用Hive分析“余额宝”躺着赚大钱背后的逻辑

## 1. 创建原始表

CREATE TABLE IF NOT EXISTS stock(  
 tradedate STRING,  
 tradetime STRING,  
 stockid INT,  
 buyprice DOUBLE,  
 buysize INT,  
 sellprice DOUBLE,  
 sellsize INT  
) ROW FORMAT DELIMITED FIELDS TERMINATED BY ','  
STORED AS TEXTFILE;  
  
LOAD DATA INPATH '/root/share/hadoop/total.csv' INTO TABLE stock;

## 2. 创建分区表

CREATE TABLE IF NOT EXISTS stock\_partition(  
 tradetime STRING,  
 stockid INT,  
 buyprice DOUBLE,  
 buysize INT,  
 sellprice DOUBLE,  
 sellsize INT  
) PARTITIONED BY (tradedate STRING) (1)  
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','  
STORED AS TEXTFILE;  
  
set hive.exec.dynamic.partition.mode=nonstrict;  
  
# 导入数据  
INSERT OVERWRITE TABLE stock\_partition partition(tradedate) SELECT tradetime, stockid, buyprice, buysize, sellprice, sellsize, tradedate FROM stock distribute by tradedate;

|  |  |
| --- | --- |
| **1** | 分区字段不应该出现在建表语句中 |

## 3. 自定义 Max,Min 函数

Min

package hive;  
  
import org.apache.hadoop.hive.ql.exec.UDF;  
  
public class Min extends UDF {  
 public Double evaluate(Double a, Double b) {  
 if (a == null)  
 a = 0.0;  
 if (b == null)  
 b = 0.0;  
 if (a >= b) {  
 return b;  
 } else {  
 return a;  
 }  
 }  
}

Max

package hive;  
  
import org.apache.hadoop.hive.ql.exec.UDF;  
  
public class Max extends UDF {  
 public Double evaluate(Double a, Double b) {  
 if (a == null)  
 a = 0.0;  
 if (b == null)  
 b = 0.0;  
 if (a >= b) {  
 return a;  
 } else {  
 return b;  
 }  
 }  
}

## 4. 注册临时函数

add jar /root/share/hadoop/maxUDF-1.0.jar;  
add jar /root/share/hadoop/minUDF-1.0.jar;  
  
create temporary function maxprice as 'hive.Max';  
create temporary function minprice as 'hive.Min';

## 5. 统计每分钟均价

select stockid,tradedate,substring(tradetime,0,4),sum(buyprice+sellprice)/count(stockid)\*2 from stock\_partition where stockid = '204001' group by stockid,tradedate,substring(tradetime,0,4);  
  
  
204001 20130722 1116 14.382499999999999  
204001 20130722 1117 14.32125  
204001 20130722 1118 14.267  
204001 20130722 1119 14.203333333333331  
204001 20130722 1120 14.185  
204001 20130722 1121 14.146363636363638  
204001 20130722 1122 14.145000000000001  
204001 20130722 1123 14.159090909090912  
204001 20130722 1124 14.17181818181818  
204001 20130722 1125 14.187272727272726  
204001 20130722 1126 14.143  
204001 20130722 1127 14.157  
204001 20130722 1128 14.116666666666667  
204001 20130722 1129 14.196363636363635