+

Starting Data Analysis on STAR

Samantha Brovko
Tutorial
14 May 2010 (Updated 4 November 2011)
Heavy Ion Group at UC Davis

Outline

- First steps
- get_file_list.pl
- MuDST
- Maker basics
  - File Organizing
- Examples

+ First Steps

- Log in to RHIC: the command line
  - ssh rssh.rhic.bnl.gov –l jrdoe
- Pulling up a terminal
  - rterm –i
  - -i is to bring up the RCF terminal in your terminal instead of opening a new window
  - http://drupal.star.bnl.gov/STAR/comp/sofi/rcf-contributions/rterm
    - For more information about rterm flags
- From here, set up your space

For more information about logging in to RCF visit:
http://drupal.star.bnl.gov/STAR/comp/sofi/facility-access/ssh-keys
https://www.racf.bnl.gov/docs/authentication/ssh/sshkeys

+

get_file_list.pl

- Finding data files on STAR memory
  - To make a file list

- Not all files are readily accessible
  - NFS disk – accessible for all rcf nodes
  - Archives (HPSS) – must be requested to load files to a node
  - Local disk – only on that specific rcf node, recommend use only with the scheduler or for testing your code

- Many flags to pull specific files
  - Flags define what triggers, libraries, detectors or participants were used in the dataset that you want to collect
  - http://www.star.bnl.gov/public/comp/prod/DataSummary.html
    - Gives all statistics and production IDs for all STAR data

You can find descriptions of all the flags here:
www.star.bnl.gov/public/comp/sofi/FileCatalog/user.html

+

get_file_list.pl

- Basic structure

```
[rcas6017] ~/> get_file_list.pl –limit 10 –keys ‘path,filename’ –cond
‘production=P07id,filetype=daq_reco_MuDst,filename~st_upsilon,storage=
NFS,tpc=1,emc=1,sanity=1’ -distinct
```

May be useful to `cat` the output into a `<name>.list` file

This is what you type at the command line.

-*keys* : path gives us the absolute path to the file of interest
filename includes the name of the file of interest

-*limit* N : limits the number of files to find to N files

-*distinct* : only show files distinct from each other

get_file_list.pl

- Basic structure

```
[rcas6017] ~/> get_file_list.pl –limit 10 –keys ‘path,filename’ –cond
‘production=P07id,filetype=daq_reco_MuDst,filename~st_upsilon,storage=
NFS,tpc=1,emc=1,sanity=1’ -distinct
```

-cond: production gives the type of collision, d+Au, Au+Au, p+p, Cu+Cu

filetype specifies the kind of file and data you want to analyze

filename again picks out files with this phrase in the name

storage describes where you want these files to be; either readily accessible or in the archives

tpc, emc true give files where these detectors are included

sanity means nothing weird happened to the file during processing

+ MuDST

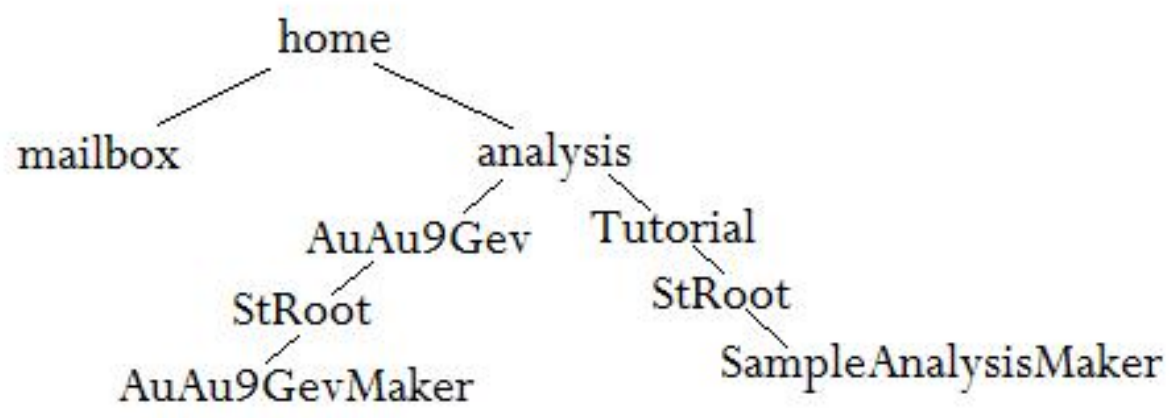
- This is the file where all the collision data is stored
  - File contains information about a number of collisions
- Has Trigger information
  - From ZDC, BBC, VPD, EMC etc.
- Has Track information
  - Fit points in tracker
  - Momentum
  - dE/dx
  - Charge
  - Global or primary type
  - DCA to vertex
  - Etc.

+ Maker Basics

- You can find a list of classes here
  - http://www.star.bnl.gov/webdata/dox/html/annotated.html
- If programming in C++ your extensions should be
  - .cxx for the code file
  - .h for the header file
- STAR coding and naming conventions can be found here
  - http://drupal.star.bnl.gov/STAR/comp/sofi/soft-n-libs/standards
- Before we get started…

File Organizing

- Need to have an StRoot folder
  - This is where cons looks to compile files
  - Within this folder is your analysis maker
- Cons is a command which compiles your maker
  - Full g++ command with all necessary flags etc.
- Should have your environment like



Maker Basics

- Four main functions: InitRun(), Init(), Make(), Finish()
  - Can make more, these are the basic four
- InitRun()
  - Called only once and calls runs one by one
- Init()
  - Called only once
  - Perform tasks that only need to be done once
    - Book histograms, define global variables etc.
- Make()
  - Done for every event: calculate values, fill Ntuples or histograms
- Finish()
  - Done once: write data to .root files, close files etc.

+ The Scheduler

- Use this to submit a job to the RCAS farm
- Must have a .xml (script) file to send to scheduler
  - `star-submit <filename>.xml`
  - Script needs to include: files to use, macro to execute, and where to put the output
- Takes your full job, breaks it down for smaller computing clusters to complete
- Full description of the scheduler can be found:
  - www.star.bnl.gov/public/comp/Grid/scheduler

Checking Jobs and Priority

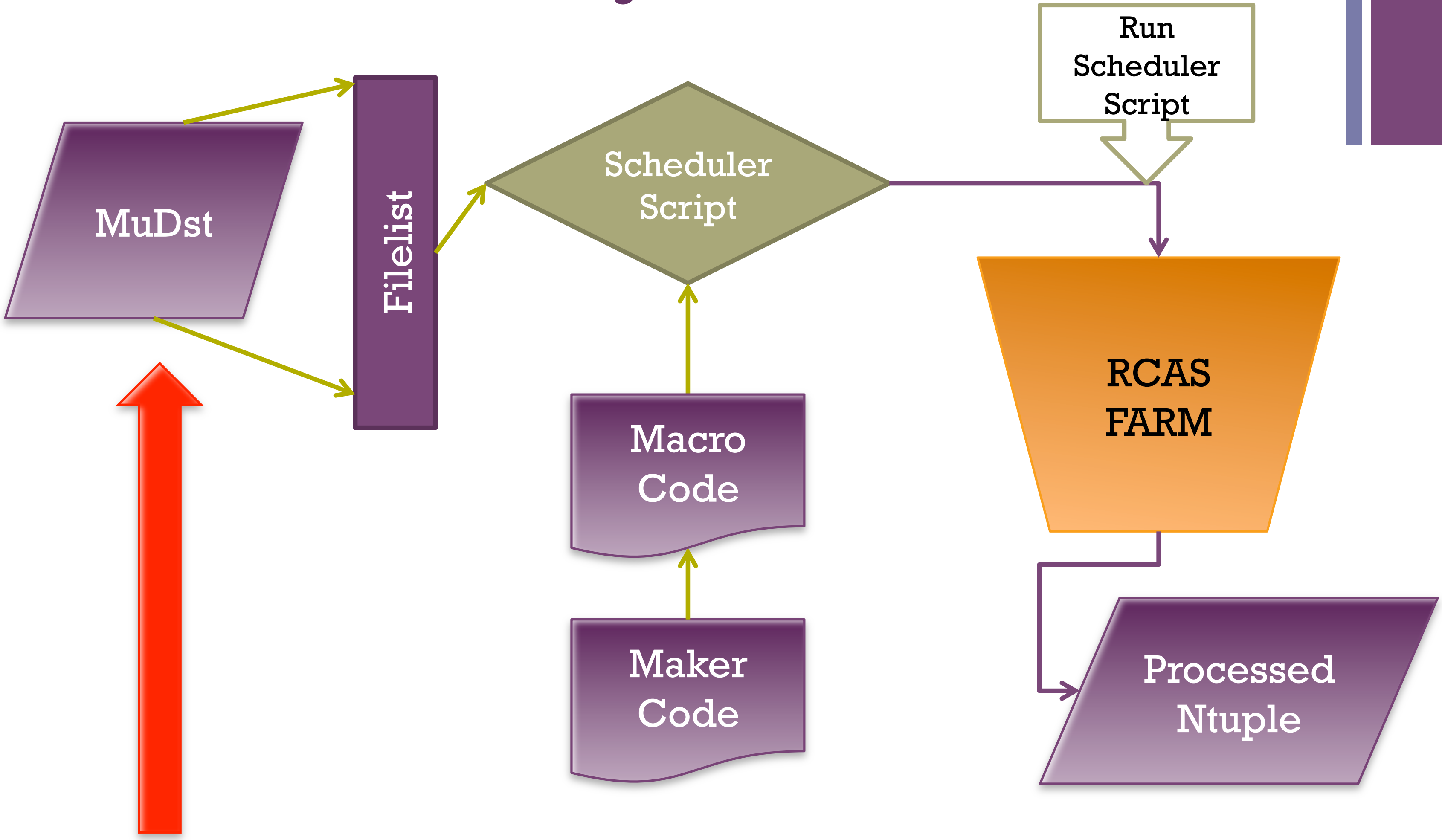
```
~/> condor_q –submitter <username>
```

- Checks the number of jobs running, idle and held
  - Useful for checking when jobs have completed
  - Idle means job is waiting for a node to run on
  - Held means something went wrong with the job

```
~/> condor_userprio –allusers | grep “<username>”
```

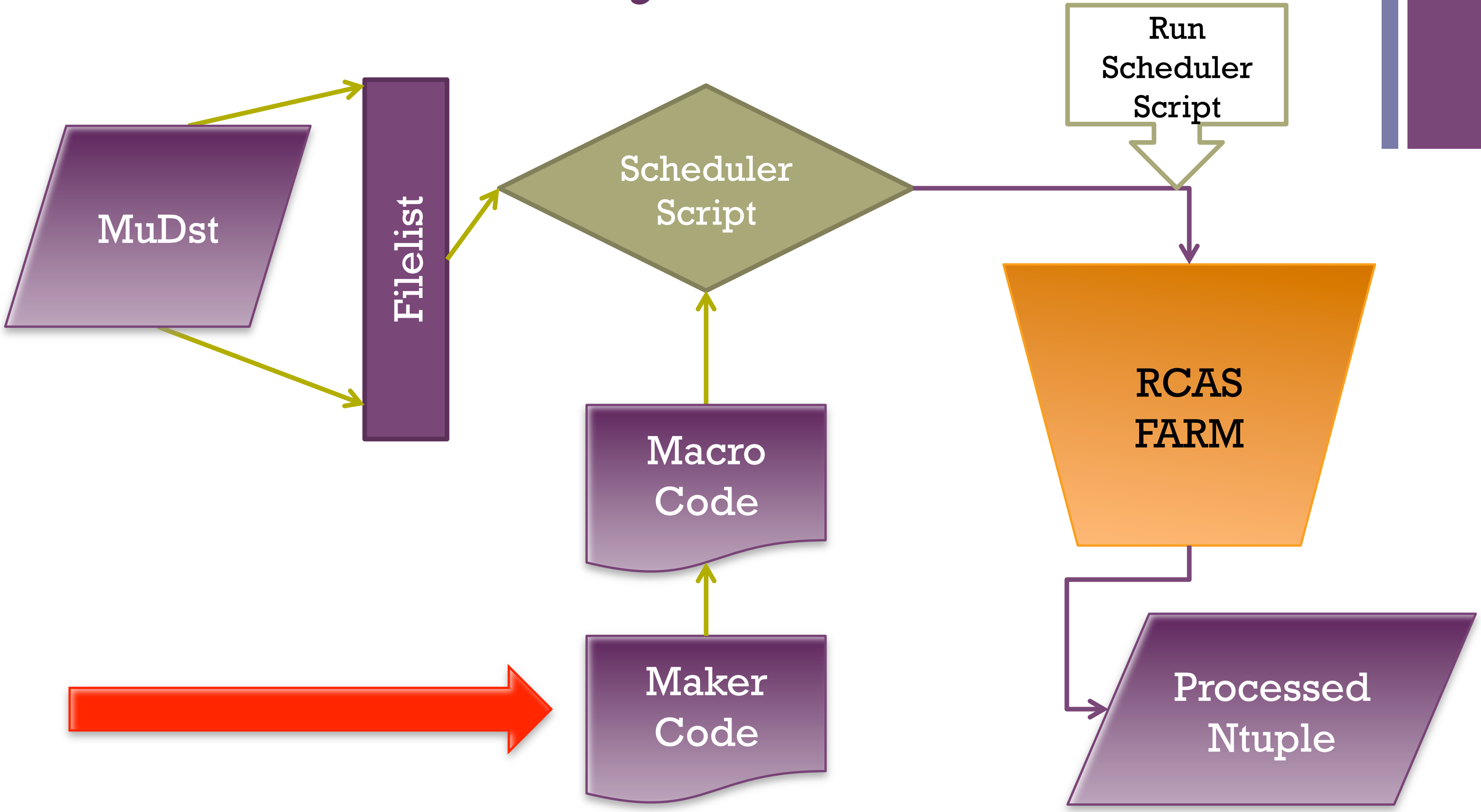
- Checks your user priority
  - Useful for estimating when your jobs will be completed

How Data Analysis Works



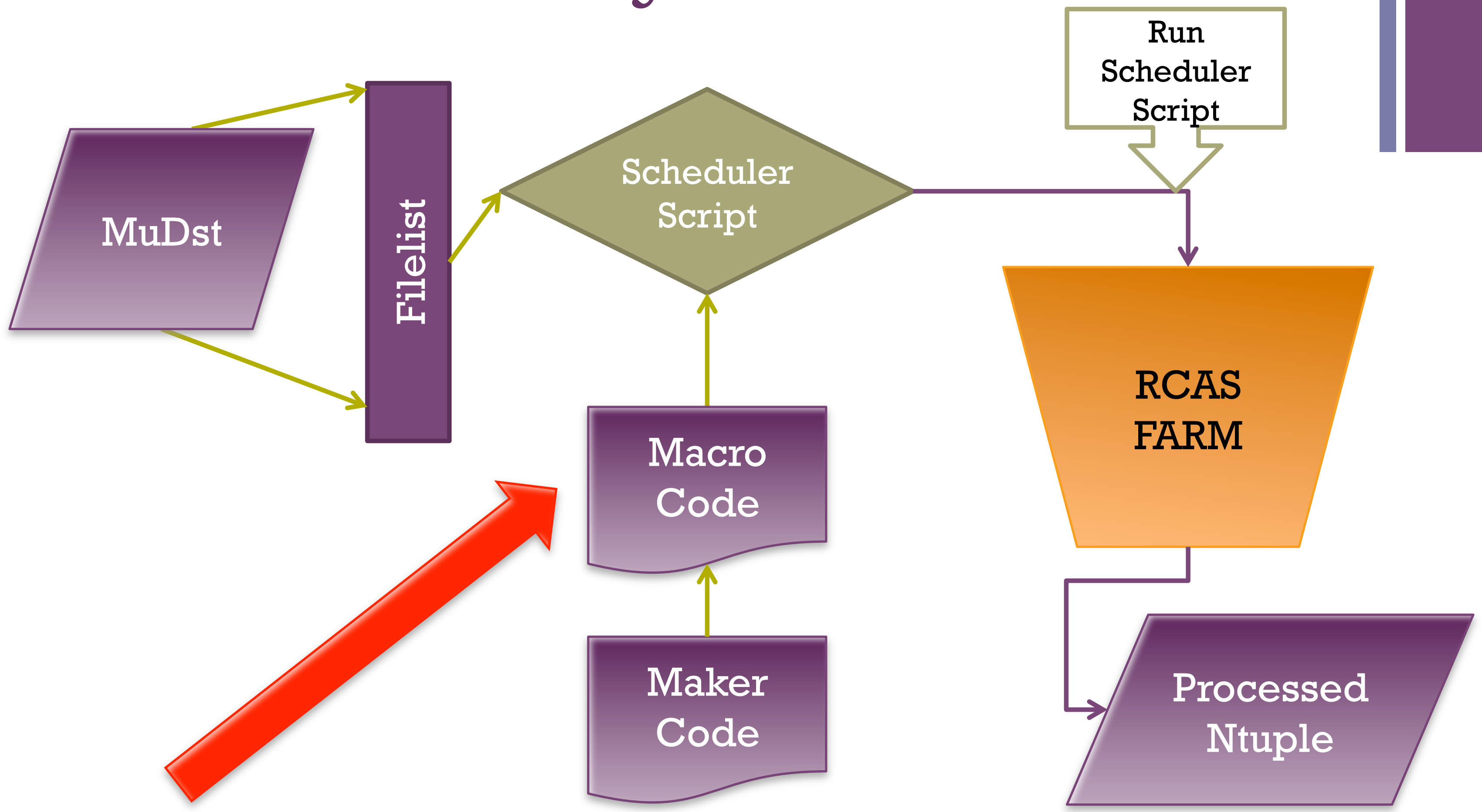
- get_file_list.pl collects locations of MuDst files and places them into a filelist

How Data Analysis Works



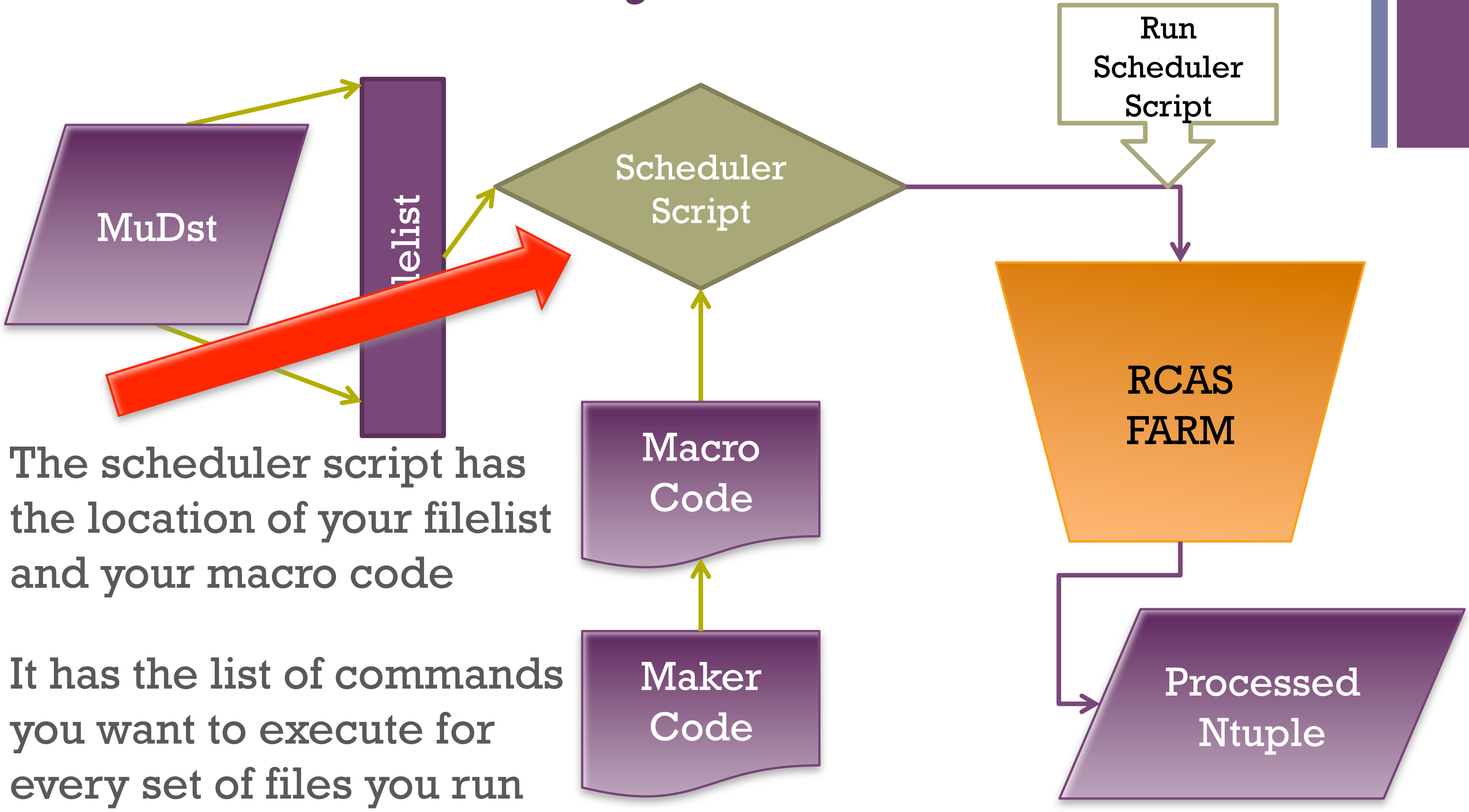
- Your Maker code describes the histograms, variables, and Ntuples/Trees of data you want to have

How Data Analysis Works



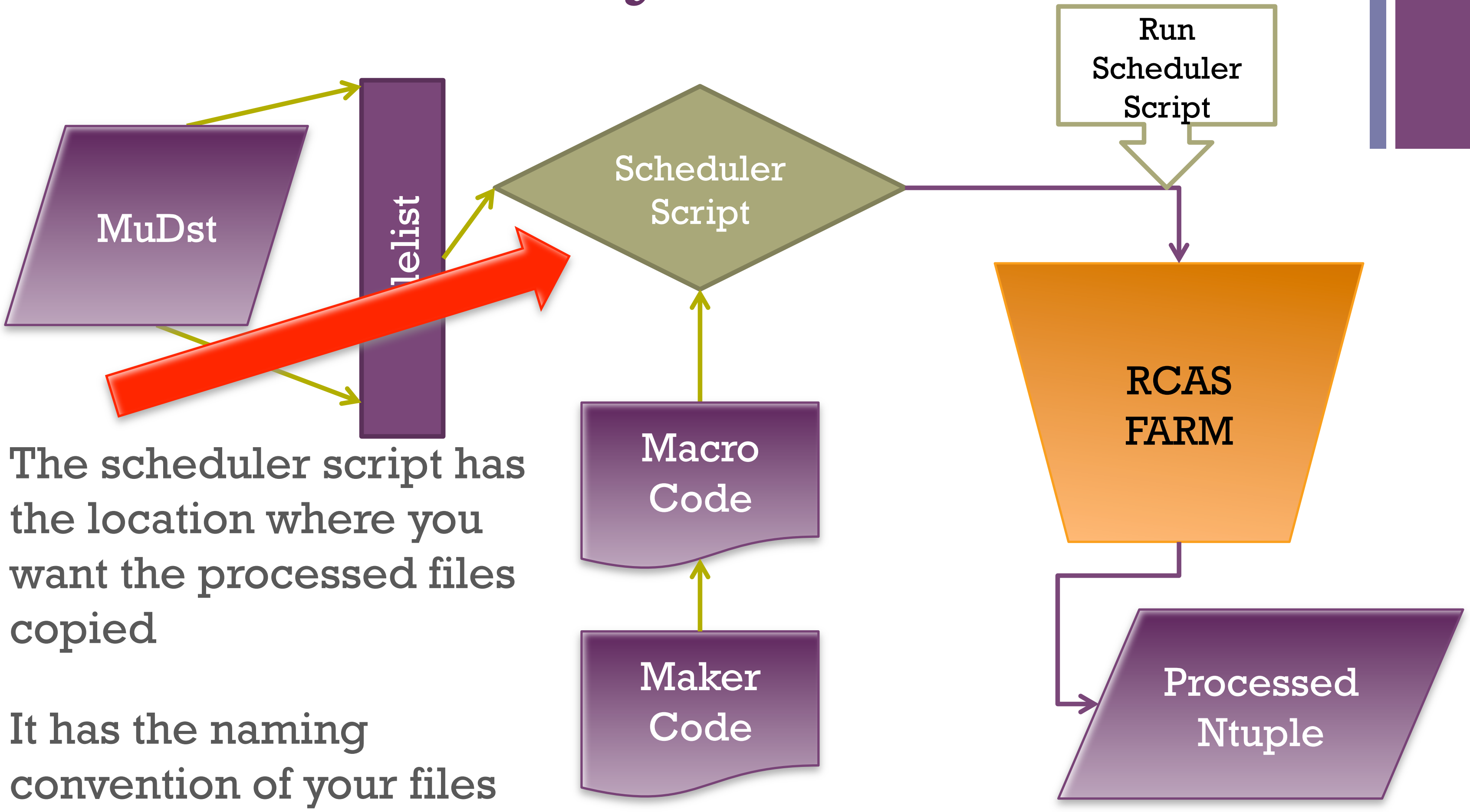
- Your Macro code describes the process which defines your Maker code, loads STAR libraries and executes your analysis

How Data Analysis Works



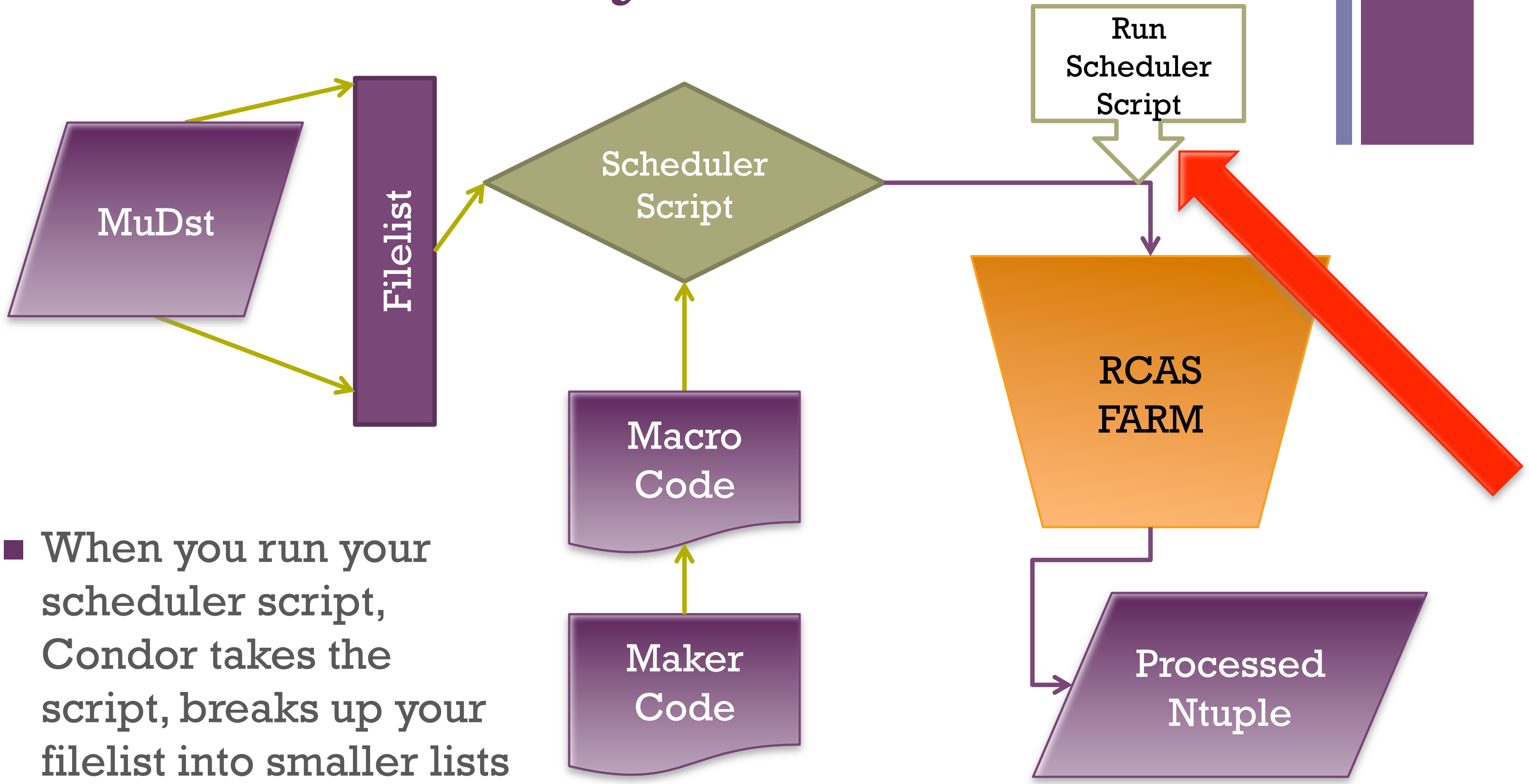
- The scheduler script has the location of your filelist and your macro code
- It has the list of commands you want to execute for every set of files you run over
- It defines the number of files per job that one RCAS node will run over

How Data Analysis Works



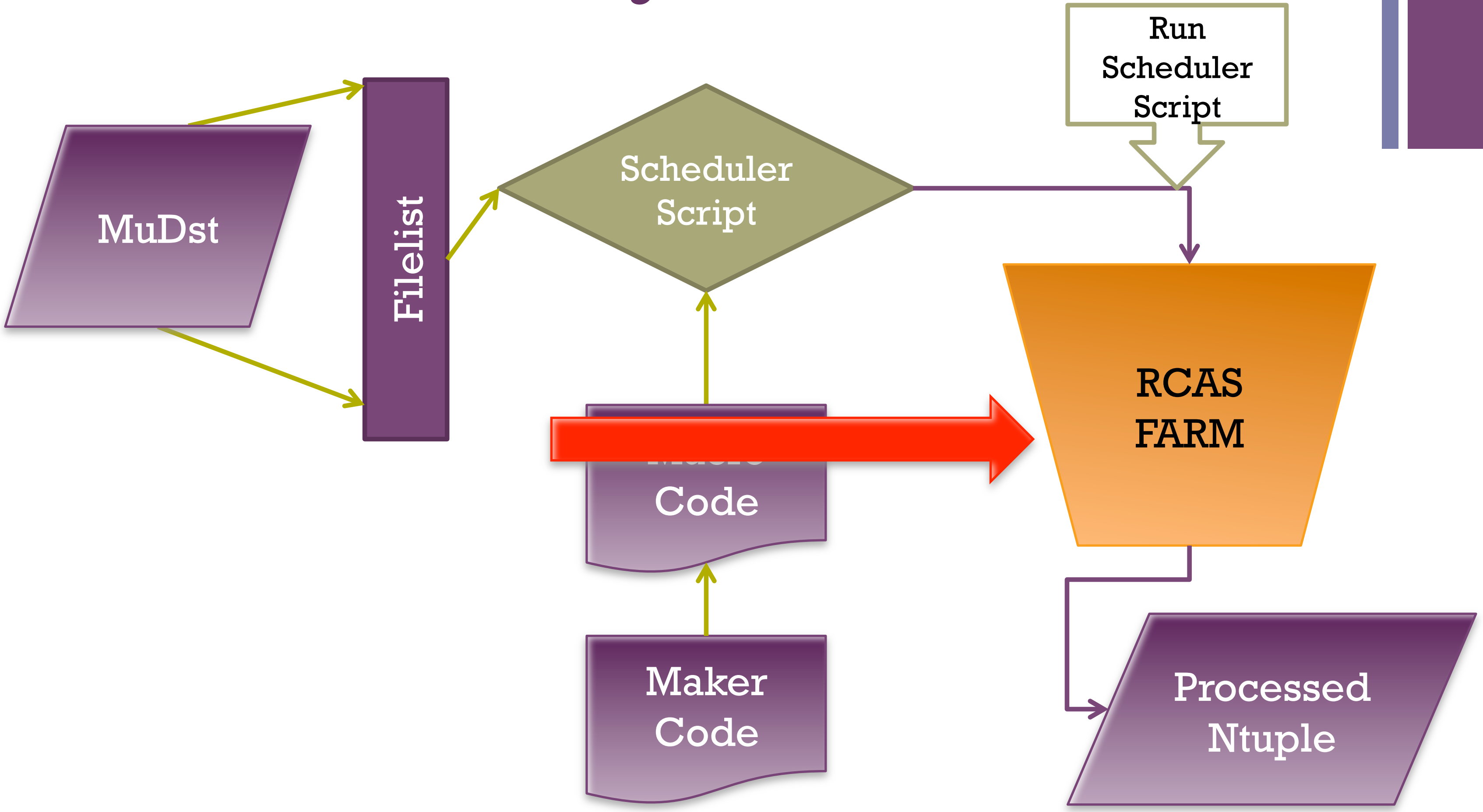
- The scheduler script has the location where you want the processed files copied
- It has the naming convention of your files and where the error and log files will be copied

How Data Analysis Works



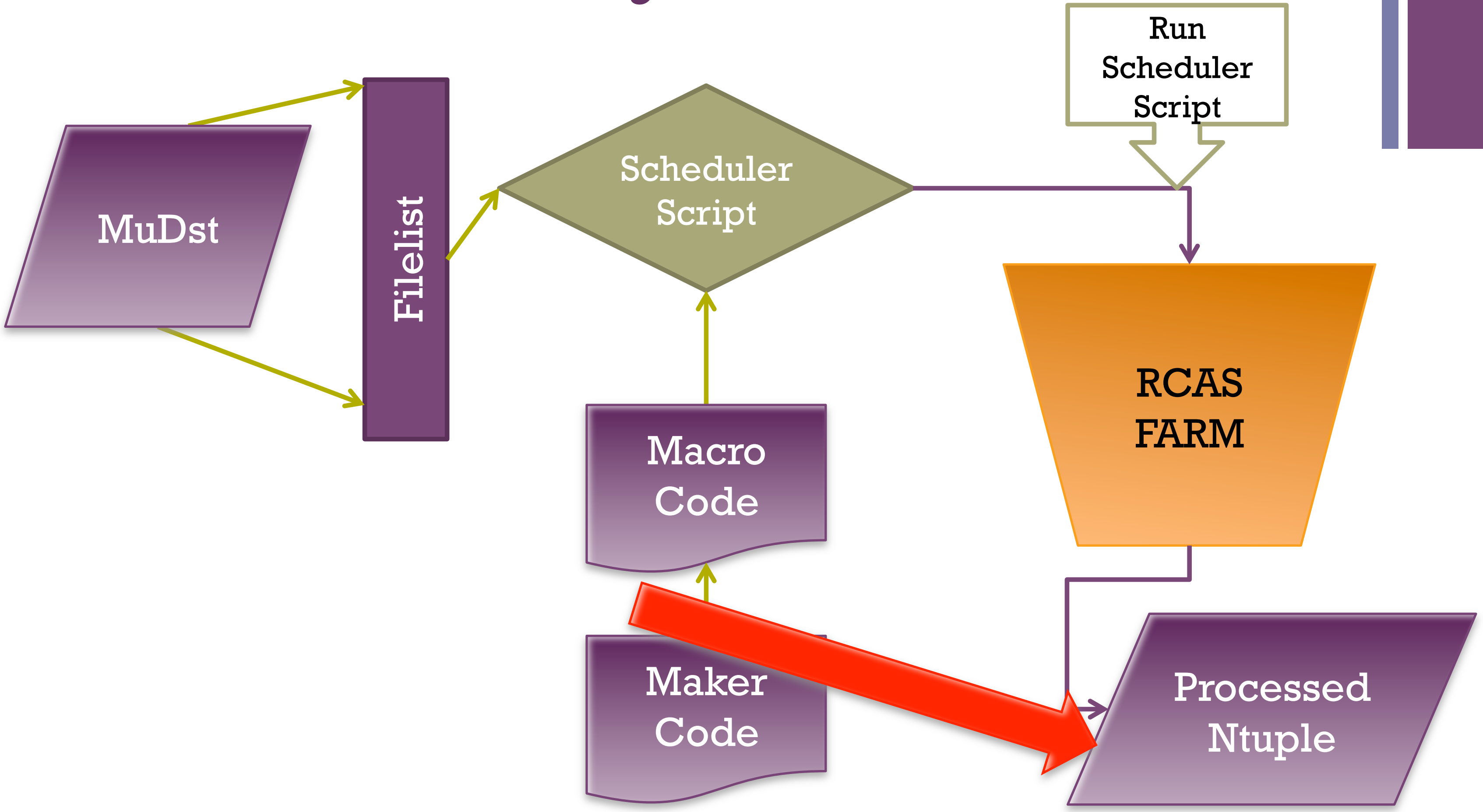
- When you run your scheduler script, Condor takes the script, breaks up your filelist into smaller lists for a single node to complete, or a job

How Data Analysis Works



- All your jobs are run on the RCAS farm

How Data Analysis Works



- Final processed data is then copied to the location specified in your scheduler script

Example 1!

- Plot the reference multiplicity in 2007 upsilon-trigger data
  - Need to pull files
  - Set up the runMaker macro
  - Run macro
  - Write macro to plot histogram
  - Plot histogram
- Starter files can be found
  - ~/> ls /star/u/sgbrovko/Tutorial/

Example 1!

- InitRun() initializes the runs
- Init()
  - Open a file for writing
  - Book the histograms
- Make()
  - Call the proper trigger ids
  - Fill histograms according to trigger id when id is accepted or rejected
- Finish() we write the histograms to a .root file and close the file

+

Example 2!

- Graph the invariant mass spectra of a K^0
  - Choose the decay mode of two pions
  - Access track information
    - Need dE/dx for particle identification (which particle?)
      - Also should be less than 600MeV
    - Need a proper number of track and fit points

Example 2!

- How to access track information?
  - Use StMuTrack
  - http://www.star.bnl.gov/webdata/dox/html/classStMuTrack.html
- Use StLorenzVector
  - Save four-momenta of the two pions
- Need to make pairs
  - Ensure you don't double count:

```
For( i=0; i<N; …){
        for(j=i+1; j<N …){
.
.
.
        }
}
```

+ Example 2!

- What next?
  - How to determine K^0 mass, of course
  - $E^2 = p^2 + m^2$
  - $p^\mu p_\mu = m^2$
  - Sum the momenta of the pions and find the mass
- What else?

Good Luck ^_^

- Questions?
- Email sgbrovko@ucdavis.edu