Veri Tabanı Yönetim Sistemleri

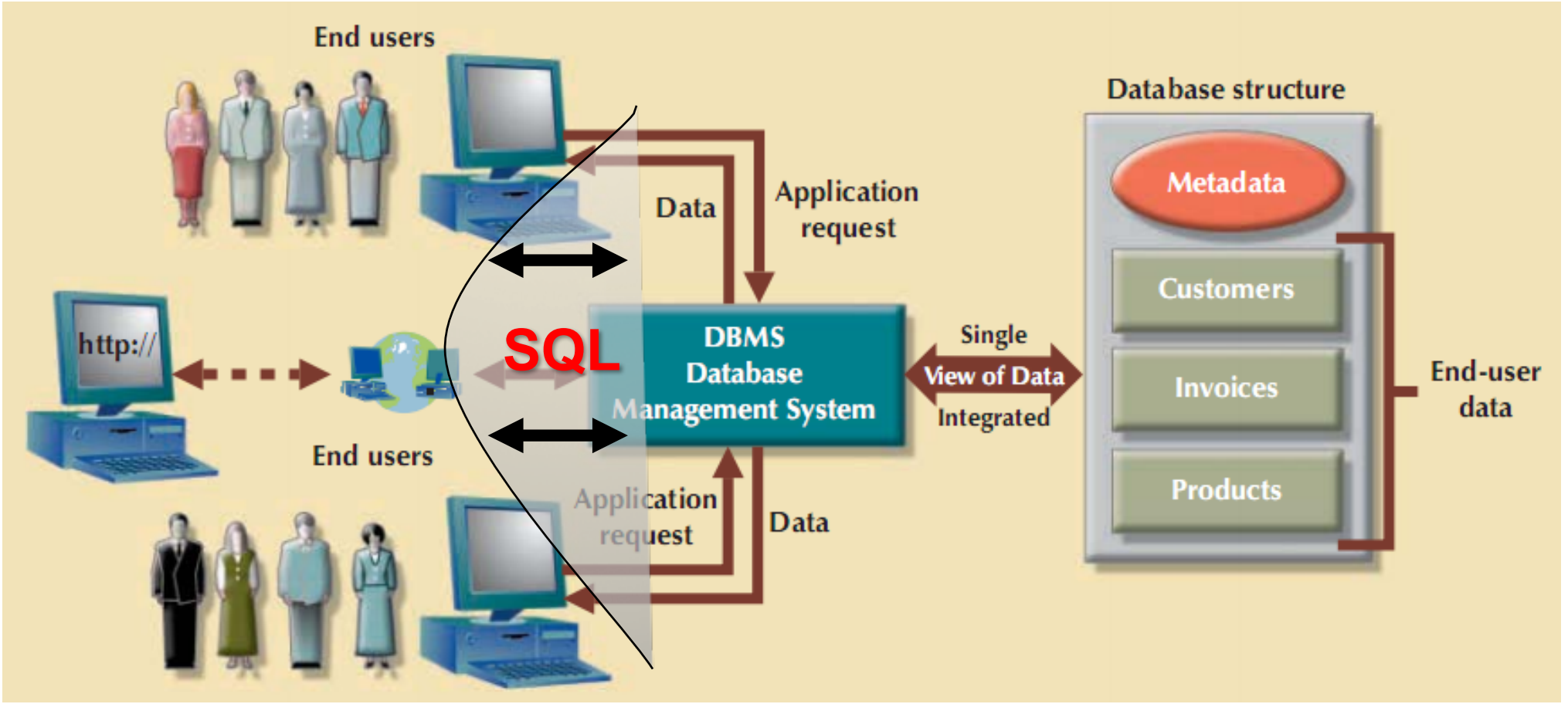
(Veri Tabanı Tasarımı)

SQL (Structured Query Language)

Konular

- ✓ Yapısal SQL Komutları
- ✓ Gruplama İşlemi
- ✓ SQL Fonksiyonları
- ✓ Kaynaklar

➢SQL (Structured Query Language)



Carlos Coronel, Steven Morris, and Peter Rob, Database Systems: Design, Implementation, and Management, sayfa 8.

SQL Fonksiyonları 2 bölüme ayrılabilir:

- ✓ Data definition language (DDL) (Yapısal Komutlar)
  - ✓ Veritabanı/tablo/ilişki v.s. oluşturma/değiştirme/silme v.s.
- ✓ Data Manipulation Language (DML) (Veri ekleme/silme/güncelleme/sorgulama v.s. komutları)

Yapısal SQL Komutları (DDL)

CREATE DATABASE

•Veritabanı oluşturmak için kullanılır.

CREATE DATABASE ***dbname*;**

```
CREATE DATABASE "OnlineStore"
ENCODING='UTF-8'
LC_COLLATE='tr_TR.UTF-8'
LC_CTYPE='tr_TR.UTF-8‘
OWNER postgres
TEMPLATE=template0;
```

CREATE SCHEMA

Şema oluşturmak için kullanılır.

CREATE SCHEMA ***schemaname*;**

```
CREATE SCHEMA sema3;
```

CREATE TABLE

Tablo oluşturmak için kullanılır.

```
CREATE TABLE "public"."Urunler" (
    "urunNo" SERIAL,
    "kodu" CHAR(6) NOT NULL,
    "adi" VARCHAR(40)  NOT NULL,
    "uretimTarihi" DATE DEFAULT '2000-01-01',
    "birimFiyati" MONEY,
    "miktari" SMALLINT DEFAULT '0',
    CONSTRAINT "urunlerPK" PRIMARY KEY ("urunNo"),
    CONSTRAINT "urunlerUnique" UNIQUE ("kodu"),
    CONSTRAINT "urunlerCheck" CHECK (miktari >= 0)
);
```

Veri Tipleri

Name	Aliases	Description
bigint	int8	signed eight-byte integer
bigserial	serial8	autoincrementing eight-byte integer
bit [(*n*)]		fixed-length bit string
bit varying [(*n*)]	varbit	variable-length bit string
boolean	bool	logical Boolean (true/false)
box		rectangular box on a plane
bytea		binary data ("byte array")
character [(*n*)]	char [(*n*)]	fixed-length character string
character varying [(*n*)]	varchar [(*n*)]	variable-length character string
cidr		IPv4 or IPv6 network address
circle		circle on a plane
date		calendar date (year, month, day)
double precision	float8	double precision floating-point number (8 bytes)
inet		IPv4 or IPv6 host address
integer	int, int4	signed four-byte integer
interval [*fields*] [(*p*)]		time span
json		JSON data
line		infinite line on a plane
lseg		line segment on a plane
macaddr		MAC (Media Access Control) address
money		currency amount
numeric [(*p*, *s*)]	decimal [(*p*, *s*)]	exact numeric of selectable precision
path		geometric path on a plane
point		geometric point on a plane
polygon		closed geometric path on a plane

http://www.postgresql.org/docs/9.5/static/datatype.html
http://www.tutorialspoint.com/postgresql/postgresql_data_types.htm

Veri Tipleri

real	float4	single precision floating-point number (4 bytes)
smallint	int2	signed two-byte integer
smallserial	serial2	autoincrementing two-byte integer
serial	serial4	autoincrementing four-byte integer
text		variable-length character string
time [(*p*)] [without time zone]		time of day (no time zone)
time [(*p*)] with time zone	timetz	time of day, including time zone
timestamp [(*p*)] [without time zone]		date and time (no time zone)
timestamp [(*p*)] with time zone	timestamptz	date and time, including time zone
tsquery		text search query
tsvector		text search document
txid_snapshot		user-level transaction ID snapshot
uuid		universally unique identifier
xml		XML data

http://www.postgresql.org/docs/9.5/static/datatype.html
http://www.tutorialspoint.com/postgresql/postgresql_data_types.htm

SEQUENCE

```
CREATE SEQUENCE "sequence1";

ALTER SEQUENCE sequence1 OWNED BY "Urunler"."urunNo"; --Urunler
tablosundaki UrunNo alanı yok edildiğinde sequence de yok edilsin diye...

SELECT NEXTVAL('sequence1');
```

Function	Return Type	Description
currval(regclass)	bigint	Return value most recently obtained with nextval for specified sequence
lastval()	bigint	Return value most recently obtained with nextval for any sequence
nextval(regclass)	bigint	Advance sequence and return new value
setval(regclass, bigint)	bigint	Set sequence's current value
setval(regclass, bigint, boolean)	bigint	Set sequence's current value and is_called flag

```
INSERT INTO "Urunler" ("urunNo", "adi", "birimFiyati", "uretimTarihi", "miktari")
VALUES (NEXTVAL('sequence1'), 'TV', '13', '1.1.2000', 5)
```

SEQUENCE

Tablo oluştururken sequence kullanma.

```
CREATE TABLE "public"."Urunler" (
    "urunNo" INTEGER DEFAULT NEXTVAL('sequence1'),
    "kodu" CHAR(6) NOT NULL,
    "adi" VARCHAR(40)  NOT NULL,
    "uretimTarihi" DATE DEFAULT '2000-01-01',
    "birimFiyati" MONEY,
    "miktari" SMALLINT DEFAULT '0',
    CONSTRAINT "urunlerPK" PRIMARY KEY ("urunNo"),
    CONSTRAINT "urunlerUnique" UNIQUE ("kodu"),
    CONSTRAINT "urunlerCheck" CHECK (miktari >= 0)
);
```

SQL KISITLARI (CONSTRAINTS)

Veri bütünlüğünün korunmasına yardımcı olurlar

❖NOT NULL

•Bu kısıt tanımlandığı alanın boş olamayacağını belirtir.

```
CREATE TABLE PersonsNotNull
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

❖DEFAULT

•“Default” tanımlandığı alana bir değerin girilmemesi durumunda alana varsayılan bir değerin atanmasını sağlayan kısıtlamadır.

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255) DEFAULT 'Sandnes'
)
```

•Varolan tablodaki sütuna “Default” kısıtını eklemek için aşağıdaki komut kullanılır.

```
ALTER TABLE Persons
  ALTER City SET DEFAULT 'SANDNES'
```

•Varolan tablodan “Default” kısıtını kaldırmak için aşağıdaki komut kullanılır.

```
ALTER TABLE Persons
ALTER City DROP DEFAULT
```

❖PRIMARY KEY

- "Primary key" kısıtı tanımlandığı alan için verilerin her kayıtta farklı olacağını belirtir.
- Tabloda "unique" ve "not null" tanımlanır.

```
CREATE TABLE PersonsUnique
(
P_Id int ,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CONSTRAINT PK_PersonID PRIMARY KEY(P_ID)
)
```

- Varolan tabloya "primary key" eklemek için aşağıdaki komut kullanılır.

```
ALTER TABLE Persons
  ADD CONSTRAINT PK_PersonID PRIMARY KEY(P_ID)
```

- Varolan tablodan "primary key" kaldırmak için aşağıdaki komut kullanılır.

```
ALTER TABLE Persons
  DROP CONSTRAINT PK_PersonID
```

❖UNIQUE

•“Unique” kısıtı tanımlandığı alandaki verilerin tekil, benzersiz olmasını sağlar.

```
CREATE TABLE PersonsUniqueMulti
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CONSTRAINT uc_PersonID UNIQUE (P_Id, LastName)
)
```

•Varolan tabloya benzersiz alanlar eklemek için aşağıdaki komut kullanılır.

```
ALTER TABLE "SiparisDetay"
ADD CONSTRAINT "urunSiparisUnique" UNIQUE ("urunNo", "siparisNo");
```

•Varolan tablodaki benzersiz alanları kaldırmak için aşağıdaki komut kullanılır.

```
ALTER TABLE Persons
  DROP CONSTRAINT uc_PersonID
```

❖CHECK

•“Check” tablodaki sütun için değer aralığını sınırlamada kullanılır.

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CONSTRAINT CH_PID CHECK (P_Id > 0)
)
```

•Varolan tablodaki sütuna “Check” kısıtını eklemek için aşağıdaki komut kullanılır.

```
ALTER TABLE Persons
ADD CONSTRAINT CH_PID CHECK (P_Id > 0)
```

•Varolan tablodan “check” kısıtını kaldırmak için aşağıdaki komut kullanılır.

```
ALTER TABLE Persons
DROP CONSTRAINT CH_PID
```

❖FOREIGN KEY

- Tablolar arasındaki ilişkileri tanımlayan bir kısıttır.

```
CREATE TABLE Orders
(
O_Id int NOT NULL,
OrderNo int NOT NULL,
P_Id int,
CONSTRAINT FK_PID FOREIGN KEY (P_Id) REFERENCES Persons (P_Id)
)
```

•Varolan tabloya yabancı anahtar eklemek için aşağıdaki komut kullanılır.

```
ALTER TABLE Orders
ADD CONSTRAINT FK_PID FOREIGN KEY (P_Id) REFERENCES Persons (P_Id)
```

•Varolan tablodan yabancı anahtar kaldırmak için aşağıdaki komut kullanılır.

```
ALTER TABLE Orders
DROP CONSTRAIN FK_PID
```

➢CREATE INDEX

•Tabloyu indekslemek için kullanılır.

•İndeks veritabanındaki tabloda bulunan bütün verileri kontrol etmeden, verilere daha hızlı ulaşmamızı sağlar.

•Aşağıdaki komut “Persons” tablosundaki “Lastname” sütununu indeksler.

```
CREATE INDEX PIndex
  ON Persons (LastName)
```

➢DROP TABLE

•Tabloyu silmek için kullanılır.

```
DROP TABLE table_name
```

➢DROP DATABASE

•Veritabanını silmek için kullanılır.

```
DROP DATABASE database_name
```

➢TRUNCATE TABLE

•Tablonun kendisini silmeden içindeki verileri silmek için kullanılır.

```
TRUNCATE TABLE table_name
```

➢ALTER TABLE

•Var olan tablodaki sütunlara ekleme, silme, düzelme işlemi yapmak için kullanılır.

•Tabloya sütun eklemek için:

```
ALTER TABLE table_name
ADD column_name datatype
```

•Tablodan sütun silmek için:

```
ALTER TABLE table_name
DROP COLUMN column_name
```

➢ALTER TABLE

•“Persons” tablosu:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

•Tabloya “DateOfBirth” sütununu eklemek için aşağıdaki SQL kullanılır:

```
ALTER TABLE Persons
  ADD DateOfBirth date
```

P_Id	LastName	FirstName	Address	City	DateOfBirth
1	Hansen	Ola	Timoteivn 10	Sandnes	
2	Svendson	Tove	Borgvn 23	Sandnes	
3	Pettersen	Kari	Storgt 20	Stavanger	

•“DateOfBirth” sütununun veri tipini değiştirmek için aşağıdaki SQL kullanılır:

```
ALTER TABLE Persons
  ALTER COLUMN DateOfBirth year
```

➢ALTER TABLE

•"Persons" tablosundaki "DateOfBirth" sütununu silmek için aşağıdaki SQL kullanılır:

```
ALTER TABLE Persons
  DROP COLUMN DateOfBirth
```

•Yukarıdaki SQL kullanılırsa "Persons" tablosu aşağıdaki gibi olur.

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

➢NULL VALUES

•Defaultta tablo sütunları null değerleri tutar.
•Tablodaki sütunlar optional ise varolan veriler üzerinde silme ve güncelleme işlemi yapabiliriz.
•Sütundaki verilere Null değerler verebiliriz.

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola		Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari		Stavanger

•Null değerleri =,<,>,<> gibi karşılaştırma operatorleri ile test edemeyiz.Bunun yerine IS NULL ve IS NOT NULL kullanılır.

❖IS NULL

•Aşağıdaki SQL adresi NULL olan değerleri döndürür.

```
SELECT LastName,FirstName,Address FROM Persons
  WHERE Address IS NULL
```

LastName	FirstName	Address
Hansen	Ola	
Pettersen	Kari	

❖IS NOT NULL

•Aşağıdaki SQL adresi NULL olmayan değerleri döndürür.

```
SELECT LastName,FirstName,Address FROM Persons
    WHERE Address IS NOT NULL
```

LastName	FirstName	Address
Svendson	Tove	Borgvn 23

➢SQL FONKSİYONLARI

❖ÇOKLU SATIR FONKSİYONLARI

•Bir sütundaki tüm satırlara uygulanırlar.
•Bir hesap yapıp geriye tek bir değer döndürürler.

✓AVG() - Aritmetik Ortalama

```
8  SELECT avg("UnitPrice") FROM "products"
9
10
11
```

avg
29.0422368300589

```
9  SELECT sum("UnitPrice")/count("ProductID") AS "Ortalama Birim Fiyat" FROM "products"
10
11
```

Ortalama Birim Fiyat
29.0422395405016

❖ÇOKLU SATIR FONKSİYONLARI

✓COUNT ()

• Count fonksiyonu ile oluşturduğumuz sorgunun ürettiği satır sayısı bulunur ve geriye döndürülür.

•Eğer COUNT fonksiyonunu sadece bir sütun için uygulanırsa NULL olmayan kayıtların sayısı bulunur.

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10265	7	2	1996-07-25	1
10266	87	3	1996-07-26	3
10267	25	4	1996-07-29	1

```
SELECT COUNT(CustomerID) AS OrdersFromCustomerID7 FROM Orders
  WHERE CustomerID=7;
```

OrdersFromCustomerID7
4

➢SQL FONKSİYONLARI

❖ÇOKLU SATIR FONKSİYONLARI

✓COUNT ()

• Aşağıdaki SQL ifadesi “Orders” tablosundaki kayıtların sayısını döndürür.

```
SELECT COUNT(*) AS NumberOfOrders FROM Orders;
```

NumberOfOrders
196

• Aşağıdaki SQL ifadesi “Orders” tablosundaki tekrar etmeyen kayıtların sayısını döndürür.

```
SELECT COUNT(DISTINCT CustomerID) AS NumberOfCustomers FROM Orders;
```

➢SQL FONKSİYONLARI

❖ÇOKLU SATIR FONKSİYONLARI

✓LIMIT

• Seçilen sütundaki ilk x satıra (order by DESC ile son x satıra) ulaşmak için kullanılır.

```
16  SELECT * FROM "products"
17  ORDER BY "ProductID"  LIMIT 5
18
```

ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInStock	UnitsOr
1	Chai	8	1	10 boxes x 30 b...	18	39	
2	Chang	1	1	24 - 12 oz bottles	19	17	
3	Aniseed Syrup	1	2	12 - 550 ml bott...	10	13	
4	Chef Anton's ...	2	2	48 - 6 oz jars	22	53	
5	Chef Anton's ...	2	2	36 boxes	21.35	0	

➢SQL FONKSİYONLARI

❖ÇOKLU SATIR FONKSİYONLARI

✓MAX()

• Seçilen sütundaki en büyük değere ulaşmak için kullanılır.

ProductID	ProductName	SupplierID	CategoryID	Unit	Price
1	Chais	1	1	10 boxes x 20 bags	18
2	Chang	1	1	24 - 12 oz bottles	19
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	21.35
5	Chef Anton's Gumbo Mix	2	2	36 boxes	25

```
SELECT MAX(Price) AS HighestPrice FROM Products;
```

HighestPrice
263.5

✓MIN ()

• Seçilen sütundaki en küçük değere ulaşmak için kullanılır.

```
SELECT MIN(Price) AS SmallestOrderPrice FROM Products;
```

SmallestOrderPrice
2.5

❖ÇOKLU SATIR FONKSİYONLARI

✓SUM ()

• Seçilen sütundaki tüm kayıtların toplamı hesaplanıp geri döndürülür.

OrderDetailID	OrderID	ProductID	Quantity
1	10248	11	12
2	10248	42	10
3	10248	72	5
4	10249	14	9
5	10249	51	40

```
SELECT SUM(Quantity) AS TotalItemsOrdered FROM OrderDetails;
```

TotalItemsOrdered
12743

➢GROUP BY

Sorgu sonucunu belirtilen alan(lar)a göre gruplar.
Aşağıdaki sorgu Ürünleri kategorilerine göre gruplar ve her kategoride bulunan ürünlerin sayısını hesaplayarak kategoriyle birlikte döndürür.

```
7  SELECT "CategoryID", count("CategoryID") AS "Ürün Sayısı" FROM "products"
8  GROUP BY "CategoryID"
9
10
11
```

CategoryID	Ürün Sayısı
6	6
4	10
8	12
5	7
1	12
2	11
7	5
3	13

Seçilecek alan, gruplama yapılan alan (CategoryID) ya da çoklu satır fonksiyonları (count) olmalı
Gruplanan alanla ilgili koşul yazılabilmesi için Having kullanılması gereklidir.

➢GROUP BY

Aşağıdaki sorgu Ürünleri kategorilerine göre gruplar ve kategorilerdeki ürünlerin birim fiyatlarının toplamını kategori adıyla birlikte döndürür.

```
SELECT CategoryName, SUM(UnitPrice)
FROM Categories INNER JOIN
Products ON Categories.CategoryID = Products.CategoryID
GROUP BY CategoryName
```

```
29  SELECT   "public"."customers"."CompanyName", count("OrderID")
30  FROM     "orders"
31  INNER JOIN "customers"  ON "orders"."CustomerID" = "customers"."CustomerID"
32  GROUP BY "CompanyName"
33  ORDER BY "CompanyName"
```

	CompanyName	count
1	Alfreds Futterk...	6
2	Ana Trujillo E...	4
3	Antonio More...	7
4	Around the Ho...	13
5	Berglunds sna...	18
6	Blauer See Del...	7
7	Blondesddsl p...	11
8	Bólido Comida...	3

➢GROUP BY

•Veri grupları üzerinde işlem yaparak her grup için bir sonuç üreten fonksiyondur.
•Çoklu Satır Fonksiyonları ile birlikte kullanılır.

•“ Shippers” tablomuz:

ShipperID	ShipperName	Phone
1	Speedy Express	(503) 555-9831
2	United Package	(503) 555-3199
3	Federal Shipping	(503) 555-9931

• “Orders” tablomuz:

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10248	90	5	1996-07-04	3
10249	81	6	1996-07-05	1
10250	34	4	1996-07-08	2

•“ Employees” tablomuz:

EmployeeID	LastName	FirstName	BirthDate	Photo	Notes
1	Davolio	Nancy	1968-12-08	EmpID1.pic	Education includes a BA..
2	Fuller	Andrew	1952-02-19	EmpID2.pic	Andrew received his BTS.
3	Leverling	Janet	1963-08-30	EmpID3.pic	Janet has a BS degree....

➢GROUP BY

•Aşağıdaki SQL “ ShipperName” sütununa göre “OrderID” sayısını hesaplar.

```
SELECT Shippers.ShipperName,COUNT(Orders.OrderID) AS NumberOfOrders FROM Orders
    LEFT JOIN Shippers
      ON Orders.ShipperID=Shippers.ShipperID
        GROUP BY ShipperName;
```

ShipperName	NumberOfOrders
Federal Shipping	68
Speedy Express	54
United Package	74

```
SELECT Shippers.ShipperName, Employees.LastName,
   COUNT(Orders.OrderID) AS NumberOfOrders FROM ((Orders
      INNER JOIN Shippers
         ON Orders.ShipperID=Shippers.ShipperID)
      INNER JOIN Employees
         ON Orders.EmployeeID=Employees.EmployeeID)
               GROUP BY ShipperName,LastName;
```

ShipperName	LastName	NumberOfOrders
Federal Shipping	Buchanan	5
Federal Shipping	Callahan	11
Federal Shipping	Davolio	13
Federal Shipping	Dodsworth	4

➢HAVING

•Gruplandırılmış verilerde filtreleme amaçlı kullanılır.
•Çoklu satır fonksiyonlarıyla birlikte “WHERE” ifadesi kullanılmaz, onun yerine “HAVING” kullanılır.
•HAVING ifadesi ile koşul yazarken gruplama fonksiyonları ya da gruplama yapılan alan kullanılabilir.

```
26  SELECT "Country", count("CustomerID") AS "Müşteri Sayısı" FROM "customers"
27  GROUP BY "Country"
28  HAVING count("CustomerID")>5
29
```

Country	Müşteri Sayısı
Germany	13
France	14
Spain	6
UK	9
Brazil	10
USA	17

➢HAVING

•“ Employees” tablomuz:

EmployeeID	LastName	FirstName	BirthDate	Photo	Notes
1	Davolio	Nancy	1968-12-08	EmpID1.pic	Education includes a BA..
2	Fuller	Andrew	1952-02-19	EmpID2.pic	Andrew received his BTS.
3	Leverling	Janet	1963-08-30	EmpID3.pic	Janet has a BS degree....

• “Orders” tablomuz:

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10248	90	5	1996-07-04	3
10249	81	6	1996-07-05	1
10250	34	4	1996-07-08	2

```
SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOfOrders FROM (Orders
    INNER JOIN Employees
        ON Orders.EmployeeID=Employees.EmployeeID)
            GROUP BY LastName
                HAVING COUNT(Orders.OrderID) > 10;
```

LastName	NumberOfOrders
Buchanan	11
Callahan	27
Davolio	29
Fuller	20

➢HAVING

```
SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOfOrders FROM Orders
  INNER JOIN Employees
    ON Orders.EmployeeID=Employees.EmployeeID
      WHERE LastName='Davolio' OR LastName='Fuller'
        GROUP BY LastName
          HAVING COUNT(Orders.OrderID) > 25;
```

LastName	NumberOfOrders
Davolio	29

Kaynaklar

http://www.postgresql.org/docs/9.3/static/datatype.html
http://www.tutorialspoint.com/postgresql/postgresql_data_types.htm
http://www.w3schools.com/sql