# Mid Practice Problems- DB

### Long Question 1: **Library Management System**

#### Schema:

1. **Books**:
   * BookID (Primary Key)
   * Title
   * Author
   * Category
2. **Members**:
   * MemberID (Primary Key)
   * Name
   * JoinDate
   * MembershipType
3. **BorrowRecords**:
   * BorrowID (Primary Key)
   * BookID (Foreign Key)
   * MemberID (Foreign Key)
   * BorrowDate
   * ReturnDate

#### Tasks:

1. Create the schema for all three tables.
2. Insert 10 records in each table, ensuring interrelated data.
3. Retrieve a list of all books borrowed by members whose membership type is "Premium".
4. List the names of members who have borrowed more than 3 books.
5. Find the most borrowed book in the library and its author.
6. Display all borrow records where the return date is NULL.
7. Retrieve all members who borrowed books in the "Science Fiction" category.
8. Update the return date of a specific borrow record where the book was borrowed 15 days ago.
9. Delete a member who has not borrowed any book since the last 6 months.
10. Find the total number of books borrowed grouped by category.

### Long Question 2: **E-Commerce System**

#### Schema:

1. **Products**:
   * ProductID (Primary Key)
   * Name
   * Price
   * Category
2. **Customers**:
   * CustomerID (Primary Key)
   * Name
   * City
   * JoinDate
3. **Orders**:
   * OrderID (Primary Key)
   * CustomerID (Foreign Key)
   * OrderDate
4. **OrderDetails**:
   * DetailID (Primary Key)
   * OrderID (Foreign Key)
   * ProductID (Foreign Key)
   * Quantity

#### Tasks:

1. Create the schema for all tables and set foreign key relationships.
2. Insert 10 records in each table with meaningful relationships.
3. Retrieve the details of customers who placed orders in the last month.
4. Display the total revenue generated for each product.
5. Find the customer who placed the maximum number of orders.
6. Retrieve the names of products purchased by customers from a specific city.
7. Update the price of all products in the "Electronics" category by 10%.
8. Delete all orders where the order date is more than 2 years old.
9. Retrieve a list of all products that were never ordered.
10. Find the average order value grouped by customer.

### Long Question 3: **University Management System**

#### Schema:

1. **Students**:
   * StudentID (Primary Key)
   * Name
   * Program
   * EnrollmentYear
2. **Courses**:
   * CourseID (Primary Key)
   * CourseName
   * Credits
3. **Enrollments**:
   * EnrollmentID (Primary Key)
   * StudentID (Foreign Key)
   * CourseID (Foreign Key)
   * Grade

#### Tasks:

1. Create the schema for all three tables and establish relationships.
2. Insert 10 records in each table ensuring relational integrity.
3. Display the names of students enrolled in more than 2 courses.
4. Find the course with the highest average grade.
5. Retrieve the names of students who have never enrolled in any course.
6. Update the program of students enrolled before 2020 to "Alumni".
7. List the total credits taken by each student.
8. Retrieve a list of all students and their grades for a specific course.
9. Delete a course that no student is currently enrolled in.
10. Find the total number of students grouped by their program.

### Long Question 4: **Hospital Management System**

#### Schema:

1. **Doctors**:
   * DoctorID (Primary Key)
   * Name
   * Specialization
   * ExperienceYears
2. **Patients**:
   * PatientID (Primary Key)
   * Name
   * Age
   * Gender
   * AdmissionDate
3. **Appointments**:
   * AppointmentID (Primary Key)
   * DoctorID (Foreign Key)
   * PatientID (Foreign Key)
   * AppointmentDate
   * Diagnosis
4. **Medications**:
   * MedicationID (Primary Key)
   * Name
   * Dosage
   * AppointmentID (Foreign Key)

#### Tasks:

1. Create the schema and set relationships with constraints.
2. Insert 15 records into each table with meaningful interrelations.
3. Retrieve the names of patients treated by doctors with more than 10 years of experience.
4. Display the count of male and female patients admitted this year.
5. Find the doctor who has treated the maximum number of patients.
6. Retrieve the names of patients who received medication for a specific diagnosis.
7. Update the specialization of doctors who have treated more than 50 patients in the last year.
8. Delete the records of appointments older than 5 years and their associated medications.
9. List all doctors and their number of current appointments sorted by the highest count.
10. Find the average age of patients grouped by gender.

### Long Question 5: **Movie Ticket Booking System**

#### Schema:

1. **Movies**:
   * MovieID (Primary Key)
   * Title
   * Genre
   * ReleaseDate
2. **Cinemas**:
   * CinemaID (Primary Key)
   * Name
   * Location
   * TotalSeats
3. **ShowTimes**:
   * ShowID (Primary Key)
   * CinemaID (Foreign Key)
   * MovieID (Foreign Key)
   * ShowTime
   * AvailableSeats
4. **Bookings**:
   * BookingID (Primary Key)
   * ShowID (Foreign Key)
   * CustomerName
   * BookingDate
   * SeatsBooked

#### Tasks:

1. Create the schema for all tables, ensuring constraints and relationships.
2. Insert 20 records into each table with well-structured interrelations.
3. Retrieve the names of customers who booked tickets for a specific movie genre.
4. Find the most booked cinema and its location.
5. Display the number of available seats for all shows happening today.
6. Update the total seats for cinemas where the available seats fall below 10.
7. List all movies with their respective total bookings and sort by the highest booking count.
8. Delete records of shows that have zero bookings and their related entries.
9. Retrieve all shows of a specific movie in a specific city.
10. Find the movie with the highest revenue generated (assume a ticket price).

### Long Question 6: **Online Education Platform**

#### Schema:

1. **Instructors**:
   * InstructorID (Primary Key)
   * Name
   * Specialization
   * ExperienceYears
2. **Courses**:
   * CourseID (Primary Key)
   * Title
   * InstructorID (Foreign Key)
   * DurationWeeks
3. **Students**:
   * StudentID (Primary Key)
   * Name
   * Email
   * EnrollmentDate
4. **Enrollments**:
   * EnrollmentID (Primary Key)
   * CourseID (Foreign Key)
   * StudentID (Foreign Key)
   * Grade
5. **Assignments**:
   * AssignmentID (Primary Key)
   * CourseID (Foreign Key)
   * Title
   * DueDate

#### Tasks:

1. Design the schema and establish relationships with proper constraints.
2. Populate each table with at least 15 records, ensuring data integrity.
3. Retrieve the names of students enrolled in courses taught by instructors with over 5 years of experience.
4. Display the total number of assignments for each course.
5. Update the grades for students who scored below a passing grade in any course.
6. Find the most popular course based on the number of enrollments.
7. Retrieve all students who have not completed any assignments.
8. Delete courses that have no enrollments and their associated assignments.
9. List all courses along with their instructors and the number of students enrolled.
10. Find the average grade for each course and sort them in descending order.

### Long Question 7: **Airline Reservation System**

#### Schema:

1. **Airlines**:
   * AirlineID (Primary Key)
   * Name
   * Country
   * Alliance
2. **Flights**:
   * FlightID (Primary Key)
   * AirlineID (Foreign Key)
   * Source
   * Destination
   * DepartureTime
   * ArrivalTime
3. **Passengers**:
   * PassengerID (Primary Key)
   * Name
   * Email
   * PhoneNumber
4. **Reservations**:
   * ReservationID (Primary Key)
   * PassengerID (Foreign Key)
   * FlightID (Foreign Key)
   * BookingDate
   * SeatClass

#### Tasks:

1. Create the schema with necessary constraints, including time validation for flights.
2. Insert at least 20 records in each table with diverse scenarios (e.g., different alliances and routes).
3. Retrieve the names of passengers who booked flights operated by airlines in the same alliance.
4. List all flights with a layover between two specified cities.
5. Find the airline with the highest number of flights departing from a specific city.
6. Display the total number of reservations grouped by seat class for flights on a specific date.
7. Update the alliance of airlines whose country has imposed new aviation regulations.
8. Delete all reservations for flights that are canceled (assume a cancellation flag).
9. Retrieve all passengers who booked flights but have never flown yet.
10. Find the route with the highest number of reservations across all airlines.

### Long Question 8: **Banking System**

#### Schema:

1. **Branches**:
   * BranchID (Primary Key)
   * Name
   * City
   * IFSCCode
2. **Accounts**:
   * AccountID (Primary Key)
   * BranchID (Foreign Key)
   * AccountHolder
   * Balance
   * AccountType
3. **Transactions**:
   * TransactionID (Primary Key)
   * AccountID (Foreign Key)
   * TransactionDate
   * Amount
   * TransactionType

#### Tasks:

1. Design the schema and include constraints for minimum account balance.
2. Populate tables with 25 records each, ensuring diverse account types and balances.
3. Retrieve the names of account holders with an account balance greater than the average balance across all accounts.
4. List all branches and their total deposits and withdrawals grouped by city.
5. Find the account with the highest transaction volume in the last quarter.
6. Update account balances based on transactions performed within a given date range.
7. Retrieve a list of all inactive accounts (accounts with no transactions in the past year).
8. Delete transactions older than 5 years but ensure balance consistency in accounts.
9. Retrieve the account holders whose transactions include amounts exceeding a specific threshold.
10. Find the city with the maximum total balance across all branches.

### Long Question 9: **Hotel Management System**

#### Schema:

1. **Hotels**:
   * HotelID (Primary Key)
   * Name
   * City
   * StarRating
2. **Rooms**:
   * RoomID (Primary Key)
   * HotelID (Foreign Key)
   * RoomType
   * PricePerNight
   * Availability
3. **Guests**:
   * GuestID (Primary Key)
   * Name
   * Email
   * PhoneNumber
4. **Bookings**:
   * BookingID (Primary Key)
   * GuestID (Foreign Key)
   * RoomID (Foreign Key)
   * CheckInDate
   * CheckOutDate

#### Tasks:

1. Create the schema, ensuring proper relationships and constraints for availability checks.
2. Insert 20 records in each table with varied scenarios, including overbooking attempts.
3. Retrieve the names of guests who stayed in 5-star hotels only.
4. List all hotels with their average room price grouped by city.
5. Find the room type that generated the highest revenue across all hotels.
6. Update the availability of rooms based on bookings for a specific date range.
7. Retrieve the details of bookings where guests stayed for more than a week.
8. Delete all bookings for hotels that are permanently closed (assume a flag).
9. List all hotels and their number of bookings in descending order.
10. Find the total revenue generated by each hotel and sort by the highest revenue.

### Long Question 10: **Employee Management System**

#### Schema:

1. **Departments**:
   * DepartmentID (Primary Key)
   * Name
   * Location
   * Budget
2. **Employees**:
   * EmployeeID (Primary Key)
   * Name
   * DepartmentID (Foreign Key)
   * JoinDate
   * Salary
3. **Projects**:
   * ProjectID (Primary Key)
   * ProjectName
   * DepartmentID (Foreign Key)
   * Budget
4. **Assignments**:
   * AssignmentID (Primary Key)
   * EmployeeID (Foreign Key)
   * ProjectID (Foreign Key)
   * HoursWorked

#### Tasks (Select Query Name: EfficientWorkers):

1. Create the schema with strict foreign key constraints.
2. Insert 25 records into each table with realistic data.
3. Write a query (EfficientWorkers) to find employees who worked more than 40 hours on projects with a budget higher than their department's average budget.
4. Retrieve the departments with the highest combined employee salaries, grouped by location.
5. Write a query to find the least-assigned project in each department.
6. Update the salary of employees in departments where the total hours worked on projects exceed a given threshold.
7. Delete all assignments where the employee’s total hours are less than 10 across all projects.
8. Write a query to find projects with at least one employee from multiple departments.
9. Retrieve employees assigned to more than three projects in a specific department.
10. Find the total budget utilization for each department, calculated from project budgets and hours worked.

### Long Question 11: **Online Retail System**

#### Schema:

1. **Products**:
   * ProductID (Primary Key)
   * Name
   * Category
   * Price
   * Stock
2. **Customers**:
   * CustomerID (Primary Key)
   * Name
   * City
   * JoinDate
3. **Orders**:
   * OrderID (Primary Key)
   * CustomerID (Foreign Key)
   * OrderDate
4. **OrderDetails**:
   * DetailID (Primary Key)
   * OrderID (Foreign Key)
   * ProductID (Foreign Key)
   * Quantity

#### Tasks (Select Query Name: PopularCategory):

1. Create the schema and relationships with constraints.
2. Populate the tables with at least 30 records each.
3. Write a query (PopularCategory) to find the product category that generated the highest revenue in the last quarter.
4. Retrieve customers who ordered products worth more than their city’s average order value.
5. Write a query to find products that are out of stock but have pending orders.
6. Update the stock for products where recent orders have exceeded 50 units.
7. Delete all orders placed by customers who haven’t ordered anything in the last year.
8. Retrieve the names of customers who ordered the same product multiple times.
9. Write a query to find the average order value for each product category.
10. List all products with their total quantity sold, sorted by the highest quantity.

### Long Question 12: **Social Media Analytics System**

#### Schema:

1. **Users**:
   * UserID (Primary Key)
   * Name
   * JoinDate
   * Country
2. **Posts**:
   * PostID (Primary Key)
   * UserID (Foreign Key)
   * Content
   * PostDate
3. **Likes**:
   * LikeID (Primary Key)
   * PostID (Foreign Key)
   * UserID (Foreign Key)
4. **Comments**:
   * CommentID (Primary Key)
   * PostID (Foreign Key)
   * UserID (Foreign Key)
   * Content
   * CommentDate

#### Tasks (Select Query Name: InfluentialUsers):

1. Design the schema and include constraints for unique likes and comments per user.
2. Populate tables with at least 40 records each.
3. Write a query (InfluentialUsers) to find users with posts that have received likes and comments from users in at least 3 different countries.
4. Retrieve posts with the highest number of likes and comments in the last month.
5. Write a query to find users who commented on their own posts more than 3 times.
6. Update the content of posts with less than 5 likes to append a "Boosted" tag.
7. Delete all posts and their associated likes/comments for users inactive for over a year.
8. Write a query to find the average likes per post for each country.
9. List all users with the number of posts, likes, and comments they have received.
10. Retrieve posts with more comments than likes, sorted by the highest comment count.

### Long Question 13: **E-Library Management System**

#### Schema:

1. **Books**:
   * BookID (Primary Key)
   * Title
   * Author
   * Category
   * CopiesAvailable
2. **Members**:
   * MemberID (Primary Key)
   * Name
   * Email
   * MembershipDate
3. **BorrowRecords**:
   * BorrowID (Primary Key)
   * MemberID (Foreign Key)
   * BookID (Foreign Key)
   * BorrowDate
   * ReturnDate
4. **Penalties**:
   * PenaltyID (Primary Key)
   * BorrowID (Foreign Key)
   * PenaltyAmount

#### Tasks (Select Query Name: FrequentBorrowers):

1. Create the schema and define relationships with constraints, including penalties for late returns.
2. Populate the tables with at least 30 records, ensuring some overdue records for penalties.
3. Write a query (FrequentBorrowers) to find members who borrowed books more than 5 times in the past year.
4. Retrieve the most borrowed book category for each month in the past year.
5. Write a query to find books that have never been borrowed.
6. Update the penalty amount for all records where the return is overdue by more than 15 days.
7. Delete all records of members whose membership expired 3 years ago and they have no active penalties.
8. Retrieve the average penalty amount grouped by book category.
9. List all members and the total number of books they borrowed, sorted by the highest count.
10. Write a query to find the member who has paid the most in penalties over the last year.

### Long Question 14: **Ride-Sharing Platform**

#### Schema:

1. **Drivers**:
   * DriverID (Primary Key)
   * Name
   * VehicleType
   * Rating
2. **Rides**:
   * RideID (Primary Key)
   * DriverID (Foreign Key)
   * PickupLocation
   * DropoffLocation
   * Fare
   * RideDate
3. **Passengers**:
   * PassengerID (Primary Key)
   * Name
   * Email
   * JoinDate
4. **RideRequests**:
   * RequestID (Primary Key)
   * PassengerID (Foreign Key)
   * RideID (Foreign Key)
   * RequestDate
   * Status

#### Tasks (Select Query Name: TopEarners):

1. Design the schema with constraints ensuring no duplicate ride requests per passenger.
2. Populate the tables with at least 50 records, ensuring diverse ride scenarios.
3. Write a query (TopEarners) to find the top 3 drivers who earned the most in the past month.
4. Retrieve the number of rides completed by drivers with an average rating greater than 4.5.
5. Write a query to find the most frequent pickup location for passengers who joined in the last 6 months.
6. Update the fare for rides where the distance between pickup and dropoff exceeds a specific threshold.
7. Delete all ride requests that were canceled and older than 30 days.
8. Retrieve the average fare for rides grouped by vehicle type.
9. List all passengers who requested rides but have never taken one.
10. Write a query to find the driver with the highest number of 5-star rated rides.

### Long Question 15: **University Course Management System**

#### Schema:

1. **Courses**:
   * CourseID (Primary Key)
   * Title
   * Credits
   * Department
2. **Students**:
   * StudentID (Primary Key)
   * Name
   * Major
   * EnrollmentDate
3. **Enrollments**:
   * EnrollmentID (Primary Key)
   * StudentID (Foreign Key)
   * CourseID (Foreign Key)
   * Grade
4. **Instructors**:
   * InstructorID (Primary Key)
   * Name
   * Department
5. **Assignments**:
   * AssignmentID (Primary Key)
   * CourseID (Foreign Key)
   * Title
   * DueDate

#### Tasks (Select Query Name: TopPerformers):

1. Create the schema with relationships and include constraints for grade validation.
2. Populate the tables with at least 40 records, ensuring diverse enrollment and grading data.
3. Write a query (TopPerformers) to find students with the highest average grades in each major.
4. Retrieve courses with the lowest enrollment count in each department.
5. Write a query to find students who have not submitted assignments in more than 3 courses.
6. Update the grades for students in courses taught by a specific instructor, following a grading curve.
7. Delete enrollment records for students who dropped out last semester.
8. Retrieve the average grade for each course grouped by department.
9. List all instructors and their total number of assignments, sorted by the highest count.
10. Write a query to find the total credits earned by students in a specific year.