Assignment 01DDL

1. Create a table with a multi-column primary key and a unique constraint that spans three columns.
2. Design a table named EmployeeWorkHours that stores each employee's work hours. Ensure that the EmployeeID foreign key references an Employee table and if an employee record is deleted, their work hours should be set to NULL.
3. Create a table Products with a Price column that ensures all prices are greater than 0 and do not exceed 10,000. Also, add a default value of 1000 for the Price column.
4. Design a partitioned table SalesData that partitions based on a date column SaleDate. Partition the table into three partitions: sales before 2010, between 2010 and 2020, and after 2020.
5. Alter an existing table Orders by adding a new column OrderStatus that can only hold values 'Pending', 'Shipped', or 'Delivered'. Ensure it defaults to 'Pending'.
6. Create a table EmployeeSalary that tracks employees' salaries. Add a constraint to ensure that the Salary column only allows values between 30,000 and 200,000, and is always divisible by 100.
7. Create a table with a composite foreign key referencing two columns in another table Courses. The foreign key should involve StudentID and CourseID from the Enrollments table.
8. Create a table StudentCourses with a CourseID column that is both a primary key and foreign key referencing the Courses table. Make sure deleting a course cascades to this table as well.
9. Alter a table Departments to add a new column ManagerID that references the Employees table. Ensure that when an employee is deleted, their ManagerID value is set to NULL.
10. Create a view TopPerformers that lists employees who have a PerformanceScore greater than 90, grouped by department and sorted by score.
11. Add a unique constraint to a table Users that ensures both Username and Email together must be unique across all rows.
12. Create a table Contracts that stores the contract duration in months and ensures that the contract duration is a multiple of 6 (i.e., only 6, 12, 18 months, etc.).
13. Write an SQL query to create a Customers table that supports full-text search on columns FirstName, LastName, and Email.
14. Design a table OrderItems that includes a primary key for OrderID and ProductID, with a foreign key to a Products table. Ensure that when a product is deleted, any corresponding OrderItems entries are automatically deleted.
15. Create a table Inventory with a column StockCount and ensure that the value is non-negative and cannot exceed 1000. Add a trigger that prevents inserting or updating StockCount beyond these limits.
16. Write a DDL query to create a table Transactions with a column TransactionID (auto-increment), a column TransactionAmount (decimal), and a foreign key to a Users table. Ensure that a user's transaction records are deleted when they are deleted.
17. Create a table Events with an EventDate column. Partition the table into 4 partitions based on the YEAR(EventDate). Each partition should store events for 5 years, starting from 2000.
18. Alter a table Invoices to add a column TaxAmount, calculated as 10% of the TotalAmount column (use a generated column).

19. Create a table Employees with a PhoneNumber column that ensures phone numbers follow a specific format (e.g., +1-xxx-xxx-xxxx).
20. Create a table Enrollments with a composite key based on StudentID and CourseID, and add a check constraint to ensure that a student cannot enroll in more than 5 courses at once.
21. Design a table Sessions with a unique constraint that prevents two sessions from having the same combination of SessionDate, StartTime, and EndTime.
22. Alter an existing table Payments by adding a new column PaymentType that can only store values 'Credit Card', 'Debit Card', 'Bank Transfer'. Add a default value of 'Credit Card'.
23. Create a Projects table where each project can have one or more employees assigned. Use a Many-to-Many relationship with a ProjectAssignments table that links Projects and Employees.
24. Create a table ProductSales with a SaleDate and ProductID. Ensure the ProductID exists in the Products table. Also, partition this table based on the month of the SaleDate.
25. Write an SQL query to create a table Customers that includes a Gender column. Use an ENUM data type with values 'Male', 'Female', 'Other'.
26. Alter an existing table Students to add a new column GraduationDate, but ensure that the column only allows values greater than the current date.
27. Create a table CarRentals where each car can be rented multiple times. Ensure that there are no overlapping rentals for the same car (i.e., ensure no two rental periods overlap).
28. Design a table Departments where each department can only have one ManagerID. Use a foreign key to reference the Employees table, and add a constraint that ensures each department has only one manager.
29. Create a table Inventory that uses an index on the ProductName column for faster searches. Ensure that updates to the ProductName column are handled efficiently.
30. Design a table Flights that stores flight information and includes a column FlightDuration. Ensure that FlightDuration is calculated as the difference between ArrivalTime and DepartureTime using a generated column.