# Databases Exam (March 2015) – Geography

Your exam consists of several parts, explained below. You may work independently on each exam part. Submit your solutions in the automated judge system: <https://judge.softuni.bg/Contests/71/Databases-Exam-1-March-2015>.

## Part I – SQL Queries

You are given a **MS SQL Server database "Geography"** holding continents, countries, currencies, rivers, mountains and peaks, available as SQL script. Your task is to write SQL queries for displaying data from the given database. In all problems, please **name the columns exactly like in the sample tables below**.

### All Mountain Peaks

Display all **ad mountain peaks** in alphabetical order. Submit for evaluation the result grid **with headers**.

|  |
| --- |
| **PeakName** |
| Aconcagua |
| Banski Suhodol |
| Batashki Snezhnik |
| … |

1 score

### Biggest Countries by Population

Find the 30 biggest countries by population **from Europe**. Display the country name and population. Sort the results by population (from biggest to smallest), then by country alphabetically. Submit for evaluation the result grid with headers.

|  |  |
| --- | --- |
| **CountryName** | **Population** |
| Russia | 140702000 |
| Germany | 81802257 |
| France | 64768389 |
| … | … |

2 score

### Countries and Currency (Euro / Not Euro)

Find all countries along with information about their currency. Display the country code, country name and information about its currency: either "**Euro**" or "**Not Euro**". Sort the results by country name alphabetically. Submit for evaluation the result grid with headers.

|  |  |  |
| --- | --- | --- |
| **CountryName** | **CountryCode** | **Currency** |
| Afghanistan | AF | Not Euro |
| Åland | AX | Euro |
| Albania | AL | Not Euro |
| … | … | … |

2 score

### Countries Holding 'A' 3 or More Times

Find all countries that holds the letter 'A' in their name at least 3 times (case insensitively), sorted by ISO code. Display the country name and ISO code. Submit for evaluation the result grid with headers.

|  |  |
| --- | --- |
| **Country Name** | **ISO Code** |
| Afghanistan | AFG |
| Albania | ALB |
| … | … |

3 score

### 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 for evaluation the result grid with headers.

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

2 score

### 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 for evaluation the result grid with headers.

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

3 score

### \* Rivers Passing through 3 or More Countries

Find all rivers that pass through to 3 or more countries. Display the river name and the number of countries. Sort the results by river name. Submit for evaluation the result grid with headers.

|  |  |
| --- | --- |
| **River** | **Countries Count** |
| Amazon | 7 |
| Amur | 3 |
| Brahmaputra | 5 |
| … | … |

4 score

### Highest, Lowest and Average Peak Elevation

Find the highest, lowest and average peak elevation. Submit for evaluation the result grid with headers.

|  |  |  |
| --- | --- | --- |
| **MaxElevation** | **MinElevation** | **AverageElevation** |
| 88… | 20… | 48… |

1 score

### 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 for evaluation the result grid with headers.

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

2 score

### 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 for evaluation the result grid with headers.

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

2 score

### \* 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 for evaluation the result grid with headers.

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

4 score

### Highest Peak and Longest River by Country

For each country, find the elevation of **the highest peak** and **the length of the longest river**, sorted by the highest peak elevation (from highest to lowest), then by the longest river length (from longest to smallest), then by country name (alphabetically). Display **NULL** when no data is available in some of the columns. Submit for evaluation the result grid with headers.

|  |  |  |
| --- | --- | --- |
| **CountryName** | **HighestPeakElevation** | **LongestRiverLength** |
| China | 8848 | 6300 |
| India | 8848 | 3180 |
| Nepal | 8848 | 2948 |
| Pakistan | 8611 | 3180 |
| Argentina | 6962 | 4880 |
| Chile | 6962 | NULL |
| … | … | … |

5 score

### Mix of Peak and River Names

Combine all peak names with all river names, so that the last letter of each peak name is the same like the first letter of its corresponding river name. Display the peak names, river names, and the obtained mix. Sort the results by the obtained mix. Submit for evaluation the result grid with headers.

|  |  |  |
| --- | --- | --- |
| **PeakName** | **RiverName** | **Mix** |
| Aconcagua | Amazon | aconcaguamazon |
| Aconcagua | Amur | aconcaguamur |
| Banski Suhodol | Lena | banski suhodolena |
| … |  | … |

5 score

### \*\* Highest Peak Name and Elevation by Country

For each country, find the name and elevation of **the highest peak**, along with its mountain. When no peaks are available in some country, display elevation **0**, "**(no highest peak)**" as peak name and "**(no mountain)**" as mountain name. When multiple peaks in some country have the same elevation, display all of them. Sort the results by country name alphabetically, then by highest peak name alphabetically. Submit for evaluation the result grid with headers.

|  |  |  |  |
| --- | --- | --- | --- |
| **Country** | **Highest Peak Name** | **Highest Peak Elevation** | **Mountain** |
| Afghanistan | (no highest peak) | 0 | (no mountain) |
| … | … | … | … |
| Argentina | Aconcagua | 6962 | Andes |
| … | … | … | … |
| Bulgaria | Musala | 2925 | Rila |
| Burkina Faso | (no highest peak) | 0 | (no mountain) |
| … | … | … | … |
| United States | Mount McKinley | 6194 | Alaska Range |
| … | … | … | … |
| Zambia | (no highest peak) | 0 | (no mountain) |
| Zimbabwe | (no highest peak) | 0 | (no mountain) |

9 score

## Part II – Changes in the Database

You are given a **MS SQL Server database "Geography"** holding continents, countries, currencies, rivers, mountains and peaks, available as SQL script. Your task is to modify the database schema and data and write SQL queries for displaying data from the database.

### Monasteries by Country

1. Create a **table Monasteries(Id, Name, CountryCode)**. Use auto-increment for the primary key. Create a **foreign key** between the tables Monasteries and Countries.
2. 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') |

1. 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.
2. Write and execute a SQL command to **mark as deleted all countries that have more than 3 rivers**.
3. Write a query to display all **monasteries** along with their **countries** sorted by monastery name. Skip all deleted countries and their monasteries. Submit for evaluation the result grid with headers.

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

13 score

### Monasteries by Continents and Countries

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

1. Write and execute a SQL command that **changes the country named "Myanmar" to its other name "Burma"**.
2. Add a **new monastery** holding the following information: Name="Hanga Abbey", Country="Tanzania".
3. Add a **new monastery** holding the following information: Name="Myin-Tin-Daik", Country="Myanmar".
4. 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 for evaluation the result grid with headers.

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

7 score

## Part III – Stored Procedures

You are given a **MS SQL Server database "Geography"** holding continents, countries, currencies, rivers, mountains and peaks, available as SQL script. Your task is to write some stored procedures, views and other server-side database objects and write some SQL queries for displaying data from the database.

**Important:** start with a clean copy of the "Geography" database. Just execute the SQL script again.

### Stored Function: Mountain Peaks JSON

Create a **stored function** fn\_MountainsPeaksJSON that lists **all mountains** alphabetically along with **all its peaks** alphabetically. Format the output as **JSON string** without any whitespace.

If your function is correct and you execute the following SQL query:

|  |
| --- |
| SELECT dbo.fn\_MountainsPeaksJSON() |

The result should be like the sample below:

|  |
| --- |
| {"mountains":[{"name":"Alaska Range","peaks":[{"name":"Mount McKinley","elevation":6194}]},{"name":"Alborz","peaks":[{"name":"Damavand","elevation":5610}]},{"name":"Andes","peaks":[{"name":"Aconcagua","elevation":6962},{"name":"Monte Pissis","elevation":6793},{"name":"Ojos del Salado","elevation":6893},{"name":"Cerro Bonete","elevation":6759},{"name":"Galán","elevation":5912},{"name":"Mercedario","elevation":6720},{"name":"Pissis","elevation":6795}]},…]} |

The same code is better understandable when it is formatted, but please do not add whitespace:

|  |
| --- |
| { "mountains": [  { "name": "Alaska Range",  "peaks": [ {"name": "Mount McKinley", "elevation": 6194} ]  },  { "name": "Alborz",  "peaks": [ {"name": "Damavand", "elevation": 5610} ]  },  { "name": "Andes",  "peaks": [ {"name": "Aconcagua", "elevation": 6962},  {"name": "Monte Pissis", "elevation": 6793},  {"name": "Ojos del Salado", "elevation": 6893},  {"name": "Cerro Bonete", "elevation": 6759},  {"name": "Galán", "elevation": 5912},  {"name": "Mercedario", "elevation": 6720},  {"name": "Pissis", "elevation": 6795} ]  },  … ]  } |

To test your output, you could use JSON some JSON visualizer like <http://jsonviewer.stack.hu>.

Submit for evaluation the output from your stored function (JSON string without whitespace, with no header row).

18 score

## Part V – Database Schema Design

Your task is to design a MySQL database schema, fill some data in it and write a query to retrieve some data.

### Design a Database Schema in MySQL and Write a Query

1. Design a **MySQL database "trainings"** to hold training centers, courses and a course timetable. **Courses** hold name and optional description. **Training centers** hold name, optional description and optional URL. The **course timetable** holds a set of timetable items, each consisting of course, training center and starting date. All tables should have auto-increment primary key – **id**. All text fields should accept Unicode characters.

-- DROP DATABASE IF EXISTS `trainings`;

CREATE DATABASE `trainings`

CHARACTER SET utf8 COLLATE utf8\_unicode\_ci;

USE `trainings`;

DROP TABLE IF EXISTS `training\_centers`;

CREATE TABLE `training\_centers` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(45) NOT NULL,

`description` text,

`url` varchar(2083),

PRIMARY KEY (`id`)

);

DROP TABLE IF EXISTS `courses`;

CREATE TABLE `courses` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`name` varchar(100) NOT NULL,

`description` text,

PRIMARY KEY (`id`)

);

DROP TABLE IF EXISTS `courses\_timetable`;

CREATE TABLE `timetable` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`course\_id` int(11) NOT NULL,

`training\_center\_id` int(11) NOT NULL,

`start\_date` date NOT NULL,

PRIMARY KEY (`id`),

CONSTRAINT `fk\_courses\_timetable\_courses`

FOREIGN KEY (`course\_id`)

REFERENCES `courses` (`id`),

CONSTRAINT `fk\_courses\_timetable\_training\_centers`

FOREIGN KEY (`training\_center\_id`)

REFERENCES `training\_centers` (`id`)

);

INSERT INTO `training\_centers` VALUES

(1, 'Sofia Learning', NULL, 'http://sofialearning.org'),

(2, 'Varna Innovations & Learning', 'Innovative training center, located in Varna. Provides trainings in software development and foreign languages', 'http://vil.edu'),

(3, 'Plovdiv Trainings & Inspiration', NULL, NULL),

(4, 'Sofia West Adult Trainings', 'The best training center in Lyulin', 'https://sofiawest.bg'),

(5, 'Software Trainings Ltd.', NULL, 'http://softtrain.eu'),

(6, 'Polyglot Language School', 'English, French, Spanish and Russian language courses', NULL),

(7, 'Modern Dances Academy', 'Learn how to dance!', 'http://danceacademy.bg');

INSERT INTO `courses` VALUES

(101, 'Java Basics', 'Learn more at https://softuni.bg/courses/java-basics/'),

(102, 'English for beginners', '3-month English course'),

(103, 'Salsa: First Steps', NULL),

(104, 'Avancée Français', 'French language: Level III'),

(105, 'HTML & CSS', NULL),

(106, 'Databases', 'Introductionary course in databases, SQL, MySQL, SQL Server and MongoDB'),

(107, 'C# Programming', 'Intro C# corse for beginners'),

(108, 'Tango dances', NULL),

(109, 'Spanish, Level II', 'Aprender Español');

INSERT INTO `timetable`(course\_id, training\_center\_id, start\_date) VALUES

(101, 1, '2015-01-31'), (101, 5, '2015-02-28'),

(102, 6, '2015-01-21'), (102, 4, '2015-01-07'), (102, 2, '2015-02-14'), (102, 1, '2015-03-05'), (102, 3, '2015-03-01'),

(103, 7, '2015-02-25'), (103, 3, '2015-02-19'),

(104, 5, '2015-01-07'), (104, 1, '2015-03-30'), (104, 3, '2015-04-01'),

(105, 5, '2015-01-25'), (105, 4, '2015-03-23'), (105, 3, '2015-04-17'), (105, 2, '2015-03-19'),

(106, 5, '2015-02-26'),

(107, 2, '2015-02-20'), (107, 1, '2015-01-20'), (107, 3, '2015-03-01'),

(109, 6, '2015-01-13');

UPDATE `timetable` t

JOIN `courses` c ON t.course\_id = c.id

SET t.start\_date = DATE\_SUB(t.start\_date, INTERVAL 7 DAY)

WHERE c.name REGEXP '^[a-j]{1,5}.\*s$';

SELECT

tc.name AS `traning center`,

t.start\_date AS `start date`,

c.name AS `course name`,

c.description AS `more info`

FROM `timetable` t

JOIN `courses` c ON t.course\_id = c.id

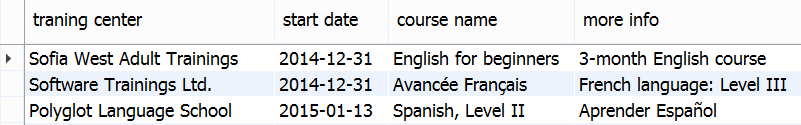
JOIN `training\_centers` tc ON t.training\_center\_id = tc.id

ORDER BY t.start\_date, t.id

1. Execute the following SQL script to load data in your tables. It should pass without any errors:

|  |
| --- |
| INSERT INTO `training\_centers` VALUES (1, 'Sofia Learning', NULL, 'http://sofialearning.org'), (2, 'Varna Innovations & Learning', 'Innovative training center, located in Varna. Provides trainings in software development and foreign languages', 'http://vil.edu'), (3, 'Plovdiv Trainings & Inspiration', NULL, NULL),  (4, 'Sofia West Adult Trainings', 'The best training center in Lyulin', 'https://sofiawest.bg'), (5, 'Software Trainings Ltd.', NULL, 'http://softtrain.eu'),  (6, 'Polyglot Language School', 'English, French, Spanish and Russian language courses', NULL), (7, 'Modern Dances Academy', 'Learn how to dance!', 'http://danceacademy.bg');  INSERT INTO `courses` VALUES (101, 'Java Basics', 'Learn more at https://softuni.bg/courses/java-basics/'), (102, 'English for beginners', '3-month English course'), (103, 'Salsa: First Steps', NULL), (104, 'Avancée Français', 'French language: Level III'), (105, 'HTML & CSS', NULL), (106, 'Databases', 'Introductionary course in databases, SQL, MySQL, SQL Server and MongoDB'), (107, 'C# Programming', 'Intro C# corse for beginners'), (108, 'Tango dances', NULL), (109, 'Spanish, Level II', 'Aprender Español');  INSERT INTO `timetable`(course\_id, training\_center\_id, start\_date) VALUES (101, 1, '2015-01-31'), (101, 5, '2015-02-28'), (102, 6, '2015-01-21'), (102, 4, '2015-01-07'), (102, 2, '2015-02-14'), (102, 1, '2015-03-05'), (102, 3, '2015-03-01'), (103, 7, '2015-02-25'), (103, 3, '2015-02-19'), (104, 5, '2015-01-07'), (104, 1, '2015-03-30'), (104, 3, '2015-04-01'), (105, 5, '2015-01-25'), (105, 4, '2015-03-23'), (105, 3, '2015-04-17'), (105, 2, '2015-03-19'), (106, 5, '2015-02-26'), (107, 2, '2015-02-20'), (107, 1, '2015-01-20'), (107, 3, '2015-03-01'), (109, 6, '2015-01-13');  UPDATE `timetable` t JOIN `courses` c ON t.course\_id = c.id  SET t.start\_date = DATE\_SUB(t.start\_date, INTERVAL 7 DAY)  WHERE c.name REGEXP '^[a-j]{1,5}.\*s$'; |

1. Write a SQL query to **list all entries from the timetable** ordered by start date and then by id. Display the **training center**, **start date**, **course name** and **more info** about the course (course details). Name the columns exactly like in the table below:



Submit for evaluation the result grid in text format, comma separated, with row headers. It should be formatted exactly like in the sample below:

|  |
| --- |
| # traning center, start date, course name, more info  Sofia West Adult Trainings, 2014-12-31, English for beginners, 3-month English course  Software Trainings Ltd., 2014-12-31, Avancée Français, French language: Level III  Polyglot Language School, 2015-01-13, Spanish, Level II, Aprender Español  … |

17 score

## Exam Information

To avoid locale-specific problems, use the "**English / United States**" as your locale. The decimal point is ".", the month names are in English, etc.

You are allowed to use any resources you have like Internet, software, existing code.

You are not allowed to get help from other people: Skype, ICQ, FB, email, talks, phone calls, etc. are forbidden.

Exam time: **5 hours**.