-- DATABASE --

-- Database (Veritabanı): Verileri anlamlı saklayan programdır.

-- SQL: Veri işlemleri içindir. Veritabanına kaydetme, okuma, güncelleme, silme (CRUD: Create,Read,Update,Delete) gibi işlemleri yaptığımız dildir.

/\*

Örnek:

\* Elbise Dolabı -> Database / Veritabanı

\* Gömlekler -> Table / Tablo

\* Tip \* Renk \* ... \* -> Column / Item / Sutun / Başlık / Eleman

- ... - ... - ... -

- Uzun - Mavi - ... - -> Row / Satır / Data / Veri

- Kısa - Mor - ... -

- ... - ... - ... -

\* Pantolonlar

- ... - ... - ... -

\* Kemerler

- ... - ... - ... -

\*/

-- YORUM --

-- Tek satır yorum için iki-tire (--) kullanabilirim. iki-tire kullandıktan sonrası komple yorum satırı olur.

SELECT 1 AS One; -- Bundan sonrası yorumdur. (Kod geçersizdir: SELECT 2;)

/\*

TersSlaş-Yıldız ile istediğim gibi yorum yazabilirim.

İster satır içi, ister çoklu satıra yazabilirim.

Kapatmak için Yıldız-TersSlaş kullanmalıyım.

\*/

SELECT 1 /\* Kod içi yorum \*/ AS One;

-- SQL --

-- SQL is case insensitive (büyük-küçük harf ayrımı gözetmez)

-- BNF kurallar: (Python PEP8 gibi.)

-- \* SQL komutları BÜYÜK HARF, tablo-sütun isimleri küçük harfle yazmamız beklenir.

-- \* String verilerde çift-tırnak ve tek-tırnak kullanabiliyor. (Her program çift-tırnak desteklemez). tek-tırnak (') kullanırın.

-- \* Noktalı-virgül (;) -> SQL işlemi yaptğınızda satır sonuna eklemeniz gereken ayıraçtır. Tek bir işlem yaptığnızda gerek yoktur. Çoklu işlemlerde mecburidir.

-- noktalı virgül:

-- SELECT 1;

-- select AlbumId, Title, ArtistId from albums; -- Çalışır ama BNF aykırı.

-- SELECT - Seç ve Getir.

-- FROM - Hangi tablodan?

-- SELECT AlbumId, Title, ArtistId FROM albums; -- Tavsiye edilen yöntemdir.

-- SELECT \* FROM albums; -- Tüm sutunları getir.

-- AS - Tablo veya Sütun isimlendirme için kullanılır. -> Lakap takma :)

-- SELECT FirstName AS isim FROM clients; -- Sütün isimlendirme

-- SELECT (1+2-3) AS lakap; -- Matematiksel işlem isimlendirme

-- SELECT c.FirstName FROM clients AS c; -- Tablo isimlendirme

-- SELECT c.FirstName isim FROM clients c; -- Kısa kullanım

-- DISTINCT - Tekrar edilen kayıtarın tekrar edilmesini engeller (tek kayıt olarak getirir)

-- SELECT DISTINCT FirstName FROM clients; -- FirstName aynı olanları 1 defa getir.

-- SELECT DISTINCT FirstName, LastName FROM clients; -- (FirstName AND LastName) aynı olanları 1 defa getir.

-- WHERE -- Filtreleme

-- SELECT \* FROM customers WHERE City = 'Berlin'; -- = Eşit olanlar.

-- SELECT \* FROM customers WHERE CustomerId > 3; -- Büyük olanlar.

-- SELECT \* FROM customers WHERE CustomerId >= 3; -- Dahil Büyük olanlar.

-- SELECT \* FROM customers WHERE CustomerId < 3; -- Küçük olanlar.

-- SELECT \* FROM customers WHERE CustomerId <= 3; -- Dahil Küçük olanlar.

-- SELECT \* FROM customers WHERE CustomerId <> 3; -- Eşit olmayanlar

-- SELECT \* FROM customers WHERE CustomerId != 3; -- Eşit olmayanlar

-- SELECT \* FROM customers WHERE CustomerId BETWEEN 3 AND 15; -- Arasında olanlar

-- SELECT CustomerId, FirstName, LastName FROM customers WHERE CustomerId BETWEEN 3 AND 5; -- Arasında olanlar

-- SELECT \* FROM invoices WHERE InvoiceDate BETWEEN '2009-01-01' AND '2009-02-15'; -- İki tarih arasındaki veriler.

-- LIMIT -- Limitler -- Default: LIMIT start=0, kaçtane=?

-- SELECT \* FROM invoices LIMIT 20; -- İlk 20 kayıt

-- SELECT \* FROM invoices LIMIT 0, 20; -- İlk 20 kayıt

-- SELECT \* FROM invoices LIMIT 20, 25; -- İlk 20 kayıt

-- SELECT \* FROM invoices LIMIT 20, 25; -- İlk 20'den sonraki (20 hariç) 25 adet kayıt getir.

-- ORDER BY - Sıralama - Default: ASC

-- SELECT \* FROM invoices ORDER BY InvoiceDate ASC; -- Yakın tarihten uzağa sırala.

-- SELECT \* FROM invoices ORDER BY InvoiceDate DESC; -- Uzak tarihten yakın sırala.

-- SELECT \* FROM invoices ORDER BY InvoiceDate; -- Default: ASC

-- SELECT \* FROM invoices ORDER BY BillingAddress ASC; -- A -> Z sırala. (Önce rakamlar)

-- SELECT \* FROM invoices ORDER BY BillingAddress DESC; -- Z -> A sırala. (Sonra rakamlar)

-- SELECT \* FROM invoices ORDER BY BillingCity DESC, BillingAddress ASC; -- Önce BillingCity DESC, sonra BillingAddress ASC

-- SELECT \* FROM invoices ORDER BY BillingCountry DESC, BillingCity ASC;

-- SELECT \* FROM invoices WHERE CustomerId = 54 ORDER BY InvoiceDate ASC -- Müşteri ID'si = 54 olan faturaları getir ve tarihe göre sırala.

-- SELECT \* FROM invoices WHERE CustomerId = 54 ORDER BY InvoiceDate ASC LIMIT 3 -- Müşteri ID'si = 54 olan faturaları getir ve tarihe göre sırala. ilk 3 kayıt

-- AND, OR, NOT -- WHERE içinde kullanılır. (ORDER BY için virgül kullanılır.)

-- SELECT \* FROM invoices WHERE CustomerId = 54 AND InvoiceDate <= '2010-09-13'; -- 54 nolu müşteri ve bu tarihte önceki kayıtlar.

-- SELECT \* FROM invoices WHERE CustomerId = 54 OR CustomerId = 42; -- Hem 42 hem de 54 numaralı müşterilerin faturaları.

-- SELECT \* FROM invoices WHERE (CustomerId = 54 AND InvoiceDate <= '2010-09-13') OR CustomerId = 42; -- AND ve OR kullanımı.

-- SELECT \* FROM invoices WHERE NOT CustomerId = 54; -- 54 olmayan müşterilerin faturalarını getir.

-- SELECT \* FROM invoices WHERE CustomerId <> 54; -- 54 eşit olmayan müşterilerin faturalarını getir.

-- SELECT \* FROM invoices WHERE InvoiceDate >= '2008-09-01' AND CustomerId = 54 AND NOT Total <= 2; -- Çoklu kullanım

-- SELECT \* FROM customers WHERE CustomerId NOT BETWEEN 3 AND 15; -- Arasında OLMAYAN

-- SELECT \* FROM invoices ORDER BY Total ASC LIMIT 10; -- En düşük fatura tutarına ait ilk 10 kayıt.

-- IN -- İçindekilerden biri mi?

-- NOT IN -- İçindekilerinden hiçbiri ?

-- SELECT \* FROM invoices WHERE CustomerId = 50 OR CustomerId = 51 OR CustomerId = 52 OR CustomerId = 53 OR CustomerId = 54; -- Daha kısa yolu var -> IN :)

-- SELECT \* FROM invoices WHERE CustomerId IN (50, 51, 52, 53, 54); -- Müşteri numaralası parantez içinde yazılı OLAN getir. (OR olarak düşünülebilir.)

-- SELECT \* FROM invoices WHERE CustomerId NOT IN (50, 51, 52, 53, 54); -- Müşteri numaralası parantez içinde yazılı OLMAYAN getir. (AND olarak düşünülebilir.)

-- SELECT \* FROM invoices WHERE CustomerId BETWEEN 50 AND 55; -- 50 ile 55 arasını getir.

-- LIKE -- İçinde ara. (insensitive -> Küçük büyük harf ayrımı yapmaz.)

-- Yüzde (%) ve alt-tire (\_) -> JOKER KARAKTERLER

-- % -> "Karakter limiti YOK ve herhangi bir karakter"

-- \_ -> "Karakter limiti VAR, her bir alt-tire bir (herhangi) karakteri temsil eder"

-- SELECT \* FROM invoices WHERE BillingCountry LIKE 'G%'; -- G ile başlayan ülkelerin faturaları

-- SELECT \* FROM invoices WHERE BillingCountry LIKE 'U%'; -- U ile başlayan ülkelerin faturaları

-- SELECT \* FROM invoices WHERE BillingCountry LIKE '%SA%'; -- İçinde SA olan ülkelerin faturaları

-- SELECT \* FROM invoices WHERE BillingCountry LIKE '%M%'; -- İçinde M olan ülkelerin faturaları

-- SELECT \* FROM invoices WHERE BillingCity LIKE '%e%o%'; -- İçinde a ve b olan (yanyana olmak zorunda değil, olabilir de)

-- SELECT \* FROM invoices WHERE BillingCity LIKE '%o'; -- O ile biten

-- SELECT \* FROM invoices WHERE BillingCity LIKE '%e%o'; -- İçinde E geçen ve O ile biten

-- SELECT \* FROM invoices WHERE BillingCity LIKE '\_e%'; -- İlk karakteri önemsiz, ama ikinci karakteri E olan

-- SELECT \* FROM invoices WHERE BillingCity LIKE '\_e\_m%'; -- 1:önemsiz, 2:E, 3. önemsiz, 4: M ... olan

-- COUNT -- Çağırdığımız verinin kayıt Sayısı.

-- SELECT COUNT(\*) FROM invoices; -- Satır sayısı -> (\*) kullanmasan olmaz mı?

-- SELECT COUNT(InvoiceId) FROM invoices;

-- SELECT COUNT(InvoiceId) AS fatura\_adet FROM invoices WHERE BillingCountry LIKE 'G%';

-- SUM - Toplama -> İçindeki veriler sayı olmak zorunda.

-- SELECT SUM(Total) AS Toplam FROM invoices;

-- SELECT SUM(Total) AS toplam FROM invoices WHERE BillingCity LIKE '\_e%';

-- AVG - Ortalama

-- SELECT AVG(Total) AS ortalama FROM invoices;

-- SELECT AVG(Total) AS ortalama FROM invoices WHERE BillingCity LIKE '\_e%';

-- ROUND -> Yuvarlar. -> ROUND(sayı, noktadanSonraHaneSayisi)

-- SELECT ROUND(AVG(Total), 2) AS ortalama FROM invoices WHERE BillingCity LIKE '\_e%';

-- SUM, AVG, MIN, MAX, COUNT

-- Veriyi tek satırda getirir.

-- SELECT BillingCity, SUM(Total) AS toplam\_tutar FROM invoices; -- SUM ->Topla

-- SELECT BillingCity, ROUND(AVG(Total), 2) AS toplam\_tutar FROM invoices; -- AVG -> Ortalama (ROUND -> Yuvarla)

-- SELECT BillingCity, MIN(Total) AS toplam\_tutar FROM invoices; -- MIN -> Min

-- SELECT BillingCity, MAX(Total) AS toplam\_tutar FROM invoices; -- MAX -> Max

-- SELECT BillingCity, COUNT(Total) AS toplam\_tutar FROM invoices; -- COUNT -> Satır Sayısı

-- SELECT BillingCity, MIN(BillingCity) AS toplam\_tutar FROM invoices; -- MIN -> Min (STRING veride çalışıyor.)

-- SELECT BillingCity, MAX(BillingCity) AS toplam\_tutar FROM invoices; -- MAX -> Max (STRING veride çalışıyor.)

-- GROUP BY -> GRUPLAMA YAPAR.

/\*

-- Şehre göre toplam tutar getir:

SELECT BillingCity, SUM(Total) AS toplam\_tutar

FROM invoices

GROUP BY BillingCity;

\*/

-- Min Tutara göre sırala:

-- SELECT BillingCity, MIN(Total) AS min\_tutar FROM invoices GROUP BY BillingCity ORDER BY min\_tutar ASC, BillingCity DESC;

-- Tutara göre sırala:

-- SELECT BillingCity, Total FROM invoices GROUP BY BillingCity ORDER BY Total ASC, BillingCity ASC;

-- Karakter uzunluğuna göre sırala:

-- SELECT LENGTH(BillingCity) AS karakter\_sayisi, BillingCity, Total FROM invoices ORDER BY BillingState ASC, BillingCity ASC, Total ASC;

-- SELECT LENGTH(12113) AS number, LENGTH('sadnashds') AS string; -- LENGTH -> HEm sayı hem de yazı -> karakter sayısı.

-- SELECT BillingCity, SUM(Total) AS toplam\_tutar FROM invoices GROUP BY BillingCity; -- Fatura toplam tutar (şehre göre)

-- SELECT BillingCity, COUNT(Total) AS toplam\_tutar FROM invoices WHERE BillingCity LIKE 'B%' GROUP BY BillingCity;

-- SELECT BillingCity, COUNT(Total) AS toplam\_tutar FROM invoices WHERE LEFT(BillingCity, 1) IN ('B', 'C') GROUP BY BillingCity;

-- LEFT, RIGHT

-- SELECT LEFT('ABCD', 1) AS ilk\_karakter;

-- SELECT RIGHT('ABCD', 1) AS son\_karakter;

/\*

-- DATA ÇAĞIRAN TEMEL KOMUTLAR:

SELECT

FROM

JOIN

WHERE

-- ÇAĞRILMIŞ DATA İŞLEMLERİ:

SELECT DISTINCT -- Aynı kayıtları 1 defa göster.

GROUP BY -- Gruplar.

HAVING -- Gelen data içinde filtreleme.

ORDER BY -- Sıralar.

LIMIT -- Sınırlı adet getir. (limitler)

-- EKSTRA

UNION (ALL) -- Çağrılmış birden fazla datayı birleştirir. (sutun sayısı ve sutun isimleri/lakapları aynı olmak ZORUNDA)

\*/

-- -- -- -- - - JOINS -- -- -- - - - - -

/\*

INNER JOIN -> Kısa Kullanım: JOIN -- Tercih: INNER JOIN - Kesişim kayıtlar

LEFT OUTER JOIN -> Kısa: LEFT JOIN - Ana tablo hepsi, kesişimleri de göster.

RIGHT OUTER JOIN -> Kısa: RIGHT JOIN - Join tablo hepsi, kesişimleri de göster.

FULL OUTER JOIN - Her iki tablo hepsi, kesişimleri de göster.

CROSS JOIN -> Kısa: FROM içinde - İlişkisiz tüm tabloları yanyana yaz.

\*/

-- INNER JOIN:

/\*

SELECT c.FirstName, c.LastName, c.Company, i.InvoiceDate, i.Total

FROM invoices AS i

INNER JOIN customers AS c ON i.CustomerId=c.CustomerId

WHERE c.FirstName LIKE 'L%';

\*/

/\*

SELECT c.FirstName, c.LastName, c.Company, i.InvoiceDate, SUM(i.Total) AS toplam\_fatura

FROM invoices AS i

INNER JOIN customers AS c ON c.CustomerId=i.CustomerId -- Sıra önemsiz, bu da çalışır -> c.CustomerId=i.CustomerId

GROUP BY c.Company, c.FirstName

ORDER BY i.InvoiceId ASC;

\*/

/\*

SELECT c.CustomerId, c.FirstName, c.LastName, c.Company, i.InvoiceDate, SUM(i.Total) AS toplam\_fatura

FROM invoices AS i

INNER JOIN customers AS c ON c.CustomerId=i.CustomerId AND c.CustomerId>5

GROUP BY c.Company, c.FirstName

ORDER BY i.InvoiceId ASC;

\*/

-- LEFT JOIN:

/\*

SELECT ar.Name, al.Title

FROM artists AS ar

LEFT JOIN albums AS al ON al.ArtistId=ar.ArtistId

GROUP BY ar.ArtistId;

\*/

-- RIGHT JOIN:

/\*

SELECT \*

FROM artists AS ar

RIGHT JOIN albums AS al ON al.ArtistId=ar.ArtistId;

\*/

-- FULL OUTER JOIN:

/\*

SELECT \*

FROM artists AS ar

FULL OUTER JOIN albums AS al ON al.ArtistId=ar.ArtistId

\*/

-- CROSS JOIN

/\*

SELECT \*

FROM invoices

CROSS JOIN customers;

\*/

/\*

-- Kısa Kullanım:

SELECT \*

FROM invoices, customers;

\*/

/\*

SELECT i.InvoiceDate, i.BillingCity, c.FirstName, c.LastName

FROM invoices AS i, customers AS c;

\*/