Python&Hadoop构建数据仓库

从开源中来,到开源中去

EasyHadoop 童小军
tongxiaojun@gmail.com
2012年10年20日

EasyHadoop

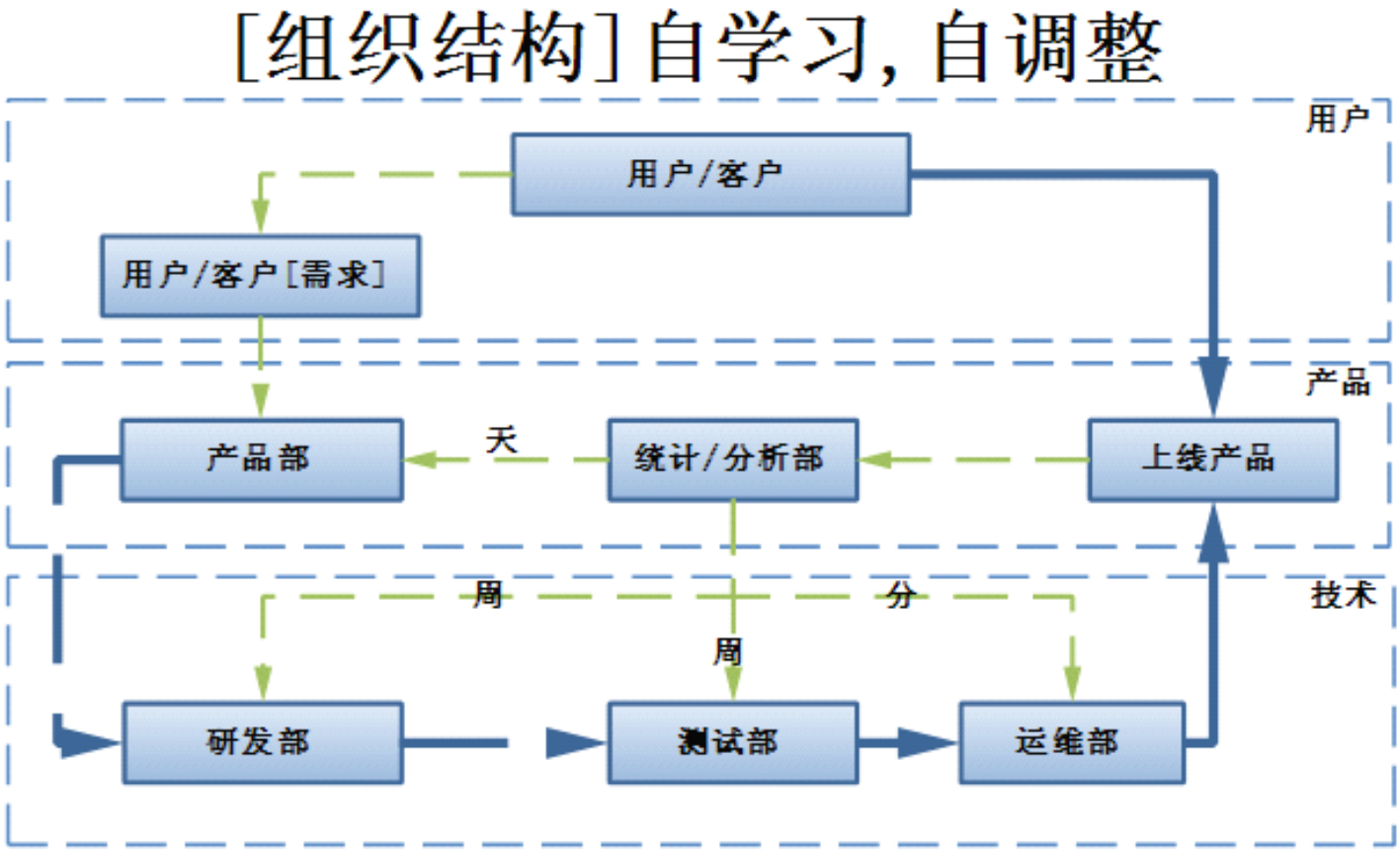
演讲大纲

- 个人介绍
- 思考数据分析系统的基本指标
- Hadoop史前和史后的数据仓库流程
- Hadoop史前和史后的数据分析流程
- 思考Hadoop解决了什么样的根本问题

- Python 如何在构建数据仓库系统的作用
  - 1. 使用Python快速构建 数据分析模块 ComETL
  - 2. 基于Python MapReduce Streaming 快速并行编程
  - 3. Hive如果内嵌Python实现自定义逻辑
  - 4. Pig内嵌JPython 实现PageRank挖掘算法
  - 5. JPython MapReduce 框架 Pydoop Happy 等。

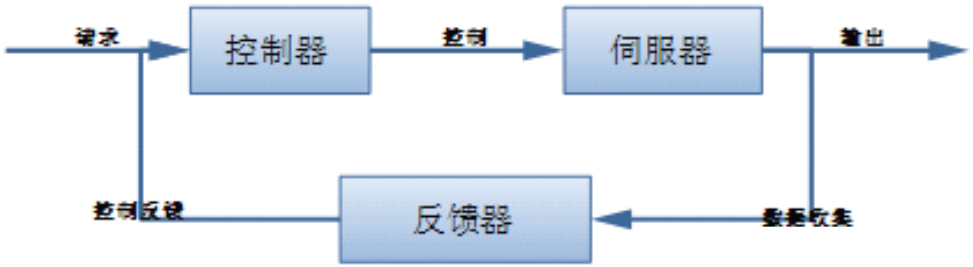
- 使用开源软件配合Python快速构建数据仓库
- EasyHadoop提供的资料[EasyHadoop部署安装手册,EasyHive手册]
- EasyHadoop开源技术聚会

思考-数据分析系统的基本指标

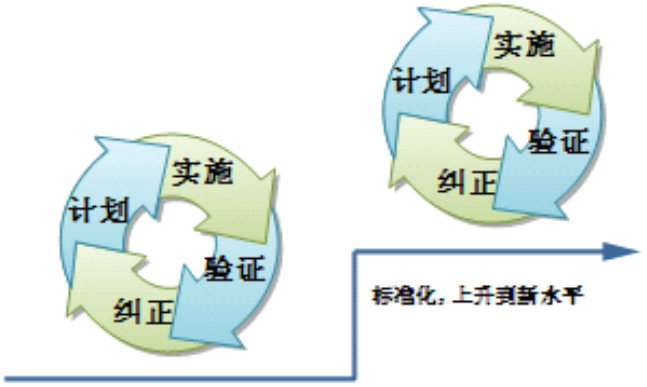


思考-数据分析系统的基本指标

[自动控制]闭环反馈



[质量持续优化]PDCA 戴明环



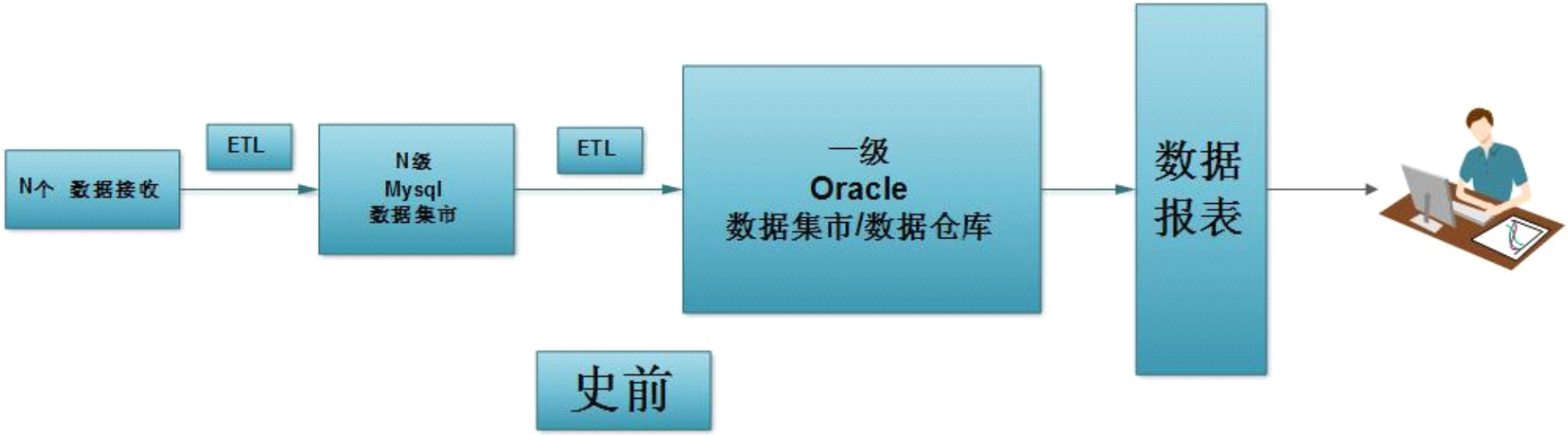
思考-数据分析系统的基本指标

反馈决策周期！快
反馈决策粒度！细
反馈决策准确性！准
反馈总体成本！廉价

数据统计/分析 是一个组织
自动控制,自学习,自调整系统
核心组成部分。机会成本! 想象空间!

Hadoop 前的数据仓库流程

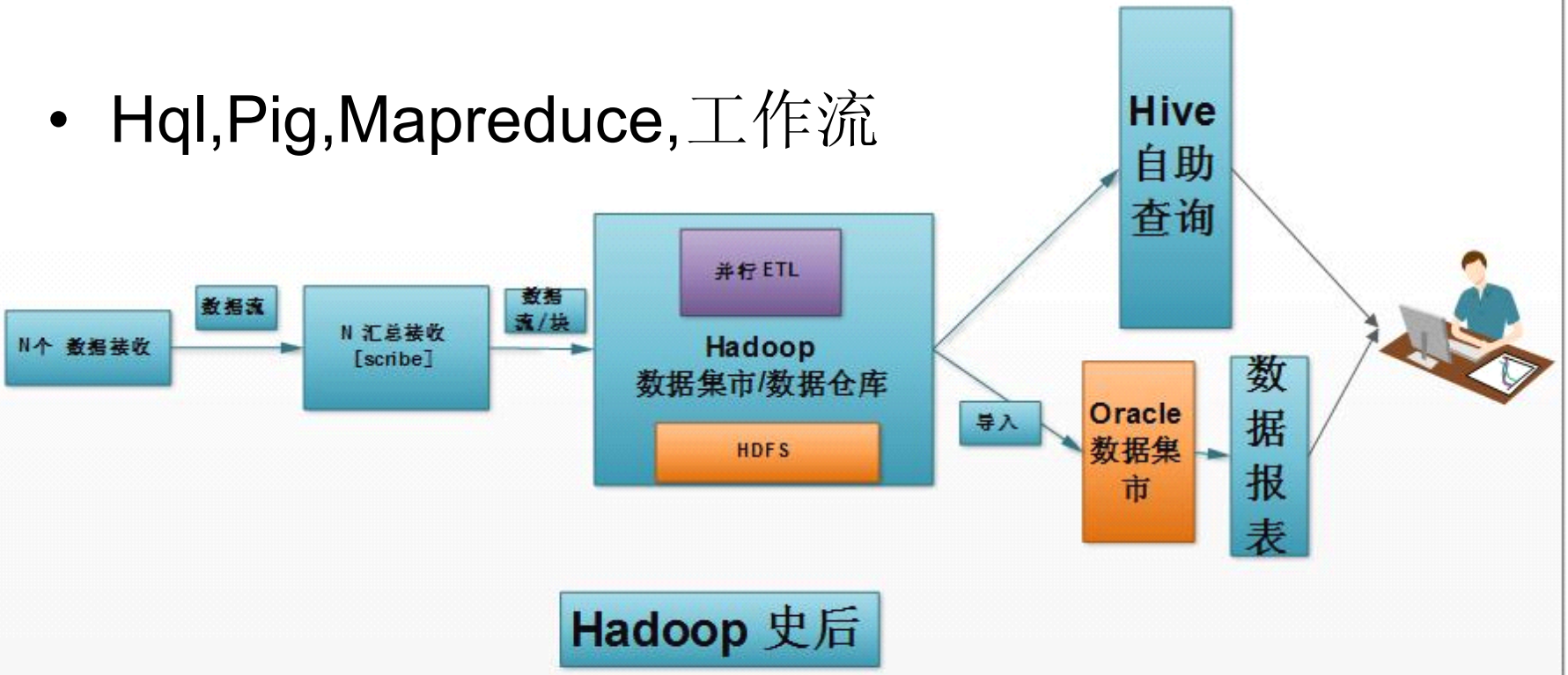
- perl,shell,awk



反馈决策周期！ 快 ？
反馈决策粒度! 细 ？
反馈决策准确性! 准 ？
反馈总体成本! 廉价 ？

Hadoop后的数据仓库流程

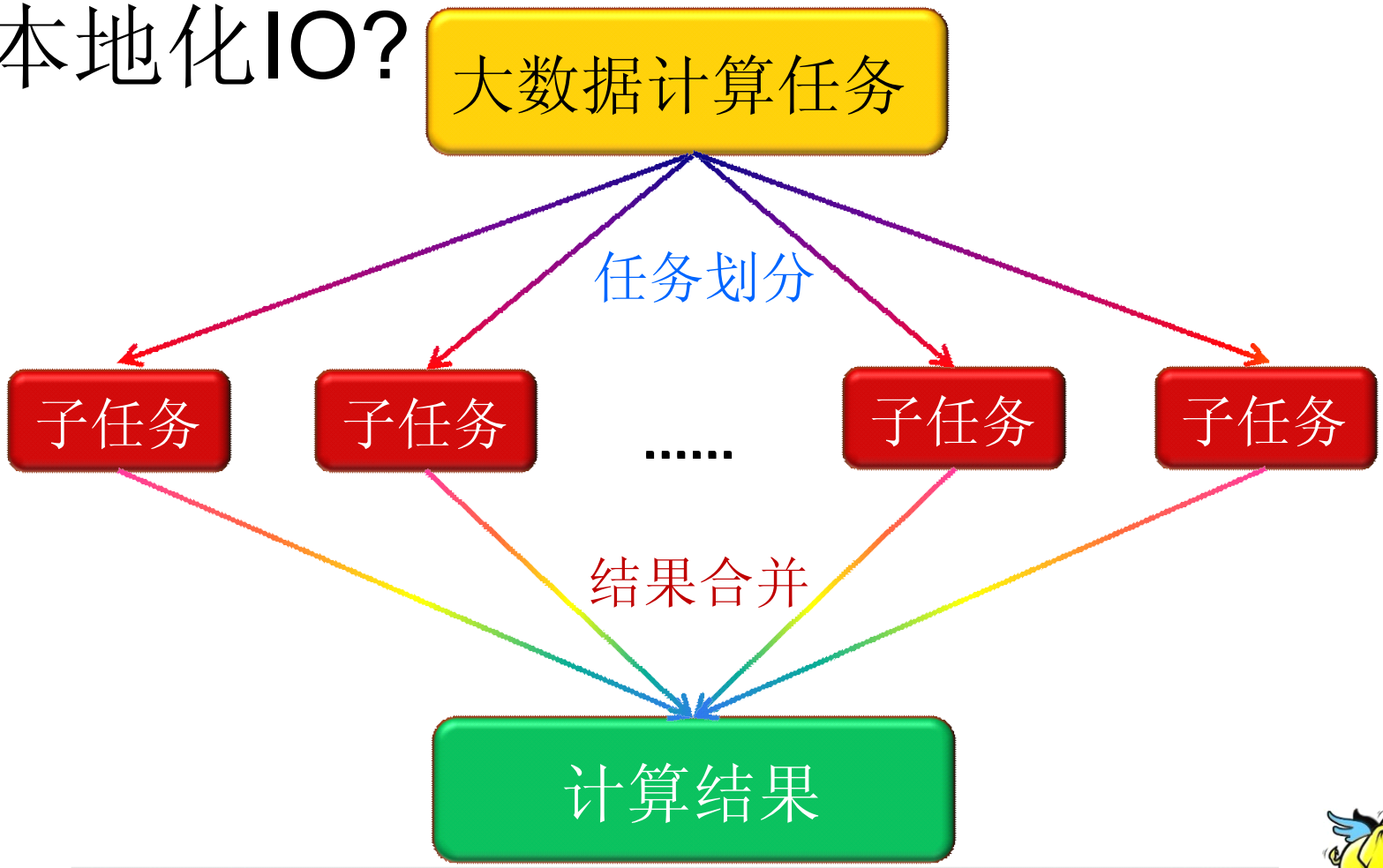
- Hql,Pig,Mapreduce,工作流

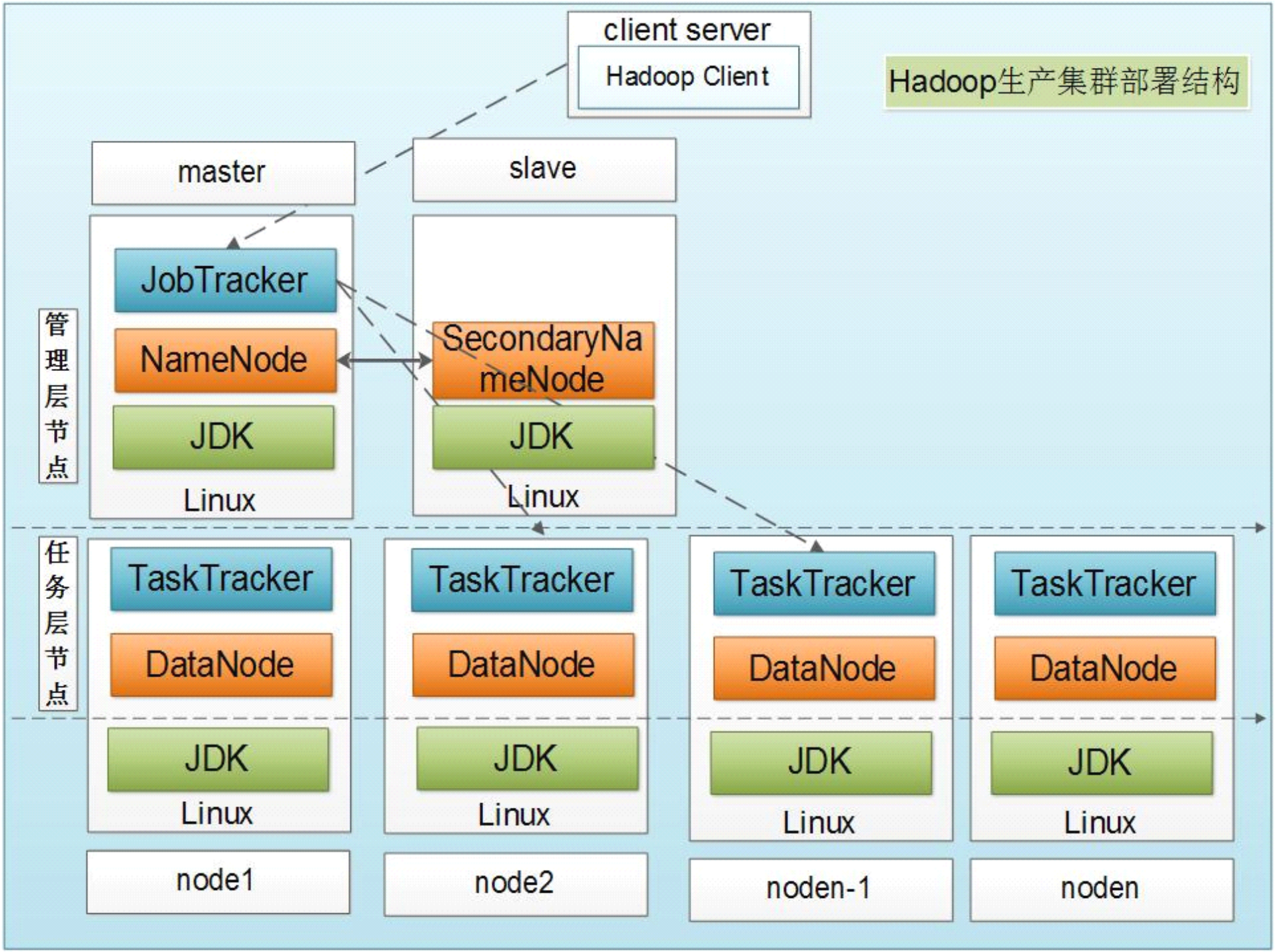


反馈决策周期！快？
反馈决策粒度！细？
反馈决策准确性！准？
反馈总体成本！廉价？
持续扩展成本？

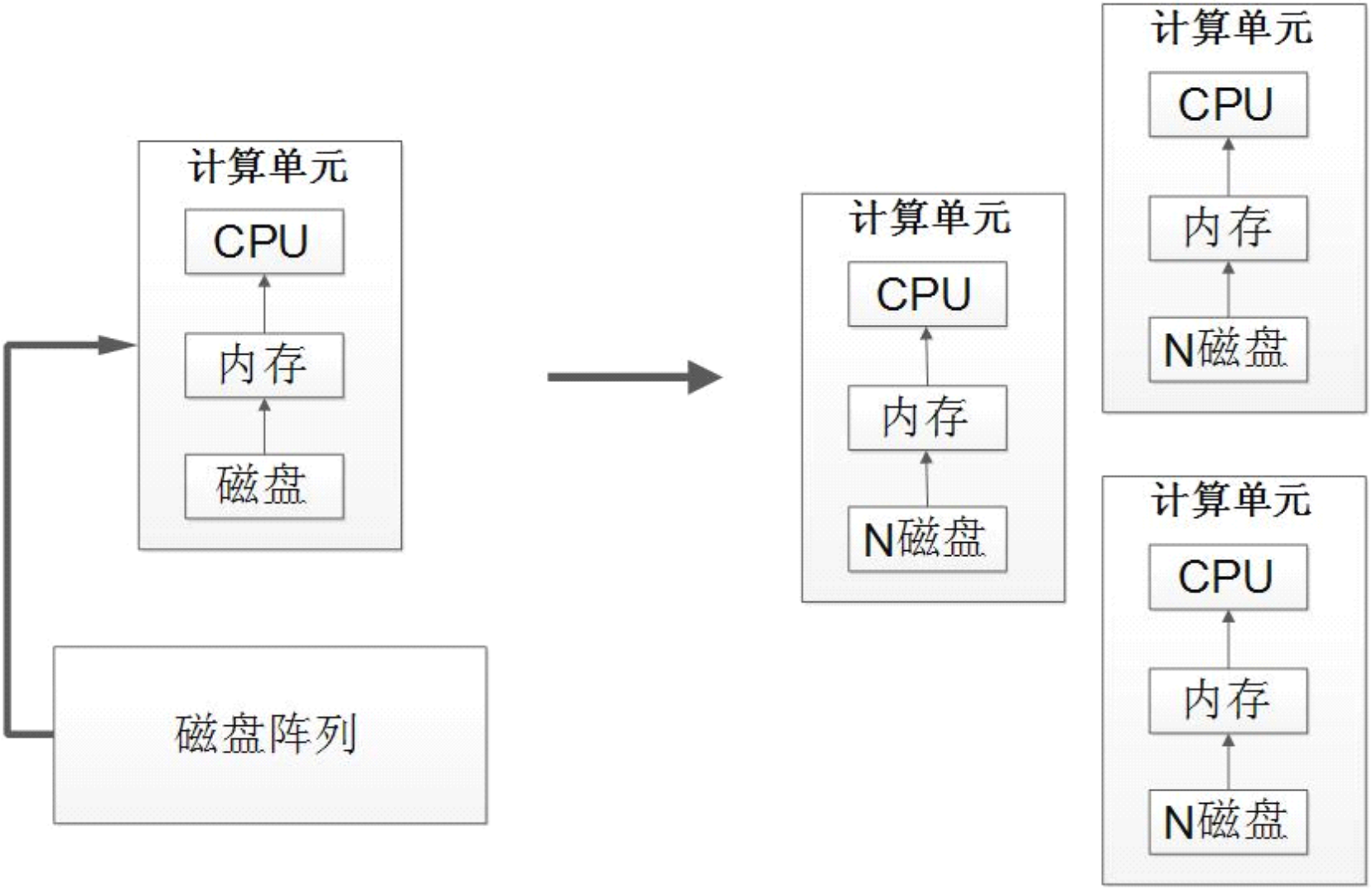
EasyHadoop

Hadoop解决了什么样的根本问题？
Hadoop为何会比数据库快？
本地化IO？

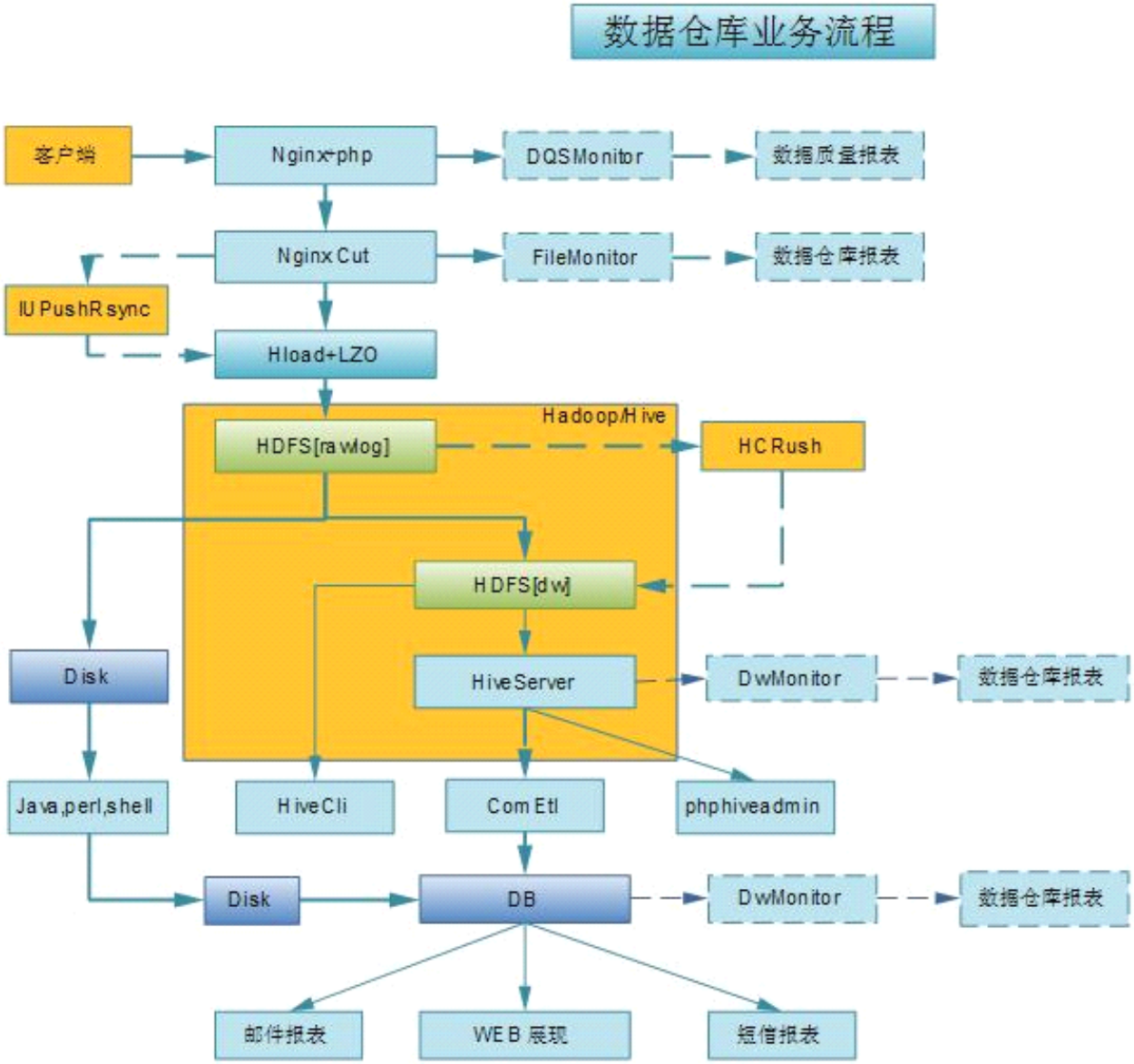



client server
Hadoop Client
Hadoop生产集群部署结构
master
slave
管理层节点
JobTracker
NameNode
SecondaryNameNode
JDK
JDK
Linux
Linux
任务层节点
TaskTracker
TaskTracker
TaskTracker
TaskTracker
DataNode
DataNode
DataNode
DataNode
JDK
JDK
JDK
JDK
Linux
Linux
Linux
Linux
node1
node2
noden-1
noden

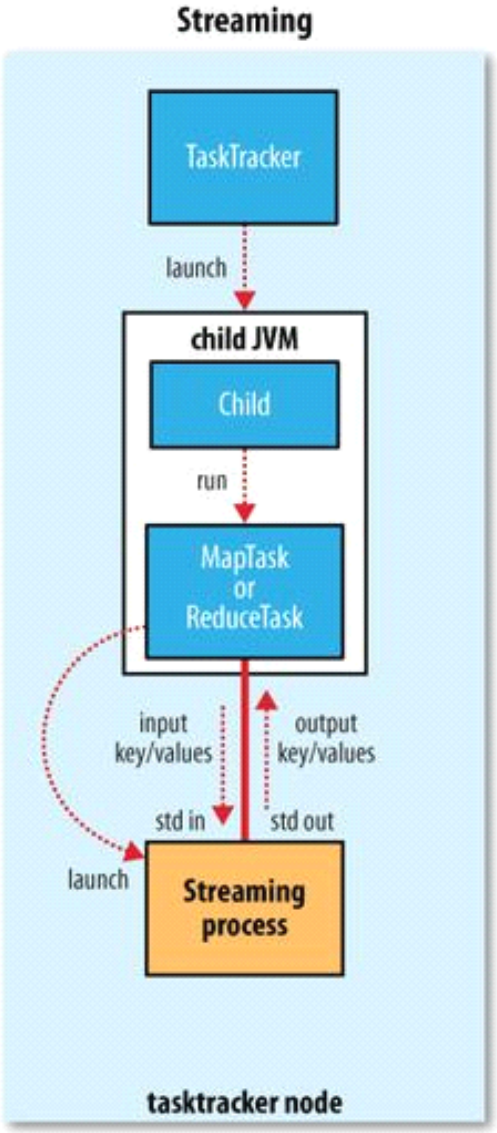
磁盘读取速度,网络IO?


计算单元
CPU
内存
磁盘
磁盘阵列
计算单元
CPU
内存
N磁盘
计算单元
CPU
内存
N磁盘
计算单元
CPU
内存
N磁盘

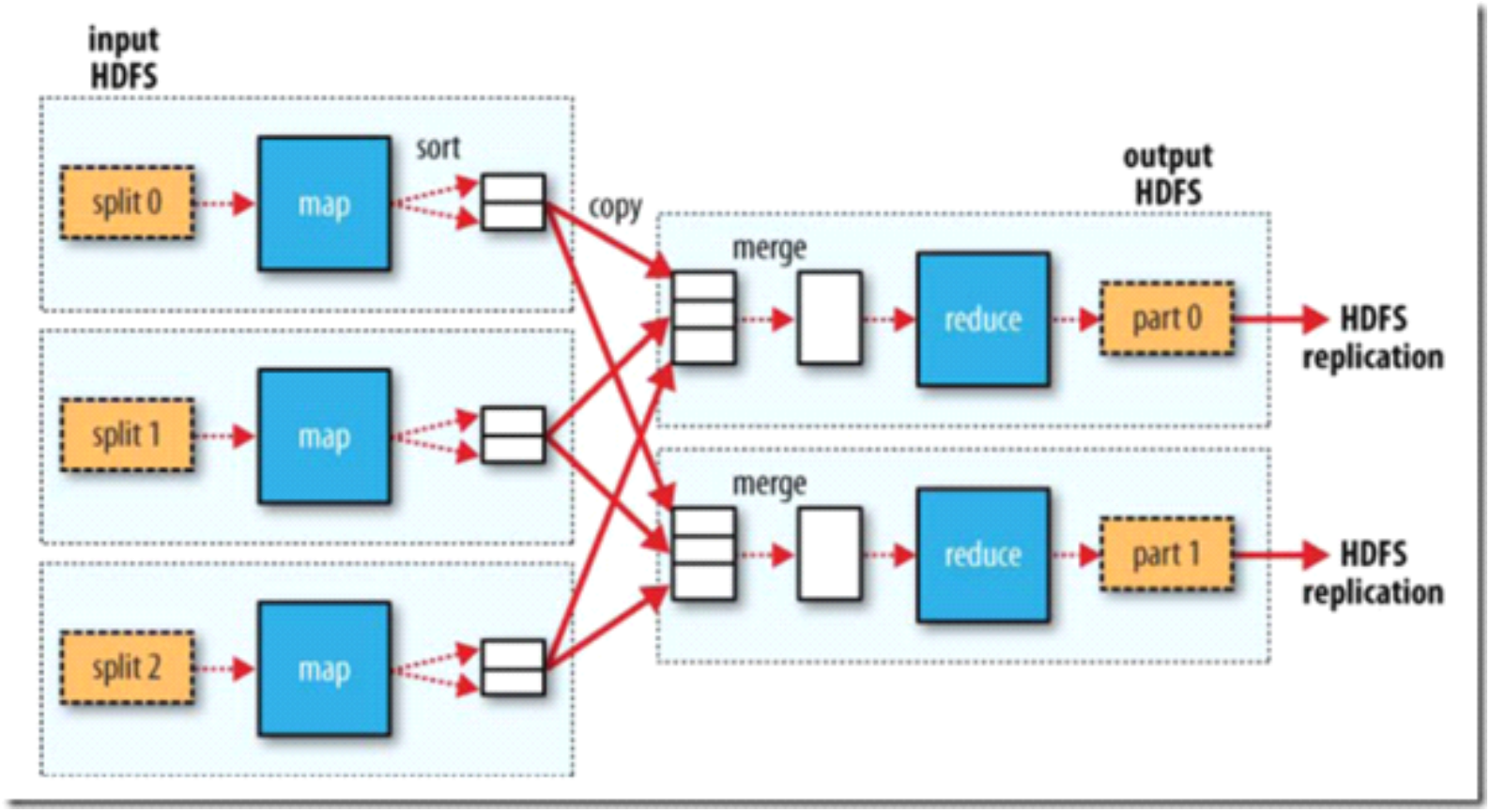
基于hadoop的数据平台总体架构



Python 结合 Hadoop Streaming



MapReduce基本流程



实现distinct

一、日志格式：

{0E3AAC3B-E705-4915-9ED4-EB7B1E963590}
{FB11E363-6D2B-40C6-A096-95D8959CDB92}
{06F7CAAB-E165-4F48-B32C-8DD1A8BA2562}
{B17F6175-6D36-44D1-946F-D748C494648A}
{06F7CAAB-E165-4F48-B32C-8DD1A8BA2562}
{B17F6175-6D36-44D1-946F-D748C494648A}

B11E363-6D2B-40C6-A096-95D8959CDB92
17F6175-6D36-44D1-946F-D748C494648A
E3AAC3B-E705-4915-9ED4-EB7B1E963590
6F7CAAB-E165-4F48-B32C-8DD1A8BA2562

4

使用python实现 distinct/count

一、日志格式：

{0E3AAC3B-E705-4915-9ED4-EB7B1E963590}
{FB11E363-6D2B-40C6-A096-95D8959CDB92}
{06F7CAAB-E165-4F48-B32C-8DD1A8BA2562}
{B17F6175-6D36-44D1-946F-D748C494648A}
{06F7CAAB-E165-4F48-B32C-8DD1A8BA2562}
{B17F6175-6D36-44D1-946F-D748C494648A}

B11E363-6D2B-40C6-A096-95D8959CDB92
17F6175-6D36-44D1-946F-D748C494648A
E3AAC3B-E705-4915-9ED4-EB7B1E963590
6F7CAAB-E165-4F48-B32C-8DD1A8BA2562

4

(distinct\count)--map

```
import sys
for line in sys.stdin:
        try:
                flags = line[1:-2]
                str = flags+'\t'+'1'
                 print str

        except Exception,e:
                print e
```

(distinct)--red

```
#!/usr/bin/python
import sys
res = {}
      for line in sys.stdin:
        try:
              flags = line[:-1].split('\t')
              if len(flags) != 2:
                    continue
              field_key = flags[0]
              if res.has_key(field_key) ==
      False:
                    res[field_key] = [0]
              res[field_key][0] = 1
        except Exception,e:
              pass
for key in res:
        print key
```

(count的优化实现)--reduce

```
#!/usr/bin/python
import sys
lastuid=""
num=1
for line in sys.stdin:
        uid,count=line[:-1].split('\t')
        if lastuid ==="":
                lastuid=uid
        if lastuid != uid:
                num+=1
                lastuid=uid
print num
```

基于Python MapReduce Streaming 快速并行编程

一、单机测试

```
head  test.log | python map.py | python red.py
```

一、将文件上传到集群

```
/bin/hadoop      fs  -copyFromLocal   test.log   /hdfs/
```

三、运行map red

```
/bin/hadoop jar contrib/streaming/hadoop-streaming-0.20.203.0.jar -file
    /path/map.py  -file  /path/red.py
-mapper map.py  -reducer  red.py
 -input   /path/test.log  -output  /path/
```

通过界面查看任务状态

Quick Links

oser622 Hadoop Map/Reduce Administration

State: RUNNING
Started: Mon Oct 17 15:02:38 CST 2011
Version: 0.20.203.0, r1099333
Compiled: Wed May 4 07:57:50 PDT 2011 by oom
Identifier: 201110171502

Cluster Summary (Heap Size is 295.5 MB/6.94 GB)

Running Map Tasks	Running Reduce Tasks	Total Submissions	Nodes	Occupied Map Slots	Occupied Reduce Slots	Reserved Map Slots	Reserved Reduce Slots	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node	Blacklisted Nodes	Graylisted Nodes	Excluded Nodes
9	2	30881	7	9	2	0	0	32	13	6.43	0	0	0

Scheduling Information

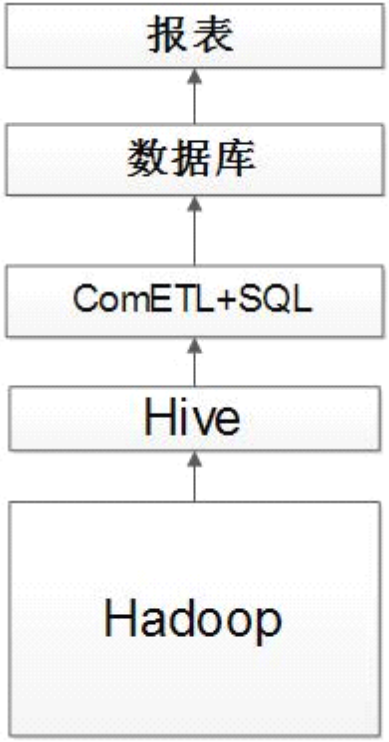
Queue Name	State	Scheduling Information
default	running	N/A

Filter (Jobid, Priority, User, Name)
Example: 'user:smith 3200' will filter by 'smith' only in the user field and '3200' in all fields

Running Jobs

Jobid	Priority	User	Name	Map % Complete	Map Total	Maps Completed	Reduce % Complete	Reduce Total	Reduces Completed	Job Scheduling Information	Diagnostic Info
job_201110171502_30934	NORMAL	rsync	streamjob43196.jar	85.41%	6	2	11.11%	1	0	NA	NA
job_201110171502_30939	NORMAL	rsync	streamjob46097.jar	55.34%	4	1	8.33%	1	0	NA	NA
job_201110171502_30940	NORMAL	rsync	streamjob43349.jar	0.62%	2	0	0.00%	1	0	NA	NA

Python快速构建 数据分析模块 ComETL



极少的代码量,几万行吧!

1.支持简单工作流
2.支持自动恢复
3.支持自定义驱动
4.支持 Hive Mysql MapReduce 等模式

类似系统 Sqoop DataX Oozie

作者:赵修湘

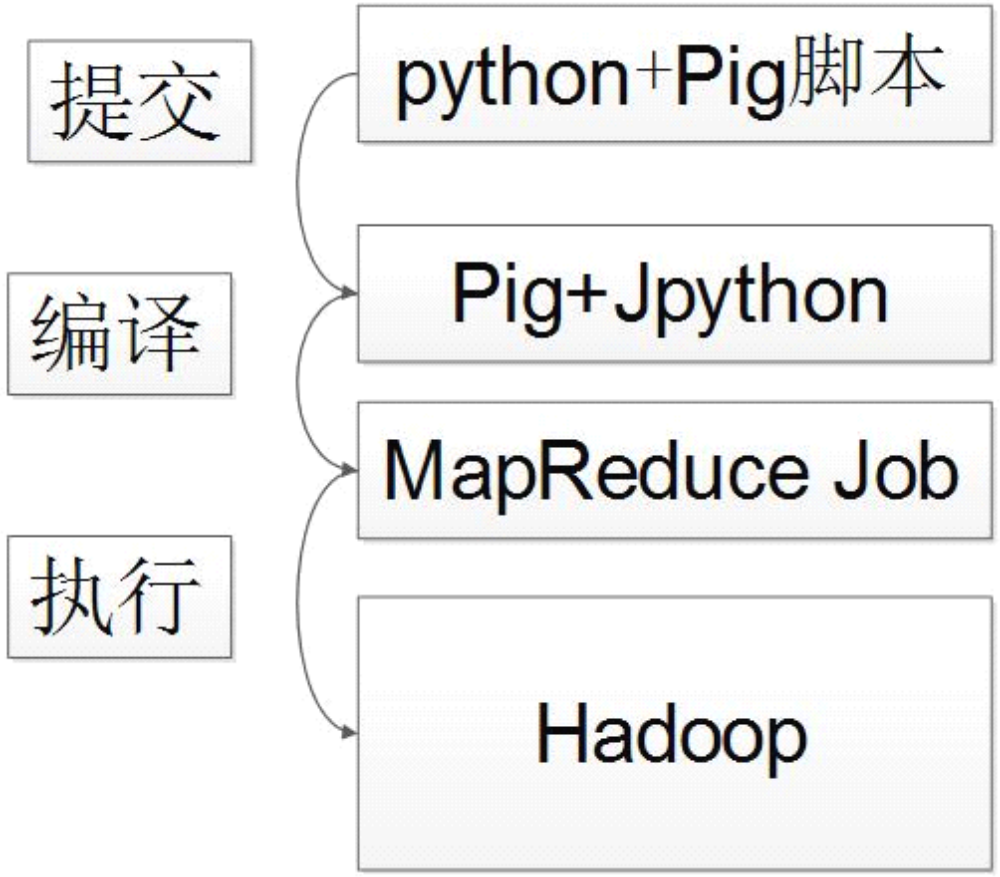
软件地址: https://github.com/zhuyeqing/ComETL

EasyHadoop

ComEtl配置样例

```
etl_op = {"run_mode":'day',
        "delay_hours":2,
        "jobs":[{"job_name":"job1",
                "analysis":[{'etl_class_name':'ExtractionEtl',
                        'step_name':'mysql_e_1',
                        'db_type'hive',
                        'db_coninfo':[{'db_ip':'192.168.1.50','db_port':3306,'db_user':'jobs','db_passwd':'hhxxttxs','db_db':'test'}],
                        'db_path':'test.a2',
                        'pre_sql':[],
                        'post_sql':[],
                        'data_save_type':'SimpleOutput',
                        "sql_assemble":'SimpleAssemble',
                        'sql':'select* from test.a2 limit 30',
                        },],
                "transform":[{'etl_class_name':'TransformEtl',
                          'step_name':'transform1',
                          'data_source':[{"job_name":"job1","step_name":'mysql_e_1','data_field':''},],
                          'data_transform_type':'SimpleTransform',
                          },],
                "loading":[{'etl_class_name':'LoadingEtl',
                        'step_name':'load1',
                        'data_source':{"job_name":"job1","step_name":'transform1'},
                        'db_type':'mysql',
                        'db_coninfo':[{'db_ip':'192.168.1.50','db_port':3306,'db_user':'jobs','db_passwd':'hhxxttxs','db_db':'test'}],
                        'db_path':'test.a2',
                        'pre_sql':[],
                        'post_sql':[],
                        'data_load_type':'SplitLoad',
                        'data_field':'a|b'},]}
        }
```

Pig内嵌JPython 实现PageRank算法



JPython+pig 代码实现演示

file | 44 lines (33 sloc) | 1.072 kb Edit Raw Blame History

```
#!/usr/bin/python
from org.apache.pig.scripting import *

P = Pig.compile("""
-- PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn))

previous_pagerank =
    LOAD '$docs_in'
    USING PigStorage('\t')
    AS ( url: chararray, pagerank: float, links:{ link: ( url: chararray ) } );

outbound_pagerank =
    FOREACH previous_pagerank
    GENERATE
        pagerank / COUNT ( links ) AS pagerank,
        FLATTEN ( links ) AS to_url;

new_pagerank =
    FOREACH
        ( COGROUP outbound_pagerank BY to_url, previous_pagerank BY url INNER )
    GENERATE
        group AS url,
        ( 1 - $d ) + $d * SUM ( outbound_pagerank.pagerank ) AS pagerank,
        FLATTEN ( previous_pagerank.links ) AS links;

STORE new_pagerank
    INTO '$docs_out'
    USING PigStorage('\t');
""")

params = { 'd': '0.5', 'docs_in': 'data/simple' }

for i in range(10):
        out = "out/pagerank_data_" + str(i + 1)
        params["docs_out"] = out
        Pig.fs("rmr " + out)
        stats = P.bind(params).runSingle()
        if not stats.isSuccessful():
                raise 'failed'
        params["docs_in"] = out

```

https://github.com/julienledem/Pig-scripting-examples/blob/

其他Python MapReduce框架

- Pydoop - Python API for Hadoop MapReduce and HDFS
- http://pydoop.sourceforge.net/docs/

- Happy - http://code.google.com/p/happy/
- datafu -Pig算法库 linkedin https://github.com/linkedin/datafu

总体数据规模

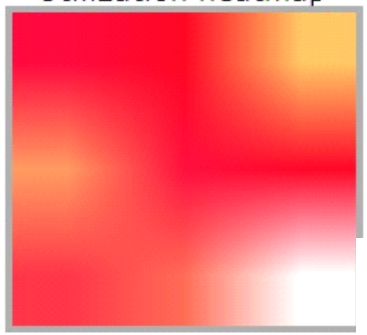
- 总空间150T以上，每日新增数据*0.5T*
- *20+* 服务器的Hadoop/hive计算平台
- 单个任务优化从 7个小时到 1个小时
- 每日 Hive 查询 1200+
- 每天处理3000+作业任务
- 每天处理*10T+*数据

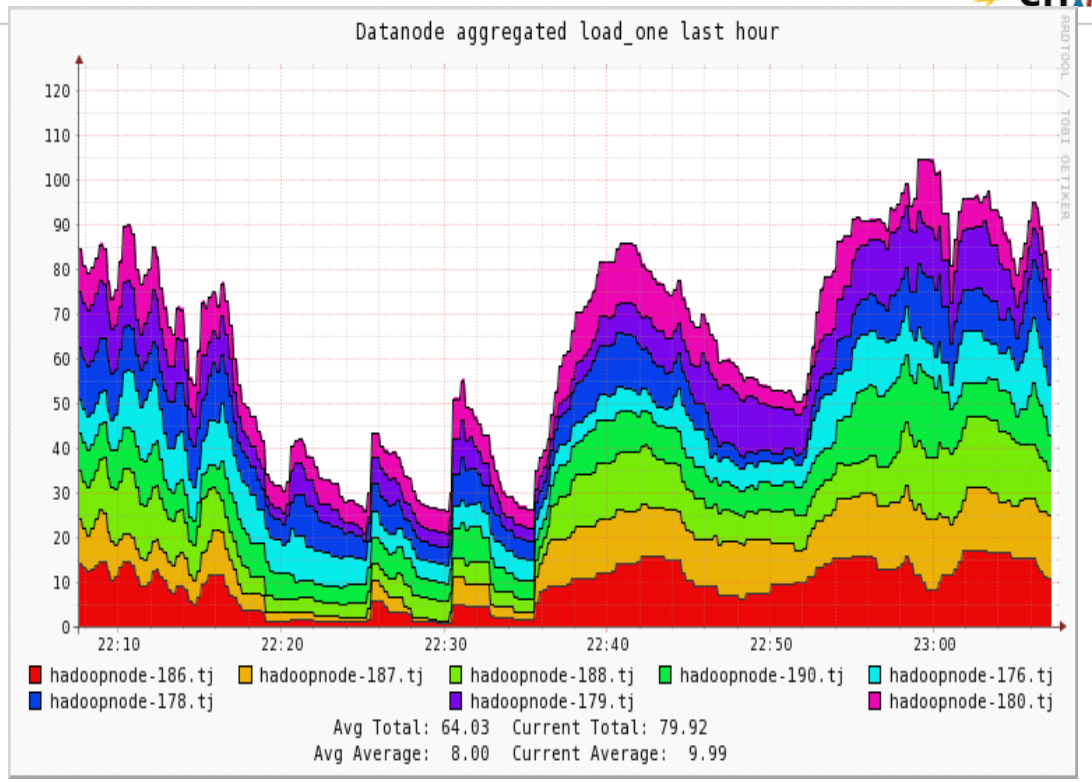
集群资源利用率

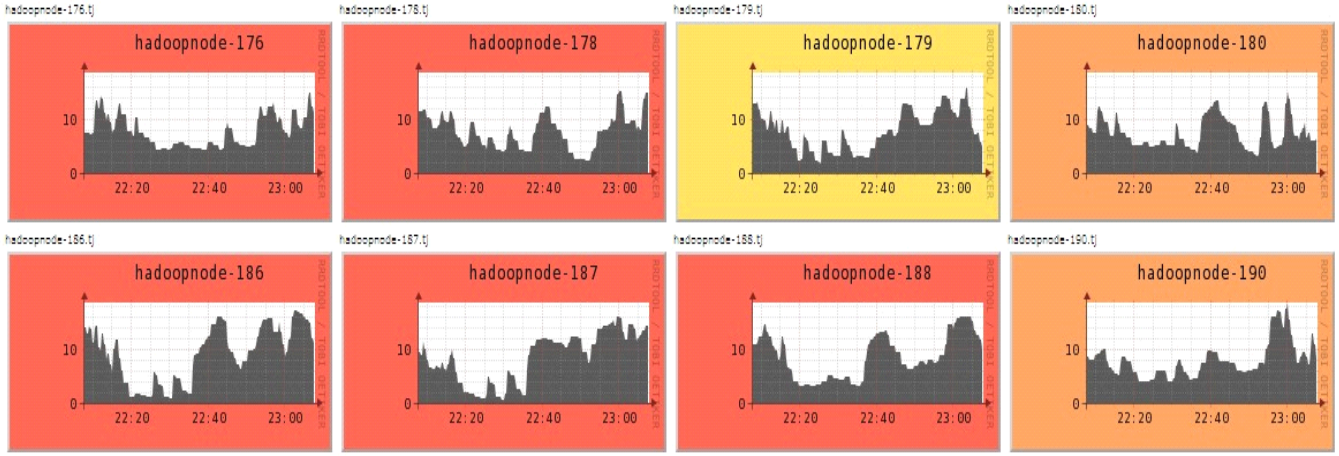
CPUs Total: **64**
Hosts up: **8**
Hosts down: **0**

Current Load Avg (15, 5, 1m):
120%, 134%, 124%
Avg Utilization (last hour):
100%

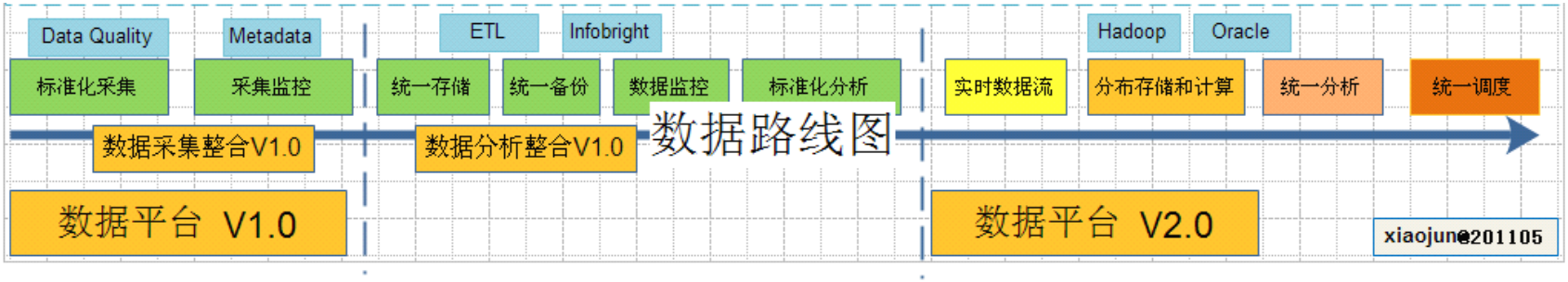
Utilization heatmap







数据平台 技术路线发展



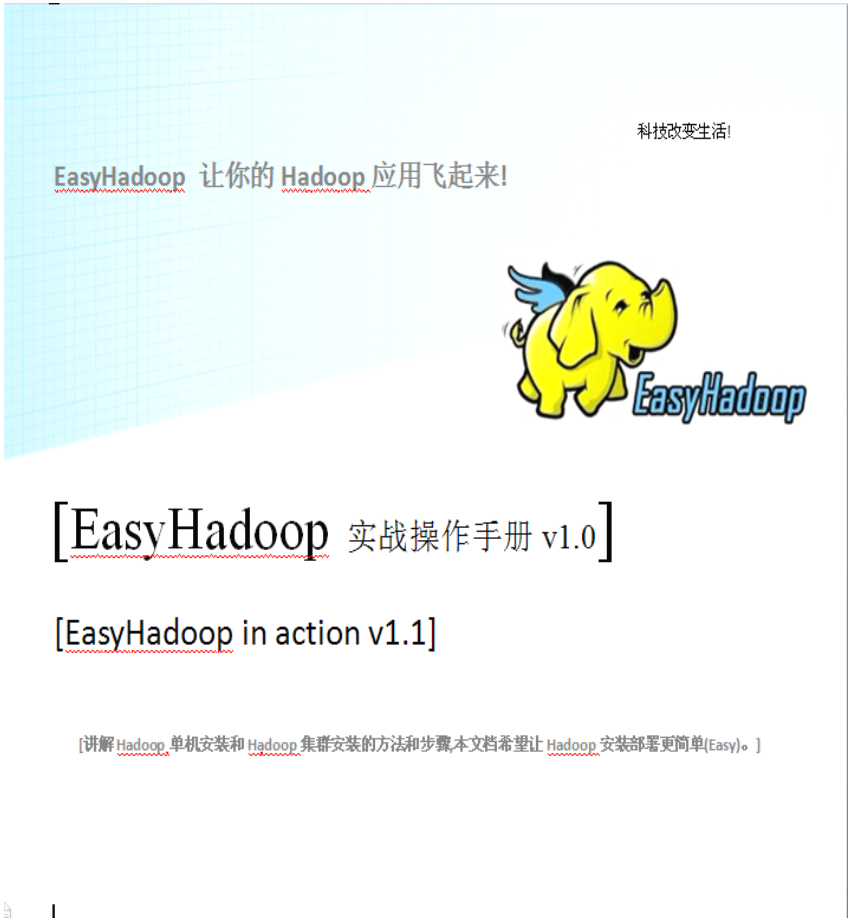
Python Hadoop最佳实践

- 通过Tornado Nginx 接受日志
- 通过Scribe 同步数据
- 使用Python 编写加载和清洗脚本
- 使用ComEtl 通过Hive做ETL
- 参考HappyEtl,Pydoop编写Python Streaming
- 使用CronHub 做定时调度
- 使用phpHiveAdmin 提供自助查询
- 使用 Mysql 存储中间结果
- 通过Tornado+highcharts/gnuplot 提供报表展现
- 使用 Python + Nagios Cacti Ganglia 监控集群
- 整体构建在 Hadoop+Hive+pig 基础平台之上。

- 参加EasyHadoop 聚会学习
- 使用EasyHadoop管理集群

EasyHadoop社区 电子出版物

www.easyhadoop.com



EasyHadoop

EasyHadoop组织了六次技术分享

HadoopCloud 开放平台计划

学习 Hadoop 需要具备三大前提资源。

- 第一:海量的数据集
- 第二:大规模的分析硬件平台
- 第三:大量真实的业务分析需求

- HadoopCloud 提供以上三个平台给用户学习使用。

谢谢!

ComETL

Happyetl

CronHub

多开放一些有趣的开源项目

www.easyhadoop.com