INFO 6210 Data Management and Database Design Project - FALL 2020

DATABASE DESIGN FOR A RENTAL E-COMMERCE PLATFORM

SUMMARY:

Database design for a rental e-commerce platform, where users can create a listing for an item or a service that they wish to rent. Other users can view the listing and contact the owner of the listing to work out a deal. When a deal is finalized, the user can place an order and the product will be delivered to the user. At the end of the rental period, the product will be obtained from the user and will be provided back to the owner of the product.

OBJECTIVES:

To design a database for a rental platform which should support the following features:

- Users must be able to login/register and user details such as username, email, password, address etc., must be stored.
- Users must be able to create a new listing and view existing listings
- Users must be able to propose and accept deals for a listing
- Order details must be stored, and users and employees must be able to access the order details.
- Maintain details of employees such as email, first name, last name, designation, pay etc.,

PROBLEM STATEMENT:

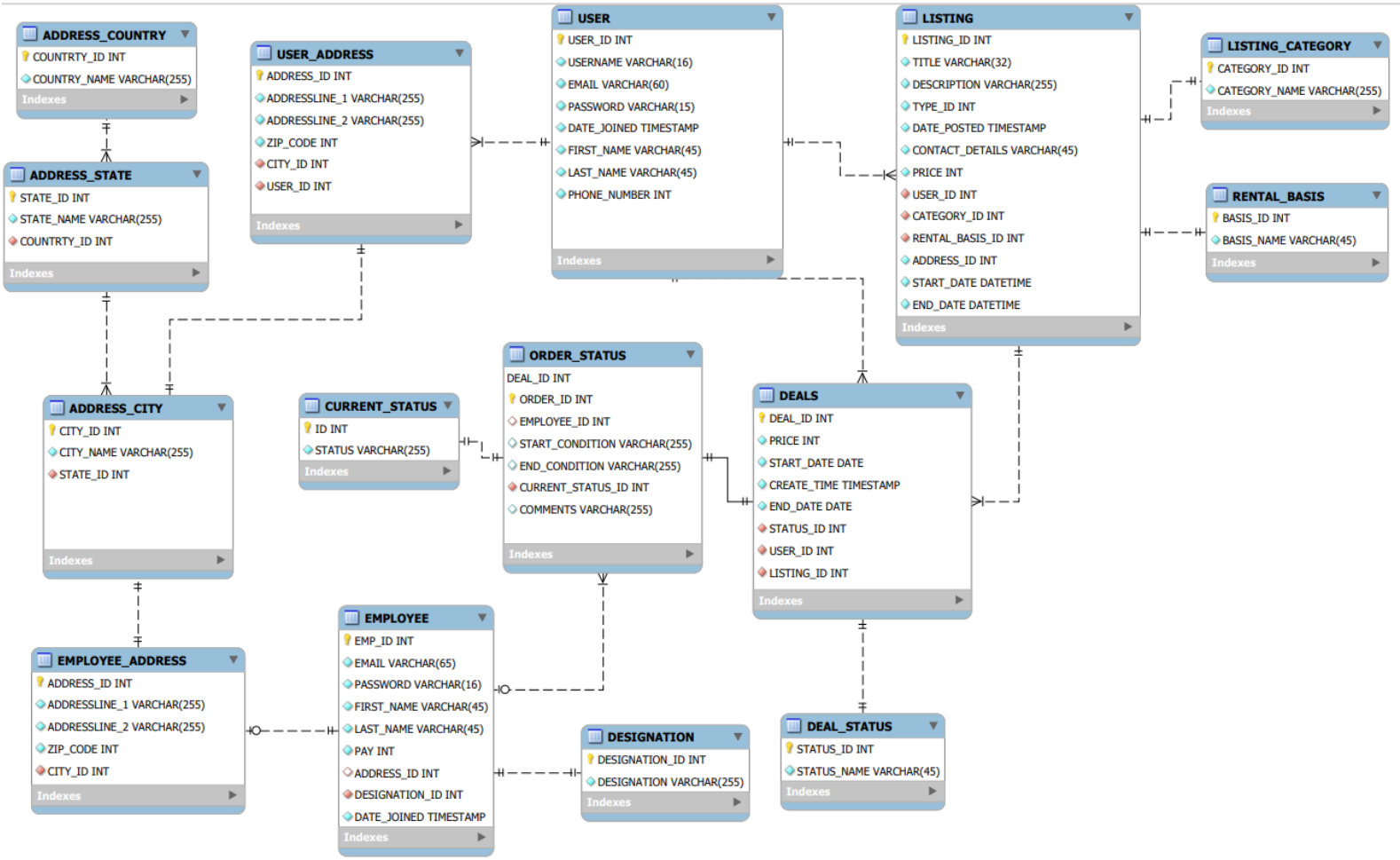
1. It is time consuming to manually remove the posts associated with a user's account when a user has deleted their account.
2. Unable to quickly retrieve all the listings posted by a particular user, since the users and listings are stored in separate files.
3. Multiple users are signing-up with the same username and it is difficult to prevent this by manually searching the existing usernames. This causes data inconsistency.
4. It is in-efficient and time-consuming to manually search and obtain the User and listing owner details of a deal.

PROPOSED SOLUTION:

To overcome the above problems, a database model can be used. The issues are resolved as follows:

1. Using a database design, a FOREIGN KEY reference with ON DELETE CASCADE property can be used to automatically delete all listings associated with a user, when a user deletes his/her account.
2. Using a database design, even though the Users and Listings entity are separate, we can join the User and Listing entities using the User ID column and obtain the data from both the entities. Thus, all listings posted by a user can be obtained.
3. Instead a manually searching for existing usernames, we can take advantage of the UNIQUE key constraint in databases to prevent users from signing-up with the same username. This guarantees data consistency.
4. With a database design, we can quickly obtain the user and listing owner details by including the User and Listing ID, which is a PRIMARY KEY that uniquely identifies a record.

DATA MODEL E-R DIAGRAM:



USER ENTITY

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
USER_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
USERNAME	VARCHAR (16)	UNIQUE KEY, NOT NULL
EMAIL	VARCHAR (60)	UNIQUE KEY, NOT NULL
PASSWORD	VARCHAR (15)	NOT NULL
DATE_JOINED	TIMESTAMP	CURRENT_TIMESTAMP
FIRST_NAME	VARCHAR (45)	NOT NULL
LAST_NAME	VARCHAR (45)	NOT NULL
PHONE_NUMBER	INTEGER	NOT NULL

LISTING ENTITY

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
LISTING_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
TITLE	VARCHAR (32)	NOT NULL
DESCRIPTION	VARCHAR (255)	NOT NULL
DATE_POSTED	TIMESTAMP	CURRENT_TIMESTAMP
START_DATE	DATE	NOT NULL
END_DATE	DATE	NOT NULL
CONTACT_DETAILS	VARCHAR (45)	NOT NULL
PRICE	INTEGER	NOT NULL
USER_ID	INTEGER	A Foreign Key which REFERENCES USER_ID from the USER ENTITY.

		This is used to fetch the details of the user. NOT NULL
RENTAL_BASIS_ID	INTEGER	A Foreign Key which REFERENCES BASIS_ID from the RENTAL_BASIS ENTITY. This is used to fetch the rental basis details. NOT NULL
CATEGORY_ID	INTEGER	A Foreign Key which REFERENCES CATEGORY_ID from the LISTING_CATEGORY ENTITY. This is used to fetch the category details of the Listing. NOT NULL

DEALS ENTITY

ATTRIBUTES	**DATA TYPE AND SIZE**	**COMMENTS**
DEAL_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
PRICE	INTEGER	NOT NULL
START_DATE	DATE	NOT NULL
CREATE_TIME	TIMESTAMP	CURRENT_TIMESTAMP
END_DATE	DATE	NOT NULL
STATUS_ID	INTEGER	A Foreign Key which REFERENCES STATUS_ID from the DEAL_STATUS ENTITY. This is used to fetch the status details of the listing. NOT NULL
USER_ID	INTEGER	A Foreign Key which REFERENCES USER_ID from the USER ENTITY. This is used to fetch the details of the user. NOT NULL

LISTING_ID	INTEGER	A Foreign Key which REFERENCES LISTING_ID from the LISTING ENTITY. This is used to fetch the listing details of the deal. NOT NULL

ORDER_STATUS ENTITY

ATTRIBUTES	**DATA TYPE AND SIZE**	**COMMENTS**
ORDER_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
START_CONDITION	VARCHAR (255)	NOT NULL
END_CONDITION	VARCHAR (255)	NOT NULL
CREATE_TIME	TIMESTAMP	CURRENT_TIMESTAMP
CURRENT_STATUS	INTEGER	A Foreign Key which REFERENCES CURRENT_STATUS_ID from the CURRENT_STATUS ENTITY. This is used to fetch the status details of the order. NOT NULL
EMPLOYEE_ID	INTEGER	A Foreign Key which REFERENCES EMP_ID from the EMPLOYEE ENTITY. This is used to fetch the details of the employee. NULL ALLOWED
DEAL_ID	INTEGER	A Foreign Key which REFERENCES DEAL_ID from the DEALS ENTITY. This is used to fetch the details of the deal. NOT NULL

CURRENT_STATUS ENTITY:

ATTRIBUTES	**DATA TYPE AND SIZE**	**COMMENTS**
ID	INTEGER	PRIMARY KEY, AUTO GENERATED
STATUS	VARCHAR (255)	NOT NULL

EMPLOYEE ENTITY

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
EMP_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
EMAIL	VARCHAR (65)	UNIQUE, NOT NULL
PASSWORD	VARCHAR (16)	NOT NULL
DATE_JOINED	TIMESTAMP	CURRENT_TIMESTAMP
FIRST_NAME	VARCHAR (45)	NOT NULL
LAST_NAME	VARCHAR (45)	NOT NULL
PAY	INTEGER	NOT NULL
DESIGNATION_ID	INTEGER	A Foreign Key which REFERENCES DESIGNATION_ID from the DESIGNATION ENTITY. This is used to fetch the details of the employee's designation. NOT NULL
ADDRESS_ID	INTEGER	A Foreign Key which REFERENCES ADDRESS_ID from EMPLOYEE_ADDRESS ENTITY. This is used to fetch the address details of the employee. NULL ALLOWED

USER_ADDRESS ENTITY

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
ADDRESS_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
ADDRESSLINE_1	VARCHAR (255)	NOT NULL
ADDRESSLINE_2	VARCHAR (255)	NOT NULL
ZIP_CODE	INTEGER	NOT NULL

CITY_ID	INTEGER	A Foreign Key which REFERENCES CITY_ID from the ADDRESS_CITY ENTITY. This is used to fetch the details of city. NOT NULL
USER_ID	INTEGER	A Foreign Key which REFERENCES USER_ID from the USER ENTITY. This is used to associate the address to a user. NOT NULL

EMPLOYEE_ADDRESS ENTITY

ATTRIBUTES	**DATA TYPE AND SIZE**	**COMMENTS**
ADDRESS_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
ADDRESSLINE_1	VARCHAR (255)	NOT NULL
ADDRESSLINE_2	VARCHAR (255)	NOT NULL
ZIP_CODE	INTEGER	NOT NULL
CITY_ID	INTEGER	A Foreign Key which REFERENCES CITY _ID from the ADDRESS_CITY ENTITY. This is used to fetch the details of city. NOT NULL

ADDRESS_CITY ENITY:

ATTRIBUTES	**DATA TYPE AND SIZE**	**COMMENTS**
CITY_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
CITY_NAME	VARCHAR (255)	NOT NULL
STATE_ID	INTEGER	A Foreign Key which REFERENCES STATE_ID from the ADDRESS_STATE ENTITY. This is

		used to fetch the details of the state. NOT NULL

ADDRESS_STATE ENTITY:

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
STATE_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
STATE_NAME	VARCHAR (255)	NOT NULL
COUNTRY_ID	INTEGER	A Foreign Key which REFERENCES COUNTRY_ID from the ADDRESS_COUNTRY ENTITY. This is used to fetch the details of the country. NOT NULL

ADDRESS_COUNTRY ENTITY:

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
COUNTRY_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
COUNTRY_NAME	VARCHAR (255)	NOT NULL

LISTING_CATEGORY ENTITY:

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
CATEGORY_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
CATEGORY_NAME	VARCHAR (255)	NOT NULL

RENTAL_BASIS ENTITY:

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
BASIS_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
BASIS_NAME	VARCHAR (45)	NOT NULL

DEAL_STATUS ENTITY:

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
STATUS_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
STATUS_NAME	VARCHAR (45)	NOT NULL

DESIGNATION ENTITY:

ATTRIBUTES	DATA TYPE AND SIZE	COMMENTS
DESIGNATION_ID	INTEGER	PRIMARY KEY, AUTO GENERATED
DESIGNATION	VARCHAR (255)	NOT NULL

Business Rules

1. **USER**s can create one or more **LISTING**s
2. A **LISTING** is created by only one **USER**
3. A **LISTING** can have its pricing as **RENTAL_BASIS** either per-hour or per-day or per-week or per-month
4. A **LISTING** can be created under **LISTING_CATEGORY** like Electronics, Clothing, Books, Sports, Tools, Vehicle, Heavy Machines and Others based on the item rented
5. Every **LISTING** has a **USER_ADDRESS** associated with it. It cannot be NULL
6. A **USER** interested in a **LISTING** can rent the item by ordering directly or submit a **DEAL** on that **LISTING** as a counteroffer with the **USER**'s specific preferences (price, rental-duration)
7. A **USER** can submit zero or more **DEAL**s offer on a **LISTING**
8. A **DEAL** offer can have only one **USER** as owner
9. A **LISTING** can have zero or more **DEAL**s submitted as counteroffer
10. **DEAL_STATUS** can have only three states: Pending, Approved, Rejected
11. The **LISTING**'s owner should approve only one **DEAL** and when a deal is approved, all other remaining **DEAL**s submitted for that **LISTING** are automatically rejected.

12. An **ORDER** is made either when a **USER** buys a **LISTING** or when a **USER**'s **DEAL** gets **DEAL_STATUS** approved by the **LISTING**'s owner
13. When a **LISTING** is booked directly, a **DEAL** is auto created, and an **ORDER** is made for that **DEAL**
14. A **LISTING**'s approved **DEAL** has only one **ORDER** created for it
15. An **ORDER** is associated only with one **DEAL**
16. Every **ORDER** has one **ORDER_STATUS** associated with it having all the details of order delivery, order pickup, rented item's condition prior delivery and after pickup etc.
17. **CURRENT_STATUS** can have only six states: Pending employee assignment, Delivery In-transit, Delivered/Rented, Pickup In-transit, Complete, disputed.
18. An **ORDER** is delivered by only one **EMPLOYEE**
19. An **EMPLOYEE** can deliver zero or more **ORDER**s
20. An **EMPLOYEE** can have only one **DESIGNATION**
21. A **DESIGNATION** can be given to one or more **EMPLOYEE**s
22. **DESIGNATION** can have only two values: Delivery executive and Manger.
23. An **EMPLOYEE** has only one **EMPLOYEE_ADDRESS**
24. An **EMPLOYEE_ADDRESS** is located only in one **ADDRESS_CITY**
25. A **USER_ADDRESS** is located only in one **ADDRESS_CITY**
26. An **ADDRESS_CITY** can have one or more **USER_ADDRESS** where the rental platform is used
27. Our rental platform service is available only in few **ADDRESS_CITY**s like Boston, New York, etc.
28. An **ADDRESS_ STATE** can have one or more **ADDRESS_ CITY**s where the rental platform is available
29. An **ADDRESS_ CITY** can be located only in one **ADDRESS_ STATE**
30. Our rental platform service is available only in just four **ADDRESS_STATE**: Massachusetts, New York, California, Virginia
31. An **ADDRESS_COUNTRY** can have one or more **ADDRESS_STATE**s where the rental platform is available
32. An **ADDRESS_STATE** can be located only in one **ADDRESS_COUNTRY**
33. Our rental platform service is available only in **ADDRESS_COUNTRY** USA

SECURITY CONSTRAINTS: (User level Access/Permissions)

USER:

1. Does not have any access to EMPLOYEE, EMPLOYEE_ADDRESS, DESIGNATION tables.
2. Has only READ access to ADDRESS_COUNTRY, ADDRESS_STATE, ADDRESS_CITY, ORDER_STATUS tables.
3. Has READ/WRITE/UPDATE access to USER_ADDRESS, LISTING, DEAL tables.

ADMIN:

1. Has full access to all the entities in the database

EMPLOYEE:

1. Has only READ access to EMPLOYEE, DEALS, LISTING, USER, USER_ADDRESS tables.
2. Has READ/WRITE/UPDATE access to ORDER_STATUS table.