在我们的数据中，我们可以看到每次广告展现都是一行数据，如果我们想知道一个站点（site）的历史点击率，我们需要读取一段时间数据进行统计。但是如果历史点击率是一个经常查询的条件，而每次查询都要计算的话则会耗费大量计算资源。因此我们需要一个数据库进行存储，这样只需要一次计算，以后查询就只需要按照key进行检索即可。

**Step1： Hbase Shell**

列举数据库中所有表

> list

创建表

create 命令

> create 'table\_name', 'cf1', 'cf2'

其中的 cf1 和 cf2 为列族名 1，列族名 2，列族需要在见表时确定，列则不需要， Column Family 是 Schema 的一部分，设计时就需要考虑。

删除表

在删除表之前需要使用 disable 命令，让表失效。在修改表结构时，也需要先执行此命令

> disable "table\_name"

删除表使用 drop 命令

> drop 'table\_name'

测试表是否存在

> exists 'table\_name'

会显示表是否存在：

hbase(main):002:0> exists 'test'

Table test does exist

0 row(s) in 0.2650 seconds

显示表结构

describe 命令查看表结构，显示 HBase 表 schema，以及 column family 设计

> describe 'table\_name'

使表有效

enable 命令，和 disable 命令对应

> enable 'table\_name'

修改表结构

alter 修改表的结构，新增列族，删除列族。在修改之前要先 disable ，修改完成后再 enable

新增列族

> alter 'table\_name', '列族'

删除列族

> alter 'table\_name', {name=>‘列族’, METHOD=>'delete'}

增加记录

put 命令

插入数据，对于同一个 rowkey，如果执行两次 put，则认为是更新操作

> put 'table\_name', 'rowkey', '列族名 1: 列名 1', 'value'

> put 't1', 'r1', 'c1', 'value', ts1

一般情况下 ts1（时间戳） 可以省略， Column 可以动态扩展，每行可以有不同的 Column。

查询表行数

计算表的行数，count 一般比较耗时，使用

> count 'table\_name'

查询所有 rowkey

> count 'table\_name', { INTERVAL => 1 }

获取指定 rowkey 的指定列族所有的数据

> get 'table\_name', 'rowkey', '列族名'

获取指定 rowkey 的所有数据

> get 'table\_name', 'rowkey'

获取指定时间戳的数据

> get 'table\_name', 'rowkey', {COLUMN=>'列族名：列', TIMESTAMP=>1373737746997}

删除指定 rowkey 的指定列族的列名数据

> delete 'table\_name', 'rowkey', '列族名：列名'

删除指定 rowkey 指定列族的数据

> delete 'table\_name', 'rowkey', '列族名‘

删除整行数据

> deleteall 'table\_name', ’rowkey'

全表扫描

> scan

hbase(main):043:0> scan 'test', {VERSIONS => 12}

ROW COLUMN+CELL

rowkey1 column=cf:a, timestamp=1487295285291, value=value 3

rowkey1 column=cf:a, timestamp=1487294839168, value=value 2

rowkey1 column=cf:a, timestamp=1487294704187, value=value 1

hbase shell 脚本

shell 命令，把所有的 hbase shell 命令写到一个文件内，类似与 Linux shell 脚本顺序执行所有命令，可以使用如下方法执行。

> hbase shell test.hbaseshell

**Step2：使用Python进行Hbase数据读写**

通过行键获取数据

1. **import** hbase
3. zk = 'sis3.ustcdm.org:2181,sis4.ustcdm.org:2181'
5. **if** \_\_name\_\_ == '\_\_main\_\_':
6. with hbase.ConnectionPool(zk).connect() as conn:
7. table = conn['mytest']['videos']
8. row = table.get('00001')
9. **print**(row)
10. exit()

扫描表

1. **import** hbase
3. zk = 'sis3.ustcdm.org:2181,sis4.ustcdm.org:2181'
5. **if** \_\_name\_\_ == '\_\_main\_\_':
6. with hbase.ConnectionPool(zk).connect() as conn:
7. table = conn['mytest']['videos']
8. **for** row **in** table.scan():
9. **print**(row)
10. exit()

写入一条记录

1. **import** hbase
3. zk = 'sis3.ustcdm.org:2181,sis4.ustcdm.org:2181'
5. **if** \_\_name\_\_ == '\_\_main\_\_':
6. with hbase.ConnectionPool(zk).connect() as conn:
7. table = conn['mytest']['videos']
8. table.put(hbase.Row(
9. '0001', {
10. 'cf:name': b'Lily',
11. 'cf:age': b'20'
12. }
13. ))
14. exit()

**Step3：Hbase保存用户数据**

1. #!/usr/bin/env python
2. **import** hbase
4. zk = '127.0.0.1:2181'
6. f = open('train-small','r')
7. st = set()
8. dt = dict()
9. **while** True:
10. line = f.readline()
11. **if** **not** line:
12. **break**
13. line = line.strip()
14. keys = line.split(',')
15. st.add(keys[8])
16. f.close()
18. cnt = 0
20. **for** key **in** st:
21. dt['map:'+key] = str(cnt).encode('utf-8')
22. cnt = cnt + 1
24. # print(dt)
26. with hbase.ConnectionPool(zk).connect() as conn:
27. table = conn['kaggle']['key']
28. table.put(hbase.Row(
29. 'app\_id', dt
30. ))