ECE356 Lab2 Part 1

Step 1: Decompose it, pre BCNF

1) Check Employee table schema:
    a. The empName is the full name of each employee, and consist of a First and Last Name, optionally a Middle Name. Since attribute empName is not atomic, the current Employee table schema violates 1NF. Therefore, we need to decompose empName into three attributes which are firstName, lastName and middleName. Since each empName has one middle name and it is optional, middleName needs a separate relation. Therefore, table Employee should be split into two tables to satisfy BCNF, as shown below.

       Employee: (empID, firstName, lastName, job, deptID, salary)
       MiddleName: (empID, middleName)

    b. Since an employee can be in more than one department, empID is no longer sufficient as a candidate key. Likewise, given that depID can be shared, depID cannot be a candidate key. However, firstName, lastName, job, salary is not shared. Therefore, we still have the functional dependency: empID → firstName, lastName, job, salary. Since this is non-trivial and empID is not a superkey, this violates BCNF. Therefore, the Employee table must be broken down into Employee: (empID, firstName, lastName, job, salary) and EmployeeDepartment: (empID, depID)

2) Check Project table schema:
    We have functional dependency projID → title, budget, fund. projID. Since projID is unique per project, projID is a candidate key. Therefore, Project table does not violate BCNF.

3) Check Assigned table schema:
    a. An employee may be assigned to more than one project, therefore, the combination (empID, projID) can act as a candidate key, since it will be unique and will uniquely determine the relation.

    b. An employee may have more than one role on a project. the combination (empID, projID, role) can act as a candidate key, since it will be unique and will uniquely determine the relation.

    Assigned table does not violate BCNF.

4) Check Department table schema:
    a. A department location is a full address, comprising the street number, street name, city name, province and postal code. Since attribute location is not atomic, the current Department table schema violate 1NF. Therefore, we need to decompose attribute location into five attributes which are street number, street name, city name, province and postal code.

Department: (depID, deptName, streetNumber, streetName, city, province, postalCode)

b. Since a department may have multiple location, deptID is not sufficient as a candidate key. However, deptName is not shared. Therefore, we still have the functional dependency: deptID → deptName. Since this is non-trivial and empID is not a superkey, this violates BCNF. Therefore, the Department table must be broken down into Department: (deptID, deptName) and DepartmentLocation: (deptID, streetNumber, streetName, city, province, postalCode)

c. Since postal code functional determines city name and province, but postal code is not a superkey in the DepartmentLocation relationship. Therefore, PostalCode needs a separate relation. Therefore, Department table must be broken down into DepartmentLocation: (depID, streetNumber, streetName, postalCode) and PostalCode: (postalCode, city, province).

After BCNF decomposition, we have relations:

1) Employee: (empID, firstName, lastName, job, salary)
2) MiddleName: (empID, middleName)
3) EmployeeDepartment: (empID, depID)
4) Project: (projID, title, budget, funds)
5) Assigned: (empID, projID, role)
6) Department: (deptID, deptName)
7) DepartmentLocation: (deptID, streetNumber, streetName, postalCode)
8) PostalCode: (postalCode, city, province)

Step 2: Determine primary keys:

1) Employee: empID
2) MiddleName: empID
3) EmployeeDepartment: empID, depID
4) Project: projID
5) Assigned: empID, projID, role
6) Department: deptID
7) DepartmentLocation: deptID, streetNumber, streetName, postalCode
8) PostalCode: postalCode

Step 3: Determine foreign keys:

1) Employee: No foreign keys
2) MiddleName: No foreign keys
3) EmployeeDepartment:
   a. empID is a foreign key, referencing the empID of Employee relation
   b. depID is a foreign key, referencing the depID of Department relation
4) Project: No foreign keys

5) Assigned:
    a. empID is a foreign key, referencing the empID of Employee relation
    b. projID is a foreign key, referencing the projID of Project relation
6) Department: No foreign keys
7) DepartmentLocation:
    a. depID is a foreign key, referencing the depID of Department relation
    b. postalCode is a foreign key, referencing the postalCode of PostalCode relation
8) PostalCode: No foreign keys