# Joins

* It allows us to bring together data from multiple tables.
* Mostly used for tables that have existing relationships.
  + But you can also join tables that does not have relationship.

## Inner join

* Return rows with matching values in both tables.

## Left join

* Returns every row from the left table and returns some row from the right table with matching values.

## Right join

* Returns every row from the right table and returns some row from the left table with matching values.

## Full join

* Return every row from both tables.

## Cross joins

* Returns the Cartesian product of the joined tables.
* Basically, will return all possible combination of all rows from each table.

# Seeding

* It is just the process of populating your database with some initial set of data.
* So just doing insert into statements for tables/adding data.

# Subquery

* Allows you to add a query that is nested inside another query such as select, insert, delete, and update.
* You can think of it as a querying result so that we can use it.

# Introduction to T-SQL

* It is superset of SQL.
* It is Microsoft’s way to extending the functionalities of SQL.

## Set Operations

* Another way of joining your tables or queries together.
* Biggest different is that they do not need to compare columns unlike join.
* Another difference is that the number and order of the columns (same data type) must the same in all queries

## Union

* It will give you all the rows from both queries **except** duplicated rows.

## Union all

* It will give you all the rows from both queries **including** duplicated rows.

## Except

* It will give you all the rows from the left query that is not included on the right query.
* Basically, it will give you the distinct rows the left query has that the right query doesn’t have.

## Intersect

* It will give you only the duplicated rows between the left and right query.
* The opposite of Union, it just gives you the duplicated rows instead.

# Function

* Also known as User Defined Functions
* They are like methods but with different restrictions and capabilities.
* You can perform multiple query statement like a transaction.
* You cannot start a transaction in a function.
  + Main reason being is that function cannot modify the database state.
* Mostly used to do some algorithm logic that can be re-usable (think of pre-build functions like avg, sum, count)

## Different types of Function

* Scalar function
  + When a function takes one, or more parameters and returns a single value.
* Tabular function
  + When a function returns a table data type.
  + You can treat the return function as a normal table, so you do DML, DQL operations on it if you want.

# Stored Procedures

* Almost like a function except has certain unique characteristics about it.
* It can accept input parameters but also output parameters.
  + Output parameters is what the stored procedure will return.
* You can also have optional parameters.
* Can perform operations on the database.
* You can also return a special output parameter that will indicate if the stored procedure was successful or not.
* It is reusable.

# Triggers

* They are a special type of stored procedure that will run when a certain event happens such as insert, update, or delete queries.
* You can then specify when you want the trigger to happen such as before an insert query or after an insert query.

## Different types of triggers

* DML trigger – they will do some sort of operations whenever a DML event happened on a specific table.
* DDL trigger – they will do some sort of operations whenever a DDL event happened.
* Logon trigger – they will run whenever a user log in to the server.

# Views

* They are virtual tables that is used to present data to a user.
* Let us you share only certain data from a table.
* They let you also alias the column names to hide that information from the user that is using the view.

# CTE

* Stands for Common Table Expression
* Note: CTE it is from SQL and not T-SQL
* It allows you to temporary create a table for you to do DDL, DML, etc.
* Very useful for doing multiple SQL queries for a generated result set.

# ACID Properties

* A set of properties of database transactions that is intended to guarantee validity even in the event of errors or power failures.
* Basically, they ensure that your database won’t be corrupted if your database crashed while doing DML, DDL operations.

## Atomicity

* Either all statements should execute or none of them will.
* Meaning all statements inside the transaction will execute and persists in the database or none of them will.

## Consistency

* There should be a transparent consistency in your database.
* Data should be consistent before and after a transaction.
  + Ex: $100 - $30 checking account to put on your saving account then
  + $70 checking account and $30 on your saving account
  + It should not result into $70 checking and $40 on your saving account.

## Isolation

* The state of a transaction should be invisible to the other transactions.
* Cannot access the result of the other transaction until the transaction is completed.

### Different bad phenomenon that occurs during concurrent transactions

* Dirty read – reading data that has not been committed.
* Non-repeated read – When data was read twice but comes out different on each time.
* Phantom read – When data is added or removed by another transaction.

### Degrees of Isolation

* Different levels of isolation hat change the interaction between transaction.
* Read uncommitted – does not protect the transaction from anything.
* Read committed – Checks if the data being read is committed by the second transaction.
* Repeatable read – Force the second transaction to wait for the first transaction to update the data.
* Serializable – Forces the second transaction to wait for an insert or delete query to be finished from the first transaction.
  + Biggest con with the highest level of isolation it will essentially prevent concurrent transactions to be happening (really slowing down your database)

## Durability

* Once a transaction completes, the changes it did to the database is permanent.
* Even if there was a system failure, that data will still be there and not removed.