



Apache Beam

introdução rápida

Agenda

- Contextualizando (Análise de dados e Processamento de dados)
- Onde o Apache Beam se encaixa
- O que é Apache Beam?
- Porque Apache Beam
- Programando em Apache Beam
 - Pipelines
 - Collections
 - Transforms
- Exemplo de pipelines
- Google Dataflow Service
- Conclusão

Sobre mim

Rafael Ribeiro

Cientista/Engenheiro de Dados



Rafael Ribeiro

Data Scientist at KaBuM!



Análise de dados & Processamento de dados

Análise de dados é a prática que compreende todo o ciclo para a geração de insights, desde captura dos dados até a qualidade e acesso a estes.

Processamento de dados é um componente dentro da análise de dados. Transforma os dados brutos em informações valiosas para o negócio.



Processamento de dados



Ingestão

Lê os dados da
cloud ou on-premise

**Processamento
ou
Transformação**

Define as lógicas
de negócio

Escrita

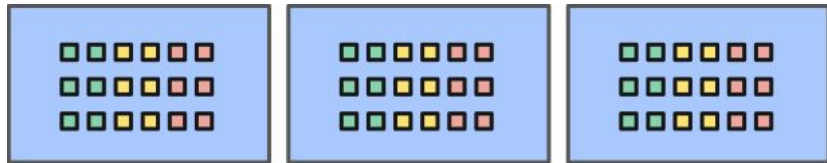
Escreve os dados no
destino escolhido

Tipos de processamento de dados

Batch

Os dados são coletados e processados em pedaços (lotes)

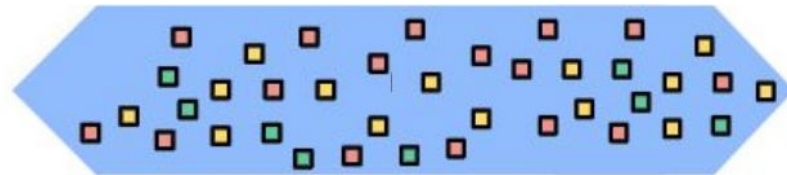
ex: Folhas de pagamento, Contas de telefone, etc



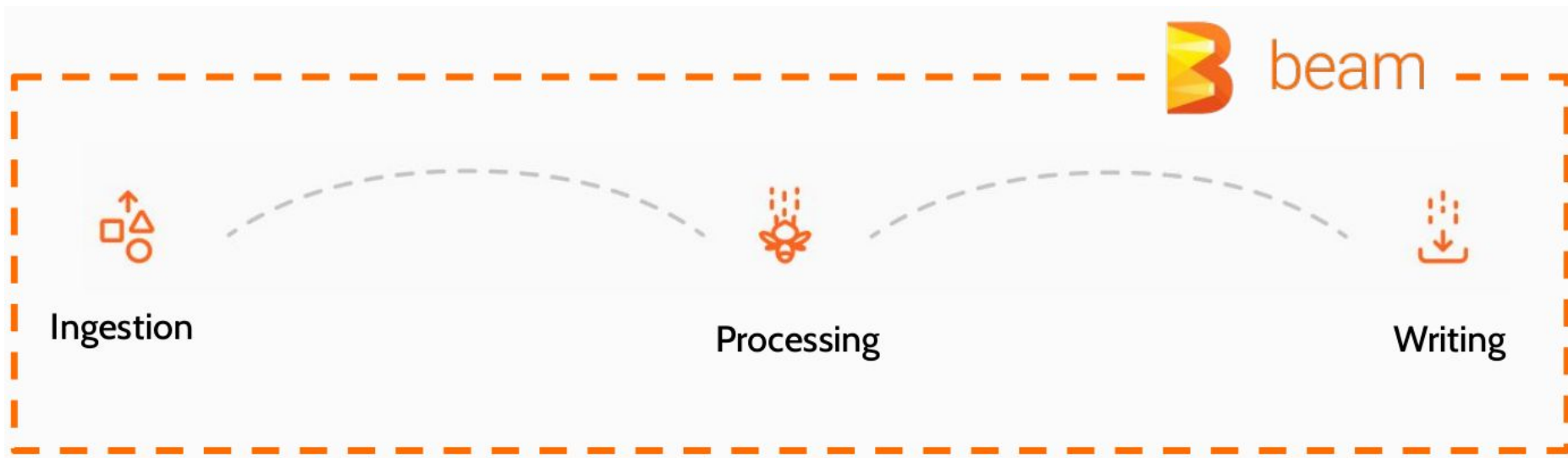
Streaming

É o processamento contínuo dos dados

ex: detecção de anomalia, alertas, etc




Onde o Apache Beam se encaixa



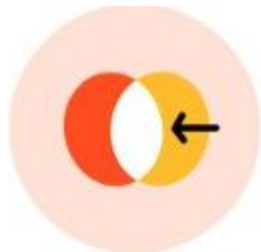
Um equívoco comum...

Apache Beam é um **substituto** para Apache Spark e/ou Apache Flink



<https://beam.apache.org/get-started/from-spark/>

Porque usar Apache Beam?



Unificado

Use um modelo único de programação para processamento em Batch ou Streaming



Extensível

Você pode escrever novas SDKs, conectores I/O e etc



Portável

Execute pipelines em múltiplos ambientes de execução

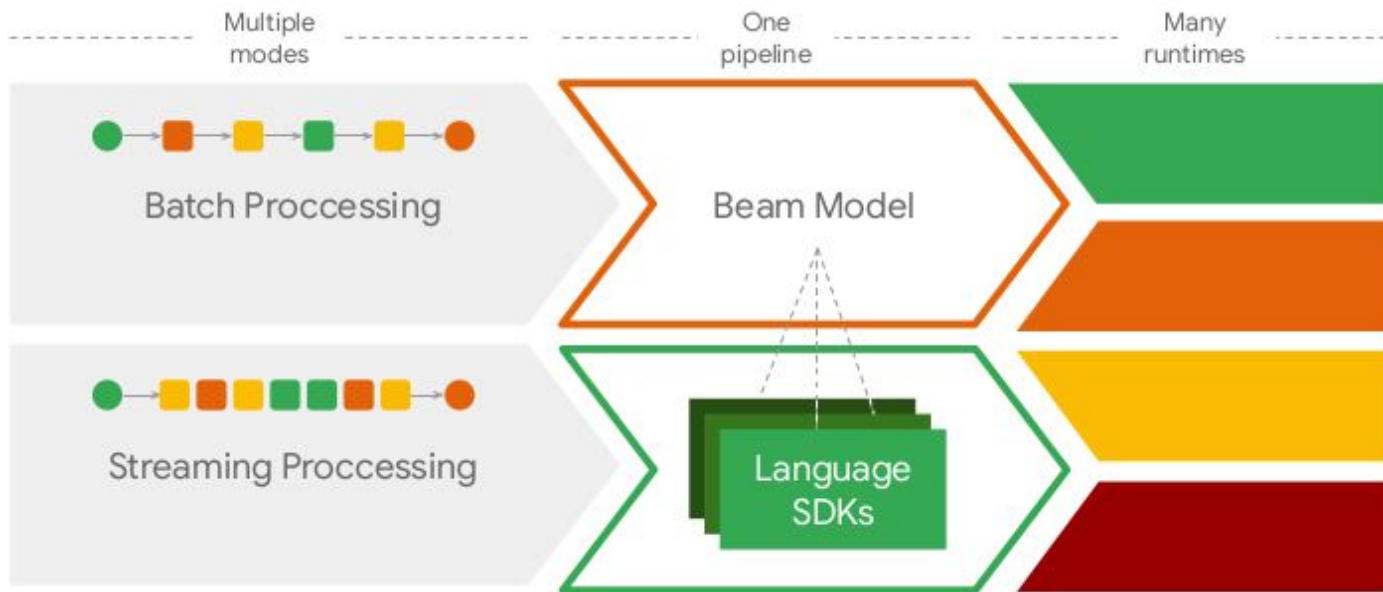


Código Aberto

Desenvolvimento e o suporte se apoia na comunidade para ajudar e evoluir o sistema.

O que é o Apache Beam

Apache Beam é um **modelo unificado** de programa para construir pipelines de processamento de dados **batch e streaming**



Linguagens suportadas (SDKs)



Maturidade dos SDKs



Java™



python™



Java > Python > Go



Scala

Runners suportados



<https://beam.apache.org/documentation/runners/capability-matrix/>

Apache Beam

Java

```
input.apply(  
    Sum.integersPerKey()  
)
```

Python

```
input | Sum.PerKey()
```

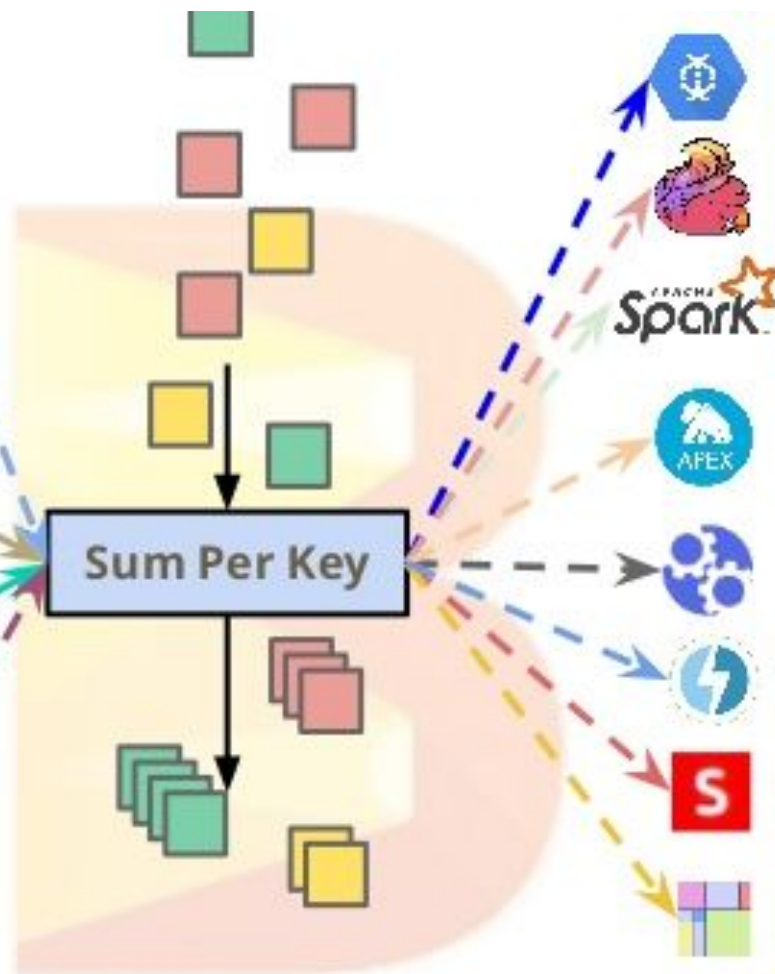
Go

```
stats.Sum(s, input)
```

SQL

```
SELECT key, SUM(value)  
FROM input GROUP BY key
```

⋮



Cloud Dataflow

Apache Flink

Apache Spark

Apache Apex

Gearpump

IBM Streams

Apache Samza

Apache Nemo
(incubating)

⋮

Ecosistema

Spring 2019

InfoWorld
FROM IDG

Technology of the year
2019 Awards

Summer 2020



Top 5 project in 4
categories

Significant Developer
Interest

4 years in a row!

Top 5 repositórios apache por número de commits

Top 5 lista de email ativa (user@ + dev@)

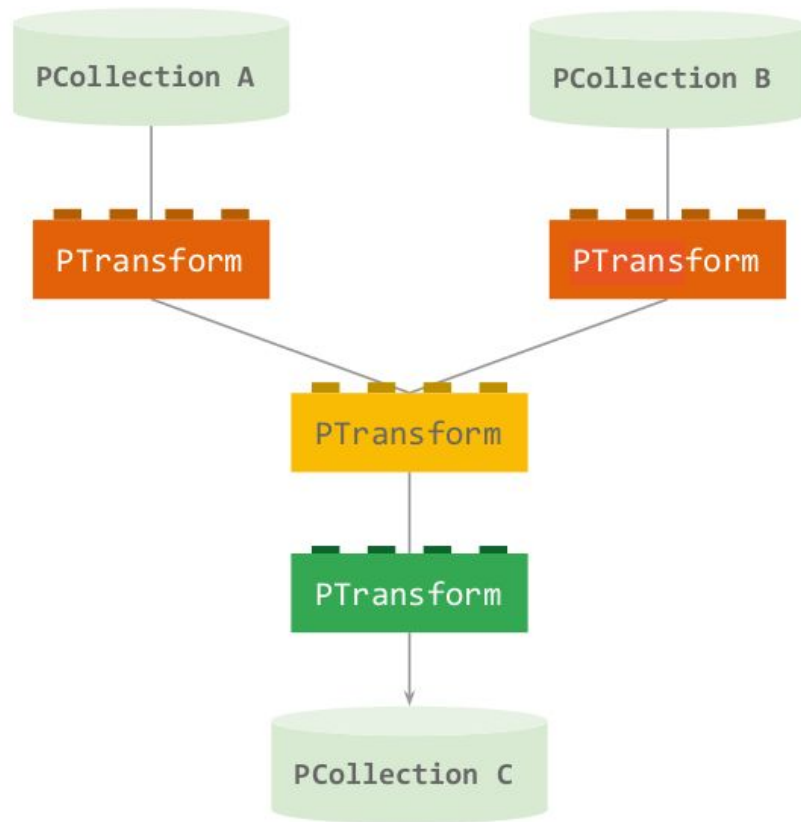
Top 5 mais ativo código do Apache - clones

Top 5 mais ativo código do Apache - visitas

O que é uma pipeline?

Pipeline é uma DAG (Directed Acyclic Graph) onde transformações (**PTransform**) são realizadas sobre os uma ou mais coleção de dados (**PCollection**).

Pode trabalhar com múltiplas entradas e saídas de dados.



Coleções - PCollection

No Apache Beam os dados vivem em uma **PCollection** (**Parallel Collection**)

Uma **PCollection** é como uma lista de elementos.

Pode conter dados com ou sem limites (bounded and unbounded)

PCollections -- *Parallel Collections* *[a]*

[]

[1, 2, 3]

[(1, 'a'), (2, 'b'), (3, 'c')]

[1, 2, 1, ...]

Definição de uma PCollection

- **PCollection** só pertence a uma pipeline e não pode ser compartilhada
- Elementos de uma **PCollection** podem ser de qualquer tipo
- Todos os elementos de uma **PCollection** devem ser do mesmo tipo
- Uma **PCollection** é imutável
- Cada elemento de uma **PCollection** tem um timestamp associado

Transformações - PTransforms

PTransforms (**Parallel Transforms**) é uma operação dentro da pipeline

PTransform é como uma função.

PTransforms -- *Parallel Transforms* $a \rightarrow b$

$x \rightarrow x * 2$

$\text{word} \rightarrow \text{word.lower}()$

$\text{xs} \rightarrow \text{sum}(\text{xs}) / \text{len}(\text{xs})$

$x \rightarrow x == 42$

Observações sobre PTransforms

- Para invocar uma transformação, você deve **aplicar** sobre uma PCollection de entrada.
- Transformações podem ser **aninhadas (nested)**
- A transformação **não altera** os dados da PCollection de entrada
- O SDK do Beam tem transformações **core** e **composições pré escritas**

Principais Transformações



<https://beam.apache.org/documentation/transforms/python/overview/>

Transformações de Entrada/Saída

Podemos ler e escrever os dados em diversas fontes diferentes

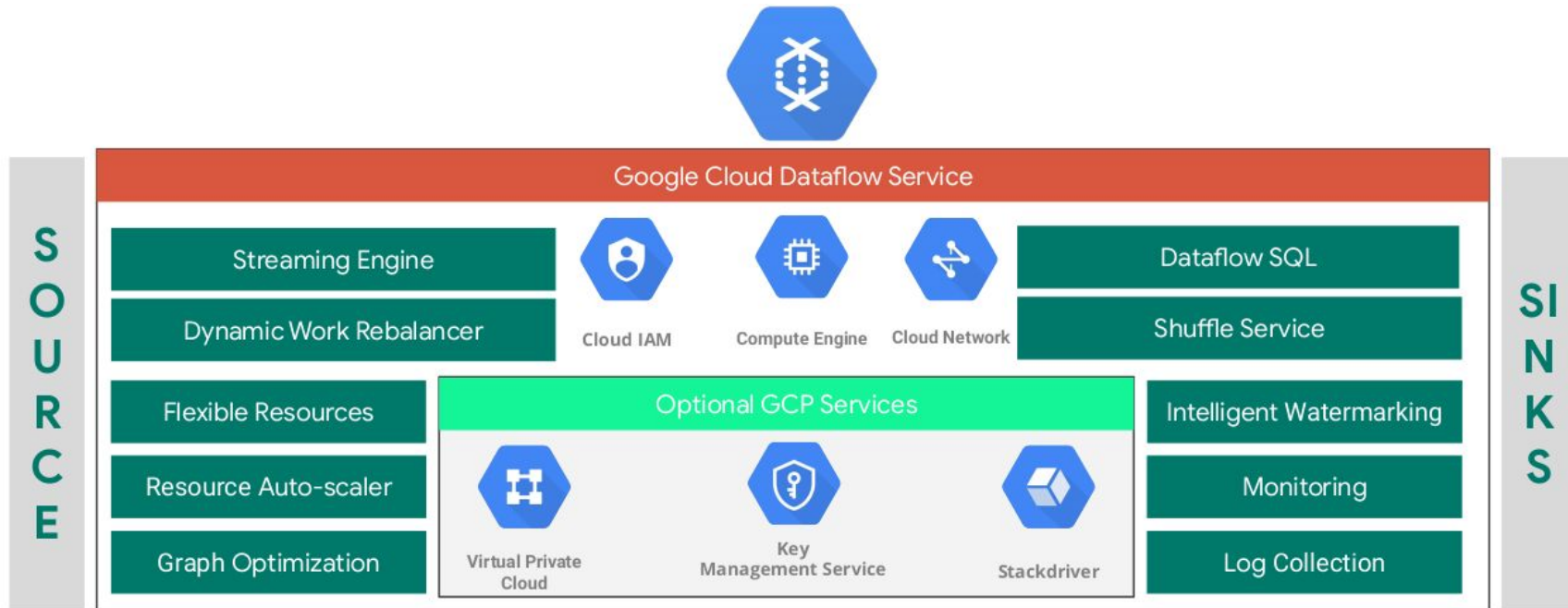
O Apache Beam disponibiliza algumas transformações para ler e escrever dados nas mais diferentes fontes.

<https://beam.apache.org/documentation/io/built-in/>

Criando Pipelines/Coleções/Transformações

Vamos criar uma pipeline completa!

Google Dataflow Service



Conclusão

- é um **modelo unificado** para pipelines **batch e streaming**
- é **portável**
- você pode codificar com sua **linguagem favorita**
- grande coleção de **conectores IO** disponível
- pode criar **seu próprio** conector ou operador
- **compatível** com **Flink, Spark, Dataflow** e etc

- pode **pagar quanto usar** (Google Dataflow)
- **auto-escalável** (Google Dataflow)

Streaming Pipelines

Fica para uma próxima!!

Dúvidas?



Referências

Apache Beam - <https://beam.apache.org/>

Beam College - <https://beamcollege.dev/>