Desafio Técnico - Analytics Engineer

Descrição

Este desafio é composto por dois exercícios , espera-se que você finalize ambos em até 24 horas a partir do momento que compartilhamos com você. Caso precise de mais tempo basta nos avisar.

Abaixo estão os desafios, dentro de cada um você encontrará a descrição e a forma que é esperado que você nos retorne.

Exercício 1 - SQL

Para este exercício, não é necessário executar nenhum código. O objetivo é escrever um SQL que considere ser a forma mais adequada de responder as perguntas abaixo, com base nas tabelas a seguir.

A tabela de compra possui toda a informação das transações efetuadas na plataforma. Abaixo estão algumas colunas da tabela.

purchase (compra corrente)

purchase_id	buyer_id	prod_item_id	order_date	release_date	producer_id	purchase_partition	prod_item_partition
55	15947	5	2022-12-01	2022-12-01	852852	5	5
56	369798	746520	2022-12-25	2022-12-25	963963	6	0
57	147	98736	2021-07-03	2021-07-03	963963	7	6
58	986533	6565	2021-10-12	NULL	200478	8	5

product_item (item_produto corrente)

prod_item_id	product_id	item_quantity	purchase_value	prod_item_partition
1	69	5	500,00	5
5	69	120	1,00	0
98736	37	69	25,00	6
3	96	369	140,00	5

A tabela de item_produto possui os dados de quantidade de itens na transação e o valor dos itens da compra.

Como você responderia às perguntas abaixo:

- Quais são os 50 maiores produtores em faturamento ($) de 2021?
- Quais são os 2 produtos que mais faturaram ($) de cada produtor?

Caso tenha dúvidas sobre os campos, consulte o diagrama disponível neste link

Entregável: arquivo sql

Exercício 2 - Modelagem e desenvolvimento

Gross Merchandising Value, ou (GMV), é o valor transacionado considerando apenas as transações cujo pagamento foi efetuado e não foi cancelado. **Nesse exercício você tem como objetivo entregar o GMV diário por subsidiária** e para isso precisará "construir" um ETL baseado nos eventos de purchase, product_item e purchase_extra_info.

purchase (eventos/cdc)

purchase (eventos)

transaction_datetime	transaction_date	purchase_id	buyer_id	prod_item_id	order_date	release_date	producer_id
2023-01-20 22:00:00	2023-01-20	55	15947	5	2023-01-20	2023-01-20	852852
2023-01-26 00:01:00	2023-01-26	56	369798	746520	2023-01-25	NULL	963963
2023-02-05 10:00:00	2023-02-05	55	160001	5	2023-01-20	2023-01-20	852852
2023-02-26 03:00:00	2023-02-26	69	160001	18	2023-02-26	2023-02-28	96967
2023-07-15 09:00:00	2023-07-15	55	160001	5	2023-01-20	2023-03-01	852852

product_item (eventos/cdc)

product_item (eventos)

transaction_datetime	transaction_date	purchase_id	product_id	item_quantity	purchase_value
2023-01-20 22:02:00	2023-01-20	55	696969	10	50,00
2023-01-25 23:59:59	2023-01-25	56	808080	120	2400,00
2023-02-26 03:00:00	2023-02-26	69	373737	2	2000,00
2023-07-12 09:00:00	2023-07-12	55	696969	10	55,00

Purchase_extra_info (eventos/cdc)

transaction_datetime	transaction_date	purchase_id	subsidiary
2023-01-23 00:05:00	2023-01-23	55	nacional
2023-01-25 23:59:59	2023-01-25	56	internacional
2023-02-28 01:10:00	2023-02-28	69	nacional
2023-03-12 07:00:00	2023-03-12	69	internacional

Caso tenha dúvidas sobre os campos, consulte o diagrama disponível neste link

O GMV, tecnicamente, é a soma do valor das transações, ou seja, apenas transações com "Data Liberação" preenchida, indicando que o pagamento foi efetuado.

A atualização dos dados pode ocorrer de maneira assíncrona e, em caso de falhas no envio de dados para o lake, o reenvio de eventos pode acontecer - tanto para corrigir algo hoje, quanto para mudar o passado. Porém, sempre haverá um registro equivalente para cada compra transacionada (ex: uma compra sempre terá um item

de compra e uma informação extra, porém, podem não chegar no mesmo dia e hora).

Como seria a modelagem histórica e imutável de uma tabela final com o GMV acumulado do dia e separado por subsidiária? A modelagem precisa ajudar pessoas que não possuem conhecimentos sólidos em SQL.

Pré-Requisitos

- A interpretação do dado das tabelas de eventos acima descritas, faz parte da solução que precisa ser desenvolvida.
- Os dados nas tabelas acima refletem um cenário real, onde podem ocorrer inconsistências, como dados faltantes. Portanto, esperamos que a sua solução seja montada em cima dos exemplos acima fornecidos.
- Todas as tabelas são gatilhos para atualização da tabela final.
- Se uma tabela sofreu atualização e as demais não, os dados ativos das demais, deverão ser repetidos.
- A atualização ocorre em D-1.
- A modelagem precisa garantir que o passado não seja alterado, mesmo com o reprocessamento full da tabela.
- É necessário que o usuários consiga navegar entre o valor de Jan/2023 em 31/03/2023 e o valor de Jan/2023 no dia de hoje e os valores retornados pela consulta não podem ser diferentes.
- É necessário ter a rastreabilidade na granularidade diária.
- A partição da tabela pode ser o transaction_date.
- É necessário recuperar facilmente quais são os registros correntes da base histórica.
- Fica ao seu critério qual linguagem de programação utilizar: ***Preferencialmente Python, Spark ou Scala.***
- Escreva um select em cima da sua tabela para trazer a resposta para o GMV com os dados ativos/correntes de hoje.

O entregável:

- Script do ETL - (preferencialmente em python, spark ou scala);
- Create table do dataset final - (DDL);
- Exemplo do dataset final populado;
- Consulta SQL, em cima do dataset final que retorne o GMV diário por subsidiária;
- Descrição sobre a tech stack que viabiliza a solução;

Pontos fortes de avaliação

- Tratamento da qualidade dos dados
- Modelagem que atenda os requisitos