

**Kazakh-British Technical University**

Faculty of Information Technology

Laboratory Work №5

Prepared by: Maratuly T.

.

**Almaty, 2021**

**Exercise 1.**

Will the conversion to BCNF be dependency preserving in any case? Proof your statement and give reasoning for choosing BCNF design.

**Answers:**

Decomposition into BCNF can prevent efficient testing of certain functional dependencies. Consider the example where student may have only one advisor (many – to – one), instructor can be associated with only a single department, and a student may have more than one advisor, but no more than one from a given department.:

Diagram

Description automatically generated

🡪 dept\_advisor (s\_ID, i\_ID, dept\_name)

Adding extra constraint that “an instructor can act as advisor for only a single department.

i\_ID 🡪 dept\_name

s\_ID, dept\_name 🡪 i\_ID

Ater converting to BCNF we obtain: (s\_ID, i\_ID), (i\_ID, dept\_name). Because the design doesn’t permit the enforcement of this functional dependency without a join, we say that our design is not dependency preserving. It is better to use BCNF in order to distribute data, it means that all redundancy based on functional dependency has been removed, but some other types of redundancy may exist.It is a more restricted form of normalization so that the database does not end in anomalies.

**Exercise 2**

Given table in 1NF, convert to 3NF if PK is UnitID.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| UnitID | StudentID | Date | Tutor ID | Topic | Room | Grade | Book | TutEmail |
| U1 | St1 | 23.02.03 | Tut1 | GMT | 629 | 4.7 | Deumlich | tut1@fhbb.ch |
| U2 | St1 | 18.11.02 | Tut3 | Gln | 631 | 5.1 | Zehnder | [tut3@fhbb.ch](mailto:tut3@fhbb.ch) |
| U1 | St4 | 23.02.03 | Tut1 | GMT | 629 | 4.3 | Deumlich | tut1@fhbb.ch |
| U5 | St2 | 05.05.03 | Tut3 | PhF | 632 | 4.9 | Dummlers | tut3@fhbb.ch |
| U4 | St2 | 04.07.03 | Tut5 | AVQ | 621 | 5.0 | SwissTopo | tut5@fhbb.ch |

**Answer:**

**Second Normal Form:**

Determine the dependencies:

StudentID , UnitID 🡪 Grade

UnitID 🡪 TutorID

UnitID 🡪 Date

UnitID 🡪 Topic

UnitID 🡪 Room

UnitID 🡪 Book

UnitID 🡪 TutEmail

TutorID 🡪 TutEmail

From the above dependencies we see that the candidate key for the whole table is (StudentId,UnitID) that becomes a primary key since it is the only one candidate key.

The table in 1 NF is divided into 2 tables:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UnitID** | **StudentID** | Date | Tutor ID | Topic | Room | Grade | Book | TutEmail |

2 expressed tables in **Second Normal Form**:

|  |  |  |
| --- | --- | --- |
| **StudentID** | **UnitID** | Grade |
| St1 | U1 | 4.7 |
| St1 | U2 | 5.1 |
| St4 | U1 | 4.3 |
| St2 | U5 | 4.9 |
| St2 | U4 | 5.0 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **UnitID** | Date | TutorID | Topic | Room | Book | TutEmail |
| U1 | 23.02.03 | Tut1 | GMT | 629 | Deumlich | [tut1@fhbb.ch](mailto:tut1@fhbb.ch) |
| U2 | 18.11.02 | Tut3 | Gln | 631 | Zehnder | [tut3@fhbb.ch](mailto:tut3@fhbb.ch) |
| U5 | 05.05.03 | Tut3 | PhF | 632 | Dummlers | tut3@fhbb.ch |
| U4 | 04.07.03 | Tut5 | AVQ | 621 | SwissTopo | tut5@fhbb.ch |

**Third Normal Form:**

Eliminate Transitive dependencies from the table above. We have UnitID 🡪 TutorID. Furthermore, TutorID 🡪 TutorEmail. 🡨 is the transitive dependency.

Tables in **Third Normal Form:**

|  |  |  |
| --- | --- | --- |
| **StudentID** | **UnitID** | Grade |
| St1 | U1 | 4.7 |
| St1 | U2 | 5.1 |
| St4 | U1 | 4.3 |
| St2 | U5 | 4.9 |
| St2 | U4 | 5.0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UnitID** | Date | TutorID | Topic | Room | Book |
| U1 | 23.02.03 | Tut1 | GMT | 629 | Deumlich |
| U2 | 18.11.02 | Tut3 | Gln | 631 | Zehnder |
| U5 | 05.05.03 | Tut3 | PhF | 632 | Dummlers |
| U4 | 04.07.03 | Tut5 | AVQ | 621 | SwissTopo |

|  |  |
| --- | --- |
| **TutorID** | TutEmail |
| Tut1 | [tut1@fhbb.ch](mailto:tut1@fhbb.ch) |
| Tut3 | [tut3@fhbb.ch](mailto:tut3@fhbb.ch) |
| Tut5 | tut5@fhbb.ch |

The tables above in **3NF.**

**Exercise 3**

Given table in 1NF, convert to 2NF if PK is {ProjectName, ProjectManager}, use decomposition:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ProjectName** | **ProjectManager** | Position | Budjet | TeamSize |
| Project1 | Manager1 | CTO | 1 kk $ | 15 |
| Project2 | Manager2 | CTO2 | 1.5 kk $ | 12 |

**Answer:**

The table is already in 1NF. Lets consider for 2NF. Define all the dependencies for 2NF (Partial):

**#1 Option of deciding Dependencies:**

ProjectName 🡪 TeamSize

ProjectManager 🡪 Position

ProjectName 🡪 Budget

Here the position is assumed as Manager career position.

|  |  |
| --- | --- |
| **ProjectManager** | Position |
| Manager1 | CTO |
| Manager2 | CTO2 |

|  |  |
| --- | --- |
| **ProjectName** | **ProjectManager** |
| Project1 | Manager1 |
| Project2 | Manager2 |

|  |  |  |
| --- | --- | --- |
| **ProjectName** | Budget | TeamSize |
| Project1 | 1 kk $ | 15 |
| Project2 | 1.5 kk $ | 12 |

**#2 Option of deciding dependencies:**

ProjectName, ProjectMan 🡪 TeamSize

ProjectName, ProjectMan 🡪 Budget

ProjectManager 🡪 Position

|  |  |
| --- | --- |
| **ProjectManager** | Position |
| Manager1 | CTO |
| Manager2 | CTO2 |

|  |  |  |  |
| --- | --- | --- | --- |
| **ProjectName** | **ProjectManager** | Budjet | TeamSize |
| Project1 | Manager1 | 1 kk $ | 15 |
| Project2 | Manager2 | 1.5 kk $ | 12 |

**Exercise 4**

Given table, convert to 3NF if PK is Group, use decomposition:

(Faculties have a number of specialties, each specialty consists of set of particular groups).

|  |  |  |
| --- | --- | --- |
| **Group** | Faculty | Speciality |
| g1 | f1 | s1 |
| g2 | f2 | s2 |

**Answer:**

**2 option of solutions:**

**OPTION #1 (Normalization):**

The provided table is already in 1NF 🡪 lets consider it for 2NF.

All the dependencies: group 🡪 specialty

group 🡪 faculty

specialty 🡪 faculty

Considering those dependencies above we can guess that the table is already in 2NF.

Lets consider it for 3NF.

For 3NF we need to consider transitive dependencies. As we see from our list of dependencies the faculty actually depends on specialty directly. The tables in 3NF:

|  |  |
| --- | --- |
| **Group** | Speciality |
| g1 | s1 |
| g2 | s2 |

|  |  |
| --- | --- |
| **Speciality** | faculty |
| s1 | f1 |
| s2 | f2 |

**OPTION #2 (ERD):**

Considering the provided expression: (Faculties have a number of specialties, each specialty consists of set of particular groups). Use it to construct ER diagram.

Diagram

Description automatically generated

**Exercise 5**

Given table, convert to BCNF if PK is {ProjectID, Department}, use decomposition:

(Curator depends on projectID and related departments, teamSize directly relates to project and related departments, ProjectGroupsNumber depends on TeamSize.)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ProjectID** | **Department** | Curator | TeamSize | ProjectGroupsNumber |
| p1 | d1 | e1 | 100 | 5 |
| p2 | d2 | e2 | 120 | 6 |

**Answer:**

Define all the dependencies using the provided information:

(ProjectID,Department) 🡪 Curator

(ProjectID, Department) 🡪 TeamSize

(ProjectId, Department) 🡪 ProjectGroupNumber

TeamSize 🡪 ProjectGroupNumber

The table is already in 1NF. Let’s consider for 2NF. For 2NF everything is already corresponded. Therefore, the table is already in 2NF. For 3NF we need to consider the transitive dependency. That’s we need to divide our table into 2 sub tables:

**3NF:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ProjectID** | **Department** | Curator | TeamSize |
| p1 | d1 | e1 | 100 |
| p2 | d2 | e2 | 120 |

|  |  |
| --- | --- |
| **TeamSize** | ProjectGroupNumber |
| 100 | 5 |
| 120 | 6 |

**BCNF:**

For the BCNF we can guess that one curator works only in the only one department. That means we can define the name of the department just by the name of the curator.

|  |  |
| --- | --- |
| **TeamSize** | ProjectGroupNumber |
| 100 | 5 |
| 120 | 6 |

|  |  |
| --- | --- |
| **Curator** | Department |
| e1 | d1 |
| e2 | d2 |

|  |  |
| --- | --- |
| **ProjectID** | **Curator** |
| p1 | e1 |
| p2 | e2 |

|  |  |  |
| --- | --- | --- |
| **ProjectID** | **Department** | TeamSize |
| p1 | d1 | 100 |
| p2 | d2 | 120 |

**Exercise 6**

List the three design goals for relational databases, and explain why each is desirable. Give an example of both desirable and undesirable types of decompositions.

**Answer:**

Three design goals are lossless-join decompositions, dependency preserving decompositions, and minimization of repetition of information. They are desirable so we can maintain an accurate database, check correctness of up-date quickly, and use the smallest amount of space possible.

There are two types of decomposition: Lossy Decomposition, Lossless Join Decomposition.

Example of **Lossy Decomposition**:

Diagram

Description automatically generated

As we can see the lossy decomposition can lead to unclear data table with as the join of tables brings the table with redundant information and we can’t find person who we are looking for.

Example of **Lossless Decomposition**:

|  |  |  |
| --- | --- | --- |
| PersonId | PersonName | CityID |
| 101 | Temirbolat | Krg09 |
| 102 | Tamerlan | Alm02 |

|  |  |  |
| --- | --- | --- |
| CityID | CityName | Country |
| Krg09 | Karaganda | Kazakhstan |
| Alm02 | Almaty | Kazakhstan |

After the inner join or natural join we do not loss the data and we get:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PersonId | PersonName | CityID | CityName | Country |
| 101 | Temirbolat | Krg09 | Karaganda | Kazakhstan |
| 102 | Tamerlan | Alm02 | Almaty | Kazakhstan |