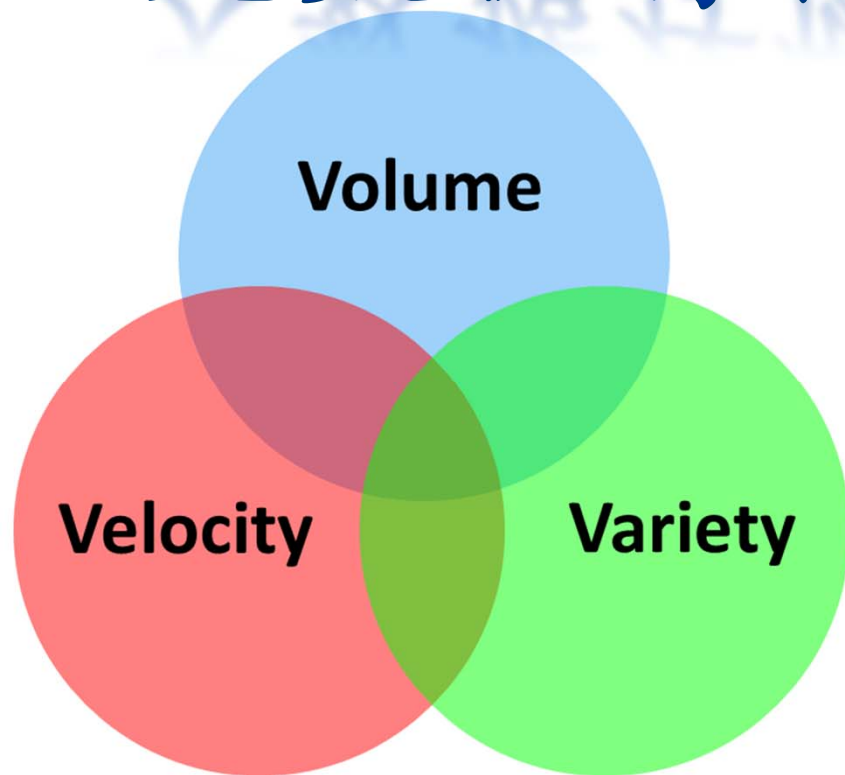


大数据系统与大规模数据分析

# 作业1: 大数据存储系统编程



陈世敏

中科院计算所  
计算机体系结构  
国家重点实验室

©2015-2017 陈世敏

# 课程相关

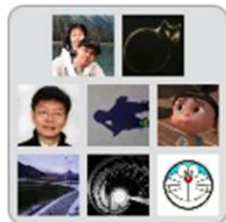
- 成绩分配

- 闭卷考试： 50%
- 作业1+作业2+作业3： 30%
- 大作业： 20%
- 课堂表现： +5%

# 助教

- 杨若雪: yangruoxue@ict.ac.cn
- 赵跃辉: zhaoyuehui14@mails.ucas.ac.cn
- 向超: xiangchao215@mails.ucas.ac.cn
- 王浩博: haobo\_kaoyan@163.com

# 微信群



2017春季大数据系统  
大规模数据分析



该二维码7天内(3月22日前)有效，重新进入将更新

# 作业1安排

- 成绩：占总成绩10%

- 时间

- 发布：2017/3/15(Wed)

- 上交：**2017/4/5(Wed)**，北京时间 6:59pm（共3周）

- 在课程系统中提交

- 晚交

- 最晚：2017/4/12(Wed)，北京时间 6:59pm，将扣除20%成绩

- 之后不再接收，成绩为0

# 分组

- 共分为6个组，每个组的作业题目有一定区别

- 分组方式如下

□ 组号 = (学号最右面6位数字) % 6

□ % 是求余数

- 举例

□ 学号 = 201618013229032

□ 学号最右面6位数字 = 229032

□ 组号 =  $229032 \% 6 = 0$

□ 所以是第0组

# 作业提交的格式

- 文件命名

- ❑ 组号\_学号\_hw1.java

- ❑ 例如：0\_201618013229032\_hw1.java

- 程序中Java class名

- ❑ Hw1GrpX，其中X为组号

- ❑ 例如：Hw1Grp0

- 上述名称注意大小写，自动检查程序会根据学号自动寻找对应的文件，重新命名为Hw1GrpX.java、编译、执行

- ❑ 如果名称不正确，将无法找到或不能执行，就没有成绩

- ❑ 注意在jdk 1.7, hadoop 2.6, hbase 0.98环境下可以正确编译执行

# 上机安排(1)

- 地点

- 计算机学院，4层
- 网络安全教学实验室（447室）：50台
- 云计算教学实验室（432室）：20台

- 机器：联想PC机M6400t, Windows 7/32bit

- 环境：每台机器安装了一个虚拟机，运行Ubuntu Linux 14.04.2, JDK 1.7, Hadoop 2.6.0, HBase 0.98等
- 本作业只需要在单机上构成伪分布环境

- 注：可以在自己的计算机上完成作业



# 上机安排(2)

- 时间

- 周五上午, 8:30-11:50am
- 周五下午, 1:00-4:20pm

- 助教

- 杨若雪, 赵跃辉, 向超, 王浩博

- 上机期间助教的职责

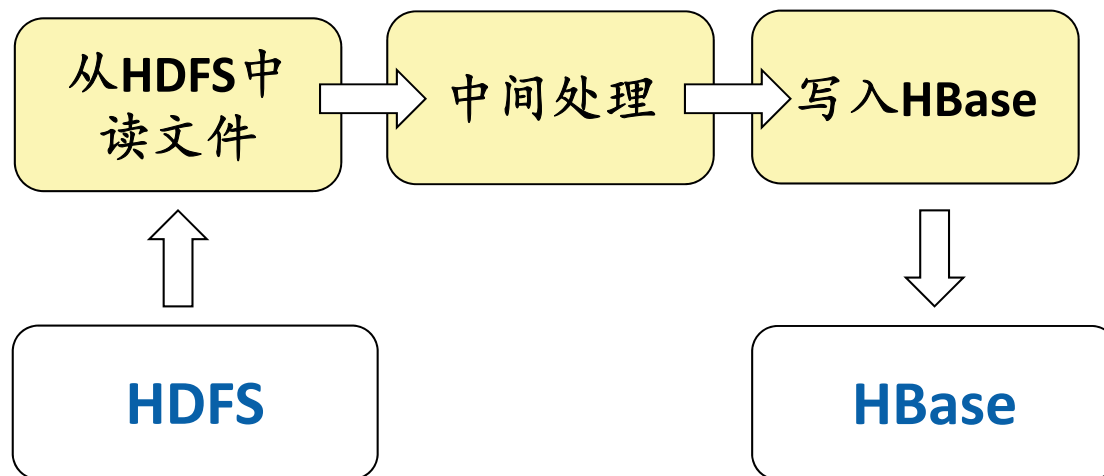
- **管理上机秩序**: 上机前找助教签到, 分配机器; 使用完毕, 找助教签出; 助教负责监督机房秩序(不得喧哗、打闹等)。
- **解答机器使用的问题**: 包括如何开机、如何登录、如何使用编辑器、如何编译和运行程序
- **不包括**: 其它关于作业内容的问题

# 作业内容

- 目的

- 学习HDFS和HBase的基本编程使用
- 巩固课堂讲授的内容

- 总体功能



# 从HDFS中读文件

- 文件格式

- 文本文件
- 每一行是一个关系型记录
- 各个列用|分开

- 例如

- **1|AMERICA|hs use ironic, even requests. s|**
- 这个是TPCH基准测试数据集中region table的一行
- 有3个列
  - 第0列: 1
  - 第1列: AMERICA
  - 第2列: hs use ironic, even requests. S

# hdfs工具

```
$ hdfs dfs -help
```

打印出所有命令的usage信息

```
-copyFromLocal [-f] [-p] [-l] <localsrc> ... <dst> :  
  Identical to the -put command.
```

```
-copyToLocal [-p] [-ignoreCrc] [-crc] <src> ... <localdst> :  
  Identical to the -get command.
```

```
-cat [-ignoreCrc] <src> ... :  
  Fetch all files that match the file pattern <src> and display  
  their content on stdout.
```

```
-ls [-d] [-h] [-R] [<path> ...] :  
  list contents
```

# 程序举例：读一个HDFS文件

```
public class HDFSTest {  
    public static void main(String[] args) throws IOException, URISyntaxException{  
        String file= "hdfs://localhost:9000/文件路径";  
  
        Configuration conf = new Configuration();  
        FileSystem fs = FileSystem.get(URI.create(file), conf);  
        Path path = new Path(file);  
        FSDataInputStream in_stream = fs.open(path);  
  
        BufferedReader in = new BufferedReader(new InputStreamReader(in_stream));  
        String s;  
        while ((s=in.readLine())!=null) {  
            System.out.println(s);  
        }  
  
        in.close();  
        fs.close();  
    }  
}
```

# 中间处理

第0组	Hash join
第1组	Sort-merge join
第2组	Hash based group-by
第3组	Sort based group-by
第4组	Hash based distinct
第5组	Sort based distinct

注：

- 假设所有数据都可以放入内存
- 可以采用Java已有的库实现hash table和sorting

# HBase数据模型

key

<row key, column family: column key, version, value>

- Key包括row key与column两个部分
- 所有row key是按顺序存储的
- 其中column又有column family前缀
  - Column family是需要事先声明的，种类有限（例如~10或~100）
  - 而column key可以有很多
- 具体存储时，每个column family将分开存储（类似列式数据库）

# Key-Value 与 Relational Schema 忽略version部分

- 简单<key, value>可以对应为一个两列的Table

Key	Value
...	...
...	...

- <row key, column family: column key, value>  
每个column family可以对应为一个3列的Table

Row Key	Column family 1' s column key	Value
...	...	...
...	...	...

Row Key	Column family 2' s column key	Value
...	...	...
...	...	...



# 写入HBase

- 本次作业的输出写入HBase，表名是**Result**，注意大小写
- 给定了表名
  - 首先检查这个表是否存在，如果存在，那么删除
  - 创建**Result**表
  - 把结果写入

# hbase shell

```
create 'mytable', 'mycf'
```

创建表, column family

```
put 'mytable', 'abc', 'mycf:a', '123'
```

0 row(s) in 0.0580 seconds

```
put 'mytable', 'def', 'mycf:b', '456'
```

0 row(s) in 0.0060 seconds

```
scan 'mytable'
```

ROW	COLUMN+CELL
-----	-------------

abc	column=mycf:a, timestamp=1427731972925, value=123
-----	---

def	column=mycf:b, timestamp=1427731990058, value=456
-----	---

2 row(s) in 0.0300 seconds

# 举例：HBase create table & Put

```
public class HBaseTest {
    public static void main(String[] args) throws MasterNotRunningException,
        ZooKeeperConnectionException, IOException {
        // create table descriptor
        String tableName= "mytable";
        HTableDescriptor htd = new HTableDescriptor(TableName.valueOf(tableName));

        // create column descriptor
        HColumnDescriptor cf = new HColumnDescriptor("mycf");
        htd.addFamily(cf);

        // configure HBase
        Configuration configuration = HBaseConfiguration.create();
        HBaseAdmin hAdmin = new HBaseAdmin(configuration);

        hAdmin.createTable(htd);
        hAdmin.close();
    }
}
```

# 举例：HBase create table & Put

```
// put "mytable","abc","mycf:a","789"

HTable table = new HTable(configuration,tableName);
Put put = new Put("abc".getBytes());
put.add("mycf".getBytes(),"a".getBytes(),"789".getBytes());
table.put(put);
table.close();
System.out.println("put successfully");
}
}
```

# 中间处理

第0组	Hash join
第1组	Sort-merge join
第2组	Hash based group-by
第3组	Sort based group-by
第4组	Hash based distinct
第5组	Sort based distinct

注：

- 假设所有数据都可以放入内存
- 可以采用Java已有的库实现hash table和sorting

# 第0组/第1组：Join

- 命令行：
  - `java Hw1GrpX R=<file 1> S=<file 2> join:R2=S3 res:R4,S5`
  - 蓝色是可变的参数
- 输入hdfs文件：<file 1>, <file 2>
  - 例如：/hw1/lineitem.tbl等
- Join key: 每个文件有一列为join key
  - 例如：R的第2列和S的第3列（从第0列数起）
- 输出：可以有1到多列
  - 在HBase中，建立Result表，row key是join key，column family是res，column是R4和S5，value是相应的值
  - 例如：一个结果join key= abc, R4= def, S5= ghi  
那么在HBase的Result表中，需要有(row key=abc, res:R4=def) (row key=abc, res:S5=ghi)
- 如何实现join？见讲义

# 相同join key的输出

- 例如：

- 结果包含：

- join key= abc, R4= def, S5= ghi
    - join key= abc, R4= 123, S5= 456
    - join key= abc, R4= 789, S5= ghi

- 那么输出到Hbase:

- (row key=abc, res:R4=def) (row key=abc, res:S5=ghi)
    - (row key=abc, res:R4.1=123) (row key=abc, res:S5.1=456)
    - (row key=abc, res:R4.2=789) (row key=abc, res:S5.2=ghi)

- 注意：groupby和distinct时每个row key仅有一个结果，所以不需要上述

## 第2组/第3组： Group-by

- 命令行：
  - `java Hw1GrpX R=<file> groupby:R2 res:count,avg(R3),max(R4)`
  - 蓝色是可变的参数
- 输入文件： <file>
  - 例如： /hw1/lineitem.tbl
- Group by key: 只有一列
  - 例如： R的第2列 (从第0列数起)
- 输出： 可以有1~多列， 数值列
  - 三种形式为(a) count, (b) avg(列), (c) max(列)
  - 在HBase中， 建立Result表， row key是group by key, column family是res, column是count、avg(R3)、max(R4), value是相应的值
  - 例如： 一个结果groupby key= abc, count=3, avg(R3)= 10,max(R4)=20  
那么在HBase的Result表中， 需要有(row key=abc, res:count=3) (row key=abc, res:avg(R3)=10) (row key=abc, res:max(R4)=20)
- 注： count和max结果是准确值， avg保留小数点后2位数



# Group by实现

- Hash based

- 建立一个hash table
- Key= group by key
- Value= 需要统计的信息
  - Count: 目前的计数
  - Avg: 目前的sum和count
  - Max: 目前的最大值
- 把输入都使用hash table完成统计，最后扫描输出hash table中的所有项

- Sort based

- 根据group by key 排序
- 然后同一个group的都会在一起
- 统计输出

## 第4组/第5组：Distinct

- 命令行：

- `java Hw1GrpX R=<file> select:R1,gt,5.1 distinct:R2,R3,R5`

- 蓝色是可变的参数

- 输入文件：<file>

- 例如：/hw1/lineitem.tbl等

- 选择：只有一列，数值列

- 6种形式(a)列,gt,值, (b)列,ge,值, (c)列,eq,值, (d)列,ne,值, (e)列,le,值, (f)列,lt,值

- 涵义：> gt; >= ge; == eq; != ne; le <=; lt <

- 例如：R的第1列大于5.1 （从第0列数起）

- 输出：可以有1~多列

- 每种组合只输出一次

- 在HBase中，建立Result表，row key是序号，column family是res，column是R2,R3和R5，value是相应的值

# 举例

R0	R1	R2	R3	R4	R5
100	3	good	nice	ok	12
101	6	abc	def	better	10
102	9	abc	def	best	10
103	12	abc	def	nicest	8

- 第0个结果(abc, def, 10), 那么在HBase的Result表中, 需要有(row key=0, res:R2=abc) (row key=0, res:R3=def)(row key=0, res:R5=10)
- 第1个结果(abc, def, 8), 那么在HBase的Result表中, 需要有(row key=1, res:R2=abc) (row key=1, res:R3=def)(row key=1, res:R5=8)

# Distinct实现

- Selection: 每个记录依次进行比较
- Hash based
  - 建立一个hash table
  - Key= distinct所有key (例如: R2,R3,R5)
  - Value= 空
  - 把输入都放入hash table一次且仅一次, 最后扫描输出hash table中的所有项
- Sort based
  - 根据distinct key 排序
  - 然后相同的都会在一起
  - 输出

# 以TPCH数据为基础的例子

- Join

- ❑ `java Hw1GrpX R=/hw1/lineitem.tbl S=/hw1/orders.tbl  
join:R0=S0 res:S1,R1,R5`
- ❑ `java Hw1GrpX R=/hw1/lineitem.tbl S=/hw1/part.tbl join:R1=S0  
res:S1,S3,R5`

- Groupby

- ❑ `java Hw1GrpX R=/hw1/lineitem.tbl groupby:R2  
res:count,sum(R5)`
- ❑ `java Hw1GrpX R=/hw1/orders.tbl groupby:R1 res:count,avg(R3)`

- Distinct

- ❑ `java Hw1GrpX R=/hw1/part.tbl select:R7,gt,1800  
distinct:R3,R4,R5`
- ❑ `java Hw1GrpX R=/hw1/lineitem.tbl select:R4,lt,5  
distinct:R13,R14,R8,R9`

# 注意事项

- 命名

- 程序名、类名、表名、Column Family名、列名等
- 注意大小写，必须按照规定

- 程序注释

- 注意程序格式，要求有Javadoc要求的注释，没有就-1
- 只能用英文

- 严禁抄袭

- 会有自动检查程序（也会比较去年的作业）
- 一旦发现，抄袭各方均为0分