Exercises: Query Basics – Update and Delete

You can check your solutions here: https://judge.softuni.org/Contests/3130/Query-Basics-Update-Delete.

Problem 1. Create Database

You now know how to create database using the GUI of the SSMS. Now it's time to create it using SQL queries. In that task (and the several following it) you will be required to create the database from the previous exercise **using only SQL queries**. Firstly, just **create new database named Minions.**

Problem 2. Create Tables

In the newly created database Minions add table **Minions (Id, Name, Age)**. Then add new table **Towns (Id, Name).** Set **Id** columns of both tables to be **primary key** as **constraint**.

Problem 3. Alter Minions Table

Change the structure of the Minions table to have **new column TownId** that would be of the same type as the **Id** column of **Towns table**. Add **new constraint** that makes **TownId foreign key** and references to **Id** column of **Towns** table.

Problem 4. Insert Records in Both Tables

Populate both tables with sample records given in the table below.

Minions			
Id	**Name**	**Age**	**TownId**
1	Kevin	22	1
2	Bob	15	3
3	Steward	NULL	2

Towns	
Id	**Name**
1	Sofia
2	Plovdiv
3	Varna

Use only SQL queries. Insert the Id manually (don't use `identity`).

Problem 5. Truncate Table Minions

Delete all the data from the Minions table using **SQL query.**

Problem 6. Drop All Tables

Delete all tables from the Minions database using **SQL query**.

Problem 7. Create Table People

Using **SQL query** create table **People** with columns:

- **Id** – unique number for every person there will be **no more than 2^{31}-1 people.** (Auto incremented)
- **Name** – full name of the person will be **no more than 200 Unicode characters**. (Not null)
- **Picture** – image with **size up to 2 MB.** (Allow nulls)
- **Height** – In meters. Real number precise up to **2 digits** after floating point. (Allow nulls)
- **Weight** – In kilograms. Real number precise up to **2 digits** after floating point. (Allow nulls)

- **Gender** – Possible states are **m** or **f.** (Not null)
- **Birthdate –** (Not null)
- **Biography** – detailed biography of the person it can contain **max allowed Unicode characters.** (Allow nulls)

Make **Id** primary key. Populate the table with only **5 records**. Submit your **CREATE** and **INSERT statements** as Run queries & check DB.

Problem 8. Create Table Users

Using **SQL query** create table **Users** with columns:

- **Id** – unique number for every user. There will be **no more than 2^{63}-1 users.** (Auto incremented)
- **Username** – unique identifier of the user will be **no more than 30 characters (non Unicode).** (Required)
- **Password** – password will be **no longer than 26 characters (non Unicode).** (Required)
- **ProfilePicture** – image with **size up to 900 KB.**
- **LastLoginTime**
- **IsDeleted** – shows if the user deleted his/her profile. Possible states are **true** or **false**.

Make **Id** primary key. Populate the table with exactly **5 records**. Submit your **CREATE** and **INSERT statements** as Run queries & check DB.

Problem 9. Change Primary Key

Using **SQL queries** modify table **Users** from the previous task. First **remove current primary key** then create **new primary key** that would be the **combination** of fields **Id** and **Username**.

Problem 10. Add Check Constraint

Using **SQL queries** modify table **Users**. Add **check constraint** to ensure that the values in the Password field are **at least 5 symbols** long.

Problem 11. Set Default Value of a Field

Using **SQL queries** modify table **Users**. Make the **default value** of **LastLoginTime** field to be the **current time.**

Problem 12. Set Unique Field

Using **SQL queries** modify table **Users**. Remove **Username** field from the primary key so only the field **Id** would be primary key. Now **add unique constraint** to the **Username** field to ensure that the values there are **at least 3 symbols** long.

Problem 13. Movies Database

Using **SQL queries** create **Movies** database with the following entities:

- **Directors** (Id, DirectorName, Notes)
- **Genres** (Id, GenreName, Notes)
- **Categories** (Id, CategoryName, Notes)
- **Movies** (Id, Title, DirectorId, CopyrightYear, Length, GenreId, CategoryId, Rating, Notes)

Set most **appropriate data types** for each column. **Set primary key** to each table. Populate each table with exactly **5 records**. Make sure the columns that are present in 2 tables would be of the **same data type**. Consider which fields

are always required and which are optional. Submit your **CREATE TABLE** and **INSERT statements** as Run queries & check DB.

Problem 14. Car Rental Database

Using **SQL queries** create **CarRental** database with the following entities:

- **Categories** (Id, CategoryName, DailyRate, WeeklyRate, MonthlyRate, WeekendRate)
- **Cars** (Id, PlateNumber, Manufacturer, Model, CarYear, CategoryId, Doors, Picture, Condition, Available)
- **Employees** (Id, FirstName, LastName, Title, Notes)
- **Customers** (Id, DriverLicenceNumber, FullName, Address, City, ZIPCode, Notes)
- **RentalOrders** (Id, EmployeeId, CustomerId, CarId, TankLevel, KilometrageStart, KilometrageEnd, TotalKilometrage, StartDate, EndDate, TotalDays, RateApplied, TaxRate, OrderStatus, Notes)

Set most **appropriate data types** for each column. **Set primary key** to each table. Populate each table with only **3 records**. Make sure the columns that are present in 2 tables would be of the **same data type**. Consider which fields are always required and which are optional. Submit your **CREATE TABLE** and **INSERT statements** as Run queries & check DB.

Problem 15. Hotel Database

Using **SQL queries** create **Hotel** database with the following entities:

- **Employees** (Id, FirstName, LastName, Title, Notes)
- **Customers** (AccountNumber, FirstName, LastName, PhoneNumber, EmergencyName, EmergencyNumber, Notes)
- **RoomStatus** (RoomStatus, Notes)
- **RoomTypes** (RoomType, Notes)
- **BedTypes** (BedType, Notes)
- **Rooms** (RoomNumber, RoomType, BedType, Rate, RoomStatus, Notes)
- **Payments** (Id, EmployeeId, PaymentDate, AccountNumber, FirstDateOccupied, LastDateOccupied, TotalDays, AmountCharged, TaxRate, TaxAmount, PaymentTotal, Notes)
- **Occupancies** (Id, EmployeeId, DateOccupied, AccountNumber, RoomNumber, RateApplied, PhoneCharge, Notes)

Set most **appropriate data types** for each column. **Set primary key** to each table. Populate each table with only **3 records**. Make sure the columns that are present in 2 tables would be of the **same data type**. Consider which fields are always required and which are optional. Submit your **CREATE TABLE** and **INSERT statements** as Run queries & check DB.

Problem 16. Increase Employees Salary

Use **SoftUni** database and **increase the salary** of all employees by **10%.** Then show **only Salary** column for all in the **Employees** table. Submit your query statements as Prepare DB & Run queries.

Problem 17. Decrease Tax Rate

Use **Hotel** database and **decrease tax rate by 3%** to all payments. Then select **only TaxRate** column from the **Payments** table. Submit your query statements as Prepare DB & Run queries.

Problem 18. Delete All Records

Use **Hotel** database and **delete all records** from the **Occupancies** table. Use SQL query. Submit your query statements as Run skeleton, run queries & check DB.