《Hive 数据仓库应用教程》

第7章 Hive 函数（hánshù）

—— 强 化 讲 义 原 稿（jiǎngyì yuángǎo）——

适用对象：专科（zhuānkē）大学生

语言：中文（口语化讲解）

【幻灯片 1】 项目7 Hive函数 — 封面（fēngmiàn）

同学们好！欢迎来到我们 Hive 数据仓库课程的第7讲。

今天我们要学习的内容是——Hive 函数（hánshù）。

什么是函数呢？简单来说，函数就像一个「小工具（gōngjù）」。
你给它一些数据，它帮你处理（chǔlǐ）一下，然后返回（fǎnhuí）一个结果。
比如，你给它一个数字 3.7，它帮你四舍五入变成 4，这就是函数的作用。

今天这一讲非常重要，因为在实际工作中，我们写 SQL 查询的时候，

几乎每条语句（yǔjù）都会用到函数。

学会了函数，你的 SQL 能力会大大提升（tíshēng）！

【幻灯片 2】 本讲目录（mùlù）

我们先看一下今天的学习目录，一共有4个任务（rènwù）：

任务 7-1：Hive 函数概述（gàishù）及分类标准（biāozhǔn）

→ 先搞清楚 Hive 里面有哪些类型的函数，怎么分类的。

任务 7-2：Hive 内置函数（nèizhì hánshù）

→ 学习 Hive 自带的各种函数，比如字符串函数、日期函数等。

任务 7-3：Hive 用户自定义函数（zìdìngyì hánshù）— UDF、UDAF、UDTF

→ 当内置函数不够用的时候，我们自己写函数！

任务 7-4：案例（ànlì）— UDF实现手机号****加密（jiāmì）

→ 通过一个实际案例来练习写自定义函数。

【幻灯片 3】 任务 7-1 Hive 函数概述及分类标准

好，我们正式开始第一个任务。

这个部分主要是让大家对 Hive 函数有一个整体（zhěngtǐ）的认识。

先不着急学具体的函数，而是先了解（liǎojiě）一下全局（quánjú）。

【幻灯片 4】 7.1.1 Hive 函数概述及分类标准 — 概述

大家都知道，我们之前学过 MySQL，对吧？

MySQL 里面有很多内置函数，比如 COUNT、SUM、CONCAT 这些。

Hive 也是一样的！

Hive 就跟 MySQL、Oracle 这些传统的关系型数据库（guānxì xíng shùjùkù）一样，

自带了大量（dàliàng）的内置函数，我们可以直接拿来用。

同时呢，如果内置函数满足不了你的需求（xūqiú），

Hive 还支持你自己开发（kāifā）函数，这就叫「用户自定义函数」。

Hive 的内置函数从功能上主要分为以下几大类：

1. 算术函数（suànshù hánshù）— 做数学运算的
2. 聚合函数（jùhé hánshù）— 比如求和、计数
3. 日期时间函数（rìqī shíjiān hánshù）— 处理日期
4. 条件函数（tiáojiàn hánshù）— 做条件判断
5. 字符串函数（zìfúchuàn hánshù）— 处理文本

使用这些函数可以让我们写程序更灵活（línghuó）、可读性更好、执行（zhíxíng）速度更快。

【幻灯片 5】 7.1.1 函数探索和说明

那 Hive 到底有多少个函数呢？我们怎么查看呢？

Hive 提供了两个非常有用的命令：

第一个命令：show functions

这个命令可以列出（lièchū）Hive 当前所有可用的函数。

你在 Hive 的命令行里输入这个，就会看到一长串函数名。

第二个命令：describe function extended 函数名

如果你想知道某个函数怎么用，就用这个命令查看它的详细说明。

【图片说明】幻灯片上展示了一个终端截图（jiétú），

画面显示的是在 Beeline 里执行 desc function extended count; 命令的结果。

可以看到 count 函数有三种用法：

- count(*) — 统计所有行数，包括 NULL 值
- count(expr) — 统计表达式（biǎodáshì）非空的行数
- count(DISTINCT expr) — 统计去重后的行数

还显示了它的类型是 BUILTIN（内置的）。

实习命令 — 函数探索

```
-- 查看所有函数
show functions;

-- 查看 count 函数的详细说明
desc function extended count;

-- 查看 concat 函数的详细说明
desc function extended concat;
```

【幻灯片 6】 7.1.1 分类标准（fēnlèi biāozhǔn）

接下来我们来看函数的分类标准。

Hive 的函数分为两大类（liǎng dà lèi）：

A类：内置函数（Built-in Functions）

就是 Hive 已经帮我们写好的函数，直接拿来用就行。

内置函数又可以细分为：

- 数值类型函数 — 做数学计算
- 日期类型函数 — 处理时间日期
- 字符串类型函数 — 处理文本字符串
- 集合函数 — 处理集合（数组、Map等）
- 条件函数 — 做判断逻辑

B类：用户定义函数 UDF（User-Defined Functions）

当内置函数不够用的时候，用户自己开发的函数。

用户定义函数又根据输入输出的行数分为3种：UDF、UDAF、UDTF。
后面我们会详细讲。

【图片说明】幻灯片上有一个树状图（shùzhuàng tú），
左边是 Built-in Functions（内置函数），包含5个子分类；
右边是 User-Defined Function（用户定义函数），包含3个子分类：
Regular UDF、UDAF、UDTF。中间有一个 Hive 的蜜蜂标志。

【幻灯片 7】 7.1.1 用户定义函数 UDF 分类标准

现在我们重点看一下用户定义函数的三种类型。
这个分类标准非常重要，一定要记住！

① UDF — 普通函数（pǔtōng hánshù），一进一出

UDF 是 User-Defined Function 的缩写。
它的特点是：你给它一行数据，它返回一行结果。
举个例子：round(3.7) → 4，输入一个数字，返回一个数字。

② UDAF — 聚合函数（jùhé hánshù），多进一出

UDAF 是 User-Defined Aggregation Function 的缩写。
A 就是 Aggregation（聚合）的意思。
它的特点是：你给它很多行数据，它返回一行结果。
举个例子：sum(salary) 把所有员工的工资加起来，返回一个总数。

③ UDTF — 表生成函数（biǎo shēngchéng hánshù），一进多出

UDTF 是 User-Defined Table-Generating Functions 的缩写。
T 就是 Table-Generating（生成表）的意思。
它的特点是：你给它一行数据，它返回多行结果。
举个例子：explode 函数可以把一个数组展开成多行。

【图片说明】幻灯片上画了三个框，分别标注 UDF、UDAF、UDTF。
UDF 框：左边一个箭头进去，右边一个箭头出来（一进一出）。
UDAF 框：左边三个箭头进去，右边一个箭头出来（多进一出）。
UDTF 框：左边一个箭头进去，右边三个箭头出来（一进多出）。

【幻灯片 8-9】 7.1.1 UDF 分类标准扩大化（kuòdàhuà）

这里有一个很重要的概念（gàiniàn），大家要注意！

UDF、UDAF、UDTF 这个分类标准，

本来是针对「用户自己开发的函数」来说的。

但是！这个标准其实可以扩大到 Hive 的所有函数！

包括内置函数和用户自定义函数。

为什么呢？因为不管是什么函数，它一定有输入和输出。

从「输入几行、输出几行」这个角度来分类，完全没有问题。

所以大家千万不要被 UD（User-Defined）这两个字母所迷惑（míhuò），

以为只有用户自己写的函数才能这么分类。其实不是的！

举个例子：Hive 官方文档里，

把内置的 count、sum 这些聚合函数，也归类为「内置 UDAF 类型」。

【图片说明】幻灯片上展示了 Hive 官方文档的截图，

标题是 Built-in Aggregate Functions (UDAF)，

表格列出了 count 和 sum 函数的返回类型、签名和描述。

【幻灯片 10】 任务 7-2 Hive 内置函数

好的，概述部分我们讲完了。

接下来进入第二个任务——学习 Hive 的内置函数。

这个部分内容比较多，但都很实用（shíyòng），
大家跟着我一个一个来看。

【幻灯片 11】 7.2.1 Hive 内置函数 — 概述

内置函数，英文叫 built-in function，

就是 Hive 开发团队已经帮我们实现好的函数，拿来直接用就可以了。

官方文档地址在幻灯片上有，大家可以记一下。

如果以后工作中遇到不会用的函数，可以去那里查。

内置函数根据功能分类，一共有8大类：

【图片说明】幻灯片上用词云（cí yún）的方式展示了8种函数类型：
日期函数、数学函数、条件函数、字符串函数、
数据脱敏函数、集合函数、类型转换函数、其他杂项函数。

下面我们一个一个来看每种类型有哪些常用的函数。

【幻灯片 12】 7.2.1 (1/8) String Functions 字符串函数

第一类是字符串函数，用来处理文本数据。

这类函数非常常用！我来给大家介绍几个重要的：

📌 length — 获取字符串的长度。比如 length('hello') = 5
📌 reverse — 反转字符串。比如 reverse('abc') = 'cba'
📌 concat — 拼接（pīnjiē）字符串。比如 concat('hi',' ','world') = 'hi world'
📌 concat_ws — 带分隔符（fēngé fú）的拼接。
比如 concat_ws('-','2024','01','15') = '2024-01-15'
📌 substr / substring — 截取（jiéqǔ）字符串的一部分
📌 upper / ucase — 转大写。比如 upper('hello') = 'HELLO'
📌 lower / lcase — 转小写
📌 trim — 去掉（qùdiào）前后的空格
📌 split — 分割字符串，返回数组

另外还有一些高级的：
📌 regexp_replace — 用正则表达式（zhèngzé biǎodáshì）替换
📌 regexp_extract — 用正则表达式提取（tíqǔ）
📌 parse_url — 解析 URL
📌 get_json_object — 解析 JSON 数据

实习命令 — 字符串函数

```
-- 获取字符串长度
select length('hello world');

-- 反转字符串
select reverse('hello');

-- 拼接字符串
select concat('hello', ' ', 'world');

-- 带分隔符拼接
select concat_ws('-', '2024', '01', '15');

-- 截取字符串（从第2个字符开始，取3个）
```

```
select substr('hello world', 2, 3);

-- 转大写
select upper('hello');

-- 去空格
select trim('  hello  ');

-- 分割字符串
select split('a-b-c-d', '-');
```

【幻灯片 13】 7.2.1 (2/8) Date Functions 日期函数

第二类是日期函数，专门处理日期和时间。
在数据仓库（shùjù cāngkù）里，日期处理特别频繁（pínfán），所以这些函数很重要。

- 📌 current_date — 获取当前日期。比如返回 '2024-01-15'
- 📌 current_timestamp — 获取当前时间戳（shíjiān chuō），精确到秒
- 📌 from_unixtime — 把 UNIX 时间戳转成日期格式
- 📌 unix_timestamp — 把日期转成 UNIX 时间戳（一个整数）
- 📌 to_date — 从时间戳里提取日期部分
- 📌 year, month, day — 提取年、月、日
- 📌 hour, minute, second — 提取小时、分钟、秒
- 📌 weekofyear — 获取是一年中的第几周
- 📌 datediff — 计算两个日期之间相差多少天
- 📌 date_add — 日期加上几天
- 📌 date_sub — 日期减去几天

比如你想知道「今天到过年还有多少天」，就可以用 datediff。

想知道「5天后是几号」，就可以用 date_add。

实习命令 — 日期函数

```
-- 获取当前日期
select current_date;

-- 获取当前时间戳
select current_timestamp;

-- 计算两个日期相差多少天
select datediff('2024-12-31', '2024-01-01');

-- 日期加10天
select date_add('2024-01-15', 10);

-- 日期减5天
select date_sub('2024-01-15', 5);
```

```
-- 提取年月日
select year('2024-01-15');
select month('2024-01-15');
select day('2024-01-15');
```

【幻灯片 14】 7.2.1 (3/8) Mathematical Functions 数学函数

第三类是数学函数，用来做数学运算（yùnsuàn）。

- 📌 round — 四舍五入（sìshě wǔrù）取整。比如 round(3.7) = 4
  还可以指定精度（jīngdù）：round(3.1415, 2) = 3.14
- 📌 floor — 向下取整。比如 floor(3.7) = 3（直接去掉小数部分）
- 📌 ceil — 向上取整。比如 ceil(3.1) = 4（只要有小数就进一位）
- 📌 rand — 生成（shēngchéng）0到1之间的随机数（suíjī shù）
- 📌 abs — 取绝对值（juéduì zhí）。比如 abs(-5) = 5
- 📌 bin — 转换成二进制（èr jìnzhì）
- 📌 conv — 进制转换，可以在任意进制间转换

实习命令 — 数学函数

```
-- 四舍五入
select round(3.7);
select round(3.1415, 2);

-- 向下取整
select floor(3.7);

-- 向上取整
select ceil(3.1);

-- 取随机数
select rand();

-- 取绝对值
select abs(-5);
```

【幻灯片 15】 7.2.1 (4/8) Collection Functions 集合函数

第四类是集合函数，用来处理数组（shùzǔ）和 Map 类型的数据。

什么是集合？简单来说就是一组数据放在一起。
比如一个数组 [1, 2, 3]，或者一个 Map {'name':'张三', 'age':'20'}。

- 📌 size — 获取集合的大小（元素个数）
  size(Map) 或 size(Array) 都可以
- 📌 map_keys — 获取 Map 的所有键（jiàn）
- 📌 map_values — 获取 Map 的所有值
- 📌 array_contains — 判断数组是否包含某个元素
- 📌 sort_array — 对数组进行排序（páixù）

实习命令 — 集合函数

```
-- 获取数组大小
select size(array(1,2,3,4,5));

-- 判断数组是否包含某个值
select array_contains(array(1,2,3), 2);

-- 数组排序
select sort_array(array(3,1,2,5,4));
```

【幻灯片 16-17】 7.2.1 (5/8) Conditional Functions 条件函数

第五类是条件函数，用来做条件判断（tiáojiàn pànduàn）和逻辑转换。
这类函数在实际工作中用得非常多！

if 函数

语法：if(条件, 真的值, 假的值)
跟 Excel 里的 IF 函数一模一样。
例如：if(1=2, 100, 200) → 结果是 200（因为 1 不等于 2）

isnull / isnotnull 函数

判断一个值是不是 NULL（空值）。
isnull('allen') → false　　isnull(null) → true

nvl 函数

空值转换函数。如果第一个值是 NULL，就返回第二个值。
nvl('allen', 'itcast') → 'allen'
nvl(null, 'itcast') → 'itcast'

COALESCE 函数

从多个值中找到第一个不为 NULL 的值返回。

CASE WHEN 函数

条件转换，类似于编程里的 switch-case。
语法：CASE a WHEN b THEN c [WHEN d THEN e] [ELSE f] END

实习命令 — 条件函数

```
-- if条件判断
select if(1=2, 100, 200);
select if(sex='男', 'M', 'W') from student limit 3;

-- 空判断函数
```

```
select isnull('allen');
select isnull(null);
select isnotnull('allen');
select isnotnull(null);

-- 空值转换函数
select nvl('allen', 'itcast');
select nvl(null, 'itcast');

-- CASE WHEN 示例
select
  case
    when score >= 90 then '优秀'
    when score >= 60 then '及格'
    else '不及格'
  end as grade
from student;
```

【幻灯片 18】 7.2.1 (6/8) Type Conversion Functions 类型转换函数

第六类是类型转换函数，主要用于数据类型之间的转换。

最重要的就是 cast 函数：
语法：cast(值 as 目标类型)

它可以把一种数据类型转换成另一种。
比如：

- cast(12.14 as bigint) → 12 （小数变整数）
- cast(12.14 as string) → '12.14' （数字变字符串）
- cast('hello' as int) → NULL （字母没法变数字，所以返回空）

⚠ 注意！如果转换失败（shībài），不会报错，而是返回 NULL。

实习命令 — 类型转换函数

```
-- 小数转整数
select cast(12.14 as bigint);

-- 数字转字符串
select cast(12.14 as string);

-- 无法转换的情况（返回NULL）
select cast('hello' as int);
```

【幻灯片 19】 7.2.1 (7/8) Data Masking Functions 数据脱敏函数

第七类是数据脱敏（tuōmǐn）函数。

什么是数据脱敏？就是把敏感数据（mǐngǎn shùjù）隐藏起来。

比如手机号 13912345678，脱敏后变成 139****5678。

再比如身份证号、银行卡号，都需要脱敏处理。

Hive 提供了以下脱敏函数：

- 📌 mask — 对字符串进行掩码（yǎnmǎ）处理
- 📌 mask_first_n — 只遮蔽前 n 个字符
- 📌 mask_last_n — 只遮蔽后 n 个字符
- 📌 mask_show_first_n — 只显示前 n 个字符，其余遮蔽
- 📌 mask_show_last_n — 只显示后 n 个字符，其余遮蔽
- 📌 mask_hash — 用哈希值（hāxī zhí）替换原数据，完全不可逆

【幻灯片 20】 7.2.1 (8/8) Misc. Functions 其他杂项函数

最后一类是其他杂项（záxiàng）函数，包含一些不好归类的函数。

📌 java_method — 在 Hive 中调用 Java 方法
📌 reflect — 反射（fǎnshè）函数，也是调用 Java 方法
📌 hash — 获取哈希值
📌 current_user() — 获取当前用户
📌 current_database() — 获取当前数据库名
📌 version() — 获取 Hive 版本信息
📌 sha1 — SHA-1 加密
📌 sha2 — SHA-2 家族加密（SHA-224, SHA-256, SHA-384, SHA-512）
📌 md5 — MD5 加密
📌 crc32 — CRC32 校验

这些函数平时用得不多，但在特殊场景（chǎngjǐng）下很有用。

比如需要对数据进行加密或者校验（jiàoyàn）的时候。

【幻灯片 21】 任务 7-3 Hive 用户自定义函数（UDF、UDAF、UDTF）

好的，8种内置函数我们都过了一遍。

接下来进入第三个任务——用户自定义函数。

有时候，Hive 自带的函数不能满足我们的需求，

这时候就需要我们自己来开发函数了！

这就是 UDF — User-Defined Function（用户自定义函数）。

【幻灯片 22】 7.3.1 用户自定义函数 — 分类

前面我们已经提到过，用户自定义函数根据输入输出分为3类。

现在我们来详细（xiángxì）学习每一类。

再强调（qiángdiào）一遍：

- ✅ UDF — 普通函数，一进一出（输入一行，输出一行）
- ✅ UDAF — 聚合函数，多进一出（输入多行，输出一行）
- ✅ UDTF — 表生成函数，一进多出（输入一行，输出多行）

【图片说明】和前面第7张幻灯片的图一样，
三个框分别表示 UDF（一进一出）、UDAF（多进一出）、UDTF（一进多出），
用箭头直观地展示了输入输出的关系。

【幻灯片 23】 7.3.1 (1/3) UDF 普通函数

先看 UDF，也就是普通函数。
它的特点非常简单：一进一出。
给它一行数据，它返回一行结果。

【图片说明】幻灯片上有两个部分：
左边是 UDF 的示意图（shìyì tú），一个箭头进一个箭头出。
右边是一个执行截图，执行了 select round(3.14159256);
结果显示 _c0 列的值为 3。

round 就是一个典型（diǎnxíng）的 UDF：
你输入一个小数 3.14159256，它返回一个整数 3。
一行进去，一行出来，非常直接（zhíjiē）。

实习命令 — UDF 普通函数

```
select round(3.14159256);
select round(3.14159256, 2);
select length('hello');
select upper('hello');
```

【幻灯片 24-25】 7.3.1 (2/3) UDAF 聚合函数

接下来看 UDAF，也就是聚合函数。

A 代表 Aggregation（聚合）。
它的特点是：多进一出。输入多行数据，输出一行结果。

最常见的聚合函数大家都用过：

- count — 统计总行数
- sum — 求和
- avg — 求平均值（píngjūn zhí）
- min — 求最小值
- max — 求最大值

这些你们之前在 SQL 里都学过了。

但 Hive 还有两个特殊的聚合函数，大家要注意：

collect_set(col) — 收集去重

把某一列的所有值收集到一个数组里，并且自动去掉重复的。

比如 student 表的 sex 列有很多「男」和「女」，

collect_set(sex) 的结果是 [「男」,「女」]，只有两个值。

collect_list(col) — 收集不去重

跟 collect_set 类似，但不去重。

collect_list(sex) 的结果会把所有的男男女女都列出来。

【图片说明】幻灯片展示了两个查询结果的截图：

collect_set(sex) 返回 [「男」,「女」]（去重了）；

collect_list(sex) 返回一长串 [「男」,「女」,「女」,「男」,「男」,...]（没去重）。

实习命令 — UDAF 聚合函数

```
-- 基本聚合函数
select count(*) from student;
```

```
select sum(age) from student;
select avg(age) from student;

-- 收集函数（去重）
select collect_set(sex) from student;

-- 收集函数（不去重）
select collect_list(sex) from student;
```

【幻灯片 26】 7.3.1 (3/3) UDTF 表生成函数

最后看 UDTF，表生成函数。
T 代表 Table-Generating（生成表）。
它的特点是：一进多出。输入一行数据，输出多行结果。

最典型的 UDTF 就是 explode 函数！

explode 是什么意思？就是「炸开（zhà kāi）」、「展开（zhǎnkāi）」的意思。
它可以把一个数组「炸开」成多行。

【图片说明】幻灯片展示了执行 select explode(`array`(11,22,33)); 的结果：
输入的是一个数组 array(11,22,33)（一行数据），
输出的结果变成了3行：11、22、33。
右边标注了「输入一行 array 类型，返回3行 string 类型」。

UDTF 在处理嵌套数据的时候非常有用，

比如一个字段里存了用逗号分隔的多个标签，

你想把它们拆开成多行来分析，就可以用 UDTF。

实习命令 — UDTF 表生成函数

```
-- explode 展开数组
select explode(array(11, 22, 33));

-- explode 展开 Map
select explode(map('name','张三','age','20'));
```

【幻灯片 27】 任务 7-4 案例：UDF实现手机号****加密

好的，前面我们学了很多理论知识。

现在进入最后一个任务——通过一个实战（shízhàn）案例来应用所学！

这个案例是用自定义函数来实现手机号的加密处理。

【幻灯片 28】 7.4.1 案例背景（bèijǐng）和需求

在企业中处理数据的时候，经常会遇到敏感数据（mǐngǎn shùjù）。

比如用户的手机号、身份证号、银行卡号等。

这些数据不能直接暴露（bàolù）出来，需要做脱敏处理。

最常见的方式就是：把手机号中间4位用 **** 替换。
比如：13912345678 → 139****5678

Hive 内置函数虽然可以通过嵌套调用来实现，

但写起来很复杂，效率也不高。

所以我们要自己开发一个 UDF 来实现这个功能。

需求（xūqiú）如下：

1. 对输入数据进行非空判断和手机号位数判断
2. 校验手机号格式，符合规则的进行 **** 处理
3. 不符合手机号规则的数据直接原样返回，不处理

【幻灯片 29】 7.4.1 UDF 实现步骤（bùzhòu）

实现这个 UDF 一共分4个步骤：

第一步：在 PyCharm 中编写 Python 脚本（jiǎoběn）

→ 写好手机号掩码处理的逻辑

第二步：将脚本上传到 HDFS

→ 因为 Hive 运行在 Hadoop 集群上，文件需要放在 HDFS 里

第三步：在 Beeline 中注册（zhùcè）脚本

→ 告诉 Hive 有一个新的脚本可以用

第四步：使用 SELECT TRANSFORM 执行（zhíxíng）

→ 在 SQL 语句中调用我们的脚本

【幻灯片 30-31】 7.4.1 Python 脚本编写

我们来看 Python 脚本的代码：

核心代码讲解：

首先，定义（dìngyì）一个手机号的正则表达式：

PHONE_PATTERN = re.compile(r'^1[3-9]\d{9}$')

这个正则表达式的意思是：

^ → 字符串的开始

1 → 第一位必须是数字 1

[3-9] → 第二位必须是 3 到 9 之间的数字

\d{9} → 后面再跟9个任意数字

$ → 字符串结束

所以合起来就是：以1开头，第二位是3-9，总共11位数字。

这正好符合（fúhé）中国手机号的格式！

处理逻辑（luójí）：

1. 先检查输入是否为空（null check）
2. 去掉前后空格（strip）
3. 检查长度是否为11位，格式是否匹配（pǐpèi）正则
4. 如果都通过，就把中间4位替换成 ****

   phone[:3] + '****' + phone[7:]

   前3位 + **** + 后4位
5. 如果不符合，就原样返回

Python 脚本代码 — phone_mask.py

```
import re
import sys

# 手机号正则表达式（以1开头，第二位为3-9，共11位）
PHONE_PATTERN = re.compile(r'^1[3-9]\d{9}$')

def mask_phone(phone):
```

```
    """将手机号中间4位替换为****进行掩码处理"""
    if not phone:
        return phone
    phone = phone.strip()
    if len(phone) != 11 or not PHONE_PATTERN.match(phone):
        return phone
    return phone[:3] + '****' + phone[7:]

for line in sys.stdin:
    line = line.strip()
    print(mask_phone(line))
```

【幻灯片 32-34】 7.4.1 上传脚本和执行

脚本写好之后，需要上传到 HDFS：

上传脚本到 HDFS

```
hdfs dfs -put phone_mask.py /tmp/
```

上传好了之后，我们就可以在 Hive 中使用了。

执行步骤：

第一步：创建一个临时表（línshí biǎo），插入测试数据

第二步：用 SELECT TRANSFORM 调用我们的 Python 脚本

在 Hive 中执行 UDF

```
-- 1. 创建临时表
CREATE TEMPORARY TABLE temp_phones (phone STRING);

-- 2. 插入测试数据
INSERT INTO temp_phones VALUES
('13912345678'),
('01012345678'),  -- 非正常(不以1开头)
('12345');         -- 非正常(长度不够)

-- 3. 执行 UDF
SELECT TRANSFORM (phone)
USING 'python phone_mask.py'
AS (masked_phone STRING)
FROM temp_phones;
```

执行结果：

- 13912345678 → 139****5678 ✅ 正常手机号，成功脱敏
- 01012345678 → 01012345678 ❌ 不以1开头，原样返回
- 12345 → 12345 ❌ 长度不够11位，原样返回

看到了吗？我们的 UDF 完美（wánměi）地实现了需求：

合法的手机号做了脱敏，不合法的原样返回。太棒了！

【幻灯片 35】 总结（zǒngjié）与回顾（huígù）

好的，到这里我们第7讲的内容就全部讲完了。

让我们来快速回顾一下今天学了什么：

🔑 核心知识点（zhīshi diǎn）：

1. Hive 函数分为两大类：内置函数 和 用户自定义函数

2. UDF 分类标准：
   UDF（一进一出）、UDAF（多进一出）、UDTF（一进多出）
   这个分类标准可以应用到所有函数，不只是用户自定义的！

3. 内置函数有8大类：
   字符串函数、日期函数、数学函数、集合函数、
   条件函数、类型转换函数、数据脱敏函数、杂项函数

4. 用户自定义函数可以用 Python 编写，
   通过 SELECT TRANSFORM 在 Hive 中调用

5. 手机号脱敏案例展示了完整的 UDF 开发流程：

   编写脚本 → 上传 HDFS → 注册 → 执行

今天的内容到此结束（jiéshù）。

同学们回去以后，一定要动手练习这些函数，多写多练才能掌握！

谢谢大家！下次课见！