Hive 数据仓库应用教程

模块五 Hive 数据操作

(Hive shùjù cāozuò)

第五次课 讲义原稿（教师用）

讲授语言：中文（附病音标注）

目 录

第一部分：课程导入 — 模块五概述（Slide 1-2）

第二部分：任务5-1 数据导入

2.1 任务描述（Slide 4）

2.2 任务准备知识 — 创建数据库（Slide 5-7）

2.3 数据导入方式总览（Slide 8）

2.4 方式一：直接向表中加载数据 INSERT INTO（Slide 9-19）

2.5 方式二：通过Load加载数据（Slide 20-49）

2.6 方式三：通过查询加载数据（Slide 50-60）

2.7 方式四：CTAS — 查询语句中创建表并加载数据（Slide 61-64）

2.8 方式五：Import导入Hive表数据（Slide 65-68）

2.9 数据导入小结（Slide 69）

第三部分：任务5-2 数据导出

3.1 任务描述（Slide 70-72）

3.2 方式一：Insert语句导出（Slide 73-87）

3.3 方式二：Hadoop命令导出（Slide 88-95）

3.4 方式三：Hive Shell命令导出（Slide 96-102）

3.5 方式四：Export语句导出（Slide 103-109）

第四部分：本章小结（Slide 110-111）

第五部分：结束语（Slide 112）

Slide 1 — 模块五 Hive数据操作（封面页）

同学们好！欢迎（huānyíng）来到我们 Hive 数据仓库的第五次课。

今天我们要学习的内容是——模块五 Hive 数据操作。

前面几次课，我们学了怎么创建数据库、怎么创建表。

那今天呢，我们就来学习一个非常实用（shíyòng）的内容——

怎么把数据放进 Hive 的表里面，还有怎么把数据从 Hive 里面拿出来。

简单说，就是“数据导入“和”数据导出“这两个核心（héxīn）操作。

> 💡 *用生活中的例子来说：数据导入就像把东西放进仓库（cāngkù），数据导出就像从仓库里把东西拿出来。*

Slide 2 — 本章内容概览（目录页）

大家看一下这个目录页，我们今天这节课主要分成四个任务：

第一个任务，任务5-1，数据导入（dǎorù）。这是最重要的部分，内容也最多。

第二个任务，任务5-2，数据导出（dǎochū）。学完导入以后，我们还要学怎么把数据取出来。

第三个任务，任务5-3，本章小结（xiǎojié）。把今天学的东西总结一下。

第四个任务，任务5-4，本章习题（xítí）。留一些练习给大家复习。

好，那我们现在就开始第一部分——数据导入。

Slide 3 — 任务5-1 数据导入（章节封面）

好的，现在我们进入任务5-1，数据导入。

数据导入，英文叫 Data Import 或者 Data Loading。

意思就是把外面的数据装载（zhuāngzài）到 Hive 的表里面去。

这个部分非常重要，因为没有数据的表就像空的仓库，什么都做不了。

Slide 4 — 5.1.1 任务描述

这个小节（xiǎojié）的目的是讨论怎么把数据装载到我们已经创建好的表里面。

我们假定（jiǎdìng）大家已经学过 SQL 的基础知识，所以这些语句对大家来说不会太陌生（mòshēng）。

如果你还不太熟悉 SQL，请回去复习（fùxí）一下前面的章节。

这里有一个很重要的知识点要记住：

Hive 和我们以前学的 MySQL 不一样——

Hive 没有行级别（háng jíbié）的数据插入（chārù）、更新（gēngxīn）和删除（shānchú）操作。

也就是说，你不能像 MySQL 那样一条一条地插入数据。

往 Hive 表里装数据，主要是用“大量（dàliàng）“装载的方式，或者直接把文件放到对应的目录下面。

> 💡 *打个比方：MySQL 像是一个个往书架上放书，而 Hive 是一箱一箱地往仓库里搬。*

Slide 5 — 5.1.2 任务准备知识

在开始今天的操作之前，我们需要先做一些准备工作。

为了避免（bìmiǎn）和前面章节的数据发生冲突（chōngtū），

我们要重新创建一个新的数据库，名字叫做 unit5db。

在本章的所有操作中，我们都会使用这个数据库。

请大家注意，后面的所有例题都是在 unit5db 这个数据库里面做的。

Slide 6 — 创建数据库 unit5db（操作步骤）

好，现在我们来一步一步地创建这个数据库。

一共有三个步骤（bùzhòu）：

Step 01：创建数据库 unit5db

第一步，我们在 Hive 里面输入创建数据库的命令：

【实习命令 / 实践操作】创建数据库

```
hive> CREATE DATABASE IF NOT EXISTS unit5db;
```

这条命令的意思是：如果 unit5db 这个数据库不存在的话，就创建它。

IF NOT EXISTS 是一个保护措施（cuòshī），防止重复创建时报错。

Step 02：查看数据库 unit5db

第二步，我们查看一下数据库是不是创建成功了：

【实习命令 / 实践操作】查看数据库

```
hive> SHOW DATABASES;
```

执行（zhíxíng）这条命令后，如果在列表里看到 unit5db，就说明创建成功了。

Step 03：使用数据库 unit5db

第三步，我们要切换到这个数据库：

【实习命令 / 实践操作】使用数据库

```
hive> USE unit5db;
```

执行完这条命令后，后面我们所有的操作都在 unit5db 这个数据库里面进行了。

Slide 7 — 创建数据库操作结果（图片页）

> **【图片说明】**图5-1 展示了创建数据库 unit5db 和使用数据库 unit5db 的操作结果截图。画面中可以看到 Hive 命令行界面，执行了 CREATE DATABASE、SHOW DATABASES 和 USE 三条命令。在 SHOW DATABASES 的结果中，可以看到 unit5db 已经出现在数据库列表里，说明创建成功。USE unit5db 执行后没有报错，说明切换成功。

Slide 8 — 5.1.3 任务实施 — 数据导入方式总览

好的，准备工作做完了。现在我们来看看 Hive 中数据导入的方式。

Hive 一共有五种（wǔ zhǒng）方式可以往表中装载数据：

第一种：直接向表中加载数据 —— 用 INSERT INTO ... VALUES 语句

第二种：通过 Load 加载数据 —— 从本地文件或 HDFS 加载

第三种：通过查询加载数据 —— 用 INSERT ... SELECT 语句

第四种：查询语句中创建表并加载数据 —— 用 CREATE TABLE ... AS SELECT

第五种：Import 导入 Hive 表数据 —— 用于集群（jíqún）之间的数据迁移（qiānyí）

> 💡 *这五种方式各有各的用处，我们一个一个来学习。*

Slide 9 — 方式一：直接向表中加载数据（INSERT INTO VALUES）

我们先来看第一种方式：直接向表中加载数据。

当我们需要插入少量（shǎoliàng）的测试数据的时候，可以使用 INSERT INTO 语句。

大家看一下它的语法格式（yǔfǎ géshì）：

【实习命令 / 实践操作】INSERT INTO 语法

```
INSERT INTO TABLE tablename
  [PARTITION (partcol1[=val1], partcol2[=val2] ...)]
VALUES values_row [, values_row ...]
```

这个语法和 MySQL 里面的 INSERT INTO 很像。

tablename 就是表的名字。

PARTITION 是分区（fēnqū），如果你的表有分区就要指定。

VALUES 后面跟上你要插入的数据。

Slide 10-12 — 例题2：向 students 表插入数据

好，我们来看第一个例题。

例题2要求我们向学生信息表 students 中插入2条新记录。

分别是 ('fred', 1.28) 和 ('barney rubble', 32, 2.32)。

操作过程（cāozuò guòchéng）很简单：

首先我们要创建 students 这个表，然后用 INSERT INTO 往里面插入数据。

> **【图片说明】**图5-2 和 图5-3 展示了操作过程和操作结果。画面中可以看到先执行了建表语句创建 students 表，然后执行 INSERT INTO 语句插入了2条数据。最后用 SELECT * FROM students 查询，可以看到2条新记录已经成功插入。

Slide 13-19 — 例题3：向 pageviews 表插入和更新数据

接下来看例题3，这个例子稍微复杂（fùzá）一点。

我们要向浏览信息表 pageviews 中插入2条新记录，

分别是 ('jsmith', 'mail.com', 'sports.com') 和 ('jdoe', 'mail.com', null)。

同时还要更新表中的2条记录。

这里大家注意一个关键点：

Hive 中的 INSERT 操作会启动（qǐdòng）MapReduce 任务来完成。

所以每次执行 INSERT 可能需要一些时间，大家不要着急（zháojí）。

> **【图片说明】**Slide 14-19 展示了完整的操作过程截图，包括建表、插入数据、查询验证等多个步骤的执行结果。每一步都能看到 Hive 命令行的输出，显示操作成功完成。

슬라이드 14:

```
--step01：创建浏览信息表pageviews
--创建本例题所需要的浏览信息表pageviews，具体操作命令如下：
CREATE TABLE pageviews (
userid string,
link string,
came_from string
) PARTITIONED BY (stat_date STRING)
CLUSTERED BY (userid) INTO 256 BUCKETS
STORED AS ORC;

--step02：查看浏览信息表pageviews
--查看创建的浏览信息表pageviews是否成功，具体操作命令如下：
SHOW TABLES;

--step03：查看浏览信息表pageviews的数据
--查看创建的浏览信息表pageviews的数据，具体操作命令如下：
SELECT * FROM pageviews;
```

슬라이드 15:

```
--step04：将数据插入到浏览信息表 pageviews 中
--将数据插入到浏览信息表 pageviews 中，具体操作命令如下：
INSERT INTO TABLE pageviews PARTITION (stat_date='2026-02-07')
VALUES ('jsmith', 'mail.com','sports.com'),
('jdoe','mail.com',null);

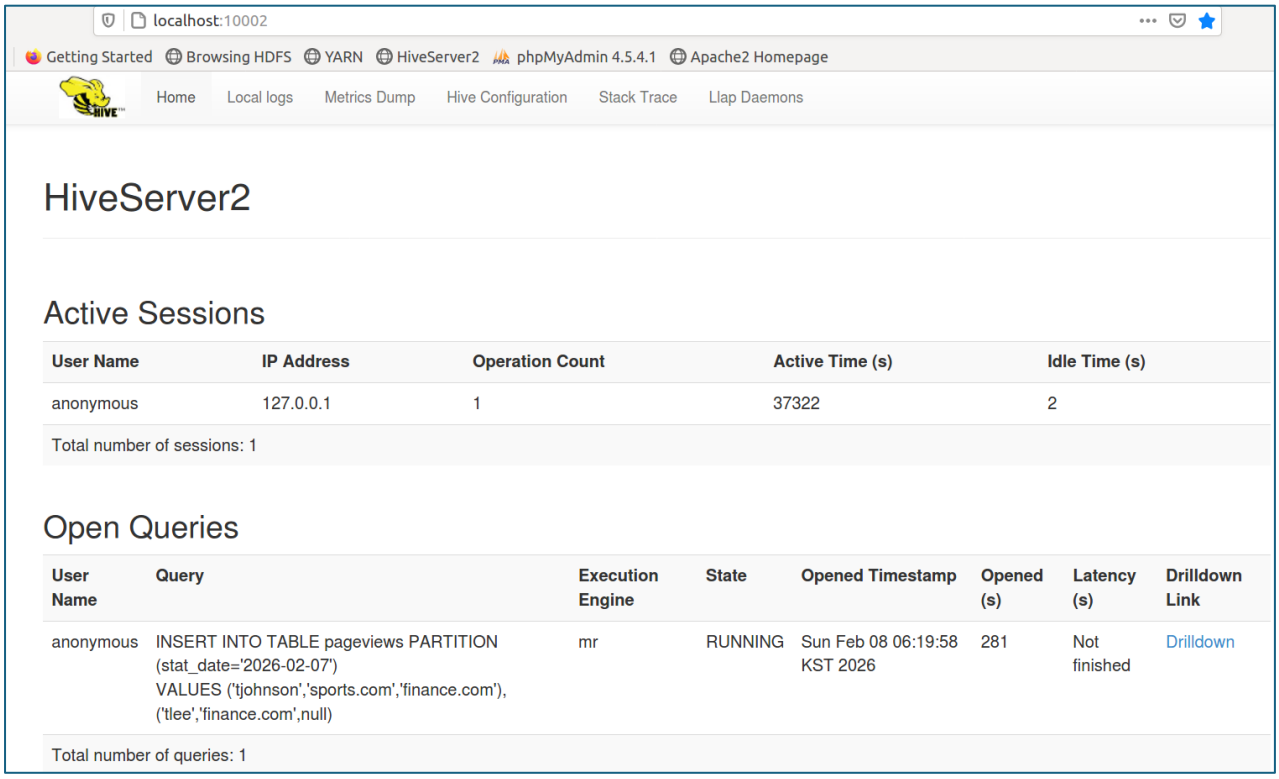

--step05：查看表中的数据
--查看浏览信息表 pageviews 中的数据，具体操作命令如下：
SELECT * FROM pageviews;
--目前，浏览信息表 pageviews 插入 2 条新数据成功，
--且 2 条数据全部插入到分区 '2026-02-07'中。
```

슬라이드 16:

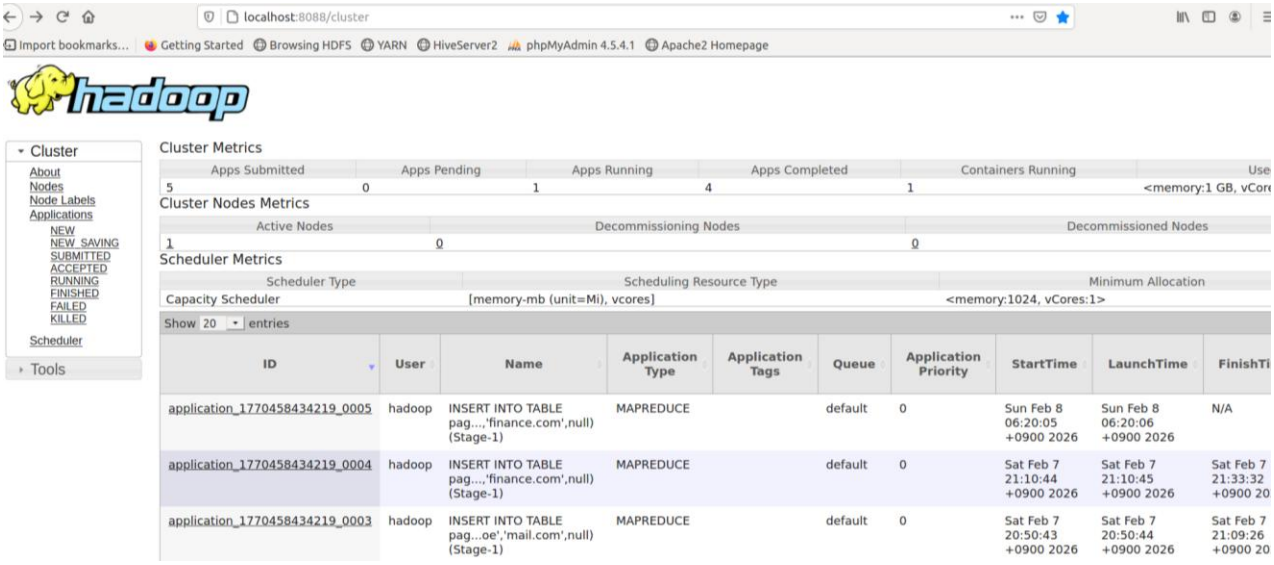
```
--step06：将更新的数据插入到浏览信息表 pageviews 中
--将需要更新的 2 条新数据插入到浏览信息表 pageviews 中，具体操作命令如下：
INSERT INTO TABLE pageviews PARTITION (stat_date='2026-02-07')
VALUES ('tjohnson','sports.com','finance.com'),
('tlee','finance.com',null);

--step07：查看表中的数据
--再次查看浏览信息表 pageviews 中的数据，具体操作命令如下：
SELECT * FROM pageviews;
--由操作结果可以看出，目前，浏览信息表 pageviews 更新插入的 2 条新数据成功，
--且 2 条数据全部插入到分区'2026-02-07'中。
```

슬라이드 17:


localhost:10002
Getting Started
Browsing HDFS
YARN
HiveServer2
phpMyAdmin 4.5.4.1
Apache2 Homepage
Home
Local logs
Metrics Dump
Hive Configuration
Stack Trace
Llap Daemons
HiveServer2
Active Sessions
User Name
IP Address
Operation Count
Active Time (s)
Idle Time (s)
anonymous
127.0.0.1
1
37322
2
Total number of sessions: 1
Open Queries
User Name
Query
Execution Engine
State
Opened Timestamp
Opened (s)
Latency (s)
Drilldown Link
anonymous
INSERT INTO TABLE pageviews PARTITION (stat_date='2026-02-07') VALUES ('tjohnson','sports.com','finance.com'), ('tlee','finance.com',null)
mr
RUNNING
Sun Feb 08 06:19:58 KST 2026
281
Not finished
Drilldown
Total number of queries: 1

슬라이드:18



슬라이드 : 19



Slide 20-21 — 方式二：通过 Load 加载数据

好的，现在我们来学第二种方式——通过 Load 加载数据。
这种方式非常常用，也是我们今天的重点（zhòngdiǎn）。

首先大家要理解（lǐjiě）一个概念：
Hive 的底层（dǐcéng）是 Hadoop，数据实际上是存储在 HDFS 分布式文件系统上的。
Hive 在加载数据的时候，不会修改（xiūgǎi）数据内容。
它只是把数据文件移动（yídòng）到 HDFS 中 Hive 指定的目录下面。

> 💡 *用搬家来比喻：Load 操作就像是把一箱东西搬到仓库指定的位置，箱子里面的东西不会被打开或修改。*

所以，Hive 不支持对数据的改写（gǎixiě）和添加（tiānjiā）。
所有的数据都是在加载的时候就确定了的。

Slide 22 — Load 加载数据 — 从本地文件加载

Load 加载数据主要有两种途径（tújìng）：
第一种，从本地文件系统加载到 Hive 表。
第二种，从 HDFS 文件系统加载到 Hive 表。

我们先看从本地文件加载的方式。
当你需要把本地的一个文本文件（wénběn wénjiàn）装载到 Hive 的表中时，
就可以使用 LOAD DATA LOCAL 语句。

```
LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename
[PARTITION (partcol1=val1, partcol2=val2 ...)] [INPUTFORMAT 'inputformat' SERDE 'serde']
(3.0 or later)
```

Slide 23 — 语法规则之 filepath（文件路径）

这里有一个重要的参数（cānshù）叫做 filepath，就是文件路径。

filepath 表示你要加载的数据文件在哪里。

filepath 支持三种形式：

第一种：相对路径（xiāngduì lùjìng），比如 project/data1

第二种：绝对路径（juéduì lùjìng），比如 /user/hive/project/data1

第三种：带有 schema 的完整 URI，比如 hdfs://namenode:9000/user/hive/project/data1

filepath 可以指向一个文件，也可以指向一个目录。

如果指向文件，Hive 会把这个文件移动到表中。

如果指向目录，Hive 会把目录下面的所有文件都移动到表中。

```
LOAD DATA [LOCAL] INPATH 'filepath' [OVERWRITE] INTO TABLE tablename
[PARTITION (partcol1=val1, partcol2=val2 ...)] [INPUTFORMAT 'inputformat' SERDE 'serde']
(3.0 or later)
```

Slide 24 — Load 操作的特点

这里有几个特点大家要记住：

第一，Load 操作只是单纯（dānchún）的复制或者移动操作。

第二，Hive 对导入的数据不做错误检查（cuòwù jiǎnchá）。

第三，如果数据格式和表的字段不匹配（pǐpèi），Hive 会用 NULL 来替代（tìdài）。

第四，对于多余的数据，Hive 会直接丢弃（diūqì）。

这就是为什么我们说 Hive 属于“读时模式（dú shí móshì）”——

数据在加载的时候不检查，只有在读取（dúqǔ）查询的时候才发现问题。

Slide 25-30 — 例题4：将本地文件加载到 Hive 表中

好，我们来做一个实际的例子。

例题4要求我们把本地文件 order-info-0208.txt 加载到 Hive 的 order_info_load 表中。

Step 01：创建本地文件

首先，我们在本地 /usr 下创建一个文件夹 datas，用来存放所有的本地文件。

【实习命令 / 实践操作】创建目录和数据文件

```
mkdir -p /usr/datas
```

然后创建 order-info-0208.txt 这个数据文件。

Step 02-04：创建表并加载数据

接下来我们要在 Hive 里面创建 order_info_load 表，然后用 LOAD 命令加载数据。

【实习命令 / 实践操作】加载本地文件到Hive表

```
hive> LOAD DATA LOCAL INPATH '/usr/datas/order-info-0208.txt'
      INTO TABLE order_info_load;
```

注意这里的关键字：

LOCAL —— 表示从本地文件系统加载（不是从 HDFS）。

INPATH —— 后面跟文件的路径。

INTO TABLE —— 后面跟表的名字。

> **【图片说明】**图5-4至5-8 展示了完整操作过程：首先创建了本地目录和数据文件，然后在Hive中创建了order_info_load表，最后执行LOAD命令将本地文件加载到表中。执行结果显示操作成功。实际上，从本地上传文件到Hive表中，底层是将文件复制到HDFS的表目录下。

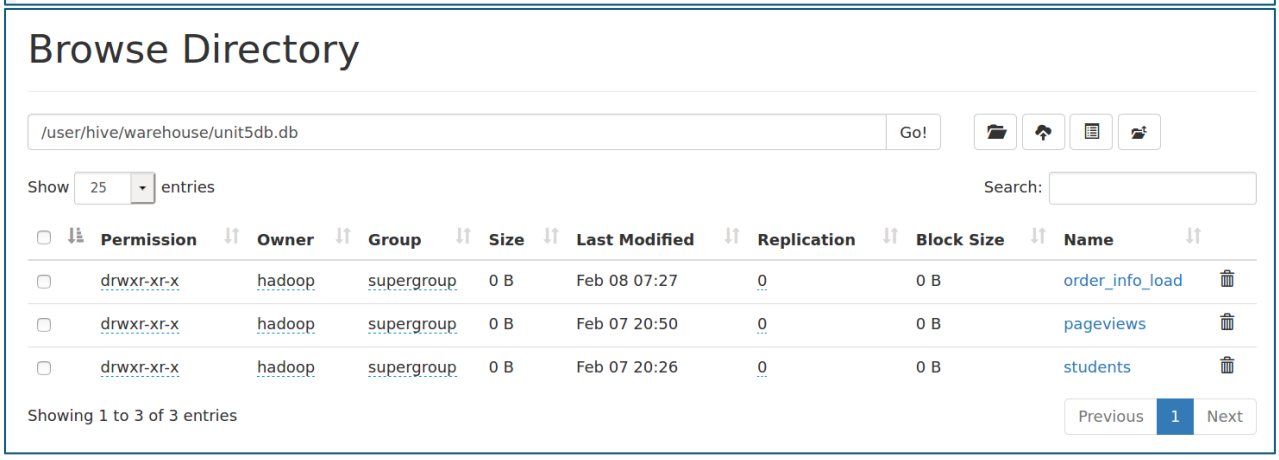
```
cd /usr
sudo mkdir datas
sudo chown hadoop:hadoop -R datas

touch order-info-0208.txt
vim order-info-0208.txt
1,待支付,2026-02-08 12:20:11,2026-02-08 12:20:11
2,待支付,2026-02-08 14:20:11,2026 02-08 14:20:11
3,待支付,2026-02-08 16:20:11,2026-02-08 16:20:11
```

```
--step02：查看数据库 unit5db 的位置
--查看数据库 unit5db 在 Hive 中的位置，具体操作命令如下：
hive-bee
SHOW DATAFILES;
USE unit5db;
DESC DATABASE unit5db;
```

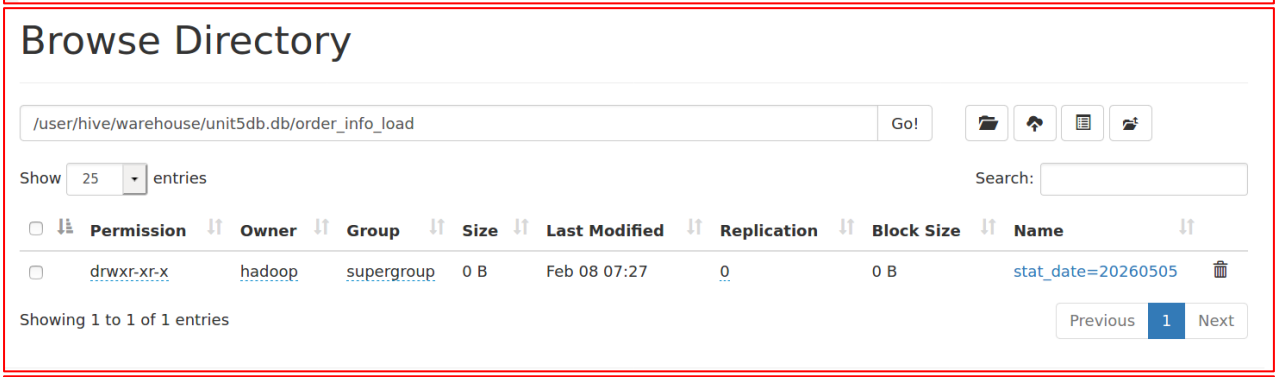
```
--step03：创建数据库中的表order_info_load
--在数据库unit5db中，创建表order_info_load，具体操作命令如下：
CREATE TABLE order_info_load(
order_id STRING COMMENT '订单id',
order_status STRING COMMENT '订单状态',
create_time STRING COMMENT '创建时间',
update_time STRING COMMENT '最后更新时间'
)
partitioned BY (stat_date string)
row format DELIMITED
FIELDS TERMINATED by ',';
```

```
--step04：查看数据库中的表
--在数据库unit5db中，查看表order_info_load是否创建成功，具体操作命令如下：
SHOW tables;
```



```
--step05：将本地文件 order-info-0208.txt 上传到服务器 Hive 中

--在数据库 unit5db 中， 加载服务器本地 order-info-0208.txt
--文本数据到order_info_load 订单表的 20260208 分区中，具体操作命令如下：
LOAD DATA LOCAL inpath '/usr/datas/order-info-0208.txt'
OVERWRITE INTO TABLE order_info_load
PARTITION(stat_date='20260505');
```




```
--step06：验证上传结果
验证 Load 加载 HDFS 数据结果集，具体操作命令如下：
SELECT * FROM order_info_load;
SELECT * FROM order_info_load WHERE stat_date = '20260505';
```

```
0: jdbc:hive2://localhost:10000> SELECT * FROM order_info_load;
+--------------------------+-----------------------------+-----------------------------+-----------------------------+---------------------------+
| order_info_load.order_id | order_info_load.order_status | order_info_load.create_time | order_info_load.update_time | order_info_load.stat_date |
+--------------------------+-----------------------------+-----------------------------+-----------------------------+---------------------------+
| 1                        | 待支付                       | 2026-02-08 12:20:11         | 2026-02-08 12:20:11         | 20260505                  |
| 2                        | 待支付                       | 2026-02-08 14:20:11         | 2026 02-08 14:20:11         | 20260505                  |
| 3                        | 待支付                       | 2026-02-08 16:20:11         | 2026-02-08 16:20:11         | 20260505                  |
+--------------------------+-----------------------------+-----------------------------+-----------------------------+---------------------------+
3 rows selected (3.313 seconds)
0: jdbc:hive2://localhost:10000> SELECT * FROM order_info_load WHERE stat_date = '20260505';
+--------------------------+-----------------------------+-----------------------------+-----------------------------+---------------------------+
| order_info_load.order_id | order_info_load.order_status | order_info_load.create_time | order_info_load.update_time | order_info_load.stat_date |
+--------------------------+-----------------------------+-----------------------------+-----------------------------+---------------------------+
| 1                        | 待支付                       | 2026-02-08 12:20:11         | 2026-02-08 12:20:11         | 20260505                  |
| 2                        | 待支付                       | 2026-02-08 14:20:11         | 2026 02-08 14:20:11         | 20260505                  |
| 3                        | 待支付                       | 2026-02-08 16:20:11         | 2026-02-08 16:20:11         | 20260505                  |
+--------------------------+-----------------------------+-----------------------------+-----------------------------+---------------------------+
```

由执行结果可以看出，上传本地文件order-info-0208.txt 到服务器 Hive 中的表 order_info_load，实际操作是向表中复制数据。

Slide 31-35 — 例题5：将本地文件上传到 Hive 分区表中

例题5和例题4类似，但这次我们要把数据加载到一个分区（fēnqū）中。

Step 01：创建本地文件

【实习命令 / 实践操作】创建数据文件

创建本地文件 /usr/datas/order-info-0209.txt

```
cd /usr/dat,as
touch order-info-0209.txt
1,待支付,2026-02-08 12:20:11,2026-02-08 12:20:11
2,已支付,2026-02-09 14:20:11,2026-02-09 14:21:11
3,已支付,2026-02-09 16:20:11,2026-02-09 16:21:11
3,待支付,2026-02-09 12:20:11,2026-02-09 12:20:11
3,待支付,2026-02-09 14:20:11,2026-02-09 14:20:11

Ctrl+C 复制

vim order-info-0209.txt
--建i打印，然后 Ctrl+Shiht+C
```

Step 02：查看数据库位置

【实习命令 / 实践操作】查看数据库位置

```
hive> DESCRIBE DATABASE unit5db;
```

```
--step02：查看数据库 unit5db 的位置
--查看数据库 unit5db 在 Hive 中的位置，具体操作命令如下：
DESC DATABASE unit5db;

--有结果可知，hdfs://hadoop01:9000/user/hive/warehouse/unit5db.db,
```

Step 03-04：加载数据到分区

这次我们加载到表的 20260506 这个分区中：

【实习命令 / 实践操作】加载数据到分区表

```
hive> LOAD DATA LOCAL INPATH '/usr/datas/order-info-0209.txt'
      INTO TABLE order_info_load PARTITION(dt='20260506');
```

```
--step03：查看数据库中的表
--在数据库 unit5db 中，查看表 order_info_load 是否存在，具体操作命令如下：
SHOW TABLES;
SELECT * FROM order_info_load;
```

```
--step04：将本地文件 order-info-0209.txt 上传到服务器 Hive 中
--在数据库 unit5db 中，
--加载服务器本地 order-info-0209.txt 文本数据到order_info_load 订单表的 20220608 分区中，
--具体操作命令如下：
LOAD DATA LOCAL inpath '/usr/datas/order-info-0209.txt'
OVERWRITE INTO TABLE order_info_load
PARTITION (stat_date='20260506');
```

```
--step05：验证上传结果
--验证 Load 加载 HDFS 数据结果集，具体操作命令如下：
SELECT * FROM order_info_load WHERE stat_date='20260506';
```

大家注意，这里多了一个 PARTITION 关键字。

dt='20260506' 表示我们把数据放到日期为 20260506 的这个分区里面。

【图片说明】展示了查看数据库位置、加载数据到分区以及验证上传结果的完整过程。

Slide 36 — 重要注意事项

这里有两个重要的注意事项（zhùyì shìxiàng）：

第一，和我们熟悉的关系型数据库（guānxìxíng shùjùkù）不同，
Hive 在早期版本中不支持 INSERT INTO ... VALUES 这种形式。
虽然现在的版本（bǎnběn）可以了，但一般不推荐（tuījiàn）用这种方式插入大量数据。

第二，在 Hive 3.0 版本之前，Load 操作只是简单的复制操作。
在 Hive 3.0 版本之后，Hive 内部会把 Load 操作重写为 INSERT AS SELECT。

Slide 37 — 通过 HDFS 将数据文件加载到 Hive 表中

除了从本地加载数据，我们还可以从 HDFS 上直接加载数据到 Hive 表中。

大家要理解一个过程：
当我们从本地文件系统导入数据到 Hive 时，
实际上是先把数据临时（línshí）复制到 HDFS 的一个临时目录下（比如 /tmp），
然后再从这个临时目录移动到 Hive 表的数据目录里面。

注意：这里说的是“移动“，不是”复制“！
所以如果数据已经在 HDFS 上了，我们可以直接从 HDFS 加载到 Hive 表。
这种方式不需要加 LOCAL 关键字。

Slide 38-43 — 例题6：从 HDFS 加载文件到 Hive 表

例题6要求我们把 HDFS 上的文件 order-info-0323.txt 加载到 Hive 表中。

Step 01-02：准备工作

首先查看 HDFS 文件系统的目录结构，确定数据文件的位置。

【实习命令 / 实践操作】查看HDFS目录

```
[root@node2 ~]# hadoop fs -ls /tmp/
```

```
--step01：上传本地文件 order-info-0209.txt 到 HDFS 中
--在本地已经创建本地文件 order-info-0209.txt,
--在本地/usr 下创建了一个文件夹 datas，用来存放本章节所有的本地文件,
--那么本地文件的具体路径/usr/datas/- order-info-0209.txt。
其中，本地文件 order-info-0209.txt 中共有 5 条数据，分别为：

1,待支付,2026-02-08 12:20:11,2026-02-08 12:20:11
2,已支付,2026-02-09 14:20:11,2026-02-09 14:21:11
3,已支付,2026-02-09 16:20:11,2026-02-09 16:21:11
3,待支付,2026-02-09 12:20:11,2026-02-09 12:20:11
3,待支付,2026-02-09 14:20:11,2026-02-09 14:20:11
```

```
--step02：查看数据库 unit5db 的位置
--查看数据库 unit5db 在 Hive 中的位置，具体操作命令如下：
DESC DATABASE unit5db;
```

Step 03-04：上传本地文件到 HDFS

【实习命令 / 实践操作】上传文件到HDFS

```
[root@node2 ~]# hadoop fs -put /usr/datas/order-info-0323.txt /tmp/
```

```
--step03：查看 HDFS 文件系统的目录结构
--查看 HDFS 文件系统的目录结构，以方便确定待上传数据在 HDFS 中的存放路径。
--具体操作命令如下：
[Main terminal에서]
hhdfs dfs -ls /
--其中，HDFS 文件系统的目录结构，如图 所示。共有 4 个文件目录，
--其中一个文件目录为/tmp，此目录为本章节待上传数据文件的 HDFS 的存储目录。
```

```
--step04：上传本地文件并查看 HDFS 文件系统的目录结构
--上传本地文件 order-info-0209.txt 到 HDFS 中的/tmp 目录中，
--并再次查看 HDFS 文件系统中的/tmp 目录结构，
--以确定本地文件 order-info-0209.txt 上传到 HDFS 文件系统中的/tmp 目录成功。
--具体操作命令如下：
[Main terminal에서]
hdfs dfs -put /usr/datas/order-info-0209.txt /tmp/
--然后，再次查看 HDFS 文件系统中的/tmp 目录结构，
--以确定本地文件order-info-0209.txt 上传到HDFS 文件系统中的/tmp 目录成功。
--具体操作命令如下：
hdfs dfs -ls /tmp/*.txt
```

Step 05：从 HDFS 加载到 Hive

注意：这里没有 LOCAL 关键字！因为数据已经在 HDFS 上了。

【实习命令 / 实践操作】从HDFS加载数据到Hive表

```
hive> LOAD DATA INPATH '/tmp/order-info-0323.txt'
      INTO TABLE order_info_load;
```

```
--step05：上传 HDFS 的文件到 Hive 服务器中
--将 HDFS 文件系统的目录/tmp/order-info-0209.txt 下的 order-info-0209.txt 文件
--加载order_info_load 订单表的 20260506 分区中。
--具体操作命令如下：
LOAD DATA inpath 'hdfs://hadoop01:9000/tmp/order-info-0209.txt'
OVERWRITE INTO TABLE order_info_load
PARTITION (stat_date='20260506');

--然后，再次查看 order_info_load 订单表的 20260208分区中的数据，
SELECT * FROM order_info_load WHERE stat_date='20260506';

[Main terminal에서]
/usr/datas$ hdfs dfs -ls /tmp/*.txt
```

大家比较（bǐjiào）一下：

从本地加载：LOAD DATA LOCAL INPATH ... （有 LOCAL）

从 HDFS 加载：LOAD DATA INPATH ... （没有 LOCAL）

这个区别一定要记住！

```
hadoop@hadoop01:/usr/datas$ hdfs dfs -ls /tmp/*.txt
-rw-r--r--   1 hadoop supergroup        260 2026-02-08 11:46 /tmp/order-info-0209.txt
hadoop@hadoop01:/usr/datas$ hdfs dfs -ls /tmp/*.txt
ls: `/tmp/*.txt': No such file or directory
```

Slide 44-49 — 例题7：将 HDFS 上的 CSV 文件加载到 Hive 表

例题7是把 HDFS 上的 Person001.csv 文件加载到 person 表中。

操作步骤和例题6类似。

Step 01：上传本地 CSV 到 HDFS

【实习命令 / 实践操作】上传CSV文件到HDFS

```
[root@node2 ~]# hadoop fs -put /usr/datas/Person001.csv /tmp/
```

```
--在本地已经创建本地文件Person001.csv,
--在本地/usr下已经创建了一个文件夹datas,
--用来存放本章节所有的本地文件,
--那么本地文件的具体路径/usr/datas/Person001.csv,
--其中，本地文件Person001.csv中共有80条数据,
--此处，仅显示前面5条记录，分别为：

2   SMITH   AARON
3   SMITH   ABDUL
4   SMITH   ABE
5   SMITH   ABEL
6   SMITH   ABRAHAM
```

Step 02-03：查看 HDFS 目录结构

【实习命令 / 实践操作】查看HDFS目录

```
[root@node2 ~]# hadoop fs -ls /
[root@node2 ~]# hadoop fs -ls /tmp/
```

【图片说明】展示了HDFS文件系统的目录结构，可以看到有4个文件目录，其中 /tmp 目录用来存放待上传的数据文件。

```
--step02：上传本地文件Person001.csv到 /usr/datas/中
[Main terminal에서]
cd /usr/datas
cp ~/Downloads/Person001.csv

hdfs dfs -ls /
```

```
--step03：上传本地文件并查看 HDFS 文件系统的目录结构
--上传本地文件 Person001.csv 到 HDFS 中的/tmp 目录中，
--并再次查看 HDFS文件系统中的/tmp 目录结构，
--以确定本地文件 Person001.csv 上传到 HDFS 文件系统中的/tmp 目录成功。
hdfs dfs -put /usr/datas/Person001.csv /tmp

--然后，再次查看 HDFS 文件系统中的/tmp 目录结构，
--以确定本地文件Person001.csv 上传到 HDFS 文件系统中的/tmp 目录成功。
hdfs dfs -ls /tmp/Person001.*
--其中，HDFS 文件系统的目录结构。
--共有 3 个文件目录，其中一个文件目录为/tmp/Person001.csv，
--本地文件 Person001.csv 上传到 HDFS指定的目录/tmp 成功。
```

Step 04：创建 person 表

【实习命令 / 实践操作】创建person表

```
hive> CREATE TABLE person (
        persid INT,
        firstname STRING,
        lastname STRING
      )
      ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

```
--step04：创建数据库中的person表
--在数据库unit5db中，创建person表，具体操作命令如下：
CREATE TABLE person (
perid int,
lastname string,
firstname string
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ',';

--然后，在数据库 unit5db 中，查看 person 表是否创建成功。
SHOW tables;
```

```
0: jdbc:hive2://localhost:10000> show tables;
+------------------+
|     tab_name     |
+------------------+
| order_info_load  |
| pageviews        |
| students         |
+------------------+
3 rows selected (0.25 seconds)
0: jdbc:hive2://localhost:10000> CREATE TABLE person (
. . . . . . . . . . . . . . . .> perid int,
. . . . . . . . . . . . . . . .> lastname string,
. . . . . . . . . . . . . . . .> firstname string
. . . . . . . . . . . . . . . .> )
. . . . . . . . . . . . . . . .> ROW FORMAT DELIMITED
. . . . . . . . . . . . . . . .> FIELDS TERMINATED BY ',';
No rows affected (3.943 seconds)
0: jdbc:hive2://localhost:10000> show tables;
+------------------+
|     tab_name     |
+------------------+
| order_info_load  |
| pageviews        |
| person           |
| students         |
+------------------+
4 rows selected (0.254 seconds)
```

Step 05：加载 HDFS 文件到 Hive 表

【实习命令 / 实践操作】加载CSV到person表

```
hive> LOAD DATA INPATH '/tmp/Person001.csv' INTO TABLE person;
```

```
--step05：上传 HDFS 的文件到 Hive 服务器中
--将 HDFS 文件系统的目录/tmp/Person001.csv 下的 Person001.csv 文件加载到person 表中。
LOAD DATA inpath 'hdfs://hadoop01:9000/tmp/Person001.csv'
OVERWRITE INTO TABLE person;

SELECT * FROM person;
```

Slide 50 — 方式三：通过查询加载数据（INSERT ... SELECT）

第三种方式是通过查询来加载数据。

简单来说，就是从一个表中查询数据，然后把查询结果插入到另一个表中。

Hive 使用 INSERT 子句（zǐjù）将查询结果插入到表中。

这种方式可以一次性（yīcìxìng）加载数据到一张表中，

也可以一次性加载数据到多张表中，非常灵活（línghuó）。

语法格式有两种：

OVERWRITE 方式 —— 覆盖（fùgài）目标表的原有数据

INTO 方式 —— 追加（zhuījiā）到目标表中

```
INSERT  OVERWRITE  TABLE  tablename1
[PARTITION (partcol1=val1, partcol2=val2 ...)
[IF NOT EXISTS]]   select_statement1   FROM    from_statement;

--多个表插入格式：
FROM employees
INSERT OVERWRITE TABLE emp_partitioned
    PARTITION (dept='HR')
    SELECT name, salary WHERE dept = 'HR'
INSERT OVERWRITE TABLE emp_partitioned
    PARTITION (dept='IT')
    SELECT name, salary WHERE dept = 'IT'
INSERT INTO TABLE emp_backup
    SELECT *;
```

Slide 51-53 — 例题8：用 OVERWRITE 方式加载数据

例题8要求我们用 OVERWRITE 方式把 person 表的数据加载到 personhub 表中。

Step 01：创建 personhub 表

【实习命令 / 实践操作】创建目标表

```
hive> CREATE TABLE personhub (
        persid INT,
        firstname STRING,
        lastname STRING
      );
```

```
CREATE TABLE personhub (persid int);

然后，在数据库 unit5db 中，查看 personhub 表是否创建成功。
SHOW TABLES;
```

Step 02：用 OVERWRITE 方式加载数据

【实习命令 / 实践操作】OVERWRITE方式加载数据

```
hive> INSERT OVERWRITE TABLE personhub
      SELECT persid, firstname, lastname FROM person;
```

```
--step02：  利用 overwrite 方式将 person 表的数据加载到 personhub 表中 在数据库unit5db 中，
--利用 overwrite 方式将person 表的数据加载到personhub表中，覆盖 personhub 表中原有数据
INSERT OVERWRITE TABLE personhub
SELECT DISTINCT perid FROM person; --이것도 시간 많이 걸림(78.912초)
```

大家注意：OVERWRITE 是覆盖的意思，会把 personhub 表中原来的数据全部清掉，

然后把查询出来的新数据放进去。

Step 03：验证数据

【实习命令 / 实践操作】验证数据

```
hive> SELECT * FROM personhub;
```

```
--step03：  验证 personhub 表中的数据
--再次查看 personhub 表中的数据

SELECT * from personhub;
```

Slide 54 — 例题9：用 INTO 方式加载数据

例题9和例题8类似，但这次用 INTO 方式。

INTO 和 OVERWRITE 的区别是：

INTO 是追加数据，不会删除原来的数据。

OVERWRITE 是覆盖数据，会删除原来的数据。

【实习命令 / 实践操作】INTO方式加载数据

```
hive> INSERT INTO TABLE personhub
      SELECT persid, firstname, lastname FROM person;
```

执行后查看 personhub 表，可以看到有80条数据（之前已有的加上新追加的）。

```
--step02：  利用 into 方式将 person 表的数据加载到 personhub 表中在数据库 unit5db 中，
--利用 into 方式将 person 表的数据加载到 personhub 表中，追加新数据到 personhub 表中原有数据后，
INSERT INTO TABLE personhub
SELECT DISTINCT perid FROM person; --이것도 시간 많이 걸림(65.947초)


--step03：  验证 personhub 表中的数据再次查看 personhub 表中的数据，
--由结果可知，整个表中共有 160 条最新的数据，加载文件成功。
SELECT * FROM personhub;
```

Slide 55-60 — 例题10：一次性加载到多个表

例题10展示了一个更高级（gāojí）的用法：

从 wyp 表中查询数据，然后根据不同的条件（tiáojiàn），

分别加载到 test 表和 test3 表中。

这样可以一次查询，把数据分别导入不同的表，效率（xiàolǜ）很高。

```
--step01：   创建数据库中的 wyp 表
--在数据库 unit5db 中，创建 wyp 表
CREATE TABLE wyp (id int, name string, age int, tel string)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ' '
STORED AS TEXTFILE;

--然后，在数据库unit5db中，查看wyp表是否创建成功。
--创建成功后，利用前面所学知识向wyp表中加载wyp.txt和add.txt文件中的数据，
-- 最后，查看wyp表中的数据，共有7条数据
--step02：创建wyp.txt使用如下：
[Main terminal에서]
cd /usr/datas
vim wyp.txt

5 wyp1 23 232323232323
6 wyp2 34 32454567678825
7 wyp3 36 23243243545656
8 wyp4 26 24324354667572
1 wyp 25 13188888888
2 test 30 13388888888
3 zs 34 899314121
```

```
--step03：上传本地文件并查看 HDFS 文件系统的目录结构
--上传本地文件 wyp.txt 到 HDFS 中的/tmp 目录中，
--并再次查看 HDFS文件系统中的/tmp 目录结构，
--以确定本地文件 wyp.txt 上传到 HDFS 文件系统中的/tmp 目录成功。
hdfs dfs -put /usr/datas/wyp.txt /tmp

hdfs dfs -ls /tmp/wyp.*

--step04：上传 HDFS 的文件到 Hive 服务器中
--将 HDFS 文件系统的目录/tmp/wyp.txt 下的 wyp.txt 文件加载到wyp 表中。
LOAD DATA inpath 'hdfs://hadoop01:9000/tmp/wyp.txt'
OVERWRITE INTO TABLE wyp;

SELECT * FROM wyp;
```

```
--step04 创建数据库中的test表
--在数据库unit5db中，创建test表
CREATE TABLE test(id int, name string, tel string)
PARTITIONED BY (age int)
ROW FORMAT DELIMITED FIELDS
TERMINATED BY ' '
STORED AS TEXTFILE;
--然后，在数据库unit5db中，查看test表是否创建成功。
--创建成功后，查看test表中的数据，表中无数据。
SELECT * FROM test;

--step05： 创建数据库中的 test3 表
--在数据库 unit5db 中，创建 test3 表，具体操作命令如下：
CREATE TABLE test3(id int, name string);
--然后，在数据库unit5db中，查看test3表是否创建成功。创建成功后，
--查看test3表中的数据，表中无数据。
SELECT * FROM test3;
```

```
--step06：按照查询要求加载 wyp 表中的数据分别到 test 表和 test3 表中 在数据库unit5db 中，
--按照查询要求加载wyp 表中的数据分别到test 表和test3表中，具体操作命令如下：
SET hive.exec.dynamic.partition.mode=nonstrict;

--在此处，test 表里面用 age 作为了分区字段，
--为了方便将 wyp 表中的查询结果插入到 test 表中，
--因此，需要设置 hive.exec.dynamic.partition.mode 的值为 nonstrict。
FROM wyp
INSERT INTO table test
partition(age)
SELECT id, name, tel, age
INSERT into table test3
SELECT id, name where age>25;     --시간 걸림 (65.468초)
```

```
--step07： 分别验证 test 表和 test3 表中的数据
--再次查看 test 表和 test3 表中的数据，如图 5-40 和图 5-41 所示。
--由结果可知，test 表和 test3 表分别加载数据成功。
--具体操作命令如下：
SELECT * FROM test;
SELECT * FROM test3;
```

【图片说明】Slide 56-60 展示了完整的操作过程：从创建wyp源表和test、test3目标表，到执行多表插入语句，最后验证各表数据的全部步骤。

Slide 61 — 方式四：CREATE TABLE AS SELECT（CTAS）

第四种方式叫做 CTAS，全称是 CREATE TABLE AS SELECT。

这种方式非常方便（fāngbiàn），可以在一条语句中同时完成两件事：

第一，创建一个新表。

第二，把查询结果加载到这个新表中。

也就是说，一步到位（yībù dàowèi），不需要先建表再导入数据。

新表的结构（jiégòu）会自动根据查询结果来创建。

```
CREATE table [IF NOT EXISTS] tablename
AS SELECT select_statement1 FROM from_statement;
```

Slide 62 — 例题11：CTAS 创建 test4 表

例题11要求把 wyp 表的数据查询后加载到 test4 表中。

【实习命令 / 实践操作】CTAS创建表并加载数据

```
hive> CREATE TABLE test4 AS SELECT * FROM wyp;
```

就这么简单！一条命令，test4 表就创建好了，而且里面已经有数据了。

```
--step01：  查看数据库中的 wyp 表的数据 其中，由图 5-37 所示，该表中共有 7 条数据。
SELECT * from wyp;


--step02：  创建数据库中的表 test4
--在数据库 unit5db 中，创建数据库中的表 test4，
--并在创建的同时，将表 wyp中的数据加载到表 test4 中：
CREATE TABLE test4 AS SELECT id, tel
FROM wyp;                    --시간걸림 (19.4초)


--step03：  验证 test4 表中的数据
--查看 test4 表中的数据。
--由结果可知，整个表中共有 7 条最新的数据，加载数据成功。
SELECT * FROM test4;
```

Slide 63-64 — 例题12：CTAS 创建 order_info_as_select_load 表

例题12是把 order_info_load 表的数据加载到新表 order_info_as_select_load 中。

Step 01：查看源表数据

【实习命令 / 实践操作】查看源表数据

```
hive> SELECT * FROM order_info_load;
```

该表中共有13条数据。

Step 02：创建新表并加载数据

【实习命令 / 实践操作】CTAS方式创建表并加载

```
hive> CREATE TABLE order_info_as_select_load
      AS SELECT * FROM order_info_load;
```

执行成功后，新表 order_info_as_select_load 就自动创建了，

而且里面的数据和 order_info_load 表一模一样（yīmú yīyàng）。

Slide 65 — 方式五：Import 导入 Hive 表数据

第五种方式是 Import 导入。

这种方式一般配合（pèihé）Export 一起使用。

先用 Export 把数据导出，然后再用 Import 把数据导入。

这种方式主要用在什么场景（chǎngjǐng）呢?

主要用于两个 Hadoop 平台集群之间的 Hive 表迁移。

比如，你要把一个集群上的数据搬到另一个集群上去。

语法格式很简单：

【实习命令 / 实践操作】Import语法

```
hive> IMPORT TABLE tablename FROM 'filepath';
```

Slide 66-68 — 例题13：Import 导入数据

例题13要求把 HDFS 上已有的文件加载到 order_info_import_load 表中。

操作过程首先确认 HDFS 上有我们之前 Export 导出的数据，

然后用 Import 命令把数据导入到新表中。

【实习命令 / 实践操作】Import导入数据

```
hive> IMPORT TABLE order_info_import_load
      FROM '/tmp/order_info_load_out5';
```

```
--step01：  查看 HDFS 文件系统中/tmp 的目录结构。导出数据之前，
--查看 HDFS 文件系统中/tmp 的目录结构，
--其中没有/tmp/order_info_load_out4 目录。
--具体操作命令如下：[root@node2 ~]# hadoop fs  -ls /tmp/
hdfs dfs -ls /tmp/

--step02：  将文件导出到 HDFS 相应目录中。
--首先，将文件导出到相应的目录中，
--本题目中所使用目录为 HDFS 上的文件系统/tmp/order_info_load_out4 目录。
Export table order_info_load to '/tmp/order_info_load_out4';
--此处，在导入之前，首先将需要导入的资源准备充分，
--此处为准备资源的部分。
--只是此次的资源，是通过导出得来的，而不是本来就准备好的，
--这样做的目的是为了读者可以更好的理解 Hive 生态系统中导入和导出的实际应用。

--step03：  再次查看 HDFS 文件系统中/tmp 的目录结构
hdfs dfs -ls /tmp/
--由结果可知，其中/tmp/order_info_load_out4 已经存在。
```

```
--step04：  创建表 order_info_import_load
--在数据库 unit5db 中，创建所需表 order_info_import_load，该表为目标表。
CREATE TABLE order_info_import_load LIKE order_info_load;
--由结果可知，其中 order_info_import_load 表中无数据。
SELECT * FROM order_info_import_load;

--step05：  验证是否导入成功。
--其中，  将导出操作所得数据 /tmp/order_info_load_out4
--导 入 到 表order_info_import_load 中
IMPORT TABLE order_info_import_load FROM '/tmp/order_info_load_out4';
--再次查看表 order_info_import_load 中的数据
--共有 13 条数据，导入操作成功
SELECT * FROM order_info_import_load;
```

【图片说明】展示了Import操作的完整过程，包括查看HDFS目录确认导出文件存在，执行Import命令，以及验证导入结果。

Slide 69 — 5.1.3 数据导入小结

好的，到这里数据导入的五种方式我们全部学完了。

让我们来回顾（huígù）一下：

第一种：INSERT INTO VALUES —— 适合插入少量测试数据

第二种：LOAD DATA —— 最常用，从本地或 HDFS 加载文件

第三种：INSERT ... SELECT —— 从其他表查询后导入

第四种：CREATE TABLE AS SELECT —— 建表和导入一步完成

第五种：IMPORT —— 用于集群间的数据迁移

💡 考试重点：LOAD DATA 和 INSERT ... SELECT 是最常考的，大家一定要掌握（zhǎngwò）。

Slide 70 — 任务5-2 数据导出（章节封面）

好的同学们，学完了数据导入，我们现在来学数据导出。

数据导出，英文叫 Data Export。

就是把 Hive 表里面的数据取出来，导出到本地文件系统或者 HDFS 上。

Slide 71 — 5.2.1 任务描述

在 Hive 的日常使用中，经常需要把表中的数据导出来。

Hive 提供了多种导出方式，但是面对不同的数据量、不同的需求（xūqiú），

如果随意（suíyì）使用某种导出方式，可能会导致导出时间过长，

导出的结果不满足需求，甚至造成集群资源（zīyuán）不必要的浪费（làngfèi）。

所以我们需要根据不同的场景选择最合适（héshì）的导出方式。

Slide 72 — 5.2.1 任务准备知识

从 Hive 中导出数据，实际上就是从 HDFS 中导出数据。

因为 Hive 的存储依赖（yīlài）于 HDFS 分布式文件系统。

语法中的关键字在前面已经讲过了，这里就不重复了。

Slide 73 — 5.2.3 数据导出方式总览

Hive 数据导出主要有四种方式：

第一种：Insert 导出 —— 通过查询语句导出

第二种：Hadoop 命令导出 —— 用 hdfs dfs -get 命令

第三种：Hive Shell 命令导出 —— 用 hive -e/-f 命令

第四种：Export 命令导出 —— 用于集群间迁移

下面我们一个一个来学习。

Slide 73 — 方式一：Insert 语句导出

第一种导出方式是用 INSERT 语句导出。

这种方式是把查询结果写入到文件系统中。

可以导出到本地文件系统，也可以导出到 HDFS。

Slide 74-78 — 例题14：导出到本地文件系统

例题14要求把 person 表中的数据导出到本地 /usr/datas/out1 目录中。

Step 01：查看本地目录

【实习命令 / 实践操作】查看本地目录结构

```
[root@node2 ~]# cd /usr/datas/
[root@node2 datas]# ll
```

查看本地目录，确认 out1 目录还不存在。

Step 02：查看源表数据

【实习命令 / 实践操作】查看person表

```
hive> SELECT * FROM person;
```

person 表中共有80条记录。

Step 03：执行导出命令

【实习命令 / 实践操作】Insert导出到本地

```
hive> INSERT OVERWRITE LOCAL DIRECTORY '/usr/datas/out1'
      SELECT persid, firstname, lastname FROM person;
```

注意关键字：

INSERT OVERWRITE —— 表示覆盖方式写入

LOCAL DIRECTORY —— 表示导出到本地目录（如果没有 LOCAL 就是导出到 HDFS）

Step 04：验证导出结果

【实习命令 / 实践操作】验证导出结果

```
[root@node2 datas]# cd /usr/datas/out1
[root@node2 out1]# ll
[root@node2 out1]# cat 000000_0
```

【图片说明】图5-49至5-54展示了完整的导出过程。先查看了本地目录（out1不存在），然后执行了INSERT OVERWRITE LOCAL DIRECTORY命令，最后验证out1目录已自动生成，里面有80条导出的数据。

Slide 79-81 — 例题15：格式化导出到本地

例题15和例题14类似，但这次我们要按照格式化（géshìhuà）标准导出。

也就是说，我们要指定字段（zìduàn）之间的分隔符（fēngéfú）。

【实习命令 / 实践操作】格式化导出

```
hive> INSERT OVERWRITE LOCAL DIRECTORY '/usr/datas/out2'
      ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
      SELECT persid, firstname, lastname FROM person;
```

这里多了 ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' 这部分，

意思是用逗号（dòuhào）作为字段的分隔符。

这样导出的文件就是 CSV 格式，更容易阅读（yuèdú）和使用。

【实习命令 / 实践操作】验证格式化导出结果

```
[root@node2 datas]# cd /usr/datas/out2
[root@node2 out2]# ll
[root@node2 out2]# cat 000000_0
```

【图片说明】图5-55至5-58展示了格式化导出的过程和结果。导出后的数据以逗号分隔，整齐排列，共80条数据。

Slide 82-86 — 例题16：导出到 HDFS 文件系统

例题16要求把 person 表的数据导出到 HDFS 的 /tmp/out3 目录中。

和导出到本地的区别就是：去掉 LOCAL 关键字。

Step 01-02：查看 HDFS 目录和源表数据

【实习命令 / 实践操作】查看HDFS目录

```
[root@node2 ~]# hadoop fs -ls /tmp/
hive> SELECT * FROM person;
```

Step 03：导出到 HDFS

【实习命令 / 实践操作】Insert导出到HDFS

```
hive> INSERT OVERWRITE DIRECTORY '/tmp/out3'
      ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
      SELECT persid, firstname, lastname FROM person;
```

注意：这里没有 LOCAL，直接写 DIRECTORY，表示导出到 HDFS。

Step 04：验证导出结果

【实习命令 / 实践操作】验证HDFS导出结果

```
[root@node2 ~]# hadoop fs -ls /tmp/out3
[root@node2 ~]# cd /usr/datas/
[root@node2 datas]# hdfs dfs -get /tmp/out3/
[root@node2 datas]# cd out3
[root@node2 out3]# cat 000000_0
```

【图片说明】图5-59至5-63展示了导出到HDFS的完整过程，包括查看HDFS目录、执行导出命令、下载到本地验证。共导出80条数据。

Slide 87 — Insert 导出使用场景

Insert 导出方式有什么特点呢?

第一，这种方式会启动 MapReduce 任务来完成导出。
所以不适合导出很少的数据，因为即使数据量很小也会消耗（xiāohào）集群资源。

第二，适合对导出数据格式有要求的场景。
比如你可以指定数据文件的类型和字段分隔符。

第三，导出的路径只能指定到文件夹，不能指定到某个具体的文件。
而且文件夹下面可能会产生多个文件，不一定只有一个。

Slide 88 — 方式二：Hadoop 命令导出

第二种导出方式是用 Hadoop 命令来导出。
这种方式很直接（zhíjiē），就是用 hdfs dfs -get 命令把 HDFS 上的文件下载到本地。

语法格式非常简单：

【实习命令 / 实践操作】Hadoop命令导出语法

```
hdfs dfs -get [HDFS文件目录] [本地目录]
```

就像从网上下载（xiàzài）文件一样，-get 就是“获取“的意思。

Slide 89-95 — 例题17：用 Hadoop 命令导出数据

例题17要求把 HDFS 上 /tmp/out3 中的数据导出到本地 /usr/datas/hadoop 目录。

Step 01：确认 HDFS 数据

【实习命令 / 实践操作】查看HDFS目录

```
[root@node2 ~]# hdfs dfs -ls /tmp/
[root@node2 ~]# hdfs dfs -ls /tmp/out3
```

Step 02：创建本地目录并下载

【实习命令 / 实践操作】创建本地目录

```
[root@node2 datas]# mkdir /usr/datas/hadoop
```

【实习命令 / 实践操作】下载HDFS数据到本地

```
[root@node2 datas]# hdfs dfs -get /tmp/out3 /usr/datas/hadoop
```

Step 03：验证数据

【实习命令 / 实践操作】验证下载的数据

```
[root@node2 datas]# cd /usr/datas/hadoop
[root@node2 hadoop]# ll
[root@node2 hadoop]# cd out3
[root@node2 out3]# cat 000000_0
```

【图片说明】图5-64至5-70展示了完整的操作过程。先确认HDFS上 /tmp/out3 数据存在，然后创建了本地目录 /usr/datas/hadoop，用 hdfs dfs -get 命令下载数据，最后验证数据共80条，下载成功。

Slide 96 — 方式三：Hive Shell 命令导出

第三种方式是用 Hive Shell 命令导出。

也就是直接在 Linux 命令行中使用 hive 命令，加上参数（cānshù） -e 或者 -f 来导出数据。

语法格式：

【实习命令 / 实践操作】Hive Shell导出语法

```
hive -e "SQL语句" > filepath
hive -f SQL文件 > filepath
```

-e 后面直接跟 SQL 语句。

-f 后面跟一个包含 SQL 语句的文件。

> 是 Linux 的重定向（chóngdìngxiàng）符号，把查询结果输出到文件中。

Slide 97-101 — 例题18：Hive Shell 导出数据

例题18要求把 order_info_load 表的数据导出到 /usr/datas/hiveshell 目录。

Step 01：查看源表数据

【实习命令 / 实践操作】查看源表数据

```
hive> SELECT * FROM order_info_load;
```

表中共有13条数据。

Step 02：创建本地目录

【实习命令 / 实践操作】创建导出目录

```
[root@node2 datas]# cd /usr/datas
[root@node2 datas]# mkdir hiveshell
```

Step 03：执行 Hive Shell 导出

【实习命令 / 实践操作】Hive Shell -e 导出

```
hive -e 'select * from unit5db.order_info_load;' >
/usr/datas/hiveshell/order_info_load_out.txt
```

注意：这条命令是在 Linux 命令行执行的，不是在 Hive 里面！

因为这里用的是数据库名.表名的格式（unit5db.order_info_load），

所以不需要先 USE unit5db。

【实习命令 / 实践操作】验证导出结果

```
[root@node2 datas]# cd hiveshell
[root@node2 hiveshell]# ll
[root@node2 hiveshell]# cat order_info_load_out.txt
```

【图片说明】图5-71至5-73展示了完整的操作过程，包括查看源表数据、创建目录、执行导出命令和验证结果。导出文件order_info_load_out.txt中包含了13条完整数据。

Slide 102 — Hive Shell 导出使用场景

Hive Shell 导出方式的特点：

第一，不会启动 MapReduce 任务，所以速度快，适合数据量较少的情况。

第二，支持指定导出的文件名，而且导出结果只有一个文件。

第三，只能导出到本地文件系统，不能导出到 HDFS。

第四，不支持指定导出文件的分隔符。

Slide 103 — 方式四：Export 语句导出

第四种方式是用 Export 语句导出。

前面我们在 Import 导入的时候提到过，Export 和 Import 是配对使用的。

主要用于两个 Hadoop 集群之间的 Hive 表迁移（qiānyí）。

语法格式：

【实习命令 / 实践操作】Export导出语法

```
hive> EXPORT TABLE tablename TO 'filepath';
```

Slide 104-108 — 例题19：Export 导出数据

例题19要求把 order_info_load 表的数据导出到 HDFS 的 /tmp/order_info_load_out5 目录。

Step 01：查看 HDFS 目录

【实习命令 / 实践操作】查看HDFS目录

```
[root@node2 ~]# hdfs dfs -ls /tmp/
```

确认 /tmp/order_info_load_out5 目录还不存在。

Step 02：执行 Export 导出

【实习命令 / 实践操作】Export导出

```
hive> EXPORT TABLE order_info_load TO '/tmp/order_info_load_out5';
```

Step 03-04：验证导出结果

【实习命令 / 实践操作】验证导出结果

```
[root@node2 ~]# hdfs dfs -ls /tmp/
[root@node2 ~]# hadoop fs -ls /tmp/order_info_load_out5
```

【图片说明】图5-74至5-76展示了Export导出的完整过程。导出前 /tmp 目录下没有 order_info_load_out5，执行Export后该目录自动生成，里面包含了导出的数据文件。

Slide 109 — Export 导出使用场景

Export 导出方式的特点：

第一，只能导出数据到 HDFS 上，不能导出到本地。

第二，导出速度比较快。

第三，主要适用于 Hive 数据的批量迁移（pīliàng qiānyí）。

简单记忆：Export 是打包搬走，适合整体迁移；Insert 导出是选择性提取，更灵活。

Slide 110 — 任务5-3 本章小结（章节封面）

好了同学们，到这里我们今天的主要内容就全部讲完了。

现在让我们来做一个总结。

Slide 111 — 5.3.1 本章小结内容

在这一章中，我们学了两大部分内容：

第一部分：数据导入（五种方式）

① 直接向表中加载数据 —— INSERT INTO ... VALUES

② 通过 Load 加载数据 —— LOAD DATA [LOCAL] INPATH ... INTO TABLE ...（重点！）

③ 通过查询加载数据 —— INSERT [OVERWRITE/INTO] TABLE ... SELECT ...（重点！）

④ 查询语句中创建表并加载数据 —— CREATE TABLE ... AS SELECT ...（重点！）

⑤ Import 导入 Hive 表数据 —— IMPORT TABLE ... FROM ...

第二部分：数据导出（四种方式）

① Insert 导出 —— INSERT OVERWRITE [LOCAL] DIRECTORY ...（重点！）

② Hadoop 命令导出 —— hdfs dfs -get ...（重点！）

③ Hive Shell 命令导出 —— hive -e/-f ... > filepath

④ Export 命令导出 —— EXPORT TABLE ... TO ...（重点！）

掌握（zhǎngwò）这些数据导入和导出的方法，

是我们日常使用 Hive 解决实际问题的重要基础（jīchǔ）。

💡 建议（jiànyì）同学们课后多练习，特别是 LOAD DATA 和 INSERT 相关的操作。

Slide 112 — 感谢您的观看

好的，同学们，今天的课就到这里了。

今天我们学了 Hive 数据操作中最核心的两个内容：数据导入和数据导出。

一共学了九种方法（五种导入 + 四种导出）。

课后请大家：

第一，回顾今天的笔记和讲义，把每种方式的语法格式记住。

第二，自己动手做一遍今天课上的例题。

第三，完成任务5-4的本章习题。

如果有任何问题，欢迎随时来问我。

谢谢大家！下课！