**使用UDF实现oracle兼容函数to\_char()功能**

# 课题背景

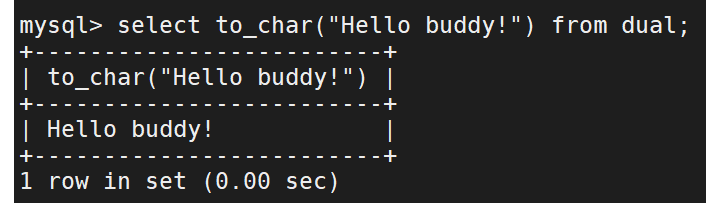
ORACLE是目前使用最广泛的商业数据库，其功能非常强大。

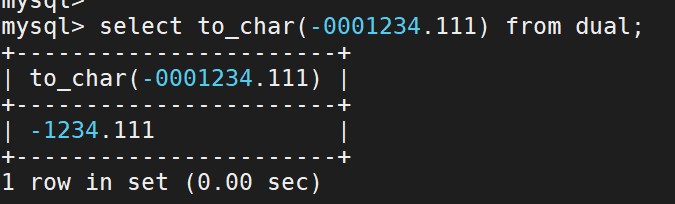
ORACLE存在这样一个函数to\_char()，该函数用于将一个表达式（数字、字符串、日期等类型）转化为char/varchar类型的字符串数据。

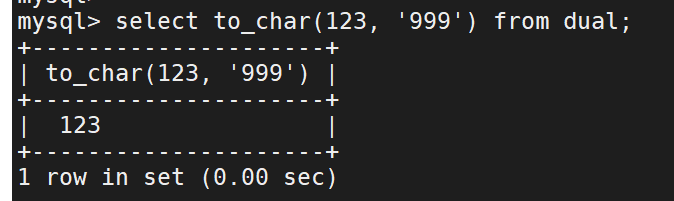
其官方定义的语法为：

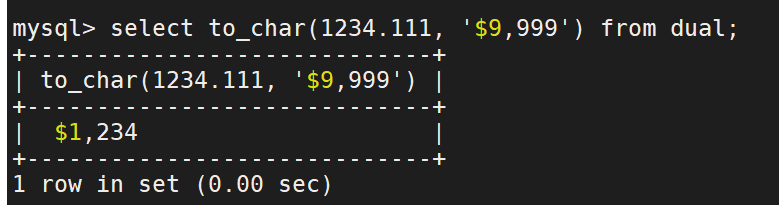
to\_char ( expr [, fmt] )

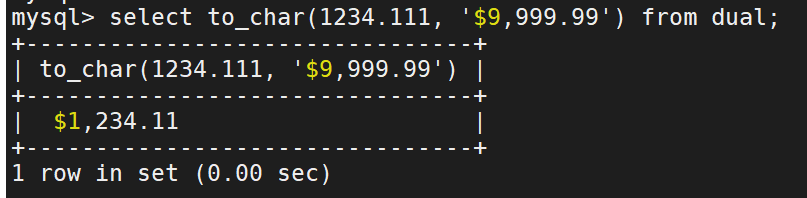
举例如下：

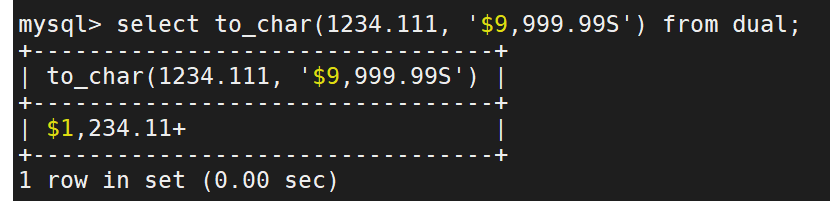












关于fmt格式的说明如下：

|  |  |
| --- | --- |
| **Fmt格式** | **说明** |
| "," | 分组符号 |
| "." | 精度符号；只能出现一次，.和D不可共存 |
| "0" | 有数字就显示数字，没有则补0到第一个非0字符 |
| "9" | 有数字就显示数字，没有指定的数则用空格代替 |
| "D" | 精度符号，默认为”.”；只能出现一次，.和D不可共存 |
| "MI" | 负数则在结束位置增加”-”，取代原正负号；出现在最右侧，与S和PR不可共存 |
| "PR" | 用尖括号包围负数，取代原正负号；出现在最右侧，与S和MI不共存 |
| "S" | 正负号，负数为”-”号，正数为”+”号；取代原正负号，出现在最左侧或者最右侧；S与MI和PR也不可共存 |
| “EEEE” | 返回科学计数法显示结果 |
| “$” | 返回的数值前多一个货币前缀，仅需支持$符号；可以出现在数字整数部分左邻位置、数字友邻位置、或者数字中间，只能出现一次 |

注意事项：

（1）数字开头的符号不占用数字位数

（2）小数截断格式输出时，需要考虑进位

（3）大小写不敏感

更详细的信息可以参考ORACLE官方手册中相关内容。

# 课题描述

目前MySQL是不支持to\_char()函数的用法的，本课题就是要在MySQL8.0.25+版本上上实现该函数的功能，无需编译MySQL源码、使用UDF的方式实现。

UDF是MySQL的一个拓展接口，UDF（User defined function）可翻译为用户自定义函数，这个是用来拓展MySQL的技术手段。

MySQL版本中默认是包含了一些UDF函数的示例可以使用，具体可以参考MySQL的官方文档：

1. <https://dev.mysql.com/doc/refman/8.0/en/function-loading.html>
2. https://dev.mysql.com/doc/extending-mysql/8.0/en/adding-loadable-function.html

全量的功能比较复杂，本次需要实现to\_char()函数支持1个入参和2个入参的部分场景，未描述的场景不做要求、统一将函数结果输出为NULL、不可以导致MySQL服务异常。

**具体要求**：

1. 基本功能，支持to\_char()一个参数的场景，入参类型为字符串或者数字类型；
2. 基本功能，支持to\_char()函数2个参数的场景；
3. 必选功能：支持以下6种fmt格式：',' '.' '0' '9' '$' 'S' 'MI' 'PR'
4. 附加功能：支持fmt格式：'EEEE'、以及其他Oracle中to\_char函数支持的格式

**输出结果**：

1. 方案设计文档；
2. 测试用例设计文档；
3. 功能demo（最后一节课随堂验收验证）；

**打分标准（满分60分，占本课程成绩60%）**：

1. 完成2个基本功能+2个设计文档，得分【30】分。
2. 满足1的基础上+必选功能，得分【30-45】分。
3. 满足2的基础上，完成分配的可选功能，得分【45-60】分。
4. 根据交付功能的编码好坏程度，额外【加分0-10】分，60分封顶；

# 部分验收示例

|  |  |
| --- | --- |
| **测试语句** | **预期结果** |
| # 单个入参 |  |
| select to\_char(123) from dual; | 123 |
| select to\_char(-0123) from dual; | -123 |
| select to\_char(-0) from dual; | 0 |
| select to\_char(-000) from dual; | 0 |
| select to\_char(0.5) from dual; | 0.5/.5 |
| select to\_char(-00.50) from dual; | -0.5/-.5 |
| select to\_char(-.50) from dual; | -0.5/-.5 |
| select to\_char('-0123') from dual; | -0123 |
| select to\_char('GoldenDB is awesome!') from dual; | GoldenDB is awesome! |
| select to\_char('') from dual; |  |
| # 源数据为正整数 |  |
| select to\_char(123, '999') from dual; | 123 |
| select to\_char(123, '99') from dual; | ### |
| select to\_char(123, '9999') from dual; | 123 |
| select to\_char(123, '999.99') from dual; | 123.00 |
| select to\_char(123, '9999.99') from dual; | 123.00 |
| select to\_char(123, '0999') from dual; | 0123 |
| select to\_char(123, '90999') from dual; | 0123 |
| select to\_char(123, '000') from dual; | 123 |
| select to\_char(12345678901234567890123456789012345678901234567890,'99999999999999999999999999999999999999999999999999') from dual; | 12345678901234567890 |
| #源数据为负整数 |  |
| select to\_char(-123, '999') from dual; | -123 |
| select to\_char(-123, '9999.99') from dual; | -123.00 |
| select to\_char(-123, '099') from dual; | -123 |
| select to\_char(-123, '90999') from dual; | -0123 |
| select to\_char(123.22, '9999') from dual; | 123 |
| select to\_char(123.66, '9999.99') from dual; | 123.66 |
| select to\_char(123.225, '909.99') from dual; | 123.23 |
| select to\_char(-123.222, '909.99') from dual; | -123.22 |
| #源数据带逗号 |  |
| select to\_char(1234, '9,999') from dual; | 1,234 |
| select to\_char(1234, '9,9,9,9') from dual; | 1,2,3,4 |
| select to\_char(1234.222, '9,999.99') from dual; | 1,234.22 |
| select to\_char(1234.666, '9,9,9,9.99') from dual; | 1,2,3,4.67 |
| #格式串中带$ |  |
| select to\_char(123, '$999') from dual; | $123 |
| select to\_char(1234.66, '$9999.99') from dual; | $1234.66 |
| select to\_char(1234.222, '$9,999.99') from dual; | $1,234.22 |
| select to\_char(-1234.66, '$0999.99') from dual; | -$1234.66 |
| #格式串中带S |  |
| select to\_char(123, 'S999') from dual; | +123 |
| select to\_char(1234.66, '9999.99S') from dual; | 1234.66+ |
| select to\_char(1234.222, 'S9,999.99') from dual; | +1,234.22 |
| select to\_char(-1234.66, '0999.99S') from dual; | 1234.66- |
| #格式串中带MI |  |
| select to\_char(123, '999MI') from dual; | 123 |
| select to\_char(1234.222, '9,999.99MI') from dual; | 1,234.22 |
| select to\_char(-1234.66, '0999.99MI') from dual; | 1234.66- |
| #格式串中带PR |  |
| select to\_char(123, '999PR') from dual; | 123 |
| select to\_char(-123, '999PR') from dual; | <123> |
| select to\_char(-1234.66, '9999.99PR') from dual; | <1234.66> |
| select to\_char(-1234.66, '90999.99PR') from dual; | <1234.66> |
| #（3）异常用例 |  |
| select to\_char(123, '') from dual; | # |
| select to\_char(123.225, '99.999') from dual; | ####### |
| select to\_char(123, '999XX') from dual; | 报错 |
| select to\_char(123, '99') from dual; | ### |
| select to\_char(123.45, '99.9D9') from dual; | 报错 |
| select to\_char(123.45, '99.9D9') from dual; | 报错 |
| select to\_char(123.45, '$$999.99') from dual; | 报错 |
| select to\_char(123.45, 'S999.99S') from dual; | 报错 |
| select to\_char(-123, 'S999S') from dual; | 报错 |
|  |  |
|  |  |

# 其它说明

1. 功能开发基于开源的mysql8025+版本，需要到官方下载安装部署：https://dev.mysql.com/downloads/mysql/。根据操作系统类型选择不同的安装包，或者使用其他途径安装MySQL也可以。
2. 功能开发需要基于linux环境，可以是实验室的linux服务器，也可以使用个人笔记本搭建linux虚拟机环境，推荐操作系统：redhat7.4。由于需要编码并编译生成.so后缀的动态链接库，可能需要额外部署gcc编译环境（如果安装的系统中没有的话）。
3. MySQL源码中包含udf示例程序udf\_example.cc，可以尝试对此进行编译安装、检查当前系统环境是否能实现udf的编译和安装（涉及到头文件等路径，根据实际情况调整命令）：

gcc -shared -I/home/yxx\_git1/DB/include -I/home/yxx\_git1/DB/build\_release/include/ udf\_example.cc -o udf\_example.so

udf\_example.cc文件：



UDF相关函数接口中使用到的类型描述如下：



编码相关指南可以从官网学习：

https://dev.mysql.com/doc/extending-mysql/8.0/en/adding-loadable-function.html

1. 最终验收，将考虑各种场景，不局限于以上的示例。
2. 由于内置函数和UDF差异，函数结果需要通过额外转换才可以按照字符串显示，UDF函数my\_to\_char的demo输出如下：

