Bookstore Database Project phase 1 Report

Project Title:

- Bookstore Database

Course:

- CSC380

Team Members:

- مهند الشهراني - 442100744
- Yasir Alfaifi - 444100878
- سلمان ال الشيخ - 444105684

---

2. Table of Contents

---

3. Introduction

This project focuses on designing and implementing a bookstore database management system. The database handles various entities such as **Books**, **Categories**, **Customers**, **Orders**, and **Payment Information**. It provides a systematic way to organize and manage books, customer orders, and payments for an online bookstore.

---

4. Project Description

This section includes a detailed explanation of each entity and the relationships:

Book:
The **Book** entity stores information about the books available in the bookstore, including attributes like Book_id, title, author, isbn, and price. It has a **many-to-one relationship** with **Category** and a **many-to-many relationship** with **Order** (through the **Books_in_Order** relationship).

Category:
The **Category** entity categorizes books by genres. Each category can have multiple books, establishing a **one-to-many relationship** with **Book**.

Customer:
The **Customer** entity represents individuals purchasing from the bookstore. A **Customer** can place multiple orders, creating a **one-to-many relationship** with **Order**.

Order:
The **Order** entity tracks customer purchases. An order can include multiple books (**many-to-many relationship** with **Book**), and each order has a single customer (**one-to-many relationship** with **Customer**). It also has a **one-to-one relationship** with **Payment_Info**.

Payment_Info:
This entity stores payment details associated with each order, establishing a **weak one-to-one relationship** with **Order**.

Books_in_Order:
This is a junction entity that represents the **many-to-many relationship** between **Order** and **Book**. It tracks the quantity of each book in an order.

Summary of Relationships:

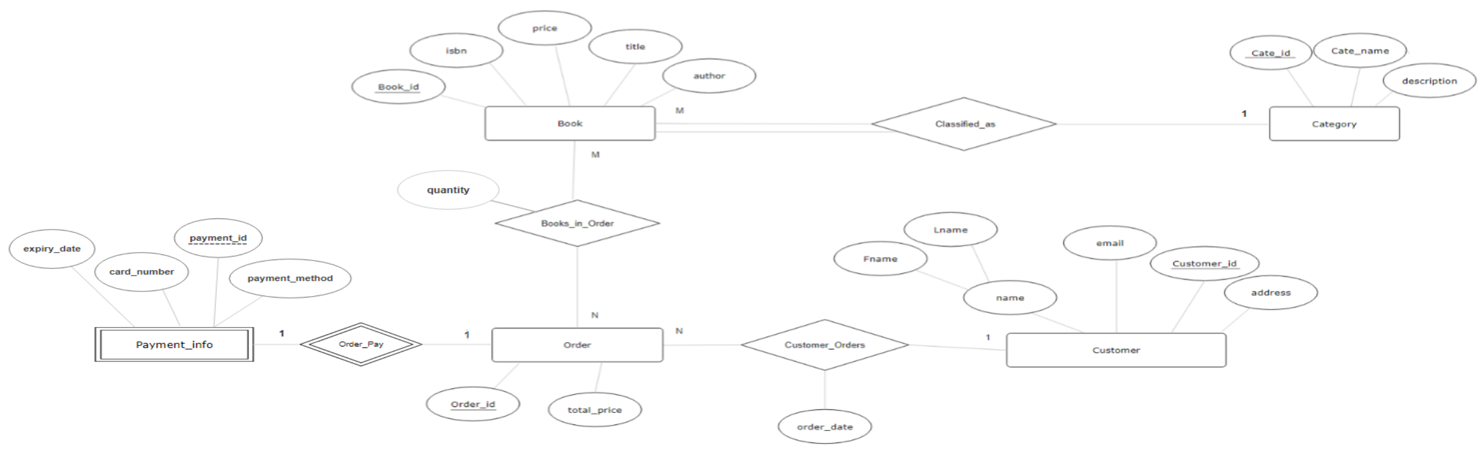
- **Book ↔ Category**: Many-to-One
- **Order ↔ Book**: Many-to-Many

- **Order ↔ Customer**: Many-to-One
- **Order ↔ Payment_Info**: One-to-One

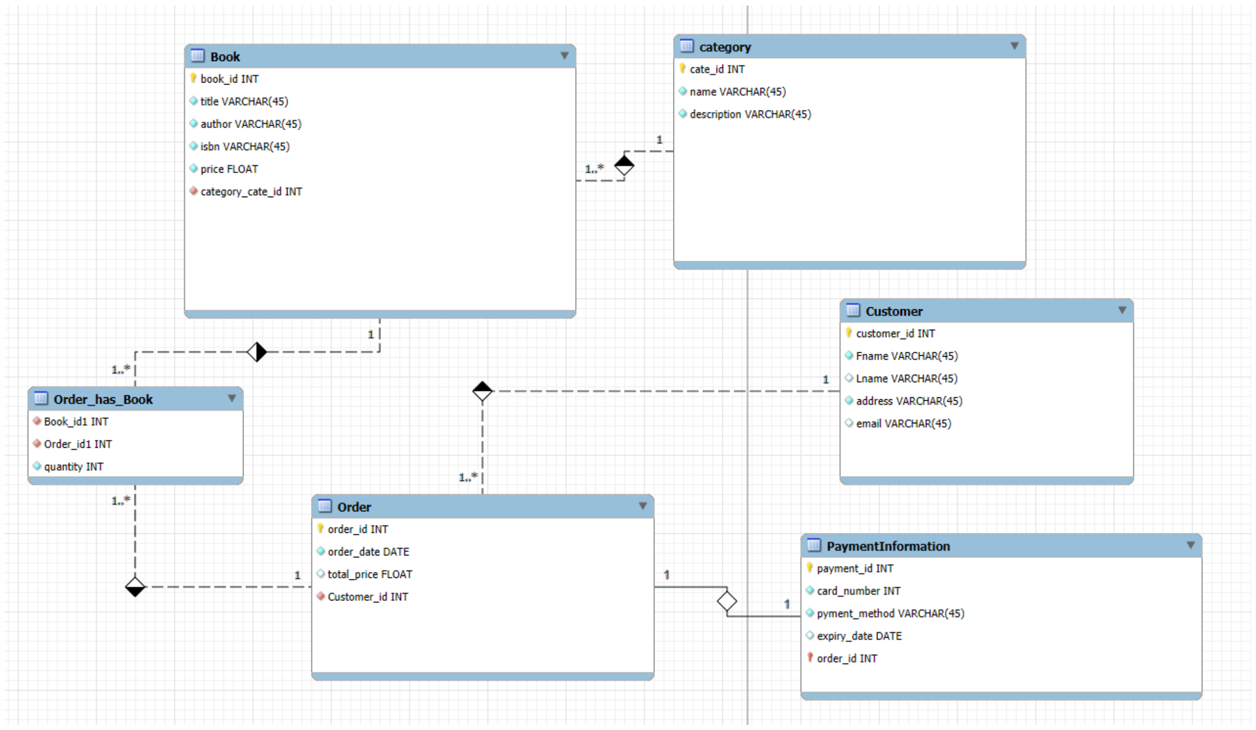
---

5. ER Diagram

Here is the **ER diagram** used in the project:

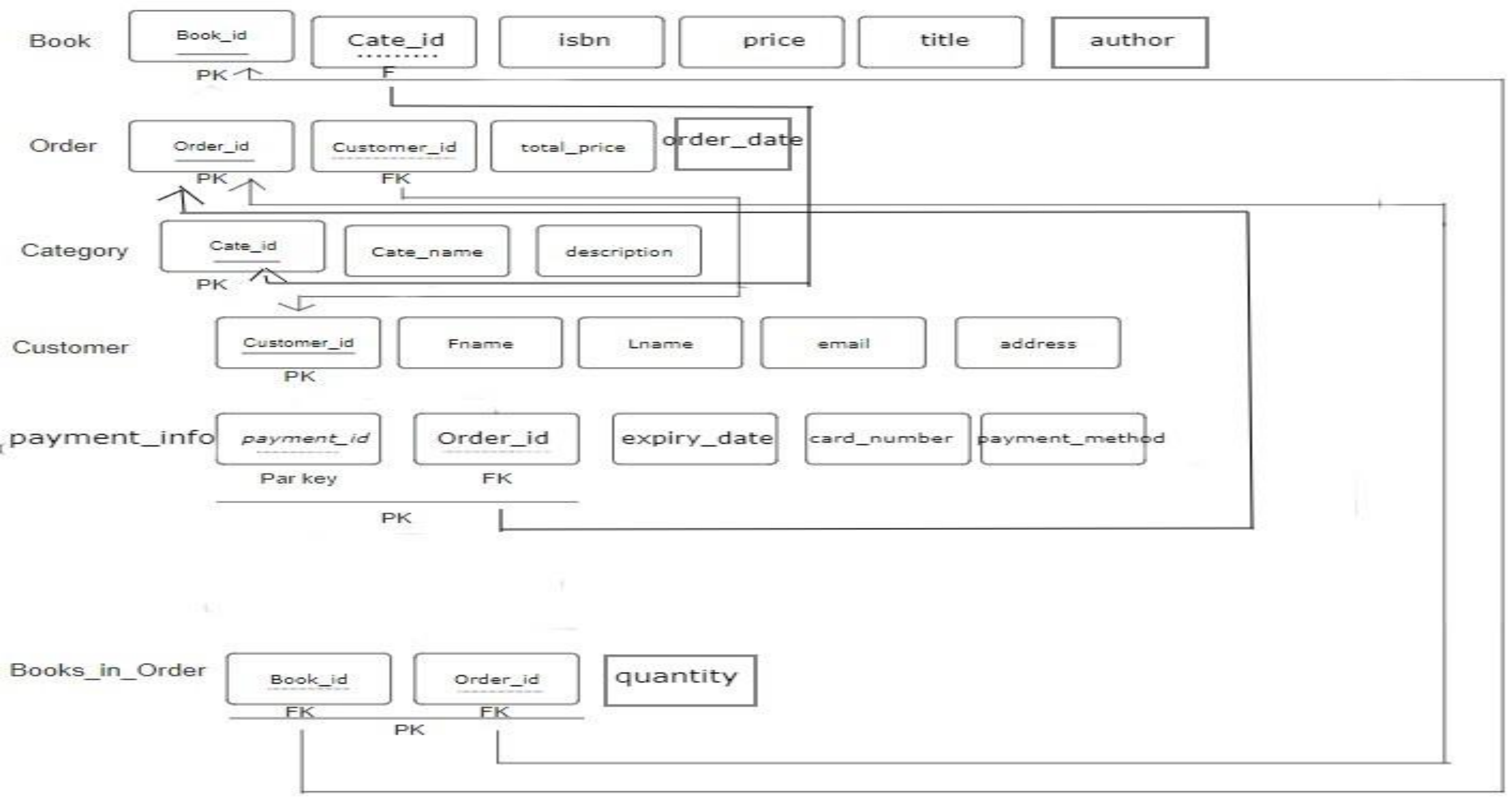


Using sql workbench:



6. Relational Schema

The relational schema was mapped from the ER diagram, and the structure includes the following tables:



Using sql workbench:

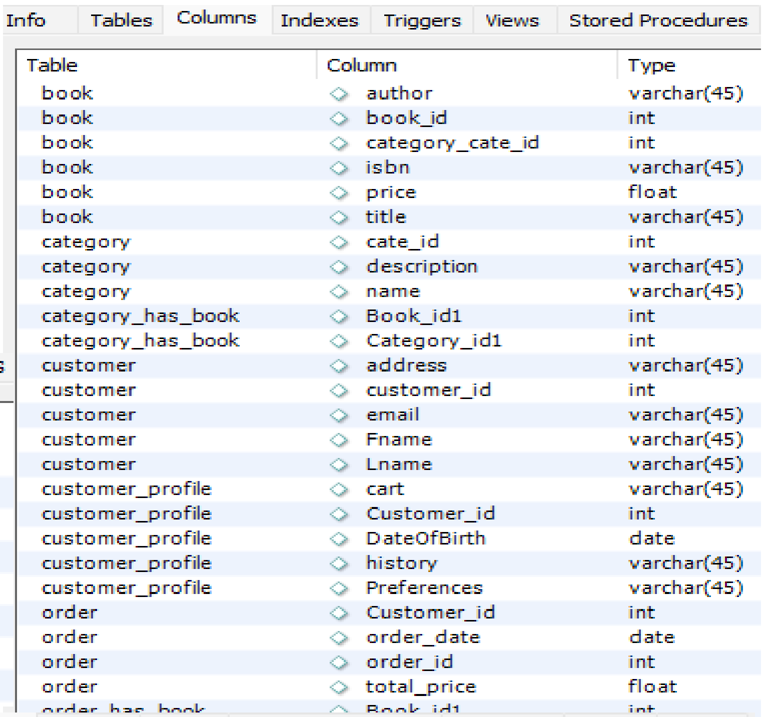
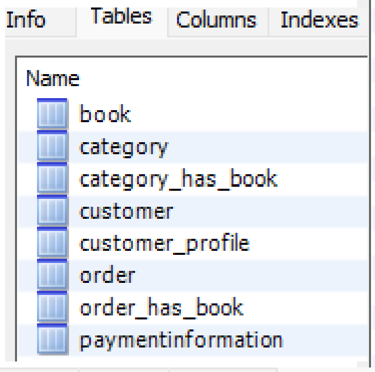


Table	Column	Type
book	author	varchar(45)
book	book_id	int
book	category_cate_id	int
book	isbn	varchar(45)
book	price	float
book	title	varchar(45)
category	cate_id	int
category	description	varchar(45)
category	name	varchar(45)
category_has_book	Book_id1	int
category_has_book	Category_id1	int
customer	address	varchar(45)
customer	customer_id	int
customer	email	varchar(45)
customer	Fname	varchar(45)
customer	Lname	varchar(45)
customer_profile	cart	varchar(45)
customer_profile	Customer_id	int
customer_profile	DateOfBirth	date
customer_profile	history	varchar(45)
customer_profile	Preferences	varchar(45)
order	Customer_id	int
order	order_date	date
order	order_id	int
order	total_price	float
order_has_book	Book_id1	int



Name
book
category
category_has_book
customer
customer_profile
order
order_has_book
paymentinformation

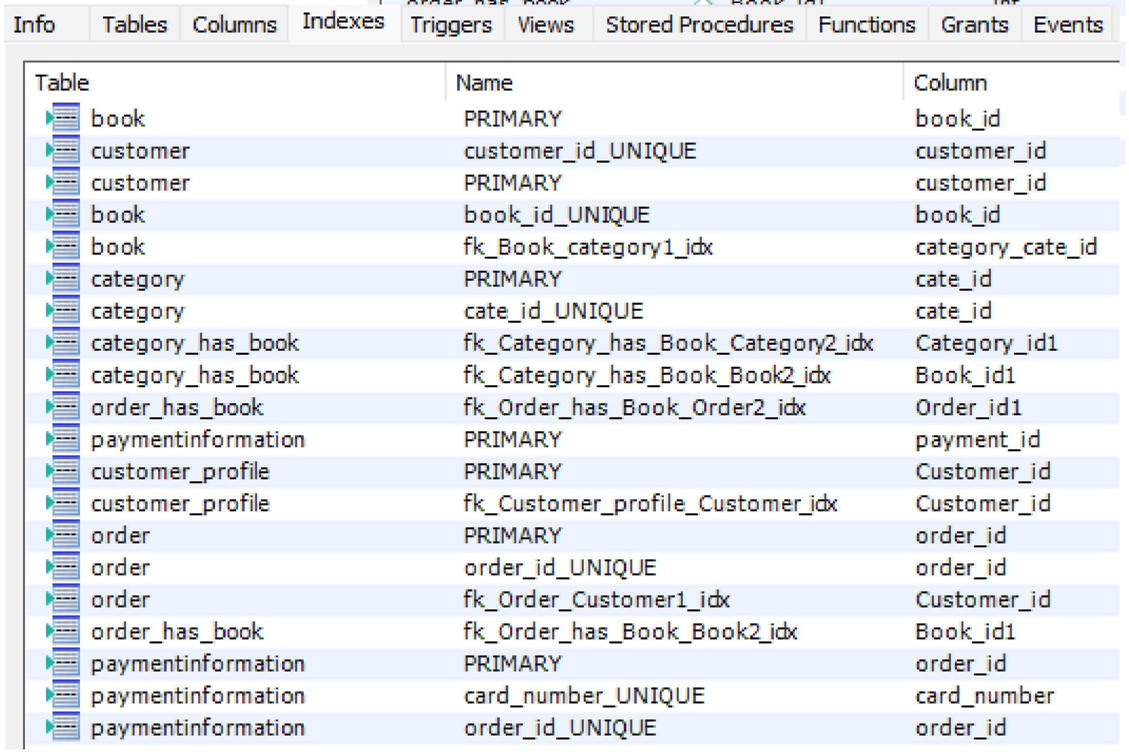


Table	Name	Column
book	PRIMARY	book_id
customer	customer_id_UNIQUE	customer_id
customer	PRIMARY	customer_id
book	book_id_UNIQUE	book_id
book	fk_Book_category1_idx	category_cate_id
category	PRIMARY	cate_id
category	cate_id_UNIQUE	cate_id
category_has_book	fk_Category_has_Book_Category2_idx	Category_id1
category_has_book	fk_Category_has_Book_Book2_idx	Book_id1
order_has_book	fk_Order_has_Book_Order2_idx	Order_id1
paymentinformation	PRIMARY	payment_id
customer_profile	PRIMARY	Customer_id
customer_profile	fk_Customer_profile_Customer_idx	Customer_id
order	PRIMARY	order_id
order	order_id_UNIQUE	order_id
order	fk_Order_Customer1_idx	Customer_id
order_has_book	fk_Order_has_Book_Book2_idx	Book_id1
paymentinformation	PRIMARY	order_id
paymentinformation	card_number_UNIQUE	card_number
paymentinformation	order_id_UNIQUE	order_id

7. SQL Script:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;

SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;

SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISIO
N_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------

-- Drop Tables (If Exists)

-- -----------------------------------------------------

DROP TABLE IF EXISTS Category;

DROP TABLE IF EXISTS Book;

DROP TABLE IF EXISTS Customer;

DROP TABLE IF EXISTS `Order`;

DROP TABLE IF EXISTS PaymentInformation;

DROP TABLE IF EXISTS Order_has_Book;

-- -----------------------------------------------------

-- Schema mydb

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8;

USE `mydb`;

-- -----------------------------------------------------

-- Table `category`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `mydb`.`category` (

  `cate_id` INT NOT NULL,

  `name` VARCHAR(45) NOT NULL,

  `description` VARCHAR(45) NOT NULL,

  PRIMARY KEY (`cate_id`),

  UNIQUE INDEX `cate_id_UNIQUE` (`cate_id` ASC) VISIBLE)

ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `Book`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Book` (
  `book_id` INT NOT NULL,
  `title` VARCHAR(45) NOT NULL,
  `author` VARCHAR(45) NOT NULL,
  `isbn` VARCHAR(45) NOT NULL,
  `price` FLOAT NOT NULL,
  `category_cate_id` INT NOT NULL,
  PRIMARY KEY (`book_id`),
  UNIQUE INDEX `book_id_UNIQUE` (`book_id` ASC) VISIBLE,
  INDEX `fk_Book_category1_idx` (`category_cate_id` ASC) VISIBLE,
  CONSTRAINT `fk_Book_category1`
    FOREIGN KEY (`category_cate_id`)
    REFERENCES `mydb`.`category` (`cate_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `Customer`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Customer` (
  `customer_id` INT NOT NULL,
  `Fname` VARCHAR(45) NOT NULL,
  `Lname` VARCHAR(45) NULL,
  `address` VARCHAR(45) NOT NULL,
  `email` VARCHAR(45) NULL,
  PRIMARY KEY (`customer_id`),
  UNIQUE INDEX `customer_id_UNIQUE` (`customer_id` ASC) VISIBLE)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `Order`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Order` (
  `order_id` INT NOT NULL,
  `order_date` DATE NOT NULL,
  `total_price` FLOAT NULL,
  `Customer_id` INT NOT NULL,
  PRIMARY KEY (`order_id`),
  INDEX `fk_Order_Customer1_idx` (`Customer_id` ASC) VISIBLE,
  UNIQUE INDEX `order_id_UNIQUE` (`order_id` ASC) VISIBLE,
  CONSTRAINT `fk_Order_Customer1`
    FOREIGN KEY (`Customer_id`)
    REFERENCES `mydb`.`Customer` (`customer_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `Order_has_Book`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`Order_has_Book` (
  `Book_id1` INT NOT NULL,
  `Order_id1` INT NOT NULL,
  `quantity` INT NOT NULL,
  INDEX `fk_Order_has_Book_Book2_idx` (`Book_id1` ASC) VISIBLE,
  INDEX `fk_Order_has_Book_Order2_idx` (`Order_id1` ASC) VISIBLE,
  CONSTRAINT `fk_Order_has_Book_Book2`
    FOREIGN KEY (`Book_id1`)
    REFERENCES `mydb`.`Book` (`book_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Order_has_Book_Order2`
    FOREIGN KEY (`Order_id1`)
    REFERENCES `mydb`.`Order` (`order_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
-- -----------------------------------------------------
-- Table `PaymentInformation`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `mydb`.`PaymentInformation` (
  `payment_id` INT NOT NULL,
  `card_number` INT NOT NULL,
  `payment_method` VARCHAR(45) NOT NULL,
  `expiry_date` DATE NULL,
  `order_id` INT NOT NULL,
  PRIMARY KEY (`order_id`, `payment_id`),
  UNIQUE INDEX `card_number_UNIQUE` (`card_number` ASC) VISIBLE,
  UNIQUE INDEX `order_id_UNIQUE` (`order_id` ASC) VISIBLE,
  CONSTRAINT `payment_id`
    FOREIGN KEY (`order_id`)
    REFERENCES `mydb`.`Order` (`order_id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;

-- Reset SQL Modes
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```