1. Create view for finding the maximum salary from each department.  
  
SELECT \* FROM 5311prep.employees;

create view maxSalaryDeparments as

select MAX(salary) as max\_salary, department\_name from employees, departments

where employees.department\_id = departments.department\_id

group by department\_name;

select \* from maxSalaryDeparmen

2.Write an procedure to display employees who earn more than the average salary in that company

DELIMITER //

CREATE PROCEDURE MoreThenAVG()

BEGIN

select first\_name, salary, (select AVG(salary) FROM employees) AS average\_salary from employees

where salary > (select AVG(salary) from employees );

END //

DELIMITER ;

call MoreThenAVG();

3.Write procedure that divide people into three groups based on their salaries (groups are input parametrs for the procedure).

Ex. group1, group2, group 3,

less that group1, between group1 and group 2, between group 2 and group 3, greather than group 3

Kiko:  
DELIMITER //

CREATE PROCEDURE salaryGroups10(IN Group1 INTEGER, IN Group2 INTEGER, IN Group3 INTEGER)

BEGIN

SELECT first\_name AS name, salary AS salary, CONCAT('less than ', Group1) as salary\_group FROM employees

WHERE salary < Group1

UNION ALL

SELECT first\_name AS name, salary AS salary, CONCAT('between ', Group1 , ' and ', Group2) AS salary\_group FROM employees

WHERE salary >= Group1 AND salary < Group2

UNION ALL

SELECT first\_name AS name, salary AS salary, CONCAT('between ', Group2 , ' and ', Group3) AS salary\_group FROM employees

WHERE salary >= Group2 AND salary < Group3

UNION ALL

SELECT first\_name AS name, salary AS salary, CONCAT('more than ', Group3) AS salary\_group FROM employees

WHERE salary >= Group3;

END //

DELIMITER ;

call salaryGroups10(3000,5000,10000)

IVAN:  
DELIMITER //

CREATE PROCEDURE salaryGroups11(IN Group1 INTEGER, IN Group2 INTEGER, IN Group3 INTEGER)

BEGIN

SELECT

first\_name AS name,

salary AS salary,

CASE

WHEN salary < Group1 THEN CONCAT('less than ', Group1)

WHEN salary >= Group1 AND salary < Group2 THEN CONCAT('between ', Group1, ' and ', Group2)

WHEN salary >= Group2 AND salary < Group3 THEN CONCAT('between ', Group2, ' and ', Group3)

WHEN salary >= Group3 THEN CONCAT('more than ', Group3)

ELSE 'Unknown Group'

END AS salary\_group

FROM employees;

END //

DELIMITER ;

call salaryGroups11(3000,5000,10000)

4. create trigger for fetching when salary is updated for some employeer

CREATE TABLE Employees\_Loggers (

employee\_id INT,

first\_name VARCHAR(60),

last\_name VARCHAR(60),

old\_salary INT,

new\_salary INT

);

DELIMITER //

CREATE TRIGGER Employees\_Salary\_Update

AFTER UPDATE ON employees

FOR EACH ROW

BEGIN

IF NEW.salary != OLD.salary THEN

INSERT INTO Employees\_Loggers (employee\_id, first\_name, last\_name, old\_salary, new\_salary)

VALUES (OLD.employee\_id, OLD.first\_name, OLD.last\_name, OLD.salary, NEW.salary);

END IF;

END //

DELIMITER ;

UPDATE employees

SET salary = 8888

WHERE first\_name = 'Guy';

select \* from Employees\_Loggers;

select \* from employees;

SHOW TRIGGERS;

drop trigger Employees\_Salary\_Updatess

mongo :

db.createCollection("employees")

db.employees.insertMany([

{ name: "Zoran", role: "CEO" },

{ name: "Blashko", role: "Manager" },

{ name: "Kristian", role: "Sales" },

{ name: "Viktor", role: "Engineer" },

{ name: "Tamara", role: "Analyst" }

])