Welcome to our webinar!

- This webinar starts in 5 minutes - please stay tuned
- This webinar will be recorded
- You will get the slides, recording and SQL snippets
- During the webinar, you may ask questions using the Q&A button - you may ask questions anonymously
- You can hop on and off on all the sessions

Day 3	15:00 - 16:00 Self-service Troubleshooting 16:15 - 17:15 Job Management Outlook

DATA VIRTUALITY

Topic: Self service troubleshooting

Agenda

- Trouble Categorization
- Where to look for information
  - Logging inside Data Virtuality studio
  - Data Virtuality Performance Monitoring
  - other Log Files (boot.log, gc.log, Configuration Database log files)
- Data Virtuality Server Log Files
  - Accessing and Parsing the Data Virtuality Server Log Files from inside Data Virtuality Server
  - Concept and usage of extended log levels and the LogMsg() procedure
- BONUS: Knowing when (not) to restart

Trouble categorization

Trouble Categorization (1)

- the Data Virtuality Server stopped
  - first action: start Data Virtuality Server
- the Data Virtuality Server runs, but is unavailable
  - first action: restart Data Virtuality Server
- the Data Virtuality Server does not correctly start
  - check boot.log and server.log for any obvious, repairable reason

Trouble Categorization (2)

- the Data Virtuality Server is available, but too slow
  - check what could cause slowness
  - restart the server
- Errors
  - check error message
- Unexpected behavior
  - create a support ticket
- further action: call support *(with boot.log and server.log)*

Where to find informations

Logging inside Data Virtuality Studio

- QueryLog and JobLog in the Data Virtuality Studio show history of queries and jubs run by Data Virtuality Server
- including duration, query plans, finishing state and errors
- valuable (beside other things) for
  - detecting changes in behavior over longer time
  - looking up graphical versions of QueryPlans
- based on syslog.JobLogs, syslog.QueryLogs and syslog.getQueryLogPlan(querPlanId)

Demo - Infos in DV studio

Data Virtuality Performance Monitoring

- system and query/job details of the of the last 24 hours
- will log queries against system schemas (QueryLog does not log all of them)
- available at http://<YourDvServer>:8080/monitoring/#/
- can be enabled/disabled via 'ENABLE_PERFORMANCE_MONITORING' property (defaultOptionValue)
- based on system tables syslog.QueryPerformanceLog and syslog.SystemPerformanceLog

Demo - Performance Monitoring

Log files

- Log files are stored at dvserver/standalone/log/*
- the main Data Virtualitity Server Logfile is `server.log`
- the main logfile is rotated on daily base, old files are kept with a timestamp appended to name
- will log - beside a lot of other details - by default error stacktraces that might be not visible in the Query-/JobLog error details
- might be configured to log more details on specific parts of the software or to log custom messages defined in own code

Other log files (1)

- boot.log
  - information collected on server start, system settings, bootstrapping errors
  - new file on each server start, old file will be overwritten
- gc.log
  - garbage collector logs, valuable source of information on performance issues
  - new file ‘gc.log’ on each server start, old file is kept and gets a timestamp appended

Other log files (2)

- .hprof files
  - heap dump files, created on JAVA OoM events
  - one file per incident
- configuration Database log files
  - located either in log folder of Data Virtuality Server or as configured in the PostgreSQL database
  - rotated as per configuration (weekly in case of embedded PostgreSQL)

Community and Support

- https://support.datavirtuality.com/hc/en-us
- support@datavirtuality.com or support@datavirtuality.de

Working with the server.log

Accessing the Data Virtuality Server Log Files from inside Data Virtuality Server (1)

- Create a data source of type *file*, which points to the directory storing the server log files

```
call SYSADMIN.createConnection
        (
                name => 'ds_serverlog'
                , jbossCliTemplateName => 'ufile'
                , connectionOrResourceAdapterProperties => 'ParentDirectory=/opt/datavirtuality/dvserver/standalone/log/'
        )
;;

call SYSADMIN.createDatasource
        (
                name => 'ds_serverlog'
                , translator => 'ufile'
                , modelProperties => 'importer.useFullSchemaName=false'
        )
;;
```

Accessing the Data Virtuality Server Log Files from inside Data Virtuality Server (2)

- Get the whole file

```
SELECT f.file
FROM
        (
                call "ds_serverlog".getFiles('server.log')
        ) f

;;
```

- The same approach can be used to access other log files than server.log (e.g. server.log.<date> files, boot.log, gc.log)
- The result of such a query can be exported into a file using the Export-to-CSV mechanism*

Parsing of Data Virtuality Server Log Files inside Data Virtuality Studio

```
SELECT
       case when "csv"."record" like '%:%:%[%]%' then cast (substring("csv"."record", 0, 12) as string) else null end as "logtime"
    ,  case when "csv"."record" like '%:%:%[%]%' then cast (trim(substring("csv"."record", 13, 7)) as string) else null end as "type"
    ,  case when "csv"."record" like '%:%:%[%]%' then cast (trim(substring("csv"."record", 20, locate(']',"csv"."record") - 19)) as string) else null end
       as "logger"
    , case when "csv"."record" like '%:%:%[%]%' then trim(substring("csv"."record", locate(']',"csv"."record") + 1)) else "csv"."record" end as "entry"
FROM
    (call "ds_serverlog".getFiles('server.log')) f
    , TEXTTABLE
        (
            to_chars(f.file,'UTF-8')
            COLUMNS "record" STRING
            DELIMITER '€'
            QUOTE ''
         ) csv;;
```

Demo - server.log in studio

extended logging

Concept and usage of extended log levels

Enable more detailed logging to see the entire stacktrace.

```
--run this query
SELECT "salesorderid", "currencycode" FROM "ds_mysql.salesorderheader" where currencycode='USD';;

--enable logger
EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=org.teiid.CONNECTOR:add(level=DEBUG)');;

--rerun the query
SELECT "salesorderid", "currencycode" FROM "ds_mysql.salesorderheader" where currencycode='USD';;

--check the result
SELECT "logtime", "type", "logger", "entry" FROM "views.parsing_log" where lower(entry) like '%binary%';;

--disable logger
EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=org.teiid.CONNECTOR:remove');;
```

→ no change in the query log
→ source query in server.log

Additional logger

```
EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.datavirtuality.METADATA:add(level=TRACE)');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.datavirtuality.METADATA.columns:add(level=DEBUG)');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=org.teiid.CONNECTOR:add(level=DEBUG)');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.google.api.client.http.HttpTransport:add(level=DEBUG)');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.datavirtuality.connectors.facebook:add(level=DEBUG)');;
```

Disabling logger

```
EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.datavirtuality.METADATA:remove');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.datavirtuality.METADATA.columns:remove');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=org.teiid.CONNECTOR:remove');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.google.api.client.http.HttpTransport:remove');;

EXEC SYSADMIN.executeCli(script => '/subsystem=logging/logger=com.datavirtuality.connectors.facebook:remove');;
```

Demo - extended logging

Concept: adding logMessages

Adding log messages to the code

```
CALL "SYSADMIN.logMsg"(
    "level" => 'string_level'/* Mandatory */,
    "context" => 'string_context'/* Mandatory */,
    "msg" => object_msg/* Mandatory */
 );;

BEGIN
        LOOP ON (SELECT SalesPersonID FROM "ds_pg.salesperson" ) AS cur
                BEGIN
                        INSERT INTO "dwh.SalesPerson" SELECT cur.salespersonID AS "SalesPerson";
                        CALL "SYSADMIN.logMsg"("level" => 'INFO',"context" => 'Salesperson', "msg" => 'Salesperson: '||cur.salespersonID||' imported');
                END
END;;
```

Demo - log messaging

BONUS: Server restart when (not)

When (not) to restart

- restart (sudo service datavirtuality stop (start))
  - server seems to be unresponsive (please create a ticket anyway for investigation)
  - Java Heap space error (please create a ticket anyway for investigation)
  - some maintenance cases (e.g. adding a new certificate to cacerts, adding system properties in standalone.conf.props(.bat))
- no restart
  - during adding or removing a datasource
  - during a start of the Data Virtuality server

Any feedback / questions?

Thank you!

Please feel free to contact us at:
info@datavirtuality.com

or

visit us at:
datavirtuality.com