# Exercises: Introduction to DB Apps

This document defines the **exercise assignments** for the ["Databases Advanced – Entity Framework" course @ Software University.](https://softuni.bg/trainings/1442/databases-advanced-entity-framework-october-2016)

## Initial Setup

Write a program that connects to your **localhost** server. Create **new database** called **MinionsDB** where we will keep information about our minions and villains.

For each **minion** we should keep information about its **name**, **age** and **town**. Each **town** has information about **in** **which** **country** is located. **Villains** have **name** and **evilness** **factor** (**good**, **bad**, **evil**, **super** **evil**). Each **minion** can **serve** to **several** **villains** and **each** **villain** can **have** **several** **minions** to **serve** **him**. Fill all tables with at least 5 records each.

## Get Villains’ Names

Write a program that prints on the console **all villains’ names** and their **number of minions** of those who has more than 3 minions **ordered descending** by number of minions.

### Example

|  |
| --- |
| **Output** |
| Gru 6  Victor 4  Jilly 4 |

## Get Minion Names

Write a program that prints on the console **all minion names** and age for given **villain id.**

### Example

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 1 | Villain: Gru  1. Bob 13  2. Kevin 14  3. Steward 19 |  | 3 | Villain: Victor  1. Bob 13  2. Simon 22 |  | 2 | Villain: Victor Jr.  *(no minions)* |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 10 | No villain with ID 10 exists in the database. |

## Add Minion

Write a program that **reads** **information** about minion and its villain and **add it to the database**. In case the town of the minion is not in the database **insert** it as well. In case the villain is not present in the database add him too with default evilness factor of “evil”. Finally set the new minion to be servant of the villain. Print appropriate messages after each operation.

**\*Bonus task:** Make sure all operations are executed successfully. In case of an error do not change the database.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| Minion: Bob 14 Berlin  Villain: Gru | Successfully added Robert to be minion of Gru. |
| Minion: Cathleen 20 Liverpool  Villain: Gru | Town Liverpool was added to the database.  Successfully added Cathleen to be minion of Gru. |
| Minion: Mars 23 Sofia  Villain: Poppy | Villain Poppy was added to the database.  Successfully added Mars to be minion of Poppy |
| Minion: Carry 20 Eindhoven  Villain: Jimmy | Town Eindhoven was added to the database.  Villain Jimmy was added to the database.  Successfully added Carry to be minion of Jimmy |

## Change Town Names Casing

Write a program that **change all town names to uppercase** for towns for given country. **Print the** **number of towns that were changed** in the format provided in examples. On the next line **print** those **names that were changed** separated with coma and space.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| Bulgaria | 3 town names were affected.  [SOFIA, VARNA, BURGAS] |
| Germany | No town names were affected. |

## \*Remove Villain

Write a program that receives **ID** of a villain, **delete him from the database** and **releases his minions** from serving to him. As an output print the name of the villain and the number of minions were released. Make sure all operations go as planned otherwise do not make any changes in the database.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 | Gru was deleted  6 minions released |
| 3 | Victor was deleted  0 minions released |
| 101 | No such villain was found |

## Print All Minion Names

Write a program that **prints all minion names** from the minions table **in order** first record, last record, first + 1, last – 1, first + 2, last – 2… first + n, last – n.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 3 | 5 | 7 | 9 | 10 | 8 | 6 | 4 | 2 |

### Example

|  |  |
| --- | --- |
| **Original Order** | **Output** |
| Bob  Kevin  Steward  Jimmy  Vicky  Becky  Jully | Bob  Jully  Kevin  Becky  Steward  Vicky  Jimmy |

## Increase Minions Age

Read from console minion IDs separated by space. **Increment age** of those minions **by 1** and make their **name title case**. Finally **print names and ages of all minions** that are in the database.

### Example

|  |  |  |
| --- | --- | --- |
| **Minions** | | |
| **Id** | **Name** | **Age** |
| 1 | bob | 14 |
| 2 | steward | 22 |
| 3 | kevin | 13 |
| 4 | jimmy | 49 |
| 5 | vicky jackson | 26 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 2 1 4 | Bob 15  Steward 23  kevin 13  Jimmy 50  vicky jackson 26 |  | 5 | bob 14  steward 22  kevin 13  jimmy 49  Vicky Jackson 27 |

## Increase Age Stored Procedure

Create stored procedure **usp\_GetOlder** (**directly in the database** using **Management Studio** or any other similar tool) that receive **MinionId** and **increase its years by 1**. Write a program that **uses that stored procedure to increase age** of a minion whose id will be given as input from the console. After that **print the name and the age** of that minion.

### Example

|  |  |  |
| --- | --- | --- |
| **Minions** | | |
| **Id** | **Name** | **Age** |
| 1 | bob | 14 |
| 2 | steward | 22 |
| 3 | kevin | 13 |
| 4 | jimmy | 49 |
| 5 | vicky jackson | 26 |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 1 | bob 15 |
| 3 | kevin 14 |
| 5 | vicky jackson 27 |