

# Big Data e Machine Learning com Hadoop e Spark



# Conteúdo

## CONTEÚDO PROGRAMÁTICO

- Visão geral da ciência de dados e aprendizado de máquina em escala
- Visão geral do ecossistema do Hadoop
- Instalação de um Cluster Hadoop
- Trabalhando com dados do HDFS e tabelas do Hive usando o Hue
- Visão geral do Python
- Visão geral do R
- Visão geral do Apache Spark 2
- Leitura e gravação de dados
- Inspeção da qualidade dos dados
- Limpeza e transformação de dados
- Resumindo e agrupando dados
- Combinando, dividindo e remodelando dados
- Explorando dados
- Configuração, monitoramento e solução de problemas de aplicativos Spark
- Visão geral do aprendizado de máquina no Spark MLlib
- Extraíndo, transformando e selecionando recursos
- Construindo e avaliando modelos de regressão
- Construindo e avaliando modelos de classificação
- Construindo e avaliando modelos de cluster
- Validação cruzada de modelos e hiperparâmetros de ajuste
- Construção de pipelines de aprendizado de máquina
- Implantando modelos de aprendizado de máquina

## MATERIAL DIDÁTICO

- Slides do treinamento em PDF
- GitHub com exercícios e códigos exemplo
- Máquinas virtuais para exercícios simulados
- Gravação das aulas disponível durante 3 meses





## INTEGRAÇÃO MYSQL & HADOOP

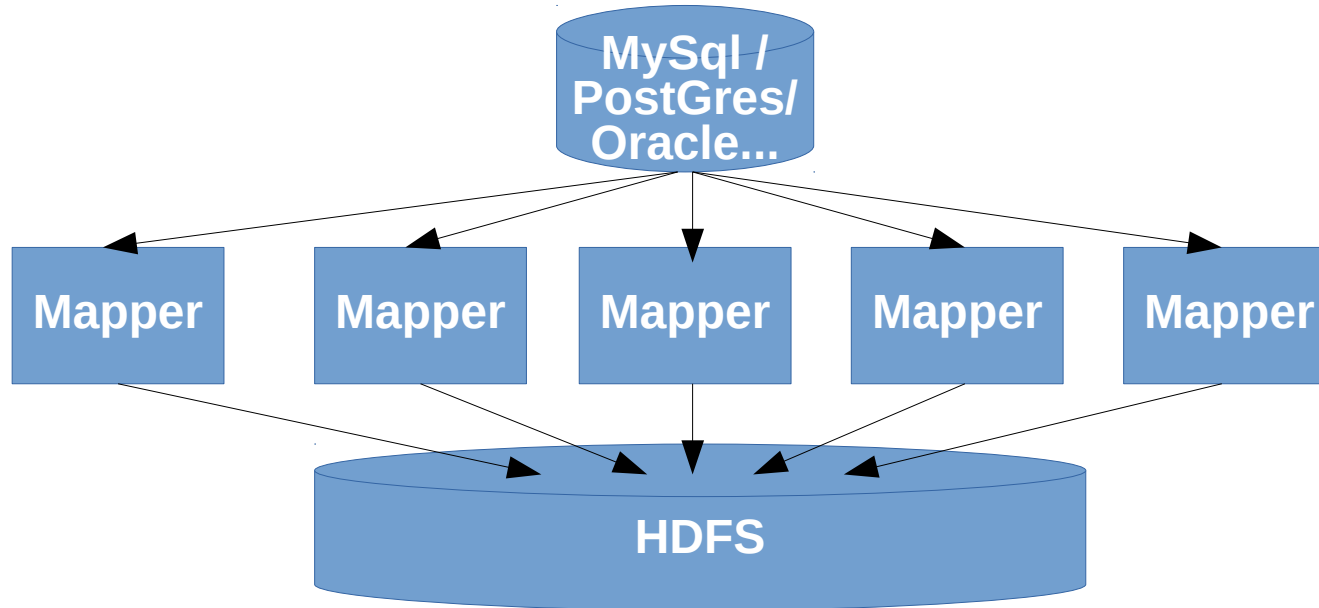
O que é o MySQL?

- Banco de dados relacional popular e gratuito
- Geralmente monolítico por natureza
- Mas, pode ser usado para OLTP - então exportar dados para o MySQL pode ser útil
- Dados existentes podem existir no MySQL que você deseja importar para o Hadoop

# Sqoop

## Sqoop pode manipular BIGDATA

■ Na verdade, inicia trabalhos do MapReduce para manipular a importação ou exportação de seus dados!



# Sqoop

## Importar dados do MySQL para HDFS

```
sqoop import --connect jdbc:mysql://localhost/movielens --driver  
com.mysql.jdbc.Driver --table movies -m 1
```

## Importar dados do MySQL diretamente no Hive

```
sqoop import --connect jdbc:mysql://localhost/movielens --driver  
com.mysql.jdbc.Driver --table movies --hive-import -m 1
```

# Sqoop

## Importações incrementais

- Você pode manter seu banco de dados relacional e o Hadoop em sincronia
- `--check-column` e `-last-value`



# Sqoop

## Sqoop: Exportar dados do Hive para MySQL

- `sqoop export --connect jdbc:mysql://localhost/movielens -m 1 --driver com.mysql.jdbc.Driver --table exported_movies --export-dir /apps/hive/warehouse/movies - --input-fields-terminated-by '\0001'`
- A tabela de destino já deve existir no MySQL, com colunas na ordem esperada

# Sqoop

Vamos praticar com o MySQL e o Sqoop

- Importe dados do MovieLens para um banco de dados MySQL
- Importe os filmes para o HDFS
- Importe os filmes para o Hive
- Exportar os filmes de volta para o MySQL





# Zookeeper

O que é o ZooKeeper?

- Basicamente, ele controla as informações que devem ser sincronizadas em todo seu cluster

- Qual nó é o mestre?

- Quais tarefas são atribuídas a quais workers?

- Quais workers estão atualmente disponíveis?

- É uma ferramenta que os aplicativos podem usar para recuperar falhas parciais no cluster.

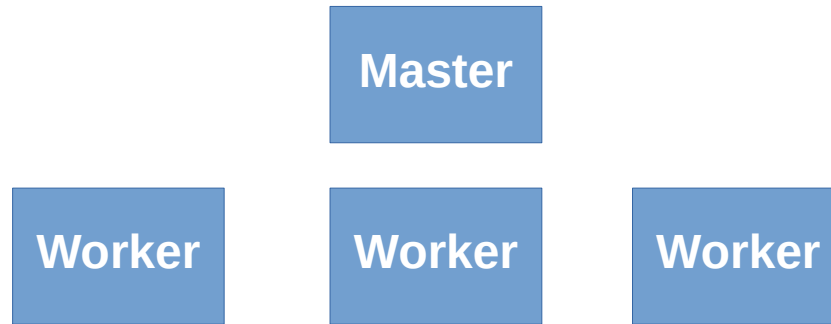
- Uma parte integrante do HBase, High-Availability (HA) MapReduce, Drill, Storm, Solr, e muito mais



# Zookeeper

## Modos de falha

- Master falha, precisa fazer failover para um backup
- Worker falha - seu trabalho precisa ser redistribuído
- Problemas de rede - parte do seu cluster não consegue ver o restante



# Zookeeper

Operações “primitivas” em um sistema distribuído

- Eleição mestre

- Um nó se registra como um mestre e mantém um "bloqueio" nesses dados
- Outros nós não podem se tornar mestre até que esse bloqueio seja liberado
- Apenas um nó permitido manter o bloqueio de cada vez

- Detecção de falhas

- Os dados "efêmeros" sobre a disponibilidade de um nó desaparecem automaticamente se o nó desconecta ou falha ao atualizar após um período de tempo limite.

- Gerenciamento de grupo

- Metadados

- Lista de tarefas pendentes, atribuições de tarefas



# Zookeeper

API do ZooKeeper:

- Criar, excluir, existir, setData, getData, getChildren

/

|

|-----/master "Master1.foobar.com:2223"

|

|-----/workers

|

|----- worker-1 "worker-2.foobar.com:2225"

|----- worker-2 "worker-5.foobar.com:2225"



# Zookeeper

## Notificações

- Um cliente pode se registrar para notificações em um znode
  - Evita a pesquisa contínua
  - Exemplo: registrar para notificação no /master - se ele parar, tente assumir como o novo mestre.



# Zookeeper

## Znodes persistentes e efêmeros

- Os znodes persistentes permanecem armazenados até serem excluídos explicitamente

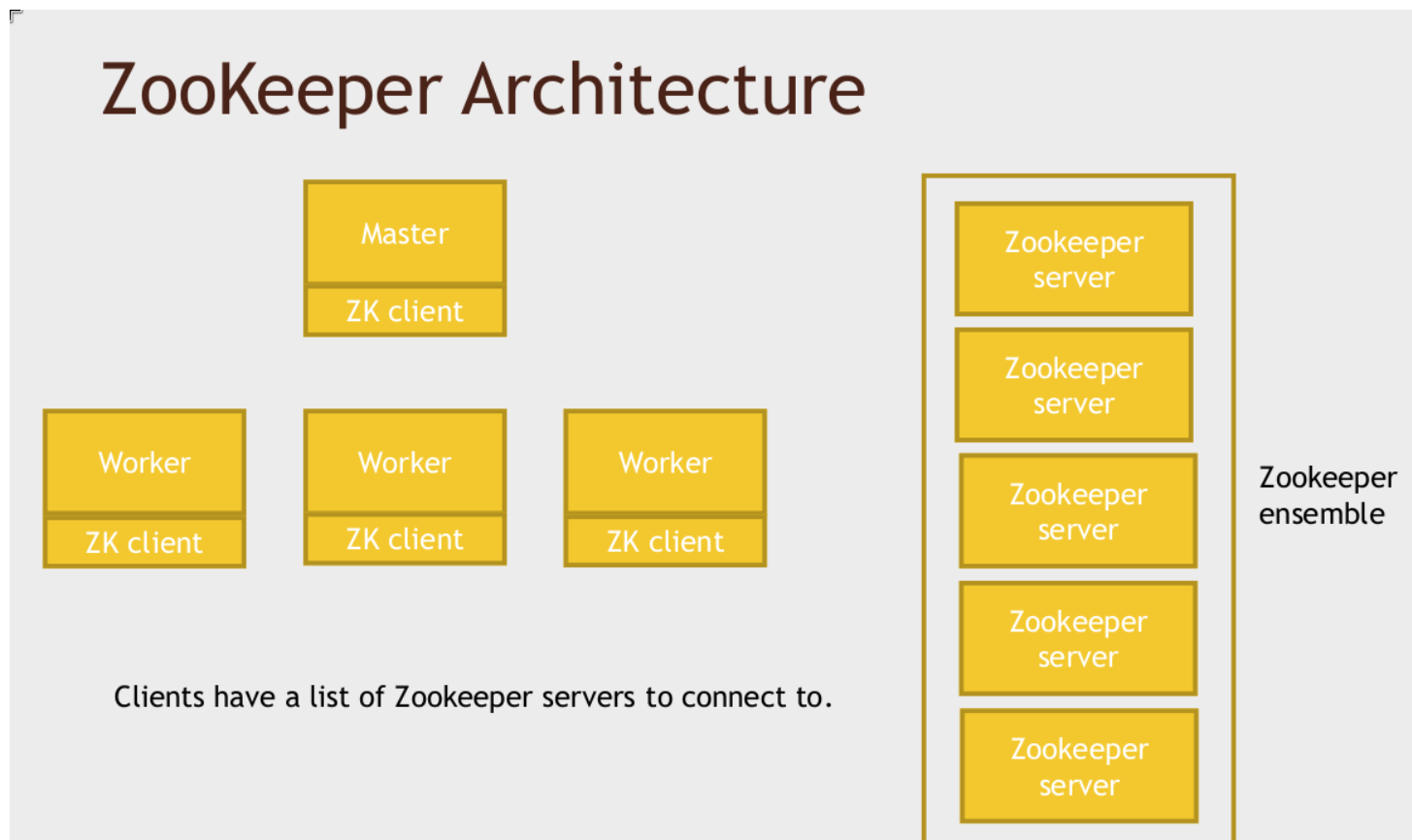
- isto é, a atribuição de tarefas aos trabalhadores deve persistir mesmo se o mestre travar

- Os znodes efêmeros desaparecem se o cliente que o criou falhar ou perder conexão ao ZooKeeper

- ou seja, se o mestre travar, ele deve liberar seu bloqueio no znode que indica qual nó é o mestre!



# Zookeeper



# Storm

Processamento de fluxo em tempo real

O que é o Apache Storm?

- Framework para processar fluxos contínuos de dados em um cluster
  - Pode rodar em cima do YARN (como Spark)
- Funciona em eventos individuais, não em micro lotes (como o Spark Streaming)
  - Se você precisa de latência de sub-segundo, Storm é a opção

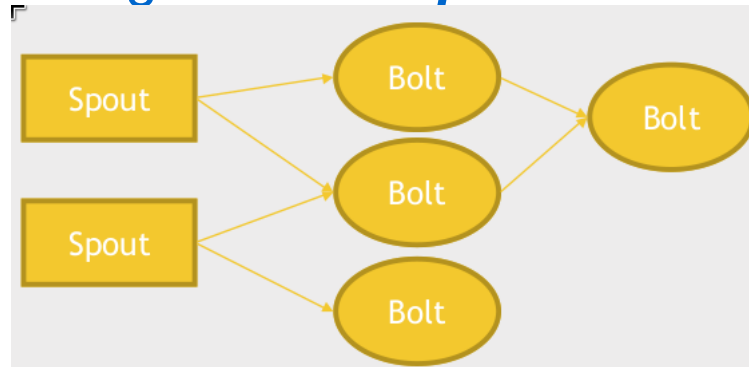




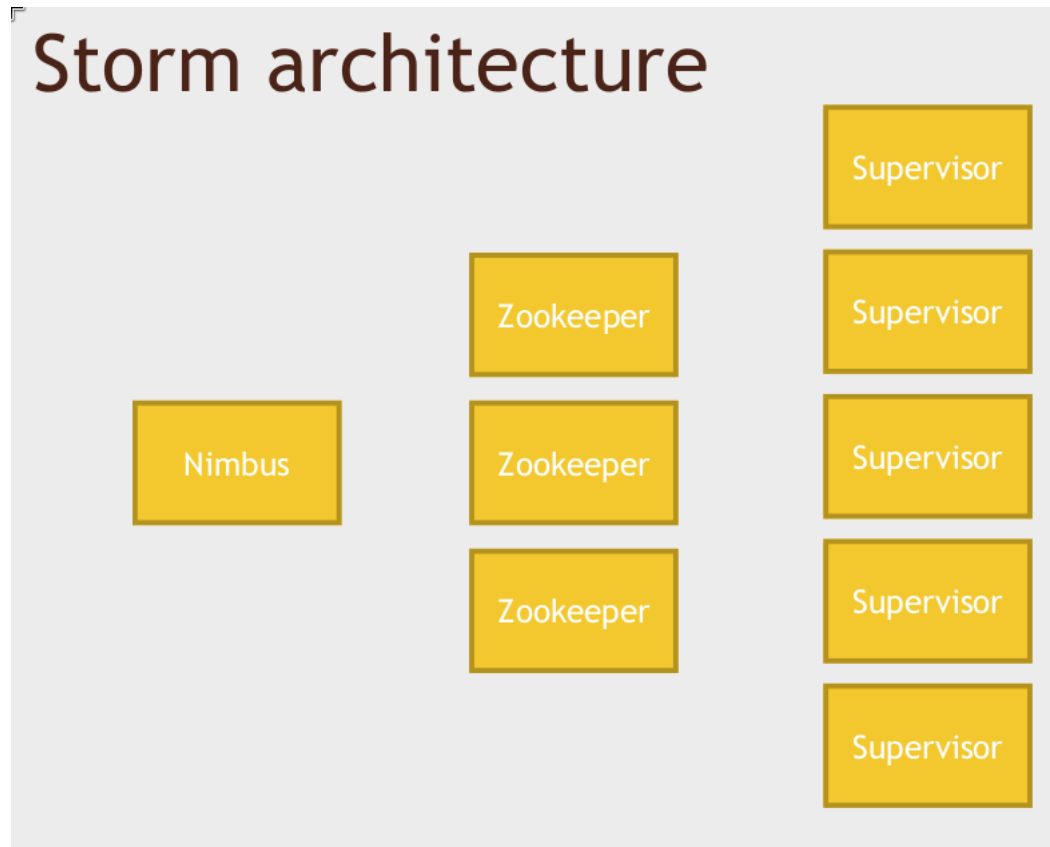
# Storm

## Terminologia do Storm

- Um **stream** consiste em tuplas que fluem através de...
- **Spouts** que são fontes de dados de fluxo (Kafka, Twitter, etc.)
- **Bolts** que processam dados de fluxo conforme são recebidos
  - Transformar, agregar, gravar em bancos de dados / HDFS
- Uma **topology** é um gráfico de **Spouts** e **Bolts** que processam seu fluxo



# Storm



# Storm

## Desenvolvendo aplicativos Storm

- Geralmente feito com Java

- Embora os bolts possam ser direcionados através de scripts em outros idiomas

## Núcleo do Storm

- A API de nível inferior para o Storm

- Semântica "pelo menos uma vez"

- Trident

- API de nível superior para tempestade

- semântica "Exactly once"

- O Storm executa seus aplicativos "para sempre" depois de enviado - até você explicitamente pare



# Storm

## Storm vs. Spark Streaming

- Apesar do Spark ter mais ferramentas...  
se você precisar de processamento em tempo real (sub-segundo) de eventos, em, Storm é sua escolha
- Kafka + Storm é uma combinação bastante popular



# Flume

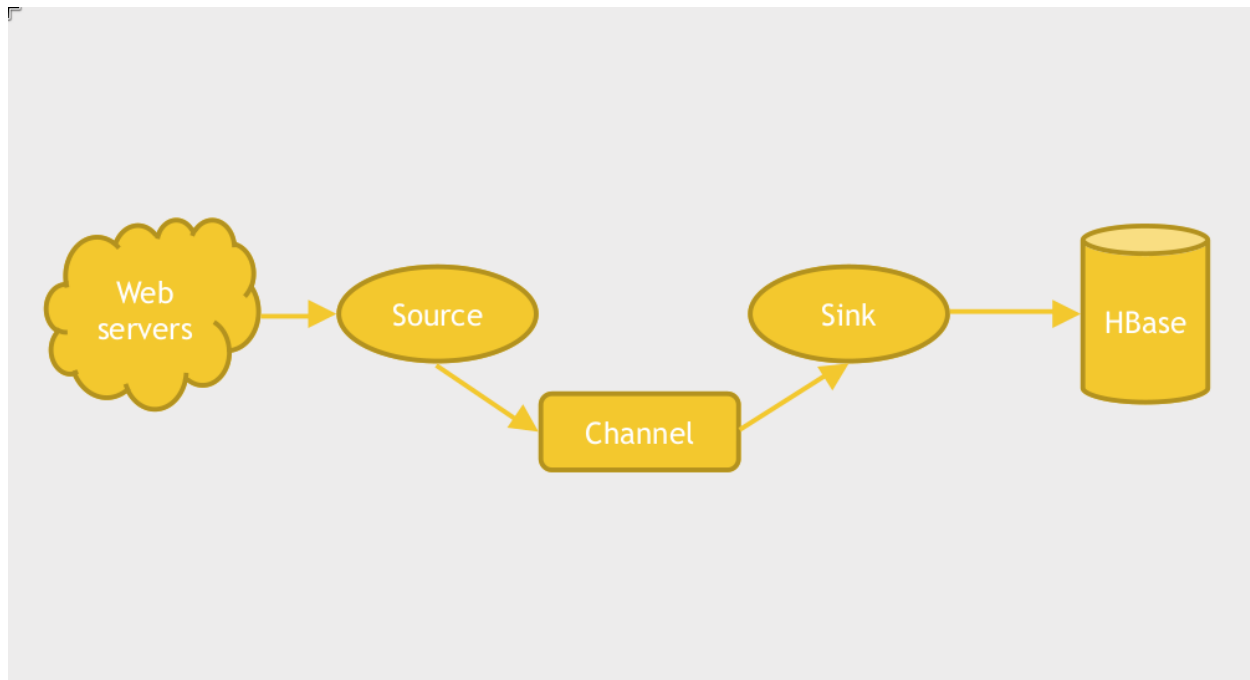
O que é o Flume?

- Outra maneira de transmitir dados em seu cluster
- Feito desde o início com o Hadoop em mente
  - Dissipadores embutidos para HDFS e Hbase
- Originalmente feito para manipular a agregação de logs



# Flume

## Anatomia de um Flume Agent and Flow



# Flume

## Componentes de um agente

### ■ Fonte

- De onde vêm os dados
- Opcionalmente, pode ter Seletores de Canal e Interceptores

### ■ Canal

- Como os dados são transferidos (via memória ou arquivos)

### ■ Pia (Sink)

- Onde os dados estão indo
- Pode ser organizado em grupos de pia
- Uma pia pode se conectar a apenas um canal



# Flume

## Componentes de um agente

### ■ Fonte

- De onde vêm os dados
- Opcionalmente, pode ter Seletores de Canal e Interceptores

### ■ Canal

- Como os dados são transferidos (via memória ou arquivos)

### ■ Pia (Sink)

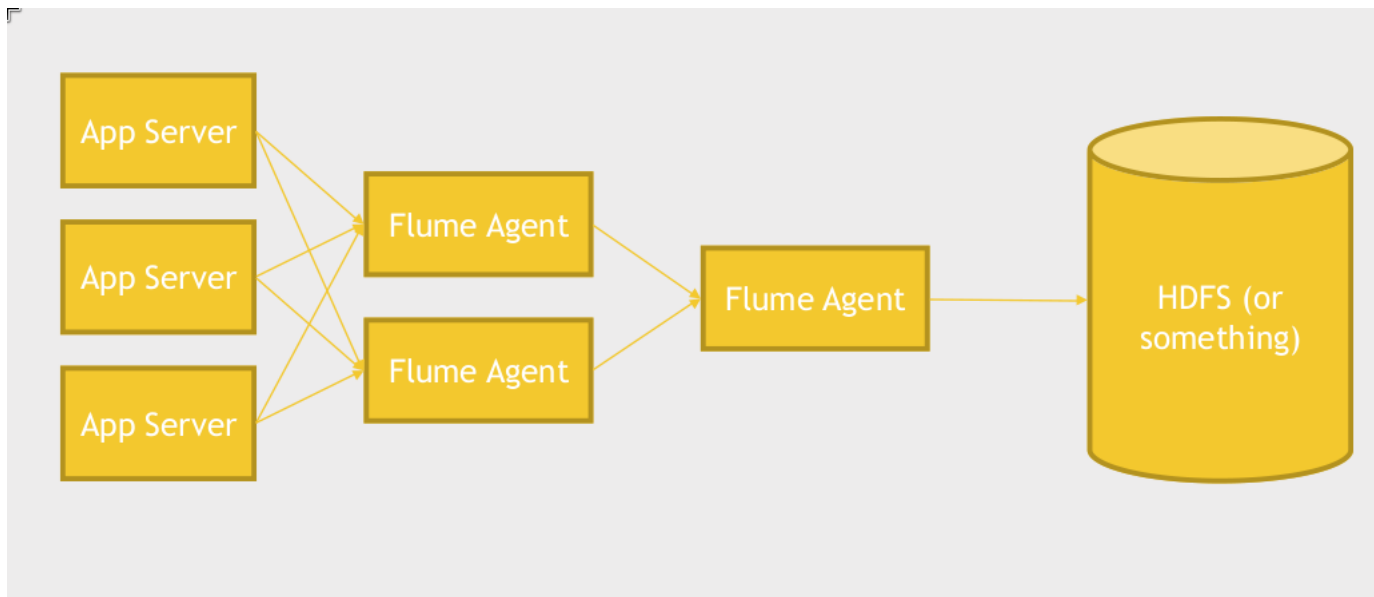
- Onde os dados estão indo
- Pode ser organizado em grupos de pia
- Uma pia pode se conectar a apenas um canal

O canal é notificado para excluir uma mensagem quando o coletor a processa.



# Flume

Usando o Avro, os agentes podem se conectar outros agentes também



# Flume

## Tipos de Sink Built-in

- HDFS
- Hive
- HBase
- Avro
- Thrift
- Elasticsearch
- Kafka
- Personalizado
- E mais!



# Kafka

## STREAMING COM KAFKA

Publicar / Assinar Mensagens com Kafka

O que é streaming?

- Conversamos sobre o processamento de big data histórico e existente
  - residente no HDFS
  - residente em um banco de dados
- Mas como os novos dados chegam ao seu cluster? Especialmente se for "Big Data"?
  - Novas entradas de log dos seus servidores da web
  - Novos dados do sensor do seu sistema IoT
  - Novas negociações de ações
- Streaming permite publicar esses dados, em tempo real, em seu cluster.
  - E você pode até processá-lo em tempo real quando ele chegar!



# Kafka

## Dois problemas

- Como obter dados de várias fontes diferentes fluindo para o cluster
- Processaestes dados quando chegarem
- Primeiro, vamos nos concentrar no primeiro problema



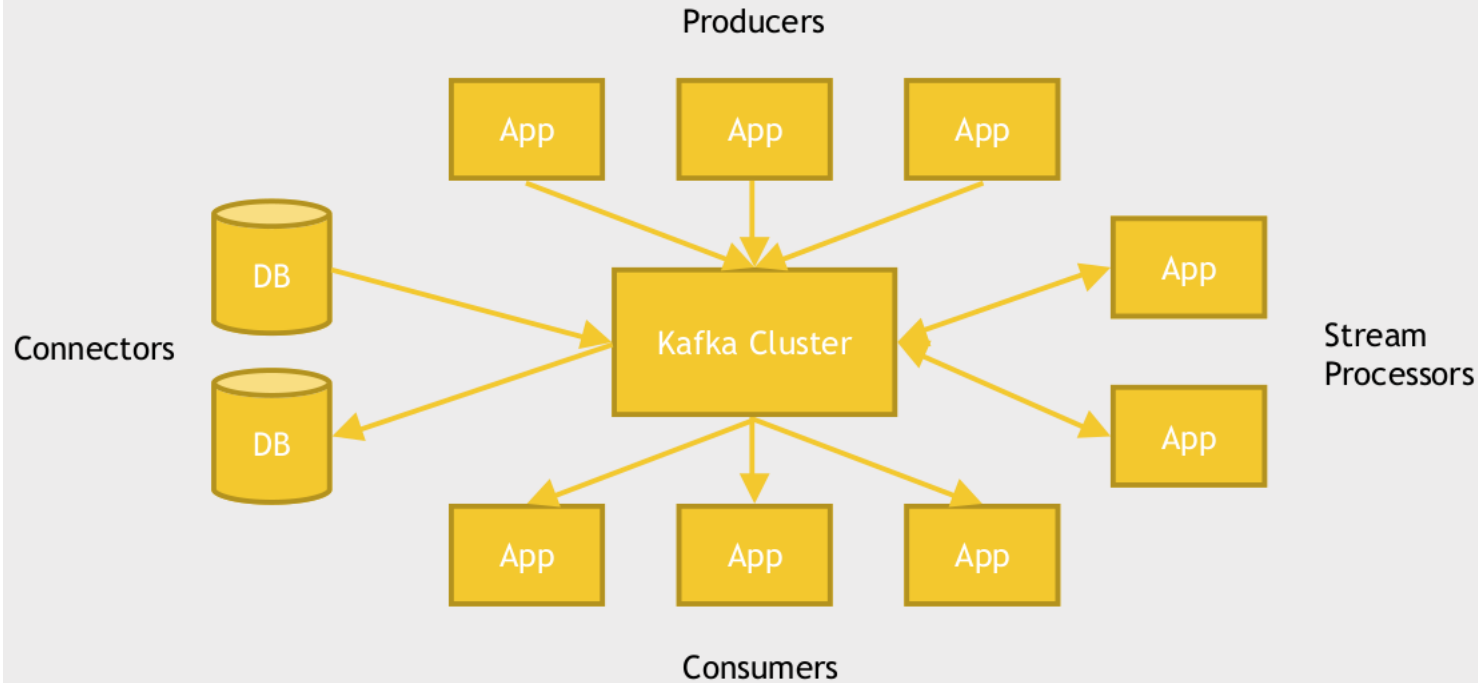
# Kafka

- Kafka é um sistema de publicação / assinatura de mensagens de propósito geral
- Os servidores Kafka armazenam todas as mensagens recebidas dos ***publishers*** por um período de tempo, e publica-os em um fluxo de dados (***stream***) chamado ***topic***.
- Os ***consumers*** se inscrevem em um ou mais ***topics*** e recebem dados a medida que são publicados
- Um stream / topic pode ter muitos ***consumers*** diferentes, todos com suas próprias posições no fluxo mantidas
- Não é apenas para o Hadoop



# Kafka

## Kafka architecture



# Kafka

Como o Kafka escala:

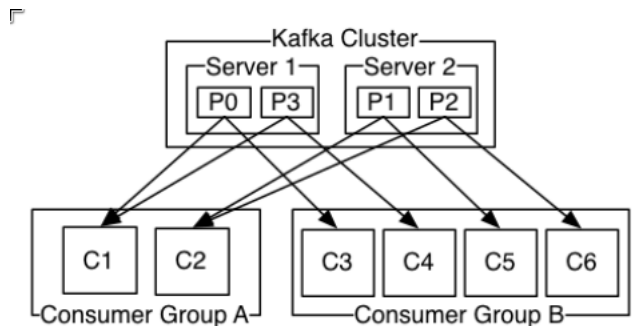
- O próprio Kafka pode ser distribuído entre muitos processos em muitos servidores

- Distribuirá o armazenamento do fluxo dados bem

- Os consumidores também podem ser distribuídos

- Consumidores do mesmo grupo tem mensagens distribuídas entre eles

- Consumidores de diferentes grupos receberão sua própria cópia de cada mensagem



Obrigado!!!

Nos vemos amanhã!!!

Bom descanso!

