

《数据仓库》练习05

说明

- 在D盘创建考生文件夹：命名为“《数仓》+练习05+学号后2位”
- 在IDEA里创建sql文件，名称为 练习05.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05.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_db5 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_db5 的数据库，并将其位置设置为 /user/hive/exam/exam_db5。

```
-- 2.1 删除 exam_db5 数据库
drop database if exists exam_db5 cascade;

-- 2.2 创建 exam_db5 数据库
create database if not exists exam_db5
    location '/user/hive/exam/exam_db5';

-- 2.3 选择 exam_db5 数据库
use exam_db5;
```

三. 创建数据表

在 `exam_db5` 数据库中创建四个表：

- 3.1 `guests` 表结构：

列	类型	描述
guest_id	int	客人ID
guest_name	string	客人姓名
phone	string	电话号码

- 3.2 `rooms` 表结构：

列	类型	描述
room_id	int	房间ID
room_type	string	房型
rate	int	房价

- 3.3 `Hotelorders` 表结构：

列	类型	描述
orders_id	int	订单ID
guest_id	int	客人ID
room_id	int	房间ID
amount	double	支付金额
check_in_date	date	入住日期
check_out_date	date	离店日期

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

列	类型	描述
booking_id	int	订单ID
guest_id	int	客人ID
amount	double	支付金额

-- 3.1 创建 guests 数据表

```
CREATE TABLE if not exists guests (
  guest_id int COMMENT '客人ID',
  guest_name string COMMENT '客人姓名',
  phone string COMMENT '电话号码'
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

-- 3.2 创建 rooms 数据表

```
CREATE TABLE if not exists rooms (
  room_id int COMMENT '房间ID',
  room_type string COMMENT '房型',
  rate int COMMENT '房价'
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

-- 3.3 创建 Hotelorders 数据表

```
CREATE TABLE if not exists hotelorders (
  orders_id string COMMENT '订单ID',
  guest_id int COMMENT '客人ID',
  room_id int COMMENT '房间ID',
  amount double COMMENT '支付金额',
  check_in_date date COMMENT '入住日期',
  check_out_date date COMMENT '离店日期'
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

-- 3.4 创建按 check_out_date 分区的payments_partitioned分区表

```
CREATE TABLE if not exists payments_partitioned (
  orders_id string COMMENT '订单ID',
  guest_id int COMMENT '客人ID',
  amount double COMMENT '支付金额'
)
PARTITIONED BY (check_out_date date)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';
```

四. 修改数据表

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

```
ALTER TABLE guests
    ADD COLUMNS (sex STRING COMMENT '性别');
```

-- 4.2 把 `rooms` 表中将列 `rate` 改名为 `price`。

```
ALTER TABLE rooms CHANGE rate price int;
```

五. 加载数据

/* 5.1 将数据 `guests.txt`、`rooms.txt`、`hotelorders.txt` 上传到 master 节点上
`/opt/apps/hive/exam/` 下 */

-- 5.2 把 `guests.txt` 数据加载到 `guests` 表中：

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

-- 5.3 把 `rooms.txt` 数据加载 `rooms` 表中：

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

-- 5.4.1 把 `hotelorders.txt` 从本地上传到 HDFS 中，路径为 `/user/hive/exam/exam_db5`；
`hdfs dfs -put /opt/apps/hive/exam/hotelorders.txt /user/hive/exam/exam_db5`

-- 5.4.2 把路径 `/user/hive/exam/exam_db5` 下的 `hotelorders.txt` 数据加载到
`hotelorders` 表中。

```
LOAD DATA INPATH '/user/hive/exam/exam_db5/hotelorders.txt' OVERWRITE INTO TABLE
hotelorders;
```

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

-- 注意：该表是按照 `check_out_date` 进行分区。

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

```
INSERT OVERWRITE TABLE payments_partitioned PARTITION(check_out_date)
SELECT orders_id, guest_id, amount, check_out_date FROM hotelorders;
```

六. 查询数据

-- 6.1 统计每种房型的房间数量

```
SELECT room_type, COUNT(*) AS room_count
FROM rooms
GROUP BY room_type;
```

-- 6.2 查询入住次数最多的客人姓名和入住次数

```
SELECT g.guest_name, COUNT(*) AS booking_count
```

```
FROM hotelorders h
    JOIN guests g ON h.guest_id = g.guest_id
GROUP BY g.guest_name
ORDER BY booking_count DESC
LIMIT 1;
```

-- 6.3 在“hotelorders”表查询“2023-01-10”的入住信息?

```
SELECT *
FROM hotelorders
WHERE check_in_date = '2023-01-12';
```

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

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

SELECT *
FROM hotelorders
    DISTRIBUTE BY room_id
    SORT BY check_in_date DESC;
```

-- 6.5 在分区表payments_partitioned中查询2023年1月的支付最低金额的信息

```
SELECT *
FROM payments_partitioned
WHERE check_out_date >= '2023-01-01' AND check_out_date <= '2023-01-31'
order by amount Asc
limit 1;
```