CUNEF ETL - Spark - Evaluación

Master DS - Leonardo Hansa

Tienes una base de datos con información sobre alojamientos de Airbnb. La tarea consiste en:

- extraer parte de esos datos,
- construir nuevas tablas transformándolos con ciertas directrices
- y subirlas de nuevo a la base de datos.

Objetivo final

Las dos tablas finales que crearás y subirás a la base de datos serán las siguientes:

- **Tabla 1.** Evolución mensual del número de críticas por distrito, con predicción para el mes siguiente.
- **Tabla 2.** Distribución del tipo de alojamiento `room_type` por distrito. Incluye:
  - Nota media ponderada (`review_scores_rating` ponderado con `number_of_reviews`).
  - Precio mediano (`price`).
  - Número de alojamientos (`id`).

Datos

Los datos los tienes repartidos en tres ficheros:

- `listings_redux.parquet` Datos sobre los alojamientos de Madrid listados en julio de 2021.
- `reviews.parquet` Listado con comentarios hechos sobre cada alojamiento, con fecha e identificador del inquilino.
- `neighborhoods.parquet` Relación entre barrio (`neighbourhood`) y distrito (`neighbourhood_group`).

Herramientas

- Realiza los ejercicios de **extracción** en Spark y genera data frames de R o Python a partir de tus consultas. **Extrae (haz collect) únicamente de los datos necesarios.** (valoraré muy negativamente que descargues datos de más).
- El resto del trabajo lo puedes hacer en R ó Python. Si lo haces todo en R, tendrás un punto más en la nota final de la asignatura.
- La entrega será un script que funcione: o sea, lo ejecutaré y al final deberé tener cargadas en la base de datos las tablas que se piden.

Puntuación

La práctica cuenta 4 puntos de la nota final de la asignatura. Si la haces en R (incluida la parte de Spark, con sparklyr), tendrás un punto más (aunque si suspendes el examen, el punto de más no influirá y la asignatura estará suspensa).

Tareas

De ahora en adelante tienes detalles de los pasos necesarios.

Extracción

1. **Extracción (listings)** Crea un data frame de `pandas` o de R a partir del fichero `listings_redux.parquet` con las consideraciones que se indican a continuación. Con Spark, haz un *join* con el fichero `neighbourhoods.parquet` para añadir el dato de distrito (`neighbourhood_group`) y asegúrate de que extraes esta columna en el data frame en lugar de `neighbourhood`. Para saber qué columnas necesitas, tendrás que seguir leyendo la práctica para ver qué es lo que irás necesitando.

2. **Extracción (reviews)** Trabaja en Spark la información del fichero `reviews.parquet` con las consideraciones siguientes. Crea un data frame de R o pandas con la tabla final.

    - Con Spark, haz un *join* con la información del fichero `neighbourhoods.parquet` para añadir el dato de distrito (`neighbourhood_group`) y asegúrate de que extraes esta columna en el data frame en lugar de `neighbourhood`.
    - También en Spark, cuenta a nivel de distrito y mes el número de reviews. Para calcular el mes a partir de una fecha te vendrá bien la documentación oficial de pyspark y el método strftime (en este link) o esta respuesta en StackOverflow sobre sparklyr.
    - Además, extrae los datos desde 2011 en adelante (también con Spark).

Transformación

3. **Transformación (listings).** Antes de realizar la agregación que se pide, tienes que tratar las columnas `price,`, `number_of_reviews` y `review_scores_rating`. Empieza con el precio (de las otras dos columnas te encargarás en el siguiente ejercicio). Necesitas pasarla a numérica. Ahora mismo es de tipo texto y lo primero que necesitamos es quitar símbolos raros. Tanto R como Python sabe convertir un texto como `"15.00"` a número, pero no saben convertir `"$1,400.00"`. Tienes que quitar tanto el símbolo del dólar como la coma. En expresiones regulares, el símbolo del dólar se usa para una cosa muy concreta, así que necesitarás usar algo como `"\\$"` (lo que se conoce como *escapar*).

4. **Transformación (listings).** Toca imputar los valores missing de `number_of_reviews` y `review_scores_rating`. Normalmente en estos casos se habla con la gente que más usa los datos y se llega con ellos a un acuerdo de cómo se imputaría esta información. En este caso, imputa los valores missing con valores reales dentro de la tabla, a nivel de `room_type`, escogidos de manera aleatoria. Es decir, si hay un valor missing en `number_of_reviews` para un registro con `room_type == "Entire home/apt"`, lo reemplazarías con un valor aleatorio de esa misma columna para los que `room_type` sea `"Entire home/apt"`. Tienes libertad para plantear esto como te resulte más cómodo. *Pista.* Yo he hecho un bucle *for()* con R base (sí, lo nunca visto en mí :P)

5. **Transformación (listings).** Con los missing imputados y el precio en formato numérico ya puedes agregar los datos. A nivel de distrito y de tipo de alojamiento, hay que calcular:

   - Nota media ponderada (`review_scores_rating` ponderado con `number_of_reviews`).
   - Precio mediano (`price`).
   - Número de alojamientos (`id`).

La tabla resultante tendrá cuatro columnas: distrito (llamada habitualmente `neighbourhood_group`), tipo de alojamiento (`room_type`), nota media y precio mediano. Esta tabla puede ser útil para estudiar diferencias entre mismo un tipo de alojamiento en función del distrito en el que esté.

6. **Transformación (reviews).** La mayor parte de la transformación para *Reviews* la has hecho ya con SQL. Vamos a añadir ahora a simular que tenemos un modelo predictivo y lo vamos a aplicar sobre nuestros datos. Así, la tabla que subamos de nuevo a la base de datos tendrá la predicción añadida. El último mes disponible es julio, así que daremos la predicción para agosto. Esto no es una asignatura de predicción de series temporales, así que nos vamos a conformar con tomar el valor de julio como predicción para agosto (a nivel de distrito). Es decir, si el dato en `"Centro"` para julio es de 888 reviews, añadiremos una fila con los valores `"Centro"`, `"2021-08"` y `888`, así para cada distrito. Tienes libertad para plantearlo como veas adecuado. **Al final, deja el data frame ordenado a nivel de distrito y mes.** *Pista.* Yo he creado un data frame nuevo con todas estas predicciones y lo he apilado al data frame original. Esto se puede

hacer con la función `bind_rows()` de dplyr o el método `append()` o función `concat()` de pandas.

7. (*este es lío*) **Transformación (reviews).** Hay casos que no tienen dato, por ejemplo, febrero de 2011 en Arganzuela. Como no hay dato, asumiremos que es 0. Siguiendo esta idea, añade todos los registros necesarios a la tabla. Puedes hacerlo de la manera que te resulte más intuitiva. **Recuerda ordenar la tabla final por distrito y mes.** *Pista.* Yo he creado primero un vector con todas las fechas posibles y otro con los posibles distritos. Con esos vectores hago un data frame de dos columnas, con todas las combinaciones posibles entre meses y distritos. Hay muchas formas de hacer eso. Luego hago un *full join* con los datos originales. Si después del *join* la columna *reviews* tiene valor missing, es que no estaba en el caso original. Sustituyo esos missing por ceros y ya tengo la tabla final.

8. **Carga.** Guarda en formato parquet las dos tablas que has creado. No sobreescibas los que tienes: crea dos ficheros nuevos. Haz una prueba de que todo está en orden, mostrando las primeras líneas de cada fichero. Si la fecha tiene un formato raro, es posible que necesites definirla en el data frame como tipo texto.

Master in DS

ETL