Beyen havva

030119112

# veritabanı oluşturma

CREATE DATABASE `havva.beyen\_Ders4`;

# Oluşturulan veritabanlarını göstermek

SHOW CREATE DATABASE `havva.beyen\_Ders4`;

SHOW DATABASES;

# Veri tabanı kullanma

USE `havva.beyen\_Ders4`;

# Koşullu gösterim

SHOW DATABASES LIKE "%Ders%";

SHOW DATABASES WHERE expression;

SHOW DATABASES LIKE "%schema";

# Bir yerden veri seçip gösterim

SELECT schema\_name FROM information\_schema.schemata;

SELECT schema\_name FROM information\_schema.schemata WHERE schema\_name LIKE "em%";

# Veritabanı silmek

DROP DATABASE [IF EXISTS] `havva.beyen\_Ders4`;

DROP SCHEMA [IF EXISTS] `havva.beyen\_Ders4`;

# Veritabanı kopyası oluşturmak

CREATE DATABASE `havva.beyen\_Ders5`;

CREATE DATABASE `havva.beyen\_Ders5\_copy`;

SHOW DATABASES;

# Tablo oluşturmak

CREATE TABLE `havva.beyen\_odev1`.`std\_tbl`(

`std\_id` INT NOT NULL AUTO\_INCREMENT,

`std\_firstname` VARCHAR(35) NOT NULL,

`std\_surname` VARCHAR(45) NOT NULL,

PRIMARY KEY (`std\_id`)

);

# Tablo göstermek

SHOW TABLES;

# Veri tabanı kullanarak tabloyı değiştirmek

USE `havva.beyen\_odev1`;

ALTER TABLE `std\_tbl`

ADD `std\_calss` int NOT null

AFTER `std\_surname`;

# Tabloya veriler eklemek

INSERT INTO `std\_tbl` (`std\_firstname`, `std\_surname`,`std\_calss`)

VALUES ('Ajeet', 'Maurya','1'), ('Deepika', 'Chopra','3'), ('Vimal', 'Jaiswal','4');

# Bütün tabloyu göster

SELECT \*FROM `std\_tbl`;

# Veri tabanı kullanarak tablo oluşturup benzersiz anahtar atamak

USE `havva.beyen\_odev1`;

CREATE TABLE student(

Stud\_ID int,

Roll\_No int,

Name varchar(45) NOT NULL,

Email varchar(45),

Age int,

City varchar(25),

CONSTRAINT uc\_rollno\_email UNIQUE(Roll\_No, Email)

);

# Bir tablodaki bir indexte olan veriyi gösterimi

SHOW INDEX FROM student;

ALTER TABLE student DROP INDEX uc\_rollno\_email;

SHOW INDEX FROM student;

# Bir tabloya benzersiz anahtar eklemek

ALTER TABLE table\_name

ADD CONSTRAINT constraint\_name UNIQUE(column\_list);

# Tablo oluşturmak sonra onu görmek (describe)

CREATE TABLE Login(

login\_id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(40),

password VARCHAR(55),

email VARCHAR(55)

);

DESCRIBE Login;

# Bir tabloya bir veriden (önce/sonra) ekleme yapmak

ALTER TABLE Login

ADD new\_column int(8) Null

AFTER username;

# Bir verinin veri türünü değiştirmek (modify)

ALTER TABLE Login

MODIFY new\_column varchar(50) NULL;

# Bir verinin veri türünü değiştirmek / eklemek (modify)

ALTER TABLE Login

MODIFY new\_column varchar(50) NULL;

FIRST

# Bir tablodan sutün silmek

ALTER TABLE employee\_table

DROP COLUMN new\_column;

# Bir sutünde değişim yapmak

ALTER TABLE std\_tbl

CHANGE COLUMN std\_class

varchar(250) NOT NULL;

# bütün tabloları göster

SHOW FULL TABLES;

# Bir veritabanı içindeki tabloları göstermek /koşullu olarak

SHOW TABLES IN `havva.beyen\_odev1`;

SHOW TABLES FROM `havva.beyen\_odev1`;

SHOW TABLES LIKE 'st%';

SHOW TABLES FROM `havva.beyen\_odev1` LIKE 'st%';

# Tablo adını değiştirmek

RENAME TABLE cus\_tbl TO std\_tbl;

RENAME TABLE emp\_tbl TO employee\_table, students TO student\_table;

# Tabloya veriler eklemek

USE `havva.beyen\_odev1`;

INSERT INTO Login( username, password, email)

VALUES('Deniz', 15343434532, 'deniz@izu.edu.tr'),( 'Mehmet', 35435479495, 'mehmet@izu.edu.tr');

# Tabloya veriler eklemek

INSERT INTO Login( username, password, email)

VALUES ('Ahmet', 15343434532, 'ahmet@izu.edu.tr');

# Tablo oluşturup benzersiz anahtar eklemek

CREATE TABLE Student\_Unique\_Key (

Stud\_ID int NOT NULL UNIQUE,

Name varchar(45),

Email varchar(45),

Age int,

City varchar(25)

);

# Içinde benzersiz anahtarı olan tabloya veri atamak

USE `havva.beyen\_odev1`;

INSERT INTO Student\_Unique\_Key (Stud\_ID, Name, Email, Age, City)

VALUES (1, 'Ahmet', 'ahmet@izu.edu.tr', 22, 'Ankara'),

(2, 'Fatma', 'fatma@izu.edu.tr', 24, 'Antalya'),

(3, 'Mehmet', 'mehmet@izu.edu.tr', 23, 'Erzurum');

# Tablo oluşturup benzersiz anahtar eklemek

CREATE TABLE Student\_Unique\_Key (

Stud\_ID int NOT NULL UNIQUE,

Name varchar(45),

Email varchar(45),

Age int,

City varchar(25)

);

USE `havva.beyen\_odev1`;

INSERT INTO Student\_Unique\_Key (Stud\_ID, Name, Email, Age, City)

VALUES (1, 'Ahmet', 'ahmet@izu.edu.tr', 22, 'Ankara'),

(2, 'Fatma', 'fatma@izu.edu.tr', 24, 'Antalya'),

(3, 'Mehmet', 'mehmet@izu.edu.tr', 23, 'Erzurum');

# Tablo oluşturup primary anahtar eklemek

CREATE TABLE Student\_Primary\_Key (

Student\_ID int,

Roll\_No int,

Name varchar(45) NOT NULL,

Age int,

City varchar(25),

PRIMARY KEY(Student\_ID, Roll\_No)

);

# Bir tabloya primary anahtarı eklemek

ALTER Student\_Primary\_Key

ADD PRIMARY KEY(Age);

# Bir tabloyu oluşturup primary anahtarı eklemek

CREATE TABLE Person (

Person\_ID int NOT NULL,

Name varchar(45),

Age int,

City varchar(25)

);

# Bir tabloya primary anahtarı eklemek

ALTER TABLE Person

ADD PRIMARY KEY(Person\_ID);

# Bir tablodan primary anahtarı silmek

ALTER TABLE Person DROP PRIMARY KEY;

# Bir tabloya PK ‘ye otomatik olarak veri atamak

USE `havva.beyen\_odev1`;

CREATE TABLE customer (

ID INT NOT NULL AUTO\_INCREMENT,

Name varchar(50) NOT NULL,

City varchar(50) NOT NULL,

PRIMARY KEY (ID)

);

# FK oluşturmak – cascade ile (2 tablodan siler )

CREATE TABLE contact (

ID INT,

Customer\_Id INT,

Customer\_Info varchar(50) NOT NULL,

Type varchar(50) NOT NULL,

INDEX par\_ind (Customer\_Id),

CONSTRAINT fk\_customer FOREIGN KEY (Customer\_Id)

REFERENCES customer(ID)

ON DELETE CASCADE

ON UPDATE CASCADE

);

# 2 tabloyu foreign Key ile bağlamak

USE `havva.beyen\_odev1ForeignKey`;

CREATE TABLE customer (

ID INT NOT NULL AUTO\_INCREMENT,

Name varchar(50) NOT NULL,

City varchar(50) NOT NULL,

PRIMARY KEY (ID)

);

CREATE TABLE contact (

ID INT,

Customer\_Id INT,

Customer\_Info varchar(50) NOT NULL,

Type varchar(50) NOT NULL,

INDEX par\_ind (Customer\_Id),

CONSTRAINT fk\_customer FOREIGN KEY (Customer\_Id)

REFERENCES customer(ID)

ON DELETE CASCADE

ON UPDATE CASCADE

);

/\*

CREATE TABLE `customer`(

`Id` int PRIMARY KEY NOT NULL,

    `Name` varchar(45) NOT NULL,

    `Product` varchar(45) DEFAULT NULL,

    `Country` varchar(25) DEFAULT NULL,

    `Year` int NOT NULL

);

INSERT INTO `customer` ( Id, Name, Product, Country, Year)

 VALUES (1, 'Stephen', 'Computer', 'USA', 2015),

(2, 'Joseph', 'Laptop', 'India', 2016),

(3, 'John', 'TV', 'USA', 2016),

(4, 'Donald', 'Laptop', 'England', 2015),

(5, 'Joseph', 'Mobile', 'India', 2015),

(6, 'Peter', 'Mouse', 'England', 2016);

\*/

# (Tabloyu Write olarak kilitler)

LOCK TABLE students2 WRITE;

# Tablo adını değiştirmek

RENAME TABLE students2 TO students3;

# (Tabloyu sadece Read olarak kilitler)

LOCK TABLE students3 READ;

# Tablo adını değiştirmek

RENAME TABLE `havva.beyen \_Ders2 `. employee TO ` havva.beyen \_Ders2`. employee\_table2;

# (Temporary tablo oluşturur geçici)

CREATE TEMPORARY TABLE Students (

`name` VARCHAR(40) NOT NULL,

`total\_marks` DECIMAL(12,2) NOT NULL DEFAULT 0.00,

`total\_subjects` INT UNSIGNED NOT NULL DEFAULT 0

);

# Temporary tablo oluşturur ve içine değer ekler sonra bütün verileri gösterir. Ondan sonra Tablonun ismini değiştirir

CREATE TEMPORARY TABLE Students ( `name` VARCHAR(40) NOT NULL, `total\_marks` DECIMAL(12,2) NOT NULL DEFAULT 0.00, `total\_subjects` INT UNSIGNED NOT NULL DEFAULT 0 );

INSERT INTO Students(`name`, `total\_marks`, `total\_subjects`) VALUES ('Joseph', 150.75, 2), ('Peter', 180.75, 2);

SELECT \* FROM Students;

ALTER TABLE Students RENAME TO student\_info;

# 2 tabloyu foreign Key ile bağlamak

USE `havva.beyen\_odev1ForeignKey`;

**CREATE** **TABLE** contact (

ID **INT**,

Customer\_Id **INT**,

Customer\_Info **varchar**(50) NOT NULL,

Type **varchar**(50) NOT NULL,

**INDEX** par\_ind (Customer\_Id),

**CONSTRAINT** fk\_customer **FOREIGN** **KEY** (Customer\_Id)

**REFERENCES** customer(ID)

**ON** **DELETE** **CASCADE**

**ON** **UPDATE** **CASCADE**

);

CREATE TABLE `customer` (    ​

    `Id` int ,     ​

    `Name` varchar(45) NOT NULL,     ​

    `Product` varchar(45) DEFAULT NULL,     ​

    `Country` varchar(25) DEFAULT NULL,     ​

    `Year` int NOT NULL​

); ​

INSERT INTO `customer` ( Id, Name, Product, Country, Year)     ​

VALUES (1, 'Stephen', 'Computer', 'USA', 2015),     ​

(2, 'Joseph', 'Laptop', 'India', 2016),     ​

(3, 'John', 'TV', 'USA', 2016),    ​

(4, 'Donald', 'Laptop', 'England', 2015),    ​

(5, 'Joseph', 'Mobile', 'India', 2015),    ​

(6, 'Peter', 'Mouse', 'England', 2016);

SELECT \* FROM customer;

**INSERT** **INTO** contact (Customer\_Id, Customer\_Info, Type)

**VALUES**

(1, 'mehmet@izu.edu.tr', 'email'),

(1, '121-121-121', 'work' ),

(1, '123-123-123', 'home'),

(2, 'ayse@izu.edu.tr', 'email'),

(2, 'ayse@izu.edu.tr', 'email'),

(2, '212-212-212', 'work'),

(3, 'can@izu.edu.tr', 'email'),

(3, '313-313-313', 'home');

# Tabloya güncelleme yapmak

**UPDATE** customer **SET** id=3 **WHERE** **Name** = 'Ayse';

# Tabloları silmek

Drop Table contact;

Drop Table customer;

Drop Table Persons;

# 2 tabloyu Foreign Key ile bağlamak ( set null ile - Sildikten sonra Yerini boş bırak )

/\*

CREATE TABLE Persons ( ID INT, Name varchar(50) NOT NULL, City varchar(50) NOT NULL, PRIMARY KEY (ID) );

CREATE TABLE Contacts ( ID INT, Person\_Id INT, Info varchar(50) NOT NULL, Type varchar(50) NOT NULL, INDEX par\_ind (Person\_Id), CONSTRAINT fk\_person FOREIGN KEY (Person\_Id) REFERENCES Persons(ID) ON DELETE SET NULL ON UPDATE SET NULL );

\*/

**CREATE** **TABLE** Persons (

 ID **INT**,

**Name** **varchar**(50) NOT NULL,

  City **varchar**(50) NOT NULL,

**PRIMARY** **KEY** (ID)

);

**CREATE** **TABLE** Contacts (

ID **INT**,

Person\_Id **INT**,

Info **varchar**(50) NOT NULL,

Type **varchar**(50) NOT NULL,

**INDEX** par\_ind (Person\_Id),

**CONSTRAINT** fk\_person **FOREIGN** **KEY** (Person\_Id)

**REFERENCES** Persons(ID)

**ON** **DELETE** **SET** NULL

**ON** **UPDATE** **SET** NULL

);

**INSERT** **INTO** Persons(**Name**, City) **VALUES**

('Mehmet', 'Ankara'),

('Ayse', 'Bodrum'),

('Ahmet', 'Erzurum');

**INSERT** **INTO** Contacts (Person\_Id, Info, Type) **VALUES**

(1, 'mehmet@izu.edu.tr', 'email'),

(1, '121-121-121', 'work' ),

(2, 'ayse@izu.edu.tr', 'email'),

(2, '212-212-212', 'work'),

(3, 'ahmet@izu.edu.tr', 'email'),

(3, '313-313-313', 'home');

**UPDATE** Persons **SET** ID=103 **WHERE** ID=3;

# Composite key create ile

**CREATE** **TABLE** Product (  ​

    Prod\_ID **int** NOT NULL,   ​

**Name** **varchar**(45),   ​

    Manufacturer **varchar**(45),  ​

**PRIMARY** **KEY**(**Name**, Manufacturer)  ​

);

 ​**INSERT** **INTO** Product (Prod\_ID, **Name**, Manufacturer)  ​

**VALUES** (101, 'Soap', 'Hamam');  ​

  ​

**INSERT** **INTO** Product (Prod\_ID, **Name**, Manufacturer)  ​

**VALUES** (101, 'Soap', 'LUX');  ​

# Composite alter ile

CREATE TABLE Student(  ​

  stud\_id int NOT NULL,  ​

  stud\_code varchar(15),  ​

  stud\_name varchar(35),  ​

  subject varchar(25),  ​

  marks int  ​

);**​**

ALTER TABLE Student ​

ADD PRIMARY KEY(stud\_id, subject);**​**

# 5.hafta

# PK

CREATE DATABASE `havva.beyen\_db1`;

USE `havva.beyen\_db1`;

CREATE TABLE customers(id int(10),

name varchar(50),

city varchar(50),

PRIMARY KEY(id)

);

# Tablo düzenleme

USE `havva.beyen\_db1`;

ALTER TABLE customers

ADD age VARCHAR(50);

# Tabloya ekleme

USE `havva.beyen\_db1`;

INSERT INTO customers VALUES(101,'rahul','delhi',20);

Select \* from customers;

# Tablonun tümünü göster

USE `havva.beyen\_db1`;

SELECT \* FROM customers

# Tabloyu güncelle

USE `havva.beyen\_db1`;

UPDATE customers SET name='bob', city='london' WHERE id=101;

Select \* from customers;

//

USE `havva.beyen\_db1`;

Select \* from customers;

UPDATE customers SET name='bob', city='london' WHERE id=101;

Select \* from customers;

//

# Tablodan silme

USE `havva.beyen\_db1`;

DELETE FROM customers WHERE id=20;

Select \* from customers;

# Tabloya ekle

USE `havva.beyen\_db1`;

INSERT INTO customers VALUES(101,'rahul','delhi',20);

Select \* from customers;

# Tabloya ekle sonra sil

USE `havva.beyen\_db1`;

INSERT INTO customers VALUES(102,'rahul','delhi',21);

Select \* from customers;

TRUNCATE TABLE customers;

Select \* from customers;

# Tablo oluştur ( not null)

USE `havva.beyen\_db1`;

CREATE TABLE Student(Id INTEGER, LastName TEXT NOT NULL, FirstName TEXT NOT NULL, City VARCHAR(35));

USE `havva.beyen\_db1`;

Select \* from Student;

INSERT INTO Student VALUES(1, 'Hanks', 'Peter', 'New York');

INSERT INTO Student VALUES(2, NULL, 'Amanda', 'Florida');

Select \* from Student;

USE `havva.beyen\_db1`;

INSERT INTO Student VALUES( 3, '', 'Joe', 'California' );

Select \* from Student;

# Tablo oluştur (unique)

USE `havva.beyen\_db1`;

CREATE TABLE ShirtBrands(Id INTEGER, BrandName VARCHAR(40) UNIQUE, Size VARCHAR(30));

Select \* from ShirtBrands;

USE `havva.beyen\_db1`;

INSERT INTO ShirtBrands(Id, BrandName, Size)

VALUES(1, 'Pantaloons', 38), (2, 'Cantabil', 40);

INSERT INTO ShirtBrands(Id, BrandName, Size)

VALUES(1, 'Raymond', 38), (2, 'Cantabil', 40);

Select \* from ShirtBrands;

# Şart ifadesi (check)

USE `havva.beyen\_db1`;

CREATE TABLE Persons (

ID int NOT NULL,

Name varchar(45) NOT NULL,

Age int CHECK (Age>=18) ); // kontrol - şart ifadesi

Select \* from Persons;

USE `havva.beyen\_db1`;

INSERT INTO Persons(Id, Name, Age)

VALUES (1,'Robert', 28), (2, 'Joseph', 35), (3, 'Peter', 40);

INSERT INTO Persons(Id, Name, Age) VALUES (1,'Robert', 19);

# Tablo sil

USE `havva.beyen\_db1`;

DROP TABLE Persons

# Tablo oluştur ( default)

USE `havva.beyen\_db1`;

CREATE TABLE Persons (

ID int NOT NULL,

Name varchar(45) NOT NULL,

Age int,

City varchar(25) DEFAULT 'New York'

);

Select \* from Persons;

USE `havva.beyen\_db1`;

INSERT INTO Persons(Id, Name, Age, City)

VALUES (1,'Robert', 15, 'Florida'),

(2, 'Joseph', 35, 'California'),

(3, 'Peter', 40, 'Alaska');

Select \* from Persons;

USE `havva.beyen\_db1`;

INSERT INTO Persons(Id, Name, Age)

VALUES (1,'Brayan', 15);

Select \* from Persons;

USE `havva.beyen\_db1`;

DROP TABLE Persons

USE `havva.beyen\_db1`;

CREATE TABLE Persons (

ID int NOT NULL PRIMARY KEY,

Name varchar(45) NOT NULL,

Age int,

City varchar(25));

USE `havva.beyen\_db1`;

INSERT INTO Persons(Id, Name, Age, City)

VALUES (1,'Robert', 15, 'Florida') ,

(2, 'Joseph', 35, 'California'),

(3, 'Peter', 40, 'Alaska');

INSERT INTO Persons(Id, Name, Age, City)

VALUES (1,'Stephen', 15, 'Florida');

USE `havva.beyen\_db1`;

INSERT INTO Persons(Id, Name, Age, City)

VALUES (4,'Stephen', 15, 'Florida');

# Tablo oluştur ( enum örneği )

USE `havva.beyen\_db1`;

CREATE TABLE Shirts (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(35),

size ENUM('small', 'medium', 'large', 'x-large')

);

USE `havva.beyen\_db1`;

INSERT INTO Shirts(id, name, size)

VALUES (1,'t-shirt', 'medium'),

(2, 'casual-shirt', 'small'),

(3, 'formal-shirt', 'large');

Select \* from Shirts;

USE `havva.beyen\_db1`;

CREATE TABLE Shirts2 (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(35),

size ENUM('small', 'medium', 'large', 'x-large')

);

Select \* from Shirts2;

USE `havva.beyen\_db1`;

INSERT INTO Shirts2(id, name, size)

VALUES (1,'t-shirt', 'medium'),

(2, 'casual-shirt', 'small'),

(3, 'formal-shirt', 'large');

Select \* from Shirts;

# Tablo oluştur ( index ekleyerek )

USE `havva.beyen\_db1`;

CREATE INDEX idx\_name ON Shirts2(name);

Select \* from Shirts2;

### // foregin key constraintleri

MySQL contains five different referential options, which are given below:

**CASCADE:**

**2 tablodan siler** It is used when we delete or update any row from the parent table, the values of the matching rows in the child table will be deleted or updated automatically.​

**SET NULL:**

**Sildikten sonra Yerini boş bırak** It is used when we delete or update any row from the parent table, the values of the foreign key columns in the child table are set to NULL.​

**RESTRICT**

**Sadece bir tabloda silmeyi Kabul eder** It is used when we delete or update any row from the parent table that has a matching row in the reference(child) table, MySQL does not allow to delete or update rows in the parent table.​

**NO ACTION:**

**Silerse diğer tablo etkilenmez**

It is similar to RESTRICT. But it has one difference that it checks referential integrity after trying to modify the table.​

**SET DEFAULT:**

**Yerine değer koy** The MySQL parser recognizes this action. However, the InnoDB and NDB tables both rejected this action.​

// date time için fonksiyonlar bir haftada tek başına işlenecek

# Update örnekleri

USE `havva.beyen\_db1`;

CREATE TABLE trainer (course\_name VARCHAR(35) PRIMARY KEY NOT NULL, trainer VARCHAR(45) NOT NULL, email VARCHAR(150) NOT NULL);

INSERT INTO trainer (course\_name, trainer, email)

VALUES ('Java', 'Mike', 'mike@izu.edu.tr'),

('Python', 'James', 'james@izu.edu.tr'),

('Android', 'Robin', 'robin@izu.edu.tr'),

('Hadoop', 'Stephen', 'stephen@izu.edu.tr'),

('Testing', 'Micheal', 'micheal@izu.edu.tr');

Select \* from trainer;

USE `havva.beyen\_db1`;

Select \* from trainer;

UPDATE trainer SET email='nlnjkn@std.izu.edu.tr' WHERE course\_name = 'Java';

Select \* from trainer;

USE `havva.beyen\_db1`;

Select \* from trainer;

UPDATE trainer

SET trainer = 'Robin', email = 'james@std.izu.edu.tr'

WHERE Course\_name = 'Python';

Select \* from trainer;

# Replace örneği

USE `havva.beyen\_db1`;

Select \* from trainer;

UPDATE trainer

SET email = REPLACE(email,'@std.izu.edu.tr','@itu.edu.tr')

WHERE course\_name = 'Python';

Select \* from trainer;

# Delete örnekleri

USE `havva.beyen\_db1`;

CREATE TABLE Employees (emp\_id INT PRIMARY KEY NOT NULL AUTO\_INCREMENT, name VARCHAR(35) NOT NULL, birthdate DATE NOT NULL, gender VARCHAR(2) NOT NULL, hire\_date DATE NOT NULL);

INSERT INTO `Employees`(`emp\_id`, `name`, `birthdate`, `gender`, `hire\_date`)

VALUES (101, 'Bryan', '1988-08-12', 'M', '2015-08-26'), (102, 'Joseph', '1978-05-12', 'M', '2014-10-21'), (103, 'Mike', '1984-10-13', 'M', '2017-10-28'), (104, 'Daren', '1979-04-11', 'F', '2006-11-01'), (105, 'Marie', '1990-02-11', 'F', '2018-10-12'), (106, 'Marco', '1988-04-11', 'M', '2010-10-12'), (107, 'Antonio', '1982-02-15', 'M', '2005-10-12');

Select \* from Employees;

USE `havva.beyen\_db1`;

CREATE TABLE Payment (payment\_id INT PRIMARY KEY NOT NULL AUTO\_INCREMENT, emp\_id INT NOT NULL, amount INT NOT NULL, payment\_date DATE NOT NULL);

INSERT INTO Payment (payment\_id, emp\_id, amount, payment\_date)

VALUES (301, 101, 1200, '2015-09-15'), (302, 101, 1200, '2015-09-30'), (303, 103, 1500, '2015-10-15'), (304, 103, 1500, '2015-10-30'), (305, 102, 1800, '2015-09-15'), (306, 102, 1800, '2015-09-30');

Select \* from Payment;

USE `havva.beyen\_db1`;

DELETE from Employees WHERE emp\_id=107;

Select \* from Employees;

USE `havva.beyen\_db1`;

Select \* from Employees;

DELETE from Employees;

Select \* from Employees;

# Select örnekleri

USE `havva.beyen\_db1`;

Select \* from Payment;

USE `havva.beyen\_db1`;

Select amount from Payment;

USE `havva.beyen\_db1`;

Select payment\_date,amount from Payment;

# Unique örnekleri

USE `havva.beyen\_db1`;

CREATE TABLE Person3(

id int,

name varchar(45)DEFAULT NULL,

email varchar(45)DEFAULT NULL UNIQUE,

city varchar(25)DEFAULT NULL

);

Select \* from Person3;​

# 6.HAFTA

// **CLAUSES:**

FFROM - WHERE - HAVING - ORDER BY - GROUP BY- SUM - MIN - MAX - AVG - INNER JOIN - OUTER JOIN - AND - OR - DISTINCT - ASC - DESC – COUNT

# DATABASE – TABLO OLUŞTURMA

create database `havva.beyen\_Lesson`;

USE `havva.beyen\_Lesson`;

CREATE TABLE Officers

(

officer\_id INT,

officer\_name VARCHAR(35),

address VARCHAR(45)

);

USE `havva.beyen\_Lesson`;

Select \* from Officers;

USE `havva.beyen\_Lesson`;

INSERT INTO Officers

(officer\_id, officer\_name, address)

VALUES

(1, 'Ajeet', 'Mau'),

(2, 'Deepika', 'Lucknow'),

(3, 'Vimal', 'Faizabad'),

(4, 'Rahul', 'Lucknow');

Select \* from Officers;

USE `havva.beyen\_Lesson`;

Select \* from Officers;

SELECT \* FROM Officers

WHERE address = 'Mau';

USE `havva.beyen\_Lesson`;

Select address from Officers;

SELECT \* FROM Officers

WHERE address = 'Mau';

# AND

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers WHERE address = 'Lucknow'

AND officer\_id < 5;

# OR

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers

WHERE address = 'Lucknow'

OR address = 'Mau';

# AND – OR

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers

WHERE (address = 'Mau' AND officer\_name = 'Ajeet')

OR (officer\_id < 5);

# DISTINCT

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers ;

SELECT DISTINCT address FROM Officers;

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers ;

SELECT DISTINCT address FROM Officers;

SELECT DISTINCT officer\_name, address

FROM Officers;

# WHERE

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers

WHERE officer\_id <= 3;

# BAZI ŞART İFADELERİNİ BU TABLO ÜZERİNDEN GERÇEKLEŞTİRDİK

USE `havva.beyen\_Lesson`;

CREATE TABLE `students` (

`student\_id` INT NOT NULL,

`student\_name` VARCHAR(35) NOT NULL,

`course\_name` VARCHAR(55) NOT NULL

);

USE `havva.beyen\_Lesson`;

INSERT INTO students

(student\_id, student\_name, course\_name)

VALUES

(1, 'Aryan', 'Java'),

(2, 'Rohini', 'Hadoop'),

(3, 'Lallu', 'MangoDB');

USE `havva.beyen\_Lesson`;

SELECT \* FROM students;

# 2 TABLOYU BİRLEŞTİRME

# inner join

USE `havva.beyen\_Lesson`;

SELECT Officers.officer\_id, students.student\_name

FROM students

INNER JOIN Officers

ON students.student\_id = Officers.officer\_id;

# outer join

USE `havva.beyen\_Lesson`;

SELECT Officers.officer\_id, students.student\_name

FROM Officers

LEFT OUTER JOIN students

ON Officers.officer\_id = students.student\_id;

# ORDER BY

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers;

SELECT \* FROM Officers WHERE address = 'Lucknow'

ORDER BY officer\_name;

# ASC

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers;

SELECT \* FROM Officers WHERE address = 'Lucknow'

ORDER BY officer\_name ASC;

# DESC

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers;

SELECT \* FROM Officers WHERE address = 'Lucknow'

ORDER BY officer\_name DESC;

# DESC - ASC

USE `havva.beyen\_Lesson`;

SELECT \* FROM Officers;

SELECT officer\_name, address FROM Officers WHERE officer\_id < 5

ORDER BY officer\_name DESC, address ASC;

# COUNT

USE `havva.beyen\_Lesson`;

SELECT address, COUNT(\*)

FROM Officers

GROUP BY address;

# GROUP BY

USE `havva.beyen\_Lesson`;

SELECT address, COUNT(\*)

FROM Officers

GROUP BY address;

# BAZI ŞART İFADELERİNİ BU TABLO ÜZERİNDEN GERÇEKLEŞTİRDİK

USE `havva.beyen\_Lesson`;

CREATE TABLE `employees` (

`ID` int NOT NULL,

`Name` VARCHAR(35) DEFAULT NULL,

`Email` VARCHAR(100) DEFAULT NULL,

`Phone` VARCHAR(15) DEFAULT NULL,

`City` VARCHAR(35) DEFAULT NULL,

`Working\_hours` INT NOT NULL

);

USE `havva.beyen\_Lesson`;

INSERT INTO `employees`

(`ID`, `Name`, `Email`, `Phone`, `City`, `Working\_hours`) VALUES

(1, 'Peter', 'peter@izu.edu.tr', '5467878467', 'Texas', 12),

(2, 'Suzi', 'suzi@izu.edu.tr', '768589489', 'California', 10),

(3, 'Joseph', 'joseph@izu.edu.tr', '587439304', 'New York', 6),

(4, 'Alex', 'alex@izu.edu.tr', '578578854', 'Los Angeles', 9),

(1, 'Peter', 'peter@izu.edu.tr', '5467878467', 'Texas', 12),

(2, 'Suzi', 'suzi@izu.edu.tr', '768589489', 'California', 10),

(3, 'Joseph', 'joseph@izu.edu.tr', '587439304', 'New York', 6),

(4, 'Alex', 'alex@izu.edu.tr', '578578854', 'Los Angeles', 9),

(1, 'Peter', 'peter@izu.edu.tr', '5467878467', 'Texas', 12),

(2, 'Suzi', 'suzi@izu.edu.tr', '768589489', 'California', 10),

(1, 'Peter', 'peter@izu.edu.tr', '5467878467', 'Texas', 12),

(2, 'Suzi', 'suzi@izu.edu.tr', '768589489', 'California', 10);

Select \* from employees;

# SUM

USE `havva.beyen\_Lesson`;

SELECT Name, SUM(Working\_hours) AS "Total working hours"

FROM employees

GROUP BY Name;

# MIN

USE `havva.beyen\_Lesson`;

SELECT Name, MIN(Working\_hours) AS "Minimum working hour" FROM employees

GROUP BY Name;

# MAX

USE `havva.beyen\_Lesson`;

SELECT Name, MAX(Working\_hours) AS "Minimum working hour" FROM employees

GROUP BY Name;

# AVG

USE `havva.beyen\_Lesson`;

SELECT Name, AVG(Working\_hours) AS "Minimum working hour" FROM employees

GROUP BY Name;

# HAVING

USE `havva.beyen\_Lesson`;

SELECT Name, SUM(Working\_hours) AS "Total working hours"

FROM employees

GROUP BY Name

HAVING SUM(Working\_hours) > 5;

// HAVING GROUP BY DEN SONRA WHERE TAM TERSİ

# Hafta 7

Joins | inner – outer – left – right – cross

CREATE DATABASE `havva.beyen\_db7`;

USE `havva.beyen\_db7`;

CREATE TABLE officers

(

officer\_id INT,

officer\_name VARCHAR(35),

address VARCHAR(45)

);

INSERT INTO officers

(officer\_id, officer\_name, address)

VALUES

(1, 'Ajeet', 'Mau'),

(2, 'Deepika', 'Lucknow'),

(3, 'Vimal', 'Faizabad'),

(4, 'Rahul', 'Lucknow');

SELECT \* FROM officers;

CREATE TABLE `students` (

`student\_id` INT NOT NULL,

`student\_name` VARCHAR(35) NOT NULL,

`course\_name` VARCHAR(55) NOT NULL

);

INSERT INTO students

(student\_id, student\_name, course\_name)

VALUES

(1, 'Aryan', 'Java'),

(2, 'Rohini', 'Hadoop'),

(3, 'Lallu', 'MangoDB');

Select \* from officers;

Select \* from students;

# İNNER

USE `havva.beyen\_db7`;

SELECT officers.officer\_name, officers.address, students.course\_name

FROM officers

INNER JOIN students

ON officers.officer\_id = students.student\_id;

# LEFT

USE `havva.beyen\_db7`;

SELECT officers.officer\_name, officers.address, students.course\_name

FROM officers

LEFT JOIN students

ON officers.officer\_id = students.student\_id;

# RİGHT

USE `havva.beyen\_db7`;

SELECT officers.officer\_name, officers.address, students.course\_name

FROM officers

RIGHT JOIN students

ON officers.officer\_id = students.student\_id;

# İNNER JOİN ÖRNEĞİ

USE `havva.beyen\_db7`;

CREATE TABLE students2 (student\_id INT PRIMARY KEY, stud\_fname VARCHAR(25), stud\_lname VARCHAR(25), city VARCHAR(25) );

CREATE TABLE technologies (student\_id INT, tech\_id INT PRIMARY KEY, inst\_name VARCHAR(25), technology VARCHAR(25) );

INSERT INTO students2 (student\_id, stud\_fname, stud\_lname, city)

VALUES (1, 'Devine', 'Putin', 'Paris'), (2, 'Micheal', 'Clark', 'Sydney'), (3, 'Ethon', 'Miller', 'London'), (4, 'Mark', 'Strauss', 'New York');

INSERT INTO technologies (student\_id, tech\_id, inst\_name, technology)

VALUES (1,1,'Java Training Inst','Java'), (2,2,'Chroma Campus','Angular'), (3,3,'CETPA Infotech','Big Data'), (4,4,'Aptron','IOS');

Select \* from students2;

Select \* from technologies;

USE `havva.beyen\_db7`;

SELECT students2.stud\_fname,

students2.stud\_lname, students2.city,

technologies.technology

FROM students2

INNER JOIN technologies

ON students2.student\_id = technologies.tech\_id;

# RİGHT ÖRNEĞİ

USE `havva.beyen\_db7`;

SELECT students2.stud\_fname,

students2.stud\_lname, students2.city,

technologies.technology

FROM students2

RIGHT JOIN technologies

ON students2.student\_id = technologies.tech\_id;

# LEFT ÖRNEĞİ

SELECT students2.stud\_fname,

students2.stud\_lname, students2.city,

technologies.technology

FROM students2

LEFT JOIN technologies

ON students2.student\_id = technologies.tech\_id;

# LEFT – USİNG ÖRNEĞİ

USE `havva.beyen\_db7`;

SELECT students2.stud\_fname,

students2.stud\_lname, students2.city,

technologies.technology

FROM students2

LEFT JOIN technologies

USING (student\_id);

# İNNER – USİNG ÖRNEĞİ

USE `havva.beyen\_db7`;

SELECT student\_id, inst\_name, city, technology

FROM students2

INNER JOIN technologies

USING (student\_id);

# İNNER – USİNG – WHERE ÖRNEĞİ

SELECT tech\_id, inst\_name, city, technology

FROM students2

INNER JOIN technologies

USING (student\_id) WHERE technology = "Java";

# İNNER ÖRNEĞİ

USE `havva.beyen\_db7`;

CREATE TABLE `contacts` (`contact\_id` INT NOT NULL, `cellphone`

VARCHAR(11) DEFAULT NULL, `homephone` VARCHAR(11) DEFAULT NULL);

INSERT INTO `contacts` (`contact\_id`, `cellphone`, `homephone`)

VALUES (1, '4673567', '7346575'), (2, '8758575', '7589488'),

(3, '7584855', '2029322'), (4, '5453432', '8753494');

Select \* from contacts;

USE `havva.beyen\_db7`;

SELECT student\_id, inst\_name, city,

technology, cellphone

FROM students2

INNER JOIN technologies USING (student\_id)

INNER JOIN contacts ORDER BY student\_id;

USE `havva.beyen\_db7`;

CREATE TABLE `customers` ( `customer\_id` INT NOT NULL, `customer\_name` VARCHAR(25),

`occupation` VARCHAR(25) DEFAULT NULL, `income` BIGINT, `qualification` VARCHAR(9) );

CREATE TABLE `employees` ( `emp\_id` INT NOT NULL, `emp\_name` VARCHAR(25), `email` VARCHAR(100));

INSERT INTO `customers` (`customer\_id`, `customer\_name`, `occupation`, `income`, `qualification`)

VALUES

(1, 'John Miller', 'Developer', 20000, 'Btech'),

(2, 'Mark Robert', 'Engineer', 40000, 'Btech'),

(3, 'Reyan Watson', 'Scientist', 60000, 'MSc'),

(4, 'Shane Trump', 'Business', 10000, 'MBA'),

(5, 'Adam Obama', 'Manager', 80000, 'MBA'),

(6, 'Rincky Ponting', 'Cricketer', 200000, 'Btech');

INSERT INTO `employees` (`emp\_id`, `emp\_name`, `email`)

VALUES

(1, 'Peter', 'peter@izu.edu.tr'),

(2, 'Suzi', 'suzi@izu.edu.tr'),

(3, 'Joseph', 'joseph@izu.edu.tr'),

(4, 'Alex', 'alex@izu.edu.tr');

# İNNER - WHERE ÖRNEĞİ

USE `havva.beyen\_db7`;

CREATE TABLE `customers7` ( `customer\_id` INT NOT NULL, `customer\_name` VARCHAR(25),

`occupation` VARCHAR(25) DEFAULT NULL, `income` BIGINT, `qualification` VARCHAR(9) );

CREATE TABLE `employees7` ( `emp\_id` INT NOT NULL, `emp\_name` VARCHAR(25), `email` VARCHAR(100));

INSERT INTO `customers7` (`customer\_id`, `customer\_name`, `occupation`, `income`, `qualification`)

VALUES

(1, 'John Miller', 'Developer', 20000, 'Btech'),

(2, 'Mark Robert', 'Engineer', 40000, 'Btech'),

(3, 'Reyan Watson', 'Scientist', 60000, 'MSc'),

(4, 'Shane Trump', 'Business', 10000, 'MBA'),

(5, 'Adam Obama', 'Manager', 80000, 'MBA'),

(6, 'Rincky Ponting', 'Cricketer', 200000, 'Btech');

INSERT INTO `employees7` (`emp\_id`, `emp\_name`, `email`)

VALUES

(1, 'Peter', 'peter@izu.edu.tr'),

(2, 'Suzi', 'suzi@izu.edu.tr'),

(3, 'Joseph', 'joseph@izu.edu.tr'),

(4, 'Alex', 'alex@izu.edu.tr');

USE `havva.beyen\_db7`;

SELECT emp\_id, occupation, income, qualification

FROM employees7

INNER JOIN customers7

WHERE income>20000 and income<80000;

# LEFT ÖRNEĞİ

USE `havva.beyen\_db7`;

CREATE TABLE orders (

order\_id int NULL,

date DATE NULL,

customer\_id int,

price int

);

INSERT INTO orders (order\_id, date,customer\_id,price)VALUES(1001, '2020-03-20', 2, 3000),

(1002, '2020-02-15', 4, 2500),

(1003, '2020-01-31', 5, 5000),

(1004, '2020-03-10', 2, 1500),

(1005, '2020-02-20', 1, 4500);SELECT\*from orders;

USE `havva.beyen\_db7`;

Select \* from customers7;

Select \* from orders;

SELECT customers7.customer\_id, customer\_name, price, date

FROM customers7

LEFT JOIN orders ON customers7.customer\_id = orders.customer\_id;

# LEFT – USİNG ÖRNEĞİ

USE `havva.beyen\_db7`;

Select \* from customers7;

Select \* from orders;

SELECT customers7.customer\_id, customer\_name, price, date

FROM customers7

LEFT JOIN orders

USING (customer\_id);

/\* bu kod hatalı

USE `havva.beyen\_db7`;

Select \* from customers7;

Select \* from orders;

SELECT customers7.customer\_id, customer\_name, price, date

FROM customers7

LEFT JOIN orders ON customers7.customer\_id = orders.customer\_id

GROUP BY price;

\*/

# LEFT ÖRNEKLERİ

USE `havva.beyen\_db7`;

SELECT customer\_id, customer\_name, occupation, price, date

FROM customers7

LEFT JOIN orders

USING(customer\_id) WHERE price>2500;

USE `havva.beyen\_db7`;

SELECT customers7.customer\_id, customer\_name, order\_id, price, cellphone

FROM customers7

LEFT JOIN contacts ON customer\_id = contact\_id

LEFT JOIN orders ON customers7.customer\_id = orders.customer\_id ORDER BY income;

USE `havva.beyen\_db7`;

SELECT customer\_name, occupation, order\_id, price, date

FROM customers7 LEFT JOIN orders ON price=2500;

# Cross join

Örnek

# Hafta 10

stored procedure

// işlemleri üzerine yapacağımız tabloyu oluşturuyoruz

CREATE DATABASE `havva.beyen\_db10`;

USE `havva.beyen\_db10`;

CREATE TABLE student\_info (

stud\_id INT AUTO\_INCREMENT NOT NULL PRIMARY KEY,

stud\_code INT NOT NULL,

stud\_name VARCHAR(24),

subject VARCHAR(24),

marks INT,

phone INT);

INSERT INTO student\_info (stud\_code, stud\_name, subject, marks, phone)

VALUES

(101, 'Mark', 'English', 68, 746756),

(102, 'Joseph', 'Physics', 70, 786876),

(103, 'John', 'Maths', 70, 657656),

(104, 'Barack', 'Maths', 90, 786786),

(105, 'Rinky', 'Maths', 85, 657664),

(106, 'Adam', 'Science', 92, 456777),

(107, 'Andrew', 'Science', 83, 425667),

(108, 'Brayan', 'Science', 85, 546544),

(109, 'Alexander', 'Biology', 67, 758675);

Select \* from student\_info;

# Yordam oluşturma

// delimiter: procedure içinde olan virvülleri tanımak için , mysql’e özel bir komuttur

USE `havva.beyen\_db10`;

DELIMITER &&

CREATE PROCEDURE get\_merit\_student ()

BEGIN

SELECT \* FROM student\_info WHERE marks > 70;

SELECT COUNT(stud\_code) AS Total\_Student FROM student\_info;

END &&

DELIMITER ;

# Yordam çağırma

USE `havva.beyen\_db10`;

CALL get\_merit\_student();

# Yordam oluşturma **IN** işlemli

// in : önce parametre al sonra işlemleri yap

USE `havva.beyen\_db10`;

DELIMITER &&

CREATE PROCEDURE get\_student (IN var1 INT)

BEGIN

SELECT \* FROM student\_info LIMIT var1;

SELECT COUNT(stud\_code) AS Total\_Student FROM student\_info;

END &&

DELIMITER ;

# **IN** işlemli yordamı çağırma

USE `havva.beyen\_db10`;

CALL get\_student(4);

# Yordam oluşturma **OUT** işlemli

// out önce işlem yap sonra göster

USE `havva.beyen\_db10`;

DELIMITER &&

CREATE PROCEDURE display\_max\_mark (OUT highestmark INT)

BEGIN

SELECT MAX(marks) INTO highestmark FROM student\_info;

END &&

DELIMITER ;

# **OUT** işlemli yordamı çağırma

CALL display\_max\_mark(@M);

SELECT @M;

# Yordam oluşturma **INOUT** işlemli

USE `havva.beyen\_db10`;

DELIMITER &&

CREATE PROCEDURE display\_marks (INOUT var1 INT)

BEGIN

SELECT marks INTO var1 FROM student\_info WHERE stud\_id = var1;

END &&

DELIMITER ;

# **INOUT** işlemli yordamı çağırma

USE `havva.beyen\_db10`;

SET @M = '3';

CALL display\_marks(@M);

SELECT @M;

# Yordamları gösterme

SHOW PROCEDURE STATUS;

SHOW **PROCEDURE** STATUS **WHERE** db = 'havva.beyen\_db10';

# Yordamları slime

USE `havva.beyen\_db10`;

SHOW PROCEDURE STATUS ;

DROP PROCEDURE display\_marks;

SHOW PROCEDURE STATUS ;

# Yordamları güncellemek

Sınırlayıcı hatasını Verdi !!

USE `havva.beyen\_db10`;

ALTER PROCEDURE get\_merit\_student ;

COMMENT `It displays all records` ;

DELIMITER &&

ALTER PROCEDURE get\_merit\_student ()

BEGIN

SELECT \* FROM student\_info WHERE marks > 60;

SELECT COUNT(stud\_code) AS Total\_Student FROM student\_info;

END &&

DELIMITER ;

# Hafta 11

Union

# UNION KULLANARAK BİRLEŞME

USE `havva.beyen\_db10`;

CREATE TABLE student(

Stud\_ID int,

Name varchar(45) NOT NULL,

Email varchar(45),

City varchar(25),

);

USE `havva.beyen\_db10`;

CREATE TABLE student2(

Stud\_ID int,

Name varchar(45) NOT NULL,

Email varchar(45),

City varchar(25),

);

INSERT INTO Student (Stud\_ID, Name, Email, City)

VALUES (1, "Peter", "peter@std.izu.edu.tr", "Texas"),

(2, "Suzi", "suzi@std.izu.edu.tr", "California"),

(3, "Joseph", "joseph@std.izu.edu.tr", "Alaska"),

(4, "Andrew", "andrew@std.izu.edu.tr", "Los Angeles"),

(5, "Brayan", "brayan@std.izu.edu.tr", "New York");

INSERT INTO Student2 (Stud\_ID, Name, Email, City)

VALUES (1, "Stephen", "stephen@std.izu.edu.tr", "Texas"),

(2, "Joseph", "joseph@std.izu.edu.tr", "Los Angeles"),

(3, "Peter", "peter@std.izu.edu.tr", "California"),

(4, "David", "david@std.izu.edu.tr", "New York"),

(5, "Maddy", "maddy@std.izu.edu.tr", "Los Angeles");

SELECT Name FROM student1

UNION

SELECT Name FROM student2;

# UNION WITH AS

USE `havva.beyen\_db10`;

CREATE TABLE student1(

Stud\_ID int,

stud\_name varchar(45) NOT NULL,

Email varchar(45),

subject varchar(45) NOT NULL,

marks int,

City varchar(25),

);

USE `havva.beyen\_db10`;

CREATE TABLE student2(

Stud\_ID int,

stud\_name varchar(45) NOT NULL,

Email varchar(45),

subject varchar(45) NOT NULL,

marks int,

City varchar(25),

);

INSERT INTO Student (Stud\_ID, stud\_name, Email, subject , marks ,City)

VALUES (1, "Peter", "peter@std.izu.edu.tr", “math”,90, "Texas"),

(2, "Suzi", "suzi@std.izu.edu.tr", “IT”,92, "California"),

(3, "Joseph", "joseph@std.izu.edu.tr",“math”,93, "Alaska"),

(4, "Andrew", "andrew@std.izu.edu.tr",“art”,94, "Los Angeles"),

(5, "Brayan", "brayan@std.izu.edu.tr", “IT”,95, "New York");

INSERT INTO Student2 (Stud\_ID, stud\_name, Email, subject , marks ,City)

VALUES (1, "Peter", "peter@std.izu.edu.tr", “math”,90, "Texas"),

(2, "Suzi", "suzi@std.izu.edu.tr", “IT”,92, "California"),

(3, "Joseph", "joseph@std.izu.edu.tr",“math”,93, "Alaska"),

(4, "Andrew", "andrew@std.izu.edu.tr",“art”,94, "Los Angeles"),

(5, "Brayan", "brayan@std.izu.edu.tr", “IT”,95, "New York");

SELECT stud\_name AS student\_name, subject AS course FROM student

UNION

SELECT stud\_name, subject FROM student2;

# UNION WITH ORDER BY

(SELECT stud\_name, subject, marks FROM students)

UNION

(SELECT stud\_name, subject, marks FROM student2)

ORDER BY marks;

# UNION ALL KULLANARAK 2 TABLO BİRLEŞİMİ

(SELECT stud\_name, subject, marks FROM student)

UNION ALL

(SELECT stud\_name, subject, marks FROM student2)

ORDER BY marks;

# UNION VE UNION ALL ÖRNEĞİ

SELECT City FROM Student

UNION

SELECT City FROM Student2

ORDER BY City;

SELECT City FROM student

UNION ALL

SELECT City FROM student2

ORDER BY City;

Minus

# MINUS KULLANARAK FARK ALMAK

SELECT city FROM student WHERE Stud\_ID =1 ;

MINUS

SELECT city FROM student2 WHERE Stud\_ID =3 ;

INTERSECT

# INTERSECT = SET ANLAMINDA

SELECT city FROM student WHERE Stud\_ID =1 ;

INTERSECT

SELECT city FROM student2 WHERE Stud\_ID = NULL ;

### Hafta 12

CREATE TABLE employee(

id INT PRIMARY KEY AUTO\_INCREMENT,

name varchar(45) NOT NULL,

occupation varchar(35) NOT NULL,

department\_id int(35) NOT NULL,

working\_date date,

working\_hours varchar(10)

);

USE `havva.beyen\_db12`;

INSERT INTO employee (name, occupation, department\_id, working\_date, working\_hours)

VALUES

('Robin', 'Scientist', 1, '2020-10-04', 12),

('Warner', 'Engineer', 1,'2020-10-04', 10),

('Peter', 'Actor', 2, '2020-10-04', 13),

('Marco', 'Doctor', 3, '2020-10-04', 14),

('Brayden', 'Teacher', 3, '2020-10-04', 12),

('Antonio', 'Business', 3, '2020-10-04', 11);

# COUNT

USE `havva.beyen\_db12`;

SELECT COUNT(name) FROM employee;

# SUM

USE `havva.beyen\_db12`;

SELECT SUM(working\_hours) AS "total working hours" FROM employee;

# AVG

USE `havva.beyen\_db12`;

SELECT AVG(working\_hours) AS "Average working hours" FROM employee;

# MIN

USE `havva.beyen\_db12`;

SELECT MIN(working\_hours) AS Minimum\_working\_hours FROM employee;

# MAX

USE `havva.beyen\_db12`;

SELECT MAX(working\_hours) AS Maximum\_working\_hours FROM employee;

# FIRST

USE `havva.beyen\_db12`;

SELECT working\_date FROM employee LIMIT 1;

# LAST

USE `havva.beyen\_db12`;

SELECT working\_hours FROM employee ORDER BY name DESC LIMIT 1;

# GROUP\_CONCAT

USE `havva.beyen\_db12`;

SELECT department\_id, occupation, working\_date, working\_hours,

GROUP\_CONCAT(occupation) as "designation"

FROM employee GROUP BY department\_id;

USE `havva.beyen\_db12`;

CREATE TABLE `employee\_detail` (

`ID` int AUTO\_INCREMENT PRIMARY KEY,

`Name` varchar(35) DEFAULT NULL,

`Email` varchar(100) DEFAULT NULL,

`Phone` varchar(15) DEFAULT NULL,

`City` varchar(35) DEFAULT NULL,

`Working\_hours` int NOT NULL

)

USE `havva.beyen\_db12`;

ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

INSERT INTO `employee\_detail` (`ID`, `Name`, `Email`, `Phone`, `City`, `Working\_hours`) VALUES

(1, 'Peter', 'peter@izu.edu.tr', '5467878467', 'Texas', 12),

(2, 'Suzi', 'suzi@izu.edu.tr', '768589489', 'California', 10),

(3, 'Joseph', 'joseph@izu.edu.tr', '587439304', 'New York', 14),

(4, 'Alex', 'alex@izu.edu.tr', '578578854', 'Los Angeles', 9),

(5, 'Mark', 'mark@izu.edu.tr', '485689053', 'Washington', 12),

(6, 'Stephen', 'stephen@izu.edu.tr', '986789799', 'New York', 10),

(7, 'Taylor', 'taylor@izu.edu.tr', '7467347', 'New York', 13);

# COUNT(EXPRESSION)

USE `havva.beyen\_db12`;

SELECT COUNT(Name) FROM employee\_detail;

# COUNT(\*)

USE `havva.beyen\_db12`;

SELECT COUNT(\*) FROM employee\_detail WHERE Working\_hours>10;

# COUNT(DISTINICT)

USE `havva.beyen\_db12`;

SELECT COUNT(DISTINCT Working\_hours) FROM employee\_detail;

# COUNT WITH GROUP BY

USE `havva.beyen\_db12`;

SELECT Name, City, COUNT(\*) FROM employee\_detail GROUP BY city;

# COUNT WITH HAVING AND ORDER BY

USE `havva.beyen\_db12`;

SELECT GROUP\_CONCAT(Name), City, COUNT(\*) FROM employees\_detail

GROUP BY City

HAVING COUNT(\*)>=2

ORDER BY COUNT(\*);

# SUM

USE `havva.beyen\_db12`;

SELECT SUM(working\_hours) AS "Total working hours" FROM employee;

# SUM WITH WHERE

USE `havva.beyen\_db12`;

SELECT SUM(working\_hours) AS "Total working hours" FROM employee WHERE working\_hours>=12;

# SUM WITH GROUP BY

USE `havva.beyen\_db12`;

SELECT department\_id, GROUP\_CONCAT(Name), occupation, SUM(working\_hours) AS "Total working hours" FROM employee GROUP BY department\_id;

# SUM WITH HAVING

USE `havva.beyen\_db12`;

SELECT ID, Name, occupation,

SUM(working\_hours) Total\_working\_hours

FROM employee

GROUP BY department\_id

HAVING SUM(working\_hours)>20;

# SUM WITH DISTINCT

USE `havva.beyen\_db12`;

SELECT Name, occupation,

SUM(DISTINCT working\_hours) Total\_working\_hours

FROM employee

GROUP BY department\_id;

# AVG

USE `havva.beyen\_db12`;

SELECT AVG(working\_hours) Avg\_working\_hours FROM employee;

# AVG WITH WHERE

USE `havva.beyen\_db12`;

SELECT AVG(working\_hours) Avg\_working\_hours FROM employee WHERE working\_hours>=12;

# AVG WITH GROUP BY

USE `havva.beyen\_db12`;

SELECT Name, department\_id, AVG(working\_hours) Avg\_working\_hours FROM employee GROUP BY department\_id;

# AVG WITH HAVING

USE `havva.beyen\_db12`;

SELECT GROUP\_CONCAT(name) Names, department\_id,

AVG(working\_hours) Avg\_working\_hours

FROM employees

GROUP BY department\_id

HAVING AVG(working\_hours)>11;

# AVG WITH DISTINCT

USE `havva.beyen\_db12`;

SELECT GROUP\_CONCAT(name) Names, department\_id,

AVG(DISTINCT working\_hours) Avg\_working\_hours

FROM employee

GROUP BY department\_id;

USE `havva.beyen\_db12`;

ALTER TABLE `employee` ADD `age` INT NOT NULL AFTER `working\_hours`;

INSERT INTO `employee` (id, age)

VALUES ('Robin', 'Scientist', 1, '2020-10-04', 12, 32),

('Warner', 'Engineer', 1,'2020-10-04', 10, 43), ('Peter', 'Actor', 2, '2020-10-04', 13, 34),

('Marco', 'Doctor', 3, '2020-10-04', 14, 55), ('Brayden', 'Teacher', 3, '2020-10-04', 12, 38),

('Antonio', 'Business', 3, '2020-10-04', 11, 46) AS vals ON DUPLICATE KEY UPDATE age=vals.age;

# MIN

USE `havva.beyen\_db12`;

SELECT MIN(working\_hours) AS Minimum\_working\_hour FROM employee;

# MIN WITH WHERE

USE `havva.beyen\_db12`;

SELECT MIN(working\_hours) AS Minimum\_working\_hours

FROM employee

WHERE age >= 32 AND age <= 40;

# MIN WITH GROUP BY

USE `havva.beyen\_db12`;

SELECT age, MIN(working\_hours) AS Minimum\_working\_hours

FROM employee

GROUP BY department\_id;

# MIN WITH HAVING

USE `havva.beyen\_db12`;

SELECT department\_id, MIN(working\_hours) AS Minimum\_working\_hours

FROM employee

GROUP BY department\_id

HAVING MIN(working\_hours) > 10;

# MON WITH DISTINCT

USE `havva.beyen\_db12`;

SELECT name, department\_id, MIN(DISTINCT working\_hours) AS Minimum\_working\_hours

FROM employee

GROUP BY department\_id;

# MAX

USE `havva.beyen\_db12`;

SELECT MAX(working\_hours) AS "Maximum Working Hours" FROM employee;

# MAX WITH WHERE

USE `havva.beyen\_db12`;

SELECT MAX(working\_hours) AS "Maximum\_working\_hours"

FROM employee

WHERE age < 45;

# MAX WITH GROUP BY

USE `havva.beyen\_db12`;

SELECT department\_id, MAX(working\_hours) AS "Maximum Working Hours"

FROM employee

GROUP BY department\_id;

# MAX WITH HAVING

USE `havva.beyen\_db12`;

SELECT department\_id, MAX(working\_hours) AS "Maximum Working Hours"

FROM employee

GROUP BY department\_id

HAVING MAX(working\_hours) >= 13;

# MAX WITH DISTINCT

USE `havva.beyen\_db12`;

SELECT department\_id, MAX(DISTINCT working\_hours) AS "Maximum Working Hours"

FROM employee

GROUP BY department\_id;

# MAX() SUBQUERY FUNCTION

USE `havva.beyen\_db12`;

// MAX() FONKSİYONU SUBQUERY

SELECT \* FROM employee WHERE

age = (SELECT MAX(age) FROM employee);

# FIRST

USE `havva.beyen\_db12`;

SELECT Name

FROM employee\_detail

LIMIT 1;

# FIRST 2 RECORDS

USE `havva.beyen\_db12`;

SELECT Name

FROM employee\_detail

LIMIT 2;

# LAST

USE `havva.beyen\_db12`;

SELECT Name

FROM employee\_detail

ORDER BY ID DESC

LIMIT 1;

# LAST

USE `havva.beyen\_db12`;

SELECT Name

FROM employee\_detail

ORDER BY Working\_hours DESC

LIMIT 1;