A blue and black logo

AI-generated content may be incorrect.

Syed Muhammad Waiz Rizvi F2023376117

Section: D2

subject: DBMS

submitted to: Sir Usama Amjad

# Scenario 1:

## Entities and Attributes

| **Entity** | **Attributes** |
| --- | --- |
| **Book** | BookID (PK), Title, Edition (multi-valued), ISBN, Publisher, Year |
| **Author** | AuthorID (PK), Name |
| **Member** | MemberID (PK), Name, Email, Phone, Address |
| **Staff** | StaffID (PK), Name, Role (Admin, Librarian), Email, Phone |
| **Borrowing** | BorrowingID (PK), BorrowDate, DueDate, ReturnDate, Fine |
| **Reservation** | ReservationID (PK), ReservationDate |
| **Login** | UserID (PK), Username, Password, Role |

## Relationships

* **Writes**: Book ⇄ Author (many-to-many)
* **Borrows**: Member ⇄ Borrowing ⇄ Book (M:N with attributes)
* **Reserves**: Member ⇄ Reservation ⇄ Book (M:N)
* **Manages**: Staff ⇄ Borrowing / Reservation / Fine
* **Has\_Login**: Each Member or Staff has one Login (weak entity — requires ID from Member/Staff)

## Multivalued Attribute

* **Book → Edition**

## Weak Entity

* **Login** (depends on Member/Staff)

## Role-Based Relationships

* Staff Role defines whether they can Manage Fines, Reservations, etc.

## 2. Normalized Relational Schema (3NF)

Here's a **3NF** version of the schema:

**Book(BookID, Title, ISBN, Publisher, Year)**

**Edition(BookID, EditionNo)**

(Multi-valued attribute → separate table)

**Author(AuthorID, Name)**

**BookAuthor(BookID, AuthorID)**

(Many-to-many)

**Member(MemberID, Name, Email, Phone, Address)**

**Staff(StaffID, Name, Role, Email, Phone)**

**Login(UserID, Username, Password, Role, LinkedID, LinkedType)**

(Weak entity, references either MemberID or StaffID based on LinkedType)

**Borrowing(BorrowingID, MemberID, BookID, StaffID, BorrowDate, DueDate, ReturnDate, Fine)**

**Reservation(ReservationID, MemberID, BookID, ReservationDate, IsFulfilled)**

## 3. Candidate Keys and Foreign Keys

### Candidate Keys

* **Book**: BookID
* **Author**: AuthorID
* **Member**: MemberID
* **Staff**: StaffID
* **Login**: UserID
* **Borrowing**: BorrowingID
* **Reservation**: ReservationID

### Foreign Keys

* **Edition.BookID → Book.BookID**
* **BookAuthor.BookID → Book.BookID**
* **BookAuthor.AuthorID → Author.AuthorID**
* **Borrowing.MemberID → Member.MemberID**
* **Borrowing.BookID → Book.BookID**
* **Borrowing.StaffID → Staff.StaffID**
* **Reservation.MemberID → Member.MemberID**
* **Reservation.BookID → Book.BookID**
* **Login.LinkedID → Member.MemberID or Staff.StaffID (conditional)**

## Extended ERD:

A black and white image of a black background

AI-generated content may be incorrect.

# Scenario 2:

## 1. ER Diagram Design

**Entities and Attributes**

| **Entity** | **Attributes** |
| --- | --- |
| **Student** | StudentID (PK), Name, Email, Phone, Gender |
| **Department** | DeptID (PK), DeptName |
| **Course** | CourseID (PK), CourseName, CreditHours |
| **Instructor** | InstructorID (PK), Name, Email, Phone |
| **Login** | UserID (PK), Username, Password (for instructors) |
| **Assessment** | AssessmentID (PK), Type (Midterm, Final, Quiz,  Assignment), Weight (%) |
| **Grade** | GradeID (PK), MarksObtained |
| **Attendance** | AttendanceID (PK), TotalClasses, ClassesAttended,  Percentage, WarningFlag |

## Relationships

* **Student belongs to → Department** (M:1)
* **Student enrolls in → Course** (M:N)
* **Course is taught by → Instructor** (M:N)
* **Instructor has → Login** (1:1)
* **Course has → Assessments** (1:M)
* **Student receives → Grades** for each (Student, Course, Assessment) tuple
* **Student has → Attendance** per course

## ER Diagram Features

* **Weak Entity**: Login (dependent on Instructor)
* **Multivalued**: Grades per Assessment
* **Role-based**: Instructors manage grades/attendance only for their courses

## 2. Normalized Relational Schema (3NF)

**Student(StudentID, Name, Email, Phone, Gender, DeptID)**

* FK: DeptID → Department

**Department(DeptID, DeptName)**

**Course(CourseID, CourseName, CreditHours)**

**Instructor(InstructorID, Name, Email, Phone)**

**Login(UserID, Username, Password, InstructorID)**

* FK: InstructorID → Instructor

**Enrollment(StudentID, CourseID)**

* Composite PK: (StudentID, CourseID)
* FK: StudentID → Student
* FK: CourseID → Course

**CourseInstructor(CourseID, InstructorID)**

* Composite PK: (CourseID, InstructorID)
* FK: CourseID → Course
* FK: InstructorID → Instructor

**Assessment(AssessmentID, CourseID, Type, Weight)**

* FK: CourseID → Course

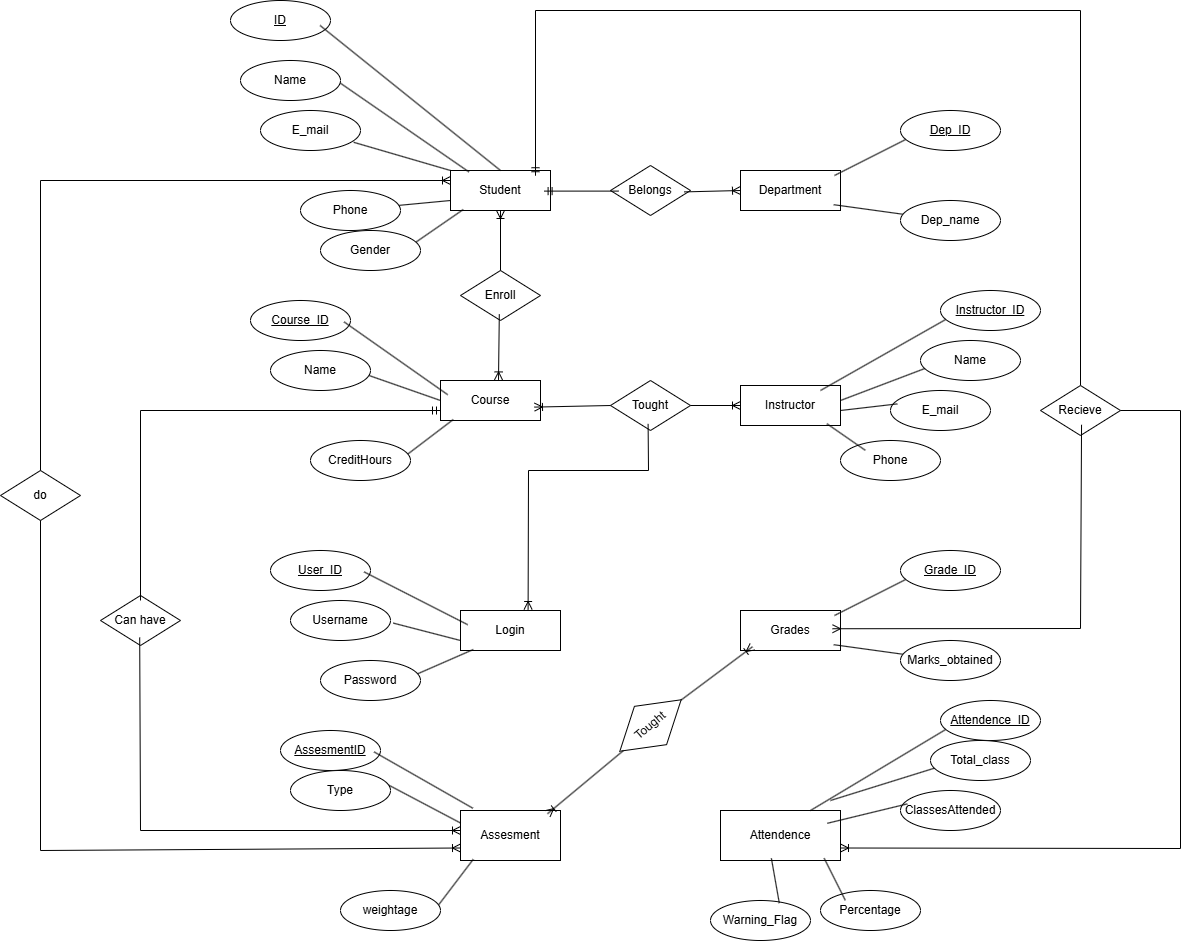
**Grade(GradeID, StudentID, AssessmentID, MarksObtained)**

* FK: StudentID → Student
* FK: AssessmentID → Assessment

**Attendance(AttendanceID, StudentID, CourseID, TotalClasses, ClassesAttended, Percentage, WarningFlag)**

* FK: StudentID → Student
* FK: CourseID → Course

## Detailed ERD:



# Scenario 3:

## 1. Comprehensive ER Diagram Design

**Entities and Attributes**

| **Entity** | **Attributes** |
| --- | --- |
| **Patient** | PatientID (PK), Name, DOB, Gender, Contact, Address |
| **Doctor** | DoctorID (PK), Name, Specialization, Contact |
| **Staff** | StaffID (PK), Name, Role (Admin, Nurse, Receptionist, Doctor), Contact |
| **Department** | DeptID (PK), DeptName |
| **Appointment** | AppointmentID (PK), PatientID (FK), DoctorID (FK), Date, Time, Purpose |
| **MedicalRecord** | RecordID (PK), PatientID (FK), Diagnosis, Procedure, Prescription, Date |
| **Room** | RoomID (PK), RoomType, ChargesPerDay |
| **Admission** | AdmissionID (PK), PatientID (FK), RoomID (FK), AdmitDate, DischargeDate |
| **Bill** | BillID (PK), PatientID (FK), TotalAmount, BillDate |
| **BillItem** | BillItemID (PK), BillID (FK), ItemType (Consultation, Lab Test, Medicine, Room), Cost |
| **DischargeSummary** | SummaryID (PK), AdmissionID (FK), Diagnosis, Procedure, FollowUp |

## Relationships

* **Patient → Appointments**, **Admissions**, **Bills**, **Medical Records**
* **Doctor → Appointments**, **Medical Records**
* **Room → Admissions**
* **Admission → DischargeSummary**
* **Staff → Department** (Many-to-One)
* **Staff can assist in → Appointments/Admissions/Billing** based on roles

## Special ER Features

* **Multivalued Attributes**: None (handled via tables like BillItems)
* **Weak Entities**: BillItem (dependent on Bill)
* **Role-based Relationships**: Staff role defines authority

## 2. Normalized Relational Schema (3NF)

**Patient(PatientID, Name, DOB, Gender, Contact, Address)**

**Doctor(DoctorID, Name, Specialization, Contact)**

**Staff(StaffID, Name, Role, Contact, DeptID)**

**Department(DeptID, DeptName)**

**Appointment(AppointmentID, PatientID, DoctorID, Date, Time, Purpose)**

* FK: PatientID → Patient
* FK: DoctorID → Doctor

**MedicalRecord(RecordID, PatientID, DoctorID, Diagnosis, Procedure, Prescription, Date)**

* FK: PatientID → Patient
* FK: DoctorID → Doctor

**Room(RoomID, RoomType, ChargesPerDay)**

**Admission(AdmissionID, PatientID, RoomID, AdmitDate, DischargeDate)**

* FK: PatientID → Patient
* FK: RoomID → Room

**DischargeSummary(SummaryID, AdmissionID, Diagnosis, Procedure, FollowUp)**

* FK: AdmissionID → Admission

**Bill(BillID, PatientID, BillDate, TotalAmount)**

* FK: PatientID → Patient

**BillItem(BillItemID, BillID, ItemType, Cost)**

* FK: BillID → Bill

## 3. Key Relationships

* **Staff.DeptID → Department.DeptID**
* **Doctor is a specialized role under Staff** (Can model via Staff.Role = 'Doctor')
* **Patient admitted to Room** via Admission
* **Bill can have multiple BillItems**
* **Discharge linked to Admission**
* **Medical Records linked to both Patient and Doctor**

## 4. SQL Queries

### Generate a Bill for a Patient

SELECT

B.BillID,

B.BillDate,

BI.ItemType,

BI.Cost,

SUM(BI.Cost) AS TotalAmount

FROM

Bill B

JOIN

BillItem BI ON B.BillID = BI.BillID

WHERE

B.PatientID = 'P001'

GROUP BY

B.BillID, B.BillDate, BI.ItemType, BI.Cost;

### List Doctors with Most Admitted Patients

SELECT

D.DoctorID,

D.Name,

COUNT(DISTINCT A.PatientID) AS TotalPatients

FROM

Doctor D

JOIN

Appointment Ap ON D.DoctorID = Ap.DoctorID

JOIN

Admission A ON A.PatientID = Ap.PatientID

GROUP BY

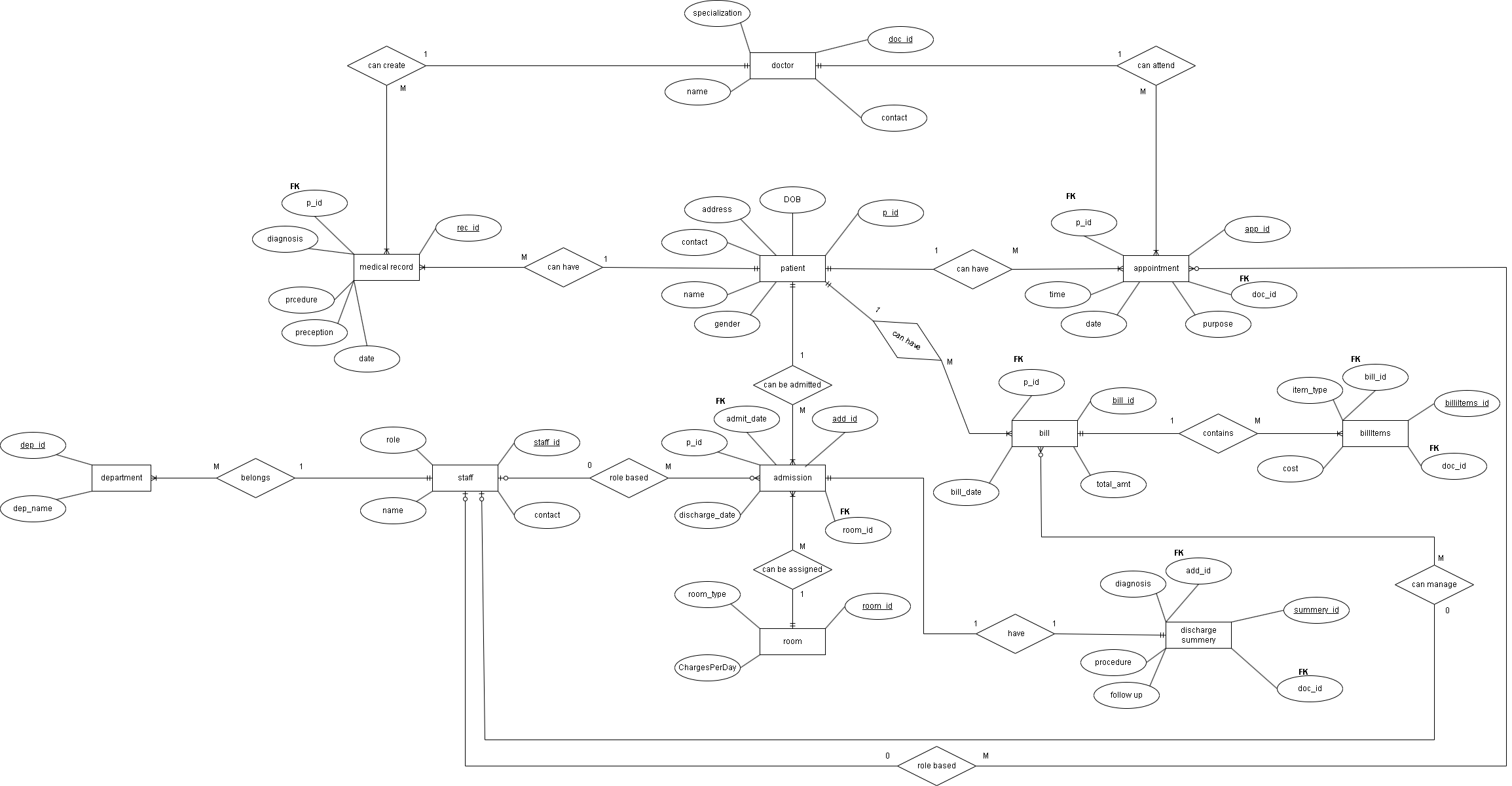
D.DoctorID, D.Name

ORDER BY

TotalPatients DESC

LIMIT 5;

## Comprehensive ERD:



# Scenario 4:

## 1. ER Diagram (Described Textually)

**Entities and Attributes**

| **Entity** | **Key Attributes** |
| --- | --- |
| **Customer** | CustomerID (PK), Name, Email, Phone, Address |
| **Product** | ProductID (PK), Name, Price, Description, StockLevel, ReorderThreshold, CategoryID (FK), VendorID (FK) |
| **Order** | OrderID (PK), CustomerID (FK), OrderDate, Status, ShippingAddress, PaymentMethod, TotalAmount |
| **OrderItem** | OrderID (FK), ProductID (FK), Quantity, PriceAtOrderTime *(Composite PK: OrderID + ProductID)* |
| **Category** | CategoryID (PK), Name, Description |
| **Vendor** | VendorID (PK), Name, Contact, Address |
| **Review** | ReviewID (PK), CustomerID (FK), ProductID (FK), Rating (1–5), Comment, ReviewDate |
| **Admin** | AdminID (PK), Username, Password |
| **Payment** | PaymentID (PK), OrderID (FK), PaymentDate, Amount, PaymentMode, Status |
| **Refund** | RefundID (PK), OrderID (FK), RefundDate, Amount, Reason |

## Relationships

* **Customer → Order** (1:M)
* **Order → OrderItem** (1:M) ⬅️ **(Many-to-Many with attributes)**
* **Product → OrderItem** (1:M)
* **Product → Category** (M:1)
* **Product → Vendor** (M:1)
* **Product → Review (Customer)** (M:M with Rating, Comment)
* **Order → Payment** (1:1 or 1:M)
* **Order → Refund** (0:1)

## Special Notes:

* **Many-to-Many with Attributes**: OrderItem is the M:N relation with attributes like quantity, price.
* **Product reviews** also form a many-to-many relationship with Customer, with attributes Rating, Comment.

## 2. Normalized Relational Schema (3NF)

**Customer(CustomerID, Name, Email, Phone, Address)**

**Product(ProductID, Name, Price, Description, StockLevel, ReorderThreshold, CategoryID, VendorID)**

* FK → Category(CategoryID)
* FK → Vendor(VendorID)

**Order(OrderID, CustomerID, OrderDate, Status, ShippingAddress, PaymentMethod, TotalAmount)**

* FK → Customer(CustomerID)

**OrderItem(OrderID, ProductID, Quantity, PriceAtOrderTime)**

* PK → (OrderID, ProductID)
* FK → Order(OrderID)
* FK → Product(ProductID)

**Category(CategoryID, Name, Description)**

**Vendor(VendorID, Name, Contact, Address)**

**Admin(AdminID, Username, Password)**

**Review(ReviewID, CustomerID, ProductID, Rating, Comment, ReviewDate)**

* FK → Customer(CustomerID)
* FK → Product(ProductID)

**Payment(PaymentID, OrderID, PaymentDate, Amount, PaymentMode, Status)**

* FK → Order(OrderID)

**Refund(RefundID, OrderID, RefundDate, Amount, Reason)**

* FK → Order(OrderID)

3. Integrity Constraints

-- Quantity in OrderItem must be positive

CHECK (Quantity > 0)

-- StockLevel must be non-negative

CHECK (StockLevel >= 0)

-- Rating between 1 and 5

CHECK (Rating >= 1 AND Rating <= 5)

-- Refunds must not exceed total order amount (logic constraint)

CHECK (Amount >= 0)

## Detailed erd:

