SQL APUNTES:

Algunos conceptos teóricos…

- Toda consulta posee como mínimo un comando y una cláusula.
- **Comando**: El que usamos principalmente es SELECT (la consulta comienza con dicho comando), otros comandos son… "INSERT, UPDATE, DELETE, REPLACE"
- **Cláusula**: Las condiciones que modifican los datos. La más usada FROM. Otras útiles son … "WHERE, GROUP/ORDER BY Y HAVING"
- **Predicado**: Función que devuelve true o false.
- Una consulta SQL DEBE acabar con punto y coma.
- **Programación imperativa:** Se modifica el código de forma totalmente manual, para conseguir que el programa realice un propósito. El programador define y crea las estructuras y variables que están implementadas para que dicho código funcione.
- **Programación declarativa:** La escritura se basa en la descripción del problema a solucionar generalmente mediante funciones matemáticas. El programador no resuelve el problema directamente, si no que mediante el uso de las funciones adecuadas este se resuelve "solo".
- **Sublenguajes SQL:** Son seis, los siguientes:
    1. **DDL**: Permite a los usuarios llevar a cabo las tareas de definición de las estructuras que almacenarán los datos así como de los procedimientos o funciones que permitan consultarlos.
    2. **DML**: permite a los usuarios llevar a cabo las tareas de consulta o modificación de los datos contenidos en las Bases de Datos del Sistema Gestor de Bases de Datos
    3. **DQL**: DQL permite expresar consultas en función de entidades y las relaciones entre ellas
    4. **DCL**: incluye una serie de comandos SQL que permiten al administrador controlar el acceso a los datos contenidos en la Base de Datos.
    5. **SCL**: (Session Control Language.)
- **Funciones de agregado:** Se añaden después de un campo y este va entrecomillado ejemplo: SELECT nombre, SUM(libros_leidos). Al hacer esto generamos una columna con dicho nombre. Esto es un problema si queremos agrupar en un orden, ya que en el ORDER BY habrá que poner ese nombre y no otro. **(Puedes solucionarlo cambiándole el nombre con AS).** Las más importantes…
    1. AVG: Calcula la media de un campo.
    2. COUNT: Cuenta los registros de un campo.
    3. SUM: Suma los valores de un campo.
    4. MAX: Devuelve el máximo de un campo.

5. MIN: Devuelve el mínimo de un campo.

- **SELECT:** Haces referencia a los tipos de datos que quieres consultar. Por ejemplo, si haces … SELECT country FROM world; Sacarás todos los países registrados en la tabla WORLD.
- **FROM:** Especificas la tabla de la que quieres sacar los datos.
- **HAVING/WHERE:** Sirven para lo mismo pero el WHERE siempre se utiliza antes de un ORDER BY mientras que el HAVING va después de este. Ambos sirven para dar condiciones y así mostrar la información deseada.
- **ORDER BY:** Ordenamos por defecto de menor a mayor tanto numérica como alfabéticamente. Se especifica la columna sobre la que tomamos referencia para ordenar. Si agregamos después de la columna "DESC" ordenamos pero de forma invertida, es decir empezando por el mayor.
- **GROUP BY:** Especificamos como queremos agrupar cierta información. **Es distinto de ORDER BY.** Un ejemplo sería si queremos sacar la población total de cada continente. Tendriamos que hacer la suma de la población de los países y luego utilizar el GROUP BY para agruparla en continentes.
- **Operadores:**
    1. **<:** Menor que
    2. **>:** Mayor que
    3. **=:** Igual que
    4. **>=:** Mayor o igual que
    5. **<=:** Menor o igual que
    6. **<>:** Distinto de
    7. **BETWEEN:** Entre (dos valores). Equivale a usar dos comparadores mayores/menores **o iguales.**
    8. **LIKE:** Como
    9. **IN:** En (usada cuando nos referimos a varios valores de un mismo tipo)
    10. **AND: "**Y" para encadenar posibilidades.
    11. **OR: "**O**"** también encadena posibilidades, la diferencia es que con el AND deben cumplirse **ambas.** Con el OR no.
    12. **NOT:** "Negación". Es común su uso junto a LIKE, para referirnos a los valores que NO queremos sacar (NOT LIKE).
- **Funciones más utilizadas:** Aunque hay muchas mas estas son las más útiles hasta el momento …
    1. **ROUND(**Valor a redondear, numero de Decimales**):** Si te mandan redondear para el otro lado, pones un negativo (a las mil = -3)
    2. **CONCAT(**"un string", "otro"**):**
    3. **DISTINCT(**"Lo que quieras que salga una vez"**):**
    4. **LENGTH(**Valor del que quieras obtener el numero de caracteres**):**
    5. **LEFT(**lo que quieras meter, numero de caracteres que quieres sacar**):**
    6. **NOW():** Te saca la fecha actual
    7. **REPLACE(**lo que sea, lo que quiero sustituir, por lo que sustituyo**):**

- **Subconsultas:** Consiste en hacer una consulta dentro de otra para referirse a valores específicos, estos e entiende mucho mejor con los ejercicios. Un ejemplo…

```
SELECT name FROM world
 WHERE population >
   (SELECT population FROM world
    WHERE name='Russia')
```

  Sacaría los países con una población superior a la de Russia
- **JOINS:**
  1. **INNER JOIN:** Refleja la información común a dos tablas relacionadas. Por ejemplo, si hay una tabla Clientes y otro Pedidos y un cliente puede hacer varios pedidos, con el join se reflejarán los clientes que hayan hecho los pedidos.
  2. **LEFT JOIN:** Devuelve todos los registros de la tabla de la izquierda + los comunes a ambas
  3. **RIGHT JOIN:** Devuelve todos los registros de la tabla de la derecha + los comunes a ambas
- **NULL:** Se refiere a los valores de las tablas que no tienen valor es decir (valor nulo). Si queremos referirnos específicamente a los valores nulos en una consulta escribimos "nombre de la columna" IS NULL.
  Usamos **COALESCE(**columna, dato que quieres meter si hay NULL**):** Muestra los valores de una columna, si pilla valores nulos les da el valor que tu hayas puesto.

  Para dar condiciones(no entra en este examen):
  CASE WHEN "lo que sea"
  THEN "muestro algo"
  WHEN (opcional, sería para dar otra condición, que debe seguirse con otro THEN)
  ELSE (muestra otra cosa que tu elijas si no se cumple el WHEN)
  END
  FROM (tiene que ir al final en estos casos)

Conceptos Prácticos…

- Cuidado si una cadena lleva el carácter " ´ ", para que en el string se reconozca como carácter debes poner dos ("´´"). Se reconocerá uno solo.
- En SQL un string debe ir entre comillas
- Si quiero que me muestren los elementos bajo dos condiciones que se cumplen en una misma columna (por ejemplo, si tengo una columna llamada deportes y en ella tengo beisbol, futbol y baloncesto y quiero que se muestren las

personas que participan en dos de estos deportes) NO puedo poner el por ejemplo … “beisbol AND futbol” porque se interpreta que se deben cumplir ambas condiciones a la vez. **DEBES UTILIZAR “OR”.**

- Usa **AS** para cambiar el nombre a una de las columnas que vas a sacar. Esto es muy útil si usamos **Funciones de agregado** ya que quedan nombres raros.
- Si usamos LIKE = “esto es una expresión regular”
- Si usamos = “esto es un string”
- HAVING/FROM/WHERE: Ejecutan predicados.
- El orden de ejecución de una consulta va desde el FROM hasta abajo y lo último en ejecutarse es el SELECT.
- **NO** puedes poner subconsultas al lado izquierdo, pues da error.
- En vez de poner x < 0 pon mejor “IS NOT NULL”

Ayudas en ejercicios…

- Si hay que acotar la información a partir de valores numéricos estos NO van entre comillas, pues no son strings.
- Cuidado con el LIKE ‘%x%’ = Sacará los valores que contengan la letra x
- Puedes concatenar LIKE´s un buen ejemplo es la siguiente consulta…

```
SELECT name
FROM world
WHERE name LIKE 'C%' AND name LIKE '%ia';
```

- Con LIKE si quieres que los valores tengan un determinado carácter en una posición determinada usa ‘_x%’ (eso sacacaría los valores con x en segunda posición)
- Si quieres sacar valores con un determinado numero de caracteres puedes usar LIKE ‘____’ En este caso, sacaría los valores con cuatro caracteres
- Puedes hacer operaciones en el propio SELECT, pero recuerda que el nombre quedará con lo puesto en esa operación (soluciona esto con un “AS”)
- NO USES “¡=” USA “<>”
- Antes de usar JOIN asegúrate de que es necesario, es decir, observa las tablas de las que realmente necesitas sacar valores.
- Para relacionar tablas usa una expresión como la siguiente…

```
FROM game JOIN goal ON (game.id=goal.matchid)
```

  Debes relacionar las tablas a partir del dato que las relaciona, en este caso, la tabla juegos tiene su id y este se relaciona con con la tabla de puntos con el dato “idpartido”.
- Puedes hacer subconsultas aun usando JOIN
- Acuérdate del DISTINCT para evitar valores repetidos
- Otra forma de usar JOIN…

```
SELECT title
  FROM movie, casting, actor
 WHERE name='Harrison Ford' AND movieid=movie.id
    AND actorid=actor.id
```

- Ten cuidado a la hora de usar LEFT/RIGHT JOIN dependiendo de que tabla menciones primero y lo que te piden.
- La consulta 15, soluciones…

  Con JOINS.

```
SELECT DISTINCT Actor1.name
FROM actor AS Actor1 JOIN casting AS Casting1
                ON Actor1.id = Casting1.actorid
                JOIN casting AS Casting2
                ON Casting1.movieid = Casting2.movieid
                JOIN actor AS Actor2
                ON Actor2.id = Casting2.actorid
WHERE Actor2.name = 'Art Garfunkel'
 AND Actor2.id  <> Actor1.id;
```

  Con JOINS Y SUBCONSULTAS.

```
SELECT actor.name
FROM actor JOIN casting ON actor.id = casting.actorid
WHERE actor.name <> 'Art Garfunkel'
 AND casting.movieid IN (
 -- 👇👇👇 AMBITO NUEVO 👇👇👇
 SELECT casting.movieid
 FROM actor JOIN casting ON actor.id = casting.actorid
 WHERE actor.name = 'Art Garfunkel'
```