

MIS582 Final Course Project

Student Name: Hardik Maru

Date: OCT-10-2023

Instructions for Students

You will create a database that stores information about ONLY the following entities.

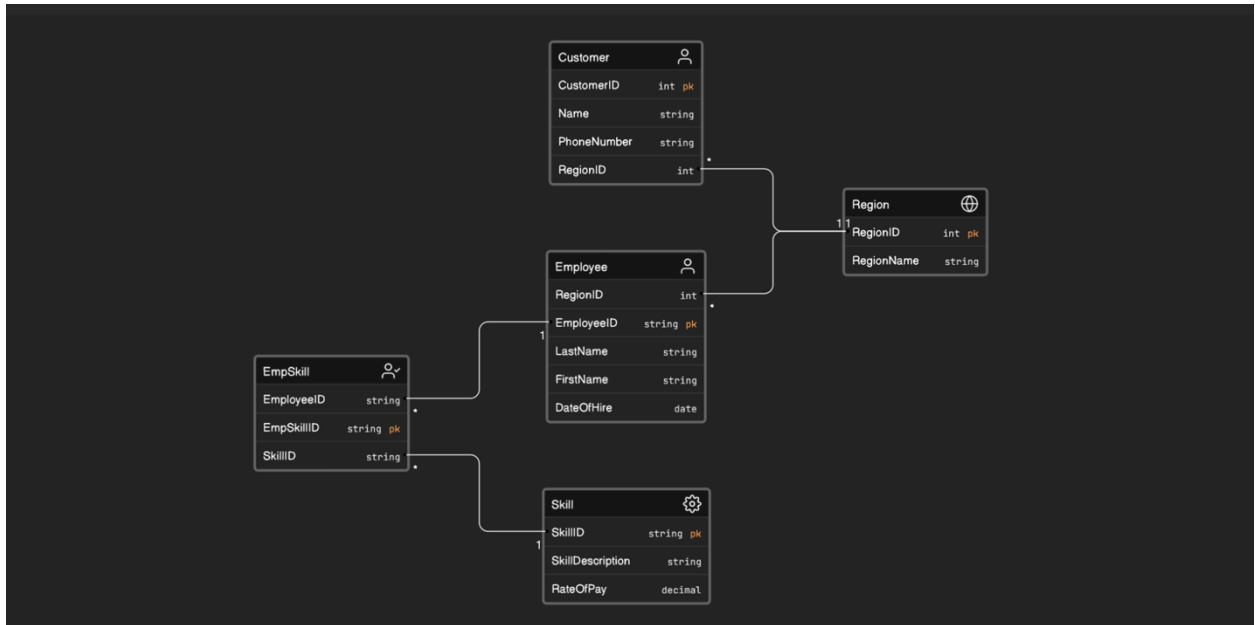
- Customer
- Region
- Employee
- Skill
- EmpSkill

The business rules below give the relationships between these entities.

- A customer is assigned to one region. One region can have several customers.
- An employee can have several skills. One skill can be learned by several employees.
- An employee works for one region. A region can have many employees.

Here are the steps you need to go through to do the project:

1. Given the business rules above, create a logical-level Crow's Foot ERD. ERD can be draw using any online tool such as:
<https://app.diagrams.net/>



Please copy and paste your ERD diagram here.

- original entities and any associative entities (intersection or intermediate tables).

Please copy and paste the script for create table and insert into statements here. Also run a select * statement and copy paste the fully populated tables here.

```

Create database projectDBfinal;

USE ProjectDBfinal;

CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR(25),
    PhoneNumber VARCHAR(20),
    RegionID INT,
    FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
);

CREATE TABLE Region (
    RegionID INT PRIMARY KEY,
    RegionName VARCHAR(25)
);

CREATE TABLE Employee (
    EmployeeID VARCHAR(20) PRIMARY KEY,
    LastName VARCHAR(25),
    FirstName VARCHAR(25),
    RegionID INT,
    DateOfHire DATE,
    FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
);
  
```

```

CREATE TABLE Skill (
    SkillID VARCHAR(20) PRIMARY KEY,
    SkillDescription VARCHAR(150),
    RateOfPay DECIMAL(10, 2)
);

CREATE TABLE EmpSkill (
    EmpSkillID VARCHAR(20) PRIMARY KEY,
    EmployeeID VARCHAR(20),
    SkillID varchar(20)
);

ALTER TABLE EmpSkill
ADD CONSTRAINT FK_EmpSkill_Employee
FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID),
ADD CONSTRAINT FK_EmpSkill_Skill
FOREIGN KEY (SkillID) REFERENCES Skill(SkillID);

ALTER TABLE Customer
ADD CONSTRAINT FK_Customer_Region
FOREIGN KEY (RegionID)
REFERENCES Region(RegionID);

INSERT INTO Region (RegionID, RegionName)
VALUES
    (1001, 'Northwest'),
    (1002, 'Southwest'),
    (1003, 'Midwest North'),
    (1004, 'Midwest South'),
    (1005, 'Northeast');

INSERT INTO Customer (CustomerID, Name, PhoneNumber, RegionID)
VALUES
    (1, 'Bellsouth', '222-333-4571', 1003),
    (2, 'Comcast', '253-444-5555', 1003),
    (3, 'Enron', '367-555-6666', 1005),
    (4, 'Exxon', '444-777-7777', 1004);

INSERT INTO Employee (EmployeeID, LastName, FirstName, RegionID, DateOfHire)
VALUES
    ('E1', 'Maru', 'Hardik', 1004, '2019-2-7'),
    ('E2', 'Craig', 'Brett', 1004, '2019-3-30'),
    ('E3', 'Williams', 'Josh', 1005, '1999-3-17'),
    ('E4', 'Cope', 'Leslie', 1002, '2017-4-21'),
    ('E5', 'Mudd', 'Roger', 1001, '2007-10-18');

alter table Employee
    modify EmployeeID VARCHAR(10) null;

INSERT INTO Skill (SkillID, SkillDescription, RateOfPay)
VALUES
    ('S1', 'Data Entry I', 12.00),
    ('S2', 'Java I', 25.00),
    ('S3', 'Python I', 25.00),

```

```

        ('S4', 'Python II', 35.00);
INSERT INTO EmpSkill (EmployeeID, SkillID)
VALUES
    ("E1", "S1"),
    ("E2", "S1"),
    ("E3", "S2"),
    ("E3", "S4"),
    ("E4", "S3");

SELECT
    FORMAT(AVG(RateOfPay), 2) AS AverageSkillRate,
    FORMAT(MAX(RateOfPay), 2) AS MaximumSkillRate,
    FORMAT(MIN(RateOfPay), 2) AS MinimumSkillRate
FROM
    Skill;

SELECT Customer.Name
FROM Customer
INNER JOIN Region ON Customer.RegionID = Region.RegionID
WHERE Region.RegionName = 'Northeast';

INSERT INTO EmpSkill (EmployeeID, SkillID)
VALUES
    ("E1", "S1"),
    ("E2", "S1"),
    ("E3", "S2"),
    ("E3", "S4"),
    ("E4", "S3");

select * from EmpSkill;

SELECT DISTINCT EmployeeID
FROM EmpSkill
WHERE SkillID IN (
    SELECT SkillID
    FROM Skill
    WHERE RateOfPay > 15
);

CREATE VIEW EmployeeSkillsView AS
SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID
FROM Employee E
INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;

SELECT * FROM EmployeeSkillsView;

```

```
select * from Customer;
```

```
select * from Employee;
```

3. Based on the ERD, write and execute a SQL script to create a database and populate it with data based on the tables below. Make sure you clearly identify the primary and foreign keys in your SQL code. Use the attributes shown in the tables below. Data must be entered for all fields except for cusPhone, which can have null values.

```
Create Database projectDBfinal;
CREATE TABLE Customer (
    CustomerID INT PRIMARY KEY,
    Name VARCHAR(25),
    PhoneNumber VARCHAR(20),
    RegionID INT,
    FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
);

CREATE TABLE Region (
    RegionID INT PRIMARY KEY,
    RegionName VARCHAR(25)
);

CREATE TABLE Employee (
    EmployeeID VARCHAR(20) PRIMARY KEY,
    LastName VARCHAR(25),
    FirstName VARCHAR(25),
    RegionID INT,
    DateOfHire DATE,
    FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
);

CREATE TABLE Skill (
    SkillID VARCHAR(20) PRIMARY KEY,
    SkillDescription VARCHAR(150),
    RateOfPay DECIMAL(10, 2)
);

CREATE TABLE EmpSkill (
    EmpSkillID VARCHAR(20) PRIMARY KEY,
    EmployeeID VARCHAR(20),
    SkillID varchar(20)
);

ALTER TABLE EmpSkill
ADD CONSTRAINT FK_EmpSkill_Employee
FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID),
ADD CONSTRAINT FK_EmpSkill_Skill
FOREIGN KEY (SkillID) REFERENCES Skill(SkillID);

ALTER TABLE Customer
ADD CONSTRAINT FK_Customer_Region
```

```
FOREIGN KEY (RegionID)
REFERENCES Region (RegionID);
```

The screenshot shows a SQL IDE interface. The main editor displays a stored procedure named `Stored procedure.sql` with the following SQL code:

```
112 WHERE SkillID IN (
113     SELECT SkillID
114     FROM Skill
115     WHERE RateOfPay > 15
116 );
117
118
119 CREATE VIEW EmployeeSkillsView AS
120 SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID
121 FROM Employee E
122 INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;
123
124 SELECT * FROM EmployeeSkillsView;
125
126 select * from EmpSkill;
```

The output window shows the results of the query `select * from EmpSkill;` with 5 rows:

EmployeeID	SkillID
E1	S1
E2	S1
E3	S2
E3	S4
E4	S3

The screenshot shows a SQL IDE interface. The main editor displays a stored procedure named `Stored procedure.sql` with the following SQL code:

```
112 WHERE SkillID IN (
113     SELECT SkillID
114     FROM Skill
115     WHERE RateOfPay > 15
116 );
117
118
119 CREATE VIEW EmployeeSkillsView AS
120 SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID
121 FROM Employee E
122 INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;
123
124 SELECT * FROM EmployeeSkillsView;
125
126 select * from EmpSkill;
127 select * from Skill;
```

The output window shows the results of the query `select * from Skill;` with 4 rows:

SkillID	SkillDescription	RateOfPay
S1	Data Entry I	12.00
S2	Java I	25.00
S3	Python I	25.00
S4	Python II	35.00

SQL Editor interface showing a stored procedure named `EmployeeSkillsView` being executed. The procedure contains the following SQL code:

```
CREATE VIEW EmployeeSkillsView AS
SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID
FROM Employee E
INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;

SELECT * FROM EmployeeSkillsView;

select * from EmpSkill;

select * from Skill;

select * from Employee;
```

The output window displays the results of the `select * from Employee;` query, showing 5 rows of employee data:

EmployeeID	LastName	FirstName	RegionID	DateOfHire
1	E1	Maru	Hardik	1004 2019-02-07
2	E2	Craig	Brett	1004 2019-03-30
3	E3	Williams	Josh	1005 1999-03-17
4	E4	Cope	Leslie	1002 2017-04-21
5	E5	Mudd	Roger	1001 2007-10-18

SQL Editor interface showing a stored procedure named `Region` being executed. The procedure contains the following SQL code:

```
CREATE VIEW EmployeeSkillsView AS
SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID
FROM Employee E
INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;

SELECT * FROM EmployeeSkillsView;

select * from EmpSkill;

select * from Skill;

select * from Employee;

select * from Customer;

select * from Region;
```

The output window displays the results of the `select * from Region;` query, showing 5 rows of region data:

RegionID	RegionName
1	1001 Northwest
2	1002 Southwest
3	1003 Northeast
4	1004 Southeast
5	1005 Central

SQL Editor Interface showing a database schema and a query execution result.

Database Schema (Left Panel):

- GCSDDB
 - tables
 - Customer
 - Employee
 - EmpSkill
 - Region
 - Skill
 - views
 - EmployeeSkillsView
- information_schema
- mysql
- performance_schema
- sys
- Server Objects

SQL Query (Main Editor):

```
117  
118  
119 CREATE VIEW EmployeeSkillsView AS  
120 SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID  
121 FROM Employee E  
122 INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;  
123  
124 SELECT * FROM EmployeeSkillsView;  
125  
126 select * from EmpSkill;  
127  
128 select * from Skill;  
129  
130 select * from Employee;  
131  
132 select * from Customer;
```

Query Execution Result (Bottom Panel):

Output: GCSDDB.Customer

CustomerID	Name	PhoneNumber	RegionID
1	Bellsouth	222-333-4571	1003
2	Comcast	253-444-5555	1003
3	Enron	367-555-6666	1005
4	Exxon	444-777-7777	1004

SQL > Launchpad procedure.sql 132:1 (23 chars) LF UTF-8 4 spa

- a. Write a query to display average, maximum and minimum skill rate.

The result of the query should be:

24.25	35	12
-------	----	----

Copy and paste script as well as result table here.

The screenshot shows a SQL IDE with a query editor and a results pane. The query editor contains the following SQL script:

```
71  
72  
73 DELIMITER ;  
74  
75 -- To execute the stored procedure, use the following statement:  
76 -- CALL GCSDB_Initialization();  
77  
78 SELECT  
79     FORMAT(AVG(RateOfPay), 2) AS AverageSkillRate,  
80     FORMAT(MAX(RateOfPay), 2) AS MaximumSkillRate,  
81     FORMAT(MIN(RateOfPay), 2) AS MinimumSkillRate  
82 FROM  
83     Skill;
```

The results pane shows the output of the query, which is a single row with three columns: AverageSkillRate, MaximumSkillRate, and MinimumSkillRate. The values are 24.25, 35.00, and 12.00 respectively.

AverageSkillRate	MaximumSkillRate	MinimumSkillRate
24.25	35.00	12.00

- b. Write a query to display the names of all customers in the region named Northeast. You must use a JOIN.

The result of the query should be:

BellSouth

ComCast

Copy and paste script as well as result table here.

The screenshot shows a SQL IDE interface. The main editor displays a SQL script with the following content:

```
DELIMITER ;
-- To execute the stored procedure, use the following statement:
-- CALL GCSDb_Initialization();
SELECT
    FORMAT(AVG(RateOfPay), 2) AS AverageSkillRate,
    FORMAT(MAX(RateOfPay), 2) AS MaximumSkillRate,
    FORMAT(MIN(RateOfPay), 2) AS MinimumSkillRate
FROM
    Skill;

SELECT Customer.Name
FROM Customer
INNER JOIN Region ON Customer.RegionID = Region.RegionID
WHERE Region.RegionName = 'Northeast';
```

The bottom panel shows the 'Output' window for the query 'GCSDb.Customer'. It displays 2 rows of results:

Name
Bellsouth
Comcast

c. Write a query to display employee ID of employees who have skills with that pay more than \$15 per hour. You must use a subquery.

The result of the query should be:

E3
E4

HINT: Use the subquery to get a list of skills that pay \$15/hour (i.e. `SELECT skillID FROM skill WHERE skillRate > 15`). Then use main query to find employees whose skill matches one of those in the list.

Copy and paste script as well as result table here.

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the server hierarchy for 'localhost' with the 'Server Objects' folder expanded. The central pane shows a SQL query script with line numbers 101 to 116. The query is as follows:

```
101 Skill;
102
103
104
105 SELECT Customer.Name
106 FROM Customer
107 INNER JOIN Region ON Customer.RegionID = Region.RegionID
108 WHERE Region.RegionName = 'Northeast';
109
110 SELECT DISTINCT EmployeeID
111 FROM EmpSkill
112 WHERE SkillID IN (
113     SELECT SkillID
114     FROM Skill
115     WHERE RateOfPay > 15
116 );
```

The right pane shows the 'Output' window for the query 'GCSDb.EmpSkill'. It displays 2 rows of results:

EmployeeID
E3
E4

d. Write a query to create view that contains employee id, employee last name, employee first name and skill ID for each employee. After the view is created. Use a SELECT command to display the view.

The result of the SELECT statement should be:

EmployeeID	LastName	FirstName	SkillID
E1	(your last name)	(your first name)	S1
E2	Craig	Brett	S1
E3	Williams	Josh	S2
E3	Williams	Josh	S4
E4	Cope	Leslie	S3

Copy and paste script as well as result table here.

The screenshot shows a SQL IDE with a script editor and an output window. The script editor contains the following SQL code:

```
111 FROM EmployeeSkillsView
112 WHERE SkillID IN (
113     SELECT SkillID
114     FROM Skill
115     WHERE RateOfPay > 15
116 );
117
118
119 CREATE VIEW EmployeeSkillsView AS
120 SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID
121 FROM Employee E
122 INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;
123
124 SELECT * FROM EmployeeSkillsView;
```

The output window shows the results of the SELECT statement, displaying 5 rows of data:

EmployeeID	LastName	FirstName	SkillID
1 E1	Maru	Hardik	S1
2 E2	Craig	Brett	S1
3 E3	Williams	Josh	S2
4 E3	Williams	Josh	S4
5 E4	Cope	Leslie	S3

Here I did additional work, you can just run one stored procedure and entire project would be executed as of requirement given for the project. I would like to shares a complimentary work executed by me.

```
DELIMITER $$
```

```

CREATE PROCEDURE DB1_Initialization()
BEGIN
    -- Create tables
    CREATE TABLE IF NOT EXISTS Region (
        RegionID INT PRIMARY KEY,
        RegionName VARCHAR(25)
    );

    CREATE TABLE IF NOT EXISTS Customer (
        CustomerID INT PRIMARY KEY,
        Name VARCHAR(25),
        PhoneNumber VARCHAR(20),
        RegionID INT,
        FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
    );

    CREATE TABLE IF NOT EXISTS Employee (
        EmployeeID VARCHAR(20) PRIMARY KEY,
        LastName VARCHAR(25),
        FirstName VARCHAR(25),
        RegionID INT,
        DateOfHire DATE,
        FOREIGN KEY (RegionID) REFERENCES Region(RegionID)
    );

    CREATE TABLE IF NOT EXISTS Skill (
        SkillID VARCHAR(20) PRIMARY KEY,
        SkillDescription VARCHAR(150),
        RateOfPay DECIMAL(10, 2)
    );

    CREATE TABLE IF NOT EXISTS EmpSkill (
        EmpSkillID VARCHAR(20) PRIMARY KEY,
        EmployeeID VARCHAR(20),
        SkillID VARCHAR(20),
        FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID),
        FOREIGN KEY (SkillID) REFERENCES Skill(SkillID)
    );

    -- Alter table
    ALTER TABLE Employee
    MODIFY EmployeeID VARCHAR(10);

    -- Insert data
    INSERT INTO Region (RegionID, RegionName)
    VALUES
        (1001, 'Northwest'),
        (1002, 'Southwest'),
        (1003, 'Midwest North'),
        (1004, 'Midwest South'),
        (1005, 'Northeast');

    INSERT INTO Customer (CustomerID, Name, PhoneNumber, RegionID)
    VALUES
        (1, 'Bellsouth', '222-333-4571', 1003),
        (2, 'Comcast', '253-444-5555', 1003),
        (3, 'Enron', '367-555-6666', 1005),

```

```

        (4, 'Exxon', '444-777-7777', 1004);

INSERT INTO Employee (EmployeeID, LastName, FirstName, RegionID,
DateOfHire)
VALUES
    ('E1', 'Maru', 'Hardik', 1004, '2019-2-7'),
    ('E2', 'Craig', 'Brett', 1004, '2019-3-30'),
    ('E3', 'Williams', 'Josh', 1005, '1999-3-17'),
    ('E4', 'Cope', 'Leslie', 1002, '2017-4-21'),
    ('E5', 'Mudd', 'Roger', 1001, '2007-10-18');

INSERT INTO Skill (SkillID, SkillDescription, RateOfPay)
VALUES
    ('s1', 'Data Entry I', 12.00),
    ('s2', 'Java I', 25.00),
    ('s3', 'Python I', 25.00),
    ('s4', 'Python II', 35.00);

INSERT INTO EmpSkill (EmployeeID, SkillID)
VALUES
    ('E1', 's1'),
    ('E2', 's1'),
    ('E3', 's2'),
    ('E3', 's4'),
    ('E4', 's3');

-- Create a view
CREATE OR REPLACE VIEW EmployeeSkillsView AS
SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID
FROM Employee E
INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;

END$$

DELIMITER ;

-- To execute the stored procedure, use the following statement:
-- CALL DB1_Initialization();
SELECT
    FORMAT(AVG(RateOfPay), 2) AS AverageSkillRate,
    FORMAT(MAX(RateOfPay), 2) AS MaximumSkillRate,
    FORMAT(MIN(RateOfPay), 2) AS MinimumSkillRate
FROM
    Skill;

SELECT Customer.Name
FROM Customer
INNER JOIN Region ON Customer.RegionID = Region.RegionID
WHERE Region.RegionName = 'Northeast';

SELECT DISTINCT EmployeeID
FROM EmpSkill
WHERE SkillID IN (
    SELECT SkillID
    FROM Skill
    WHERE RateOfPay > 15

```

```
);  
  
CREATE VIEW EmployeeSkillsView AS  
SELECT E.EmployeeID, E.LastName, E.FirstName, ES.SkillID  
FROM Employee E  
INNER JOIN EmpSkill ES ON E.EmployeeID = ES.EmployeeID;  
  
SELECT * FROM EmployeeSkillsView;  
  
select * from EmpSkill;  
  
select * from Skill;  
  
select * from Employee;  
  
select * from Customer;  
  
select * from Region;
```

Thank you
Hardik Maru