# Part A: ER Diagram

Diagram

Description automatically generated

Figure 1: E R diagram of task A

## Task 1 and 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Employee | | | | |
| \*emp\_id | first\_name | last\_name | phone\_nu | past\_history |

Given scenario at task 1 can be represented by constructing six table, employee, department, jobtype, review, worker and supervise.

Table employee consist emp\_id attribute as varchar (string), primary key represent by \* symbol in table which store the unique id of employee. The first\_name and last\_name store the first name and last name of the employee respectively are varchar length 40 each. Credential phone number is store in phone\_nu attribute which is multivalued attribute and is integer with and the past history is store in attribute past\_history with varchar.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| worker | | | | |
| \*work\_id | emp\_id | work\_type | talent | duty |

The worker table contain work\_id as primary attribute and varchar. It is linked with one to one relationship with employee table with foreign key as emp\_id which is a varchar. It contains work\_type as attribute it store the data to employee whether he work in floor or office. Attribute like talent and duty are all string variable to store the talent and duty of the worker.

|  |  |  |
| --- | --- | --- |
| Department | | |
| \*dep\_id | emp\_id | dep\_name |

Table department has three attribute, dep\_id is a varchat with length 10 and a primary key. Department name is store in attribute dep\_name as varchar of length 40. It is linked with employee table with the foreign key emp\_id. The relation between both tables is a one-to-many relationship which show that one department contain many employee.

|  |  |  |  |
| --- | --- | --- | --- |
| job | | | |
| \*job\_id | #emp\_id | position | grade |

Job position and grade are store in another table name job in attribute job\_name and grade respectively both are varchar of length 40. It consists of emp\_id as foreign key to link the job table to the employee table. One to one relationship exists between both table.

|  |  |  |  |
| --- | --- | --- | --- |
| review | | | |
| \*review\_id | #emp\_id | #reviewed\_by | date |

The data of review of the employee by reviewer is store in the attribute in emp\_id and reviewed\_by respectively and attributes are varchar of length 10, both these are foreign key to employee and department table. It has one to many relationship with employee and department table which show that the many record of one employee and department can be found in review table. It contains the date of the review in attribute date.

|  |  |  |  |
| --- | --- | --- | --- |
| supervise | | | |
| \*super\_id | #mag\_id | #ass\_mag | employee |

The supervise table contain four attribute super\_id as primary key and it’s store the string datatype as varchar. It contain emp\_id as a linked between department table and has a relation of one to one relationship. It contain mag-id and ass\_mag to store the value of the manager id and assistant manager id.

# Part B: SQL Programming

## Task 1:

-- Creating table Employee

CREATE TABLE Employee (

empid VARCHAR(10) PRIMARY KEY,

name VARCHAR(50),

address VARCHAR(50),

DOB DATE,

job VARCHAR(50),

salaryCode VARCHAR(10),

depid VARCHAR(10),

manager VARCHAR(50) ,

schemeid VARCHAR(10)

);

-- Creating table Department

CREATE TABLE Department (

depid VARCHAR(10) PRIMARY KEY,

name VARCHAR(50)

);

-- Creating table SalaryGrade

CREATE TABLE SalaryGrade (

salaryCode VARCHAR(10) PRIMARY KEY,

startSalary INT(10),

finishSalary INT(10)

);

-- Creating table PensionScheme

CREATE TABLE PensionScheme (

schemeid VARCHAR(10) PRIMARY KEY,

name VARCHAR(50),

rate DECIMAL(5,2)

);

-- Altering the foreign key of employee table

ALTER TABLE Employee ADD FOREIGN KEY(depid) REFERENCES Department(depid) ON DELETE SET NULL;

ALTER TABLE Employee ADD FOREIGN KEY(schemeid) REFERENCES PensionScheme(schemeid) ON DELETE SET NULL;

ALTER TABLE Employee ADD FOREIGN KEY(salaryCOde) REFERENCES SalaryGrade(salaryCode) ON DELETE SET NULL;

-- First inserting the foregien key table data

INSERT INTO Department VALUES

('D10', 'Administration'),

('D20', 'Finance'),

('D30', 'Sales'),

('D40', 'Maintenance'),

('D50', 'IT Support');

INSERT INTO PensionScheme VALUES

('S110', 'AXA' ,0.5),

( 'S121', 'Premier', 0.6),

( 'S124' ,'Stakeholder', 0.4),

('S116', 'Standard' ,0.4);

INSERT INTO SalaryGrade VALUES

('S1' ,15000 ,18000),

('S2' ,18001 ,22000),

('S3', 22001 ,25000),

('S4', 25001, 29000),

('S5', 29001, 38000);

-- Inserting the data in table employee

INSERT INTO employee VALUES

('E101', 'Young, S.', '199 London Road', '2076/03/05' ,'Clerk' ,'S1', 'D10', 'E110', 'S116'),

('E301', 'April, H.', '20 Glade close', '2079/03/10' ,'Sales Person', 'S2', 'D30', 'E310', 'S124'),

('E310', 'Newgate, E.', '10 Heap Street', '2080/11/28' ,'Manager', 'S5', 'D30', NULL ,'S121'),

('E501', 'Teach, E', '22 railway road', '2072/02/12', 'Analyst' ,'S5', 'D50', NULL ,'S121'),

('E102', 'Hawkins, M.', '3 High Street', '2074/07/13', 'Clerk', 'S1', 'D10', 'E110', 'S116'),

('E110', 'Watkins, J.',' 11 crescent road' ,'2069/06/25', 'Manager', 'S5', 'D10', NULL ,'S121');

## Task 2: a

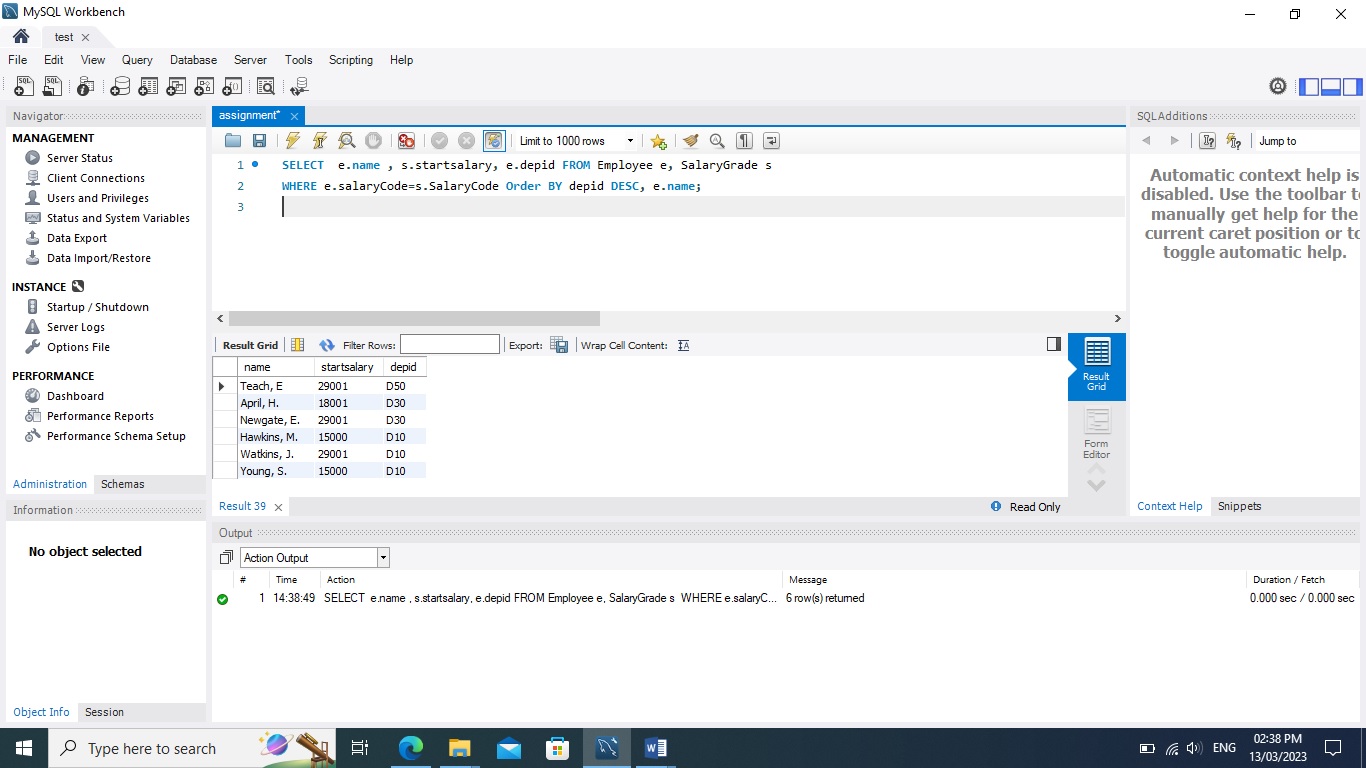
1. The name (in ascending order), the starting salary and department id of each employee within a descending order of department ids.

For this query we use select name, startsalary and depid from table employee and salarygrade putting condition using where salary code of employee = salarycode of salarygrade and order be depid in descending order and ascending order by employee name.

SELECT e.name , s.startsalary, e.depid FROM Employee e, SalaryGrade s

WHERE e.salaryCode=s.SalaryCode Order BY depid DESC, e.name;

|  |  |  |
| --- | --- | --- |
| **name** | **startsalary** | **depid** |
| Teach, E | 29001 | D50 |
| April, H. | 18001 | D30 |
| Newgate, E. | 29001 | D30 |
| Hawkins, M. | 15000 | D10 |
| Watkins, J. | 29001 | D10 |
| Young, S. | 15000 | D10 |



## Task 2: b

1. Give the number of employees for each of the pension schemes offered by the company. Result listing should include the name of each scheme and its corresponding number of employees who join the scheme.

In this second query we use select name from pensionscheme and use count function to schemeid of employee using where condition to check schemeid of pensionscheme and employee is equal and the grouping the result on schemeid of employee.

SELECT p.name, COUNT(e.schemeid) numberofemp From Pensionscheme p, Employee e WHERE p.schemeid =e.schemeid GROUP BY e.schemeid;

|  |  |
| --- | --- |
| **name** | **numberofemp** |
| Standard | 2 |
| Premier | 3 |
| Stakeholder | 1 |

Graphical user interface, text, application, email

Description automatically generated

## Task 2: c

1. Give the total number of employees who are not managers but currently receive an annual salary of over £35,000.

First we select name and empid from employee and checking condition using where and forming a sub query selecting salarycode from table salarygrade where finishsalary>35000. The given result is the compare with salarycode of employee. The salarycode of both should be equal. And we cancel the result which match empid E110 and E310 from employee table.

SELECT name, e.empid FROM Employee e where (SELECT s.SalaryCode FROM SalaryGrade s WHERE FinishSalary >35000 ) =e.SalaryCode AND e.empid!= ('E110') And e.empid!= ('E310') ;

|  |  |
| --- | --- |
| name | empid |
| Teach, E | E501 |

Graphical user interface, text, application, email

Description automatically generated

## Task 2: d

1. List the id and name of each employee along with his/her manager’s name.

For this query we select empid and name from employee table and name from table tt as manager\_ name. Table tt is subquery where we select empid and name from employee where managerid is null and join both table on manager from employee = to empid from table tt.

SELECT e.empid, e.name, tt.name AS Manager\_name FROM Employee e

JOIN

(SELECT a.empid, a.name FROM Employee a WHERE a.manager is null) AS tt ON e.manager = tt.empid ;

|  |  |  |
| --- | --- | --- |
| **empid** | **name** | **Manager\_name** |
| E101 | Young, S. | Watkins, J. |
| E102 | Hawkins, M. | Watkins, J. |
| E301 | April, H. | Newgate, E. |

Graphical user interface, text, application, email

Description automatically generated