DATABASE SYSTEMS (CS F212)
SEMESTER PROJECT

DBS_PR_07:
Laundry Management System

Submitted By:

Vansh Chhabra 2019B1A71039P

Sujay Patni 2019B3A70575P

Project Coordinators:

Mr. Pravin Kumar Saha (TA), Ms. Trishna Paul (Ph.D.)

INTRODUCTION:

Project Description: Here the admin of the store will keep records of all the users , and simultaneously will update the status of the clothes as per availability which will scale the business of laundry for the owner.

→ In this project, we have tried to implement the Laundromat service provided in our BITS Pilani Campus. We have implemented many entities and relationships to model the working of Laundromat in our campus. Further information can be found in the Software Requirements Specification (SRS) section.

Database aspects it will cover: ER modeling, Data Normalization, Functions/Procedures, Views, Insert, Update, and Delete Queries, Concurrency

SOFTWARE REQUIREMENTS SPECIFICATION (SRS):

The basic entities that we created to model the working of Laundromat are as follows: Branch, Order, Rate, Customer, Account, Login, and Monthly Revenue. The corresponding relationships are: BranchOrder, CustomerAccount, CustomerLogin, OrderRate, BranchRevenue.
These entities and relationships are further made more meaningful by removing redundancies and by performing normalization.

- **Branch:** This represents a specific branch of Laundromat in our campus. We have considered the current 3 branches located at: CVR Bhawan, Malviya Bhawan, and Meera Bhawan.
- **Order:** This represents the order placed by a customer which has to be fulfilled by the laundry service. It also shows the status of the order. The order can be either “delivered” or “pending”.
- **Rate:** This represents the different rates/costs associated with the orders placed by customers. We have considered 3 possible types of orders having different rates: Clothes (normal laundry clothes), Blankets, and Suits.
- **Customer:** The Customer entity represents a customer of the laundry service who is a BITS Pilani student having a unique BITS ID no. in our case.
- **Account:** This entity provides information to the customer like the date he/she opened the account, and the amount they owe to the laundry service.
- **Login:** Every customer that registers is provided with unique login credentials and this entity takes care of that.
- **Monthly Revenue:** The admin can view the revenue earned by laundry service in different months using this particular entity.

There are 2 user levels in our Laundry Management System:

a. Admin (Has access to the entire system)
b. Customer (Can register, login, and place orders)

Constraints and Conditions to run the model:

- MySQL should be installed on the respective system and preferably MySQL Workbench should be used for running the SQL code.
- All the SQL code has been provided in a single file and running that file will be sufficient.
- The explanations for the defined relationships are provided in the "Relationship Explanations" section.
- The Key Constraints and Functional Dependencies (FDs) are provided in Schema Design and Data Normalization sections respectively.

Functional Requirements:

FR1: The customer should be able to register with their details.
FR2: The system should record the following customer details: BITS ID, Name, Hostel, Room No., Phone No.
FR3: The system should verify the customer username & password against the member database while logging in.
FR4: The customer should be able to place an order and the admin will receive OrderID, Laundry Weight and the customer will be shown "Pending" status for his/her order.
FR5: Once the customer places an order, depending on the rate of the order, the amount owed by the customer as well as the monthly revenue of the particular branch should change.

Data Requirements:

DR1: Customer
BITS ID: One of the following forms:
20**XX**B**X**PS**XXXX**P/20**XX**B**X**A**XXXXX**P/20**XX**A**X**PS**XXXX**P
Name: FirstName LastName
Hostel: One of {'KR','GD','VK','RM','BD','MR','SH','VY','AK','RP','BH','ML'}
Room No.: Valid positive integer
Phone No.: 10 digits long

DR2: Account

AccountID: Of the form “ACC**X**”
Owe: Valid positive integer
DateOpened: Of the form YYYY-MM-DD

DR3: Login
LoginID: Of the form “ID**X**”
Pwd: Set as “ PASS**X**” in our data. Can be chosen as anything by the user.
Username: FirstNameLastName

DR4: Order
OrderID: Of the form “ORD**X**”
Laundry Weight: A valid positive decimal number. (Measured in kilograms)
Status: “Pending” or “Delivered”

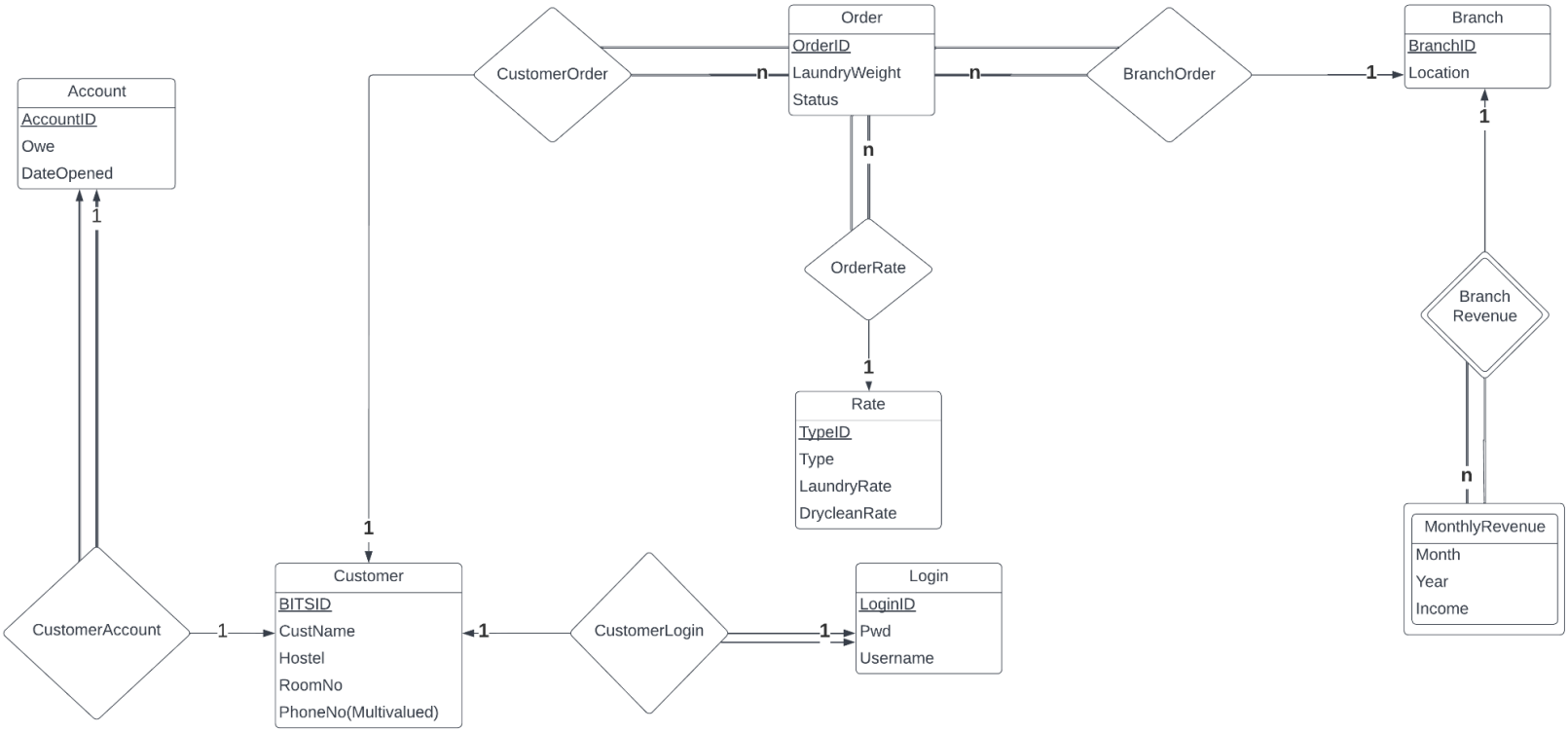
DR5: Rate
TypeID: One of {‘CLOTHES123’, ‘BLANKET123’, ‘SUIT123’}
Type: One of {‘CLOTHES’, ‘BLANKET’, ‘SUIT’}
Laundry Rate: A valid positive integer
Dryclean Rate: A valid positive integer

DR6 Branch:
BranchID: One of {‘CVR123’, ‘MB123’, ‘MEERA123’}
Location: One of {‘CVR BHAWAN’, ‘MALVIYA BHAWAN’, ‘MEERA BHAWAN’}

DR7 Monthly Revenue:
Month: One of {'JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OCT','NOV','DEC'}
Year: A valid year integer
Income: A valid positive integer

SYSTEM MODELING

A. Entity-Relationship(ER) Diagram



<u>Relationship Explanations:</u>

CustomerAccount:

- A customer can have exactly one account.
- An account can be associated with exactly one customer.
- It is possible that a customer does not have an account.

CustomerOrder:

- A customer can have multiple (N) orders.
- An order can be associated with exactly one customer.
- It is possible that a customer does not have an order.

CustomerLogin

- A customer can have exactly one login credential.
- A login credential can be associated with exactly one customer.
- It is possible that a customer does not have a login credential.

OrderRate:

- A rate can have/be associated with multiple (N) orders.
- An order can have/be associated with exactly one rate.
- It is possible that a rate is not associated with an order.

BranchOrder:

- A branch can have multiple (N) orders.
- An order can be associated with exactly one branch.
- It is possible that a branch does not have an order.

BranchRevenue:

- A branch can have multiple (N) monthly revenues. (Ex. for January, for February, etc.).
- A monthly revenue can be associated with exactly one branch.
- It is possible that a branch does not have a monthly revenue.

B. Schema Design

1. Branch(BranchID, Location)
2. MonthlyRevenue(BranchID, Month, Year, Income)
3. Account(AccountID, Owe, DateOpened)
4. Customer(BITSID, CustName, Hostel, RoomNo, PhoneNo):
    a. Customer(BITSID, CustName, Hostel, RoomNo)
    b. Customer(BITSID, PhoneNo)
5. Login(LoginID, Pwd, Username)
6. Order(OrderID, LaundryWeight, Status)
7. Rate(TypeID, Type, LaundryRate, DrycleanRate)
8. BranchRevenue(BranchID, Month, Year)
9. CustomerAccount(BITSID, AccountID)
10. CustomerLogin(BITSID, LoginID)
11. OrderRate(OrderID, TypeID)
12. BranchOrder(OrderID, BranchID)
13. CustomerOrder(OrderID, BITSID)

Removing Redundancies:

1. Branch(BranchID, Location)

2. MonthlyRevenue(BranchID, Month, Year, Income)
3. Account_Customer(AccountID, BITSID, Owe, DateOpened)
4. Customer(BITSID, CustName, Hostel, RoomNo, PhoneNo):
    a. Customer(BITSID, CustName, Hostel, RoomNo)
    b. Customer_PhoneNo(BITSID, PhoneNo)
5. Login_Customer(LoginID,BITSID, Pwd, Username)
6. Order_Rate_Branch_Customer(OrderID, BITSID, TypeID, BranchID, LaundryWeight, Status)
7. Rate(TypeID, Type, LaundryRate, DrycleanRate)
8. ~~BranchRevenue(BranchID, Month, Year)~~
9. ~~CustomerAccount(BITSID, AccountID)~~
10. ~~CustomerLogin(BITSID, LoginID)~~
11. ~~OrderRate(OrderID, TypeID)~~
12. ~~BranchOrder(OrderID, BranchID)~~
13. ~~CustomerOrder(OrderID, BITSID)~~

C. Data Normalization

a. Branch(BranchID, Location):

Primary Key: BranchID

Functional Dependencies: BranchID→Location

i. Conversion into 1NF

All attributes are atomic so it is already in 1NF.

ii. 1NF to 2NF

No Partial Dependency(single attribute PK) so the table is in 2NF.

iii. 2NF to 3NF

No Transitive Dependency so the table is in 3NF

b. MonthlyRevenue(BranchID, Month, Year, Income):

Candidate Keys: BranchID+Month+Year

Functional Dependencies:

BranchID+Month+Year→Location

i. Conversion into 1NF

All attributes are atomic so it is already in 1NF.

ii. 1NF to 2NF

No Partial Dependency so the table is in 2NF.

iii. 2NF to 3NF

No Transitive Dependency so the table is in 3NF

c. Account_Customer(AccountID, BITSID, Owe, DateOpened):

Candidate Keys: AccountID, BITSID

Functional Dependencies:
AccountID→BITSID,Owe,DateOpened
BITSID→AccountID,Owe,DateOpened

i. **Conversion into 1NF**
All attributes are atomic so it is already in 1NF.

ii. **1NF to 2NF**
No Partial Dependency(single attribute PK) so the table is in 2NF.

iii. **2NF to 3NF**
No Transitive Dependency so the table is in 3NF

d. Customer(<u>BITSID</u>, CustName, Hostel, RoomNo):

Candidate Keys: BITSID
Functional Dependencies:
BITSID→CustName,Hostel,RoomNo

i. **Conversion into 1NF**
All attributes are atomic so it is already in 1NF.

ii. **1NF to 2NF**
Primary Key: BITSID
No Partial Dependency(single attribute PK) so the table is in 2NF.

iii. **2NF to 3NF**
No Transitive Dependency so the table is in 3NF

e. Customer_PhoneNo(<u>BITSID</u>, <u>PhoneNo</u>):

Candidate Keys: BITSID+PhoneNo
Functional Dependencies: None

i. **Conversion into 1NF**
All attributes are atomic so it is already in 1NF.

ii. **1NF to 2NF**
Primary Key: BITSID + PhoneNo
No Partial Dependency so the table is in 2NF.

iii. **2NF to 3NF**
No Transitive Dependency so the table is in 3NF

f. Login_Customer(<u>LoginID</u>, BITSID, Pwd, Username):

Candidate Keys: LoginID, BITSID
Functional Dependencies:
LoginID→BITSID,Pwd,Username
BITSID→LoginID,Pwd,Username

i. **Conversion into 1NF**
All attributes are atomic so it is already in 1NF.

ii. **1NF to 2NF**
Candidate Keys: LoginID, BITSID
No Partial Dependency(single attribute PK) so the table is in 2NF.

iii. **2NF to 3NF**
No Transitive Dependency so the table is in 3NF

g. **Order_Rate_Branch_Customer(<u>OrderID</u>, BITSID, TypeID, BranchID, LaundryWeight, Status):**
Candidate Keys: OrderID
Functional Dependencies:
OrderID→BITSID,TypeID,BranchIDmLaundryWeught,Status

i. **Conversion into 1NF**
All attributes are atomic so it is already in 1NF.

ii. **1NF to 2NF**
Primary Key: OrderID
No Partial Dependency(single attribute PK) so the table is in 2NF.

iii. **2NF to 3NF**
No Transitive Dependency so the table is in 3NF

h. **Rate(<u>TypeID</u>, Type, LaundryRate, DrycleanRate):**
Candidate Keys: TypeID, Type
Functional Dependencies:
TypeID→Type,LaundryRate,DrycleanRate
Type→TypeID,LaundryRate,DrycleanRate

i. **Conversion into 1NF**
All attributes are atomic so it is already in 1NF.

ii. **1NF to 2NF**
Primary Key: TypeID
No Partial Dependency(single attribute PK) so the table is in 2NF.

iii. **2NF to 3NF**
No Transitive Dependency so the table is in 3NF

D. List of Tables Required

a. Branch(<u>BranchID</u>, Location)
b. MonthlyRevenue(<u>BranchID, Month, Year</u>, Income)
c. Account_Customer(<u>AccountID</u>,BITSID,Owe,DateOpened)
d. Customer(<u>BITSID</u>, CustName, Hostel, RoomNo)
e. Customer_PhoneNo(<u>BITSID, PhoneNo</u>)

f. Login_Customer(LoginID,BITSID,Pwd,Username)
g. Order_Rate_Branch_Customer(OrderID, BITSID, TypeID, BranchID, LaundryWeight, Status)
h. Rate(TypeID,Type,LaundryRate,DrycleanRate)

E. Additional Components:

a. **Procedure To Register Customer:**
REG_CUST(BITSID,CUSTNAME,HOSTEL,ROOMNO,AccountID,PHONENO ,LOGINID,PWD ,USERNAME)
b. **Procedure to Make New Order:**
NEW_ORDER(ORDERID,BITSID,TYPEID,BRANCHID, LAUNDRYWEIGHT,MONTH,YEAR)
c. **Procedure To Complete Delivery:**
Complete_Delivery(OrderID)
d. **Procedure To Add Month in MonthlyRevenue:**
UPDATE_MONTH(BRANCHID,MONTH,YEAR)