Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут атомної та теплової енергетики
Кафедра цифрових технологій в енергетиці

ЗВІТ

з лабораторної роботи №9

з дисципліни «Розробка застосунків інтернету речей та сенсорних мереж»

Тема: «Застосування SparkSQL, робота з даними з використанням Dataframes і Dataset»

Варіант №17

Виконав:
Студент групи ТР-12
Ковальов Олександр Олексійович

Дата здачі: 13.03.2025

КИЇВ – 2025

Мета роботи. Набуття навичок роботи з датасетом RDD та інструментом SparkSQL. Отримання вмінь роботи з даними через використання Dataframes та Dataset, а також застосування практичних навичок.

Індивідуальне завдання:

1. Побудувати індекс типу RDD[(A, B)], в якому ключем є слово, а значенням – список посилань, на сторінках яких міститься це слово.
2. Побудувати тимчасову таблицю Spark SQL, в якій є 2 стовпчики:
   - значення метаданого;
   - посилання на сторінку, що має це значення метаданого.
3. Реалізувати пошук за словами на сторінці.
4. Реалізувати пошук за метаданими сторінок.

Хід роботи.

Лабораторна робота була виконана на Linux, дистрибутив Ubuntu. Якщо використовувати Windows, виникають проблеми пов’язані зі Spark.

Був встановлений Maven:

```
xairaven@Ubuntu-D:~/Documents/IOT$ mvn -v
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfcdc97d260186937)
Maven home: /usr/bin/apache-maven-3.9.9
Java version: 11.0.26, vendor: Ubuntu, runtime: /usr/lib/jvm/jav
64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "6.8.0-52-generic", arch: "amd64", fa
xairaven@Ubuntu-D:~/Documents/IOT$
```

Також, був встановлений JDK 11 (з будь яким JDK версії 11+ у Spark виникають проблеми):

```
xairaven@Ubuntu-D:~$ java --version
openjdk 11.0.26 2025-01-21
OpenJDK Runtime Environment (build 11.0.26+4-post-Ubuntu-1ubuntu124.04)
OpenJDK 64-Bit Server VM (build 11.0.26+4-post-Ubuntu-1ubuntu124.04, mixe
 sharing)
xairaven@Ubuntu-D:~$
```

Згенерований шаблон проекту:

```
...
[INFO] Parameter: basedir, Value: /home/xairaven/Documents/IOT/Lab9
[INFO] Parameter: package, Value: com.example
[INFO] Parameter: groupId, Value: com.example
[INFO] Parameter: artifactId, Value: myproject
[INFO] Parameter: packageName, Value: com.example
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: /home/xairaven/Document
/IOT/Lab9/myproject
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  05:08 min
[INFO] Finished at: 2025-03-13T17:38:38Z
[INFO] ------------------------------------------------------------------------
xairaven@Ubuntu-D:~/Documents/IOT/Lab9$ ls
myproject
```

Був доданий код до App.java:

```
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject/src/main/java/com/example$ cat
App.java | head -n5
package com.example;

import org.apache.spark.SparkConf;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaRDD;
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject/src/main/java/com/example$
```

Були зроблені зміни у файлі pom.xml:

```
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject$ cat pom.xml | head -n5
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/
2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/
maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.example</groupId>
  <artifactId>myproject</artifactId>
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject$
```

Створений jar файл:

```
[INFO]
[INFO] --- assembly:3.7.1:single (default-cli) @ myproject ---
[INFO] Building jar: /home/xairaven/Documents/IOT/Lab9/myproject/target/myproje
t-1.0-SNAPSHOT-jar-with-dependencies.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  24.620 s
[INFO] Finished at: 2025-03-13T18:05:05Z
[INFO] ------------------------------------------------------------------------
```

Був встановлений Spark:

```
xairaven@Ubuntu-D:~/Documents/IOT$ spark-submit --version
25/03/13 18:44:08 WARN Utils: Your hostname, Ubuntu-D resolves to a loopback address: 127.0.1.1;
8.0.106 instead (on interface enp0s8)
25/03/13 18:44:08 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.5
      /_/

Using Scala version 2.12.18, OpenJDK 64-Bit Server VM, 11.0.26
Branch HEAD
Compiled by user ubuntu on 2025-02-23T20:30:46Z
Revision 7c29c664cdc9321205a98a14858aaf8daaa19db2
Url https://github.com/apache/spark
Type --help for more information.
xairaven@Ubuntu-D:~/Documents/IOT$
```

Виведена таблиця в логах програми:

```
25/03/13 18:46:11 INFO CodeGenerator: Code generated in 362.66274 ms
25/03/13 18:46:12 INFO CodeGenerator: Code generated in 24.616186 ms
+---------+-----+
| metadata| link|
+---------+-----+
|metadata1|link1|
|metadata2|link2|
|metadata3|link3|
+---------+-----+
```

Згенеровані файли:

```
done
├── task1
│   ├── part-00000
│   ├── part-00001
│   └── _SUCCESS
├── task2
│   ├── part-00000-121e4ca0-f69c-4948-b781-67be4ffa1957-c000.csv
│   ├── part-00001-121e4ca0-f69c-4948-b781-67be4ffa1957-c000.csv
│   └── _SUCCESS
├── task3
│   ├── part-00000
│   ├── part-00001
│   └── _SUCCESS
└── task4
    ├── part-00000-0ffa30d0-8041-4a58-9122-614fe7ea8dd8-c000.csv
    └── _SUCCESS
```

Результати:

```
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject/target/done$ cat task1/part-00000
(word2,[page1, page2])
(word4,[page2, page3])
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject/target/done$ cat task1/part-00001
(word3,[page1, page2, page3])
(word1,[page1])
(word5,[page3])
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject/target/done$ cat task3/part-00000
page1
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject/target/done$ cat task3/part-00001
page2
xairaven@Ubuntu-D:~/Documents/IOT/Lab9/myproject/target/done$
```

На основі наданого коду всі завдання лабораторної роботи виконані. Побудовано індекс типу RDD[(A, B)], де ключем є слово, а значенням – список посилань на сторінки, що містять це слово. Реалізація здійснена за допомогою JavaPairRDD, де дані групуються за словами та відповідними сторінками, а результати збережені у файлі "task1".

Також створено тимчасову таблицю Spark SQL, що містить два стовпці: значення метаданого та посилання на сторінку, що має це значення. Таблиця була сформована у вигляді Dataset<Row> та зареєстрована через createOrReplaceTempView, після чого здійснено вибірку всіх даних. Отримані результати збережені у файлі "task2".

Реалізовано пошук за словами на сторінці шляхом фільтрації JavaPairRDD за конкретним словом, що дозволяє знаходити сторінки, де воно зустрічається. Відфільтровані результати записані у файл "task3".

Окрім цього, виконано пошук за метаданими сторінок за допомогою SQL-запиту до тимчасової таблиці Spark SQL. Було здійснено вибірку рядків, де значення метаданого відповідає заданому критерію, а результати збережено у файлі "task4".

Висновок: У ході виконання лабораторної роботи було набуте практичне розуміння роботи з Apache Spark, зокрема з RDD, Spark SQL, DataFrames та Dataset. Було реалізовано індексування слів у текстах за допомогою RDD, створено тимчасову таблицю Spark SQL для збереження метаданих сторінок, а також реалізовано пошук за словами та метаданими.

Отримані результати підтверджують ефективність використання розподіленої обробки даних у Spark. Завдяки використанню RDD вдалося організувати зручне представлення даних у вигляді індексу, а Spark SQL дозволив швидко виконувати запити до структурованих даних. Робота продемонструвала основні можливості Spark для обробки великих обсягів інформації, що є важливим для розробки сучасних систем аналізу даних.

Контрольні питання:

1. *Як Spark використовує RDD (стійкі розподілені набори даних), і чому RDD є ключовим компонентом у Spark?*
   Spark використовує RDD як основну структуру для зберігання та обробки даних у розподіленому середовищі. RDD забезпечує стійкість даних через механізм відновлення після збоїв та дозволяє виконувати обчислення в пам’яті, що значно прискорює обробку. Завдяки лінійній розбитості та паралельному виконанню RDD є ключовим компонентом, який дозволяє ефективно обробляти великі обсяги інформації.
2. *Як Spark SQL дає можливість обробляти структуровані дані, і як вона інтегрується з іншими компонентами Spark?*
   Spark SQL дозволяє обробляти структуровані дані за допомогою DataFrames і Datasets, використовуючи декларативний підхід, подібний до традиційного SQL. Вона інтегрується з іншими компонентами Spark, такими як RDD та MLlib, що дозволяє комбінувати SQL-запити з гнучкими обчисленнями та алгоритмами машинного навчання. Завдяки оптимізатору Catalyst забезпечується висока продуктивність запитів.
3. *Які основні операції можна виконувати за допомогою Spark SQL, і які переваги це надає порівняно з традиційними SQL-запитами?*
   За допомогою Spark SQL можна виконувати операції вибірки, фільтрації, групування, сортування та об’єднання даних. Порівняно з традиційними SQL-запитами, Spark SQL працює в розподіленому середовищі, що дає змогу

обробляти великі обсяги даних паралельно. Крім того, завдяки кешуванню та оптимізації Catalyst обчислення стають більш ефективними.

4. *Які є ключові компоненти Apache Spark, крім Spark SQL, і яку роль вони відіграють у всьому проєкті?*
   Крім Spark SQL, основними компонентами Apache Spark є Spark Core, який відповідає за розподілене виконання, Spark Streaming для обробки потокових даних, MLlib для машинного навчання та GraphX для роботи з графовими структурами. Кожен з них додає можливості для різних типів аналітики, забезпечуючи гнучкість та масштабованість платформи.
5. *Як Spark Streaming використовується для обробки потокових даних, і які джерела даних підтримуються?*
   Spark Streaming використовується для обробки потокових даних у реальному часі. Він підтримує різні джерела даних, такі як Apache Kafka, Flume, HDFS та сокети. Дані обробляються у вигляді мікробатчів, що дозволяє використовувати ті самі API, що й для обробки статичних даних, спрощуючи інтеграцію потокової та пакетної аналітики.
6. *Які можливості надає MLlib (Бібліотека машинного навчання) у контексті Spark, і які завдання з обробки даних вона допомагає розв'язувати?*
   MLlib надає широкий набір інструментів для машинного навчання, включаючи алгоритми класифікації, кластеризації, регресії та зниження розмірності. Вона дозволяє ефективно обробляти великі обсяги даних, використовуючи розподілені обчислення. MLlib спрощує побудову моделей машинного навчання та їх розгортання в середовищі Spark.
7. *Як GraphX допомагає подавати та обробляти графові структури даних у Spark?*
   GraphX дозволяє обробляти та аналізувати графові структури даних, такі як соціальні мережі або взаємозв'язки між об'єктами. Вона підтримує стандартні алгоритми обробки графів, такі як PageRank і трикутне злиття, а також забезпечує можливість перетворення даних між графами та іншими структурами Spark, такими як RDD та DataFrames.
8. *Які трансформації можна застосовувати до RDD, і яку роль вони відіграють у процесі обробки даних?*
   До основних трансформацій RDD належать map, filter, flatMap, groupByKey, reduceByKey, join та union. Вони дозволяють змінювати, фільтрувати та комбінувати дані у розподіленому середовищі. Трансформації є "ледачими", тобто виконуються лише при виклику дій, що допомагає оптимізувати виконання та ефективно використовувати ресурси.