ECE 356 - Lab 2 Part 2

Manthan Shah (20658832), Eric Song (20658437), Stephanie Meler (20662056)

Part A: 1NF

It can be observed that the relations Employee, Project, and Assigned are all in 1NF, as they do not have any value domains that can be broken down further (i.e. value domains of all attributes in each relation are atomic).

The Department relation is not in 1NF. This is because the location attribute is not atomic, as information is repeated. We can break the location attribute into a new set of attributes; streetNum, streetName, cityName, province, and postalCode. The Department schema should be updated as shown in Figure 1.

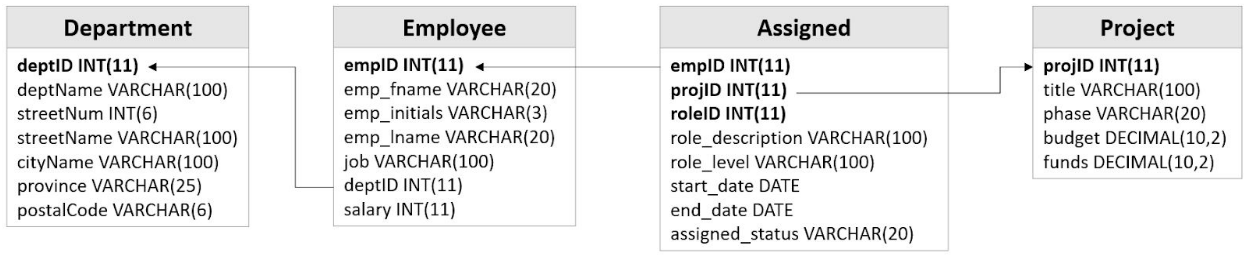

Figure 1. Relational schema diagram after 1NF verification with primary keys in bold and foreign keys depicted by arrows

To account for the 1NF decomposition of Department, we must add a new functional dependency,

$$\text{location} \rightarrow \text{streetNum, streetName, cityName, province, postalCode}$$

to the set of functional dependencies F. Using augmentation and transitivity, we can simplify the functional dependencies related to the table Department as such:

$$\text{deptID} \rightarrow \text{deptName, streetNum, streetName, cityName, province, postalCode}$$

This particular operation is permissible since the attribute ‘location’ no longer exists for the Department relation nor any other relation.

Thus, the final set of functional dependencies F for Part A is shown in Table 1.

Table 1. Set of Functional Dependencies F for Part A

Table Name	**Functional Dependencies**
Department	deptID → deptName, streetNum, streetName, cityName, province, postalCode
Project	projID → title, phase, budget, funds
Employee	empID → emp_fname, emp_initials, emp_lname, job, deptID, salary
Assigned	empID, projID, roleID → role_description, role_level, start_date, end_date, assigned_status roleID → role_description, role_level

Part B1: 3NF

To determine if each relation is in 3NF, we can apply the 3NF test on each relation over their respective functional dependencies.

B1.1: Employee relation

The set of functional dependencies F (from Table 1) covers attributes not within the Employee relation, so the simplified 3NF test cannot be used. To start the general 3NF test, the attribute closures for all attributes of Employee with respect to F must be computed.

We will define a new set of *relevant* functional dependencies as **F'**. For a functional dependency to be relevant to a particular relation, it must have at least one attribute on the left and one attribute on the right from schema of interest. Otherwise, it does not describe anything and is trivial, making it *irrelevant*. For the relation Employee, the functional dependency

empID → emp_fname, emp_initials, emp_lname, job, deptID, salary

is a relevant functional dependency since empID is on the left and emp_fname, emp_initials, etc are on the right. On the other hand, in

deptID → deptName, streetNum, streetName, cityName, province, postalCode

deptID is the only attribute in the Employee relation. Since it is missing an Employee attribute on the right, this functional dependency is not relevant with respect to Employee. For example, the above functional dependency w.r.t. Employee would be equivalent to:

deptID → {}

Similarly, every other functional dependency in F only has one attribute in Employee, or no attributes of Employee. Thus, they do not belong in F'.

Looking at the empID → … functional dependency, we can understand that empID *must* be part of any candidate key since it is never on the right side of a functional dependency in F'. Likewise, all other attributes of Employee are only found on the right side of F' so they are not necessarily part of the candidate key. First, we can compute the attribute closure of empID:

$$(\text{empID})^{+} = \text{empID, emp\_fname, emp\_initials, emp\_lname, job, deptID, salary}$$

We can see that $(\text{empID})^{+}$ is a candidate key. This is also the *only* candidate key as any closures with more attributes would not be the simplest form. Furthermore, every other single attribute closure would not be valid since empID must be in the candidate key as explained previously.

Since empID was identified as a candidate key, and the α (empID) of the only relevant functional dependency is a super key, it does not violate 3NF. All non-relevant functional dependencies are also in 3NF with respect to Employee since they are trivial. As all of F is in 3NF, we can now conclude that the Employee relation is in 3NF.

B1.2: Project relation

Similarly to before, we cannot apply the simplified 3NF test over the Project relation since F contains functional dependencies with attributes not covered by Project.

Before computing the attribute closures, we can discard any functional dependency with only one attribute of Project since they are not relevant (or those without any attributes of Project). The remaining *relevant* functional dependencies denoted by F' (see B1.1 for an explanation of F') are:

$$\text{projID} \rightarrow \text{title, phase, budget, funds}$$

In F', the attribute projID must be part of any candidate key since it does not appear on the right side. Thus, the attribute closure of projID is:

$$(\mathit{projID})^{+} = \text{projID, title, phase, budget, funds}$$

We can see that projID is a candidate key. Additionally, it must be the only candidate key since any multi-attribute closures would be more complex versions of $(\text{projID})^{+}$, and other single attribute closures would be invalid since projID needs to be part of the candidate key.

Inspecting the relevant functional dependencies in F',

$$\text{projID} \rightarrow \text{title, phase, budget, funds}$$

the ⍺ (projID) is a superkey of Project so it does not violate 3NF. All non-relevant functional dependencies (F - F') are all trivial with respect to Project, so they do not violate 3NF. As all F does not violate 3NF, we can conclude that the Project relation is in 3NF.

B1.3: Assigned relation

Likewise to the previous relations, we cannot apply the simplified 3NF test over the Assigned relation since F contains attributes not covered by Assigned.

Like before, we can discard non-relevant functional dependencies. The resulting set of relevant functional dependencies F' with respect to the schema of interest (see B1.1 for an explanation of F') are:

empID, projID, roleID → role_description, role_level, start_date, end_date, assigned_status
roleID → role_description, role_level

By inspecting F', the attributes empID, projID, and roleID never appear on the right side so they must be part of any candidate key.

$(empID,\ projID,\ roleID)^+$ = empID, projID, roleID, role_description, role_level, start_date, end_date, assigned_status

This is also the only candidate key since longer attribute closures (of 4 or more attributes) would be more complex. Also, smaller attribute closures (2 or less) cannot be candidate keys since these three attributes must be part of any candidate key.

Inspecting the first functional dependency in F',

empID, projID, roleID → role_description, role_level, start_date, end_date, assigned_status

the ⍺ (empID, projID, roleID) was identified as a candidate key and thus a superkey, so it is in 3NF.

The next functional dependency in the F' is:

roleID → role_description, role_level

This ⍺ (empID) is not a superkey (or candidate key) since roleID does not functionally determine all other attributes. Also, each attribute from $\beta - \alpha$ = {role_description, role_level} is not contained in any candidate key. Thus, this functional dependency violates the rules for 3NF, and the Assigned relation is not in 3NF.

To get the Assigned relation into 3NF, we can perform the 3NF decomposition procedure. First, we must determine the canonical cover. Observe that role_description in the first functional dependency is extraneous on the RHS:

Compute $(empID, projID, roleID)^+$ over F' (defined at beginning of B1.3)

$(empID, projID, roleID)^+ =$ role_description, role_level, start_date, end_date, assigned_status

Since role_description is in the attribute set closure, it is extraneous, and the canonical cover now becomes:

F_c = { empID, projID, roleID → role_level, start_date, end_date, assigned_status,
roleID → role_description, role_level }

Observe that role_level in the first functional dependency in F_c is extraneous on the RHS:

Compute $(empID, projID, roleID)^+$ over F'

$(empID, projID, roleID)^+ =$ role_description, role_level, start_date, end_date, assigned_status

Since role_level is in the attribute set closure, it is extraneous, and the canonical cover now becomes:

F_c = { {empID, projID, roleID → start_date, end_date, assigned_status},
{roleID → role_description, role_level} }

Since F_c cannot be reduced any further, it is the final canonical cover for F with respect to Assigned. Now, as part of the 3NF decomposition procedure, we can develop the new relations (dependency preserving relations of Assigned):

$Assigned = R_1$ (empID, projID, roleID, start_date, end_date, assigned_status)

$Role = R_2$ (roleID, role_description, role_level)

B1.4: Department relation

For the Department relation, we can apply the same methodology as with the Project and Employee relations. We are not able to use the simplified 3NF test over the Department relation as there are attributes that appear in the functional dependencies that are not contained in Department.

Before computing attribute closures, the functional dependencies with only one attribute of Department can be discarded as they are not relevant.

The remaining functional dependencies denoted by F' (see B1.1 for an explanation of F') are:

deptID → deptName, streetNum, streetName, cityName, province, postalCode

We can identify that the candidate key is deptID based on its attribute set closure over Project:

$(deptID)^+$ = deptID, deptName, streetNum, streetName, cityName, province, postalCode

No other candidate key can exist since this is the simplest attribute closure containing deptID.

Since the only functional dependency in F' has an α (deptID) which is the candidate key, it does not violate 3NF. All other dependencies in F not in F' are trivial w.r.t. Department, hence they are in 3NF. We can now conclude that the Department relation is in 3NF.

B1.5: 3NF Schema

Since the Employee, Project, and Department relations were all found to be in 3NF, their relation schemas do not need to be modified. However, the Assigned relation would need to be updated, with the full relation schema shown in Figure 2:

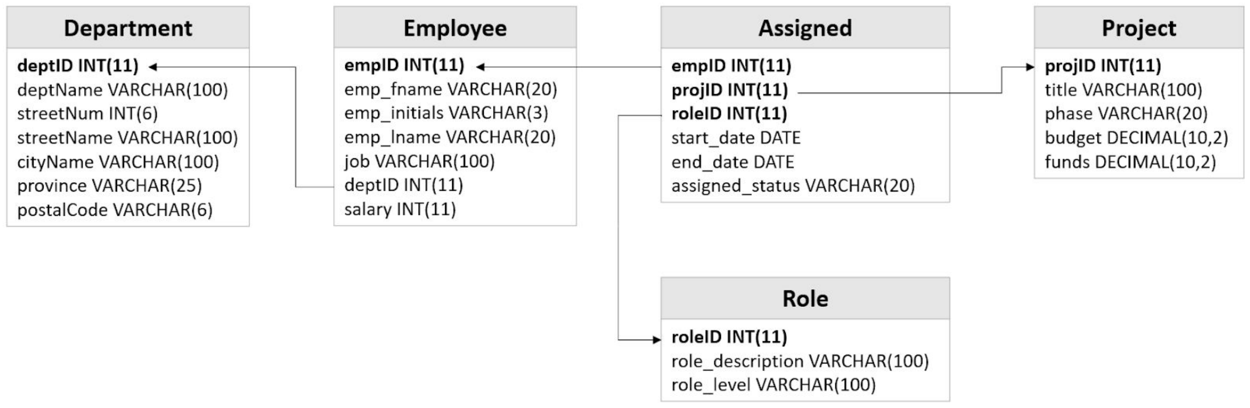

Figure 2. Relational schema diagram after 3NF decomposition with primary keys in bold and foreign keys depicted by arrows

Part B2: BCNF

To determine if each relation is in BCNF, we can apply the BCNF test on each relation over their respective functional dependencies, and decompose non-compliant relations.

B2.1: Employee relation

The set of functional dependencies F (from Table 1) covers attributes not within the Employee relation, so the simplified BCNF test cannot be used.

We can eliminate non-relevant functional dependencies with only one attribute of Employee (or none). This is because for a functional dependency to have meaning with regards to the selected schema of interest (Employee), it must have at least one attribute on both left and right sides (i.e. $A \rightarrow B$), as described in B1.1. Thus, the relevant functional dependencies denoted by the set F', are

$$\text{empID} \rightarrow \text{emp\_fname, emp\_initials, emp\_lname, job, deptID, salary}$$

Inspecting F', the only attribute not on the right side of any functional dependency is empID. Consequently, it must be part of any candidate key. We can start by computing $(\text{empID})^+$:

$$(empID)^+ = \text{empID, emp\_fname, emp\_initials, emp\_lname, job, deptID, salary}$$

This shows that empID is the only candidate key since any larger attribute closure containing empID would not be simpler than $(\text{empID})^+$, and any single attribute closure cannot be valid since empID must be in any candidate key.

Looking at the only functional dependency in F',

$$\text{empID} \rightarrow \text{emp\_fname, emp\_initials, emp\_lname, job, deptID, salary}$$

we see that α (empID) is a superkey so it does not violate BCNF. The remaining functional dependencies in F are trivial with respect to Employee, so they also do not violate BCNF. Thus, the Employee relation is in BCNF.

B2.2: Project relation

Similarly to Employee, the set of functional dependencies F (from Table 1) covers attributes not within the Project relation so the simplified BCNF test cannot be used.

We define a set F' of relevant functional dependencies w.r.t. Project (as defined in B1.1) containing the functional dependencies:

$$\text{projID} \rightarrow \text{title, phase, budget, funds}$$

Examining F', we can see that projID is the only attribute not on the right side of any functional dependency. Consequently, projID must be a part of any candidate key of Project. We can compute $(\text{projID})^+$ as:

$$(\mathit{projID})^+ = \text{projID, title, phase, budget, funds}$$

This must be the only candidate key since any larger attribute closure would not be simpler, and any other single attribute closure cannot be valid since it would not contain projID.

Looking at the only functional dependency in F'.

$$\text{projID} \rightarrow \text{title, phase, budget, funds}$$

where ɑ (projID) is a superkey, F' does not violate BCNF. All other functional dependencies in F are trivial w.r.t. Project so they also do not violate BCNF. Thus, the Employee relation is in BCNF.

B2.3: Assigned relation

Likewise to the previous relations, the set of functional dependencies F (from Table 1) covers attributes not within the Assigned relation so the simplified BCNF test cannot be used.

We start by defining the set of relevant functional dependencies F' (as defined in B1.1) w.r.t Assigned, which are:

$$\text{empID, projID, roleID} \rightarrow \text{role\_description, role\_level, start\_date, end\_date, assigned\_status}$$
$$\text{roleID} \rightarrow \text{role\_description, role\_level}$$

By inspecting F', the attributes empID, projID, and roleID never appear on the right side so they must be part of any candidate key. The attribute closure of this is:

$$(\mathit{empID},\ \mathit{projID},\ \mathit{roleID})^+ = \text{empID, projID, roleID, role\_description, role\_level, start\_date, end\_date, assigned\_status}$$

This must also be the only candidate key since any larger attribute closure would not be simpler. Also, other attribute closures with three or less attributes would not be valid as a candidate key since they cannot contain all three empID, projID, and roleID.

Looking at the two functional dependencies in F', the first,

$$\text{empID, projID, roleID} \rightarrow \text{role\_description, role\_level, start\_date, end\_date, assigned\_status}$$

is in BCNF form because ɑ (empID, projID, roleID) was identified as a candidate key and thus a superkey. However, the second functional dependency in F',

$$\text{roleID} \rightarrow \text{role\_description, role\_level}$$

violates BCNF because it is not a trivial functional dependency and α (roleID) is not a superkey for Assigned. Therefore, we must decompose Assigned using this functional dependency to put it in BCNF form.

The resulting R_1 and R_2 will be :

Role = R_1 (roleID, role_description, role_level), which is the union of α (roleID) and β (role_description, role_level).
Assigned = R_2 (empID, projID, roleID, star_date, end_date, assigned_status), which is R - β (role_description, role_level).

Looking at the Role = R_1 schema, the only functional dependency that is relevant in evaluating the schema will be,

$$roleID \rightarrow role\_description, role\_level$$

where roleID now becomes a candidate key $(roleID)^+ =$ roleID, role_description, role_level. Consequently,. we can confirm that α (roleID) is a superkey and thus R_1 is now in BCNF.

Looking at Assigned = R_2, the functional dependency

$$empID, projID, roleID \rightarrow role\_description, role\_level, start\_date, end\_date, assigned\_status$$

holds on R_2 but α (empID, projID, roleID) is a superkey of R_2. The other functional dependency

$$roleID \rightarrow role\_description, role\_level$$

is no longer relevant to R_2. Thus, R_2 is in BCNF.

For the remaining functional dependencies in F (i.e. F - F'), they are trivial to both new relations R_1 and R_2. Thus, both R_1 and R_2 are fully BCNF compliant.

B2.4: Department relation

Again, the simplified BCNF test cannot be used since the set of functional dependencies F (from Table 1) covers attributes not within the Department relation. We start by defining the relevant set of functional dependencies w.r.t to Department, F' (as defined in B1.1):

$$deptID \rightarrow deptName, streetNum, streetName, cityName, province, postalCode$$

By inspecting F', deptID never appears on the right side so it must be part of any candidate key. The attribute closure of this is:

$$(deptID)^{+} = \text{deptID, deptName, streetNum, streetName, cityName, province, postalCode}$$

which determines deptID as a candidate key. Also, this must be the only candidate key since any larger attribute closure would not be simpler, and any other single attribute closure cannot be valid since it would not contain deptID.

Looking at the only functional dependency in F'.

$$\text{deptID} \rightarrow \text{deptName, streetNum, streetName, cityName, province, postalCode}$$

α (deptID) is a superkey so F' does not violate BCNF. All other functional dependencies in F are trivial w.r.t. Department so they also do not violate BCNF. Thus, the Department relation is in BCNF.

B2.5: BCNF Schema

Since the Employee, Project, and Department relations were all found to be in BCNF, their relation schemas do not need to be modified. However, the Assigned relation would need to be updated, with the full relation schema shown in Figure 3:

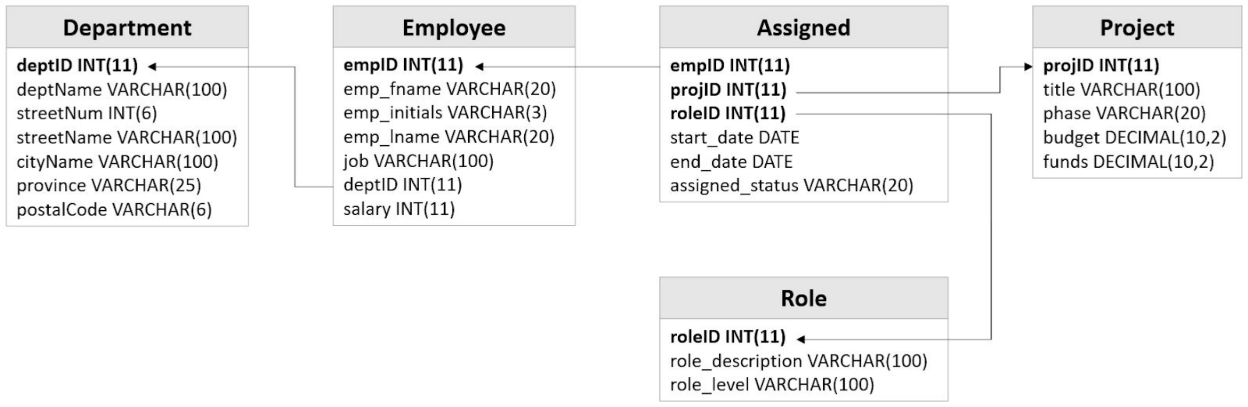

Figure 3. Relational schema diagram after BCNF decomposition with primary keys in bold and foreign keys depicted by arrows