

Framework Luigi para ETL com Python

The word "Luigi" is written in a stylized, green, blocky font that resembles pipes or tubes. The letters are interconnected, with the 'L' having a vertical pipe on its left and the 'u' having a horizontal pipe on its top.

Prof. Dr. Diego Bruno

Education Tech Lead na DIO

Doutor em Robótica e *Machine Learning* pelo ICMC-USP

Framework de ETLs

Prof. Dr. Diego Bruno

Machine Learning



Framework

Luigi é um framework de execução criado pelo Spotify que cria pipelines de dados em Python. É um pacote Python (2.7, 3.6, 3.7 testado) que ajuda a construir pipelines complexos de trabalhos em lote. Ele lida com resolução de dependências, gerenciamento de fluxo de trabalho, visualização, tratamento de falhas, integração de linha de comando e muito mais.



Framework

Luigi é um **framework** de execução criado pelo **Spotify** que cria pipelines de dados em Python. Em tese, é um pacote Python (2.7, 3.6, 3.7 testado) que ajuda a construir pipelines complexos de trabalhos em lote. Ele lida com resolução de dependências, gerenciamento de fluxo de trabalho, visualização, tratamento de falhas, integração de linha de comando e muito mais.



The word "Luigi" is written in a stylized, 3D font that looks like it's made of green pipes. The letters have a metallic texture and some green pipe fittings attached to them, particularly on the 'i' and 't'. The 'L' is a vertical pipe section, the 'u' is a horizontal U-shaped pipe, the 'i' is a T-junction pipe, and the 'g' is a horizontal pipe section.

Tópicos

Target: Em palavras simples, um alvo contém a saída de uma tarefa. Um destino pode ser um local (por exemplo: um arquivo), (MySQL etc);



Tópicos

Task: - Tarefa é algo onde o trabalho real ocorre. Uma tarefa pode ser independente ou dependente. O exemplo de uma tarefa dependente é despejar os dados em um arquivo ou banco de dados. Antes de carregar os dados, os dados devem estar lá por qualquer meio (*scraping*, API, etc). Cada tarefa é representada como uma classe Python que contém certas funções-membro obrigatórias. Uma função de tarefa contém os seguintes métodos:



Tópicos

Task:

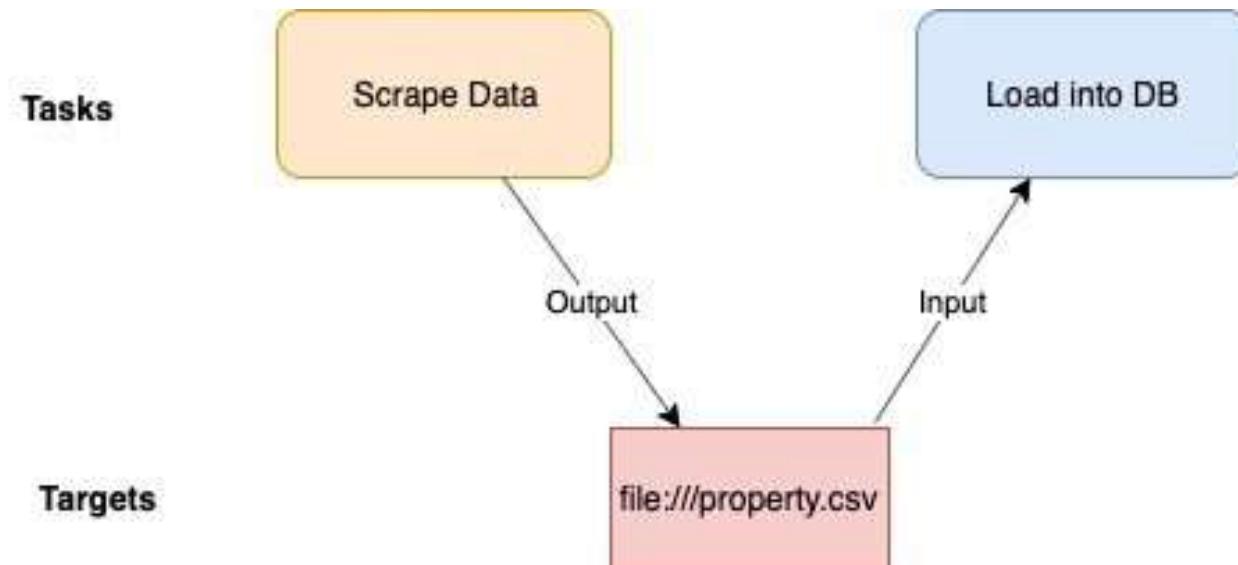
require():- Esta função membro da classe *task* contém todas as instâncias de tarefas que devem ser executadas antes da tarefa atual.

output():- Este método contém o destino onde a saída da tarefa será armazenada. Isso pode conter um ou mais objetos de destino.

run():- Este método contém a lógica real para executar uma tarefa.

Tópicos

A representação do processo será algo como abaixo:



TL-Python-Luigi) C:\Users\diego>luigid
luigid' não é reconhecido como um comando interno
externo, um programa operável ou um arquivo em lotes.

TL-Python-Luigi) C:\Users\diego>luigid
22-07-21 16:23:35,144 luigi[13052] INFO: logging configured by default settings
22-07-21 16:23:35,146 luigi.scheduler[13052] INFO: No prior state file exists at /var/lib/luigi-server/state.pickle
Starting with empty state
22-07-21 16:23:35,156 luigi.server[13052] INFO: Scheduler starting up
22-07-21 16:24:34,950 tornado.access[13052] INFO: 302 GET / (::1) 1.00ms
22-07-21 16:24:35,086 tornado.access[13052] INFO: 200 GET /static/visualiser/index.html (::1) 132.56ms
22-07-21 16:24:35,126 tornado.access[13052] INFO: 200 GET /static/visualiser/css/luigi.css (::1) 9.54ms
22-07-21 16:24:35,143 tornado.access[13052] INFO: 200 GET /static/visualiser/lib/jquery-1.10.0.min.js (::1) 14.00ms

22-07-21 16:24:35,150 tornado.access[13052] INFO: 200 GET /static/visualiser/lib/AdminLTE/css/skin-green-light.min.s (::1) 5.99ms
22-07-21 16:24:35,165 tornado.access[13052] INFO: 200 GET /static/visualiser/lib/bootstrap3/css/bootstrap.min.css (1) 13.06ms
22-07-21 16:24:35,171 tornado.access[13052] INFO: 200 GET /static/visualiser/css/font-awesome.min.css (::1) 18.70ms

22-07-21 16:24:35,175 tornado.access[13052] INFO: 200 GET /static/visualiser/css/tipsy.css (::1) 22.73ms
22-07-21 16:24:35,181 tornado.access[13052] INFO: 200 GET /static/visualiser/lib/bootstrap3/css/bootstrap-theme.min.ss (::1) 28.93ms
22-07-21 16:24:35,188 tornado.access[13052] INFO: 200 GET /static/visualiser/lib/AdminLTE/css/AdminLTE.min.css (::1) 34.25ms
22-07-21 16:24:35,195 tornado.access[13052] INFO: 200 GET /static/visualiser/lib/datatables/css/jquery.dataTables.m

Luigi Task Visualiser

localhost:8082/static/visualiser/index.html

Luigi Task Status

Task List Dependency Graph Workers Resources Running

TASK FAMILIES Clear selection

PENDING TASKS 0	RUNNING TASKS 0	BATCH RUNNING TASKS 0	DONE TASKS 0
FAILED TASKS 0	UPSTREAM FAILURE 0	DISABLED TASKS 0	UPSTREAM DISABLED 0

Show 10 entries Filter table: Filter on Server

Name	Details	Priority	Time	Actions
No data available in table				

Showing 0 to 0 of 0 entries Previous Next

→ Localhost:8082

30°C Ensolarado

16:36 21/07/2022

Framework

O framework Luigi tem suporte para trabalho de forma gráfica

The screenshot shows the Luigi Task Status interface. At the top, there's a navigation bar with tabs: Luigi Task Status, Task List, Dependency Graph, Workers, Resources, and a status indicator Running. Below the navigation bar is a section titled "TASK FAMILIES" with a "Clear selection" button. This section contains eight cards arranged in a 2x4 grid, each representing a task family with an icon, a title, and a count of zero tasks:

PENDING TASKS	RUNNING TASKS	BATCH RUNNING ...	DONE TASKS
0	0	0	0
FAILED TASKS	UPSTREAM FAILU...	DISABLED TASKS	UPSTREAM DISA...
0	0	0	0

At the bottom of the screen, there's a table with columns: Name, Details, Priority, Time, and Actions. The table displays the message "No data available in table". Below the table, it says "Showing 0 to 0 of 0 entries" and has navigation links for Previous and Next.

Obrigado!

Prof. Dr. Diego Bruno
Machine Learning



ETL: Manipulando dados com Pandas

Pandas

Prof. Dr. Diego Bruno

Education Tech Lead na DIO

Doutor em Robótica e *Machine Learning* pelo ICMC-USP



Leitura de dados

Prof. Dr. Diego Bruno

Pandas



Leitura dos dados

[]

```
import pandas as pd
url = 'https://raw.githubusercontent.com/Muralimekala/python/master/Resp2.csv'
df1 = pd.read_csv(url)
# Dataset is now stored in a Pandas Dataframe
```

[]

```
import pandas as pd
url = 'https://raw.githubusercontent.com/Muralimekala/python/master/Salaries.csv'
sf = pd.read_csv(url)
#salaries CSV
sf.head()
```

Tabela gerada

		EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes	Agency	Status
0	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0.00	400184.25	NaN	567595.43	567595.43	2011	NaN	San Francisco	NaN
1	2	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	NaN	538909.28	538909.28	2011	NaN	San Francisco	NaN
2	3	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.60	NaN	335279.91	335279.91	2011	NaN	San Francisco	NaN
3	4	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916.00	56120.71	198306.90	NaN	332343.61	332343.61	2011	NaN	San Francisco	NaN
4	5	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT,(FIRE DEPARTMENT)	134401.60	9737.00	182234.59	NaN	326373.19	326373.19	2011	NaN	San Francisco	NaN

Obrigado!

Prof. Dr. Diego Bruno



Biblioteca **Scikit-learn**



Prof. Dr. Diego Bruno

Education Tech Lead na DIO

Doutor em Robótica e *Machine Learning* pelo ICMC-USP



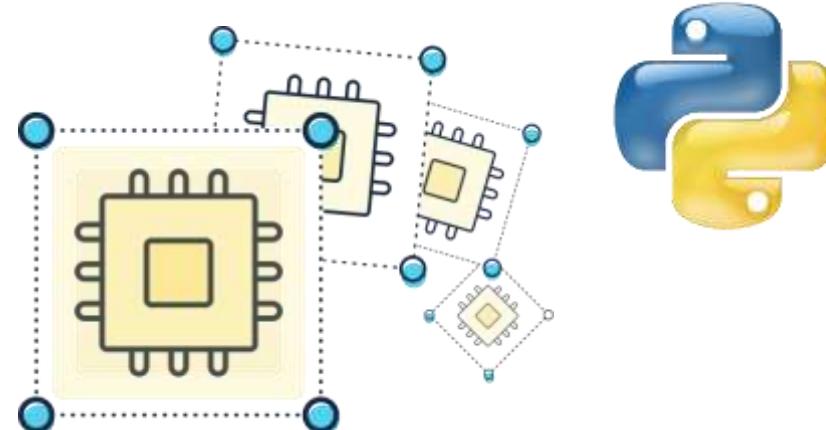
Biblioteca **Scikit-learn**

Prof. Dr. Diego Bruno

Machine Learning

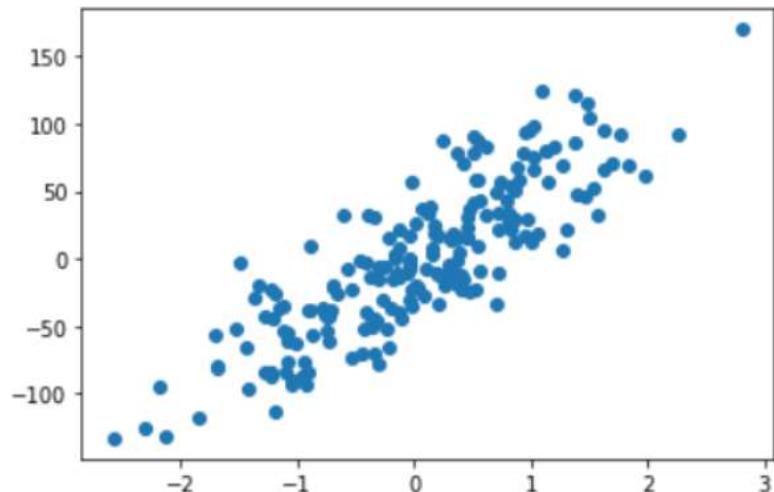
Utilizando a biblioteca

Esta biblioteca dispõe de ferramentas simples e eficientes para análise preditiva de dados, é reutilizável em diferentes situações, possui código aberto, sendo acessível a todos e foi construída sobre os pacotes **NumPy**, **SciPy** e **matplotlib**.



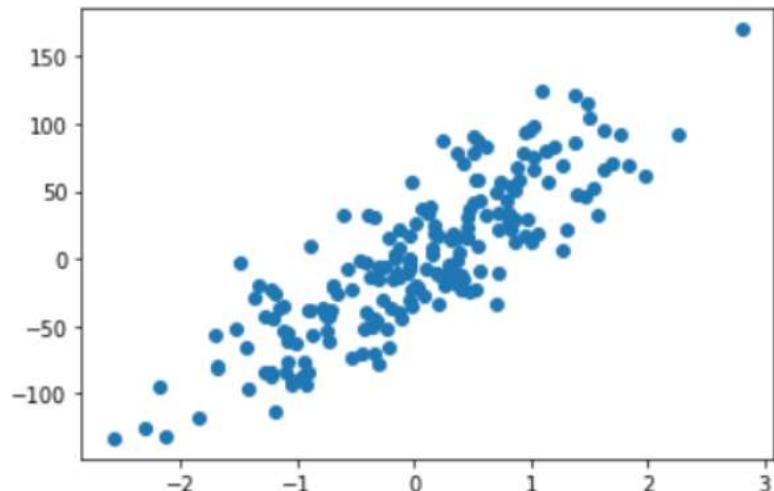
Scikit-learn

Neste exemplo iremos criar uma massa de dados com 200 observações, com apenas uma variável preditora, que será a variável x e a variável target, que será a y . Para isso indicamos os parâmetros $n_samples = 200$ e $n_features = 1$. O parâmetro $noise$ define o quanto dispersos os dados estarão um dos outros.



Scikit-learn – Etapa 1

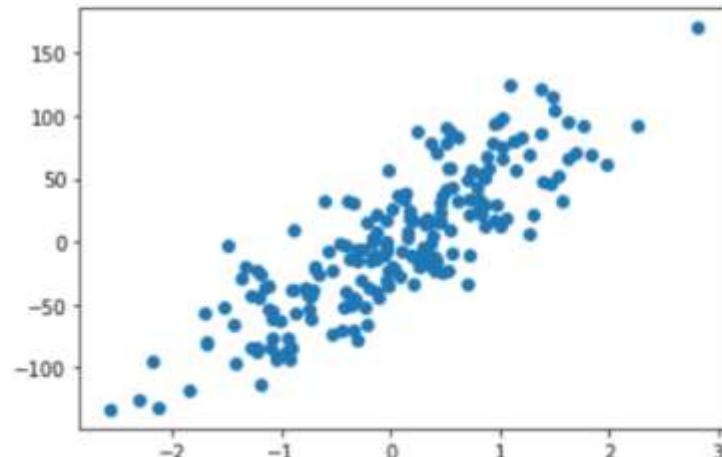
```
from sklearn.datasets import make_regression  
#gerando uma massa de dados:  
x, y = make_regression(n_samples=200, n_features=1, noise=30).
```



Scikit-learn

Utilizaremos o pacote *matplotlib*, com o módulo *pyplot* e a função *scatter()*, que criará o gráfico, e função *show()* que o exibirá na tela.

```
In [2]: import matplotlib.pyplot as plt  
  
# mostrando no gráfico:  
plt.scatter(x,y)  
plt.show()
```



Scikit-learn

Com os dados gerados, já podemos iniciar a criação de nosso modelo de machine learning. Para isso utilizaremos o módulo *linear_model*, e a função *LinearRegression()*.

```
from sklearn.linear_model import LinearRegression  
# Criação do modelo  
modelo = LinearRegression()
```

Scikit-learn

Com os dados gerados, já podemos iniciar a criação de nosso modelo de machine learning. Para isso utilizaremos o módulo *linear_model*, e a função *LinearRegression()*.

```
from sklearn.linear_model import LinearRegression  
# Criação do modelo  
modelo = LinearRegression()
```

Scikit-learn

Após esta execução, o objeto *modelo* que acabamos de criar está pronto para receber os dados que darão origem ao modelo. Como não indicamos nenhum parâmetro específico na função, estamos utilizando suas configurações padrão.

Agora precisamos apenas apresentar os dados ao modelo, e para isso temos o *método fit()*. Na documentação da função podemos conferir todos os métodos que ela possui.

Scikit-learn – Etapa 2

Após esta etapa, nosso modelo de *machine learning* está pronto e **podemos utilizá-lo para prever dados desconhecidos**. Simplificando este primeiro entendimento, vamos apenas visualizar a **reta de regressão linear** que o modelo gera, com os mesmos dados que criaram o modelo. Para isso iremos utilizar o método ***predict()***, indicando que queremos aplicar a previsão nos valores de **x**. O resultado do método será uma previsão de **y** para cada valor de **x** apresentado.

Scikit-learn – Etapa 2

```
modelo.predict(x)
```

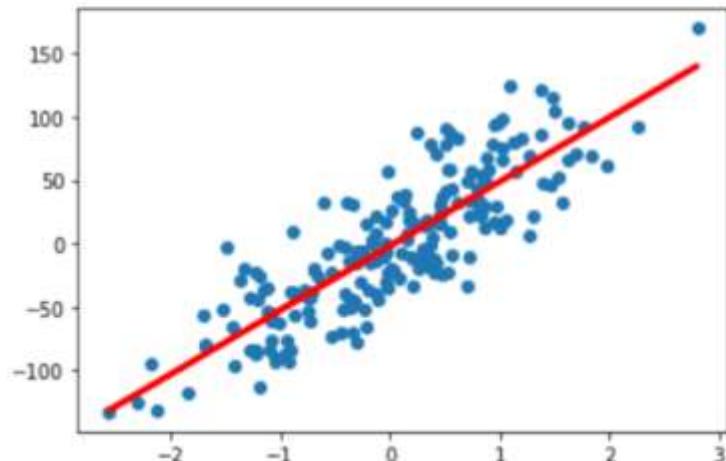
```
In [5]: modelo.predict(x)
```

```
Out[5]: array([ 35.6203308 , -3.17423057, -45.61500036, -35.55761336,
   -36.64951896, -0.26583513, -109.2048109 , -4.03112327,
   -74.775194 , -40.03149364, -63.76819387, -79.38003634,
   15.28045518,  76.15050169, -9.89632462,  16.9883385 ,
  -17.40250852,  45.81916612, -56.95678911, -3.81705028,
  -8.97268015,  6.0599397 , -38.26310679,  87.85259224,
  34.68692551,  19.82857138, -63.84119353, -2.66023192,
  38.05629487, -28.88635211,  8.68916995, -131.85857672,
  80.7606594 ,  29.64818836,  21.09928963, -8.59112398,
  84.28593443, -22.35286161, -39.30567938, -16.93842323,
  -1.18700474,  13.989157 , -17.13834937,  112.76375682,
  77.7204409 , -13.03318035, -19.1297018 , -8.05979909,
  8.41315509, -58.33337327,  42.41831639,  6.5197212 ,
  72.27310044,  7.52191056,  43.53561478,  56.02630839,
```

Scikit-learn – Etapa 2

A função `plot()` do pacote `pyplot` gera uma reta com os dados apresentados. Como já temos os dados de x e y , basta indicá-los na função. Assim, primeiramente montamos novamente o gráfico de x e y original com a função `scatter()`, e somamos a ele a reta de

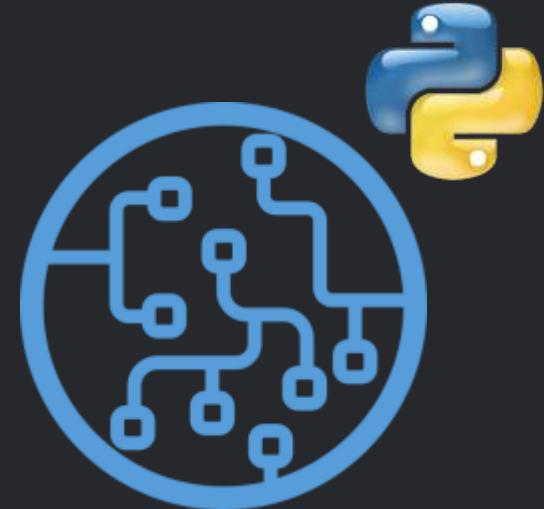
```
In [6]: plt.scatter(x,y)
plt.plot(x, modelo.predict(x), color='red', linewidth=3)
plt.show()
```



Obrigado!

Machine Learning

Prof. Dr. Diego Bruno



Biblioteca Pandas

Prof. Dr. Diego Bruno

Education Tech Lead na DIO

Doutor em Robótica e *Machine Learning* pelo ICMC-USP



Biblioteca Pandas

Prof. Dr. Diego Bruno

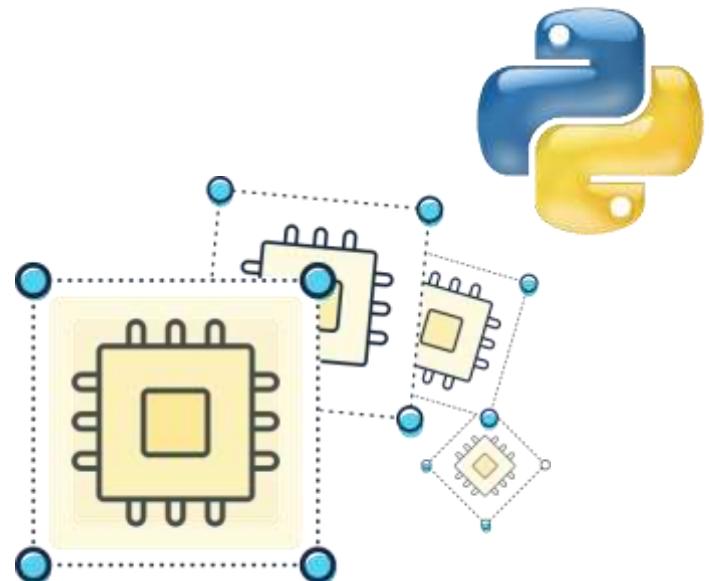
Machine Learning



Utilizando a biblioteca

O pandas permite trabalhar com diferentes tipos de dados, por exemplo:

- Dados tabulares, como uma planilha Excel ou uma tabela SQL;
- Dados ordenados de modo temporal ou não;
- Matrizes;
- Qualquer outro conjunto de dados,
que não necessariamente precisem estar rotulados;



Bibliotecas de exemplo:





Estruturas de Dados

- Os dois principais objetos da biblioteca Pandas são as **Series** e os **DataFrames**:
- → Uma Serie é uma matriz unidimensional que contém uma sequência de valores que apresentam uma indexação (que podem ser numéricos inteiros ou rótulos), muito parecida com uma única coluna no Excel.
- → Já o DataFrame é uma estrutura de dados tabular, semelhante a planilha de dados do Excel, em que tanto as linhas quanto as colunas apresentam rótulos.



Estruturas de Dados

- Os dois principais objetos da biblioteca Pandas são as **Series** e os **DataFrames**:

Series

Índice	nome	idade
0	Maria	25
1	José	56
2	Ana	31
3	Paulo	43

DataFrame

Colunas

	nomes	idade
0	Maria	25
1	José	56
2	Ana	31
3	Paulo	43



Quais as vantagens?

- Comandos equivalentes, com a mesma funcionalidade no Pandas;

R	pandas
dim(df)	df.shape
head(df)	df.head()
slice(df, 1:10)	df.iloc[:9]
filter(df, col1 == 1, col2 == 1)	df.query('col1 == 1 & col2 == 1')
df[df\$col1 == 1 & df\$col2 == 1,]	df[(df.col1 == 1) & (df.col2 == 1)]



DataFrame

- Agora se visualizarmos novamente os primeiros 4 dados do nosso conjunto, veremos que todos os valores passados para ***na_values***, além dos próprios dados ausentes, foram substituídos por NaN.

```
1 df.head(n=4)
```

	id	data_aq	produto	quantidade	valor UN	Total	setor
0	0	01/01/2019	toalha	6	R\$ 35,00	R\$ 210,00	Mesa_banho
1	1	02/01/2019	toalha	6	R\$ 35,00	R\$ 210,00	NaN
2	2	03/01/2019	toalha	2	R\$ 35,00	R\$ 70,00	mesa_banho
3	3	01/02/2019	toalha	5	R\$ 35,00	R\$ 175,00	NaN



dio._

df.shape

```
1 df.shape
2 (549, 7)
```



df.info

```
1 df.info()
```

Já para saber que formato se encontram os dados em cada coluna, além da quantidade de memória para ler esse conjunto de dados, podemos utilizar o comando *info*:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 549 entries, 0 to 548
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   id          549 non-null    int64  
 1   data_aq     549 non-null    object  
 2   produto     542 non-null    object  
 3   quantidade  549 non-null    int64  
 4   valor_UN   549 non-null    object  
 5   Total       549 non-null    object  
 6   setor       535 non-null    object  
dtypes: int64(2), object(5)
memory usage: 30.1+ KB
```



Alterações

Na sequência, podemos visualizar quais são nossas colunas existentes e até mesmo alterar esses nomes, basta passar o novo conjunto de nomes desejados com a mesma quantidade de colunas existente no conjunto original:

```
1 df.columns
2 Index(['id', 'data_aq', 'produto', 'quantidade', 'valor UN', 'Total', 'setor'], dtype='object')
3
4 df.columns = ['id', 'data_aq', 'produto', 'quantidade', valor_un', 'valor_total', setor']
```



Verificação

Para verificar quantos dados faltantes existem em nosso conjunto

```
1 df.isnull().sum()
```

```
id          0
data_aq     0
produto     7
quantidade  0
valor_un    0
valor_total 0
setor       14
dtype: int64
```



Valores únicos

No nosso objeto Serie podemos verificar quais os valores únicos existem naquela coluna, com o método *unique*:

```
1 df['setor'].unique()
```

```
array(['Mesa_banho', 'mesa_banho', 'mesa_Banho', 'perfumaria',
       'informatica', 'Informatica', 'informaticA', 'brinquedos',
       'brinquEDos', 'Brinquedos'], dtype=object)
```

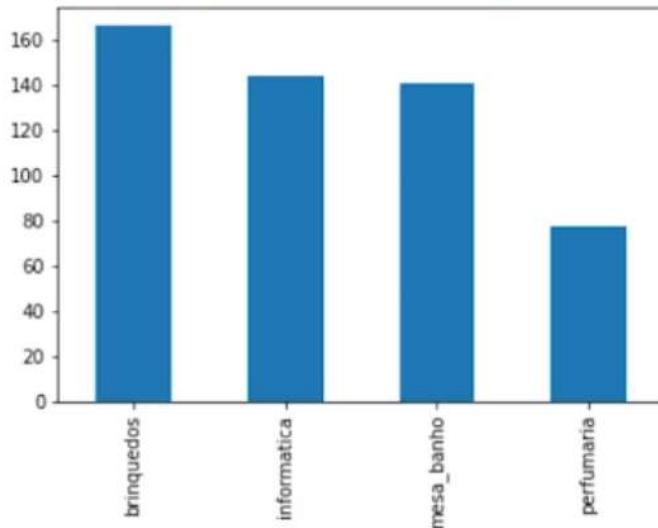


Agrupamentos

Ainda a partir desse método podemos gerar uma visualização simples e rápida com o resultado. Como? Com o método *plot*.

```
1 df['setor'].value_counts().plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f18fd27be90>
```





Dados estatísticos

o Pandas colabora na exibição de algumas informações estatísticas a respeito do nosso conjunto de dados e permite que possamos realizar facilmente uma análise com o nosso objeto DataFrame

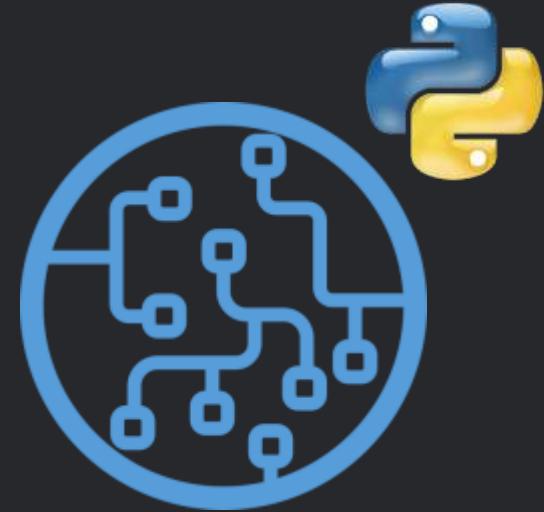
```
1 df.describe()
```

	id	quantidade	valor_total
count	528.000000	528.000000	528.000000
mean	275.229167	6.032197	280.739848
std	157.431587	3.207100	383.826301
min	0.000000	1.000000	2.990000
25%	137.750000	3.000000	64.950000
50%	276.000000	6.000000	142.890000

Obrigado!

Machine Learning

Prof. Dr. Diego Bruno



Ferramentas Aplicadas para ETL

Prof. Dr. Diego Bruno

Education Tech Lead na DIO

Doutor em Robótica e *Machine Learning* pelo ICMC-USP



Ferramentas de ETLs

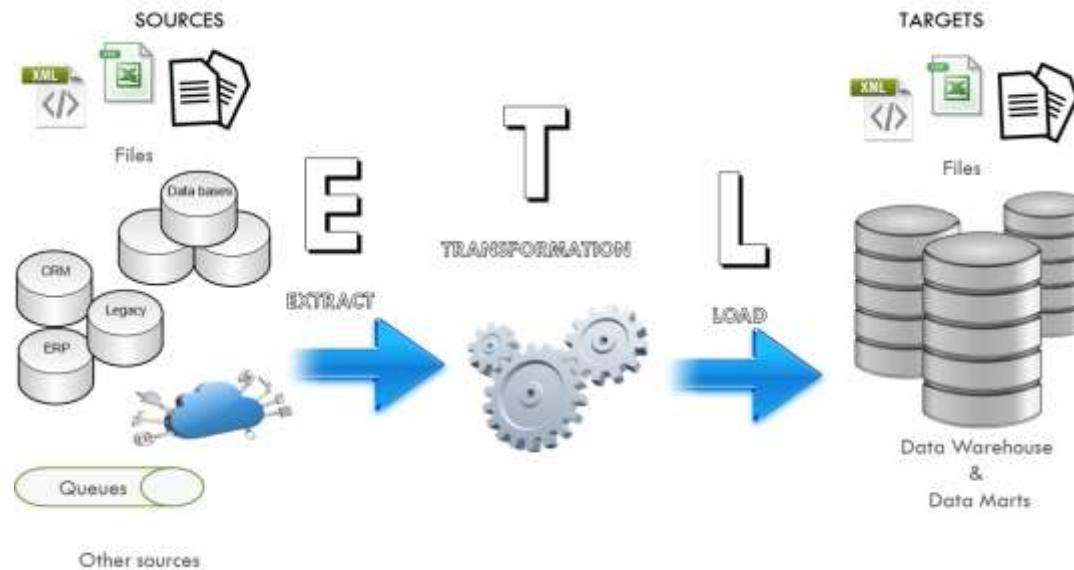
Prof. Dr. Diego Bruno

Machine Learning



Ferramentas Utilizadas

As ferramentas são softwares utilizados para facilitar o processo de integração de dados. Inicialmente muito usados em projetos de **Data Warehouse** e **Business Intelligence** em geral, ultimamente tem sido utilizados em processos de integração de software, bancos de dados, etc.



Ferramentas Utilizadas

Existem diversas ferramentas de ETL, como :

IBM Data Stage – <https://www.cetax.com.br/dastage-ibm-ferramenta-de-etl/> Informatica
Power Center – <https://www.cetax.com.br/power-center-informatica-ferramenta-de-etl/> SSIS
Sql Server Integration Services – <https://www.cetax.com.br/ssis-sql-server-integration->
Talend ETL – <https://www.cetax.com.br/criando-job-simples-no-talend/>



ETL para BIG Data

Hoje com o crescimento dos projetos de Big Data aumenta-se mais ainda a necessidade de fazer ETL entre plataformas heterogêneas, para isso, projetos como o *Hadoop*, possuem ferramentas próprias para carga de dados, como :

SQOOP – Ferramenta para movimentar dados dentre bancos de dados relacionais e o ambiente *Hadoop*.

HIVE – Ambiente de SQL sobre um cluster *Hadoop*.

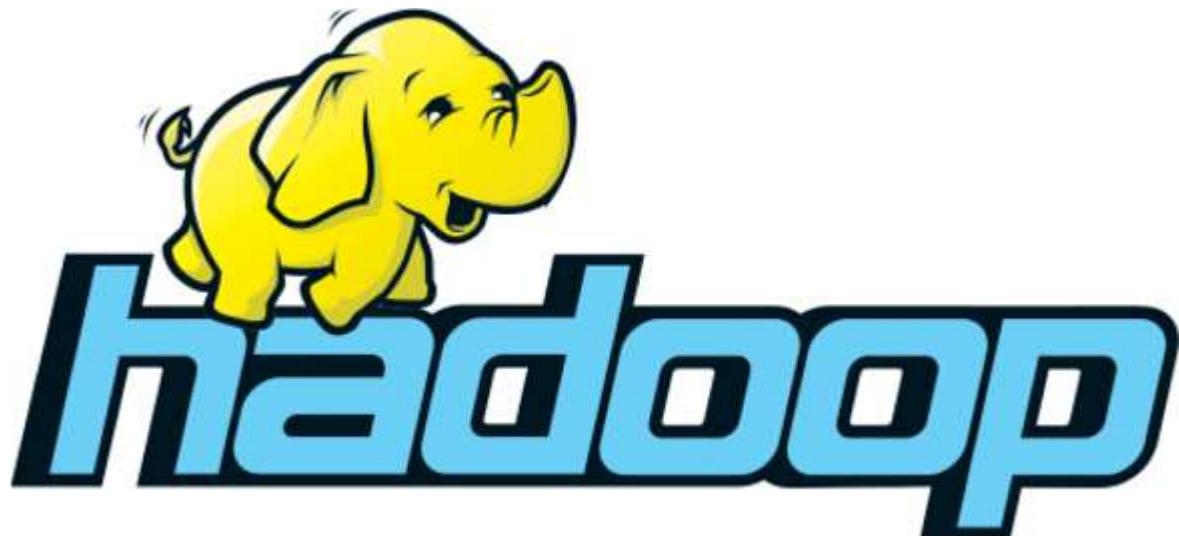
PIG – Ferramenta de Script para transformação e processamento de dados.

SPARK – Framework de processamento em memória.

,

Mas o que é Hadoop?

Hadoop é uma plataforma de software em Java de computação distribuída voltada para clusters e processamento de grandes volumes de dados, com atenção a tolerância a falhas.



ETL para BIG Data

Mesmo com todas as possibilidades acima, vemos as ferramentas de ETL se adaptando para **Big Data** ou gerando códigos para serem rodados nessas ferramentas do Ecosistema Hadoop.

Em Big Data, o processo de carga também é conhecido como Ingestão de Dados, que geralmente é a primeira parte da carga (Extract) a parte mais simples do processo, que consiste em extrair dados dos sistemas de origem e trazer para dentro do Data Lake ou ambiente de dados utilizado.

Obrigado!

Prof. Dr. Diego Bruno
Machine Learning



Etapas para um Processo ETL

Prof. Dr. Diego Bruno

Education Tech Lead na DIO

Doutor em Robótica e *Machine Learning* pelo ICMC-USP



Etapas de ETLs

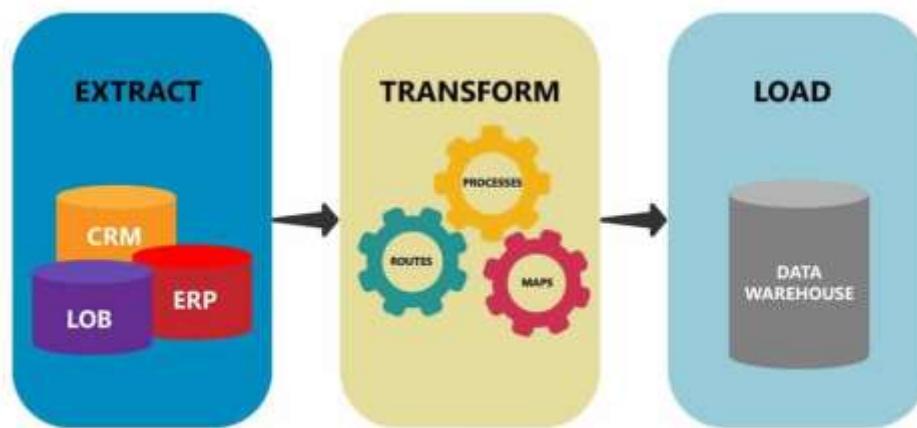
Prof. Dr. Diego Bruno

Machine Learning



Temos 3 Etapas:

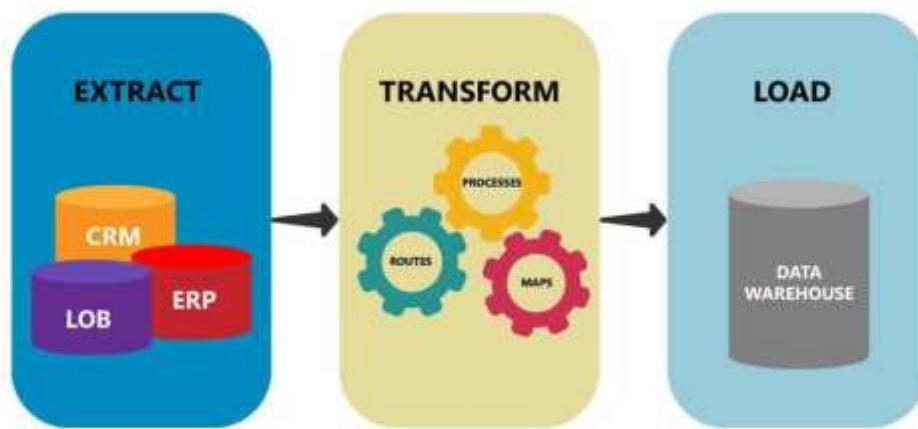
- 1) Extract
- 2) Transform
- 3) Load.



ETL - Extract, Transform, Load

1) Extract

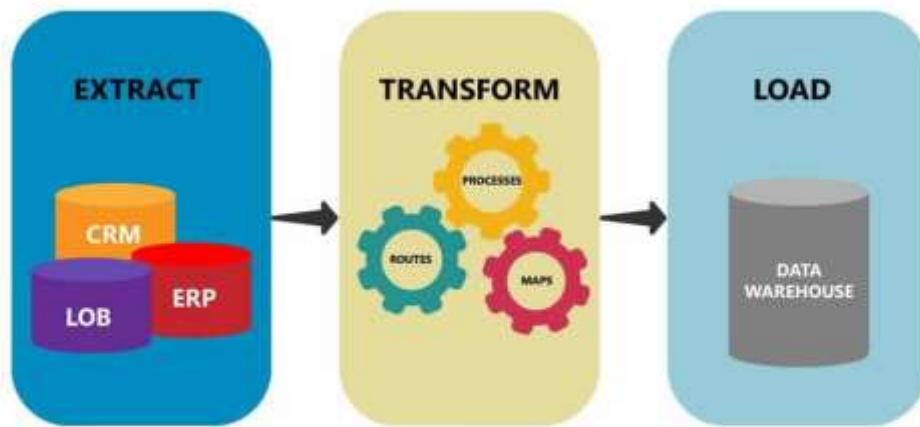
O processo de **Extração de dados** consiste em se comunicar com outros sistemas ou bancos de dados para capturar os dados que serão inseridos no destino, seja uma *Staging Area* ou outro sistema.



ETL - Extract, Transform, Load

2) Transform

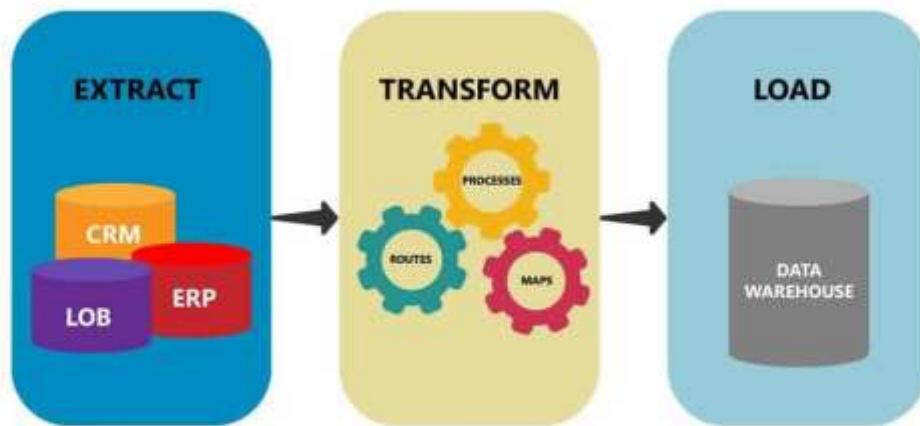
O processo de **Transformação de Dados** é composto por várias etapas : padronização, limpeza, qualidade. Dados vindos de sistemas diferentes tem padrões diferentes seja de nomenclatura ou mesmo de tipos de dados (VARCHAR2 Oracle ou VARCHAR Sql Server.



ETL - Extract, Transform, Load

3) Load

O processo de **Load** é a etapa final onde os dados são lidos das áreas de **staging** e preparação de dados, carregados no **Data Warehouse ou Data Mart Final**.



ETL - Extract, Transform, Load

Algumas vantagens para ETL

Garantia significativa da qualidade dos dados

A Ferramentas de ETL, através de sequências de operações e instruções tem condições de solucionar problemas de maior complexidade.



Algumas vantagens para ETL

Funcionalidade de execução

Uma ferramenta de ETL já possui suas funções específicas, sendo necessária apenas a atenção no fluxo de dados.



Algumas vantagens para ETL

Desenvolvimento das cargas

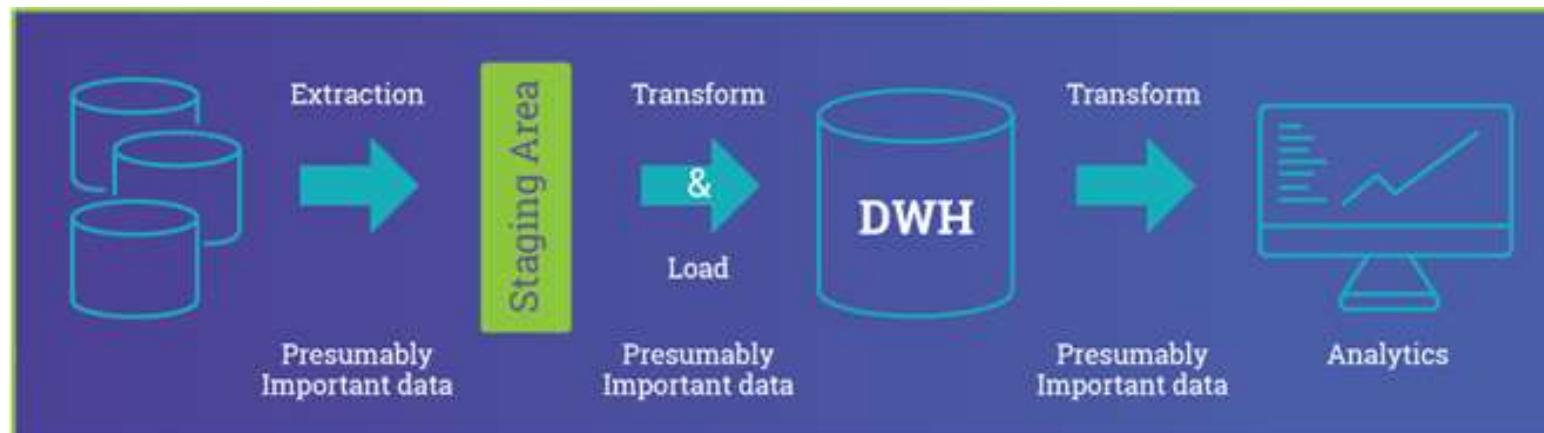
Mesmo que o usuário não seja técnico poderá desenvolver uma rotina de carga em uma ferramenta de ETL, devido a facilidade e rapidez para codificação.



Algumas vantagens para ETL

Manutenção das cargas

As tarefas de manutenção de uma rotina de carga são mais simples de realizar em relação à manutenção de código.



Algumas vantagens para ETL

Metainformação

Os metadados (informações úteis para identificar, localizar, entender e gerenciar os dados) são gerados e mantidos de forma automática com a ferramenta, evitando problemas de geração de informações incorretas na finalização do processo.



Algumas vantagens para ETL

Performance

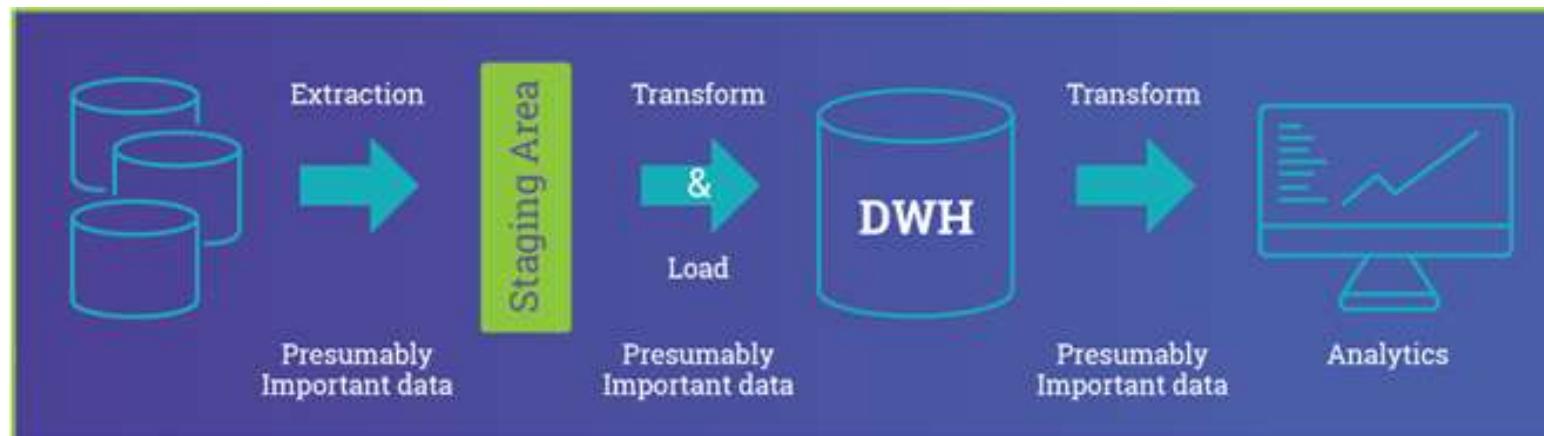
Os métodos mais usados para trabalhar com grandes volumes conseguem extrair, transformar e carregar dados com maior velocidade e menos recursos, como gravações em bloco e operações não logadas.



Algumas vantagens para ETL

Transferência

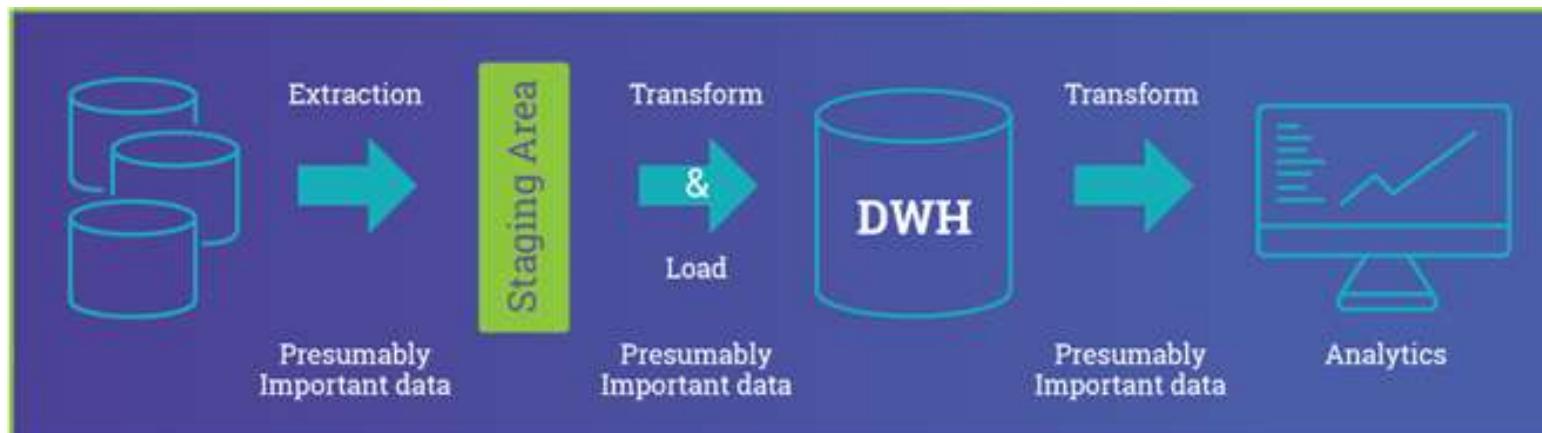
Ferramentas de ETL podem ser deslocadas de um servidor mais facilmente ou distribuídas entre vários servidores.



Algumas vantagens para ETL

Conectividade

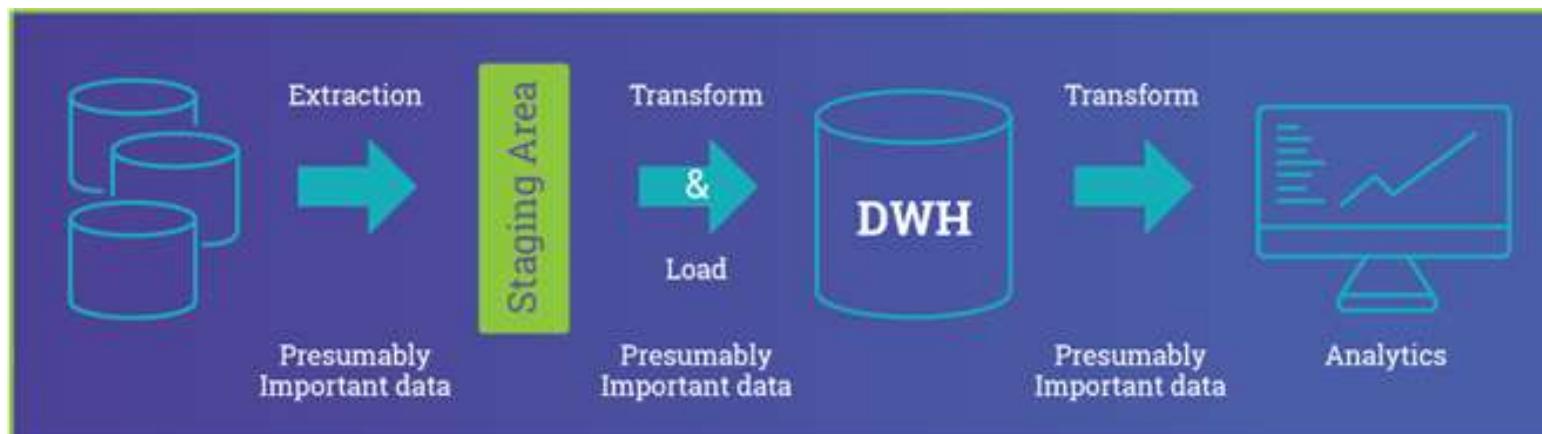
A conexão de uma ferramenta de ETL com múltiplas fontes de dados é transparente. Caso sejam precisas mais fontes como o SAP, VSAM, Mainframe ou qualquer outra, basta a aquisição do conector sem a necessidade de codificar um.



Algumas vantagens para ETL

Reinicialização

Ferramentas possuem a capacidade de reiniciar a carga de onde pararam sem a necessidade de codificação.



Algumas vantagens para ETL

Segurança e Estabilidade

É possível articular melhor a segurança tornado-a mais modular, dividindo as finalidades (criação de cargas, execução de cargas, agendamento, etc.)



Obrigado!

Prof. Dr. Diego Bruno
Machine Learning



Introdução para ETL (Extract, Transform, Load)



Prof. Dr. Diego Bruno

Education Tech Lead na DIO

Doutor em Robótica e *Machine Learning* pelo ICMC-USP

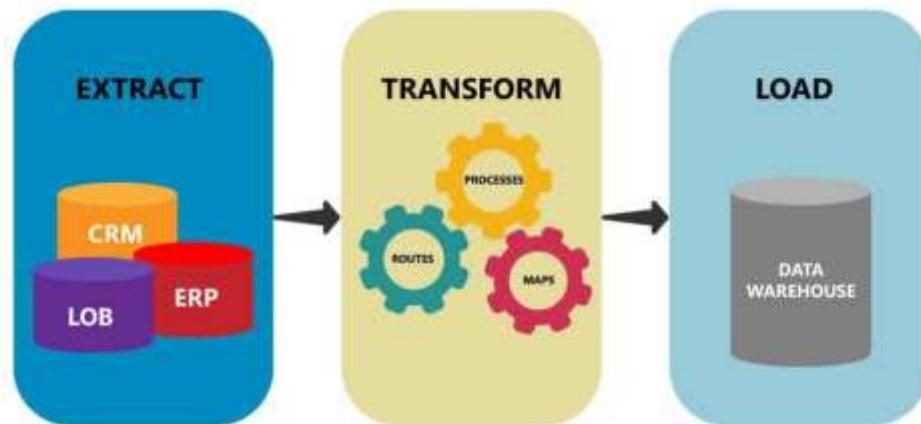
Introdução para ETLs

Prof. Dr. Diego Bruno



Mas o que é ETL?

ETL é um tipo de *data integration* em três etapas (extração, transformação, carregamento) usado para combinar dados de diversas fontes. Ele é comumente utilizado para construir um *data warehouse*.



ETL - Extract, Transform, Load

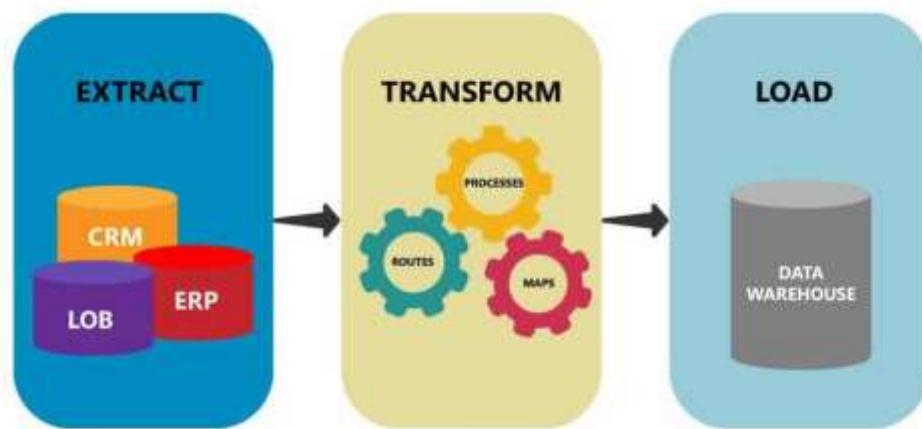
Mas o que é ETL?

ETL é um tipo de ***data integration*** em três etapas (extração, transformação, carregamento) usado para combinar dados de diversas fontes. Ele é comumente utilizado para construir um ***data warehouse***.



Mas o que é ETL?

ETLs, são ferramentas de software cuja função é a extração de dados de diversos sistemas, transformação desses dados conforme regras de negócios e por fim o carregamento dos dados geralmente para um *Data Mart* e/ou *Data Warehouse*.



ETL - Extract, Transform, Load

Mas o que é ETL?

Nesse processo, os dados são retirados (extraídos) de um sistema-fonte, convertidos (transformados) em um formato que possa ser analisado, e armazenados (carregados) em nuvem ou outro sistema. Extração, carregamento, transformação (ELT) é uma abordagem alternativa, embora relacionada, projetada para jogar o processamento para o banco de dados, de modo a aprimorar a performance.



Ferramentas

Existem muitas ferramentas de ETL disponíveis no mercado como: **IBM Information Server (Data Stage)**, o **Oracle Data Integrator (ODI)**, o **Informatica Power Center**, o **Microsoft Integration Services (SSIS)**. Existe também um conjunto de Ferramentas de ETL Open Source como o PDI – Pentaho Data Integrator e Talend ETL.



Processo de ETL

O processo de extração, transformação e carregamento (ETL) abrange alguns passos importantes. Como exemplo, podemos considerar um Banco de dados de Clientes Especiais com todas as informações essenciais.



Processo de ETL

No mapeamento, a extração de origem deve conter a especificação da identidade e seus atributos detalhados, tudo armazenado numa zona temporária. Quando forem efetuadas as análises e filtragens dos dados, a nova versão poderá ser comparada com a cópia da versão prévia.



Processo de ETL

A transformação inclui limpeza, racionalização e complementação dos registros. O processo de limpeza removerá erros e padronizará as informações. O processo de complementação implicará no acréscimo de dados.



Obrigado!

Prof. Dr. Diego Bruno

