# SQL Commands and Explanations

## Problem 1: Domain Constraints

### a. Create Table Statement for Students

CREATE TABLE Students (  
 StudentID INT PRIMARY KEY,  
 Name VARCHAR(50) NOT NULL,  
 Age INT CHECK (Age BETWEEN 18 AND 25),  
 Grade DECIMAL(3, 1) CHECK (Grade BETWEEN 0.0 AND 10.0)  
);

### b. Insert Statements Violating Constraints

-- Violating Age constraint  
INSERT INTO Students (StudentID, Name, Age, Grade)  
VALUES (1, 'John Doe', 17, 8.5);  
  
-- Violating Grade constraint  
INSERT INTO Students (StudentID, Name, Age, Grade)  
VALUES (2, 'Jane Smith', 20, 12.0);

### c. Explanation of Errors

1. For the first INSERT statement, an error occurs because the Age value 17 is outside the allowed range (18 to 25).  
 Error Message: CHECK constraint failed: Students  
2. For the second INSERT statement, an error occurs because the Grade value 12.0 exceeds the allowed range (0.0 to 10.0).  
 Error Message: CHECK constraint failed: Students

## Problem 2: Entity Integrity Constraints

### a. Create Table Statement for Courses

CREATE TABLE Courses (  
 CourseID INT PRIMARY KEY,  
 CourseName VARCHAR(100),  
 Credits INT NOT NULL  
);

### b. Insert Statement Without CourseID

INSERT INTO Courses (CourseName, Credits)  
VALUES ('Mathematics', 3);

### c. Explanation of Entity Integrity

Entity integrity ensures that each row in a table is uniquely identifiable, which is critical for maintaining data consistency and preventing ambiguity in the database.

## Problem 3: Referential Integrity Constraints

### a. SQL Commands to Create Products and Orders

CREATE TABLE Products (  
 ProductID INT PRIMARY KEY,  
 ProductName VARCHAR(100)  
);  
  
CREATE TABLE Orders (  
 OrderID INT PRIMARY KEY,  
 ProductID INT,  
 OrderQuantity INT,  
 FOREIGN KEY (ProductID) REFERENCES Products(ProductID)  
);

### b. Insert Statement Violating Referential Integrity

INSERT INTO Orders (OrderID, ProductID, OrderQuantity)  
VALUES (1, 999, 5);

### c. Valid Insert Statements

INSERT INTO Products (ProductID, ProductName)  
VALUES (1, 'Laptop'), (2, 'Phone');  
  
INSERT INTO Orders (OrderID, ProductID, OrderQuantity)  
VALUES (1, 1, 2), (2, 2, 3);

## Problem 4: Key Constraints

### a. SQL Commands to Create Employees

CREATE TABLE Employees (  
 EmployeeID INT PRIMARY KEY,  
 Email VARCHAR(100) UNIQUE,  
 Name VARCHAR(50)  
);

### b. Insert Statements Violating Unique Constraint

INSERT INTO Employees (EmployeeID, Email, Name)  
VALUES (1, 'john@example.com', 'John Doe');  
  
INSERT INTO Employees (EmployeeID, Email, Name)  
VALUES (2, 'john@example.com', 'Jane Doe');

### c. Insert Statement Without Email

INSERT INTO Employees (EmployeeID, Name)  
VALUES (3, 'Mark Smith');

## Problem 5: Combination of Constraints

### a. SQL Commands to Create Accounts and Transactions

CREATE TABLE Accounts (  
 AccountID INT PRIMARY KEY,  
 MobileNumber VARCHAR(10) CHECK (LENGTH(MobileNumber) = 10),  
 Balance DECIMAL(10, 2) CHECK (Balance >= 0)  
);  
  
CREATE TABLE Transactions (  
 TransactionID INT PRIMARY KEY,  
 AccountID INT,  
 TransactionAmount DECIMAL(10, 2) CHECK (TransactionAmount <> 0),  
 FOREIGN KEY (AccountID) REFERENCES Accounts(AccountID)  
);

### b. Insert Statement Violating Referential Integrity

INSERT INTO Transactions (TransactionID, AccountID, TransactionAmount)  
VALUES (1, 999, 100.0);

### c. Valid Insert Statements

INSERT INTO Accounts (AccountID, MobileNumber, Balance)  
VALUES (1, '1234567890', 500.00);  
  
INSERT INTO Transactions (TransactionID, AccountID, TransactionAmount)  
VALUES (1, 1, 50.00);