Выполнил: Бойко Илья Группа: ПИН-б-о-22-1

Лабораторная работа №5: Потоковая обработка данных со Structured Streaming

1. Цель работы Освоить основы потоковой обработки данных с помощью Apache Spark Structured Streaming. Получить практические навыки чтения потоковых данных, выполнения скользящих агрегаций и визуализации результатов в реальном времени.

2. Используемый стек технологий

Язык программирования: Python (PySpark)

Фреймворк: Apache Spark

Язык программирования: Python

Инструментарий:Jupyter Lab / Jupyter Notebook

Данные: Структурированные данные в форматах JSON/CSV

3. Теоретические сведения

Structured Streaming — это масштабируемый и отказоустойчивый механизм потоковой обработки данных, построенный на движке Spark SQL.

Ключевые концепции:

- **Непрерывные инкрементальные вычисления:** Обработка данных происходит по мере их поступления
- **Event-time обработка:** Возможность обработки данных по времени события, а не времени получения
- **Водяные знаки (Watermarks):** Механизм обработки задержанных данных
- **Output modes:** Complete, Update, Append - различные режимы вывода результатов

Основные источники данных:

- Socket source (для тестирования)
- File source
- Kafka source

4. **Ход выполнения работы**

Часть 1: Настройка потока данных через Socket

1.1 Запуск netcat-сервера

nc -lk 9999

1.2 Создание тестовых данных

import time from datetime import datetime

test_data = ["2024-01-15 10:00:01,user1,click,5", "2024-01-15 10:00:02,user2,purchase,100", "2024-01-15 10:00:03,user1,click,3", "2024-01-15 10:00:04,user3,view,1", "2024-01-15 10:00:05,user2,click,7", "2024-01-15 10:00:06,user1,purchase,50", "2024-01-15 10:00:07,user4,click,2", "2024-01-15 10:00:08,user3,purchase,200", "2024-01-15 10:00:09,user5,view,1", "2024-01-15 10:00:10,user2,click,4", "2024-01-15 10:01:01,user1,click,6", "2024-01-15 10:01:02,user6,purchase,150", "2024-01-15 10:01:03,user3,click,3", "2024-01-15 10:01:04,user2,view,1", "2024-01-15 10:01:05,user4,purchase,80", "2024-01-15 10:02:01,user5,click,5", "2024-01-15 10:02:02,user1,purchase,120", "2024-01-15 10:02:03,user7,click,2", "2024-01-15 10:02:04,user3,view,1", "2024-01-15 10:02:05,user6,click,8"]

print("Тестовые данные подготовлены:") print("Формат: timestamp,user_id,action,value") print("Примеры:") for i, line in enumerate(test_data[:5], 1): print(f"{i}. {line}")

Часть 2: Чтение потоковых данных

2.1 Создание сессии Spark с поддержкой Streaming

from pyspark.sql import SparkSession from pyspark.sql.functions import * from pyspark.sql.types import * import warnings warnings.filterwarnings('ignore')

spark = SparkSession.builder
.appName("StructuredStreamingLab")
.config("spark.sql.shuffle.partitions", "2")
.config("spark.sql.streaming.schemaInference", "true")
.getOrCreate()

spark.sparkContext.setLogLevel("WARN")

print("Spark Session создана успешно!") print(f"Версия Spark: {spark.version}")

2.2 Определение схемы данных

schema = StructType([StructField("timestamp", TimestampType(), True), StructField("user_id", StringType(), True), StructField("action", StringType(), True), StructField("value", IntegerType(), True)])

print(" Схема данных определена:") print(schema)

2.3 Создание потокового DataFrame

stream_df = spark.readStream
.format("socket")

.option("host", "localhost")
.option("port", 9999)
.option("includeTimestamp", "true")
.load()

print(" Потоковый DataFrame создан") print(f" Схема потоковых данных: {stream_df.schema}")
print(f" Является ли потоковым: {stream_df.isStreaming}")

print("\n Пример данных (первые 5 строк):") if stream_df.isStreaming: # Для демонстрации создаем временный статический DataFrame demo_data = spark.createDataFrame([("2024-01-15 10:00:01,user1,click,5",), ("2024-01-15 10:00:02,user2,purchase,100",), ("2024-01-15 10:00:03,user1,click,3",), ("2024-01-15 10:00:04,user3,view,1",), ("2024-01-15 10:00:05,user2,click,7",)], ["value"])