# Exercises: Data Definition and Data Types

This document defines the **exercise assignments** for the ["Databases Basics - MSSQL" course @ Software University.](https://softuni.bg/courses/databases-basics-ms-sql-server)

## 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.**

CREATE DATABASE Minions

## 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**.

CREATE TABLE Minions (

Id INT NOT NULL,

[Name] NVARCHAR(50) NOT NULL,

Age INT NOT NULL

)

CREATE TABLE Towns (

Id INT NOT NULL,

[Name] NVARCHAR(50) NOT NULL

)

## 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.

ALTER TABLE Minions

ADD CONSTRAINT Pk\_Id

PRIMARY KEY(Id)

ALTER TABLE Towns

ADD CONSTRAINT Pk\_TownId

PRIMARY KEY(Id)

ALTER TABLE Minions

ADD TownId INT

ALTER TABLE Minions

ADD CONSTRAINT FK\_MinionTownId

FOREIGN KEY (TownId) REFERENCES Towns(Id)

## Insert Records in Both Tables

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Minions** | | | |  | **Towns** | |
| **Id** | **Name** | **Age** | **TownId** |  | **Id** | **Name** |
| 1 | Kevin | 22 | 1 |  | 1 | Sofia |
| 2 | Bob | 15 | 3 |  | 2 | Plovdiv |
| 3 | Steward | NULL | 2 |  | 3 | Varna |

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

INSERT INTO Towns(Id, [Name]) VALUES

(1, 'Sofia'),

(2, 'Plovdiv'),

(3, 'Varna')

INSERT INTO Minions(Id, [Name], Age, TownId) VALUES

(1, 'Kevin', 22, 1),

(2, 'Bob', 15, 3),

(3, 'Steward', NULL, 2)

## Truncate Table Minions

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

TRUNCATE TABLE Minions

## Drop All Tables

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

DROP TABLE Minions

## 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 PRIMARY KEY IDENTITY,

[Name] NVARCHAR(200) NOT NULL,

Picture VARBINARY,

Height DECIMAL(3,2),

[Weight] DECIMAL(5,2),

Gender CHAR NOT NULL,

Birthdate DATE NOT NULL,

Biography NVARCHAR(Max)

)

INSERT INTO People

([Name], Picture, Height, [Weight], Gender, Birthdate, Biography) VALUES

('Pesho Peshev', 0, 1.73, 97.42, 'm', '1967-04-02', 'Do we have [to]?'),

('Gosho Goshev', 0, 1.84, 87.34, 'm', '1976-02-03', 'Oh, the horror!'),

('Goshina Gosheva',0, 1.56, 67.33, 'f', '1982-08-09', 'Do give me a break'),

('Pesho Goshev', 0, 1.93, 83.22, 'm', '1987-12-12', 'One more to go'),

('Fritz Schönberg', 0, 1.67, 89.72, 'm', '1976-02-28', 'Finally, we are here')

## 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)
* **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.

CREATE TABLE Users(

Id BIGINT PRIMARY KEY IDENTITY,

Username VARCHAR(30) UNIQUE NOT NULL,

[Password] VARCHAR(26) UNIQUE NOT NULL,

ProfilePicture VARBINARY(MAX),

CHECK(DATALENGTH(ProfilePicture) <= 921600),

LastLoginTime DATETIME2,

IsDeleted BIT NOT NULL

)

INSERT INTO Users

(Username, [Password], ProfilePicture, LastLoginTime, IsDeleted)

VALUES

('Gosho Goshev', 'Oh, the horror!', 0, NULL, 0),

('Goshina Gosheva','Do give me a break', 0, NULL, 1),

('Pesho Goshev', 'One more to go', 0, NULL, 1),

('Fritz Schönberg', 'Finally, we are here', 0, NULL, 0),

('Pesho Peshev', 'Do we have [to]?', NULL, NULL, 0)

## 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\_\_3214EC071EF6F709

ALTER TABLE Users

ADD CONSTRAINT PK\_CompositeIdUsername

PRIMARY KEY(Id, Username)

## 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\_Password

CHECK (DATALENGTH([Password]) >= 5)

## 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\_LastLonginTime

DEFAULT GETDATE() FOR LastLoginTime

## 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\_CompositeIdUsername;

ALTER TABLE Users

ADD CONSTRAINT PK\_Id

PRIMARY KEY(Id)

ALTER TABLE Users

ADD CONSTRAINT Unique\_Username

UNIQUE(Username)

ALTER TABLE Users

ADD CONSTRAINT CHK\_Users

CHECK (DATALENGTH(Username) >= 3)

## 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 DATABASE Movies

USE Movies

CREATE TABLE Directors(

Id INT PRIMARY KEY NOT NULL,

DirectorName NVARCHAR(60) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Directors

(Id, DirectorName, Notes)

VALUES

(1, 'Gosho Goshev', Null),

(2, 'Pesho Peshev', 'The best director ever'),

(3, 'Gergana Gerganova', Null),

(4, 'Petranka Gosheva', 'Current number of movies over 50'),

(5, 'Pesho Goshev', Null)

CREATE TABLE Genres(

Id INT PRIMARY KEY NOT NULL,

GenreName NVARCHAR(30) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Genres

(Id, GenreName, Notes)

VALUES

(1, 'Comedy', 'Tell God your plans'),

(2, 'Drama', 'After you say "I do!"'),

(3, 'SCI-FI', 'Just heaven!'),

(4, 'Horror', 'Where the brave may live forever'),

(5, 'Crime', 'Where nobody lives')

CREATE TABLE Categories(

Id INT PRIMARY KEY NOT NULL,

CategoryName NVARCHAR(30) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Categories

(Id, CategoryName, Notes)

VALUES

(1, 'For kids', NUll),

(2, 'For teens', NUll),

(3, 'For young adults', NUll),

(4, 'Adults', NUll),

(5, 'Good God', NUll)

CREATE TABLE Movies(

Id INT PRIMARY KEY NOT NULL,

Title NVARCHAR(90) NOT NULL,

DirectorID INT FOREIGN KEY REFERENCES Directors(Id),

CopyrightYear INT,

[Length] NVARCHAR(8),

GenreID INT FOREIGN KEY REFERENCES Genres(Id),

CategoryID INT FOREIGN KEY REFERENCES Categories(Id),

Rating INT,

Notes NVARCHAR(MAX)

)

INSERT INTO Movies

(Id, Title, DirectorID, CopyrightYear, GenreID, CategoryID, Rating)

VALUES

(1, 'Title 1', 1, 1987, 2, 3, 5),

(2, 'Title 2', 2, 1928, 2, 2, 2),

(3, 'Title 3', 4, 1982, 3, 3, 10),

(4, 'Title 4', 5, 1955, 5, 5, 8),

(5, 'Title 5', 3, 1956, 3, 3, NULL)

## 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.

CREATE TABLE Categories(

Id INT PRIMARY KEY NOT NULL,

CategoryName NVARCHAR(50) NOT NULL,

DailyRate DECIMAL(10,2) NOT NULL,

WeeklyRate DECIMAL(10,2) NOT NULL,

MonthlyRate DECIMAL(10,2) NOT NULL,

WeekendRate DECIMAL(10,2) NOT NULL

)

INSERT INTO Categories

(Id, CategoryName, DailyRate, WeeklyRate, MonthlyRate, WeekendRate)

VALUES

(1, 'Category 1', 10, 70, 30, 5),

(2, 'Category 2', 20, 80, 40, 10),

(3, 'Category 3', 30, 70, 20, 1)

CREATE TABLE Cars(

Id INT PRIMARY KEY NOT NULL,

PlateNumber VARCHAR(30) NOT NULL,

Manufacturer VARCHAR(30) NOT NULL,

Model VARCHAR(30) NOT NULL,

CarYear INT NOT NULL,

CategoryId INT FOREIGN KEY REFERENCES Categories(Id),

Doors TINYINT NOT NULL,

Picture VARBINARY(MAX),

Condition NVARCHAR(50),

Available BIT NOT NULL

)

INSERT INTO Cars

(Id, PlateNumber, Manufacturer, Model, CarYear, CategoryId, Doors, Available)

VALUES

(1, 'PK2343OK', 'Mercedes-Benz', 'B200', 2006, 1, 5, 1),

(2, 'C0101CS', 'Opel', 'Astra', 2003, 2, 5, 0),

(3, 'A0192PO', 'Volkswagen', 'Polo', 1998, 3, 5, 0)

CREATE TABLE Employees(

Id INT PRIMARY KEY NOT NULL,

FirstName NVARCHAR(30) NOT NULL,

LastName NVARCHAR(30) NOT NULL,

Title VARCHAR(30) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Employees

(Id, FirstName, LastName, Title)

VALUES

(1, 'Ivan', 'Ivanov', 'Director'),

(2, 'Gosho', 'Peshov', 'Saler'),

(3, 'Pesho', 'Goshev', 'Engineer')

CREATE TABLE Customers(

Id INT PRIMARY KEY NOT NULL,

DriverLicenceNumber VARCHAR(30) NOT NULL,

FullName NVARCHAR(60) NOT NULL,

[Address] NVARCHAR(100) NOT NULL,

City NVARCHAR(20) NOT NULL,

ZIPCode NVARCHAR(20) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Customers

(Id, DriverLicenceNumber, FullName, [Address], City, ZIPCode)

VALUES

(1, 27282837, 'Ivan Goshov', 'Somewhere over the rainbow', 'Golden City', 8888),

(2, 27226267, 'Pesho Ivanov', 'Catch me if you can', 'Underhill', 2637),

(3, 25267262, 'Gosho Peshov', 'Who actually cares', 'Hell or heaven', 3765)

CREATE TABLE RentalOrders(

Id INT PRIMARY KEY NOT NULL,

EmployeeId INT FOREIGN KEY REFERENCES Employees(Id),

CustomerId INT FOREIGN KEY REFERENCES Customers(Id),

CarId INT FOREIGN KEY REFERENCES Cars(Id),

TankLevel DECIMAL (10,2) NOT NULL,

KilometrageStart INT NOT NULL,

KilometrageEnd INT NOT NULL,

TotalKilometrage INT NOT NULL,

StartDate DATE NOT NULL,

EndDate DATE NOT NULL,

TotalDays INT NOT NULL,

RateApplied DECIMAL (10,2),

TaxRate DECIMAL (10,2),

OrderStatus NVARCHAR(60) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO RentalOrders

(Id, EmployeeId, CustomerId, CarId, TankLevel, KilometrageStart,

KilometrageEnd, TotalKilometrage, StartDate, EndDate, TotalDays, OrderStatus)

VALUES

(1, 1, 1, 1, 100, 200, 300, 400, '1982.11.11', '1982.11.12', 30, 'completed'),

(2, 1, 2, 1, 200, 200, 1000, 10, '1987.11.11', '1987.11.12', 20, 'in progress'),

(3, 2, 2, 3, 300, 200, 100, 100, '2007.11.11', '2008.11.12', 20, 'completed')

## 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 TABLE Employees(

Id INT PRIMARY KEY NOT NULL,

FirstName NVARCHAR(30) NOT NULL,

LastName NVARCHAR(30) NOT NULL,

Title NVARCHAR(60) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Employees

(Id, FirstName, LastName, Title)

VALUES

(1, 'Ivan', 'Ivanov', 'Director'),

(2, 'Gosho', 'Peshov', 'Saler'),

(3, 'Pesho', 'Goshev', 'Engineer')

CREATE TABLE Customers(

AccountNumber INT PRIMARY KEY NOT NULL,

FirstName NVARCHAR(60) NOT NULL,

LastName NVARCHAR(60) NOT NULL,

PhoneNumber VARCHAR(20),

EmergencyName NVARCHAR(60) NOT NULL,

EmergencyNumber VARCHAR(20) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Customers

(AccountNumber, FirstName, LastName, EmergencyName, EmergencyNumber)

VALUES

(1, 'Ivan', 'Goshov', 'Somewhere over the rainbow', 8888),

(2, 'Pesho', 'Ivanov', 'Catch me if you can', 2637),

(3, 'Gosho', 'Peshov', 'Who actually cares', 3765)

CREATE TABLE RoomStatus(

RoomStatus NVARCHAR(30) PRIMARY KEY NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO RoomStatus

(RoomStatus)

VALUES

('Free'),

('Occupied'),

('Reserved')

CREATE TABLE RoomTypes(

RoomType NVARCHAR(30) PRIMARY KEY NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO RoomTypes

(RoomType)

VALUES

('Economy'),

('Luxury'),

('Honeymoon suit')

CREATE TABLE BedTypes(

BedType NVARCHAR(20) PRIMARY KEY NOT NULL,

Notes NVARCHAR(60)

)

INSERT INTO BedTypes

(BedType)

VALUES

('Single'),

('Double'),

('King sized')

CREATE TABLE Rooms(

RoomNumber INT PRIMARY KEY NOT NULL,

RoomType NVARCHAR(30) REFERENCES RoomTypes(RoomType),

BedType NVARCHAR (20) REFERENCES BedTypes(BedType),

Rate DECIMAL(2,1),

RoomStatus NVARCHAR(30) REFERENCES RoomStatus(RoomStatus),

Notes NVARCHAR(MAX)

)

INSERT INTO Rooms

(RoomNumber, RoomType, BedType, RoomStatus)

VALUES

(101, 'Economy', 'Single', 'Occupied'),

(102, 'Luxury', 'King sized', 'Reserved'),

(103, 'Honeymoon suit', 'King sized', 'Free')

CREATE TABLE Payments(

Id INT PRIMARY KEY NOT NULL,

EmployeeId INT FOREIGN KEY REFERENCES Employees(Id),

PaymentDate DATE NOT NULL,

AccountNumber INT NOT NULL,

FirstDateOccupied DATE NOT NULL,

LastDateOccupied DATE NOT NULL,

TotalDays INT NOT NULL,

AmountCharged DECIMAL(10,2) NOT NULL,

TaxRate DECIMAL(10,2) NOT NULL,

TaxAmount DECIMAL(10,2) NOT NULL,

PaymentTotal DECIMAL(10,2) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Payments

(Id, EmployeeId, PaymentDate, AccountNumber, FirstDateOccupied, LastDateOccupied, TotalDays, AmountCharged,

TaxRate, TaxAmount, PaymentTotal)

VALUES

(1, 1, '2010.11.11', 108272678, '2009.11.11', '2009.12.12', 60, 9760.208, 29.0, 20.1, 890880.22),

(2, 3, '2010.11.11', 108272678, '2009.02.11', '2009.03.12', 30, 9760.208, 29.0, 20.1, 4553.22),

(3, 2, '2010.11.11', 108272678, '2009.11.11', '2009.11.15', 4, 9760.208, 29.0, 20.1, 3478.22)

CREATE TABLE Occupancies(

Id INT PRIMARY KEY NOT NULL,

EmployeeId INT FOREIGN KEY REFERENCES Employees(Id),

DateOccupied DATE NOT NULL,

AccountNumber INT FOREIGN KEY REFERENCES Customers(AccountNumber),

RoomNumber INT FOREIGN KEY REFERENCES Rooms(RoomNumber),

RateApplied DECIMAL(10, 2) NOT NULL,

PhoneCharge VARCHAR(30) NOT NULL,

Notes NVARCHAR(MAX)

)

INSERT INTO Occupancies

(Id, EmployeeId, DateOccupied, AccountNumber, RoomNumber, RateApplied, PhoneCharge)

VALUES

(1, 1, '2018.11.11', 2, 102, 22.2, 29386473),

(2, 1, '2019.11.11', 1, 101, 33.3, 28462933),

(3, 2, '2011.11.11', 1, 103, 22.1, 29382022)

## 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.

## 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.

**Hint:** [**https://support.microsoft.com/en-gb/help/2019698/how-to-schedule-and-automate-backups-of-sql-server-databases-in-sql-se**](https://support.microsoft.com/en-gb/help/2019698/how-to-schedule-and-automate-backups-of-sql-server-databases-in-sql-se)

## 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 |

## 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 Towns;

SELECT \* FROM Departments;

SELECT \* FROM Employees;

## 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 Towns

ORDER BY [Name] ASC;

SELECT \*

FROM Departments

ORDER BY [Name] ASC;

SELECT \*

FROM Employees

ORDER BY Salary DESC;

## 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 Towns

ORDER BY [Name] ASC;

SELECT [Name]

FROM Departments

ORDER BY [Name] ASC;

SELECT FirstName,

LastName,

JobTitle,

Salary

FROM Employees

ORDER BY Salary DESC;

## 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 Employees

SET Salary \*= 1.10;

SELECT Salary

FROM Employees;

## 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.

USE Hotel

UPDATE Payments

SET

TaxRate = TaxRate - (TaxRate \* 0.03)

SELECT TaxRate

FROM Payments

## 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.

TRUNCATE TABLE Occupancies;