Pools in HIPE

Why would I use *that*

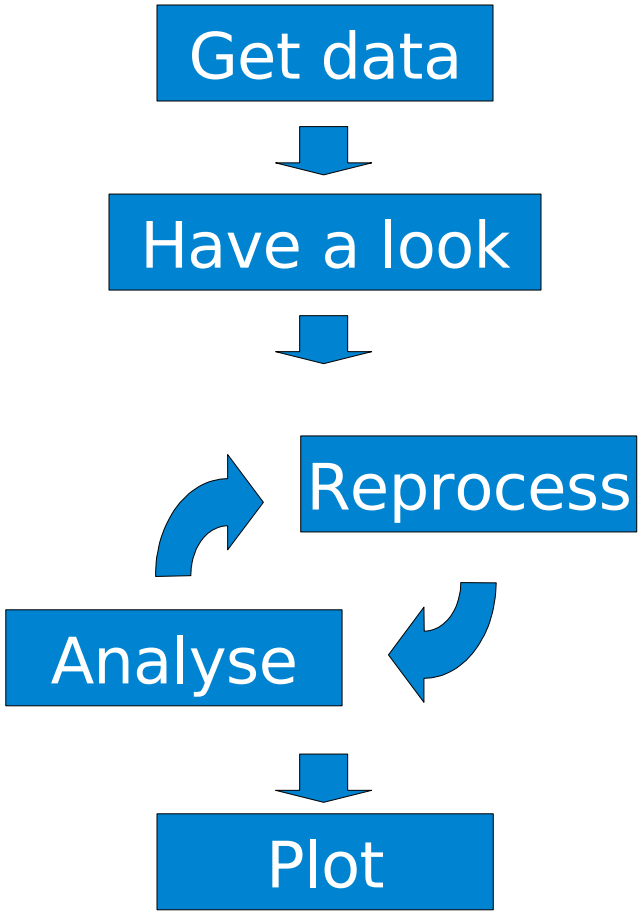
Paul Balm

DP Workshop 2012

February 2012

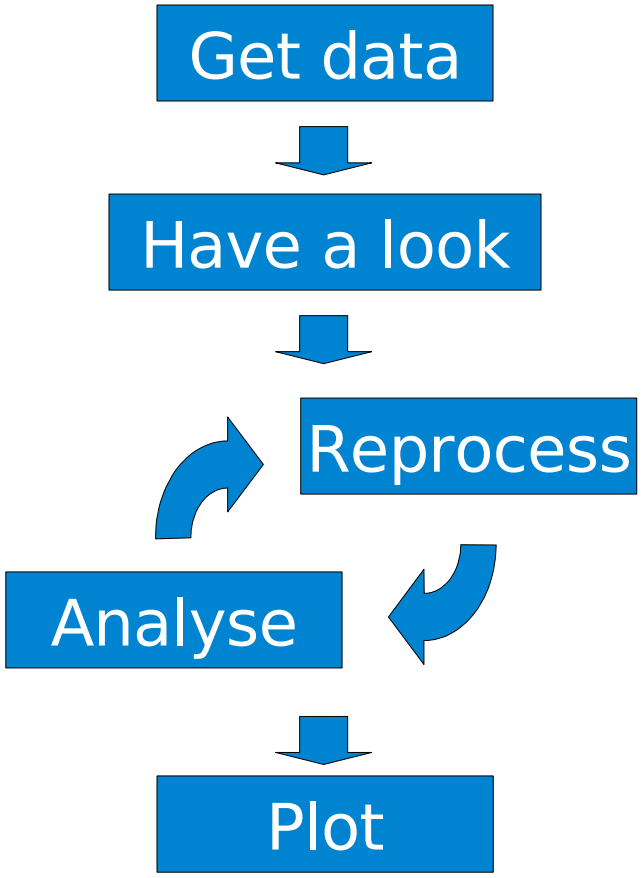
Introduction

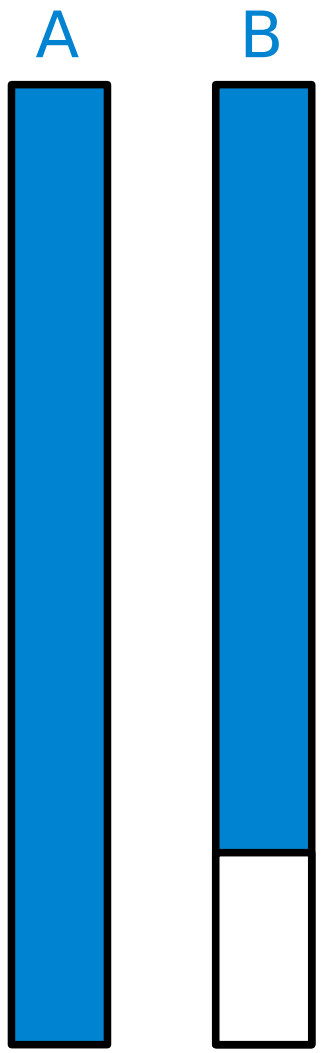
Example workflow



Introduction

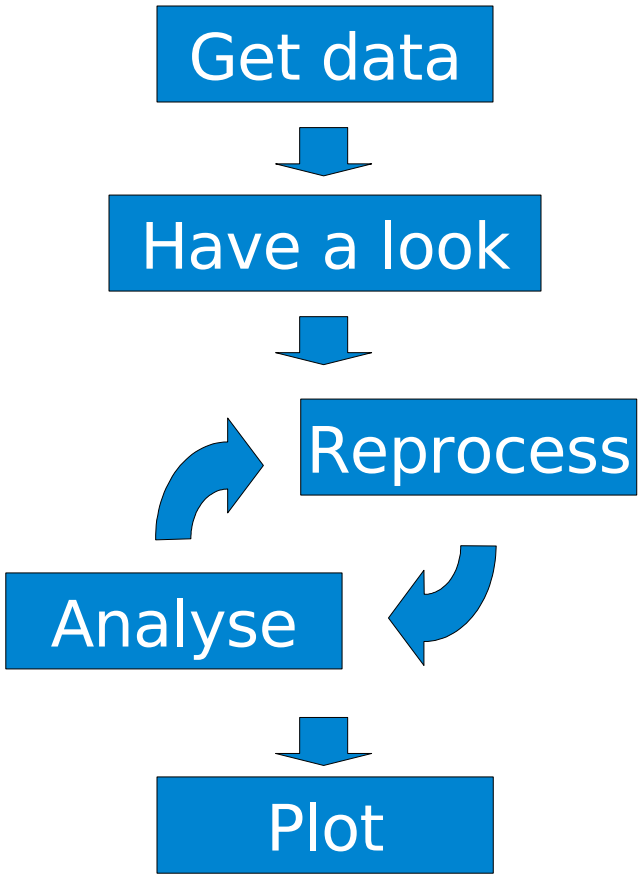
Example workflow

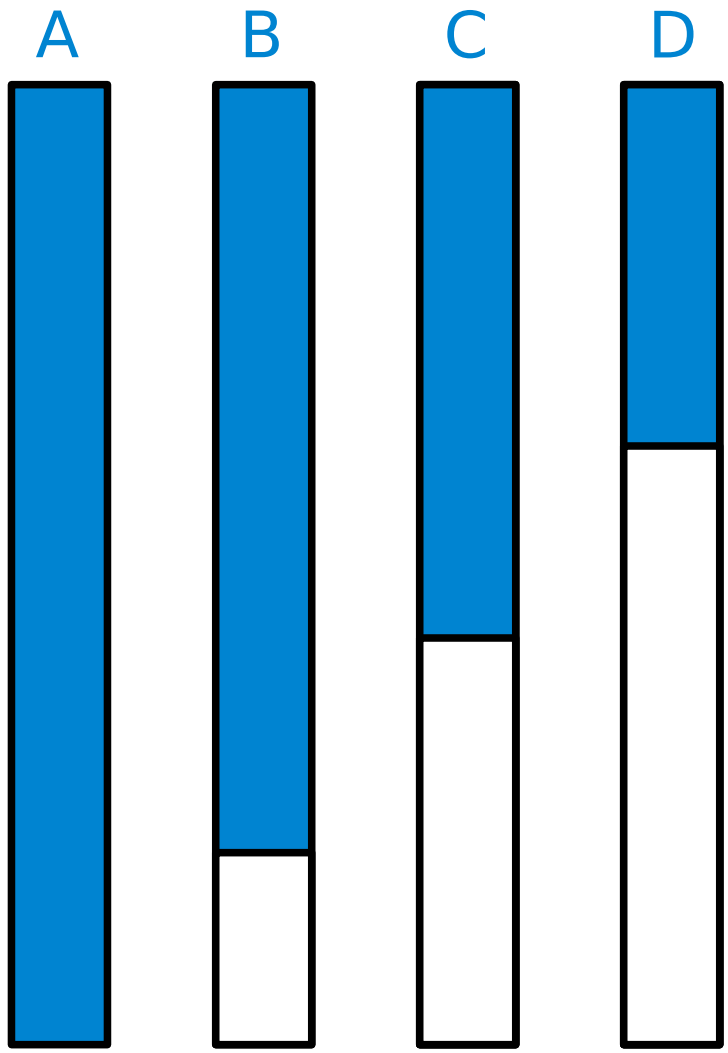




Introduction

Example workflow





Storing Your Data

Scenario D: Only use HIPE “to have a look”
Export your data as FITS files.
(See “Basics on data access: …” by Eva Verdugo)

Scenario A, B, C: You will want to store your data.
HIPE offers:

- Store as (FITS) files & work with them in HIPE
- Store in pools & work with data from pools

Goal of this talk

To allow you to make an informed decision how to store your data:
- as FITS files
- in HIPE pools*

* I mean “local pools” in this talk.

What's a pool?

A pool is a database:

- store data, load it again later
- searching (“querying”)
- etc.

- Pool data is stored in files
- Data should be accessed using HIPE, **not using the file system**

Advantages of files

- **Simplicity**
  Easy to understand what you're doing.

- **Trust**
  Not worried about losing your data.

- **Familiar tools** to work with files:
  - Command-line (cp, rm, ls...)
  - GUIs (file explorers)

Advantages of pools

Pools offer:

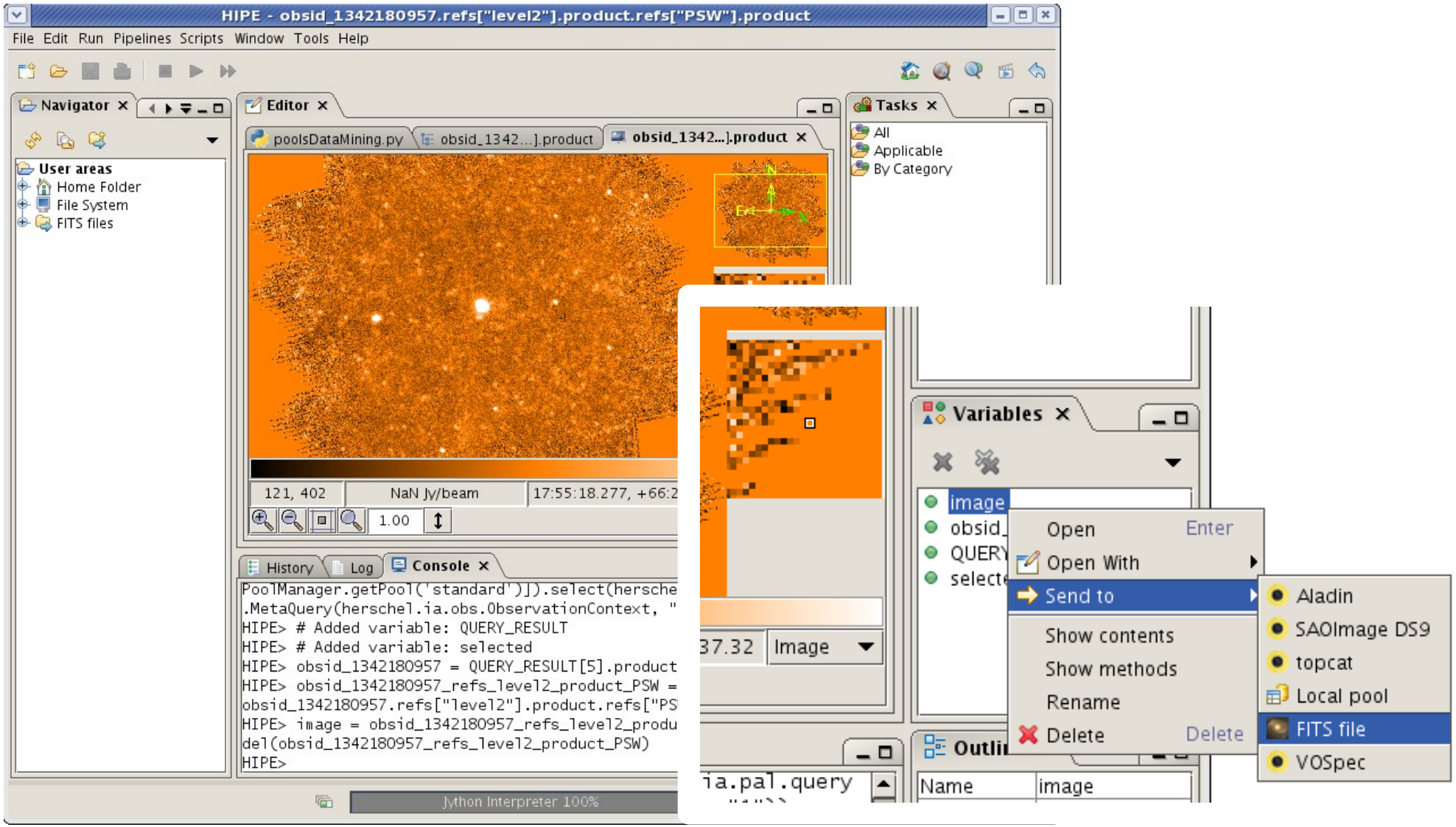
- **Automatic versioning**
  Keep track of different versions of your data.
- **Efficient use of disk space**
  Save only the changes
- **Searching your data/Data mining**
  E.g. find all images covering (ra, dec)...
- **Efficient memory usage**
  Using references, load only what you need
- **Tagging**
  Assign labels to your data

Conclusion

Disadvantages of one are lack of the advantages of the other...

- Files: Lack of features offered by pools
- Pools: Learning curve.

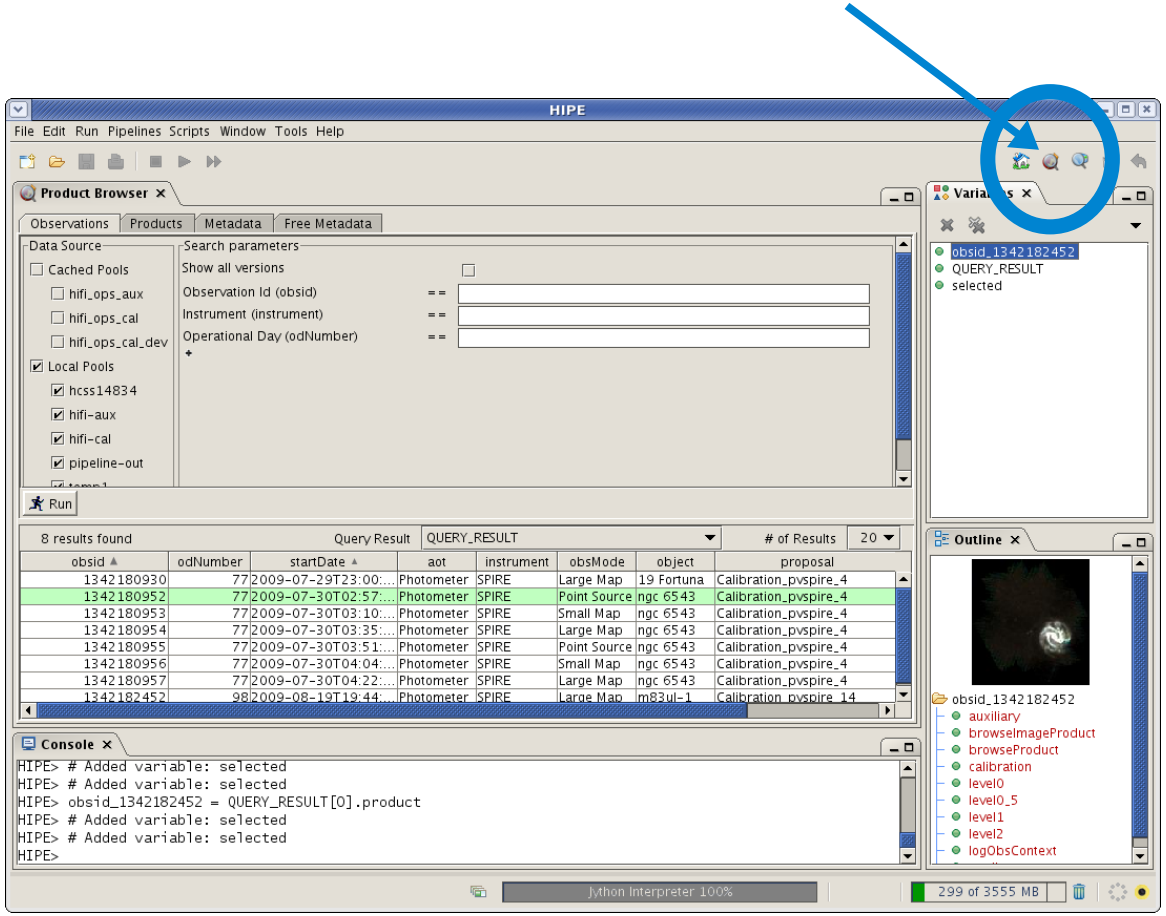
File I/O


HIPE - obsid_1342180957.refs["level2"].product.refs["PSW"].product
File Edit Run Pipelines Scripts Window Tools Help
Navigator
User areas
Home Folder
File System
FITS files
Editor
poolsDataMining.py
obsid_1342...].product
obsid_1342...].product
121, 402
NaN Jy/beam
1.00
Tasks
All
Applicable
By Category
History
Log
Console
PoolManager.getPool('standard')]).select(hersche
.MetaQuery(herschel.ia.obs.ObservationContext, "
HIPE> # Added variable: QUERY_RESULT
HIPE> # Added variable: selected
HIPE> obsid_1342180957 = QUERY_RESULT[5].product
HIPE> obsid_1342180957_refs_level2_product_PSW =
obsid_1342180957.refs["level2"].product.refs["PS
HIPE> image = obsid_1342180957_refs_level2_produ
del(obsid_1342180957_refs_level2_product_PSW)
HIPE>
Jython Interpreter 100%
Variables
image
obsid_
QUERY
select
Open Enter
Open With
Send to
Show contents
Show methods
Rename
Delete Delete
Aladin
SAOImage DS9
topcat
Local pool
FITS file
VOSpec
37.32
Image
ia.pal.query
Name
image

Pool GUI: Product Browser

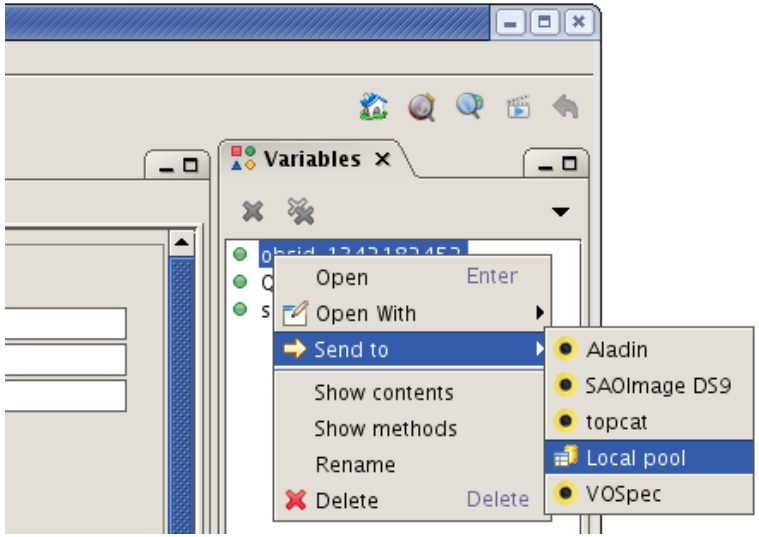
GUI for working with files: File explorer
- Find files
- Remove, open, ...

GUI for working with pools: "Product Browser"

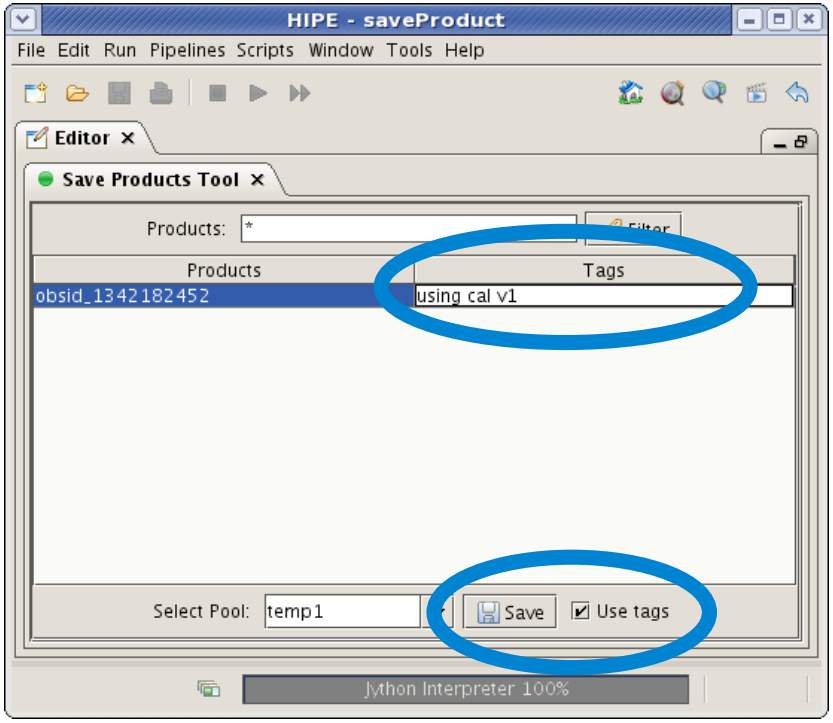


Saving products with tags

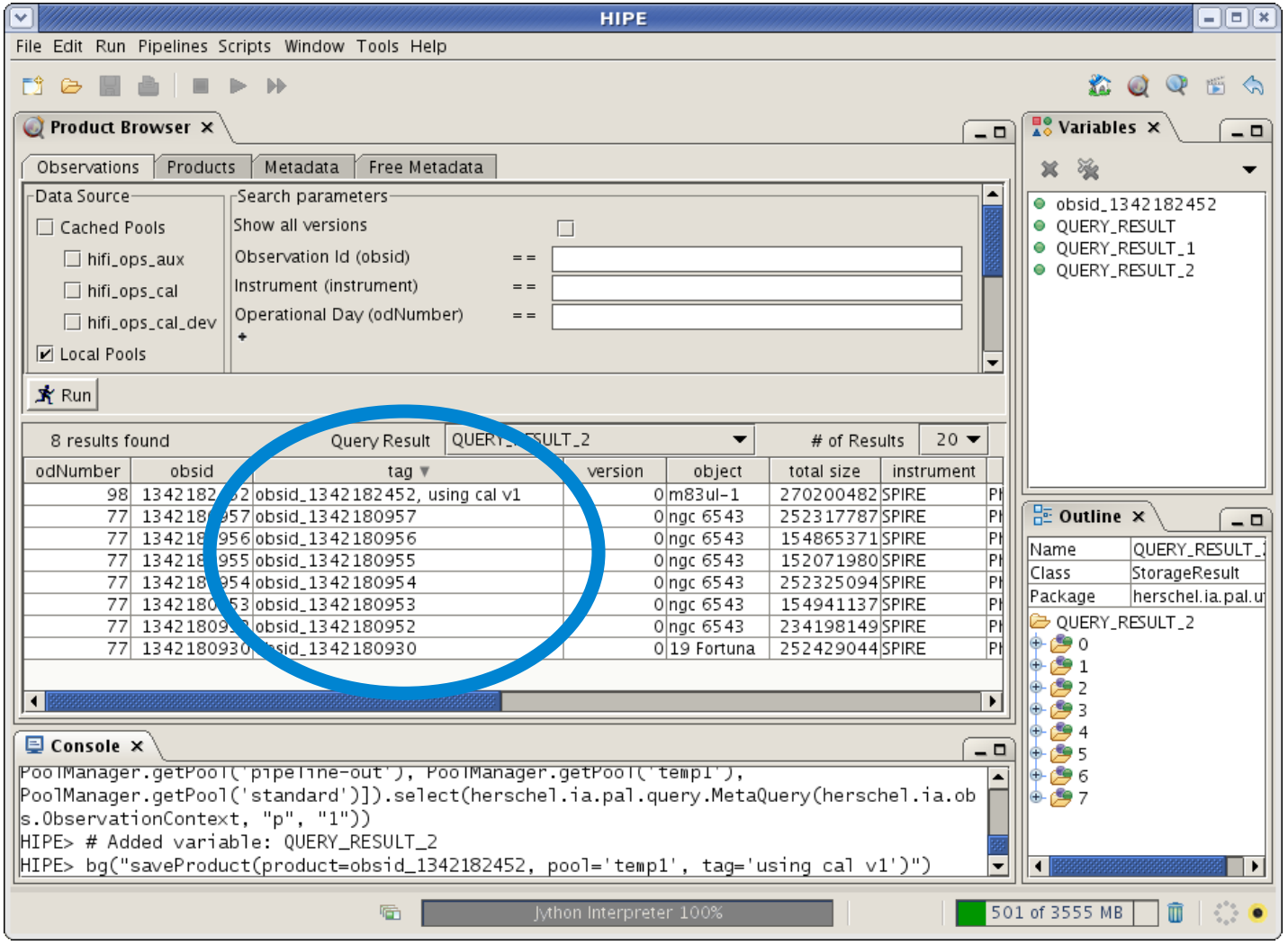
Use right-click menu:



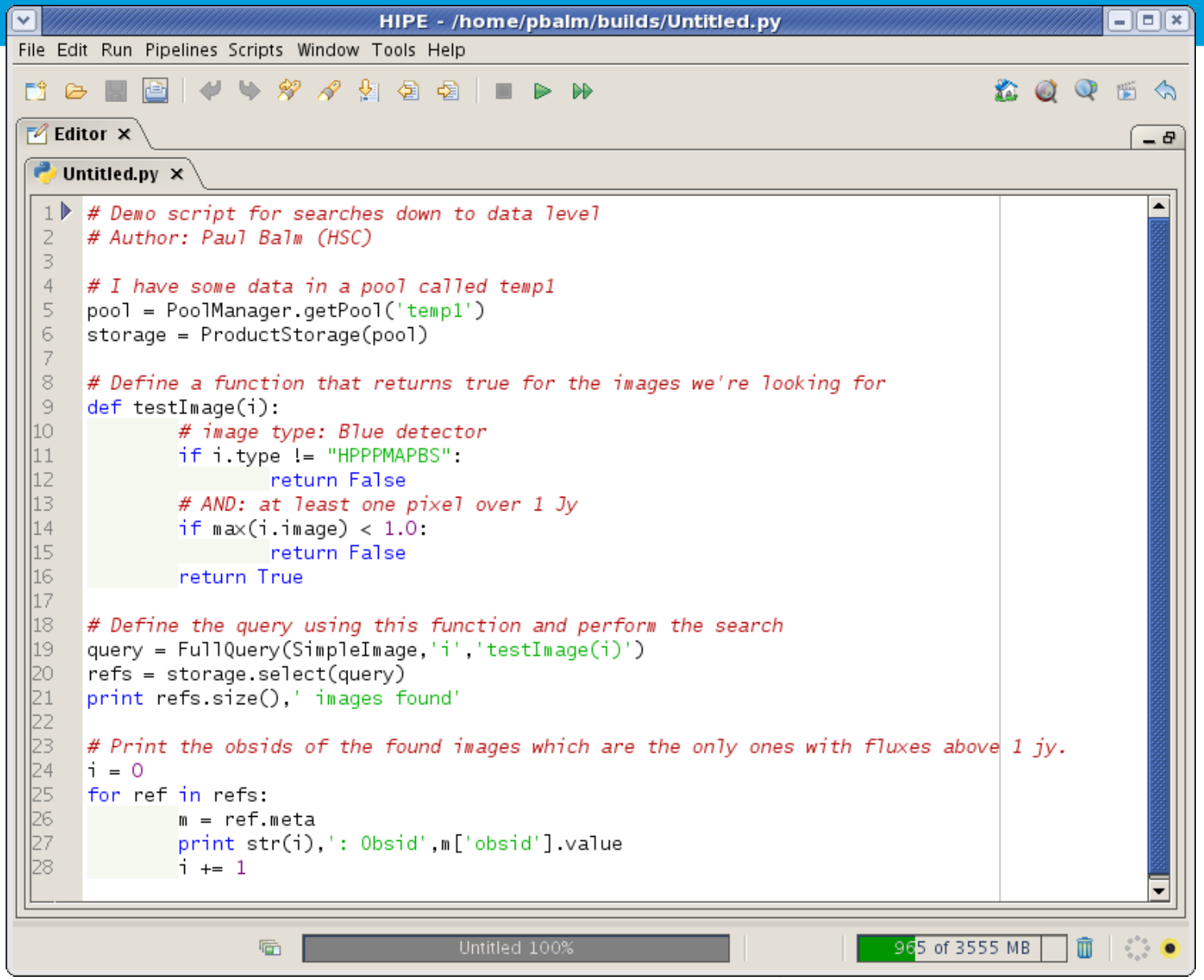
Or: Window-->Show View-->
Data Access-->Save Products to Pool



Using tags


HIPE
File Edit Run Pipelines Scripts Window Tools Help
Product Browser
Observations Products Metadata Free Metadata
Data Source
Cached Pools
hifi_ops_aux
hifi_ops_cal
hifi_ops_cal_dev
Local Pools
Search parameters
Show all versions
Observation Id (obsid) ==
Instrument (instrument) ==
Operational Day (odNumber) ==
Run
8 results found
Query Result
# of Results 20
odNumber obsid tag version object total size instrument
98 obsid_1342182452, using cal v1 0 m83ul-1 270200482 SPIRE
77 obsid_1342180957 0 ngc 6543 252317787 SPIRE
77 obsid_1342180956 0 ngc 6543 154865371 SPIRE
77 obsid_1342180955 0 ngc 6543 152071980 SPIRE
77 obsid_1342180954 0 ngc 6543 252325094 SPIRE
77 obsid_1342180953 0 ngc 6543 154941137 SPIRE
77 obsid_1342180952 0 ngc 6543 234198149 SPIRE
77 obsid_1342180930 0 19 Fortuna 252429044 SPIRE
Variables
obsid_1342182452
QUERY_RESULT
QUERY_RESULT_1
QUERY_RESULT_2
Outline
Name QUERY_RESULT_
Class StorageResult
Package herschel.ia.pal.u
QUERY_RESULT_2
0 1 2 3 4 5 6 7
Console
PoolManager.getPool('pipeline-out'), PoolManager.getPool('temp1'),
PoolManager.getPool('standard')]).select(herschel.ia.pal.query.MetaQuery(herschel.ia.obs.ObservationContext, "p", "1"))
HIPE> # Added variable: QUERY_RESULT_2
HIPE> bg("saveProduct(product=obsid_1342182452, pool='temp1', tag='using cal v1')")
Jython Interpreter 100%
501 of 3555 MB

Data mining using pools



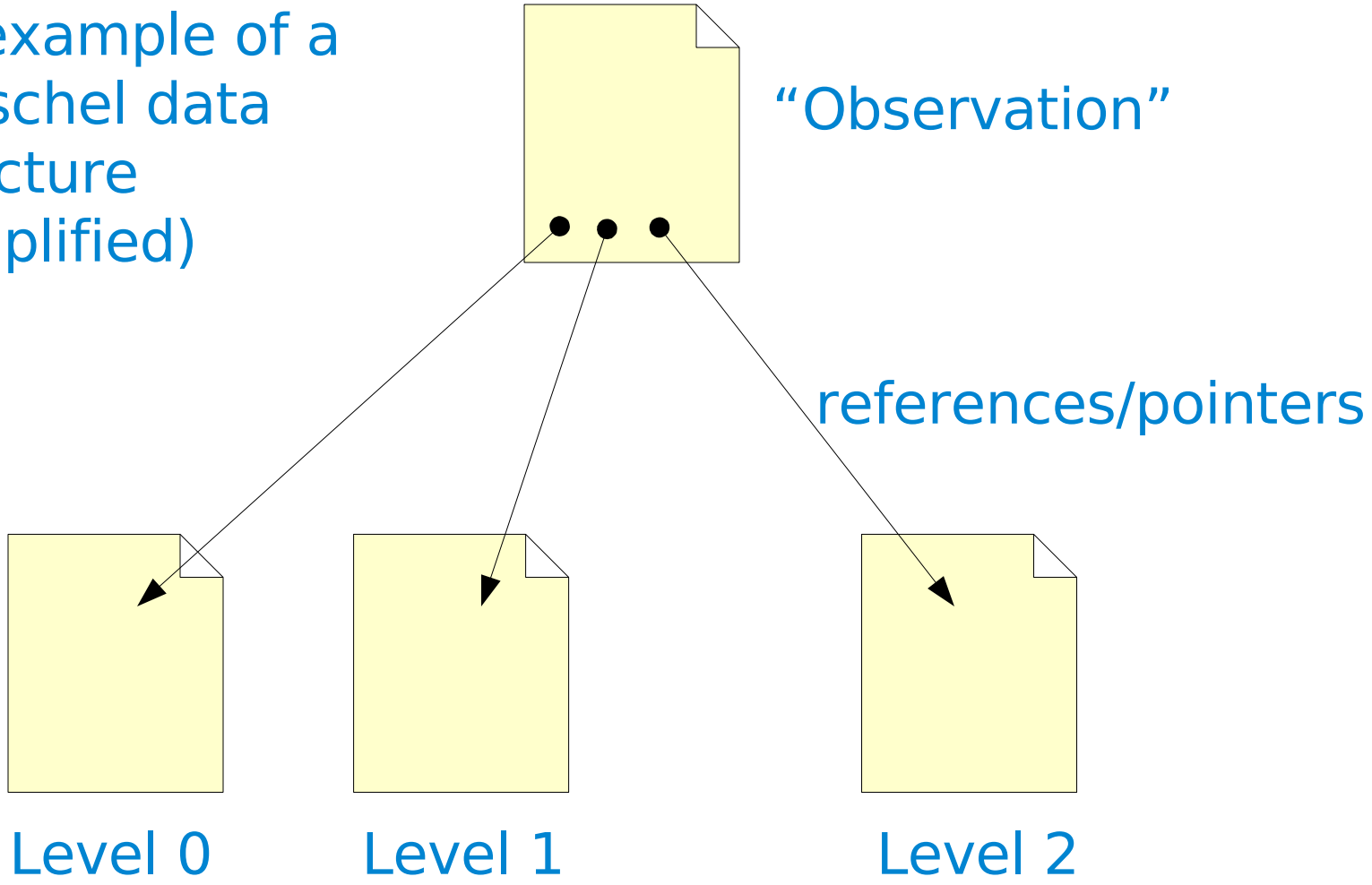
Data mining using pools

Data mining from start to finish:

- Get observations from HSA
  (see "*Basics on data handling...*")
- Save them to a pool
  - Having data in multiple pools is ok as well
- Run the data-mining script

Efficient use of memory & disk space

An example of a Herschel data structure (simplified)



Efficient use of memory & disk space

Reprocessed the Level 2 data and saved observation again
- saved only new data