Desafio
Programa de Trainee iBlue

Projeto: Modelagem de Dados com Postgres

Introdução

Uma startup chamada Sparkify quer analisar os dados que eles vêm coletando em músicas e atividades do usuário em seu novo aplicativo de streaming de música.

A equipe de análise está particularmente interessada em entender quais músicas os usuários estão ouvindo. Atualmente, eles não têm uma maneira fácil de consultar seus dados, que reside em um diretório de registros JSON sobre a atividade do usuário no aplicativo, bem como um diretório com metadados JSON nas músicas em seu aplicativo.

Eles gostariam que um engenheiro de dados criasse um banco de dados postgres com tabelas projetadas para otimizar consultas na análise de reprodução de músicas e trazê-lo no projeto. Sua função é criar um esquema de banco de dados e um pipeline ETL para esta análise. Você poderá testar seu banco de dados e pipeline ETL executando consultas dadas a você pela equipe de análise da Sparkify e comparar seus resultados com os resultados esperados.

Descrição do projeto

Para concluir o projeto, você precisará definir tabelas de fatos e dimensões para um esquema estelar para um determinado foco analítico e escrever um pipeline ETL que transfere dados de arquivos em dois diretórios locais para essas tabelas em Postgres usando Python e SQL.

Conjunto de dados de música

O primeiro conjunto de dados é um subconjunto de dados reais do Million Song Dataset. Cada arquivo está no formato JSON e contém metadados sobre uma música e o artista dessa canção. Os arquivos são particionados pelas três primeiras letras do ID da faixa de cada canção. Por exemplo, aqui estão os filepaths para dois arquivos neste conjunto de dados.

```
song_data/A/B/C/TRABCEI128F424C983.json
song_data/A/A/B/TRAABJL12903CDCF1A.json
```

E abaixo está um exemplo de como é um único arquivo de música, TRAABJL12903CDCF1A.json.

```
{"num_songs": 1, "artist_id": "ARJIE2Y1187B994AB7", "artist_latitude": null,
"artist_longitude": null, "artist_location": "", "artist_name": "Line Renaud",
"song_id": "SOUPIRU12A6D4FA1E1", "title": "Der Kleine Dompfaff", "duration": 152.92036,
"year": 0}```
```

```
# Log Dataset
The second dataset consists of log files in JSON format generated by this [event
simulator](https://github.com/Interana/eventsim) based on the songs in the dataset
above. These simulate activity logs from a music streaming app based on specified
configurations.

The log files in the dataset you'll be working with are partitioned by year and month.
For example, here are filepaths to two files in this dataset.
```

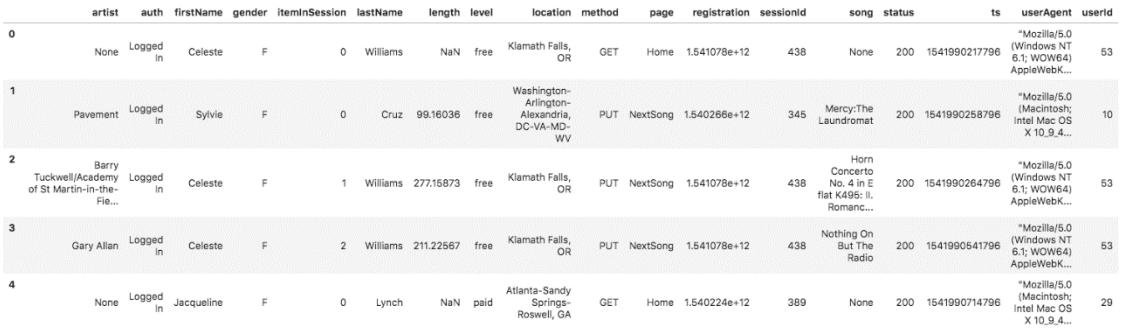
txt

log_data/2018/11/2018-11-12-events.json

log_data/2018/11/2018-11-13-events.json

'''

E abaixo está um exemplo do que os dados em um arquivo de registro, 2018-11-12-eventos.j, parece.



	artist	auth	firstName	gender	itemInSession	lastName	length	level	location	method	page	registration	sessionId	song	status	ts	userAgent	userId
0	None	Logged In	Celeste	F	0	Williams	NaN	free	Klamath Falls, OR	GET	Home	1.541078e+12	438	None	200	1541990217796	"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebK...	53
1	Pavement	Logged In	Sylvie	F	0	Cruz	99.16036	free	Washington-Arlington-Alexandria, DC-VA-MD-WV	PUT	NextSong	1.540266e+12	345	Mercy:The Laundromat	200	1541990258796	"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_4...	10
2	Barry Tuckwell/Academy of St Martin-in-the-Fie...	Logged In	Celeste	F	1	Williams	277.15873	free	Klamath Falls, OR	PUT	NextSong	1.541078e+12	438	Horn Concerto No. 4 in E flat K495: II. Romanc...	200	1541990264796	"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebK...	53
3	Gary Allan	Logged In	Celeste	F	2	Williams	211.22567	free	Klamath Falls, OR	PUT	NextSong	1.541078e+12	438	Nothing On But The Radio	200	1541990541796	"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebK...	53
4	None	Logged In	Jacqueline	F	0	Lynch	NaN	paid	Atlanta-Sandy Springs-Roswell, GA	GET	Home	1.540224e+12	389	None	200	1541990714796	"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_4...	29

Se você quiser olhar para os dados JSON dentro de log_data arquivos, você precisará criar um dataframe pandas para ler os dados. Lembre-se de primeiro importar bibliotecas JSON e pandas.

```
df = pd.read_json(filepath, lines=True)
```

Por exemplo, leria o arquivo de dados 2018-11-01-events.json. `df =`

```
pd.read_json('data/log_data/2018/11/2018-11-01-events.json', lines=True)
```

Caso você precise de uma atualização nos formatos de arquivo JSON, aqui está um vídeo útil.

Esquema para análise de reprodução de música

Usando os conjuntos de dados de música e log, você precisará criar um esquema estelar otimizado para consultas na análise de reprodução de músicas. Isso inclui as seguintes tabelas.

Tabela de Fatos

1. **songplays** - registros em dados de registro associados a reproduções de músicas, ou seja, registros com página NextSong
    - *songplay_id, start_time, user_id, nível, song_id, artist_id, session_id, localização, user_agent*

Tabelas de dimensão

2. **usuários** - usuários no aplicativo
    - *user_id, first_name, last_name, gênero, nível*
3. **músicas** - músicas no banco de dados de música
    - *song_id, título, artist_id, ano, duração*
4. **artistas** - artistas em banco de dados de música
    - *artist_id, nome, localização, latitude, longitude*
5. **tempo** - timestamps de discos em **songplays** divididos em unidades específicas
    - *start_time, hora, dia, semana, mês, ano, dia da semana*

Modelo de projeto

O projeto inclui seis arquivos:

1. `test.ipynb` exibe as primeiras linhas de cada tabela para que você verifique seu banco de dados.
2. `create_tables.py` drop e crie suas tabelas. Você executa este arquivo para redefinir suas tabelas antes de cada vez que você executar seus scripts ETL.
3. `etl.ipynb` lê e processa um único arquivo e carrega os dados em suas tabelas. Este notebook contém instruções detalhadas sobre o processo ETL para cada uma das tabelas.`song_datalog_data`
4. `etl.py` lê e processa arquivos e os carrega em suas tabelas. Você pode preencher isso com base no seu trabalho no notebook ETL.`song_datalog_data`
5. `sql_queries.py` contém todas as suas consultas sql, e é importado para os últimos três arquivos acima.
6. `README.md` fornece discussão sobre o seu projeto.

Etapas do Projeto

Abaixo estão as etapas que você pode seguir para concluir o projeto:

Criar tabelas

1. Escreva declarações para criar cada tabela.`CREATEsql_queries.py`
2. Escreva declarações para derrubar cada tabela se ela existir. `DROP sql_queries.py`
3. Criar seu banco de dados e tabelas.`create_tables.py`

4. Para confirmar a criação de suas tabelas com as colunas corretas. Certifique-se de clicar em "Reiniciar o kernel" para fechar a conexão ao banco de dados depois de executar este notebook. `test.ipynb`

Construir processos ETL

Siga as instruções no notebook para desenvolver processos de ETL para cada tabela. No final de cada seção de tabela, ou no final do notebook, rode para confirmar que os registros foram inseridos com sucesso em cada tabela. Lembre-se de repetir para redefinir suas tabelas antes de cada vez que você executar este notebook. `etl.ipynb` `;etl.py; create_tables.py; etl.py; test.ipynb.`