7. Normalization

7.2.1 Violation of 1NF

Consider the relation schema:
Employees(EmpNo, Jobs)

where Jobs is a multivalued attribute containing one or more jobs.
Make changes needed to ensure 1NF.

Solution:

Normalization to 1NF:
Employees(EmpNo, Job)
The attribute Job is made part of the Primary Key, and this will allow an employee to have any number of jobs.

7.2.2 Violation of 2NF

Consider the relation schema:
Clients(ClientNo, SalesRepNo, CName, SName, Date)
where ClientNo and CName is the number and name of a client, SalesRepNo and SName is the number and name of a sales representative, and Date is the date where the sales representative was assigned to the client.

Explain why *Clients* is not in 2NF. Hint: First identify the minimal, non-trivial functional dependences. Normalize to 2NF.

Solution:

We identify the following functional dependencies:

ClientNo -> CName,

SalesRepNo -> SName,

ClientNo SalesRepNo -> Date

We assume that the relation between clients and sales representatives is many to many (otherwise *ClientNo, SalesRepNo* would not both have been needed in the primary key). So we neither have ClientNo -> SalesRepNo nor SalesRepNo -> ClientNo

Clients is not in 2NF as two non-key attributes, Cname and Sname, depend partly on the primary key: CName does not depend on the Full Primary Key: it depends only on ClientNo, but not on SalesRepNo. Likewise, SName only depends on SalesRepNo, but not on ClientNo. The attribute Date is OK, since it depends on the full primary key.

Normalization to 2NF:
SalesPersons(SalesRepNo, SName)
Clients(ClientNo, CName)
ClientContacts(ClientNo, SalesRepNo, Date)

7.2.3 Violation of 3NF

Consider the relation schema:
Winners(Tournament, Year, Winner, Birthday)
where each row in the table tells who was the Winner of a particular Tournament in a particular Year. The Birthday of the Winner is also given.
Explain why Winners is not in 3NF and why that is a problem. Is Winners in 2NF? Normalize Winners to 3NF.

Solution:
We assume the following, minimal functional dependencies:
Tournament Year -> Winner,
Winner -> Birthday

Winners **is in 2NF** as no non-primary-key attribute, i.e. neither Winner nor Birthday, depend only on either just Tournament or Year – they depend on the complete key.

Winners is **not in 3NF** as there is a transitive dependency of Birthday on the primary key:
Tournament Year -> Winner -> Birthday
This is a problem because the same person can appear as winner in several rows with different dates of birth.

Normalization to 3NF gives:
Winner1(Tournament, Year, Winner)
Winner2(Winner, Birthday)

Now a winner can only have one birthday. ☺

7.2.4 Violation of 4NF

We are told that in a company, an employee can work on many projects and be employed in many departments. Departments as well as projects can have many employees.

Consider the relation schema:
Company(<u>EmpNo</u>, <u>DepNo</u>, <u>ProjectNo</u>)
with the dependency:
EmpNo →→ ProjectNo
Explain why Company is not in 4NF.

Normalize Company to 4NF.

Solution:

The information EmpNo →→ ProjectNo means that for each of the departments (DepNo) an employee is employed in, the set of projects (given by ProjectNo) the employee is participating in, should be the same.

Company is not in 4NF, as EmpNo →→ ProjectNo, but EmpNo is not a key.
Normalization to 4NF:
Employees(<u>EmpNo</u>, <u>DepNo</u>)
Works(<u>EmpNo</u>, <u>ProjectNo</u>)

7.2.5 Functional Dependencies

How many functional dependencies can be proposed for the schema: R(A, B, C, D, E)?

Solution:
Of a set of 5 attributes, 2^5 = 32 subsets can be made like {}, {D}, {B, E}, {A, C, E}, {B, C, D, E} and {A, B, C, D, E}. Each subset can be a Determinant or Dependent, thus 32x32 = 1024 Functional Dependencies can be proposed, like {A, B} → {C} and {D} → {B, E}.

7.2.6 Primary Key with Armstrong's rules

Use Armstrong's rules to propose a primary key for the following relation schema:
R(A, B, C, D, E, F, G, H)
with the following set of functional dependencies:
{A → BC, E → FG, AB → D, EG → H}
Please specify which of Armstrong's rules are used in each step.

Solution:

Given the relation schema:
R(A, B, C, D, E, F, G, H)
And the Set of Functional Dependencies:
{A → BC, E → FG, AB → D, EG → H}
Armstrong's Rules in action:
A → A Self-determination
A → B Decomposition of A → BC
A → C Decomposition of A → BC
A → D Augmentation of A → B to AA → AB and Transitivity with AB → D gives AA → D and AA = A.
E → E Self-determination
E → F Decomposition of E → FG
E → G Decomposition of E → FG
E → H Augmentation of E →G to EE → EG and Transitivity with EG → H gives EE → H and EE = E.
A → A, B, C, D Union
E → E, F, G, H Union
AE -> A, B, C, D, E, F, G, H Composition

As all attributes are dependent on AE, AE is a super key. As it is minimal, it is a candidate key and can therefore be chosen as the primary key.