# Exercises: Subqueries and JOINs

This document defines the **exercise assignments** for the ["Java DB - MySQL" course @ Software University.](https://softuni.bg/trainings/2352/mysql-may-2019)

Submit your solutions here: <https://judge.softuni.bg/Contests/606/Subqueries-and-JOINs-Exercise>.

For problems from 1 to 11 (inclusively) use "**soft\_uni**" database and for the others – "**geography**".

## Employee Address

Write a query that selects:

* employee\_id
* job\_title
* address\_id
* address\_text

Return the first 5 rows sorted by address\_id **in ascending order.**

### Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **job\_title** | **address\_id** | **address\_text** |
| 142 | Production Technician | 1 | 108 Lakeside Court |
| 30 | Human Resources Manager | 2 | 1341 Prospect St |
| … | … | … | … |

## Addresses with Towns

Write a query that selects:

* first\_name
* last\_name
* town
* address\_text

Sort the result by first\_name **in** **ascending order** then by last\_name. Select first 5 employees.

### Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **first\_name** | **last\_name** | **town** | **address\_text** |
| A.Scott | Wright | Newport Hills | 1400 Gate Drive |
| Alan | Brewer | Kenmore | 8192 Seagull Court |
| … | … | … | … |

## Sales Employee

Write a query that selects:

* employee\_id
* first\_name
* last\_name
* department\_name

Sort the result by employee\_id **in descending order**. Select only **employees** from the “**Sales**” department.

### Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **first\_name** | **last\_name** | **department\_name** |
| 290 | Lynn | Tsoflias | Sales |
| 289 | Rachel | Valdez | Sales |
| … | … | … | … |

## Employee Departments

Write a query that selects:

* employee\_id
* first\_name
* salary
* department\_name

Filter only **employees** with salary higher than 15000. Return the first 5 rows sorted by department\_id **in descending order.**

### Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **first\_name** | **salary** | **department\_name** |
| 109 | Ken | 125500.00 | Executive |
| 140 | Laura | 60100.00 | Executive |
| … | … | … | … |

## Employees Without Project

Write a query that selects:

* employee\_id
* first\_name

Filter only **employees** without a project. Return the first 3 rows sorted by employee\_id **in descending order.**

### Example:

|  |  |
| --- | --- |
| **employee\_id** | **first\_name** |
| 293 | George |
| 292 | Martin |
| … | … |

## Employees Hired After

Write a query that selects:

* first\_name
* last\_name
* hire\_date
* dept\_name

Filter only **employees** hired after 1/1/1999 and from either the **"Sales"** or the **"Finance"** **departments**. Sort the result by hire\_date **(ascending).**

### Example:

|  |  |  |  |
| --- | --- | --- | --- |
| **first\_name** | **last\_name** | **hire\_date** | **dept\_name** |
| Debora | Poe | 2001-01-19 00:00:00 | Finance |
| Wendy | Kahn | 2001-01-26 00:00:00 | Finance |
| … | … | … | … |

## Employees with Project

Write a query that selects:

* employee\_id
* first\_name
* project\_name

Filter only **employees** with a project, which has started after **13.08.2002** and it is still **ongoing** (no end date). Return the first 5 rows sorted by first\_name **then by** project\_name **both in ascending order.**

### Example

|  |  |  |
| --- | --- | --- |
| **employee\_id** | **first\_name** | **project\_name** |
| 44 | A. Scott | Hitch Rack - 4-Bike |
| 170 | Alan | LL Touring Handlebars |
| … | … | … |

## Employee 24

Write a query that selects:

* employee\_id
* first\_name
* project\_name

Filter all the **projects** of employee with **id 24**. If the project has started after **2005 inclusively** the return value should be NULL. Sort the result by project\_name **alphabetically.**

### Example

|  |  |  |
| --- | --- | --- |
| **employee\_id** | **first\_name** | **project\_name** |
| 24 | David | NULL |
| 24 | David | NULL |
| … | … | … |

## Employee Manager

Write a query that selects:

* employee\_id
* first\_name
* manager\_id
* manager\_name

Filter all **employees** with a manager who has id **equal to 3 or 7**. Return all rows sorted by **employee** first\_name **in ascending order.**

### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **first\_name** | **manager\_id** | **manager\_name** |
| 122 | Bryan | 7 | JoLynn |
| 158 | Dylan | 3 | Roberto |
| … | … | … | … |

## Employee Summary

Write a query that selects:

* employee\_id
* employee\_name
* manager\_name
* department\_name

Show the first 5 **employees** (only for employees who have a manager) with their **managers** and the **departments** they are in (show the departments of the **employees**). Order by employee\_id**.**

### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **employee\_name** | **manager\_name** | **department\_name** |
| 1 | Guy Gilbert | Jo Brown | Production |
| 2 | Kevin Brown | David Bradley | Marketing |
| … | … | … | … |

## Min Average Salary

Write a query that returns the value of the **lowest average** salary of all **departments**.

### Example:

|  |
| --- |
| **min\_average\_salary** |
| 10866.6666 |

## Highest Peaks in Bulgaria

Write a query that selects:

* country\_code
* mountain\_range
* peak\_name
* elevation

Filter all **peaks** in **Bulgaria** with **elevation** over **2835**. Return all rows sorted by elevation **in descending order.**

### Example

|  |  |  |  |
| --- | --- | --- | --- |
| **country\_code** | **mountain\_range** | **peak\_name** | **elevation** |
| BG | Rila | Musala | 2925 |
| BG | Pirin | Vihren | 2914 |
| … | … | … | … |

## Count Mountain Ranges

Write a query that selects:

* country\_code
* mountain\_range

Filter the **count** of the **mountain** **ranges** in the **United States, Russia and Bulgaria.** Sort result by mountain\_rangecountin **decreasing order**.

### Example

|  |  |
| --- | --- |
| **country\_code** | **mountain\_range** |
| BG | 6 |
| RU | 1 |
| … | … |

## Countries with Rivers

Write a query that selects:

* country\_name
* river\_name

Find the first 5 **countries** with or without **rivers** in **Africa**. Sort them by **country\_name in ascending order.**

### Example

|  |  |
| --- | --- |
| **country\_name** | **river\_name** |
| Algeria | Niger |
| Angola | Congo |
| Benin | Niger |
| Botswana | NULL |
| Burkina Faso | Niger |

## \*Continents and Currencies

Write a query that selects:

* **continent\_code**
* **currency\_code**
* **currency\_usage**

Find all **continents** and their most used **currency**. Filter any **currency** that is used in only one **country**. Sort the result by continent\_code **and** currency\_code**.**

### Example

|  |  |  |
| --- | --- | --- |
| **continent\_code** | **currency\_code** | **currency\_usage** |
| AF | XOF | 8 |
| AS | AUD | 2 |
| AS | ILS | 2 |
| EU | EUR | 26 |
| NA | XCD | 8 |
| OC | USD | 8 |

## Countries without any Mountains

Find the count of all **countries** which don’t have a **mountain**.

### Example

|  |
| --- |
| **country\_count** |
| 231 |

## 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. Limit only the **first 5 rows**.

### Example

|  |  |  |
| --- | --- | --- |
| **country\_name** | **highest\_peak\_elevation** | **longest\_river\_length** |
| China | 8848 | 6300 |
| India | 8848 | 3180 |
| Nepal | 8848 | 2948 |
| Pakistan | 8611 | 3180 |
| Argentina | 6962 | 4880 |