Library Management System

An Oracle SQL & PL/SQL Implementation

-Sanket Mali (E265)
-Pinak Salunke(E247)
-Abhijit Borse(E251)

Introduction

- It's about design and implement a robust and scalable database system for managing all core library operations, from user and media management to the rental and return processes.
- To demonstrate the practical application of relational database principles, SQL, and procedural programming with PL/SQL, showcasing how these technologies can be used to solve real-world data management challenges.

Core Features

1)User and Media Management

- **User Management:** This covers everything related to people in the system. When a new person joins, they're registered as a **Customer** or an **Employee**. A **Card** is automatically created for them using a trigger. The system also handles updates to their personal information like their address or phone number.
- **Media Management:** This is about keeping track of the library's collection. The system allows an employee to easily **add new books and videos** to the database, ensuring all relevant details like ISBN, title, and location are recorded. If an item is lost or damaged, it can also be removed from the catalog.

Core Features

2)Item Rental and Return Processes

- **Rental Process:** When a user wants to borrow an item, the system first checks their card status. If their card is **active** and the item's availability is marked as **'A' (Available)**, the transaction is approved. A new record is inserted into the **Rent** table, and the item's availability is updated to **'O' (On Loan)**.

- **Return Process:** When an item is returned, an employee uses a procedure to remove the record from the **Rent** table. The system then automatically changes the item's availability back to **'A' (Available)**, making it ready to be borrowed by someone else.

Core Features

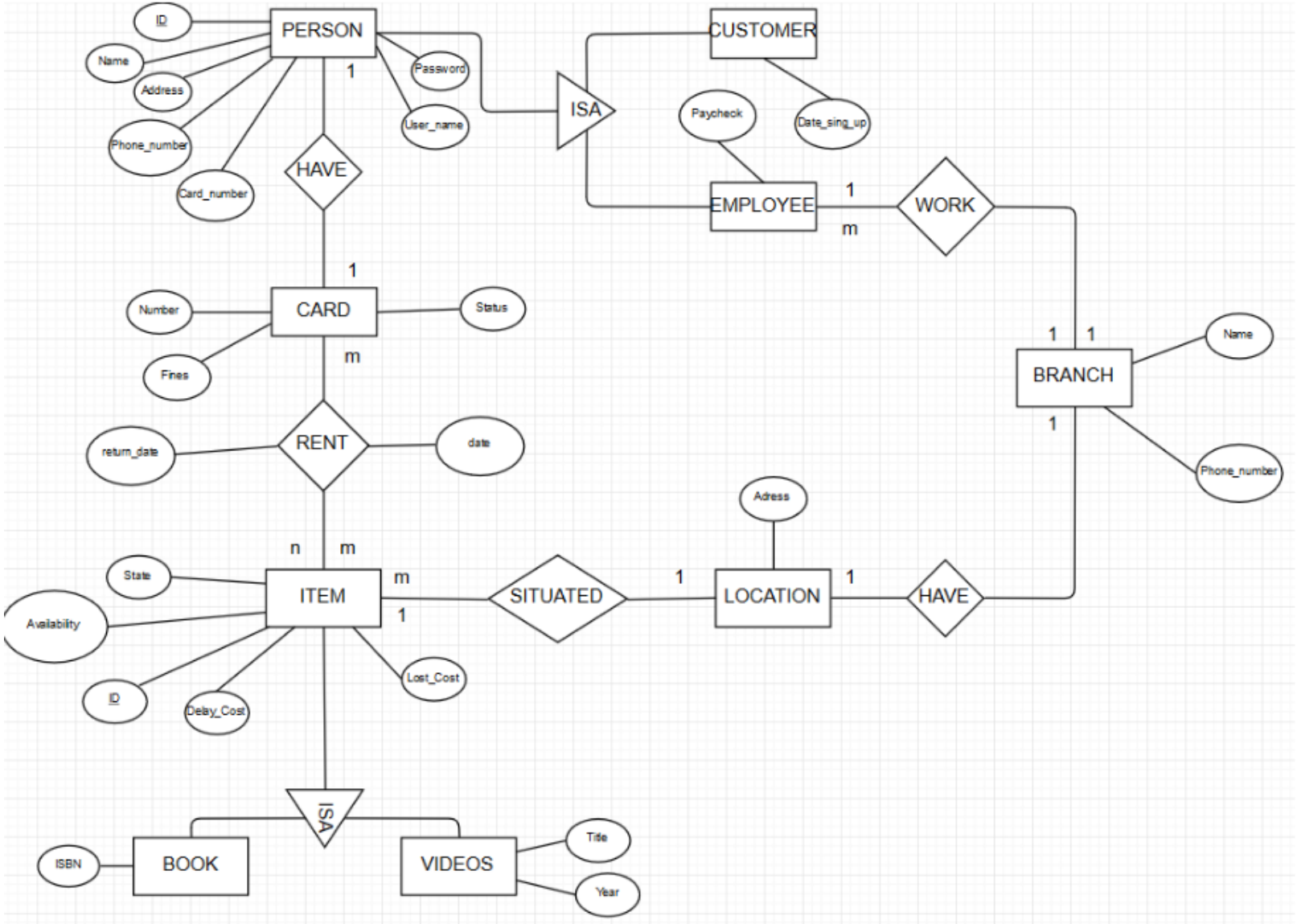
3)Automated Fine Calculation

Fines on Late Returns: This is handled by a trigger that activates when an item is returned (i.e., when a record is deleted from the Rent table). The trigger compares the current date with the item's due date. If the return is late, it automatically calculates the fine based on the item's **debyCost.**

Blocking Accounts: After a late return, the system updates the user's Card status to **'B' (Blocked).** This prevents them from borrowing any more items until the fine is paid.

Paying Fines: Users can pay their fines through a dedicated procedure. Once the fine is settled, the system updates their card status back to **'A' (Active)**.

EER Diagram


ID
Name
Address
Phone_number
Card_number
PERSON
1
Password
User_name
HAVE
ISA
CUSTOMER
Paycheck
Date_sing_up
EMPLOYEE
1
m
WORK
1
Number
CARD
Status
Fines
m
1
1
Name
BRANCH
1
Phone_number
return_date
RENT
date
Adress
n
m
State
ITEM
m
1
SITUATED
1
LOCATION
1
HAVE
Availability
ID
Delay_Cost
Lost_Cost
ISA
ISBN
BOOK
VIDEOS
Title
Year

Relational Schema

- Relational Schema
- Card (cardID, status, fines)
- Customer (customerID, name, customerAddress, phone, password, userName, dateSignUp, cardNumber)
- Employee (employeeID, name, employeeAddress, phone, password, userName, paycheck, branchName, cardNumber)
- Branch (name, address, phone)
- Location (address)
- Rent (cardID, itemID, apporpriationDate, returnDate)
- Book (ISBN, bookID, state, avalability, debyCost, lostCost, address)
- Video (title, year, videoID, state, avalability, debyCost, lostCost, address)