**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**

1. Introduction
2. Project Description
3. ER Diagram
4. Relational Schema
5. SQL Scripts
6. Description of the Environment
7. User Interface and Queries
8. Conclusion

**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**:

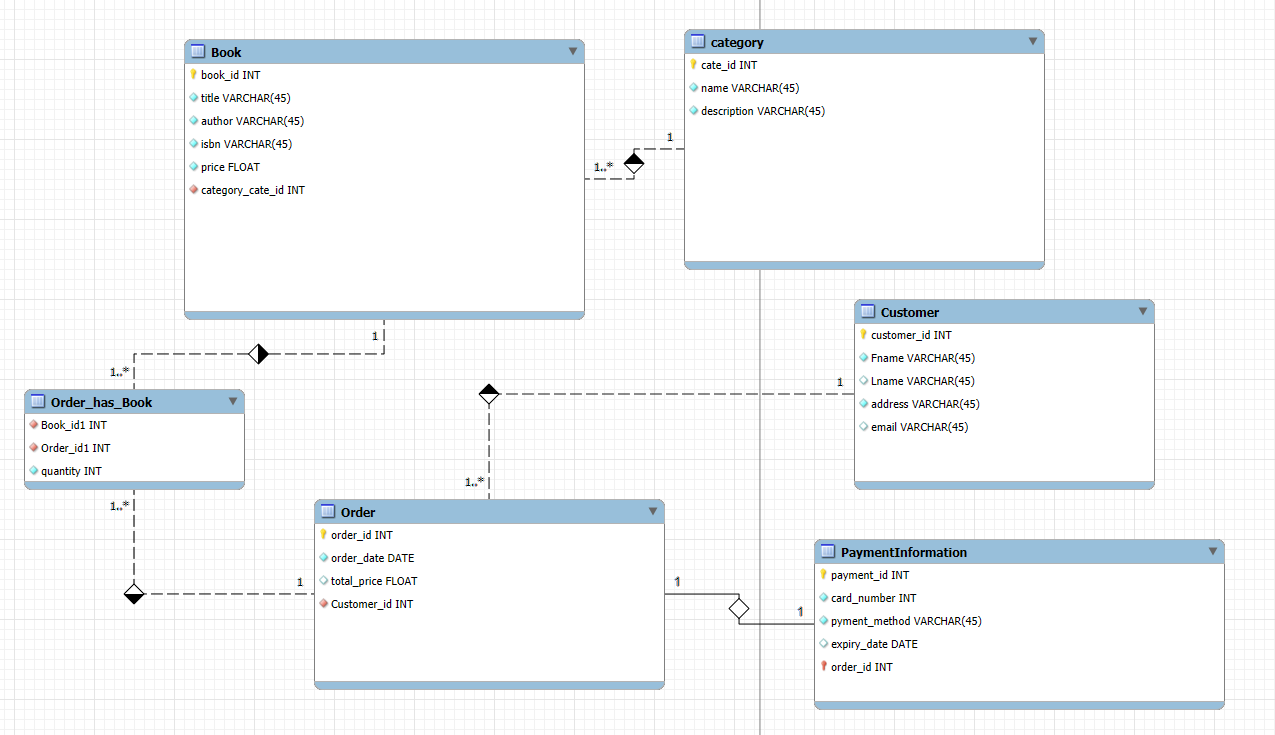
* **Book ↔ Category**: Many-to-One
* **Order ↔ Book**: Many-to-Many
* **Order ↔ Customer**: Many-to-One
* **Order ↔ Payment\_Info**: One-to-One

A screenshot of a computer

Description automatically generated**5. ER Diagram**

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

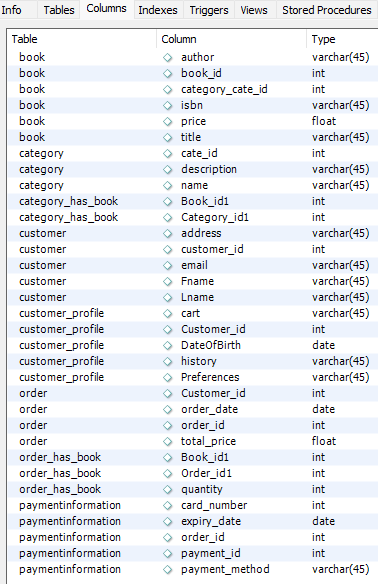
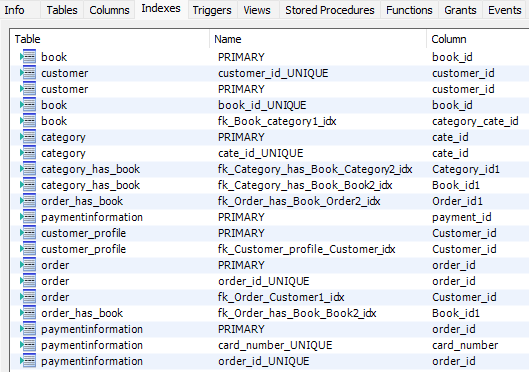
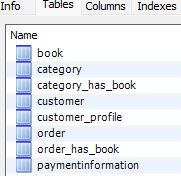
Using sql workbench:



**6. Relational Schema**

A screenshot of a computer

Description automatically generatedThe relational schema was mapped from the ER diagram, and the structure includes the following tables:

Using sql workbench:

**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\_DIVISION\_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;