Команда № 25.

Участники команды:

Деданов Алексей

Конева Анастасия

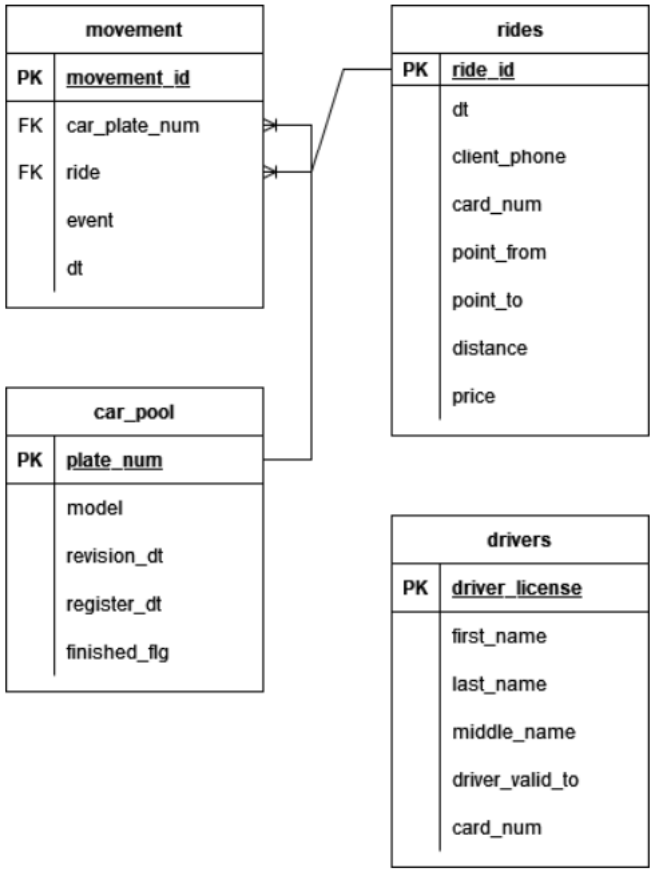
Конева Анна

Есаков Вячеслав

Техническое задание на разработку хранилища данных. Этап 1.

1. Настроить ETL процесс, который будет забирать данные из источника и раскладывать их по целевым таблицам в описанную в документации структуру в хранилище

Первым шагом для извлечения данных с сервера необходимо создать базу данных на сервере PostgreSQL (таблицы work_movement, work_rides, work_car_pool, work_drivers) при помощи базовых команд SQL (используя данные из таблиц сущностей)



С помощью этих таблиц будет извлекаться информация с исходного сервера в нашу базу данных (через команды питона)

После создания таблиц подключимся к серверу по предоставленным нам реквизитам

Реквизиты подключения к источнику техническим пользователем:

- host: de-edu-db.chronosavant.ru
- port: 5432
- database: taxi
- schema: main
- user: etl_tech_user
- password: etl_tech_user_password

Для того, чтобы подключиться к серверу с данными необходимо подключить и импортировать библиотеку для работы с *psycopg2*, которая предназначена для адаптеров баз данных (субд PosgreSQL)

```
#Сначала подключаемся к их бд и извлекаем нужные данные
conn = psycopg2.connect(dbname='taxi', user='etl_tech_user',
                        password='etl_tech_user_password', host='de-edu-db.chronosavant.ru')
cursor = conn.cursor()
```

Вводим необходимые реквизиты и извлекаем данные, которые постоянно обновляются (добавляются)

Поскольку изначально (после создания пустых таблиц) в нашей таблице отсутствуют данные используем счетчик для каждой из таблиц (чтобы подгружать только что добавившиеся данные)

```
In [21]:
cur_count_movement = 0 # Изначально кол-во данных в нашей бд равно нулю
cur_count_rides = 0
cur_count_car_pool = 0
cur_count_drivers = 0
```

Для каждой из таблиц извлекаем данные с сервера, чтобы потом добавить их на другой сервер

Демонстрация извлечения данных для каждой из 4ех таблиц:

```
# SQL запрос извлекающий новые данные
query = f'''
        select * from main.movement
        limit 100000 offset {cur_count_movement};
        '''
cursor.execute(query)
data_movement = list(cursor.fetchall()) # Формируем список новых только что полученных данных

query = f'''
        select * from main.rides
        limit 100000 offset {cur_count_rides};
        '''
cursor.execute(query)
data_rides = list(cursor.fetchall())

query = f'''
        select * from main.car_pool
        limit 100000 offset {cur_count_car_pool};
        '''
cursor.execute(query)
data_car_pool = list(cursor.fetchall())

query = f'''
        select * from main.drivers
        limit 100000 offset {cur_count_drivers};
        '''
cursor.execute(query)

data_drivers = list(cursor.fetchall())
```

После того, как данные извлеклись, их необходимо добавить на другой сервер и загрузить в пустые таблицы (при этом новые данные также должны добавляться)

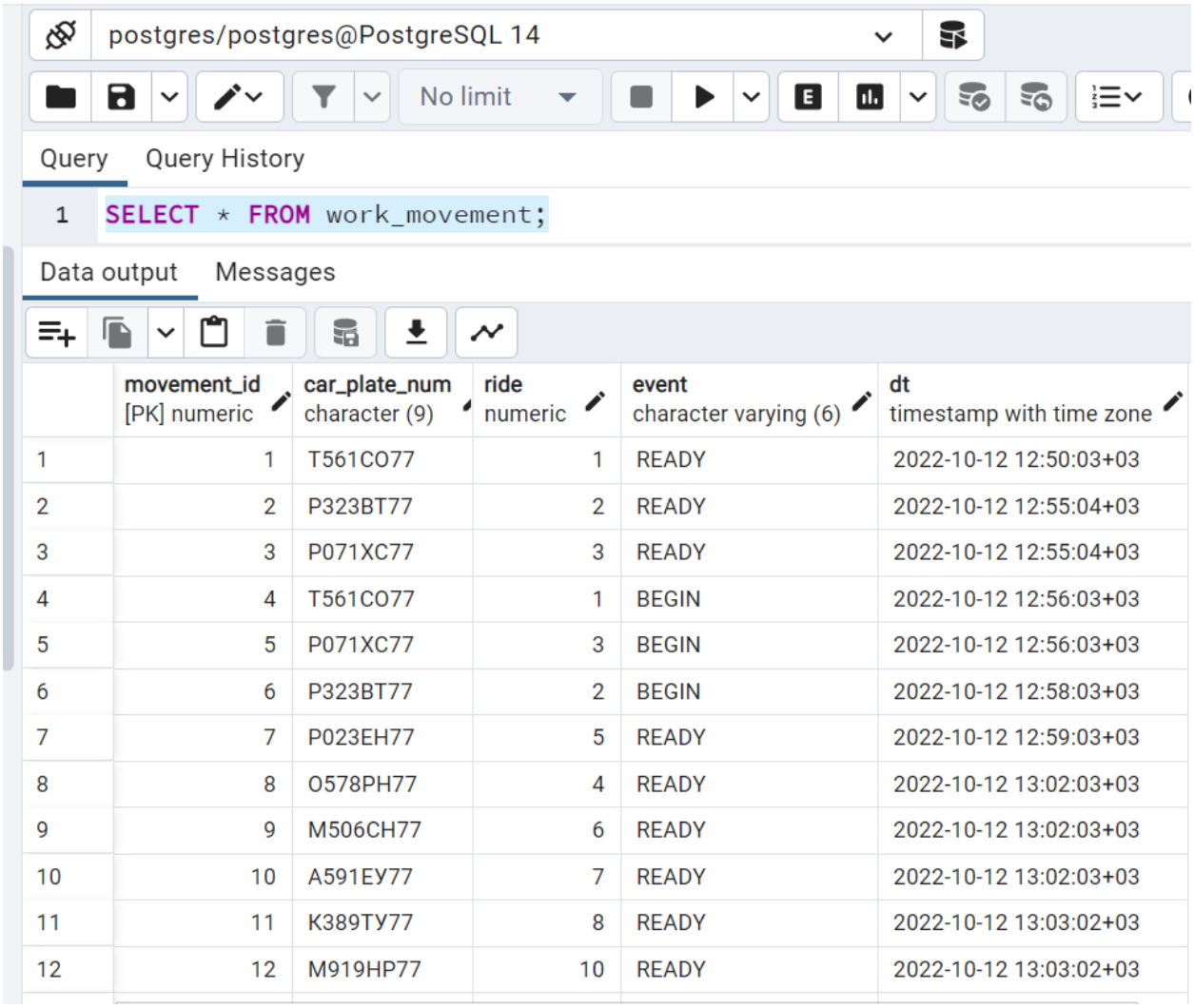
Для того, чтобы загрузить извлечённые данные на другой сервер, нужно подключиться к этому серверу (аналогично с извлечением данных)

```
conn = psycopg2.connect(dbname='taxi', user='etl_tech_user',
                        password='etl_tech_user_password', host='de-edu-db.chronosavant.ru')
cursor = conn.cursor()

```

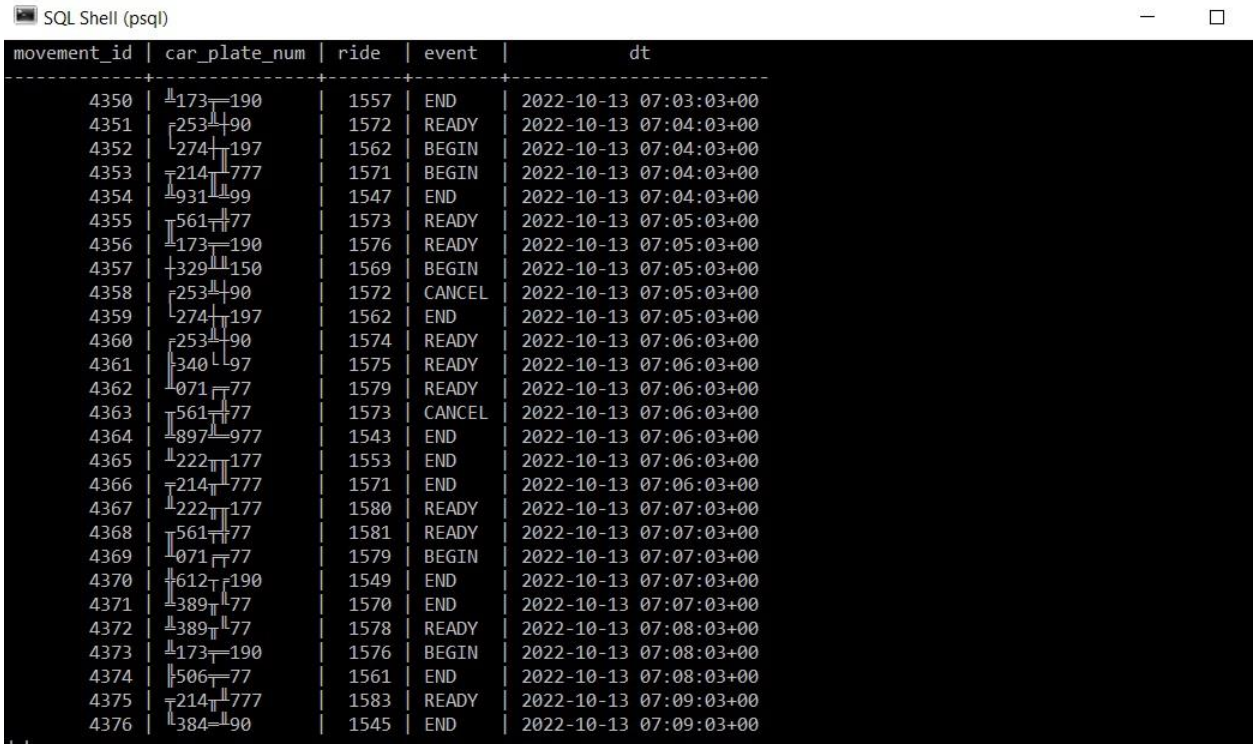
Далее идет процесс загрузки данных:

```python
# movement

query = 'INSERT INTO work_movement (movement_id, car_plate_num, ride, event, dt) VALUES (%s, %s, %s, %s, %s)'
for val in data_movement:
    cursor.execute(query, val)
conn.commit() #фиксируем изменения в бд

cursor.execute("select count(movement_id) from work_movement;")

cur_count_movement = cursor.fetchall()[0][0] #Обновляем количество данных в нашей бд.
print(cur_count_movement)
data_movement[:5]

# rides

query = 'INSERT INTO work_rides (ride_id, dt, client_phone, card_num, point_from, point_to, distance, price) VALUES (%s, %s,
for val in data_rides:
    cursor.execute(query, val)
conn.commit() #фиксируем изменения в бд


cursor.execute("select count(ride_id) from work_rides;")

cur_count_rides = cursor.fetchall()[0][0] #Обновляем количество данных в нашей бд.
print(cur_count_rides)
data_rides[:5]


## drivers

query = 'INSERT INTO work_drivers (driver_license, first_name, last_name, middle_name, driver_valid_to, card_num, update_dt)
for val in data_drivers:
    cursor.execute(query, val[:7])
conn.commit() #фиксируем изменения в бд


cursor.execute("select count(driver_license) from work_drivers;")

cur_count_rides = cursor.fetchall()[0][0] #Обновляем количество данных в нашей бд.
print(cur_count_drivers)
data_drivers[:5]


## car_pool

query = 'INSERT INTO work_car_pool (plate_num, model, rivision_dt, register_dt, finished_flg) VALUES (%s, %s, %s, %s, %s)'
for val in data_car_pool:
    cursor.execute(query, val[:5])
conn.commit() #фиксируем изменения в бд


cursor.execute("select count(plate_num) from work_car_pool;")

cur_count_rides = cursor.fetchall()[0][0] #Обновляем количество данных в нашей бд.
print(cur_count_car_pool)
data_car_pool[:5]
```

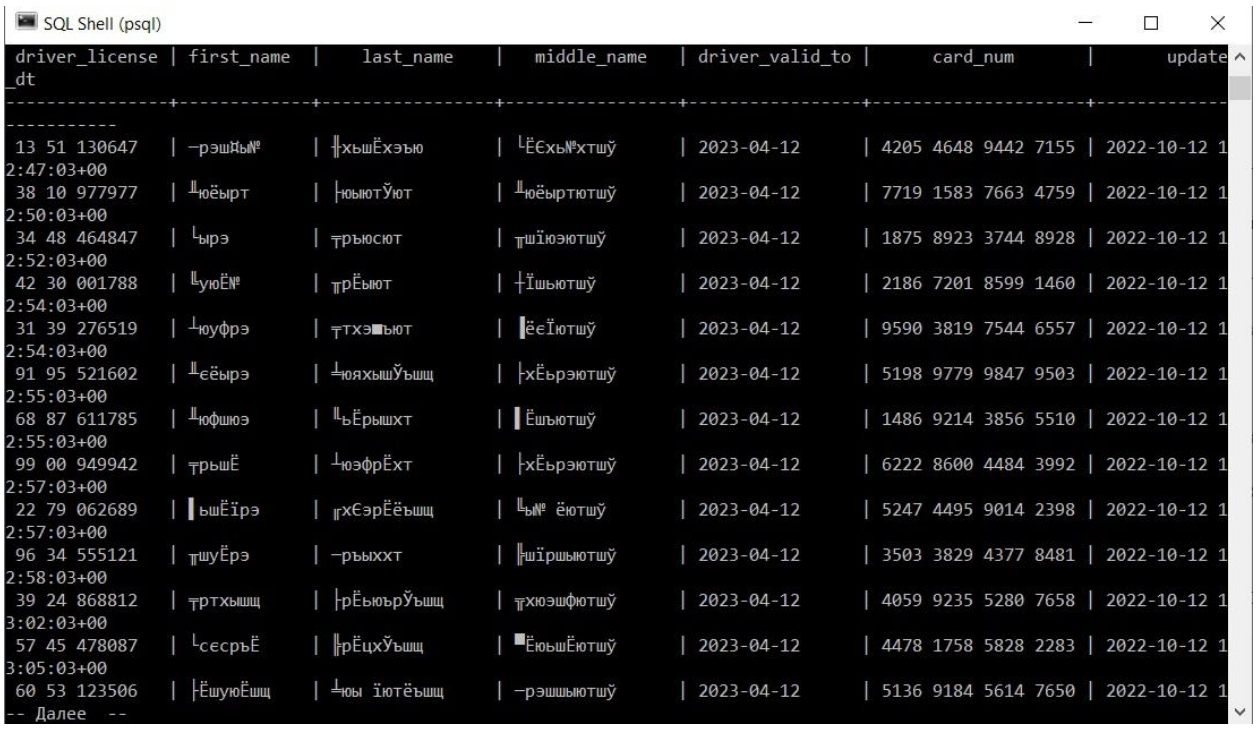

Демонстрация извлечения и загрузки данных на сервер (данные извлеклись одного сервера и загрузились на другой). Аналогично для остальных таблиц.



	movement_id [PK] numeric	car_plate_num character (9)	ride numeric	event character varying (6)	dt timestamp with time zone
1	1	T561CO77	1	READY	2022-10-12 12:50:03+03
2	2	P323BT77	2	READY	2022-10-12 12:55:04+03
3	3	P071XC77	3	READY	2022-10-12 12:55:04+03
4	4	T561CO77	1	BEGIN	2022-10-12 12:56:03+03
5	5	P071XC77	3	BEGIN	2022-10-12 12:56:03+03
6	6	P323BT77	2	BEGIN	2022-10-12 12:58:03+03
7	7	P023EH77	5	READY	2022-10-12 12:59:03+03
8	8	O578PH77	4	READY	2022-10-12 13:02:03+03
9	9	M506CH77	6	READY	2022-10-12 13:02:03+03
10	10	A591EY77	7	READY	2022-10-12 13:02:03+03
11	11	K389TY77	8	READY	2022-10-12 13:03:02+03
12	12	M919HP77	10	READY	2022-10-12 13:03:02+03



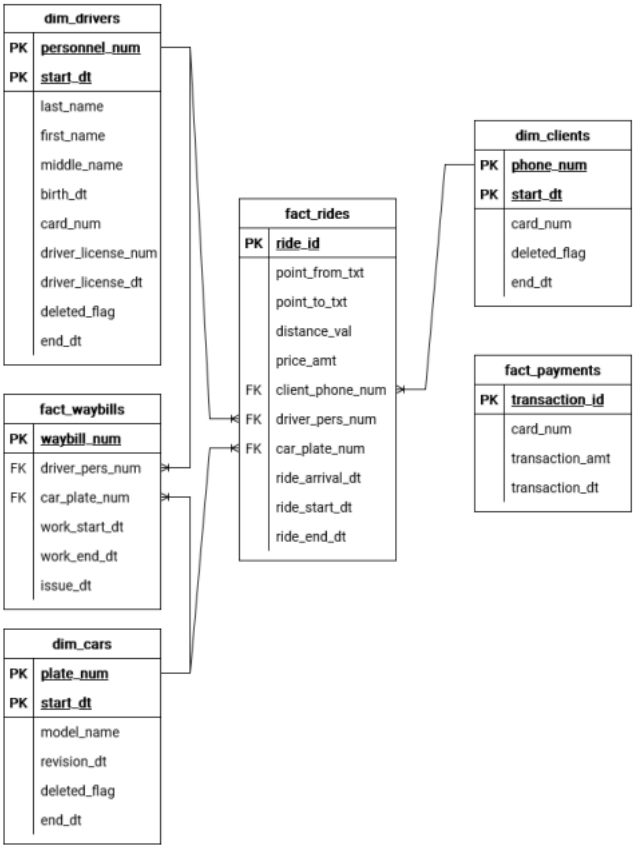

```
movement_id | car_plate_num | ride | event |           dt
------------+---------------+------+-------+------------------------
       4350 | ╙173╖190      | 1557 | END   | 2022-10-13 07:03:03+00
       4351 | ┌253┼90        | 1572 | READY | 2022-10-13 07:04:03+00
       4352 | └274┬197      | 1562 | BEGIN | 2022-10-13 07:04:03+00
       4353 | ╒214╥777      | 1571 | BEGIN | 2022-10-13 07:04:03+00
       4354 | ╙931╙99       | 1547 | END   | 2022-10-13 07:04:03+00
       4355 | ┌561╪77       | 1573 | READY | 2022-10-13 07:05:03+00
       4356 | ╙173╖190      | 1576 | READY | 2022-10-13 07:05:03+00
       4357 | ┼329╙150      | 1569 | BEGIN | 2022-10-13 07:05:03+00
       4358 | ┌253┼90        | 1572 | CANCEL| 2022-10-13 07:05:03+00
       4359 | └274┬197      | 1562 | END   | 2022-10-13 07:05:03+00
       4360 | ┌253┼90        | 1574 | READY | 2022-10-13 07:06:03+00
       4361 | ├340└97       | 1575 | READY | 2022-10-13 07:06:03+00
       4362 | ╙071┌77        | 1579 | READY | 2022-10-13 07:06:03+00
       4363 | ┌561╪77       | 1573 | CANCEL| 2022-10-13 07:06:03+00
       4364 | ╙897═977      | 1543 | END   | 2022-10-13 07:06:03+00
       4365 | ╙222╖177      | 1553 | END   | 2022-10-13 07:06:03+00
       4366 | ╒214╥777      | 1571 | END   | 2022-10-13 07:06:03+00
       4367 | ╙222╖177      | 1580 | READY | 2022-10-13 07:07:03+00
       4368 | ┌561╪77       | 1581 | READY | 2022-10-13 07:07:03+00
       4369 | ╙071┌77        | 1579 | BEGIN | 2022-10-13 07:07:03+00
       4370 | ╞612┌190       | 1549 | END   | 2022-10-13 07:07:03+00
       4371 | ╙389┐77        | 1570 | END   | 2022-10-13 07:07:03+00
       4372 | ╙389╖77        | 1578 | READY | 2022-10-13 07:08:03+00
       4373 | ╙173╖190      | 1576 | BEGIN | 2022-10-13 07:08:03+00
       4374 | ╟506╖77       | 1561 | END   | 2022-10-13 07:08:03+00
       4375 | ╒214╥777      | 1583 | READY | 2022-10-13 07:09:03+00
       4376 | ╙384═90       | 1545 | END   | 2022-10-13 07:09:03+00
```



Работоспособность кода проверяли на локальном сервере PjsgreSQl, после чего подключились к нужному серверу.

Данные извлеклись и загрузились.

Следующим шагом мы создаем в субд PosgreSQL данные таблицы (с помощью базовых команд языка SQL).



С помощью библиотеки *ftp lib* мы можем подключаться к серверу для выгрузки путевых листов, требуется подключиться к *ftp* серверу. Подключиться к серверу нам удалось, но извлечь данные не получилось.

```
:
1 #from ftplib import FTP_TLS
2 import ftplib
3
4 import ssl
5 import sys, os
```

```python
host = 'de-edu-db.chronosavant.ru'
ftp_user = 'etl_tech_user'
ftp_password = 'etl_tech_user_password'
port = 21
ftps = ftplib.FTP_TLS(host)

ftps.auth()
ftps.prot_p()
ftps.login(ftp_user, ftp_password)

ftps.cwd('/waybills')
ftps.cwd('/payments')

print(ftps.pwd())
#ftps.retrlines('LIST')
#ftps.nlst()
print(ftps.sendcmd('PASV'))
#ftps.set_pasv(21)
print(ftps.sendcmd('pwd'))

ftps.quit()
ftps.close()

```

```
/payments
227 Entering Passive Mode (10,128,0,31,171,46).
257 "/payments" is the current directory
```

ИТОГ:
Была проделана работа по извлечению и переносу данных с одного сервера на другой в необходимую базу данных

Данные были извлечены и загружены. Реализовано динамическое обновление данных.