Controlling script processes using threads and nodes

(Part 2 of External script technical training)

Click here to download the supporting SQL script

External Script Training Overview

This is the R script version

1. Introduction to external script interface
2. Controlling Script Processes – threads and nodes
3. Controlling Data Processing – partitions
4. Passing parameters through to scripts

Kognitio

These examples illustrate functionality but are not the best way to approach getting the overall average, why?

Controlling script processes:
threads and nodes

```
create external script Kog_R_AvgOverAllRows environment
rscript
receives(part_id int, value decimal(10,6))
input 'column_headers on, column_header_format 0'
sends(average decimal(10,6))
limit 1 threads
script S'EOF(
… <identical R script to part 1> …
)EOF';
```

Limit the external script to 1 thread only. All data streams through a single invocation yielding 1 average over all the data

Parallelism is lost and the performance that comes from it

```
create external script Kog_R_AvgOverAllRows environment
rscript
receives(part_id int, value decimal(10,6))
input 'column_headers on,column_header_format 0'
sends(average decimal(10,6))
limit 1 threads per node
run on node ('nodename')
script S'EOF(
… <identical R script to part 1> …
)EOF';
```

The same single result is obtained using a combination of limiting threads per node and specifying which node to run the script on

As well as loosing parallelism the script becomes tied to the named node. Script is no longer portable between systems. Kognitio automatically manages where to execute processes so generally not required

Kognitio

Controlling script processes:
when to limit threads and nodes

```
create external script Kog_R_AvgOverAllRows
environment rscript
receives(part_id int, value decimal(10,6))
input 'column_headers on, column_header_format 0'
sends(average decimal(10,6))
limit 1 threads
script S'EOF(
… <identical R script to part 1> …
)EOF';
```

There are some processes that cannot be parallelised or broken down into parallel steps. Limiting 1 threads allows users to run these scripts on Kognitio

```
create external script Kog_R_AvgOverAllRows
environment rscript
receives(part_id int, value decimal(10,6))
input 'column_headers on,column_header_format 0'
sends(average decimal(10,6))
limit 1 threads per node
run on 'nodename'
script S'EOF(
… <identical R script to part 1> …
)EOF';
```

Limiting threads per node is very useful for processes
requiring larger amounts of RAM. The parallelism is cut but not lost. The script invocations do not compete for RAM within the node. This is particularly useful if your processing makes use of multiple processors.

For use when the external executable is only installed on a subset of nodes. Due to licensing costs for example.

What happens when we try to run all data through a single process? (Remove the “where” clause)

Kognitio

Your query might simply take a long time to run or…

Controlling script processes:
R can be RAM intensive

…you may see this error

R can be RAM hungry as using read.csv tries to import all input rows at once. If the data is too large to fit into the RAM allocated to script invocation the execution will fail

Checking the logs you can see R cannot allocate vector

```
--
-- Run the same query over all data by removing predicate
--
select * from
(external script Kog_R_AvgOverAllRows from
   (select prodno, price/100.00 from demo_ret.v_ret_sale
--         where storeno = 1
    )
) ext;
```

Error | Results (0 rows) | Metadata | Info

Error: Go To Error

HY000[Kognitio][WX2 Driver][POC12] RS0023: Error writing to external script pipe

Info:
Data could not be properly sent to the external script. It may have exited without consuming all data.

Action:
Check the server logs and debug the script.

```
external script DEMODATA.DEBUG_LOG
order by 1 desc;
```

Results (100 rows) | Metadata | Info

	LOG_OUTPUT
1	T_2014-01-27_10:35:12_GMT: AM id 0x2f00000f: abort request received f … session=645119, tno=-1, session aborted...
2	T_2014-01-27_10:35:12_GMT: AM #2f00000f aborting now
3	T_2014-01-27_10:34:52_GMT: RS 0 S 645000 R 33700B8 LO:Script stderr: Ex…ution halted
4	T_2014-01-27_10:34:52_GMT: RS 0 S 645000 R 33700B8 LO:Script stderr: Error: cannot allocate vector of size 250.0 Mb

What is the best way to compute the average over all the data?

Contact your system administrator regarding the external script limits or learn more about Kognitio RAM management here

Kognitio

Controlling script processes:
making use of parallelism

```
create external script Kog_r_AvgOverAllRows environment
rscript
receives(part_id int, value decimal(10,6))
input 'column_headers on,column_header_format 0'
sends(totalvalue decimal(18,3), numrows int)
script
S'EOF(
          read_in<-read.csv(file=file("stdin"),
          header=TRUE)
          total <- sum(read_in$VALUE)
          counter <- nrow(read_in)
          cat(file="",paste(total, counter,
               sep=",", collapse="\n"),"\n")
)EOF';

select sum(totalvalue)/sum(numrows) Avg1
from (external script Kog_rAvgOverAllRows
      from (select prodno, price/100.00
            from demo_ret.v_ret_sale
            )
      )ext;
```

Design your scripts so that you make full use of the parallelism whenever possible.

If you can code the problem in SQL it will generally be faster.

This code emulates how Kognitio breaks down and parallelises the SQL avg function.

It is easy to utilise Kognitio to further process external script output; using SQL (like here) or passing it into another external script. This allows data to be streamed through a series of scripts to perform very complex tasks

The “where” clause is removed as parallelism means larger data sets can be handled

As well as controlling the script processes in Kognitio it is easy to control the data flow. See part 3 for details

Kognitio