# 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 231-1** **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.

CREATE TABLE People (

Id INT NOT NULL IDENTITY(1,1) PRIMARY KEY,

[Full Name] NVARCHAR(200) NOT NULL ,

Picture VARBINARY(2000),

Height DECIMAL(8,2),

[Weight] DECIMAL(8,2),

Gender CHAR(1) NOT NULL,

Birthday DATE NOT NULL,

Biography NVARCHAR(MAX),

)

INSERT INTO [dbo].[People] ([Full Name], [Height], [Weight], Gender, Birthday, Biography)

VALUES ('Bakoi Georgiev', 1.76, 71, 'm', '1987-10-2', 'Skilled like sharp knife'),

('Dancho Keca', 1.96, 100, 'm', '1991-11-8', NULL),

('Iveta Ilieva', 1.70, 62, 'f', '1994-5-12', 'Very good skater girl'),

('Irina Peteva', null, NULL, 'f', '1993-3-8', 'Good piano skills'),

('Stefan Ivanov', 1.83, 83, 'm', '1990-6-14', 'Skilled and goal oriented')

# **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 263-1 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)
* **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.

CREATE TABLE Users (

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

Username VARCHAR(30) UNIQUE NOT NULL,

Password VARCHAR(30) NOT NULL,

LastLogIn smalldatetime NOT NULL,

IsDeleted BIT

)

INSERT INTO dbo.Users

VALUES ('Someone', 'parolata', '2017-8-19', 0),

('Keke', 'nqmam', '2018-5-19', 1),

('Ikardi', 'ashdh', '2018-3-3', 1),

('Olele', 'oleliq', '2017-3-10', 1),

('Acera', 'tuzle', '2018-2-11', 1)

# **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**.

ALTER TABLE Users

DROP CONSTRAINT PK\_\_Users\_\_3214EC07E26E52A8

ALTER TABLE Users

ADD PRIMARY KEY (Id, 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.

ALTER TABLE Users

# ADD CONSTRAINT CHK\_PasswordLen CHECK (DATALENGTH([Password]) > 5)

# **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.**

# ALTER TABLE Users

# ADD CONSTRAINT DF\_LastLoginTime DEFAULT GETDATE() FOR LastLogIn;

# **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.

ALTER TABLE Users

DROP CONSTRAINT PK\_\_Users\_\_7722245988E34500

# **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.

CREATE TABLE Directors (

Id INT PRIMARY KEY NOT NULL IDENTITY(1,1),

DirectorName NVARCHAR(50) NOT NULL,

Notes NVARCHAR(MAX),

)

CREATE TABLE Genres (

Id INT PRIMARY KEY NOT NULL IDENTITY(1,1),

GengreName NVARCHAR(20) NOT NULL,

Notes NVARCHAR(MAX)

)

CREATE TABLE Categories (

Id INT PRIMARY KEY NOT NULL IDENTITY(1,1),

CategoryName NVARCHAR(20) NOT NULL,

Notes NVARCHAR(MAX)

)

CREATE TABLE Movies (

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

Title NVARCHAR(30),

CopyrightYear DATE,

[Length] INT,

Rating CHAR(1),

Notes NVARCHAR(MAX)

)

ALTER TABLE Movies

ADD DirectorId INT

ALTER TABLE Movies

ADD CONSTRAINT FK\_DirectorId

FOREIGN KEY (DirectorId) REFERENCES Directors(Id)

ALTER TABLE Movies

ADD CategoryId INT

ALTER TABLE Movies

ADD CONSTRAINT FK\_Category

FOREIGN KEY (CategoryId) REFERENCES Categories(Id)

# **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 o ptional. Submit your **CREATE TABLE** and **INSERT statements** as Run queries & check DB.

CREATE DATABASE CarRental

CREATE TABLE Categories (

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

CategoryName NVARCHAR(20),

DailyRate DECIMAL(8,2) NOT NULL,

WeeklyRate DECIMAL(8,2),

MonthlyRate DECIMAL(8,2) NOT NULL,

WeekendRate DECIMAL(8,2),

)

CREATE TABLE Cars (

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

PlateNumber NVARCHAR(15) NOT NULL,

Manifacturer NVARCHAR(10) NOT NULL,

Model NVARCHAR(10) NOT NULL,

CarYear SMALLINT NOT NULL,

CategoryId INT FOREIGN KEY REFERENCES Categories(Id),

Doors TINYINT,

Condition NVARCHAR(100) NOT NULL,

Available BIT NOT NULL

)

CREATE TABLE Employees (

Id INT PRIMARY KEY IDENTITY(1,1) NOT NULL,

FirstName NVARCHAR(20) NOT NULL,

LastName NVARCHAR(20) NOT NULL,

Title NVARCHAR(20) NOT NULL,

Notes NVARCHAR(20)

)

CREATE TABLE Customers (

Id INT PRIMARY KEY NOT NULL IDENTITY(1,1),

DriverLicenceNumber INT NOT NULL UNIQUE,

FullName NVARCHAR(50) NOT NULL,

[Address] NVARCHAR(50),

City NVARCHAR(20),

ZIPCode INT,

Notes NVARCHAR(MAX)

)

CREATE TABLE RentalOrders (

Id INT NOT NULL PRIMARY KEY IDENTITY(1,1),

EmployeeId INT FOREIGN KEY REFERENCES Employees(Id),

CustomerId INT FOREIGN KEY REFERENCES Customers(Id),

CarId INT FOREIGN KEY REFERENCES Cars(Id),

TankLevel SMALLINT,

KilometrageStart INT NOT NULL,

KilometrageEnd INT NOT NULL,

TotalKilometrage INT,

StartDate DATE NOT NULL,

EndDate DATE NOT NULL,

TotalDays TINYINT NOT NULL,

RateApplied CHAR(1),

TaxRate TINYINT,

OrderStatus NVARCHAR(20) NOT NULL,

Notes NVARCHAR(MAX),

)

# **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.

CREATE DATABASE Hotel

CREATE TABLE Employees (

Id INT IDENTITY(1,1) PRIMARY KEY NOT NULL,

FirstName NVARCHAR(20) NOT NULL,

LastName NVARCHAR(20) NOT NULL,

Title NVARCHAR(15),

Notes NVARCHAR(MAX)

)

CREATE TABLE Customers (

AccountNumber INT NOT NULL,

FirstName NVARCHAR(20) NOT NULL,

LastName NVARCHAR(20) NOT NULL,

PhoneNumber INT UNIQUE,

EmergencyName NVARCHAR(30)

)

CREATE TABLE RoomStatus (

RoomStatus NVARCHAR(20) UNIQUE NOT NULL,

Notes NVARCHAR(MAX)

)

CREATE TABLE RoomTypes (

RoomType NVARCHAR(20) UNIQUE NOT NULL,

Notes NVARCHAR(MAX)

)

CREATE TABLE BedTypes (

BedType NVARCHAR(20) UNIQUE NOT NULL,

Notes NVARCHAR(MAX)

)

CREATE TABLE Rooms (

RoomNumber SMALLINT NOT NULL,

RoomType NVARCHAR(20) FOREIGN KEY REFERENCES RoomTypes(RoomType),

BedType NVARCHAR(20) FOREIGN KEY REFERENCES BedTypes(BedType),

Rate SMALLINT,

RoomStatus NVARCHAR(20) FOREIGN KEY REFERENCES RoomStatus(RoomStatus),

Note NVARCHAR(MAX)

)

CREATE TABLE Payments (

Id INT NOT NULL PRIMARY KEY IDENTITY(1,1),

EmployeeId INT FOREIGN KEY REFERENCES Employees(Id),

PaymentDate DATE,

AccountNumber INT NOT NULL,

FirstDateOccupied DATE,

LastDateOccupied DATE,

TotalDays SMALLINT,

AmountCharged MONEY NOT NULL,

TaxRate SMALLINT,

TaxAmoun INT NOT NULL,

PaymentTotal MONEY NOT NULL,

Notes NVARCHAR(MAX)

)

CREATE TABLE Occupancies (

Id INT NOT NULL PRIMARY KEY IDENTITY(1,1),

EmployeeId INT FOREIGN KEY REFERENCES Employees(Id),

DateOccupied DATE,

AccountNumber INT NOT NULL UNIQUE,

RoomNumber INT NOT NULL,

RateApplied MONEY,

PhoneCharge MONEY,

Notes NVARCHAR(MAX)

)

# Problem 16: Create SoftUni Database

Now create bigger database called **SoftUni.** You will use same database in the future tasks. It should hold information about

* **Towns** (Id, Name)
* **Addresses** (Id, AddressText, TownId)
* **Departments** (Id, Name)
* **Employees** (Id, FirstName, MiddleName, LastName, JobTitle, DepartmentId, HireDate, Salary, AddressId)
* **Id** columns are **auto incremented** starting from 1 and increased by 1 (1, 2, 3, 4…). Make sure you **use appropriate data types** for each column. Add **primary** and **foreign keys** **as constraints** for each table. Use **only SQL queries**. Consider which fields are always required and which are optional.

CREATE DATABASE SoftUni

CREATE TABLE Towns (

Id INT NOT NULL PRIMARY KEY IDENTITY(1,1),

Name NVARCHAR(25) NOT NULL,

)

CREATE TABLE Adresses (

Id INT NOT NULL PRIMARY KEY IDENTITY(1,1),

AdressText NVARCHAR(50) NOT NULL,

TownId INT FOREIGN KEY REFERENCES Towns(Id),

)

CREATE TABLE Departments (

Id INT NOT NULL PRIMARY KEY IDENTITY(1,1),

[Name] NVARCHAR(20) NOT NULL,

)

CREATE TABLE Employees (

Id INT NOT NULL PRIMARY KEY IDENTITY(1,1),

FirstName NVARCHAR(20) NOT NULL,

MiddleName NVARCHAR(20),

LastName NVARCHAR(20),

JobTitle NVARCHAR(20),

DepartmentId INT FOREIGN KEY REFERENCES Departments(Id),

HireDate DATE,

Salary MONEY,

AddressId INT FOREIGN KEY REFERENCES Adresses(Id)

)

## **Problem 17: Backup Database**

Backup the database **SoftUni** from the previous tasks into a file named “**softuni-backup.bak**”. Delete your database from SQL Server Management Studio. Then restore the database from the created backup.

## **Problem 18: Basic Insert**

Use the **SoftUni** database and insert some data **using SQL queries**.

* **Towns:** Sofia, Plovdiv, Varna, Burgas
* **Departments:** Engineering, Sales, Marketing, Software Development, Quality Assurance
* **Employees:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Job Title** | **Department** | **Hire Date** | **Salary** |
| Ivan Ivanov Ivanov | .NET Developer | Software Development | 01/02/2013 | 3500.00 |
| Petar Petrov Petrov | Senior Engineer | Engineering | 02/03/2004 | 4000.00 |
| Maria Petrova Ivanova | Intern | Quality Assurance | 28/08/2016 | 525.25 |
| Georgi Teziev Ivanov | CEO | Sales | 09/12/2007 | 3000.00 |
| Peter Pan Pan | Intern | Marketing | 28/08/2016 | 599.88 |

INSERT INTO Towns (Name)

VALUES ('Sofia'), ('Plovdiv'),('Varna'),('Burgas');

INSERT INTO Departments

VALUES ('Engineering'), ('Sales'), ('Marketing'), ('Software Development'),('QA');

INSERT INTO Adresses (AdressText, TownId)

VALUES

('Merlow 23', 3),

('Shadow str. 39', 1),

('Kapchy str. 5', 2),

('Shushana bul. 14', 4),

('Kapinka #10', 3)

INSERT INTO Employees (FirstName, MiddleName, LastName, JobTitle, DepartmentId, HireDate, Salary, AddressId)

VALUES

('Ivan', 'Ivanov', 'Ivanov', '.NET Developer', 4, '2013-02-01', 3500.00, 2),

('Petar', 'Petrov', 'Petrov', 'Senior Engineer', 1, '2004-03-02', 4000.00, 1),

('Maria', 'Petrova', 'Ivanova', 'Intern', 5,'2016-08-28', 525.25, 4 ),

('Georgi', 'Terziev', 'Ivanov', 'CEO', 2, '2007-12-09', 3000.00, 2),

('Peter', 'Pan', 'Pan', 'Intern', 3, '2016-08-28', 599.88, 3 )

# Problem 19: Basic Select All Fields

Use the **SoftUni** database and first select all records from the **Towns**, then from **Departments** and finally from **Employees** table. Use SQL queries and submit them to Judge at once. Submit your query statements as Prepare DB & Run queries.

SELECT \* FROM dbo.Towns

SELECT \* FROM dbo.Departments

SELECT \* FROM dbo.Employees

# Problem 20: Basic Select All Fields and Order Them

Modify queries from previous problem by sorting:

* **Towns** - alphabetically by name
* **Departments** - alphabetically by name
* **Employees** - descending by salary

Submit your query statements as Prepare DB & Run queries.

SELECT \* FROM [dbo].[Towns]

ORDER BY Name ASC

SELECT \* FROM [dbo].[Departments]

ORDER BY Name ASC

SELECT \* FROM [dbo].[Employees]

ORDER BY Salary DESC

# Problem 21: Basic Select Some Fields

Modify queries from previous problem to show only **some of the columns**. For table:

* **Towns** – Name
* **Departments** – Name
* **Employees** – FirstName, LastName, JobTitle, Salary

**Keep the ordering** from the previous problem. Submit your query statements as Prepare DB & Run queries.

SELECT Name FROM dbo.Towns

SELECT Name FROM dbo.Departments

SELECT FirstName, LastName, JobTitle, Salary FROM dbo.Employees

# Problem 22: 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.

UPDATE [dbo].[Employees]

SET Salary = Salary + (Salary \* 0.1);

SELECT Salary FROM [dbo].[Employees]