**DDL exercises**

1. Create a table named "Employee" with columns "EmployeeID", "FirstName", "LastName", and "DepartmentID", where "EmployeeID" is the primary key.

Create Table with Primary Key Constraint:

CREATE TABLE Employee ( EmployeeID INT PRIMARY KEY, FirstName VARCHAR(50), LastName VARCHAR(50), DepartmentID INT );

1. Create a table named "Customer" with columns "CustomerID", "FirstName", "LastName", and "Email", ensuring that "CustomerID" and "Email" are unique.

Create Table with Unique Constraint:

CREATE TABLE Customer ( CustomerID INT UNIQUE, FirstName VARCHAR(50), LastName VARCHAR(50), Email VARCHAR(100) UNIQUE );

1. Create a table named "Product" with columns "ProductID", "ProductName", "UnitPrice", and "UnitsInStock", where "ProductName" and "UnitsInStock" cannot be NULL.

Create Table with Not Null Constraint:

CREATE TABLE Product ( ProductID INT PRIMARY KEY, ProductName VARCHAR(100) NOT NULL, UnitPrice DECIMAL(10, 2), UnitsInStock INT NOT NULL );

1. Create a table named "Orders" with columns "OrderID", "OrderDate", and "TotalAmount", ensuring that "TotalAmount" is non-negative.

Create Table with Check Constraint:

CREATE TABLE Orders ( OrderID INT PRIMARY KEY, OrderDate DATE, TotalAmount DECIMAL(10, 2), CONSTRAINT CHK\_TotalAmount CHECK (TotalAmount >= 0) );

Or

CREATE TABLE Orders ( OrderID INT PRIMARY KEY, OrderDate DATE, TotalAmount DECIMAL(10, 2) CHECK (TotalAmount >= 0) );

1. Create a table named "OrderDetails" with columns "OrderDetailID", "OrderID", "ProductID", and "Quantity", where "OrderID" and "ProductID" are foreign keys referencing other tables.

Create Table with Foreign Key Constraint:

CREATE TABLE OrderDetails ( OrderDetailID INT PRIMARY KEY, OrderID INT, ProductID INT, Quantity INT, FOREIGN KEY (OrderID) REFERENCES Orders(OrderID), FOREIGN KEY (ProductID) REFERENCES Products(ProductID) );

Or

CREATE TABLE OrderDetails (

OrderDetailID INT PRIMARY KEY,

OrderID INT,

ProductID INT,

Quantity INT,

CONSTRAINT FK1 FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),

CONSTRAINT FK2 FOREIGN KEY (ProductID) REFERENCES Products(ProductID) );

1. Alter the "Employee" table to add a primary key constraint on the "EmployeeID" column.

Alter Table to Add Primary Key Constraint:

ALTER TABLE Employee ADD CONSTRAINT PK\_EmployeeID PRIMARY KEY (EmployeeID);

1. Alter the "Customer" table to drop the unique constraint on the "Email" column.

Drop Unique Constraint:

ALTER TABLE Customer DROP CONSTRAINT UQ\_Email;

Note: For dropping an individual unnamed default constrain on a column, you need to write code.

To add the unique constraint on the "Email" column, we write:

ALTER TABLE Customer ADD CONSTRAINT UQ\_Email UNIQUE (Email);

1. Alter the "Product" table to allow NULL values in the "ProductName" column.

Alter Table to Add Not Null Constraint:

ALTER TABLE Product ALTER COLUMN ProductName VARCHAR(100) NOT NULL;

1. Add a check constraint to the "Orders" table to ensure that the "OrderDate" is not in the future.

Alter Table to Add Check Constraint:

ALTER TABLE Orders ADD CONSTRAINT CHK\_OrderDate CHECK (OrderDate <= GETDATE());

1. Add a foreign key constraint to the "OrderDetails" table referencing the "Orders" table.

Alter Table to Add Foreign Key Constraint:

ALTER TABLE OrderDetails ADD CONSTRAINT FK\_OrderID FOREIGN KEY (OrderID) REFERENCES Orders(OrderID);

1. Drop the primary key constraint from the "Employee" table.

Drop Primary Key Constraint:

ALTER TABLE Employee DROP CONSTRAINT PK\_EmployeeID;

1. Drop the check constraint on the "OrderDate" column in the "Orders" table.

Drop Check Constraint:

ALTER TABLE Orders DROP CONSTRAINT CHK\_OrderDate;

1. Drop the foreign key constraint on the "OrderID" column in the "OrderDetails" table.

Drop Foreign Key Constraint:

ALTER TABLE OrderDetails DROP CONSTRAINT FK\_OrderID;

1. Truncate all records from the "Employee" table.

Truncate Table:

TRUNCATE TABLE Employee;

1. Create a table named "OrderDetails" with a composite primary key consisting of "OrderID" and "ProductID".

Create Table with Composite Primary Key Constraint:

CREATE TABLE OrderDetails ( OrderID INT, ProductID INT, Quantity INT, PRIMARY KEY (OrderID, ProductID) );

Or

CREATE TABLE OrderDetails (

OrderID INT,

ProductID INT,

Quantity INT,

CONSTRAINT PK PRIMARY KEY (OrderID, ProductID));

1. Create a unique index on the "CustomerID" column in the "Customer" table.

Create Table with Unique Index Constraint:

CREATE TABLE Customer ( CustomerID INT, FirstName VARCHAR(50), LastName VARCHAR(50), Email VARCHAR(100), CONSTRAINT UQ\_CustomerID UNIQUE(CustomerID), CONSTRAINT UQ\_Email UNIQUE(Email) );

1. Add a composite primary key constraint on the "OrderDetails" table for "OrderID" and "ProductID".

Alter Table to Add Composite Primary Key Constraint:

ALTER TABLE OrderDetails ADD CONSTRAINT PK\_OrderDetails PRIMARY KEY (OrderID, ProductID);

1. Create a table named "Employee" with a default value of 1 for the "DepartmentID" column.

Create Table with Default Constraint:

CREATE TABLE Employee ( EmployeeID INT PRIMARY KEY, FirstName VARCHAR(50), LastName VARCHAR(50), DepartmentID INT DEFAULT 1 );

1. Add a default constraint on the "DepartmentID" column in the "Employee" table.

Alter Table to Add Default Constraint:

ALTER TABLE Employee ADD CONSTRAINT DF\_DepartmentID DEFAULT 1 FOR DepartmentID;

1. Drop the default constraint on the "DepartmentID" column in the "Employee" table.

Drop Default Constraint:

ALTER TABLE Employee DROP CONSTRAINT DF\_DepartmentID;

1. Drop the identity constraint on the "ProductID" column in the "Product" table.

Drop Identity Constraint:

ALTER TABLE Product DROP CONSTRAINT DF\_ProductID;

1. Create a table named "Sales" with a unique constraint on the combination of "OrderID" and "ProductID".

Create Table with Unique Constraint on Multiple Columns:

CREATE TABLE Sales ( OrderID INT, ProductID INT, SaleDate DATE, Quantity INT, CONSTRAINT UQ\_OrderProduct UNIQUE (OrderID, ProductID) );

1. Create a table named "Student" with a check constraint to allow only grades 'A', 'B', 'C', 'D', or 'F'.

Create Table with Check Constraint for Enumerated Values:

CREATE TABLE Student ( StudentID INT PRIMARY KEY, FirstName VARCHAR(50), LastName VARCHAR(50), Grade CHAR(1), CONSTRAINT CHK\_Grade CHECK (Grade IN ('A', 'B', 'C', 'D', 'F')) );

1. Add a check constraint on the "Grade" column in the "Student" table.

ALTER TABLE Student ADD CONSTRAINT CHK\_Grade CHECK (Grade > 0);

**DML exercise**

Create table STUDENTS having attributes: StudentID (primary key), FirstName (can’t be null, can’t have more than 50 characters), LastName (can’t be null, can’t have more than 50 characters), Age (must be 18 or more), GPA (not negative and not more than 4, can have 3 digits out of which 2 digits can be decimal point), and Major (can’t have more than 50 characters).

Table 1: Students

CREATE TABLE Students (

StudentID INT PRIMARY KEY,

FirstName VARCHAR(50) NOT NULL,

LastName VARCHAR(50) NOT NULL,

Age INT CHECK (Age >= 18),

GPA DECIMAL(3, 2) CHECK (GPA BETWEEN 0 AND 4),

Major VARCHAR(100)

);

Table 2: Courses

Create table Courses having attributes: CourseID (primary key), CourseName (can’t be null, can’t have more than 100 characters), Instructor (can’t have more than 100 characters), Credits (must be positive), Department (can’t have more than 100 characters).

CREATE TABLE Courses (

CourseID INT PRIMARY KEY,

CourseName VARCHAR(100) NOT NULL,

Instructor VARCHAR(100),

Credits INT CHECK (Credits > 0),

Department VARCHAR(100)

);

INSERT INTO Students (StudentID, FirstName, LastName, Age, GPA, Major) VALUES

(1, 'John', 'Doe', 20, 3.5, 'Computer Science'),

(2, 'Jane', 'Smith', 22, 3.8, 'Biology'),

(3, 'Alex', 'Johnson', 21, 3.2, 'Mathematics'),

(4, 'Emily', 'Williams', 19, 3.9, 'Physics'),

(5, 'Chris', 'Brown', 23, 3.6, 'Chemistry'),

(6, 'Sarah', 'Davis', 20, 3.7, 'English'),

(7, 'Michael', 'Taylor', 21, 3.4, 'History'),

(8, 'Amanda', 'Anderson', 22, 3.1, 'Psychology');

INSERT INTO Courses (CourseID, CourseName, Instructor, Credits, Department) VALUES

(101, 'Introduction to Programming', 'Dr. Smith', 3, 'Computer Science'),

(102, 'Biology 101', 'Dr. Johnson', 4, 'Biology'),

(103, 'Calculus I', 'Prof. Brown', 4, 'Mathematics'),

(104, 'Physics Fundamentals', 'Dr. Williams', 3, 'Physics'),

(105, 'Organic Chemistry', 'Prof. Davis', 4, 'Chemistry'),

(106, 'Literature Survey', 'Dr. Taylor', 3, 'English'),

(107, 'World History', 'Prof. Anderson', 3, 'History'),

(108, 'Introduction to Psychology', 'Dr. White', 3, 'Psychology');

1. How would you update the GPA of student with ID 3 to 3.5?

UPDATE Students SET GPA = 3.5 WHERE StudentID = 3;

1. How can you update the instructor of the course with ID 102 to 'Prof. Adams'?

UPDATE Courses SET Instructor = 'Prof. Adams' WHERE CourseID = 102;

1. How would you delete the record of the student with ID 7?

DELETE FROM Students WHERE StudentID = 7;

1. How can you delete the course with ID 108 from the Courses table?

DELETE FROM Courses WHERE CourseID = 108;

1. How would you update the Major of the student with ID 6 to 'Communications'?

UPDATE Students SET Major = 'Communications' WHERE StudentID = 6;

1. How can you update the Credits of the course with ID 101 to 4?

UPDATE Courses SET Credits = 4 WHERE CourseID = 101;

1. How would you delete all records of students with a GPA less than 3.0?

DELETE FROM Students WHERE GPA < 3.0;

1. How can you delete the course with ID 105 from the Courses table?

DELETE FROM Courses WHERE CourseID = 105;

1. How would you update the Instructor of the course with ID 104 to 'Prof. Smith'?

UPDATE Courses SET Instructor = 'Prof. Smith' WHERE CourseID = 104;

1. How can you update the Department of the course with ID 107 to 'Social Studies'?

UPDATE Courses SET Department = 'Social Studies' WHERE CourseID = 107;

1. How would you delete the record of the student with ID 8?

DELETE FROM Students WHERE StudentID = 8;

1. How can you delete the course with ID 103 from the Courses table?

DELETE FROM Courses WHERE CourseID = 103;

1. How would you update the Age of the student with ID 1 to 21?

UPDATE Students SET Age = 21 WHERE StudentID = 1;

1. How can you update the Instructor of the course with ID 106 to 'Dr. Johnson'?

UPDATE Courses SET Instructor = 'Dr. Johnson' WHERE CourseID = 106;

1. How would you delete all records of students with a Major of 'History'?

DELETE FROM Students WHERE Major = 'History';

1. How can you update the Credits of all courses taught by 'Dr. Smith' to 5?

UPDATE Courses SET Credits = 5 WHERE Instructor = 'Dr. Smith';

1. How can you update the Age of students older than 20 to 25?

UPDATE Students SET Age = 25 WHERE Age > 20;

1. Update the Instructor of all courses with a Credits value greater than 3 to 'Dr. White'?

UPDATE Courses SET Instructor = 'Dr. White' WHERE Credits > 3;

1. Update Age to 23 and GPA to 3.8 of Student with ID 1 if GPA is Less than 3.5 or Age is Greater than 22:

UPDATE Students SET Age = 23, GPA = 3.8 WHERE StudentID = 1 AND (GPA < 3.5 OR Age > 22);

1. Update Credits to 4 and Instructor of Courses (to Prof. Adams) with Department 'Computer Science' if Credits are Less than 4 or Instructor is 'Dr. Smith':

UPDATE Courses SET Credits = 4, Instructor = 'Prof. Adams' WHERE Department = 'Computer Science' AND (Credits < 4 OR Instructor = 'Dr. Smith');

1. Update Instructor to Prof. White and Department to Physics of Courses if Credits are Greater than 3 and Department is 'Mathematics':

UPDATE Courses SET Instructor = 'Prof. White', Department = 'Physics' WHERE Credits > 3 AND Department = 'Mathematics';

1. Update Major to Chemistry and GPA to 3,8 of Students if Major is 'Biology' and GPA is Greater than 3.5:

UPDATE Students SET Major = 'Chemistry', GPA = 3.8 WHERE Major = 'Biology' AND GPA > 3.5;

1. Update CourseName of Courses whose Instructor's name contains 'Brown':

UPDATE Courses SET CourseName = 'Advanced Biology' WHERE Instructor LIKE '%Brown%';

1. Update Major of Students whose LastName ends with 'son' and Age is less than 25:

UPDATE Students SET Major = 'History' WHERE LastName LIKE '%son' AND Age < 25;