项目7 Hive函数



《Hive数据仓库应用教程》

目 录 / CONTENTS

1 任务7-1 Hive 函数概述及分类标准

2 任务7-2 Hive 内置函数

3 任务7-3 Hive 用户自定义函数（UDF、UDAF、UDTF）

4 任务7-4 案例：UDF实现手机号****加密

任务 7-1

Hive 函数概述及分类标准

7.1.1 Hive 函数概述及分类标准

Hive数据库同MySQL、Oracle等传统关系型数据库一样，都含有大量的内置函数,用户可以直接使用,同时也支持用户自定义函数以满足实际应用的特殊需求。函数是Hive数据库中非常重要的一-部分,使用函数可以极大地提高对数据库的管理效率。

Hive内部提供了大量丰富的函数供用户使用，这些函数从功能上主要分为算术函数、聚合函数、日期时间函数、条件函数以及字符串函数，这些函数被称为内置函数。通过Hive内置函数对数据进行处理，可以提高程序灵活性、可读性及执行速度。本任务将详细介绍Hive内置函数的使用方法。







7.1.1 Hive 函数概述及分类标准

概述:函数探索和说明

- Hive内建了不少函数， 用于满足用户不同使用需求，提高SQL编写效率：

1. 使用show functions查看当下的所有函数；
2. 通过describe function extended funcname来查看函数的使用方式

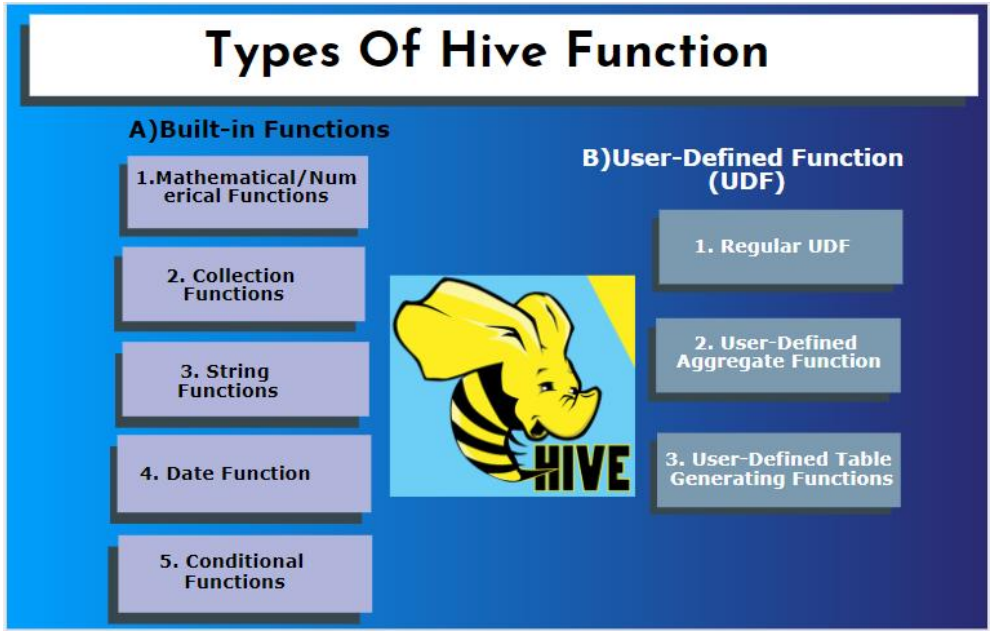
```
0: jdbc:hive2://localhost:10000> desc function extended count;
+----------------------------------------------------+
|                      tab_name                      |
+----------------------------------------------------+
| count(*) - Returns the total number of retrieved rows, including row
| count(expr) - Returns the number of rows for which the supplied expr
| count(DISTINCT expr[, expr...]) - Returns the number of rows for whi
| Function class:org.apache.hadoop.hive.ql.udf.generic.GenericUDAFCoun
| Function type:BUILTIN                              |
+----------------------------------------------------+
5 rows selected (0.077 seconds)
0: jdbc:hive2://localhost:10000>
```

Manual

7.1.1 Hive 函数概述及分类标准

分类标准

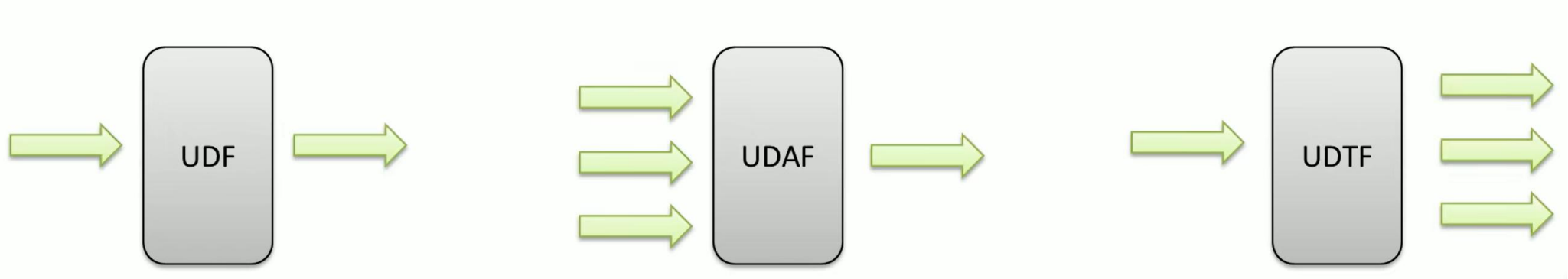
- Hive的函数分为两大类：**内置函数**（Built-in Functions）、**用户定义函数UDF**（User-Defined Functions）；
  - 内置函数可分为：数值类型函数、日期类型函数、字符串类型函数、集合函数、条件函数等;
  - 用户定义函数根据输入输出的行数可分为3类：UDF、UDAF、UDTF。



Manual

用户定义函数UDF分类标准

- UDF（User-Defined Function）普通函数，一进一出
- UDAF（User-Defined Aggregation Function）聚合函数，多进一出
- UDTF（User-Defined Table-Generating Functions）表生成函数，一进多出



7.1.1 Hive 函数概述及分类标准

UDF分类标准扩大化

- UDF分类标准本来针对的是用户自己编写开发实现的函数。UDF分类标准可以扩大到Hive的所有函数中：包括内置函数和用户自定义函数。

  因为不管是什么类型的函数，一定满足于输入输出的要求，那么从输入几行和输出几行上来划分没有任何问题。千万不要被UD（User-Defined）这两个字母所迷惑，造成视野的狭隘。

- 比如Hive官方文档中，针对聚合函数的标准就是内置的UDAF类型。

Built-in Aggregate Functions (UDAF)

The following built-in aggregate functions are supported in Hive:

Return Type	Name(Signature)	Description
BIGINT	count(*), count(expr), count(DISTINCT expr[, expr...])	count(*) - Returns the total number of retrieved rows, including rows containing NULL values. count(expr) - Returns the number of rows for which the supplied expression is non-NULL. count(DISTINCT expr[, expr]) - Returns the number of rows for which the supplied expression(s) are unique and non-NULL. Execution of this can be optimized with hive.optimize.distinct.rewrite.
DOUBLE	sum(col), sum(DISTINCT col)	Returns the sum of the elements in the group or the sum of the distinct values of the column in the group.

UDF分类标准扩大化

- UDF分类标准本来针对的是用户自己编写开发实现的函数。UDF分类标准可以扩大到Hive的所有函数中：包括内置函数和用户自定义函数。

  因为不管是什么类型的函数，一定满足于输入输出的要求，那么从输入几行和输出几行上来划分没有任何问题。千万不要被UD（User-Defined）这两个字母所迷惑，造成视野的狭隘。

- 比如Hive官方文档中，针对聚合函数的标准就是内置的UDAF类型。

任务 7-2

Hive 内置函数

7.2.1 Hive 内置函数

概述

- **内置函数（build-in）**指的是Hive开发实现好，直接可以使用的函数，也叫做内建函数。
- 官方文档地址：https://cwiki.apache.org/confluence/display/Hive/LanguageManual+UDF
- 内置函数根据应用归类整体可以分为8大种类型，我们将对其中重要的，使用频率高的函数使用进行详细讲解。



7.2.1 Hive 内置函数

（1/8） String Functions 字符串函数。

- 字符串长度函数：length
- 字符串反转函数：reverse
- 字符串连接函数：concat
- 带分隔符字符串连接函数：concat_ws
- 字符串截取函数：substr,substring
- 字符串转大写函数：upper,ucase
- 字符串转小写函数：lower,lcase
- 去空格函数：trim
- 左边去空格函数：ltrim
- 右边去空格函数：rtrim

- 正则表达式替换函数：regexp_replace
- 正则表达式解析函数：regexp_extract
- URL解析函数：parse_url
- json解析函数：get_json_object
- 空格字符串函数：space
- 重复字符串函数：repeat
- 首字符ascii函数：ascii
- 左补足函数：lpad
- 右补足函数：rpad
- 分割字符串函数: split
- 集合查找函数: find_in_set

7.2.1 Hive 内置函数

（2/8） Date Functions 日期函数。

- 获取当前日期: current_date
- 获取当前时间戳: current_timestamp
- UNIX时间戳转日期函数: from_unixtime
- 获取当前UNIX时间戳函数: unix_timestamp
- 日期转UNIX时间戳函数: unix_timestamp
- 指定格式日期转UNIX时间戳函数: unix_timestamp
- 抽取日期函数: to_date
- 日期转年函数: year

- 日期转月函数: month
- 日期转天函数: day
- 日期转小时函数: hour
- 日期转分钟函数: minute
- 日期转秒函数: second
- 日期转周函数: weekofyear
- 日期比较函数: datediff
- 日期增加函数: date_add
- 日期减少函数: date_sub

（3/8）Mathematical Functions 数学函数。

- 取整函数: round
- 指定精度取整函数: round
- 向下取整函数: floor
- 向上取整函数: ceil
- 取随机数函数: rand
- 二进制函数: bin
- 进制转换函数: conv
- 绝对值函数: abs

（4/8）Collection Functions 集合函数。

- 集合元素size函数: size(Map<K.V>) size(Array<T>)
- 取map集合keys函数: map_keys(Map<K.V>)
- 取map集合values函数: map_values(Map<K.V>)
- 判断数组是否包含指定元素: array_contains(Array<T>, value)
- 数组排序函数:sort_array(Array<T>)

（5/8）Conditional Functions 条件函数。

- 主要用于条件判断、逻辑判断转换这样的场合

•if条件判断: if(boolean testCondition, T valueTrue, T valueFalseOrNull)

•空判断函数: isnull(a)

•非空判断函数: isnotnull (a)

•空值转换函数: nvl(T value, T default_value)

•非空查找函数: COALESCE(T v1, T v2, ...)

•条件转换函数: CASE a WHEN b THEN c [WHEN d THEN e]* [ELSE f] END

•nullif(a, b): 如果a = b，则返回NULL；否则返回NULL。否则返回一个

•assert_true: 如果'condition'不为真，则引发异常，否则返回null

（5/8）Conditional Functions 条件函数。

```
----Conditional Functions 条件函数----------
describe function extended assert_true;

--使用之前课程创建好的student表数据
select *from student limit 3;

--if条件判断: if(boolean testCondition, T valueTrue, T valueFalseOrNull)
select if(1=2,100,200);
select if(sex ='男','M','W') from student limit 3;

--空判断函数: isnull( a )
select isnull("allen");
select isnull(null);

--非空判断函数: isnotnull ( a )
select isnotnull("allen");
select isnotnull(null);

--空值转换函数: nvl(T value, T default_value)
select nvl("allen","itcast");
select nvl(null,"itcast");
```

（6/8）Type Conversion Functions 条件函数。

- 主要用于显示的数据类型转换

```
---Type Conversion Functions 类型转换函数---
--任意数据类型之间转换：cast
desc function extended cast;
select cast(12.14 as bigint);
select cast(12.14 as string);
select cast("hello" as int));
```

（7/8）Data Masking Functions 数据脱敏函数 。

- 主要完成对数据脱敏功能， 屏蔽原始数据， 主要如下：

- mask
- mask_first_n(string str[, int n]
- mask_last_n(string str[, int n])
- mask_show_first_n(string str[, int n])
- mask_show_last_n(string str[, int n])
- mask_hash(string|char|varchar str)

（8/8）Misc. Functions 其他杂项函数

- hive调用java方法: java_method(class, method[, arg1[, arg2..]])
- 反射函数: reflect(class, method[, arg1[, arg2..]])
- 取哈希值函数:hash
- current_user()、logged_in_user()、current_database()、version()
- SHA-1加密: sha1(string/binary)
- SHA-2家族算法加密：sha2(string/binary, int) (SHA-224, SHA-256, SHA-384, SHA-512)
- crc32加密:
- MD5加密: md5(string/binary)

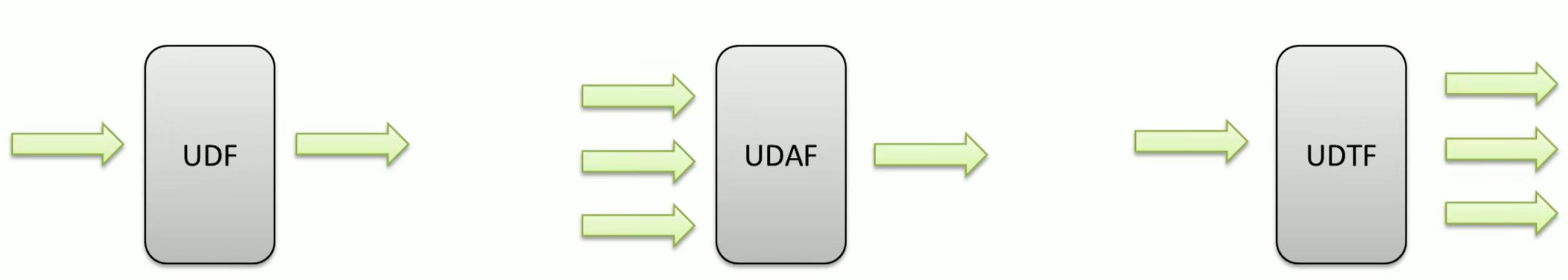
任务 7-3

Hive 用户自定义函数（UDF、UDAF、UDTF）

7.3.1 Hive 用户自定义函数

分类

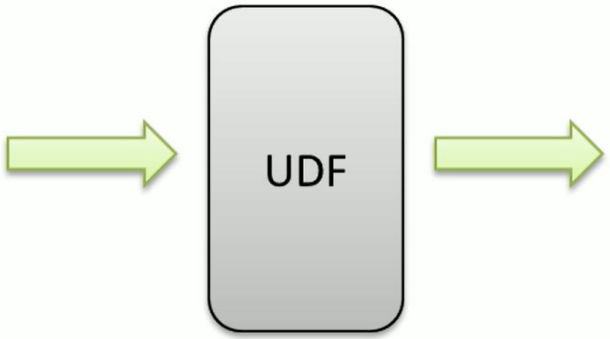
- 用户自定义函数简称UDF, 源自于英文user-defined function
- 根据函数输入输出的行数可以分为3类，分别是：
  - UDF（User-Defined Function）普通函数，一进一出
  - UDAF（User-Defined Aggregation Function）聚合函数，多进一出
  - UDTF（User-Defined Table-Generating Functions）表生成函数，一进多出



(1/3) UDAF 普通函数

- UDF 普通函数
- 特点是一进一出，也就是输入一行输出一行。

比如 `round` 这样的取整函数，接收一行数据，输出的还是一行数据。用户自定义函数简称UDF, 源自于英文 user-defined function




```
select round(3.14159256);
```

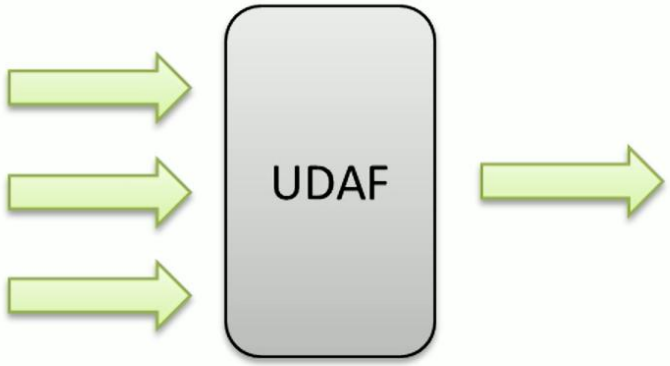
1 row

	_c0
1	3

(2/3) UDF 聚合函数

- UDAF 聚合函数， A所代表的单词就是Aggregation聚合的意思。
- 多进一出，也就是输入多行输出一行。

比如count、sum这样的函数。



- count:统计检索到的总行数。
- sum:求和
- avg:求平均
- min:最小值
- max:最大值
- 数据收集函数（去重）: collect_set(col)
- 数据收集函数（不去重）: **collect_list(col)**

(2/3) UDF 聚合函数

```
select sex from student;

select collect_set(sex) from student;
select collect_list(sex) from student;
```

```
select collect_set(sex) from student;
```

1 row

	_c0
1	["男","女"]

```
select collect_list(sex) from student;
```

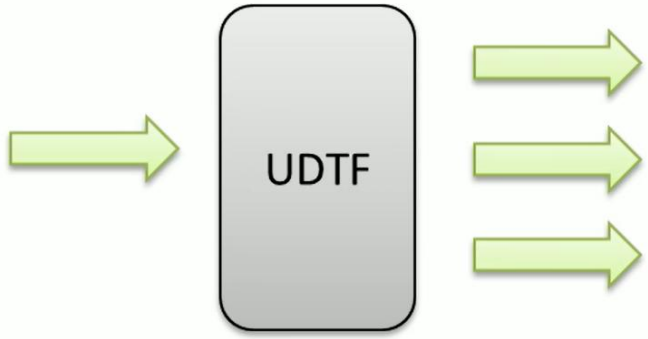
1 row

	_c0
1	["男","女","女","男","男","男","女","女","女","男","男","女","男","女","男","男"

(3/3) UDF 聚合函数

- UDTF **表生成函数**，T所代表的单词是Table-Generating表生成的意思。
- 特点是一进多出，也就是输入一行输出多行。

这类型的函数作用返回的结果类似于表，同时，UDTF函数也是我们接触比较少的函数。比如explode函数。




```
select explode(`array`(11,22,33));
```

3 rows

	col
1	11
2	22
3	33

输入一行array类型
返回3行string类型

任务 7-4

案例：UDF实现手机号****加密

案例：开发Hive UDF实现手机号****加密

在企业中处理数据的时候，对于敏感数据往往需要进行脱敏(masking)处理。比如手机号。我们常见的处理方式是将手机号中间4位进行****处理。

Hive中没有这样的函数可以直接实现功能，虽然可以通过各种函数的嵌套调用(nested call)最终也能实现，但是效率不高，现要求自定义开发实现Hive函数，满足上述需求。

1. 能够对输入数据进行非空判断(null check)、手机号位数判断
2. 能够实现校验手机号格式，把满足规则的进行****处理
3. 对于不符合手机号规则的数据直接返回，不处理

7.4.1案例：UDF实现手机号****加密

UDF实现步骤

1. 在 PyCharm 中编写 Python 脚本（手机号掩码处理）
2. 将脚本和环境上传至 HDFS
3. 在 Beeline 中使用 ADD FILE / ADD ARCHIVE 命令进行注册
4. 使用 SELECT TRANSFORM 执行

7.4.1案例：UDF实现手机号****加密

在 PyCharm 中编写

首先，在 PyCharm 中创建如下 Python 脚本：

```
# 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:]
```

7.4.1案例：UDF实现手机号****加密

在 PyCharm 中编写



🔍 正则表达式 `^1[3-9]\d{9}$` 详细解释

正则表达式	说明	含义
`^`	**起始锚点**	表示字符串的**开始**
`1`	**数字1**	第一位必须是数字 `1`
`[3-9]`	**字符类**	第二位必须是 `3,4,5,6,7,8,9` 中的一个（即3到9之间的**数字**）
`\d`	**数字**	等同于 `[0-9]`，表示一位**数字**
`{9}`	**重复量词**	前面的 `\d` **重复9次**
`$`	**结束锚点**	表示字符串的结束

7.4.1案例：UDF实现手机号****加密

将脚本和环境上传至 HDFS

上传至 HDFS

hdfs dfs -put phone_mask.py /tmp/

7.4.1案例：UDF实现手机号****加密

执行

```
CREATE TEMPORARY TABLE temp_phones (phone STRING);
INSERT INTO temp_phones VALUES
('13912345678'),
('01012345678'),  -- 비정상(1로 시작 안함)
('12345');        -- 비정상(길이 오류)

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

-- 결과:
-- 139****5678
-- 01012345678
-- 12345
```

7.4.1案例：UDF实现手机号****加密

将脚本和环境上传至 HDFS

上传至 HDFS

hdfs dfs -put phone_mask.py /tmp/

感谢您的观看

