EDA - 603 Major Assignment 1 – DB Modeling

Parul Bhutani Wadhwa (000504532)

Saskatchewan Polytechnic

Abstract

Normalization is a design approach which is used to create structured organization of data by reducing data redundancy and improving data integrity. It ensures that the dataset is saved only once to manage and update it easily.

Normalization is achieved by dividing a huge dataset into smaller and more manageable tables, which are connected to each other. By adding constraints, data redundancy is reduced, duplicate data is removed, and data consistency is improved. Normalization levels, commonly known as normal forms, vary from first normal form (1NF) to fifth normal form (5NF) (5NF). To ensure that the database is properly normalized, all the levels must follow a set of rules and principles. The first normal form (1NF) needs atomic values in each column of a table, implying that the data cannot be split down into smaller components. Each non-key column in second normal form (2NF) must be completely dependent on the primary key. Each non-key column must be independent of the other non-key columns in the third normal form (3NF).

Table of Contents

Normalization

The technique of structuring data in a database to reduce redundancy and increase data integrity is known as normalization.

Process of Normalization

The following steps are involved in the normalization process:

1. Determine entities and attributes: In the first stage we will identify which entities and characteristics will be stored in the database. In the employee excel sheet, the entities are employee, employee roles, committee, employee committee, and supervisor, and the attributes are EmpId, SinNo, FirstName, LastName, Street, City, Province, Postal, Birth Date, HiringDate, Inc. Tax, Hours, OT, Person Hours Worked, Supervisor, Supervisor Cell, Committee Id, Committee Name, and Meeting Night.
2. We will create distinct tables for each entity once the entities and attributes are decided. Each table should have a primary key that identifies each unique record in the table.
3. Identify relationships: Now we'll build the connections between the tables. For example, a supervisor may have numerous employees, which means there's a one-to-many link between the supervisor and employee tables. Also, an employee can be a member of multiple Committees, stating a one-to-many relationship between the person and the Committee table.

4. Creating 2NF: Now identify each non-key column is completely dependent on the primary key and the dependent columns are shifted to a separate table in case of any partial dependencies.

   The 2NF contains below tables:

Employee Table	**Payroll Table**	**Job Table**	**Committee Table**	**EmployeeCommittee Involvement**
EmpID, Last, First, Street, City, Prov, Postal, SIN, Birth Date	EmpID, Pay_Week_End_Date, Days_available, Hours, OT, Person_Hours_Worked	EmpId, Job_Code, Position, Payrate,Inc Tax,Hire Date, Supervisor,Supervisor cell	ComId, ComName, MeetingNight	compositeId, EmpId, Committee Id

5. Creating 3NF: Now, we must ensure that each non-key column is independent of the others and the dependent columns are moved to a separate table when there are any transitive dependencies.

Employee Table (Employee_ID, Last, First, Street, City, Prov, Postal, SIN Birth_Date)
Payroll Table (EmpID, Pay_Week_End_Date,Days_Available, Hours, OT)
Job Table (EmpId, Job_Code, Inc.Tax, Hire Date)
Comittees Table (Com_Id, Com_Name, Meeting_Night)
Supervisor Table (sId, Sname, Scell)
Position Table (job code,Position,payrate,SId)
EmployeeCommitteeInvolvement(CompositeId,EmpID,CommitteeId)

Questions

1) **There is a concern that the spreadsheet will get out of hand, as the number of pay periods increase, how does your 3NF solve this issue?**

➔ In the 3NF we have created a separate table “Payroll Table” which will take care of the employee incomes for every Pay Period. So for every new pay period the data can be inserted to this table and it can be handled easily.

2) **Supervisors often manage several departments, for example, Chad Long is managing butchers and bakers, how did you address this problem?**

➔ I have created a separate table “Position Table” for various departments and “Supervisor Table” for all the supervisor details and created a composite key “supervisorId” which becomes the primary key for the table. In “Position Table” every unique department is linked to supervisor Id. If the user wants to get details of supervisor through any department, he can use inner join in position table with supervisor table based on supervisor Id as foreign key. Hence, one can find all the departments linked to one supervisor and viz-a-viz.

3) **Employees often serve on multiple committees at once, how did you address this?**

➔ I have created a table “Employee Committee Involvement”. In this table each employee id is linked with its committee id. Also, a composite key “EC_Id” is created which is unique for all the records and serves as the primary key in the table. Also, even if an employee is serving multiple committees, then committee meetings happen on different days which will not create any clashes.

4) **Person hours worked is calculated (it represents information, not data) for all hours worked by all employees for that week, is there a better way to handle this rather than record it in the database? Implement your solution in the queries section.**

➔ Person hours worked is calculated by adding the hours of all the employees in one pay period date. So, this field does not need to be stored in a database. Rather, we can create a procedure and it can be called whenever we need this field.

➔
```
Create Procedure PersonHoursWorked
As
BEGIN
    Select PayWeekendDate, sum(HoursWorked) as totalHoursWorked
    From employeePayroll
    Where PaywEekendDate != ' '
    Group by PayWeekendDate;
END;
```

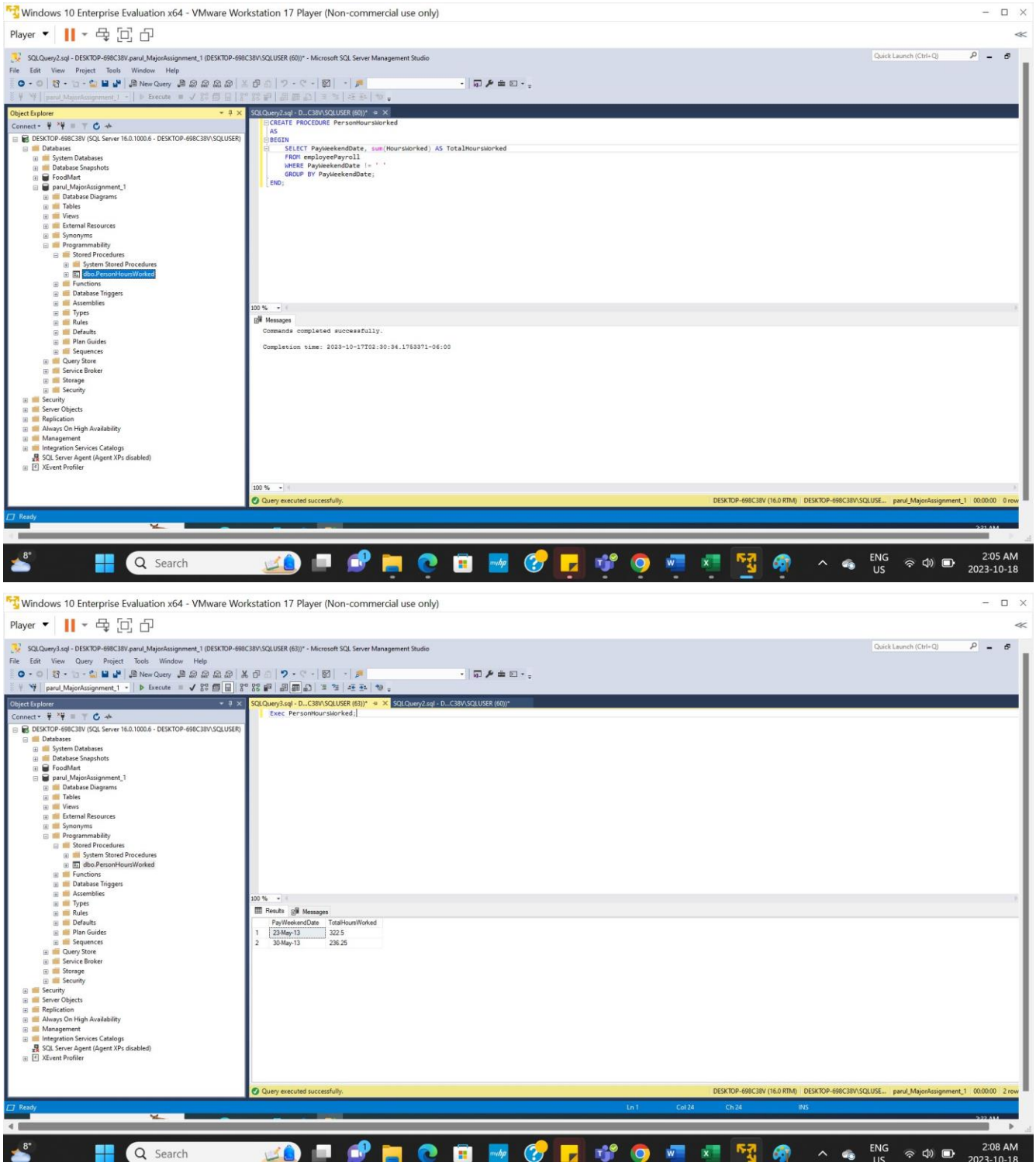

Fig 1. Store Procedure to Calculate Person Hours Worked and its result

Entity Relationship Diagram

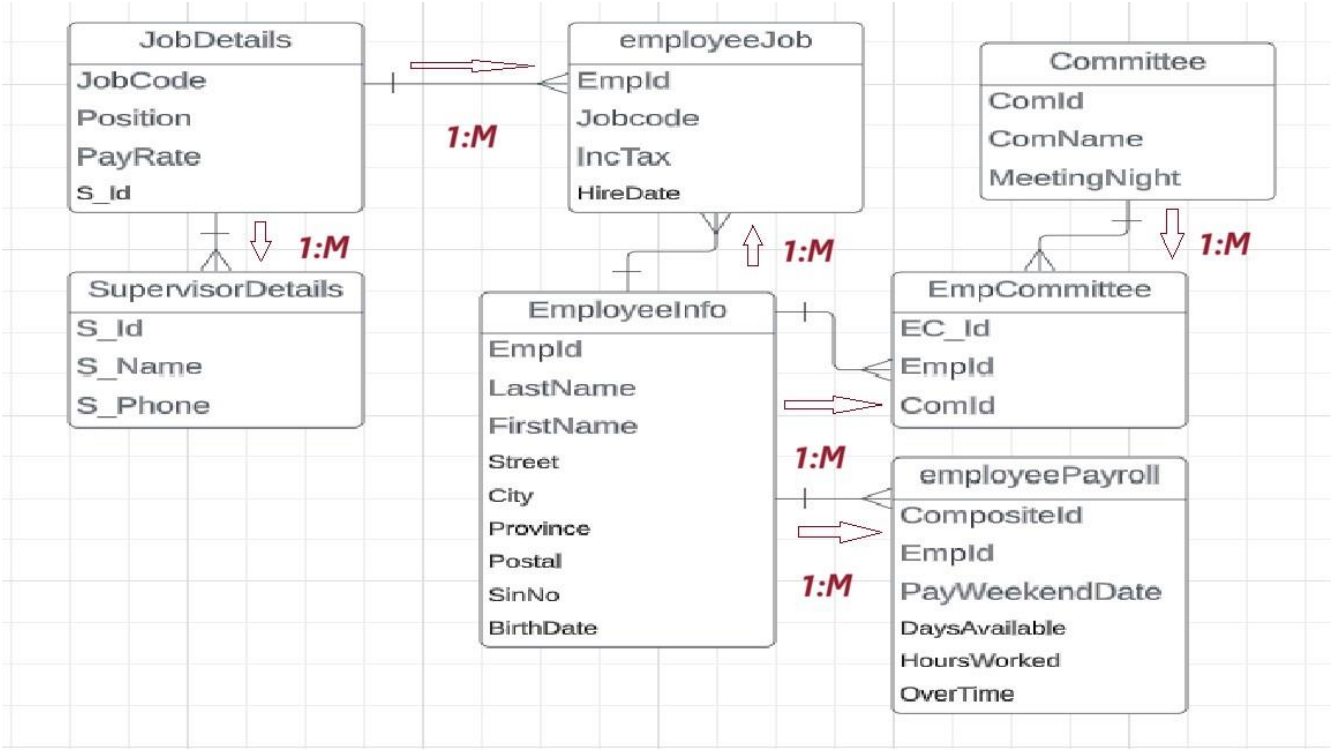

Fig 2. ER Diagram of 3NF.

Queries

1) Who worked more than 25 hours in any pay period?

➔ select ep.EmpId, CONCAT(ef.FirstName,' ',ef.LastName) as 'Full Name', sd.S_Name as 'Supervisor Name', ep.HoursWorked as 'Hours' from employeePayroll as ep
inner join employeeInfo as ef on ep.EmpId=ef.EmpId
inner join employeeJob as ej on ej.EmpId=ep.EmpId
inner join JobDetails as jd on jd.JobCode=ej.JobCode
inner join SupervisorDetails as sd on sd.S_Id=jd.S_Id
where ep.HoursWorked>25 and ef.LastName like '%s%';

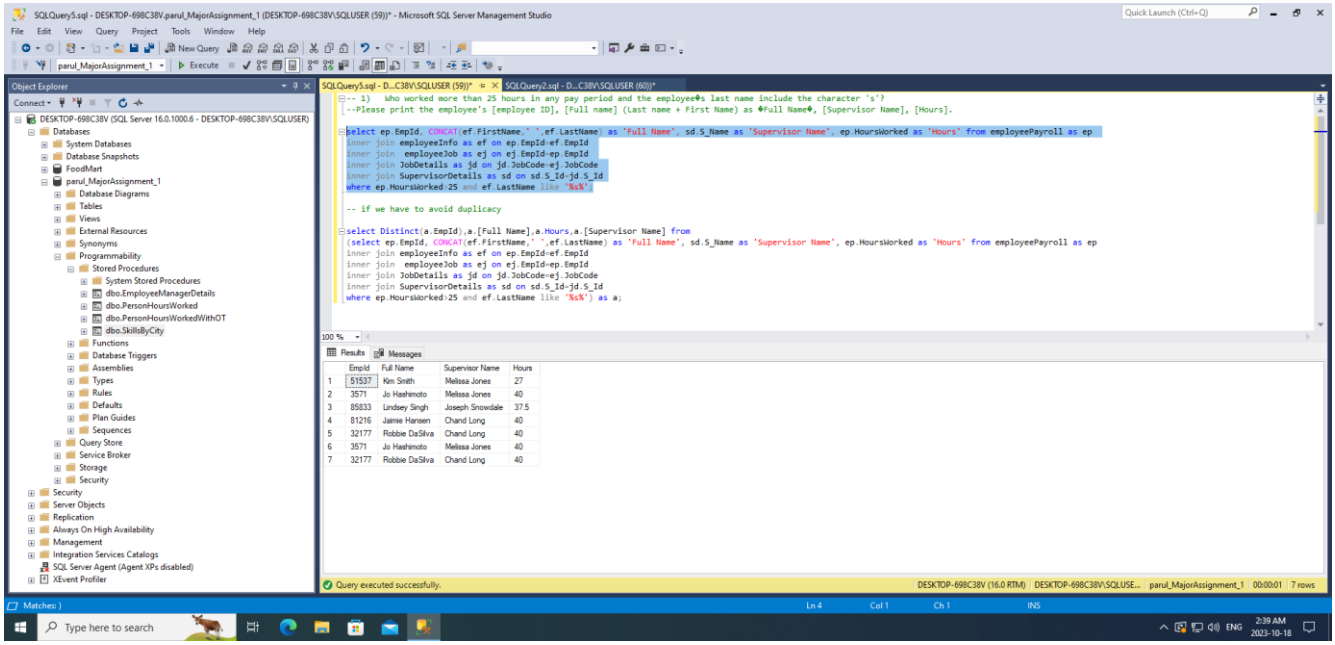

Fig 3. Employee who worked more than 35 hours (with duplicate records).

➔ select Distinct(a.EmpId),a.[Full Name],a.Hours,a.[Supervisor Name] from
(select ep.EmpId, CONCAT(ef.FirstName,' ',ef.LastName) as 'Full Name', sd.S_Name as 'Supervisor Name', ep.HoursWorked as 'Hours' from employeePayroll as ep
inner join employeeInfo as ef on ep.EmpId=ef.EmpId
inner join employeeJob as ej on ej.EmpId=ep.EmpId
inner join JobDetails as jd on jd.JobCode=ej.JobCode
inner join SupervisorDetails as sd on sd.S_Id=jd.S_Id
where ep.HoursWorked>25 and ef.LastName like '%s%') as a;

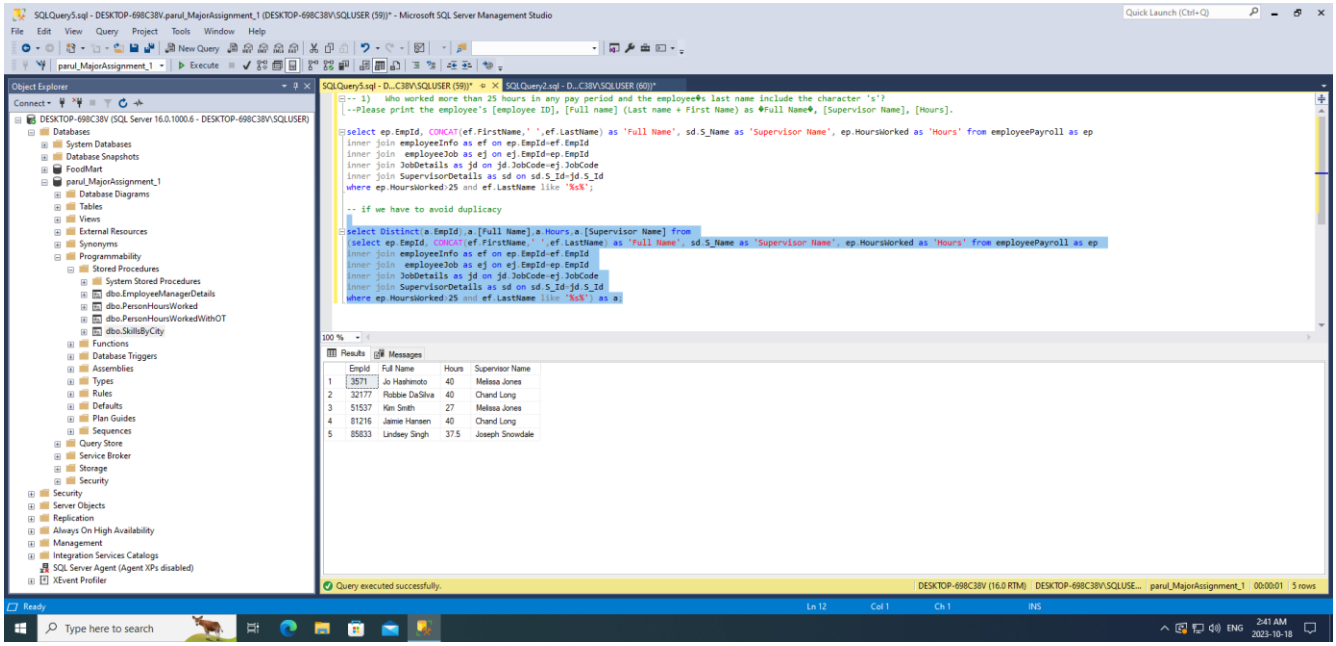

Fig 4. Employees who worked more than 25 hours (without duplicate records).

2) Who worked overtime hours in the 30-May-13 pay period?

➔ select ep.EmpId, CONCAT(ef.FirstName,' ',ef.LastName) as 'Full Name',
ep.PayWeekendDate, ep.OverTime from employeePayroll as ep
inner join employeeInfo as ef on ep.EmpId=ef.EmpId
where PayWeekendDate = '30-May-13' and OverTime > 0;

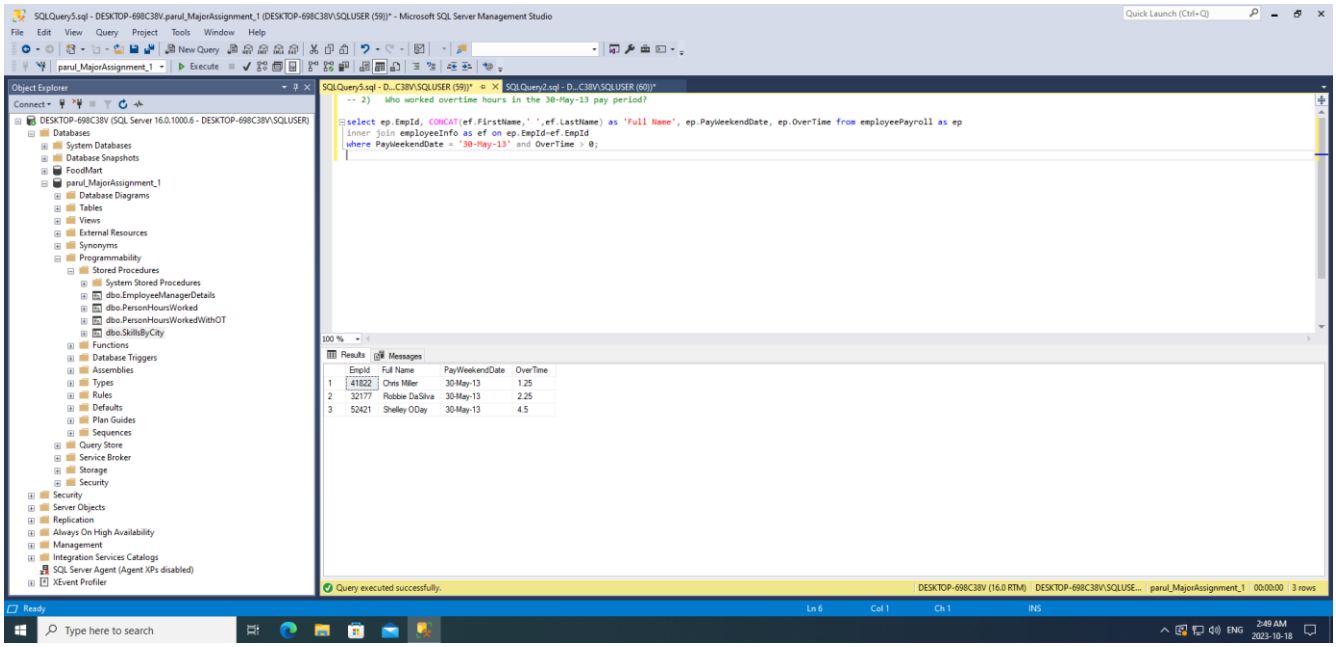

Fig 5. Employee who worked Overtime for the pay period 30-May-13.

3) I want to enter an employee id and find their job code, their manager's name and their manager's cell number

➔ select ej.JobCode, sd.S_Name as 'Manager Name', sd.S_Phone as 'Manager Cell Number' from employeeInfo as ep
inner join employeeJob as ej on ep.EmpId=ej.EmpId
inner join JobDetails as jd on ej.JobCode=jd.JobCode
inner join SupervisorDetails as sd on sd.S_Id=jd.S_Id
where ep.EmpId='33982';

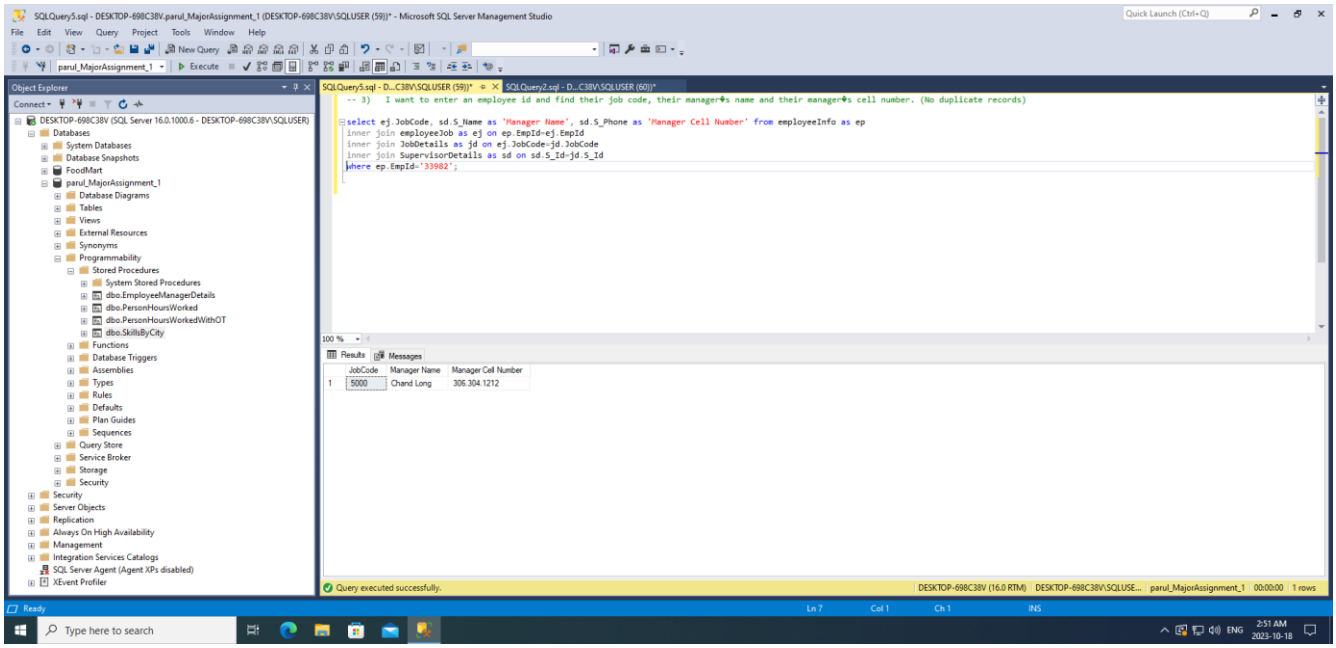

Fig 5. Query for retrieving JobCode, Managers details based on EmployeeId.

Also, this can be resolved using procedure:

➔ Create Procedure EmployeeManagerDetails @empId nvarchar(50)
as
BEGIN
select ej.JobCode, sd.S_Name as 'Manager Name', sd.S_phone as 'Manager Cell Number' from employeeInfo as ep
inner join employeeJob as ej on ep.EmpId=ej.EmpId inner join JobDetails as jd on ej.JobCode=jd.JobCode
inner join SupervisorDetails as sd on sd.S_Id=jd.S_Id where ep.EmpId=@empId;
END;

➔ Exec EmployeeManagerDetails @empId='33982';

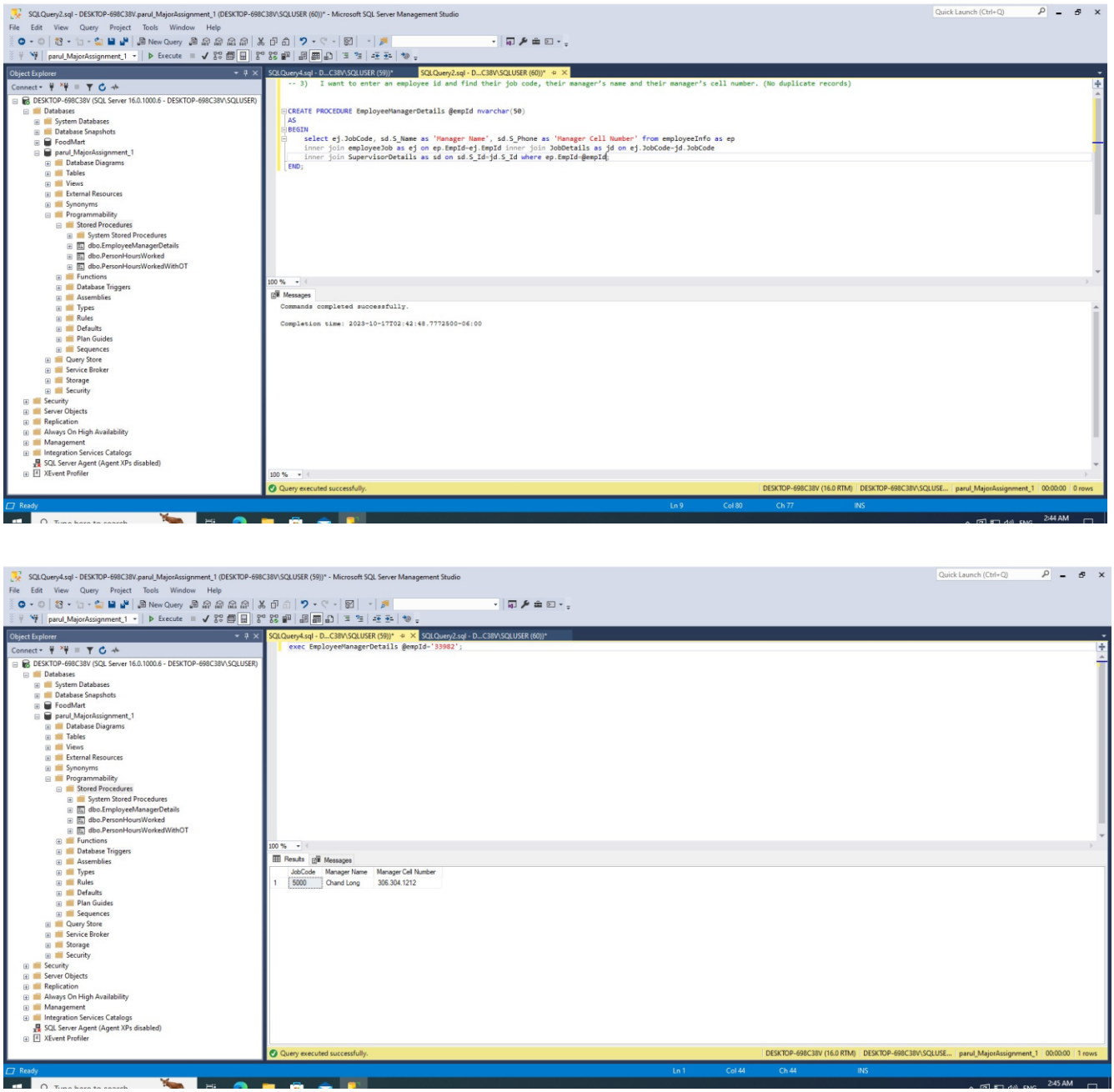

Fig 6. Procedure for retrieving JobCode, Managers details based on EmployeeId with results.

4) I'd like to know the skill sets we have by city. I want to enter a city and geta list of the people who live there and their job descriptions.

➔ select ei.EmpId, CONCAT(ei.FirstName,' ',ei.LastName) as 'Full Name', jd.Position as 'Skill'
from employeeInfo as ei inner join employeeJob as ej on ei.EmpId=ej.EmpId
inner join JobDetails as jd on jd.JobCode=ej.JobCode
where ei.City='Moose Jaw' ;

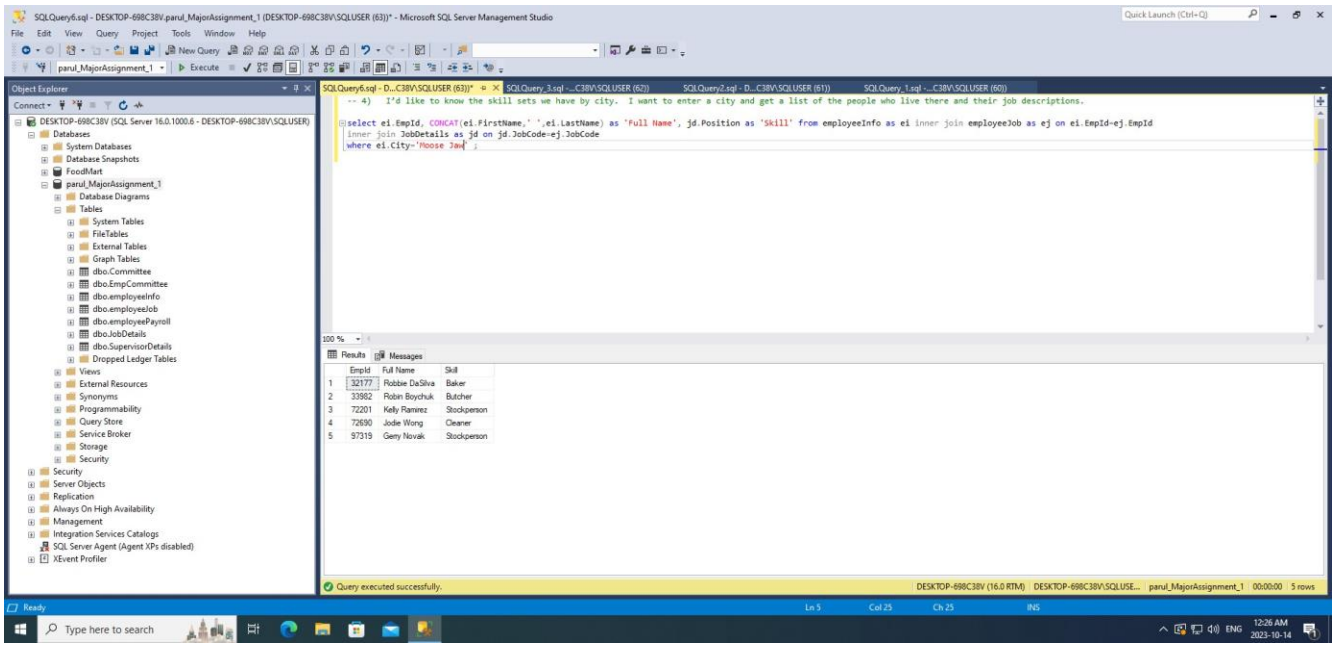

Fig 7. Query for retrieving Person Name and Job Description based on City.

This can also be resolved using a stored procedure.

➔ CREATE PROCEDURE SkillsByCity @city nvarchar(50)
AS
BEGIN
select ei.EmpId, CONCAT(ei.FirstName,' ',ei.LastName) as 'Full Name', jd.Position as 'Skill'
from employeeInfo as ei inner join employeeJob as ej on ei.EmpId=ej.EmpId
inner join JobDetails as jd on jd.JobCode=ej.JobCode
where ei.City=@city
END;

➔ Exec SkillsByCity @city='Moose Jaw';

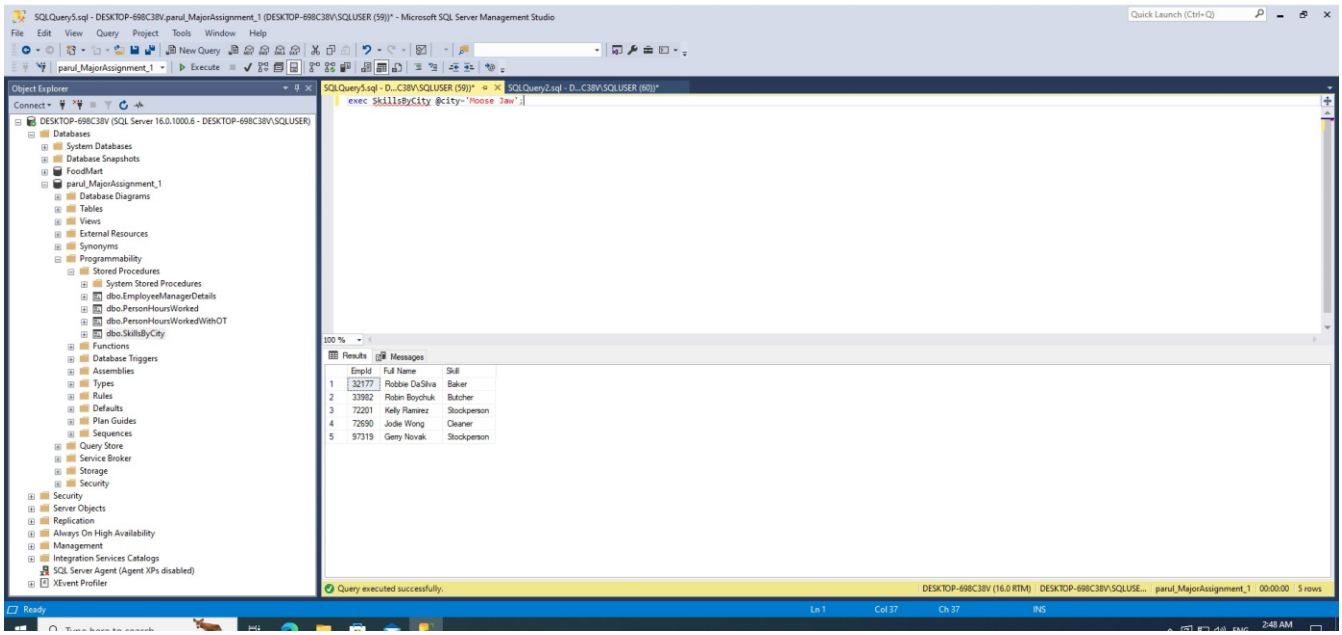

Fig 8. Procedure for retrieving Person Name and Job Description based on City with results.

5) Given a committee name 'OH&S', I'd like a list of people who work on it and their job description.

➔ select ei.EmpId,CONCAT(ei.FirstName,' ',ei.LastName) as 'Full Name', jd.Position as 'Job Description' from Committee as c inner join EmpCommittee as ec on ec.ComId = c.ComId inner join employeeJob as ej on ej.EmpId=ec.EmpId
inner join JobDetails as jd on jd.JobCode=ej.JobCode inner join employeeInfo as ei on ei.EmpId=ej.EmpId
where c.ComName = 'OH&S';

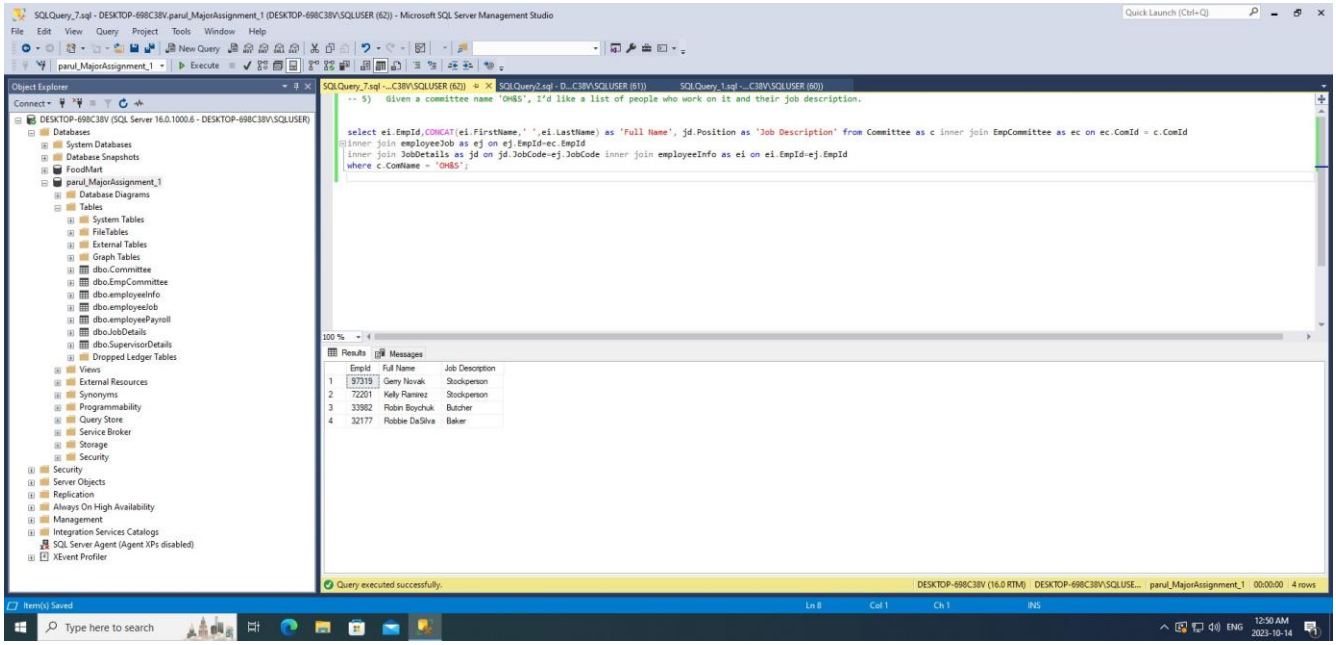

Fig 9. Query to retrieve Person Name and Job Description based on Committee Name.

6) Given a supervisor last name ‘Goldberg’ , provide a list of employees being supervised and their job code.

➔ select ei.EmpId, CONCAT(ei.FirstName,' ',ei.LastName) as 'Full Name', jd.JobCode from SupervisorDetails as sd inner join JobDetails as jd on jd.S_Id=sd.S_Id
inner join employeeJob ej on ej.JobCode=jd.JobCode inner join employeeInfo as ei on ei.EmpId=ej.EmpId
where sd.S_Name like '%Goldberg%' ;

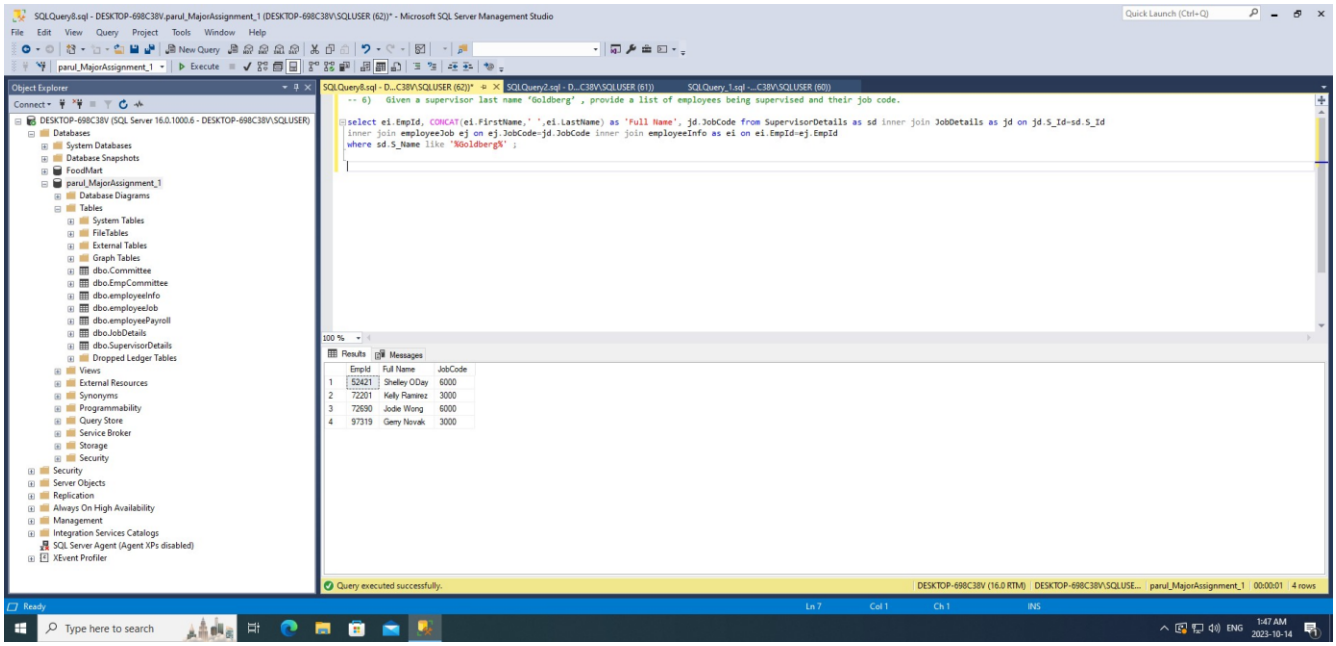

Fig 10. Query to retrieve Employee List along with Job Code based on Supervisor LastName.

7) A query that calculated person hours worked (as mentioned above). It should also include overtime.

➔ Select PayWeekendDate, sum(HoursWorked+OverTime) as PersonHoursWorked from employeePayroll where PayWeekendDate != ' ' group by PayWeekendDate;

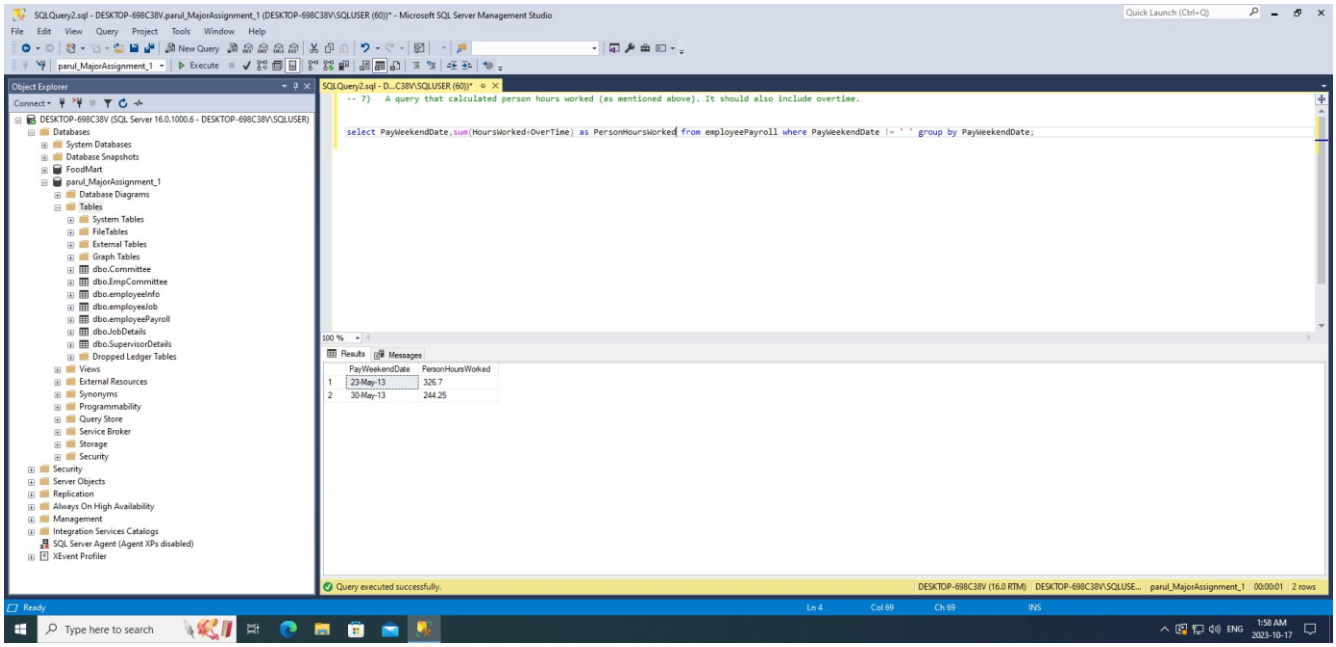

Fig 11. Query to calculate Person Hours Worked Including OverTime.

This can also be resolved using a stored procedure:

➔ Create Procedure PersonHoursWorkedWithOT
AS
BEGIN
SELECT PayWeekendDate, sum(HoursWorked+OverTime) AS
TotalHoursWorked FROM employeePayroll where PayWeekendDate != ‘ ’
group by PayWeekendDate;
END;

➔ Exec PersonHoursWorkedWithOT;

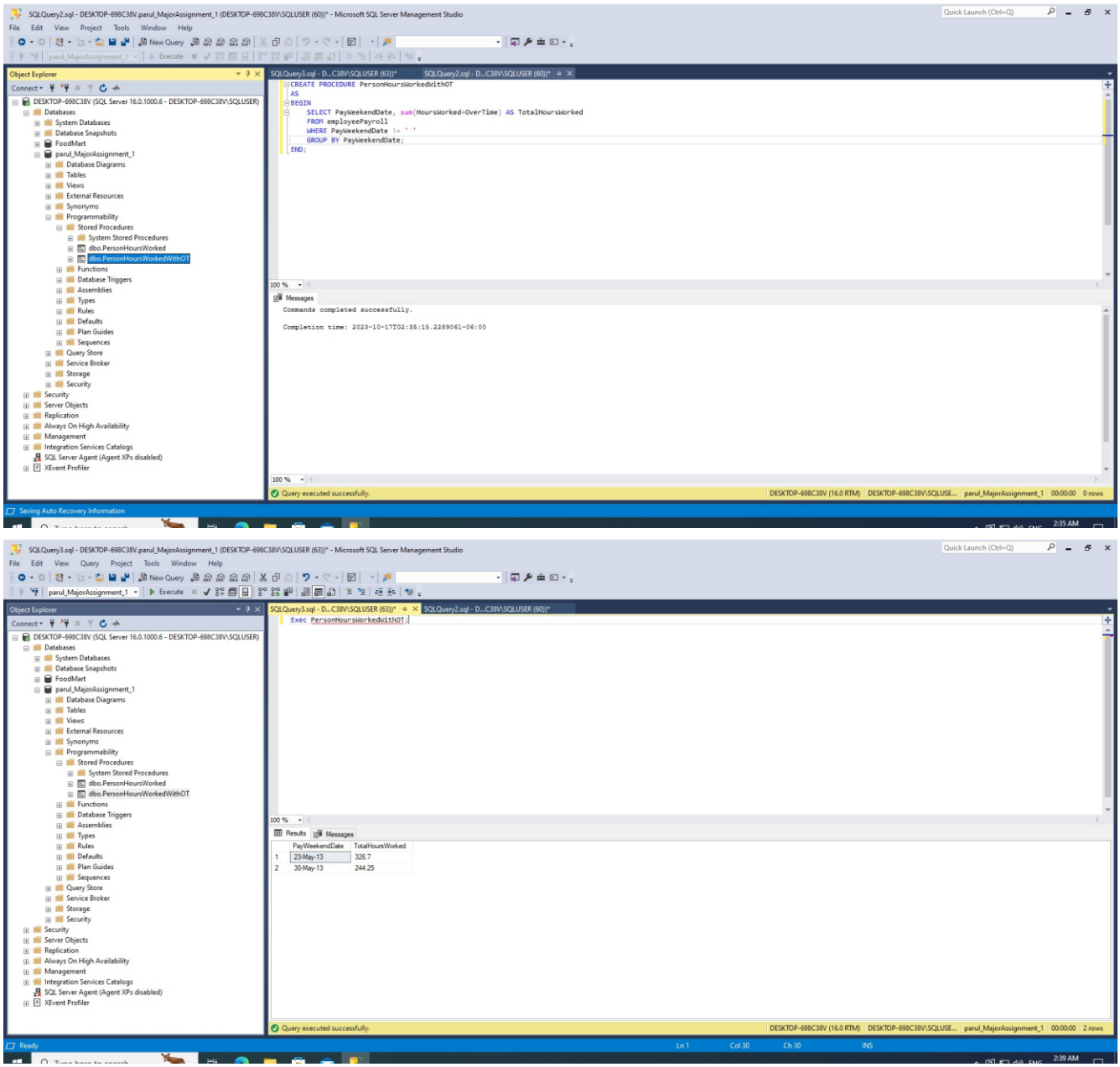

Fig 12. Procedure to calculate Person Hours Worked Including OverTime with results.

8) List the number of employees under each supervisor. Only include the supervisor with more than 2 employees. Print out [SupervisorID],[Number of Employees].

➔ select jd.s_id as 'Supervisor Id', count(distinct(ej.empId)) as 'No. Of Employees' from employeeJob ej inner join JobDetails jd on ej.JobCode=jd.JobCode
group by jd.S_Id having count(ej.empId)>2 ;

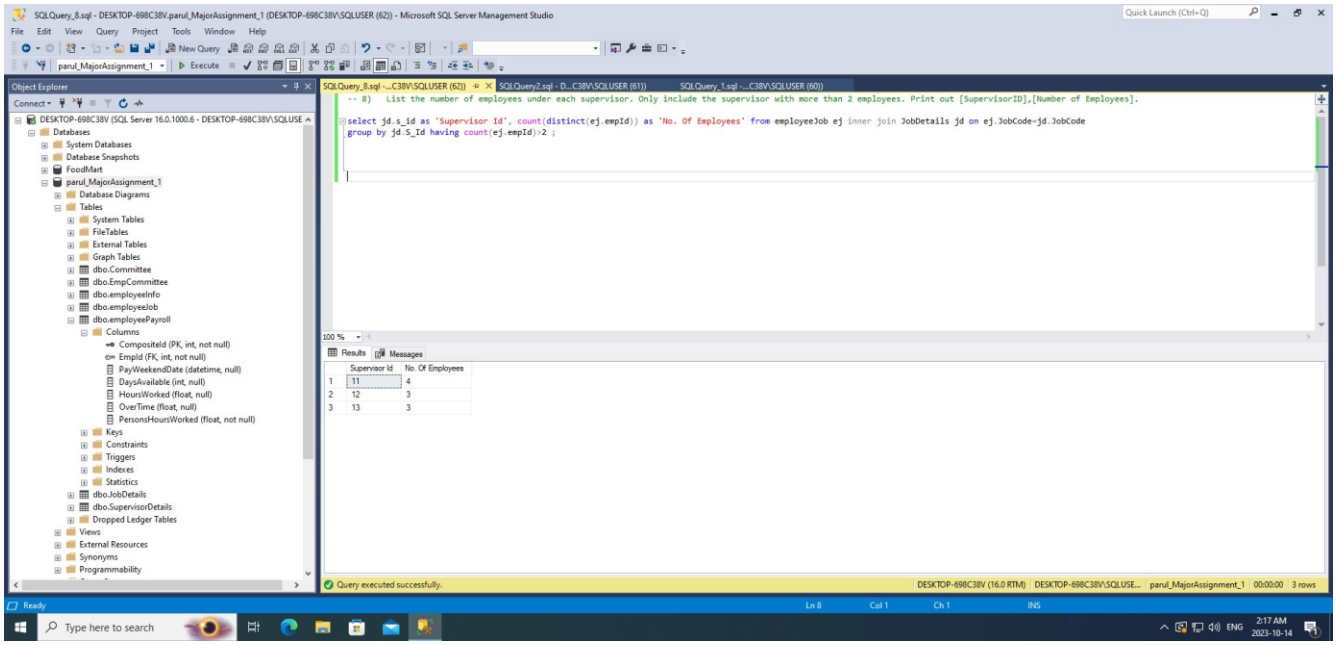

Fig 13. Query to list employees under each supervisor.

9) List the supervisor's ID and name, who supervises more than 2 employees. (hint: using EXISTS clause)

➔ select S_Id,S_Name from supervisordetails as sd where exists
(select jd.S_Id from employeeJob ej
inner join JobDetails jd on ej.JobCode=jd.JobCode
where jd.S_Id=sd.S_Id
group by jd.S_Id having count(distinct(ej.empId))>2) ;

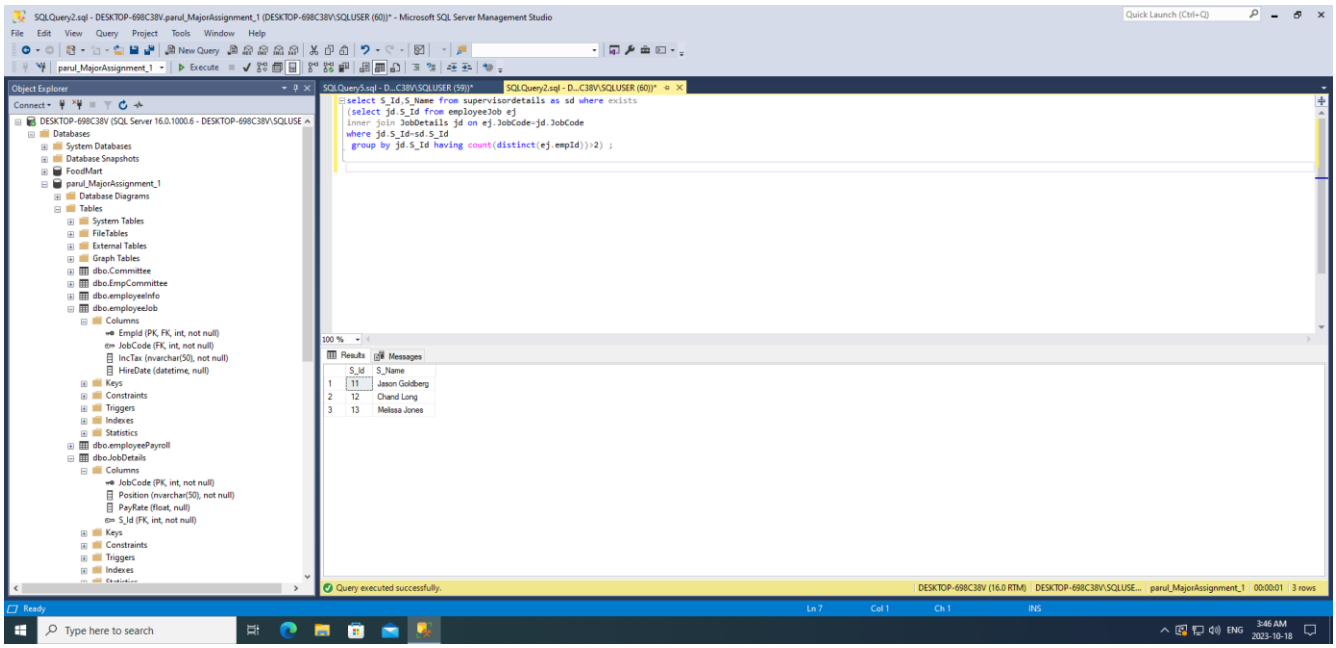

Fig 14. Query to list supervisors with more than two employees.

10) Create a view to show the employee ID, employee's full name, committee ID, meeting night, committee name, supervisor id and supervisor's name, where the condition is the meeting night is 'Tues'. No duplicate records.

➔ Create view CommitteeMeetingForDay as
select ei.EmpId,Concat(ei.FirstName,' ',ei.LastName) as 'Full Name', c.ComId as 'Committee Id',
c.MeetingNight, c.ComName as 'Committee Name', sd.S_Id as 'Supervisor Id',sd.s_Name as 'Supervisor Name'
from committee c
inner join EmpCommittee ec on c.ComId=ec.ComId
inner join employeeInfo ei on ei.EmpId=ec.EmpId
inner join employeeJob ej on ej.EmpId=ei.EmpId
inner join JobDetails jd on jd.JobCode=ej.JobCode
inner join SupervisorDetails sd on sd.S_Id=jd.S_Id;

➔ Select * from CommitteeMeetingForDay where Meetingnight='Mon';

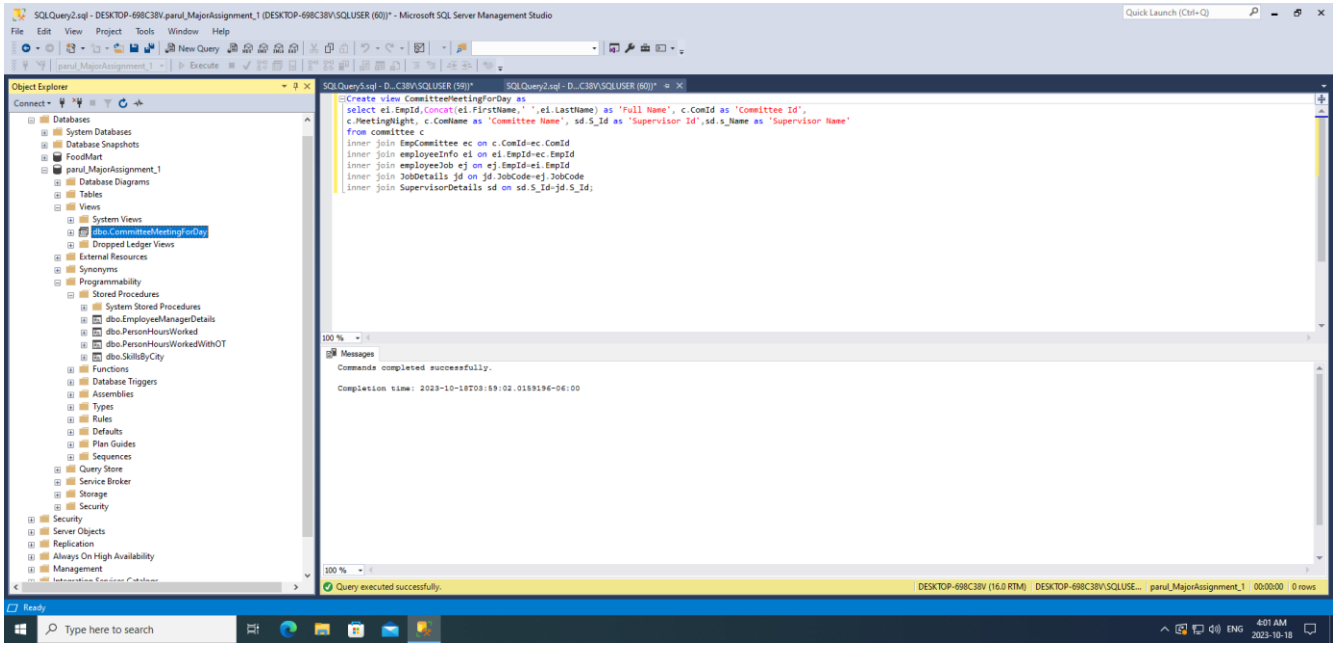

Fig 15. View to show details using MeetingNight filter.

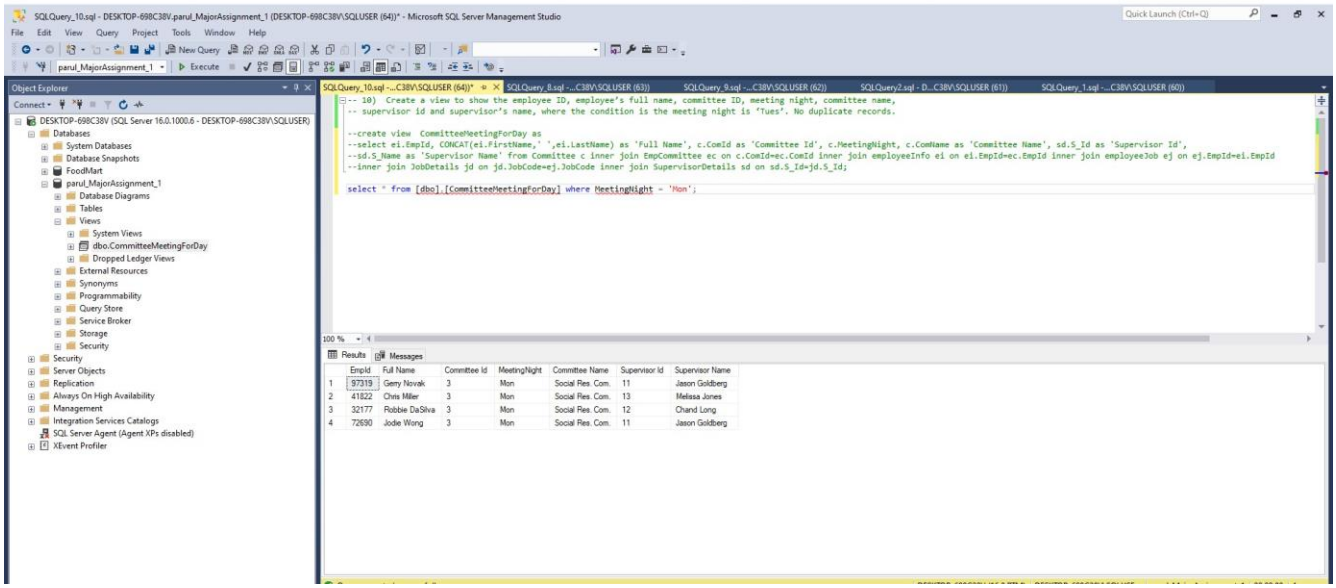

Fig 16. Query to fetch details from the View.