Proyecto BDA

1 EXTRACCIÓN de datos

En la extracción de datos se cargará información a s3. Se hará en su totalidad usando Spark, no es válido utilizar librerías como boto3, salvo para crear el bucket.

1.1 Archivos

Será necesario cargar los siguientes archivos de datos **directamente** a S3.

- **restaurantes.json**: Este archivo contiene una lista de datos de restaurantes generados. Cada entrada de restaurante incluye un ID, un nombre (empresa) y un ID de hotel asignado aleatoriamente.
- **habitaciones.csv:** Este archivo contiene datos de habitaciones generados en formato CSV. Cada entrada de habitación incluye un número de habitación, categoría y tarifa nocturna.

2 MongoDB

Será necesario **obtener** los datos desde **MongoDB** y posteriormente mandar esa información a través de **Kafka**.

Tendréis que insertar los datos en MongoDB con un archivo que se encontrará en data_generation: **mongoInsert.py** y en otro archivo mandar los datos directamente por Kafka leyendo desde MongoDB, sin usar archivos de por medio: **mongoProducer.py.** Estos archivos no usarán Spark.

El **consumer,** que leerá los datos del stream, sí **utiliza spark.**

- **Stream**: clientes_stream
- **clientes.json:** Este archivo contiene una lista de datos de clientes generados. Cada entrada de cliente incluye un ID, nombre, dirección y preferencias dietéticas.

3 Neo4j

Tendréis que insertar los datos en Neo4j con un archivo que se encontrará en data_generation: **neo4jInsert.py** y en otro archivo mandar los datos directamente por Kafka desde Neo4j, sin usar archivos de por medio: **neo4jProducer.py.** Estos archivos no usarán Spark.

El **consumer,** que leerá los datos del stream, sí **utiliza spark.**

- **Stream**: menus_stream
- **menu.csv:** este archivo contiene los datos de menú generados en formato CSV. Cada entrada de menú incluye un ID de menú, precio, disponibilidad y el ID del restaurante asociado.
- **platos.csv**: este archivo contiene datos de platos generados en formato CSV. Cada entrada de plato incluye un ID de plato, nombre, ingredientes y alérgenos.
- **relaciones.json:** Este archivo contiene una lista de relaciones generadas entre menús y restaurantes. Cada entrada incluye un ID de menú y un ID de restaurante.

3.1 Postgres

Esta información vendrá desde Postgres, para ellos tendréis que leer dos archivos csv y cargarlos en una BBDD de postgres con un archivo que se encontrará en data_generation: **postgresInsert.py**.

Más adelante tendréis que leer esa información utilizando Spark a través de la BBDD.

- **Puerto BBDD:** 9999
- **Nombre BBDD:** PrimOrd
- **Nombre usuario:** primOrd
- **Password usuario:** bdaPrimOrd

- **empleados.csv:** Este archivo contiene los datos generados de los empleados en formato CSV. Cada entrada de empleado incluye un ID de empleado, nombre, puesto y fecha de contratación.

- **hoteles.csv:** este archivo contiene datos de hoteles generados en formato CSV. Cada entrada de hotel incluye un ID de hotel, el nombre del hotel, la dirección y una lista de ID de empleados asociados a él.

3.2 Kafka

Esta información vendrá desde kafka, para ellos tendréis que **leer un archivo txt, convertirlo** al formato **json** y **enviarlo** por **kafka**.

3.2.1 reservas.txt

Descripción: Fichero de texto para datos de reservas con separación única:

**** Reserva 1 ****
ID Cliente: 23
Fecha Llegada: 2024-04-20
Fecha Salida: 2024-04-25
Tipo Habitacion: Suite
Preferencias Comida: Vegetariano
Id Habitacion: 15
ID Restaurante: 17

Cada reserva se indica con "*** Reserva {id} ***".
Para cada reserva, se indica el ID del cliente, la fecha de llegada, la fecha de salida, el tipo de habitación, las preferencias dietéticas, el ID de la habitación y el ID del restaurante.

La información se leerá una vez introducida en kafka a través de Spark, es decir, el **productor** de la información **no utiliza spark,** pero el **consumer** sí **utiliza spark.**

4 TRANSFORMACIÓN de datos

En esta etapa, los datos ya estarán en s3 con LocalStack y tendréis que prepararlos para el apartado de análisis, modificad lo que veais conveniente. **No hace falta hacer ningún tratamiento de errores, vacíos, etc. Es opcional.**

5 LOAD: Data Warehouse

5.1 Data loading

Los datos transformados se cargarán en Postgres para su análisis posterior en 4 tablas distintas que responderán a las preguntas del Data analytics. Solo poner la información de cada tabla que sea interesante para resolver estas preguntas.

5.2 Data analytics

Usando Apache Spark tenéis que obtener los datos a través de postgres y realizar consultas que contengan análisis avanzados sobre los datos almacenados en el almacén de datos.

5.2.1 Análisis de las preferencias de los clientes

¿Cuáles son las preferencias alimenticias más comunes entre los clientes?

5.2.2 Análisis del rendimiento del restaurante:

¿Qué restaurante tiene el precio medio de menú más alto?
¿Existen tendencias en la disponibilidad de platos en los distintos restaurantes?

5.2.3 Patrones de reserva

¿Cuál es la duración media de la estancia de los clientes de un hotel?
¿Existen periodos de máxima ocupación en función de las fechas de reserva?

5.2.4 Gestión de empleados

¿Cuántos empleados tiene de media cada hotel?

5.2.5 Ocupación e ingresos del hotel

¿Cuál es el índice de ocupación de cada hotel y varía según la categoría de habitación?
¿Podemos estimar los ingresos generados por cada hotel basándonos en los precios de las habitaciones y los índices de ocupación?

5.2.6 Análisis de menús

¿Qué platos son los más y los menos populares entre los restaurantes?

¿Hay ingredientes o alérgenos comunes que aparezcan con frecuencia en los platos?

5.2.7 Comportamiento de los clientes

¿Existen pautas en las preferencias de los clientes en función de la época del año?
¿Los clientes con preferencias dietéticas específicas tienden a reservar en restaurantes concretos?

5.2.8 Garantía de calidad

¿Existen discrepancias entre la disponibilidad de platos comunicada y las reservas reales realizadas?

5.2.9 Análisis de mercado

¿Cómo se comparan los precios de las habitaciones de los distintos hoteles y existen valores atípicos?

6 Hadoop

Hacer un MapReduce y un archivo Pig Latin que responda a la siguiente pregunta usando un archivo csv dado en los apartados anteriores:
¿Cuántas reservas se hicieron al mes?

7 Estructura del proyecto

Para este proyecto tendréis que seguir la siguiente estructura:

data_Prim_ord/: Esta carpeta contiene todos los datos necesarios para el análisis.

- **json/**: Aquí se almacenan los archivos json.
- **text/:** Aquí se almacena el archivo txt
- **csv/:** Esta carpeta almacena los archivos csv.

data_generation/: Esta carpeta contiene todos los archivos python que generen los archivos anteriores, **tienen que generarlos en la carpeta especificada.**

apps_Prim_ord/: Esta carpeta contiene el resto de archivos py.

- **data_integration.py**: Almacena datos en s3
- **data_transformation.py**: Organiza/une los datos, es opcional.

- **data_load.py**: Guarda los datos en las 4 tablas distintas de Postgres.
- **data_analysis.py**: documento donde se realiza el análisis de datos.

hadoop/: Esta carpeta contiene los archivos asociados a Hadoop, el .Java y .pig

docker/: Esta carpeta tiene todos los Dockerfile usados.

localstack/: Esta carpeta tiene todos los archivos que trabajan con localstack usando boto3, solo debería ser de test o para crear los buckets.

scripts/: Esta carpeta contiene todos los archivos de ejecución de servicios del Dockerfile.

docker-compose.yml: el yml de vuestro cluster.

Si existe algún archivo más que sea necesario **comentarlo** con el profesor.

8 Evaluación

Me tendréis que pasar un **enlace a GitHub** con vuestro repo con el código. Si veo que hay muy pocos commits, o un commit con toda la información y el resto cambiando poco este trabajo se considerará suspendido.

El proyecto se evaluará en una presentación el día **17 de mayo** donde tendréis que hacer una demo de la aplicación y responder a dudas del proyecto.