2017

"Take one pill twice a day hidden in some cheese."

Project Report: Pharmacy

CS 6360.002 – TEAM 23

ISHAN SHARMA – IXS171130, RAHUL NALAWADE - RSN170330

Introduction

Pharmacies are essential component of healthcare in the United States and handle the function of selling medical drugs. Even though the pharmacies do not seem different than any other shop, their functioning is very different due to various laws regarding drugs.

For example, most of the drugs available in a pharmacy cannot be purchased without a prescription. Even with a signed prescription, there is a limit on the quantity that can be purchased. Additionally, pharmacist can do a background check on customer's medical history to ensure that they are not involved in drug abuse.

In addition, there are other laws on the operations of pharmacy like requirement for safe disposal of expired medicine and requirement of license for employees that mix/prepare the drugs.

Thus, preparing a Database Management System for a pharmacy not only requires study of how things are handled from a customer or employee point of view but also the relevant laws. With this project, our aim was to develop a comprehensive system that could deal with challenges faced in day to day operation of a modern pharmacy. We studied the relevant laws and prepared a system that complies with the required Federal and State laws.

Table of Contents

Requirements

During research phase, we arrived at following requirements based on the pharmacy flow:

Customer

When a customer arrives in the pharmacy, we identify them based on their SSN. If they are a new customer, they are asked for their name, date of birth, phone number, gender and address. The address and date of birth are required to be recorded for drug control purposes under Omnibus Budget Reconciliation Act of 1990.

Insurance

89.6% of US population has health insurance coverage (National Center for Health Statistics, 2017). The coverage was significantly boosted by ACA(also called Obamacare). One reason for high coverage is that medical costs are very high in the country and without insurance, a large portion of population can't afford the healthcare.

If a customer has health insurance, we store the insurance ID (unique for each customer), company name, start date, end date and Co-Insurance. Co-Insurance is a percentage amount that insurance company pays for a medicinal purchase (Managing your healthcare costs, n.d.). Given the customer SSN and insurance ID, the system should be able to automatically calculate the amount paid by insurance company and customer.

Employee

An employee has same details as a customer but they are also given a company ID, that is unique for them. An employee has to have one of the following roles:

1. Pharmacist
2. CPhT (Certified Pharmacy Technician)
3. Intern (can work in the pharmacy part time)
4. Cashier

Apart from cashier, all other roles require a license from State's Medical Board as they directly deal with mixing and preparation of drugs. For the purpose of this report, we follow laws of Texas state.

Prescription

Most of the drugs in the pharmacy can only be sold with a prescription. A prescription contains customer's SSN, the prescribing Doctor's ID (required by law) and when the prescription was prescribed.

Each prescription contains a number of prescribed drugs with drug name, quantity and refill limit of each of them. By law, a pharmacy cannot sell more than prescribed quantity or anything that is not listed on prescription.

The prescription is required to be stored under CASA (1970).

Order

An order is created from the prescription. This data has to be stored separately because customer may:

1. Buy less medicine than prescription specifies
2. Come back for refills based on same prescription

Each order has a unique Order ID that is automatically assigned by the system. Each order can have multiple drugs, each with their ordered quantity and price. We also record batch number of the drug. This data can be requested by the government under Comprehensive Drug Abuse Prevention and Control Act (CASA) and has to be stored.

Bill

Once an order has been completed, a bill is generated by the system. This bill is handed over to the customer and contains order information, insurance information as well as breakdown of amount paid.

The breakdown should be automatically calculated by the system based on insurance, customer and medicine data.

Medicine(Inventory)

Drugs are divided into “over the counter”, “restricted” and “prescription only”. Federal Law only divides restricted drugs into 5 schedules and require “readily accessible” inventory for schedule 2 drugs.

While not needed by law everywhere, it is beneficial to store an up to date inventory for record keeping as well knowing when we run out of stock.

Notifications

The system should be able to generate notifications based on the following four events:

1. Stock for a medicine is low (less than 100 tablets)
2. Some medicine will expire in next 60 days
3. Drugs are marked for disposal
4. Drugs are successfully disposed

The notifications are sent to all the employees who are Pharmacists.

Laws Affecting Pharmacies

Since drugs are highly regulated, a system designed to manage a pharmacy has to follow all the required laws. The following are most important laws that we discovered during course of our research for the project:

Food, Drug and Cosmetic (FDC) Act of 1938

This law is more concerned with purity of drugs than regulation. It requires medicine makers to perform pretesting and sell only drugs that pass strict processes. We record the batch number for every order and all the medicine in stock. It helps us keep track of manufacturers who pharmacy purchases drugs from and in case of discrepancy in drug quality, find the responsible party.

Comprehensive Drug Abuse Prevention and Control Act(CASA) of 1970

CASA (Virgil Van Dusen, 2006) is one of the most impactful laws affecting a pharmacy. It requires comprehensive record keeping for restricted drugs. This act also separates restricted drugs into 5 categories, from Schedule I to V.

Schedule I drugs are not allowed to be sold by any pharmacy under any condition. If a drug is reclassified as Schedule I from any other category, the pharmacy has to destroy it responsibly.

Schedule II drugs can be sold, but accurate records have to be kept. The records need to be "readily retrievable" in case of an enquiry by law enforcement agencies. Schedule II drugs can't have refills and each purchase requires a separate prescription signed by the doctor.

Schedule III to V drugs can have refills for up to 6 months.

Poison Prevention Packaging Act(PPPA) of 1970

PPPA is concerned with ensuring drug safety for children. Before the act, there were many cases where children accidently consumed a drug when an adult was not around. To work around this, PPPA requires all drug packaging to be hard to open for children. All packaging has to be tested on at least 200 children below 5 years of age. If more than 20% cannot open the package in less than 5 minutes and more than 90% of the adults can comfortably open the packaging, it is deemed safe for use.

This is something that pharmacist and other staff have to take care of. We do not model this in our system.

Omnibus Budget Reconciliation Act (OBRA) of 1990

OBRA-90 (Sandra Christensen, 1991) focuses on how the pharmacists interact with customers and reducing the money wasted on drug related issues. Under the act, a pharmacist can access the central medication history of a patient if they need that a review is required before dispensing a medicine.

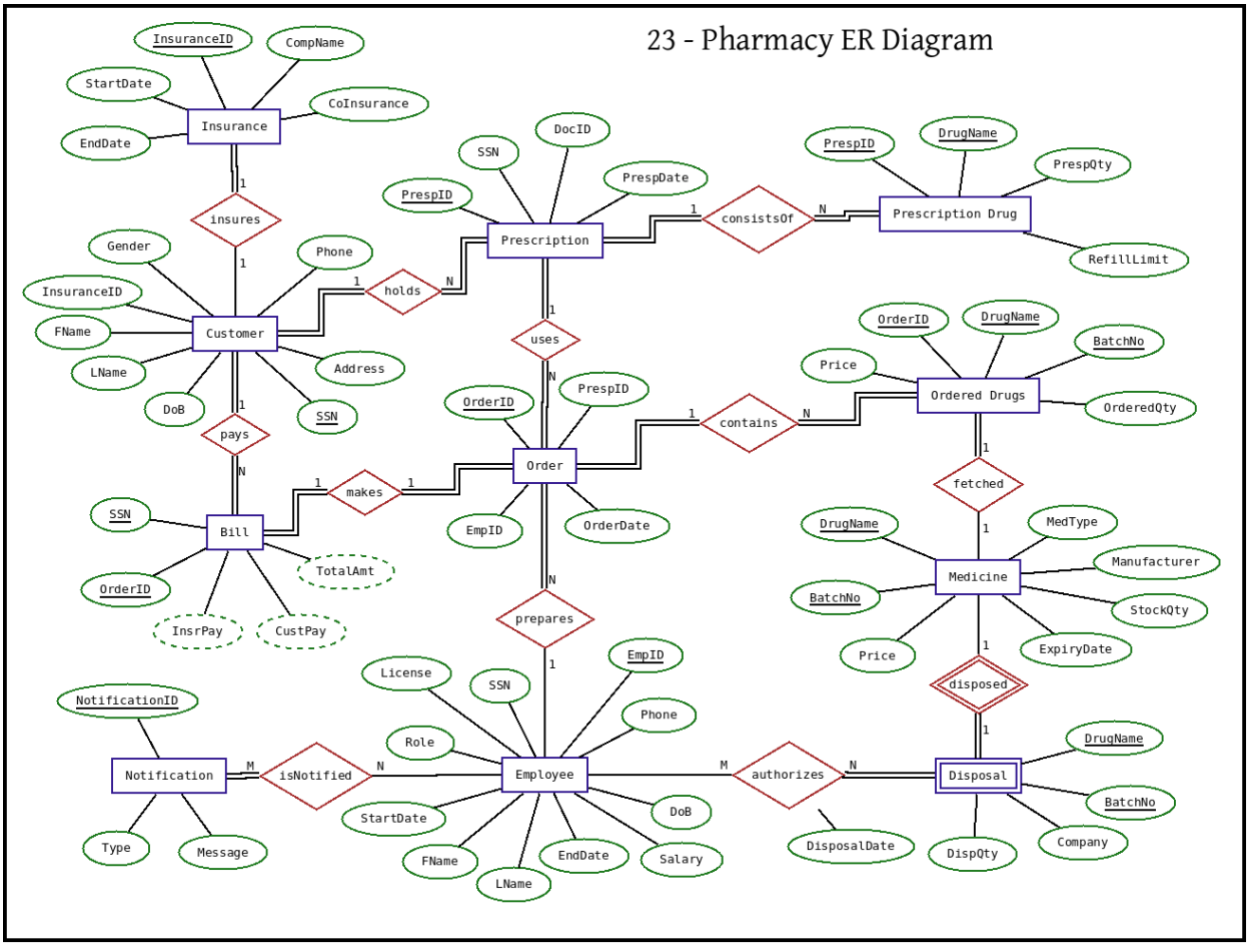
Health Insurance Portability and Accountability Act(HIPPA) of 1996

HIPPA is concerned with ensuring security of sensitive patient data. It requires the IT infrastructure to be secure and in case of any breaches, the affected parties have to be notified. Apart from that, proper authorization has to be put in place to ensure that only authorized employees have access to the patient data. Also, patient health conditions should not be revealed to anyone except the patient.

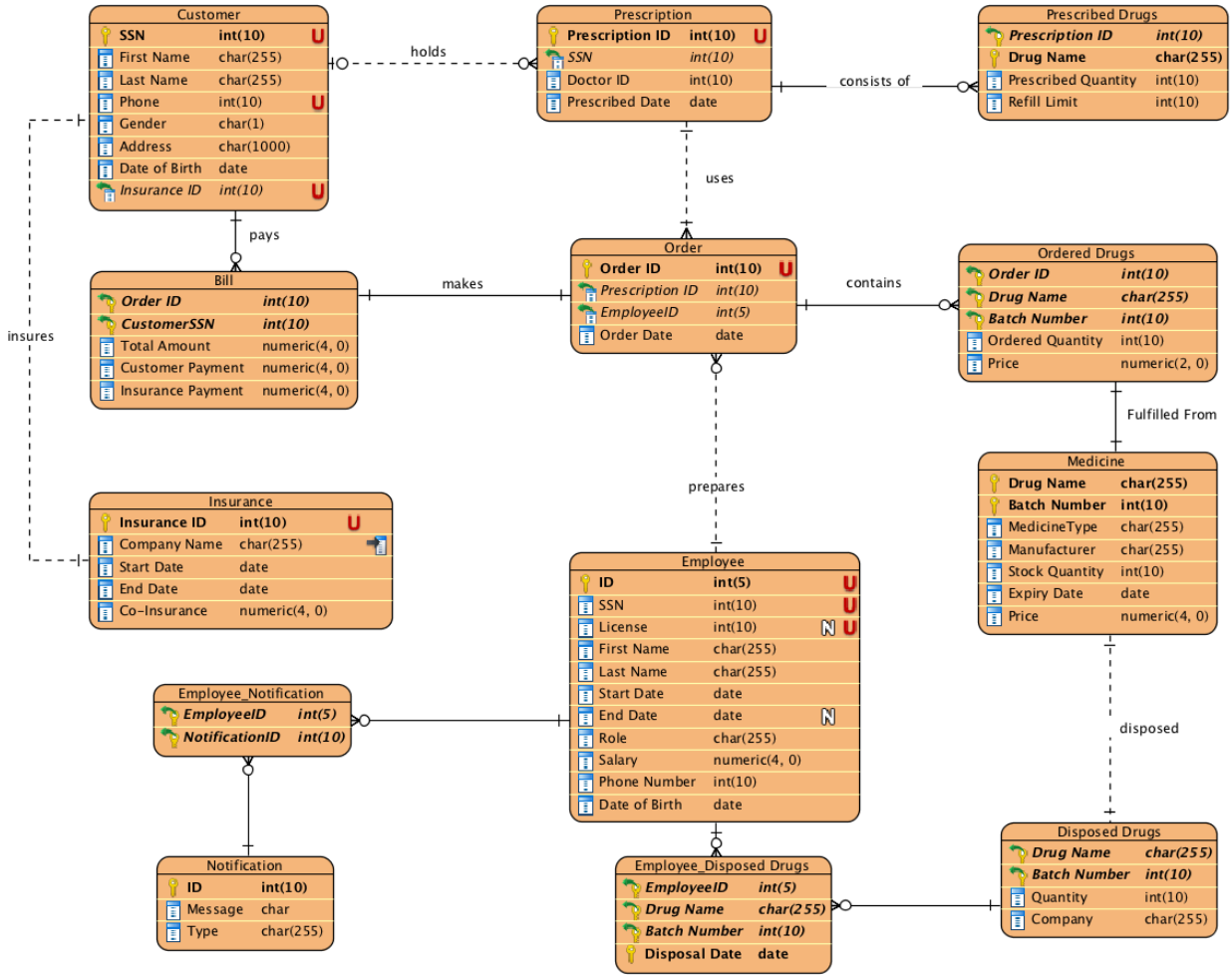
ER Modelling

The final ER diagram and UML diagram are shown below with explanations.

ER Diagram



UML



1. A single customer can have multiple prescriptions. Thus, the relation between them is one to many.
2. A prescription consists of multiple drugs, so the relation is one to many. In case of refills, a prescription can generate multiple orders. So, this relation is one to many as well.
3. A single order can contain multiple drugs, thus relationship is one to many. One order, however, can generate only one bill. Thus, the relation between bill and order is one to one.
4. A customer can make multiple purchases and hence, the relation between customer and bill is one to many. This is due to the fact that every bill has only one customer.
5. In medicine table (stock), drug name and batch number can uniquely identify every drug we have in inventory. Batch number is assumed to be unique among manufacturers.
6. Disposed drugs are weak entity and use foreign key Drug Name and Batch Number as their primary key.
7. One employee can receive multiple notifications and one notification can be sent to multiple employees, thus relationship is many to many.

8. Multiple employees can dispose same drug. Similarly, one employee can dispose multiple drugs. Hence, relationship is many to many.
9. One employee can prepare multiple orders. However, a specific order can only be prepared by one employee. Thus, relationship is one to many.

Relations and Normalization

Relations

The final relations are listed below:

Customer

SSN	First Name	Last Name	Phone	Gender	Address	Date of Birth	Insurance ID

Primary Key: SSN

Foreign Key: Customer(Insurance ID) → Insurance(Insurance ID)

Insurance

Insurance ID	Company Name	Start Date	End Date	Co-Insurance

Primary Key: Insurance ID

Employee

ID	SSN	License	First Name	Last Name	Start Date	End Date	Role
Salary	Phone Number	Date of Birth					

Primary Key: ID

Prescription

Prescription ID	SSN	Doctor ID	Prescription Date

Primary Key: Prescription ID
Foreign Key: Prescription(SSN) → Customer(SSN)

Prescribed Drugs

Prescription ID	Drug Name	Prescribed Quantity	Refill Limit

Primary Key: Prescription ID, Drug Name
Foreign Key: Prescribed Drugs(Prescription ID) → Prescription(Prescription ID)

Order

Order ID	Prescription ID	EmployeeID	Order Date

Primary Key: Order ID
Foreign Key: Order(Prescription ID) → Prescription(Prescription ID), Order(Employee ID) → Employee(ID)

Ordered Drugs

Order ID	Drug Name	Batch Number	Quantity	Price

Primary Key: Order ID, Drug Name, Batch Number
Foreign Key: Ordered Drugs(Order ID) → Order(Order ID), Ordered Drugs(Drug Name, Batch Number) → Medicine(Drug Name, Batch Number)

Bill

Order ID	CustomerSSN	Total Amount	Customer Payment	Insurance Payment

Primary Key: Order ID, Customer SSN
Foreign Key: Bill(Order ID) → Order(Order ID), Bill(Customer SSN) → Customer(SSN)

Medicine

Drug Name	Batch Number	Medicine Type	Manufacturer	Quantity	Expiry Date	Price

Primary Key: Drug Name, Batch Number

Disposed Drugs

Drug Name	Batch Number	Quantity	Company

Primary Key: Drug Name, Batch Number
Foreign Key: Disposed Drugs(Drug Name, Batch Number) → Medicine(Drug Name, Batch Number)

Notification

ID	Message	Type

Primary Key: ID

Employee_Disposed Drugs

Employee ID	Drug Name	Batch Number	Disposal Date

Primary Key: Employee ID, Drug Name, Batch Number, Disposal Date
Foreign Key: Employee_Disposed Drugs(Employee ID) → Employee (Employee ID), Employee_Disposed Drugs(Drug Name, Batch Number) → Disposed Drugs(Drug Name, Batch Number)

Employee Notification

Employee ID	Notification ID

Primary: Employee ID, Notification ID
Foregin Key: Employee Notification(Employee ID) → Employee(ID), Employee Notification(Notification ID) → Notification(Notification ID)

Normalization

The following dependencies exist in our schema:

1. Insurance(Insurance ID, Company Name, Start Date, End Date, Co-Insurance)
Insurance ID → Company Name, Start Date, End Date, Co-Insurance

2. Customer(SSN, First Name, Last Name, Phone, Gender, Address, Dateof Birth, Insurance ID)
SSN → First Name, Last Name, Phone, Gender, Address, Date of Birth, Insurance ID

3. Prescription(Prescription ID, SSN, Doctor ID, Prescribed Date)
Prescription ID → SSN, Doctor ID, Prescribed Date

4. Prescribed_Drugs(Prescription ID, Drug Name, Prescribed Quantity, Refill Limit)
Prescription ID, Drug Name → Prescribed Quantity, Refill Limit

5. Order(Order ID, Prescriptio ID, Employee ID, Order Date)
Order ID → Prescriptio ID, Employee ID, Order Date

6. Ordered Drugs(Order ID, Dug Name, Batch Number, Ordered Quantity, Price)
Order ID, Dug Name, Batch Number → Ordered Quantity, Price

7. Bill(Order ID, CustomerSSN, Total Amount, Customer Payment, Insurance Payment)
Order ID, CustomerSSN → Total Amount, Customer Payment, Insurance Payment

8. Employee(Employee ID, SSN, First Name, Last Name, Start Date, End Date, Role, Salary, Phone Number, Date of Birth)
Employee ID → SSN, First Name, Last Name, Start Date, End Date, Role, Salary, Phone Number, Date of Birth

9. Employee_Notification(Employee ID, Notification ID)
All Keys.

10. Notification(Notification ID, Type, Message)
Notification ID → Type, Message

11. Employee_Disposed_Drugs(Employee ID, Drug Name, Batch Number, Disposal Date)
All Keys.

12. Disposed Drugs(Drug Name, Batch Number, Quantity, Company)
Drug Name, Batch Number → Quantity, Company

13. Medicine(Drug Name, Batch Number, Medicine Type, Manufacturer, Stock Quantity, Expiry Date, Price)
Drug Name, Batch Number → Medicine Type, Manufacturer, Stock Quantity, Expiry Date, Price

None of the above dependencies violate 3NF rules, so above relations are in 3NF.

Table Creation

SQL commands for creating the tables in our database:

```
CREATE TABLE CUSTOMER (
  SSN           NUMBER(10) NOT NULL,
  first_name    CHAR(255)  NOT NULL,
  last_name     CHAR(255)  NOT NULL,
  phone         NUMBER(10) NOT NULL UNIQUE,
  gender        CHAR(1)    NOT NULL,
  address       CHAR(1000) NOT NULL,
  date_of_birth DATE       NOT NULL,
  insurance_id  NUMBER(10) NOT NULL UNIQUE,
  PRIMARY KEY (SSN)
);
ALTER TABLE Customer
  ADD CONSTRAINT insures FOREIGN KEY (insurance_id) REFERENCES Insurance (insurance_id) ON DELETE
SET NULL;

CREATE TABLE Prescription (
  prescription_id NUMBER(10) NOT NULL,
  SSN             NUMBER(10) NOT NULL,
  doctor_id       NUMBER(10) NOT NULL,
  prescribed_date DATE       NOT NULL,
  PRIMARY KEY (prescription_id)
);
ALTER TABLE Prescription
  ADD CONSTRAINT holds FOREIGN KEY (SSN) REFERENCES Customer (SSN);

CREATE TABLE "PRESCRIBED_DRUGS" (
  prescription_id     NUMBER(10) NOT NULL,
  drug_name           CHAR(255)  NOT NULL,
  prescribed_quantity NUMBER(10) NOT NULL,
  refill_limit        NUMBER(10) NOT NULL,
  PRIMARY KEY (prescription_id,
               drug_name)
);
ALTER TABLE "PRESCRIBED_DRUGS"
```

```
  ADD CONSTRAINT "consists of" FOREIGN KEY (prescription_id) REFERENCES Prescription (prescription_id) ON
DELETE CASCADE;

CREATE TABLE "Order" (
  order_id        NUMBER(10) NOT NULL,
  prescription_id NUMBER(10) NOT NULL,
  EmployeeID      NUMBER(5)  NOT NULL,
  order_date      DATE       NOT NULL,
  PRIMARY KEY (order_id)
);
ALTER TABLE "Order"
  ADD CONSTRAINT prepares FOREIGN KEY (EmployeeID) REFERENCES Employee (ID);
ALTER TABLE "Order"
  ADD CONSTRAINT uses FOREIGN KEY (prescription_id) REFERENCES Prescription (prescription_id);

CREATE TABLE "ORDERED_DRUGS" (
  order_id         NUMBER(10) NOT NULL,
  drug_name        CHAR(255)  NOT NULL,
  batch_number     NUMBER(10) NOT NULL,
  ordered_quantity NUMBER(10) NOT NULL,
  Price            NUMBER(2)  NOT NULL,
  PRIMARY KEY (order_id,
               drug_name,
               batch_number)
);
ALTER TABLE "ORDERED_DRUGS"
  ADD CONSTRAINT "contains" FOREIGN KEY (order_id) REFERENCES "Order" (order_id) ON DELETE
CASCADE;
ALTER TABLE "ORDERED_DRUGS"
  ADD CONSTRAINT "Fulfilled From" FOREIGN KEY (drug_name, batch_number) REFERENCES Medicine
(drug_name, batch_number);

CREATE TABLE Insurance (
  insurance_id NUMBER(10) NOT NULL,
  company_name CHAR(255)  NOT NULL,
  start_date   DATE       NOT NULL,
  end_date     DATE       NOT NULL,
```

```
  co_insurance NUMBER(4)  NOT NULL,
  PRIMARY KEY (insurance_id)
);
CREATE INDEX "Insurance_Company Name"
  ON Insurance (company_name);

CREATE TABLE Employee (
  ID           NUMBER(5)  NOT NULL,
  SSN          NUMBER(10) NOT NULL UNIQUE,
  License      NUMBER(10) UNIQUE,
  first_name   CHAR(255)  NOT NULL,
  last_name    CHAR(255)  NOT NULL,
  start_date   DATE       NOT NULL,
  end_date     DATE,
  role         CHAR(255)  NOT NULL,
  salary       NUMBER(4)  NOT NULL,
  phone_number NUMBER(10) NOT NULL,
  date_of_birth DATE      NOT NULL,
  PRIMARY KEY (ID)
);

CREATE TABLE Medicine (
  drug_name     CHAR(255)  NOT NULL,
  batch_number  NUMBER(10) NOT NULL,
  MedicineType  CHAR(255)  NOT NULL,
  Manufacturer  CHAR(255)  NOT NULL,
  stock_quantity NUMBER(10) NOT NULL,
  expiry_date   DATE       NOT NULL,
  Price         NUMBER(4)  NOT NULL,
  PRIMARY KEY (drug_name,
               batch_number)
);

CREATE TABLE Bill (
  order_id         NUMBER(10) NOT NULL,
  CustomerSSN      NUMBER(10) NOT NULL,
  total_amount     NUMBER(4)  NOT NULL,
```

```
  customer_payment  NUMBER(4)  NOT NULL,
  insurance_payment NUMBER(4)  NOT NULL,
  PRIMARY KEY (order_id,
               CustomerSSN)
);
ALTER TABLE Bill
  ADD CONSTRAINT makes FOREIGN KEY (order_id) REFERENCES "Order" (order_id);
ALTER TABLE Bill
  ADD CONSTRAINT pays FOREIGN KEY (CustomerSSN) REFERENCES Customer (SSN);

CREATE TABLE "Disposed Drugs" (
  drug_name    CHAR(255)  NOT NULL,
  batch_number NUMBER(10) NOT NULL,
  Quantity     NUMBER(10) NOT NULL,
  Company      CHAR(255)  NOT NULL,
  PRIMARY KEY (drug_name,
               batch_number)
);
ALTER TABLE "Disposed Drugs"
  ADD CONSTRAINT disposed FOREIGN KEY (drug_name, batch_number) REFERENCES Medicine (drug_name,
batch_number);

CREATE TABLE Notification (
  ID      NUMBER(10) NOT NULL,
  Message CHAR(255)  NOT NULL,
  Type    CHAR(255)  NOT NULL,
  PRIMARY KEY (ID)
);

CREATE TABLE Employee_Notification (
  EmployeeID     NUMBER(5)  NOT NULL,
  NotificationID NUMBER(10) NOT NULL,
  PRIMARY KEY (EmployeeID,
               NotificationID)
);
ALTER TABLE Employee_Notification
  ADD CONSTRAINT FKEmployee_N849182 FOREIGN KEY (EmployeeID) REFERENCES Employee (ID) ON
```

```
DELETE CASCADE;
ALTER TABLE Employee_Notification
  ADD CONSTRAINT FKEmployee_N664471 FOREIGN KEY (NotificationID) REFERENCES Notification (ID) ON
DELETE CASCADE;

CREATE TABLE "EMPLOYEE_DISPOSED_DRUGS" (
  EmployeeID    NUMBER(5)  NOT NULL,
  drug_name     CHAR(255)  NOT NULL,
  batch_number  NUMBER(10) NOT NULL,
  disposal_date DATE       NOT NULL,
  PRIMARY KEY (EmployeeID,
               drug_name,
               batch_number,
               disposal_date)
);
ALTER TABLE "EMPLOYEE_DISPOSED_DRUGS"
  ADD CONSTRAINT FKEmployee_D470142 FOREIGN KEY (EmployeeID) REFERENCES Employee (ID);
ALTER TABLE "EMPLOYEE_DISPOSED_DRUGS"
  ADD CONSTRAINT FKEmployee_D990025 FOREIGN KEY (drug_name, batch_number) REFERENCES "Disposed
Drugs" (drug_name, batch_number);
```

Procedures & Triggers

Procedures

We have 4 stored procedures:

1. Generate Bill
2. Add Drug to Order
3. Report Expiring Drugs
4. Dispose Expired Drugs

Generate Bill

Making payments is slightly different in the pharmacy as majority of customers have insurance that supports co-payment. It means that insurance company pays a part of the bill while the patient pays rest of it. The data is stored in the database in the insurance table.

When the bill is generated, total amount is calculated based on ordered drugs and then copayment and customer payment is automatically calculated.

```
CREATE OR REPLACE
PROCEDURE GENERATE_BILL
 (
   order_id     IN INT,
   ssn          IN INT,
   insurance_id IN INT
 )
AS
 total_amount        NUMBER;
 copayment_percentage NUMBER;
 copayment_amount    NUMBER; -- this is the amount insurance company pays
 customer_payment    NUMBER; -- this is the amount customer pays
 BEGIN
   -- do a total of all orders
   SELECT SUM('price')
   INTO total_amount
   FROM ORDERED_DRUGS
   WHERE 'order_id' = order_id;

   -- get insurance details
   SELECT co_insurance
   INTO copayment_percentage
```

```
  FROM INSURANCE
  WHERE 'insurance_id' = insurance_id;

  -- the insurance company will pay this amount
  copayment_amount = total_amount * copayment_percentage;

  -- the customer will pay this amount
  customer_payment = total_amount * (1 - copayment_percentage);

  -- Insert data
  INSERT INTO BILL VALUES (order_id, ssn, total_amount, customer_payment, copayment_amount);
END;
```

Add Drug to Order

When adding drug to the order, specified quantity has to be added to order and same quantity has to be removed from the stock. If the quantity to be added is more than that in stock, an exception is thrown and order is not created.

```
CREATE OR REPLACE
PROCEDURE ADD_DRUG_TO_ORDER
 (
  order_id  IN INT,
  drug_name IN CHAR(255),
  quantity  IN INT
 )
AS
 drug MEDICINE%ROWTYPE;
  insufficient_quantity EXCEPTION;
BEGIN
  SELECT *
  INTO drug
  FROM MEDICINE
  WHERE 'drug_name' = drug_name;

  IF drug.quantity < quantity
  THEN
    RAISE insufficient_quantity;
  ELSE
```

```
    INSERT INTO ORDERED_DRUGS
    VALUES (order_id, drug.drug_name, drug.batch_number, quantity, drug.price);
    DBMS_OUTPUT.PUT_LINE("Drug added successfully to the order");
  END IF;

  EXCEPTION
  WHEN insufficient_quantity THEN
  DBMS_OUTPUT.PUT_LINE(
      "Request drug " || drug_name || " is not available. Maximum order possible is " || drug.quantity
  );
END;
```

Report Expiring Drugs

Any drugs that are going to expire within 60 days are displayed on screen along with their quantity and batch number.

```
CREATE OR REPLACE
PROCEDURE REPORT_EXPIRING_DRUGS
AS
  BEGIN
    DBMS_OUTPUT.PUT_LINE('ALL DRUGS EXPIRING IN NEXT 60 DAYS');
    -- date calculations from
    -- http://www.oracle.com/technetwork/issue-archive/2012/12-jan/o12plsql-1408561.html
    FOR item IN
    (
    SELECT
      'drug_name',
      'batch_number',
      'manufacturer',
      'stock_quantity',
      'expiry_date'
    FROM MEDICINE
    WHERE 'expiry_date' < SYSDATE + 60
    )
    LOOP
      DBMS_OUTPUT.PUT_LINE(
        item.drug_name || item.batch_number || item.manufacturer || item.stock_quantity || item.expiry_date)
```

```
  END LOOP;
END;
```

Dispose Expired Drugs

This procedure will move any drugs that have expired in past or are expiring today from stock and move them to disposed drugs. Additionally, it will also send a notification to the pharmacists.

```
CREATE OR REPLACE
PROCEDURE DISPOSE_DRUGS
 (
   drug_name    IN CHAR(255),
   batch_number IN INT,
   employee_id  IN INT
 )
AS
 medicine            MEDICINE%ROWTYPE;
 new_notification_id  INT;
 notification_message CHAR(1000);
 BEGIN
   SELECT *
   INTO medicine
   FROM MEDICINE
   WHERE MEDICINE.drug_name = drug_name AND MEDICINE.batch_number = batch_number
       AND MEDICINE.expiry_date <= SYSDATE;
   IF medicine%FOUND
   THEN
    INSERT INTO DISPOSE_DRUGS VALUES (drug_name, batch_number, medicine.quantity,
medicine.manufacturer);
    INSERT INTO "EMPLOYEE_DISPOSED_DRUGS" VALUES (employee_id, drug_name, batch_number,
SYSDATE);
    DELETE FROM MEDICINE
    WHERE MEDICINE.drug_name = drug_name AND MEDICINE.batch_number AND MEDICINE.expiry_date <
SYSDATE;

    -- Operations are done, we have to send notification now
    SELECT MAX(ID) + 1
```

```
    INTO new_notification_id
    FROM NOTIFICATION;
    notification_message = 'Successfully Disposed: ' || drug_name || '(' || batch_number || ') by Employee' ||
    INSERT INTO NOTIFICATION VALUES (new_notification_id, notification_message, 'DISPOSAL_SUCCESS');

    -- SEND NOTIFICATION TO EMPLOYEES
    EXECUTE IMMEDIATE SEND_NOTIFICATIONS(new_notification_id, 'pharmacist');
  END IF;
 END;
```

Send Notifications

Sending of notifications is recorded in “EMPLOYEE_NOTIFICATION” relation. We are assuming that a third party tool will pick notifications from there and send to the relevant employees. Email/Text is out of scope for the project.

```
CREATE OR REPLACE
PROCEDURE SEND_NOTIFICATIONS
 (
   notification_id IN INT,
   employee_role   IN CHAR(100)
 )
AS
 BEGIN
   FOR employee IN
   (
   SELECT 'ID'
   FROM EMPLOYEE
   WHERE LOWER(EMPLOYEE.role) = employee_role
   )
   LOOP
     INSERT INTO EMPLOYEE_NOTIFICATIONS VALUES (employee, notification_id);
   END LOOP;
 END;
```

Triggers

Notifications

We have four types of notifications in our system. A notification is generated when a medicine is added to the order from stock and stock goes below 100. Additionally, there's a check to see when the medicine will expire. If it expires within next 60 days, a notification is generated. Some of the notifications are handled inside procedures.

Low stock notification can be automatically sent when adding a medicine to the order makes the quantity too low for a medicine.

```
CREATE OR REPLACE TRIGGER Low_Stock_Alert
AFTER INSERT OR UPDATE ON MEDICINE
FOR EACH ROW
 DECLARE
   new_notification_id INT;
 BEGIN

   IF :NEW.stock_quantity < 100
   THEN
     SELECT MAX(ID) + 1
     INTO new_notification_id
     FROM NOTIFICATION;
     INSERT INTO NOTIFICATION VALUES (new_notification_id,
                                :OLD.drug_name || ' batch - ' || :OLD.batch_number || ' has low stock. Only ' ||
                                :NEW:quantity || ' in stock', 'LOWSTOCK');

     -- Send notification to Pharmacists
     EXECUTE IMMEDIATE SEND_NOTIFICATIONS(new_notification_id, 'pharmacist');
   END IF;
 END;
```

Employee Validation

When an employee is added, the employee type should be one of:

1. Pharmacist
2. CPhT
3. Intern
4. Cashier

Except cashier, every other role requires a license.

```
CREATE OR REPLACE TRIGGER Validate_Employee
BEFORE INSERT OR UPDATE ON EMPLOYEE
FOR EACH ROW
 BEGIN
  IF LOWER(:NEW.role) != 'cashier' OR LOWER(:NEW.role != 'pharmacist') OR LOWER(:NEW.role != 'cpht') OR
    LOWER(:NEW.role != 'intern')
  THEN
    RAISE_APPLICATION_ERROR(-20000, 'Invalid role given for employee');
  END IF;

  IF :NEW.license := NULL AND LOWER(:new.role) != 'cashier'
  THEN
    RAISE_APPLICATION_ERROR(-20000, 'Can not leave license blank for anyone except cashiers');
  END IF;
 END;
```

Conclusion

The pharmacy project was a good learning experience for implementing a real world DBMS and helped us understand the nuances of a full implementation.

The most interesting part was the experience of starting from real world and then translating the concepts into the terms of a DBMS. The final implementation is robust and can handle various edge cases and scenarios. Paired with a capable application front end, it can handle day to day operations for a pharmacy.

Citations and References

Managing your healthcare costs. (n.d.). Retrieved from Humana: https://www.humana.com/all-products/understanding-insurance/health-insurance-cost

National Center for Health Statistics. (2017, March 31). Retrieved from Centers for Disease Control and Prevention: https://www.cdc.gov/nchs/fastats/health-insurance.htm

Sandra Christensen, H. L. (1991, Feburary). *MEDICARE AND THE OMNIBUS BUDGET RECONCILIATIONACT OF 1990: IMPACT ON ENROLLEES, HOSPITALS, AND PHYSICIANS.* Retrieved from Congressional Budget Office: https://www.cbo.gov/sites/default/files/102nd-congress-1991-1992/reports/199102medicareandobra.pdf

Virgil Van Dusen, R. J. (2006, Dec 01). A Review of Federal Legislation Affecting Pharmacy Practice. *Pharmacy Times*, p. 1.