2 April SQL

SQL stands for Structured Query Language,
a language for manipulating and talking about data in databases.
It first came into use in 1970 and became a standard in 1986 by IBM in conjunction with several projects for the US government and for many years it remained a government-only project. It's a programing language which is used to access data stored in a database.
The word "SQL" is an acronym, which is associated, today, with "Structured Query Language". It was originally called SEQUEL with a slightly different meaning. Some people still enunciate the acronym as SEQUEL and some people enunciate each individual letter, S-Q-L. They mean the same thing.
SQL, as a data retrieval language, is an industry standard; All relational database products, SQL is the mechanism, language and syntax used to retrieve data from ar database.

▼ LIKE

```
SELECT column_name(s)
FROM table_name
WHERE column_1 LIKE pattern;
```

patterns ⇒ wildcars olarak geçer araştırmaya değer... Başka dillerde REGEX diye geçiyor...

Percent(%) ⇒ sıfır veya daha fazla karakteri seçer. W% ile başlayıp herhangi bir şekilde tamamlananları seç

Underscore(_) ⇒ Tek bir karakteri ifade eder. 'El_is' 3 karakteri farketmez.

Aggregate Functions ⇒ AVG MAX MIN COUNT SUM olmak üzere toplam 5 adettir. SUM ve AVG numeric değerleri ile kullanılabilir. Diğerleri farketmez.

NULL kendine bile eşit değildir. Yokluktur.

```
SELECT COUNT(column_name)
FROM table_name
/* kaç adet olduğunu verir.COUNT(*) yapılırsa NULL'ları da sayar. Sütun adı belirtilirse
saymaz.*/
```

counter(*) as takma_isim ⇒ as ile bir sütuna isim verilebilir. Sadece result'ta görünür.

from tracks as t ⇒ as kullanılmayada bilir. t ismi ile kullanılabilir.

```
SELECT MIN(column_name)
FROM table_name
```

```
/* minimum değeri verir. MAX da aynı şekilde kullanılır. Maksimumu verir.*/
```

```
SELECT SUM(column_name)
FROM table_name
/* sütun toplamını verir. AVG kullanılırsa ortalama verir.*/
```

- round yuvarlıyor. round(sum(total),2) virgulden sonra 2 basamak göstermesi için.

▼ GROUP BY Clause

Bir Aggregate function ile birlikte kullanılır.

```
SELECT column_1, aggregate_function(column_2)
FROM table_name
GROUP BY column_1;
/* Gruplanan sütunları aggregate kullanılarak anlamlı hale getirilir.*/
```

```
/*=====================================================
LIKE
=====================================================*/
/* tracks tablosunda Composer sutunu Bach ile biten kayıtların Name bilgilerini
listeyen sorguyu yazınız*/
SELECT name
FROM tracks
WHERE Composer like '%Bach'


/* albums tablosunda Title (başlık) sutununda Greatest içeren kayıtların tüm
bilgilerini listeyen sorguyu yazınız*/
SELECT *
FROM albums
WHERE Title like '%Greatest%' -- başında ve sonunda ne olduğu önemli değil

/* invoices tablosunda, 2010 ve 2019 arası bir tarihte (InvoiceDate) Sadece
şubat aylarında gerçekleşmiş olan faturaların tüm bilgilerini listeleyen
sorguyu yazınız*/
SELECT *
FROM invoices
WHERE InvoiceDate like '201_-02%';


/* customers tablosunda, isimleri (FirstName) üç harfli olan müşterilerin
FirstName, LastName ve City bilgilerini listeleyen sorguyu yazınız*/
SELECT FirstName, LastName, City
FROM customers
WHERE FirstName like '___';


/* customers tablosunda, soyisimleri Sch veya Go ile başlayan müşterilerin
FirstName, LastName ve City bilgilerini listeleyen sorguyu yazınız*/
SELECT FirstName, LastName, City
FROM customers
```

```
WHERE LastName like 'Sch%' or LastName like 'Go%' %; -- birden fazla sorgulanan
--varsa like hepsine yazılmalıdır. Like case-insensitive çalışıyor. Edit programa
-- kısmından değiştirilebilir.


/*================================================
  AGGREGATE FUNCTION COUNT,SUM,MIN,MAX, AVG)
================================================*/

-- COUNT
-------------------------------------------------------------------------------
/* invoices tablosunda kaç adet fatura bulunduğunu döndüren sorgu yazınız */
SELECT count(*)   -- Tamamını sayar null dahildir.
FROM invoices;

SELECT * from invoices;

SELECT count(BillingState) -- NULL olanlar sayılmaz
FROM invoices;

 -- Kaç tane NULL girilmiş??
  SELECT count(*) as num_null
  FROM invoices
  WHERE BillingState IS NOT NULL;
  -- IS NULL null mu? * ile kullanılması gerekir..
  -- IS NOT NULL null olmayanları.

 /* tracks tablosunda kaç adet farklı Composer bulunduğunu döndüren sorguyu
 yazınız*/
 SELECT count(DISTINCT Composer) as
 FROM tracks;


-- MIN,MAX
-------------------------------------------------------------------------------------------------------
/* tracks tablosundaki şarkı süresi en kısa olan şarkının adını ve süresi
listeleyen sorguyu yaziniz */
SELECT name, min(Milliseconds)/1000.0 as min_duration_sc
FROM tracks;

/*students tablosundaki en düşük ve en yüksek notu listeleyen sorguyu yazınız */
SELECT min(Grade) as min_grade, max(Grade) as max_grade
FROM students;

-- SUM,AVG
-------------------------------------------------------------------------------------------------------
/* invoices  tablosundaki faturaların toplam değerini listeyiniz */
SELECT round(sum(total)) as total_amount
FROM invoices;
-- round yuvarlıyor. round(sum(total),2) virgulden sonra 2 basamak göstermesi için.

/* invoices  tablosundaki faturaların ortalama değerini listeyiniz */
SELECT round(AVG(total),2) as avg_amount
FROM invoices;

/* tracks tablosundaki şarkıların ortalama süresinden daha uzun olan şarkıların
adlarını listeleyiniz.*/
SELECT AVG(Milliseconds) as avg_duration
FROM tracks;
```

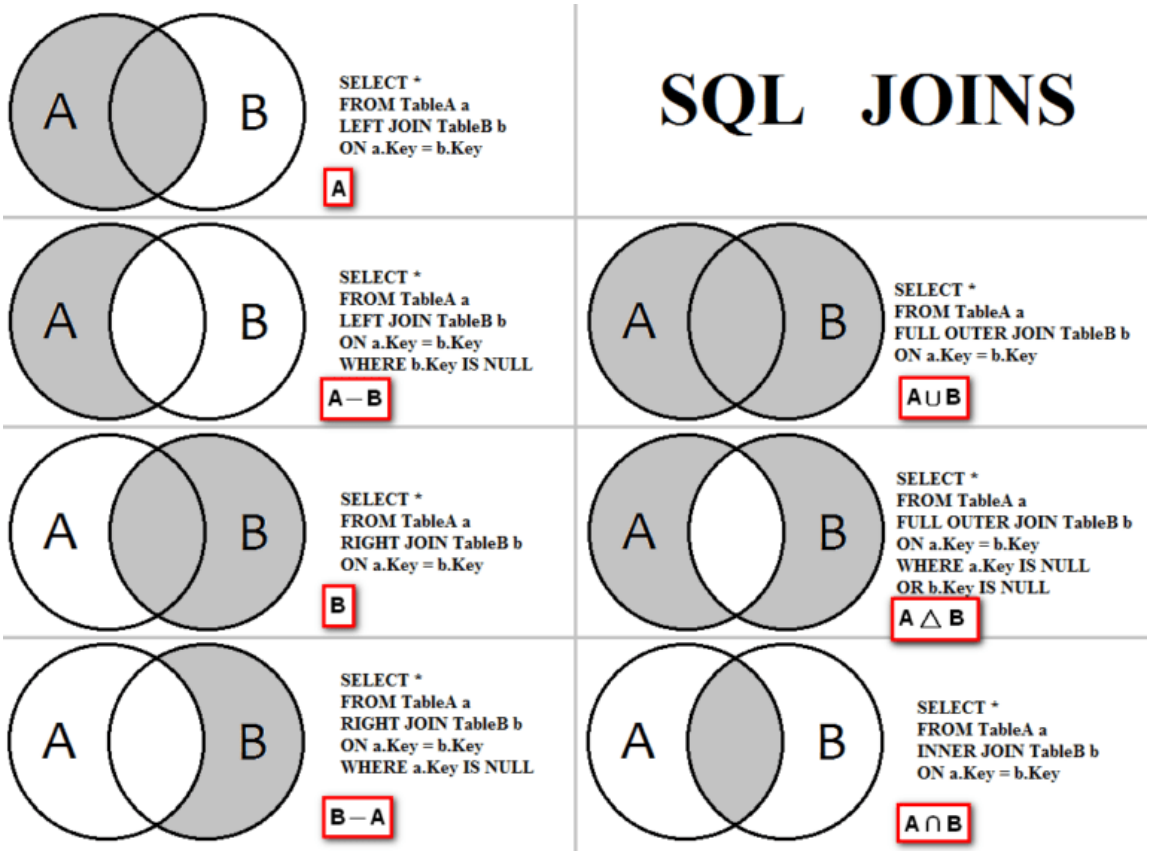
```
SELECT name, Milliseconds
FROM tracks
WHERE Milliseconds > 393599.212103911;

/*Bu yöntem hard-coded olduğu için çok mantıklı bir çözüm değil.
alt-sorgu(sub-query) ile daha doğru bir yaklaşım olacaktır sonraki bölümlerde
alt-sorguyu ayrıntılı bir şekilde inceleyeceğiz.*/
SELECT name, Milliseconds
FROM tracks
WHERE Milliseconds > (SELECT AVG(Milliseconds) FROM tracks);
-- iç içe sorgularda önce içtekini yapar.

/*=====================================================
 GROUP BY
=====================================================*/

/* tracks tablosundaki her bir Bestecisinin (Composer) toplam şarkı sayısını
Besteci adına göre gruplandırarak listeleyen sorguyu yazınız. */
SELECT Composer,count(name)
FROM tracks
WHERE Composer IS NOT NULL
GROUP by Composer;

/* invoices tablosundaki her bir ülkenin (BillingCountry) fatura ortalamalarını
hesaplayan ve ülke bilgileri ile listeleyen sorguyu yazınız.*/

SELECT BillingCountry, round(avg(total),2) as avg_amount
FROM invoices
WHERE InvoiceDate BETWEEN '2009-01-01' AND '2011-12-31'
GROUP BY BillingCountry
ORDER by avg_amount DESC; -- bunu ben ekledim sıralamak için
```

▼ JOİN

Farklı tablodaki ortak verilere göre veri çekilmesi yapılmasıdır. İki veya daha fazla tablodan verileri çekip tek bir tabla olarak sonuç gösterilmesidir. Foreign key'ler aracılığı bu işlemi yapar.

SQLite özel bir durum ⇒ Foreign key'in refer ettiği sütun primary key olmak zorundadır.!!!!!! Foreign key'de olmayını diğer tabloda primary key yapamayız. Yapmaya çalışırsak hata alırız.!!!! Hatırlatıcı primary key eşsiz olmalı, tekrarsız... Right Join aslında INNER JOIN'İ verir. FULL JOIN ise LEFT JOIN'i vermiş oluyor. Bu sadece SQLite'a özel durum key'ler durumundan dolayı...

Başka tablodan veri getirmek için iki farklı yöntem vardır. 1. Subquery 2.Join



Çeşitleri ;

- INNER JOIN, ⇒ iki tablonun kesişimini verir.
- LEFT OUTER JOIN, ⇒ sol tabloyu direk alır. sağ tablonun kesişimini
- RIGHT OUTER JOIN, ⇒ left'in tam tersi
- FULL OUTER JOIN ⇒ Hepsini alır.

```
SELECT columns
FROM table_A
INNER JOIN table_B ON join_conditions
/* columns=> tablo_A.salary gibi yazılması gerekir. */
/* join_conditions genellikle => table_A.common_field = table_B.common_field */
```