Main Project

COMP 353 Section CD

Aydin Azari Farhad

Deniz Dinchdonmez

Jason Forte

Ryan Feher

# Section 1

In developing our schema, we had to make a number of assumptions about requirements that either weren’t specified at all or left ambiguous. The following is a summary of our planning and thought process for each relation in the final schema.

### Regions

Straight forward relation, only has one attribute which is the region name. Already normalized to 3NF.

### Countries

Since a region (continent) has multiple countries, we established a 1-many relationship between Countries and Regions, where Country references regions. Each country also has a 1-1 relationship with an organization, so Organizations gets referenced too. Every country is assumed to have a unique name, so with those 3 attributes and countryName as the candidate key we’ve arrived at a normalized relation.

### ProStaTers

Just like how Regions has a 1-many relationship with Countries, Countries has a 1-many relationship with ProStaTers. Every prostater has a name, but we assumed there’s a possibility that 2 prostaters from different countries can have the same name. We also assumed that no 2 prostaters would have the same name in the same country, so prostaterName and countryName make up the candidate key for this relation. This relation is in 3NF.

### Reports

Reports is where government agencies store COVID vaccinations, infections, and deaths in the country by province/state/territory and vaccine type over time. If vaccine type is None, then that means the record is keeping track of unvaccinated covid cases. Originally we had Cases and Vaccinations as separate relations that would be referenced in the Reports relation, which we originally called UpdateLogs. In the end, we chose to cut those relations since they only held the latest information on covid cases and vaccinations per province/state/territory, which the Reports relation already stores. Each record is uniquely identified by the government agency who uploaded the record and the datetime it was uploaded. Datetime and gov agency will always be unique but prostater and vaccine can repeat since this relation stores the covid and vaccine status over time. This relation is in 3NF.

### Vaccines

This relation only has one attribute and that’s the name of each type of vaccine, including None, which is used by the Reports relation to track covid cases of unvaccinated people. As it has only one attribute it is already in 3NF.

### Articles

According to the project requirements, the Articles relation needed to reference the author, who could either be a researcher or an organization, and if they are a researcher, they should be identified by their first and last name. We assumed since the author is a researcher and researchers are a defined role within our system, every article published would be authored by either a user or organization stored in the database. Therefore, we couldn’t let the author attribute be one field since it would have to reference 2 different relations, so we had articles identified by AuthorName and Organization, but to not break 1NF, we had to define the author by authorUsername and organization. Since the author can be a user, their first and last names can be obtained by referencing the Users relation, and organization is referenced by orgID in the Organizations relation. Once we worked out how authors would be defined, coming up with a candidate key came easy and we ended up with a relation that was already normalized into 3NF.

### Users

Since every user has a username we assumed the username would have to be unique, which is typical for many websites where users can login to, and therefore username was a fitting minimal candidate key. We also made user-role an enum that can either be a regular user, administrator, researcher, or delegate, and the rights they are entitled to would be handled in the controller logic. In our original design for Users, we had defined user roles by their privileges and defined a separate relation that would map users to a right they had over a subject, which could either be another user or an article. In this design, finding administrators in the system, for example, would mean searching for users in the Privileges relations which had add/delete/edit rights over articles and users. However due to time constraints, we cut out this relation and moved permission handling in the controller. The user role, as well as all other attributes in the final schema, are solely determined by the username and nothing else, making this relation 3NF.

### EmailLogs

This relation is where automated emails sent to users who have subscribed to an author would be stored. Each email log has the recipient’s username and the datetime the email was sent, as well as the subject and the body of the email. The candidate key for this relation is username and datetime, and it’s in 3NF.

### Subscriptions

Subscriptions is a relationship between users and authors, which can be researchers (who are also users), and organizations. Each subscription holds the username of the subscriber, the username of the author they are subscribed to (if the author is a researcher), and the organization id (if the author is an organization). Every attribute in this relation is prime, therefore the relation is automatically in 3NF.

### Employees

Employees is a relationship between users and organizations. It conveys what organizations users such as employees and delegates are employed at. Since every attribute in this relation is prime the relation is automatically in 3NF.

### Organizations

This relation holds all relevant information about the organizations in the system, which is the organization name and the organization type. We assumed there’s a reasonable possibility that organizations of the same type with the same name might exist, so we added an organization ID to uniquely identify every organization in the system. This relation is in 3NF.

# Section 2.1

# 

# Section 2.2

| **Name** | **Contributions** |
| --- | --- |
| Jason Forte | Created the first draft of the ER diagram.  Worked on updating the ER diagram.  Created the written portion of the report.  Created the FDs for the schema.  Contributed to the PHP programming. |
| Aydin Azari Farhad | Created summary of requirements.  Worked on creating the ER diagram.  Created tables.  Created test data.  Populated tables with test data.  Debugged data structure to meet requirements.  Maintained ER diagram and database against updated approaches. |
|  |  |
|  |  |