法律声明

□ 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

□ 课程详情请咨询

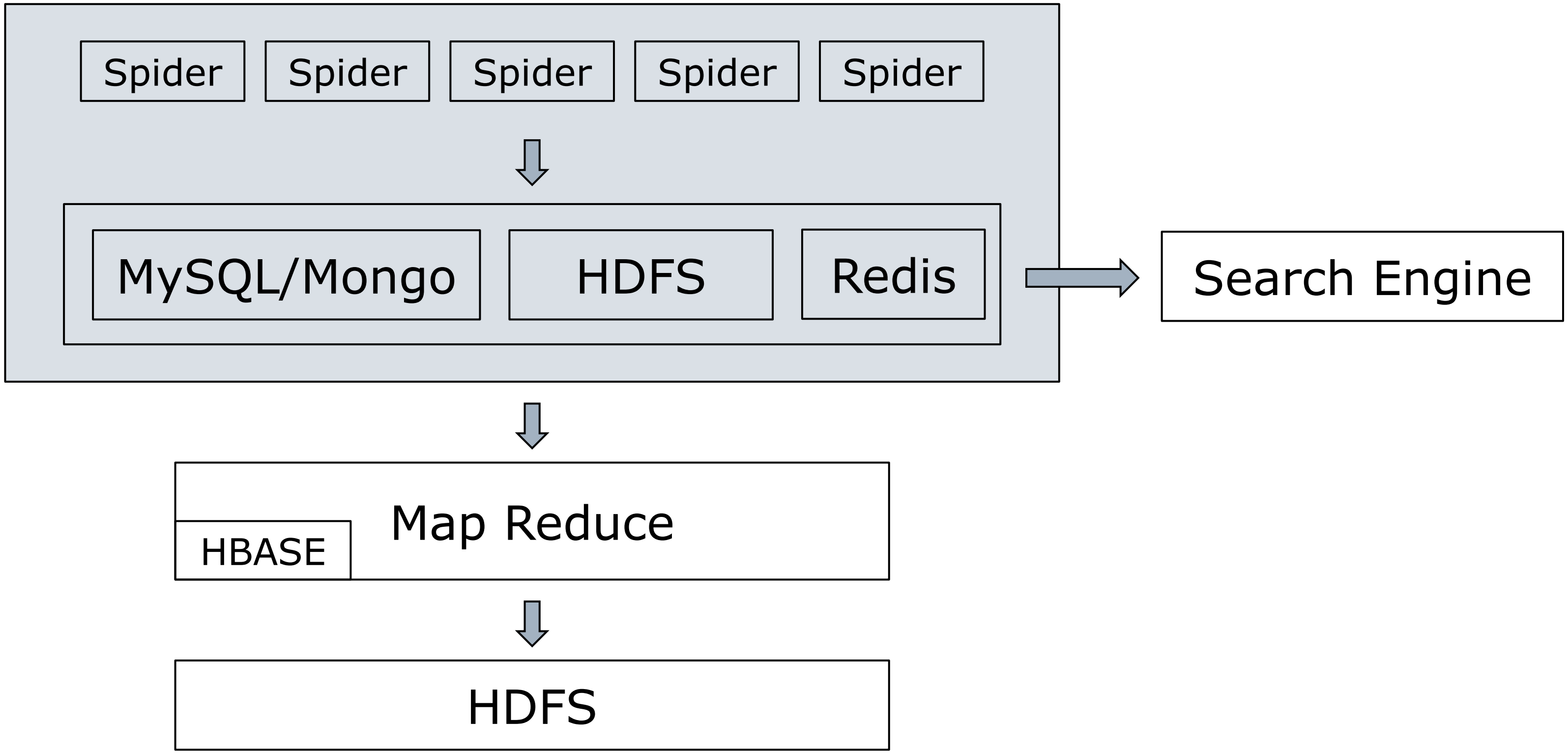
- ■ 微信公众号：大数据分析挖掘
- ■ 新浪微博：ChinaHadoop

分布式爬虫

大纲

- 一个简单的分布式爬虫
- 分布式存储
- 分布式数据库及缓存
- 完整的分布式爬虫

分布式爬虫系统


Spider
Spider
Spider
Spider
Spider
MySQL/Mongo
HDFS
Redis
Search Engine
Map Reduce
HBASE
HDFS

一个简单的分布式爬虫

分布式爬虫的重要性

- 解决目标地址对IP访问频率的限制
- 利用更高的带宽，提高下载速度
- 大规模系统的分布式存储和备份
- 数据的扩展能力

将多进程爬虫部署到多台主机上

- 将数据库地址配置到统一的服务器上

- 数据库设置仅允许特定IP来源的访问请求

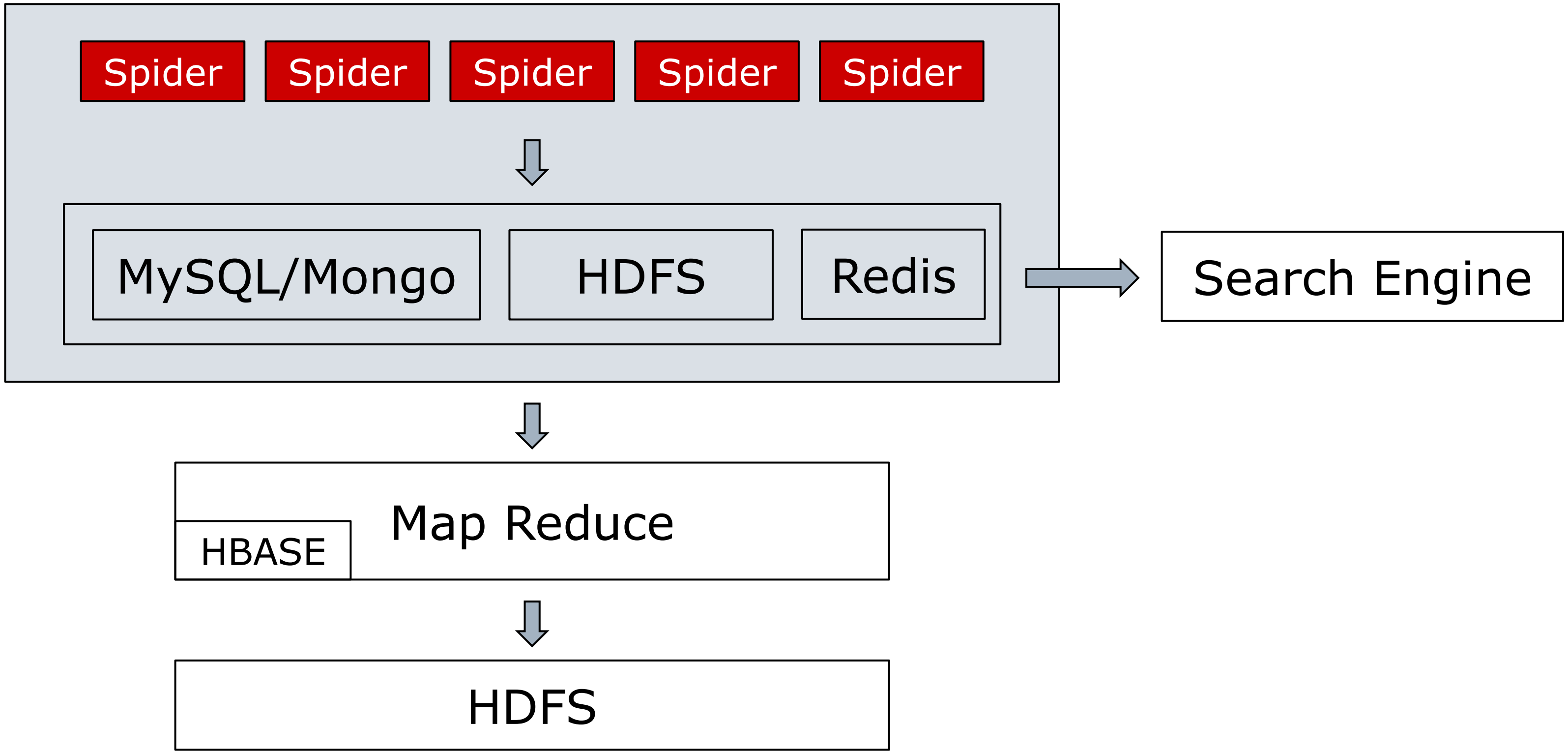
  1. GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'password' WITH GRANT OPTION; FLUSH PRIVILEGES;

  2. my.cnf
  ```
  #bind-address = 127.0.0.1
  ```

- 设置防火墙，允许端口远程连接

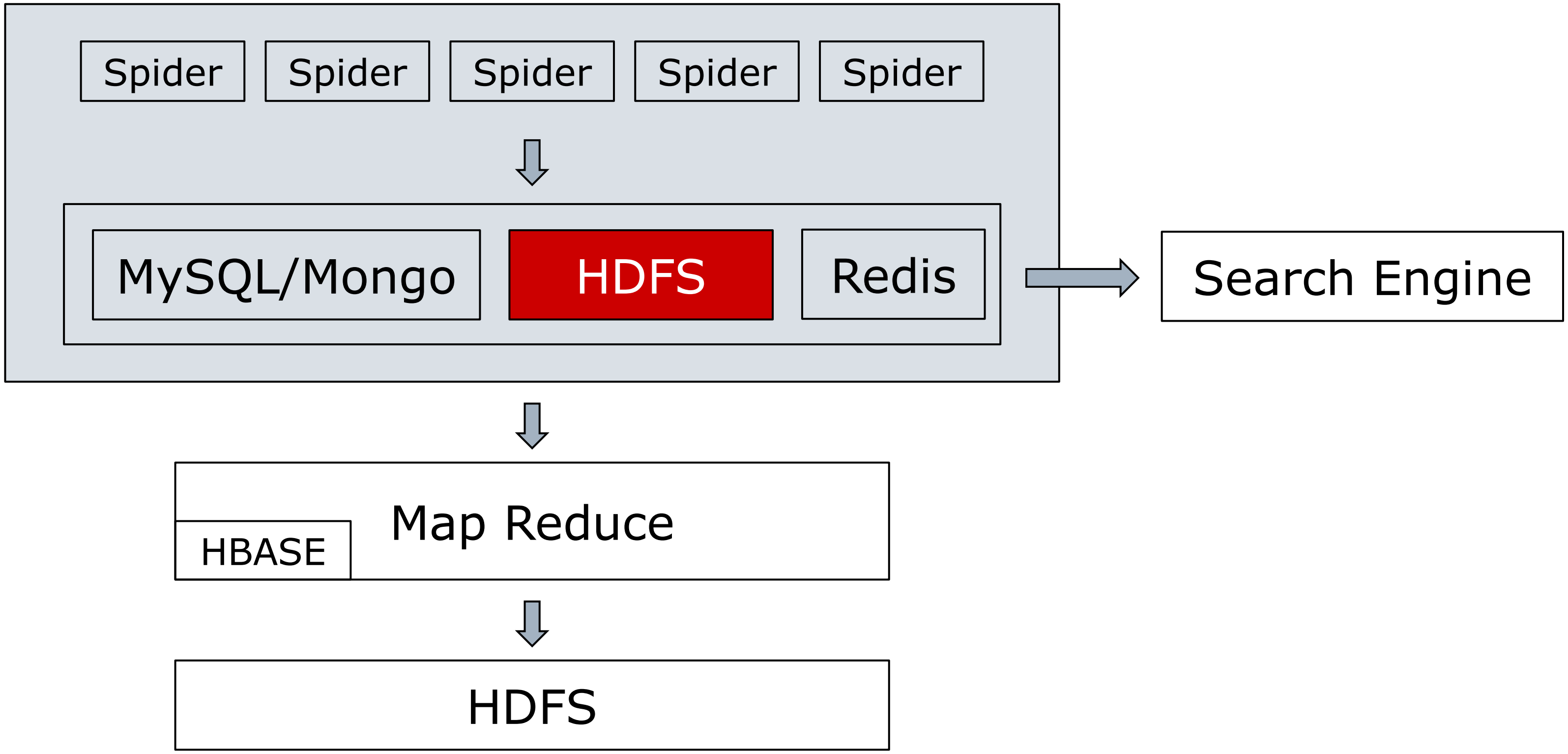
  ```
  iptables -A INPUT -i eth0 -p tcp -m tcp --dport 3306 -j ACCEPT
  ```

分布式爬虫系统 – 爬虫


Spider
Spider
Spider
Spider
Spider
MySQL/Mongo
HDFS
Redis
Search Engine
Map Reduce
HBASE
HDFS

分布式存储

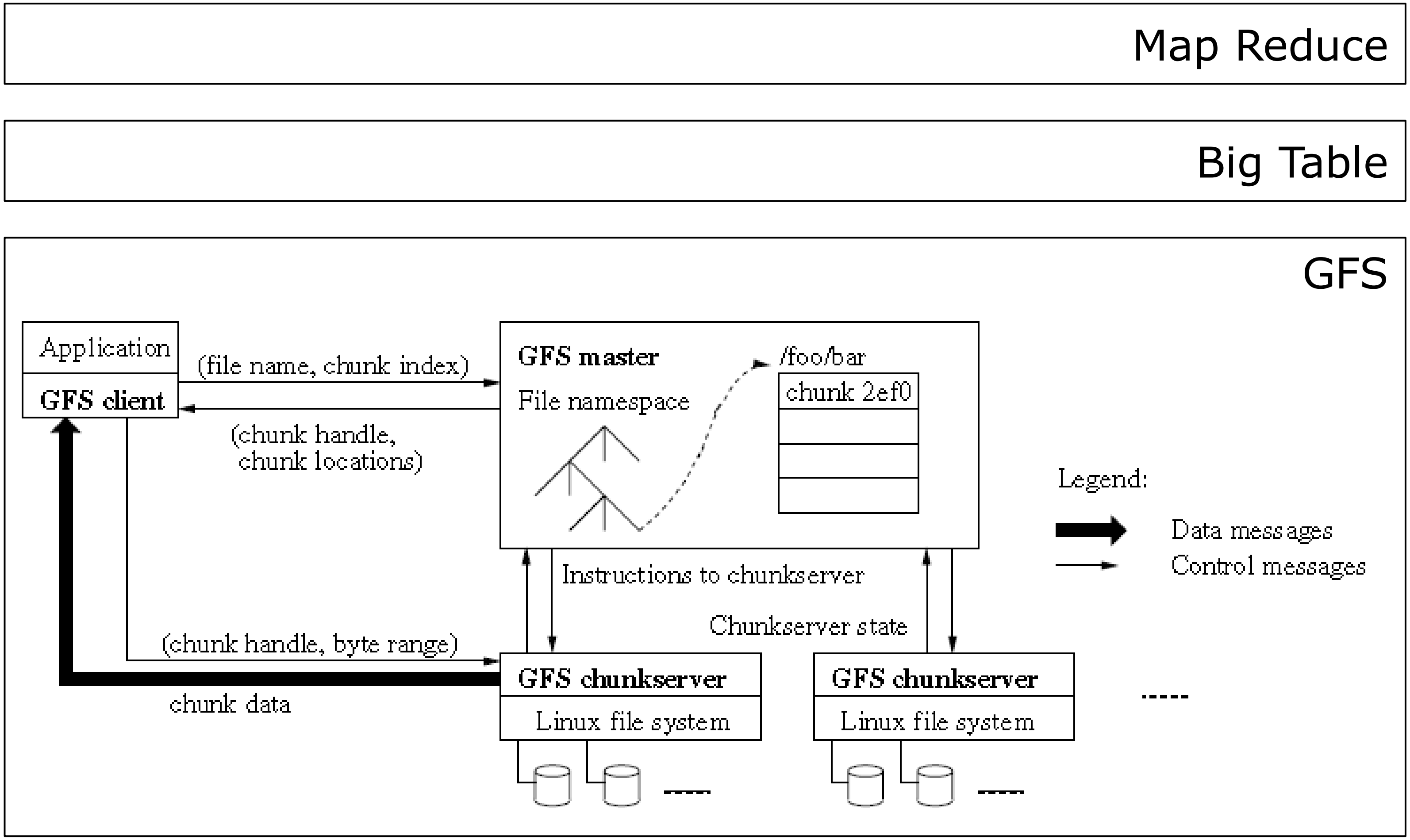
分布式爬虫系统 – 存储


Spider
Spider
Spider
Spider
Spider
MySQL/Mongo
HDFS
Redis
Search Engine
Map Reduce
HBASE
HDFS

爬虫原始数据存储特点

- 文件小，大量 KB 级别的文件
- 文件数量大
- 增量方式一次性写入，极少需要修改
- 顺序读取
- 并发的文件读写
- 可扩展

Google FS



Map Reduce

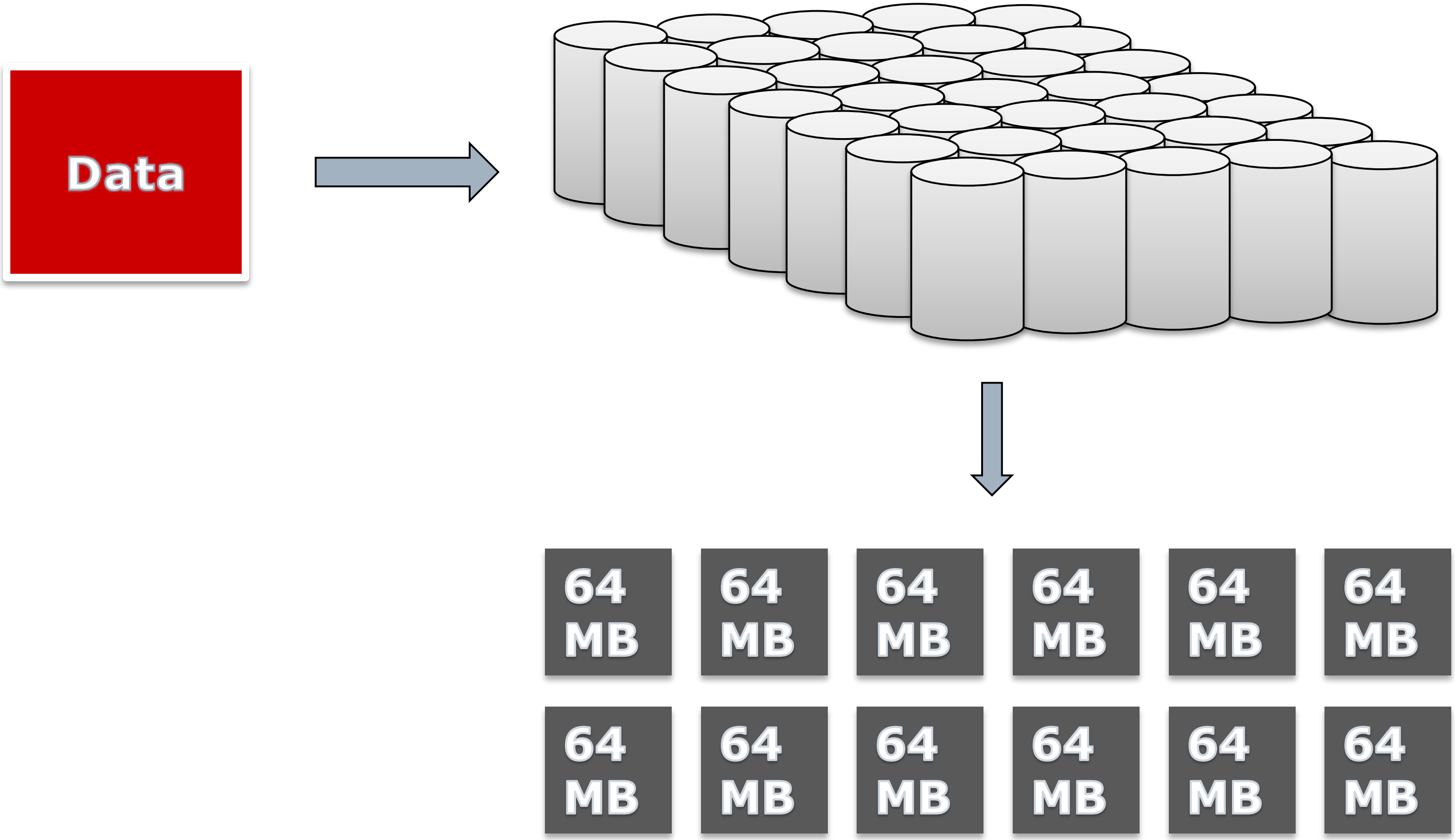
Big Table

GFS

HDFS

- Distributed, Scalable, Portable File System
- Written in Java
- Not fully POSIX-compliant
- Replication: 3 copies by default
- Designed for immutable files
- Files are cached and chunked, chunk size 64MB

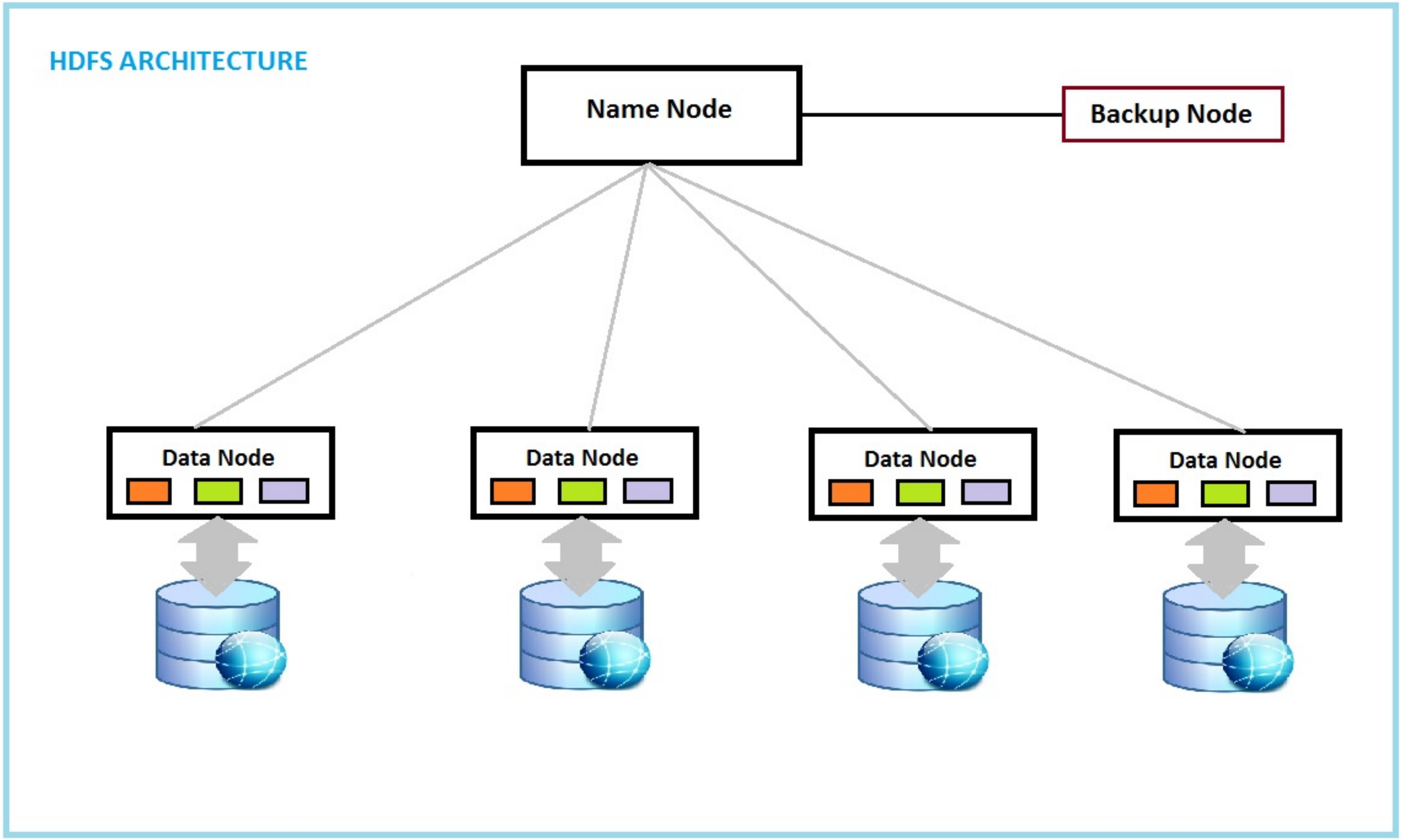
HDFS



Data

64 MB	64 MB	64 MB	64 MB	64 MB	64 MB
64 MB	64 MB	64 MB	64 MB	64 MB	64 MB

HDFS


HDFS ARCHITECTURE
Name Node
Backup Node
Data Node
Data Node
Data Node
Data Node

Python hdfs module

Installation: pip install hdfs

Methods	Desc
read()	read a file
write()	write to file
delete()	Remove a file or directory from HDFS.
rename()	Move a file or folder.
download	Download a file or folder from HDFS and save it locally.
list()	Return names of files contained in a remote folder.
makedirs()	Create a remote directory, recursively if necessary.
rename()	Move a file or folder.
resolve()	Return absolute, normalized path, with special markers expanded.
upload()	Upload a file or directory to HDFS.
walk()	Depth-first walk of remote filesystem.

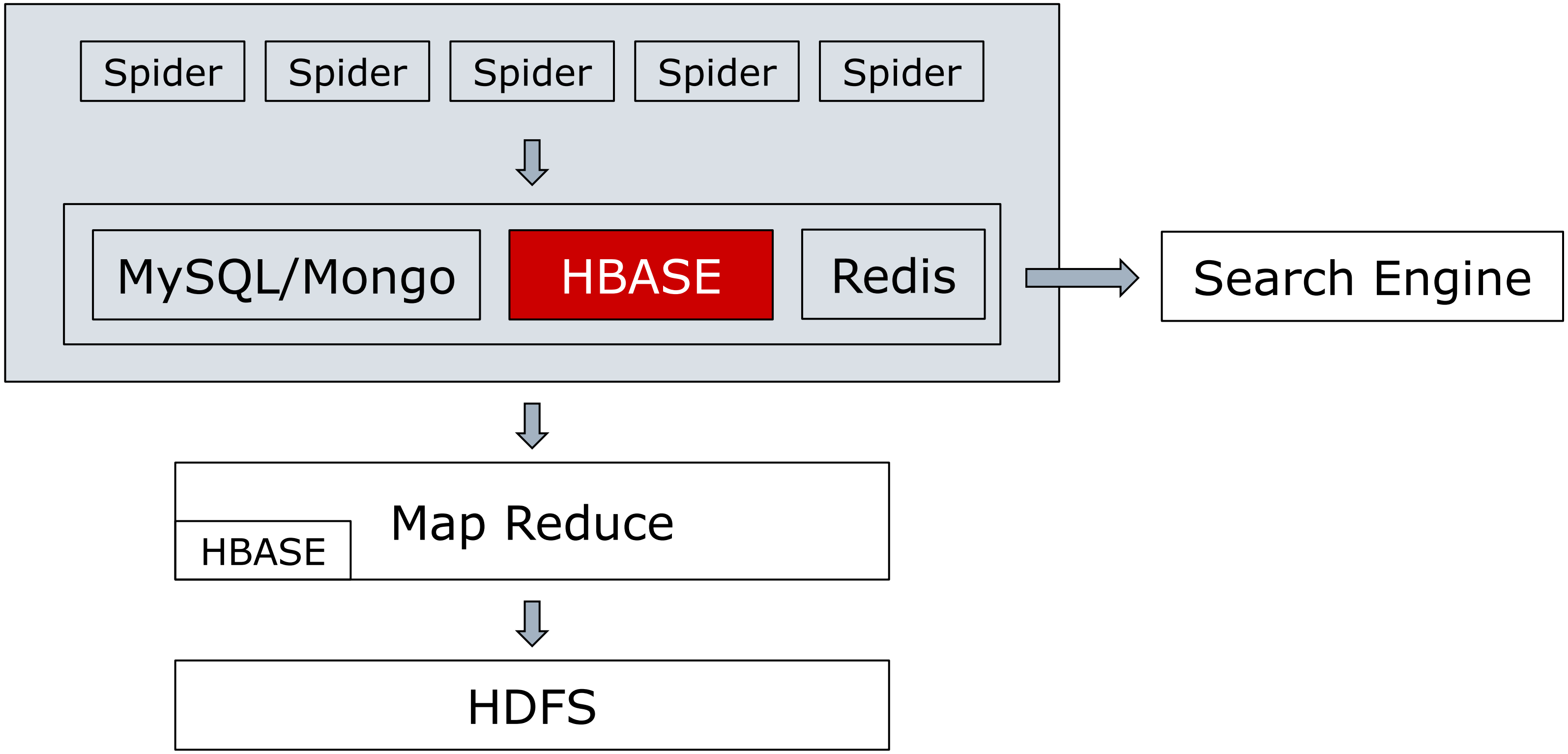
存储到HDFS

```
from hdfs import *
from hdfs.util import HdfsError

hdfs_client = InsecureClient(’[host]:[port]', user=’user')


with hdfs_client.write('/htmls/mfw/%s.html' % (filename)) as writer:
            writer.write(html_page)


except HdfsError, Arguments:
            print Arguments
```

HBASE

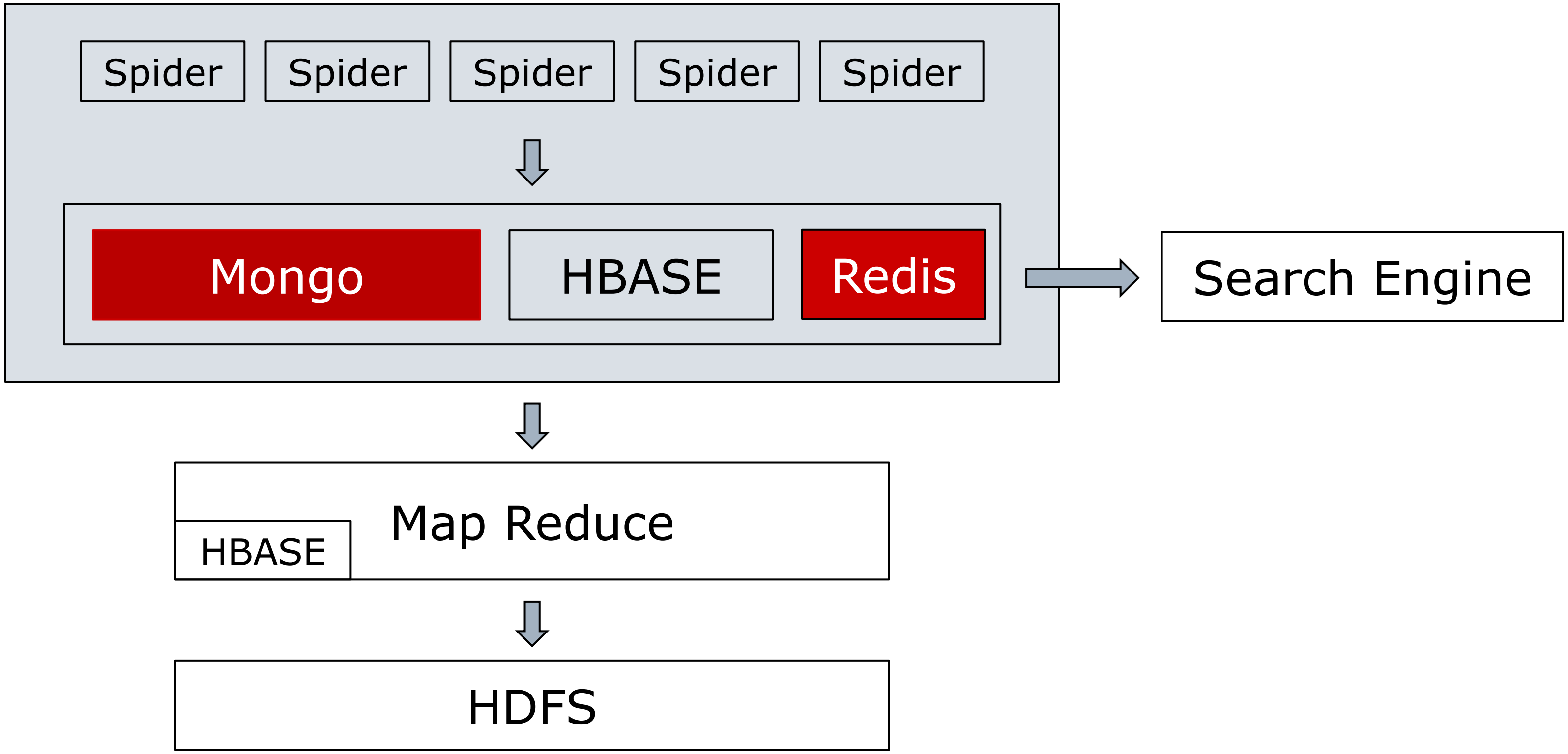
- On top of HDFS
- Column-oriented database
- Can store huge size raw data
- KEY-VALUE

分布式爬虫系统 – 存储


Spider
Spider
Spider
Spider
Spider
MySQL/Mongo
HBASE
Redis
Search Engine
Map Reduce
HBASE
HDFS

分布式数据库

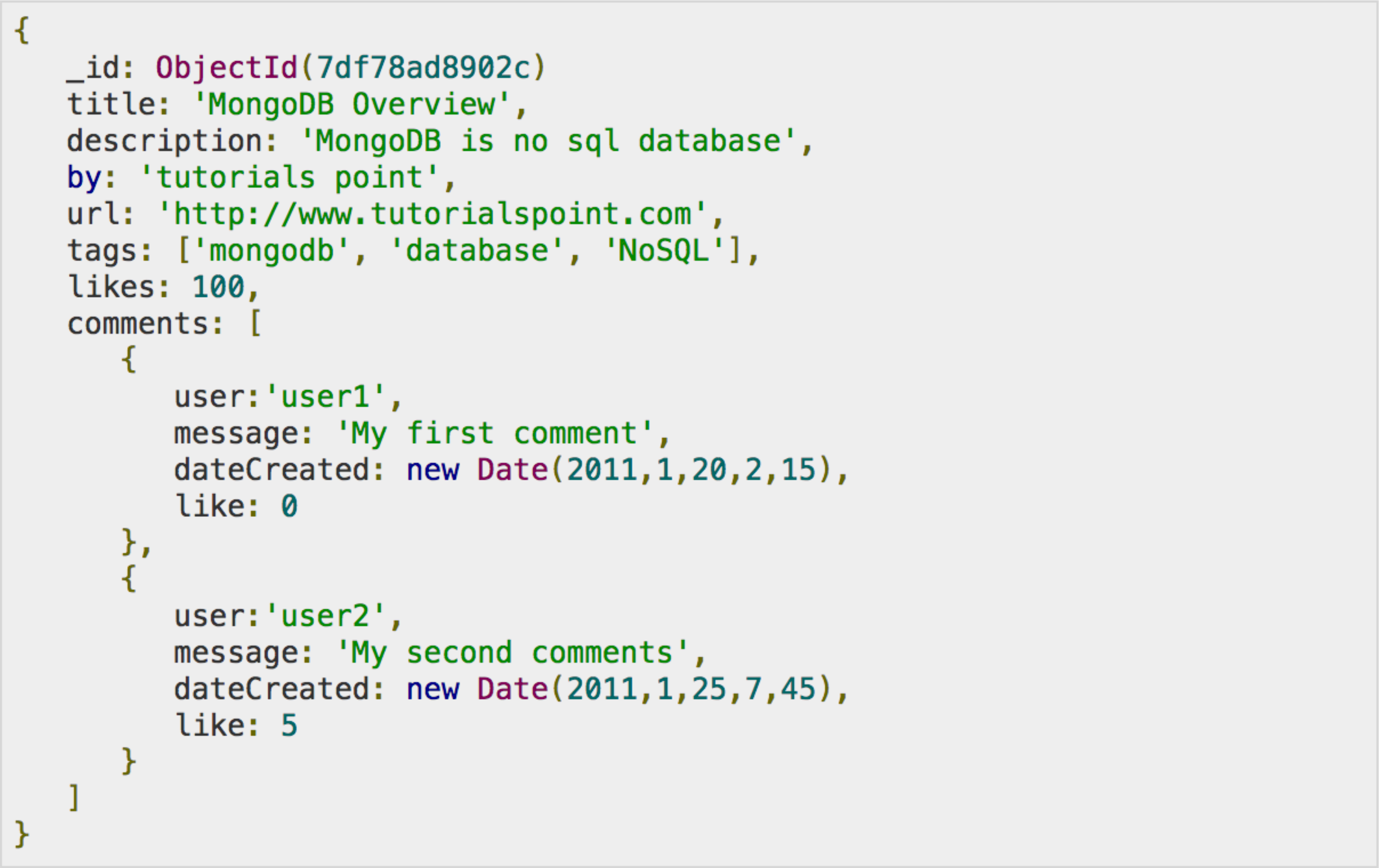
分布式爬虫系统 - 数据库


Spider
Spider
Spider
Spider
Spider
Mongo
HBASE
Redis
Search Engine
Map Reduce
HBASE
HDFS

MongoDB

RDBMS	MongoDB
Database	Database
Table	Collection
Tuple/Row	Document
column	Field
Table Join	Embedded Documents
Primary Key	Primary Key (Default key _id provided by mongodb itself)

MongoDB Document



```
{
   _id: ObjectId(7df78ad8902c)
   title: 'MongoDB Overview',
   description: 'MongoDB is no sql database',
   by: 'tutorials point',
   url: 'http://www.tutorialspoint.com',
   tags: ['mongodb', 'database', 'NoSQL'],
   likes: 100,
   comments: [
      {
         user:'user1',
         message: 'My first comment',
         dateCreated: new Date(2011,1,20,2,15),
         like: 0
      },
      {
         user:'user2',
         message: 'My second comments',
         dateCreated: new Date(2011,1,25,7,45),
         like: 5
      }
   ]
}
```

MongoDB

- **Schema less** – MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- Structure of a single object is clear.
- No complex joins.
- Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
- **Ease of scale-out** – MongoDB is easy to scale.
- Conversion/mapping of application objects to database objects not needed.

Installation

Download

https://www.mongodb.com/download-center?jmp=nav#community
https://fastdl.mongodb.org/linux/mongodb-linux-x86_64-amazon-3.4.2.tgz

Setup

```
mkdir mongodb
tar xzvf mongodb-linux-x86_64-amazon-3.4.2.tgz –C mongodb

# create default db folder, may need to change owner to current user
mkdir –p /data/db
nohup mongod&
```

client

```
mongo
```

MongoDB

db.collection.findOneAndUpdate(**filter, update, options**)

- Returns one document that satisfies the specified query criteria.
- Returns the first document according to natural order, means insert order
- Find and update are done atomically

```
db.mfw.findOneAndUpdate(
    { "status" : ”new" },
    { $set: { ”status" : ”downloading"} },
    { upsert:false, returnNewDocument : false}
);
```

MongoClient methods:

db.spider.mfw.find_one_and_update()

数据库类型

Type	Databases
RDBMS	Oracle, MySQL
Key-Value	BerkeleyDB
In Memory Key-Value	MemoryCached, Redis
Document	MongoDB
Column	HBase
Graphic	Neo4j, Titan

Redis Overview

- 基于 KEY VALUE 模式的内存数据库
- 支持复杂的对象模型（MemoryCached 仅支持少量类型）
- 支持 Replication，实现集群 （ MemoryCached 不支持分布式部署）
- 所有操作都是原子性 （MemoryCached 多数操作都不是原子的）
- 可以序列化到磁盘 （ MemoryCached 不能序列化）

Redis Environment Setup

Download

```
$ wget http://download.redis.io/releases/redis-3.2.7.tar.gz
$ tar xzf redis-3.2.7.tar.gz
$ cd redis-3.2.7
$ make
```

Start server and cli

```
$ nohup src/redis-server&
$ src/redis-cli
```

Test it

```
redis> set foo bar
OK
redis> get foo
"bar"
```

Python Redis

Installation

```
$ sudo pip install redis
```

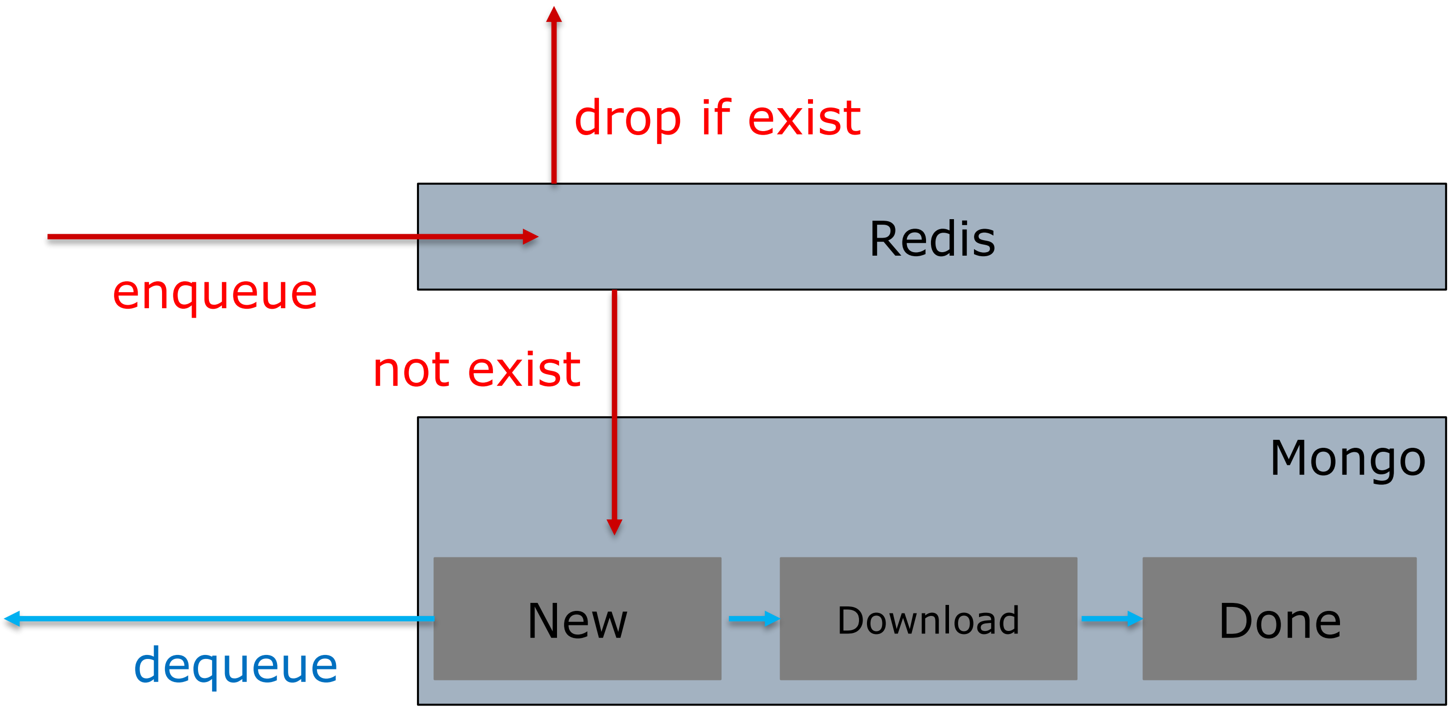
Sample Code

```
>>> import redis
>>> r = redis.StrictRedis(host='localhost', port=6379, db=0)
>>>
>>> r.set('foo', 'bar')
True
>>> r.get('foo')
'bar'
```

Mongo 的优化

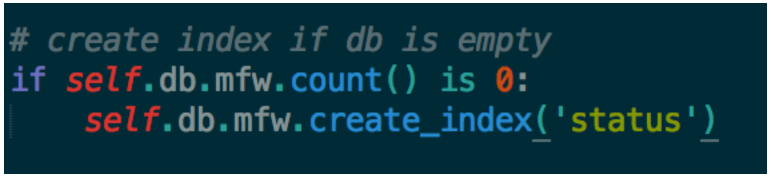
- url 作为 _id，默认会被创建索引，创建索引是需要额外的开销的
- index 尽量简单， url 长了一些
- dequeueUrl find_one() 并没有利用 index，会全库扫描，但是仍然会很快，因为扫描到第一个后就停止了，但是当下载完成后的数量特别大的时候，扫描依然是很费时的，考虑一下能不能进一步优化
- 插入的操作很频繁，每一个网页对应着几百次插入，到了 depth = 3 的时候，基数网页是百万级，插入检查将是亿级，考虑使用更高效的方式来检查

Mongo with Redis

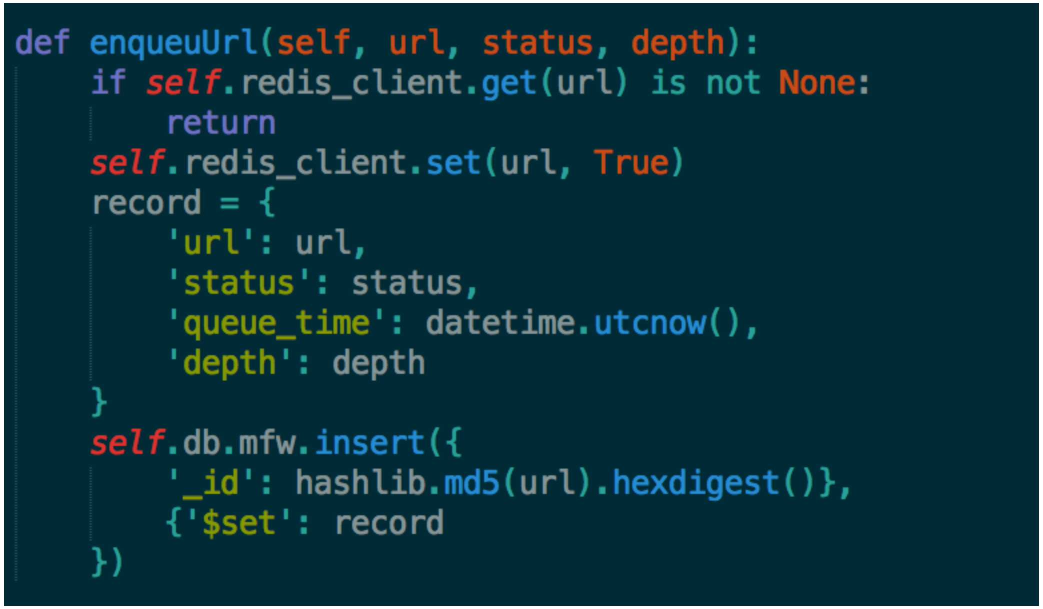


status: create index OR in different collections

Code Snippet



```python
# create index if db is empty
if self.db.mfw.count() is 0:
    self.db.mfw.create_index('status')
```



```python
def enqueuUrl(self, url, status, depth):
    if self.redis_client.get(url) is not None:
        return
    self.redis_client.set(url, True)
    record = {
        'url': url,
        'status': status,
        'queue_time': datetime.utcnow(),
        'depth': depth
    }
    self.db.mfw.insert({
        '_id': hashlib.md5(url).hexdigest()},
        {'$set': record
    })
```

疑问

- 问题答疑： http://www.xxwenda.com/
  - 可邀请老师或者其他人回答问题

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：大数据分析挖掘
- 新浪微博：ChinaHadoop