

《数据仓库》练习03

说明

- 在D盘创建考生文件夹：命名为“《数仓》+练习03+学号后2位”
- 在IDEA里创建sql文件，名称为 练习03.sql
- 在IDEA里答题时，每道题前必须加上题号的备注，如‘-- 1.2’。
- 答题结束后，需要提交以下资料
 - 1.需在IDEA中导出 练习03.sql 文件为 HTML 文件，并保存在考生文件夹
 - 2.需对HDFS在web UI 中的 /user/hive/exam/exam_db3 路径进行截图，命名“HDFS.jpg”保存在考生文件夹
 - 3.需对第六题的6道题的查询结果进行截图【截图的范围包括整个窗口和运行结果】，并按题号命名保存在考生文件夹
 - 4.将考生文件夹打包压缩按监考老师的要求进行提交。

一. 启动环境

```
# 1.1
能启动VMware和FinalShell连接虚拟机即可得分

# 1.2 启动hadoop集群
start-dfs.sh
start-yarn.sh

# 1.3 启动Hive服务
nohup hive --service metastore &
nohup hive --service hiveserver2 &

# 1.4 检查进程
jps -m

# 1.5 IDEA能正确连接hive
```

二. 创建数据库

- 删除并创建一个名为 exam_db3 的数据库，并将其位置设置为 /user/hive/exam/exam_db3。

```
-- 2.1 删除 exam_db3 数据库
drop database if exists exam_db3 cascade;

-- 2.2 创建 exam_db3 数据库
create database if not exists exam_db3
    location '/user/hive/exam/exam_db3';

-- 2.3 选择 exam_db3 数据库
use exam_db3;
```

三. 创建数据表

在 `exam_db3` 数据库中创建四个表：

- 3.1 `customers` 表结构：

列	类型	描述
customer_id	int	顾客ID
customer_name	string	顾客姓名
address	string	顾客地址

- 3.2 `product` 表结构：

列	类型	描述
product_id	int	商品ID
name	string	商品名称
price	double	商品价格
category	string	商品类别

- 3.3 `orders` 表结构：

列	类型	描述
order_id	int	订单ID
customer_id	string	客户ID
product_id	int	商品ID
amount	double	销售额
purchase_date	date	购买日期

- 3.4 分区表 `order_partitioned`，该表以 `purchase_date` 分区列进行分区，分区列为日期类型。

列	类型	描述
order_id	int	订单ID
customer_id	string	客户ID
product_id	int	商品ID

-- 3.1 创建 customers 数据表

```
CREATE TABLE if not exists customers (
  customer_id int COMMENT '顾客ID',
  customer_name string COMMENT '顾客姓名',
  address string COMMENT '顾客地址'
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

-- 3.2 创建 product 数据表

```
CREATE TABLE if not exists product (
  product_id int COMMENT '商品ID',
  name string COMMENT '商品名称',
  price double COMMENT '商品价格',
  category string COMMENT '商品类别'
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

-- 3.3 创建 orders 数据表

```
CREATE TABLE if not exists orders (
  order_id int COMMENT '订单ID',
  customer_id string COMMENT '客户ID',
  product_id int COMMENT '商品ID',
  amount double COMMENT '销售额',
  purchase_date date COMMENT '购买日期'
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

-- 3.4 创建按 purchase_date 分区的order_partitioned分区表

```
CREATE TABLE if not exists order_partitioned (
  order_id int COMMENT '订单ID',
  customer_id string COMMENT '客户ID',
  product_id int COMMENT '商品ID',
  amount double COMMENT '销售额'
)
PARTITIONED BY (purchase_date date)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

四. 修改数据表

-- 4.1 在 `customers` 数据表中添加一个列名为 `phone` 的列，类型为字符串。

```
ALTER TABLE customers
    ADD COLUMNS (phone STRING COMMENT '电话号码');
```

-- 4.2 把 `product` 表中将列 `name` 改名为 `product_name`。

```
ALTER TABLE product CHANGE name product_name STRING;
```

五. 加载数据

/* 5.1 将数据 `customers.txt`、`product.txt`、`order.txt` 上传到master节点上的
`/opt/apps/hive/exam/`下 */

-- 5.2 把 `customers.txt` 数据加载到 `customers` 表中；

```
LOAD DATA LOCAL INPATH '/opt/apps/hive/exam/customers.txt' OVERWRITE INTO TABLE
customers;
```

-- 5.3 把 `product.txt` 数据加载 `product` 表中；

```
LOAD DATA LOCAL INPATH '/opt/apps/hive/exam/product.txt' OVERWRITE INTO TABLE
product;
```

-- 5.4.1 把 `orders.txt` 从本地上传到HDFS中，路径为 `/user/hive/exam/exam_db3`；

```
hdfs dfs -put /opt/apps/hive/exam/orders.txt /user/hive/exam/exam_db3
```

-- 5.4.2 把路径 `/user/hive/exam/exam_db3` 下的 `orders.txt` 数据加载到 `orders` 表中。

```
LOAD DATA INPATH '/user/hive/exam/exam_db3/orders.txt' OVERWRITE INTO TABLE
orders;
```

-- 5.5 从表 `orders` 执行动态分区插入数据到 `order_partitioned` 分区表中

-- 注意：该表是按照 `purchase_date` 进行分区。

```
SET hive.exec.dynamic.partition=true;
```

```
SET hive.exec.dynamic.partition.mode=nonstrict;
```

```
INSERT OVERWRITE TABLE order_partitioned PARTITION(purchase_date)
```

```
SELECT order_id, customer_id, product_id, amount , purchase_date FROM orders;
```

六. 查询数据

-- 6.1 统计每个商品类别的商品数量

```
SELECT category, COUNT(*) AS product_count
FROM product
GROUP BY category;
```

-- 6.2 查询销售额最低的商品名称和销售额

```
SELECT p.product_name AS product_name, SUM(o.amount) AS total_sales
FROM orders o
      JOIN product p ON o.product_id = p.product_id
GROUP BY p.product_name
ORDER BY total_sales ASC
LIMIT 1;
```

-- 6.3 查询类别为“服装”的商品信息

```
SELECT *
FROM product
WHERE category = '服装';
```

-- 6.4 将表orders按product_id划分到3个节点（reducer）上，并在每个节点内按 amount 字段降序排序。

```
SET mapreduce.job.reduces=3;
```

```
SELECT order_id, product_id, amount
FROM orders
      DISTRIBUTE BY product_id
      SORT BY amount DESC;
```

-- 6.5 在分区表order_partitioned中查询2023年2月的最高金额的订单信息

```
SELECT *
FROM order_partitioned
WHERE purchase_date >= '2023-02-01' AND purchase_date <= '2023-02-28'
order by amount desc
limit 1;
```