Big Data

Pós Disruptiva

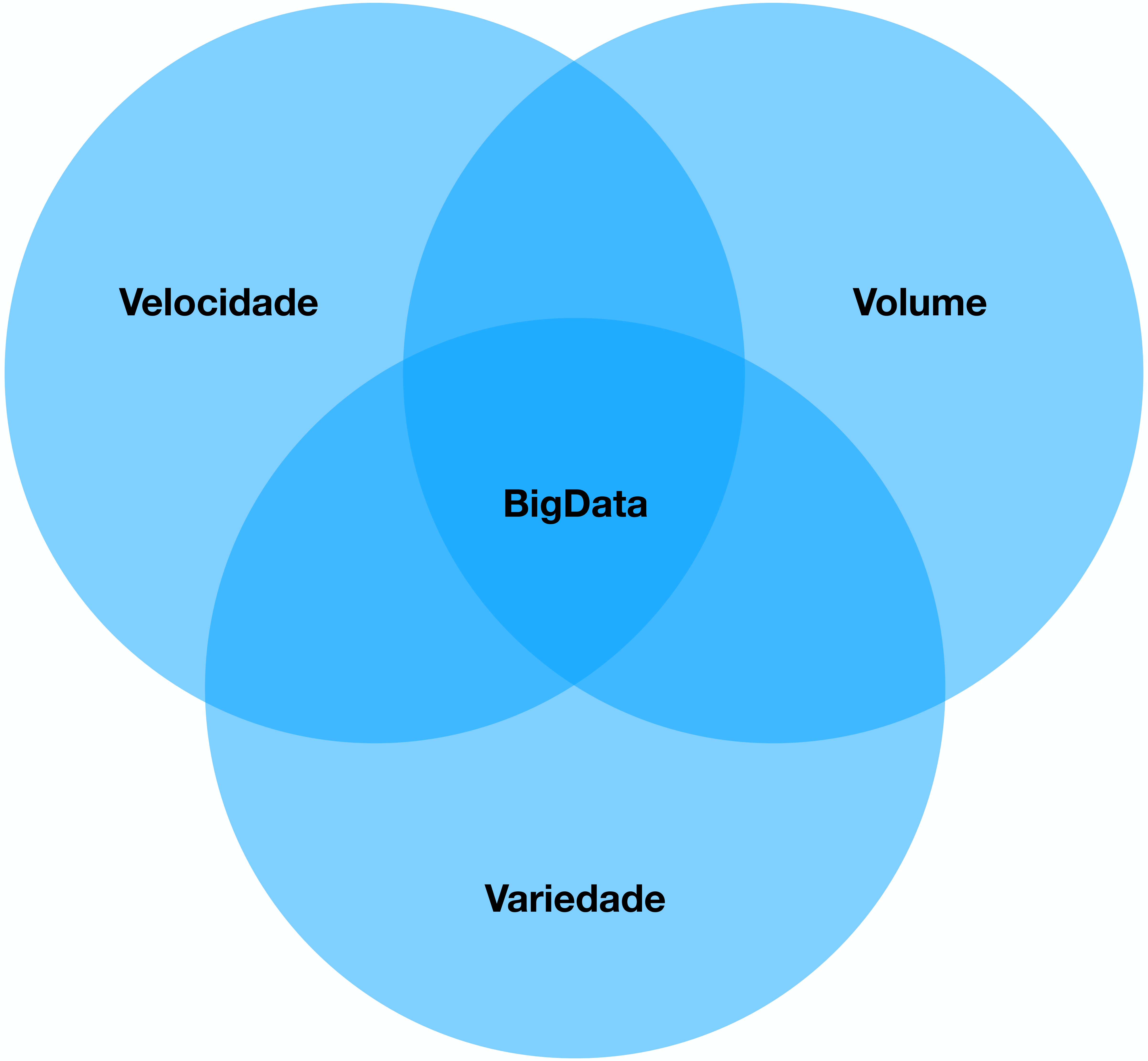
AULA 1

Revisão
MR HO
Database

Modulo 1

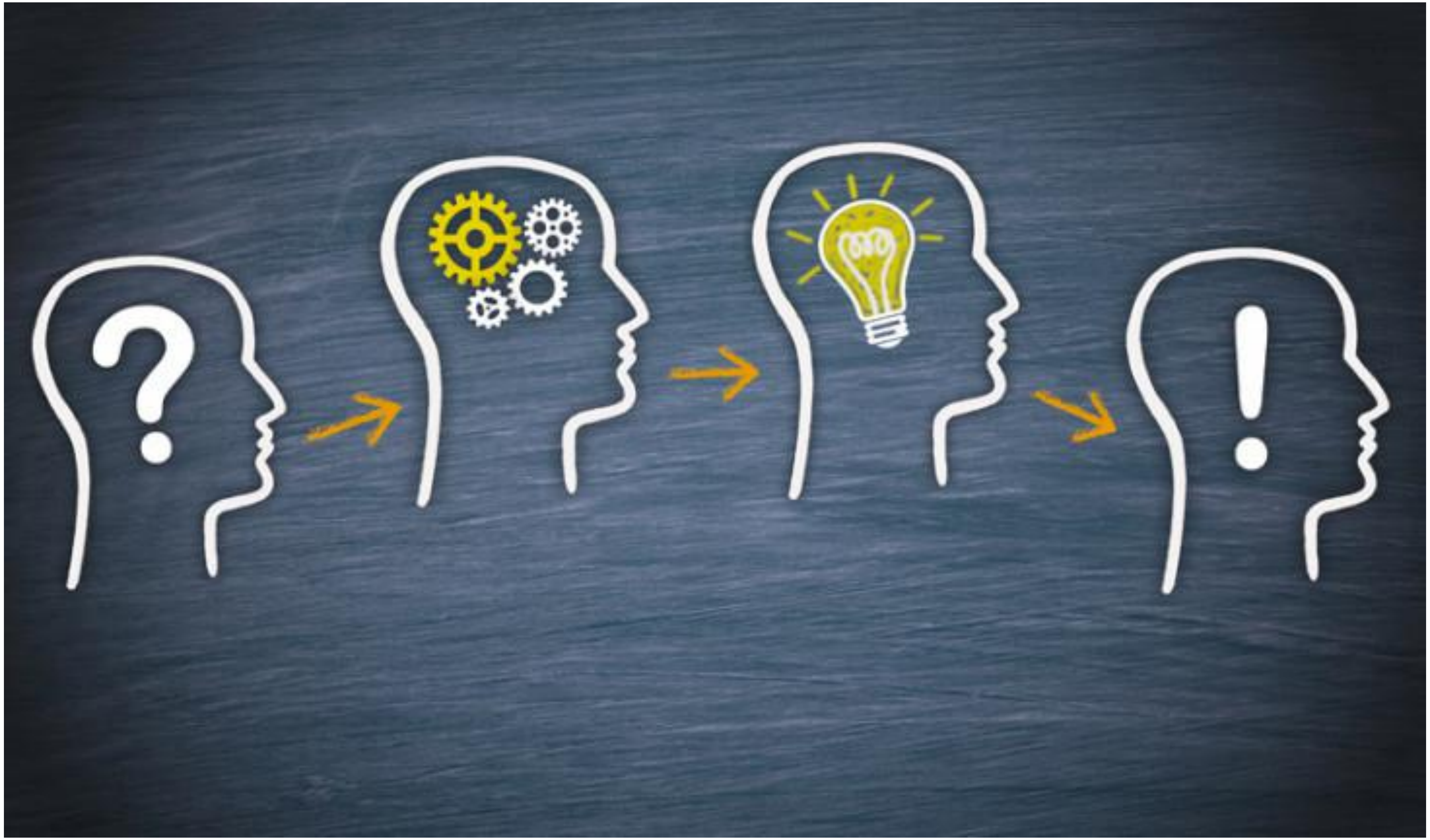
Revisão

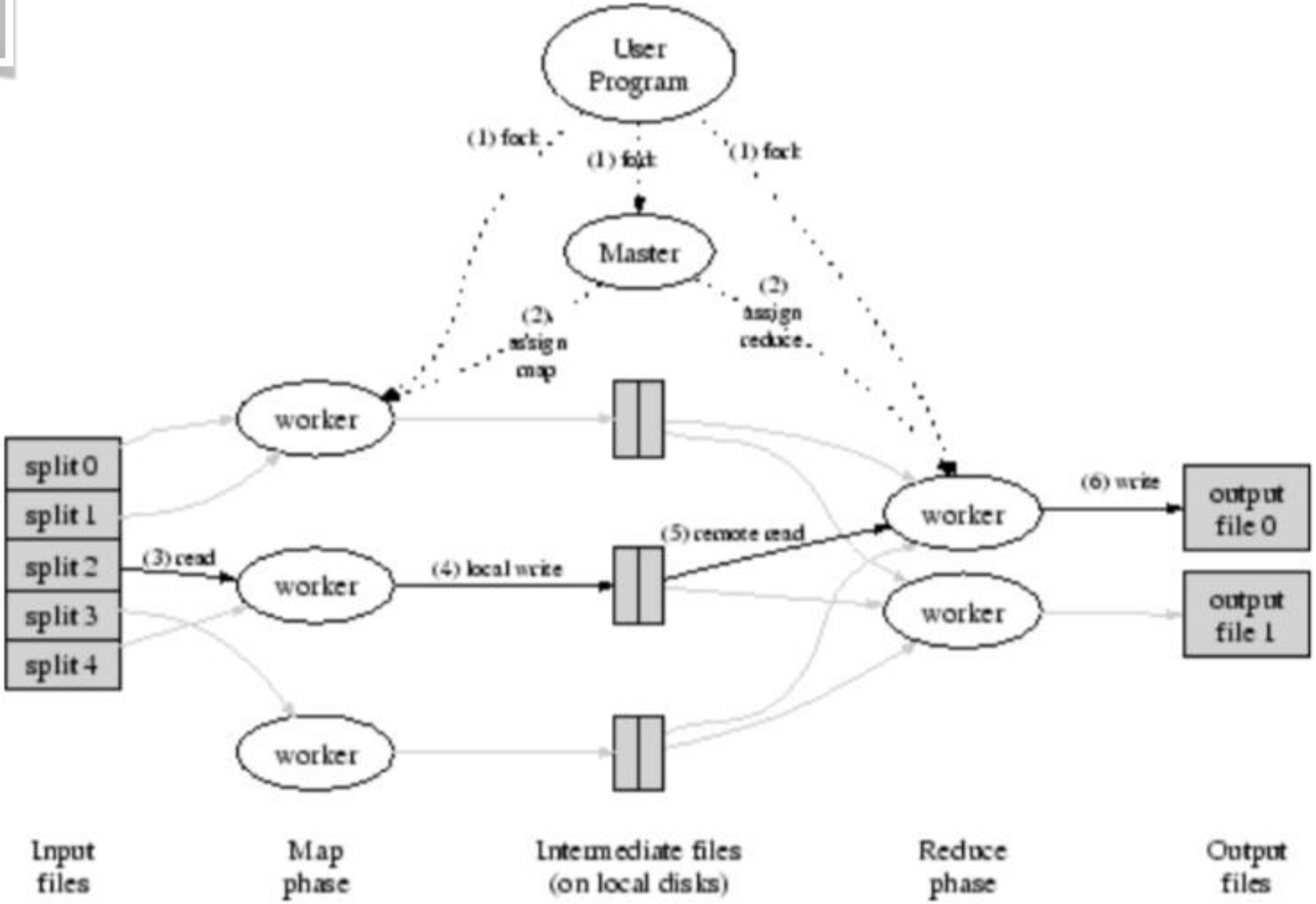
Bigdata é um problema, não uma solução!

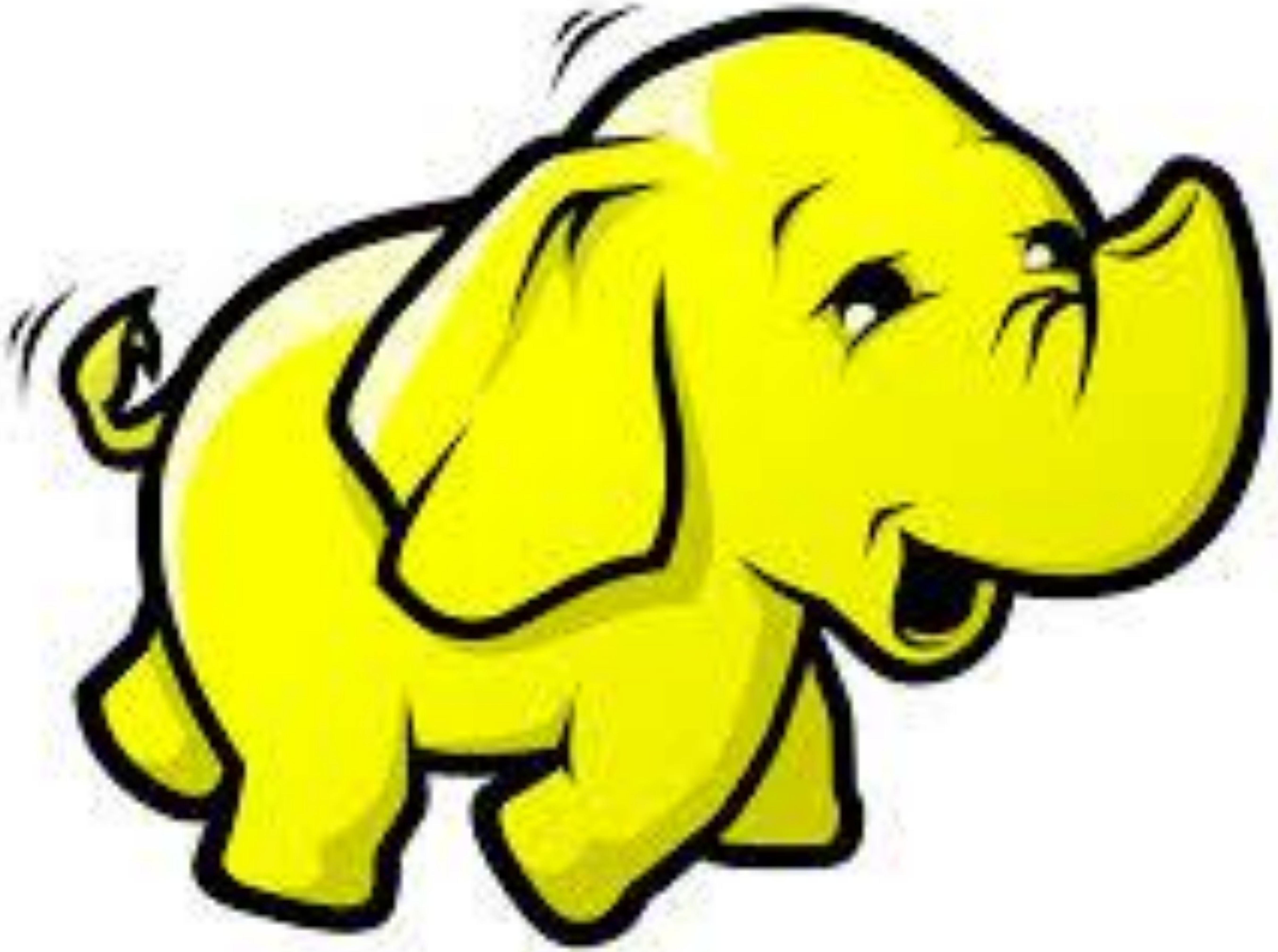


KEEP IT

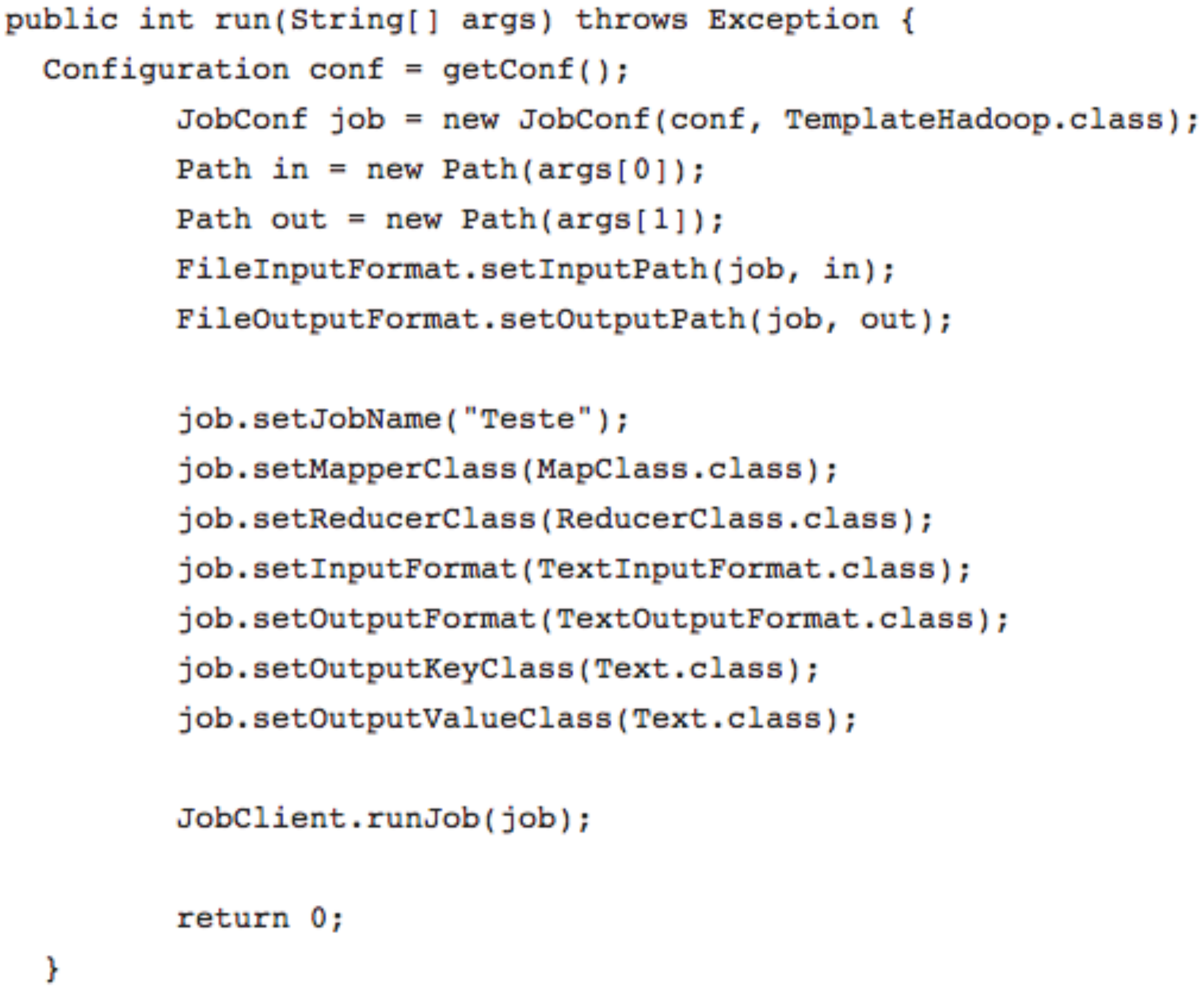
SIMPLE




User Program
(1) fork
(1) fork
(1) fork
Master
(2) assign map
(2) assign reduce
worker
worker
worker
split 0
split 1
split 2
split 3
split 4
(3) read
(4) local write
(5) remote read
worker
worker
(6) write
output file 0
output file 1
Input files
Map phase
Intermediate files (on local disks)
Reduce phase
Output files

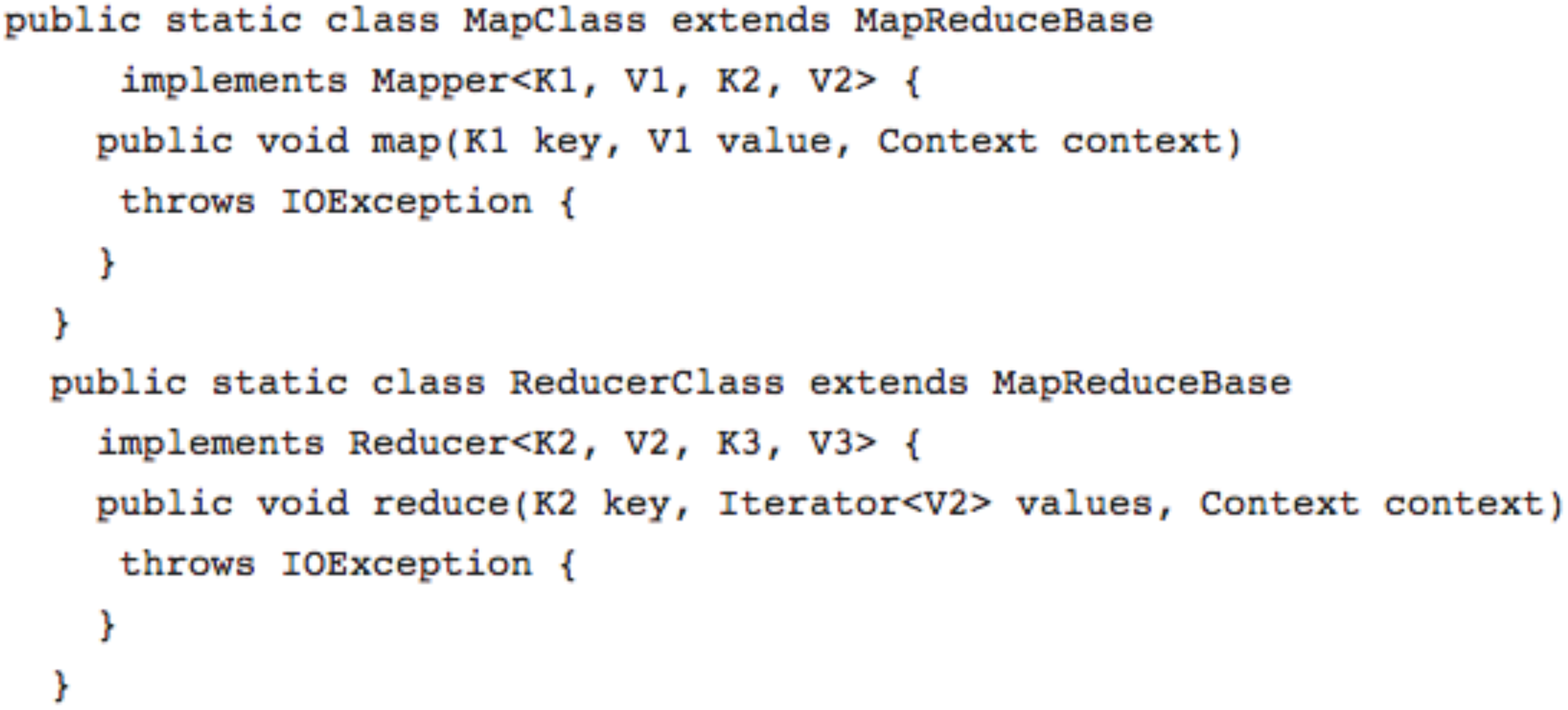


THE CHALLENGE



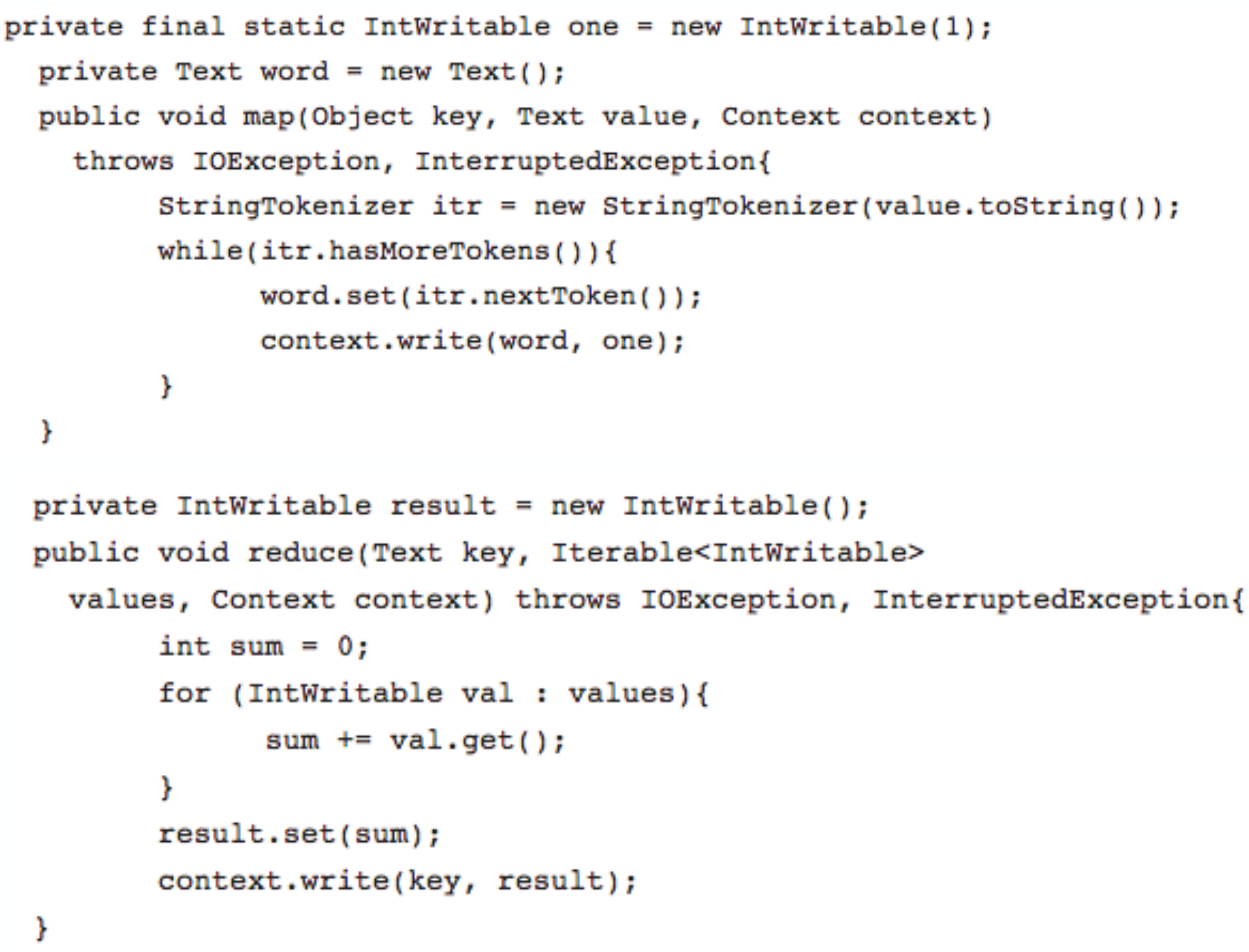
```
public int run(String[] args) throws Exception {
  Configuration conf = getConf();
         JobConf job = new JobConf(conf, TemplateHadoop.class);
         Path in = new Path(args[0]);
         Path out = new Path(args[1]);
         FileInputFormat.setInputPath(job, in);
         FileOutputFormat.setOutputPath(job, out);

         job.setJobName("Teste");
         job.setMapperClass(MapClass.class);
         job.setReducerClass(ReducerClass.class);
         job.setInputFormat(TextInputFormat.class);
         job.setOutputFormat(TextOutputFormat.class);
         job.setOutputKeyClass(Text.class);
         job.setOutputValueClass(Text.class);

         JobClient.runJob(job);

         return 0;
  }
```

THE CHALLENGE



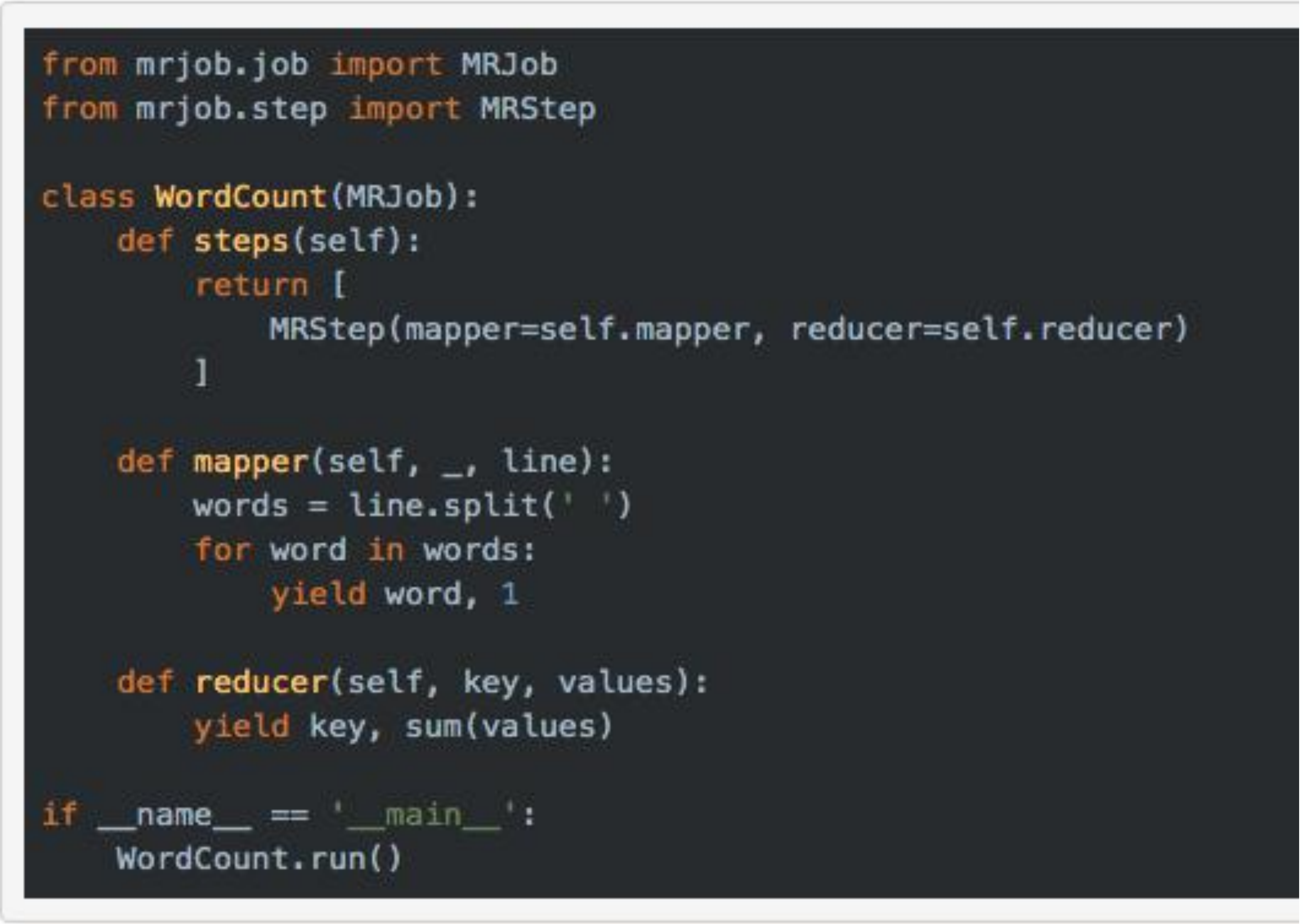
```
public static class MapClass extends MapReduceBase
     implements Mapper<K1, V1, K2, V2> {
    public void map(K1 key, V1 value, Context context)
     throws IOException {
    }
  }
  public static class ReducerClass extends MapReduceBase
    implements Reducer<K2, V2, K3, V3> {
    public void reduce(K2 key, Iterator<V2> values, Context context)
     throws IOException {
    }
  }
```

THE CHALLENGE



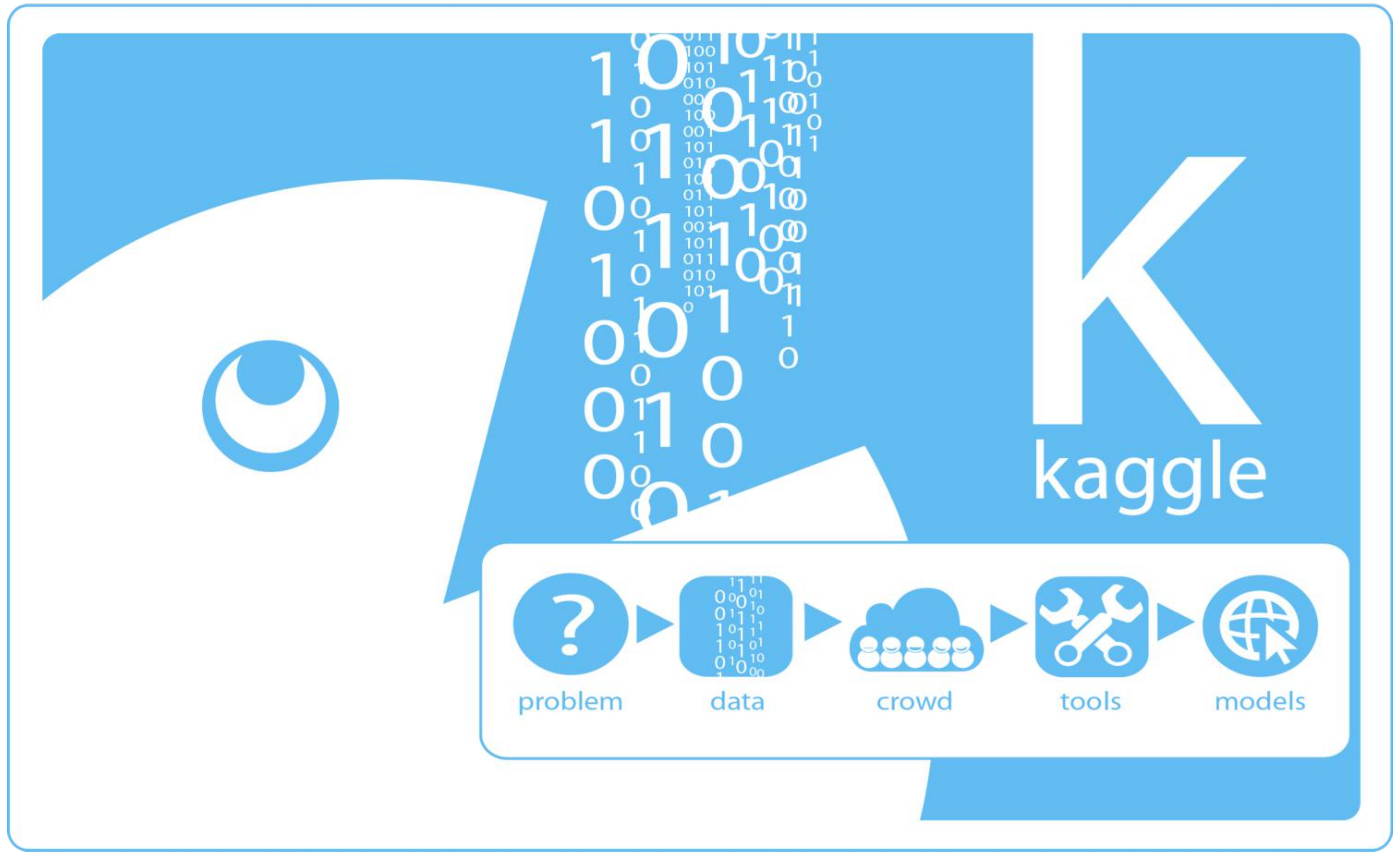
```
private final static IntWritable one = new IntWritable(1);
  private Text word = new Text();
  public void map(Object key, Text value, Context context)
    throws IOException, InterruptedException{
         StringTokenizer itr = new StringTokenizer(value.toString());
         while(itr.hasMoreTokens()){
               word.set(itr.nextToken());
               context.write(word, one);
         }
  }

  private IntWritable result = new IntWritable();
  public void reduce(Text key, Iterable<IntWritable>
    values, Context context) throws IOException, InterruptedException{
         int sum = 0;
         for (IntWritable val : values){
               sum += val.get();
         }
         result.set(sum);
         context.write(key, result);
  }
```

THE CHALLENGE



```python
from mrjob.job import MRJob
from mrjob.step import MRStep

class WordCount(MRJob):
    def steps(self):
        return [
            MRStep(mapper=self.mapper, reducer=self.reducer)
        ]

    def mapper(self, _, line):
        words = line.split(' ')
        for word in words:
            yield word, 1

    def reducer(self, key, values):
        yield key, sum(values)

if __name__ == '__main__':
    WordCount.run()
```

THE CHALLENGE

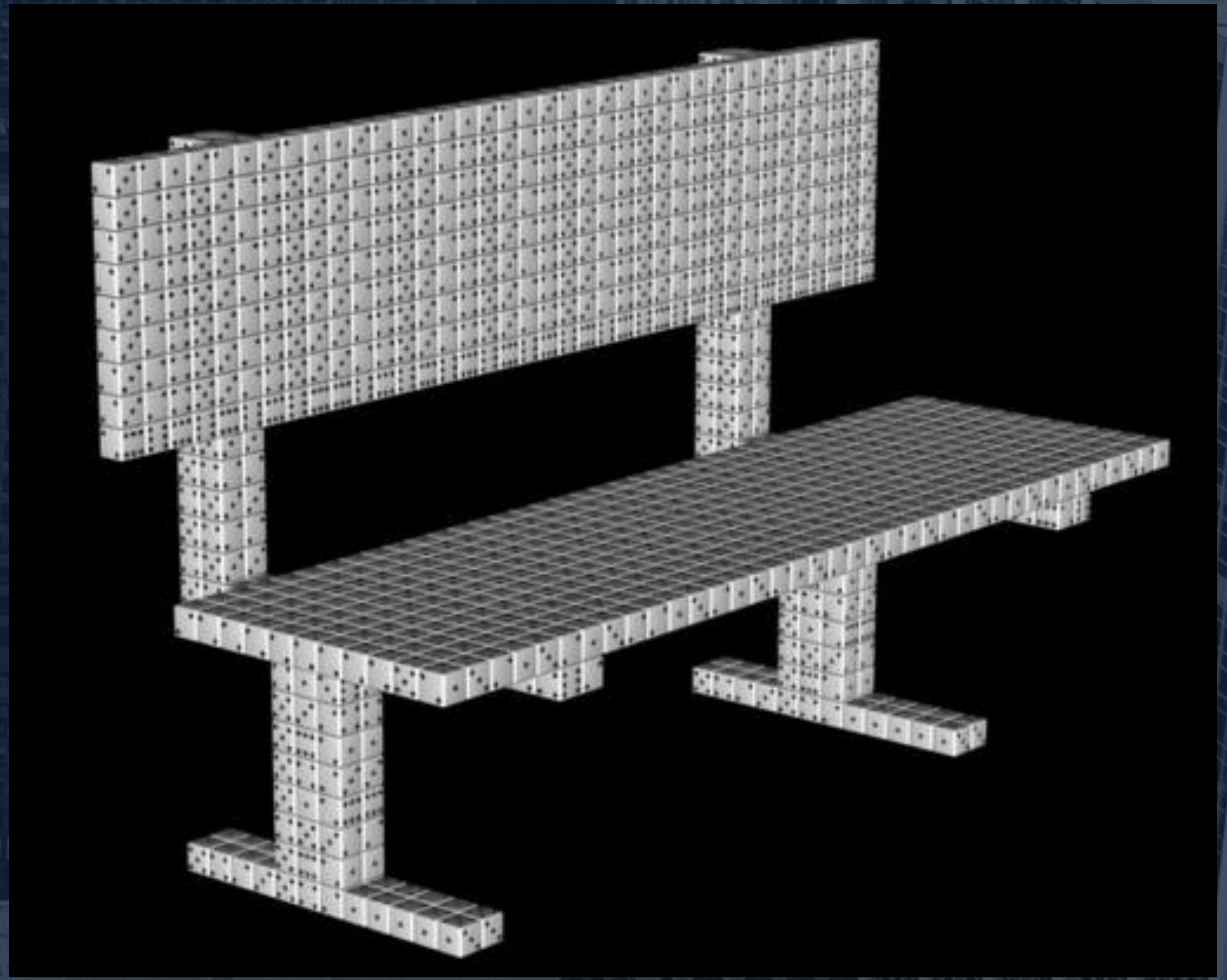

k
kaggle
problem
data
crowd
tools
models

HADOOP
BUSINESS
INTERNET
ANALYSIS
SECURITY
STORAGE
DATA
BIG DATA
CLOUD
ANALYTICS
SYSTEMS
LEARNING
COMPUTING
PROJECTS
ENTERPRISE
MANAGE
VOLUME
OPEN SOURCE
SOLVED
SOFTWARE
TECHNOLOGIES
SOCIAL
MEDIA
TREND
DATABASE
COMPLEX
RISK
NETWORKS
SHARE

docker

Modulo 2

Banco de dados



NoSQL | SQL

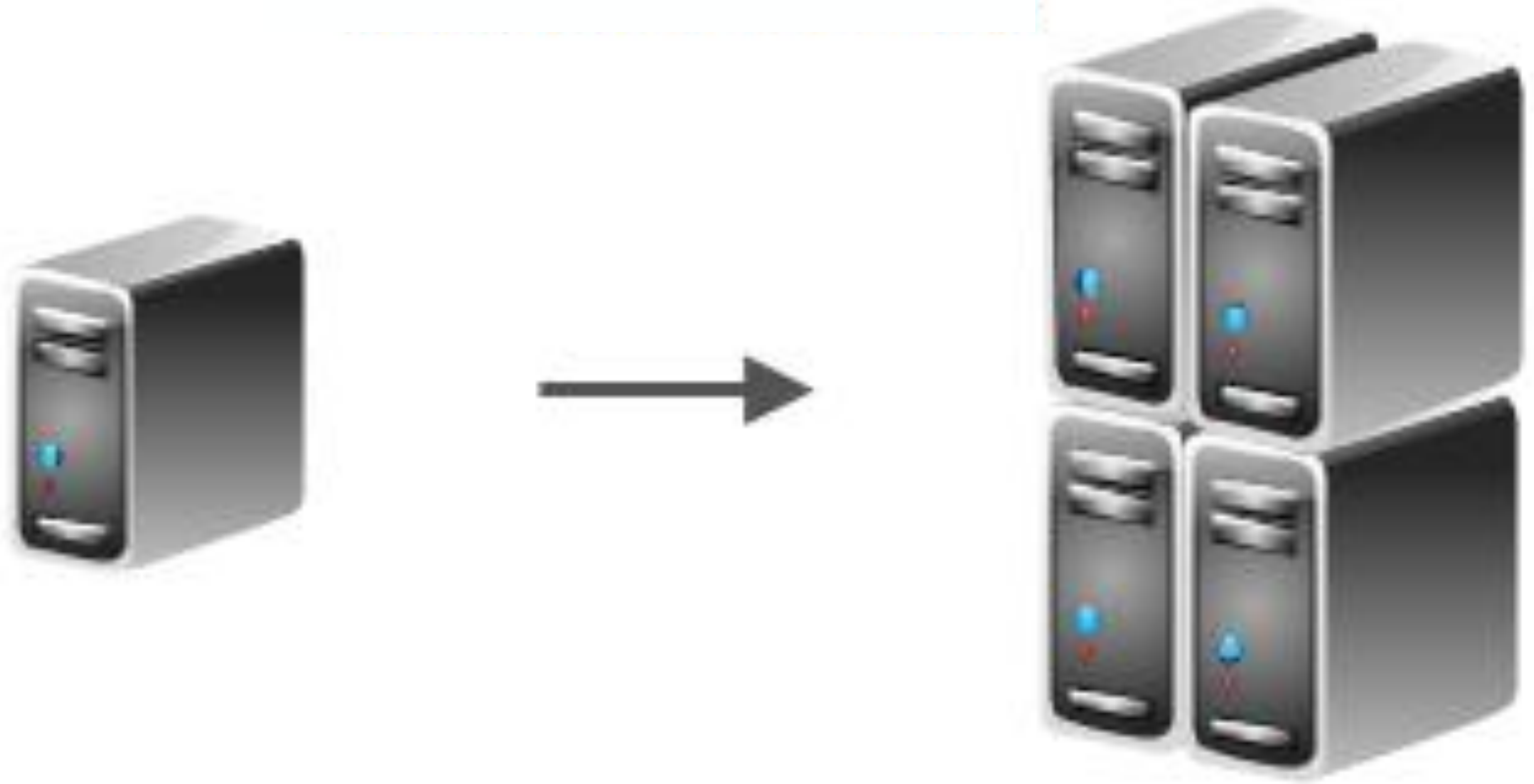
mongoDB
cassandra
HIVE
APACHE
HBASE
MEMCACHED
redis

SQL
MySQL
PostgreSQL
ORACLE
DATABASE

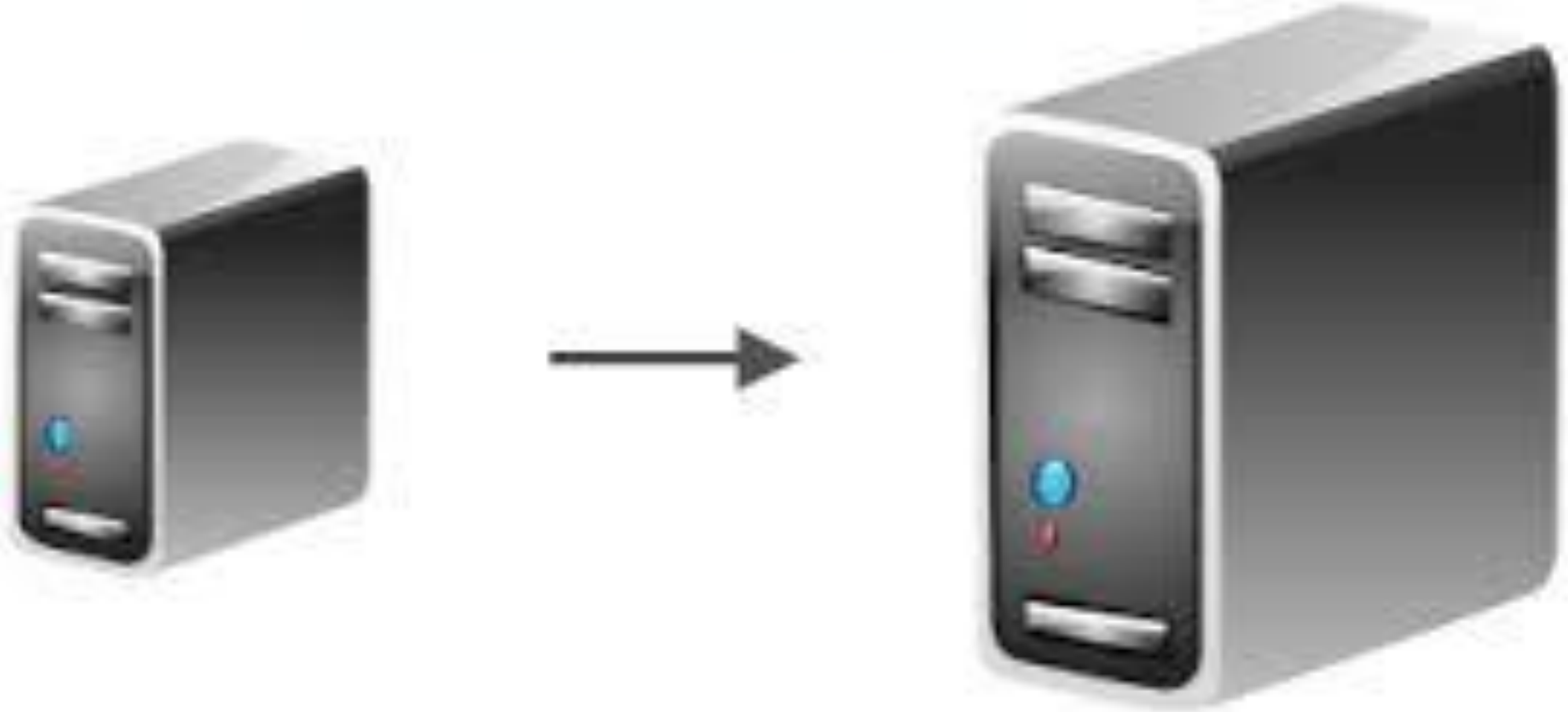
NoSQL

SQL

NoSQL



SQL



NoSQL

cassandra

```
CREATE KEYSPACE MyKeySpace
  WITH REPLICATION = { 'class' :
'SimpleStrategy',
'replication_factor' : 3 };

USE MyKeySpace;

CREATE COLUMNFAMILY MyColumns (id
text, Last text, First text, PRIMARY
KEY(id));
```

SQL

MySQL®

```
UPDATE users SET status = 'C'
WHERE age > 25
```

```
SELECT * FROM users
```

```
INSERT INTO users (user_id, age, status)
VALUES ('bcd001', 45, 'A')
```

cassandra

ebay

NETFLIX

NoSQL

mongoDB

```
db.users.insert({
  user_id: 'bcd001',
  age: 45,
  status: 'A'
})
```

```
db.users.update(
  { age: { $gt: 25 } },
  { $set: { status: 'C' } },
  { multi: true }
)
```

SQL

MySQL

```
UPDATE users SET status = 'C'
WHERE age > 25
```

```
SELECT * FROM users
```

```
INSERT INTO users (user_id, age, status)
VALUES ('bcd001', 45, 'A')
```

mongoDB®

ebay

McAfee®
An Intel Company

intuit.

Adobe

NoSQL

APACHE HBASE

Row Key	Timestamp	Customer		Sales	
Customer Id		Name	City	Product	Amount
101	T1	Suresh	Hyderabad		300
101	T2	Suresh Reddy		Books	
102	T1	Lavya Gavshinde	Indore	Fan	600
102	T2	Lavya			570
102	T3		Bhopal		
103	T1	Anurag	Raipur	Laptop	40000
104	T1	Deepesh	Delhi	Bike	32000

SQL

MySQL®

Row Key	Timestamp	Customer		Sales	
Customer Id		Name	City	Product	Amount
101	T1	Suresh	Hyderabad		300
101	T2	Suresh Reddy		Books	
102	T1	Lavya Gavshinde	Indore	Fan	600
102	T2	Lavya			570
102	T3		Bhopal		
103	T1	Anurag	Raipur	Laptop	40000
104	T1	Deepesh	Delhi	Bike	32000

APACHE HBASE

Adobe

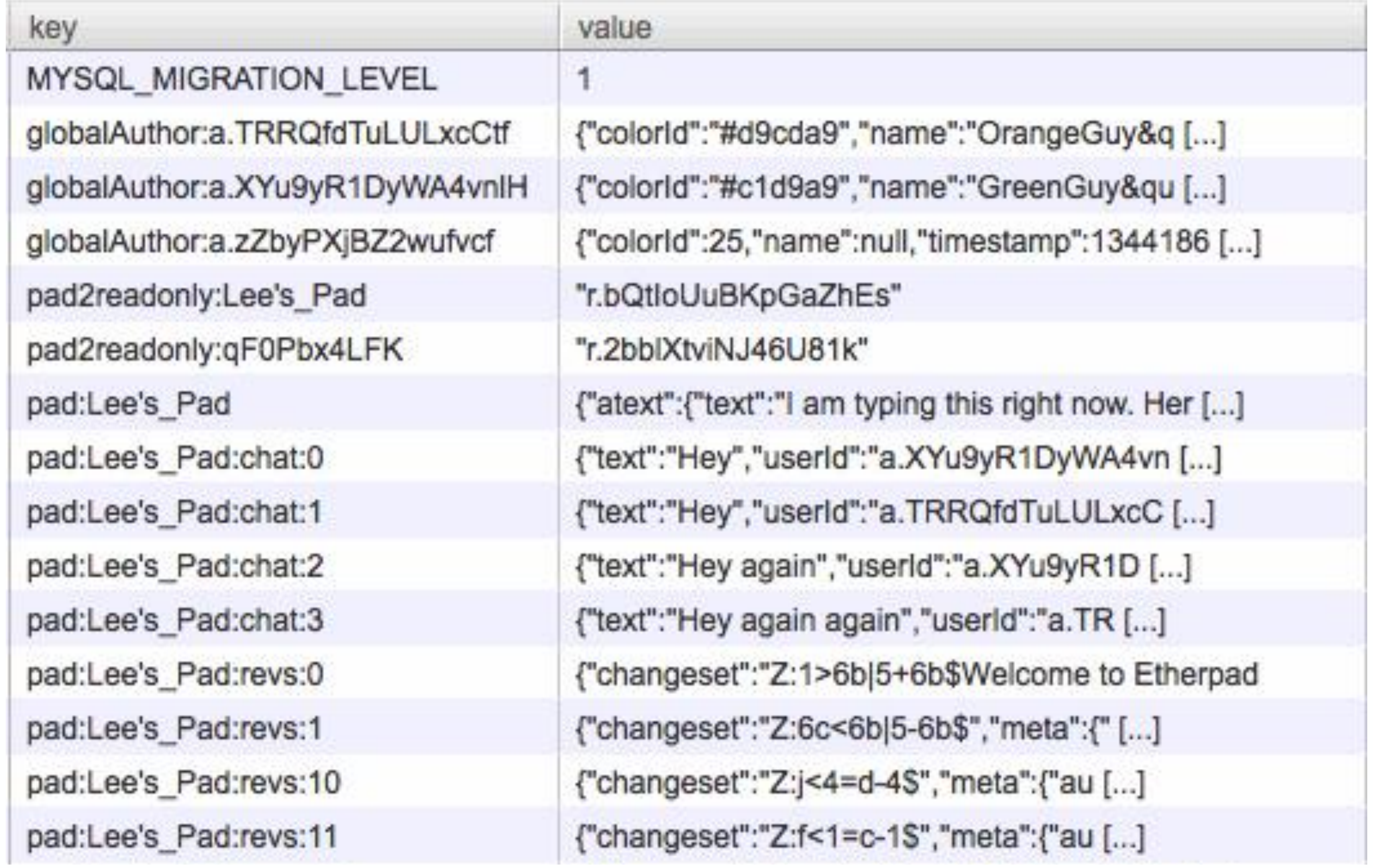
salesforce

facebook

Spotify®

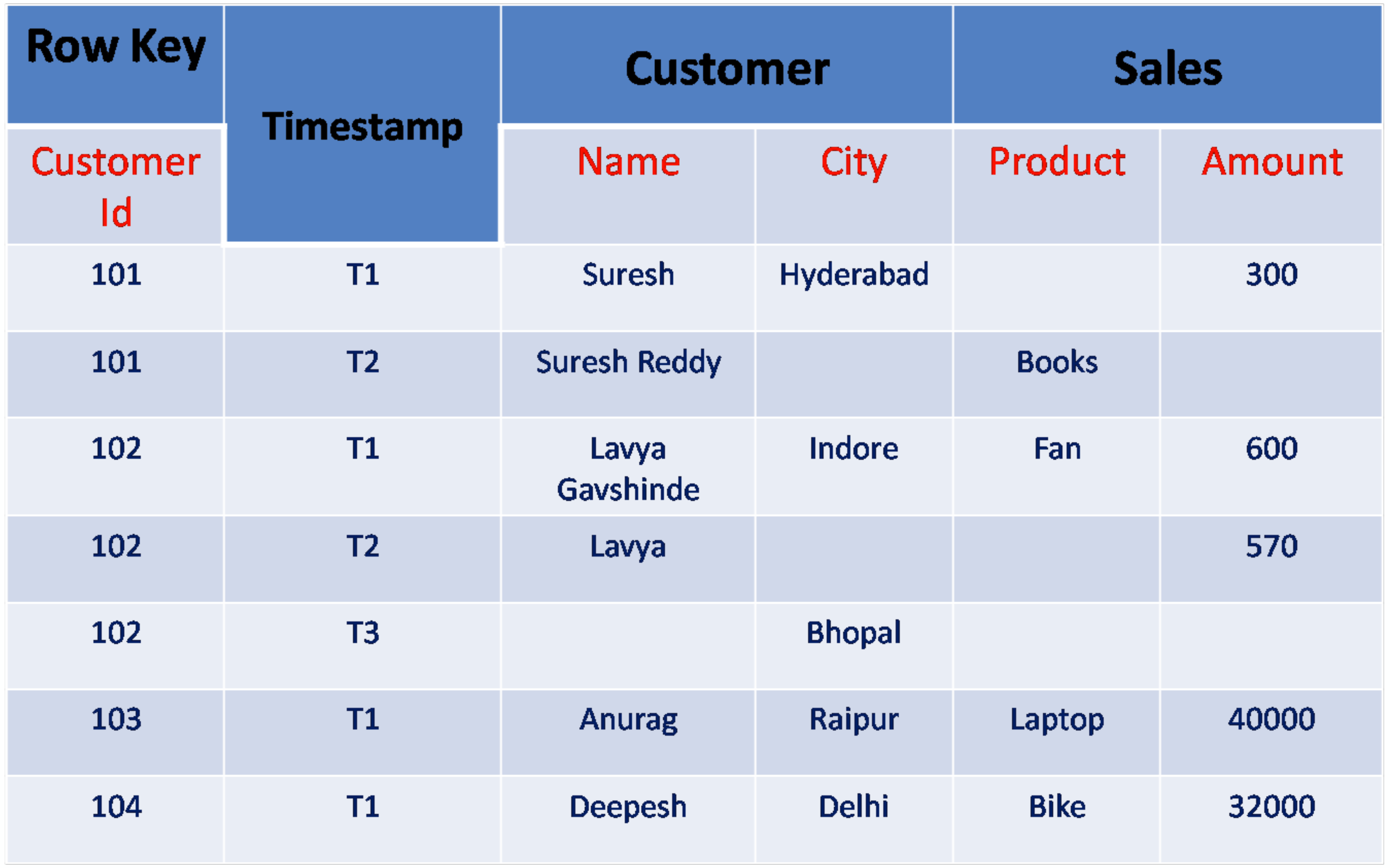
NETFLIX

NoSQL



key	value
MYSQL_MIGRATION_LEVEL	1
globalAuthor:a.TRRQfdTuLULxcCtf	{"colorId":"#d9cda9","name":"OrangeGuy&q [...]
globalAuthor:a.XYu9yR1DyWA4vnlH	{"colorId":"#c1d9a9","name":"GreenGuy&qu [...]
globalAuthor:a.zZbyPXjBZ2wufvcf	{"colorId":25,"name":null,"timestamp":1344186 [...]
pad2readonly:Lee's_Pad	"r.bQtloUuBKpGaZhEs"
pad2readonly:qF0Pbx4LFK	"r.2bblXtviNJ46U81k"
pad:Lee's_Pad	{"atext":{"text":"I am typing this right now. Her [...]
pad:Lee's_Pad:chat:0	{"text":"Hey","userId":"a.XYu9yR1DyWA4vn [...]
pad:Lee's_Pad:chat:1	{"text":"Hey","userId":"a.TRRQfdTuLULxcC [...]
pad:Lee's_Pad:chat:2	{"text":"Hey again","userId":"a.XYu9yR1D [...]
pad:Lee's_Pad:chat:3	{"text":"Hey again again","userId":"a.TR [...]
pad:Lee's_Pad:revs:0	{"changeset":"Z:1>6b|5+6b$Welcome to Etherpad
pad:Lee's_Pad:revs:1	{"changeset":"Z:6c<6b|5-6b$","meta":{" [...]
pad:Lee's_Pad:revs:10	{"changeset":"Z:j<4=d-4$","meta":{"au [...]
pad:Lee's_Pad:revs:11	{"changeset":"Z:f<1=c-1$","meta":{"au [...]

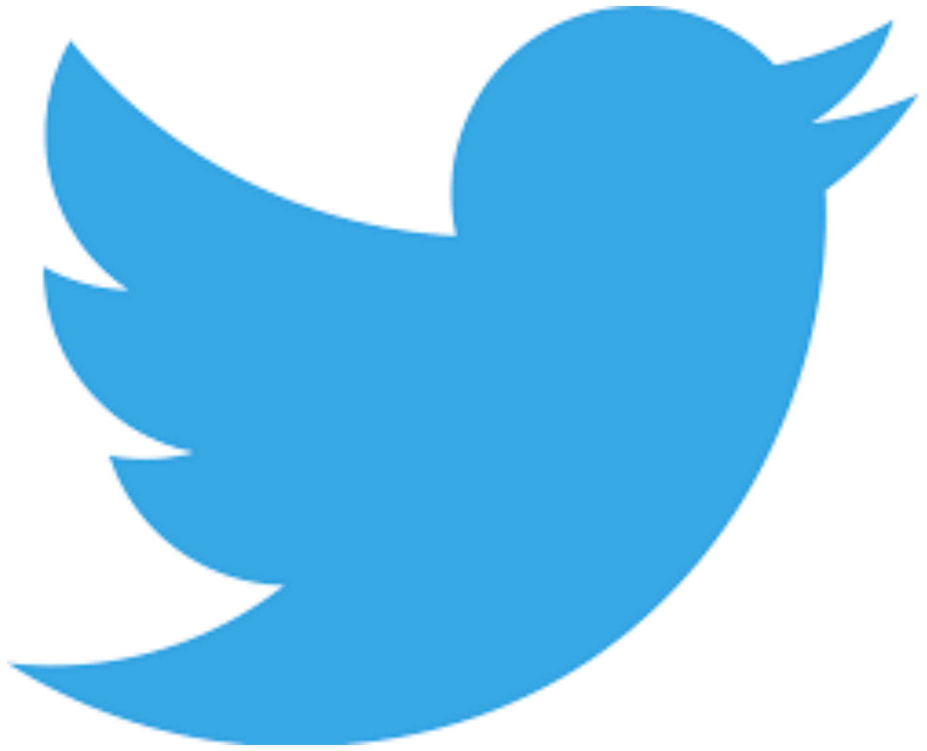
SQL



Row Key	Timestamp	Customer		Sales	
Customer Id		Name	City	Product	Amount
101	T1	Suresh	Hyderabad		300
101	T2	Suresh Reddy		Books	
102	T1	Lavya Gavshinde	Indore	Fan	600
102	T2	Lavya			570
102	T3		Bhopal		
103	T1	Anurag	Raipur	Laptop	40000
104	T1	Deepesh	Delhi	Bike	32000

redis

stackoverflow



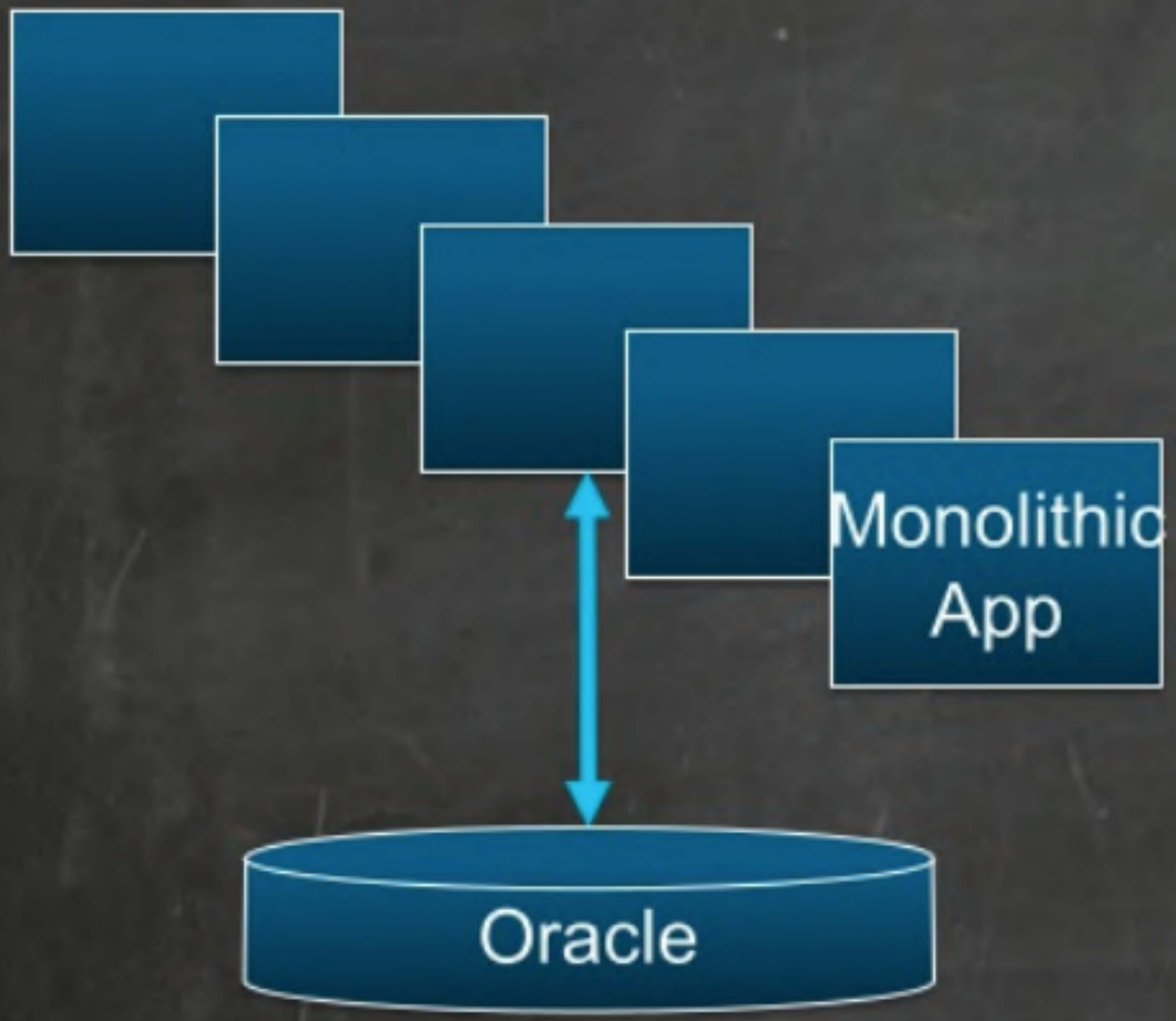
NoSQL

Celtics
Great Checking. Legendary
Boston Celtics
CAVS 23
THOMAS 4

SQL

Just Pampers.
Pampers
Cruisers

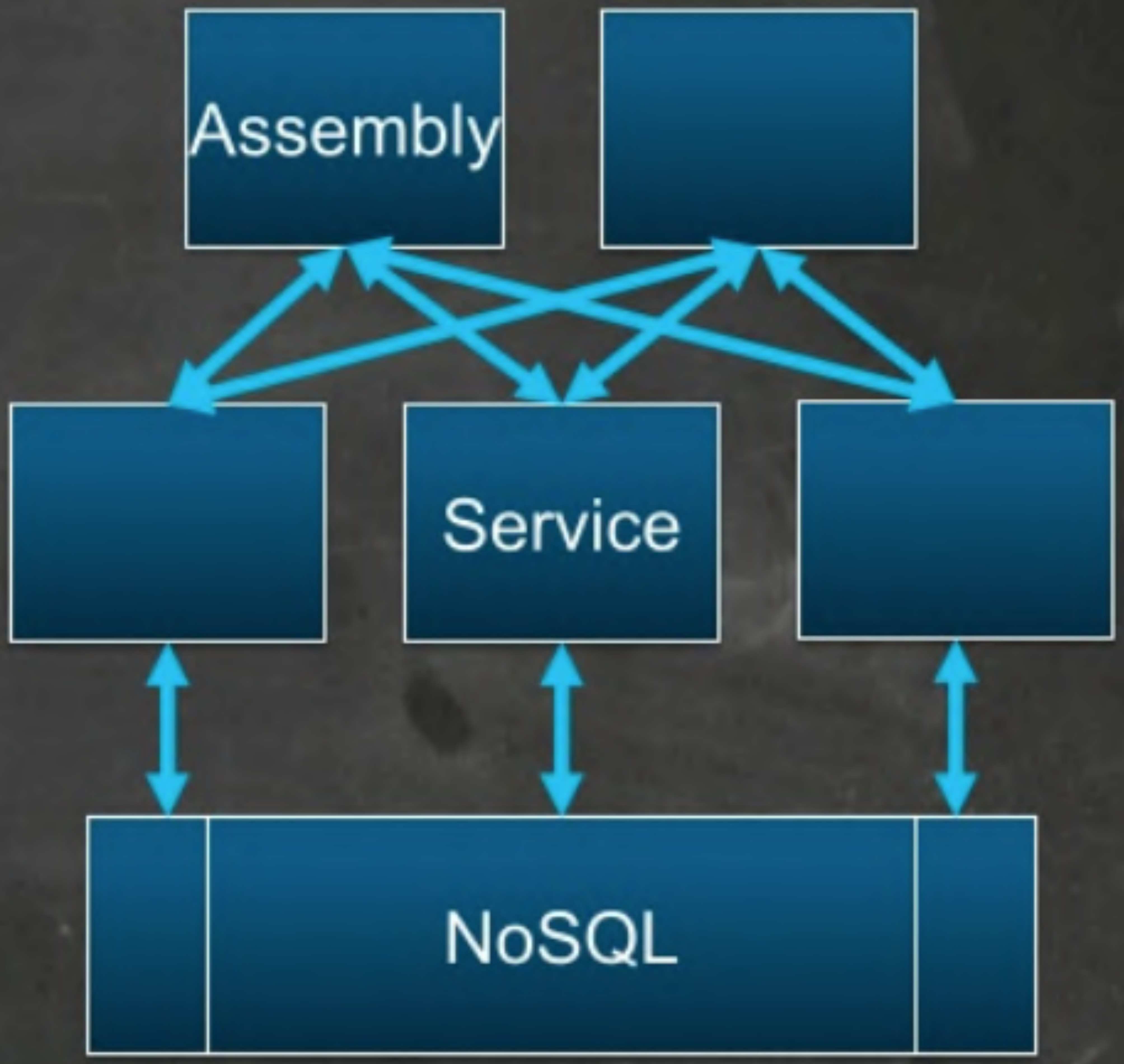
OLD



Monolithic App

Oracle

NEW

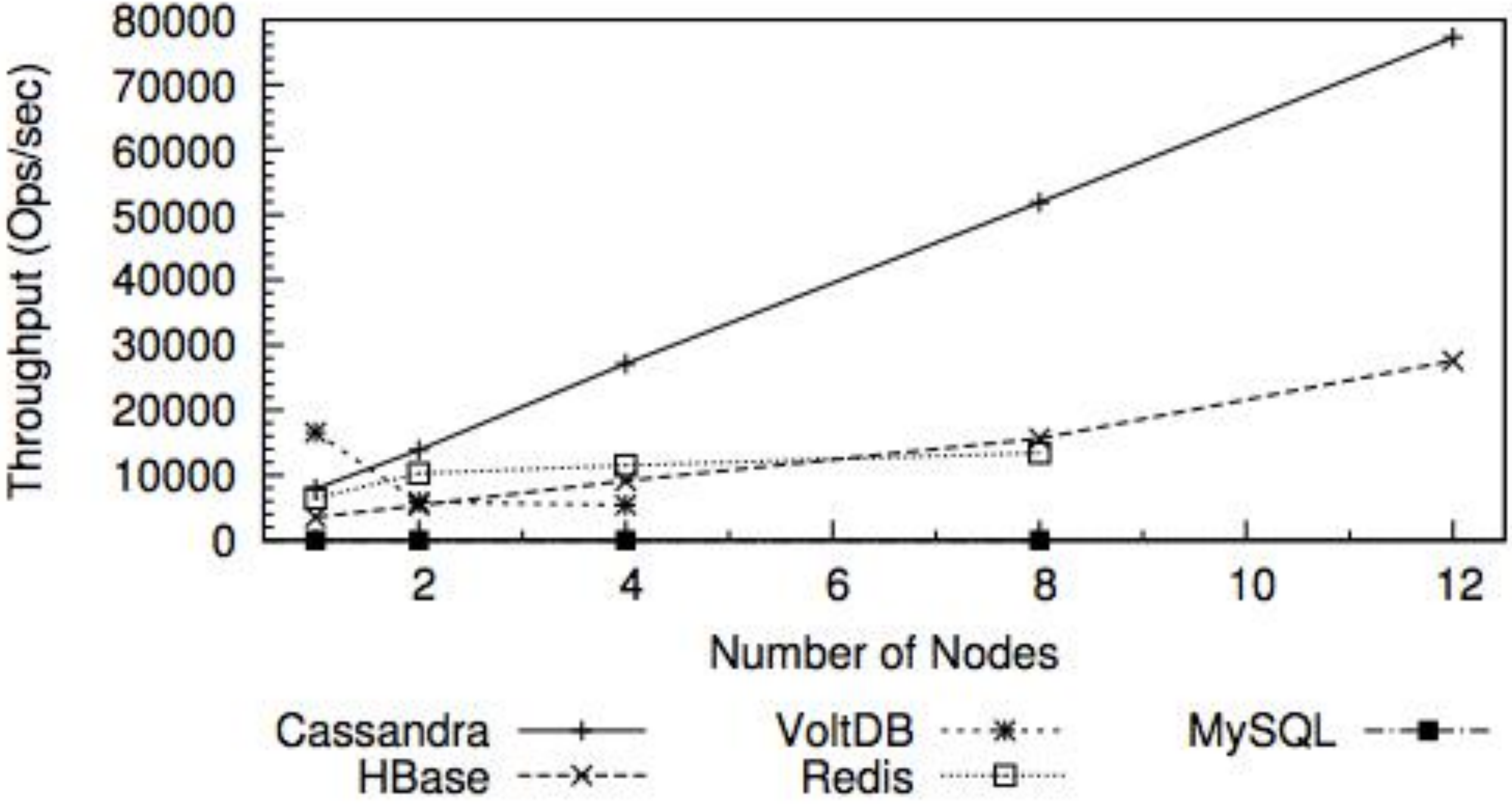


Assembly

Service

NoSQL

Throughput for workload Read/Write



Por hoje é tudo

Até a próxima aula ;)