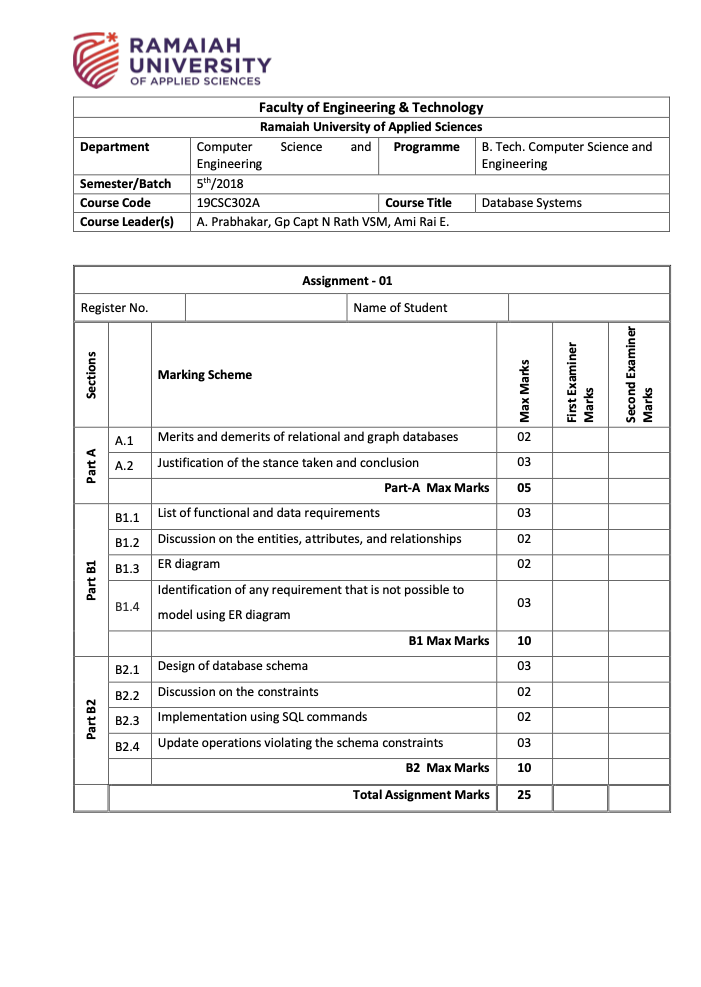


|  |  |
| --- | --- |
| **ASSIGNMENT - 1** | |
| **Course Code** | 19CSC302A |
| **Course Name** | Database Systems |
| **Programme** | B. Tech |
| **Department** | CSE |
| **Faculty** | FET |

#### 

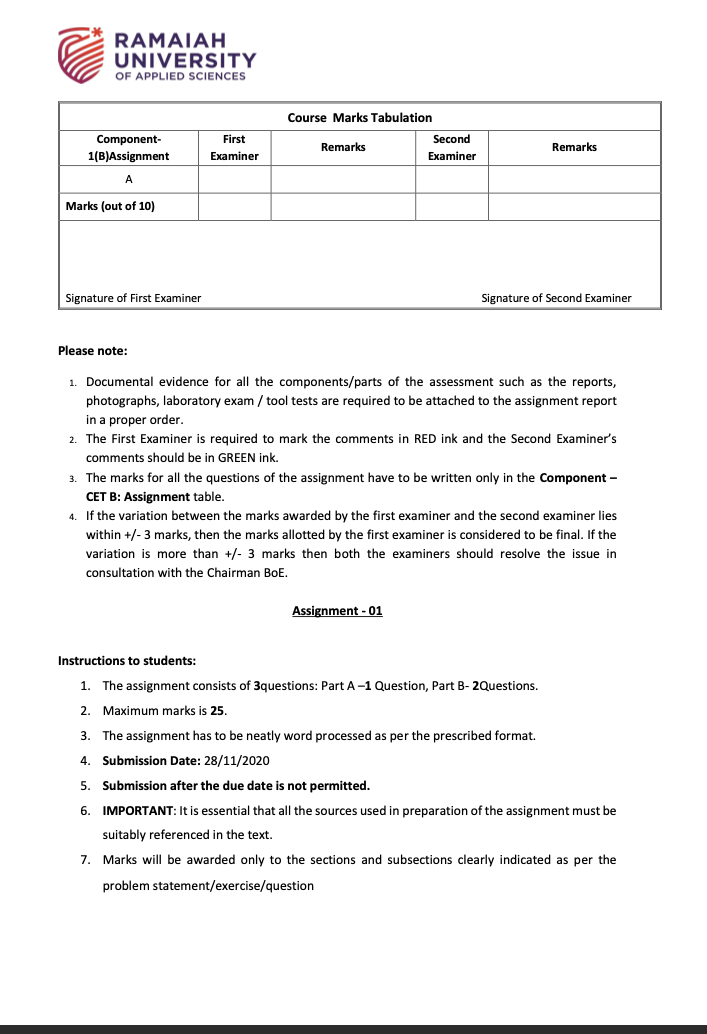
|  |  |
| --- | --- |
| **Name of the Student** | K Srikanth |
| **Reg. No** | 17ETCS002124 |
| **Semester/Year** | 5th Semester/3rd Year |
| **Course Leader/s** | Ami Rai E. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Declaration Sheet** | | | | | | | | |
| Student Name | K Srikanth | | | | | | | |
| Reg. No | 17ETCS002124 | | | | | | | |
| Programme | B.Tech | | | | | Semester/Year | 5th Semester/ 3rd Year | |
| Course Code | 19CSC302A | | | | | | | |
| Course Title | Database Systems | | | | | | | |
| Course Date | 14/09/2020 | | to | | 16/02/2021 | | | |
| Course Leader | Ami Rai E. | | | | | | | |
| **Declaration**  The assignment submitted herewith is a result of my own investigations and that I have conformed to the guidelines against plagiarism as laid out in the Student Handbook. All sections of the text and results, which have been obtained from other sources, are fully referenced. I understand that cheating and plagiarism constitute a breach of University regulations and will be dealt with accordingly. | | | | | | | | |
| Signature of the Student | |  | | | | | Date |  |
| Submission date stamp  (by Examination & Assessment Section) | |  | | | | | | |
| Signature of the Course Leader and date | | | | Signature of the Reviewer and date | | | | |
|  | | | |  | | | | |



17ETCS002124

K Srikanth



**Question A**

**A1.1)**

**Introduction**

**Relational Database:** A relational database is a collection of data items with pre-defined relationships between them. These items are organized as a set of tables with columns and rows.

**Example:** MySql which uses relational database concept and is written using Structured Query Language

****

Figure 1 Example of a Relational Database

**Graph Database:** A graph database is a database designed to treat the relationships between data as equally important to the data itself. Here all the data is stored in nodes and connected using relations with a pre-defined model.

**Example:** Neo4j which uses Graph Database concept and is written using cypher query language

****

Figure 2 Example of a Graph Database

**Now let’s talk about the Merits and demerits of relational and graph databases**

**Merits of using relational database**

* Easy to use when compared with graph database
* If you have simple model which is consistent then its better to go for relational database
* Data Manipulation is easy to do in relational database when compared with graph database

**Merits of using graph database**

* If the data has lot of relations represented as many to many it is easier to connect when you lot of relations using a graph database
* If you have a large number of data scale like “Facebook”. Graph database is the best way to go because of low latency when you have a large scale of data.
* Graph databases can be used for real time data when the latency and processing time matter as they use tree techniques like Breadth First Search and Depth First Search

**Demerits of using relational database**

* Performance is slow for a large-scale data compared to graph database
* The memory to create tables and rows take a lot of physical memory to store the data

**Demerits of using graph database**

* If there are any unrelated data then graph database is not the right way
* If you are looking for a fixed data set like a train reservation system where you have fixed number of columns then graph database is not the right way

**A1.2)**

To start off with both the databases have their own significance depending on the data the user has but using a relational database is easy when compared with graph database I feel like graph databases are somehow going to replace relational database at any point just like **Python over C language** because of its latency and efficiency and handling a large data set is easier when compared with a relational database so I can conclude saying that both of the databases have their own significance in their way it all depends on the data.

Graph Database was really fun to try (Figure 2)

**Question B**

**Question B1)**

**B1.1) Functional Requirements**

**Table 1.1: Functional Requirement 1**

|  |  |
| --- | --- |
| Requirement Tag | FR1 |
| Requirement Description | The system should have an interface where user can be able to create an account |
| Dependent on Requirements | - |
| User/System interacting with the requirement | User |

**Table 1.2: Functional Requirement 2**

|  |  |
| --- | --- |
| Requirement Tag | FR2 |
| Requirement Description | The system should have an interface where user can be able to login using existed or newly created account |
| Dependent on Requirements | FR1 |
| User/System interacting with the requirement | User |

**Table 1.3: Functional Requirement 3**

|  |  |
| --- | --- |
| Requirement Tag | FR3 |
| Requirement Description | The system should be able to store user data. |
| Dependent on Requirements | FR1 |
| User/System interacting with the requirement | System |

**Table 1.4: Functional Requirement 4**

|  |  |
| --- | --- |
| Requirement Tag | FR4 |
| Requirement Description | The system should have an interface where user can see all the products |
| Dependent on Requirements | - |
| User/System interacting with the requirement | User |

**Table 1.5: Functional Requirement 5**

|  |  |
| --- | --- |
| Requirement Tag | FR5 |
| Requirement Description | The System should have a search bar where user can search for a particular product |
| Dependent on Requirements | FR4 |
| User/System interacting with the requirement | User |

**Table 1.6: Functional Requirement 6**

|  |  |
| --- | --- |
| Requirement Tag | FR6 |
| Requirement Description | The System should have an interface where they can view their cart |
| Dependent on Requirements | FR4, FR5 |
| User/System interacting with the requirement | User |

**Table 1.7: Functional Requirement 7**

|  |  |
| --- | --- |
| Requirement Tag | FR7 |
| Requirement Description | The System should have an interface where user can make payments for their order |
| Dependent on Requirements | FR6 |
| User/System interacting with the requirement | User |

**Table 1.8: Functional Requirement 8**

|  |  |
| --- | --- |
| Requirement Tag | FR8 |
| Requirement Description | The System should have an interface where Admin can Add / Update and delete the products |
| Dependent on Requirements | FR4 |
| User/System interacting with the requirement | Staff / Admin |

**Table 1.9: Functional Requirement 9**

|  |  |
| --- | --- |
| Requirement Tag | FR9 |
| Requirement Description | The System should display to user if the product is available or unavailable |
| Dependent on Requirements | FR4.FR5 |
| User/System interacting with the requirement | User / System |

**Data Requirements**

**Table 2.1: Data Requirement 1**

|  |  |
| --- | --- |
| Requirement Tag | DR1 |
| Item Name | User ID |
| Item Description (Where/How used) | This is Unique Number used to identify users |
| Item type | Int |
| User/System interacting with the item | User |

**Table 2.2: Data Requirement 2**

|  |  |
| --- | --- |
| Requirement Tag | DR2 |
| Item Name | Password |
| Item Description (Where/How used) | To Login the user has to enter their password which can consist of numbers, alphabets and special characters |
| Item type | Char Array |
| User/System interacting with the item | User |

**Table 2.3: Data Requirement 3**

|  |  |
| --- | --- |
| Requirement Tag | DR3 |
| Item Name | First Name |
| Item Description (Where/How used) | This data contains First Name of the user |
| Item type | Char Array |
| User/System interacting with the item | User |

**Table 2.4: Data Requirement 4**

|  |  |
| --- | --- |
| Requirement Tag | DR4 |
| Item Name | Last Name |
| Item Description (Where/How used) | This data contains Last Name of the user |
| Item type | Char Array |
| User/System interacting with the item | User |

**Table 2.5: Data Requirement 5**

|  |  |
| --- | --- |
| Requirement Tag | DR5 |
| Item Name | Phone Number |
| Item Description (Where/How used) | This data contains Phone Number of the user |
| Item type | Long Int |
| User/System interacting with the item | User |

**Table 2.6: Data Requirement 6**

|  |  |
| --- | --- |
| Requirement Tag | DR6 |
| Item Name | Address |
| Item Description (Where/How used) | This data contains Address of the user |
| Item type | Char Array |
| User/System interacting with the item | User |

**Table 2.7: Data Requirement 7**

|  |  |
| --- | --- |
| Requirement Tag | DR7 |
| Item Name | Admin ID |
| Item Description (Where/How used) | This is Unique Number used to identify Admin |
| Item type | Int |
| User/System interacting with the item | Admin |

**Table 2.8: Data Requirement 8**

|  |  |
| --- | --- |
| Requirement Tag | DR8 |
| Item Name | Admin Password |
| Item Description (Where/How used) | To Login the Admin has to enter their password which can consist of numbers, alphabets and special characters |
| Item type | Char Array |
| User/System interacting with the item | Admin |

**Table 2.9: Data Requirement 9**

|  |  |
| --- | --- |
| Requirement Tag | DR9 |
| Item Name | Product ID |
| Item Description (Where/How used) | This is Unique Number used to identify Products |
| Item type | Int |
| User/System interacting with the item | Admin / User |

**Table 2.10: Data Requirement 10**

|  |  |
| --- | --- |
| Requirement Tag | DR10 |
| Item Name | Product Name |
| Item Description (Where/How used) | This data contains Product Name |
| Item type | Char Array |
| User/System interacting with the item | Admin / User |

**Table 2.11: Data Requirement 11**

|  |  |
| --- | --- |
| Requirement Tag | DR11 |
| Item Name | Availability Status |
| Item Description (Where/How used) | This data contains if the product is available or out of stock |
| Item type | Boolean |
| User/System interacting with the item | System / User |

**Table 2.8: Data Requirement 12**

|  |  |
| --- | --- |
| Requirement Tag | DR8 |
| Item Name | Order ID |
| Item Description (Where/How used) | This data contains the Order ID which is generated after a successful transaction |
| Item type | Int |
| User/System interacting with the item | System / User |

**B1.2)**

**Entities**

**1. Admin:** This Entity contains all the attributes required for an administrator

**2. User:** This Entity contains all the attributes required for a user

**3. Products:** This Entity contains all the attributes required for a product

**Attributes**

**User Entity**

**1. User ID:** This attribute is used to stores the User ID which is auto incremented and this attribute is also our primary key for this entity.

**2. Password:** This attribute is used to stores the password of the user to check if the user entered password is valid or invalid.

**3. First Name:** This attribute contains the first name of the user.

**4. Last Name:** This attribute contains the last name of the user.

**5. Phone Number:** This attribute contains the phone number of the user.

**6. Address:** This attribute contains the address of the user.

**Admin Entity**

**1. Admin ID:** This attribute is used to stores the Admin ID and this attribute is also our primary key for this entity

**2. Password:** This attribute is used to stores the password of the admin to check if the user entered password is valid or invalid.

**Products Entity**

**1. Product ID:** This attribute is used to stores the Product ID which is auto incremented and this attribute is also our primary key for this entity.

**2. Product Name:** This attribute contains the name of the product.

**3. Manufacturer:** This attribute contains the manufacturer name of the product.

**4. Availability Status:** This attribute contains a flag which tells us if the product is available or out of stock.

**5. Quantity:** This attribute contains the current stock quantity of the product

**6. Description:** This attribute contains the description of the product in a detailed view.

**7. Product Price:** This attribute contains the price of the product

**Relationships**

**1. Add Product:** This relation is 1: N from admin to product where admin can add product.

**2. Update Product: T**his relation is 1: N from admin to product where admin can update product.

**3. Delete Product: T**his relation is 1: N from admin to product where admin can delete product.

**4. Orders:** This relation is M: N from User to Products such that **“M”** number of users can order **“N”** number of Productswith attributes containing **Order ID** and a primary key and Time/ Date as an attribute.

**Note: Here M is not equal to N**

**B1.3)**

**Entity Relation Diagram**

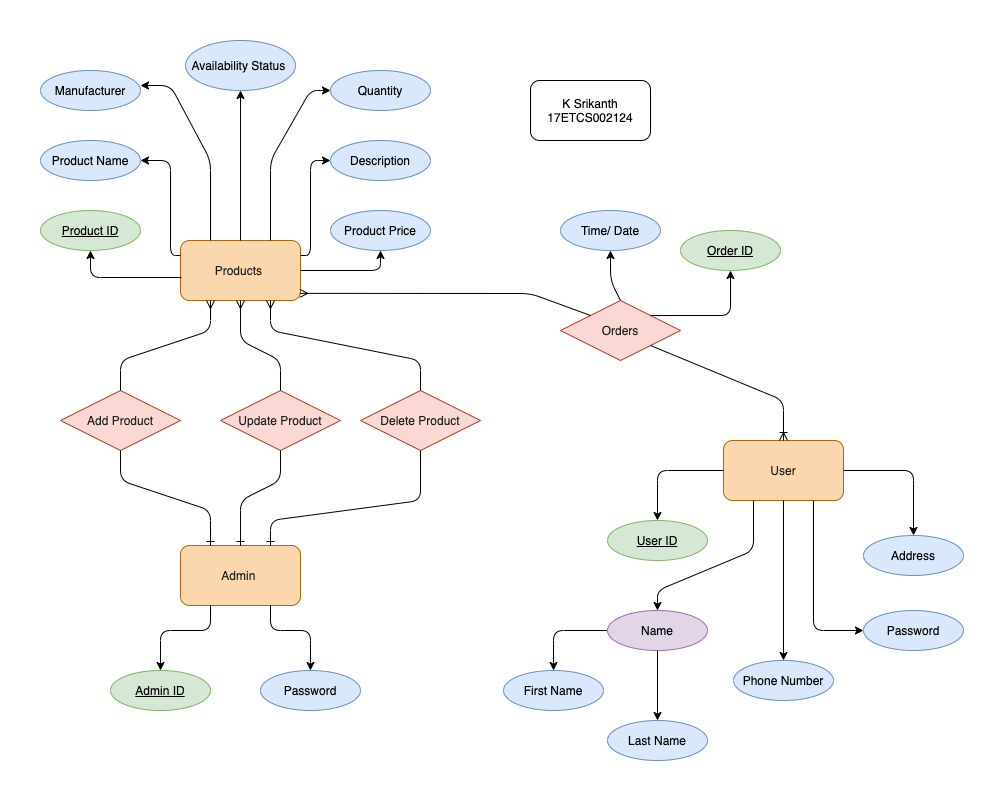
****

Figure 3 Entity Relation Diagram for the given problem statement

**B1.4)**

**Identification of the requirement:**

Entity Relation diagram doesn’t stratify some of the requirements constraints some of them include,

1. Entity relation diagram cannot show us the response we get after the input

2. Verification process can’t be shown using entity relation diagram

3. Entity relation diagram doesn’t let us know the sequence of which the user has to follow

**Alternative Suggestion**

There are many UML conceptual data models but I choose to go with Sequence Diagram

**Sequence Diagram**

Sequence diagrams are the better way of doing UML because they focus on lifelines, or the processes and objects that live simultaneously, and the messages exchanged between them to perform a function before the lifeline end

Sequential Diagrams are used to represent time sequence of the objects like what order is the user is going to interact with the system,

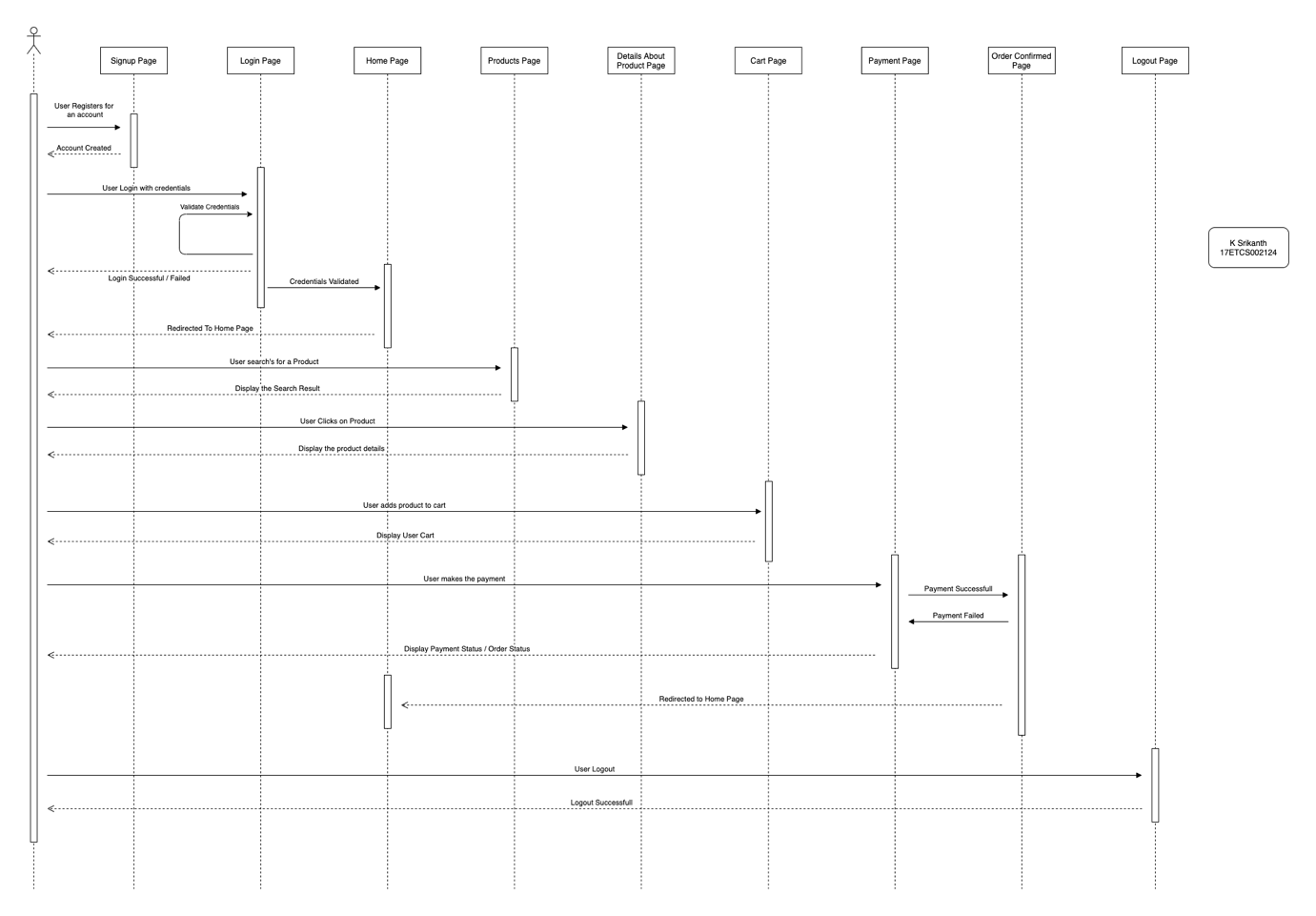
**Sequence Diagram (User)**

Figure 4 Sequence Diagram of the given problem statement for User Access

There are two types of users in our case here

**1. Customer / User**

Let’s see how a customer interacts with our system (Furniture Website),

Firstly, the user has to register using the interface provide by the system which includes filling up all the necessary details to get started. After registration now user can be able to sign in using the interface provide by our system and system is going to verify if the user’s credentials are valid or not and it will display a prompt to the user with a dialog box stating that the user has logged in or failed to login. If the user has failed to login then the user can again enter their credentials on login page. If dialog box says that successfully logged in then the user will be redirected to the homepage of furniture shopping website where the user can search for their preferred furniture product and system will display the results. Now when user clicks on a product it will be able to displays fully detailed information about the product and user can know about the product much better if they are not aware of the product that they are looking at and user can add the product to the cart section of the website after knowing the details about the product now user have an option where they can be able to buy or using a payment gateway if the payment is successful then user will receive an order ID followed by the list of products they have purchased after this they would be redirected to the home page of Furniture Home Page and if the payment is failed then user will have to try the payment again. Finally, if the user is done with their shopping, they can logout of the system.

**Sequence Diagram (Admin)**

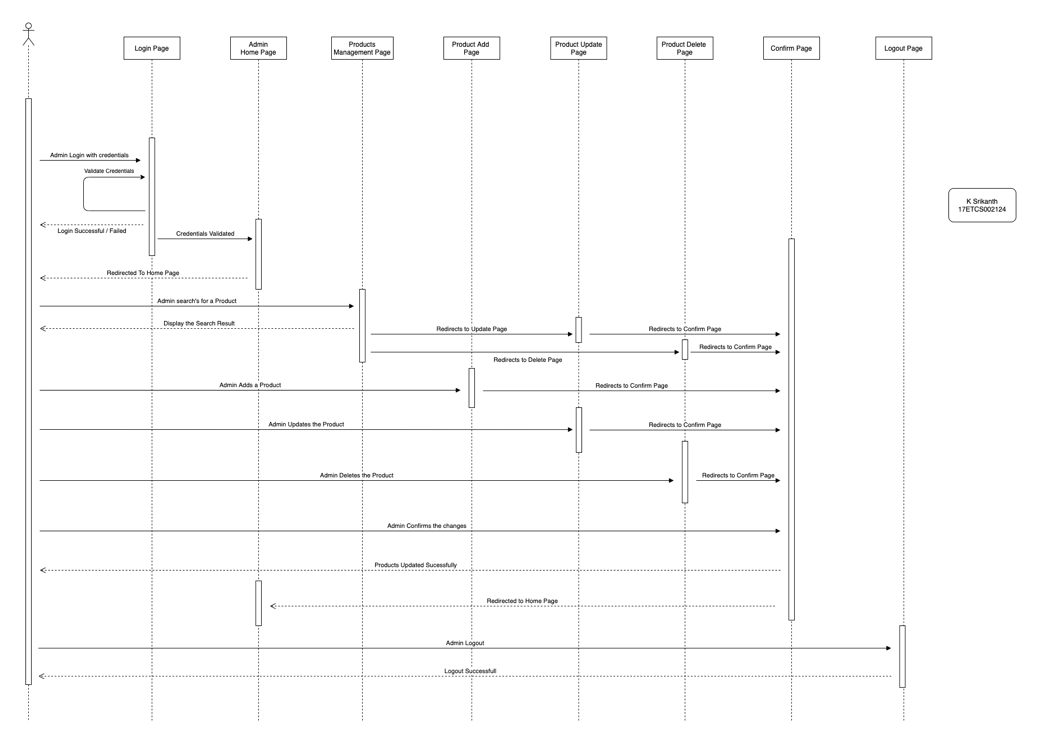


Figure 5 Sequence Diagram of the given problem statement for Administrator Access

The staff administration are going to provide the credentials for admin account where they can make changes to the websites products stock now the admin has to sign in using the interface provide by the system and the system is going to verify if the admin’s credentials are valid or not and it will display a prompt to a user with a dialog box stating that the user has logged in or failed to login if the user has failed to login then the user can again enter their credentials on login page. If dialog box says that successfully logged in then It will be redirected to the homepage of our admin where the admin where admin can search for their furniture product and system will display the results. The four major operations that admin can perform are

**1. Add**

If admin wants to add a new product for their website, they can do it via creating a new product and uploading the particular details about the product.

**2. Update**

If admin wants to update details of the product which is up online, they can be able to do it with an edit option on the product page.

**3. Delete**

If admin wants to delete the product which is up online, they can be able to do it with a delete option on the product page.

After Performing all the operations, the admin can verify everything like what changes that they made all together and make it online. Finally, if the admin is done making changes with their website, they can logout of the system.

**Question B2)**

**B2.1)**

**To convert entity relation diagram into a relational schema there are 7 steps to be followed,**

**Step – 1)**

**Convert all the strong entities into tables**

In Step 1 we just convert strong entities in to tables

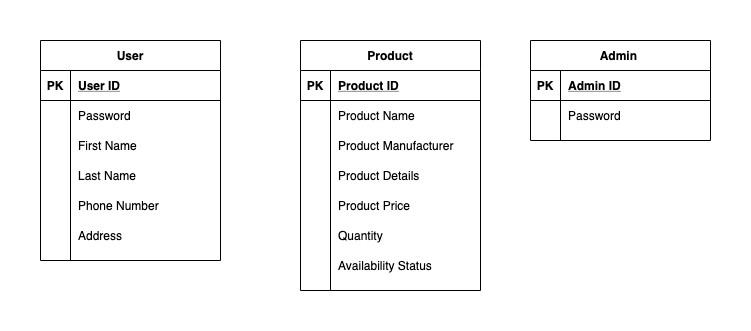
****

Figure 6 Converting Strong Entities into table from Image 1

**Step – 2)**

**Convert all the weak entities into tables**

In Step 2 we convert all the weak entities into a table in my case I don’t have any weak entities from my entity relation diagram (Image 1)

**Step – 3)**

**Convert all the 1:1 relations into tables**

In Step 3 we would be converting all the 1:1 relations into table in my case I don’t have any 1:1 relations from my entity relation diagram (Image 1)

**Step – 4)**

**Convert all the 1:N relations into tables**

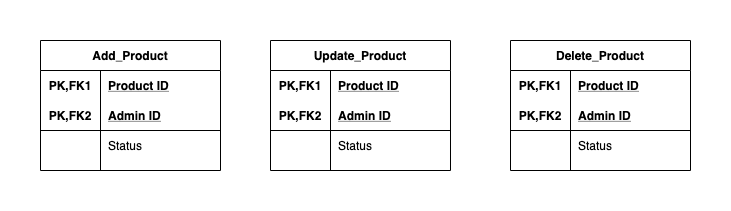
****

Figure 7 Converting 1:N Relation into table from Image 1

In Step 4 we would be converting all the 1:N relations into table which basically means that adding foreign keys if your relation supports. In my case one admin can add a product or update a product or delete product .

**Step – 5)**

**Convert all the M:N relations into tables**

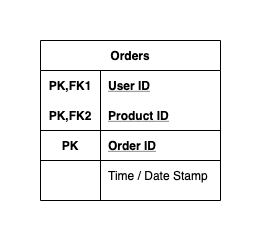
****

Figure 8 Converting M:N Relations into table from image 1

In Step 5 we would be converting all the M:N relations into table which has a separate table and have both the primary keys from the entities and also a attributes (if it has a attributes) in my case “M” number of customers can order “N” number of products.

**Note: Here M is not equal to N**

**Step – 6)**

**Convert all the Multivalued attributes into tables**

In Step 6 we should convert all the multivalued attributes into a separate table my case I don’t have any Multivalued attributes from my entity relation diagram (Image 1)

**Step – 7)**

**Convert all the N-Ary relationships into tables**

In Step 7 we should convert all the N-Ary relationships into a table my case I don’t have any N-Ary relationships from my entity relation diagram (Image 1)

**Relational Schema**

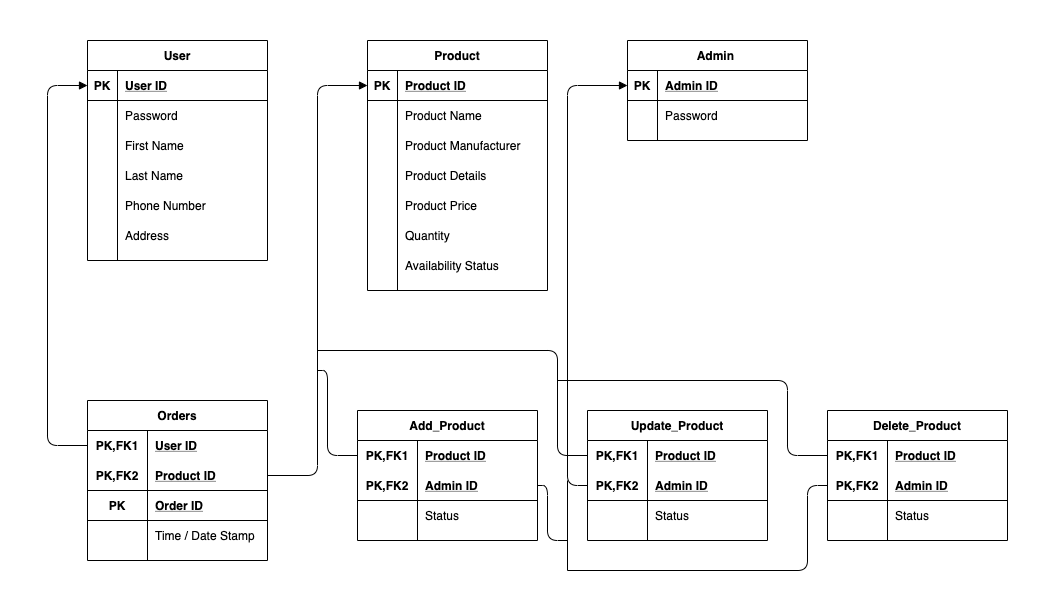
****

Figure 9 Complete Relational Schema to ER Diagram(Image 1)

**B2.2)**

**User Table**

**1. User ID:** PRIMARY KEY AUTO\_INCREMENT <Type> INT

**2. Password:** NOT NULL <Type> VARCHAR

**3. First Name:** NOT NULL <Type> VARCHAR

**4. Last Name:** <Type> VARCHAR

**5. Phone Number:** NOT NULL UNIQUE <Type> LONG INT

**6. Address:** <Type> VARCHAR

**Admin Table**

**1. Admin ID:** PRIMARY KEY <Type> INT

**2. Password:** NOT NULL <Type> VARCHAR

**Products Table**

**1. Product ID:** PRIMARY KEY AUTO\_INCREMENT <Type> INT

**2. Product Name:** NOT NULL <Type> VARCHAR

**3. Manufacturer:** NOT NULL <Type> VARCHAR

**4. Availability Status:** NOT NULL <Type> BOOLEAN

**5. Quantity:** NOT NULL <Type> INT

**6. Description:** NOT NULL <Type> VARCHAR

**7. Product Price:** NOT NULL <Type> DOUBLE

**Orders Table**

**1. User ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**2. Product ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**3. Order ID:** PRIMARY KEY AUTO\_INCREMENT <Type> INT

**4. Date Stamp:** NOT NULL <Type> DATE STAMP

**Add\_Product Table**

1. **Product ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**2. Admin ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**3. Status:** NOT NULL <TYPE> BOOLEAN

**Update\_Product Table**

1. **Product ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**2. Admin ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**3. Status:** NOT NULL <TYPE> BOOLEAN

**Delete\_Product Table**

1. **Product ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**2. Admin ID:** PRIMARY KEY & FOREIGN KEY <Type> INT

**3. Status:** NOT NULL <TYPE> BOOLEAN

**B2.3)**

**MySQL Implementation for Question B2.2**

**1. User Table**

**MySQL Query**

create table user (

User\_ID int auto\_increment,

Password varchar (40) not null,

First\_Name varchar (40) not null,

Last\_Name varchar (40),

Phone\_Number bigint unique not null,

Address varchar (40),

primary key (User\_ID));

**Result**

**Describe Table**

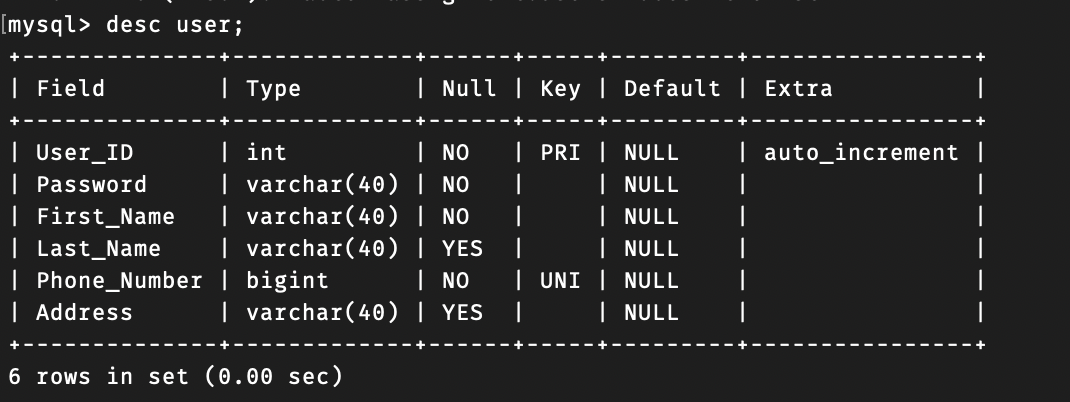
****

Figure 10 MySQL Metadata for user table.

**Data in the table**

**MySQL Query**

insert into user(Password,FIRST\_Name,Last\_Name,Phone\_Number,Address) values ("\*\*\*\*\*\*","K","Srikanth",9493364308,"Nagwara,Bangalore");

insert into user(Password,FIRST\_Name,Last\_Name,Phone\_Number,Address) values ("\*\*\*\*\*\*","GN","Naveen Kumar",7019462108,"Bangalore");

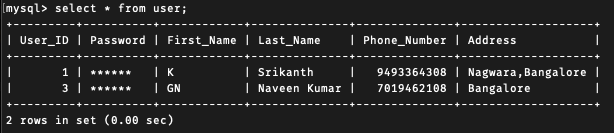


Figure 11 MySQL data from user table

**2. Admin Table**

**MySQL Query**

create table admin(

Admin\_ID int,

Password varchar (40) not null,

primary key (Admin\_ID));

**Result**

**Describe Table**

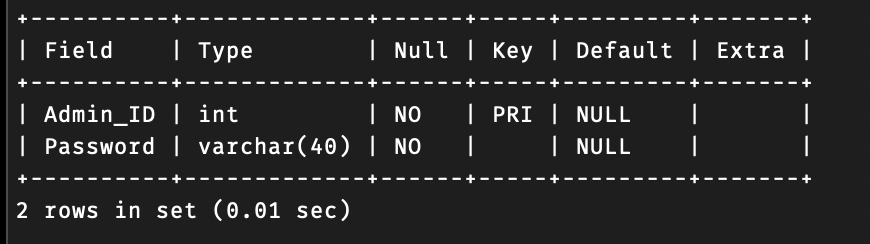
****

Figure 12 MySQL Metadata for Admin table.

**Data in the table**

**MySQL Query**

insert into admin values ("17002124","\*\*\*\*\*\*\*");

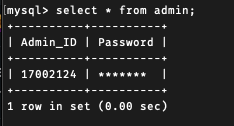
****

Figure 13 MySQL data from Admin table

**3. Products Table**

**MySQL Query**

create table product21 (

Product\_ID int auto\_increment,

Product\_Name varchar (40) not null,

Manufacturer varchar (40) not null,

Availability\_Status Bool not null,

\_Description varchar (40) not null,

Quantity int not null,

Product\_price Double not null,

primary key (Product\_ID));

**Result**

**Describe Table**

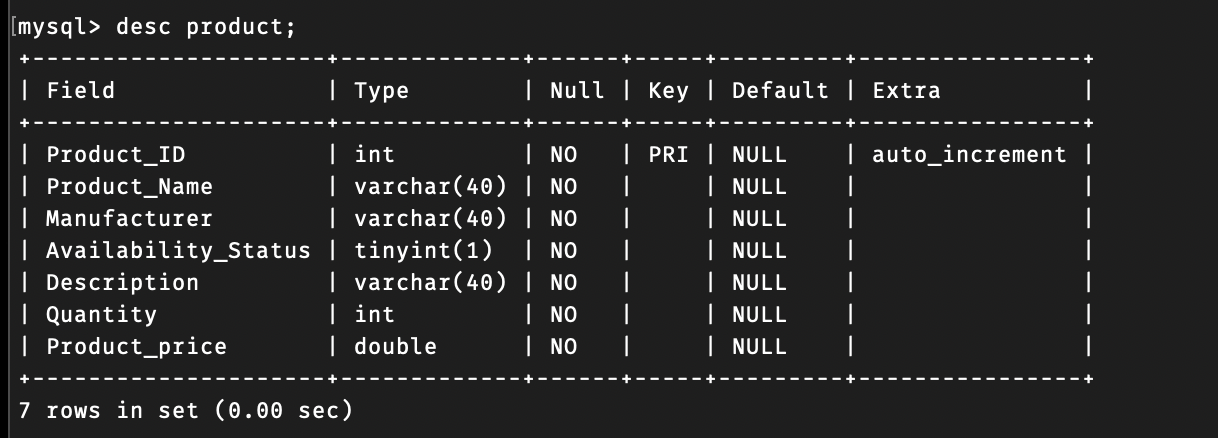
****

Figure 14 MySQL Metadata for Product table.

**Data in the table**

**MySQL Query**

insert into product(Product\_Name,Manufacturer,Availability\_Status,Description,Quantity,Product\_price) VALUES ("Oak Desk","IKEA",true,"Good Table",10,4000.00);

insert into product(Product\_Name,Manufacturer,Availability\_Status,Description,Quantity,Product\_price) VALUES ("Maple Desk","IKEA",true,"Also a Good Table",5,10000.00);



Figure 15 MySQL data from Product table

**Note:** Here Availability\_Status is a Boolean data type If it is **1 then it is true** **(Available)** or if it is **0 then it is false (Out of stock),**

**4. Orders Table**

**MySQL Query**

create table orders (

Order\_ID int auto\_increment,

User\_ID int,

Product\_ID int,

Date\_Time TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

**FOREIGN KEY** (User\_ID) **REFERENCES** user (User\_ID),

**FOREIGN KEY** (Product\_ID) **REFERENCES** product (Product\_ID),

primary key (Order\_ID));

**Result**

**Describe Table**

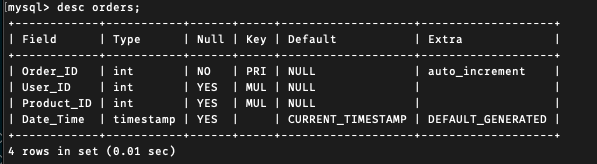
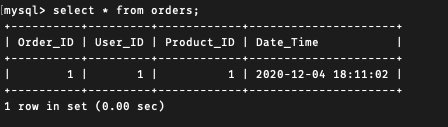


Figure 16 MySQL Metadata for Order table.

**Data in the table**

**MySQL Query**

insert into orders(user\_ID,Product\_ID) values (1,1);

****

*Figure 17 MySQL data from Order table*

**5. Add\_Product Table**

MySQL Query

create table add\_product (

Product\_ID int,

Admin\_ID int,

Status bool not null,

**FOREIGN KEY** (Product\_ID) **REFERENCES** product (Product\_ID),

**FOREIGN KEY** (Admin\_ID) **REFERENCES** admin (Admin\_ID));

**Result**

**Describe Table**

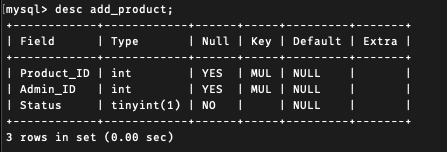


Figure 18 MySQL Metadata for add\_product table.

**Data in the table**

**MySQL Query**

insert into add\_product (Product\_ID,Admin\_ID,Status) values (1,17002124,true);

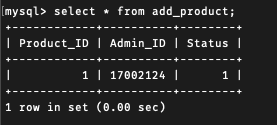
****

Figure 19 MySQL data from add\_product table

**6. Update\_Product Table**

MySQL Query

create table update\_product (

Product\_ID int,

Admin\_ID int,

Status bool not null,

**FOREIGN KEY** (Product\_ID) **REFERENCES** product (Product\_ID),

**FOREIGN KEY** (Admin\_ID) **REFERENCES** admin (Admin\_ID));

**Result**

**Describe Table**

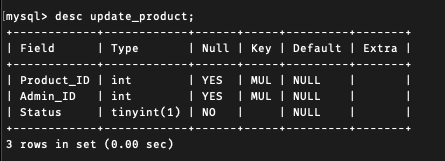


Figure 20 MySQL Metadata for update\_product table.

**Data in the table**

**MySQL Query**

insert into update\_product(Product\_ID,Admin\_ID,Status) values (1,17002124,true);

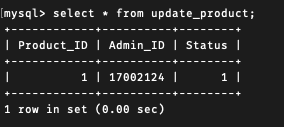


Figure 21 MySQL data from update\_product table

**7. Delete\_Product Table**

MySQL Query

create table delete\_product (

Product\_ID int,

Admin\_ID int,

Status bool not null,

**FOREIGN KEY** (Product\_ID) **REFERENCES** product (Product\_ID),

**FOREIGN KEY** (Admin\_ID) **REFERENCES** admin (Admin\_ID));

**Result**

**Describe Table**

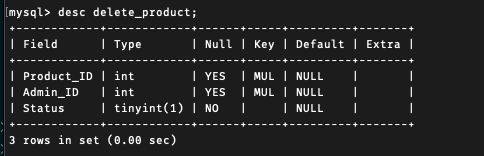


Figure 22 MySQL Metadata for delete\_product table.

**Data in the table**

**MySQL Query**

insert into delete\_product(Product\_ID,Admin\_ID,Status) values (1,17002124,true);

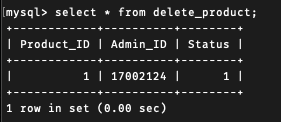


Figure 23 MySQL data from delete\_product table

**B2.4)**

**Schema Violating Constraints**

**1. Not Null Constraint**

**MySQL Query**

update add\_product set status = Null where status = true;

**Output**

****

Figure 24 MySQL Constraint Error for Not Null

The violation we see is that status is a not null Boolean value when you try to update it cannot be null it violates not null constraint.

**2. Domain Constraint**

**MySQL Query**

update user set Phone\_Number = "GN "where Phone\_Number = 7019462108;

**Output**

****

Figure 25 MySQL Constraint Error for domain constraint

The violation we see is that phone number is a integer value and it cannot accept a string or a character array it violates domain constraint

**3. Primary Key**

**MySQL Query**

update orders set order\_id= null where order\_id = 1;

**Output**

****

Figure 26 MySQL Constraint Error for Entity integrity constraint

The violation we see is that primary key cannot be null when you try update it violates Entity integrity constraint.

**4. Foreign Key**

**MySQL Query**

update orders set user\_id = 7 where user\_id = 1;

**Output**

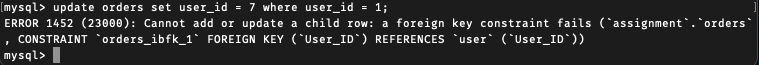
****

Figure 27 MySQL Constraint Error for referral integrity constraint

The violation we see is that if the foreign key doesn’t exist from user table it cannot map to that attribute and it violates Referential Integrity Constraint.