# Exercises: Additional Exercises

This document defines the **exercise** assignments for the ["Databases Basics - MSSQL" course @ Software University](https://softuni.bg/modules/22/csharp-db-may-2024/1473).   
You can check your solutions in the [Judge system](https://judge.softuni.org/Contests/804/Additional-Exercises).

# PART I – Queries for Diablo Database

## Number of Users for Email Provider

Find number of users for email provider from the largest to smallest, then by Email Provider in ascending order.

### Output

|  |  |
| --- | --- |
| **Email Provider** | **Number Of Users** |
| yahoo.com | 14 |
| dps.centrin.net.id | 5 |
| softuni.bg | 5 |
| indosat.net.id | 4 |
| … | … |

## All User in Games

Find all **user in games** with information about them. Display the game name, game type, username, level, cash and character name. Sort the result by level in descending order, then by username and game in alphabetical order. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Game** | **Game Type** | **Username** | **Level** | **Cash** | **Character** |
| Calla lily white | Kinky | obliquepoof | 99 | 7527.00 | Monk |
| Dubai | Funny | rateweed | 99 | 7499.00 | Barbarian |
| Stonehenge | Kinky | terrifymarzipan | 99 | 4825.00 | Witch Doctor |
| … |  | … | … | … | … |

## Users in Games with Their Items

Find all users in games with their items count and items price. Display the username, game name, items count and items price. Display only user in games with items count more or equal to **10**. Sort the results by items count in descending order, then by price in descending order, and by username in ascending order. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Username** | **Game** | | **Items Count** | | **Items Price** | |
| skippingside | Rose Fire & Ice | | 23 | | 11065.00 | |
| countrydecay | | Star of Bethlehem | | 18 | | 8039.00 | |
| obliquepoof | | Washington D.C. | | 17 | | 5186.00 | |

## \*User in Games with Their Statistics

Find information about **every game** a user has played with their **statistics**. Each user may have participated in several games. Display the username, game name, character name, strength, defence, speed, mind and luck.  
Every statistic (strength, defence, speed, mind and luck) should be a sum of the **character statistic, game type statistic** and **items** for user **in game** statistic. One user may have multiple characters in a single game.

What you should do in order to calculate the statistics properly is to sum the following:

1. Get the sum of all items – of all characters that the user may have (**SUM**).
2. For all of his characters get the character stats which are the biggest (**MAX**).
3. For all of his game types stats select only these which are again the biggest (**MAX**).

Order the results by **Strength**, **Defence**, **Speed**, **Mind**, **Luck** – all in descending order. Submit your query statement as Prepare DB & run queries in Judge.

### Example

Let's say that we have user "**Ana**" and she is in the game "**Star of Bethlehem**", having two characters: **Sorceress** and **Paladin**. In tables below, you will be given their statistics:

Paladin

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Type of Stats\Statistics** | **Strength** | **Defence** | **Speed** | **Mind** | **Luck** |
| Item A Stats | 15 | 10 | 3 | 14 | 20 |
| Game Type Stats | 5 | 5 | 7 | 4 | 5 |
| Character Stats | 20 | 17 | 10 | 8 | 6 |

Sorceress

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Type of Stats\Statistics** | **Strength** | **Defence** | **Speed** | **Mind** | **Luck** |
| Item B Stats | 8 | 4 | 10 | 22 | 12 |
| Game Type Stats | 6 | 6 | 6 | 4 | 6 |
| Character Stats | 8 | 6 | 13 | 23 | 10 |

What we should get as a result is:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Username** | **Game** | **Character** | **Strength** | **Defence** | **Speed** | **Mind** | **Luck** |
| Ana | Star of Bethlehem | Sorceress | 49 | 37 | 33 | 63 | 48 |

Now let's see how the **Strength** is calculated:

**Strength** – (Item A's **Strength** + Item B's **Strength**) + **MAX** (**Paladin** Game Type's **Strength**, **Sorceress** GameType's **Strength**) + **MAX** (**Paladin** Character's **Strength**, **Sorceress** Character's **Strength**) = **15** + **8** + **6** + **20** = **49**.

Here we sum up first the **items** **stats** (**15 + 8**), then we add the biggest one between the game type stats (6 > 5 → 6), then we add the biggest one between the character stats (**20** > **8** → **20**). So, (**15** + **8**) + **6** + **20** = **49**.

Same goes for the **Luck**:

**Luck** = (Item A's **Luck** + Item B's **Luck**) + **MAX** (**Paladin** Game Type's **Luck**, **Sorceress** GameType's **Luck**) + **MAX** (**Paladin** Character’s **Luck**, **Sorceress** Character’s **Luck**) = **20** + **12** + **6** + **10** = **49**.

Here we sum up first the **items** **stats** (**20 + 12**), then we add the biggest one between the game type stats (**6** > **5** → **6**), then we add the biggest one between the character stats (**10** > 6 → **10**). So, (**20** + **12**) + **6** + **10** = **48**.

### Output

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Username** | **Game** | **Character** | **Strength** | **Defence** | **Speed** | **Mind** | **Luck** |
| skippingside | Rose Fire & Ice | Sorceress | 258 | 215 | 246 | 240 | 263 |
| countrydecay | Star of Bethlehem | Sorceress | 221 | 163 | 216 | 153 | 196 |
| obliquepoof | Washington D.C. | Paladin | 204 | 200 | 183 | 185 | 185 |

Note that for the **Character** column you should select the character name which is **alphabetically** "bigger" than others. In other words, if you have two characters: '**A**' and '**Z**', **pick** '**Z**' because alphabetically is after 'A'.

### Hints

You have to join **GameType** with **Statistics**, **Characters** with **Statistics** and **Items** with their **Statistics** in a single **query** (and that for every **user** in a game). After that use aggregate functions (like **MAX** and **SUM**) to calculate the above statistics.

For the character name use **MAX**(**characterName**).

## All Items with Greater than Average Statistics

Find all items with statistics larger than average. Display only items that have **Mind, Luck** and **Speed**, greater than average **Items** **mind**, **luck** and **speed**. Sort the results by item names in alphabetical order. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Price** | **MinLevel** | **Strength** | **Defence** | **Speed** | **Luck** | **Mind** |
| Aether Walker | 473.00 | 46 | 2 | 10 | 15 | 11 | 13 |
| Ancient Parthan Defenders | 566.00 | 38 | 5 | 7 | 10 | 19 | 18 |
| Aquila Cuirass | 405.00 | 76 | 5 | 7 | 10 | 19 | 18 |
| Arcstone | 613.00 | 50 | 2 | 10 | 15 | 11 | 13 |

## Display All Items with Information about Forbidden Game Type

Find **all** **items** and information whether and what forbidden game types they have. Display item name, price, min level and forbidden game type. Display all items. Sort the results by game type in descending order, then by item name in ascending order. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Price** | **MinLevel** | **Forbidden Game Type** |
| Archfiend Arrows | 531.00 | 8 | Kinky |
| Behistun Rune | 611.00 | 67 | Kinky |
| Boots | 782.00 | 44 | Kinky |
| … | … | … | … |

## Buy Items for User in Game

User **Alex** isin theshop of the game "**Edinburgh**" and she wants to buy some items. She likes **Blackguard**, **Bottomless Potion of Amplification**, **Eye of Etlich (Diablo III)**, **Gem of Efficacious Toxin**, **Golden Gorget of Leoric** and **Hellfire Amulet**. Buy the items. You should add the data in the right tables. Get the money for the items from the user in column **Cash**.

Select all users in the current game with their items. Display username, game name, cash and item name. Sort the result by item name.

Submit your query statements as Prepare DB & run queries in Judge.

### Output

|  |  |  |  |
| --- | --- | --- | --- |
| **Username** | **Name** | **Cash** | **Item Name** |
| Alex | Edinburgh | \*\*\*\*.\*\* | Akanesh, the Herald of Righteousness |
| … | … | … | … |
| corruptpizz | Edinburgh | \*\*\*\*.\*\* | Broken Crown |
| … | … | … | … |
| printerstencils | Edinburgh | \*\*\*\*.\*\* | Envious Blade |

# PART II – Queries for Geography Database

## Peaks and Mountains

Find all **peaks along with their mountain** sorted by elevation (from the highest to the lowest), then by peak name alphabetically. Display the peak name, mountain range name and elevation. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |
| --- | --- | --- |
| **PeakName** | **Mountain** | **Elevation** |
| Everest | Himalayas | 8848 |
| K2 | Karakoram | 8611 |
| Kangchenjunga | Himalayas | 8586 |
| … |  | … |

## Peaks with Their Mountain, Country and Continent

Find all peaks along with their mountain, country and continent. When a mountain belongs to multiple countries, display them all. Sort the results by peak name alphabetically, then by country name alphabetically. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |  |
| --- | --- | --- | --- |
| **PeakName** | **Mountain** | **CountryName** | **ContinentName** |
| Aconcagua | Andes | Argentina | South America |
| Aconcagua | Andes | Chile | South America |
| Banski Suhodol | Pirin | Bulgaria | Europe |
| … | … | … | … |

## Rivers by Country

For each country in the database, display the number of rivers passing through that country and the total length of these rivers. When a country does not have any river, display **0** as rivers count and as total length. Sort the results by rivers count (from largest to smallest), then by total length (from largest to smallest), then by country alphabetically. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |  |
| --- | --- | --- | --- |
| **CountryName** | **ContinentName** | **RiversCount** | **TotalLength** |
| China | Asia | 8 | 35156 |
| Russia | Europe | 6 | 26427 |
| … |  | … | … |

## Count of Countries by Currency

Find the **number of countries for each currency**. Display three columns: currency code, currency description and number of countries. Sort the results by number of countries (from highest to lowest), then by currency description alphabetically. Name the columns exactly like in the table below. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |
| --- | --- | --- |
| **CurrencyCode** | **Currency** | **NumberOfCountries** |
| EUR | Euro Member Countries | 35 |
| USD | United States Dollar | 17 |
| AUD | Australia Dollar | 8 |
| XOF | Communauté Financière Africaine (BCEAO) Franc | 8 |
| … | … | … |

## Population and Area by Continent

For each continent, display the total area and total population of all its countries. Sort the results by population from highest to lowest. Submit your query statement as Prepare DB & run queries in Judge.

### Output

|  |  |  |
| --- | --- | --- |
| **ContinentName** | **CountriesArea** | **CountriesPopulation** |
| Asia | 31603228 | 4130318467 |
| Africa | 30360296 | 1015470588 |
| … | … | … |

## Monasteries by Country

Create a **table Monasteries(Id, Name, CountryCode)**. Use auto-increment for the primary key. Create a **foreign key** between the tables **Monasteries** and **Countries**.

Execute the following SQL script (it should pass without any errors):

|  |
| --- |
| INSERT INTO Monasteries(Name, CountryCode) VALUES  ('Rila Monastery “St. Ivan of Rila”', 'BG'),  ('Bachkovo Monastery “Virgin Mary”', 'BG'),  ('Troyan Monastery “Holy Mother''s Assumption”', 'BG'),  ('Kopan Monastery', 'NP'),  ('Thrangu Tashi Yangtse Monastery', 'NP'),  ('Shechen Tennyi Dargyeling Monastery', 'NP'),  ('Benchen Monastery', 'NP'),  ('Southern Shaolin Monastery', 'CN'),  ('Dabei Monastery', 'CN'),  ('Wa Sau Toi', 'CN'),  ('Lhunshigyia Monastery', 'CN'),  ('Rakya Monastery', 'CN'),  ('Monasteries of Meteora', 'GR'),  ('The Holy Monastery of Stavronikita', 'GR'),  ('Taung Kalat Monastery', 'MM'),  ('Pa-Auk Forest Monastery', 'MM'),  ('Taktsang Palphug Monastery', 'BT'),  ('Sümela Monastery', 'TR') |

Write a SQL command to add a new **Boolean** column IsDeleted in the Countries table (defaults to false). Note that there is no "**Boolean**" type in SQL server, so you should use an alternative and make sure you set the **default** value properly.

Write and execute a SQL command to **mark as deleted all countries that have more than 3 rivers**.

Write a query to display all **monasteries** along with their **countries**, sorted by monastery name. Skip all deleted countries and their monasteries.

Submit your query statements **without the INSERT statement from above** as Prepare DB & run queries in Judge.

### Output

|  |  |
| --- | --- |
| **Monastery** | **Country** |
| Bachkovo Monastery “Virgin Mary” | Bulgaria |
| Benchen Monastery | Nepal |
| Kopan Monastery | Nepal |
| … | … |

## Monasteries by Continents and Countries

This problem assumes that **the previous task is completed successfully without errors**.

Write and execute a SQL command that **changes the country named "Myanmar" to its other name "Burma"**.

Add a **new monastery**, holding the following information: **Name**="Hanga Abbey", **Country**="Tanzania".

Add a **new monastery**, holding the following information: **Name**="Myin-Tin-Daik", **Country**="Myanmar".

Find the **count of monasteries for each continent and not deleted country**. Display the **continent name**, the **country name** and the **count of monasteries**. Include also the countries with **0** monasteries. Sort the results by monasteries count (from largest to lowest), then by country name alphabetically. Name the columns exactly like in the table below.

Submit all your query statements at once as Prepare DB & run queries in Judge.

**NOTE**: when you **insert** the **monasteries** make sure to specify the country code by the country name (use **subquery**).

### Output

|  |  |  |
| --- | --- | --- |
| **ContinentName** | **CountryName** | **MonasteriesCount** |
| Asia | Nepal | 4 |
| Europe | Bulgaria | 3 |
| Asia | Burma | 2 |
| Europe | Greece | 2 |
| … | … | … |