Python ORM Exam - 03 August 2024



*Create three models that allow you to manage, manipulate, and extract data from the Database. These models together form the basis of a simple Space Exploration Platform.*

*Your project will manage astronauts, missions, and spacecrafts.*

# **Skeleton**

You are provided with a ready-to-use skeleton. Do not change the folder and file names! You are allowed to add additional files.

**Judge Submissions:**

* Once you have **completed a task**, you must **archive** the project files (**zip** format) and upload the zip file to the contest (for the **corresponding judge task**). You **do not need to include** your **venv**, **.idea**, **pycache**, and **\_\_MACOSX** (for Mac users), so you do **not exceed** the maximum allowed size of **31.25** **KB**
* Submit a solution (archived project files) for **each** **task**!

A screenshot of a computer

Description automatically generated

# **Database - 100 points**

You will need to create **three models** in the **models.py** file:

## Astronaut

* + **name**
    - A **character** field.
    - Represents the full name of the astronaut.
    - Validation: **Minimum length** of **2** characters, **maximum length** of **120** characters.
* **phone\_number**
  + - A **character** field.
    - Represents the phone number of the astronaut.
    - Validation: **Must** **contain only digits**, **maximum length** of **15** characters.
    - Additional Note: This field must contain **unique** values.
* **is\_active**
  + - A **boolean** field.
    - Indicates whether the astronaut is currently **active**.
    - Default Value: **True** (Active).
* **date\_of\_birth**
  + - A **date** field.
    - Represents the birth date of the astronaut.
    - Additional Note: This field is **not required** and can have **NULL** values.
* **spacewalks**
  + - An **integer** field.
    - Represents the number of spacewalks the astronaut has participated in.
    - Default value: **0**
    - Validation: **Minimum value 0**.
* **updated\_at**
  + - A **datetime** field.
    - **Automatically** updated to the **current datetime** every time the object is **saved**.

## Spacecraft

* **name**
* A **character** field.
* Represents the name of the spacecraft.
* Validation: **Minimum** **length** of **2** characters, **maximum length** of **120** characters.
* **manufacturer**
  + A **character** field.
* Represents the manufacturer of the spacecraft.
* Validations: **Maximum length** of **100** characters.
* **capacity**
  + A **small** **positive integer** field.
* Represents the number of astronauts the spacecraft can carry.
* Validation: **Minimum value 1**.
* **weight**
  + A **float** field.
* Represents the weight of the spacecraft in kilograms.
* Validation: **Minimum value 0.0**.
* **launch\_date**
  + A **date** field.
* Represents the launch date of the spacecraft.
* **updated\_at**
  + - A **datetime** field.
    - **Automatically** updated to the **current datetime** every time the object is **saved**

## Mission

* **name**
* A **character** field.
* Represents the name of the mission.
  + Validation: **Minimum** **length** of **2** characters, **maximum length** of **120** characters
* **description**
* A **text** field.
* Provides a detailed description of the mission.
* Additional Note: This field is **not required** and can have **NULL** values.
* **status**
  + A **character** field with **pre-defined choices**.
* Represents the status of the mission.
* Valid choices: "**Planned**", "**Ongoing**", or "**Completed**".
* Validation: **Maximum length** of **9** characters.
* Default Value: "**Planned**".
* **launch\_date**
  + A **date** field.
  + Represents the launch date of the mission.
* **updated\_at**
  + - A **datetime** field.
* **Automatically** updated to the **current datetime** every time the object is **saved**
* **spacecraft**
  + A **foreign key** to the **Spacecraft** model.
  + Establishes a **many-to-one** relationship with the Spacecraft model, associating each mission with a spacecraft.
  + **ON DELETE** constraint must be set to **CASCADE**.
* **astronauts**
  + A **many-to-many** field to the **Astronaut** model.
* Establishes a **many-to-many** relationship with the Astronaut model, allowing multiple astronauts to participate in a mission.
* **commander**
  + A **foreign key** to the **Astronaut** model.
* Establishes a **many-to-one** relationship with the Astronaut model, indicating the mission commander.
* **ON DELETE** constraint must be set to **SET\_NULL**.
* This field **can** have **NULL** values.

# **Customizing Django Admin Site - 30 points**

Register your models to the Django Admin Site (**admin.py** file) and make the following customizations which will enhance the admin interface by providing more meaningful and searchable information:

## AstronautAdmin

* **Display fields**: Specify the fields to be displayed in the list view of the admin site for the **Astronaut model**.
  + Fields: **'name'**, **'spacewalks'**, **'is\_active'**
* **Filters**: Add a filter for **'is\_active'** in the admin site.
* **Search fields**: Enable search by **'name'** and **'phone\_number'** in the admin site.
* **Ordering**: Set the default ordering by '**name**'.

## SpacecraftAdmin

* **Display fields**: Specify the fields to be displayed in the list view of the admin site for the **Spacecraft model**.
  + Fields: **'name'**, **'manufacturer'**, **'launch\_date'**
* **Filters**: Add a filter for **'capacity'** in the admin site.
* **Search fields**: Enable search by **'name'** in the admin site.
* **Read**-**only** **fields**: **'updated\_at'**.

## **MissionAdmin**

* **Display fields**: Specify the fields to be displayed in the list view of the admin site for the **Mission model**.
  + Fields: **'name'**, **'status'**, **'description'**, **'launch\_date'**
* **Filters**: Add filters for **'status'** and **'launch\_date'** in the admin site.
* **Search fields**: Enable search by **commander's 'name'** (searching missions by commander's name).
* **Read-only fields**: **'updated\_at'**.

# **Custom Model Manager - 20 points**

Create a **custom model manager** for the **Astronaut** **model** and add your **custom method**:

### get\_astronauts\_by\_missions\_count()

This method **retrieves** and **returns** all **astronaut objects**, **ordered descending by** the **number** of **missions** each astronaut has participated in, **then by** their **phone numbers ascending**.

# **Django Queries I - 75 points**

In the **caller.py** file create the following functions:

### get\_astronauts(search\_string=None)

This function accepts the following argument with default **None** value:

* **search\_string** - string value or **None**

It **retrieves** astronaut objects by **partially** and **case-insensitively** matching the given searching criteria for **name** or **phone number**. Check if **any** of these two field values **contain** the searched string.

**If there are** astronaut objects that match the criteria, **order** them by **name** **ascending**.

**Return** a **string** in the following format, each astronaut's info on a new line:

**"Astronaut: {name1}, phone number: {phone\_number1}, status: {Active/Inactive}**

**Astronaut: {name2}, phone number: {phone\_number2}, status: {Active/Inactive}**

**…**

**Astronaut: {nameN}, phone number: {phone\_numberN}, status: {Active/Inactive}"**

* If **no astronauts** **match** the criteria or the **search\_string** is **None**, **return** an **empty string** (**""**).
* Consider the search scenario when the **search\_string** is an **empty string**.
  + **Hint**: Empty strings are always considered to be a substring of any other string.
* The **status** (**'Active'** or **'Inactive'**) depends on the **is\_active** value (**True** or **False**).

### get\_top\_astronaut()

This function accepts no arguments.

It **retrieves** the astronaut with the **greatest number** of **participated missions**.

Ifthere is **more than one astronaut** with the **same number** ofmissions, **order** them by **phone number ascending**, and **return** the **first one's** info.

**Return** a **string** in the following format:

**"Top Astronaut: {name} with {num\_of\_missions} missions."**

* If there are **no astronauts** or **no missions** in the database, **return** the following **string:**

**"No data."**

### get\_top\_commander()

This function accepts no arguments.

It **retrieves** the astronaut with the **greatest number** of **commanded missions.**

Ifthere is **more than one astronaut** with the **same number** ofcommandedmissions, **order** them **by phone number ascending**, and **return** the **first one's** info.

**Return** a **string** in the following format:

**"Top Commander: {name} with {num\_of\_missions} commanded missions."**

* If there are **no astronauts** or **no missions** in the database,or the **missions** in the database **have no commanders** yet, **return** the following **string**:

**"No data."**

# **Django Queries II - 75 points**

### get\_last\_completed\_mission()

This function accepts no arguments.

It **retrieves** information about the **last completed mission** (**latest** **launch date** and **mission status "Completed"**) and **returns** a **string** in the following format:

**"The last completed mission is: {mission\_name}. Commander: {commander\_name/TBA}. Astronauts: {astronaut1\_name}, …, { astronautN\_name}. Spacecraft: {spacecraft\_name}. Total spacewalks: {total\_spacewalks}."**

* If the **commander's name** **is NULL**, **return "TBA" instead**.
* Astronauts' **names** must be **separated** by a **comma and space** (**", "**)and **ordered by name**, **ascending**.
* The **total spacewalks** representthe **sum of all spacewalks** of the **associated astronauts**.
* If there are **no completed missions** in the database, **return** the following **string:**

**"No data."**

### get\_most\_used\_spacecraft()

This function accepts no arguments.

It **retrieves** the **most used** **spacecraft** by consideringthe **number** **of missions** it was assigned to. If you have **spacecrafts** used in the **same number of missions**, **order** them **by spacecraft's name** **ascending**, and get the **first one**.

**Return** a **string** in the following format:

**"The most used spacecraft is: {spacecraft\_name}, manufactured by {manufacturer}, used in {num\_missions} missions, astronauts on missions: {num\_astronauts}."**

* The **'astronauts on missions'** represent the **number of unique astronauts** who havebeen **on missions** with this **spacecraft**.
* If **no missions** exist in the database, **return** the following **string**:

**"No data."**

### decrease\_spacecrafts\_weight()

This function accepts no arguments.

It **filters** the **unique** **spacecraft objects** currently **assigned** to **planned missions** (mission status "**Planned**") and **decreases** their **weight** by **200.0 kilograms**.

The **spacecraft's weight** should **not drop** **below zero**(**0.0**), therefore **filter** further **only** those **objects** that **weigh** **at least 200.0** **kg**.

**Decrease** the **weight only once** if the **spacecraft participates** in **multiple** **planned missions**.

**Return** a **string** in the following format:

**"The weight of {num\_of\_spacecrafts\_affected} spacecrafts has been decreased. The new average weight of all spacecrafts is {avg\_weight}kg"**

* The **average weight** represents the **average value**, calculated based on the **weight of all spacecrafts after updates**.
* **Format** the **average weight** to the **first decimal place**.
* If there are **no affected** spacecrafts, **return**:

**"No changes in weight."**

# **Testing Data Constraints**

* There will always be astronauts and spacecrafts when creating missions.
* The number of participating astronauts will always correspond to spacecrafts capacity. No validation is needed.
* The following outputs show the **expected behavior** of the functions. Populate the database with your own testing data and then check if the functions produce the expected results.

# **Examples**

|  |
| --- |
| **Sample Data in Database** |
| ***# Astronauts***   1. *John Deer is an active astronaut with 3 spacewalks, born on January 1, 1980, and can be reached at phone number 853967.* 2. *Jane Smith is an active astronaut with 1 spacewalk, born on May 15, 1985, and can be reached at phone number 123456.* 3. *Josie Stam is an inactive astronaut with no spacewalks, born on March 12, 1990, and can be reached at phone number 111111.*   ***# Spacecrafts***   1. *A spacecraft named "Explorer I" was manufactured by SpaceTech Inc., has a capacity of 5, and was launched on January 1, 2022, weighing 12000.5 kilograms.* 2. *A spacecraft named "Explorer II" was manufactured by SpaceX, has a capacity of 2, and was launched on May 1, 2023, weighing 10000.2 kilograms*   ***# Missions***   1. *A planned mission named "Moon Landing" with a launch date of October 10, 2024, led by John Deer. It's aimed at landing on the moon and has astronauts John Deer and Jane Smith assigned to it.* *The spacecraft "Explorer I" manufactured by SpaceTech Inc. is assigned to this mission.* 2. *A completed mission named "Moon Landing2" was launched on March 1, 2024, led by Josie Stam. It's also aimed at landing on the moon and has astronauts Jane Smith and Josie Stam assigned to it. The spacecraft "Explorer I" manufactured by SpaceTech Inc. is assigned to this mission.* |

|  |
| --- |
| **Test Code** |
| print(Astronaut.objects.get\_astronauts\_by\_missions\_count()) |
| **Output** |
| <QuerySet [<Astronaut: Jane Smith>, <Astronaut: Josie Stam>, <Astronaut: John Deer>]> |
| **Test Code** |
| print(get\_astronauts(search\_string='jO')) |
| **Output** |
| Astronaut: John Deer, phone number: 853967, status: Active  Astronaut: Josie Stam, phone number: 111111, status: Inactive |
| **Test Code** |
| print(get\_astronauts(search\_string='zzz')) |
| **Output** |
|  |
| **Test Code** |
| print(get\_top\_astronaut()) |
| **Output** |
| Top Astronaut: Jane Smith with 2 missions. |
| **Test Code** |
| print(get\_top\_commander()) |
| **Output** |
| Top Commander: Josie Stam with 1 commanded missions. |
| **Test Code** |
| print(get\_last\_completed\_mission()) |
| **Output** |
| The last completed mission is: Moon Landing2. Commander: Josie Stam. Astronauts: Jane Smith, Josie Stam. Spacecraft: Explorer I. Total spacewalks: 1. |
| **Test Code** |
| print(get\_most\_used\_spacecraft()) |
| **Output** |
| The most used spacecraft is: Explorer I, manufactured by SpaceTech Inc., used in 2 missions, astronauts on missions: 3. |
| **Test Code** |
| print(decrease\_spacecrafts\_weight()) |
| **Output** |
| The weight of 1 spacecrafts has been decreased. The new average weight of all spacecrafts is 10900.4kg |