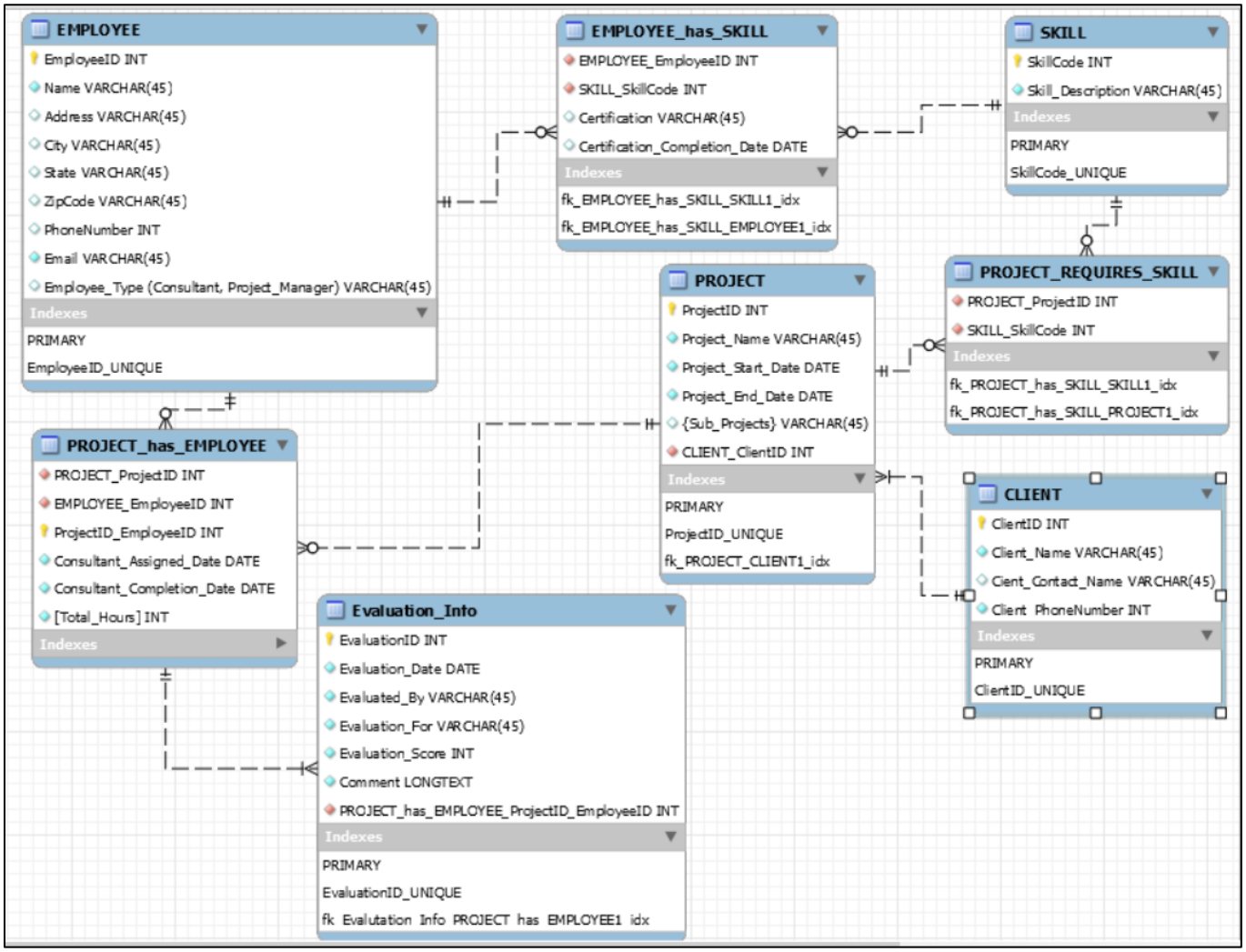
Q2. Logical design of the database: E-R Diagram (Part – I)



Q3. SQL codes for creating all tables (with all PK and FK constraints specified).

There are 8 tables:

i. Employee table
ii. Skill table
iii. Employee_has_skill table
iv. Client table
v. Project table
vi. Project_requires_skill table
vii. Project_has_employee table
viii. Evaluation_Info table

i. SQL Code for Creation of Employee Table

Query:

```
CREATE TABLE EMPLOYEE (
  EmployeeID INT NOT NULL,
  Name VARCHAR(45) NOT NULL,
  Address VARCHAR(45),
  City VARCHAR(45),
  State VARCHAR(45),
  ZipCode VARCHAR(45),
  PhoneNumber INT,
  Email VARCHAR(45) NOT NULL,
  Employee_Type VARCHAR(45),
  CONSTRAINT Employee_PK PRIMARY KEY (EmployeeID));
```

ii. SQL Code for Creation of Skill Table

Query:

```
CREATE TABLE SKILL (
   SkillCode INT NOT NULL,
   Skill_Description VARCHAR(45) NOT NULL,
   CONSTRAINT Skill_PK PRIMARY KEY (SkillCode));
```

```
-- TABLE SKILL
CREATE TABLE SKILL (
    SkillCode INT NOT NULL,
    Skill_Description VARCHAR(45) NOT NULL,
    CONSTRAINT Skill_PK PRIMARY KEY (SkillCode));
```

Results | Explain | Describe | Saved SQL | History

```
Table created.

0.08 seconds
```

iii. SQL Code for Creation of Employee_has_Skill Table

Query:

```
CREATE TABLE EMPLOYEE_has_SKILL (
  EmployeeID INT NOT NULL,
  SkillCode INT NOT NULL,
  Certification VARCHAR(45),
  Certification_Completion_Date DATE,
 CONSTRAINT FK_EMPLOYEE_has_SKILL_EMPLOYEE1 FOREIGN KEY (EmployeeID)
  REFERENCES EMPLOYEE(EmployeeID),
   CONSTRAINT FK_EMPLOYEE_has_SKILL_SKILL1 FOREIGN KEY (SkillCode)
  REFERENCES SKILL (SkillCode));
```

```
-- Table EMPLOYEE_has_SKILL
CREATE TABLE EMPLOYEE_has_SKILL (
   EmployeeID INT NOT NULL,
   SkillCode INT NOT NULL,
   Certification VARCHAR(45),
   Certification_Completion_Date DATE,
  CONSTRAINT FK_EMPLOYEE_has_SKILL_EMPLOYEE1 FOREIGN KEY (EmployeeID)
    REFERENCES EMPLOYEE(EmployeeID),
    CONSTRAINT FK_EMPLOYEE_has_SKILL_SKILL1 FOREIGN KEY (SkillCode)
    REFERENCES SKILL (SkillCode));
```

Results | Explain | Describe | Saved SQL | History

```
Table created.

0.07 seconds
```

iv. SQL Code for Creation of Client table

Query:

```
CREATE TABLE CLIENT (
  ClientID INT NOT NULL,
  Client_Name VARCHAR(45) NOT NULL,
  Cient_Contact_Name VARCHAR(45) NULL,
  Client_PhoneNumber INT NOT NULL,
  CONSTRAINT Client_PK PRIMARY KEY (ClientID));
```

v. SQL Code for Creation of Project table

Query:

```
CREATE TABLE PROJECT (
  ProjectID INT NOT NULL,
  Project_Name VARCHAR(45) NOT NULL,
  Project_Start_Date DATE NOT NULL,
  Project_End_Date DATE NOT NULL,
  Sub_Projects VARCHAR(45) NULL,
  ClientID INT NOT NULL,
 CONSTRAINT PROJECT_PK PRIMARY KEY (ProjectID),
 CONSTRAINT FK_PROJECT_CLIENT1 FOREIGN KEY (ClientID)
  REFERENCES CLIENT(ClientID));
```

```
--TABLE PROJECT
CREATE TABLE PROJECT (
   ProjectID INT NOT NULL,
   Project_Name VARCHAR(45) NOT NULL,
   Project_Start_Date DATE NOT NULL,
   Project_End_Date DATE NOT NULL,
   Sub_Projects VARCHAR(45) NULL,
   ClientID INT NOT NULL,
  CONSTRAINT PROJECT_PK PRIMARY KEY (ProjectID),
  CONSTRAINT FK_PROJECT_CLIENT1 FOREIGN KEY (ClientID)
    REFERENCES CLIENT(ClientID));
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

0.11 seconds

vi. SQL Code for Project_requires_skill Table

Query:

```
CREATE TABLE PROJECT_REQUIRES_SKILL (
  ProjectID INT NOT NULL,
  SkillCode INT NOT NULL,
  CONSTRAINT FK_PROJECT_has_SKILL_PROJECT1 FOREIGN KEY (ProjectID)
   REFERENCES PROJECT(ProjectID),
 CONSTRAINT FK_PROJECT_has_SKILL_SKILL1 FOREIGN KEY (SkillCode)
   REFERENCES SKILL (SkillCode));
```

```
-- Table PROJECT_REQUIRES_SKILL
CREATE TABLE PROJECT_REQUIRES_SKILL (
   ProjectID INT NOT NULL,
   SkillCode INT NOT NULL,
   CONSTRAINT FK_PROJECT_has_SKILL_PROJECT1 FOREIGN KEY (ProjectID)
    REFERENCES PROJECT(ProjectID),
  CONSTRAINT FK_PROJECT_has_SKILL_SKILL1 FOREIGN KEY (SkillCode)
    REFERENCES SKILL (SkillCode));
```

Results | Explain | Describe | Saved SQL | History

```
Table created.
```

0.10 seconds

vii. SQL Code for Project_has_employee Table

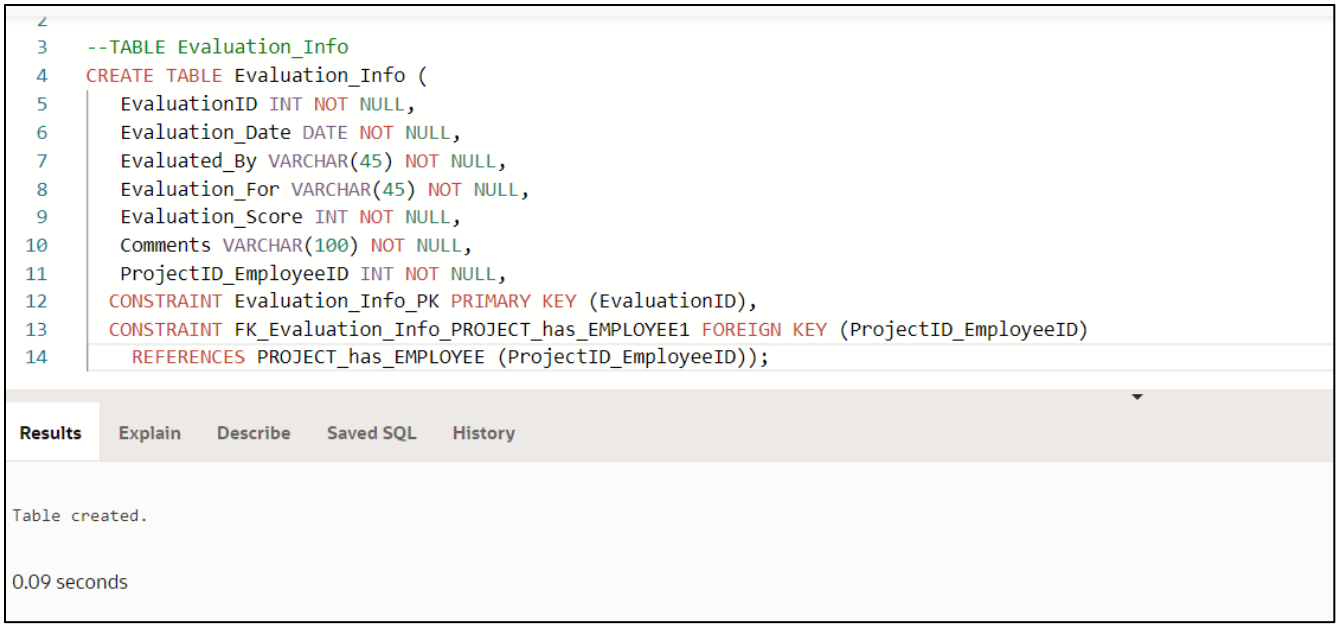
Query:

```
CREATE TABLE PROJECT_has_EMPLOYEE (
  ProjectID INT NOT NULL,
  EmployeeID INT NOT NULL,
  ProjectID_EmployeeID INT NOT NULL,
  Consultant_Assigned_Date DATE NOT NULL,
  Consultant_Completion_Date DATE NOT NULL,
  Total_Hours INT as ((to_date(Consultant_Completion_Date, 'MM/dd/yyyy') - to_date(Consultant_Assig
ned_Date, 'MM/dd/yyyy'))*24) NOT NULL,
 CONSTRAINT Project_has_Employee_PK PRIMARY KEY (ProjectID_EmployeeID),
 CONSTRAINT FK_PROJECT_has_EMPLOYEE_PROJECT1 FOREIGN KEY (ProjectID)
  REFERENCES PROJECT (ProjectID),
 CONSTRAINT FK_PROJECT_has_EMPLOYEE_EMPLOYEE1 FOREIGN KEY (EmployeeID)
  REFERENCES EMPLOYEE (EmployeeID));
```

viii. SQL Code for Creation of Evaluation_Info

Query:

```
CREATE TABLE Evaluation_Info (
  EvaluationID INT NOT NULL,
  Evaluation_Date DATE NOT NULL,
  Evaluated_By VARCHAR(45) NOT NULL,
  Evaluation_For VARCHAR(45) NOT NULL,
  Evaluation_Score INT NOT NULL,
  Comments VARCHAR(100) NOT NULL,
  ProjectID_EmployeeID INT NOT NULL,
 CONSTRAINT Evaluation_Info_PK PRIMARY KEY (EvaluationID),
 CONSTRAINT FK_Evaluation_Info_PROJECT_has_EMPLOYEE1 FOREIGN KEY (ProjectID_EmployeeID)
  REFERENCES PROJECT_has_EMPLOYEE (ProjectID_EmployeeID));
```



Q4. SQL codes for inserting at least 5 rows of data into each table (You must insert data for Consultant ID 100, Mark Meyers. You must also insert data for SKILL ID 1, 2, and 3).

i. SQL code for inserting at least 5 rows of data into employee table

Query:

```
Insert into Employee(EmployeeID,Name,Address,City,State,ZipCode,PhoneNumber,Email,Employee_Type )
 values (100,'Mark Meyers','123 Main Street','Plano','Texas','75074',5151234567,'mark.meyers@gmail.com','Consultant');

 Insert into Employee(EmployeeID, Name, Address, City, State, ZipCode, PhoneNumber, Email, Employee_Type )
 values (101, 'Bruce Ernst', '234 State Street', 'Dallas', 'Texas', '75201', 5151237568, 'bruce.ernst@gmail.com', 'Consultant');

 Insert into Employee(EmployeeID, Name, Address, City, State, ZipCode, PhoneNumber, Email, Employee_Type )
 values (102, 'Liza Ozer', '112 Allen Street', 'Frisco', 'Texas', '75033', 5904234567, 'liza.ozer@gmail.com', 'Consultant');

 Insert into Employee(EmployeeID, Name, Address, City, State, ZipCode, PhoneNumber, Email, Employee_Type )
 values (103, 'William Smith', '750 Ave C', 'Denton', 'Texas', '76201', 6501231224, 'william.smith@gmail.com', 'Consultant');

 Insert into Employee(EmployeeID, Name, Address, City, State, ZipCode, PhoneNumber, Email, Employee_Type )
 values (104, 'Laura Bissot', '220 Bryce Ave', 'Frisco', 'Texas', '75035', 6505071876, 'laura.bissot@gmail.com', 'Project Manager');


SELECT *
  FROM Employee;
```

Result:

Results | Explain | Describe | Saved SQL | History

EMPLOYEEID	NAME	ADDRESS	CITY	STATE	ZIPCODE	PHONENUMBER	EMAIL	EMPLOYEE_TYPE
100	Mark Meyers	123 Main Street	Plano	Texas	75074	5151234567	mark.meyers@gmail.com	Consultant
102	Liza Ozer	112 Allen Street	Frisco	Texas	75033	5904234567	liza.ozer@gmail.com	Consultant
103	William Smith	750 Ave C	Denton	Texas	76201	6501231224	william.smith@gmail.com	Consultant
101	Bruce Ernst	234 State Street	Dallas	Texas	75201	5151237568	bruce.ernst@gmail.com	Consultant
104	Laura Bissot	220 Bryce Ave	Frisco	Texas	75035	6505071876	laura.bissot@gmail.com	Project Manager

5 rows returned in 0.00 seconds Download

ii. SQL code for inserting at least 5 rows of data into skill table

Query:

```
Insert into Skill(SkillCode, Skill_Description) values (1, 'Microsoft Office');
 Insert into Skill(SkillCode, Skill_Description) values (2, 'Cloud Datawarehouse');
 Insert into Skill(SkillCode, Skill_Description) values (3, 'Web analytics service like Google Analytics');
 Insert into Skill(SkillCode, Skill_Description) values (4, 'Project Management');
 Insert into Skill(SkillCode, Skill_Description) values (5, 'Financial analysis and planning');


SELECT *
  FROM Skill;
```

Result:

Results | Explain | Describe | Saved SQL | History

SKILLCODE	SKILL_DESCRIPTION
4	Project Management
2	Cloud Datawarehouse
1	Microsoft Office
3	Web analytics service like Google Analytics
5	Financial analysis and planning

5 rows returned in 0.03 seconds Download

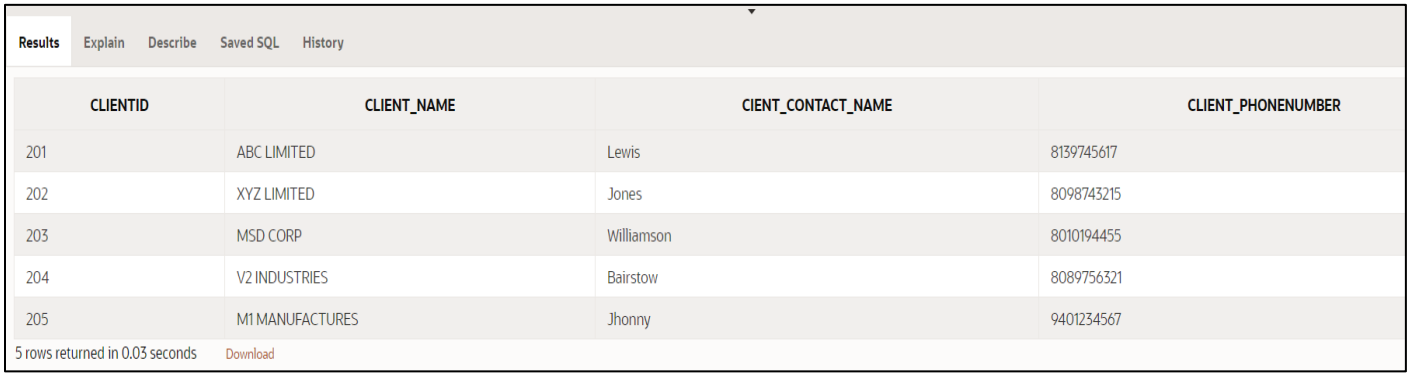
iii. SQL code for inserting at least 5 rows of data into client table

Query:

```
INSERT INTO client (CLIENTID,CLIENT_NAME,CIENT_CONTACT_NAME,CLIENT_PHONENUMBER) values (2
01,'ABC LIMITED', 'Lewis','8139745617');
INSERT INTO client (CLIENTID,CLIENT_NAME,CIENT_CONTACT_NAME,CLIENT_PHONENUMBER) values (2
02,'XYZ LIMITED', 'Jones','8098743215');
INSERT INTO client (CLIENTID,CLIENT_NAME,CIENT_CONTACT_NAME,CLIENT_PHONENUMBER) values (2
03,'MSD CORP', 'Williamson','8010194455');
INSERT INTO client (CLIENTID,CLIENT_NAME,CIENT_CONTACT_NAME,CLIENT_PHONENUMBER) values (2
04,'V2 INDUSTRIES', 'Bairstow','8089756321');
INSERT INTO client (CLIENTID,CLIENT_NAME,CIENT_CONTACT_NAME,CLIENT_PHONENUMBER) values (2
05,'M1 MANUFACTURES', 'Jhonny','9401234567');


SELECT *
  FROM client
    ORDER BY CLIENTID;
```

Result:



Results | Explain | Describe | Saved SQL | History

CLIENTID	CLIENT_NAME	CIENT_CONTACT_NAME	CLIENT_PHONENUMBER
201	ABC LIMITED	Lewis	8139745617
202	XYZ LIMITED	Jones	8098743215
203	MSD CORP	Williamson	8010194455
204	V2 INDUSTRIES	Bairstow	8089756321
205	M1 MANUFACTURES	Jhonny	9401234567

5 rows returned in 0.03 seconds Download

iv. SQL code for inserting at least 5 rows of data into project table

Query:

```
INSERT INTO project (ProjectID, Project_Name, Project_Start_Date, Project_End_Date, Sub_Projects, Cli
entID) VALUES ('11', 'DBA Project', '09/25/2020', '03/30/2021', 'Sb1, Sb2', '201');
INSERT INTO project (ProjectID, Project_Name, Project_Start_Date, Project_End_Date, Sub_Projects, Cli
entID) VALUES ('12', 'Housing Project', '06/05/2020', '08/09/2020', 'Sb3', '202');
INSERT INTO project (ProjectID, Project_Name, Project_Start_Date, Project_End_Date, Sub_Projects, Cli
entID) VALUES ('13', 'Payment Processing Project', '07/03/2020', '12/09/2020', 'Sb4, Sb5', '203');
```

```
INSERT INTO project (ProjectID, Project_Name, Project_Start_Date, Project_End_Date, Sub_Projects, Cli
entID) VALUES ('14', 'ETL Billing Project', '05/04/2019', '01/02/2021', 'Sb6', '204');
INSERT INTO project (ProjectID, Project_Name, Project_Start_Date, Project_End_Date, Sub_Projects, Cli
entID) VALUES ('15', 'Philadelphia Project', '11/12/2019', '03/04/2020', 'Sb7, Sb8', '205');
```

```
SELECT *
   FROM Project;
```

Result:

Results | Explain | Describe | Saved SQL | History

PROJECTID	PROJECT_NAME	PROJECT_START_DATE	PROJECT_END_DATE	SUB_PROJECTS	CLIENTID
11	DBA Project	09/25/2020	03/30/2021	Sb1, Sb2	201
12	Housing Project	06/05/2020	08/09/2020	Sb3	202
13	Payment Processing Project	07/03/2020	12/09/2020	Sb4, Sb5	203
14	ETL Billing Project	05/04/2019	01/02/2021	Sb6	204
15	Philadelphia Project	11/12/2019	03/04/2020	Sb7, Sb8	205

5 rows returned in 0.01 seconds Download

v. SQL code for inserting at least 5 rows of data into project_requires_skill table

Query:

```
INSERT INTO PROJECT_REQUIRES_SKILL VALUES (11, 2);
INSERT INTO PROJECT_REQUIRES_SKILL VALUES (12, 3);
INSERT INTO PROJECT_REQUIRES_SKILL VALUES (13, 4);
INSERT INTO PROJECT_REQUIRES_SKILL VALUES (14, 1);
INSERT INTO PROJECT_REQUIRES_SKILL VALUES (15, 5);
```

```
SELECT *
    FROM Project_Requires_Skill;
```

Result:

Results | Explain | Describe | Saved SQL | History

PROJECTID	SKILLCODE
11	2
12	3
13	4
14	1
15	5

5 rows returned in 0.01 seconds Download

vi. SQL code for inserting at least 5 rows of data into project_has_employee table

Query:

```
INSERT INTO project_has_employee (ProjectID, EmployeeID, ProjectID_EmployeeID,
Consultant_Assigned_Date, Consultant_Completion_Date) VALUES (11, 100, 11100, '03/02/2018',
'03/03/2020');

INSERT INTO project_has_employee (ProjectID, EmployeeID, ProjectID_EmployeeID,
Consultant_Assigned_Date, Consultant_Completion_Date) VALUES (12, 101, 12101, '01/03/2019',
'01/05/2020');

INSERT INTO project_has_employee (ProjectID, EmployeeID, ProjectID_EmployeeID,
Consultant_Assigned_Date, Consultant_Completion_Date) VALUES (13, 102, 13102, '12/24/2020',
'12/27/2020');

INSERT INTO project_has_employee (ProjectID, EmployeeID, ProjectID_EmployeeID,
Consultant_Assigned_Date, Consultant_Completion_Date) VALUES (14, 103, 14103, '01/02/2020',
'01/03/2021');

INSERT INTO project_has_employee (ProjectID, EmployeeID, ProjectID_EmployeeID,
Consultant_Assigned_Date, Consultant_Completion_Date) VALUES (15, 104, 15104, '02/02/2021',
'02/05/2021');

INSERT INTO project_has_employee (ProjectID, EmployeeID, ProjectID_EmployeeID,
Consultant_Assigned_Date, Consultant_Completion_Date) VALUES (11, 101, 11101, '01/03/2019',
'01/05/2019');

INSERT INTO project_has_employee (ProjectID, EmployeeID, ProjectID_EmployeeID,
Consultant_Assigned_Date, Consultant_Completion_Date) VALUES (11, 102, 11102, '12/24/2020',
'12/25/2020');


SELECT *
  FROM project_has_employee;
```

Result:

Results | Explain | Describe | Saved SQL | History

PROJECTID	EMPLOYEEID	PROJECTID_EMPLOYEEID	CONSULTANT_ASSIGNED_DATE	CONSULTANT_COMPLETION_DATE	TOTAL_HOURS
11	102	11102	12/24/2020	12/25/2020	24
11	101	11101	01/03/2019	01/05/2019	48
11	100	11100	03/02/2018	03/03/2020	17568
12	101	12101	01/03/2019	01/05/2020	8808
13	102	13102	12/24/2020	12/27/2020	72
14	103	14103	01/02/2020	01/03/2021	8808
15	104	15104	02/02/2021	02/05/2021	72

7 rows returned in 0.02 seconds Download

vii. SQL code for inserting at least 5 rows of data into evaluation_info table

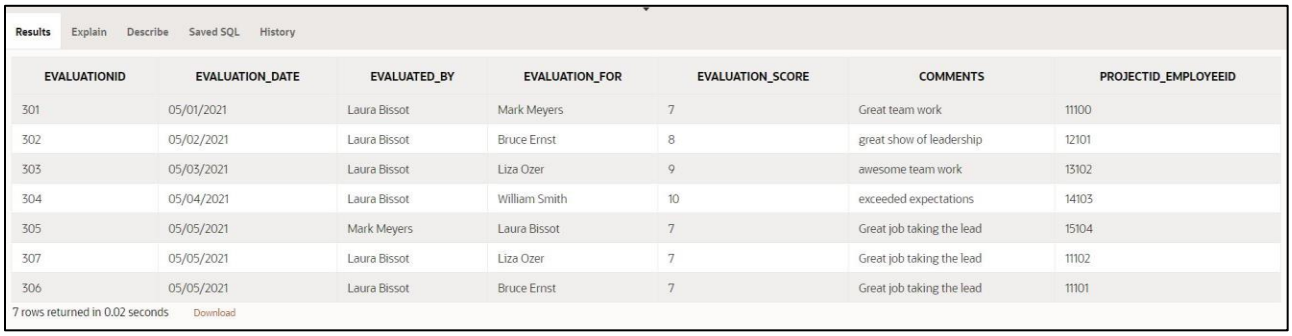
Query:

```
INSERT INTO EVALUATION_INFO VALUES (301, '05/01/2021', 'Laura Bissot', 'Mark Meyers', 7, 'Great team work', 11100);
INSERT INTO EVALUATION_INFO VALUES (302, '05/02/2021', 'Laura Bissot', 'Bruce Ernst', 8, 'great show off leadership', 12101);
INSERT INTO EVALUATION_INFO VALUES (303, '05/03/2021', 'Laura Bissot', 'Liza Ozer', 9, 'awesome team work', 13102);
INSERT INTO EVALUATION_INFO VALUES (304, '05/04/2021', 'Laura Bissot', 'William Smith', 10, 'exceeded expectations', 14103);
INSERT INTO EVALUATION_INFO VALUES (305, '05/05/2021', 'Mark Meyers', 'Laura Bissot', 7, 'Great job taking the lead', 15104);
INSERT INTO EVALUATION_INFO VALUES (306, '05/05/2021', 'Laura Bissot', 'Bruce Ernst', 7, 'Great job ta king the lead', 11101);

INSERT INTO EVALUATION_INFO VALUES (307, '05/05/2021', 'Laura Bissot', 'Liza Ozer', 7, 'Great job takin g the lead', 11102);

SELECT *
  FROM Evaluation_Info;
```

Result:



Results | Explain | Describe | Saved SQL | History

EVALUATIONID	EVALUATION_DATE	EVALUATED_BY	EVALUATION_FOR	EVALUATION_SCORE	COMMENTS	PROJECTID_EMPLOYEEID
301	05/01/2021	Laura Bissot	Mark Meyers	7	Great team work	11100
302	05/02/2021	Laura Bissot	Bruce Ernst	8	great show of leadership	12101
303	05/03/2021	Laura Bissot	Liza Ozer	9	awesome team work	13102
304	05/04/2021	Laura Bissot	William Smith	10	exceeded expectations	14103
305	05/05/2021	Mark Meyers	Laura Bissot	7	Great job taking the lead	15104
307	05/05/2021	Laura Bissot	Liza Ozer	7	Great job taking the lead	11102
306	05/05/2021	Laura Bissot	Bruce Ernst	7	Great job taking the lead	11101

7 rows returned in 0.02 seconds Download

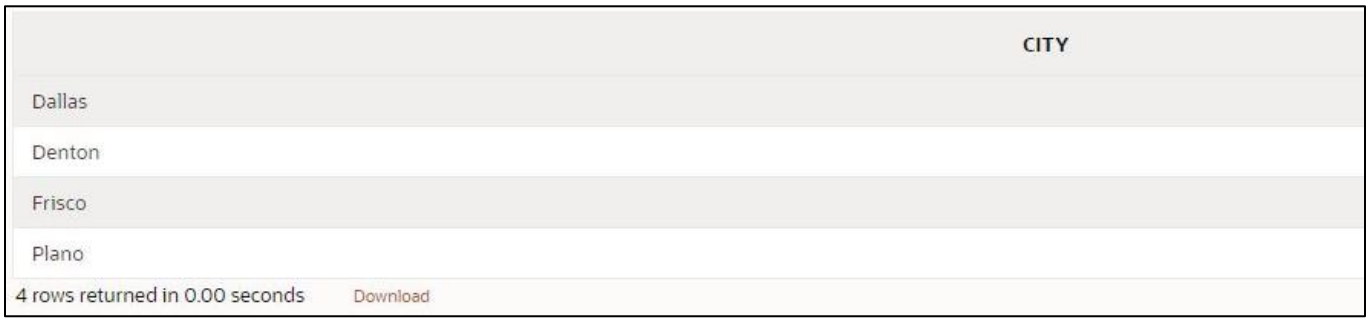
Q5. Create SQL queries for the following scenarios (even if you do not have data in your tables that will produce output, you must write the query correctly).

a. Retrieve the name of each city where a consultant lives. Suppress duplicate output and display the values in alphabetical order;

Query:

```
SELECT DISTINCT(city)
  FROM employee
    ORDER BY city;
```

Result:



CITY
Dallas
Denton
Frisco
Plano

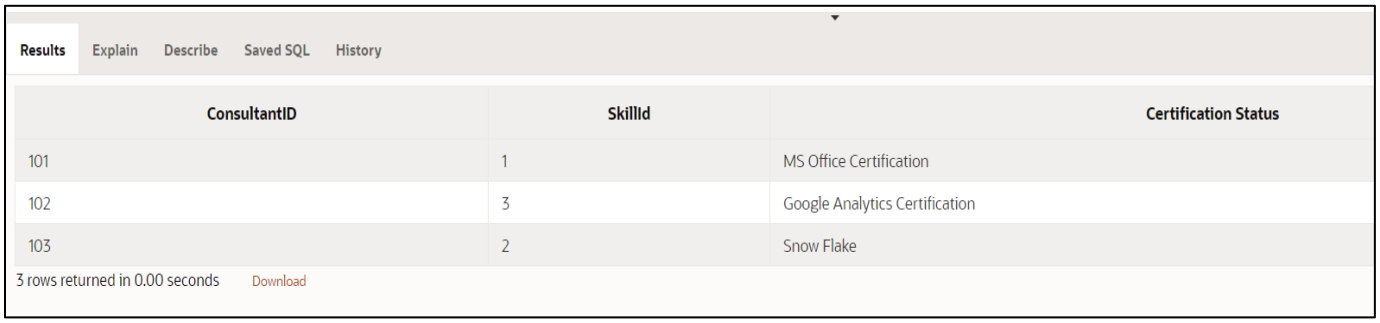
4 rows returned in 0.00 seconds Download

b. Retrieve the consultant id, skill id and certification status for every consultant who is proficient with skill ID 1, 2, or 3;

Query:

```
SELECT EmployeeID as "ConsultantID", SkillCode as "SkillId", Certification as "Certification Status"
  FROM Employee_has_skill
    WHERE SkillCode <4;
```

Result:



Results | Explain | Describe | Saved SQL | History

ConsultantID	SkillId	Certification Status
101	1	MS Office Certification
102	3	Google Analytics Certification
103	2	Snow Flake

3 rows returned in 0.00 seconds Download

c. Retrieve the first and last names of those consultants whose last names begin with an "M";

Query:

```
SELECT SUBSTR(name,1, INSTR(name,' ')-1) AS "First Name",
       SUBSTR(name, INSTR(name,' ')+1 )  AS "Last Name"
FROM employee
WHERE SUBSTR(name, INSTR(name,' ')+1 )  LIKE 'M%';
```

Result:

Results | Explain | Describe | Saved SQL | History

First Name	Last Name
Mark	Meyers

1 rows returned in 0.02 seconds Download

d. Retrieve the last name of each consultant and the name of each project they work on (it's ok to have duplicate consultant names);

Query:

```
SELECT SUBSTR(e.name, INSTR(e.name,' ')+1 ) as "Last Name", project_name as "Name of Project"
  FROM project p, Project_has_employee pe, employee e
    WHERE p.projectid = pe.projectid
      AND pe.employeeid = e.employeeid;
```

Result:

Results | Explain | Describe | Saved SQL | History

Last Name	Name of Project
Meyers	DBA Project
Bissot	Philadelphia Project
Ernst	DBA Project
Ernst	Housing Project
Ozer	DBA Project
Ozer	Payment Processing Project
Smith	ETL Billing Project

7 rows returned in 0.03 seconds Download

e. List the first and last name of every consultant who has worked with Mark Meyers (use a subquery);

Query:

```
SELECT SUBSTR(O_E.name,1, INSTR(O_E.name,' ')-
1) as "First Name", SUBSTR(O_E.name, INSTR(name,' ')+1 ) as "Last Name"
FROM Employee O_E
LEFT JOIN Project_has_Employee O_PHE ON O_E.EMPLOYEEID = O_PHE.EMPLOYEEID
WHERE O_PHE.PROJECTID IN (Select PROJECTID
                  From Employee E
                    Left join Project_has_Employee PHE
                      ON E.EMPLOYEEID = PHE.EMPLOYEEID
                        Where Trim(E.NAME) = 'Mark Meyers'
                                )
AND Trim (O_E.NAME) <> 'Mark Meyers';
```

Result:

Results | Explain | Describe | Saved SQL | History

First Name	Last Name
Bruce	Ernst
Liza	Ozer

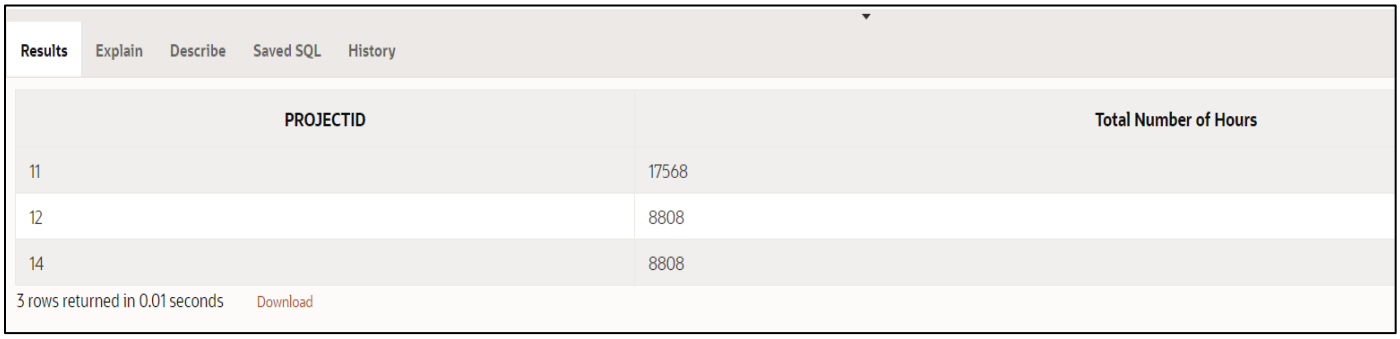
2 rows returned in 0.07 seconds Download

f. Display each project id and the total number of hours that all consultants have spent on that project (only display data for projects that have accumulated over 100 hours);

Query:

```
SELECT projectid, total_hours as "Total Number of Hours"
   FROM project_has_employee
      WHERE TOTAL_HOURS > 100;
```

Result:



Results | Explain | Describe | Saved SQL | History

PROJECTID	Total Number of Hours
11	17568
12	8808
14	8808

3 rows returned in 0.01 seconds Download

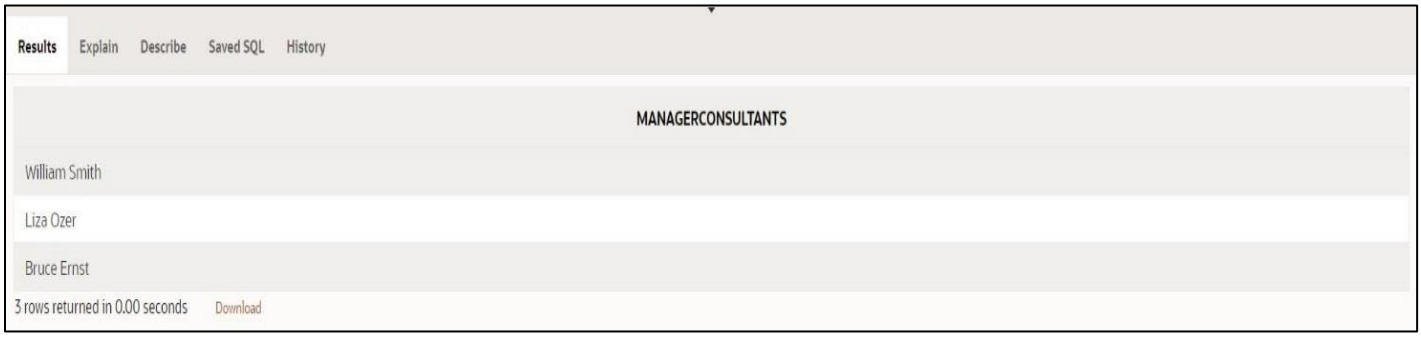
g. Develop your own query that uses an outer join;

(Get all consultants or managers who did not provide evaluation).

Query:

```
SELECT TRIM(E.Name) as ManagerConsultants
 FROM Evaluation_Info EI
FULL OUTER JOIN Employee E ON TRIM(E.Name)=TRIM(EI.EVALUATED_BY)
WHERE EVALUATED_BY is NULL;
```

Result:



Results | Explain | Describe | Saved SQL | History

MANAGERCONSULTANTS
William Smith
Liza Ozer
Bruce Ernst

3 rows returned in 0.00 seconds Download

h. Develop your own query that uses a group by with the having clause.

(Display the name of the evaluator who has evaluated more than 1 employee and the number of evaluations completed)

Query:

```
SELECT Evaluated_By as "Evaluator", COUNT(EvaluationID) as "Number of Evaluations Completed"
   FROM Evaluation_info
      GROUP BY Evaluated_By
         HAVING COUNT(EvaluationID) > 1;
```

Result:

Results | Explain | Describe | Saved SQL | History

Evaluator	Number of Evaluations Completed
Laura Bissot	6

1 rows returned in 0.00 seconds Download

i. Develop your own query that uses the LIKE keyword.

(Display name, certification, and certification_completion_date of all employees who have certification in Google Analytics).

Query:

```
SELECT name, certification, certification_completion_date
   FROM employee e, employee_has_skill es
      WHERE e.employeeid = es.employeeid
         AND certification LIKE '%Analytics%';
```

Result:

Results | Explain | Describe | Saved SQL | History

NAME	CERTIFICATION	CERTIFICATION_COMPLETION_DATE
Liza Ozer	Google Analytics Certification	11/30/2020

1 rows returned in 0.02 seconds Download

j. Develop your own query that uses the DISTINCT keyword.
(Display names of all the evaluators. Suppress duplicate output)

Query:

```
SELECT DISTINCT(evaluated_by) as "Evaluators"
  FROM evaluation_info;
```

Result: