221275043 谢俊言 lab3

I. 拉取hive4.0.0镜像，启动容器

```
1. docker pull apache/hive:4.0.0
2. export HIVE_VERSION=4.0.0
3. docker run -d -p 10000:10000 -p 10002:10002 --env
SERVICE_NAME=hiveserver2 --name hive4 apache/hive:${HIVE_VERSION}
4. docker exec -it hive-server2 beeline -u 'jdbc:hive2://localhost:10000/'
```

II. 数据准备

```
1. docker cp /User/jyxie/user_profile.csv hive4:/tmp/user_profile.csv
2. docker cp /User/jyxie/user_balance.csv hive4:/tmp/user_balance.csv
```

III. 任务1：创建user_profile_table和user_balance_table并载入数据

```
1.  CREATE TABLE user_profile_table (
    user_id STRING,
    sex STRING,
    city STRING,
    constellation STRING
    )
    ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ','
    STORED AS TEXTFILE
    TBLPROPERTIES ("skip.header.line.count"="1");

2.  CREATE TABLE user_balance_table (
    user_id STRING,
    report_date STRING,
    tBalance DOUBLE,
    yBalance DOUBLE,
    total_purchase_amt DOUBLE,
    direct_purchase_amt DOUBLE,
    purchase_bal_amt DOUBLE,
    purchase_bank_amt DOUBLE,
    total_redeem_amt DOUBLE,
    consume_amt DOUBLE,
    transfer_amt DOUBLE,
    tftobal_amt DOUBLE,
    tftocard_amt DOUBLE,
    share_amt DOUBLE,
    category1 DOUBLE,
    category2 DOUBLE,
```

```
    category3 DOUBLE,
    category4 DOUBLE
    )
    ROW FORMAT DELIMITED
    FIELDS TERMINATED BY ','
    STORED AS TEXTFILE
    TBLPROPERTIES ("skip.header.line.count"="1");

3.  LOAD DATA LOCAL INPATH '/tmp/user_profile.csv' INTO TABLE
user_profile_table;
    LOAD DATA LOCAL INPATH '/tmp/user_balance.csv' INTO TABLE
user_balance_table;

4.  SELECT * FROM user_profile_table LIMIT 5;
    SELECT * FROM user_balance_table LIMIT 5;
    SELECT COUNT(*) FROM user_profile_table;
    SELECT COUNT(*) FROM user_balance_table;
```

依次得到如下四张图：

user_profile_table.user_id	user_profile_table.sex	user_profile_table.city	user_profile_table.constellation
2	1	6411949	狮子座
12	1	6412149	摩羯座
22	1	6411949	双子座
23	1	6411949	双鱼座
25	1	6481949	双鱼座

user_balance_table.user_id	user_balance_table.report_date	user_balance_table.tbalance	user_balance_table.ybalance	user_balance_table.total_purchase_amt	user_balance_table.direct_purchase_amt	user_balance_table.purchase_bal_amt	user_balance_table.purchase_bank_amt	user_balance_table.total_redeem_amt	user_balance_table.consume_amt	user_balance_table.transfer_amt	user_balance_table.tftobal_amt	user_balance_table.tftocard_amt	user_balance_table.share_amt	user_balance_table.category1	user_balance_table.category2	user_balance_table.category3	user_balance_table.category4
1	20140805	20385.0	20383.0	2.0	0.0	0.0	0.0	0.0	0.0	2.0	NULL	NULL	NULL	NULL	NULL	NULL	NULL
1	20140808	20391.0	20389.0	2.0	0.0	0.0	0.0	0.0	0.0	2.0	NULL	NULL	NULL	NULL	NULL	NULL	NULL
1	20140811	20397.0	20395.0	2.0	0.0	0.0	0.0	0.0	0.0	2.0	NULL	NULL	NULL	NULL	NULL	NULL	NULL
1	20140814	20403.0	20401.0	2.0	0.0	0.0	0.0	0.0	0.0	2.0	NULL	NULL	NULL	NULL	NULL	NULL	NULL
1	20140817	20409.0	20407.0	2.0	0.0	0.0	0.0	0.0	0.0	2.0	NULL	NULL	NULL	NULL	NULL	NULL	NULL

_c0
28041

_c0
2840421

IV. 任务2: 基本数据查询

任务2.1：查询星座用户数量

```
SELECT
  constellation,
  COUNT(*) AS user_count
FROM
```

```
  user_profile_table
GROUP BY
  constellation
ORDER BY
  user_count DESC;
```

结果如下图：

constellation	user_count
天秤座	2910
天蝎座	2640
处女座	2497
狮子座	2387
射手座	2336
水瓶座	2336
摩羯座	2280
双鱼座	2265
巨蟹座	2202
金牛座	2108
双子座	2088
白羊座	1992

任务2.2：查询特定日期的资金流入和流出情况

```
1.  CREATE TABLE daily_flow_table AS
    SELECT
    report_date,
    SUM(total_purchase_amt) AS total_inflow,
    SUM(total_redeem_amt) AS total_outflow
    FROM
    user_balance_table
    GROUP BY
    report_date;
2.  SELECT * FROM daily_flow_table LIMIT 20;
```

结果如下图：

daily_flow_table.report_date	daily_flow_table.total_inflow	daily_flow_table.total_outflow
20130701	3.2488348E7	5525022.0
20130702	2.903739E7	2554548.0
20130703	2.727077E7	5953867.0
20130704	1.8321185E7	6410729.0
20130705	1.1648749E7	2763587.0
20130706	3.6751272E7	1616635.0
20130707	8962232.0	3982735.0
20130708	5.7258266E7	8347729.0
20130709	2.6798941E7	3473059.0
20130710	3.0696506E7	2597169.0
20130711	4.4075197E7	3508800.0
20130712	3.4183904E7	8492573.0
20130713	1.5164717E7	3482829.0
20130714	2.2615303E7	2784107.0
20130715	4.8128555E7	1.3107943E7
20130716	5.0622847E7	1.1864981E7
20130717	2.9015682E7	1.0911513E7
20130718	2.4234505E7	1.1765356E7
20130719	3.3680124E7	9244769.0
20130720	2.0439079E7	4601143.0

V. 任务3：数据聚合分析

任务3.1：按星座统计总购买量和总赎回量

```sql
SELECT
  up.constellation,
  SUM(ub.total_purchase_amt) AS total_purchase_amt,
  SUM(ub.total_redeem_amt) AS total_redeem_amt
FROM
  user_profile_table up
JOIN
  user_balance_table ub
ON
  up.user_id = ub.user_id
GROUP BY
  up.constellation;
```

结果如下图：

up.constellation	total_purchase_amt	total_redeem_amt
双子座	6.925452079E9	5.348014629E9
双鱼座	6.926003786E9	5.361595639E9
处女座	6.745578008E9	5.293924154E9
天秤座	9.88203188E9	7.989306193E9
天蝎座	1.0046042993E10	7.958903337E9
射手座	8.833197253E9	6.88952704E9
巨蟹座	7.359871686E9	5.992055191E9
摩羯座	7.418393917E9	5.724242577E9
水瓶座	8.096474293E9	6.492539527E9
狮子座	7.069134334E9	5.5723382E9
白羊座	6.655483821E9	5.012246637E9
金牛座	6.633419406E9	5.083604446E9

任务3.2:按城市统计2014.3.1的平均余额

```
SELECT
  up.city,
  AVG(ub.tBalance) AS avg_balance
FROM
  user_profile_table up
JOIN
  user_balance_table ub
ON
  up.user_id = ub.user_id
WHERE
  ub.report_date = '20140301'
GROUP BY
  up.city
ORDER BY
  avg_balance DESC
LIMIT 10;
```

结果如下图:

up.city	avg_balance
6281949	2795923.837298216
6301949	2650775.0664451825
6081949	2643912.7566638007
6481949	2087617.2136986302
6411949	1929838.5617977527
6412149	1896363.471625767
6581949	1526555.5551020408

VI.任务4:复杂查询与分析

任务4.1:活跃用户分析

```
SELECT COUNT(DISTINCT user_id) AS active_user_count
FROM (
  SELECT
    user_id,
    COUNT(DISTINCT report_date) AS active_days
  FROM
    user_balance_table
  WHERE
    report_date BETWEEN '20140801' AND '20140831'
  GROUP BY
    user_id
  HAVING
    active_days >= 5
) AS temp;
```

结果如下图:

active_user_count
12767

任务4.2:统计每个城市总流量前3高的用户

```
1.  CREATE VIEW user_monthly_flow AS
    SELECT
```

```
    user_id,
    SUM(total_purchase_amt + total_redeem_amt) AS total_flow
    FROM
    user_balance_table
    WHERE
    report_date BETWEEN '20140801' AND '20140831'
    GROUP BY
    user_id;

2.  SELECT
    city,
    user_id,
    total_flow
    FROM (
    SELECT
        up.city,
        umf.user_id,
        umf.total_flow,
        ROW_NUMBER() OVER (PARTITION BY up.city ORDER BY umf.total_flow
DESC) AS rank
    FROM
        user_monthly_flow umf
    JOIN
        user_profile_table up
    ON
        umf.user_id = up.user_id
    ) sub
    WHERE
    sub.rank <= 3;
```

结果如下图:

city	user_id	total_flow
6081949	27235	1.0847568E8
6081949	27746	7.6065458E7
6081949	18945	5.5304049E7
6281949	15118	1.49311909E8
6281949	11397	1.24293438E8
6281949	25814	1.04428054E8
6301949	2429	1.09171121E8
6301949	26825	9.537403E7
6301949	10932	7.4016744E7
6411949	662	7.5162566E7
6411949	21030	4.9933641E7
6411949	16769	4.9383506E7
6412149	22585	2.00516731E8
6412149	14472	1.3826279E8
6412149	25147	7.0594902E7
6481949	12026	5.1161825E7
6481949	670	4.9626204E7
6481949	14877	3.4488733E7
6581949	9494	3.8854436E7
6581949	26876	2.3449539E7
6581949	21761	2.113644E7

VII. 实验感受

遇到的问题：hive3.1.3与java11兼容性存在着一些问题

解决方法：使用官方的hive4.0.0 docker镜像