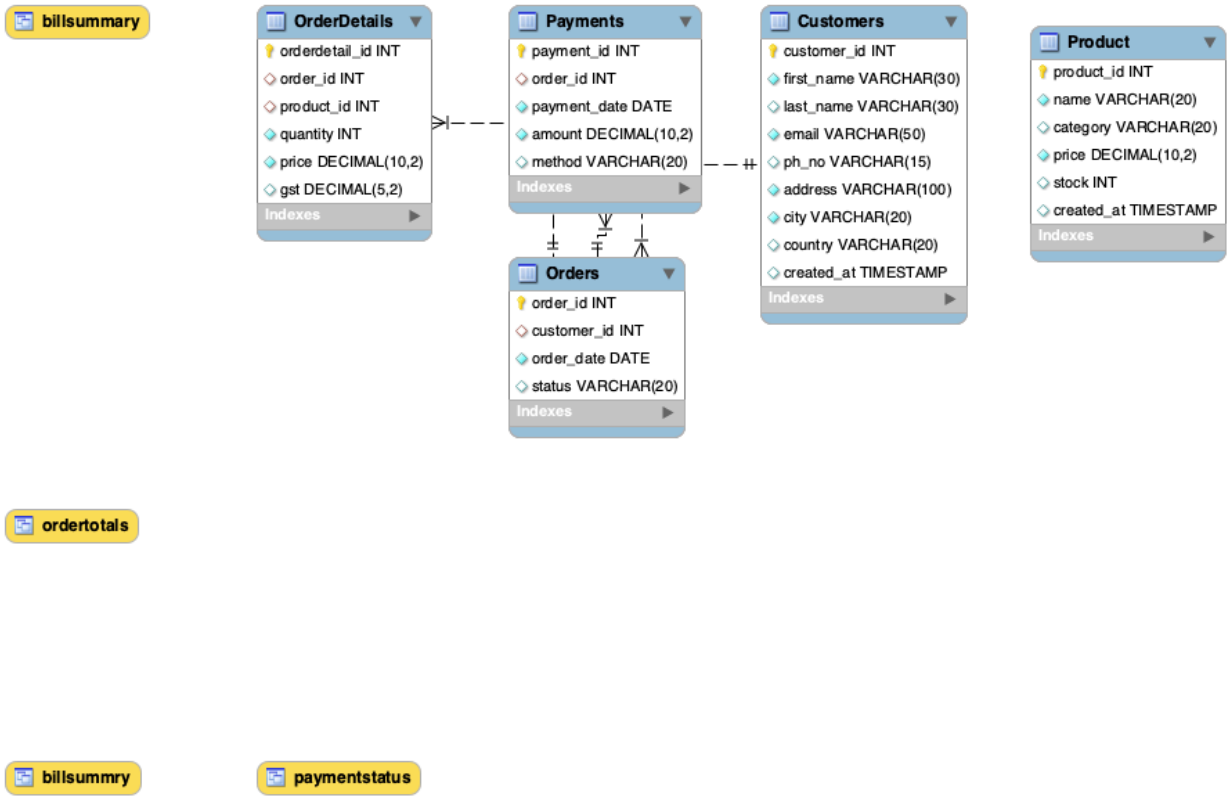
Project Report ~JINESH SHARMA

Title: Online Retail Database Management System



ER-DIAGRAM

1. Introduction

This project focuses on designing and implementing a relational database for an **online retail system**. The database is built using SQL and supports managing customers, products, orders, payments, and invoices with GST calculation.

The system ensures proper normalization, referential integrity using primary/foreign keys, and real-time bill summaries through SQL views.

---

2. Objectives

- To design a normalized database schema for an e-commerce platform.
- To store and manage information about customers, products, orders, payments.
- To maintain relationships between entities using foreign keys and constraints.
- To implement GST calculation within invoices.
- To create useful **views** (BillSummary, OrderTotals, PaymentStatus) for business analysis.

---

3. Tools & Technologies Used

- **Database:** MySQL
- **Language:** SQL
- **Design Tool:** MySQL Workbench (for ER diagram)
- **Version Control:** GitHub

---

4. Database Design

4.1 Entities & Attributes

1. **Customers** – Customer details (name, email, phone, address).

2. **Product** – Product catalog with price, stock, category.
3. **Orders** – Orders placed by customers.
4. **OrderDetails** – Individual product details in each order.
5. **Payments** – Payment transactions linked with orders.

4.2 ER Diagram

(Insert your ER diagram image here — the one you exported earlier)

---

5. Implementation

5.1 Database & Tables

- Created schema: `retail_db`.
- Tables: `Customers`, `Product`, `Orders`, `OrderDetails`, `Payments`.
- Constraints:
    - Primary Keys (`customer_id`, `product_id`, `order_id`, etc.).
    - Foreign Keys with `ON DELETE CASCADE`.
    - Default values for timestamps, GST, and status.

5.2 Sample Data

Inserted sample records for:

- **Customers** (5 entries).
- **Products** (4 entries, with price updates).
- **Orders & OrderDetails** (3 sample orders with multiple products).
- **Payments** (Credit Card, UPI, Cash).

5.3 GST & Invoice Logic

- Added `gst` column in `OrderDetails`.
- Implemented GST calculation in `BillSummary` view.

---

6. Views & Queries

6.1 BillSummary View

Shows product-wise billing with GST.

```
create or replace view BillSummary as
select
  od.order_id,
  p.name as product,
  od.quantity,
  od.price,
  od.gst,
  (od.quantity * od.price) as base_amount,
  (od.quantity * od.price * od.gst / 100) as gst_amount,
  (od.quantity * od.price * (1 + od.gst / 100)) as total_with_gst
from OrderDetails od
join Orders o on od.order_id = o.order_id
join Customers c on o.customer_id = c.customer_id
join Product p on od.product_id = p.product_id;
```

6.2 OrderTotals View

Summarizes order totals per customer including GST.

6.3 PaymentStatus View

Checks if payment matches the billed amount.

---

7. Sample Output Screenshots

(You can paste your query results / screenshots from MySQL Workbench here — e.g., BillSummary, PaymentStatus)

---

8. Conclusion

The project successfully demonstrates how a normalized database can be designed and implemented for an online retail system. Key features include:

- Efficient data organization and relationships.
- Automatic GST calculation.
- Useful reporting through SQL views.
- Support for analyzing payment status.

This system can be further extended with triggers, stored procedures, and integration with a web application.