Exercises — PlanetBase

version #

Assistants C/Unix 2022 <assistants@tickets.assistants.epita.fr>

Copyright

This document is for internal use at EPITA (website) only.

Contents

*https://intra.assistants.epita.fr

1 PlanetBase

Files to submit:

- planet_base/views.sql
- planet_base/procedures.sql
- planet_base/triggers.sql

Provided files:

- planet_base/schema.sql
- planet_base/data.sql

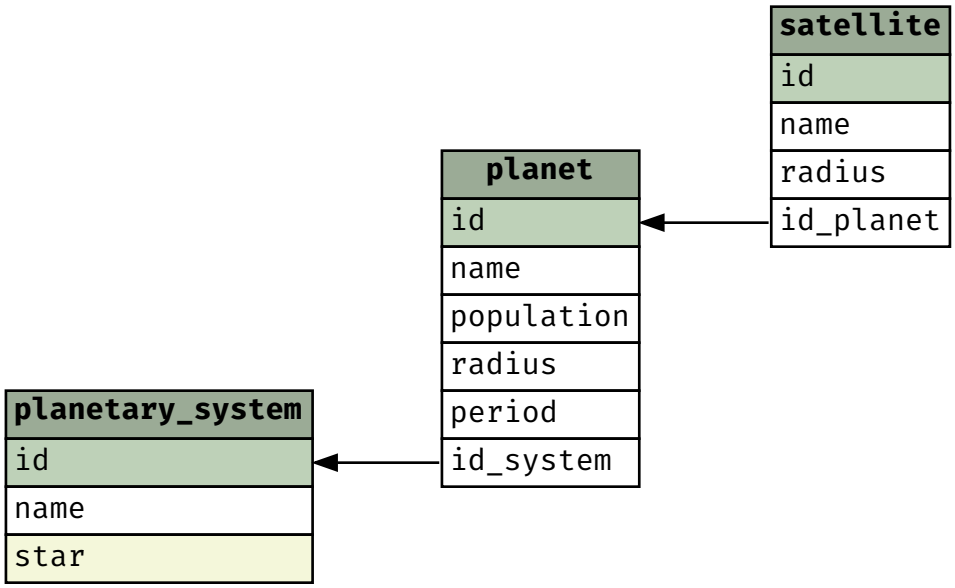
1.1 Goal

In this exercise you will have to manage a little part of the universe. You will only focus on planetary systems, planets and satellites.

Tables

Look at the `schema.sql` to fully understand what is expected of you.

Relationships



Procedures

All procedures must be written in `procedures.sql`.

space_travel:

This procedure allows humans to travel from a planet to another planet of the same planetary system. You must update the population of both involved planets. The amount of travelers in the spaceship must be strictly positive.

This procedure returns true if the travel is a success, false otherwise - and in this case, the population of both planets must not change.

```
FUNCTION space_travel(origin INT, destination INT, quantity BIGINT)
RETURNS boolean;
```

list_satellite_inf_750:

This procedure returns a list of all the satellites having a radius of 750 km or less, and belonging to the given planetary system. We expect three columns:

- The name of the satellite
- The name of the planet corresponding to the satellite
- The radius of the satellite

Order them by planet name, then by satellite radius in descending order and finally by satellite name.

```
FUNCTION list_satellite_inf_750(syst INT)
RETURNS TABLE(satellite VARCHAR(32), planet VARCHAR(32), radius INT);
```

```
 satellite | planet  | radius
-----------+---------+--------
 Proteus   | Neptune |    210
 Nereid    | Neptune |    170
 Iapetus   | Saturn  |    734
 Dione     | Saturn  |    561
 Tethys    | Saturn  |    531
 Enceladus | Saturn  |    252
 Mimas     | Saturn  |    198
 Umbriel   | Uranus  |    584
 Ariel     | Uranus  |    578
 Miranda   | Uranus  |    235
(10 rows)
```

Views

All views must be written in `views.sql`.

view_nearest_planet_to_sun:

Display the three nearest planets from the star matching the name “sun”, with an insensitive case. We will not test the case of several planetary systems with stars called *Sun*. We also consider that planets with the shortest periods are the closest to the star.

Order these planets by period.

```
 planet
---------
 Mercury
 Venus
 Earth
(3 rows)
```

view_nb_satellite_per_planet:

Display the number of satellites per planet. Satellites with a radius of less than 500 km must be ignored and their matching planets must be displayed with a null number. Sort by number of satellites then by planet name.

```
 planet  | number of satellites
---------+----------------------
 Mars    |                    0
 Mercury |                    0
 Venus   |                    0
 Earth   |                    1
 Neptune |                    1
 Jupiter |                    4
 Uranus  |                    4
```

(continues on next page)

(continued from previous page)

```
 Saturn  |                   5
(8 rows)
```

view_average_period:

Display the average period of the planets in each planetary system, sorted by average period, then by system name. Round the average period to 2 decimals. Do not forget systems without a single planet.

You will have to create new planetary system to test this view.

```
    system    | average_period
--------------+----------------
 Empty system |              0
 Solar        |       13353.88
 Epita        |       86506.73
(3 rows)
```

view_biggest_entities:

Display the 10 biggest entities (planets and satellites) from any planetary system, sorted by radius in descending order, then by name.

```
   type    | system |   name   | radius
-----------+--------+----------+--------
 planet    | Solar  | Jupiter  |  69911
 planet    | Solar  | Saturn   |  58232
 planet    | Solar  | Uranus   |  25362
 planet    | Solar  | Neptune  |  24622
 planet    | Solar  | Earth    |   6371
 planet    | Solar  | Venus    |   6051
 planet    | Solar  | Mars     |   3389
 satellite | Solar  | Ganymede |   2634
 satellite | Solar  | Titan    |   2576
 planet    | Solar  | Mercury  |   2439
(10 rows)
```

Triggers

All triggers must be written in `triggers.sql`.

This file must contain any eventual table and procedure that helps you produce the expected result.

store_earth_population_updates:

This trigger stores all information about earth population modifications:

- The date of the modification (TIMESTAMP). Use `now()`
- The old population (BIGINT)
- The new population (BIGINT)
- An id

You will have to create the associated view **view_earth_population_evolution** in `triggers.sql`. This view lists all modifications, ordered by the date of the update. Pay attention to the format of the date: `DD/MM/YYYY HH24:MI:SS`.

```
 id |        date         | old population | new population
----+---------------------+----------------+----------------
  1 | 01/02/2017 12:53:18 |     7000000000 |     6999500000
  2 | 02/02/2017 15:35:37 |     6999500000 |     6999000000
  3 | 03/02/2017 07:51:28 |     6999000000 |     6999004500
  4 | 04/02/2017 09:41:29 |     6999004500 |     6999039900
  5 | 13/02/2017 11:09:46 |     6999039900 |     6999040500
(5 rows)
```

It is my job to make sure you do yours.