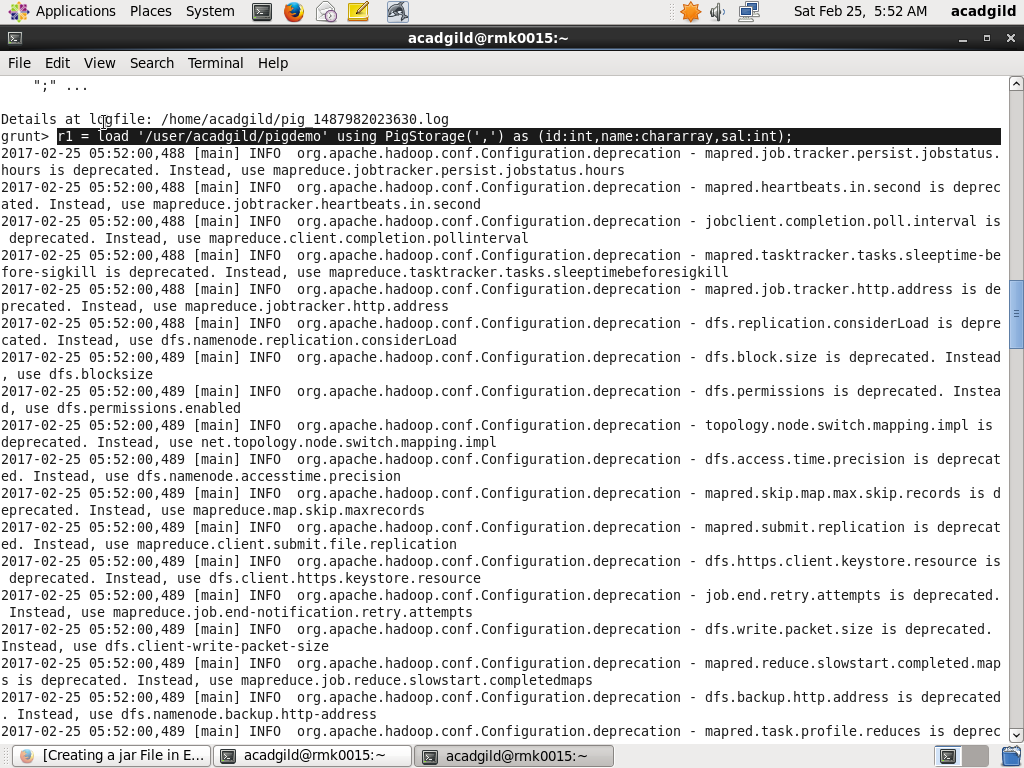
Load:

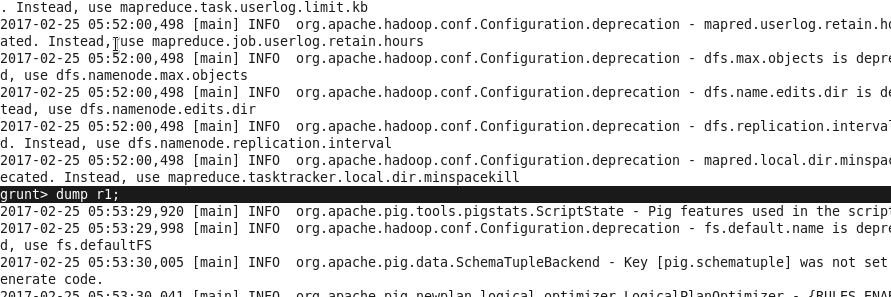
* Loads data from the file system.
* We can also specify relative path names to load the data.
* We can specify in which format the data should be loaded and saved by the using PigStorage command.
* The syntax is:

LOAD 'data' [USING function] [AS schema];

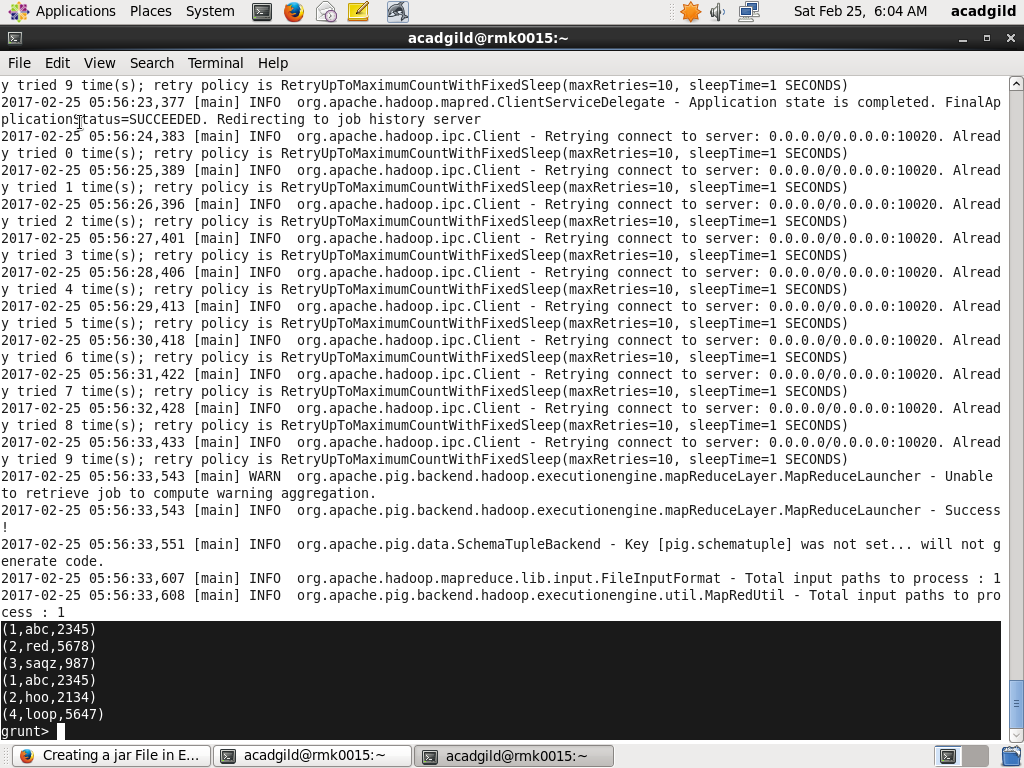
In the below example we are loading the file into relation:



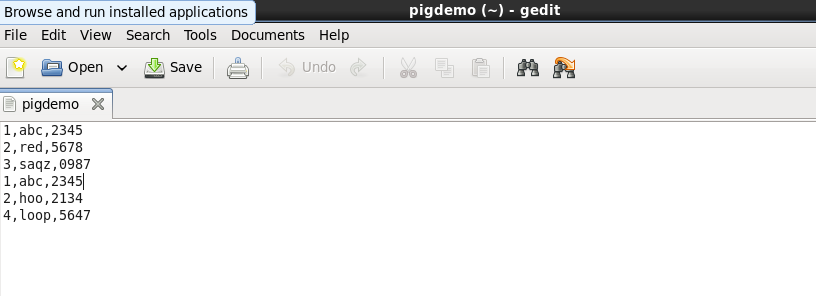
In the below example we are using the dump command to retrieve the details of relation which is loaded currently



In the below screen shot after executing the dump command the contents of the files are retrieved and displayed



Contents of file pigdemo:



STORE:

* Stores or saves results to the file system.
* Syntax :

STORE alias INTO 'directory' [USING function];

\*STORE A INTO 'myoutput' USING PigStorage ('\*');

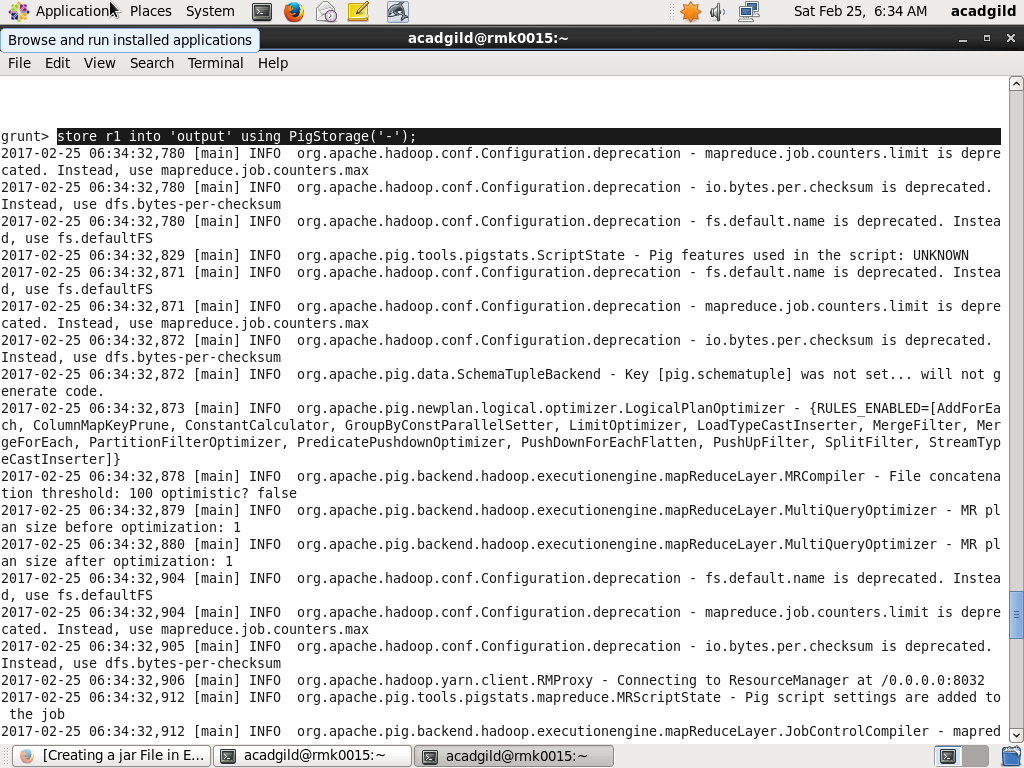
\*CAT myoutput;

1\*2\*3

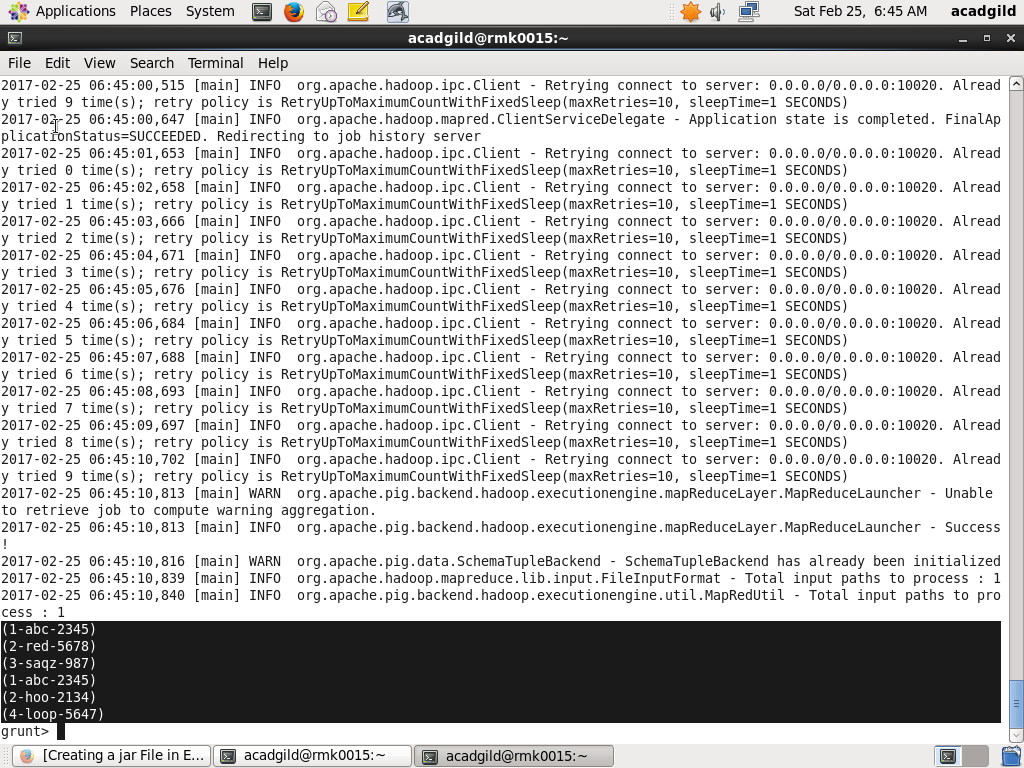
8\*3\*4

4\*2\*1

In the below snap store command is executed that is already created relation r1 is store in output using Storage(‘-‘)



In the below snap as mentioned in the store command that the data should be stored using ‘-‘ in output ,the dump command is used to display the output and it is



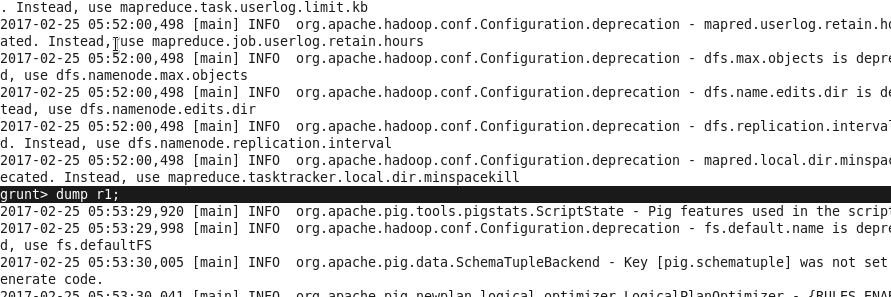
DUMP:

* Dumps or displays results to screen.

Syntax

• DUMP alias;

The below snap shows how the dump command is used to display the contents of r1



FOREACH:

* Generates data transformations based on columns of data.

SYNTAX:

alias = FOREACH generate\_operations [AS schema];

Terms

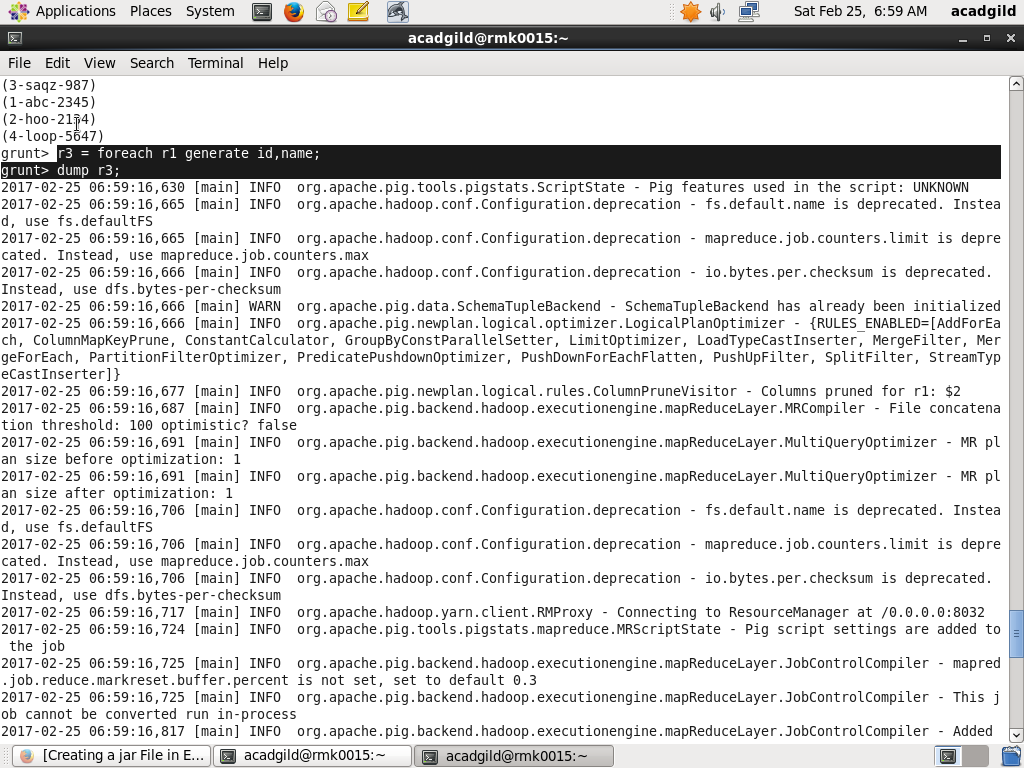
• alias

• The name of relation (outer bag).

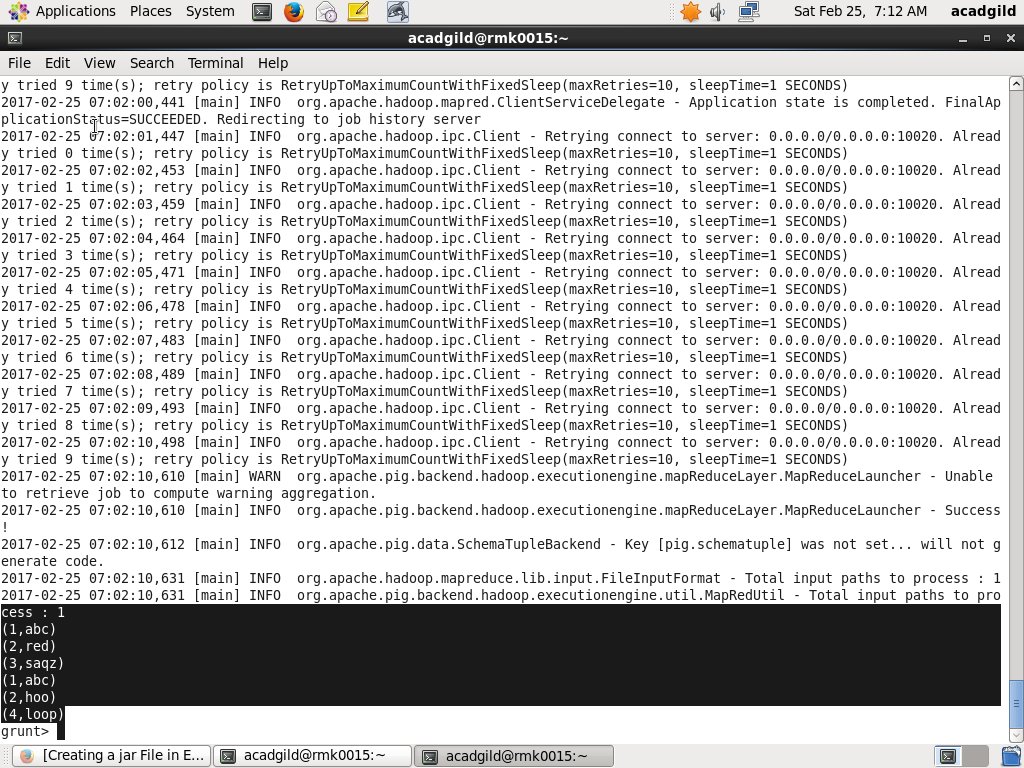
• generate operations

• FOREACH … GENERATE used with a relation.

In the below snap the foreach command is executed and dumped



The below snap shows after foreach command results:



FILTER:

* Selects tuples from a relation based on some condition.

Syntax

alias = FILTER alias BY expression;

Terms

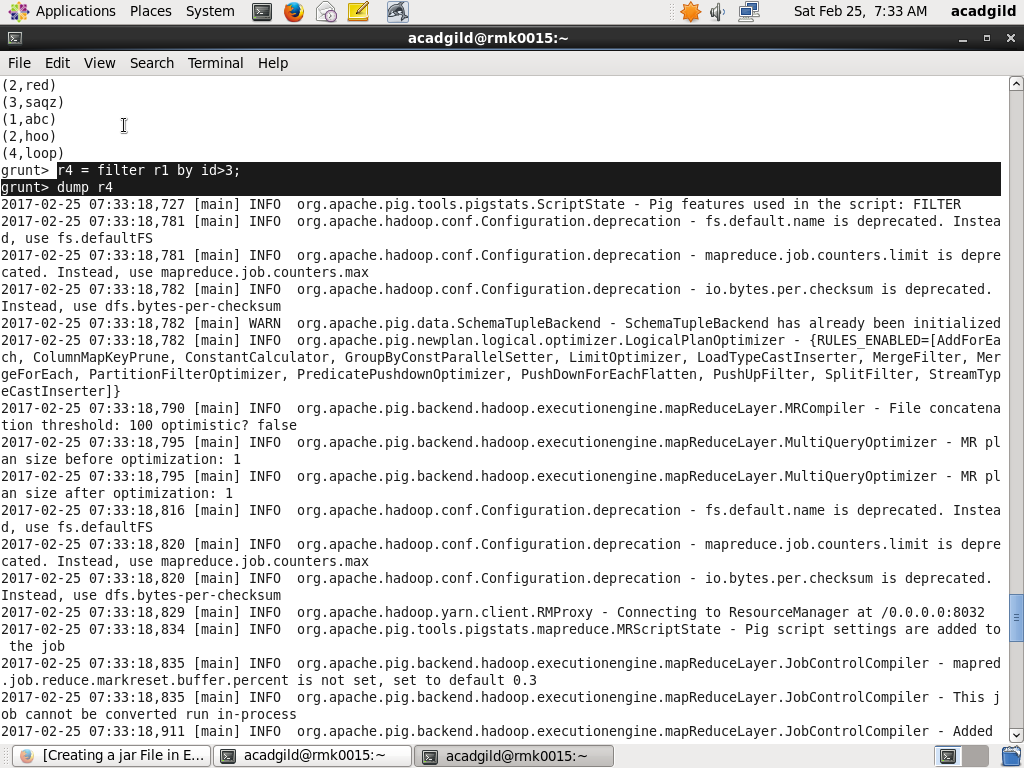
• alias

• The name of the relation

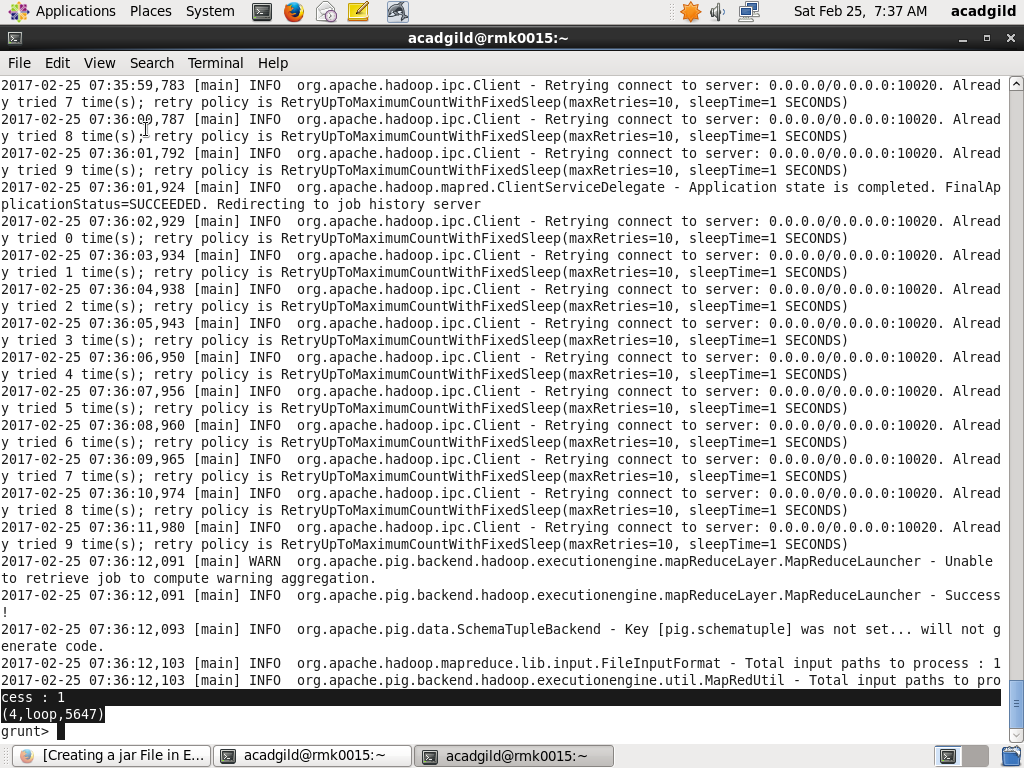
BY

• Required keyword.

In the below snap, the filter is applied to retrieve elements greater than 3



In the below snap, the output of the filter command is displayed



GROUP BY:

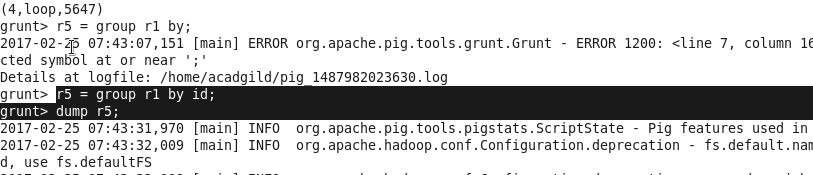
* Groups the data in one or multiple relations.

Syntax:

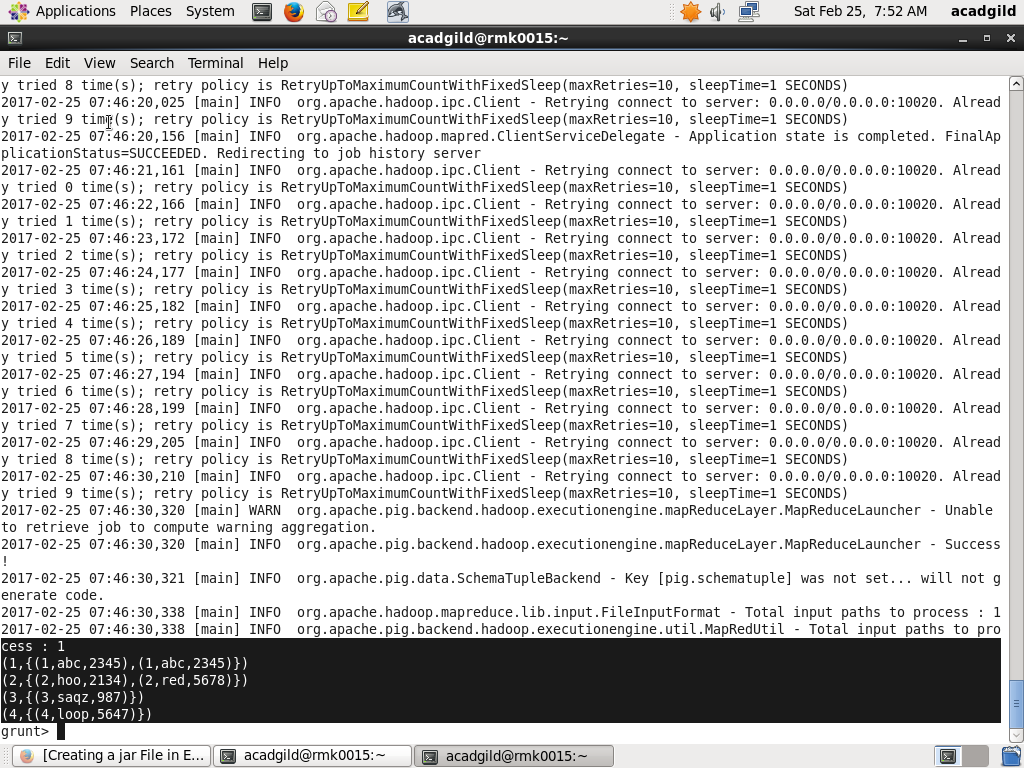
alias = GROUP alias { ALL | BY expression} [ alias ALL | BY expression …] [PARALLEL n].

* The GROUP operator groups together tuples that have the same group key.
* The key field will be a tuple if the group key has more than one field, otherwise it will be the same type as that of the group key.
* The result of a GROUP operation is a relation that includes one tuple per group. This tuple contains two fields:
* The first field is named "group" (do not confuse this with the GROUP operator) and is the same type as the group key.
* The second field takes the name of the original relation and is type bag.

The below snap shows the group by being applied to relation r1.



The below snap shows the output of the group by command



ORDER BY:

* Sorts a relation based on one or more fields.

Syntax

* alias = ORDER alias BY { \* [ASC|DESC] | field\_alias [ASC|DESC] [, field\_alias [ASC|DESC] …] } [PARALLEL n];

Terms alias

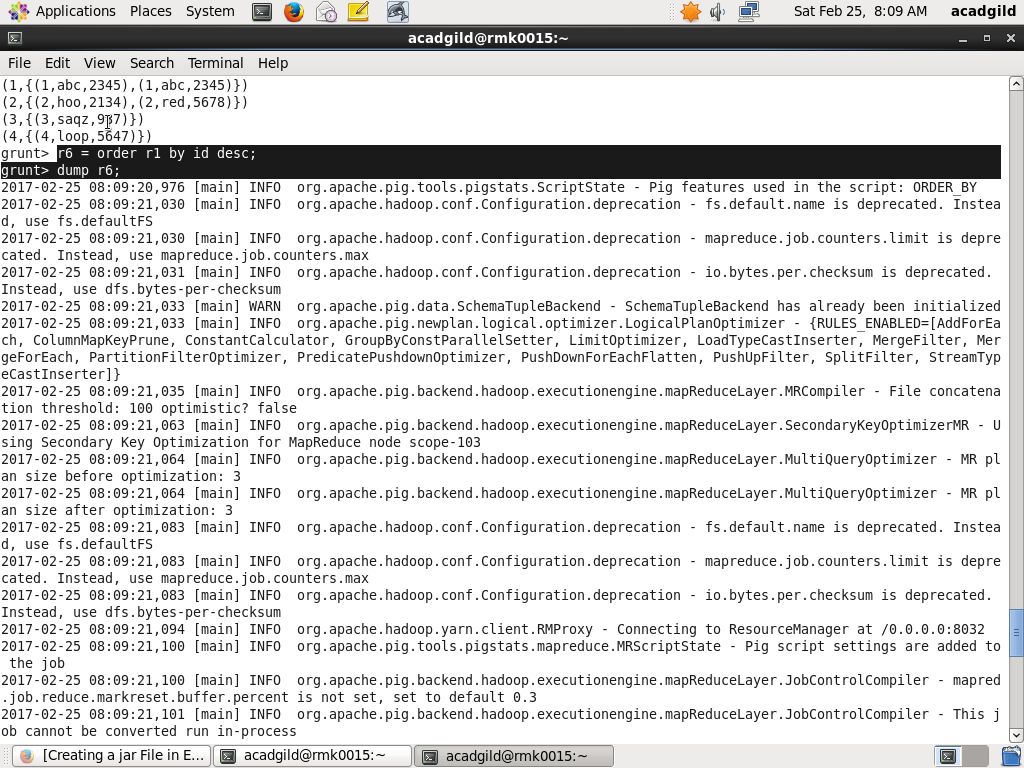
• The name of a relation. ASC

• Sort in ascending order. DESC

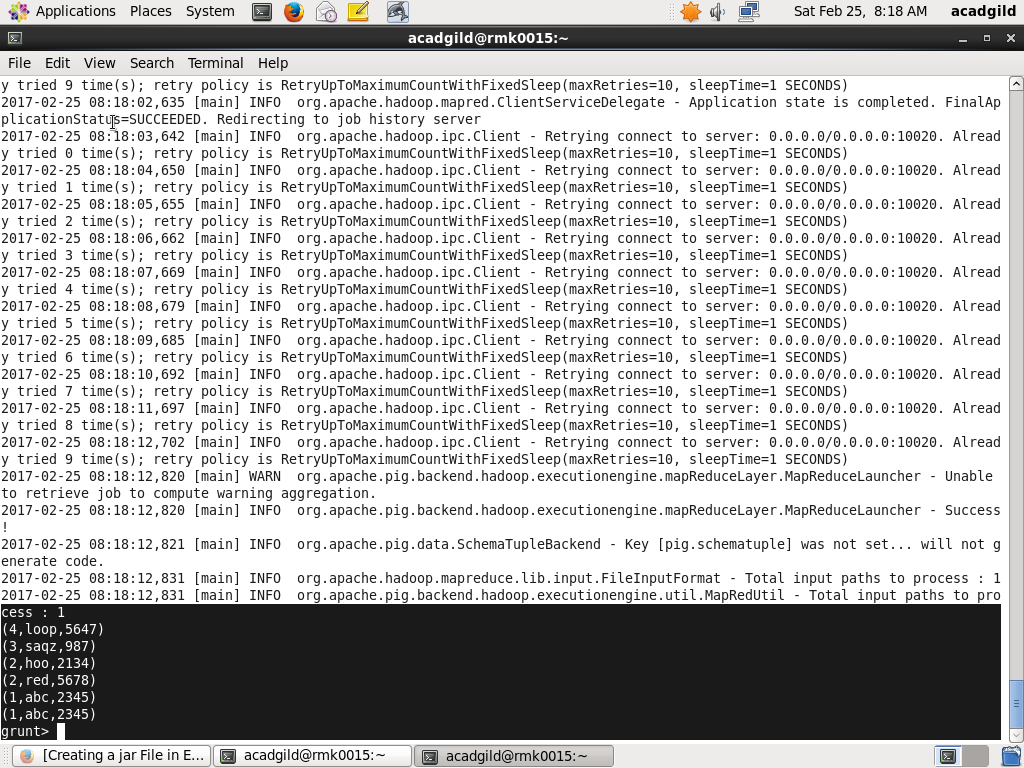
• Sort in descending order. field\_alias • A field in the relation. PARALLEL n

• Increase the parallelism of a job by specifying the number of reduce tasks, n. The default value for n is 1 (one reduce task)

The below snap shows the order by command being executed



The below snap shows the output of the order by command:



DESCRIBE :

* Returns the schema of an alias.

Syntax

• DESCRIBE alias;

The below snap shows execution of describe command

