

Ingestão de Dados com Python

Erika Kinjo



Mini Bio



- Mestranda em Informática e Gestão do Conhecimento pela Uninove;
- Especialização em Projeto de Sistemas pela FATEC;
- Graduação em Análise e Projeto de Sistemas pelo IFSP (Instituto Federal de São Paulo)



- Especialista em People Analytics na Centauro;
- + 8 anos na área de TI, no 90 % voltada a dados.



- Amo cultura japonesa ^^
- Amo viajar

Agenda

- ❖ O que é Ingestão de Dados?
- ❖ Ingestão de Dados X ETL
- ❖ Data Warehouse X Data Lake
- ❖ Pipeline
- ❖ Spark
- ❖ PySpark
- ❖ Case
- ❖ Considerações
- ❖ Por onde começar??

O que é ingestão de dados?

O processamento de ingestão de dados refere-se ao processo de obtenção e importação de dados para uso imediato ou armazenamento em um banco de dados.

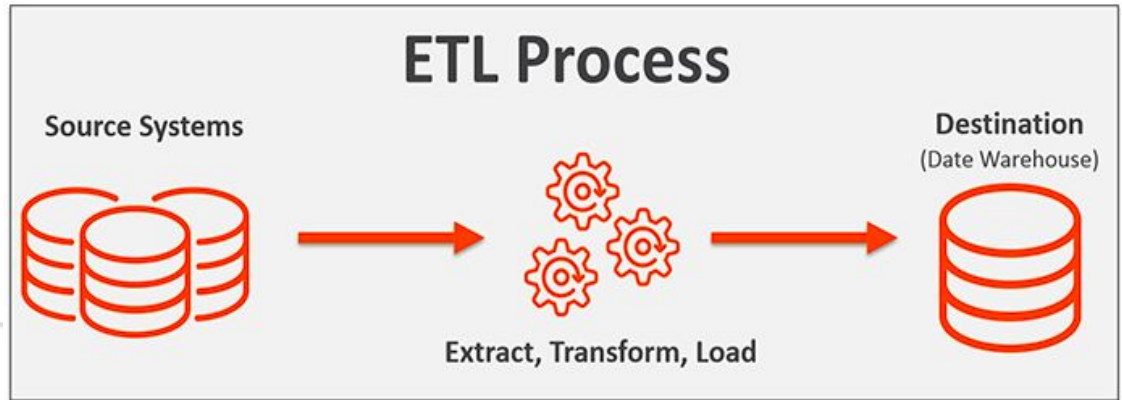
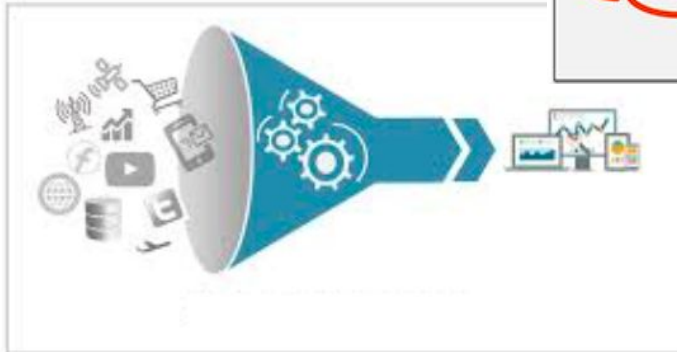
Transmissão pode ser em:

- **Streaming** (tempo real): os dados são importados conforme são gerado na origem
- **Batch** (lotes): os dados são importados em partes distintas em intervalos de tempo periódicos

Ingestão de dados X ETL

Ingestão:

- Dados não estruturados
- Streaming
- Novas origens



ETL:

- Dados estruturados
- Armazenamento fixo

Data Warehouse (DW) x Data Lake

Característica	DW	Data Lake
Tipo de Dados	Estruturado	Estruturado, Semi-Estruturado e Não Estruturado
Agilidade	Configuração Fixa	Configuração mais flexível
Usuário	Áreas de negócio	Data Science, Áreas de negócio, usuário final

Fonte: DBABrasil

O pipeline

- Planejamento da ingestão dos dados pré-existentes para o Data Lake.
- No processo de transição de uma solução para o Data Lake o pipeline de ingestão de dados é fundamental.

O pipeline

Define como a lógica de negócios será comunicada. Contém as seguintes informações:

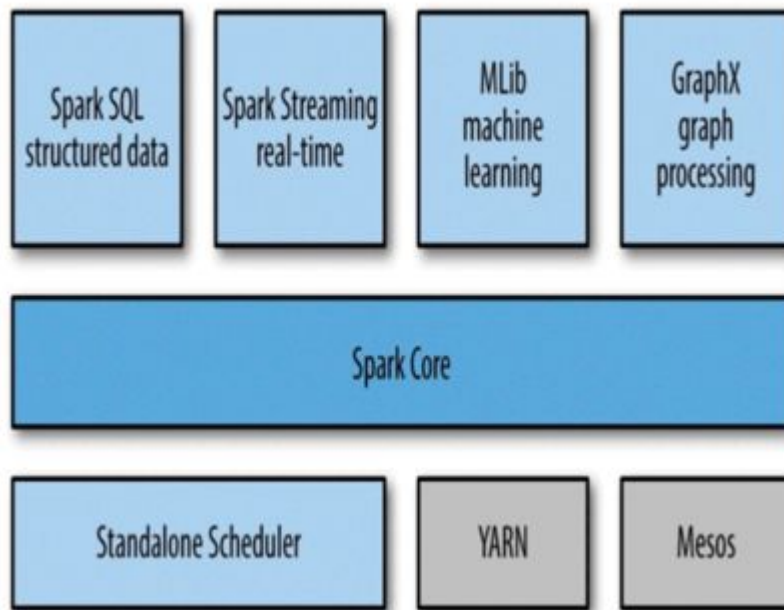
- Nomes, locais e formatos das origens dos dados
- Atividades de transformação dos dados **
- A programação dessas atividades
- Recursos que executam essas atividades e precondições
- Precondições que precisam ser atendidas antes que as atividades sejam programadas
- Maneiras de alertar os responsáveis com atualizações de status à medida que a execução do pipeline prossegue

Spark




- O Apache Spark é uma ferramenta Big Data que tem o objetivo de processar grandes conjuntos de dados de forma paralela e distribuída.
- Permite a programação em três linguagens: Java, Scala e Python.
- O Spark pode ser configurado para rodar em um cluster Hadoop em conjunto com um gerenciador, como YARN ou standalone
- Além da parte Core do Spark, que é o responsável pelo gerenciamento dos processamentos e das bases de dados. E possui alguns componentes:
 - Spark Streaming: que permite ao desenvolvedor manipular e processar dados oriundos de algum streaming.
 - Spark MLlib: que é uma biblioteca de algoritmos de machine learning distribuídos.
 - Spark GraphX: que é uma biblioteca capaz de processar grafos utilizando a arquitetura distribuída do Spark.

Spark



Python + Spark = PySpark

PySpark - Vantagens

- Processamento distribuído de grandes volumes de dados.
- Capacidade de extrair dados diretamente do HDFS, a partir de uma instância Spark configurada na máquina do Data Scientist ou Data Engineer, sem necessidade de download ou cópia local dos dados;
- Implementação em Python  Facilidade de integrar os modelos matemáticos desenvolvido pelo time de Data Scientists.
- Framework bastante completo, pois permite tratar processamento de dados em batch, streaming, além de fornecer suporte a algoritmos distribuídos de Machine Learning e grafos.

HDFS (Hadoop Distributed File System)

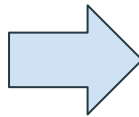
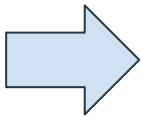
Um projeto da Apache Software Foundation e um subprojeto do projeto Apache Hadoop, trata-se de um sistema de arquivos distribuído do Hadoop.

Entre os objetivos destacam-se:

- Escalabilidade para armazenar e processar de modo confiável grandes quantidades de dados
- Eficiência pela distribuição de dados e pela lógica para processá-los em paralelo nos nós em que os dados estão localizados



PySpark



```
from pyspark.sql import HiveContext
from pyspark import SparkContext
from pyspark.sql import SQLContext
from pyspark.sql.functions import *
from pyspark.sql.types import *
import os
import logging

sc = SparkContext()
sqlContext = HiveContext(sc)

tableHive = 'default.teste'
cmd = 'select * from {} limit 0'.format(tableHive)

query = sqlContext.sql(cmd)
schema = query.schema

df = sqlContext.read.format(com.databricks.spark.csv) \
    .options(header='true', delimiter=',') \
    .load('/tmp/teste.csv', schema=schema)

df.write.mode("append") \
    .format("orc") \
    .option("compression", "zlib") \
    .insertInto(tableHive)
```

Case

- Medir o quanto um cliente utiliza o serviço



Case

```
from pyspark import SparkContext
from pyspark.sql import SQLContext
from pyspark.sql.functions import *
from pyspark.sql.types import *
```

```
sc = SparkContext()
```

```
df = sc.read.csv("data.csv")
```

```
df_novo = df\
    .groupBy("campo1", "campo2")\
    .agg(sum("valor1").alias("soma_valor1"))
```

```
df_novo.show()
```


PySpark - Pontos de Atenção

- Configurações do Spark: para cada Job uma configuração específica
 - Sobrecarga
 - Alocar recursos desnecessários

```
spark.app.name          PYLADIES_APP
spark.driver.cores      1
spark.driver.memory     1G
spark.executor.memory   1G
```



.properties

Considerações

- Só existe o Spark para fazer ingestão de dados no universo Big Data?
 - Não. Tem o Sqoop também!!
- Eu preciso aprender Python para começar?
 - Claro que não, se souber SQL já dá usar o Spark

E por onde começar????



Meu começo...

- Instalando o Anaconda e instalando os pacotes....
- Criando arquivos .csv e transformando em .parquet



Dicas de Cursos

- **Data Science Academy**

<https://www.datascienceacademy.com.br/pages/home>

- **Udemy**

<https://www.udemy.com/big-data-com-apache-spark-e-pyspark/>

- **Coursera**

<https://www.coursera.org/learn/scala-spark-big-data>

Dicas de Cursos

- **FIAP**

Envisioning Big Data

Machine Learning e Data Mining

- **FIA**

Análise de Big Data com Spark e Python

DÚVIDAS ??

Contatos



Erika Kinjo



@midorikinjo

Feedback

