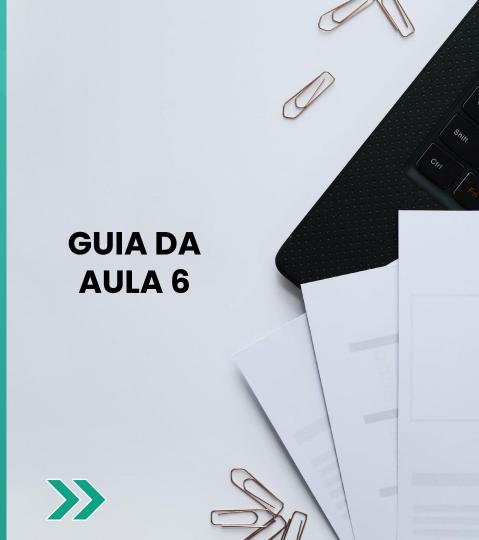


## Profissão: Analista de dados





4° PROJETO: PIPELINE DE DADOS DO TELEGRAM







# Faça a apresentação

- Introdução
- AWS Athena

Analytics



Acompanhe aqui os temas que serão tratados na videoaula







### 1. Introdução

A etapa de apresentação é responsável por entregar o dado para os usuários (analistas, cientistas, etc.) e sistemas (dashboards, motores de consultas, etc.), idealmente através de uma interface de fácil uso, como o SQL, logo, essa é a única etapa que a maioria dos usuários terá acesso. Além disso, é importante que as ferramentas da etapa entreguem dados armazenados em camadas refinadas, pois assim as consultas são mais baratas e os dados mais consistentes.





#### 2. AWS Athena

Na etapa de apresentação, o AWS Athena tem função de entregar o dados através de uma interface SQL para os usuários do sistema analítico. Para criar a interface, basta criar uma tabela externa sobre o dado armazenado na camada mais refinada da arquitetura, a camada enriquecida.





```
CREATE EXTERNAL TABLE `telegram`(
  `message id` bigint,
   `user id` bigint,
  `user is bot` boolean,
  `user first name ` string,
  `chat id` bigint,
  `chat type` string,
  `text` string,
  `date` bigint) PARTITIONED BY (
   `context date ` date)
ROW FORMAT SERDE
  'org.apache.hadoop.hive.gl.io.parquet.serde.ParquetHiveSerDe' STORED AS INPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetInputFormat' OUTPUTFORMAT
  'org.apache.hadoop.hive.ql.io.parquet.MapredParquetOutputFormat'
LOCATION
  's3://<bucket-enriquecido>/'
```





Por fim, adicione as partições disponíveis.

Importante: Toda vez que uma nova partição é adicionada ao repositório de dados, é necessário informar o AWS Athena para que a ela esteja disponível via SQL. Para isso, use o comando SQL MSCK REPAIR TABLE <nome-tabela>para todas as partições (mais caro) ou ALTER TABLE <nome-tabela> ADD PARTITION <coluna-partição> = <valor- partição> para uma única partição (mais barato), link documentação no https://docs.aws.amazon.com/athena/latest/ug/alter-table-add-parti tion.html).





```
MSCK REPAIR TABLE `telegram`;
```

#### E consulte as 10 primeiras linhas para observar o resultado.

```
SELECT * FROM `telegram` LIMIT 10;
```





### 3. Analytics

Com o dado disponível, usuários podem executar as mais variadas consultas analíticas. Seguem alguns exemplos:

Quantidade de mensagens por dia.

```
SELECT
   context_date,
   count(1) AS "message_amount"
FROM "telegram"
GROUP BY context_date
ORDER BY context_date DESC
```





#### • Quantidade de mensagens por usuário por dia.

```
user_id, user_first_name, context_date,
  count(1) AS "message_amount"

FROM "telegram"

GROUP BY
  user_id, user_first_name, context_date

ORDER BY context_date DESC
```





• Média do tamanho das mensagens por usuário por dia.

```
SELECT
    user_id, user_first_name, context_date,
    CAST(AVG(length(text)) AS INT) AS "average_message_length"
FROM "telegram"
GROUP BY
    user_id, user_first_name, context_date
ORDER BY context_date DESC
```





 Quantidade de mensagens por hora por dia da semana por número da semana.

```
SWITH
parsed date cte AS (
     SELECT
          CAST(date format(from unixtime("date"),'%Y-%m-%d %H:%i:%s')
AS timestamp) AS parsed date
     FROM "telegram"
),
hour week cte AS (
     SELECT
          EXTRACT (hour FROM parsed date) AS parsed date hour, EXTRACT (dow FROM parsed date) AS
          parsed date weekday, EXTRACT (week FROM parsed date) AS parsed date weeknum
     FROM parsed date cte
```





#### Continuação:

```
SELECT
    parsed_date_hour, parsed_date_weekday, parsed_date_weeknum,
    count(1) AS "message_amount"

FROM hour_week_cte

GROUP BY
    parsed_date_hour, parsed_date_weekday,
    parsed_date_weeknum

ORDER BY
    parsed_date_weeknum, parsed_date_weekday
```

