# PostgreSQL Exam Preparation I

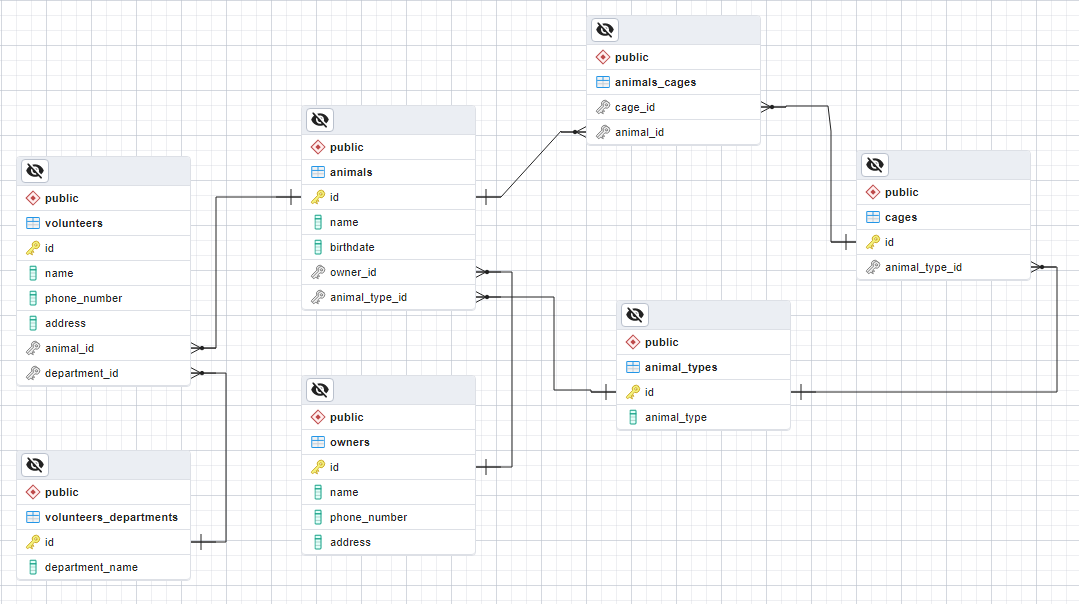
**Submit your solutions** to the SoftUni [**Judge Canimalontest**](https://alpha.judge.softuni.org/contests/exam-preparation-i/4296).

# Zoo

*As an employee of the local Zoo, you have been given the opportunity to design a management system to track the animals and the people involved in the Zoo. This system will help streamline operations and ensure efficient management of the Zoo's resources.*

# Section 1. Data Definition Language (DDL) - (30 pts)

You have been provided with the E/R Diagram of the **Zoo**. This diagram represents the relationships between various entities within the Zoo and serves as a visual representation of the database structure.



Create a PostgreSQL database named **"zoo\_db"** that consists of seven tables:

* **"owners"** - stores information about the owners of the animals;
* **"animal\_types"** - hold data about the different types of animals in the zoo;
* **"cages"** - stores information about the animal cages;
* **"animals"** - contain information about the animals;
* **"volunteers\_departments"** - hold data about the departments of the volunteers;
* **"volunteers"** - contain information about the volunteers in the zoo;
* **"animals\_cages"** - serves as a many-to-many mapping table between animals and cages.

**NOTE: Please ensure that you use the exact data types specified in the model tables when working with dates. For instance, if a column is of type "DATE," make sure to use the "DATE" type. Similarly, if a column is of type "TIMESTAMP" use the "TIMESTAMP" data type. Failure to use the correct data types may result in the Judge system rejecting your submission as incorrect.**

Your first assignment is to create the database tables based on the provided models. Follow the given specifications to create the tables.

## ****owners****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **name** | **String** up to **50** symbols | **NULL** is **not** allowed |
| **phone\_number** | **String** up to **15** symbols | **NULL** is **not** allowed |
| **address** | **String** up to **50** symbols | **NULL** is permitted |

## ****animal\_types****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **animal\_type** | **String** up to **30** symbols | **NULL** is **not** allowed |

## ****cages****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **animal\_type\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **animal\_types**, Cascade Operations, **NULL** is **not** allowed |

## ****animals****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **name** | **String** up to **30** symbols | **NULL** is **not** allowed |
| **birthdate** | **DATE** | **NULL** is **not** allowed |
| **owner\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **owners**, Cascade Operations, **NULL** is permitted |
| **animal\_type\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **animal\_types**, Cascade Operations, **NULL** is **not** allowed |

## ****volunteers\_departments****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **department\_name** | **String** up to **30** symbols | **NULL** is **not** allowed |

## ****volunteers****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **id** | **Integer** from **0** to **2,147,483,647** | Primary Key, Unique table identification, Auto-increment |
| **name** | **String** up to **50** symbols | **NULL** is **not** allowed |
| **phone\_number** | **String** up to **15** symbols | **NULL** is **not** allowed |
| **address** | **String** up to **50** symbols | **NULL** is permitted |
| **animal\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **animals**, Cascade Operations, **NULL** is permitted |
| **department\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **volunteers\_departments**, Cascade Operations, **NULL** is **not** allowed |

## ****animals\_cages****

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Constraints** |
| **cage\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **cages**, Cascade Operations, **NULL** is **not** allowed |
| **animal\_id** | **Integer** from **0** to **2,147,483,647** | Relationship with table **animals**, Cascade Operations, **NULL** is **not** allowed |

## Database Design

Submit only your **CREATE** statements for all tables to the Judge.

# Section 2. Data Manipulation Language (DML) - (10 pts)

**Before beginning, it is necessary to import "dataset.sql". If the structure has been properly created, the data should be inserted successfully.**

In this particular situation, various tasks require making changes to the database, including data modifications and the addition of new records.

## Insert

Let's insert sample data into the database by writing a query that adds the provided records into their **respective tables**, with **all "id" values generated automatically**.

### ****volunteers****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **name** | **phone\_number** | **address** | **animal\_id** | **department\_id** |
| Anita Kostova | 0896365412 | Sofia, 5 Rosa str. | 15 | 1 |
| Dimitur Stoev | 0877564223 | NULL | 42 | 4 |
| Kalina Evtimova | 0896321112 | Silistra, 21 Breza str. | 9 | 7 |
| Stoyan Tomov | 0898564100 | Montana, 1 Bor str. | 18 | 8 |
| Boryana Mileva | 0888112233 | NULL | 31 | 5 |

### ****animals****

|  |  |  |  |
| --- | --- | --- | --- |
| **name** | **birthdate** | **owner\_id** | **animal\_type\_id** |
| Giraffe | 2018-09-21 | 21 | 1 |
| Harpy Eagle | 2015-04-17 | 15 | 3 |
| Hamadryas Baboon | 2017-11-02 | NULL | 1 |
| Tuatara | 2021-06-30 | 2 | 4 |

## Update

**Kaloqn Stoqnov**, an existing owner in the database, visited the zoo to adopt all animals that currently do not have an owner assigned to them. Please update the records by correctly assigning the **"owner\_id"** to these animals.

### Example

Before update

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **name** | **birthdate** | **owner\_id** | **animal\_type\_id** |
| 1 | Brown bear | 2017-07-17 | 3 | 1 |
| 2 | Chimpanzee | 2010-01-21 | 6 | 1 |
| … | … | … | … | … |
| 11 | Hippo | 2017-09-07 | [null] | 1 |
| 12 | Koala | 2018-06-30 | 24 | 1 |
| … | … | … | … | … |
| 14 | Banded Archer Fish | 2022-01-15 | [null] | 2 |
| 15 | Cichlid | 2021-01-21 | 5 | 2 |
| 16 | Koi | 2021-07-05 | [null] | 2 |
| … | … | … | … | … |
| 20 | Saddlebill Stork | 2019-08-21 | [null] | 3 |
| … | … | … | … | … |
| 44 | Desert Hairy Scorpion | 2020-05-13 | [null] | 6 |
| … | … | … | … | … |
| 50 | Tuatara | 2021-06-30 | 2 | 4 |

After update

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **name** | **birthdate** | **owner\_id** | **animal\_type\_id** |
| 1 | Brown bear | 2017-07-17 | 3 | 1 |
| 2 | Chimpanzee | 2010-01-21 | 6 | 1 |
| … | … | … | … | … |
| 11 | Hippo | 2017-09-07 | 4 | 1 |
| 12 | Koala | 2018-06-30 | 24 | 1 |
| … | … | … | … | … |
| 14 | Banded Archer Fish | 2022-01-15 | 4 | 2 |
| 15 | Cichlid | 2021-01-21 | 5 | 2 |
| 16 | Koi | 2021-07-05 | 4 | 2 |
| … | … | … | … | … |
| 20 | Saddlebill Stork | 2019-08-21 | 4 | 3 |
| … | … | … | … | … |
| 44 | Desert Hairy Scorpion | 2020-05-13 | 4 | 6 |
| … | … | … | … | … |
| 50 | Tuatara | 2021-06-30 | 2 | 4 |

## Delete

The Zoo has decided to close down the **'Education program assistant'** department. Your task is to remove this department from the database. Please note that there might be **foreign key constraint conflicts** that need to be considered during the deletion process.

### Example

Before delete

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **id** | **name** | **phone\_number** | **address** | **animal\_id** | **department\_id** |
| 1 | Kiril Kostadinov | 0896541233 | Sofia , 213 Tsarigradsko shose str. | 7 | 2 |
| 2 | Boyan Boyanov | 0896321546 | Plovdiv, 15 Arda str. | 14 | 1 |
| 3 | Mariya Petkova | 0874563201 | Kalofer, 2 Tsar Simeon str. | 4 | 3 |
| … | … | … | … | … | … |
| 6 | Anton Antonov | 0877456123 | Varna, 2 Dobrotitsa str. | 11 | 3 |
| 7 | Yanko Totev | 0896369258 | Sofia, 54 Hristo Botev str. | 1 | 2 |
| 8 | Katerina Dimitrova | 0874589665 | [null] | 5 | 6 |
| … | … | … | … | … | … |
| 29 | Boryana Mileva | 0888112233 | [null] | 31 | 5 |

|  |  |
| --- | --- |
| **id** | **department\_name** |
| 1 | Guest engagement |
| 2 | Education program assistant |
| 3 | Zoo events |
| … | … |

After delete

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **id** | **name** | **phone\_number** | **address** | **animal\_id** | **department\_id** |
| 2 | Boyan Boyanov | 0896321546 | Plovdiv, 15 Arda str. | 14 | 1 |
| 3 | Mariya Petkova | 0874563201 | Kalofer, 2 Tsar Simeon str. | 4 | 3 |
| … | … | … | … | … | … |
| 6 | Anton Antonov | 0877456123 | Varna, 2 Dobrotitsa str. | 11 | 3 |
| 8 | Katerina Dimitrova | 0874589665 | [null] | 5 | 6 |
| … | … | … | … | … | … |
| 29 | Boryana Mileva | 0888112233 | [null] | 31 | 5 |

|  |  |
| --- | --- |
| **id** | **department\_name** |
| 1 | Guest engagement |
| 3 | Zoo events |
| … | … |

# Section 3. Querying - (40 pts)

**In the upcoming assignment, the Zoo management has requested you to retrieve specific data. It is crucial to mention that the sample results provided are based on a fresh database. To ensure a clean dataset, you need to recreate the database and import the sample data again using the provided "dataset.sql" file. It is important not to make any modifications or include any data from the INSERT, UPDATE, and DELETE tasks.**

## Volunteers

Retrieve details about all the **"volunteers"**, including their **"name"**, **"phone\_number"**, **"address"**, the animal they are responsible for (**"animal\_id"**), and the department they are involved in (**"department\_id"**). Sort the results by the **volunteer's** **"name"** in **ascending order**. If a volunteer is responsible for multiple animals, the results should be further sorted by the **"animal\_id"** in **ascending order**. Additionally, sort the results by the **"department\_id"** in **ascending order**.

### Example

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **name** | **phone\_number** | **address** | **animal\_id** | **department\_id** |
| Anton Antonov | 0877456123 | Varna, 2 Dobrotitsa str. | 11 | 3 |
| Boyan Boyanov | 0896321546 | Plovdiv, 15 Arda str. | 14 | 1 |
| Darina Petrova | 0889654236 | Sofia, 39 Bratya Buxton str. | 31 | 3 |
| Dilyana Stoeva | 0889412025 | Sofia, 15 Lyulyak str. | [null] | 2 |
| … | … | … | … | … |
| Zdravko Asenov | 0889652365 | Sofia, 6 Neven str. | 19 | 2 |

## Animals Data

The goal of this task is to retrieve a list of all **animals** and their **corresponding types**. Extract the **"name"** of the animal, **"animal\_type"**, and the **"birthdate"** formatted as **'DD.MM.YYYY'**. Sort the results in **ascending order** based on the **animal's** **"name"**.

### Example

|  |  |  |
| --- | --- | --- |
| **name** | **animal\_type** | **birthdate** |
| African Penguin | Birds | 17.07.2017 |
| African Spurred Tortoise | Reptiles | 26.09.2009 |
| American Kestrel | Birds | 27.04.2019 |
| … | … | … |
| Bald Eagle | Birds | 29.06.2014 |
| … | … | … |
| California Condor | Birds | 19.12.2014 |
| … | … | … |
| Woma Python | Reptiles | 26.04.2019 |

## Owners and Their Animals

Write an SQL query that retrieves the animals for each owner. Find the **top 5** owners who have the highest count of animals. Select the owner's **"name"** and the **"count\_of\_animals"** they own. Order the result by the **count of animal**s owned in **descending order**. If there are multiple owners with the same count of animals, order them by the **owner'**s **"name"** in **ascending order**.

### Example

|  |  |
| --- | --- |
| **owner** | **count\_of\_animals** |
| Kaloqn Stoqnov | 4 |
| Kiril Peshev | 4 |
| Kamelia Yancheva | 3 |
| Martin Genchev | 3 |
| Metodi Dimitrov | 3 |

## Owners, Animals and Cages

As part of your responsibilities, you are required to retrieve information about the owners of **'Mammals',** including the name of their animal and the cage in which these animals are located. The query should select the owner's **"name"** and the animal's "**name"** in the format **"owner - animal"**, along with the owner's **"phone\_number"** and the cage **"id"**. The results should be ordered by the **"name"** of the owner in **ascending order**. If an owner has more than one animal, the results should be ordered by their **animal`s** **"name"** in **descending order**.

### Example

|  |  |  |
| --- | --- | --- |
| **owners - animals** | **phone\_number** | **cage\_id** |
| Anelia Mihova - Koala | 0897856147 | 16 |
| Borislava Kamenova - Fennec Fox | 0877477112 | 21 |
| Gergana Mancheva - Brown bear | 0897412123 | 26 |
| Kaloqn Stoqnov - Leopard | 0878325642 | 32 |
| Kaloqn Stoqnov - Elephant | 0878325642 | 37 |
| … | … | … |
| Petya Dobreva - Giant Panda | 0874547896 | 27 |

## Volunteers in Sofia

Create a SQL query that retrieves information about the **"volunteers"** who are involved in the **'Education** **program assistant'** department and live in Sofia. The query should select their **"name"**, **"phone\_number"**, and **"address"** in Sofia (**excluding the city's name**). The results should be ordered by the **"name"** of the **volunteers** in **ascending order**.

### Example

|  |  |  |
| --- | --- | --- |
| **volunteers** | **phone\_number** | **address** |
| Dilyana Stoeva | 0889412025 | 15 Lyulyak str. |
| Kiril Kostadinov | 0896541233 | 213 Tsarigradsko shose str. |
| Yanko Totev | 0896369258 | 54 Hristo Botev str. |
| Zdravko Asenov | 0889652365 | 6 Neven str. |

## Animals for Adoption

Retrieve all animals that are **not owned**, and are **younger than 5 years** (as of **'01/01/2022'**), excluding **Birds**. Select their **"name"**, **year of birth**, and **"animal\_type"**. Order the results by the **"name"** of the **animal** in **ascending order**.

### Example

|  |  |  |
| --- | --- | --- |
| **animal** | **birth\_year** | **animal\_type** |
| Banded Archer Fish | 2022 | Fish |
| Chameleon | 2018 | Reptiles |
| Desert Hairy Scorpion | 2020 | Invertebrates |
| Goliath Frog | 2020 | Amphibians |
| Hippo | 2017 | Mammals |
| Koi | 2021 | Fish |
| Poison Frog | 2020 | Amphibians |

# Section 4. Programmability - (20 pts)

## All Volunteers in a Department

Your task is to create a user-defined function called **fn\_get\_volunteers\_count\_from\_department()**. This function should accept a parameter named **"searched\_volunteers\_department"** with a **maximum length of 30**, which represents a **specific department**. The function should return the **count of volunteers** who are involved in that department.

For this task, please only submit your **user-defined** **function** in the Judge system.

### Examples

|  |  |
| --- | --- |
| **Query** | **Output** |
| **SELECT fn\_get\_volunteers\_count\_from\_department('Education program assistant')** | **6** |
| **SELECT fn\_get\_volunteers\_count\_from\_department('Guest engagement')** | **4** |
| **SELECT fn\_get\_volunteers\_count\_from\_department('Zoo events')** | **5** |

## 12. Animals with Owner or Not

To address the task, you should create a stored procedure named **sp\_animals\_with\_owners\_or\_not()**. This procedure accepts a parameter called **"animal\_name"** of type **VARCHAR(30)** and retrieves the name of the owner associated with the specified animal. If the animal does not have an owner, the procedure will return the value **'For adoption'**.

For this task, please only submit your **stored procedure** in the Judge system.

### Example

|  |  |
| --- | --- |
| **Query** | **Output** |
| **CALL sp\_animals\_with\_owners\_or\_not('Pumpkinseed Sunfish')** | **Kamelia Yancheva** |
| **CALL sp\_animals\_with\_owners\_or\_not('Hippo')** | **For adoption** |
| **CALL sp\_animals\_with\_owners\_or\_not('Brown bear')** | **Gergana Mancheva** |