## **Работа с базой данных**

### **Задание 1**

Представь: тебе нужно проверить, отображается ли созданный заказ в базе данных.

Для этого: выведи список логинов курьеров с количеством их заказов в статусе «В доставке» (поле inDelivery = true).

**PowerShell**

PS C:\Windows\System32> ssh -V

OpenSSH\_for\_Windows\_9.5p1, LibreSSL 3.8.2

PS C:\Windows\System32> *export PATH=$PATH:/usr/bin:/bin:/c/Windows/System32/OpenSSH*

ParserError:

Line |

1 | export PATH=$PATH:/usr/bin:/bin:/c/Windows/System32/OpenSSH

| ~~~~~~

| Variable reference is not valid. ':' was not followed by a valid variable name character. Consider using ${} to

| delimit the name.

PS C:\Windows\System32> *ssh -i ~/.ssh/id\_rsa a347efd9-7eac-4603-a4ab-da1bf1434118@serverhub.praktikum-services.ru -p 4554*

morty@891c9a9ca0a1:~$ *psql -U morty -d scooter\_rent*

Password for user morty: *smith*

psql (11.18 (Debian 11.18-0+deb10u1))

Type "help" for help.

scooter\_rent=# *\dt*

List of relations

Schema | Name | Type | Owner

--------+---------------+-------+-------

public | Couriers | table | root

public | Orders | table | root

public | SequelizeMeta | table | root

(3 rows)

scooter\_rent=# *\d "Couriers"*

Table "public.Couriers"

Column | Type | Collation | Nullable | Default

--------------+--------------------------+-----------+----------+----------------------------------------

id | integer | | not null | nextval('"Couriers\_id\_seq"'::regclass)

login | character varying(255) | | not null |

passwordHash | character varying(255) | | |

firstName | character varying(255) | | |

createdAt | timestamp with time zone | | not null |

updatedAt | timestamp with time zone | | not null |

Indexes:

"Couriers\_pkey" PRIMARY KEY, btree (id)

"Couriers\_login\_key" UNIQUE CONSTRAINT, btree (login)

scooter\_rent=# *\d "Orders"*

Table "public.Orders"

Column | Type | Collation | Nullable | Default

--------------+--------------------------+-----------+----------+--------------------------------------

id | integer | | not null | nextval('"Orders\_id\_seq"'::regclass)

courierId | integer | | |

firstName | character varying(255) | | |

lastName | character varying(255) | | |

address | character varying(255) | | |

metroStation | character varying(255) | | |

phone | character varying(255) | | |

rentTime | integer | | |

deliveryDate | timestamp with time zone | | |

track | integer | | |

inDelivery | boolean | | |

color | character varying(255)[] | | |

comment | character varying(255) | | |

cancelled | boolean | | |

finished | boolean | | |

createdAt | timestamp with time zone | | not null |

updatedAt | timestamp with time zone | | not null |

Indexes:

"Orders\_pkey" PRIMARY KEY, btree (id)

scooter\_rent=# *SELECT \* FROM "Couriers";*

id | login | passwordHash | firstName | createdAt | updatedAt

----+-------+--------------+-----------+-----------+-----------

(0 rows)

scooter\_rent=# *SELECT \* FROM "Orders"*;

scooter\_rent=# scooter\_rent=#

scooter\_rent=# *INSERT INTO "Couriers" (id, login, "createdAt", "updatedAt")*

scooter\_rent-# *VALUES*

scooter\_rent-#  *(1, 'courier\_1', NOW(), NOW()),*

scooter\_rent-#  *(2, 'courier\_2', NOW(), NOW());*

INSERT 0 2

scooter\_rent=#  *INSERT INTO "Orders" ("courierId", "inDelivery", "createdAt", "updatedAt")*

scooter\_rent-#  *VALUES*

scooter\_rent-#  *(1, true, NOW(), NOW()),*

scooter\_rent-#  *(2, true, NOW(), NOW()),*

scooter\_rent-# (1, true, NOW(), NOW());

INSERT 0 3

scooter\_rent=# *SELECT*

scooter\_rent-# *c.login AS courier\_login,*

scooter\_rent-# *COUNT(o.id) AS orders\_in\_delivery*

scooter\_rent-# *FROM*

scooter\_rent-# *"Couriers" c*

scooter\_rent-# *JOIN*

scooter\_rent-#  *"Orders" o ON c.id = o."courierId"*

scooter\_rent-# *WHERE*

scooter\_rent-# *o."inDelivery" = true*

scooter\_rent-# *GROUP BY*

scooter\_rent-#  *c.login*

scooter\_rent-# *ORDER BY*

scooter\_rent-# *orders\_in\_delivery DESC;*

courier\_login | orders\_in\_delivery

---------------+--------------------

courier\_1 | 2

courier\_2 | 1

(2 rows)

scooter\_rent=#

### **Задание 2**

Ты тестируешь статусы заказов. Нужно убедиться, что в базе данных они записываются корректно.Для этого: выведи все трекеры заказов и их статусы.

Статусы определяются по следующему правилу:

Если поле **finished == true,** то вывести статус **2**.

Если поле **canсelled == true**, то вывести статус **-1**.

Если поле **inDelivery == true**, то вывести статус **1**.

Для остальных случаев вывести **0**.

**Технические примечания:**

Доступ к базе осуществляется с помощью команды *psql -U morty -d scooter\_rent.* Пароль: smith.У psql есть особенность: если таблица в базе данных с большой буквы, то её в запросе нужно брать в кавычки. Например, **select \* from “Orders”**.

PowerShell 7.4.6

PS C:\Windows\System32> export PATH=$PATH:/usr/bin:/bin:/c/Windows/System32/OpenSSH

ParserError:

Line |

1 | export PATH=$PATH:/usr/bin:/bin:/c/Windows/System32/OpenSSH

| ~~~~~~

| Variable reference is not valid. ':' was not followed by a valid variable name character. Consider using ${} to

| delimit the name.

PS C:\Windows\System32> ssh -i ~/.ssh/id\_rsa 0ed72b8f-179b-4139-8462-e8a647f48e9c@serverhub.praktikum-services.ru -p 4554

morty@253090ba24c8:~$ psql -U morty -d scooter\_rent

Password for user morty: smith

psql (11.18 (Debian 11.18-0+deb10u1))

Type "help" for help.

scooter\_rent=# \d "Orders"

Table "public.Orders"

Column | Type | Collation | Nullable | Default

--------------+--------------------------+-----------+----------+--------------------------------------

id | integer | | not null | nextval('"Orders\_id\_seq"'::regclass)

courierId | integer | | |

firstName | character varying(255) | | |

lastName | character varying(255) | | |

address | character varying(255) | | |

metroStation | character varying(255) | | |

phone | character varying(255) | | |

rentTime | integer | | |

deliveryDate | timestamp with time zone | | |

track | integer | | |

inDelivery | boolean | | |

color | character varying(255)[] | | |

comment | character varying(255) | | |

cancelled | boolean | | |

finished | boolean | | |

createdAt | timestamp with time zone | | not null |

updatedAt | timestamp with time zone | | not null |

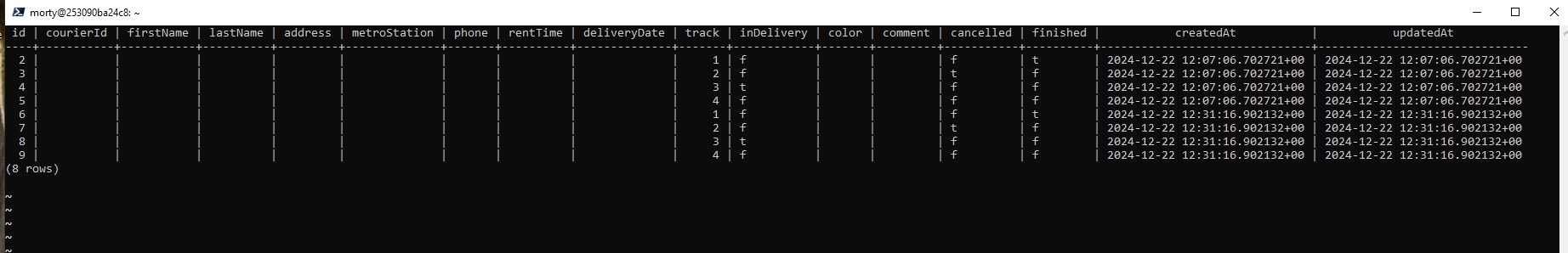
Indexes:

"Orders\_pkey" PRIMARY KEY, btree (id)

scooter\_rent=# INSERT INTO "Orders" (track, finished, cancelled, "inDelivery", "createdAt", "updatedAt") VALUES (1, true, false, false, NOW(), NOW()), (2, false, true, false, NOW(), NOW()), (3, false, false, true, NOW(), NOW()), (4, false, false, false, NOW(), NOW());

INSERT 0 4

scooter\_rent=# SELECT\*FROM"Orders";



scooter\_rent=# scooter\_rent=#

scooter\_rent=# ^C

scooter\_rent=# SELECT

scooter\_rent-# track AS tracker\_id,

scooter\_rent-# CASE

scooter\_rent-# WHEN finished = true THEN 2

scooter\_rent-# WHEN cancelled = true THEN -1

scooter\_rent-# WHEN "inDelivery" = true THEN 1

scooter\_rent-# ELSE 0 END AS status FROM "Orders";

tracker\_id | status

------------+--------

1 | 2

2 | -1

3 | 1

4 | 0

(4 rows)

scooter\_rent=#

## **Автоматизация теста к API**

Теперь автоматизируй сценарий, который подготовили коллеги-тестировщики:

* Клиент создает заказ.
* Проверяется, что по треку заказа можно получить данные о заказе.

**Шаги автотеста:**

* Выполнить запрос на создание заказа.
* Сохранить номер трека заказа.
* Выполнить запрос на получения заказа по треку заказа.
* Проверить, что код ответа равен 200.

**Технические примечания:**

* К проекту добавь файлы .gitignore и README.MD .
* Логи лежат в файле *error.log* в папке */var/www/backend/logs.*