Akira Takada, Jason Chen, Luke Rogers, Julia Bechtel

Stage IV – Elaboration: Design

Demonstrate that all the relations in the relational schema are normalized to Boyce–Codd normal form (BCNF).

- INTERVIEW Table:
  - This table is in BCNF because we have the primary key id which all of the other attributes depend on, meaning we would not know what the title or audio for example of an interview is without the id. The conditions below are also all satisfied:
  - 1NF: All attributes (title, date, audio, thumbnail, script, uID) depend on the id primary key
  - 2NF: Being the only single primary key, it does not have multiple attributes.
  - 3NF: All the nonkey attributes do not depend on each other aside from the primary key, id (i.e date does not depend on the audio nor the thumbnail, etc.)
- ASSETS Table:
  - ASSETS table is in BCNF because all the conditions below are satisfied:
  - 1NF: The attributes timestamp, hyperlink, image, and text all depend of the primary key fid
  - 2NF: The candidate key {Fid, timestamp} is depended on by all nonkey attributes. The nonkey attributes need the timestamp key as well to determine which time under the specific interview id it is relevant to. Having Fid as the sole key will not help determine which tuple the nonkey attributes are related to.
  - 3NF: All of the nonkey attributes only depend on the keys in the table and not on each other, the other nonkey attributes
  - BCNF: {Fid, timestamp} are prime attributes and is a functional dependency of all the nonkey attributes and {Fid, timestamp} is also a superkey therefore it is in BCNF
- USER
  - 1NF: Both attributes username and password depend on the key userID
  - 2NF: All the nonkey attributes depend on the whole key as there is no partial key dependencies
  - 3NF: Username and password attributes do not depend on each other and only depend on the key userID
  - BCNF: being that username and password only depend on the one key userID, this design also fulfills BCNF
- ADMINISTRATOR
  - 1NF: The nonkey attribute permissions depend on the userID
  - 2NF: There are no multi-attribute keys in this relation
  - 3NF: There is only a single nonkey attribute that depends on the key, userID
  - BCNF: permissions are the only nonkey attribute which solely relies on the primary key userID
- CREATES

- 1NF: Since the attributes of CREATES all depend on the key then it is in 1NF
- 2NF: There are no partial key dependencies with the attributes in this table
- 3NF: Similar to both the USER and ADMINISTRATOR tables, all the nonkey attributes cannot depend on each other but rather only on the key
- BCNF: permissions are the only nonkey attribute which solely relies on the primary key userID

Define the different views (virtual tables) required. For each view list the data and transaction requirements. Give a few examples of queries, in English, to illustrate.

- A view called *interview_view* that fetches the assets from the ASSETS table which includes fid, timestamp, hyperlink, image, and text for a given interview. This creates a view of all the assets that are related to the selected interview. An example query is to create the *interview_view* query that selects the attributes timestamp, hyperlink, image, and text from the ASSETS table for the interview with ID as 0. This will list all of that specific interview's corresponding assets
- A view called *admin_info_view* that fetches the username and userID. This creates the view of all the usernames and IDs of all administrators. An example query is to create the *admin_info_view* view that uses the attributes userID and username from the USER table and right join it with ADMINISTRATOR. This results in the view of username and ID of all valid administrators

Design a complete set of SQL queries to satisfy the transaction requirements identified in the previous stages, using the relational schema and views defined in tasks 2 and 3 above.

```
CREATE VIEW   interview_view   AS
SELECT        Fid, timestamp, hyperlink, image, text
FROM          ASSETS
WHERE         Fid=0;

CREATE VIEW   admin_info_view AS
SELECT        userID, username
FROM          USER
RIGHT JOIN    ADMINISTRATOR
ON USER.userID = ADMINISTRATOR.userID
```