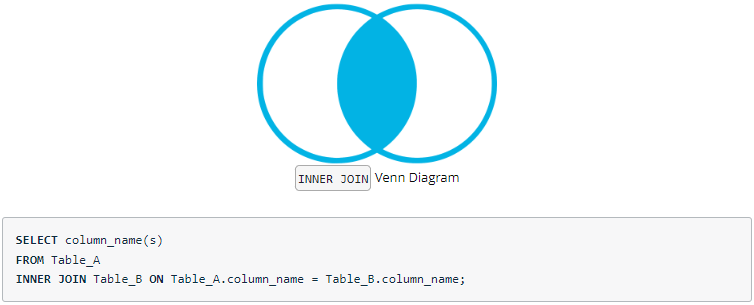
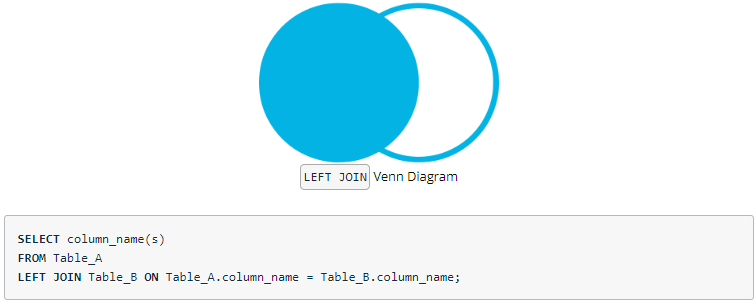
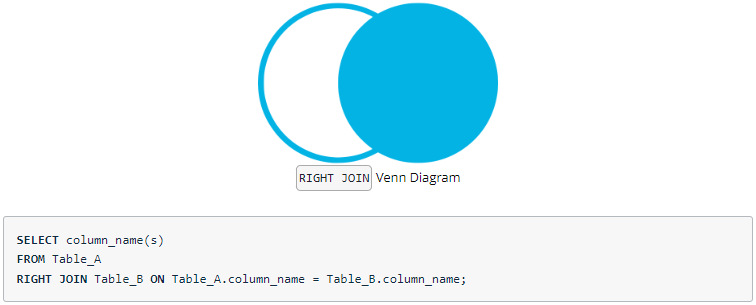
**INTRODUCTION ADVANCED SQL**

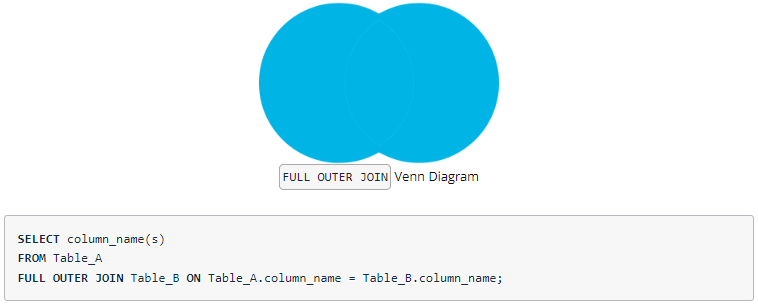
**FULL OUTER JOIN**

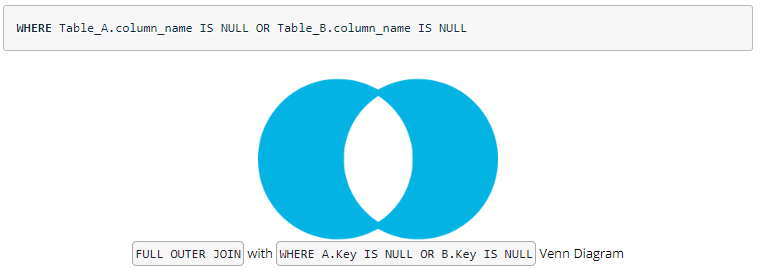
Antes de nada vamos a repasar los tipos de JOIN que hemos visto hasta ahora con los diagramas Venn:











Ver ejercicios en SQL\_7\_introductionAdvancedSQL.sql

**UNION**

Este statement nos permite unir las filas de dos tablas, pero se deben cumplir dos condiciones: ambas tablas deben tener el mismo número de columnas y cada columna debe tener el mismo de tipo de dato en cada tabla.

Algo a tener en cuenta es que UNION no duplica los datos, es decir, si se encuentra dos filas en cada tabla que son exactamente iguales, esta fila igual solo aparecerá una vez en la tabla resultante. Se puede usar UNION ALL para duplicar los datos. Esto tiene sentido porque normalmente si se usa UNION es porque se guarda la misma información en dos tablas distintas y hay veces que para hacer los análisis hay que juntar todos los datos desde distintas tablas.

**PERFORMANCE TUNING**

Hay formas de hacer que las consultas se hagan de forma más rápida reduciendo la cantidad de cálculos que se hace. Tres aspectos que influyen mucho en este tiempo son:

-Tamaño de la tabla

-Joins

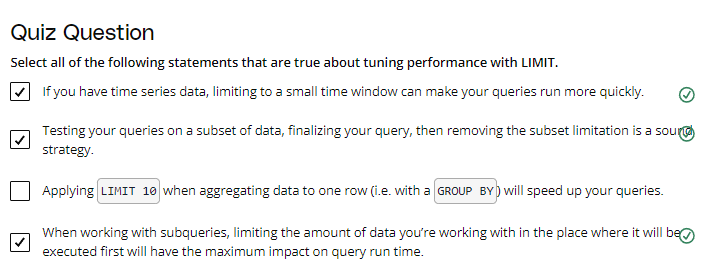
-Agregaciones

Por ejemplo, conviene hacer limitaciones (fechas, valores, etc), especialmente si trabajamos con subquerys. También conviene hacer las agregaciones en subquerys u otras tablas antes de hacer un JOIN para reducir considerablemente el número de filas con el que el JOIN tiene que trabajar.

También depende de otros factores que no podemos controlar:

-Que haya otras personas ejecutando consultas en la base de datos

-La optimización del software (por ejemplo Postgres está optimizado para leer y escribir nuevas filas, en cambio Redshift para ejecutar más rápido las agregaciones).



EXPLAIN

Si ponemos esto antes de la consulta nos explica los pasos que se van a seguir en la ejecución.