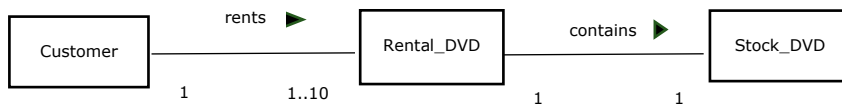
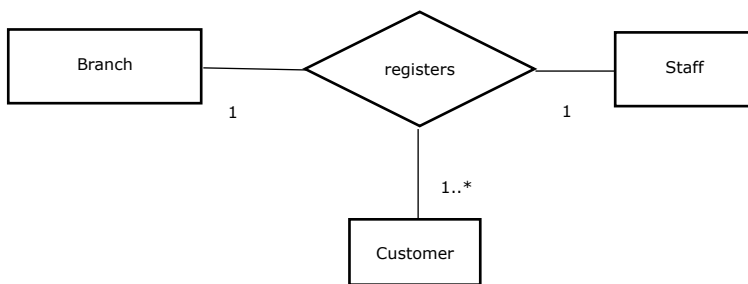
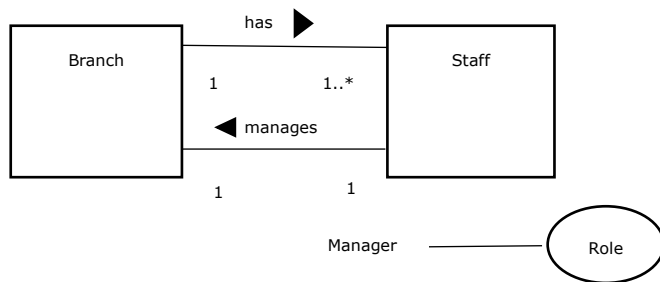
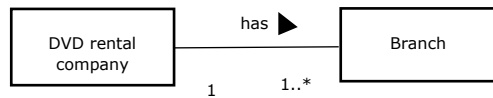


- Identify the main entity types of the DVD rental company
 - DVD rental company
 - Branch
 - Staff
 - Rental DVD
 - Stock DVD
- Identify the main relationship types between entity types and multiplicity constraints



➤ Determine candidate key and primary key

Branch: branchNo is the candidate key and primary key

Staff: staffNo is the candidate key and primary key

Customer: memberNo is the primary key

memberNo, firstName, lastName is the candidate key

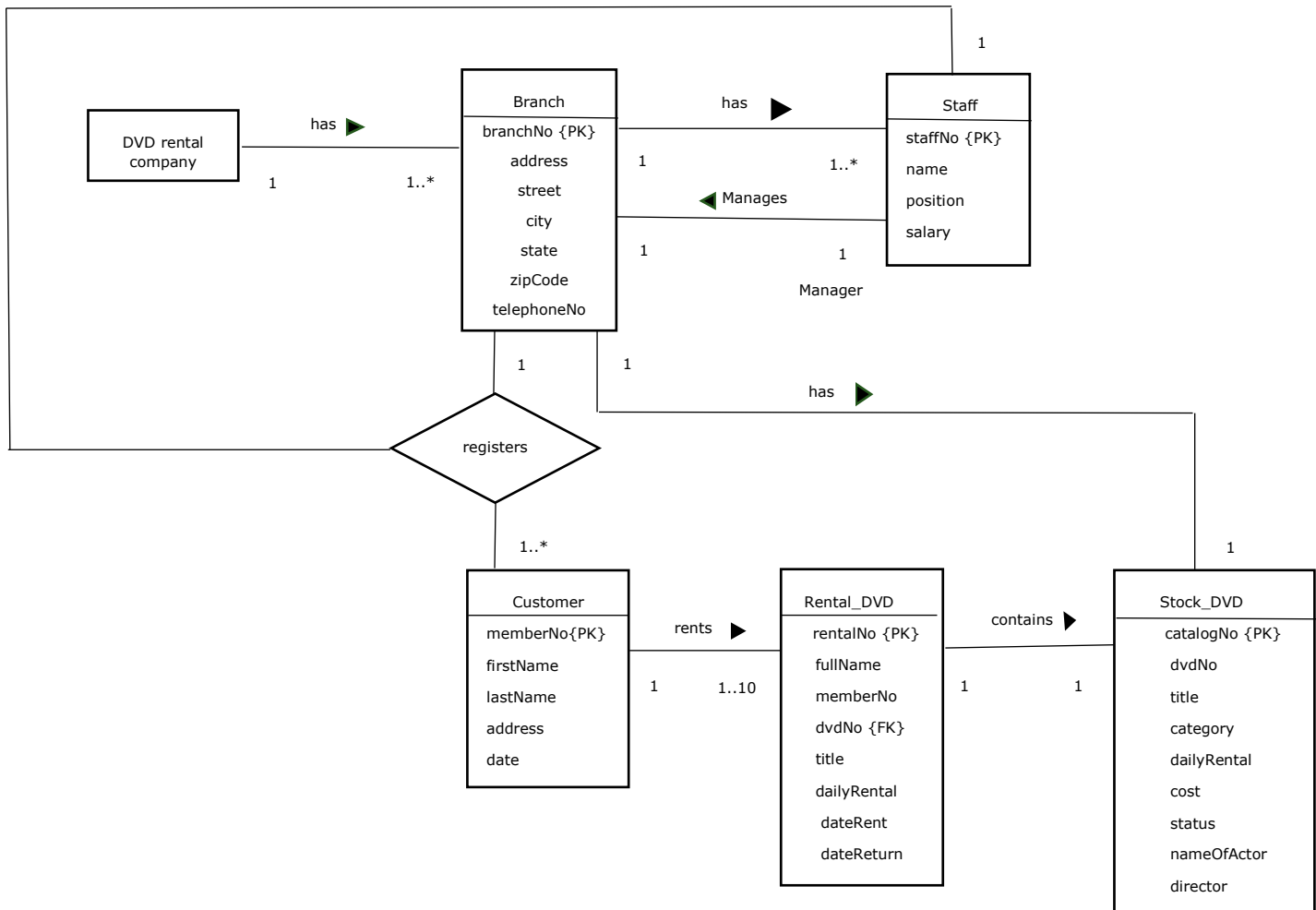
Stock_DVD: catalogNo is the primary key

catalogNo, dvdNo is the candidate key

Rental_DVD: rentalNo is the primary key

memberNo, rentalNo is the candidate key

Assumption: A staff can help register one to many customers.



Emp1 table:

The screenshot shows the SQL Developer interface with the Emp1 table selected in the Schemas pane. The SQL Editor contains the query `SELECT * FROM Emp1;`. The Result Grid displays the following data:

EmpNo	ENAME	JOB	MGR	HireDate	Salary	Comment	Dept	Location
7369	SMITH	CLERK	7902	1995-12-17	5000		40	
7499	JOHN	SALESMAN	7698	1995-12-17	7000		10	
7521	KIM	CLERK	7698	1995-12-17	3000		20	
7566	KING	Manager	7839	1993-07-14	5000		10	
7654	Tim	Sales Person	7698	1995-12-17	6500		20	
7698	Kim	Manager	7839	1993-06-17	5500		20	
7782	Mike	Manager	7839	1993-06-14	6000		30	
7788	Justin	Sales person	7566	1995-12-17	7200		40	
7839	Blake	President		1993-06-14	9000			
7844	Adams	CLERK	7698	1995-12-17	2900		30	
7876	Ford	CLERK	7788	1995-12-17	2700		40	
7900	Turner	Sales person	7698	1995-12-17	7400		40	
7902	Jim	Manager	7566	1995-06-14	5200		40	
7934	Scott	CLERK	7782	1995-12-17	3500		30	

The Output pane shows the execution of the query, indicating that 14 rows were returned in 0.016 seconds.

Emp2 table:

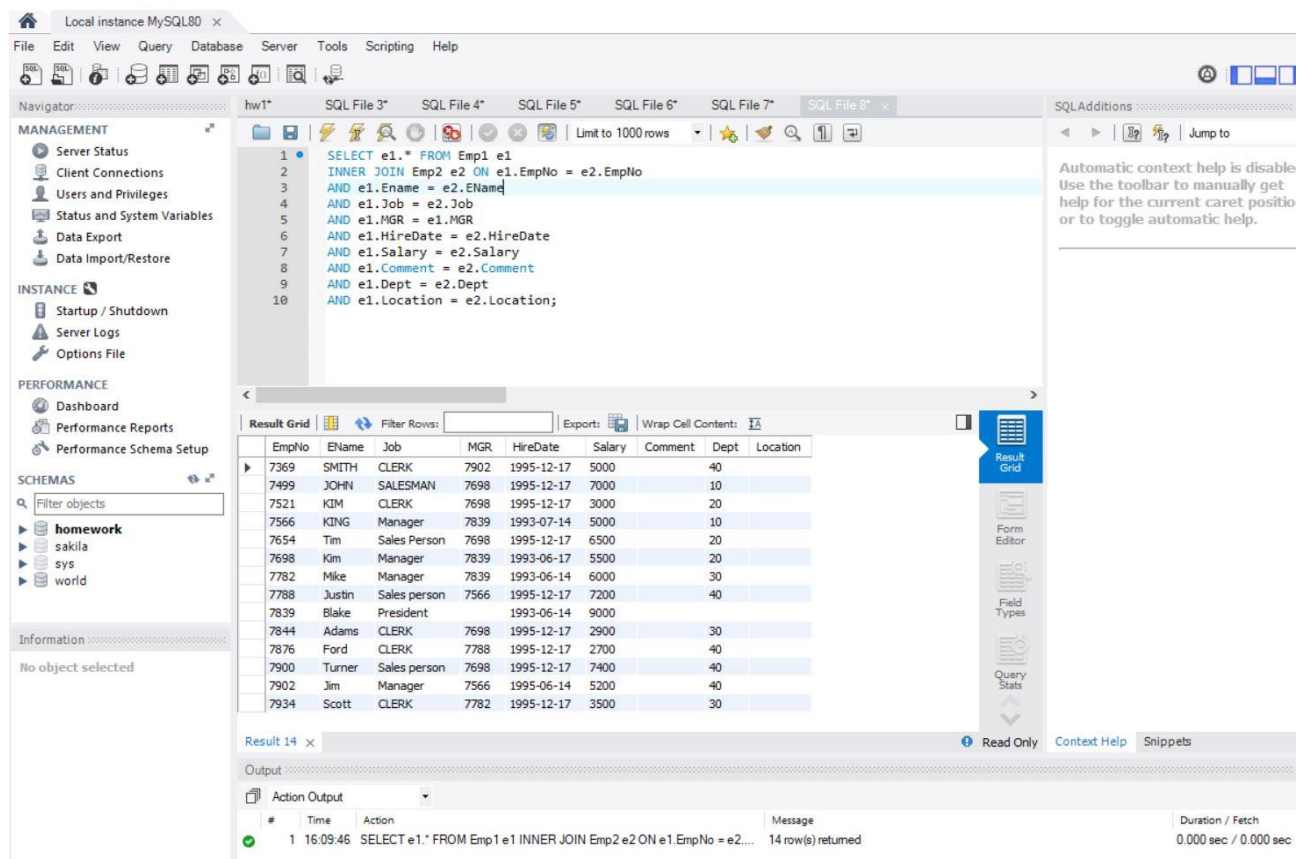
The screenshot shows the SQL Developer interface with the Emp2 table selected in the Schemas pane. The SQL Editor contains the query `SELECT * FROM Emp2;`. The Result Grid displays the following data:

EmpNo	ENAME	JOB	MGR	HireDate	Salary	Comment	Dept	Location
7369	SMITH	CLERK	7902	1995-12-17	5000		40	
7499	JOHN	SALESMAN	7698	1995-12-17	7000		10	
7521	KIM	CLERK	7698	1995-12-17	3000		20	
7566	KING	Manager	7839	1993-07-14	5000		10	
7654	Tim	Sales Person	7698	1995-12-17	6500		20	
7698	Kim	Manager	7839	1993-06-17	5500		20	
7782	Mike	Manager	7839	1993-06-14	6000		30	
7788	Justin	Sales person	7566	1995-12-17	7200		40	
7839	Blake	President		1993-06-14	9000			
7844	Adams	CLERK	7698	1995-12-17	2900		30	
7876	Ford	CLERK	7788	1995-12-17	2700		40	
7900	Turner	Sales person	7698	1995-12-17	7400		40	
7902	Jim	Manager	7566	1995-06-14	5200		40	
7934	Scott	CLERK	7782	1995-12-17	3500		30	

The Output pane shows the execution of the query, indicating that 14 rows were returned in 0.000 seconds.

- Compare the data and find out if the two tables have identical data

```
SELECT e1.* FROM Emp1 e1
INNER JOIN Emp2 e2 ON e1.EmpNo = e2.EmpNo
AND e1.Ename = e2.EName
AND e1.Job = e2.Job
AND e1.MGR = e1.MGR
AND e1.HireDate = e2.HireDate
AND e1.Salary = e2.Salary
AND e1.Comment = e2.Comment
AND e1.Dept = e2.Dept
AND e1.Location = e2.Location;
```



The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL query:

```
SELECT e1.* FROM Emp1 e1
INNER JOIN Emp2 e2 ON e1.EmpNo = e2.EmpNo
AND e1.Ename = e2.EName
AND e1.Job = e2.Job
AND e1.MGR = e1.MGR
AND e1.HireDate = e2.HireDate
AND e1.Salary = e2.Salary
AND e1.Comment = e2.Comment
AND e1.Dept = e2.Dept
AND e1.Location = e2.Location;
```

The query has been executed, and the results are displayed in the Result Grid. The grid shows 14 rows of employee data. The columns are: EmpNo, EName, Job, MGR, HireDate, Salary, Comment, Dept, and Location.

EmpNo	EName	Job	MGR	HireDate	Salary	Comment	Dept	Location
7369	SMITH	CLERK	7902	1995-12-17	5000		40	
7499	JOHN	SALESMAN	7698	1995-12-17	7000		10	
7521	KIM	CLERK	7698	1995-12-17	3000		20	
7566	KING	Manager	7839	1993-07-14	5000		10	
7654	Tim	Sales Person	7698	1995-12-17	6500		20	
7698	Kim	Manager	7839	1993-06-17	5500		20	
7782	Mike	Manager	7839	1993-06-14	6000		30	
7788	Justin	Sales person	7566	1995-12-17	7200		40	
7839	Blake	President		1993-06-14	9000			
7844	Adams	CLERK	7698	1995-12-17	2900		30	
7876	Ford	CLERK	7788	1995-12-17	2700		40	
7900	Turner	Sales person	7698	1995-12-17	7400		40	
7902	Jim	Manager	7566	1995-06-14	5200		40	
7934	Scott	CLERK	7782	1995-12-17	3500		30	

The Output pane at the bottom shows the execution message: "SELECT e1.* FROM Emp1 e1 INNER JOIN Emp2 e2 ON e1.EmpNo = e2.EmpNo... 14 row(s) returned".

- List the duplicate rows exists in Emp1 and Emp2

```
SELECT e1.* FROM Emp1 e1
INNER JOIN Emp2 e2 ON e1.EmpNo = e2.EmpNo
AND e1.Ename = e2.EName
AND e1.Job = e2.Job
AND e1.MGR = e1.MGR
AND e1.HireDate = e2.HireDate
AND e1.Salary = e2.Salary
```

AND e1.Comment = e2.Comment

AND e1.Dept = e2.Dept

AND e1.Location = e2.Location;

The screenshot shows the MySQL Workbench interface for a local instance of MySQL80. The query editor displays a SELECT statement with an INNER JOIN between Emp1 and Emp2 tables, filtering for matching EmpNo, EName, Job, MGR, HireDate, Salary, Comment, Dept, and Location. The Results Grid shows 14 rows of data from the Emp1 table. The left sidebar contains navigation panels for Management, Instance, Performance, and Schemas. The bottom panel shows the Action Output with the executed query and its duration.

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

SCHEMAS

- Filter objects
- homework
- sakila
- sys
- world

Information

No object selected

SQL File 3* SQL File 4* SQL File 5* SQL File 6* SQL File 7* SQL File 8*

SQL File 3*

```
1 SELECT e1.* FROM Emp1 e1
2 INNER JOIN Emp2 e2 ON e1.EmpNo = e2.EmpNo
3 AND e1.EName = e2.EName
4 AND e1.Job = e2.Job
5 AND e1.MGR = e1.MGR
6 AND e1.HireDate = e2.HireDate
7 AND e1.Salary = e2.Salary
8 AND e1.Comment = e2.Comment
9 AND e1.Dept = e2.Dept
10 AND e1.Location = e2.Location;
```

Result Grid

EmpNo	EName	Job	MGR	HireDate	Salary	Comment	Dept	Location
7369	SMITH	CLERK	7902	1995-12-17	5000		40	
7499	JOHN	SALESMAN	7698	1995-12-17	7000		10	
7521	KIM	CLERK	7698	1995-12-17	3000		20	
7566	KING	Manager	7839	1993-07-14	5000		10	
7654	Tim	Sales Person	7698	1995-12-17	6500		20	
7698	Kim	Manager	7839	1993-06-17	5500		20	
7782	Mike	Manager	7839	1993-06-14	6000		30	
7788	Justin	Sales person	7566	1995-12-17	7200		40	
7839	Blake	President		1993-06-14	9000			
7844	Adams	CLERK	7698	1995-12-17	2900		30	
7876	Ford	CLERK	7788	1995-12-17	2700		40	
7900	Turner	Sales person	7698	1995-12-17	7400		40	
7902	Jim	Manager	7566	1995-06-14	5200		40	
7934	Scott	CLERK	7782	1995-12-17	3500		30	

Result 14 x

Read Only Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	16:09:46	SELECT e1.* FROM Emp1 e1 INNER JOIN Emp2 e2 ON e1.EmpNo = e2.EmpNo	14 row(s) returned	0.000 sec / 0.000 sec

- List the Data that exists in Emp1 but not in Emp2

SELECT e1.* FROM Emp1 e1

WHERE NOT EXISTS (SELECT e2.EmpNo FROM Emp2 e2 where e1.EmpNo = e2.EmpNo);

Inserting two rows in Emp1 table which are not in Emp2 table:

INSERT INTO Emp1 (EmpNo, EName, Job, MGR, HireDate, Salary, Comment, Dept, Location)

VALUES (7940, 'James', 'CLERK', 7782, '1995-12-17', 4000, '', 40, ''),

(7945, 'Mac', 'SALESMAN', 7698, '1995-12-17', 6000, '', 10, '');

Navigator: SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9* Administration - Dashboard SQL File 10*

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

SCHEMAS

- Filter objects
- homework
- sakila
- sys
- world

SQL File 5*

```

1 /*INSERT INTO Emp1 (EmpNo, EName, Job, MGR, HireDate, Salary, Comment, Dept, Location)
2 VALUES (7940, 'James', 'CLERK', 7782, '1995-12-17', 4000, '', 40, '');
3 (7945, 'Mac', 'SALESMAN', 7698, '1995-12-17', 6000, '', 10, '');*/
4
5 Select * from Emp1;

```

Result Grid

EmpNo	EName	Job	MGR	HireDate	Salary	Comment	Dept	Location
7369	SMITH	CLERK	7902	1995-12-17	5000		40	
7499	JOHN	SALESMAN	7698	1995-12-17	7000		10	
7521	KIM	CLERK	7698	1995-12-17	3000		20	
7566	KING	Manager	7839	1993-07-14	5000		10	
7654	Tim	Sales Person	7698	1995-12-17	6500		20	
7698	Kim	Manager	7839	1993-06-17	5500		20	
7782	Mike	Manager	7839	1993-06-14	6000		30	
7788	Justin	Sales person	7566	1995-12-17	7200		40	
7839	Blake	President		1993-06-14	9000			
7844	Adams	CLERK	7698	1995-12-17	2900		30	

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	19:07:06	Select * from Emp1 LIMIT 0, 1000	16 row(s) returned	0.000 sec / 0.000 sec

File Edit View Query Database Server Tools Scripting Help

Navigator: SQL File 5* SQL File 6* SQL File 7* SQL File 8* SQL File 9* Administration - Dashboard SQL File 10*

MANAGEMENT

- Server Status
- Client Connections
- Users and Privileges
- Status and System Variables
- Data Export
- Data Import/Restore

INSTANCE

- Startup / Shutdown
- Server Logs
- Options File

PERFORMANCE

- Dashboard
- Performance Reports
- Performance Schema Setup

SCHEMAS

- Filter objects
- homework
- sakila
- sys
- world

SQL File 5*

```

1 SELECT e1.* FROM Emp1 e1
2 WHERE NOT EXISTS (SELECT e2.EmpNo FROM Emp2 e2 where e1.EmpNo = e2.EmpNo);

```

Result Grid

EmpNo	EName	Job	MGR	HireDate	Salary	Comment	Dept	Location
7940	James	CLERK	7782	1995-12-17	4000		40	
7945	Mac	SALESMAN	7698	1995-12-17	6000		10	

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	19:09:26	SELECT e1.* FROM Emp1 e1 WHERE NOT EXISTS (SELECT e2.EmpNo ...	2 row(s) returned	0.000 sec / 0.000 sec

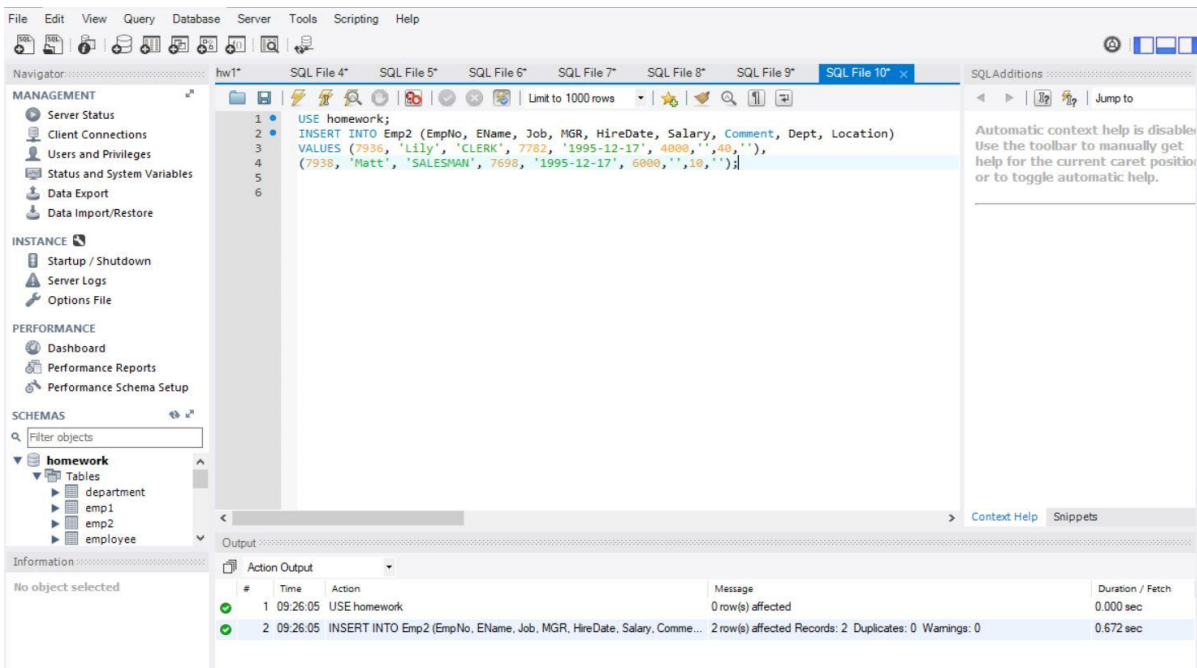
- List the Data that exists in Emp2 but not in Emp1

First of all, inserting two rows which are in Emp2 but not in Emp1

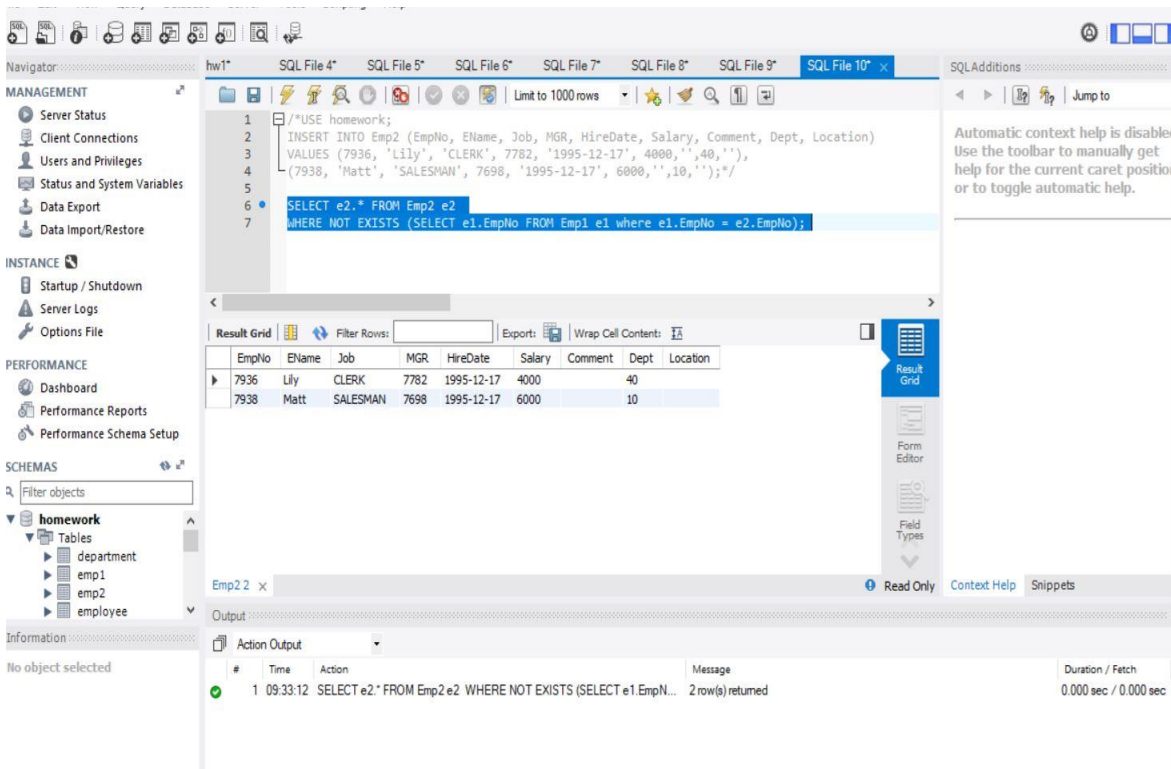
INSERT INTO Emp2 (EmpNo, EName, Job, MGR, HireDate, Salary, Comment, Dept, Location)

VALUES (7936, 'Lily', 'CLERK', 7782, '1995-12-17', 4000, '', 40, ''),


```
(7938, 'Matt', 'SALESMAN', 7698, '1995-12-17', 6000, '', 10, '' );
```



```
SELECT e2.* FROM Emp2 e2
WHERE NOT EXISTS (SELECT e1.EmpNo FROM Emp1 e1 where e1.EmpNo = e2.EmpNo);
```



- List all the employees and their reporting hierarchy

```
USE homework;
SELECT
concat(e1.EName, ' reports to ', e2.EName, ' who reports to ', e3.EName)
FROM Emp1 e1, Emp1 e2, Emp1 e3 WHERE (e1.MGR = e2.EmpNo)
```

AND (e2.MGR = e3.EmpNo)

ORDER BY e1.EmpNo;

SQL Developer interface showing a query execution. The query is:

```
USE homework;
SELECT
  concat(e1.EName, ' reports to ', e2.EName, ' who reports to ', e3.EName)
FROM Emp1 e1, Emp1 e2, Emp1 e3 WHERE (e1.MGR = e2.EmpNo)
AND (e2.MGR = e3.EmpNo)
ORDER BY e1.EmpNo;
```

The result grid shows the following output:

concat(e1.EName, ' reports to ', e2.EName, ' who reports to ', e3.EName)
SMITH reports to Jim who reports to KING
JOHN reports to Kim who reports to Blake
KIM reports to Kim who reports to Blake
Tim reports to Kim who reports to Blake
Justin reports to KING who reports to Blake
Adams reports to Kim who reports to Blake
Ford reports to Justin who reports to KING
Turner reports to Kim who reports to Blake
Jim reports to KING who reports to Blake
Scott reports to Mike who reports to Blake
James reports to Mike who reports to Blake
Mac reports to Kim who reports to Blake

The action output shows the following results:

#	Time	Action	Message	Duration / Fetch
1	22:37:27	USE homework	0 row(s) affected	0.000 sec
2	22:37:27	SELECT concat(e1.EName, ' reports to ', e2.EName, ' who reports to ', ...	12 row(s) returned	0.000 sec / 0.000 sec