**行业数据**

**一、行业分类代码查询**

|  |
| --- |
| 【行业分类】：行业分类有3种：证监会行业，申万行业，以及全球行业分类系统GICS。中国常用的是证监会行业分类和申万行业分类。同时，申万行业又分为三级。 |

**调用方法**

|  |
| --- |
| Python context.get\_industry\_category\_code(industry\_name=None) |

**方法说明**

|  |
| --- |
| 根据行业分类的**名称**，查询相应的分类行业**代码。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* industry\_name：分类行业的名称。可选项为：【"申万一级行业", "申万二级行业", "申万三级行业", "证监会行业"】

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_industry\_category\_code("申万一级行业")  context.log(data)  context.count += 1 |

**返回数据**

|  |
| --- |
| Nginx sw\_l1 |

**二、行业列表查询**

|  |
| --- |
| 【行业】：具体指商务服务业、租赁业、航空运输业等。 |

**调用方法**

|  |
| --- |
| Lua context.get\_industries(industry\_category\_code, date) |

**方法说明**

|  |
| --- |
| 根据**行业类别代码**，查询在**不超过指定时间**内最近有数据的一天，该行业类别下所**包含的行业的代码。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* industry\_category\_code：分类行业的代码。可选项为：【"sw\_l1"："申万一级行业", "sw\_l2"："申万二级行业", "sw\_l3"："申万三级行业", "zjw"："证监会行业"】
* date：查询时间，datetime或者日期字符串类型

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_industries(industry\_category\_code="zjw", date="2021-09-30")  context.log(data)  context.count += 1 |

**返回数据示例**

格式：dataframe



**点击图片可查看完整电子表格**

**三、行业成分股查询**

|  |
| --- |
| 【行业成分股】：某个行业所包含的股票（对应特定的上市公司）。 |

**调用方法**

|  |
| --- |
| Lua context.get\_industry\_stocks(industry\_code, date) |

**方法说明**

|  |
| --- |
| 根据**具体行业的代码**，查询在**不超过指定时间**内最近有数据的一天，该行业所**包含的标的的代码。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* industry\_code：具体行业的代码
* date：查询时间，datetime或者日期字符串类型

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_industry\_stocks("I64", "2021-09-30")  context.log(data)  context.count += 1 |

**返回数据**

|  |
| --- |
| Scheme ['000503.XSHE', '000606.XSHE', '000676.XSHE', '000835.XSHE', '002072.XSHE', '002095.XSHE', '002113.XSHE', '002131.XSHE', '002168.XSHE', '002174.XSHE', '002235.XSHE', '002247.XSHE', '002306.XSHE', '002315.XSHE', '002354.XSHE', '002425.XSHE', '002447.XSHE', '002464.XSHE', '002467.XSHE', '002517.XSHE', '002530.XSHE', '002555.XSHE', '002558.XSHE', '002605.XSHE', '002619.XSHE', '002624.XSHE', '002803.XSHE', '002995.XSHE', '003010.XSHE', '300031.XSHE', '300038.XSHE', '300043.XSHE', '300052.XSHE', '300104.XSHE', '300113.XSHE', '300148.XSHE', '300226.XSHE', '300242.XSHE', '300295.XSHE', '300315.XSHE', '300392.XSHE', '300418.XSHE', '300431.XSHE', '300459.XSHE', '300467.XSHE', '300494.XSHE', '300571.XSHE', '300766.XSHE', '300773.XSHE', '300785.XSHE', '300792.XSHE', '300921.XSHE', '300987.XSHE', '301001.XSHE', '600070.XSHG', '600226.XSHG', '600228.XSHG', '600242.XSHG', '600358.XSHG', '600556.XSHG', '600633.XSHG', '600634.XSHG', '600640.XSHG', '600652.XSHG', '600804.XSHG', '600899.XSHG', '600986.XSHG', '601360.XSHG', '603000.XSHG', '603258.XSHG', '603444.XSHG', '603533.XSHG', '603613.XSHG', '603825.XSHG', '603881.XSHG', '603888.XSHG', '688158.XSHG'] |

**四、概念板块列表查询**

|  |
| --- |
| 【概念板块】：与业绩股相对，依靠题材支撑价格，是一类有某种内涵的股票。如奥运概念股，是指与一系列承办奥运或与奥运有商业关联的公司。 |

**调用方法**

|  |
| --- |
| Python context.get\_concepts() |

**方法说明**

|  |
| --- |
| 查询概念板块的列表，只能在 choose\_stock 函数（标的模块）内部调用，不能单独使用，该函数在每根 k 线都会执行一次。 |

**代码示例**

|  |
| --- |
| Fortran def indicators(context):  context.count = 0   def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]  if context.count == 0:  data = context.get\_concepts()  context.log(data)  context.count += 1 |

**返回数据示例**

格式：dataframe



**点击图片可查看完整电子表格**

**五、概念成分股查询**

|  |
| --- |
| 【概念成分股】：某个概念所包含的股票（对应特定的上市公司）。 |

**调用方法**

|  |
| --- |
| Lua context.get\_concept\_stocks(concept\_code, date) |

**方法说明**

|  |
| --- |
| 根据**具体概念的代码**，查询在**不超过指定时间**内最近有数据的一天，该概念所**包含的标的的代码。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* concept\_code：具体概念的代码
* date：查询时间，datetime或者日期字符串类型

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_concept\_stocks("SC0001", "2021-09-30")  context.log(data)  context.count += 1 |

**返回数据**

|  |
| --- |
| Scheme ['000009.XSHE', '000068.XSHE', '000413.XSHE',...] |

**六、股票所在行业查询**

**调用方法**

|  |
| --- |
| Lua context.get\_symbol\_industries(symbol\_exchange, date) |

**方法说明**

|  |
| --- |
| 根据**选定股票的代码**，查询在**不超过指定时间**内最近有数据的一天，成分股中**包含该股票的所有行业的代码。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* symbol\_exchange：股票代码
* date：查询时间，datetime或者日期字符串类型

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_symbol\_industries("600519.XSHG", "2021-09-30")  context.log(data)  context.count += 1 |

**返回数据**

|  |
| --- |
| Scheme ['801120', '801123', '851231', 'C15', 'HY005', 'HY05003'] |

**七、股票所在概念查询**

**调用方法**

|  |
| --- |
| Lua context.get\_symbol\_concepts(symbol\_exchange, date) |

**方法说明**

|  |
| --- |
| 根据**选定股票的代码**，查询在**不超过指定时间**内最近有数据的一天，成分股中**包含该股票的所有概念的代码。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* symbol\_exchange：股票代码
* date：查询时间，datetime或者日期字符串类型

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_symbol\_concepts("000001.XSHE", "2021-09-30")  context.log(data)  context.count += 1 |

**返回数据**

|  |
| --- |
| Scheme ['GN1101', 'GN1126', 'GN1133',...] |

**八、资金流向查询**

|  |
| --- |
| 【资金流向】：股票在某个时间段内的资金流向，具体包含涨跌幅、主力净额、大单净额、中单净额等。 |

**调用方法**

|  |
| --- |
| Python context.get\_net\_amount(symbol\_exchange, start\_date, end\_date) |

**方法说明**

|  |
| --- |
| 获取**指定股票**在**指定时间段内**的资金流向**。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* symbol\_exchange：股票代码
* start\_date：查询起始时间，datetime或者日期字符串类型
* end\_date：查询截止时间，datetime或者日期字符串类型

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_net\_amount("000005.XSHE", "2009-10-31", "2021-09-30")  context.log(data)  context.count += 1 |

**返回数据**

**格式**

* Dataframe

**列名及含义**



**点击图片可查看完整电子表格**

**九、大股东增减持查询**

**调用方法**

|  |
| --- |
| Python context.get\_share\_holders\_change(symbol\_exchange, start\_date, end\_date) |

**方法说明**

|  |
| --- |
| 获取**指定股票**在**指定时间段内**的大股东的增减持数据**。**只能在choose\_stock函数内部调用，不能单独使用，该函数在每根k线都会执行一次。 |

**参数说明**

* symbol\_exchange：股票代码
* start\_date：查询起始时间，datetime或者日期字符串类型
* end\_date：查询截止时间，datetime或者日期字符串类型

**代码示例**

|  |
| --- |
| Python def indicators(context):  context.count = 0  def choose\_stock(context):  context.symbol\_list = ["600000.XSHG"]    if context.count == 0:  data = context.get\_share\_holders\_change("603991.XSHG", "2009-10-31", "2021-09-30")  context.log(data)  context.count += 1 |

**返回数据**

**格式**

* Dataframe

**列名及含义**



**点击图片可查看完整电子表格**