DB yapısı: Tablo-->kolon-->değer

Kolon-->Alan/column

Condition🡪 column=‘value’

Tablo Adı-->>FROM T, İNSERT İNTO T, UPDATE T SET,

\*\*\*\*\*\*\*\*\*\*\***SELECT**\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SELECT column1, column2, ...

FROM table\_name;

SELECT \* FROM Customers;

SELECT CustomerName, \* FROM Customers; (*CustomerName* kolonu başta gelsin)

SELECT CustomerName, City FROM Customers; (SELECT)

SELECT DISTINCT Country FROM Customers; (DISTINCT)-->Tekrarları siler)

SELECT COUNT(DISTINCT Country) FROM Customers; (COUNT) -->Sayısını verir. "21")

\* **WHERE,LIKE,BETWEEN,IN,AND,OR, NOT, İS NULL, İS NOT NULL, AS, TOP, SUM**, **MAX\***

SELECT column1, column2, ...

FROM table\_name

WHERE condition;

SELECT \* FROM Customers WHERE Country='Mexico'; (WHERE)

SELECT \* FROM Customers WHERE CustomerID=1; (WHERE)

SELECT \* FROM Customers WHERE City LIKE 's%'; (LIKE)

SELECT \* FROM Customers WHERE City LIKE '\_\_b%'; (LIKE) (Tek harf yerine-->Tek altçizgi(\_) )

SELECT \* FROM Customers WHERE City LIKE '[ks]%'; (LIKE)(baş harfleri k ve s olan)

SELECT \* FROM Customers WHERE City LIKE '[!ks]%'; (LIKE)(! baş harfleri ks olmayan)

SELECT \* FROM Customers WHERE City **LIKE** '[a-c]%'; (LIKE)(! baş harfleri a,b,c olan)

SELECT\* FROM Customers WHERE City LIKE 'L\_n\_on';

SELECT \* FROM Customers WHERE City LIKE '[bsp]%'; (City den Baş harfleri B, S, P olanları getir)

SELECT \* FROM Products WHERE Price **BETWEEN** 50 AND 60; (BETWEEN)(50-60 Arasında 50 ile 60 da dahil)

SELECT \* FROM Customers WHERE City **IN** ('Paris'); (IN)(İN(value))

SELECT \* FROM Customers WHERE City IN ('Paris','Londra'); (IN)

SELECT \* FROM Customers WHERE Country='Germany' **AND** City='Berlin'; (AND column=‘value’ )

SELECT \* FROM Customers WHERE Country='Germany' **OR** City='Berlin'; (OR)

SELECT \* FROM Customers WHERE **NOT** Country='Germany';

SELECT \* FROM Customers WHERE NOT Country **İS NULL**; (İS NULL)

SELECT \* FROM Customers WHERE Country **İS NOT NULL**; (İS NOT NULL)

SELECT **TOP** 3 \* FROM Customers; (TOP 3-->ilk 3 ü getir. ***SİSTEMi*** *yormaz*) (TOP)

SELECT \* FROM Customers LIMIT 3; (LİMİT)

SELECT TOP 50 PERCENT \* FROM Customers; (PERCENT--%50’si 91 adet ise 45.5tan--->46 olur)

SELECT MIN(Price) AS SmallestPrice FROM Products; (Kolondaki minimum değer)

SELECT MAX(Price) AS SmallestPrice FROM Products; (Kolondaki maximum değer)

SELECT SUM(Quantity) FROM OrderDetails; (Toplam)

SELECT CustomerID AS ID, CustomerName AS Customer FROM Customers; (2 sanal kolon yanyana)

SELECT CustomerName AS Customer, ContactName AS [Contact Person] FROM Customers;

(2 sanal kolon tabloyu 2 eşit parçaya ayırır)

SELECT CustomerName, Address + ', ' + PostalCode + ' ' + City + ', ' + Country AS Address

FROM **Customers**; (2 kolon. 2.sinde alanları birleştirme - Alan Birleştirme)

SELECT o.OrderID, o.OrderDate, c.CustomerName

FROM Customers AS c, Orders AS o

WHERE c.CustomerName='Around the Horn' AND c.CustomerID=o.CustomerID;

SELECT Orders.OrderID, Orders.OrderDate, Customers.CustomerName

FROM Customers, Orders

WHERE Customers.CustomerName='Around the Horn'

\*\*\*\*\*\*\*\*\*\*\***İNSERT İNTO**\*\*\*\*\*\*\*\*\*\*\*\*\*\*

INSERT INTO table\_name (column1, column2, column3, ...)

VALUES (value1, value2, value3, ...);

INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)

VALUES ('Cardinal','Tom B. Erichsen','Skagen 21','Stavanger','4006','Norway');

\*\*\*\*\*\*\*\*\*\*\***DELETE**\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DELETE FROM *table\_name*WHERE *condition*;

Tablo oluşturma

CREATE TABLE Persons (

PersonID int,

LastName varchar(255),

FirstName varchar(255),

Address varchar(255),

City varchar(255)

);

CREATE TABLE Dash (

PersonID int,

LastDash varchar(255),

FirstDash varchar(255),

Email varchar(255),

Country varchar(255)

);

Tabloya veri ekleme

INSERT INTO Persons (PersonID, LastName, FirstName, Address, City)

VALUES (1,'Tom B. Erichsen','Skagen 21','Stavanger','LA');

INSERT INTO Persons (PersonID, LastName, FirstName, Address, City)

VALUES (1,'Tom B. Erichsen','Skagen 21','Stavanger','Cali');

INSERT INTO Persons (PersonID, LastName, FirstName, Address, City)

VALUES (5,'MURA','Seta','akka','ist');

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PersonID** | **LastName** | **FirstName** | **Address** | **City** |
| 1 | Tom B. Erichsen | Skagen 21 | Stavanger | LA |
| 1 | Tom B. Erichsen | Skagen 21 | Stavanger | Cali |
| 5 | MURA | Seta | akka | ist |

INSERT INTO Dash (PersonID, LastDash, FirstDash, Email )

VALUES (1,'mark','er','Stavanger');

INSERT INTO Dash (PersonID, LastDash, FirstDash, Email )

VALUES (1,'david','ess','Stavan');

INSERT INTO Dash (PersonID, LastDash, FirstDash, Email )

VALUES (3,'esmana','erk','Sta');

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PersonID** | **LastDash** | **FirstDash** | **Email** | **Country** |
| 1 | mark | er | Stavanger | *null* |
| 1 | david | ess | Stavan | *null* |
| 3 | esmana | erk | Sta | *null* |

\*\*\*\*\*\*\*\*\*\*\***UPDATE**\*\*\*\*\*\*\*\*\*\*\*\*\*\*

UPDATE table\_name

SET column1 = value1, column2 = value2, ...

WHERE condition;

/// UPDATE Dash SET FirstDash='duman' where PersonID='1';

/// update dash set firstdash='serrano' where firstdash='erk';

/// update Persons set LastName='mark' where City='Cali';

PERSON TABLOSU

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PersonID** | **LastDash** | **FirstDash** | **Email** | **Country** |
| 1 | mark | duman | Stavanger | *null* |
| 1 | david | duman | Stavan | *null* |
| 3 | esmana | serrano | Sta | *null* |

DASH TABLOSU

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PersonID** | **LastName** | **FirstName** | **Address** | **City** |
| 1 | Tom B. Erichsen | Skagen 21 | Stavanger | LA |
| 1 | mark | Skagen 21 | Stavanger | Cali |
| 5 | MURA | Seta | akka | ist |

\*\*\*\*\*\*\*\*\*\*\* **ORDER BY/ASC** \*\*\*\*\*\*\*\*\*\*\*\*\*\*SIRALAMAK-Verileri Artan Sırada sıralamak için kullanılır.(ASC=Ascending, DESC=Descending)

SELECT Price,\* FROM Products ORDER BY Price ;

Number of Records: 77

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Price** | **ProductID** | **ProductName** | **SupplierID** | **CategoryID** | **Unit** |
| 2.5 | 33 | Geitost | 15 | 4 | 500 g |
| 4.5 | 24 | Guaraná Fantástica | 10 | 1 | 12 - 355 ml cans |
| 6 | 13 | Konbu | 6 | 8 | 2 kg box |

**\*\*\*\*\*\* İNNER JOİN/JOİN/NATURAL JOİN \*\*\*\*\*\*\*** Sadece kesişenleri. **ID ye göre. Sondaki eşitliğe göre kesişim.**

Not: iNNER JOİN 1.tablodakini 2.tablo ile sırayla kesiştirir. 1 e 1, 1 e 2, 2ye 1, 2ye 2..

SELECT Persons.City, Dash.LastDash

FROM Persons

INNER JOIN Dash ON Dash.PersonID = Persons.PersonID;

Number of Records: 4

|  |  |
| --- | --- |
| **City** | **LastDash** |
| LA | david |
| LA | mark |
| Cali | david |
| Cali | mark |

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*** **LEFT JOİN / LEFT OUTER JOİN \*\*\*\*\*\*\*\*\*\*\*\*\*** Kesişenler ve sol taraf. Olmayan yerine “null” yazar.

SELECT column\_name(s)  
FROM table1  
LEFT  JOIN table2  
ON table1.column\_name = table2.column\_name;

SELECT Persons.City, Dash.LastDash

FROM Persons

LEFT JOIN Dash ON Dash.PersonID = Persons.PersonID;

|  |  |
| --- | --- |
| **City** | **LastDash** |
| LA | david |
| LA | mark |
| Cali | david |
| Cali | mark |
| ist | *null* |

**\*\*\*\*\*\*\*\*\*\*RIGHT JOİN/ RIGHT OUTER JOİN \*\*\*\*\*\*\*\*\*\*** Kesişenleri ve sağdakini alır. Misal:Employees

SELECT column\_name(s)  
FROM table1  
RIGHT JOIN table2  
ON table1.column\_name = table2.column\_name;

SELECT Orders.OrderID, Employees.LastName, Employees.FirstName

FROM Orders

RIGHT JOIN Employees

ON Orders.EmployeeID = Employees.EmployeeID

ORDER BY Orders.OrderID;

Number of Records: 197

|  |  |  |
| --- | --- | --- |
| **OrderID** | **LastName** | **FirstName** |
|  | West | Adam |
| 10248 | Buchanan | Steven |
| 10249 | Suyama | Michael |
| 10250 | Peacock | Margaret |

**\*\*\*\*\*\*\*\*\*\*FULL OUTER JOIN/FULL JOIN\*\*\*\*\*\*\*\*\***Tablonun tamamını getirir.

**SELECT** column\_name(s)  
**FROM** table1  
**FULL OUTER** **JOIN** table2**ON** table1.column\_name = table2.column\_name;

**\*\*\*\*\*\*\*\*UNION\*\*\*\*\*\*\*\*\*\*** Tablonun tamamını getirir. NOT:”**UNION”** aynıları getirmez. Aynı olanları “**UNION ALL”** getirir.

**SELECT** column\_name(s)  
**FROM** table1  
**LEFT** **JOIN** table2**ON** table1.column\_name = table2.column\_name;

**UNION**

**SELECT** column\_name(s)  
**FROM** table1  
**RIGHT** **JOIN** table2**ON** table1.column\_name = table2.column\_name

**WHERE** table1.column\_name **IS NULL**;

**\*\*\*\*\*\*\*\*\*\*CARTESIAN JOIN** **\*\*\*\*\*\*\*\*\*\*** 2 TABLODAN alanlar çekmek.

SQL> SELECT ID, NAME, AMOUNT, DATE

FROM CUSTOMERS, ORDERS;

**\*\*\*\*\*\*\*\*\*\*SELF JOİN \*\*\*\*\*\*\*\*\*\***Aynı tablodan veri çekmek için. **Alias:** “AS” Kullanmaya gerek yok, kod kolay anlaşılsın diye yazılır.

**2 aynı Tablo..**

SELECT A.CustomerName AS CustomerName1, B.CustomerName AS CustomerName2, A.City

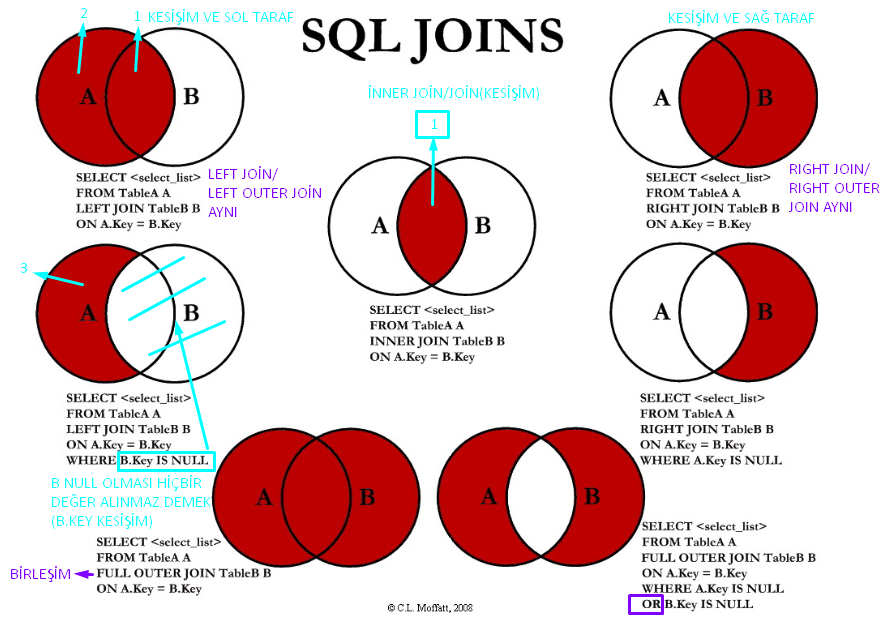
FROM Customers A, Customers B

WHERE A.CustomerID <> B.CustomerID and A.City='São Paulo' //ID ler aynı zaten

AND A.City = B.City

Number of Records: 12 (SELF JOİN)

|  |  |  |
| --- | --- | --- |
| **CustomerName1** | **CustomerName2** | **City** |
| Comércio Mineiro | Familia Arquibaldo | São Paulo |
| Comércio Mineiro | Queen Cozinha | São Paulo |
| Comércio Mineiro | Tradição Hipermercados | São Paulo |
| Familia Arquibaldo | Comércio Mineiro | São Paulo |
| Familia Arquibaldo | Queen Cozinha | São Paulo |
| Familia Arquibaldo | Tradição Hipermercados | São Paulo |
| Queen Cozinha | Comércio Mineiro | São Paulo |
| Queen Cozinha | Familia Arquibaldo | São Paulo |
| Queen Cozinha | Tradição Hipermercados | São Paulo |
| Tradição Hipermercados | Comércio Mineiro | São Paulo |
| Tradição Hipermercados | Familia Arquibaldo | São Paulo |
| Tradição Hipermercados | Queen Cozinha | São Paulo |



\*\*\*\*\*\*\*\*\*\*\***CASE WHEN**\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**CASE**  
    **WHEN** condition1 **THEN** result1  
    **WHEN** condition2 **THEN** result2  
    **WHEN** conditionN **THEN** resultN  
    **ELSE** result  
**END**;

SELECT OrderID,ProductID,

**CASE**

WHEN ProductID>20 THEN 'KALSIN'

WHEN ProductID<20 THEN 'SATILSIN'

ELSE 'BOŞVER'

**END** AS MURAT

FROM OrderDetails;

Şehre göre sıralama: (şehir null ise ülkeye göre değilse şehre göre)

SELECT CustomerName, City, Country FROM Customers

ORDER BY (CASE

WHEN City IS NULL THEN Country

ELSE City

END);

\*\*\*\*\*\*\*\*\*\*\* **GROUP BY** \*\*\*\*\*\*\*\*\*\*\*\*\*\*GRUPLAMAK. (Alt gruplara ayırma)

SELECT COUNT(CustomerID), Country, City

FROM Customers

GROUP BY Country, City;

Number of Records: 69

|  |  |  |
| --- | --- | --- |
| **COUNT(CustomerID)** | **Country** | **City** |
| 3 | Argentina | Buenos Aires |
| 1 | Austria | Graz |
| 1 | Austria | Salzburg |
| 1 | Belgium | Bruxelles |
| 1 | Belgium | Charleroi |

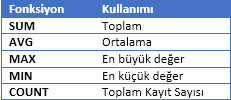
\*\*\*\*\*\*\*\*\*\*\***HAVING**\*\*\*\*\*\*\*\*\*\*\*\*\*\*GROUP BY kesinlikle kullanılmalı. Where yerine kullanılır.

**SELECT birim, maas**

**FROM personel**

**GROUP BY birim**

**HAVING MAX(maas)>1200;**



**\*\*\*\*\*\*\*\*\*\*\*** **EXISTS** **\*\*\*\*\*\*\*\***2 tabloyu birleştir ve price göre veri çek.

SELECT SupplierName

FROM Suppliers

WHERE EXISTS (SELECT ProductName FROM Products WHERE Products.SupplierID = Suppliers.supplierID AND **Price < 20**);

Number of Records: 24

|  |
| --- |
| **SupplierName** |
| Exotic Liquid |
| New Orleans Cajun Delights |

**\*\*\*\*\*\*\*\*\*\*\*** **ANY** **\*\*\*\*\*\*\*\***herbiri tek tek eşit mi?Eşit

**SELECT** ProductName  
**FROM** Products  
**WHERE** ProductID = **ANY** (**SELECT** ProductID **FROM** OrderDetails **WHERE** Quantity = 10);

Number of Records: 31

|  |
| --- |
| **ProductName** |
| Chais |
| Chang |
| Chef Anton's Cajun Seasoning |
| Uncle Bob's Organic Dried Pears |

\*\*\*\*\*\*\*\*\*\*\* ALL \*\*\*\*\*\*\*\* her iki tarafta birebir eşit mi? Değil. Solda olanların tamamı sağ taraftada yok.

SELECT ProductName  
FROM Products  
WHERE ProductID = ALL (SELECT ProductID FROM OrderDetails WHERE Quantity = 10);

Number of Records: 0

|  |
| --- |
| **ProductName** |

**\*\*\*\*\*\*\*\*\*\*\*TABLONUN TAMAMINI ARDINA EKLEME\*\*\*\*\*\*\*\***suppliers(29) sin 3 alanının tamamını customers(91) a ekle. Sonuç:120

**INSERT INTO** Customers (CustomerName, City, Country)

**SELECT** SupplierName, City, Country **FROM** Suppliers;

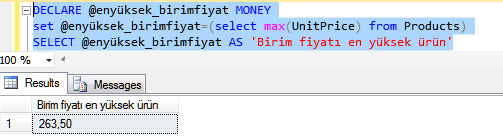
**\*\*\*\*\*\*\*\*\*\*\* NULLIF / COALESCE \*\*\*\*\*\*\*\***

**NULLIF** istemediğimiz değerleri null olarak döndürmek için.

**COALESCE** bir koşul gerçekleşmezse diğeri,o da değilse diğerine bakar.

KOMUT SATIRI: --

**\*\*\*\*\*\*\*\*\*\*\*DECLARE\*\*\*\*\*\*\*\***Değişken tanımlamada kullanılır



**\*\*\*\*\*\*\*\*\*\* VIEW \*\*\*\*\*\*\*\***

CREATE VIEW view\_adı

AS

SELECT \* FROM Customers

View, kullanıcıya Tabloya erişim izni vermeden sonuç elde etmemizi sağlar. Tabloların tamamının görünmesini engeller ve sanal tablo oluşturur. DB view ile oluşan veriyi saklamaz. Her çalıştırıldığında yeniden oluşur. VIEW üzerinde yapılan değişikler kendilerini oluşturan kaynak tabloları da etkiler ve aynı değişiklikler kaynak tablolara da yansır. Karmaşık sorguları basitleştirmek, sorgu süresini kısaltmak ve ağ üzerindeki trafiği düşürmek, erişim izinlerini düzenlemek ve farklı sunuculardaki benzer verileri karşılaştırmak içinde kullanılır.

**View in İçerisinde Yapılamayacak İşlemler Nelerdir?**

1. İsimsiz bir kolon kullanılamaz. (sql aggregate(sum,avg..) fonksiyon kullanımından sonra kolon ismi boş gelir)
2. Sql stored procedure yapısı gibi, view içerisinde parametre gönderilemez.
3. View yapısı içerisinde dml kodları (insert into, update, delete) kullanılamaz.
4. View yapısı içerisinde, sadece select ile başlayan ifadeler kullanılabilir.
5. View içerisinde order by (sıralama) fonksiyonu kullanılamaz.

**\*\*\*\*\*\*\*\*\*\* MATERİALİZED VIEW \*\*\*\*\*\*\*\***

Viewlerden farklı olarak sadece data dictionary tutulmayan aynı zamanda fiziksel olarak veri veya obje tutan viewdir. O anki veriyi kaydeder.

**\*\*\*\*\*\*\*\*\*\*** Stored Procedure**\*\*\*\*\*\*\*\*Saklı Yordam\*\*** Kullanıcı

Veritabanı serverlerında saklanan bir tür scripttir. Kullanıcı tarafından tetiklenir.

AMAÇ:Karmaşık sql kodlarını depolamak için kullanılır.

CREATE [OR REPLACE] PROCEDURE procedure\_name

[(parameter\_name [IN | OUT | IN OUT] type [, ...])]

{IS | AS}

BEGIN

< procedure\_body >

END procedure\_name;

Ör:

CREATE PROCEDURE greetings

AS

BEGIN

dbms\_output.put\_line('Hello World!');

END;

Çalıştırmak için:

EXECUTE greetings;

Silmek için:

DROP PROCEDURE greetings;

**\*\*\*\*\*\*\*\*\*\*** TRiGER**\*\*\*\*\*\*\*\*Tetikleyici (otomatik)-INSERT/UPDATE/DELETE**

Bir sorgu sonucuna göre diğer bir sorgunun çalışması. Procedure den farkı kullanıcı tarafından tetiklenmez. İki şekilde çalışır:

1. Ana sorguyu engellemek için(instead of)
2. Birlikte çalışmak için (after-for)

CREATE [OR REPLACE ] TRIGGER trigger\_name

{BEFORE | AFTER | INSTEAD OF }

{INSERT [OR] | UPDATE [OR] | DELETE}

[OF col\_name]

ON table\_name

[REFERENCING OLD AS o NEW AS n]

[FOR EACH ROW]

WHEN (condition)

DECLARE

Declaration-statements

BEGIN

Executable-statements

EXCEPTION

Exception-handling-statements

END;

\*\*\*\*\*\*\*\*\*\*\* PRIMARY KEY/UNIQUE\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PRIMARY KEY tablo alanında benzersiz kayıtların tutulmasını istediğimiz durumlarda kullanılır.(bir sütun veya alan)

- Birden fazla alan tek bir PRIMARY KEY ile tanımlanabilir. Ancak her tabloda sadece bir tane olabilir.

UNIQUE yapısı bir tabloda birden fazla olabilir.(birden fazla sütun veya alan)

- PRIMARY KEY boş kayıtlara izin vermez. UNIQUE boş kayıtlara izin verir.

- PRIMARY KEY yapısı ile tablo üzerinde bir index tanımı oluşturulur her kaydın benzersiz bir tanımı yapılır.

FOREIGN KEY: 2 tablo arasında bağ kurmak gerektiğinde kullanılır. Yardımcı index oluşturmak için kullanılır.

\*\*\*\*\*\*\*\*\*\*\*\*DATABASE\*\*\*\*\*\*\*\*\*\*\*\*

DDL - Data Definition L.(Veri Tanımlama Dili): CREATE TABLE/ ALTER TABLE/ DROP TABLE \*\*(CAD)\*\*

DQL - Data Query Language: SELECT

DML - Data Manipulation L.(Veri İşlemeD.): DELETE/ UPDATE/ İNSERT

DCL - Data Control L.(Veri KontrolD.): GRANT/ DENY/ REVOKE

TCL - Transaction Control L.(İşlem KontrolD.) : COMMIT/ ROLLBACK/ SAVEPOİNT

fiziksel(log-dosyalar) ve Mantıksal(tablo, sıra, eşanlamlı) 2 ye ayrılır

Kesin Sayısal Veri Tipleri (Exact numerics)

bit: 1 bayt yer kaplar. 1 ve 0 değerlerini alır.true/false değerlerini bu tipte saklayabiliriz.

tinyint: 1 bayt 0 ile 255 arası tam sayıları tutar.

smallint: 2 bayt -32.768 ile 32.767 arası tam sayıları tutar.

int: 4 bayt -2.147.483.648 ile 2.147.483.647 arası tam sayıları tutar

bigint: 8 bayt -9.223.372.036.854.775.808 ile 9.223.372.036.854.775.807 arası tam sayıları tutar.

char sabit.fazlası boşlukla tamamlanır. n x 1 bayt

varchar fazlası sayılmaz. kaydedilenVerininKarakterSayisi x 1 bayt) + 2 bayt

\*\*\*\*\*\*\*\*\*\*\* Tekrar eden sorguları ekrana getirme\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
SELECT KategoriAd FROM Kategori GROUP BY KategoriAd HAVING COUNT(\*) > 1

\*\*\*\*\*\*\*\*\*\*\* DATABASE tamamında mevcut alanı(misal: address) bulmak\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 select    TABLE\_CATALOG, TABLE\_SCHEMA, TABLE\_NAME, COLUMN\_NAME    
 from      INFORMATION\_SCHEMA.COLUMNS      
 where    COLUMN\_NAME like '%address%'

SELECT \* FROM İNFORMATİON\_SCHEMA.TABLES WHERE TABLE\_NAME='İNVOİCE'; (Schema altındaki tablolardan invoice olanları getir)