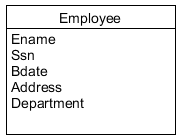
System Development 2

**Exercise (case): Update Anomalies**

### 

### Design 1: Table EMP\_DEPT

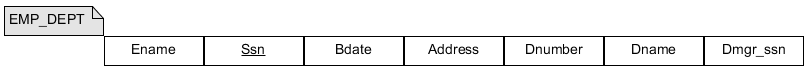
In a domain model we have the concept of an Employee – as shown below:



The Department attribute represents the department that the employee is part of.

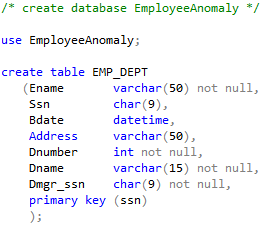
When we’re to transform to a relational database schema we realize that the Department attribute is a composite attribute that consist of number, name and manager id.

Somehow we end up with this relational database schema:

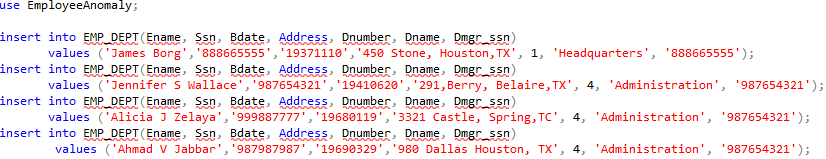


Cf. [Elmasri] fig. 14.11 (b)

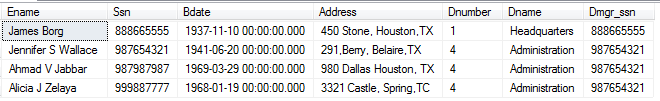
As we bypass normalization we implement the EMP\_DEPT table by this sql script:



And insert some data:



After the insertion the content is:



***Exercise 1***

Consider and evaluate these manipulations:

1. Insert another employee in department 4
2. Insert a new employee with unknown department
3. Insert a new department with these data:  
   5, 'Research', '333445555'
4. Delete ‘James Borg’

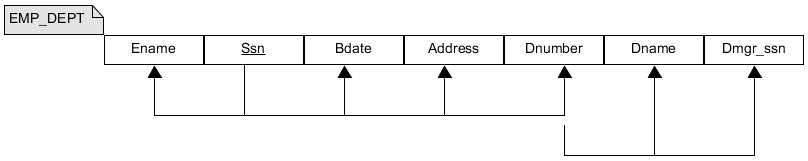
What would happen / is it possible?

What are the consequences?

### Design 2: Decompose table EMP\_DEPT

We find working with the EMP\_DEPT table cumbersome and decide to remedy the problems.

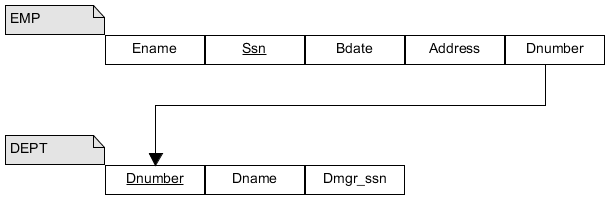
We certainly could modify the domain model, but instead we normalize the relational database schema.  
  
We examine the functional dependencies (FD) in EMP\_DEPT:



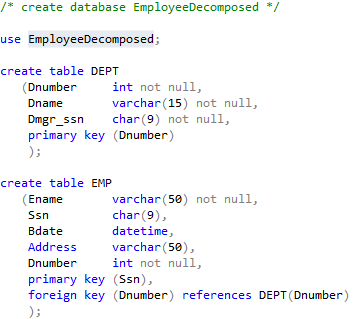
[Elmasri] fig. 14.11 (b)

The schema shows that Ssn determines the values of Ename, Bdate, Address and Dnumber, while Dnumber determines the values of Dname and Dmgr\_ssn.

On that basis this relational database schema is suggested:



We can implement the EMP and DEPT tables by this sql script:



***Exercise 2***

Based on the suggested schema and sql script consider and evaluate the same manipulations as you did in exercise 1 (assume similar data as previously):

1. Insert another employee in department 4
2. Insert a new employee with unknown department
3. Insert a new department with these data:  
   5, 'Research', '333445555'
4. Delete ‘James Borg’

What would happen / is it possible?

What are the consequences?

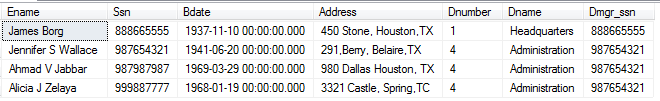
***Exercise 3***

We want to meet these principles:

* Information preservation
  + no data must be lost
  + avoid adding spurious data
* Minimum redundancy
  + avoid update anomalies
  + safe space

It’s now your job to substantiate that the suggested schema meets the information preservation principle!

Do that by example:

1. Implement a database based on the suggested schema
2. Insert data like the data previously inserted in EMP\_DEPT table  
   (2 rows in DEPT and then 4 rows in EMP)
3. Create the exact same information as you had in the EMP\_DEPT table, by joining the EMP and DEPT tables. Result should be:  
   

(If you want to walk through the whole case you can create the EmployeeAnomaly database incl. the EMP\_DEPT table and its content first).

**Resources to help:**

The file *SqlUpdateAnomalies.zip* contain a sql-script to create the EmployeeDecomposed database incl. the tables EMP and DEPT.

It also contains the scripts concerning the EMP\_DEPT table. They can be used as inspiration or to recreate the whole case.