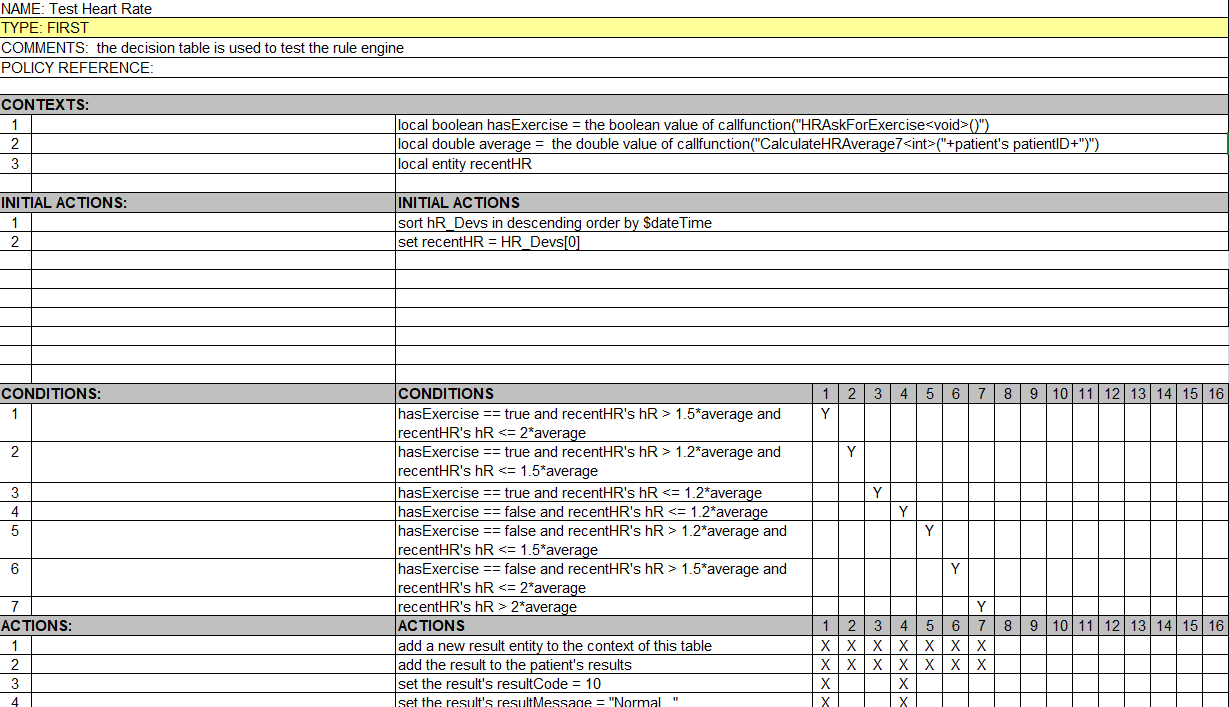
# 结构



上图表示的是一个决策表的简单示例，我们从上至下进行说明。

1. NAME：表示一张表的表名，图中该表的表名即为“Test Heart Rate”。

注：我们在使用表名的时候需要将空格转变为下划线“\_”，如下图所示的perform Evaluate\_zipcode，意为执行Evaluate zipcode这张表（也可以是Evaluate\_zipcode）：



1. TYPE：表示表的执行类型，有FIRST、ALL类型。

注：FIRST表示自左向右扫描到的符合要求的第一项执行。

ALL表示自左向右扫描到的符合要求的所有项都执行。

1. COMMENTS：表示描述一张决策表的注释。
2. POLICY REFERENCE：相当于注释，可以不用管它。
3. CONTEXTS：表示上下文，该部分可以选择不填写任何内容。
4. INITIAL ACTIONS：表示在具体执行rules之前的初始化操作，该部分也可以选择不填写任何内容。
5. CONDITIONS：表示rules的具体判断逻辑。
6. ACTIONS：表示一条rule具体需要执行的操作。

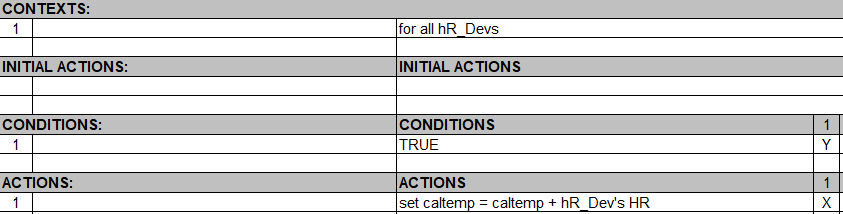
# CONTEXTS

Contexts一栏主要用于申明之后用于操作的上下文变量，下面是只能在contexts一栏中的语句。

## for all

for all表示对一个列表进行循环，该语句只能添加在CONTEXTS一栏中，并且可以嵌套。每一次循环都会执行一次rules的判断执行。为for all <列表变量>。

关于每一次循环的子实体数据访问：你可以使用定义的对应类型的名称来进行数据访问，或者直接使用属性名进行访问。例如下图所示：



或者您也可以直接使用set caltemp = caltemp + HR来代替上面的action。

## for all whose…

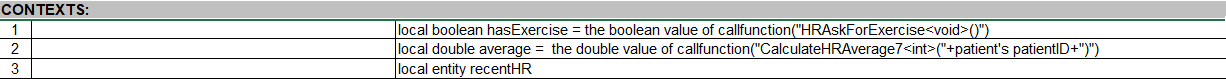
for all whose句式可以帮助筛选一些你所需要的符合条件的子实体。例如：for all HR\_Devs whose HR > 0. 为for all <变量名> whose <布尔句式>。

## for the first….

for the first句式可以帮助筛选出第一个符合条件的子实体。例如：for the first of the HR\_Devs whose HR > 0. 为for the first <变量名> whose <布尔句式>。

## local

local句式可以帮助你添加一个局部变量用以临时存储或者计算，您可以直接对其进行赋值或者仅申明，在INITIAL ACTIONS中进行赋值或者计算，为local <变量类型> <变量名>( = <值>)。例如：local double average = 0或者local entity recentHR.



# INITIAL ACTIONS & ACTIONS

ACTIONS栏中主要申明需要做的操作，不同的是INITIAL ACTIONS是在判断CONDITIONS并执行ACTIONS之前发生的。两栏的操作语句句式都是可以相同的。

## Set

set句式主要用来对变量进行赋值，为set <变量名> = <值>。例如set recentHR = HR\_Devs[0]或者set average = 0.

## Add

add句式主要用来添加变量到某个列表变量中。为add <变量/值> to <列表变量/数值变量>。例如add recentHR to HR\_Devs或者add a new HR\_Dev entity to HR\_Devs。或者我们也可以添加一个变量到该表的上下文变量中add a new HR\_Dev entity to the context of the table。



另外，add句式还可以表示加法，例如add 1 to the counter。但是我们更推荐下面这种做法：set counter = counter + 1以避免add句式的混淆。

## Subtract

subtract句式用于数值型的减法。为subtract <值> from <值/变量>。例如subtract 1 from counter。

## Remove

remove句式用于移除列表变量中的元素。为remove the <整数值> element from the <列表变量> array或者remove <变量> from the <列表变量> array。例如remove the 10 element from the HR\_Devs array或者在循环体中remove HR\_Dev from the HR\_Devs array。

## Randomize

randomize句式用于打乱数组列表，为randomize <列表变量>。例如randomize HR\_Devs。

## Clear

clear句式用于清空数组列表，为clear <列表变量>。例如clear HR\_Devs。

## Sort

sort句式用于对数组列表进行排序，为sort <列表变量> in ascending/descending order by $<属性名>。例如sort HR\_Devs in descending order by $DateTime。

## New

new句式用于新建一个变量实体，为new <类名> entity用以创建一个新的实体变量，或者new <类名> array用以创建一个数组变量。例如set recentHR = new HR\_Dev entity。

## Date相关

**日期转换**：可以直接使用(Date)”MM/DD/YYYY”的格式进行日期的转换，返回日期。例如set DateTime = (Date)”10/3/2007”。

**获取当前日期**：current date可以直接使用，表示当前日期时间。

**日期操作**：add <整数数值> days/months/years to <日期类型>，例如：add 20 days to cdate，其中cdate是定义好的日期类型实体。注：此处的日期类型不能是复杂的嵌套表达式，必须是申明过的直接实体。

subtract <整数数值> days/months/years from <日期类型>，同上。

<日期类型> +/- <整数数值> days/months/years，返回日期。例如：set cdate = cdate + 10 days。

我们可以直接使用>,<,==来判断两个日期是否相同，返回bool值。

## String相关

**字符串的转换**：(String)<数值型>，返回字符串

**字符串拼接**：<字符串>+<数值型>+……返回字符串

**字符串大小写转换**：change <字符串> to lower/upper case，返回字符串。例如：set str = change str to lower case

**字符串截取**：substring of <字符串> from <整数数值> to <整数数值>，返回字符串。例如：set str = substring of str from 0 to 5

**字符串去除前后空白**：trim(<字符串>)，返回字符串。

**获取子串起始位置**：index of <字符串> in <字符串>，返回字符串。例如：set indexer = index of “asdf” in str

**是否符合某种正则表达式**：<字符串> matches <字符串>，返回bool值。

# CONDITIONS

此处可以是任何有效的布尔表达式，通常为>,<,==。

## 自定义函数

当然如果您无法使用上述介绍的语句来完成您所需要的复杂逻辑，我们也提供了自定义函数的调用来帮助您完成这一复杂逻辑的实现。

为：callfunction(“函数名<参数类型，参数类型……>(参数1，参数2……)”)。其中参数类型为简单类型，包括了int型，double型，string型和char型。其中string类型必须使用单引号包裹。之后您可以在edu.dhu.lib.Utils类中实现该函数名的函数。



另外提供了callfunctionwithobj(“函数名<参数类型，参数类型……>”，参数1，参数2……)，其中参数类型可以是复杂的类，但是目前不支持列表类型的传入，不过您可以传入整个patient进来，对此我们提供了IRObjConverter以对patient进行转换。

例如：local double average = the double value of callfunctionwithobj(“CalculateHRAverage<entity>”, patient)。此处的patient是我们所定义的整个上下文的数据实体。

您可以在编写完自己的逻辑之后进行测试。

# 使用的数据类

本项目所使用的数据类是固定的，不过如果出现数据库设计上的更变的话，也就需要作出相应的调整。具体涉及到DevEDD下的Dev\_edd.xls文件，xml下的Dev\_map.xml文件，以及相对应的所有java项目内的数据类，目标位置为edu.dhu.DTRules.entities。相应的.net项目的数据转换也需要作出相应的改变。

# 参照示例

如果您不知如何下手编写自己的决策表，您可以参照我们给出的例子，位置在samples文件夹下。