Exercise 4) Simple Normalization

There is a website that searches daily for good deals on ebooks. You can signup for this service, and they will send you an email regarding today's deals. Your notifications can be narrowed down by author or genre. You can specify when your favorite authors release a new book. You can also specify what genres interest you.
Normalize this database into 3NF. Draw the ER diagram and relational schema. Include the SQL CREATE statements for all entities designed.

Username	Home email	Work email	School Email	Genre	Author
Mimi	mimi@home.com	mimi@work.com	mimi@school.com	Mystery, Thriller, Suspense, Crime	James Patterson, Robert Ludlum, Jeffery Archer, Ken Follett
Jack	jack@home.com		jack@school.com	Short Stories, Classic Fiction	Alexandre Dumas
Rainn			rainn@school.com	Rhyming Fiction	Dr. Seuss, PD Eastman

Stages of Normalization
Normalization is a process in database design that organizes tables to reduce redundancy and improve data integrity.

1. **First Normal Form (1NF):**
    - Ensure that each column contains only atomic (indivisible) values.
    - Each record must be unique.
2. **Second Normal Form (2NF):**
    - Ensure that the database is in 1NF.
    - Remove partial dependencies, meaning all non-key attributes should depend on the entire primary key.
3. **Third Normal Form (3NF):**
    - Ensure that the database is in 2NF.
    - Remove transitive dependencies, meaning non-key attributes should not depend on other non-key attributes.

Initial Data Structure

Initially, the data was presented in a single table format with multiple email addresses, genres, and authors stored in a non-atomic form. This format contained redundancy and lacked proper structure for relational database management.

***Step 1: First Normal Form (1NF)* - Convert to 1NF - Ensure atomic values:**

- **Objective:** Ensure that each column contains only atomic (indivisible) values and each record is unique.
- **Action Taken:**
    - Separated multiple email addresses, genres, and authors into individual rows.
    - Ensured that each field contains only a single value.
- **Result:**
    - Created a table with each user having a single email, genre, and author entry per row.

Username	Email	Genre	Author
Mimi	mimi@home.com	Mystery	James Patterson
Mimi	mimi@work.com	Thriller	Robert Ludlum
Mimi	mimi@school.com	Suspense	Jeffery Archer
Jack	jack@home.com	Short Stories	Alexandre Dumas
Jack	jack@school.com	Classic Fiction	
Rainn	rainn@school.com	Rhyming Fiction	Dr. Seuss
Rainn	rainn@school.com	Rhyming Fiction	PD Eastman

Step 2: Second Normal Form (2NF) - Convert to 2NF: 2NF - Remove partial dependencies:

- **Objective:** Ensure that the database is in 1NF and remove partial dependencies, meaning all non-key attributes should depend on the entire primary key.
- **Action Taken:**
    - Split the data into multiple related tables to eliminate partial dependencies.
    - Created separate tables for **AppUser**, **Email**, **Genre**, **Author**, **UserGenre**, and **UserAuthor**.
    - Introduced a unique identifier (**user_id**) for each user.
- **Results:**

- Established relationships between users and their emails, genres, and authors.

AppUser Table:

user_id	username
1	Mimi
2	Jack
3	Rainn

Email Table:

email_id	user_id	email	email_type_id
1	1	mimi@home.com	1
2	1	mimi@work.com	2
3	1	mimi@school.com	3
4	2	jack@home.com	1
5	2	jack@school.com	3
6	3	rainn@school.com	3

Genre Table:

genre_id	genre_name
1	Mystery
2	Thriller
3	Suspense
4	Short Stories
5	Classic Fiction
6	Rhyming Fiction

Author Table:

author_id	author_name
1	James Patterson
2	Robert Ludlum
3	Jeffery Archer
4	Alexandre Dumas
5	Dr. Seuss
6	PD Eastman

UserGenre Table:

user_id	genre_id
1	1
1	2
1	3
2	4
2	5
3	6

UserAuthor Table:

user_id	author_id
1	1
1	2
1	3
2	4

user_id	author_id
3	5
3	6

Step 3: Third Normal Form (3NF)

- **Objective:** Ensure that the database is in 2NF and remove transitive dependencies, meaning non-key attributes should not depend on other non-key attributes.
- **Action Taken:**
    - Ensured that all non-key attributes depend only on the primary key.
    - Refined the relationships between the tables to maintain data integrity and reduce redundancy.
- **Result:**
    - Finalized the schema with tables for users, emails, email types, genres, authors, user genres, and user authors, with proper foreign key constraints to maintain relationships.

Final Tables in 3NF:

- AppUser Table
- Email Table
- EmailType Table
- Genre Table
- Author Table
- UserGenre Table
- UserAuthor Table

Normalization Summary

By normalizing the data to 3NF, we ensured that each piece of information is stored only once, minimizing redundancy. The data is now structured in a way that each table has a single purpose, and relationships between tables are maintained through foreign keys. This structure enhances data integrity, consistency, and scalability.

Descriptions of Tables and Attributes

AppUser Table

- Table Name: AppUser
    - user_id (Primary Key, INT): An integer value uniquely identifying each user.
    - username (VARCHAR(100), Not Null): A string representing the username of the user.

Email Table

- Table Name: Email
    - email_id (Primary Key, INT): An integer value uniquely identifying each email entry.
    - user_id (Foreign Key, INT): An integer value referencing the user.
    - email (VARCHAR(100), Not Null): A string representing the email address.
    - email_type_id (Foreign Key, INT): An integer value referencing the email type.

EmailType Table

- Table Name: EmailType
    - email_type_id (Primary Key, INT): An integer value uniquely identifying each email type.
    - type (VARCHAR(50), Not Null): A string representing the type of email.

Genre Table

- Table Name: Genre

- genre_id (Primary Key, INT): An integer value uniquely identifying each genre.
- genre_name (VARCHAR(100), Not Null): A string representing the name of the genre.

Author Table

- Table Name: Author
    - author_id (Primary Key, INT): An integer value uniquely identifying each author.
    - author_name (VARCHAR(100), Not Null): A string representing the name of the author.

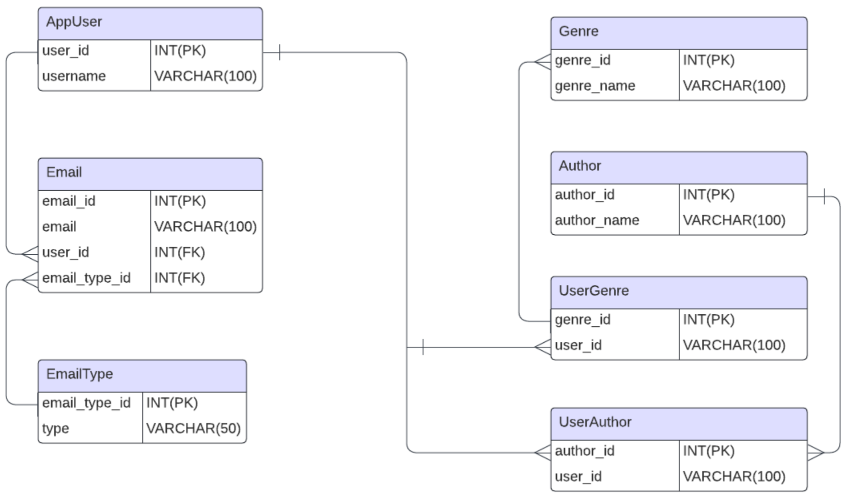
UserGenre Table

- Table Name: UserGenre
    - user_id (Foreign Key, INT): An integer value referencing the user.
    - genre_id (Foreign Key, INT): An integer value referencing the genre.
    - PRIMARY KEY (user_id, genre_id): Composite primary key.

UserAuthor Table

- Table Name: UserAuthor
    - user_id (Foreign Key, INT): An integer value referencing the user.
    - author_id (Foreign Key, INT): An integer value referencing the author.
    - PRIMARY KEY (user_id, author_id): Composite primary key.

Entity Relationship Diagram (ERD)



Relationship Summary

- **AppUser to Email (1:N):** One user can have multiple email addresses.
- **Email to EmailType (N:1):** Each email address has one type, but each type can apply to many emails.
- **AppUser to UserGenre (1:N):** One user can be interested in multiple genres.
- **UserGenre to Genre (N:1):** Each genre can be referenced by multiple users.
- **AppUser to UserAuthor (1:N):** One user can follow multiple authors.
- **UserAuthor to Author (N:1):** Each author can be followed by multiple users.