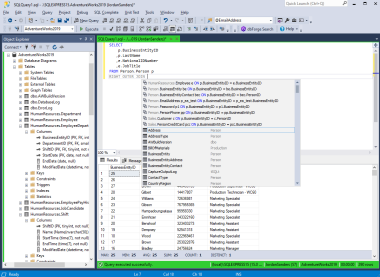


**RIGHT OUTER JOIN**

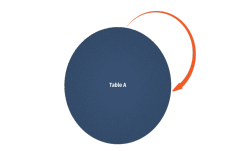
The RIGHT OUTER JOIN works by the same principle as the LEFT OUTER JOIN. The RIGHT OUTER JOIN selects data from the right table (Table B) and matches this data with the rows from the left table (Table A). The RIGHT JOIN returns a result set that

includes all rows in the right table, whether or not they have matching rows from the left table. In case, a row in the right table does not have any matching rows in the left table, the column of the left table in the result set will have nulls.

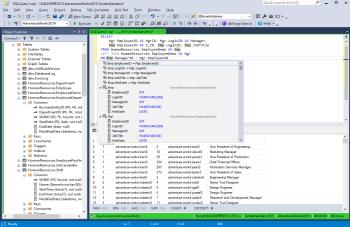


**SELF JOIN**

The SELF JOIN allows you to join a table to itself. This implies that each row of the table is combined with itself and with every other row of the table. The SELF JOIN can be viewed as a join of two copies of the same table. The table is not actually copied, but SQL performs the command as though it were. This is accomplished by using table name aliases to give each instance of the table a separate name. It is most useful for extracting hierarchical data or comparing rows within the same table.

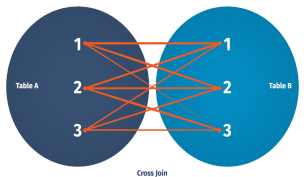


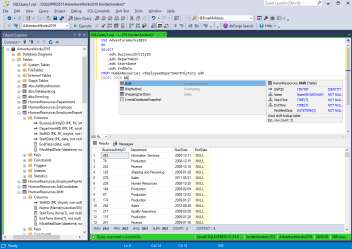
In our example, we want to retrieve a list of all the territories and the salespeople working in them from the Sales.SalesPerson table



**CROSS JOIN**

The CROSS JOIN command in SQL, also known as a cartesian join, returns all combinations of rows from each table. Envision that you need to find all combinations of size and color. In that case, a CROSS JOIN will be an asset. Note, that this join does not need any condition to join two tables. In fact, CROSS JOIN joins every row from the first table with every row from the second table and its result comprises all combinations of records in two tables.





**2)List and explain the set operations 2A)SQL Set Operations:**

The SQL Set operation is used to combine the two or more SQL SELECT statements.

Types of Set Operations

1.Union

2.UnionAll

3.Intersect

4.Minus



**1.Union**

The SQL Union operation is used to combine the result of two or more SQL SELECT queries.

In the union operation, all the number of datatype and columns must be same in both the tables on which UNION operation is being applied.

The union operation eliminates the

duplicate rows from its resultset

**Syntax**

SELECT column\_name FROM table1

UNION

SELECT column\_name FROM table2

**Example:**

**The First table**

| **ID** | **NAME** |
| --- | --- |
| 1 | Jack |
| 2 | Harry |

| 3 | Jackson |
| --- | --- |

**The Second table**

| **ID** | **NAME** |
| --- | --- |
| 3 | Jackson |
| 4 | Stephan |
| 5 | David |

**Union SQL query will be:**

SELECT \* FROM First

UNION

SELECT \* FROM Second**;**

The resultset table will look like:

| **ID** | **NAME** |
| --- | --- |
| 1 | Jack |

| 2 | Harry |
| --- | --- |
| 3 | Jackson |
| 4 | Stephan |
| 5 | David |

**2.Union All**

Union All operation is equal to the Union operation. It returns the set without removing duplication and sorting the data.

**Syntax:**

SELECT column\_name FROM table1

UNION ALL

SELECT column\_name FROM table2;

**Example:** Using the above First and Second table.

**Union All query will be like:**

SELECT \* FROM First

UNION ALL

SELECT \* FROM Second;

The resultset table will look like:

| **ID** | **NAME** |
| --- | --- |
| 1 | Jack |
| 2 | Harry |
| 3 | Jackson |
| 4 | Stephan |
| 5 | David |

**3.Intersect**

It is used to combine two SELECT

statements. The Intersect operation returns the common rows from both the SELECT statements.

In the Intersect operation, the number of datatype and columns must be the same.

It has no duplicates and it arranges the data in ascending order by default.

**Syntax**

SELECT column\_name FROM table1

INTERSECT

SELECT column\_name FROM table2;

**Example**:

Using the above First and Second table. **Intersect query will be:**

SELECT \* FROM First

INTERSECT

SELECT \* FROM Second;

The resultset table will look like:

| **ID** | **NAME** |
| --- | --- |
| 3 | Jackson |

**4. Minus**

It combines the result of two SELECT statements. Minus operator is used to display the rows which are present in the first query but absent in the second query.

It has no duplicates and data arranged in ascending order by default.

**Syntax**:

SELECT column\_name FROM table1

MINUS

SELECT column\_name FROM table2;

**Example**

Using the above First and Second table. **Minus query will be:**

SELECT \* FROM First

MINUS

SELECT \* FROM Second;

The resultset table will look like:

| **ID** | **NAME** |
| --- | --- |
| 1 | Jack |
| 2 | Harry |

**3)How to rename a column?**

**3A)Rename Columns in SQL Server:**

SQL Server allows us to change the column whenever we need. We will rename the table columns when the column name is non-meaningful or does not fulfill the

purpose of its creation. It must ensure that we have ALTER permission on the object before changing the column's name.

**Limitations and Restrictions:**

SQL Server has some restrictions while changing the column name because when we rename a column, it does not imply that all references to that column will be renamed as well. We must manually modify all objects that belong to the renamed column.

For example, if we want to change the column of a table that is also referenced in a trigger, it is required to modify the trigger for reflecting the new column name as well. The sys.sql\_expression\_dependencies can be used for listing all dependencies on the object before changing the name.

We can rename the table columns in SQL Server using mainly two ways:

1.Transact-SQL

2.SQL Server Management Studio (SSMS)

**1.Transact-SQL**

SQL Server provides a standard stored procedure called SP\_RENAME for changing the name of a user-created object in the current database. The user-created object can be a table, column, index, alias data type, etc.

Scripts and stored procedures may be broken when we change some portion of an object's name. We advise you to drop the object and re-create it with the new name rather than using this statement to change the name of stored procedures, triggers, user-defined functions, or views.

The syntax to change the column name using this approach is:

EXEC SP\_RENAME '[Table Name].[Old Column Name]', '[New Column Name]', COLUMN

**Rename Column Example**

The following example demonstrates the SQL Server rename column using the SP\_RENAME stored procedure. To do this, we will take a 'student' table that contains the below data:



If we want to change the 'city' column with the new name 'city\_name' of this table, we can use the above-specified SQL Server syntax or stored procedure as follows:

EXEC SP\_RENAME 'Student.city', 'city\_name', 'COLUMN'

After executing this script, we will get the following message:

When we verify the 'student' table, we can see that the column name of 'city' is changed successfully:



**2.SQL Server Management Studio (SSMS)**

SSMS is a windows software tool used to connect and work with our SQL Server from a graphical interface instead of using the command line. The management studio

allows us to rename the table columns in the following ways:

● Rename a column using Object Explorer ● Double click on the column name.

● Rename a column using Table Designer.

Let us discuss each of them in detail.

**Rename a column using Object Explorer:**

The following steps are used to change the column name using Object Explorer:

**Step 1:** Go to the Databases -> Tables menu and expand it.

**Step 2:** Select the desired table and expand it.

**Step 3:** Select the Columns and expand them.

**Step 4:** Right-click on the column name you want to change and choose the Rename option.

The below image explains all the above steps where we have chosen the 'student' table:



**Step 5:** Type a new name for your selected column.

**Step 6:** Refresh the database or table to finish the renaming steps.

**Step 7:** Execute the SELECT statement to verify the changed column name.

**Rename a column using Table Designer**

The following steps are used to change the column name using Object Explorer:

**Step 1:** Go to the Databases -> Tables menu and expand it.

**Step 2:** Select the desired table in which you want to change the column name, right-click on it, and choose the Design option from the context menu.



**Step 3:** Once we select the Design option, we will see the table in design mode like the below image.



**Step 4:** Select the column that you want to change and type the new name for this column. Let me rename the first\_name column to f\_name.

**Step 5:** To save the changes you made in the design window, click the Close button and then the Yes button in the displayed pop-up window.

We can save the changes made in the design window in another way by navigating to the File menu and click on the Save table name or press CTRL+S on the keyboard.

**Step 6:** Execute the SELECT statement to verify the changed column name

**Double click on the column name**

We can use the following steps to rename the column using a double click:

**Step 1:** Go to the Databases -> Tables -> Columns.

**Step 2:** Select the column name that you want to change and double-click. The below image explains it more clearly:



**Step 3:** Type a new name for your chosen column.

**Step 4:** Refresh the database or table to finish the renaming steps.

**Step 5:** Execute the SELECT statement to verify the changed column name.

**4)Assume you need to create a large table for collecting patient information and retrieving the available doctors for them in a multi-speciality hospital • Create a table with at least 20 attributes • Assign primary key for a specific column •Generate 20 tuples. • Now select the doctors and their specialisation: o match patients with doctors’ specialisation and retrieve**

**4A)**

****

**Assignment 2: Use the ER diagram for Healthcare Industry which yu have created**

**in previous assignment and construct a table consists of “CORONA” Prevention entities along with suitable attributes. Assume the various constraints and cardinality ratios among entities. 1.Now retrieve patients location wise 2. Now retrieve patients age wise 3. Now retrieve patients gender wise Now retrieve patients with other diseases wise**

**Note: Generate 20 matching records.**

**A)**

****