Chapter 8

Exercise 8.1

Consider the relation in Figure 8.19 and identify the functional dependencies of the corresponding application. Identify possible redundancies and anomalies in the relation.

Tutor	Department	Faculty	HeadOfDept	Course
Thomson	Math	Engineering	Jackson	Statistic
Thomson	Math	Engineering	Jackson	Number Theory
Robinson	Physics	Engineering	Jackson	Statistic
Robinson	Physics	Science	Johnson	Statistic
MacKay	Physics	Science	Johnson	Relativity

Figure 8.19

Solution:

A key for this relation is **Department, Faculty, Course**; a functional dependency, which does not involves the entire key, is **Department, Faculty** $\rightarrow$**HeadOfDept**; this functional dependence introduces a redundancy in the relation, because for each course in the same department and faculty, the head of department must be repeated.
The relation has an update anomaly, because if we want to change the head of a department, we have to update all the rows in which this information is present, and not only one row.
The relation contains also a deletion anomaly, because if we want to delete an head of department, we lose all information about tutors in that department.

Exercise 8.2

Identify the key(s) and functional dependencies of the relation shown in Exercise 8.1 and then identify a decomposition into Boyce-Codd normal form.

Solution:

A key for this relation is **Department, Faculty, Course**.
Also the attributes **Tutor, Faculty, Course** seem to form a key in this instance of the relation, but generally speaking this is not correct because the same tutor can teach the same course in different departments of a Faculty.

Decomposition:

Tutor	Department	Faculty	Course
Thomson	Math	Engineering	Statistic
Thomson	Math	Engineering	Number Theory
Robinson	Physics	Engineering	Statistic
Robinson	Physics	Science	Statistic
MacKay	Physics	Science	Relativity

Department	**Faculty**	**HeadofDept**
Math	Engineering	Jackson
Physics	Engineering	Jackson
Physics	Science	Johnson

This decomposition is correct, because with a join between the two relation, we obtain all and only the rows of the original relation.
Moreover, the decomposition resolves all problem about anomalies, because of the Boyce-Codd normal form.

Exercise 8.3

Consider the relation shown in Figure 8.20, which represents information on the products of a carpentry firm and their components. The following are given: the type of component of a product (attribute **Type**), the quantity of the component necessary for a certain product (attribute **Quantity),** the unit price of the component of a certain product (attribute **PriceOfC**), the supplier of the component (attribute **Supplier**) and the total price of the single product (attribute **PriceOfP**). Identify the functional dependencies and the key(s) of the relation.

Product	**Component**	**Type**	**Quantity**	**PriceOfC**	**Supplier**	**PriceOfP**
Bookcase	Wood	Walnut	5	10.00	Smith	400
Bookcase	Screw	B212	200	0.10	Brown	400
Bookcase	Glass	Crystal	3	5.00	Jones	400
Seat	Wood	Oak	5	15.00	Smith	300
Seat	Screw	B212	250	0.10	Brown	300
Seat	Screw	B414	150	0.30	Brown	300
Desk	Wood	Walnut	10	8.00	Quasimodo	250
Desk	Handle	H621	10	20.00	Brown	250
Table	Wood	Walnut	4	10.00	Smith	200

Figure 8.20

Solution:

Supposing that a Type refers only to one component, a key for the relation is **Product, Type**; so all sets of attributes which contain **Product, Type**, are superkeys for the relation.
The attributes **Quantity** and **PriceOfC** seem to be another key, but this could be not true in all instance of this database.
Another apparent key is **Type, PriceOfP**.

The functional dependencies are:

- **Product→PriceOfP**
- **Type, Supplier→PriceOfC**
- **Type→Component**

Exercise 8.4

With reference to the relation in Figure 8.20 consider the following update operations:

- Insertion of a new product
- Deletion of a product
- Addition of a component in a product
- Modification of the price of a product.

Discuss the types of anomaly that can be caused by these operations.

Solution:

1) The insertion of a new product requires the addition of a row for each component's type. The total price, which is function of the product, must be repeated in each row. Also the price of component may be a redundancy, because if the same type of component, with the same supplier is used for other products, the price of component is already present in the relation. This is an insertion anomaly.
2) The deletion of a product implicates that all rows which refers to the product must be deleted; so if a product has more than one component, the deletion of 1 product implicates the deletion of many rows; moreover this operation deletes information about the supplier of the components: if there aren't any other rows which refer to these supplier, the information about them will be lost. This is a deletion anomaly.
3) The addition of a new component implicates to add a new row to the relation. This is another insertion anomaly because, as point 1, the total price and (eventually) the price of component must be repeated.
4) The modification of the price of a product is an update anomaly, because the updating of 1 attribute implicates to update many rows in the relation (a row for each type of component of the same product).

Exercise 8.5

Consider again the relation in Figure 8.20. Describe the redundancies present and identify a decomposition of the relation that removes these redundancies. Show the schema obtained. Then verify that is possible to reconstruct the original table for this schema.

Solution:

The redundancies present in the relation are related to the functional dependencies. The redundant attributes are:

- **PriceOfP**, which is repeated in each row which refers to the same product;
- **PriceOfC,** which is repeated in each row which has the same values on **Type** and **Supplier**
- **Component**, which is repeated in each row which has the same **Type**.

A possible decomposition is:

R1

Product	Type	Quantity	Supplier
Bookcase	Walnut	5	Smith
Bookcase	B212	200	Brown
Bookcase	Crystal	3	Jones
Seat	Oak	5	Smith
Seat	B212	250	Brown
Seat	B414	150	Brown
Desk	Walnut	10	Quasimodo
Desk	H621	10	Brown
Table	Walnut	4	Smith

R2

Product	PriceOfP
Bookcase	400
Seat	300
Desk	250
Table	200

R3

Type	Component
Walnut	Wood
B212	Screw
B414	Screw
Oak	Wood
Crystal	Glass
H621	Handle

R4

Supplier	Type	PriceOfC
Smith	Walnut	10.00
Quasimodo	Walnut	8.00
Brown	B212	0.10
Brown	B414	0.30
Jones	Crystal	5.00
Smith	Oak	15.00
Brown	H261	20.00

Relation R1 has the key of the original relation, but does not contains any redundancies. Relations R2, R3, and R4 have as keys the left hand sides of the functional dependencies (see Exercise 8.3). Making joins on these keys it is possible to reconstruct exactly the information of the original schema.
All the dependencies are preserved in the decomposition, because each of them is represented with a different relation.

Exercise 8.6

Consider the schema of the relation in Figure 8.21. Its key is made up of the attributes **Title** and **CopyNo,** and on this relation we have the dependency **Title→ Author, Genre**. Verify whether the schema is in third normal form, and if not, decompose it appropriately. Verify whether the decomposition also satisfies the Boyce-Codd normal form.

Title	Author	Genre	CopyNo	Shelf
Decameron	Boccaccio	Stories	1	A75
Rubaiyat	Omar Khayyam	Poem	1	A90
Rubaiyat	Omar Khayyam	Poem	2	A90
Le Bourgeois Gentilhomme	Moliere	Play	1	A90
Le Bourgeois Gentilhomme	Moliere	Play	2	A22
Washington Square	James	Novel	1	B20
Richard III	Shakespeare	Play	1	B10

Figure 8.21

Solution:

The relation is not in Third Norma Form, because in the functional dependency
Title→Author, Genre neither of **Author** and **Genre** is part of the key.
A possible decomposition is:

R1

Title	CopyNo	Shelf
Decameron	1	A75
Rubaiyat	1	A90
Rubaiyat	2	A90
Le Bourgeois Gentilhomme	1	A90
Le Bourgeois Gentilhomme	2	A22
Washington Square	1	B20
Richard III	1	B10

R2

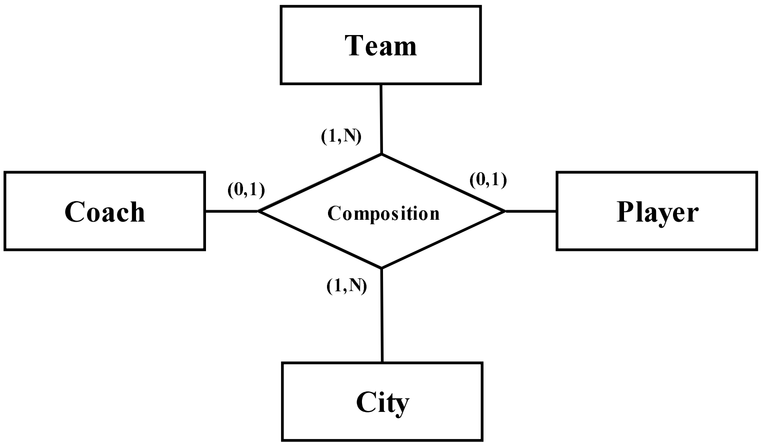
Title	Author	Genre
Decameron	Boccaccio	Stories
Rubaiyat	Omar Khayyam	Poem
Le Bourgeois Gentilhomme	Moliere	Play
Washington Square	James	Novel
Richard III	Shakespeare	Play

The relation is in Boyce-Codd normal form, because the key for R2 is **Title**, which is also the left hand side in the functional dependency.

Exercise 8.7

Consider the Entity.Relationship schema in Figure 8.22. The following properties are valid:

- A player can play only for one team (or none)
- A trainer can train only ine team (or none)
- A team belongs to one and only one city



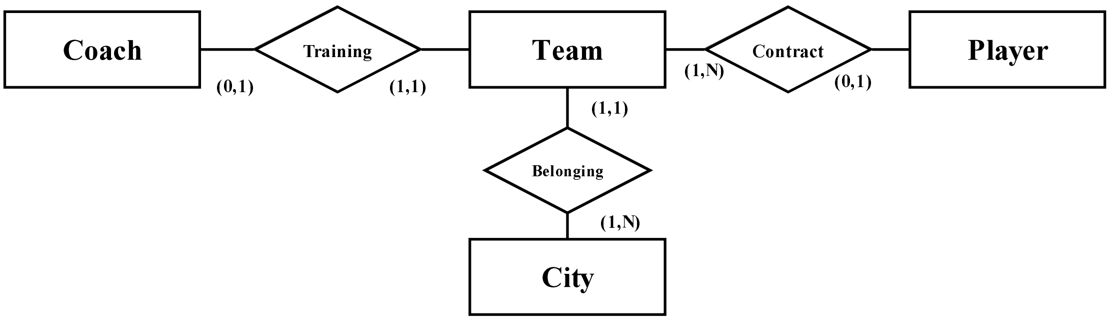
Solution:

The functional dependencies present in this schema are:

- **Player→Team**
- **Coach→Team**
- **Team→City**

The key for relationship **Composition** is **Player**, and so the schema is not in Boyce-Codd normal form.

A possible restructuring is:



In this new schema there are only binary relationship, and so now it respect the Boyce-Codd normal form.

Exercise 8.8

Consider the relation in Figure 8.23 and the following possible decomposition:

- **Department, Surname** in one relation and **Surname, FirstName, Address** in the other;
- **Department, Surname, FirstName** in one relation and **FirstName, Address** in the other;
- **Department, Surname,FirstName** in one relation and **Surname, FirstName, Address** in the other;

Department	Surname	FirstName	Address
Sales	Eastland	Fred	6 High Street
Purchasing	Eastland	Fred	6 High Street
Accounts	Watson	Ethel	27 Acacia Avenue
Personnel	Eastland	Sydney	27 Acacia Avenue

Figure 8.23

With reference both to the specific instance and to the possible instances on the same schema, identify which of these decomposition are lossless.

Solution:

The key for this relation is **Department.** We assume that people are identified by Surname and FirstName.
The relation has a functional dependency: **Surname, FirstName→ Address**.

1) this decomposition is not lossless in general; the join between the two relations produces spurious information. In fact the attribute **Surname** does not identify a person, and so the join will associate with a department all persons with the same surname. In this instance we will obtain:

Department	Surname
Sales	Eastland
Purchasing	Eastland
Accounts	Watson
Personnel	Eastland

Surname	FirstName	Address
Eastland	Fred	6 High Street
Watson	Ethel	27 Acacia Avenue
Eastland	Sydney	27 Acacia Avenue

Department	Surname	FirstName	Address
Sales	Eastland	Fred	6 High Street
Sales	Eastland	Sydney	27 Acacia Avenue
Purchasing	Eastland	Fred	6 High Street
Purchasing	Eastland	Sydney	27 Acacia Avenue
Accounts	Watson	Ethel	27 Acacia Avenue
Personnel	Eastland	Fred	6 High Street
Personnel	Eastland	Sydney	27 Acacia Avenue

2) This decomposition is lossless in this particular instance of the database, because there aren’t two persons with the same first name, and so the join between the two relations gives again the original relation, but generally speaking **FirstName** does not identify a person and so the join could give a relation with spurious information;

3) This decomposition is always lossless, because both the attributes **Surname** and **FirstName** are present in the relations, and the second relation has **Surname, FirstName** as key.
This decomposition gives a database in Boyce-Codd normal form.

Department	**Surname**	**FirstName**
Sales	Eastland	Fred
Purchasing	Eastland	Fred
Accounts	Watson	Ethel
Personnel	Eastland	Sydney

Surname	**FirstName**	**Address**
Eastland	Fred	6 High Street
Watson	Ethel	27 Acacia Avenue
Eastland	Sydney	27 Acacia Avenue

Department	**Surname**	**FirstName**	**Address**
Sales	Eastland	Fred	6 High Street
Purchasing	Eastland	Fred	6 High Street
Accounts	Watson	Ethel	27 Acacia Avenue
Personnel	Eastland	Sydney	27 Acacia Avenue

Exercise 8.9

Reconsider the relation in Figure 8.23. Verify whether the following decomposition preserve the dependencies:

- A relation on **Department, Surname** and **FirstName** and the other on **Surname** and **Address;**
- A relation on **Department, Surname** and **FirstName** and the other on **Surname, FirstName** and **Address**;
- A relation on **Department,** and **Address** and the other on **Department, Surname** and **Firstname**.

Solution:

1) This decomposition does not preserve the dependency **Surname, FirstName→Addreess**, because the attributes involved are divided into the two relations. So, if we need, for example, to change the address which refers to Sales department, we can only do this operation changing all rows in second relation which have “Eastland” as surname; but this is not correct because there are two persons with this surname and only one refers to department “Sales”. This decomposition is also not lossless.

2) This decomposition is correct, because the second relation contains all the atributes of the functional dependency.

3) This decomposition is not correct, because, as in the case of point 1, the attributes of the functional dependency are divided into the two relations; in this case it’s possible to add rows to the first relations, associating to the departments an address which does not refers to the correct person(while this is impossible in the original relation).