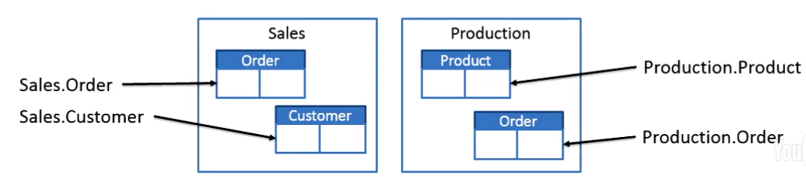
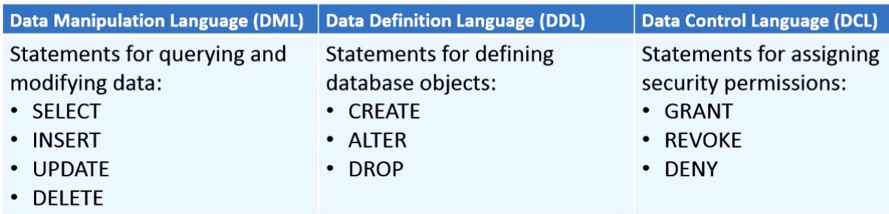
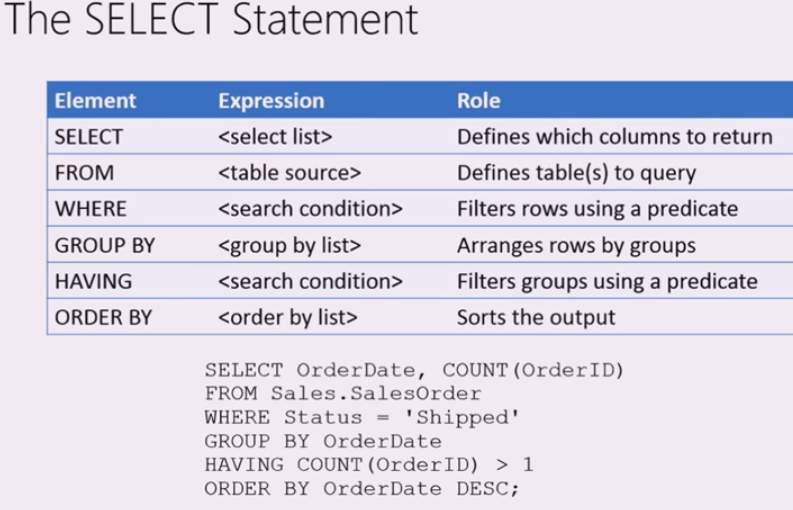
Set based, Entities(tables), row is a instance of that entity, primary key uniquely identifies a row(entity instance) in the table, schema is namespace for db objects like tables, [srvr name][db name][schema name][object name]. two different schemas can have tables with same name

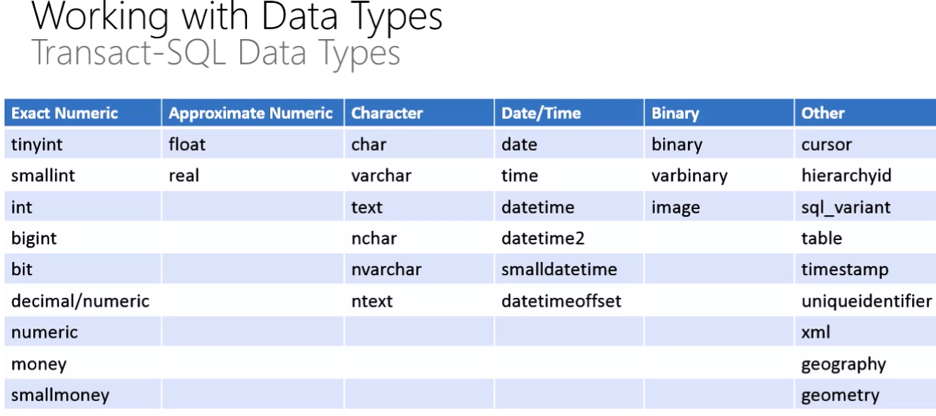




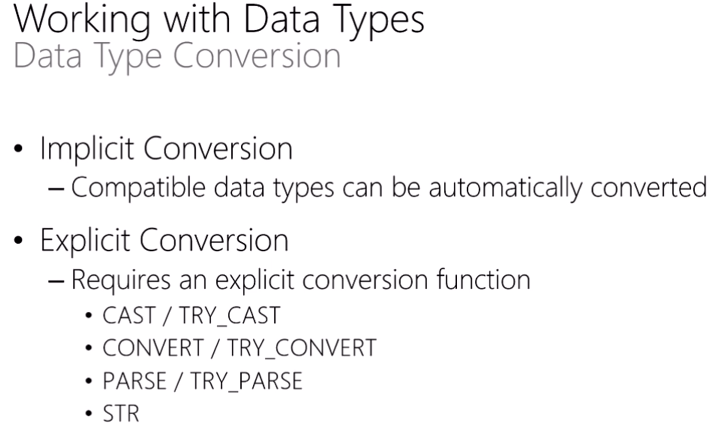
Select total number of orders grouped by order date where there is more than order.



Specify the cols u want explicitly instead of \* to reduce network traffic



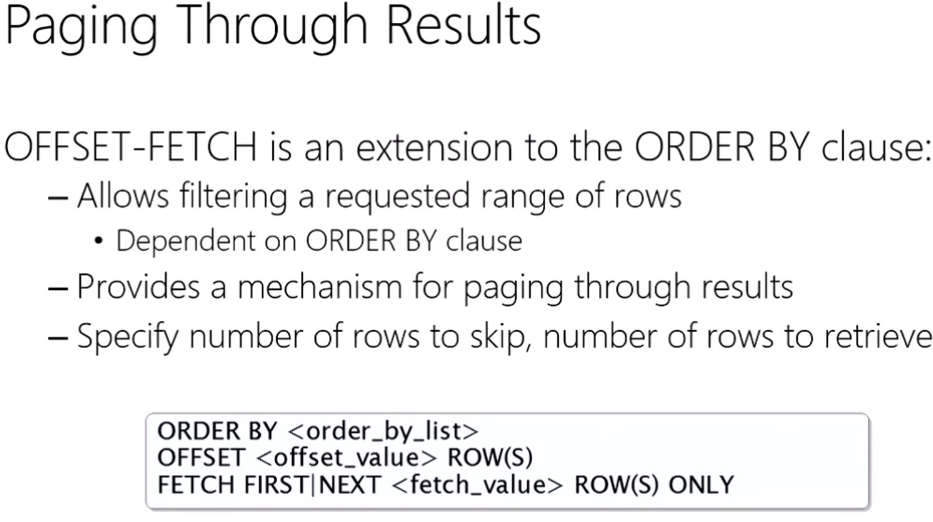
Char and tinyint uses 1 bytes. Nchar uses 2 bytes(equal to wchar in c++)



Cast and convert are pretty much the same. Prefer convert as it has more options for date conversion. parse and str are opposites. Nulls mean unknown or missing value but u can change the default in sql server to get them to mean 0 or empty space. Name is null, isnull (name, value), coalesce selects first not null from a list, case has two forms: one in which the col is outside the when statements and other when it is inside the when statements.

* NULL is used to indicate an unknown or missing value. NULL is **not** equivalent to zero or an empty string.
* Arithmetic or string concatenation operations involving one or more NULL operands return NULL. For example, 12 + NULL = NULL.
* If you need to compare a value to NULL, use the **IS** operator instead of the **=** operator.
* The **ISNULL** function returns a specified alternative value for NULL columns and variables.
* The **NULLIF** function returns NULL when a column or variable contains a specified value.
* The **COALESCE** function returns the first non-NULL value in a specified list of columns or variables).

I think the top function in select statement runs after the order by clause…So order by is not exactly the last thing to happen in all circumstances.???



--THESE QUERIES ARE SAME

SELECT

Color

FROM Production.Product

WHERE Color IS NOT NULL

GROUP BY Color;

SELECT

DISTINCT Color

FROM Production.Product

WHERE Color IS NOT NULL

--THESE QUERIES ARE SAME ARE NOT SAME!!

SELECT

COUNT(Color)

FROM Production.Product

WHERE Color IS NOT NULL

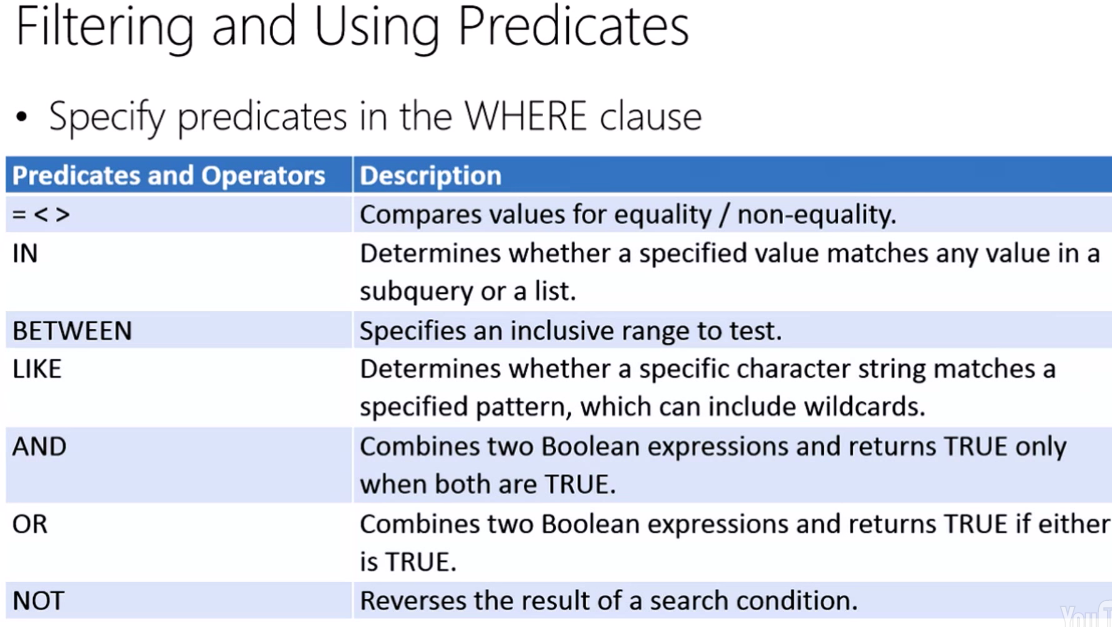
GROUP BY Color;

SELECT

COUNT(DISTINCT Color)

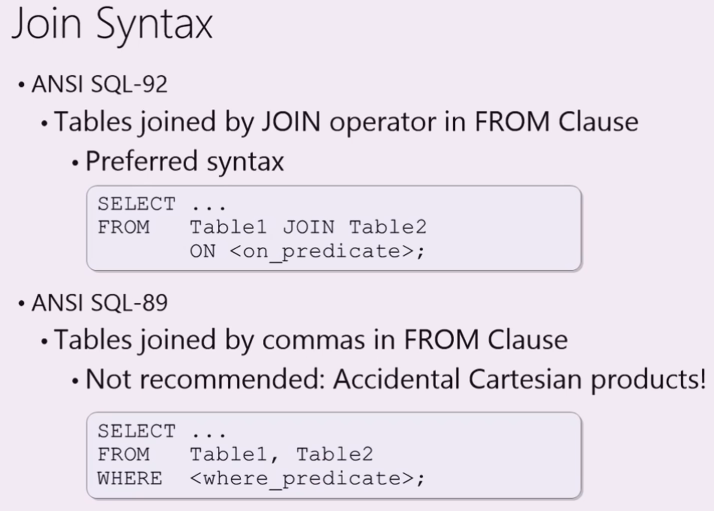
FROM Production.Product

WHERE Color IS NOT NULL



We could use IN once or OR multiple times for the same logic. Between in inclusive but could be replaced with <= and >= which is more readable.

Database can be configured to be case-sensitive or not case-sensitive.



* Joins are used to match rows in one table to rows in another table.
* The query engine supports two ways to define joins: the ANSI SQL-92 syntax (in which the join is specified in the FROM clause) and the older ANSI SQL-89 syntax (in which the join is specified in the WHERE clause). The ANSI SQL-92 syntax is the preferred approach.
* Use a Left Outer Join to include all rows from the first table and values from matched rows in the second table. Columns in the second table for which no matching rows exist are populated with NULLs.
* Use a Right Outer Join to include all rows from the second table and values from matched rows in the first table. Columns in the first table for which no matching rows exist are populated with NULLs.
* Use a Full Outer Join to include all rows from the first and second tables. Columns in the either table for which no matching rows exist are populated with NULLs.