**1、实验数据集说明**

数据集压缩包是淘宝2015年双11前6个月(包含双11)的交易数据user\_log.csv，约有5000万条日志，其中的字段定义如下：

1. user\_id | 买家id

2. item\_id | 商品id

3. cat\_id | 商品类别id

4. merchant\_id | 卖家id

5. brand\_id | 品牌id

6. month | 交易时间:月

7. day | 交易事件:日

8. action | 行为,取值范围{0,1,2,3},0表示点击，1表示加入购物车，2表示购买，3表示关注商品

9. age\_range | 买家年龄分段：1表示年龄<18,2表示年龄在[18,24]，3表示年龄在[25,29]，4表示年龄在[30,34]，5表示年龄在[35,39]，6表示年龄在[40,49]，7和8表示年龄>=50,0和NULL则表示未知

10. gender | 性别:0表示女性，1表示男性，2和NULL表示未知

11. province| 收货地址省份

**注意该数据集未经清洗，其中有可能有null字段，会导致查询失败。可自行编写Linux脚本或者Python程序删除有null字段的行。**

**2、使用Hive完成以下功能：**

1、计算出表内有多少条行数据；

2、查询双11那天有多少人购买了商品（要去掉同一个user\_id的重复购买）；

3、给定品牌brand\_id为2661，求双11当天此品牌商品的销售数量；

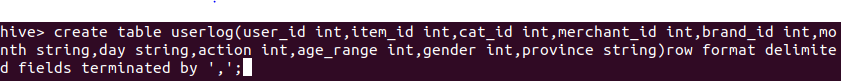
4、查询双11那天女性购买商品的数量；

5、根据收货地址，统计各个省双11那天用户下单的总次数，并进行排序。

6、 思考题：对比在Hive和MySQL中以上查询语句的执行时间，分析Hive和MySQL查询效率差异产生的原因。

**3、Hive数据导入和数据查询的主要语句和输出结果示例；**

* **建表语句：**



* **数据导入：**



* **计算表内有多少行数据：**

语句：



结果：

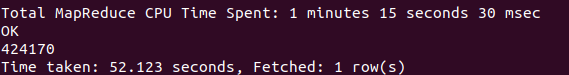


* **查询双11那天有多少人购买了商品（要去掉同一个user\_id的重复购买）**

语句：



结果：

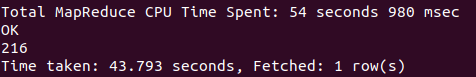


* **给定品牌brand\_id为2661，求双11当天此品牌商品的销售数量：**

语句：



结果：

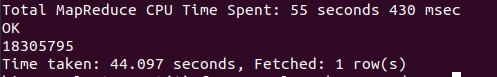


* **查询双11那天女性购买商品的数量：**

语句：



结果：

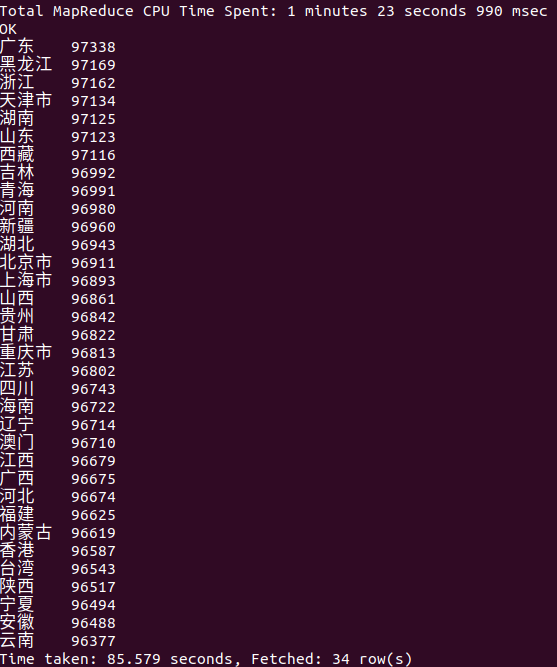


**根据收货地址，统计各个省双11那天用户下单的总次数，并进行排序：**

语句：



结果：



* **对比在Hive和MySQL中以上查询语句的执行时间，分析Hive和MySQL查询效率差异产生的原因**

索引方面，Hive在加载数据的过程中不会对数据进行任何处理，甚至不会对数据进行扫描，因此也没有对数据中的某些Key建立索引。Hive要访问数据中满足条件的特定值时，需要暴力扫描整个数据，因此访问延迟较高。由于 MapReduce 的引入， Hive 可以并行访问数据，因此即使没有索引，对于大数据量的访问，Hive 仍然可以体现出优势。数据库中，通常会针对一个或者几个列建立索引，因此对于少量的特定条件的数据的访问，数据库可以有很高的效率，较低的延迟。

执行方面，Hive 在查询数据的时候，由于没有索引，需要扫描整个表，因此延迟较高。另外一个导致 Hive 执行延迟高的因素是 MapReduce框架。由于MapReduce 本身具有较高的延迟，因此在利用MapReduce 执行Hive查询时，也会有较高的延迟。相对的，数据库的执行延迟较低。