COMPSCI 2DB3 Assignment 6

Elite Lu lue13

# Question 1

The minimal cover is shown below with the explanations:

|  |  |
| --- | --- |
| Functional Dependency | Reasoning |
| pet\_id → name | Each pet has a unique name associated with them. |
| pet\_id → pet\_age | Each pet only has exactly one age. Having multiple would not be plausible. |
| pet\_id, owner\_id → since | For each owner\_id and pet\_id, there is one start date for when the person became an owner to the pet. |
| owner\_id → owner\_age | Each owner has exactly one age. |

# Question 2

The pets are independent of the owners that are present in the table. This independence is expressed via either the multivalued dependency “pet\_id, name, pet\_age ↠ owner\_id, owner\_age, since”, and the multivalued dependency “pet\_id, name, pet\_age ↠ pet\_friend\_id” or the join dependency “⋈{X, Y}” where X = {pet\_id, name, pet\_age, owner\_id, owner\_age } and Y = {pet\_id, name, pet\_age, pet\_friend\_id}

# Question 3

This table is not in 3NF. An example of a dependency that violates this property is “user\_id → user\_location”. This dependency is not trivial because “user\_id” is not a key nor a superkey of the relational schema and along with this, “user\_location” is not part of any other key.

In order to decompose this schema, I will use the algorithm DECOMPOSE-3NF to put the relational schema into 3NF. This can be achieved by determining a minimal cover for all the provided functional dependencies.

Note that there is one problem where in order to achieve the minimal cover, I must remove all the redundant schemas such that they will not impact the keys and dependencies. This is present with id → user\_location, id → user\_id, and user\_id → user\_location. This is a scenario where A → B, A → C, and B → C. in order to preserve the dependencies and make this the minimal cover, I will remove A → C since removing this will not impact the keys and superkeys along with the dependencies. The same can be said for book\_id → age\_rating, book\_id → child\_friendly, and age\_rating → child\_friendly. The minimal cover is shown as such:

𝔖’ = {id → user\_id, id → date,

user\_id → user\_location,

book\_id → title, book\_id → publisher, book\_id → age\_rating,

id, book\_id, format → amount,

book\_id, format → isbn,

isbn → book\_id, isbn → format,

age\_rating → child\_friendly}

With this, I will then use the for-loop of the decomposition algorithm and construct a relational schema A → X in the minimal cover with attributes 𝐴 ∪ 𝐵, where 𝐵 = {𝑌 | 𝐴 −→ 𝑌 ∈ 𝔖′}. From this, I will get the following:

|  |  |
| --- | --- |
| Schemas | Dependencies |
| r0 | r0(id, user\_id, date) from id → user\_id, id → date |
| r1 | r1(user\_id, user\_location) from user\_id → user\_location |
| r2 | r2(book\_id, title, publisher, age\_rating, child\_friendly) from book\_id → title, book\_id → publisher, book\_id → age\_rating |
| r3 | r3(id, book\_id, format, amount) from id, book\_id, format → amount |
| r4 | r4(book\_id, format, isbn) from book\_id, format → isbn |
| r5 | r5(book\_id, format, isbn) from isbn → book\_id, isbn → format |
| r6 | r6(age\_rating, child\_friendly) from age\_rating → child\_friendly |

With this, we must consider the keys that have not been used and any redundant relational schemas. The keys that have not been used include category and author. Because of this. I will create another relational schema, shown below:

|  |  |
| --- | --- |
| r7 | r7(id, book\_id, format, author, category) |

Next, when I will remove the redundant schemas. This would include r4. This is because r4 is the same as r5. Because of this, the following functional schemas hold each relational schema of the resulting decomposition:

|  |  |
| --- | --- |
| Relational Scheme | Functional Dependencies |
| r0(id, user\_id, date) | “id → user\_id, date”. |
| r1(user\_id, user\_location) | “user\_id → user\_location” |
| r2(book\_id, title, publisher, age\_rating) | “book\_id → title, publisher, age\_rating”. |
| r3(id, book\_id, format, amount) | “id, book\_id, format → amount”. |
| r5(book\_id, format, isbn) | “isbn → book\_id, format”.  “isbn → book\_id, isbn → format”. |
| r6(age\_rating, child\_friendly) | “age\_rating → child\_friendly”. |
| r7(id, book\_id, format, author, category) |  |

Not that the decomposition is lossless-join and dependency-preserving, as we have used the DECOMPOSE-3NF algorithm, which guarantees lossless-join and dependency-preservation.

Finally, the decomposition of the dataset is shown below. Note that the naming convention for the columns is with respect to the original document:

## r0

|  |  |  |
| --- | --- | --- |
| Id | Ui | D |
| 1 | 1 | 24 Dec. |
| 2 | 2 | 25 Dec. |
| 3 | 1 | 26 Dec. |

## r1

|  |  |
| --- | --- |
| Ui | Ul |
| 1 | Hamilton |
| 2 | Toronto |

## r2

|  |  |  |  |
| --- | --- | --- | --- |
| B | T | P | Ar |
| 2 | Book! | ThePrinter | 18+ |
| 5 | Comic! | TheCopier | 5+ |

## r3

|  |  |  |  |
| --- | --- | --- | --- |
| Id | B | F | Am |
| 1 | 2 | paperback | 3 |
| 1 | 2 | hardcover | 6 |
| 2 | 2 | e-book | 1 |
| 3 | 5 | paperback | 1 |

## r5

|  |  |  |
| --- | --- | --- |
| B | F | Is |
| 2 | paperback | 1234 |
| 2 | hardcover | 1237 |
| 2 | e-book | 1241 |
| 5 | paperback | 1298 |

## r6

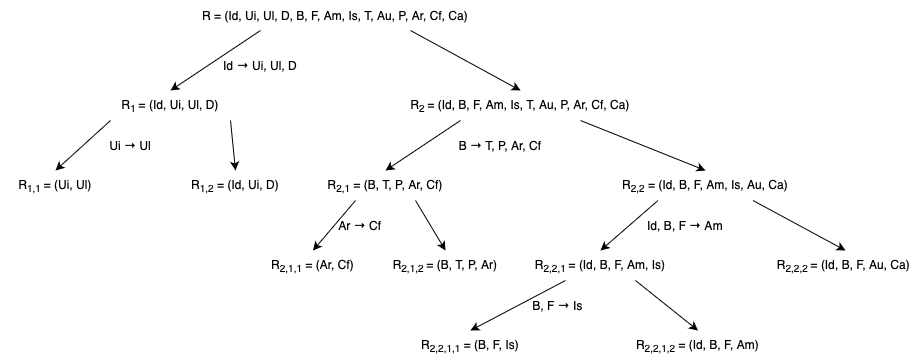
|  |  |
| --- | --- |
| Ar | Cf |
| 18+ | no |
| 5+ | yes |

## r7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | B | F | Au | Ca |
| 1 | 2 | paperback | Alicia | CS |
| 1 | 2 | paperback | Alicia | Theory |
| 1 | 2 | paperback | Dafni | CS |
| 1 | 2 | paperback | Dafni | Theory |
| 1 | 2 | hardcover | Dafni | CS |
| 1 | 2 | hardcover | Dafni | Theory |
| 1 | 2 | hardcover | Dafni | CS |
| 1 | 2 | hardcover | Dafni | Theory |
| 2 | 2 | e-book | Alicia | CS |
| 2 | 2 | e-book | Alicia | Theory |
| 2 | 2 | e-book | Dafni | CS |
| 2 | 2 | e-book | Dafni | Theory |
| 2 | 5 | paperback | Bo | Comedy |

# Question 4

The schema Order is not in BCNF. This is because it is not in 3NF a BCNF is also in BNF.

Because of this, I will use the algorithm DECOMPOSE-BCNF to put the schema into BCNF. First, I will construct the table and then explain each of the splits and show how the steps make the schema become a composition of binary relations. Here are the following steps:

1. We begin with R = (Id, Ui, Ul, D, B, F, Am, Is, T, Au, P, AR, Cf, Ca). Because ‘Id’ is not a key of this R, this means we must split. The relationship that will be used for the split is id → user\_id, user\_location, date. With this, I am left two relations R1 and R2 where R1 = (Id, Ui, Ul, D) and R2 = (Id, B, F, Am, is, T, Au, P, Ar, Cf, Ca).
2. The relational schema R1 is still not in BCNF. An example of a violation is user\_id → user\_location. As a result, we split again to get R1,1 and R1,2 where R1,1 = (Ui, Ul) and R1,2 = (Id, Ui, D). Both R1,1 and R1,2 are in BCNF as R1,1 is binary and R1,2 has the key Id.
3. R2 is not in BCNF. The relational dependency book\_id → title, publisher, age\_rating, child\_friendly violates BCNF. Another split occurs here to get R2,1 and R2,2 where R2,1 = (B, T, P, Ar, Cl) and R1,2 = (Id, B, F, Am, Is, Au, Ca).
4. With the previous split, I will split on R2,1 because it is not in BCNF as age\_rating → child\_friendly violates it. A split occurs here to get R2,1,1 and R2,1,2 where R2,1,1 = (Ar, Cf) and R2,1,2 = (B, T, P, Ar). R2,1,1 is binary, so it is in BCNF. R2,1,2 is in BCNF now because it has a key which is B.
5. On the rightmost part of the graph, R2,2 is still not in BCNF. Because of this, I will split on id, book\_id, format → amount. However, this split will be a bit different since I will be including isbn in the dependency that contains the previously aforementioned relation because a later dependency. As a result, I will get R2,2,1 and R2,2,2 where R2,2,1 = (Id, B, F, Am, Is) and R2,2,2 = (Id, B, F, Au, Ca). R2,2,2 is in BCNF.
6. R2,2,1 is still not in BCNF. Using the dependency book\_id, format → isbn, I get R2,2,1,1 and R2,2,1,2 where R2,2,1,1 = (B, F, Is) and R2,2,1,2 = (Id, B, F, Am). Finally, both are in BCNF since they each have a key, which are B, F and Id, B, F respectively.

With this, the following dependencies hold in the relational schema of the resulting decomposition:

|  |  |
| --- | --- |
| Relation Scheme | Functional Dependencies |
| R1,1(id, location) | user\_id → user\_location |
| R1,2(id, user\_id, date) | From split and id → user\_id, user\_location, date |
| R2,1,1(age\_rating, child\_friendly) | age\_rating → child\_friendly |
| R2,1,2(book\_id, title, publisher, age\_rating) | From split and dependency book\_id → title, publisher, age\_rating, child\_friendly |
| R2,2,1,1(book\_id, format, isbn) | book\_id, format → isbn  isbn → book\_id, format |
| R2,2,1,2(id, book\_id, format, amount) | From split and id, book\_id, format → amount |
| R2,2,2(id, book\_id, format, amount, category) |  |

The decomposition is a lossless-join since we used the DECOMPOSE-BCNF decomposition algorithm which guarantees a lossless-join decomposition. Surprisingly, this was also dependency preserving. This is because any dependency can be traced back. However, this is not always the case as DECOMPOSE-BCNF does not guarantee this.

Finally, the example dataset is decomposed with accordance to the information above:

## R1,1

|  |  |
| --- | --- |
| Ui | Ul |
| 1 | Hamilton |
| 2 | Toronto |
| 1 | Hamilton |

## R1,2

|  |  |  |
| --- | --- | --- |
| Id | Ui | D |
| 1 | 1 | 24 Dec. |
| 2 | 2 | 25 Dec. |
| 3 | 1 | 26 Dec. |

## R2,1,1

|  |  |
| --- | --- |
| Ar | Cf |
| 18+ | no |
| 5+ | yes |

## R2,1,2

|  |  |  |  |
| --- | --- | --- | --- |
| B | T | P | Ar |
| 2 | Book! | ThePrinter | 18+ |
| 5 | Comic! | TheCopier | 5+ |

## R2,2,1,1

|  |  |  |
| --- | --- | --- |
| B | F | Is |
| 2 | paperback | 1234 |
| 2 | hardcover | 1237 |
| 2 | e-book | 1241 |
| 5 | paperback | 1298 |

## R2,2,1,2

|  |  |  |  |
| --- | --- | --- | --- |
| Id | B | F | Am |
| 1 | 2 | paperback | 3 |
| 1 | 2 | paperback | 6 |
| 2 | 2 | e-book | 1 |
| 3 | 5 | paperback | 1 |

## R2,2,2

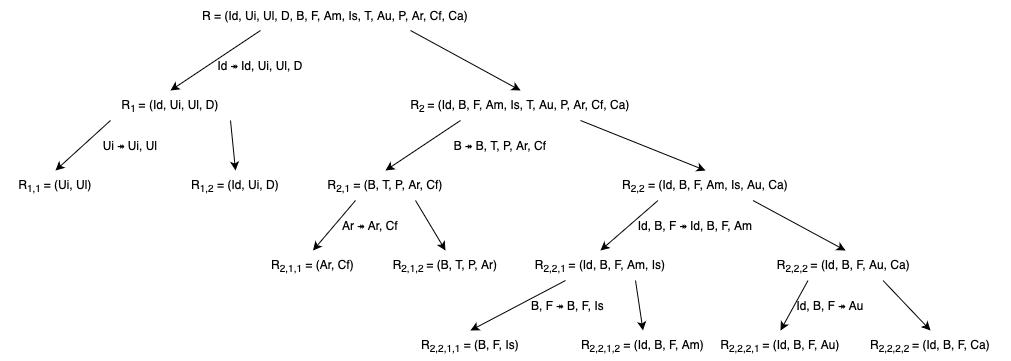
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | B | F | Au | Ca |
| 1 | 2 | paperback | Alicia | CS |
| 1 | 2 | paperback | Alicia | Theory |
| 1 | 2 | paperback | Dafni | CS |
| 1 | 2 | paperback | Dafni | Theory |
| 1 | 2 | hardcover | Dafni | CS |
| 1 | 2 | hardcover | Dafni | Theory |
| 1 | 2 | hardcover | Dafni | CS |
| 1 | 2 | hardcover | Dafni | Theory |
| 2 | 2 | e-book | Alicia | CS |
| 2 | 2 | e-book | Alicia | Theory |
| 2 | 2 | e-book | Dafni | CS |
| 2 | 2 | e-book | Dafni | Theory |
| 2 | 5 | paperback | Bo | Comedy |

# Question 5

Like the previous two question, this question is not in 4NF. This is because the table is not in BCNF. For something to be a 4NF, it must also be in BCNF as all 4NFs are BCNF.

To make this 4NF, I will use the DECOMPOSE-4NF algorithm to make this a 4NF. We use the violation α ↠ α+, where α+ is the attribute closure of αwith respect to only the functional dependencies. In addition to this, I will use the attribute that α → α+ implies α ↠ α+ because of replication.

I will reuse some of BCNF and along with that, I will evaluate non-binary relation schemes to determine if they introduce new dependencies for relational schemas.



The beginning is very similar to the BCNF as shown below with the steps and method.

1. We begin with R = (Id, Ui, Ul, D, B, F, Am, Is, T, Au, P, AR, Cf, Ca). Because ‘Id’ is not a key of this R, this means we must split. The relationship that will be used for the split is id ↠ id, user\_id, user\_location, which is obtained through replication of id → id+. With this, I am left two relations R1 and R2 where R1 = (Id, Ui, Ul, D) and R2 = (Id, B, F, Am, is, T, Au, P, Ar, Cf, Ca).
2. The relational schema R1 is still not in 4NF. An example of a violation is user\_id ↠ user\_id, user\_location, obtained from the replication of user\_id → user\_id+. As a result, we split again to get R1,1 and R1,2 where R1,1 = (Ui, Ul) and R1,2 = (Id, Ui, D). R1,1 is binary, so it is 4NF. R1,2 will be discussed later.
3. R2 is not in 4NF. The relational dependency book\_id ↠ book\_id, title, publisher, age\_rating, child\_friendly, derived from the replication of book\_id → book\_id+ violates 4NF. Another split occurs here to get R2,1 and R2,2 where R2,1 = (B, T, P, Ar, Cl) and R1,2 = (Id, B, F, Am, Is, Au, Ca).
4. With the previous split, I will split on R2,1 because it is not in 4NF as age\_rating ↠ age\_rating, child\_friendly violates it, derived from the replication of age\_rating → age\_rating+. A split occurs here to get R2,1,1 and R2,1,2 where R2,1,1 = (Ar, Cf) and R2,1,2 = (B, T, P, Ar). R2,1,1 is in 4NF since it is binary. I will discuss R2,1,2 later.
5. On the rightmost part of the graph, R2,2 is still not in 4NF. Because of this, I will split on id, book\_id, format ↠ id, book\_id, format, amount, derived from the replication of id, book\_id, format → (id, book\_id, format)+. However, this split will be a bit different since I will be including isbn in the dependency that contains the previously aforementioned relation because a later dependency. As a result, I will get R2,2,1 and R2,2,2 where R2,2,1 = (Id, B, F, Am, Is) and R2,2,2 = (Id, B, F, Au, Ca). R2,2,2 will be discussed later.
6. R2,2,1 is still not in 4NF. Using the dependency book\_id, format ↠ book\_id, format, isbn which was derived from replication of book\_id, format → (book\_id, format)+, I get R2,2,1,1 and R2,2,1,2 where R2,2,1,1 = (B, F, Is) and R2,2,1,2 = (Id, B, F, Am). Both will be discussed later.

From here one, I must analyze each of the non-binary scheme and discuss if it is in 4NF.

1. R2,2,2 = (Id, B, F, Au, Ca). We have “id → 𝔍”, which we can apply replication on this to get “id, B, F ↠ 𝔍”. Using transitivity on “id, B, F ↠ 𝔍” and “𝔍 ↠ author” to get “id, B, F ↠ author”. However, id, B, F are not keys, which means I must split to get R2,2,2,1 = (Id, B, F, Au) and R2,2,2,2 = (Id, B, F, Ca).
2. With the other multivalued dependencies, they do not introduce any new dependencies for the relational schemes. This includes R1,2 = (Id, Ui, D), R2,1,2 = (B, T, P, Ar), R2,2,1,1 = (B, F, Is), and R2,2,1,2 = (Id, B, F, Am). Because of this, all these schemes are in 4NF.

|  |  |
| --- | --- |
| Relation Scheme | Functional Dependencies |
| R1,1(id, location) | user\_id → user\_location |
| R1,2(id, user\_id, date) | From split and id → user\_id, user\_location, date |
| R2,1,1(age\_rating, child\_friendly) | age\_rating → child\_friendly |
| R2,1,2(book\_id, title, publisher, age\_rating) | From split and dependency book\_id → title, publisher, age\_rating, child\_friendly |
| R2,2,1,1(book\_id, format, isbn) | book\_id, format → isbn  isbn → book\_id, format |
| R2,2,1,2(id, book\_id, format, amount) | From split and id, book\_id, format → amount |
| R2,2,2,1(id, book\_id, format, author) |  |
| R2,2,2,2(id, book\_id, format, category) |  |

It is noted that non-trivial multivalued dependencies hold in this decomposition as that cannot be derived from the above functional dependencies. This decomposition is lossless-join since we strictly abided by the algorithm. However, this is not dependency preserving since “id, book\_id, format ↠ author” and “id, book\_id, format ↠ category” are not preserved since they cannot be derived from the functional dependencies in the above table.

Finally, here are the tables shown below:

## R1,1

|  |  |
| --- | --- |
| Ui | Ul |
| 1 | Hamilton |
| 2 | Toronto |
| 1 | Hamilton |

## R1,2

|  |  |  |
| --- | --- | --- |
| Id | Ui | D |
| 1 | 1 | 24 Dec. |
| 2 | 2 | 25 Dec. |
| 3 | 1 | 26 Dec. |

## R2,1,1

|  |  |
| --- | --- |
| Ar | Cf |
| 18+ | no |
| 5+ | no |

## R2,1,2

|  |  |  |  |
| --- | --- | --- | --- |
| B | T | P | Ar |
| 2 | Book! | ThePrinter | 18+ |
| 5 | Comic! | TheCopier | 5+ |

## R2,2,1,1

|  |  |  |
| --- | --- | --- |
| B | F | Is |
| 2 | paperback | 1234 |
| 2 | hardcover | 1237 |
| 2 | e-book | 1241 |
| 5 | paperback | 1298 |

## R2,2,1,2

|  |  |  |  |
| --- | --- | --- | --- |
| Id | B | F | Am |
| 1 | 2 | paperback | 3 |
| 1 | 2 | paperback | 6 |
| 2 | 2 | e-book | 1 |
| 3 | 5 | paperback | 1 |

## R2,2,2,1

|  |  |  |  |
| --- | --- | --- | --- |
| Id | B | F | Au |
| 1 | 2 | paperback | Alicia |
| 1 | 2 | paperback | Dafni |
| 1 | 2 | hardcover | Dafni |
| 1 | 2 | hardcover | Dafni |
| 2 | 2 | e-book | Alicia |
| 2 | 2 | e-book | Dafni |
| 3 | 5 | paperback | Bo |

## R2,2,2,2

|  |  |  |  |
| --- | --- | --- | --- |
| Id | B | F | Ca |
| 1 | 2 | paperback | CS |
| 1 | 2 | paperback | Theory |
| 1 | 2 | hardcover | CS |
| 1 | 2 | hardcover | Theory |
| 2 | 2 | e-book | CS |
| 2 | 2 | e-book | Theory |
| 3 | 5 | paperback | Comedy |

# Question 6

For each of these decompositions, not all issues are resolved. For the 4NF, author and category are in the same table as id, however there may not be a correlation between them so it would be better to model these with book\_id and format.

One small issue of redundancy is the child\_friendly and age\_rating. This is because if something is rated 18+, then it is not child friendly and if it is 5+, then it is. It is very redundant to use both and if you remove one, you can derive the it with the other field.