Create, Worker table with following fields: Worker_Id INT FirstName CHAR(25), LastName CHAR(25), Salary INT(15), JoiningDate DATETIME, Department CHAR(25))

```
CREATE TABLE WORKER(WORKER_ID INT ,
            FIRST_NAME CHAR(25) ,
            LAST_NAME CHAR (25) ,
            SALARY INT(15) ,
            JOINING_DATE DATETIME ,
            DEPARTMENT CHAR(25));
```

1. Create a stored procedure that takes in IN parameters for all the columns in the Worker table and adds a new record to the table and then invokes the procedure call.

```
DELIMITER $$
CREATE PROCEDURE ADD_WORKER(IN WORKER_USERID INT, IN WORKER_FNAME CHAR(25),
                            IN WORKER_LNAME CHAR (25),
                            IN WORKER_SALARY INT(15),
                            IN WORKER_JOINING_DATE DATETIME,
                            IN WORKER_DEPARTMENT CHAR(25))
BEGIN
INSERT INTO WORKER
VALUES(WORKER_USERID,WORKER_FNAME,WORKER_LNAME,WORKER_SALARY,WORKER_JOINING_DATE,WORKER_DEPARTMENT) ;
END $$
DELIMITER ;

CALL ADD_WORKER(1007,'SUNIL','JOSEPH',68000,'2012-01-01','HR');
```

Write stored procedure takes in an IN parameter for WORKER_ID and an OUT parameter for worker details. It should retrieve the details of the worker with the given ID. Then make the procedure call.

```
DELIMITER $$
CREATE PROCEDURE GET_WORKER_DETAILS(IN USER_ID INT)
BEGIN
SELECT * FROM WORKER WHERE WORKER_ID=USER_ID;
END $$
DELIMITER ;

CALL GET_WORKER_DETAILS(1002);
```

2. Write stored procedure takes in an IN parameter for WORKER_ID and an OUT parameter for SALARY. It should retrieve the salary of the worker with the given ID and returns it in the p_salary parameter. Then make the procedure call.

```
DELIMITER $$
CREATE PROCEDURE GET_SALARY(IN ID INT, OUT G_SALARY INT)
BEGIN
SELECT SALARY INTO G_SALARY FROM WORKER WHERE WORKER_ID= ID;
END $$
DELIMITER ;

SET @V_SALARY = 0;
CALL GET_SALARY(1002,@V_SALRY);
SELECT @V_SALRY AS SALARY;
```

3. Create a stored procedure that takes in IN parameters for WORKER_ID and DEPARTMENT. It should update the department of the worker with the given ID. Then make a procedure call.

```
DELIMITER $$
CREATE PROCEDURE UPDATE_DEPARTMENT(IN ID INT, IN GET_DEPARTMENT VARCHAR(25))
BEGIN
UPDATE WORKER SET DEPARTMENT=GET_DEPARTMENT  WHERE WORKER_ID=ID ;
SELECT DEPARTMENT INTO GET_DEPARTMENT FROM WORKER WHERE WORKER_ID=ID;
END $$
DELIMITER ;

SET @D_DEPARTMENT = 'IT';
CALL UPDATE_DEPARTMENT(1002,@D_DEPARTMENT);
SELECT @D_DEPARTMENT AS DEPARTMENT;
```

4. Write a stored procedure that takes in an IN parameter for DEPARTMENT and an OUT parameter for p_workerCount. It should retrieve the number of workers in the given department and returns it in the p_workerCount parameter. Make procedure call.

```
DELIMITER $$
CREATE PROCEDURE COUNT_DEPARTMENT(IN IN_DEPARTMENT VARCHAR(25), OUT
WORKER_COUNT INT)
BEGIN
        SELECT COUNT(DEPARTMENT) INTO WORKER_COUNT FROM WORKER WHERE
        DEPARTMENT=IN_DEPARTMENT;
END $$
DELIMITER ;

SET @WORKER_COUNT = 0;
CALL COUNT_DEPARTMENT('HR',@WORKER_COUNT);
SELECT @WORKER_COUNT AS NO_OF_WORKERS;
```

5. Write a stored procedure that takes in an IN parameter for DEPARTMENT and an OUT parameter for p_avgSalary. It should retrieve the average salary of all workers in the given department and returns it in the p_avgSalary parameter and call the procedure.

```
DELIMITER $$
                CREATE PROCEDURE AVERAGE_SALARY(IN IN_DEPARTMENT VARCHAR(25),
                OUT AVG_SALARY INT)
BEGIN
        SELECT AVG(SALARY) INTO AVG_SALARY FROM WORKER WHERE
        DEPARTMENT=IN_DEPARTMENT;
END $$
DELIMITER ;

SET @AVG_SAL = 0;
CALL AVERAGE_SALARY('HR',@AVG_SAL);
SELECT @AVG_SAL AS AVERAGE_SALARY_OF_DEPARTMENT;
```