****

**ISTE.230.602**

**Introduction to Database and Data Modelling**

**Final Project - Group 1**

By

Sara Hijazi, **ID:** 387006935

Jamile Obeid, **ID:** 762004935

Lana Kendakji, **ID:** 757001591

**Supervised by**

Dr. Zainab Al-Zanbouri

Department of Electrical Engineering and Computing

Rochester Institute of Technology, Dubai

United Arab Emirates

May 2025

**Table of Contents**

**Business Rules** 3

Business Scenario 3

Core Business and User Requirements 3

Database Schema and Entity Relationships 3

**Relationships** 8

**ER Diagram** 9

**Transposition** 10

**Normalization** 12

The First Normal Form 12

The Second Normal Form 12

The Third Normal Form 14

**Updated ER Diagram** 17

**Updated Transposition** 17

**Database Implementation** 18

Data Definition Language (DDL) - Defining the tables 19

Data Manipulation Language (DML) - Inserting tables with data 23

**Relational Algebra Statements** 28

**Composing SQL Queries** 29

**Business Rules**

## **1. Business Scenario**

Cozy Bookstore is a small, independent bookstore with a physical location that offers a range of products including books (fiction, non-fiction, and textbooks), CDs, vinyl records, stationery, and toys. To efficiently manage its inventory, process sales, and track customer purchases, the bookstore uses a relational database management system (RDBMS). This database system supports business operations and smooth customer transactions and data integrity.

**2. Core Business and User Requirements**

### 2.1 Customer Management

* Employees register customers by collecting their phone number and name to create a unique customer account.
* Customers may opt for a membership subscription, granting them discounts on purchases.

### 2.2 Sales Processing

* Employees can enter a Product ID to retrieve product details such as price and stock quantity.
* The system records which employee processes a sale and which customer makes a purchase.
* Sold product quantities are automatically updated in the inventory.

### 2.3 Inventory Management

* Employees can add new products to the inventory.
* Books are categorized into:
  + Fiction (by genre: Romance, Thriller, Sci-Fi, etc.)
  + Non-Fiction (classified using the Dewey Decimal System)
  + Textbooks (categorized by subject and grade level)
* CDs and vinyl records are categorized by music genre.
* Stationery is classified into:
  + School Stationery (e.g., Pencils, Pens, Erasers, Sharpeners, Rulers)
  + Art Supplies (e.g., Paint, Crayons, Canvas, Brushes)
* Toys are categorized into:
  + Regular Toys
  + Language Games
  + Board Games

### 2.4 Employee Role and Work Tracking

* Each employee has a unique Employee ID and is classified as either Full-Time or Part-Time.
* The system records which employee processes each sale.
* An experienced employee can be assigned a new employee to train/mentor

## 

## **3. Database Schema and Entity Relationships**

Key:

BLUE = table PINK = Attribute

### 3.1 Customer Management Tables

#### **Customers Table**

* Stores customer details.
* Attributes:
  + Customer\_ID (Primary Key)
  + Name
  + Phone\_Number

#### **Subscribed Customers Table**

* Tracks customers who have opted for a membership subscription.
* Attributes:
  + Customer\_ID (Primary Key, Foreign Key referencing Customers.Customer\_ID)
  + Discount\_Percentage

### 3.2 Employee Management Tables

#### **Employees Table**

* Stores employee details and employment type.
* Attributes:
  + Employee\_ID (Primary Key)
  + Name
  + Employment\_Type (Full-Time/Part-Time)

#### **Full-Time Employees Table**

* Stores additional details for full-time employees.
* Attributes:
  + Employee\_ID (Primary Key, Foreign Key referencing Employees.Employee\_ID)
  + Salary

#### **Part-Time Employees Table**

* Stores additional details for part-time employees.
* Attributes:
  + Employee\_ID (Primary Key, Foreign Key referencing Employees.Employee\_ID)
  + Hourly\_Rate

### 3.3 Sales Processing Tables

#### **Sales Table**

* Manages sales transactions.
* Attributes:
  + Sale\_ID (Primary Key)
  + Customer\_ID (Foreign Key referencing Customers.Customer\_ID)
  + Employee\_ID (Foreign Key referencing Employees.Employee\_ID)
  + Sale\_Date

#### **Sale Products Table**

* Handles the Many-to-Many relationship between sales and products.
* Attributes:
  + Sale\_ID (Primary Key, Foreign Key referencing Sales.Sale\_ID)
  + Product\_ID (Primary Key, Foreign Key referencing Products.Product\_ID)
  + Quantity

3.4 Product Management Tables

#### **Products Table**

* Stores details of all products.
* Attributes:
  + Product\_ID (Primary Key)
  + Category (Book, CD, Vinyl, Stationery, Toy, etc.)
  + Price
  + Stock\_Quantity

### 3.5 Book Management Tables

#### **Books Table**

* Stores book-specific attributes.
* Attributes:
  + Product\_ID (Primary Key, Foreign Key referencing Products.Product\_ID)
  + ISBN
  + Title
  + Publisher
  + Book\_Type (Fiction, Non-Fiction, Textbook)
  + Nobel\_Prize\_Winner
  + New\_York\_Times\_Bestseller
  + Author\_ID (Foreign Key referencing Authors.Author\_ID)
  + Author\_ Name
  + Author\_Nationality
  + Author\_Birth\_Year
  + Author\_Award\_ID
  + Author\_Award\_Name
  + Author\_Year\_Won

#### **Book Subcategories**

* Fiction Books: Product\_ID, Genre
* Non-Fiction Books: Product\_ID, Dewey\_Category
* Textbooks: Product\_ID, Subject, Grade\_Level

### 3.6 Music Inventory Management Tables

#### **CDs Table**

* Attributes:
  + Product\_ID (Primary Key, Foreign Key referencing Products.Product\_ID)
  + Artist
  + Album\_Name
  + Genre

#### **Vinyl Records Table**

* Attributes:
  + Product\_ID (Primary Key, Foreign Key referencing Products.Product\_ID)
  + Artist
  + Album\_Name
  + Genre

### 3.7 Toy Inventory Management Tables

#### **Toys Table**

* Attributes:
  + Product\_ID (Primary Key, Foreign Key referencing Products.Product\_ID)
  + Toy\_Type (Board Game, Language Game)

### Toy Subcategories

**Board Games**

* **Attributes:**
  + Product\_ID (Primary Key, Foreign Key references Toys.Product\_ID)
  + Age\_Recommendation

**Language Games**

* **Attributes:**
  + Product\_ID (Primary Key, Foreign Key references Toys.Product\_ID)
  + Language (e.g., English, French, Spanish)

### 3.8 Stationery Inventory Management Tables

#### **Stationery Table**

* Attributes:
  + Product\_ID (Primary Key, Foreign Key referencing Products.Product\_ID)
  + Stationery\_Type (School Stationery, Art Supplies)

### 3.9 Stationery Subcategories

**School Stationery**

* **Attributes:**
  + Product\_ID (Primary Key, Foreign Key references Stationery.Product\_ID)
  + Item\_Type (e.g., Notebook, Pen, Eraser)

**Art Supplies**

* **Attributes:**
  + Product\_ID (Primary Key, Foreign Key references Stationery.Product\_ID)
  + Material (e.g., Watercolor, Acrylic, Sketching Pencils)

## 4. Entity-Relationship Model

The relational database schema ensures proper normalization and relationships between tables, with One-to-Many (1:M), Many-to-Many (M:M), and One-to-One (1:1) relationships appropriately implemented. The ER diagram represents these connections visually.

**Relationships**

**Employees and Sales** (*1-to-Many*):

* Each sale is handled by one employee, but an employee can handle zero or multiple sales.

**Customers and Sales** (*1-to-Many*):

* A customer can make 1 or multiple purchases (sales), but a sale belongs to only one customer.

**Sales and Products** (*Many-to-Many*):

* A sale contains 1 or multiple products, and a product can be in 0 (new product) or multiple sales. Sale\_Product solves this problem.

**Sales and Sale\_Details** *(1-to-Many*):

* One sale can include multiple products, each entry in Sale\_Details belongs to a single sale.

**Products and Sale\_Details** (*1-to-Many*):

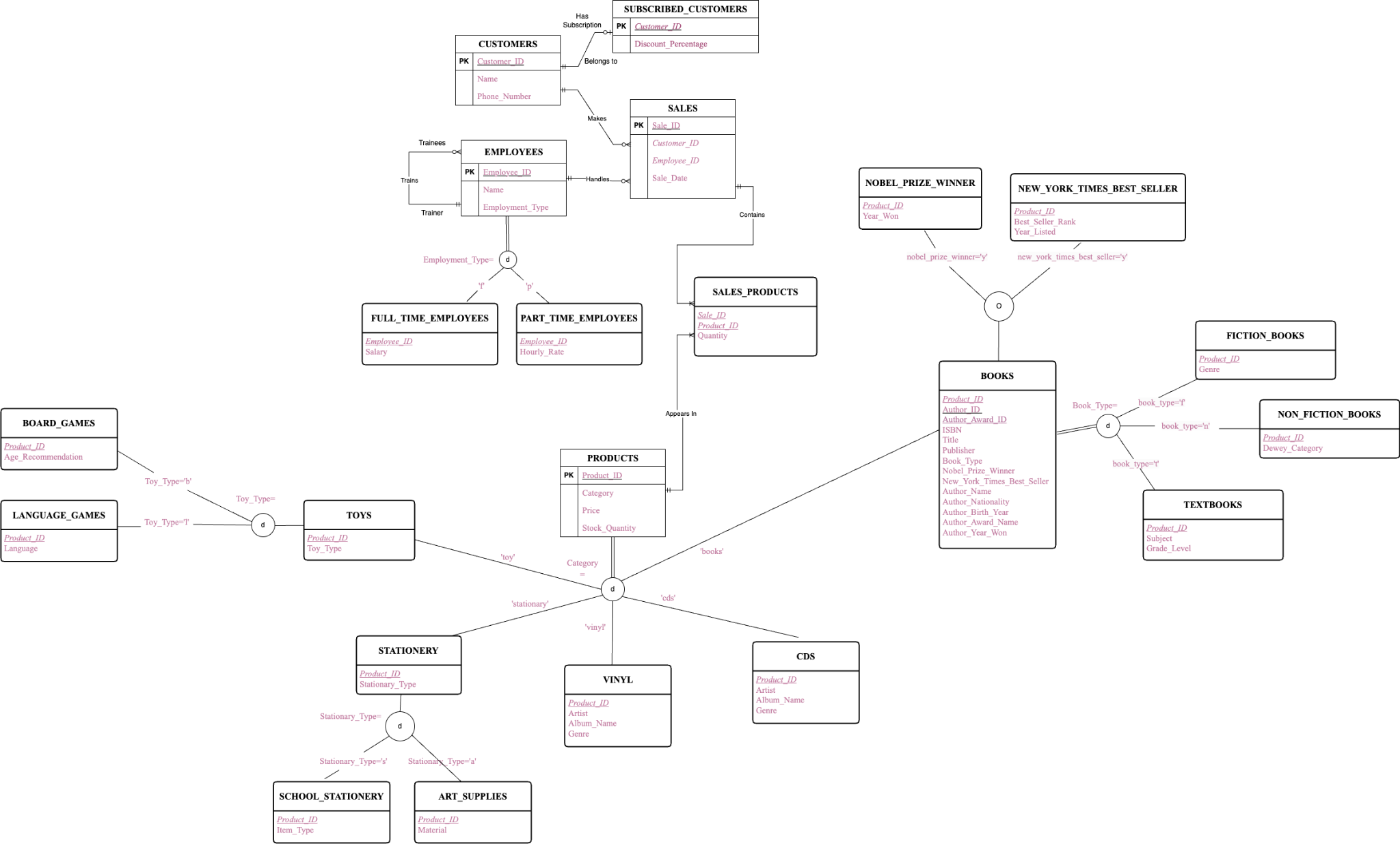
* A product can appear in multiple sales, each entry in Sale\_Details belongs to a single product.

**Employees and Employees** (*Unary/Recursive 1-to-Many*)**:**

* An experienced employee can train zero or many new employees, while each trainee is trained by one employee.

**ER Diagrams**

The below Entity-Relationship Diagram was created using the ‘*draw io*’ website (<https://app.diagrams.net/>) by following the business rules outlined in the previous pages. The ER-Diagrams created incorporate one-to-many, many-to-many, and unary relationships as well as partial and total specializations. This ERD will be transposed and subsequently normalized to ensure data integrity, reduce data redundancy, and support efficient database implementation.



Following the normalization process, an updated version of the Entity-Relationship Diagram will be created to reflect the refined structure of the database.

**Transposition**

**CUSTOMERS(**Customer\_ID, Name, Phone\_Number**)**

**SUBSCRIBED\_CUSTOMERS(***Customer\_ID,* Discount\_Percentage**)**

**SUBSCRIBED\_CUSTOMERS(**Customer\_ID**) mei CUSTOMERS(**Customer\_ID**)**

**EMPLOYEES(**Employee\_ID, Name, Employment\_Type, *Trainer\_ID***)**

**EMPLOYEES(**Trainer\_ID**) mei EMPLOYEES(**Employee\_ID**)**

**FULL\_TIME\_EMPLOYEES(***Employee\_ID,* Salary**)**

**FULL\_TIME\_EMPLOYEES(**Employee\_ID**) mei EMPLOYEES(**Employee\_ID**)**

**PART\_TIME\_EMPLOYEES(***Employee\_ID,* Hourly\_Rate**)**

**PART\_TIME\_EMPLOYEES(**Employee\_ID**) mei EMPLOYEES(**Employee\_ID**)**

**SALES(**Sale\_ID, *Customer\_ID, Employee\_ID,* Sale\_Date**)**

**SALES(**Customer\_ID**) mei CUSTOMERS(**Customer\_ID**)**

**SALES(**Employee\_ID**) mei EMPLOYEES(**Employee\_ID**)**

**PRODUCTS(**Product\_ID, Category, Price, Stock\_Quantity**)**

**SALES\_PRODUCTS(***Sale\_ID, Product\_ID,* Quantity**)**

**SALES\_PRODUCTS(**Sale\_ID**) mei SALES(**Sale\_ID**)**

**SALES\_PRODUCTS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**BOOKS(**Author\_ID, Author\_Award\_ID, *Product\_ID*, ISBN, Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller, Author\_Name, Author\_Nationality, Author\_Birth\_Year, Author\_Award\_Name, Author\_Year\_Won**)**

**BOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**FICTION\_BOOKS(***Product\_ID,* Genre**)**

**FICTION\_BOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**NON\_FICTION\_BOOKS(***Product\_ID*, Dewey\_Category**)**

**NON\_FICTION\_BOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**TEXTBOOKS(***Product\_ID*, Subject, Grade\_Level**)**

**TEXTBOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**NOBEL\_PRIZE\_WINNER(***Product\_ID*, Year\_Won**)**

**NOBEL\_PRIZE\_WINNER(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**NEW\_YORK\_TIMES\_BEST\_SELLER(***Product\_ID*, Best\_Seller\_Rank, Year\_Listed**)**

**NEW\_YORK\_TIMES\_BEST\_SELLER(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**CDS(***Product\_ID,* Artist, Album\_Name, Genre**)**

**CDS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**VINYL(***Product\_ID,* Artist, Album\_Name, Genre**)**

**VINYL(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**TOYS(***Product\_ID,* Toy\_Type**)**

**TOYS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**BOARD\_GAMES(***Product\_ID,* Age\_Recommendation**)**

**BOARD\_GAMES(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**LANGUAGE\_GAMES(***Product\_ID,* Language**)**

**LANGUAGE\_GAMES(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**STATIONERY(***Product\_ID,* Stationery\_Type**)**

**STATIONERY(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**SCHOOL\_STATIONERY(***Product\_ID,* Item\_Type**)**

**SCHOOL\_STATIONERY(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**ART\_SUPPLIES(***Product\_ID,* Material**)**

**ART\_SUPPLIES(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**Normalization**

**The First Normal Form**

* *Criteria*
  + Atomic Values
  + Unique Rows
  + Unique Attributes
  + Consistent Domain
* *Applicable Tables*
  + All tables in the database as they fit the specified criteria

**The Second Normal Form**

* *Criteria*
  + Satisfies first normal form
  + All non-key attributes are dependent on the entire primary key
  + No partial dependencies
* *Applicable Tables*
  + All tables in the database except for BOOKS as it violates the criteria by having partial dependencies
* *Transitioning to Second Normal Form*
  + **BOOKS Table**

| **Initial Functional Dependencies** | **Transitioning to Second Normal Form (Step 1)** |
| --- | --- |
| * Product\_ID, Author\_ID, Author\_Award\_ID → ISBN, Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller, Author\_Name, Author\_Nationality, Author\_Birth\_Year, Author\_Award\_ID, Author\_Award\_Name, Author\_Year\_Won * Author\_ID, Author\_Award\_ID → Author\_Name, Author\_Nationality, Author\_Birth\_Year, Author\_Award\_Name, Author\_Year\_Won *(Partial Dependency)* | **BOOKS (***Product\_ID***,** ISBN, Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller, *Author\_ID***)**  **BOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**  **BOOKS(**Author\_ID**) mei AUTHORS(**Author\_ID**)**  **AUTHORS(**Author\_ID, Author\_Award\_ID, Author\_Name, Author\_Nationality, Author\_Birth\_Year, Author\_Award\_Name, Author\_Year\_Won **)** |
| **Updated Functional Dependencies** | **Transitioning to Second Normal Form (Step 2)** |
| * *(BOOKS)* Product\_ID → ISBN, Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller, Author\_ID * *(AUTHORS)* Author\_ID, Author\_Award\_ID → Author\_Name, Author\_Nationality, Author\_Birth\_Year, Author\_Award\_Name, Author\_Year\_Won * *(AUTHORS)* Author\_ID → Author\_Name, Author\_Nationality, Author\_Birth\_Year *(Partial Dependency)* * *(AUTHORS)* Author\_Award\_ID → Author\_Award\_Name, Author\_Year\_Won *(Partial Dependency)* | **BOOKS (***Product\_ID***,** ISBN, Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller, *Author\_ID***)**  **BOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**  **BOOKS(**Author\_ID**) mei AUTHORS(**Author\_ID**)**  **AUTHORS(**Author\_ID, Author\_Name, Author\_Nationality, Author\_Birth\_Year, *Author\_Award\_ID***)**  **AUTHORS(**Author\_Award\_ID**) mei AUTHOR\_AWARDS(**Author\_Award\_ID)  **AUTHOR\_AWARDS(**Author\_Award\_ID, Author\_Award\_Name, Author\_Year\_Won**)** |

**The Third Normal Form**

* *Criteria*
  + Satisfies first normal form
  + Satisfies second normal form
  + No transitive dependencies
* *Applicable Tables*
  + All tables in the database except for BOOKS and CUSTOMERS as they violate the criteria by having transitive dependencies
* *Transitioning to Third Normal Form*
  + **BOOKS Table**

| **Initial Functional Dependencies** | **Transitioning to Third Normal Form** |
| --- | --- |
| * Product\_ID → ISBN, Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller, Author\_ID * ISBN → Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller *(Transitive Dependency)* | **BOOKS(***Product\_ID*,*ISBN,Author\_ID***)**  **BOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**  **BOOKS(**Author\_ID**) mei AUTHORS(**Author\_ID**)**  **BOOKS(**ISBN**) mei BOOKS\_INFO(**ISBN**)**  **BOOKS\_INFO(**ISBN,Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller**)** |

* + **CUSTOMERS Table**

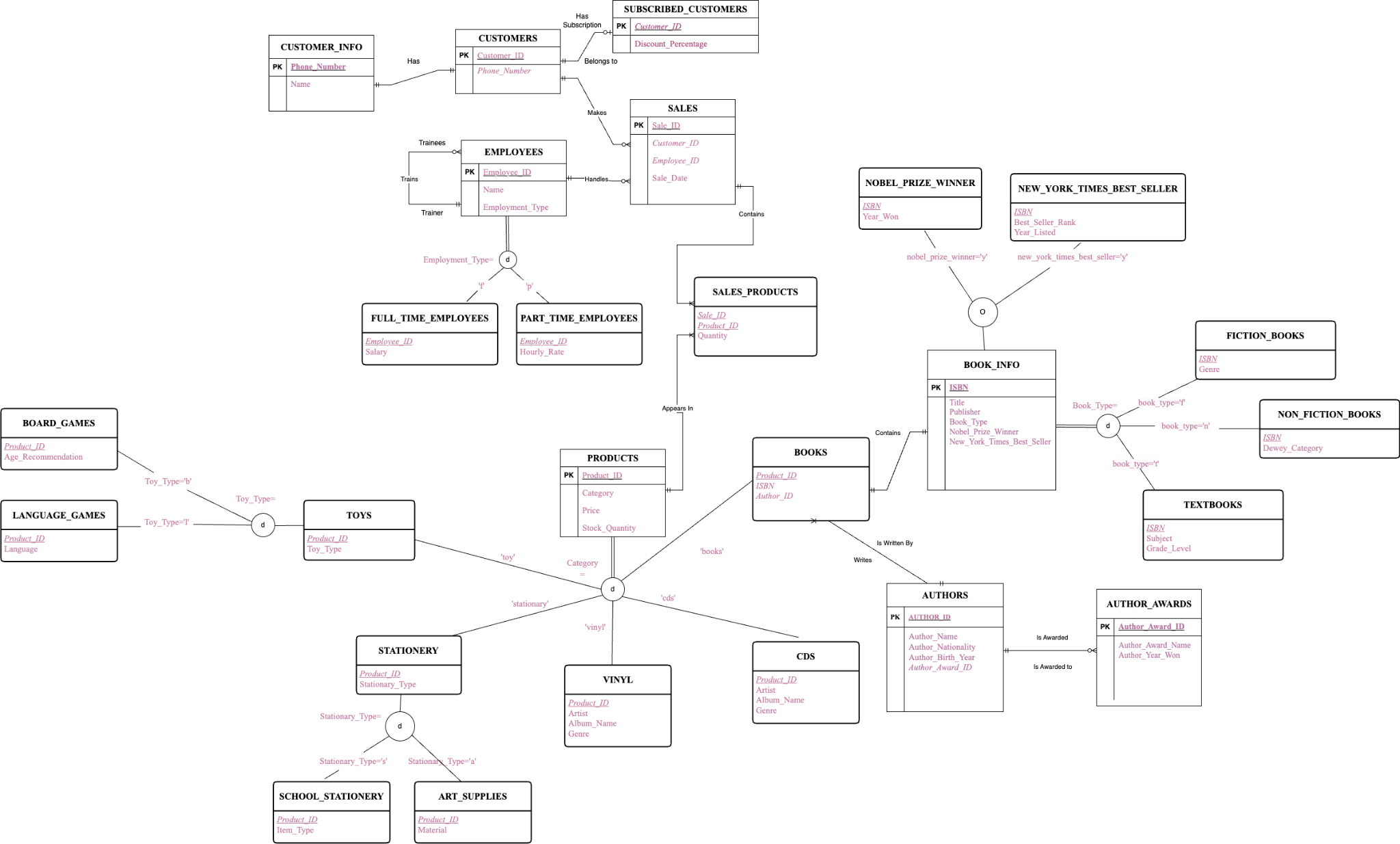
| **Initial Functional Dependencies** | **Transitioning to Third Normal Form** |
| --- | --- |
| * Customer\_ID → Name, Phone\_Number * Phone\_Number → Name *(Transitive Dependency)* | **CUSTOMERS(**Customer\_ID,*Phone\_Number***)**  **CUSTOMERS(**Phone\_Number**) mei CUSTOMERS\_INFO(**Phone\_Number**)**  **CUSTOMER\_INFO(**Phone\_Number,Name**)** |

* + **Finalized Functional Dependencies**

| **Table** | **Functional Dependency** |
| --- | --- |
| Customers | Customer\_ID → Phone\_Number |
| Customer\_Info | Phone\_Number → Name |
| Subscribed\_Customers | Customer\_ID → Discount\_Percentage |
| Employees | Employee\_ID → Name, Employment\_Type, Trainer\_ID |
| Full\_Time\_Employees | Employee\_ID → Salary |
| Part\_Time\_Employees | Employee\_ID → Hourly\_Rate |
| Sales | Sale\_ID → Customer\_ID, Employee\_ID, Sale\_Date |
| Products | Product\_ID → Category, Price, Stock\_Quantity |
| Sales\_Products | Sale\_ID, Product\_ID → Quantity |
| Books | Product\_ID → ISBN, Author\_ID |
| Books\_Info | ISBN → Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller |
| Authors | Author\_ID → Author\_Name, Author\_Nationality, Author\_Birth\_Year, Author\_Award\_ID |
| Author\_Awards | Author\_Award\_ID → Author\_Award\_Name, Author\_Year\_Won |
| Fiction\_Books | Product\_ID→ Genre |
| Non\_Fiction\_Books | Product\_ID → Dewey\_Category |
| Textbooks | Product\_ID → Subject, Grade-Level |
| Nobel\_Prize\_Winner | Product\_ID → Year\_Won |
| New\_York\_Times\_Best\_Seller | Product\_ID → Best\_Seller\_Rank, Year\_Listed |
| CDs | Product\_ID → Artist, Album\_Name, Genre |
| Vinyl | Product\_ID → Artist, Album\_Name, Genre |
| Toys | Product\_ID → Toy\_Type |
| Board\_Games | Product\_ID → Age\_Recommendations |
| Language\_Games | Product\_ID → Language |
| Stationery | Product\_ID → Stationery\_Type |
| School\_Stationery | Product\_ID → Item\_Type |
| Art\_Supplies | Product\_ID → Material |

As we can see, no partial or transitive dependencies exist in any of these tables, meaning they’re all in third normal form.

**Updated ER Diagram**

****

**Updated Transposition**

After completing the normalization and updating the ER diagram, new tables and relationships have formed, making it necessary to update some details in the transposition. The transposition of the unincluded entities remains the same.

* *New Relations Transposed*

**CUSTOMERS(**Customer\_ID,*Phone\_Number***)**

**CUSTOMERS(**Phone\_Number**) mei CUSTOMERS\_INFO(**Phone\_Number**)**

**CUSTOMER\_INFO(**Phone\_Number,Name**)**

**BOOKS(***Product\_ID*,*ISBN,Author\_ID***)**

**BOOKS(**Product\_ID**) mei PRODUCTS(**Product\_ID**)**

**BOOKS(**Author\_ID**) mei AUTHORS(**Author\_ID**)**

**BOOKS(**ISBN**) mei BOOKS\_INFO(**ISBN**)**

**BOOK\_INFO(**ISBN,Title, Publisher, Book\_Type, Nobel\_Prize\_Winner, New\_York\_Times\_Best\_Seller**)**

* *Foreign key was switched from Author\_Award\_ID referenced in Authors, to Author\_ID referenced in Author\_Awards due to the one-to-many relationship between the tables.*

**AUTHORS(**Author\_ID, Author\_Name, Author\_Nationality, Author\_Birth\_Year,**)**

**AUTHOR\_AWARDS(**Author\_Award\_ID, Author\_Award\_Name, Author\_Year\_Won, *Author\_ID***)**

**AUTHOR\_AWARDS(**Author\_ID**) mei AUTHOR(**Author\_ID)

* *Tables have become children of Book\_Info and so their foreign key has been changed to ISBN, the primary key of Book\_Info.*

**FICTION\_BOOKS(***ISBN,* Genre**)**

**FICTION\_BOOKS(**ISBN**) mei BOOK\_INFO(**ISBN**)**

**NON\_FICTION\_BOOKS(***ISBN*, Dewey\_Category**)**

**NON\_FICTION\_BOOKS(**ISBN**) mei BOOK\_INFO(**ISBN**)**

**TEXTBOOKS(***ISBN*, Subject, Grade\_Level**)**

**TEXTBOOKS(**ISBN**) mei BOOK\_INFO(**ISBN**)**

**NOBEL\_PRIZE\_WINNER(***ISBN*, Year\_Won**)**

**NOBEL\_PRIZE\_WINNER(**ISBN**) mei BOOK\_INFO(**ISBN**)**

**NEW\_YORK\_TIMES\_BEST\_SELLER(***ISBN*, Best\_Seller\_Rank, Year\_Listed**)**

**NEW\_YORK\_TIMES\_BEST\_SELLER(**ISBN**) mei BOOK\_INFO(**ISBN**)**

**Database Implementation**

1. **Data Definition Language - Defining the Tables**

| **Creating the Database** | CREATE DATABASE CozyBookStore;  USE CozyBookStore; |
| --- | --- |
| **Employees Table** | CREATE TABLE employees (  employee\_id INT PRIMARY KEY,  NAME VARCHAR(100) NOT NULL,  employment\_type VARCHAR(20) NOT NULL,  trainer\_id INT,  FOREIGN KEY (trainer\_id) REFERENCES employees(employee\_id),  CHECK(employment\_type IN('f','p'))); |
| **Customer\_Info Table** | CREATE TABLE customer\_info(  phone\_number BIGINT PRIMARY KEY,  NAME VARCHAR(100) NOT NULL); |
| **Customers Table** | CREATE TABLE customers(  customer\_id INT PRIMARY KEY,  phone\_number BIGINT,  FOREIGN KEY (phone\_number) REFERENCES customer\_info(phone\_number)); |
| **Subscribed\_Customers Table** | CREATE TABLE subscribed\_customers(  customer\_id INT PRIMARY KEY,  discount\_percentage DECIMAL(2,2) NOT NULL,  FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id)); |
| **Products Table** | CREATE TABLE products(  product\_id INT PRIMARY KEY,  price DECIMAL(5,2) NOT NULL,  stock\_quantity INT NOT NULL DEFAULT 0,  category VARCHAR(50) NOT NULL,  CHECK (category IN("toy","stationery","vinyl","cds","books"))); |
| **Sales Table** | CREATE TABLE sales(  sale\_id INT PRIMARY KEY,  customer\_id INT,  employee\_id INT,  sale\_date DATE NOT null,  FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id),  FOREIGN KEY (employee\_id) REFERENCES employees(employee\_id)); |
| **Sales\_Products Table** | CREATE TABLE sales\_products(  sale\_id INT,  product\_id INT,  quantity INT NOT NULL,  PRIMARY KEY(sale\_id,product\_id),  FOREIGN KEY (sale\_id) REFERENCES sales(sale\_id),  FOREIGN KEY (product\_id) REFERENCES products(product\_id)); |
| **Full\_Time\_Employee Table** | CREATE TABLE full\_time\_employee(  employee\_id INT PRIMARY KEY,  salary BIGINT NOT NULL,  FOREIGN KEY (employee\_id) REFERENCES employees(employee\_id)); |
| **Part\_Time\_Employee Table** | CREATE TABLE part\_time\_employee(  employee\_id INT PRIMARY KEY,  hourly\_rate INT NOT NULL,  FOREIGN KEY(employee\_id) REFERENCES employees(employee\_id)); |
| **Toys Table** | CREATE TABLE toys(  product\_id int PRIMARY KEY,  toy\_type VARCHAR(20),  CHECK (toy\_type IN ("b","l")),  FOREIGN KEY(product\_id) REFERENCES products(product\_id)); |
| **Language\_Games Table** | CREATE TABLE language\_games(  product\_id INT PRIMARY KEY,  LANGUAGE VARCHAR(20) NOT NULL,  FOREIGN KEY(product\_id) REFERENCES toys(product\_id)); |
| **Board\_Games Table** | CREATE TABLE board\_games(  product\_id INT PRIMARY KEY,  age\_recommendation VARCHAR(20) NOT NULL,  FOREIGN KEY (product\_id) REFERENCES toys(product\_id),  CHECK (age\_recommendation IN ('3+', '5+', '7+', '10+', '12+', '16+', '18+', '7-12', '10-14'))); |
| **Stationery Table** | CREATE TABLE stationery(  product\_id INT PRIMARY KEY,  stationery\_type VARCHAR(20),  FOREIGN KEY (product\_id) REFERENCES products(product\_id),  CHECK (stationery\_type IN("s","a"))); |
| **School\_Stationery Table** | CREATE TABLE school\_stationery(  product\_id INT PRIMARY KEY,  item\_type VARCHAR(100) NOT NULL,  FOREIGN KEY (product\_id) REFERENCES stationery(product\_id)); |
| **Art\_Supplies Table** | CREATE TABLE art\_supplies(  product\_id INT PRIMARY KEY,  material VARCHAR(100) NOT NULL,  FOREIGN KEY (product\_id) REFERENCES stationery(product\_id)); |
| **Vinyl Table** | CREATE TABLE vinyl(  product\_id INT PRIMARY KEY,  artist VARCHAR(100) NOT NULL,  album\_name VARCHAR(100) NOT NULL,  genre VARCHAR(50) NOT NULL,  FOREIGN KEY (product\_id) REFERENCES products(product\_id)); |
| **Cds Table** | CREATE TABLE cds(  product\_id INT PRIMARY KEY,  artist VARCHAR(100) NOT NULL,  album\_name VARCHAR(100) NOT NULL,  genre VARCHAR(50) NOT NULL,  FOREIGN KEY (product\_id) REFERENCES products(product\_id)); |
| **Book\_Info Table** | CREATE TABLE book\_info(  isbn CHAR(13) PRIMARY KEY,  title VARCHAR(100) NOT NULL,  publisher VARCHAR(100) NOT NULL,  book\_type VARCHAR(20) NOT NULL,  nobel\_prize\_winner VARCHAR(20) NOT NULL,  new\_york\_times\_best\_seller VARCHAR(20) NOT NULL,  CHECK (book\_type IN("f","n","t")),  CHECK (nobel\_prize\_winner IN("y","n")),  CHECK (new\_york\_times\_best\_seller IN ("y","n"))); |
| **Author\_Awards Table** | CREATE TABLE author\_awards(  author\_award\_id INT PRIMARY KEY,  author\_award\_name VARCHAR(100) NOT NULL,  author\_year\_won YEAR NOT NULL); |
| **Authors Table** | CREATE TABLE authors(  author\_id INT PRIMARY KEY,  author\_name VARCHAR(100) NOT NULL,  author\_nationality VARCHAR(100) NOT NULL,  author\_birth\_year YEAR NOT NULL,  author\_award\_id INT,  FOREIGN KEY (author\_award\_id) REFERENCES author\_awards(author\_award\_id)); |
| **Books Table** | CREATE TABLE books(  product\_id int PRIMARY KEY,  isbn CHAR(13),  author\_id INT,  FOREIGN KEY (product\_id) REFERENCES products(product\_id),  FOREIGN KEY (isbn) REFERENCES book\_info(isbn),  FOREIGN KEY (author\_id) REFERENCES authors(author\_id)); |
| **Textbooks Table** | CREATE TABLE textbooks(  isbn CHAR(13) PRIMARY KEY,  SUBJECT VARCHAR(20) NOT NULL,  grade\_level VARCHAR(20) NOT NULL,  FOREIGN KEY (isbn) REFERENCES book\_info(isbn)); |
| **Non\_Fiction\_Books Table** | CREATE TABLE non\_fiction\_books(  isbn CHAR(13) PRIMARY KEY,  dewey\_category VARCHAR(10) NOT NULL,  FOREIGN KEY (isbn) REFERENCES book\_info(isbn)); |
| **Fiction\_Books** | CREATE TABLE fiction\_books(  isbn CHAR(13) PRIMARY KEY,  genre VARCHAR(50) NOT NULL,  FOREIGN KEY (isbn) REFERENCES book\_info(isbn)); |
| **Nobel\_Prize\_Winner Table** | CREATE TABLE nobel\_prize\_winner(  isbn CHAR(13) PRIMARY KEY,  year\_won YEAR NOT NULL,  FOREIGN KEY (isbn) REFERENCES book\_info(isbn)); |
| **New\_York\_Times\_Best\_Seller Table** | CREATE TABLE new\_york\_times\_best\_seller(  isbn CHAR(13) PRIMARY KEY,  best\_seller\_rank INT NOT NULL,  year\_won YEAR NOT NULL,  FOREIGN KEY (isbn) REFERENCES book\_info(isbn)); |
| **Updating Stock\_Quantity Automatically** | DELIMITER //  CREATE TRIGGER reduce\_stock\_quantity  BEFORE INSERT ON sales\_products  FOR EACH ROW  BEGIN  UPDATE products  SET stock\_quantity = stock\_quantity - NEW.quantity  WHERE product\_id = NEW.product\_id;  END;  //  DELIMITER ; |

1. **Data Manipulation Language - Inserting Tables with Data**

| **Employees Table** | INSERT INTO employees (employee\_id, name, employment\_type, trainer\_id)  VALUES  (1, 'Alice Johnson', 'f', NULL),  (2, 'Bob Smith', 'p', 1),  (3, 'Clara Lee', 'f', 1),  (4, 'John Carter', 'p', NULL): |
| --- | --- |
| **Customer\_Info Table** | INSERT INTO customer\_info (phone\_number, name)  VALUES  (9876543210, 'Emily Brown'),  (8765432190, 'David Green'),  (2004522801, 'Juliette Dubois'),  (7467920283, 'Sophie Boucher'); |
| **Customers Table** | INSERT INTO customers (customer\_id, phone\_number)  VALUES  (101, 9876543210),  (102, 8765432190),  (103, 2004522801),  (104, 7467920283); |
| **Subscribed\_Customers Table** | INSERT INTO subscribed\_customers (customer\_id, discount\_percentage)  VALUES  (101, 0.15),  (102, 0.10); |
| **Products Table** | INSERT INTO products (product\_id, price, stock\_quantity, category) VALUES  (101, 14.99, 50, 'toy'),  (102, 12.49, 60, 'toy'),  (201, 3.99, 100, 'stationery'),  (202, 7.99, 80, 'stationery'),  (301, 22.99, 25, 'vinyl'),  (302, 24.50, 20, 'vinyl'),  (303, 21.75, 18, 'vinyl'),  (304, 23.00, 15, 'vinyl'),  (305, 25.00, 12, 'vinyl'),  (311, 11.99, 30, 'cds'),  (312, 12.49, 35, 'cds'),  (313, 13.99, 40, 'cds'),  (314, 14.50, 28, 'cds'),  (315, 12.00, 33, 'cds'),  (401, 15.99, 45, 'books'),  (402, 16.99, 40, 'books'),  (403, 18.50, 35, 'books'),  (404, 19.99, 30, 'books'),  (405, 13.75, 25, 'books'),  (406, 12.50, 38, 'books'),  (407, 17.25, 32, 'books'),  (408, 14.99, 34, 'books'),  (409, 21.00, 28, 'books'),  (410, 15.49, 36, 'books'); |
| **Sales Table** | INSERT INTO sales (sale\_id, customer\_id, employee\_id, sale\_date)  VALUES  (301, 101, 2, '2025-04-10'),  (302, 102, 3, '2025-04-12'); |
| **Sales\_Products Table** | INSERT INTO sales\_products (sale\_id, product\_id, quantity)  VALUES  (301, 201, 2),  (301, 202, 1); |
| **Full\_Time\_Employee Table** | INSERT INTO full\_time\_employee (employee\_id, salary)  VALUES  (1, 70000),  (3, 68000); |
| **Part\_Time\_Employee Table** | INSERT INTO part\_time\_employee (employee\_id, hourly\_rate)  VALUES  (2, 20),  (4, 30); |
| **Toys Table** | INSERT INTO toys (product\_id, toy\_type) VALUES  (101, 'b'),  (102, 'l'); |
| **Language\_Games Table** | INSERT INTO language\_games (product\_id, language)  VALUES  (102, 'French'); |
| **Board\_Games Table** | INSERT INTO board\_games (product\_id, age\_recommendation)  VALUES  (101, '7+'); |
| **Stationery Table** | INSERT INTO stationery (product\_id, stationery\_type) VALUES  (201, 's'),  (202, 'a'); |
| **School\_Stationery Table** | INSERT INTO school\_stationery (product\_id, item\_type) VALUES (201, 'Notebook'); |
| **Art\_Supplies Table** | INSERT INTO art\_supplies (product\_id, material) VALUES (202, 'Watercolors'); |
| **Vinyl Table** | INSERT INTO vinyl (product\_id, artist, album\_name, genre) VALUES  (301, 'The Beatles', 'Abbey Road', 'Rock'),  (302, 'Pink Floyd', 'The Dark Side of the Moon', 'Progressive Rock'),  (303, 'Fleetwood Mac', 'Rumours', 'Pop Rock'),  (304, 'Nirvana', 'Nevermind', 'Grunge'),  (305, 'David Bowie', 'The Rise and Fall of Ziggy Stardust', 'Glam Rock'); |
| **Cds Table** | INSERT INTO cds (product\_id, artist, album\_name, genre) VALUES  (311, 'Adele', '21', 'Pop'),  (312, 'Taylor Swift', '1989', 'Pop'),  (313, 'Ed Sheeran', 'Divide', 'Pop'),  (314, 'Imagine Dragons', 'Evolve', 'Alternative Rock'),  (315, 'Bruno Mars', '24K Magic', 'R&B'); |
| **Book\_Info Table** | INSERT INTO book\_info (isbn, title, publisher, book\_type, nobel\_prize\_winner, new\_york\_times\_best\_seller) VALUES  ('9780141182636', '1984', 'Penguin Classics', 'f', 'n', 'y'),  ('9780439139601', 'Harry Potter and the Goblet of Fire', 'Scholastic', 'f', 'n', 'y'),  ('9780062316097', 'Sapiens: A Brief History of Humankind', 'Harper', 'n', 'n', 'y'),  ('9780131103627', 'The C Programming Language', 'Prentice Hall', 't', 'n', 'n'),  ('9780385490818', 'The Alchemist', 'HarperOne', 'f', 'n', 'y'),  ('9780143039433', 'The Diary of a Young Girl', 'Penguin Books', 'n', 'y', 'y'),  ('9780525555370', 'Becoming', 'Crown Publishing Group', 'n', 'n', 'y'),  ('9780747532743', 'Harry Potter and the Philosopher\'s Stone', 'Bloomsbury', 'f', 'n', 'y'),  ('9780262033848', 'Introduction to Algorithms', 'MIT Press', 't', 'n', 'n'),  ('9780307279460', 'The Omnivore\'s Dilemma', 'Penguin Press', 'n', 'n', 'y'); |
| **Author\_Awards Table** | INSERT INTO author\_awards (author\_award\_id, author\_award\_name, author\_year\_won) VALUES  (1, 'Pulitzer Prize', 1994),  (2, 'Man Booker Prize', 2005),  (3, 'Hugo Award', 1984),  (4, 'National Book Award', 2018),  (5, 'Nobel Prize in Literature', 1949); |
| **Authors Table** | INSERT INTO authors (author\_id, author\_name, author\_nationality, author\_birth\_year, author\_award\_id) VALUES  (1, 'George Orwell', 'British', 1903, 5),  (2, 'J.K. Rowling', 'British', 1965, 2),  (3, 'Yuval Noah Harari', 'Palestinian', 1976, NULL),  (4, 'Brian Kernighan', 'Canadian', 1942, 1),  (5, 'Paulo Coelho', 'Brazilian', 1947, NULL),  (6, 'Anne Frank', 'German-Dutch', 1929, NULL),  (7, 'Michelle Obama', 'American', 1964, 4),  (8, 'Thomas H. Cormen', 'American', 1956, NULL),  (9, 'Michael Pollan', 'American', 1955, NULL); |
| **Books Table** | INSERT INTO books (product\_id, isbn, author\_id) VALUES  (401, '9780141182636', 1),  (402, '9780439139601', 2),  (403, '9780062316097', 3),  (404, '9780131103627', 4),  (405, '9780385490818', 5),  (406, '9780143039433', 6),  (407, '9780525555370', 7),  (408, '9780747532743', 2),  (409, '9780262033848', 8),  (410, '9780307279460', 9); |
| **Textbooks Table** | INSERT INTO textbooks (isbn, subject, grade\_level) VALUES  ('9780131103627', 'Computer Science', 'University'),  ('9780262033848', 'Algorithms', 'University'); |
| **Non\_Fiction\_Books Table** | INSERT INTO non\_fiction\_books (isbn, dewey\_category) VALUES  ('9780062316097', '909'),  ('9780143039433', '940.53'),  ('9780525555370', '921'),  ('9780307279460', '394'); |
| **Fiction\_Books** | INSERT INTO fiction\_books (isbn, genre) VALUES  ('9780141182636', 'Dystopian'),  ('9780439139601', 'Fantasy'),  ('9780385490818', 'Philosophical Fiction'),  ('9780747532743', 'Fantasy'); |
| **Nobel\_Prize\_Winner Table** | INSERT INTO nobel\_prize\_winner (isbn, year\_won) VALUES  ('9780143039433', 1947); |
| **New\_York\_Times\_Best\_Seller Table** | INSERT INTO new\_york\_times\_best\_seller (isbn, best\_seller\_rank, year\_won) VALUES  ('9780141182636', 4, 1950),  ('9780439139601', 1, 2000),  ('9780062316097', 3, 2015),  ('9780385490818', 2, 2001),  ('9780143039433', 5, 1952),  ('9780525555370', 1, 2018),  ('9780747532743', 1, 1997),  ('9780307279460', 6, 2006); |

**Relational Algebra Statements**

π Projection (select columns)

σ Selection (filter rows)

⨝ Join (combine tables)

Business Scenario 2.1: Membership subscription and discount

* Find the names and phone numbers of subscribed customers who are eligible for a discount.

π name, phone\_number (

( σ subscribed\_customers.discount\_percentage > 0) (

subscribed\_customers ⨝ subscribed\_customers.customer\_id = customers.customer\_id customers

⨝ customers.phone\_number = customer\_info.phone\_number customer\_info

)

)

Business Scenario 2.2: Sales Processing

* Get product IDs, names of employees, and sale dates for all products sold by employees on a given day (e.g., '2025-04-10').

π product\_id, name, sale\_date (

(σ sales.sale\_date = '2025-04-10') (

sales\_products ⨝ sales\_products.sale\_id = sales.sale\_id sales

⨝ sales.employee\_id = employees.employee\_id employees

)

)

**Composing SQL Queries**

**Query 1:**

We will compose an SQL query that deletes customers who are not subscribed to the membership program. Before performing the deletion, a temporary backup of unsubscribed customer data is created. Following the deletion, a union operation is used to display a combined list containing the names of both the deleted (unsubscribed) customers and the remaining subscribed customers.

1. **Creating a temporary table to store unsubscribed customers before deleting.**

CREATE TEMPORARY TABLE unsubscribed\_backup AS

SELECT c.customer\_id, ci.name FROM customers c

JOIN customer\_info ci ON c.phone\_number = ci.phone\_number

WHERE c.customer\_id NOT IN (SELECT customer\_id FROM subscribed\_customers);

1. **Now, we will delete those unsubscribed customers.**

DELETE FROM customers

WHERE customer\_id NOT IN (SELECT customer\_id FROM subscribed\_customers);

1. **Use union to combine unsubscribed and subscribed customer names**

SELECT name AS 'Customer Name', 'Unsubscribed' AS Status

FROM unsubscribed\_backup

UNION

SELECT ci.name AS 'Customer Name', 'Subscribed' AS Status

FROM customers c

JOIN customer\_info ci ON c.phone\_number = ci.phone\_number

JOIN subscribed\_customers sc ON c.customer\_id = sc.customer\_id;

**Query 2:**

We have a new part time employee called John at the store. He was not assigned any trainer and thus made no sales contributions. Here we will search for this new part time employee in the system and assign him our most experienced trainer Alice in order for him to make his first sale.

UPDATE employees

SET trainer\_id = 1

WHERE employment\_type = 'p'

AND NAME LIKE '%John%'

AND trainer\_id IS NULL

AND employee\_id IN (SELECT employee\_id FROM employees)

AND NOT EXISTS (

SELECT 1

FROM sales s

WHERE s.employee\_id = employees.employee\_id

);

**Query 3:**

The book store employees have noticed that certain fantasy books are always selling quickly, especially those that are new york time best sellers. This query will adjust and update the prices of those books by an increase of 15% to adjust our business profits.

UPDATE products

SET price = price \* 1.15

WHERE product\_id IN (

SELECT b.product\_id

FROM books b

JOIN book\_info bi ON b.isbn = bi.isbn

JOIN fiction\_books fb ON bi.isbn = fb.isbn

JOIN new\_york\_times\_best\_seller ny ON bi.isbn = ny.isbn

WHERE fb.genre LIKE '%fantasy%'

);