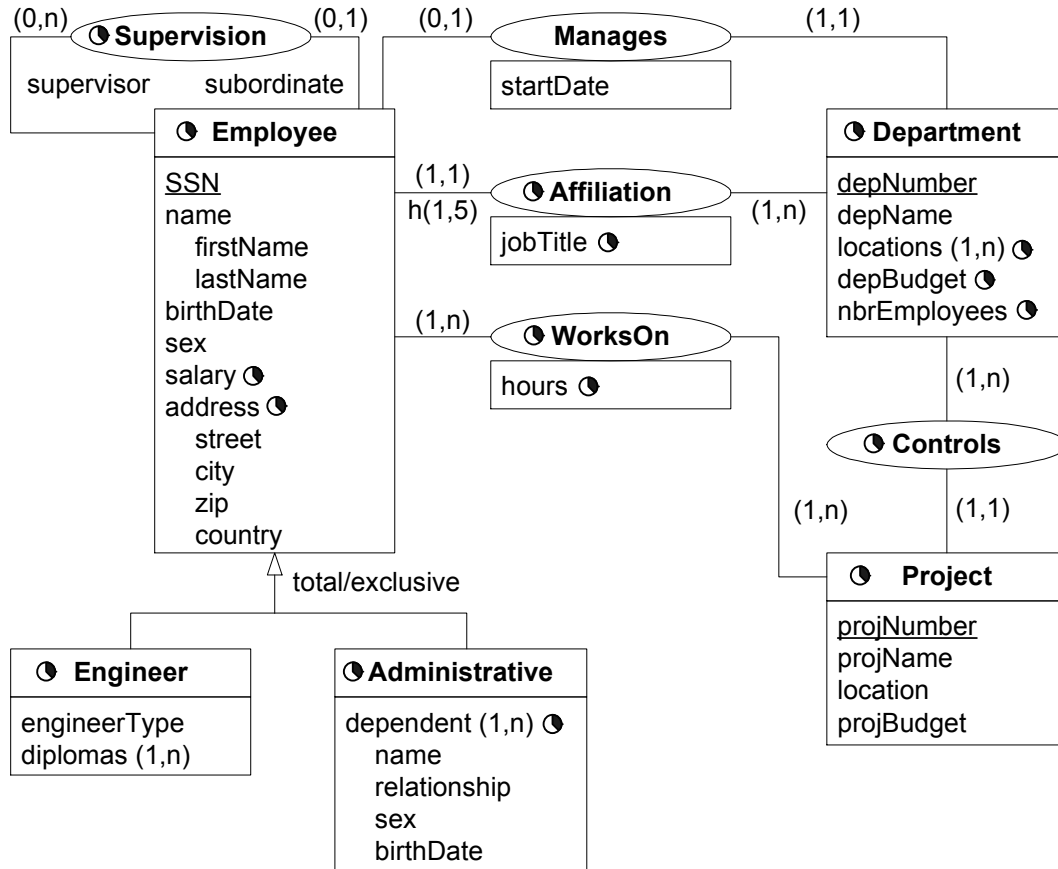


# SQL Server Temporal Databases



Consider the above temporal conceptual schema

- Define a relational schema corresponding to the conceptual schema.
- Ensure the following integrity constraints.
  - (1) An employee works in at most one department at any point in time.
  - (2) At any point in time an employee cannot work more than once in a project.
  - (3) The lifecycle of affiliation is included in the lifecycle of employee
  - (4) The lifecycle of an employee is equal to the union of his/her affiliations.
  - (5) Employees have a contiguous lifecycle.
  - (6) The lifecycle of an employee is equal to the union of his/her affiliations, now taking into account that the lifecycle of employees is contiguous.
- Write the following queries in SQL:
  - (1) Give the name of managers living currently in Houston
  - (2) Give the name of employees working currently in the 'Research' department having a salary greater than 45000
  - (3) Give the name of current employees who does not work currently in any department
  - (4) Give the name of the employee(s) that had the highest salary in 1/1/2002

- (5) Provide the salary and affiliation history for all employees
- (6) Give the name of employees and the period of time in which they were supervisors but did not work in any project during the same period
- (7) Give the name of supervisors who had work on a project at some time
- (8) Give the name of employees and the date they changed their affiliation
- (9) Give the name of employees and the periods they worked on any project
- (10) Give the history of the maximum salary
- (11) Give by department the history of the maximum salary
- (12) Give the history of the number of projects of a department
- (13) Give the name of employees and the periods they worked on all projects of their department

## Relational Schema

- Employee(SSN, FName, MInit, LName, BDate, Sex)
- EmployeeLifecycle(SSN, FromDate, ToDate)  
SSN references Employee(SSN)
- EmployeeSalary(SSN, Salary, FromDate, ToDate)  
SSN references Employee(SSN)
- EmployeeAddress(SSN, Street, City, Zip, Country, FromDate, ToDate)  
SSN references Employee(SSN)
- Engineer(SSN, EngineerType, FromDate, ToDate)  
SSN references Employee(SSN)

In this table the lifecycle of Engineer is kept as well as the attribute EngineerType. There will be redundancy if the lifecycle of Engineer is not continuous.

- EngineerDiplomas(SSN, Diploma)  
SSN references Engineer(SSN)
- AdministrativeLifecycle(SSN, FromDate, ToDate)  
SSN references Employee(SSN)
- AdminDependent(SSN, Name, Relationship, Sex, BirthDate, FromDate, ToDate)  
SSN references AdministrativeLifecycle(SSN)

It is supposed that an employee does not have two dependents of the same name and the same relationship. An alternative will be to put BirthDate instead of Relationship as part of the key.

- Supervision(SSN, SuperSSN, FromDate, ToDate)  
SSN references Employee(SSN)  
SuperSSN references Employee(SSN)
- Affiliation(SSN, JobTitle, DNumber, FromDate, ToDate)  
SSN references Employee(SSN)

It is supposed that the temporality of attribute JobTitle is the same as the lifecycle of the association. In this case two different tables are not necessary. To obtain the lifecycle of the association independently of attribute JobTitle a temporal projection is needed.

- Department(DNumber, DName, MgrSSN, MgrStartDate, FromDate, ToDate)  
MgrSSN references Employee(SSN)

It is supposed that the lifecycle of departments is continuous. In this case an additional table for the lifecycle is not necessary.

- DeptLocations(DNumber, Location, FromDate, ToDate)  
DNumber references Department(DNumber)
- DepartmentBudget(DNumber, DBudget, FromDate, ToDate)  
DNumber references Department(DNumber)
- DepartmentNbEmp(DNumber, NbEmp, FromDate, ToDate)  
DNumber references Department(DNumber)
- Project(PNumber, PName, PLocation, PBudget, FromDate, ToDate)  
It is supposed that the lifecycle of Project is continuous.
- Controls(PNumber, DNumber, FromDate, ToDate)  
PNumber references Project(PNumber)  
DNumber references Department(DNumber)

- WorksOn(ESSN, PNo, Hours, FromDate, ToDate)  
PNo references Project(PNumber)  
ESSN references Employee(SSN)

It is supposed that the temporality of attribute hours is the same as the lifecycle of the association. In this case two different tables are not necessary. To obtain the lifecycle of the association independently of the attribute hours a temporal projection is needed.

# Example Database

Partial schema where not all entities and attributes are taken into account.

| Employee  |          |       |         |            |     | EmployeeLifecycle |            |            |
|-----------|----------|-------|---------|------------|-----|-------------------|------------|------------|
| SSN       | FName    | MInit | LName   | BDate      | Sex | SSN               | FromDate   | ToDate     |
| 123456789 | John     | B     | Smith   | 09-05-1955 | M   | 123456789         | 01-01-1985 | 01-01-2079 |
| 333445555 | Franklin | T     | Wong    | 08-12-1945 | M   | 333445555         | 01-01-1982 | 01-01-2079 |
| 999887777 | Alicia   | J     | Zelaya  | 19-07-1958 | F   | 999887777         | 01-01-1985 | 01-01-2079 |
| 987654321 | Jennifer | S     | Wallace | 20-06-1931 | F   | 987654321         | 01-01-1982 | 01-01-2079 |
| 666884444 | Ramesh   | K     | Narayan | 15-09-1952 | M   | 666884444         | 01-01-1985 | 01-01-2079 |
| 453453453 | Joyce    | A     | English | 31-07-1962 | F   | 453453453         | 01-01-1985 | 01-01-2079 |
| 987987987 | Ahmad    | V     | Jabbar  | 29-03-1959 | M   | 987987987         | 01-01-1985 | 01-01-2079 |
| 888665555 | James    | A     | Borg    | 10-11-1927 | M   | 888665555         | 01-01-1980 | 01-01-2079 |

| EmployeeSalary |        |            |            |
|----------------|--------|------------|------------|
| SSN            | Salary | FromDate   | ToDate     |
| 123456789      | 30000  | 01-01-1985 | 01-01-2079 |
| 333445555      | 40000  | 01-01-1982 | 01-01-1983 |
| 333445555      | 45000  | 01-01-1983 | 01-01-2079 |
| 999887777      | 25000  | 01-01-1985 | 01-01-2079 |
| 987654321      | 43000  | 01-01-1982 | 01-01-2079 |
| 666884444      | 38000  | 01-01-1985 | 01-01-2079 |
| 453453453      | 25000  | 01-01-1985 | 01-01-2079 |
| 987987987      | 25000  | 01-01-1985 | 01-01-2079 |
| 888665555      | 55000  | 01-01-1980 | 01-01-1981 |
| 888665555      | 58000  | 01-01-1981 | 01-01-2079 |

| EmployeeAddress |              |          |      |       |            |            |
|-----------------|--------------|----------|------|-------|------------|------------|
| SSN             | Street       | City     | Zip  | State | FromDate   | ToDate     |
| 123456789       | 731 Fondren  | Houston  | 1000 | TX    | 01-01-1985 | 01-01-2079 |
| 333445555       | 638 Voss     | Houston  | 1000 | TX    | 01-01-1982 | 01-01-2079 |
| 999887777       | 3321 Castle  | Spring   | 1000 | TX    | 01-01-1985 | 01-01-2079 |
| 987654321       | 291 Berry    | Bellaire | 1000 | TX    | 01-01-1982 | 01-01-2079 |
| 666884444       | 975 Fire Oak | Humble   | 1000 | TX    | 01-01-1985 | 01-01-2079 |
| 453453453       | 5631 Rice    | Houston  | 1000 | TX    | 01-01-1985 | 01-01-2079 |
| 987987987       | 980 Dallas   | Houston  | 1000 | TX    | 01-01-1985 | 01-01-2079 |
| 888665555       | 450 Stone    | Houston  | 1000 | TX    | 01-01-1980 | 01-01-2079 |

| Supervision |           |            |            |
|-------------|-----------|------------|------------|
| SSN         | SuperSSN  | FromDate   | ToDate     |
| 123456789   | 333445555 | 01-01-1985 | 01-01-2079 |
| 333445555   | 888665555 | 01-01-1982 | 01-01-2079 |
| 999887777   | 987654321 | 01-01-1985 | 01-01-2079 |
| 987654321   | 888665555 | 01-01-1982 | 01-01-2079 |
| 666884444   | 333445555 | 01-01-1985 | 01-01-2079 |
| 453453453   | 333445555 | 01-01-1985 | 01-01-2079 |
| 987987987   | 987654321 | 01-01-1985 | 01-01-2079 |

| Affiliation |         |            |            |
|-------------|---------|------------|------------|
| SSN         | DNumber | FromDate   | ToDate     |
| 123456789   | 1       | 01-01-1985 | 01-01-1986 |
| 123456789   | 5       | 01-01-1986 | 01-01-2079 |
| 333445555   | 4       | 01-01-1982 | 01-01-1984 |
| 333445555   | 5       | 01-01-1984 | 01-01-2079 |
| 999887777   | 4       | 01-01-1985 | 01-01-2079 |
| 987654321   | 4       | 01-01-1982 | 01-01-2079 |
| 666884444   | 5       | 01-01-1985 | 01-01-2079 |
| 453453453   | 5       | 01-01-1985 | 01-01-2079 |
| 987987987   | 4       | 01-01-1985 | 01-01-2079 |
| 888665555   | 1       | 01-01-1980 | 01-01-2079 |

Department

| <u>DNumber</u> | DName          | MgrSSN    | MgrStartDate | FromDate   | ToDate     |
|----------------|----------------|-----------|--------------|------------|------------|
| 1              | Headquarters   | 888665555 | 19-06-1980   | 01-01-1980 | 01-01-2079 |
| 4              | Administration | 987654321 | 01-01-1982   | 01-01-1981 | 01-01-2079 |
| 5              | Research       | 333445555 | 22-05-1984   | 01-01-1982 | 01-01-2079 |

DepartmentNbEmp

| <u>DNumber</u> | NbEmp | FromDate   | ToDate     |
|----------------|-------|------------|------------|
| 5              | 4     | 01-01-1980 | 01-01-2079 |
| 4              | 3     | 01-01-1980 | 01-01-2079 |
| 1              | 1     | 01-01-1980 | 01-01-2079 |

DeptLocations

| <u>DNumber</u> | DLocation | FromDate   | ToDate     |
|----------------|-----------|------------|------------|
| 1              | Houston   | 01-01-1980 | 01-01-2079 |
| 4              | Stafford  | 01-01-1980 | 01-01-2079 |
| 5              | Bellaire  | 01-01-1980 | 01-01-2079 |
| 5              | Sugarland | 01-01-1980 | 01-01-2079 |
| 5              | Houston   | 01-01-1980 | 01-01-2079 |

Project

| <u>PNumber</u> | PName           | PLocation | FromDate   | ToDate     |
|----------------|-----------------|-----------|------------|------------|
| 1              | ProductX        | Bellaire  | 01-01-1980 | 01-01-2079 |
| 2              | ProductY        | Sugarland | 01-01-1980 | 01-01-2079 |
| 3              | ProductZ        | Houston   | 01-01-1980 | 01-01-2079 |
| 10             | Computerization | Stafford  | 01-01-1980 | 01-01-2079 |
| 20             | Reorganization  | Houston   | 01-01-1980 | 01-01-2079 |
| 30             | Newbenefits     | Stafford  | 01-01-1980 | 01-01-2079 |

Controls

| <u>PNumber</u> | DNumber | FromDate   | ToDate     |
|----------------|---------|------------|------------|
| 1              | 5       | 01-01-1980 | 01-01-2079 |
| 2              | 5       | 01-01-1980 | 01-01-2079 |
| 3              | 5       | 01-01-1980 | 01-01-2079 |
| 10             | 4       | 01-01-1980 | 01-01-2079 |
| 20             | 1       | 01-01-1980 | 01-01-2079 |
| 30             | 4       | 01-01-1980 | 01-01-2079 |

WorksOn

| <u>ESSN</u> | <u>PNo</u> | Hours | <u>FromDate</u> | <u>ToDate</u> |
|-------------|------------|-------|-----------------|---------------|
| 123456789   | 1          | 32.5  | 01-01-1980      | 01-01-2079    |
| 123456789   | 2          | 7.5   | 01-01-1980      | 01-01-2079    |
| 333445555   | 1          | 10    | 01-01-1980      | 01-01-2079    |
| 333445555   | 2          | 10    | 01-01-1980      | 01-01-2079    |
| 333445555   | 3          | 20    | 01-01-1980      | 01-01-2079    |
| 453453453   | 1          | 20    | 01-01-1980      | 01-01-2079    |
| 453453453   | 2          | 20    | 01-01-1980      | 01-01-2079    |
| 666884444   | 3          | 40    | 01-01-1980      | 01-01-2079    |
| 888665555   | 20         | 30.0  | 01-01-1980      | 01-01-2079    |
| 987654321   | 10         | 5.0   | 01-01-1980      | 01-01-2079    |
| 987654321   | 20         | 15.0  | 01-01-1980      | 01-01-2079    |
| 987654321   | 30         | 20.0  | 01-01-1980      | 01-01-2079    |
| 987987987   | 10         | 35.0  | 01-01-1980      | 01-01-2079    |
| 987987987   | 30         | 5.0   | 01-01-1980      | 01-01-2079    |
| 999887777   | 10         | 10.0  | 01-01-1980      | 01-01-2079    |
| 999887777   | 30         | 30.0  | 01-01-1980      | 01-01-2079    |

## Constraints

- (1) An employee works in at most one department at any point in time.  
In other terms SSN is a sequenced primary key for Affiliation.

```
create trigger Seq_PK_Affiliation on Affiliation
  after insert, update as
if exists ( select * from Inserted A1
  where 1 < ( select count(*) from Affiliation A2
    where A1.SSN = A2.SSN
    and A1.FromDate < A2.ToDate and A2.FromDate < A1.ToDate ) )
begin
  raiserror 13000
  'An employee works in at most one department at any point in time'
  rollback transaction
end
```

- (2) At any point in time an employee cannot work more than once in a project.  
In other terms (ESSN,PNo) is a sequenced primary key for WorksOn

```
create trigger Seq_PK_WorksOn on WorksOn
  after insert, update as
if exists ( select * from Inserted W1
  where 1 < ( select count(*) from WorksOn W2
    where W1.ESSN = W2.ESSN and W1.PNo = W2.PNo
    and W1.FromDate < W2.ToDate and W2.FromDate < W1.ToDate ) )
begin
  raiserror 13000
  'At any point in time an employee cannot work more than once in a project'
  rollback transaction
end
```

- (3) The lifecycle of affiliation is included in the lifecycle of employee.  
In the following triggers it is assumed that the table EmployeeLifecycle is coalesced.  
Therefore, every line in Affiliation must be covered by one line in EmployeeLifecycle.

```
create trigger Seq_FK_Affiliation_EmployeeLifecycle_1 on Affiliation
  after insert, update as
if exists ( select * from Inserted A
  where not exists ( select * from EmployeeLifecycle E
    where A.SSN = E.SSN
    and E.FromDate <= A.FromDate and A.ToDate <= E.ToDate ) )
begin
  raiserror 13000
  'The lifecycle of affiliation must be included in the lifecycle of employee'
  rollback transaction
end
```

```
create trigger Seq_FK_Affiliation_EmployeeLifecycle_2 on EmployeeLifecycle
  after update, delete as
if exists ( select * from Affiliation A
  where A.SSN IN ( select SSN from Deleted)
  and not exists ( select * from EmployeeLifecycle E
    where A.SSN = E.SSN
    and E.FromDate <= A.FromDate and A.ToDate <= E.ToDate ) )
begin
  raiserror 13000
```

```

    'The lifecycle of affiliation must be included in the lifecycle of employee'
    rollback transaction
end

```

- (4) The lifecycle of an employee is equal to the union of his/her affiliations. It is supposed that the previous trigger is activated, therefore it is sufficient to monitor that an employee must be affiliated to a department throughout his/her lifecycle.

```

create trigger Seq_FK_EmployeeLifecycle_Affiliation_1 on Affiliation
after update, delete as
if exists ( select * from EmployeeLifecycle E
where E.SSN in ( select SSN from Deleted )
and not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate <= E.FromDate and E.FromDate < A.ToDate )
or not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate < E.ToDate and E.ToDate <= A.ToDate )
or exists ( select * from Affiliation A
where E.SSN = A.SSN
and E.FromDate < A.ToDate and A.ToDate < E.ToDate
and not exists ( select * from Affiliation A2
where A2.SSN = A.SSN
and A2.FromDate <= A.ToDate and A.ToDate < A2.ToDate ) ) )
begin
raiserror 13000
'An employee must be affiliated to a department throughout his/her lifecycle'
rollback transaction
end

```

```

create trigger Seq_FK_EmployeeLifecycle_Affiliation_2 on EmployeeLifecycle
after insert, update as
if exists ( select * from Inserted E
where not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate <= E.FromDate and E.FromDate < A.ToDate )
or not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate < E.ToDate and E.ToDate <= A.ToDate )
or exists ( select * from Affiliation A
where E.SSN = A.SSN
and E.FromDate < A.ToDate and A.ToDate < E.ToDate
and not exists ( select * from Affiliation A2
where A2.SSN = A.SSN
and A2.FromDate <= A.ToDate and A.ToDate < A2.ToDate ) ) )
begin
raiserror 13000
'An employee must be affiliated to a department throughout his/her lifecycle'
rollback transaction
end

```

- (5) Employees have a contiguous lifecycle.

```

alter table EmployeeLifecycle
drop constraint PK_EmployeeLifecycle
alter table EmployeeLifecycle
add constraint PK_EmployeeLifecycle primary key (SSN)

```



- (6) The lifecycle of an employee is equal to the union of his/her affiliations, now taking into account that the lifecycle of employees is contiguous.

It is necessary to ensure that (1) the affiliations of an employee define a contiguous history, and (2) an employee must be affiliated to a department throughout his/her lifecycle.

The following trigger ensures the affiliations of an employee define a contiguous history.

```
create trigger Contiguous_Hist_Affiliation on Affiliation
after insert, update, delete as
if exists ( select * from Affiliation A1, Affiliation A2
where A1.SSN = A2.SSN and A1.ToDate < A2.FromDate
and not exists ( select * from Affiliation A3
where A1.SSN = A3.SSN
and ( ( A3.FromDate <= A1.ToDate and A1.ToDate < A3.ToDate )
or ( A3.FromDate < A2.FromDate and A2.FromDate <= A3.ToDate ) ) ) )
begin
raiserror 13000
'The affiliations of an employee define a contiguous history'
rollback transaction
end
```

The following two triggers replaces those of question (4).

```
alter trigger Seq_FK_EmployeeLifecycle_Affiliation_1 on Affiliation
after update, delete as
if exists ( select * from EmployeeLifecycle E
where E.SSN in ( select SSN from Deleted )
and not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate <= E.FromDate and E.FromDate < A.ToDate )
or not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate < E.ToDate and E.ToDate <= A.ToDate ) ) )
begin
raiserror 13000
'An employee must be affiliated to a department throughout his/her lifecycle'
rollback transaction
end
```

```
alter trigger Seq_FK_EmployeeLifecycle_Affiliation_2 on EmployeeLifecycle
after insert, update as
if exists ( select * from Inserted E
where not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate <= E.FromDate and E.FromDate < A.ToDate )
or not exists ( select * from Affiliation A
where E.SSN = A.SSN
and A.FromDate < E.ToDate and E.ToDate <= A.ToDate ) ) )
begin
raiserror 13000
'An employee must be affiliated to a department throughout his/her lifecycle'
rollback transaction
end
```

## Queries

- (1) Give the name of managers living currently in Houston

```
select E.FName, E.LName
from Employee E, EmployeeAddress A, Department D
where E.SSN = A.SSN and E.SSN = D.MgrSSN
and A.City = 'Houston'
and A.FromDate <= getdate() and getdate() < A.ToDate
and D.FromDate <= getdate() and getdate() < D.ToDate
```

- (2) Give the name of employees working currently in the 'Research' department having a salary greater than 45000

```
select E.FName, E.LName
from Employee E, EmployeeSalary S, Affiliation A, Department D
where E.SSN = S.SSN and E.SSN = A.SSN and A.DNumber = D.DNumber
and D.DName = 'Research' and S.Salary > 45000
and S.FromDate <= getdate() and getdate() < S.ToDate
and A.FromDate <= getdate() and getdate() < A.ToDate
```

- (3) Give the name of current employees who does not work currently in any department

```
select distinct E.FName, E.LName
from Employee E, EmployeeLifecycle L
where E.SSN = L.SSN
and L.FromDate <= getdate() and getdate() < L.ToDate
and not exists (
    select * from Affiliation A
    where E.SSN = A.SSN and
    and A.FromDate <= getdate() and getdate() < A.ToDate )
```

- (4) Give the name of the employee(s) that had the highest salary in 1/1/2002

```
select E.FName, E.LName
from Employee E, EmployeeSalary E1
where E.SSN = E1.SSN
and salary = ( select max(salary) from EmployeeSalary S
                where A.FromDate <= '2002-01-01' and '2002-01-01' < A.ToDate )
and E1.FromDate <= '2002-01-01' and '2002-01-01' < E1.ToDate )
```

- (5) Provide the salary and affiliation history for all employees

```
create function first_instant
    (@one smalldatetime, @two smalldatetime)
returns smalldatetime as
begin
    return CASE WHEN @one < @two then @one else @two end
end
go
create function last_instant
    (@one smalldatetime, @two smalldatetime)
returns smalldatetime as
begin
    return CASE WHEN @one > @two then @one else @two end
end
go
select E.FName, E.LName, D.DepName, S.Salary,
    'Start Date' = dbo.last_instant(S.FromDate,A.FromDate),
```

```

    'End Date'= dbo.first_instant(S.ToDate,A.ToDate)
from Employee E, EmployeeSalary S, Affiliation A, Department D
where E.SSN = S.SSN and E.SSN = A.SSN and A.DepNumber = D.DepNumber
and dbo.last_instant(S.FromDate,A.FromDate) <
    dbo.first_instant(S.ToDate,A.ToDate)
order by E.FName, E.LName

```

- (6) Give the name of employees and the period of time in which they were supervisors but did not work in any project during the same period

```

--Case 1
select S.SSN, S.FromDate, W1.FromDate as ToDate
from Supervision S, WorksOn W1
where S.SSN = W1.ESSN and S.FromDate < W1.FromDate
and not exists ( select * from WorksOn W2 where S.SSN = W2.ESSN
    and S.FromDate < W2.ToDate and W2.FromDate < W1.FromDate )
union
--Case 2
select S.SSN, W1.ToDate as FromDate, S.ToDate
from Supervision S, WorksOn W1
where S.SSN = W1.ESSN and W1.ToDate < S.ToDate
and not exists ( select * from WorksOn W2 where S.SSN = W2.ESSN
    and W1.ToDate < W2.ToDate and W2.FromDate < S.ToDate )
union
--Case 3
select S.SSN, W1.ToDate as FromDate, W2.FromDate as ToDate
from Supervision S, WorksOn W1, WorksOn W2
where S.SSN = W1.ESSN and S.SSN = W2.ESSN and W1.ToDate < W2.FromDate
and S.FromDate < W1.ToDate and W2.FromDate < S.ToDate
and not exists ( select * from WorksOn W3 where S.SSN = W3.ESSN
    and W1.ToDate < W3.ToDate and W3.FromDate < W2.FromDate )
union
--Case 4
select SSN, FromDate, ToDate from Supervision S
where not exists ( select * from WorksOn W where S.SSN=W.ESSN
    and S.FromDate < W.ToDate and W.FromDate < S.ToDate )

```

- (7) Give the name of supervisors who had work on a project at some time

```

select distinct E.FName, E.LName
from Employee E, Supervision S, WorksOn W
where E.SSN = S.SSN and E.SSN = W.ESSN

```

- (8) Give the name of employees and the date they changed their affiliation

```

select distinct E.FName, E.LName, A1.ToDate
from Employee E, Affiliation A1, Affiliation A2
where E.SSN = A1.SSN and E.SSN = A2.SSN
and A1.ToDate = A2.FromDate and A1.DNumber <> A2.DNumber

```

- (9) Give the name of employees and the periods they worked on any project

```

create procedure EmployeeProject as
create table #Temp ( ESSN char(9), FName varchar(15), LName varchar(15),
    FromDate smalldatetime, ToDate smalldatetime )
insert into #Temp
select E.FName, E.LName, E.SSN, W.FromDate, W.ToDate
from Employee E, WorksOn W
where E.SSN = W.ESSN

```

```

select distinct F.ESSN, F.FName, F.LName, F.FromDate, L.ToDate
from #Temp F, #Temp L
where F.ESSN = L.ESSN and F.FromDate < L.ToDate
and not exists ( select * from #Temp M
  where M.ESSN = F.ESSN
  and F.FromDate < M.FromDate and M.FromDate <= L.ToDate
  and not exists ( select * from #Temp T1
    where T1.ESSN = F.ESSN
    and T1.FromDate < M.FromDate and M.FromDate <= T1.ToDate ) )
and not exists ( select * from #Temp T2
  where T2.ESSN = F.ESSN
  and ( ( T2.FromDate < F.FromDate and F.FromDate <= T2.ToDate )
    or ( T2.FromDate <= L.ToDate and L.ToDate < T2.ToDate ) ) )

```

(10) Give the history of the maximum salary

```

create procedure HistoryMaximumSalary as
-- First step: Construct intervals during which no salary change occurred
create table #Instants(Instant smalldatetime)
insert into #Instants
  select distinct E.FromDate from EmployeeSalary E
  union select distinct E.ToDate from EmployeeSalary E
create table #Intervals(FromDate smalldatetime, ToDate smalldatetime)
insert into #Intervals
  select I1.Instant, I2.Instant
  from #Instants I1, #Instants I2
  where I1.Instant < I2.Instant
  and not exists ( select * from #Instants I3
    where I1.Instant < I3.Instant
    and I3.Instant < I2.Instant )
-- Second step: Compute the maximum salary for these intervals
create table #TempMax(SalaryMax decimal(18,2),
  FromDate smalldatetime, ToDate smalldatetime)
insert into #TempMax
  select max(E.Salary) as SalaryMax, I.FromDate, I.ToDate
  from EmployeeSalary E, #Intervals I
  where E.FromDate <= I.FromDate and I.ToDate <= E.ToDate
  group by I.FromDate, I.ToDate
-- Third step: Coalescing the above table
select distinct F.SalaryMax, F.FromDate, L.ToDate
from #TempMax F, #TempMax L
where F.FromDate < L.ToDate and F.SalaryMax = L.SalaryMax
and not exists ( select *
  from #TempMax M
  where M.SalaryMax = F.SalaryMax
  and F.ToDate < M.FromDate and M.FromDate <= L.FromDate
  and not exists ( select *
    from #TempMax T1
    where T1.SalaryMax = F.SalaryMax
    and T1.FromDate < M.FromDate and M.FromDate <= T1.ToDate ) )
and not exists ( select *
  from #TempMax T2
  where T2.SalaryMax = F.SalaryMax
  and ( ( T2.FromDate < F.FromDate and F.FromDate <= T2.ToDate )
    or ( T2.FromDate <= L.ToDate and L.ToDate < T2.ToDate ) ) )
order by F.FromDate

```

(11) Give by department the history of the maximum salary

```
create procedure HistMaxSalaryDept as
-- First step: Construct by department the intervals during
-- which the maximum salary must be calculated.
create table #Aff_Sal(DNumber int, Salary decimal(18,2),
   FromDate smalldatetime, ToDate smalldatetime)
insert into #Aff_Sal
select distinct A.DNumber, S.Salary,
    dbo.last_instant(S.FromDate,A.FromDate),
    dbo.first_instant(S.ToDate,A.ToDate)
from Affiliation A, EmployeeSalary S
where A.SSN = S.SSN
and dbo.last_instant(S.FromDate,A.FromDate) <
    dbo.first_instant(S.ToDate,A.ToDate)
create table #SalChanges(DNumber int, Instant smalldatetime)
insert into #SalChanges
select distinct DNumber, FromDate from #Aff_Sal
union select distinct DNumber, ToDate from #Aff_Sal
create table #SalIntervals(DNumber int, FromDate smalldatetime,
    ToDate smalldatetime)
insert into #SalIntervals
select P1.DNumber, P1.Instant, P2.Instant
from #SalChanges P1, #SalChanges P2
where P1.DNumber=P2.DNumber and P1.Instant<P2.Instant
and not exists ( select * from #SalChanges P3
    where P1.DNumber = P3.DNumber and P1.Instant < P3.Instant
    and P3.Instant < P2.Instant )
-- Second step: Compute the maximum salary for the
-- above periods.
create table #TempMaxDep(DNumber int, MaxSalary decimal(18,2),
    FromDate smalldatetime, ToDate smalldatetime)
insert into #TempMaxDep
select P.DNumber, max(Salary), P.FromDate, P.ToDate
from #Aff_Sal A, #SalIntervals P
where A.DNumber = P.DNumber
and A.FromDate <= P.FromDate and P.ToDate <= A.ToDate
group by P.DNumber, P.FromDate, P.ToDate
-- Third step: Coalescing the above table
select distinct F.DNumber, F.MaxSalary, F.FromDate, L.ToDate
from #TempMaxDep F, #TempMaxDep L
where F.DNumber = L.DNumber and F.MaxSalary = L.MaxSalary
and F.FromDate < L.ToDate
and not exists ( select *
    from #TempMaxDep M
    where F.DNumber = M.DNumber and F.MaxSalary = M.MaxSalary
    and F.ToDate < M.FromDate and M.FromDate <= L.FromDate
    and not exists ( select *
        from #TempMaxDep T1
        where F.DNumber = T1.DNumber and F.MaxSalary = T1.MaxSalary
        and T1.FromDate < M.FromDate and M.FromDate <= T1.ToDate ) )
and not exists ( select *
    from #TempMaxDep T2
    where F.DNumber = T2.DNumber and F.MaxSalary = T2.MaxSalary
    and ( ( T2.FromDate < F.FromDate and F.FromDate <= T2.ToDate )
        or ( T2.FromDate <= L.ToDate and L.ToDate < T2.ToDate ) ) )
```

```
order by F.DNumber, F.FromDate
```

(12) Give the history of the number of projects of a department

```
create procedure HistoryNumberProjects as
-- First step: Construct intervals during which the number of
-- projects of a department does not change
create table #Instants(DNumber int, Instant smalldatetime)
insert into #Instants
  select distinct DNumber, FromDate from Controls
  union
  select distinct DNumber, ToDate from Controls
create table #Intervals(DNumber int, FromDate smalldatetime,
  ToDate smalldatetime)
insert into #Intervals
  select I1.DNumber, I1.Instant, I2.Instant
  from Instants I1, Instants I2
  where I1.DNumber = I2.DNumber
  and I1.Instant < I2.Instant
  and not exists ( select * from Instants I3
    where I1.DNumber = I3.DNumber
    and I1.Instant < I3.Instant
    and I3.Instant < I2.Instant )
-- Second step: Compute the number of projects for these intervals
create table #TempCountDep(DNumber int, NbProjects int,
  FromDate smalldatetime, ToDate smalldatetime)
insert into #TempCountDep
  select I.DNumber, count(I.PNumber), I.FromDate, I.ToDate
  from Controls C, #Intervals I
  where C.DNumber = I.DNumber
  and ( C.FromDate <= I.FromDate and I.ToDate <= C.ToDate )
  group by I.DNumber, I.FromDate, I.ToDate
-- Third step: Coalescing the above table
select distinct F.DNumber, F.NbProjects, F.FromDate, L.ToDate
from #TempCountDep F, #TempCountDep L
where F.DNumber = L.DNumber and F.FromDate < L.ToDate
and F.NbProjects = L.NbProjects
and not exists ( select *
  from #TempCountDep M
  where M.DNumber = F.DNumber and M.NbProjects = F.NbProjects
  and F.ToDate < M.FromDate and M.FromDate <= L.FromDate
  and not exists ( select *
    from #TempCountDep T1
    where T1.DNumber = F.DNumber and T1.NbProjects = F.NbProjects
    and T1.FromDate < M.FromDate and M.FromDate <= T1.ToDate ) )
and not exists ( select *
  from #TempCountDep T2
  where T2.DNumber = F.DNumber and T2.NbProjects = F.NbProjects
  and ( ( T2.FromDate < F.FromDate and F.FromDate <= T2.ToDate )
    or ( T2.FromDate <= L.ToDate and L.ToDate < T2.ToDate ) ) )
order by F.DNumber, F.FromDate
```

(13) Give the name of employees and the periods they worked on all projects of their department

```
create procedure HistoryAllProjects as
-- First step: Construct intervals during which the number of projects
```

```

-- of an employee does not change
create table #Aff_Cont(SSN char(9), DNumber int, PNumber int,
   FromDate smalldatetime, ToDate smalldatetime)
insert into #Aff_Cont
select distinct A.SSN, A.DNumber, C.PNumber,
    dbo.last_instant(A.FromDate,C.FromDate),
    dbo.first_instant(A.ToDate,C.ToDate)
from Affiliation A, Controls C
where A.DNumber = C.DNumber
and dbo.last_instant(A.FromDate,C.FromDate) <
    dbo.first_instant(A.ToDate,C.ToDate)
create table #Aff_Cont_WO(SSN char(9), DNumber int, PNumber int,
   FromDate smalldatetime, ToDate smalldatetime)
insert into #Aff_Cont_WO
select distinct A.SSN, A.DNumber, W.PNumber,
    dbo.last_instant(A.FromDate,W.FromDate),
    dbo.first_instant(A.ToDate,W.ToDate)
from #Aff_Cont A, WorksOn W
where A.PNumber = W.PNumber and A.SSN = W.SSN
and dbo.last_instant(A.FromDate,W.FromDate) <
    dbo.first_instant(A.ToDate,W.ToDate)
create table #ProjChanges(SSN char(9), DNumber int,
    Instant smalldatetime)
insert into #ProjChanges
select distinct SSN, DNumber, FromDate from #Aff_Cont
union select distinct SSN, DNumber, ToDate from #Aff_Cont
union select distinct SSN, DNumber, FromDate from #Aff_Cont_WO
union select distinct SSN, DNumber, ToDate from #Aff_Cont_WO
union select SSN, DNumber, FromDate from Affiliation
union select SSN, DNumber, ToDate from Affiliation
create table #ProjIntervals(SSN char(9), DNumber int,
    FromDate smalldatetime, ToDate smalldatetime)
insert into #ProjIntervals(
select P1.SSN, P1.DNumber, P1.Instant, P2.Instant
from #ProjChanges P1, #ProjChanges P2
where P1.SSN = P2.SSN
and P1.DNumber = P2.DNumber and P1.Instant < P2.Instant
and not exists ( select * from #ProjChanges P3
    where P1.SSN = P3.SSN and P1.DNumber = P3.DNumber
    and P1.Instant < P3.Instant and P3.Instant < P2.Instant )
-- Second step: Compute the number of projects for these intervals
create table #TempUnivQuant(SSN char(9), FromDate smalldatetime,
    ToDate smalldatetime)
insert into #TempUnivQuant
select distinct P.SSN, P.FromDate, P.ToDate
from #ProjIntervals P
where not exists ( select * from Controls C
    where P.DNumber = C.DNumber
    and C.FromDate <= P.FromDate and P.ToDate <= C.ToDate
    and not exists ( select * from WorksOn W
        where C.PNumber = W.PNumber and P.SSN = W.SSN
        and W.FromDate <= P.FromDate and P.ToDate <= W.ToDate ) )
-- Third step: Coalescing the above table
select distinct F.SSN, F.FromDate, L.ToDate
from #TempUnivQuant F, #TempUnivQuant L

```

```

where F.SSN = L.SSN and F.FromDate < L.ToDate
and not exists ( select *
  from #TempUnivQuant M
  where M.SSN = F.SSN
  and F.ToDate < M.FromDate and M.FromDate <= L.FromDate
  and not exists ( select *
    from #TempUnivQuant T1
    where T1.SSN = F.SSN
    and T1.FromDate < M.FromDate and M.FromDate <= T1.ToDate ) )
and not exists ( select *
  from #TempUnivQuant T2
  where T2.SSN = F.SSN
  and ( ( T2.FromDate < F.FromDate and F.FromDate <= T2.ToDate )
    or ( T2.FromDate <= L.ToDate and L.ToDate < T2.ToDate ) ) )
order by F.SSN, F.FromDate

```