MovieDB System
Analysis of Relational Schemas

Arda Arslan 2020400078

Volkan Öztürk 2019400033

May 22, 2023

1 Introduction

In this paper, each schema defined to construct tables and relations for MovieDB System will be examined in order to determine functional dependencies present in the schema and then conclude the type of its form.
We divide present schemas in four categories:

1. Schemas having one attribute which is a primary key
2. Schemas having two attributes one of which is a primary key
3. Schemas having two attributes which together form a composite key
4. Schemas having three attributes.
5. Schemas having four attributes.

Since Boyce-Codd Normal Form is the most desired normal form we will demonstrate if relations comply with the definition given in the next section.

2 Definition of BCFN Form

Let R be a relation schema, F be the set of functional dependencies given to hold over R, X be a subset of the attributes of R, and A be an attribute of R. R is in Boyce-Codd Normal Form if, for every functional dependency $X \rightarrow A$ in F, one of the following statements is true:

- $A \in X$; that is, it is a trivial functional dependency, (2.1)
- X is a superkey. (2.2)

3 Schemas

3.1 Schemas having one attribute which is a primary key

- audience(<u>username</u>)
- movie_session(<u>session_id</u>)

For all relation schemas R listed above, we can denote R as X . Only integrity constraint that is known to hold in above schemas is primary key constraint with X being the key. Thus only functional dependency we can write is:

$$(i) \quad X \rightarrow X$$

Since X is a key and a primary key is also a superkey, the condition (2.2) given in the definition at the Section 2 is satisfied. Thus listed relations are in BCNF form. Note that functional dependency *(i)* is trivial because $X \subseteq X$ holds.

3.2 Schemas having two attributes one of which is a primary key

- director(<u>username</u>, nationality)
- rating_platform(<u>platform_id</u>,platform_name)
- genre(<u>genre_id</u>, genre_name)
- directs(<u>movie_id</u>, username)

For all relation schemas R listed above, we can denote R as XY . Only integrity constraint that is known to hold in above schemas is primary key constraint with X being the key. For each schema, no other functional dependency is created between attributes which is a result of our design choice thus only functional dependency we can write is:

$$X \rightarrow XY$$

We can infer several additional functional dependencies that are in the closure by using decomposition.

$$(i) \quad X \rightarrow X$$

$$(ii) \quad X \rightarrow Y$$

Since X is a key and a primary key is also a superkey, the condition (2.2) given in the definition at the Section 2 is satisfied for functional dependencies *(i),(ii)* obtained above after decomposition. Thus listed relations are in BCNF form. Note that functional dependency *(i)* is trivial because $X \subseteq X$ holds.

3.3 Schemas having two attributes which together form a composite key

- predecessors(<u>predecessor _movie_id</u>, <u>succeeds _movie_id</u>)
- subscriptions(<u>username</u>, <u>platform_id</u>)
- registered(<u>username</u>,<u>platform_id</u>)
- place(<u>session_id</u>, <u>occupience_id</u>)
- genre_list(<u>movie_id</u>, <u>genre_id</u>)
- bought_tickets(<u>username</u>, <u>genre_id</u>)
- has_movie(<u>session_id</u>,<u>movie_id</u>)

For all relation schemas R listed above, we can denote R as XY . Only integrity constraint that is known to hold in above schemas is primary key constraint with XY being the key. For each schema, no other functional dependency is created between attributes which is a result of our design choice thus only functional dependency we can write is:

$$XY \rightarrow XY$$

We can infer several additional functional dependencies that are in the closure by using decomposition.

$$(i) \quad XY \rightarrow X$$

$$(ii) \quad XY \rightarrow Y$$

Since $X \in XY$ and $Y \in XY$, the condition (2.1) given in the definition at the Section 2 is satisfied for functional dependencies *(i),(ii)*, which makes them trivial FDs. Thus listed relations are in BCNF form. Note that the condition (2.1) is also satisfied for both FDs.

3.4 Schemas having three attributes.

- database_manager(username, manager_number password)

For the relation schema Rabove, we can denote R as XYZ . Only integrity constraint that is known to hold in above schema is primary key constraint with X being the key. No other functional dependency is created between attributes which is a result of our design choice thus only functional dependency we can write is:

$$X \rightarrow XYZ$$

We can infer several additional functional dependencies that are in the closure by using decomposition.

$$(i) \quad X \rightarrow X$$

$$(ii) \quad X \rightarrow Y$$

$$(iii) \quad X \rightarrow Z$$

Since X is a key and a primary key is also a superkey, the condition (2.2) given in the definition at the Section 2 is satisfied for functional dependencies *(i),(ii),(iii)* obtained above after decomposition. Thus listed relations are in BCNF form. Note that functional dependency *(i)* is trivial because $X \subseteq X$ holds.

3.5 Schemas having four attributes.

- user1(username, password, surname, name)
- movie(movie_id, movie_name, overall_rating, duration)
- theater(theater_id, district, theater_capacity, theater_name)
- ratings(rating_id, username, movie_id, rating)
- occupience_info(occupience_id, theater_id, slot_number, time1)

For all relation schemas R listed above, we can denote R as $XYZW$. Only integrity constraint that is known to hold in above schemas is primary key constraint with X being the key. For each schema, no other functional dependency is created between attributes which is a result of our design choice thus only functional dependency we can write is:

$$X \rightarrow XYZW$$

We can infer several additional functional dependencies that are in the closure by using decomposition.

$$(i) \quad X \rightarrow X$$

$$(ii) \quad X \rightarrow Y$$

$$(iii) \quad X \rightarrow Z$$

$$(iv) \quad X \rightarrow W$$

Since X is a key and a primary key is also a superkey, the condition (2.2) given in the definition at the Section 2 is satisfied for four functional dependencies *(i),(ii),(iii),(iv)* obtained above after decomposition. Thus listed relations are in BCNF form. Note that functional dependency *(i)* is trivial because $X \subseteq X$ holds.

4 Conclusion

We were able to group all relational schemas used in our design into 5 different categories. Within all categories exact same notation is used to represent schemas thanks to our design choice of not including any other functional dependency within a relation other than one satisfying the key constraint. After examining FDs in the closure we conclude that all schemas are in Boyce-Codd Normal Form. Further refinement of schemas is not possible.