**Set ansi\_nulls on do -> sql command**

The SET ANSI\_NULLS ON statement is a SQL Server setting that determines how comparison operations behave with NULL values. When ANSI\_NULLS is set to ON, SQL Server follows the ANSI SQL standard for handling NULLs during comparisons. Here's what it means:

**Comparisons with NULL:**

With SET ANSI\_NULLS ON, equality and inequality comparisons with NULL values using the = and <> operators return UNKNOWN instead of TRUE or FALSE.

For example, if you have a query like WHERE column1 = NULL, it will not return rows where column1 is NULL.

**Concatenation with NULL:**

When ANSI\_NULLS is ON, concatenating a NULL value with a string using the + operator results in a NULL value.

Aggregate Functions and NULLs:

Aggregate functions like COUNT, SUM, AVG, etc., exclude NULL values in their calculations when ANSI\_NULLS is ON.

**Indexed Views:**

Indexed views (materialized views) require SET ANSI\_NULLS ON for indexed views to be created.

It's important to note that the ANSI\_NULLS setting is specific to Transact-SQL (T-SQL) and Microsoft SQL Server. Different database systems may have different behaviors regarding NULLs and may or may not adhere to the ANSI SQL standard.

By default, ANSI\_NULLS is usually set to ON in modern versions of SQL Server, but it's always a good practice to explicitly set it to avoid any potential compatibility issues. However, keep in mind that changing this setting can affect the behavior of existing queries, so it's essential to test the impact on your specific queries and applications.