**Лабораторная работа №9**

**Олейников Михаил Николаевич (olejnikov)**

В текущей работе вам потребуется использовать продукт RT.WareHouse и RT.Streaming, который развернут в рамках нашего курса. Если появятся вопросы, пожалуйста напишите в общий учебный чат или личным сообщением преподавателю

**Цель задания**: попрактиковаться с генерацией данных для очереди сообщений и сохранением данных в таблицах в GreenPlum.

1. Используйте Kafka сервер, который доступен на сервере “vm-strmng-s-1.test.local” и по порту 9092.

Напишите Python скрипт (используя библиотеку Python **kafka**) который будет генерировать поток данных (ровно одну тысячу сообщений) в формате json и складывать данные в ваш topic (используйте шаблон именования “lab10\_фамилия”, например **lab10\_yakupov**) очереди сообщения Kafka.

Протокол данных json модели:

{

**client:** “любое имя клиента”,

**opened:** “дата открытия заявки в тех поддержку”,

**priority:** 0 | 1 | 2

}

Где priority принимает одно значение из 3 возможных

- 0 - низкий приоритет заявки клиента

- 1 - средний приоритет заявки клиента

- 2 - высокий приоритет заявки клиента

Пример json данных

{

**client:** “Иванов”,

**opened:** “21.08.2023 13:00”,

**priority:** 2

}

Добейтесь в коде Python, чтобы ваш генератор создавал заявки в следующих пропорциях по приоритетам (значение для поля priority)

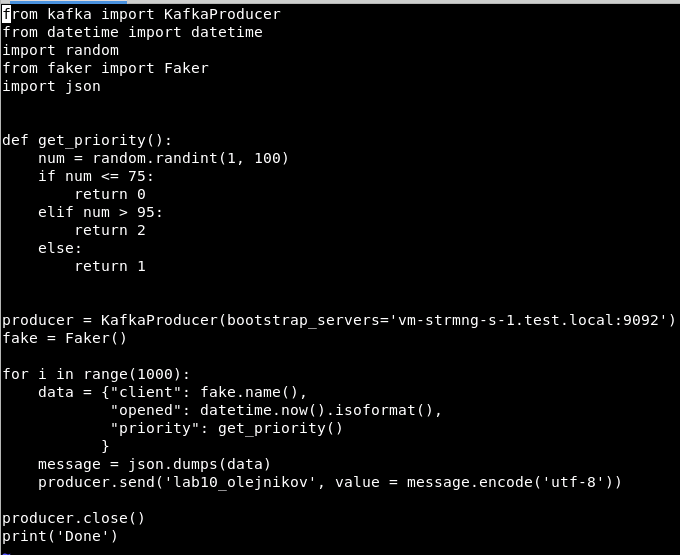
- низкий приоритет - 75 % всех сгенерированных данных

- средний приоритет - 20 % всех сгенерированных данных

- высокий приоритет - 5 % всех сгенерированных данных

пример подключения библиотеки и инициализации / закрытия продюсера

| from kafka import KafkaProducer producer = KafkaProducer(bootstrap\_servers='vm-strmng-s-1.test.local:9092') ... producer.close() |
| --- |



| from kafka import KafkaProducer from datetime import datetime import random from faker import Faker import json  def get\_priority():  num = random.randint(1, 100)  if num <= 75:  return 0  elif num > 95:  return 2  else:  return 1   producer = KafkaProducer(bootstrap\_servers='vm-strmng-s-1.test.local:9092') fake = Faker()  for i in range(1000):  data = {"client": fake.name(),  "opened": datetime.now().isoformat(),  "priority": get\_priority()  }  message = json.dumps(data)  producer.send('lab10\_olejnikov', value = message.encode('utf-8'))  producer.close() |
| --- |

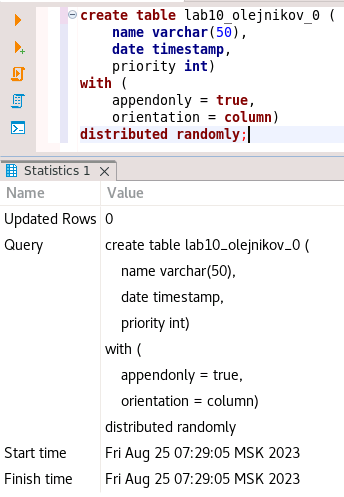
1. Создайте 3 физических таблицы в GreenPlum (формат: columnar, append only), используя шаблон именования

- “lab10\_фамилия\_0” (таблица для хранения заявок с низким приоритетом)

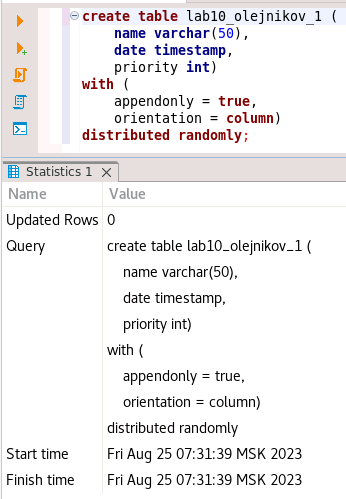
- “lab10\_фамилия\_1” (таблица для хранения заявок со средним приоритетом)

- “lab10\_фамилия\_2” (таблица для хранения заявок с высоким приоритетом)

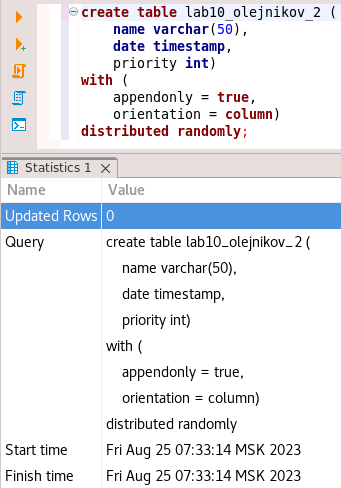
(например **lab10\_yakupov\_0**) Выберите ключ дистрибуции, какой по вашему мнению необходим для предотвращения SKEW аномалии (самый простой - DISTRIBUTED RANDOMLY).



| create table lab10\_olejnikov\_0 (  name varchar(50),  date timestamp,  priority int) with (  appendonly = true,  orientation = column) distributed randomly; |
| --- |



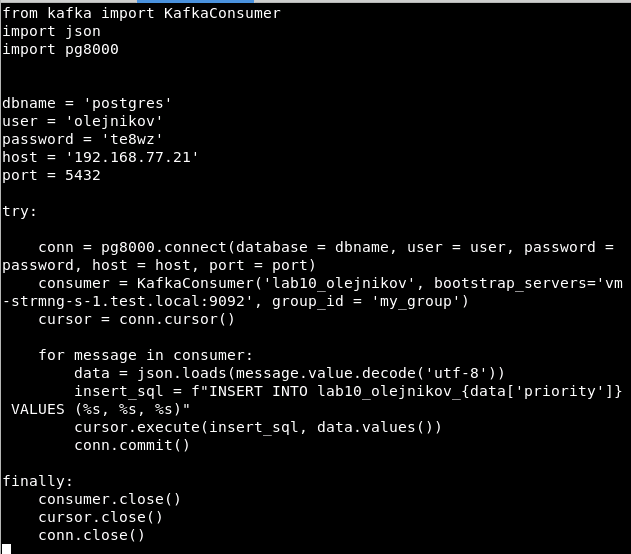
create table lab10\_olejnikov\_1 (  
 name varchar(50),  
 date timestamp,  
 priority int)  
with (  
 appendonly = true,  
 orientation = column)  
distributed randomly;



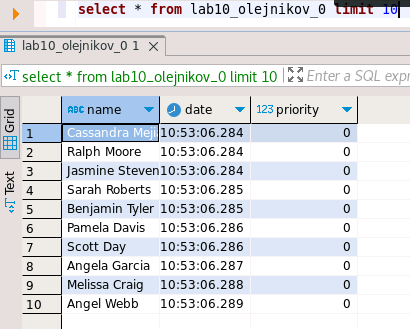
create table lab10\_olejnikov\_2 (  
 name varchar(50),  
 date timestamp,  
 priority int)  
with (  
 appendonly = true,  
 orientation = column)  
distributed randomly;

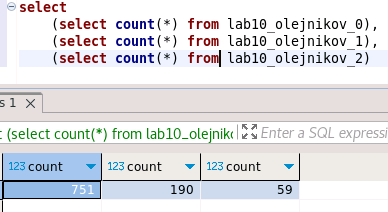
1. Создайте consumer на языке Python, который будет вычитывать данные из вашей очереди сообщений с шаблоном имени “lab10\_фамилия” и раскидать данные по таблицам в зависимости от значения поля **priority** в json данных.

| from kafka import KafkaConsumer import time import pg8000 dbname = 'postgres' user = '...' password = '...' host = '192.168.77.21' port = 5432 try: conn = pg8000.connect(database = dbname, user = user, password = password, host = host, port = port) consumer = KafkaConsumer('ваш topic', bootstrap\_servers = 'vm-strmng-s-1.test.local:9092', group\_id='my\_group') ... finally: consumer.close() conn.close() |
| --- |



| from kafka import KafkaConsumer import json import pg8000  dbname = 'postgres' user = 'olejnikov' password = 'te8wz' host = '192.168.77.21' port = 5432  try:  conn = pg8000.connect(database = dbname, user = user, password = password, host = host, port = port)  consumer = KafkaConsumer('lab10\_olejnikov', bootstrap\_servers='vm-strmng-s-1.test.local:9092', group\_id = 'my\_group')  cursor = conn.cursor()   for message in consumer:  data = json.loads(message.value.decode('utf-8'))  insert\_sql = f"INSERT INTO lab10\_olejnikov\_{data['priority']} VALUES (%s, %s, %s)"  cursor.execute(insert\_sql, data.values())  conn.commit()  finally:  consumer.close()  cursor.close()  conn.close() |
| --- |





(Предоставить преподавателю) приложите, пожалуйста, скрипты всех шагов в гугл документ и прикрепите ссылку в анкете. Не забудьте открыть доступ по ссылке.