**Exercise 1: Create a Database and Tables**

**Exercise:**

1. Create a database named company\_db.
2. Create a table named employees with the following columns:
   * emp\_id (INT, Primary Key, Auto Increment)
   * first\_name (VARCHAR(50), Not Null)
   * last\_name (VARCHAR(50), Not Null)
   * department\_id (INT)
   * salary (DECIMAL(10, 2), Not Null)
3. Create a table named departments with the following columns:
   * department\_id (INT, Primary Key, Auto Increment)
   * department\_name (VARCHAR(50), Not Null, Unique)

**Exercise 2: Alter Table by Adding Columns**

**Exercise:**

1. Add a email column to the employees table with VARCHAR(100) and make it unique.
2. Add a hire\_date column to the employees table as DATE.
3. Modify the salary column to be DECIMAL(12, 2).

**Exercise 3: Add Foreign Key Constraints**

**Exercise:**

1. Add a foreign key constraint from employees.department\_id to departments.department\_id.
2. Ensure that when a department is deleted, employees from that department are set to NULL for their department\_id.

**Exercise 4: Rename Tables and Columns**

**Exercise:**

1. Rename the employees table to company\_employees.
2. Rename the first\_name and last\_name columns in the company\_employees table to fname and lname.

**Exercise 5: Drop a Column and a Foreign Key**

**Exercise:**

1. Drop the email column from the company\_employees table.
2. Drop the foreign key constraint that links department\_id between company\_employees and departments.

**Exercise 6: Create an Index and Drop It**

**Exercise:**

1. Create an index on the lname column of the company\_employees table.
2. Drop the created index on the lname column.

**Exercise 7: Truncate and Drop Tables**

**Exercise:**

1. Truncate the departments table.
2. Drop the departments table.

**Exercise 8: Alter a Table with Multiple Changes**

**Exercise:**

1. In the company\_employees table, add a new column manager\_id as INT.
2. Add a foreign key on manager\_id that references company\_employees(emp\_id).
3. Modify the hire\_date column to DATETIME.

**Exercise 9: Drop Multiple Tables at Once**

**Exercise:**

1. Drop the company\_employees and departments tables if they exist.

**Exercise 10: Add a CHECK Constraint and Drop It**

**Exercise:**

1. Add a CHECK constraint to the company\_employees table that ensures salary is greater than 0.
2. Drop the CHECK constraint from the salary column.

**Exercise 11: Rename Multiple Tables at Once**

**Exercise:**

1. Rename the company\_employees table to employees, and rename the departments table to company\_departments in a single query.

**Exercise 12: Create a Table with Composite Keys**

**Exercise:**

1. Create a table named project\_assignments with the following columns:
   * emp\_id (INT, Foreign Key to employees(emp\_id))
   * project\_id (INT, Not Null)
   * assigned\_date (DATE, Not Null)
   * Composite Primary Key on emp\_id and project\_id

**Exercise 13: Add Multiple Constraints and Indexes**

**Exercise:**

1. In the employees table, add a column phone\_number (VARCHAR(15)) and ensure it is unique.
2. Add a CHECK constraint that ensures the salary is at least 30000.
3. Add an index on the hire\_date column.

**Exercise 14: Modify and Rename Multiple Columns**

**Exercise:**

1. In the employees table, rename the column phone\_number to contact\_number.
2. Modify the contact\_number column to be VARCHAR(20).
3. Rename the hire\_date column to start\_date and change its data type to DATETIME.

**Exercise 15: Create Tables with Foreign Key Constraints**

**Exercise:**

1. Create a projects table with the following columns:
   * project\_id (INT, Primary Key, Auto Increment)
   * project\_name (VARCHAR(100), Not Null)
   * start\_date (DATE, Not Null)
   * end\_date (DATE)
2. Create a project\_assignments table with the following columns:
   * assignment\_id (INT, Primary Key, Auto Increment)
   * emp\_id (INT, Foreign Key references employees(emp\_id))
   * project\_id (INT, Foreign Key references projects(project\_id))
   * role (VARCHAR(50))

**Exercise 16: Drop Columns and Foreign Keys**

**Exercise:**

1. Drop the contact\_number column from the employees table.
2. Drop the foreign key constraint that references employees(emp\_id) from the project\_assignments table.

**Exercise 17: Truncate Tables with Foreign Key Constraints**

**Exercise:**

1. Truncate the project\_assignments table.
2. Truncate the projects table.

**Note:** MySQL does not allow truncating tables that are referenced by foreign keys, so you may need to drop the foreign key constraint first.

**Exercise 18: Rename Multiple Tables and Columns**

**Exercise:**

1. Rename the project\_assignments table to employee\_projects.
2. Rename the role column in the employee\_projects table to job\_role.
3. Rename the projects table to company\_projects.

**Exercise 19: Drop Tables and Constraints**

**Exercise:**

1. Drop the company\_projects table.
2. Drop the employee\_projects table.
3. Drop the check\_salary\_min constraint from the employees table.

**Exercise 20: Add and Drop Multiple Columns**

**Exercise:**

1. Add the following columns to the employees table:
   * department\_name (VARCHAR(50))
   * job\_title (VARCHAR(50))
2. Drop both department\_name and job\_title columns.

**Exercise 21: Create a Table with Composite Primary Key and Foreign Key**

**Exercise:**

1. Create a project\_employees table with the following columns:
   * emp\_id (INT, Foreign Key references employees(emp\_id))
   * project\_id (INT, Foreign Key references projects(project\_id))
   * assignment\_date (DATE, Not Null)
   * Primary Key on (emp\_id, project\_id).

**Exercise 22: Add Check Constraints and Drop Them**

**Exercise:**

1. Add a CHECK constraint on the salary column of the employees table to ensure it’s greater than 50000.
2. Add a CHECK constraint on the assignment\_date in the project\_employees table to ensure the date is not in the future.
3. Drop both CHECK constraints.

**Exercise 23: Truncate and Drop Multiple Tables**

**Exercise:**

1. Truncate the project\_employees table.
2. Truncate the employees table.
3. Drop both project\_employees and employees tables.

**Exercise 24: Alter Table with Composite Foreign Keys**

**Exercise:**

1. Alter the project\_employees table to add a composite foreign key (emp\_id, project\_id) that references the employees and projects tables, respectively.

**Exercise 25: Alter and Add a Composite Primary Key**

**Exercise:**

1. Create a departments table with the following columns:
   * dept\_id (INT, Primary Key, Auto Increment)
   * dept\_name (VARCHAR(100), Not Null)
2. Create an employees table with the following columns:
   * emp\_id (INT, Auto Increment)
   * emp\_name (VARCHAR(100), Not Null)
   * dept\_id (INT, Foreign Key references departments(dept\_id))
3. Alter the employees table to add a **composite primary key** on both emp\_id and dept\_id.

**Exercise 26: Drop and Re-add Foreign Key Constraint**

**Exercise:**

1. Create a projects table with the following columns:
   * project\_id (INT, Primary Key, Auto Increment)
   * project\_name (VARCHAR(100), Not Null)
   * dept\_id (INT, Foreign Key references departments(dept\_id))
2. Alter the projects table to **drop** the foreign key constraint on dept\_id.
3. Re-add the foreign key constraint on dept\_id that references departments(dept\_id) with ON DELETE CASCADE.

**Exercise 27: Alter Foreign Key and Change Data Type**

**Exercise:**

1. Create a tasks table with the following columns:
   * task\_id (INT, Primary Key, Auto Increment)
   * task\_name (VARCHAR(100), Not Null)
   * emp\_id (INT, Foreign Key references employees(emp\_id))
2. Alter the tasks table to change the data type of emp\_id to BIGINT.
3. Re-add the foreign key constraint to reference the modified column.

**Exercise 28: Drop Primary Key and Add New Composite Primary Key**

**Exercise:**

1. Create a projects\_tasks table with the following columns:
   * project\_id (INT, Foreign Key references projects(project\_id))
   * task\_id (INT, Foreign Key references tasks(task\_id))
   * assignment\_date (DATE, Not Null)
2. Drop the existing primary key (if any) from the projects\_tasks table.
3. Add a **composite primary key** on both project\_id and task\_id.

**Exercise 29: Alter Foreign Key to Include ON UPDATE CASCADE**

**Exercise:**

1. Create a managers table with the following columns:
   * manager\_id (INT, Primary Key, Auto Increment)
   * manager\_name (VARCHAR(100), Not Null)
   * dept\_id (INT, Foreign Key references departments(dept\_id))
2. Alter the foreign key constraint on dept\_id to add ON UPDATE CASCADE.

**Exercise 30: Add and Modify Multiple Foreign Keys**

**Exercise:**

1. Create a tasks\_log table with the following columns:
   * log\_id (INT, Primary Key, Auto Increment)
   * task\_id (INT, Foreign Key references tasks(task\_id))
   * manager\_id (INT, Foreign Key references managers(manager\_id))
   * log\_date (DATE, Not Null)
2. Modify the foreign key on task\_id to include ON DELETE CASCADE.
3. Modify the foreign key on manager\_id to include ON UPDATE CASCADE.

**Exercise 31: Add and Drop Foreign Key in a Large Schema**

**Exercise:**

1. Create a workshops table with the following columns:
   * workshop\_id (INT, Primary Key, Auto Increment)
   * workshop\_name (VARCHAR(100), Not Null)
2. Create a workshop\_attendance table with the following columns:
   * attendance\_id (INT, Primary Key, Auto Increment)
   * emp\_id (INT, Foreign Key references employees(emp\_id))
   * workshop\_id (INT, Foreign Key references workshops(workshop\_id))
3. Drop the foreign key on emp\_id from the workshop\_attendance table and re-add it with ON DELETE CASCADE.

**Exercise 32: Add and Drop Composite Foreign Key**

**Exercise:**

1. Create a task\_allocations table with the following columns:
   * allocation\_id (INT, Primary Key, Auto Increment)
   * project\_id (INT, Foreign Key references projects(project\_id))
   * task\_id (INT, Foreign Key references tasks(task\_id))
2. Alter the task\_allocations table to add a **composite foreign key** on both project\_id and task\_id.
3. Drop the composite foreign key.

**Exercise 33: Add a Composite Foreign Key on Two Columns**

**Exercise:**

1. Create a training\_sessions table with the following columns:
   * session\_id (INT, Primary Key, Auto Increment)
   * session\_name (VARCHAR(100), Not Null)
   * trainer\_id (INT, Foreign Key references employees(emp\_id))
2. Create a session\_attendance table with the following columns:
   * attendance\_id (INT, Primary Key, Auto Increment)
   * emp\_id (INT, Foreign Key references employees(emp\_id))
   * session\_id (INT, Foreign Key references training\_sessions(session\_id))
3. Alter the session\_attendance table to add a **composite foreign key** on emp\_id and session\_id.

**Exercise 34: Modify Primary Key and Foreign Key on Multiple Columns**

**Exercise:**

1. Create a course\_catalog table with the following columns:
   * course\_id (INT, Primary Key, Auto Increment)
   * course\_name (VARCHAR(100), Not Null)
2. Create a student\_courses table with the following columns:
   * student\_id (INT, Foreign Key references employees(emp\_id))
   * course\_id (INT, Foreign Key references course\_catalog(course\_id))
3. Alter the student\_courses table to drop the current primary key and create a **composite primary key** on student\_id and course\_id.

**Exercise 35: Add Multiple Foreign Keys with ON DELETE and ON UPDATE Options**

**Exercise:**

1. Create a clients table with the following columns:
   * client\_id (INT, Primary Key, Auto Increment)
   * client\_name (VARCHAR(100), Not Null)
2. Create a client\_projects table with the following columns:
   * client\_id (INT, Foreign Key references clients(client\_id))
   * project\_id (INT, Foreign Key references projects(project\_id))
3. Alter the client\_projects table to:
   * Add ON DELETE CASCADE to the client\_id foreign key.
   * Add ON UPDATE CASCADE to the project\_id foreign key.

**Exercise 36: Add and Drop Primary Key with Altered Columns**

**Exercise:**

1. Create a vendors table with the following columns:
   * vendor\_id (INT, Primary Key, Auto Increment)
   * vendor\_name (VARCHAR(100), Not Null)
2. Create a vendor\_products table with the following columns:
   * product\_id (INT, Foreign Key references projects(project\_id))
   * vendor\_id (INT, Foreign Key references vendors(vendor\_id))
3. Alter the vendor\_products table to drop the primary key, modify vendor\_id and product\_id data types to BIGINT, and then re-add the composite primary key on vendor\_id and product\_id.

**Exercise 37: Rename and Modify Foreign Key Constraints**

**Exercise:**

1. Create a contractors table with the following columns:
   * contractor\_id (INT, Primary Key, Auto Increment)
   * contractor\_name (VARCHAR(100), Not Null)
2. Create a contractor\_assignments table with the following columns:
   * contractor\_id (INT, Foreign Key references contractors(contractor\_id))
   * project\_id (INT, Foreign Key references projects(project\_id))
3. Rename the foreign key on contractor\_id to fk\_contractor\_assignment and add the ON DELETE CASCADE option.

**Exercise 38: Create, Drop, and Re-add Foreign Key with ON UPDATE CASCADE**

**Exercise:**

1. Create a consultants table with the following columns:
   * consultant\_id (INT, Primary Key, Auto Increment)
   * consultant\_name (VARCHAR(100), Not Null)
2. Create a consultant\_tasks table with the following columns:
   * consultant\_id (INT, Foreign Key references consultants(consultant\_id))
   * task\_id (INT, Foreign Key references tasks(task\_id))
3. Alter the consultant\_tasks table to:
   * Drop the current foreign key on consultant\_id.
   * Re-add the foreign key with ON UPDATE CASCADE.

**Exercise 39: Alter Multiple Foreign Keys Simultaneously**

**Exercise:**

1. Create a meetings table with the following columns:
   * meeting\_id (INT, Primary Key, Auto Increment)
   * meeting\_name (VARCHAR(100), Not Null)
   * client\_id (INT, Foreign Key references clients(client\_id))
2. Create a meeting\_attendees table with the following columns:
   * attendee\_id (INT, Primary Key, Auto Increment)
   * meeting\_id (INT, Foreign Key references meetings(meeting\_id))
   * emp\_id (INT, Foreign Key references employees(emp\_id))
3. Alter the meeting\_attendees table to:
   * Drop both foreign keys on meeting\_id and emp\_id.
   * Re-add both foreign keys with ON DELETE CASCADE for meeting\_id and ON UPDATE CASCADE for emp\_id.

**Exercise 40: Alter Columns to Add DEFAULT Values and NOT NULL Constraint**

**Exercise:**

1. Create a departments table with the following columns:
   * dept\_id (INT, Primary Key, Auto Increment)
   * dept\_name (VARCHAR(100), Not Null)
   * location (VARCHAR(50))
2. Alter the table to:
   * Add a NOT NULL constraint on the location column.
   * Add a default value of 'Unknown' for the location column.

**Exercise 41: Add and Drop a Unique Constraint**

**Exercise:**

1. Create a customers table with the following columns:
   * customer\_id (INT, Primary Key, Auto Increment)
   * email (VARCHAR(100), Not Null)
   * phone\_number (VARCHAR(15))
2. Alter the table to:
   * Add a **UNIQUE** constraint on the email column.
   * Later, drop the UNIQUE constraint on email.

**Exercise 42: Rename a Table and a Column**

**Exercise:**

1. Create a suppliers table with the following columns:
   * supplier\_id (INT, Primary Key, Auto Increment)
   * supplier\_name (VARCHAR(100), Not Null)
   * supplier\_location (VARCHAR(100))
2. Rename:
   * The table from suppliers to vendor\_suppliers.
   * The column supplier\_location to location.

**Exercise 43: Add and Drop an Index**

**Exercise:**

1. Create a product\_inventory table with the following columns:
   * product\_id (INT, Primary Key, Auto Increment)
   * product\_name (VARCHAR(100), Not Null)
   * price (DECIMAL(10,2))
2. Add an index on the price column to speed up queries, then later drop the index.

**Exercise 44: Truncate a Table**

**Exercise:**

1. Create a project\_logs table with the following columns:
   * log\_id (INT, Primary Key, Auto Increment)
   * project\_id (INT)
   * log\_message (TEXT)
2. Insert a few records into the table. Then, use the TRUNCATE command to remove all the records without deleting the table structure.

**Exercise 45: Drop a Column**

**Exercise:**

1. Create a vendor\_payments table with the following columns:
   * payment\_id (INT, Primary Key, Auto Increment)
   * vendor\_id (INT, Not Null)
   * payment\_date (DATE, Not Null)
   * payment\_method (VARCHAR(50))
2. Drop the payment\_method column from the table.

**Exercise 46: Add Multiple Foreign Keys and Drop One**

**Exercise:**

1. Create a client\_contracts table with the following columns:
   * contract\_id (INT, Primary Key, Auto Increment)
   * client\_id (INT, Foreign Key references clients(client\_id))
   * project\_id (INT, Foreign Key references projects(project\_id))
2. Drop the foreign key constraint on the client\_id column while retaining the project\_id foreign key.

**Exercise 47: Alter a Column to Change Data Type and Size**

**Exercise:**

1. Create an orders table with the following columns:
   * order\_id (INT, Primary Key, Auto Increment)
   * customer\_name (VARCHAR(50))
   * order\_total (DECIMAL(10,2))
2. Modify the customer\_name column to increase its size to 100 characters.

**Exercise 48: Drop a Table**

**Exercise:**

1. Create a temp\_data table with the following columns:
   * temp\_id (INT, Primary Key, Auto Increment)
   * data\_value (VARCHAR(100))
2. Drop the entire table using the DROP command.

**Exercise 49: Add a Primary Key to an Existing Table**

**Exercise:**

1. Create a sales table with the following columns:
   * sale\_id (INT, Not Null)
   * sale\_amount (DECIMAL(10,2))
2. Alter the table to add a **primary key** on the sale\_id column.

**Exercise 50: Add and Drop a Foreign Key Constraint**

**Exercise:**

1. Create an order\_items table with the following columns:
   * item\_id (INT, Primary Key, Auto Increment)
   * order\_id (INT, Foreign Key references orders(order\_id))
2. Drop the foreign key constraint on the order\_id column.

**Exercise 51: Alter Table to Add and Drop Constraints**

**Exercise:**

1. Create a students table with the following columns:
   * student\_id (INT, Primary Key, Auto Increment)
   * first\_name (VARCHAR(50), Not Null)
   * last\_name (VARCHAR(50))
   * email (VARCHAR(100))
2. Perform the following tasks:
   * Add a UNIQUE constraint on the email column.
   * Add a CHECK constraint to ensure the student\_id is greater than 100.
   * Drop the UNIQUE constraint on email.

**Exercise 52: Alter Multiple Columns and Add Foreign Keys**

**Exercise:**

1. Create a books table with the following columns:
   * book\_id (INT, Primary Key, Auto Increment)
   * title (VARCHAR(100), Not Null)
   * author\_id (INT)
2. Create an authors table:
   * author\_id (INT, Primary Key, Auto Increment)
   * name (VARCHAR(100), Not Null)
3. Perform the following tasks:
   * Alter the books table to add a foreign key reference to authors.
   * Modify the title column to have a new size of 150 characters.
   * Add another column published\_year to books with the data type YEAR.

**Exercise 53: Rename Multiple Columns and Add Index**

**Exercise:**

1. Create an orders table with the following columns:
   * order\_id (INT, Primary Key, Auto Increment)
   * order\_date (DATE)
   * customer\_name (VARCHAR(100))
2. Perform the following tasks:
   * Rename the customer\_name column to client\_name.
   * Add an INDEX on the order\_date column for faster querying.
   * Rename the table to customer\_orders.

**Exercise 54: Truncate and Drop Table**

**Exercise:**

1. Create a temp\_sales table with the following columns:
   * temp\_id (INT, Primary Key, Auto Increment)
   * temp\_value (INT)
2. Insert some data into the table. Then:
   * Truncate the temp\_sales table.
   * Drop the temp\_sales table completely.

**Exercise 55: Add and Drop Composite Primary Key**

**Exercise:**

1. Create a student\_courses table with the following columns:
   * student\_id (INT, Not Null)
   * course\_id (INT, Not Null)
   * grade (CHAR(1))
2. Perform the following tasks:
   * Add a **composite primary key** on the student\_id and course\_id columns.
   * Drop the composite primary key.

**Exercise 56: Add Multiple Foreign Keys and Drop One**

**Exercise:**

1. Create a task\_assignments table with the following columns:
   * assignment\_id (INT, Primary Key, Auto Increment)
   * task\_id (INT, Foreign Key references tasks(task\_id))
   * employee\_id (INT, Foreign Key references employees(emp\_id))
2. Drop the foreign key on task\_id while retaining the foreign key on employee\_id.

**Exercise 57: Alter Multiple Columns in One Statement**

**Exercise:**

1. Create an invoices table with the following columns:
   * invoice\_id (INT, Primary Key, Auto Increment)
   * amount (DECIMAL(10, 2))
   * due\_date (DATE)
2. Alter the table to:
   * Modify the amount column to DECIMAL(12, 2).
   * Change the due\_date column to allow NULL values.

**Exercise 58: Drop and Add a New Primary Key**

**Exercise:**

1. Create a suppliers table with the following columns:
   * supplier\_id (INT)
   * supplier\_name (VARCHAR(100))
2. Perform the following tasks:
   * Add a **primary key** on the supplier\_id column.
   * Drop the primary key and add a new composite primary key on both supplier\_id and supplier\_name.

**Exercise 59: Add and Drop a Unique Constraint**

**Exercise:**

1. Create a projects table with the following columns:
   * project\_id (INT, Primary Key, Auto Increment)
   * project\_code (VARCHAR(50))
   * client\_id (INT)
2. Perform the following tasks:
   * Add a **UNIQUE** constraint on the project\_code column.
   * Drop the **UNIQUE** constraint from the project\_code column.

**Exercise 60: Rename Table and Drop a Column**

**Exercise:**

1. Create a payments table with the following columns:
   * payment\_id (INT, Primary Key, Auto Increment)
   * amount (DECIMAL(10, 2))
   * payment\_date (DATE)
   * payment\_status (VARCHAR(20))
2. Perform the following tasks:
   * Rename the table to customer\_payments.
   * Drop the payment\_status column.

**Exercise 61: Alter Table to Add Multiple Constraints and Modify Data Types**

**Exercise:**

1. Create the following tables:
   * employees:
     + emp\_id (INT, Primary Key, Auto Increment)
     + first\_name (VARCHAR(50), Not Null)
     + last\_name (VARCHAR(50))
     + department\_id (INT)
     + email (VARCHAR(100))
   * departments:
     + department\_id (INT, Primary Key, Auto Increment)
     + department\_name (VARCHAR(100), Not Null)
2. Perform the following tasks:
   * Add a **foreign key** constraint between employees.department\_id and departments.department\_id.
   * Add a **UNIQUE** constraint on the employees.email column.
   * Modify the last\_name column in employees to allow a maximum length of 80 characters.
   * Add a **CHECK** constraint that ensures the emp\_id in employees is greater than 1000.

**Exercise 62: Create Complex Tables and Modify Multiple Columns**

**Exercise:**

1. Create the following tables:
   * projects:
     + project\_id (INT, Primary Key, Auto Increment)
     + project\_name (VARCHAR(100))
     + start\_date (DATE)
     + client\_id (INT)
   * clients:
     + client\_id (INT, Primary Key, Auto Increment)
     + client\_name (VARCHAR(100))
2. Perform the following tasks:
   * Add a **foreign key** constraint between projects.client\_id and clients.client\_id.
   * Modify the project\_name column in projects to be of length 150.
   * Add a new column budget to projects with a data type of DECIMAL(15,2).
   * Add a **CHECK** constraint to ensure the budget is greater than 10,000.
   * Modify the start\_date column to allow NULL values.

**Exercise 63: Create Multiple Tables and Drop Constraints**

**Exercise:**

1. Create the following tables:
   * orders:
     + order\_id (INT, Primary Key, Auto Increment)
     + order\_date (DATE)
     + customer\_id (INT)
   * customers:
     + customer\_id (INT, Primary Key, Auto Increment)
     + customer\_name (VARCHAR(100))
   * products:
     + product\_id (INT, Primary Key, Auto Increment)
     + product\_name (VARCHAR(100))
     + price (DECIMAL(10, 2))
   * order\_details:
     + order\_id (INT, Foreign Key references orders)
     + product\_id (INT, Foreign Key references products)
     + quantity (INT)
2. Perform the following tasks:
   * Drop the **foreign key** constraint between order\_details.order\_id and orders.order\_id.
   * Drop the **foreign key** constraint between order\_details.product\_id and products.product\_id.
   * Drop the order\_details table.

**Exercise 64: Rename Table, Alter Multiple Columns, and Add Constraints**

**Exercise:**

1. Create the following table:
   * payments:
     + payment\_id (INT, Primary Key, Auto Increment)
     + amount (DECIMAL(10, 2))
     + payment\_date (DATE)
     + status (VARCHAR(20))
2. Perform the following tasks:
   * Rename the table to customer\_payments.
   * Add a new column customer\_id (INT).
   * Add a **foreign key** constraint between customer\_payments.customer\_id and customers.customer\_id.
   * Modify the amount column to allow up to 12 digits.
   * Add a **CHECK** constraint to ensure the amount is greater than 0.

**Exercise 65: Composite Foreign Keys and Renaming Columns**

**Exercise:**

1. Create the following tables:
   * tasks:
     + task\_id (INT, Primary Key, Auto Increment)
     + task\_name (VARCHAR(100))
     + assigned\_to (INT, Foreign Key referencing employees.emp\_id)
   * subtasks:
     + subtask\_id (INT, Primary Key, Auto Increment)
     + task\_id (INT, Foreign Key referencing tasks.task\_id)
     + subtask\_name (VARCHAR(100))
     + subtask\_assigned\_to (INT, Foreign Key referencing employees.emp\_id)
2. Perform the following tasks:
   * Add a **composite foreign key** between subtasks.task\_id and tasks.task\_id and subtasks.subtask\_assigned\_to and employees.emp\_id.
   * Rename the task\_name column in tasks to task\_description.

**Exercise 66: Project Management Schema**

**Exercise:**

1. Create the following tables:
   * **projects**:
     + project\_id (INT, Primary Key, Auto Increment)
     + project\_name (VARCHAR(100))
     + project\_manager (INT, Foreign Key referencing employees.emp\_id)
   * **tasks**:
     + task\_id (INT, Primary Key, Auto Increment)
     + task\_name (VARCHAR(100))
     + project\_id (INT, Foreign Key referencing projects.project\_id)
     + assigned\_to (INT, Foreign Key referencing employees.emp\_id)
2. Perform the following tasks:
   * Add a composite foreign key between tasks.project\_id and projects.project\_id.
   * Rename the project\_name column in projects to name.

**Exercise 67: School Database Schema**

**Exercise:**

1. Create the following tables:
   * **students**:
     + student\_id (INT, Primary Key, Auto Increment)
     + student\_name (VARCHAR(100))
     + class\_id (INT, Foreign Key referencing classes.class\_id)
   * **classes**:
     + class\_id (INT, Primary Key, Auto Increment)
     + class\_name (VARCHAR(100))
   * **enrollments**:
     + enrollment\_id (INT, Primary Key, Auto Increment)
     + student\_id (INT, Foreign Key referencing students.student\_id)
     + class\_id (INT, Foreign Key referencing classes.class\_id)
2. Perform the following tasks:
   * Add a composite foreign key between enrollments.student\_id and students.student\_id, and enrollments.class\_id and classes.class\_id.
   * Rename the class\_name column in classes to name.

**Exercise 68: E-commerce Order System**

**Exercise:**

1. Create the following tables:
   * **customers**:
     + customer\_id (INT, Primary Key, Auto Increment)
     + customer\_name (VARCHAR(100))
   * **orders**:
     + order\_id (INT, Primary Key, Auto Increment)
     + customer\_id (INT, Foreign Key referencing customers.customer\_id)
     + order\_date (DATE)
   * **order\_items**:
     + order\_item\_id (INT, Primary Key, Auto Increment)
     + order\_id (INT, Foreign Key referencing orders.order\_id)
     + product\_name (VARCHAR(100))
2. Perform the following tasks:
   * Add a composite foreign key between order\_items.order\_id and orders.order\_id.
   * Rename the customer\_name column in customers to name.

**Exercise 69: Library Management System**

**Exercise:**

1. Create the following tables:
   * **books**:
     + book\_id (INT, Primary Key, Auto Increment)
     + title (VARCHAR(100))
     + author\_id (INT, Foreign Key referencing authors.author\_id)
   * **authors**:
     + author\_id (INT, Primary Key, Auto Increment)
     + author\_name (VARCHAR(100))
   * **loans**:
     + loan\_id (INT, Primary Key, Auto Increment)
     + book\_id (INT, Foreign Key referencing books.book\_id)
     + borrower\_id (INT, Foreign Key referencing borrowers.borrower\_id)
   * **borrowers**:
     + borrower\_id (INT, Primary Key, Auto Increment)
     + borrower\_name (VARCHAR(100))
2. Perform the following tasks:
   * Add a composite foreign key between loans.book\_id and books.book\_id, and loans.borrower\_id and borrowers.borrower\_id.
   * Rename the title column in books to book\_title.

**Exercise 70: Inventory Management System**

**Exercise:**

1. Create the following tables:
   * **categories**:
     + category\_id (INT, Primary Key, Auto Increment)
     + category\_name (VARCHAR(100), UNIQUE)
   * **products**:
     + product\_id (INT, Primary Key, Auto Increment)
     + product\_name (VARCHAR(100))
     + category\_id (INT, Foreign Key referencing categories.category\_id)
     + stock\_quantity (INT CHECK (stock\_quantity >= 0))
     + price (DECIMAL(10, 2) CHECK (price >= 0))
2. Perform the following tasks:
   * Add a UNIQUE constraint on product\_name in products.
   * Rename category\_name in categories to name.
   * Add a new column supplier\_id to products (INT).
   * Alter the products table to set supplier\_id as NOT NULL.

**Exercise 71: Hotel Reservation System**

**Exercise:**

1. Create the following tables:
   * **hotels**:
     + hotel\_id (INT, Primary Key, Auto Increment)
     + hotel\_name (VARCHAR(100), UNIQUE)
     + location (VARCHAR(100))
   * **rooms**:
     + room\_id (INT, Primary Key, Auto Increment)
     + hotel\_id (INT, Foreign Key referencing hotels.hotel\_id)
     + room\_number (VARCHAR(10))
     + room\_type (VARCHAR(50))
     + price (DECIMAL(10, 2))
2. Perform the following tasks:
   * Add a composite unique constraint on hotel\_id and room\_number in rooms.
   * Rename room\_type in rooms to type.
   * Add a CHECK constraint on price to ensure it is positive.

**Exercise 72: Customer Feedback System**

**Exercise:**

1. Create the following tables:
   * **feedback\_types**:
     + feedback\_type\_id (INT, Primary Key, Auto Increment)
     + type\_name (VARCHAR(100), UNIQUE)
   * **feedbacks**:
     + feedback\_id (INT, Primary Key, Auto Increment)
     + customer\_name (VARCHAR(100))
     + feedback\_type\_id (INT, Foreign Key referencing feedback\_types.feedback\_type\_id)
     + comments (TEXT)
     + feedback\_date (DATE)
2. Perform the following tasks:
   * Add a foreign key constraint on feedback\_type\_id.
   * Rename customer\_name in feedbacks to name.
   * Create an index on feedback\_date in feedbacks.

**Exercise 73: Employee Performance Tracking**

**Exercise:**

1. Create the following tables:
   * **departments**:
     + department\_id (INT, Primary Key, Auto Increment)
     + department\_name (VARCHAR(100), UNIQUE)
   * **employees**:
     + employee\_id (INT, Primary Key, Auto Increment)
     + name (VARCHAR(100))
     + department\_id (INT, Foreign Key referencing departments.department\_id)
     + hire\_date (DATE)
     + salary (DECIMAL(10, 2))
   * **performance\_reviews**:
     + review\_id (INT, Primary Key, Auto Increment)
     + employee\_id (INT, Foreign Key referencing employees.employee\_id)
     + review\_date (DATE)
     + score (INT CHECK (score BETWEEN 1 AND 5))
2. Perform the following tasks:
   * Add a composite foreign key between performance\_reviews.employee\_id and employees.employee\_id.
   * Rename score in performance\_reviews to rating.
   * Add a CHECK constraint on salary to ensure it is greater than or equal to the minimum wage (assume 1000).

**Exercise 74: Conference Management System**

**Exercise:**

1. Create the following tables:
   * **conferences**:
     + conference\_id (INT, Primary Key, Auto Increment)
     + conference\_name (VARCHAR(100), UNIQUE)
     + date (DATE)
   * **attendees**:
     + attendee\_id (INT, Primary Key, Auto Increment)
     + attendee\_name (VARCHAR(100))
   * **registrations**:
     + registration\_id (INT, Primary Key, Auto Increment)
     + conference\_id (INT, Foreign Key referencing conferences.conference\_id)
     + attendee\_id (INT, Foreign Key referencing attendees.attendee\_id)
2. Perform the following tasks:
   * Add a composite foreign key between registrations.conference\_id and conferences.conference\_id, and registrations.attendee\_id and attendees.attendee\_id.
   * Rename conference\_name in conferences to name.
   * Create an index on date in conferences.

**Exercise 75: Online Learning Platform**

**Exercise:**

1. Create the following tables:
   * **courses**:
     + course\_id (INT, Primary Key, Auto Increment)
     + course\_title (VARCHAR(100), UNIQUE)
     + duration (INT CHECK (duration > 0)) -- duration in hours
   * **students**:
     + student\_id (INT, Primary Key, Auto Increment)
     + student\_name (VARCHAR(100))
   * **enrollments**:
     + enrollment\_id (INT, Primary Key, Auto Increment)
     + course\_id (INT, Foreign Key referencing courses.course\_id)
     + student\_id (INT, Foreign Key referencing students.student\_id)
     + enrollment\_date (DATE)
2. Perform the following tasks:
   * Add a composite foreign key between enrollments.course\_id and courses.course\_id, and enrollments.student\_id and students.student\_id.
   * Rename course\_title in courses to title.
   * Create a CHECK constraint on duration to ensure it is at least 1 hour.

**Exercise 76: Fitness Center Management**

**Exercise:**

1. Create the following tables:
   * **members**:
     + member\_id (INT, Primary Key, Auto Increment)
     + member\_name (VARCHAR(100))
     + membership\_start\_date (DATE)
     + membership\_type (VARCHAR(50))
   * **trainers**:
     + trainer\_id (INT, Primary Key, Auto Increment)
     + trainer\_name (VARCHAR(100))
     + specialization (VARCHAR(100))
   * **sessions**:
     + session\_id (INT, Primary Key, Auto Increment)
     + session\_date (DATE)
     + trainer\_id (INT, Foreign Key referencing trainers.trainer\_id)
     + member\_id (INT, Foreign Key referencing members.member\_id)
     + session\_duration (INT CHECK (session\_duration > 0)) -- duration in minutes
2. Perform the following tasks:
   * Add a UNIQUE constraint on membership\_type in members.
   * Rename specialization in trainers to expertise.
   * Create an index on session\_date in sessions.

**Exercise 77: E-commerce Order Management**

**Exercise:**

1. Create the following tables:
   * **customers**:
     + customer\_id (INT, Primary Key, Auto Increment)
     + customer\_name (VARCHAR(100))
     + email (VARCHAR(100) UNIQUE)
   * **orders**:
     + order\_id (INT, Primary Key, Auto Increment)
     + customer\_id (INT, Foreign Key referencing customers.customer\_id)
     + order\_date (DATE)
     + total\_amount (DECIMAL(10, 2) CHECK (total\_amount > 0))
   * **order\_items**:
     + order\_item\_id (INT, Primary Key, Auto Increment)
     + order\_id (INT, Foreign Key referencing orders.order\_id)
     + product\_name (VARCHAR(100))
     + quantity (INT CHECK (quantity > 0))
     + price (DECIMAL(10, 2) CHECK (price >= 0))
2. Perform the following tasks:
   * Add a composite foreign key between order\_items.order\_id and orders.order\_id.
   * Rename total\_amount in orders to amount\_due.
   * Create an index on customer\_id in orders.

**Exercise 78: Library Management System**

**Exercise:**

1. Create the following tables:
   * **books**:
     + book\_id (INT, Primary Key, Auto Increment)
     + title (VARCHAR(100))
     + author (VARCHAR(100))
     + isbn (VARCHAR(20) UNIQUE)
   * **members**:
     + member\_id (INT, Primary Key, Auto Increment)
     + member\_name (VARCHAR(100))
     + membership\_date (DATE)
   * **loans**:
     + loan\_id (INT, Primary Key, Auto Increment)
     + book\_id (INT, Foreign Key referencing books.book\_id)
     + member\_id (INT, Foreign Key referencing members.member\_id)
     + loan\_date (DATE)
     + return\_date (DATE)
2. Perform the following tasks:
   * Add a CHECK constraint on return\_date to ensure it is greater than or equal to loan\_date.
   * Rename title in books to book\_title.
   * Create an index on membership\_date in members.

**Exercise 79: Event Management System**

**Exercise:**

1. Create the following tables:
   * **events**:
     + event\_id (INT, Primary Key, Auto Increment)
     + event\_name (VARCHAR(100) UNIQUE)
     + event\_date (DATE)
   * **participants**:
     + participant\_id (INT, Primary Key, Auto Increment)
     + participant\_name (VARCHAR(100))
   * **registrations**:
     + registration\_id (INT, Primary Key, Auto Increment)
     + event\_id (INT, Foreign Key referencing events.event\_id)
     + participant\_id (INT, Foreign Key referencing participants.participant\_id)
     + registration\_date (DATE)
2. Perform the following tasks:
   * Add a composite foreign key between registrations.event\_id and events.event\_id, and registrations.participant\_id and participants.participant\_id.
   * Rename event\_name in events to name.
   * Create a CHECK constraint on event\_date to ensure it is not in the past.

**Exercise 80: Transportation Management System**

**Exercise:**

1. Create the following tables:
   * **vehicles**:
     + vehicle\_id (INT, Primary Key, Auto Increment)
     + vehicle\_type (VARCHAR(50))
     + license\_plate (VARCHAR(20) UNIQUE)
   * **drivers**:
     + driver\_id (INT, Primary Key, Auto Increment)
     + driver\_name (VARCHAR(100))
     + license\_number (VARCHAR(50) UNIQUE)
   * **trips**:
     + trip\_id (INT, Primary Key, Auto Increment)
     + vehicle\_id (INT, Foreign Key referencing vehicles.vehicle\_id)
     + driver\_id (INT, Foreign Key referencing drivers.driver\_id)
     + trip\_date (DATE)
     + trip\_duration (INT CHECK (trip\_duration > 0)) -- duration in minutes
2. Perform the following tasks:
   * Add a composite foreign key between trips.vehicle\_id and vehicles.vehicle\_id, and trips.driver\_id and drivers.driver\_id.
   * Rename vehicle\_type in vehicles to type.
   * Create an index on trip\_date in trips.