**Lucrare de laborator nr. 1: Instalarea și Configurarea unui Cluster Hadoop**

**Obiectiv:**

Instalarea și configurarea unui cluster Hadoop și realizarea prelucrării datelor folosind programe MapReduce.

**Etape:**

1. **Instalarea Hadoop pe un cluster**
	* **Pașii:**
		1. Descărcarea și instalarea Hadoop pe mai multe noduri.
		2. Configurarea HDFS (Hadoop Distributed File System) și maparea directoarelor locale.
		3. Setarea și configurarea fișierelor de configurare (core-site.xml, hdfs-site.xml, yarn-site.xml).
	* **Rezultat așteptat:** Cluster Hadoop funcțional, cu noduri interconectate.
2. **Importul unui set de date în HDFS**
	* **Pașii:**
		1. Selectarea unui set de date din surse publice.
		2. Încărcarea setului de date în sistemul HDFS.
	* **Rezultat așteptat:** Setul de date este accesibil în HDFS pentru prelucrare ulterioară.
3. **Scrierea și rularea unui program MapReduce**
	* **Pașii:**
		1. Scrierea unui program simplu MapReduce (ex.: numărarea cuvintelor).
		2. Compilarea și rularea programului pe clusterul Hadoop.
		3. Vizualizarea rezultatelor procesării în HDFS.
	* **Rezultat așteptat:** Programul MapReduce rulează cu succes, iar rezultatele sunt stocate și accesibile în HDFS.

**Lucrare de laborator nr. 2: Prelucrarea și Analiza Datelor cu Apache Spark**

**Obiectiv:**

Dezvoltarea aplicațiilor Spark pentru analiza datelor, folosirea SparkSQL și implementarea soluțiilor de procesare a fluxurilor de date în timp real.

**Etape:**

1. **Crearea unei aplicații Spark pentru prelucrarea datelor**
	* **Pașii:**
		1. Crearea unui program Spark în Python/Scala pentru prelucrarea unui set mare de date.
		2. Executarea operațiilor de transformare (ex.: map, filter, reduce).
	* **Rezultat așteptat:** Datele sunt prelucrate cu succes și se generează output-ul dorit.
2. **Interogarea datelor folosind SparkSQL**
	* **Pașii:**
		1. Importul unui set de date în Spark.
		2. Crearea unui DataFrame și utilizarea SparkSQL pentru interogarea datelor.
	* **Rezultat așteptat:** Interogări SQL realizate pe setul de date cu succes, returnând rezultate exacte.
3. **Procesarea fluxurilor de date în timp real**
	* **Pașii:**
		1. Configurarea Spark Streaming pentru a primi date în flux.
		2. Scrierea unui program Spark Streaming pentru procesarea datelor în timp real (de exemplu, date de la senzori sau loguri).
	* **Rezultat așteptat:** Datele în flux sunt procesate și rezultatele sunt generate în timp real.
4. **Dezvoltarea vizualizărilor complexe și interactive**
	* **Pașii:**
		1. Utilizarea unor librării precum Matplotlib/Plotly pentru a vizualiza datele prelucrate.
		2. Dezvoltarea unor dashboard-uri interactive care să prezinte datele analizate.
	* **Rezultat așteptat:** Vizualizările interactive prezintă datele în mod clar și intuitiv.

**Lucrare de laborator Nr. 3: Utilizarea HBase și Cassandra**

**Cerințe**

1. **Documentarea (descrierea) caracteristicilor HBase și Cassandra și realizarea unei comparații succinte între ele.**
2. **Dezvoltarea unui scenariu practic de utilizare pe un set de date existent (Big Data).**
3. **Instalarea și configurarea uneia dintre tehnologii (HBase sau Cassandra).**
4. **Încărcarea setului de date ales în baza de date.**
5. **Procesarea și analiza datelor folosind interogări și operații de manipulare specifice tehnologiei selectate.**
6. **Prezentarea rezultatelor obținute prin capturi de ecran sau scripturi.**
7. **Formularea concluziilor privind utilitatea tehnologiei selectate pentru analiza Big Data.**

**Algoritmul de realizare**

1. **Alegerea setului de date – se utilizează un dataset public mare (ex. senzori IoT, date climatice, tranzacții, log-uri).**
2. **Selectarea tehnologiei – HBase sau Cassandra.**
3. **Instalarea și configurarea bazei de date alese (local, cluster sau Docker).**
4. **Crearea schemelor:**
	* **HBase – definirea unei tabele cu familii de coloane.**
	* **Cassandra – definirea unei tabele cu coloane și tipuri de date.**
5. **Încărcarea datelor în tabel (bulk load sau inserare progresivă).**
6. **Procesarea și analiza datelor:**
	* **selectarea unui subset de date,**
	* **filtrare după condiții,**
	* **agregări (count, average, max, min etc.),**
	* **actualizare și ștergere.**
7. **Interpretarea rezultatelor – ce informații relevante pot fi extrase din date.**
8. **Concluzii – argumentarea alegerii HBase sau Cassandra pentru scenariul respectiv.**

**Lucrare de laborator nr. 4: Analiza și Modelarea Datelor în Neo4j**

**Obiectiv:**

Utilizarea bazei de date graf Neo4j pentru crearea și interogarea grafurilor, preprocesarea datelor prin Neo4j dezvoltarea și implementarea unui model de învățare automată pentru seturi masive de date.

**Etape:**

1. **Crearea și interogarea unei baze de date graf în Neo4j**
	* **Pașii:**
		1. Instalarea Neo4j și configurarea bazei de date (instalarea bibliotecilor APOC și GDS).
		2. Definirea nodurilor (entităților) și relațiilor în baza de date graf.
		3. Scrierea interogărilor Cypher pentru a analiza relațiile dintre entități, precum:
			+ Găsirea nodurilor cu cele mai multe conexiuni.
			+ Identificarea drumurilor cele mai scurte între două entități.
			+ Detectarea comunităților sau clusterelor din graf.
		4. Importul unui set de date.
	* **Rezultat așteptat:** Baza de date graf este creată, iar interogările Cypher returnează corect relațiile și analizele dorite.
2. **Preprocesarea datelor direct în Neo4j**
	* **Pașii:**
		1. Utilizarea Neo4j pentru curățarea și normalizarea datelor (de exemplu, eliminarea nodurilor duplicate sau a relațiilor inutile).
		2. Optimizarea structurii grafurilor prin adăugarea de etichete suplimentare sau proprietăți pentru noduri și relații.
		3. Pregătirea datelor pentru analize mai complexe, cum ar fi calcularea metricilor de centralitate (centralitatea gradului, centralitatea intermediară etc.).
	* **Rezultat așteptat:** Datele sunt preprocesate și structura grafului sunt pregătite pentru antrenarea modelelor de învățare automată.
3. **Implementarea unui model de machine learning pe baza grafurilor în Neo4j**
	* **Pașii:**
		1. Utilizarea algoritmilor încorporați în Neo4j pentru analize grafice (de exemplu, PageRank, algoritmi de centralitate, comunități grafice).
		2. Selectarea unui set de caracteristici extrase din graf (cum ar fi centralitatea unui nod) pentru a antrena un model de predicție bazat pe rețelele de relații.
		3. Utilizarea funcționalităților Neo4j de ML pentru antrenarea unui model de învățare automată (de ex., pentru a prezice legăturile între entități sau pentru a clasifica nodurile).
	* **Rezultat așteptat:** Modelul de ML este antrenat pe baza datelor din graf și poate face clasificări relevante sau predicții.
4. **Testarea, evaluarea și vizualizarea rezultatelor în Neo4j**
	* **Pașii:**
		1. Testarea modelului antrenat pe un set de date de test, folosind metrici cum ar fi precizia, recall-ul sau AUC (pentru probleme de clasificare).
		2. Vizualizarea rezultatelor direct în Neo4j, cu ajutorul interfeței de vizualizare grafică pentru a înțelege performanța modelului și a observa relațiile de predicție.
		3. Salvarea și prezentarea vizuală a grafurilor rezultate, inclusiv modificările apărute după aplicarea modelului de predicție (de exemplu, relații noi sau rearanjarea nodurilor).
	* **Rezultat așteptat:** Rezultatele modelului de ML sunt vizualizate și analizate, iar predicțiile sunt corelate cu structura grafului.

**Лабораторная работа № 1: Обработка данных с помощью Hadoop и HDFS**

**Цель:**

Установка и настройка кластера Hadoop и выполнение обработки данных с помощью программ MapReduce.

**Этапы:**

1. **Установка Hadoop на кластер**
	* **Шаги:**
		1. Загрузка и установка Hadoop на несколько узлов.
		2. Настройка HDFS (Hadoop Distributed File System) и маппинг локальных каталогов.
		3. Настройка конфигурационных файлов (core-site.xml, hdfs-site.xml, yarn-site.xml).
	* **Ожидаемый результат:** Рабочий кластер Hadoop с подключенными узлами.
2. **Импорт набора данных в HDFS**
	* **Шаги:**
		1. Выбор набора данных из публичных источников (например, климатические или социальные данные).
		2. Загрузка набора данных в систему HDFS.
	* **Ожидаемый результат:** Набор данных доступен в HDFS для последующей обработки.
3. **Написание и выполнение программы MapReduce**
	* **Шаги:**
		1. Написание простой программы MapReduce (например, подсчет слов).
		2. Компиляция и запуск программы на кластере Hadoop.
		3. Просмотр результатов обработки в HDFS.
	* **Ожидаемый результат:** Программа MapReduce успешно выполнена, результаты хранятся и доступны в HDFS.

**Лабораторная работа № 2: Обработка и анализ данных с помощью Apache Spark**

**Цель:**

Разработка приложений Spark для анализа данных, использование SparkSQL и реализация решений для обработки потоков данных в реальном времени.

**Этапы:**

1. **Создание приложения Spark для обработки данных**
	* **Шаги:**
		1. Написание программы Spark на Python/Scala для обработки большого набора данных.
		2. Выполнение операций трансформации (например, map, filter, reduce).
	* **Ожидаемый результат:** Данные успешно обработаны и сгенерирован нужный результат.
2. **Запросы к данным с использованием SparkSQL**
	* **Шаги:**
		1. Импорт набора данных в Spark.
		2. Создание DataFrame и выполнение запросов SQL с использованием SparkSQL.
	* **Ожидаемый результат:** Запросы SQL успешно выполнены, возвращая правильные результаты.
3. **Обработка потоков данных в реальном времени**
	* **Шаги:**
		1. Настройка Spark Streaming для получения данных в реальном времени.
		2. Написание программы Spark Streaming для обработки данных в реальном времени (например, данные с сенсоров или логов).
	* **Ожидаемый результат:** Потоковые данные обработаны, результаты генерируются в реальном времени.
4. **Создание сложных и интерактивных визуализаций**
	* **Шаги:**
		1. Использование библиотек, таких как Matplotlib/Plotly, для визуализации обработанных данных.
		2. Разработка интерактивных панелей для представления проанализированных данных.
	* **Ожидаемый результат:** Интерактивные визуализации представляют данные в понятной и наглядной форме.

**Лабораторная работа № 3: Использование HBase и Cassandra**

**Требования**

1. **Документирование** характеристик HBase и Cassandra и краткое сравнение этих технологий.
2. **Разработка** практического сценария использования на основе существующего набора данных (Big Data).
3. **Установка и настройка** одной из технологий (HBase или Cassandra).
4. **Загрузка** выбранного набора данных в базу данных.
5. **Обработка и анализ данных** с использованием запросов и операций, специфичных для выбранной технологии.
6. **Представление результатов** в виде скриншотов или скриптов.
7. **Формулирование выводов** о полезности выбранной технологии для анализа больших данных.

**Алгоритм выполнения**

1. **Выбор набора данных** – используется публичный крупный датасет (например, данные сенсоров IoT, климатические данные, транзакции, логи).
2. **Выбор технологии** – HBase или Cassandra.
3. **Установка и настройка** выбранной базы данных (локально, в кластере или через Docker).
4. **Создание схемы**:
	* HBase – определение таблицы с семействами колонок.
	* Cassandra – определение таблицы с колонками и типами данных.
5. **Загрузка данных** в таблицу (bulk load или поэтапная вставка).
6. **Обработка и анализ данных**:
	* выборка подмножества данных,
	* фильтрация по условиям,
	* агрегаты (count, average, max, min и т.д.),
	* обновление и удаление данных.
7. **Интерпретация результатов** – какие значимые сведения можно извлечь из данных.
8. **Выводы** – аргументация выбора HBase или Cassandra для данного сценария.

**Лабораторная работа №4: Анализ и моделирование данных в Neo4j**
**Цель:**
Использование графовой базы данных Neo4j для создания и запросов графов, предобработки данных с помощью Neo4j, разработки и внедрения модели машинного обучения для массивных наборов данных.

**Этапы:**

1. Создание и выполнение запросов в графовой базе данных Neo4j
**Шаги:**
a) Установка Neo4j и настройка базы данных (установка библиотек APOC и GDS).
b) Определение узлов (сущностей) и отношений в графовой базе данных.
c) Написание запросов Cypher для анализа отношений между сущностями, например:
	* Поиск узлов с наибольшим количеством связей.
	* Определение кратчайших путей между двумя сущностями.
	* Обнаружение сообществ или кластеров в графе.
	d) Импорт набора данных.

**Ожидаемый результат:** Графовая база данных создана, и запросы Cypher правильно возвращают запрашиваемые отношения и анализы.

1. Предобработка данных напрямую в Neo4j
**Шаги:**
a) Использование Neo4j для очистки и нормализации данных (например, удаление дублирующихся узлов или ненужных связей).
b) Оптимизация структуры графов путем добавления дополнительных меток или свойств к узлам и отношениям.
c) Подготовка данных для более сложного анализа, такого как расчет метрик центральности (центральность по степени, центральность посредничества и т.д.).

**Ожидаемый результат:** Данные предобработаны, и структура графа готова для обучения моделей машинного обучения.

1. Внедрение модели машинного обучения на основе графов в Neo4j
**Шаги:**
a) Использование встроенных в Neo4j алгоритмов для графового анализа (например, PageRank, алгоритмы центральности, графовые сообщества).
b) Выбор набора признаков, извлеченных из графа (например, центральность узла), для обучения предсказательной модели на основе сетевых связей.
c) Использование функционала машинного обучения Neo4j для обучения модели машинного обучения (например, для предсказания связей между сущностями или классификации узлов).

**Ожидаемый результат:** Модель машинного обучения обучена на данных графа и способна выполнять релевантные классификации или предсказания.

1. Тестирование, оценка и визуализация результатов в Neo4j
**Шаги:**
a) Тестирование обученной модели на тестовом наборе данных с использованием таких метрик, как точность, полнота или AUC (для задач классификации).
b) Визуализация результатов непосредственно в Neo4j с помощью графического интерфейса для понимания производительности модели и наблюдения за предсказанными связями.
c) Сохранение и визуальное представление итоговых графов, включая изменения после применения предсказательной модели (например, новые связи или перестановка узлов).

**Ожидаемый результат:** Результаты модели машинного обучения визуализированы и проанализированы, а предсказания сопоставлены со структурой графа.