Article
Yuri Marx · Jul 20 6m read

Open Exchange

# Create Stored Procedures using Embedded Python

Python has become the most used programming language in the world (source: https://www.tiobe.com/tiobe-index/ ) and SQL continues to lead the way as a database language. Wouldn't it be great for Python and SQL to work together to deliver new functionality that SQL alone cannot? After all, Python has more than 380,000 published libraries (source: https://pypi.org/) with very interesting capabilities to extend your SQL queries within Python. This article details how to create new SQL Stored Procedures in InterSystems IRIS Database using Embedded Python.

## Python libraries used as samples

This article will use two very useful libraries for anyone working with SQL in IRIS: Geopy and Chronyk.

Geopy is a library used to apply geocoding (qualification of addresses and geographic coordinates) to address data. With it, it is possible, from the name of the street, to obtain the postal code and the complete address, in the format of the post office. Very useful, since many records have an address.

Chronyk is used to process dates and times using human language. This is very useful, as internally, for both IRIS and Python, a date is a number that represents the amount of time that has passed since an initial date. For humans, a date is July 20th, or yesterday, or tomorrow, or two hours ago. Chronyk accepts to receive the date like this and then converts it to universal date format.

## Python support into InterSystems IRIS

Since version 2021.1 it is possible to use Python to create class methods, stored procedures, interop productions and native calls between Python and IRIS (ObjectScript) in a bidirectional way. I don't know of any other Data Platform that works so deeply with Python. The requirement for this to work is that Python is installed on the same physical or virtual machine or container as IRIS. Veja mais detalhes em: https://docs.intersystems.com/iris20221/csp/docbook/DocBook.UI.Page.cls?.... For install python run:

```
# install libraries required for python and pip
RUN apt-get -y update \
  && apt-get -y install apt-utils \
  && apt-get install -y build-essential unzip pkg-config wget \
  && apt-get install -y python3-pip
```

## Python libraries support into InterSystems IRIS

For InterSystems IRIS to be able to use a Python library it is mandatory that it be installed inside <installdir>/mgr/python. Where installdir is the folder where IRIS is installed. To install new packages run:

```
# use pip3 (the python zpm) to install geopy and chronyk packages
RUN pip3 install --upgrade pip setuptools wheel
RUN pip3 install --target /usr/irissys/mgr/python geopy chronyk
```

Pip3 is Python's most popular package manager and installer, Pip.

# Creating Stored Procedures in Python language

One of the possibilities for using Python in InterSystems IRIS is to create Stored Procedures using Python. There are two possibilities:

1. Creation of Stored Procedure Python using SQL Sentence of Create Function or Procedure;
2. Creation of ClassMethod inside ObjectScript Class with sqlProc and language=Python tags.

## Creation of Stored Procedure Python using SQL Sentence of Create Procedure

According to the InterSystems documentation you can also write a SQL function or stored procedure using Embedded Python by specifying the LANGUAGE PYTHON argument in the CREATE statement, as is shown below (source: https://docs.intersystems.com/iris20221/csp/docbook/DocBook.UI.Page.cls?...):

```
CREATE FUNCTION tzconvert(dt TIMESTAMP, tzfrom VARCHAR, tzto VARCHAR)
    RETURNS TIMESTAMP
    LANGUAGE PYTHON
{
    from datetime import datetime
    from dateutil import parser, tz
    d = parser.parse(dt)
    if (tzfrom is not None):
        tzf = tz.gettz(tzfrom)
        d = d.replace(tzinfo = tzf)
    return d.astimezone(tz.gettz(tzto)).strftime("%Y-%m-%d %H:%M:%S")
}
```

When you run this new SQL Function:

```
SELECT tzconvert(now(), 'US/Eastern', 'UTC')
```

The function returns something like:

```
2022-07-20 15:10:05
```

## Creation of ClassMethod inside ObjectScript Class with sqlProc and language=Python tags

I confess that this approach is my favorite: creating a ClassMethod with the sqlProc and language=Python tags. In my opinion it is easier to maintain, is better documented, evident and with better management of source code versions. For this approach, I've published a sample application: https://openexchange.intersystems.com/package/Python-IRIS-SQL-Procedures.... I will use it to demonstrate this second approach in detail.

# Sample application installation

To install the sample application, follow the steps below:

1. Clone/git pull the repo into any local directory

```
$ git clone https://github.com/yurimarx/iris-sql-python-sample.git
```

2. Open a Docker terminal in this directory and run:

```
$ docker-compose build
```

3. Run the IRIS container:

```
$ docker-compose up -d
```

Another installation possibility is to use ZPM:

```
zpm "install iris-sql-python-sample"
```

## Samples of Stores Procedures using Python

The first example is a stored procedure to handle geocoding of addresses, see the source code:

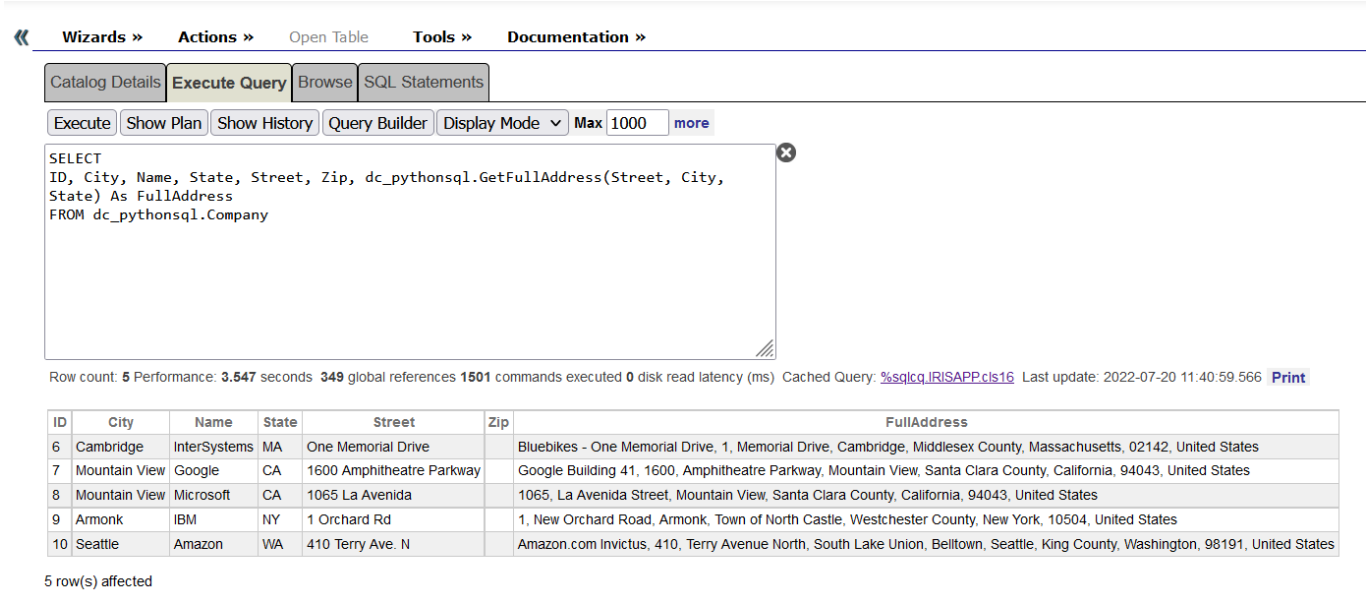
```
  ClassMethod GetFullAddress(Street As %String, City As %String, State As
%String)
As %String [ Language = python, SqlName = GetFullAddress, SqlProc ]
{
  import geopy.geocoders
  from geopy.geocoders import Nominatim
  geopy.geocoders.options.default_timeout = 7
  geolocator = Nominatim(user_agent="intersystemsiris"  )
  location = geolocator.geocode(Street + ", " + City + ",
" + State, country_codes="US")
  return location.address
}
```

See that a ClassMethod was declared (inside the dc.pythonsql.Company class) with the [ Language = python, SqlProc] tags.
The SqlName tag allows setting a name for the new stored procedure in SQL sentences.

Go to Management Portal, System > SQL and run the following code:

```
SELECT
ID, City, Name, State, Street, Zip, dc_pythonsql.GetFullAddress(Street, City, State)
As FullAddress
FROM dc_pythonsql.Company
```

You will see these results:




Now the incomplete addresses returns as "full" addresses (complete and qualified).

*Note: if nothing returns execute #class(dc.pythonsql.Company).CreateFiveCompanies(). It will create five companies to use on tests.*

This package can work with the main open and market geocoding services. In this example we use the open service Nominatim, but it is possible to use Bing, Google, ArcGIS and others. See the possibilities at https://geopy.readthedocs.io/en/stable/#module-geopy.geocoders.

The second example is a date and time package in humanized format, Chronyk.

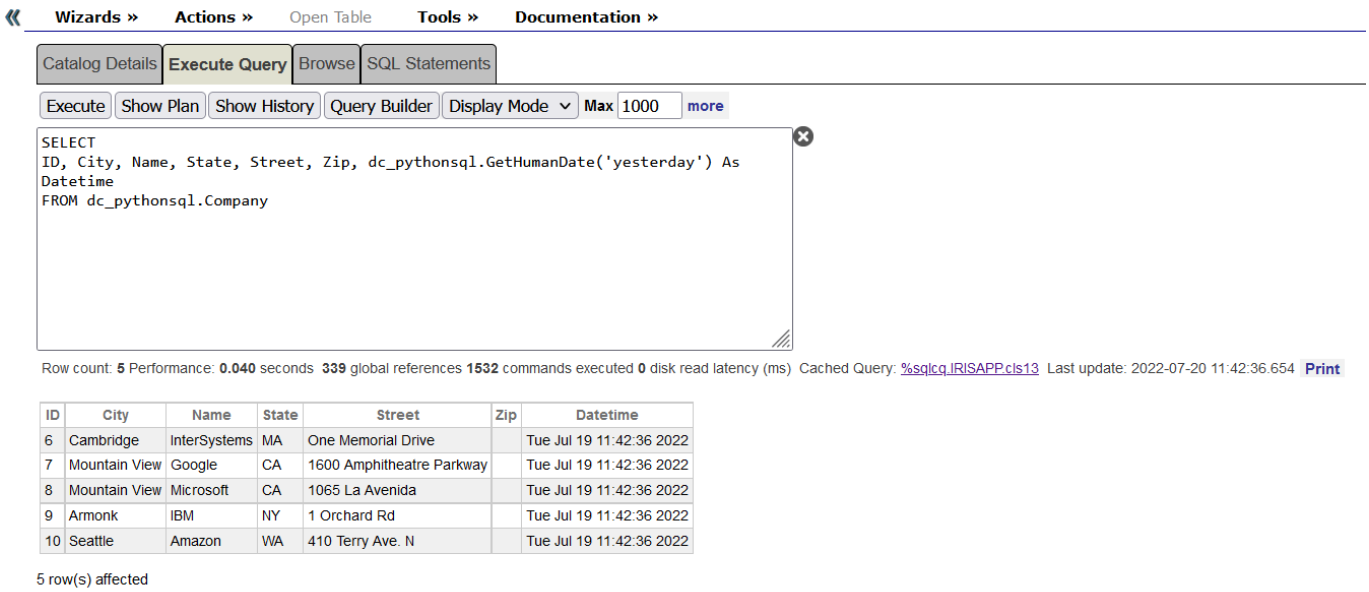
It allows you to send sentences like "tomorrow", "yesterday", "4 hours from now", "July 4, 2022" and get the result in universal date format. See the creation of the stored procedure:

```
ClassMethod GetHumanDate(Sentence As %String) As %String [
Language = python, SqlName = GetHumanDate, SqlProc ]
{
  from chronyk import Chronyk
  t = Chronyk(Sentence)
  return t.ctime()
}
```

Execute the following call in Management Portal > System > SQL:

```
SELECT
ID, City, Name, State, Street, Zip, dc_pythonsql.GetHumanDate('yesterday') As Datetim
e
FROM dc_pythonsql.Company
```

See results like it:




If you want just call the stored procedure you can use this SQL sentence:

```
select dc_pythonsql.GetHumanDate('yesterday') as Datetime
```

This library has several possibilities for humanized dates and times, see https://github.com/KoffeinFlummi/Chronyk.

So, it is easy create Python Stored Procedures, enjoy it!

#Embedded Python #Python #SQL #InterSystems IRIS #InterSystems IRIS for Health
Check the related application on InterSystems Open Exchange

---

**Source URL:**https://community.intersystems.com/post/create-stored-procedures-using-embedded-python