

NoSQL como suporte à análise de dados não- normalizados e de grande volume

Joel Alexandre

Orientador: Professor Luís Cavique

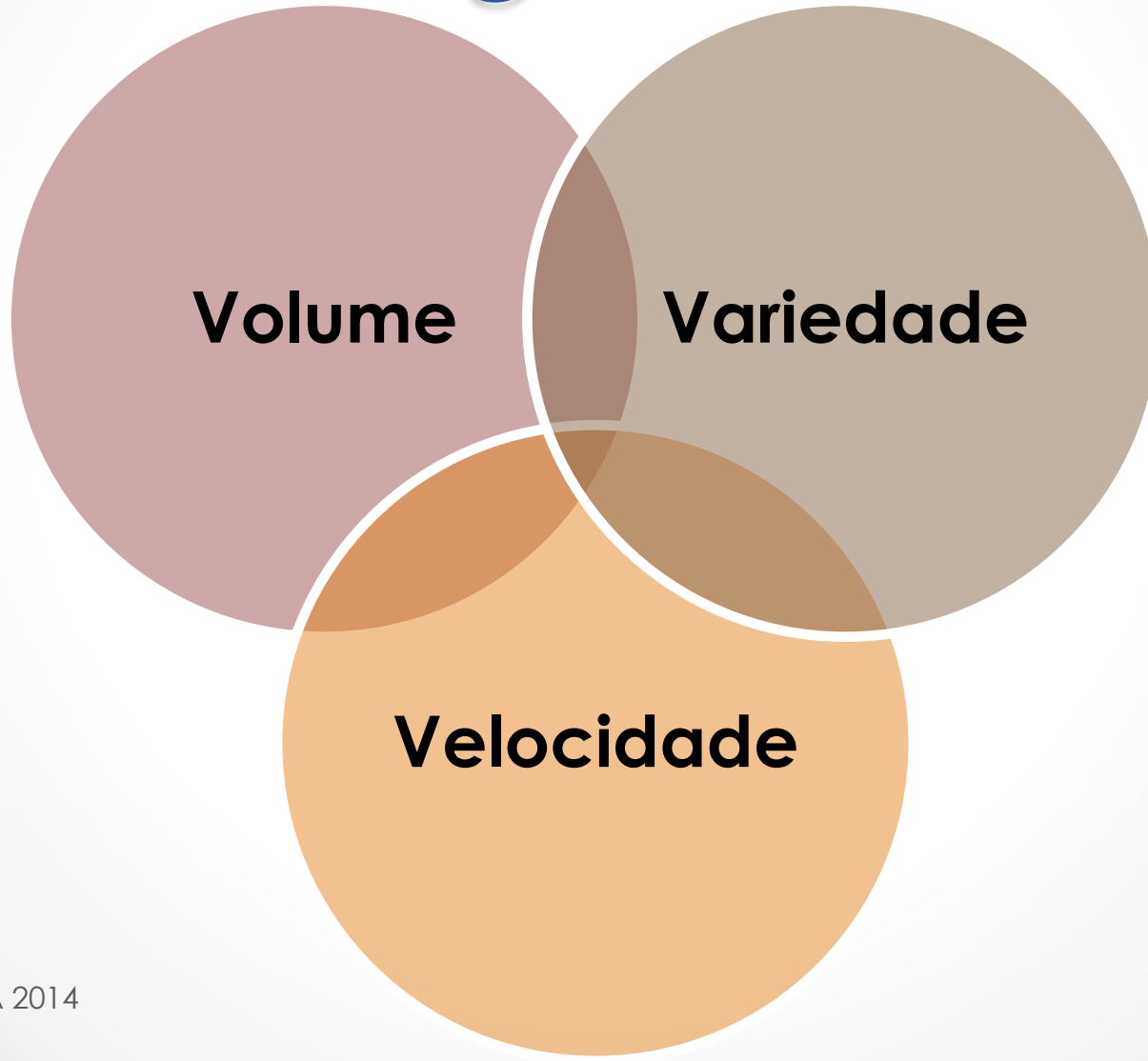
Motivação

- Grande quantidade de registos
- “Tudo” é registado
- *Internet of things* – “*tudo ligado*”
- Informação diversificada
- Processos de análise lentos

Objectivo

Agregação de dados em Big Data

Big Data



Tipos de NoSQL

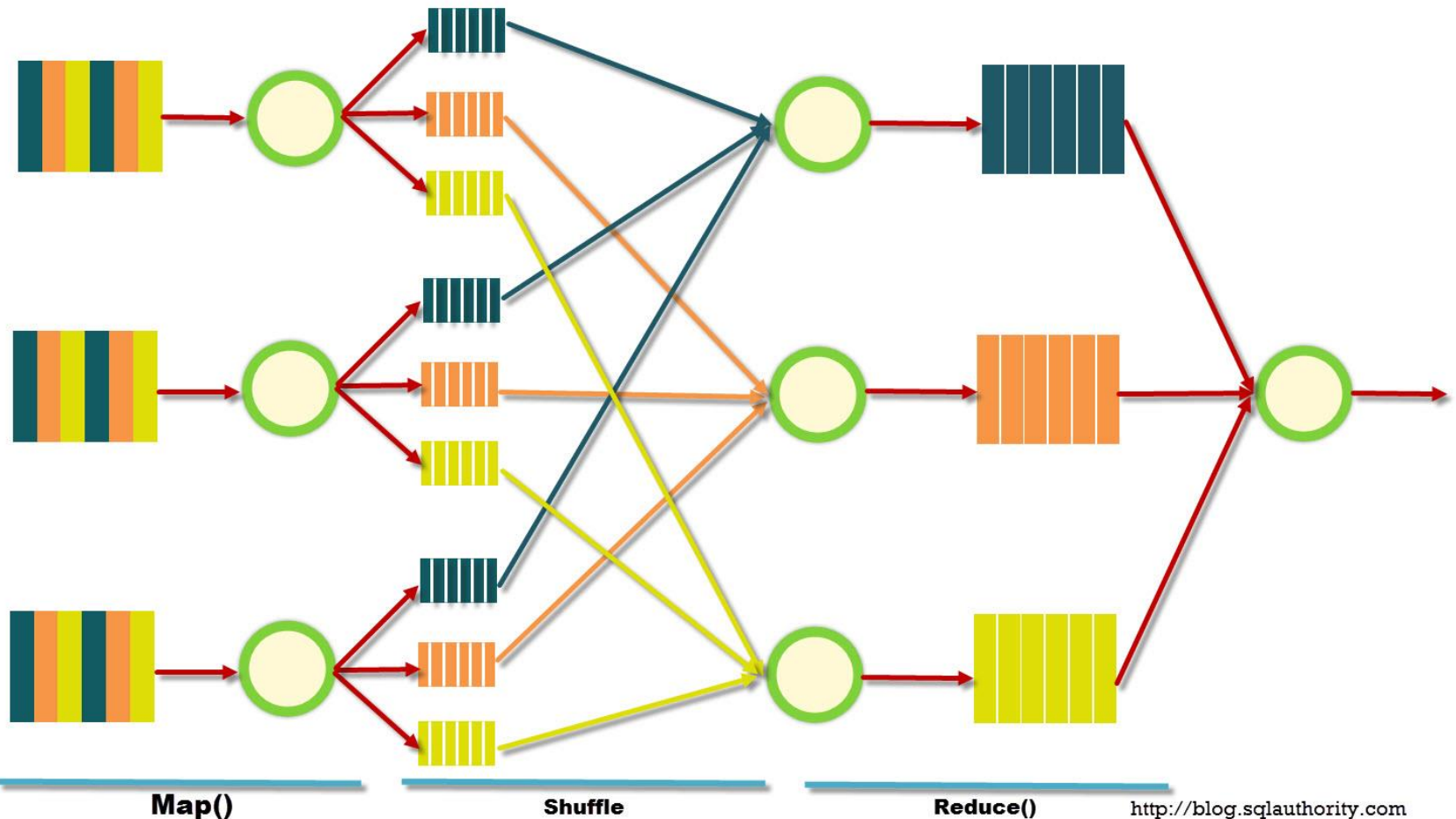
- Chave/Valor (Key/Value)
- Super colunas
- Grafos
- Documentos (xml, json)
- Objectos

Map Reduce

- Google 2004
- Permite tratar grandes volumes de informação de forma distribuida
- Não há um standard entre fornecedores

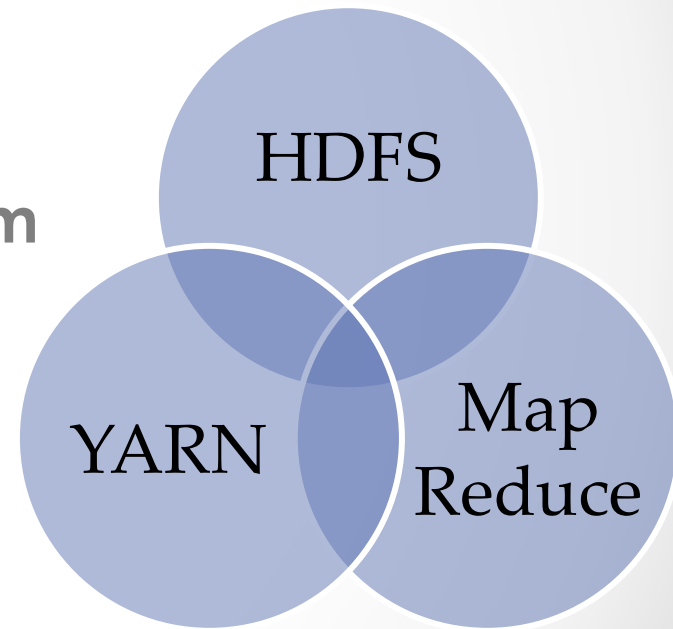
Map Reduce

How MapReduce Works?



Hadoop

- Processo central que gere e coordena o processamento paralelo dos diferentes nós
- Processamento de informação em cada nó
- Central que gere e coordena o armazenamento da informação
- Trata do armazenamento de informação em cada nó



HBase

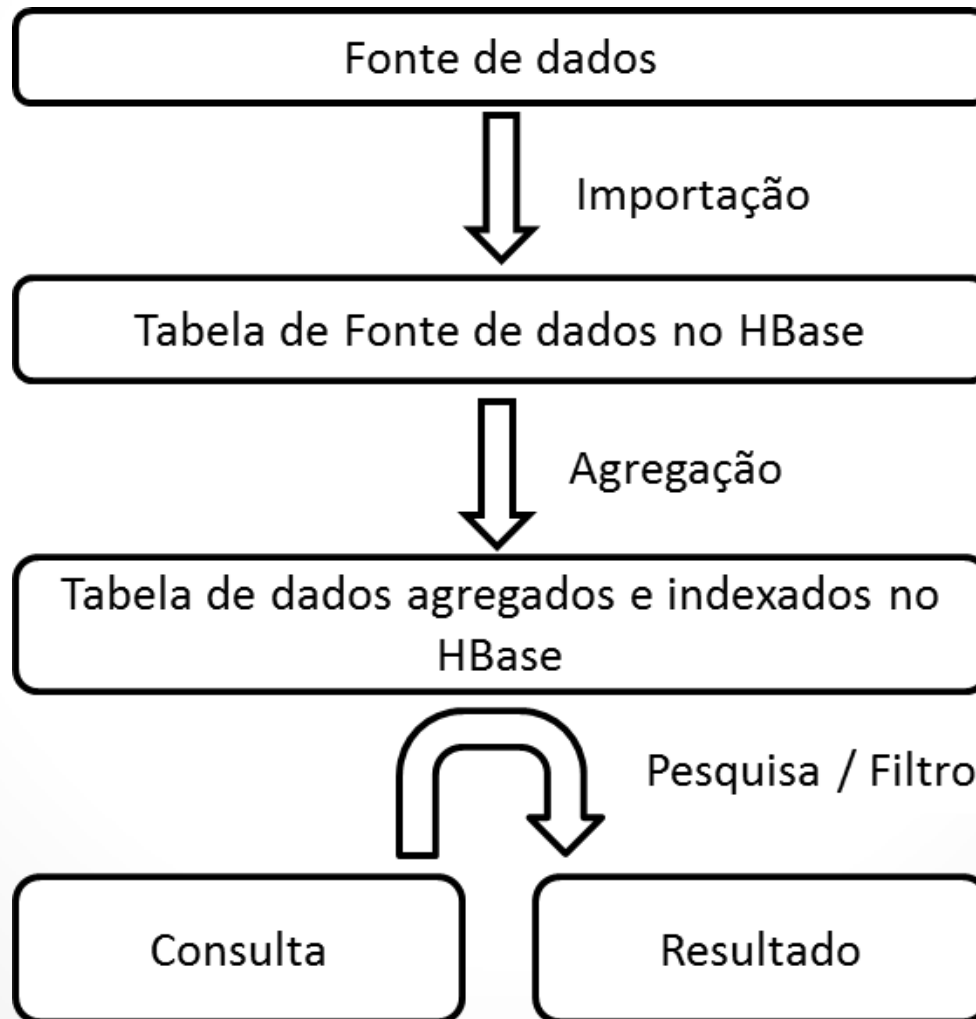
- NoSQL do tipo Super Colunas sobre Hadoop
- Table / Row / Column Family / Column Qualifier / Cell / Version

	Família de colunas – info			
Rowkey	Data	Loja	Distrito	Volume Vendas
Linha A	01/05/2013	Oeiras	Lisboa	5000.00
Linha B	01/05/2013	Cascais	Lisboa	5900.00
Linha C	02/06/2013	Cascais	Lisboa	TS:1372010537=6100.00 TS:1372010530=6150.00

Importância rowkey

- Só a rowkey está indexada
- Só permite ordenação lexicográfica (“alfabética”)
- Boa definição antes
- Se não se usar, *full scan*

Arquitetura



Importação

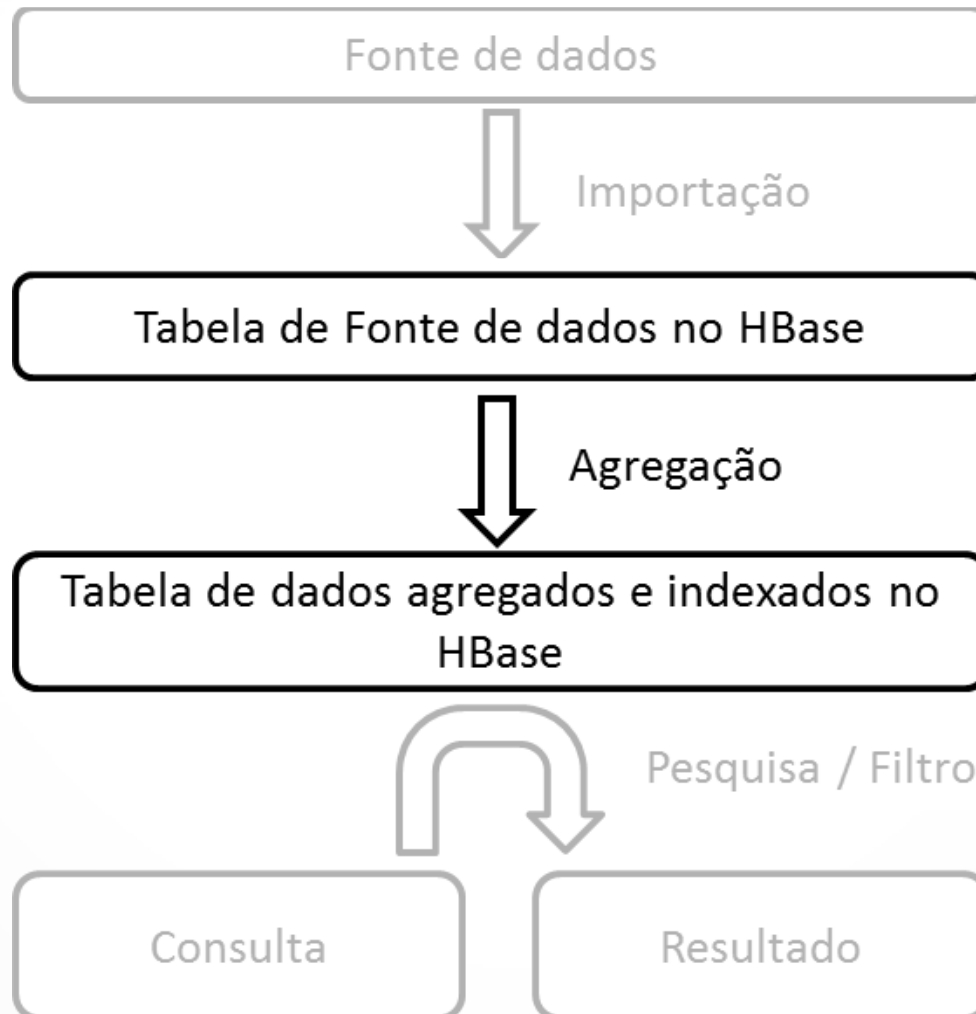
Formatos de Entrada

- **CSV**
- **Multi linha**

Comportamento

- **Concatena**
- **Substitui**

Agregação



Cubo OLAP

- **Dimensão**
- **Medidas**
 - **count** – contagem do número de elementos
 - **sum** – soma do valor dos elementos
 - **max** – o valor máximo dos elementos
 - **min** – o valor mínimo dos elementos
 - **avg** – a média dos valores dos elementos

Row Key da Agregação

- Pesquisas rápidas
- Permitir ordenação
- Permitir escolher dimensões e medidas

Row Key da Agregação

<Nome de dimensão 1> (...) <Nome de dimensão N>
<Nome métrica agregada>
<Tipo agregação da métrica >
<Ordenação do valor da métrica >
<Valor da métrica>
<Valor da dimensão 1> (...) <Valor da dimensão N>

Ordenação Descendente

O valor maior tem de aparecer primeiro na ordenação

Número muito grande – valor

Exemplo:

$V1 = 17$ menor que $V2 = 18$

$999\ 999\ 999\ 999\ 999 - 17 = 99999999999999982 = V1'$

$999\ 999\ 999\ 999\ 999 - 18 = 99999999999999981 = V2'$

$V2'$ menor que $V1'$

Conteúdo da Agregação

```
{  
  "count": <valor>,  
  "max": <valor>,  
  "min": <valor>,  
  "sum": <valor>,  
  "avg": <valor>,  
  "metric": <nome da métrica agregada>  
}
```

Caso Prático

- On-Time Performance
(<http://www.transtats.bts.gov>)
- informações sobre todos os voos comerciais com origem ou destino nos EUA entre 2000 e 2005
- 109 indicadores por registo
- 30 milhões de registos

Caso Prático

- Dimensões: Year e UniqueCarrier
- Medida: DepDelayMinutes

Rowkey
Year
UniqueCarrier
DepDelayMinutes
avg desc
999999999999999981,3950
2000 UA

Caso Prático

- Dimensões: Year e UniqueCarrier
- Medida: DepDelayMinutes

```
scan <tabela>, {LIMIT => X,  
STARTROW=>  
'Year UniqueCarrier DepDelayMinutes avg  
desc`  
ENDROW =>  
'Year UniqueCarrier DepDelayMinutes avg  
desc0`  
}
```

Resultado

Rowkey	Valor
Year UniqueCarrier DepDelayMinutes avg desc 9999999999999981,3950 2000 UA	{ "count":776559, "max":1277.0, "min":0.0, "sum":1.3671339E7, "avg":17.605022928071143 }
Year UniqueCarrier DepDelayMinutes avg desc 9999999999999984,39082 2000 HP	{ "count":219160, "max":729.0, "min":0.0, "sum":3201748.0, "avg":14.60918050739186 }

Interface - Seleção

Cube Report Generator for **movies**

Available Data

cf:review/helpfulness

cf:review/profileName

cf:review/summary

cf:review/text

cf:N#review/time

Columns

Rows

cf:product/productId

cf:review/userId

Values

cf:N#review/score

Limit results to

ex: 50

Order

results by
value

☒ Count

☐ Sum

☐ Avg

☐ Min

☐ Max

Show Value

☐ Count

☒ Sum

☐ Avg

☐ Min

☐ Max

Direction

☒ Top

☐ Bottom

► Generate

Data shown.... complete

Interface - Resultados

Result		
B0007807OY	A3KF4IP2MUS8QQ	48
B0000TB03W	A3FVISMTESSRUL	585
B0000TB04G	A2FUSHGFOEZ8HA	65
B00143XE00	A192KEPM0HW6AC	50
B00005M2IM	A3KF4IP2MUS8QQ	48
B0000TB04G	A2YIHBD7MPNTC9	280
B0000TB04G	A3W1RSMWYYQ1RC	122
630311203X	A3KF4IP2MUS8QQ	48

Map Reduce

```
Job job = new Job(config, jobName);  
Scan scan = new Scan();
```

```
TableMapReduceUtil.initTableMapperJob(  
    Bytes.toBytes(tableName), // input table  
    scan, // scan  
    AggregationMapper.class, // mapper class  
    Text.class, // mapper output key  
    DoubleWritable.class, // mapper output value  
    job);
```

```
TableMapReduceUtil.initTableReducerJob(  
    Bytes.toBytes(aggregationTable), // output table  
    AggregationReducer.class, // reducer class  
    job);
```

```
job.waitForCompletion(true);
```

Map

```
public void map(ImmutableBytesWritable row, Result value,  
    Context context) {  
  
    DoubleWritable doubleWritable = new DoubleWritable(0);  
  
    byte[] bVal = value.getValue(columnRef.getCfAsByteArray(),  
        columnRef.getCnAsByteArray());  
    doubleWritable.set(Bytes.toDouble(bVal));  
  
    keyText.set(rowKey.toString());  
    context.write(keyText, doubleWritable);  
}
```

Reduce

```
public void reduce(Text key, Iterable<DoubleWritable> values,  
    Context context) {  
  
    AggregatorValues outValues = new AggregatorValues(sMetrics[0]);  
  
    for (DoubleWritable val : values) {  
        outValues.addValue(val.get());  
    }  
  
    byte[] outValuesBytes = Bytes.toBytes(outValues.toJson());  
    String rowKey = key.toString();  
    Put put = new Put(Bytes.toBytes(rowKey));  
    put.add(AGG_CF, metricsColumnNameByte[0], outValuesBytes);  
    context.write(null, put);  
  
    writeAggregateData(context, rowKey, "sum", "asc",  
        sMetrics[0], outValuesBytes, outValues.getSum())  
}
```

Obrigado

joel.alexandre@gmail.com