1. TABLO OLUSTURMA.

CREATE TABLE aile\_tablosu(

kod\_no char(2),

isim varchar2(15) PRIMARY KEY,

soyisim varchar2(15),

yas number(2),

maas char(4),

sehir varchar2(15)

);

**=============================================================================**

1. UC TURLU PRIMARY KEY OLUSTURMA(YANDA/EN ALT SATIRDA/BIRDEN FAZLA SUTUN KULLANARAK).

CREATE TABLE ev\_durumu(

kod\_no char(2),

isim varchar2(15),

ev\_fiyat number(10),

kiraci char(2),

ev\_sahibi char(2),

CONSTRAINTS kod\_no\_pk PRIMARY KEY(kod\_no)

);

**=============================================================================**

1. FOREIGN KEY OLUSTURMA VE SUTUNLARDAN BAZILARINI UNIQUE YA DA NOT NULL YAPMA

CREATE TABLE is\_durumu(

isim varchar2(15) NOT NULL,

kod\_no char(2) UNIQUE,

is\_bilgisi varchar2(15),

CONSTRAINTS is\_isim\_pk PRIMARY KEY (isim,is\_bilgisi),

CONSTRAINTS isim\_fk FOREIGN KEY (isim) REFERENCES aile\_tablosu(isim)

);

1. VERI GIRISI YAPMA/EKSIK VERI GIRISI YAPMA

TUM SUTUNLARA:

INSERT INTO aile\_tablosu VALUES('1','AYDIN','CAN',72,'3300','ANKARA');

BELIRLI SUTUNLARA:

INSERT INTO aile\_tablosu (kod\_no,isim,soyisim,yas,sehir) VALUES ('1','AYDIN2','CAN',72,'ANKARA');

TEK BIR SUTUNA VERI GIRME:

INSERT INTO ev\_durumu (kod\_no)VALUES('2');

TARIH GIRME

INSERT INTO students VALUES(‘234567890’, ‘Veli Han’, ‘Istanbul’, 95, ‘13-Aug-20’);

**=============================================================================**

1. TABLOYU KONSOLDA GORMEK ICIN KOMUT

TUM TABLE ICIN:

SELECT \*

FROM aile\_tablosu;

\*\*\*\* \*\*\*\* \*\*\*\*

ISTENEN BELIRLI BIR SATIR ICIN:

SELECT \*

FROM ev\_durumu

WHERE isim='MEHMET';

\*\*\*\* \*\*\*\* \*\*\*\*

ISTENEN BELIRLI SATIRLARI GORMEK ICIN **OR** KULLANILIR. (BIRDEN FAZLA ISE):

SELECT \*

FROM ev\_durumu

WHERE kod\_no='1' OR isim='MEHMET';

\*\*\*\* \*\*\*\* \*\*\*\*

ISTENEN BELIRLI SATIRLARI GORMEK ICIN **OR** YERINE **IN** KULLANILABILIR:

SELECT \*

FROM ev\_durumu

WHERE kod\_no IN ('1','3') \* IN sagina kume olustururuz.

BIRDEN FAZLA SATIRI GORMEK ICIN < ya da **>** ISARETLERI KULANILABILIR.

SELECT \*

FROM products

WHERE supplier\_id < 103;

TABLODAKI HERHANGI BIR **TEK SUTUNUN** TAMAMINI GORMEK ICIN:

SELECT kod\_no

FROM ev\_durumu;

TABLODAKI **BIRDEN FAZLA SUTUNU** GORMEK ICIN:

SELECT kod\_no,isim

FROM ev\_durumu;

TABLODA **BIRDEN FAZLA SUTUN VE 1 SATIR** GORMEK ICIN KOMUT:

SELECT kod\_no,isim

FROM ev\_durumu

WHERE kiraci='E';

TABLODA **BIRDEN FAZLA SATIR VE 1 SUTUN** GORMEK ICIN KOMUT:

SELECT ev\_fiyat

FROM ev\_durumu

WHERE kod\_no IN('1','2');

**=============================================================================**

1. TABLODAKI BIR VERIYI GUNCELLEMEK/DATASINI DEGISTIRMEK ICIN KOMUT:

UPDATE ev\_durumu

SET ev\_fiyat=450000

WHERE kod\_no='1';

\*\*\*\* \*\*\*\* \*\*\*\*

UPDATE aile\_tablosu

SET yas=88

WHERE isim='AYDIN2';

\*\*\*\* \*\*\*\* \*\*\*\*

\*\*product\_id'si 1001 olan product'in product\_name'ini Kemal Can'in temsilcisi oldugu marka yapin.

UPDATE products

SET product\_name=(SELECT supplier\_name

FROM suppliers

WHERE contact\_name='Kemal Can')

WHERE product\_id=1001;

**=============================================================================**

1. TABLE’DAN SATIR SILMENIN **BIRINCI** KOMUTU: **DELETE**

\* Silme isleminden sonra bazi komutlarla silme islemi **GERI ALINABILIR.**

* 1. **DELETE: \* tek satiri silme**

DELETE FROM ev\_durumu

WHERE kod\_no=2;

* 1. **DELETE: \*tum satirlari silme**

DELETE FROM ev\_durumu;

* 1. **DELETE: \*sartli slime**

DELETE FROM ev\_durumu

WHERE kod\_no<'3';

* 1. **DELETE:\*AND ile silme**

DELETE FROM ev\_durumu

WHERE kod\_no='1' AND isim='ALI';

\*Bu iki sartin da saglanmasi lazim.

* 1. **DELETE: \*OR ile slime**

DELETE FROM ev\_durumu

WHERE kod\_no='1' AND isim='ALI';

\*Bu iki sartin hangisi saglanirsa saglanani siler.

1. TABLE’DAN SATIR SILMENIN **IKINCI** KOMUTU: **TRUNCATE**

\* Silme isleminden sonra silme islemi **GERI ALINAMAZ**.

\* Tablodaki tum satirlari siler.

TRUNCATE TABLE ev\_durumu;

1. TABLE’DAN SATIR SILMENIN **UCUNCU** KOMUTU: **DROP**

\* Bir **table'i yapisi ve icindeki datalari** ile silmek icin kullanilir.

\* Silme isleminden sonra bazi komutlarla silme islemi **GERI ALINABILIR.**

DROP TABLE ev\_durumu;

\* Silme isleminden sonra **GERI ALINMASI ISTENMIYORSA:**

DROP TABLE ev\_durumu PURGE;

**=============================================================================**

CREATE TABLE employees

(

id number(9),

name varchar2(50),

state varchar2(50),

salary number(20),

company varchar2(20)

);

INSERT INTO employees VALUES(123456789, 'John Walker', 'Florida', 2500, 'IBM');

INSERT INTO employees VALUES(234567890, 'Brad Pitt', 'Florida', 1500, 'APPLE');

INSERT INTO employees VALUES(345678901, 'Eddie Murphy', 'Texas', 3000, 'IBM');

INSERT INTO employees VALUES(456789012, 'Eddie Murphy', 'Virginia', 1000, 'GOOGLE');

INSERT INTO employees VALUES(567890123, 'Eddie Murphy', 'Texas', 7000, 'MICROSOFT');

INSERT INTO employees VALUES(456789012, 'Brad Pitt', 'Texas', 1500, 'GOOGLE');

INSERT INTO employees VALUES(123456710, 'Mark Stone', 'Pennsylvania', 2500, 'IBM');

CREATE TABLE companies

(

company\_id number(9),

company varchar2(20),

number\_of\_employees number(20)

);

​

INSERT INTO companies VALUES(100, 'IBM', 12000);

INSERT INTO companies VALUES(101, 'GOOGLE', 18000);

INSERT INTO companies VALUES(102, 'MICROSOFT', 10000);

INSERT INTO companies VALUES(100, 'APPLE', 21000);

\* Isci sayisi 15000'den fazla olan company'lerin company isimlerini ve isci isimlerini gosteren kod

SELECT company,name

FROM employees

WHERE company IN (SELECT company

FROM companies

WHERE number\_of\_employees>15000);

\*\*\*\* \*\*\*\* \*\*\*\*

\* Company\_id'si 102 den kucuk olan company'lerin salary'lerini ve state'lerini gosteren kod

SELECT salary,state

FROM employees

WHERE company IN(SELECT company

FROM companies

WHERE company\_id<102);

\*\*\*\* \*\*\*\* \*\*\*\*

\* Florida'daki company'lerin company\_id'lerini ve isci sayilarini gosteren kodu yaziniz.

SELECT company\_id,number\_of\_employees

FROM companies

WHERE company IN ( SELECT company

FROM employees

WHERE state='Florida');

**\* NOT:** SUBQUERY'ler WHERE'den sonra kullanilabildigi gibi SELECT'den sonra da kullanilabilirler.

\* Her sirketteki isci sayisini ve ortalama isci ucretlerini gosteren kodu yaziniz.

SELECT company,number\_of\_employees,(SELECT AVG (salary)

FROM employees

WHERE employees.company=companies.company) avarage\_salary

FROM companies;

\* Her sirketteki toplam state sayisini gosteren kodu yaziniz ( COUNT(state) )

SELECT company,(SELECT COUNT (state)

FROM employees

WHERE companies.company=employees.company) number\_of\_states

FROM companies;

\* SUM, COUNT, AVG, MIN, MAX gibi fonksiyonlar AGGREGATE fonksiyonlar olarak adlandirilir.

\* Her sirketin company\_id'sini ve iscilerine yaptigi toplam odemeyi gosteren kodu yaziniz.

SELECT company,company\_id, (SELECT SUM(salary)

FROM employees

WHERE companies.company=employees.company) total\_salary

FROM companies;

CREATE TABLE customers\_products

(

product\_id number(10),

customer\_name varchar2(50),

product\_name varchar2(50)

);

INSERT INTO customers\_products VALUES (10, 'Mark', 'Orange');

INSERT INTO customers\_products VALUES (10, 'Mark', 'Orange');

INSERT INTO customers\_products VALUES (20, 'John', 'Apple');

INSERT INTO customers\_products VALUES (30, 'Amy', 'Palm');

INSERT INTO customers\_products VALUES (20, 'Mark', 'Apple');

INSERT INTO customers\_products VALUES (10, 'Adem', 'Orange');

INSERT INTO customers\_products VALUES (40, 'John', 'Apricot');

INSERT INTO customers\_products VALUES (20, 'Eddie', 'Apple');

CREATE TABLE customers\_likes

(

product\_id number(10),

customer\_name varchar2(50),

liked\_product varchar2(50)

);

INSERT INTO customers\_likes VALUES (10, 'Mark', 'Orange');

INSERT INTO customers\_likes VALUES (50, 'Mark', 'Pineapple');

INSERT INTO customers\_likes VALUES (60, 'John', 'Avocado');

INSERT INTO customers\_likes VALUES (30, 'Lary', 'Cherries');

INSERT INTO customers\_likes VALUES (20, 'Mark', 'Apple');

INSERT INTO customers\_likes VALUES (10, 'Adem', 'Orange');

INSERT INTO customers\_likes VALUES (40, 'John', 'Apricot');

INSERT INTO customers\_likes VALUES (20, 'Eddie', 'Apple');

\* EXISTS komutu SUBQUERY'ler ile beraber kullanilir.

\* IN komutu OR komutunun kisa yazilmis halidir.

\* IN komutunu tek basina SUBQUERY'lerle beraber kullanmayiz.

\* SUBQUERY kullanacaksaniz EXISTS kullanmaniz gerekir.

\* Product\_id'leri ayni olan musterilerin isimlerini gosteriniz.

SELECT customer\_name

FROM customers\_products

WHERE EXISTS (SELECT product\_id

FROM customers\_likes

WHERE customers\_products.product\_id=customers\_likes.product\_id);

\* Product\_name ile liked\_product'i ayni olan musterilerin isimlerini gosteren kod

SELECT customer\_name

FROM customers\_products

WHERE EXISTS (SELECT customer\_name

FROM customers\_likes

WHERE customers\_likes.liked\_product=customers\_products.product\_name);

1. BETWEEN KOMUTU

\* BETWEEN(Arasinda ==> 10 ile 20 arasinda:

BETWEEN 10 AND 20) Komutu AND komutunun daha anlasilir halidir.

\* BETWEEN **komutunda sinirlar dahildir**.

\*\* Alfabetik siraya gore dizer.

\* BETWEEN kullanirken ilk datanin ikinciden kucuk olmasina dikkat edin.

\*\* product\_id'si 20 ile 40 arasinda olan product'larin isimlerini ve product\_id'lerini gosteren kodu yaziniz.

SELECT product\_name, product\_id

FROM customers\_products

WHERE product\_id BETWEEN 20 AND 40;

\*\*Ismi J'den T'ye kadar harflerle baslayan musterilerin tum bilgilerini gosteren kodu yaziniz.

SELECT \*

FROM customers\_products

WHERE customer\_name BETWEEN 'J' AND 'T';

**=============================================================================**

1. NOT BETWEEN KOMUTU

\*\*NOT BETWEEN, BETWEEN'in tersi olarak calisir.

\*\*NOT BETWEEN kullanirsaniz **sinirlar dahil olmaz**.

​

\* product\_id'si 20 ile 40 arasinda OLMAYAN product'larin tum bilgilerini gosteren kod yaziniz.

SELECT \*

FROM customers\_products

WHERE product\_id NOT BETWEEN 20 AND 40;

​

\* Asagidaki kod yukaridaki ile aynidir ama yukaridaki daha anlasilirdir.

SELECT \*

FROM customers\_products

WHERE product\_id<20 OR product\_id>40;

​

\* product\_name'inin ilk harfi 'M' den 'S' ye kadar olan harfleri icermeyen product'larin tum bilgilerini gosteren kodu yaziniz.

SELECT \*

FROM customers\_products

WHERE product\_name NOT BETWEEN 'M' AND 'S';

**=============================================================================**

1. IS NULL / IS NOT NULL KOMUTU

\*\* IS NULL data girilmemis olan satirlari secmek icin kullanilir.

\* customers\_products tablosundan musteri ismi girilmemis datalarin tamamini gosteren kodu yaziniz.

SELECT \*

FROM customers\_products

WHERE customer\_name IS NULL;

\*\* IS NOT NULL data'si bos olmayan satirlari secmek icin kullanilir.

\* customers\_products tablosundan musteri ismi girilmemis datalarin tamamini gosteren kodu yaziniz.

SELECT \*

FROM customers\_products

WHERE customer\_name IS NULL;

\*\* IS NULL, UPDATE komutu ile de kullanilir.

UPDATE customers\_products

SET customer\_name = 'Isim girilmemis'

WHERE customer\_name IS NULL;

\* customer\_name'i bos olan datanin product\_name'ini 'Watermellon' yapiniz.

UPDATE customers\_products

SET product\_name = 'Watermellon'

WHERE customer\_name IS NULL;

​

\* customers\_products tablosundan musteri ismi bos olmayan datalarin tamamini gosteren kodu yaziniz.

SELECT \*

FROM customers\_products

WHERE customer\_name IS NOT NULL;

**=============================================================================**

1. ORDER BY KOMUTU

\*\* ORDER BY datalarin belli bir field'a gore natural order seklinde siralanmasina yarar.

\*\* ORDER BY dan sonra field ismi kullanilabildigi gibi field numarasi da kullanilabilir.

\*\* Yani;

ORDER BY product\_id;

ile

ORDER BY 1; ayni seydir.

\* customers\_products'daki tum datalari product\_name'e gore siralayan kodu yaziniz.

SELECT \*

FROM customers\_products

ORDER BY product\_name;

\* customers\_products'daki customer\_name'i 'Mark' olan datalari product\_id'e gore siralayan kodu yaziniz.

SELECT \*

FROM customers\_products

WHERE customer\_name = 'Mark'

ORDER BY product\_id;

\*\*\*AYNI\*\*\*

SELECT \*

FROM customers\_products

WHERE customer\_name = 'Mark'

ORDER BY 1;

\* customers\_products table'indaki tum datalari product\_id'lerine gore buyukten kucuge (reverse order / descending order) siralayiniz.

SELECT \*

FROM customers\_products

ORDER BY product\_id DESC;

\*\*\*AYNI\*\*\*

SELECT \*

FROM customers\_products

ORDER BY 1 DESC;

\* customers\_products table'indaki tum datalari product\_name'lerine gore buyukten kucuge (reverse order / descending order)

\* customer\_name'lerine gore de natural order'da siralayiniz.

SELECT \*

FROM customers\_products

ORDER BY product\_name DESC, customer\_name ASC;

1. ALIASES KOMUTLARI

\*\* ALIASES table'daki field isimlerini farkli gormek icin kullanilir

\*\* AS kullandiginizda database'deki tablonun sutun isimleri degismez siz sadece yeni sutun isimlerine sahip rapor alirsiniz.

\*\* DATABASE I DEGISTIRMEZ.

\* Field isimlerini Turkce'lestirin

SELECT product\_id AS urun\_kodu, customer\_name AS musteri\_ismi, product\_name AS urun\_ismi

FROM customers\_products;

​

\* Field isimlerini Turkcelestirin ve product\_id ile product\_name ayni sutunda olsun.

SELECT customer\_name AS musteri\_ismi, product\_id || product\_name AS urun\_kodu\_ismi

FROM customers\_products;

1. GROUP BY KOMUTU

\*\* GROUP BY datalari gruplandirarak gormemizi saglar

​

\* Herbir product'i alan musteri sayisini gosteren kodu yaziniz.

SELECT product\_name, count(product\_name) AS number\_of\_customers

FROM customers\_products

GROUP BY product\_name;

​

\* Herbir product\_id nin kac kere kullanildigini gosteren kodu yaziniz.

SELECT product\_id, count(product\_id) AS number\_of\_usage

FROM customers\_products

GROUP BY product\_id;

CREATE TABLE employees

(

id number(9),

name varchar2(50),

state varchar2(50),

salary number(20),

company varchar2(20)

);

​

INSERT INTO employees VALUES(123456789, 'John Walker', 'Florida', 2500, 'IBM');

INSERT INTO employees VALUES(234567890, 'Brad Pitt', 'Florida', 1500, 'APPLE');

INSERT INTO employees VALUES(345678901, 'Eddie Murphy', 'Texas', 3000, 'IBM');

INSERT INTO employees VALUES(456789012, 'Eddie Murphy', 'Virginia', 1000, 'GOOGLE');

INSERT INTO employees VALUES(567890123, 'Eddie Murphy', 'Texas', 7000, 'MICROSOFT');

INSERT INTO employees VALUES(456789012, 'Brad Pitt', 'Texas', 1500, 'GOOGLE');

INSERT INTO employees VALUES(123456710, 'Mark Stone', 'Pennsylvania', 2500, 'IBM');

\* employees tablosundan her iscinin aldigi toplam ucreti bulunuz.

SELECT name, SUM(salary) AS total\_salary

FROM employees

GROUP BY name;

\* Herbir state'de kac isci oldugunu gosteren kodu yaziniz

SELECT state, COUNT(name) AS total\_worker

FROM employees

GROUP BY state;

\* Herbir sirkette maasi 2000'in uzerinde olan kac isci oldugunu gosteren kodu yaziniz.

SELECT company, COUNT(name) AS number\_of\_employees

FROM employees

WHERE salary>2000

GROUP BY company;

​

\* Herbir sirkette verilen maximum ve minimum ucretleri gosteren kodu yaziniz.

SELECT company, MIN(salary) AS min\_salary, MAX(salary) AS max\_salary

FROM employees

GROUP BY company;

1. HAVING KOMUTU

\*\* HAVING, GROUP BY'dan sonra AGGREGATE functionlarla beraber filtreleme yapmak icin kullanilir.

\* Herbir sirketin min salary'lerini 2000'in uzerinde ise goster ve max salary'lerini goster.

SELECT company, MIN(salary) AS min\_salary, MAX(salary) AS max\_salary

FROM employees

GROUP BY company

HAVING MIN(salary)>2000;

​

\* Toplam geliri 2500'den fazla olan herbir isciyi gosteren kodu yaziniz.

SELECT name, SUM(salary) AS total\_income

FROM employees

GROUP BY name

HAVING SUM(salary)>2500;

1. UNION/UNION ALL/INTERSECT KOMUTU

\*\* UNION Operation:Iki farkli sorgulamanin sonuclarini birlestirme islemidir

\*\* UNION kullanirken her iki sorgudaki sutun data type'lari ortusmelidir.

\*\* UNION islemi yaparken:

-Her iki query'den elde edeceginiz tablonun sutun sayilari esit olmali

-Her iki query'den elde edeceginiz tablonun karsilikli sutunlarinin data type'lari ayni olmali.

-Ayni query koyarsan yukarisi ve asagisi ayni oldugu icin bir kere yazar.

-Ama illa ki tekrar li gormek istiyorum diyorsan

**UNION ALL** dersin ve **iki kere** yazdirir.

\*\* **Tekrarli** record gormek isterseniz **UNION ALL** kullanmalisiniz.

\*\* Iki query nin **ortak sonuclarini** gormek icin komut **INTERSECT** kullanilir.

\*\* **INTERSECT** Isleminde ortak eleman bulamazsa NO DATA FOUND YAZAR. HATA VERMEZ.

​

\* Salary'si 3000'den fazla olan state ve isci isimlerini gosteren kodu yaziniz.

SELECT state AS name\_and\_state, salary

FROM employees

WHERE salary>3000

UNION

SELECT name AS name\_and\_state, salary

FROM employees

WHERE salary>3000;

\* state'lerden odenen ucreti 3000 den fazla olan ve iscilerden odenen ucreti 2000 den az olanlari bir tabloda gosteren kodu yaziniz.

SELECT state AS name\_and\_state, salary

FROM employees

WHERE salary>3000

UNION

SELECT name AS name\_and\_state, salary

FROM employees

WHERE salary<2000;

​

\* Students tablosundan Ali Can'in addresini ve grade'ini aliniz, students\_information tablosundan da Ali Can'in telefon numarasi ve average score'unu aliniz ve bunlari bir tabloda birlestirerek gosteriniz**.(TEKRARSIZ OLUR SONUC)**

SELECT students\_name AS id\_name, students\_address AS address\_phone, students\_grade AS avg\_score\_grade

FROM students

WHERE students\_id = '123456789'

UNION

SELECT students\_id AS id\_name, students\_phone AS address\_phone, students\_avg\_score AS avg\_score\_grade

FROM students\_information

WHERE students\_id = '123456789';

\*\* Tekrarli record gormek isterseniz UNION ALL kullanmalisiniz.

SELECT students\_name,students\_address , students\_grade  
FROM students  
Where students\_id =‘123456789’

UNION ALL

SELECT students\_name,students\_address , students\_grade  
FROM students  
Where students\_id =‘123456789’;

\*\* Iki query nin **ortak sonuclarini** gormek icin komut **INTERSECT** kullanilir.(INTERVIEW)

\* Grade’i 94'ten kucuk olanlarla, average score’u 80'den buyuk olanlarin kesisimi olanlari gosteriniz.

SELECT students\_grade AS grade\_avg\_score  
FROM students  
WHERE students\_grade<94

INTERSECT

SELECT students\_avg\_score  
FROM students\_information  
WHERE students\_avg\_score>80;

\* salary si 3000 den cok 2000 den az olan iscilerin tum bilgilerini ***tekrar olmadan*** gosteren kodu yaziniz.

SELECT \*

FROM employees

WHERE salary>3000

**UNION**

SELECT \*

FROM employees

WHERE salary<2000;

\* salary si 3000 den cok 2000 den az olan iscilerin ortak isimleri olusturan kod

SELECT name

FROM employees

WHERE salary>3000

**INTERSECT**

SELECT name

FROM employees

WHERE salary<2000;

\* IBM, APPLE, ve GOOGLE da calisip 3000'den fazla salary’si olan iscileri gosteren kodu yaziniz

SELECT name

FROM employees

WHERE company IN ('IBM','APPLE','GOOGLE')

**INTERSECT**

SELECT name

FROM employees

WHERE salary >3000;

1. MINUS KOMUTU

\* Google da calismayip 2000 den az alanlarin isimlerini yazan kod

\*\* ILK SORGUNUN ICINDEKI IKINCI SORGUYA AIT OLAN SORGU SONUCLARINI CIKARIR.

\*\*MINUS islemi ilk query'nin sonuclari arasindan ikinci query'de olanlari silmek icin kullanilir.

SELECT name,company

FROM employees

WHERE salary<2000

MINUS

SELECT name,company

FROM employees

WHERE company='GOOGLE'

\*\* ISMI eddie morphy olup texas ta yasamayanlari isimlerini ve statelerini gosteren kod

SELECT name, state

FROM employees

WHERE name='Eddie Murphy'

MINUS

SELECT name,state

FROM employees

WHERE state='Texas';

1. JOINS KOMUTU

\*\* Tablolari birlestirmeye yarar..

\*\* Sorgulari degil TABLOLARI BIRLESTIRIR, unionlarla karistirmamak lazim.(ONEMLI)

TURLERI:

--INNER JOIN:

--LEFT JOIN

--RIGHT JOIN

--FULL JOIN

--SELF JOIN

\*\* 1. INNER JOIN:

\* companies table daki company\_id leri ile orders table daki company\_id leri ayni olan orderlarin company\_name lerini, order\_id lerini ve order\_date lerini gosteren bir tablo olusturan kod…

SELECT companies.company\_name, orders.order\_id, orders.order\_date

FROM companies INNER JOIN orders

ON companies.company\_id = orders.company\_id;

1)SELECT’den sonra tabloda gormek istediginiz sutun isimlerini yazarken tablo ismi + nokta + sutun ismi seklinde yazin.

2)Iki tablo ile calistigimizdan FROM’dan sonra tablo ismi yazarken

birinci tablo ismi + INNER JOIN + ikinci tablo ismi yazmaliyiz

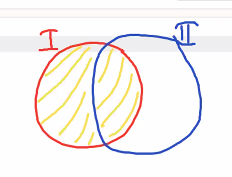
3)Join’i hangi kurala gore yapacaginizi belirtmelisiniz bunun icin de ON + kuraliniz yazmalisiniz

4) Sadece JOIN yazilabilir. Onu INNERJOIN olarak Kabul eder.

\*\* 2. LEFT JOIN

\* Ilk tablodaki tum datalar gosterilir, bunlara ilave olarak ikinci tablodan gelen ek data varsa, bu ek datalar ortak datalar icin gosterilir ancak ortak olmayan datalar icin o kisimlar bos birakilir..

LEFT JOIN solundaki tabloyu esas alir.



SELECT companies.company\_name, orders.order\_id, orders.order\_date

FROM companies LEFT JOIN orders

ON companies.company\_id = orders.company\_id;

\*\* 3. RIGHT JOIN

\* ikinci tablodaki tum datalar gosterilir,

\* RIGHT JOIN in kelimesinin sagindaki tabloyu esas alir.

SELECT companies.company\_name, orders.order\_id, orders.order\_date

FROM companies RIGHT JOIN orders

ON companies.company\_id = orders.company\_id;

\*\* 4. FULL JOIN

\* Iki tabloda da var olan tum datalar gosterilir.

\* genellikle bazi datalar bos kalabilir.

SELECT companies.company\_name, orders.order\_id, orders.order\_date

FROM companies FULL JOIN orders

ON companies.company\_id = orders.company\_id;

\*\* 5. SELF JOIN

1. WILDCARDS

**\*\* YUZDE ISARETI : %**

\*\* Sifir veya sifirdan fazla karakter gosterir.

\*\* LIKE ILE BERABER KULLANILIR.

\* Ismi 'J' harfi ile baslayan musterilerin tum bilgilerini gosteren bit tablo olusturmak icin kod yaziniz.

SELECT \*

FROM customers

WHERE customer\_name **LIKE** 'J%';

\*\*ILK HARFI J, YUZDE SONDA

​

\* Ismi 'e' ile biten musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name **LIKE** '%e';

\*\*SON HARFI e, YUZDE BASTA

\* Isminin icinde 'n' olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name **LIKE** '%n%';

**\*\* UNDERSCORE \_**

\*\*Sadece 1 karakteri gosterir

​

\* Ismi 4 harfli olup son 3 harfi 'ohn' olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name LIKE '\_ohn';

​

\* Ismi 4 harfli olup son 2 harfi 'ne' olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name LIKE '\_\_ne';

\* Ucuncu harfi 'n' olan en az 5 harfli isme sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name LIKE '\_\_n\_\_%';

\* 'B' ile baslayip 3. harfi 'a' olan isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name LIKE 'B\_a%';

**\*\* KOSELI PARANTEZ [ ]**

\* Ilk harfi 'C' olan son harfi 'n' olan ikinci harfi 'a' veya 'i' olan 3 harfli isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE REGEXP\_LIKE(customer\_name, 'C[ai]n');

---- [ ] bunun icine yazdigin harflerin hepsinin opsiyonunu yazdirir.

---- can ve cin yazdirir.

\* Ilk harfi 'C' olan son harfi 'n' olan ikinci harfi **'a'dan 'k' ya kadar** tum harfler olan 3 harfli isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE REGEXP\_LIKE(customer\_name, 'C[a-k]n');

\* Icinde **'a' veya 'n'** olan tum isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE REGEXP\_LIKE(customer\_name, '[an](\*)');

\* 'J' veya 'M' ile **baslayan** tum isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE REGEXP\_LIKE(customer\_name, '^[JM](\*)');

---Jack John Manse

\* Ilk harfi 'J' olmayan tum isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name NOT LIKE 'J%';

\* 'a' icermeyen tum isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

SELECT customer\_name, income

FROM customers

WHERE customer\_name NOT LIKE '%a%';

​

\* Ikinci harfi 'a' olmayan isimlere sahip olan musterilerin musteri adi ve gelirlerini gosteren tabloyu olusturan kodu yaziniz.

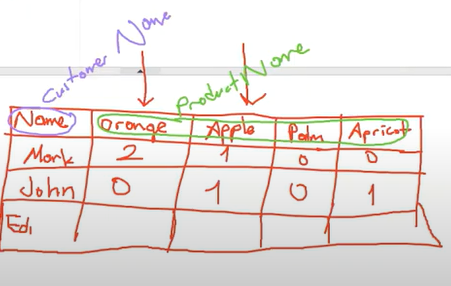
SELECT customer\_name, income

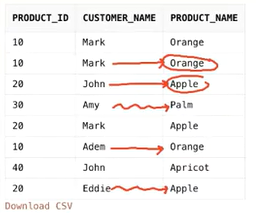
FROM customers

WHERE customer\_name NOT LIKE '\_a%';

1. PIVOT

\*\* PIVOT KULLANARAK SATIRLARI SUTUNA CEVIREBILIRIZ.





​

SELECT \* FROM (SELECT customer\_name, product\_name FROM customers\_products)

PIVOT

(COUNT(product\_name) FOR product\_name IN ('Orange', 'Apple', 'Palm','Apricot'));

​

SELECT \* FROM (SELECT customer\_name, product\_name FROM customers\_products)

PIVOT

(COUNT(customer\_name) FOR customer\_name IN ('Mark', 'Amy', 'Adem','John'));

​

SELECT \* FROM (SELECT product\_id, product\_name FROM customers\_products)

PIVOT

(COUNT(product\_id) FOR product\_id IN (10, 20, 30, 40));

1. DISTINCT

\*\*TEKRARLI OLAN URUNLERI BIR KERE GORMEK ICIN KULLANILIR

--product\_name'leri tekrarsiz gosteren kodu yaziniz

SELECT DISTINCT product\_name

FROM customers\_products;

--Kac farkli meyve var?

SELECT COUNT(DISTINCT product\_name) AS meyve\_cesit\_sayisi

FROM customers\_products;

1. ALTER TABLE

CREATE TABLE **employees**

(

id number(9),

name varchar2(50),

state varchar2(50),

salary number(20),

company varchar2(20)

);

--1)Tabloya yeni sutun ekleyebiliriz

ALTER TABLE employees

ADD gender varchar2(20);

\*\* BU KODDAN SONRA TABLOYA SUTUN EKLER AMA VERI EKLEMEDIGIMIZ ICIN BOS OLUR.

\*\* BU EKLEDIGIMIZ SUTUN GECICI BIR DEGISIKLIK DEGILDIR. TABLONUN KENDISINI DEGISTIRIR. GENETIGINI DEGISTIRIR.

​

--Tabloya ulke sutunu ekleyin

ALTER TABLE employees

ADD country varchar2(50) DEFAULT 'The USA';

​

--Tabloye ayni anda 1'den fazla sutun ekleyiniz.

ALTER TABLE employees

ADD ( number\_of\_kid number(2),

marital\_status varchar2(30) DEFAULT 'single'

);

--2)Tablodan sutun silme

ALTER TABLE employees

DROP COLUMN country;

​

--3)Column ismini degistirme

ALTER TABLE employees

RENAME COLUMN gender to gender\_male\_or\_female;

​

--4)Tablonun ismini degistirme

ALTER TABLE employees

RENAME TO workers;

​

--5)Sutunlarin yapisini degistirmek

ALTER TABLE workers

MODIFY id number(9) NOT NULL;

​

--1'den fazla sutunun yapisini ayni anda degistirmek

ALTER TABLE workers

MODIFY (state char(45) NOT NULL,

company char(30)

);

1. LIMIT ve OFFSETT

SELECT name,salary

FROM workers

ORDER BY salary DESC

**LIMIT 3;**

\*\* ILK 3 ADETINI ALIR.

SELECT name,salary

FROM workers

ORDER BY salary DESC

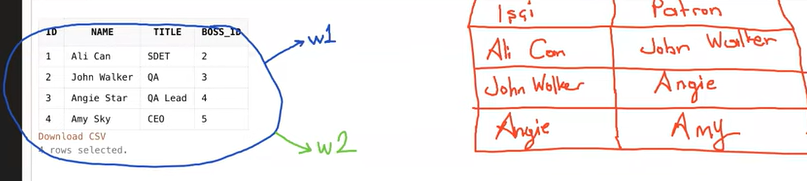
**LIMIT 3 OFFSET 2;**

\*\* ILK 3 ADETINI ALIR.

\*\* AMA ILK IKIYI ALMAZ, ucuncuden baslar.

1. SELF JOIN

Her iscinin patronunu gosteren tabloyu olusturunuz…



SELECT w1.name AS worker\_name, w2.name AS boss\_name

FROM workers01 w1 INNER JOIN workers01 w2

ON w1.boss\_id=w2.id;

Yukardaki boyali yerler kopya olusturmak icin yapildi. SQL de kopya olusturmak icin bu sekilde yapiliyor.

**\*\* SQL DE YUKARDAN ASAGI CALISMAZ.**

**\*\* ONCE FROM CALISIR.**

**\*\*SONRA WHERE CALISIR.**

**\*\* EN SON SELECT CALISIR.**

**\*\* SEC TABLODAN SU SARTLAR ALTINDA…**

**SELECT INITCAP(name), UPPER(state), LOWER(company)**

**FROM workers,**

**INITCAP🡺 ilk harfleri buyutur…**

**UPPER🡺 hepsini buyuk harf yapar…**

**LOWER🡺 hepsini kucuk harf yapar…**

**INTERVIEW SORULARI**

--Her iki tablodaki ortak id ve isimleri gosteren kodu yaziniz.

SELECT id, name

FROM students

​

INTERSECT

​

SELECT id, name

FROM workers01;

​

--students tablosunda kac farkli state'den student var.

SELECT COUNT(DISTINCT state) AS num\_of\_states

FROM students;

​

--id'si cift sayi olan ogrencilerin tum bilgilerini gosteren kodu yaziniz.

SELECT \*

FROM students

WHERE MOD(id,2)=0;

​

--id'si tek sayi olan ogrencilerin tum bilgilerini gosteren kodu yaziniz.

SELECT \*

FROM students

WHERE MOD(id,2)=1;

​

--Table'da kac tane record oldugunu gosteren kodu yaziniz

SELECT COUNT(\*) AS num\_of\_records

FROM students;

​

--En yuksek ucreti gosteren kodu yaziniz.

SELECT MAX(salary) AS Max\_salary

FROM workers01;

​

--En dusuk ucreti gosteren kodu yaziniz.

SELECT MIN(salary) AS Min\_salary

FROM workers01;

​

--En yuksek ucreti alan iscinin tum bilgilerini gosteren kodu yaziniz. (SUBQUERY)

SELECT \*

FROM workers01

WHERE salary = (SELECT MAX(salary) AS Max\_salary

FROM workers01

);

--En dusuk ucreti alan iscinin tum bilgilerini gosteren kodu yaziniz. (SUBQUERY)

SELECT \*

FROM workers01

WHERE salary = (SELECT MIN(salary) AS Min\_salary

FROM workers01

);

--Ikinci en yuksek maasi gosteren kodu yaziniz.

SELECT MAX(salary)

FROM workers01

WHERE salary< (SELECT MAX(salary)

FROM workers01

);

​

--Ikinci en dusuk maasi gosteren kodu yaziniz.

SELECT MIN(salary)

FROM workers01

WHERE salary > (SELECT MIN(salary)

FROM workers01

);

\*\* SALARY SI EN YUKSEK OLAN ISCI DISINDAKI ISCILERIN TUM BILGILERINI

\*\* SALARY E GORE BUYUKTEN KUCUGE DIZEREK GOSTEREN KODU YAZINIZ.

SELECT \*

FROM employees

WHERE salary ! = (SELECT MAX(salary)

FROM employees)

ORDER BY salary DESC;

\*\* IKINCI EN YUKSEK MAASI ALANIN TUM BILGILERINI GOSTEREN KODU YAZINIZ

SELECT \*

FROM (SELECT \*

FROM employees

WHERE salary ! = (SELECT MAX(salary)

FROM employees)

ORDER BY salary DESC)

WHERE ROWNUM=1;

**PRACTISE SITESI…**

[**https://sqlbolt.com/lesson/select\_queries\_introduction**](https://sqlbolt.com/lesson/select_queries_introduction)

[**https://www.sqlteaching.com/**](https://www.sqlteaching.com/)