**Introducción a Consultas a bases de Datos**

**Sobre la importancia de las consultas (o Queries) en una base de datos relacional.**

Las consultas son la parte fundamental de una base de datos. De qué te sirve hacer una consulta o un Query efectivo?

Pues bien, una consulta bien hecha podría, incluso, *salvar un negocio o una empresa.* Las empresas tienen una gran necesidad por información correcta, exacta, oportuna; dicha información, se puede extraer generalmente de las bases de datos que ya tenemos (de aquí la importancia de las *consultas*).

En las bases de datos tenemos muchas veces *clientes, productos, proveedores* y diferentes tipos de entidades que, por separado, no tendrían mucho sentido; pero, cuando se comienzan a unir, a través de Queries, puedes presentar: un informe, o una tendencia, etc...; de tal forma que, las personas que están encargadas de *maniobrar* la empresa para un lado o para el otro,... tengan la información que necesitan para tomar las decisiones correctas.

Entonces, alrededor de toda esa cultura de “*extraer y transformar información”,* se han creado varias especialidades en el esquema de datos que tiene que ver con, por ejemplo, ***ETL*** (transformación de datos), ***Business Intelligent*** (extracción & utilización de datos para la toma correcta de decisiones) e, incluso, que tiene que ver con: ***Machine leearning*** (toma los datos existentes de una base de datos y los presenta de tal manera que haga sentido, que sea coherente y, sobre todo, que sea útil).

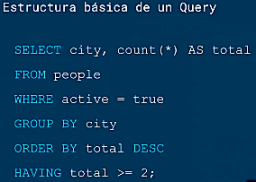
*En resumen, los Queries son una parte fundamental de una base de datos;* pues, nos permiten conocer las estructuras de los **Query** y resolver cualquier duda que tenga cualquier miembro de la empresa que pueda ser trabajada a través de una base de datos.

En resumen, con los ***Queries*** es que se brinda valor a una organización al final del día; pues, es lo que nos trae la información, *el diario vivir dentro de una organización*; con lo cual, es que se podrán tomar decisiones de gestión e inversión diariamente. Entonces, para traer toda esa información de nuestro interés, de una base de datos, necesitas hacer ***Queries.*** Los ***Queries*** son la forma en la que estructuramos las preguntas que le vamos a hacer a la base de datos; es decir, al final del día, nuestro camino a seguir para encontrar todas las respuestas.

Ahora, lo complicado es que hay muchas formas de hacer un ***Queri,*** desde lo más básico a lo más complejo.

***Vamos de las Queries básicas a las complejas.***

Empecemos a practicar la habilidad de convertir nuestras preguntas en ***Queries.*** El ***Queri*** tiene básicamente dos partes: la sentencia ***Select*** y la sentencia ***From*** y, en varias ocasiones, nos encontramos con la sentencia ***Where.***



***Select*** nos permite traer los datos que queremos mostrar, todos ***(\*)*** o de alguna ***columna*** en especifica; en este caso puntual, queremos traer los datos de la columna o atributo ***city*** y una cuenta ***count(\*)*** la cual se proyecta como ***total*** (en adelante explicaremos más a fondo los dos últimos elementos señalados en negrilla).

*Un dato:* En caso que usted no sólo desee ver los datos de una única columna, de una tabla, puede agregar más. Simplemente, dentro de la misma sentencia ***Select,*** debe llamar al otro u otros campos en cuestión por medio de separaciones de coma; por ejemplo: ***SELECT city, name*** (Ahora veríamos, ademas de ***city,*** la columna ***name).***

En ***From*** indicamos de dónde vamos a seleccionar esos datos que queremos mostrar con el ***select;*** es decir, más precisamente, de que ***tabla***; en contexto, entonces, estamos indicando que queremos los datos de la columna o atributo ***city*** de la tabla ***people.***

Ahora, en ***Where,*** el filtro de la consulta y recolección de datos se vuelve más exigente; pues, con ***where,*** lo que se establece es una condición; en consecuencia, dicha condición debe cumplirse para poder seleccionar los datos deseados de una columna para una tabla en especifica. En contexto, cuando definimos un where con ***active = true*** le estás diciendo a nuestra base de datos de trabajo actual que desea solamente los datos de la columna ***city*** de la tabla ***people*** que cumplan con la condición ***active = true.*** De hecho, ***active*** vendría siendo también otra columna o atributo de la misma tabla ***people*** y ***true*** es uno de los posibles valores que puede tener, dentro de dicha tabla, la columna ***active.***

***Group by*** lo que nos permite es agrupar los datos, justamente, según un criterio pasado; en este caso, el criterio pasado fue: ***city;*** es decir, vamos a agrupar los datos de la tabla ***people***, que cumplen con la condición ***active = true*** y son de la la columna o atributo ***city,*** dentro de la misma columna ***city*** (nos mostrará la info de ***city***).

***Order by*** define de qué manera se organizarán los datos recolectados; en este caso, nos ayuda a definir de qué manera ordenar los datos seleccionados de la columna ***city***, de la tabla ***people***, que cumplen con la condición ***active = true*** (la totalidad de ellos); por ejemplo, definir el orden de los datos seleccionados de tal manera que: se *enlisten de mayor a menor* las ciudades ***(city)*** según la cantidad de personas que tenga cada una.

Por último, en ***Having,*** sedefine otro parámetro de limitación, va a aplicar otro filtro o condición mucho más especifico; en este caso, va a filtrar (elegir) el ***total*** de los datos seleccionados de la columna ***city***, de la tabla ***people*** y que tengan un ***active = true;*** pero, además, *cuando haya 2 o más de 2 personas en cada ciudad* ***(total >= 2).*** Tengo la sensación que ***total*** es otra columna o atributo de la tabla ***people.***

***un dato:*** La sentencia ***Select \**** significa *selecciona todos los datos;* sin embargo, qué es *todo?* SQL interpreta esta solicitud, por sí sola, como muy ambigua y tiene razón (por lo cual te lanza un error). Es decir, a ***Select \**** se le debe acompañar en todos los casos con otra sentencia. Por ejemplo, evaluemos nuestro primer caso de ***Query:***



En efecto, con esto estamos diciendole a la base de datos que nos traiga absolutamente todo ***(SELECT \*)***; pero, recuerde, falta algo más,... Queremos todo, sí, pero de la tabla ***posts (FROM posts),*** que nos muestre absolutamente todo de ahí (sus campos, columnas o atributos).

De hecho, generalmente, te encuentras las ***Queries*** con una tercera sentencia declarada que es, como te contabamos, ***Where;*** que nos permite filtrar cuáles de esos datos, de la tabla ***posts***, queremos ver.



En este caso, sólo hemos filtrado los datos de la tabla ***posts*** que tienen como fecha de publicación una fecha igual o superior al año ***2024.*** Es decir, la base de datos nos arrojará todas las noticias, ***posts,*** que sean del ***2024*** para ***adelante*** en su ***fecha\_publicacion.***

***Nota:*** *para SQL “>” es igual a “>=”, por lo que en el caso anterior se incluye también al año 2024. Sin embargo, “<” no es lo mismo que “<=”.* **En resumen, con *Queries* y *Select,* filtramos información.**

En adelante, probaremos ***Queries*** más complejas según el nivel de complejidad que tenga, propiamente, la consulta de los datos que se desean ver; que, obviamente, deberían de ser en la medida unos datos con unas caracteristicas más puntuales o con mayor especificaciones. Sin embargo, para hacer consultas, se requiere tener primeramente datos; dicho eso, empecemos por agregar datos de prueba a nuestras tablas de modelación.

***Propiedad AS***

Ahora, existe una propiedad que no es propiamente de ***Select,*** pero que sirve de asistencia a las consultas o ***Queries*** que hagamos con ***Select;*** esa propiedad es: ***AS.***

***AS*** nos permite pasarle un *alias,* o nombre temporal *que no sustituye de manera permanente al nombre original,* a una o varias columnas; esto, si se acompaña con la sentencia ***Select***; es decir, ***AS*** le pasaría un nombre temporal, cambiando temporalmente el nombre original, a una o varias columnas seleccionadas a la hora de querer hacer una consulta de ellas con ***Select***. Es decir, ***AS*** nos permite sustituir temporalmente, por lo menos sólo en el momento de hacer propiamente la consulta, el nombre original de una o de varias columna; de tal manera que se le asigne un nombre temporal a la columna (o columnas) de nuestro interés.

El proceso es más o menos así: Por medio de ***Select,*** se llama a la *o a las columnas* de nuestro interés con su nombre original (no se olvide hacer separaciones por coma por cada columna llamada)**;**posteriormente, se le debe pasar a cada nombre por separado, e inmediatamente después del nombre de cada columna llamada, la sentencia ***AS;*** entonces, luego de declarar ***AS,*** se le pasaría un nuevo nombre (que sería el *Alias*), el de nuestro interés, a cada columna llamada en cuestión,... tal que así, por ejemplo:



Advertir que lo único que cambia, del *dataset,* y de forma *temporal,* es el nombre original al hacer una consulta con ***Select;*** del resto, todo sigue igual, los mismos datos para las mismas columnas: sólo que ahora está viendo que esas columnas afectadas están siendo llamadas con otro nombre; los cuales, fueron pasados con la propiedad ***AS.***

**Ahora, algo interesante.** En adelante, de hecho, usted podrá llamar cualquiera de las dos formas a sus columnas de interés para manipular sobre ellas, ya sea que las llame con su nombre original o por medio del *Alias* recientemente pasado. Por ejemplo, en futuras ***Queries*** más complejas puedes referirte a los campos en cuestión, ya no con su nombre original; sino, también, con su *Alias* asignado (si así lo deseas).

***~~Group by (en edición)~~***

~~Ahora vamos a introducir la función de~~ *~~Agrupación~~*~~, algo propio de la sentencia~~ *~~Group by~~* ~~(un tema que veremos en adelante; sin embargo, ahora sólo nos interesa hacer el tipo de~~ *~~Agrupación~~* ~~más básico).~~

***Propiedad Count***

***Select,*** también, nos permite hacer datos que se crean en el momento y que nos pueden agregar información; un buen ejemplo de eso sería combinar el ***Select*** con el uso de la sentencia ***Count(),*** ¿y qué hace ***Count()***?

La sentencia ***Count()*** se encarga de contar, ¿contar qué? Contar ***Registros.*** Es decir, a partir de unos parametros pasados para ***Count(),*** éste últimose encargará de contar la totalidad de los registros que hay dentro de una tabla, si y sólo si, se cumplen a cabalidad unas condiciones establecidas según dichos parametros pasados en cuestión, ¿qué parametros? Pues, en principio, los que establecemos dentro de los parentesis de ***Count().***

El ejemplo más básico de uso de la sentencia ***Count(),*** en connivencia con la sentencia ***Select,...*** es que en ***Count()*** se le pase, como parametro, que cuente absolutamente todos (sin filtro alguno) los registros existentes para una tabla seleccionada con la sentencia ***From***; entonces, en este caso, ese “nuevo dato que se crearía en el momento y que nos podría agregar información” sería el dato númerico, que se nos arroja, equivalente a la totalidad de registros que se alcanzarón a contar, en su totalidad, para una tabla seleccionada en cuestión.

Pongamos, por ejemplo, el siguiente caso: 

Esto nos arrojaría un nuevo dato, un simple número, que reflejaría el conteo total de los *registros* que tenemos en nuestra tabla *posts;* es decir, prácticamente, la cantidad de *posts* que tenemos publicados. Tal que así:

***, 22*** sería el número de *registros,* o posts, que tenemos en nuestra tabla *posts* parcialmente.

Ahora, si se da cuenta, ese nuevo dato que se ha creado, el dato númerico, está sustentado por un nuevo atributo, campo o columna que también ha sido creado; esa nueva columna si se percata se llama: ***COUNT(\*)***

Luego, se dará cuenta que, a la nueva columna resultante ***COUNT(\*)*** se le puede pasar un *Alias* tambiéncon la sentencia ***AS;*** esto para recordar o identificar, a dicha columna, de una manera más facil e *identificativa*; pues, es *engorroso* tener que llamar a una columna con una palabra clave como ***Count(\*****).* Probemos:



Esto quiere decir que, nuestra columna inicialmente llamada ***COUNT(\*),*** ahora puede ser llamada también como: ***numero\_posts***; lo cual, tendría mucho más sentido pues ***numero\_posts***, como nueva columna, nos estaría mostrando justamente el número de postsque tiene la tabla ***posts***,... Quedando así:



Por supuesto, hay que decir que realmente este dato no está almacenado de forma permanente en la base de datos. Con todo y eso, lo podemos aún así consultar por medio de ***Select.***