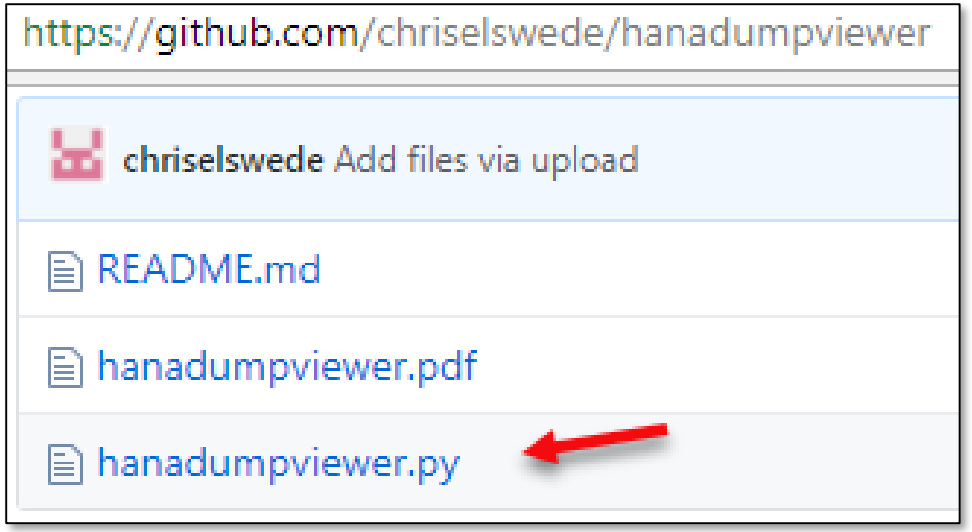
Monitoring

HANADumpViewer – SAP Note 2491748

SAP Note 2491748 presents a tool that can help view SAP HANA Dump files

2491748 - How-To: Analyzing Runtime Dumps with SAP HANADumpViewer

- It is a python script to be downloaded from https://github.com/chriselswede/hanadumpviewer
- It is intended to be executed as <sid>adm on your SAP HANA Server
  (since then the proper python version is in your path, installed together with SAP HANA)
- It does not need any user in hdbuserstore since it never connects to SAP HANA



- So it can also be executed from your local laptop (e.g. on windows, see later slides)

For more about the SAP HANADumpViewer see SAP Note 2491748

HANADumpViewer – Dot Files

The SAP HANA Dump Viewer creates .dot files out of .trc files:

Dump File (.trc)

```
[STACK_SHORT]  Short call stacks and pending exceptions
335413741[thr=40822]: JobWrk0058 is inactive
--
50744[thr=45394]: SqlExecutor at
 1: 0x0000000000000000 in <no symbol>+0x0 (<unknown>)
 2: 0x00007faaa1855ed6 in AnalyticalAuthorization::SqlA
 3: 0x00007faaa0700f35 in ptime::qo_PredGrantInjection:
 4: 0x00007faaa0702d3c in ptime::qo_PredGrantInjection:
 5: 0x00007faaa06f676a in ptime::qo_Rule::checkAndApply
 6: 0x00007faaa030933c in ptime::qo_Normalizer::applyRu
 7: 0x00007faaa0309b0e in ptime::qo_Normalizer::rewrite
 8: 0x00007faaa030a0f1 in ptime::qo_Normalizer::normali
 9: 0x00007faaa028b5a6 in ptime::QueryOptimizer::optimi
```

Dot File (.dot)

```
digraph StackGraph {
ratio=compress
rankdir=BT
nlegend [shape=record,label="{{#T =
process}}",style=filled,fillcolor="
nC0
[shape=record,label="{ltt::allocate
lled,fillcolor="#ffffff",fontname=s
nC1
```

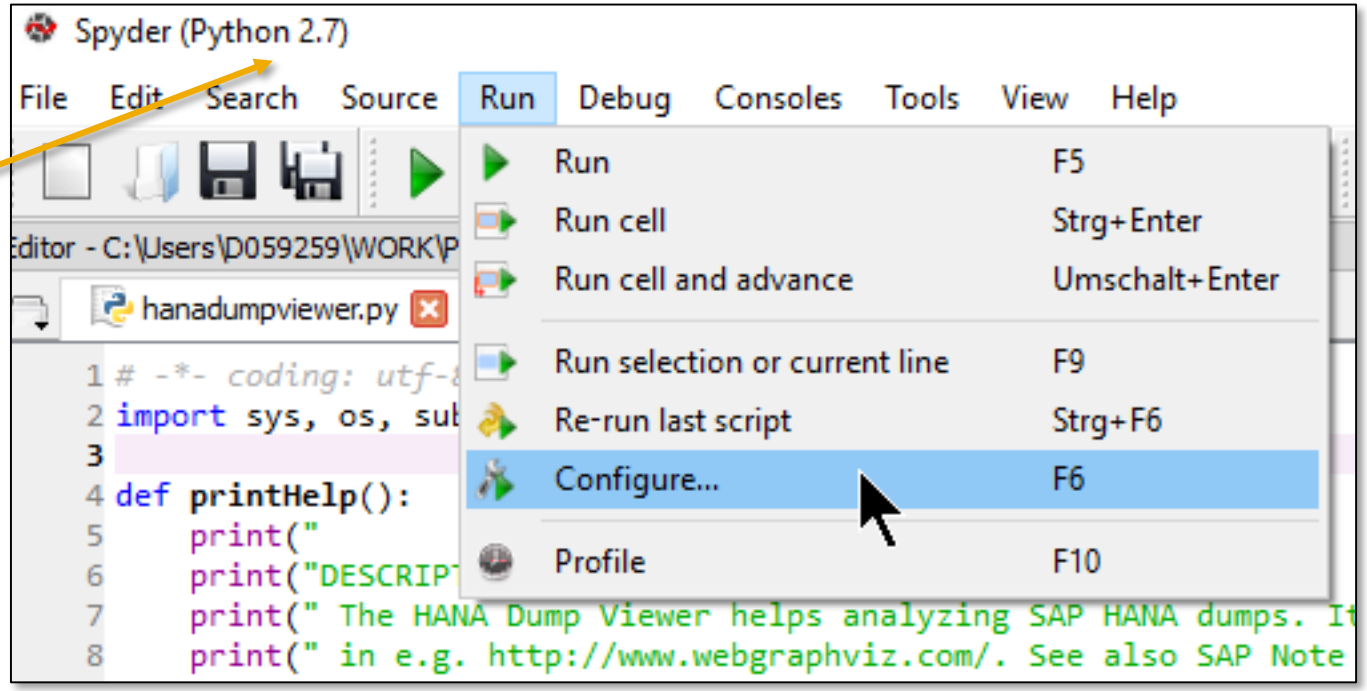
The Dot File can then easily be viewed in e.g. http://www.webgraphviz.com (see next slide)

HANADumpViewer – On Windows (1/3)

There are uncountable ways of running a python program on Windows
Here is one example of how HANADumpViewer can run from Windows

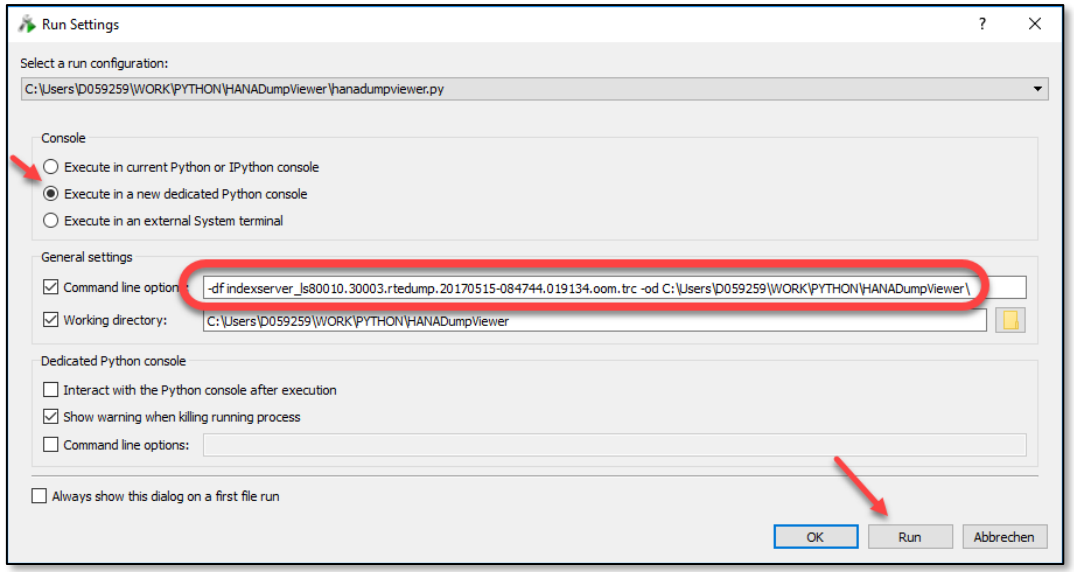
Download and install Python(x,y) with e.g. these instructions , then open handumpviewer.py in Spyder (that comes with Python(x,y)) and Run → Configure

Make sure you get a 2.7.x version as this is still the default version on your HANA server and this is the version HANADumpViewer was developed for



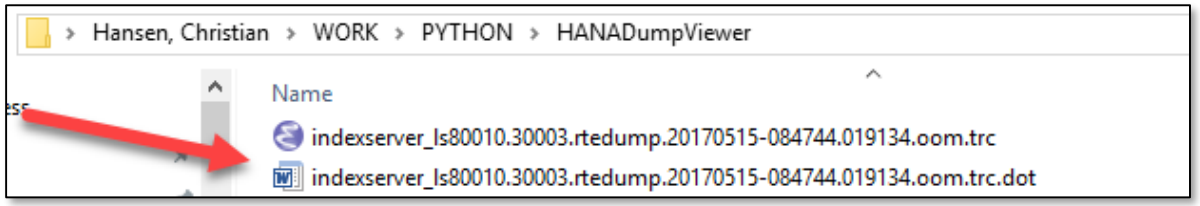
HANADumpViewer – On Windows (2/3)

Specify the input file with the `-df` flag and output folder with the `-od` flag



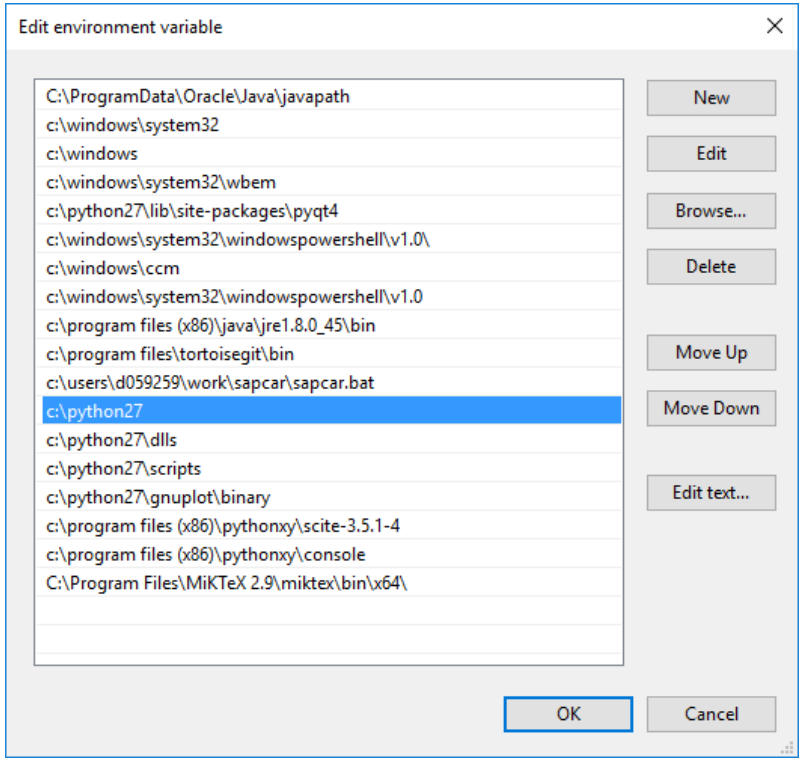
In the python console we see that the .dot file was created

And we find it in Windows Explorer



HANADumpViewer – On Windows (3/3)

Make sure the python binaries can be found in your path (Control Panel → System → Advanced Settings → Environmental Variables → Path → Edit)

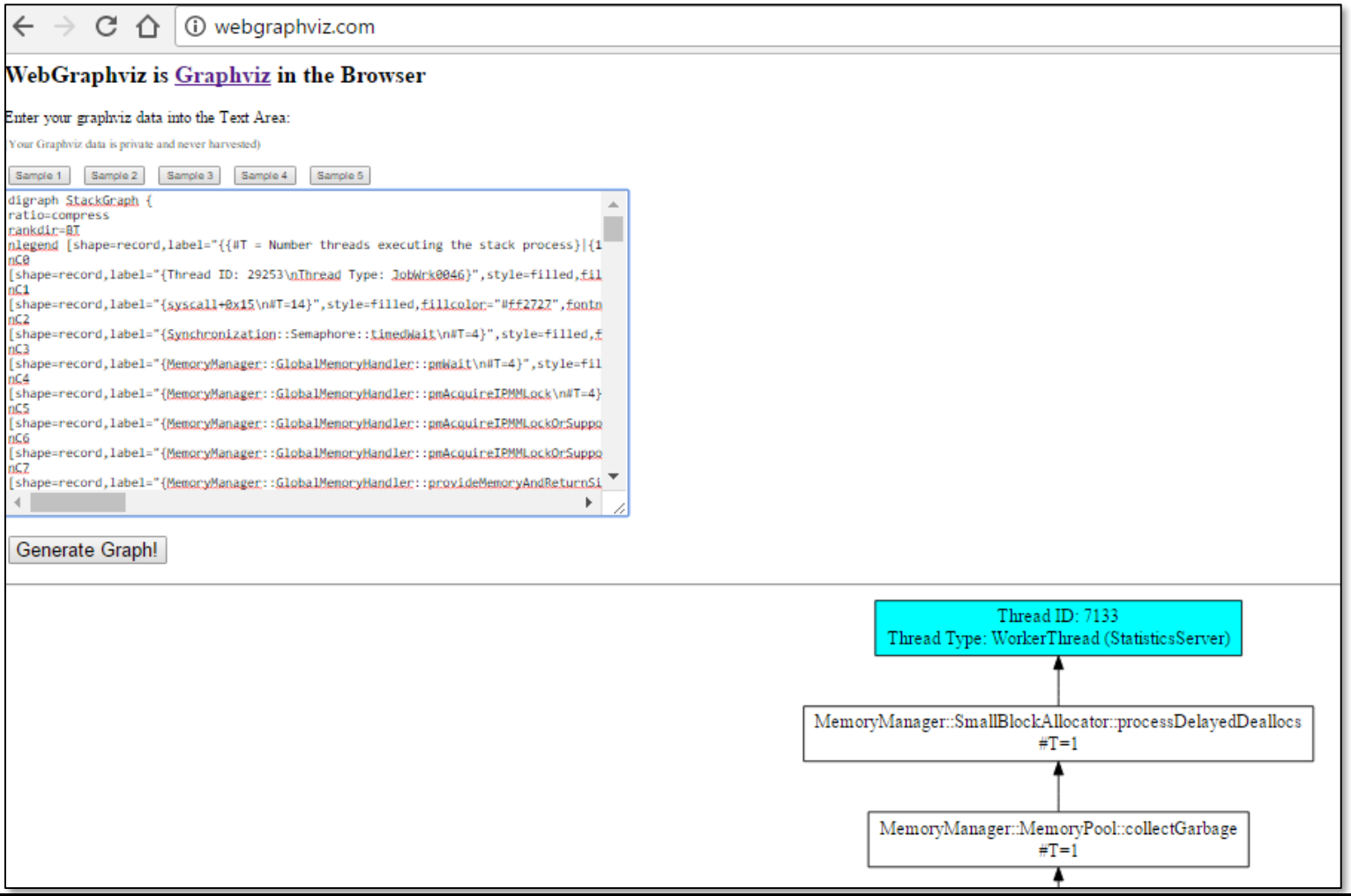


Then HANADumpViewer can be executed from your CMD prompt:

```
C:\Users\D059259\WORK\PYTHON\HANADumpViewer>python hanadumpviewer.py -df rtedump.txt -od C:\Users\D059259\WORK\PYTHON\HANADumpViewer\
File C:\Users\D059259\WORK\PYTHON\HANADumpViewer\/rtedump.txt.dot was created
```

HANADumpViewer – Graph Viz

The Dot Files can easily be viewed in e.g. http://www.webgraphviz.com



HANADumpViewer – Dump Files

HANA Dump Viewer takes either the full paths of dump files or finds them in the folder defined by the alias “cdtrace”

Flag	Unit	Details	Explanation	Default
-df		dump file names	List of dumpfile path names, separated by a comma only	‘’
-nd		number indexserver dumpfiles	Number of indexserver .trc files that HANA Dump Viewer will try to collect from “cdtrace” and create .dot files for	0
-dt		dump type	HANA Dump Viewer will only look for indexserver .trc files of this type, i.e. this string has to be part of the dump file name, e.g. rte, oom, …	‘’

Example:
Here 2 indexserver .trc files are specified and for each a .dot file is created:

```
mo-fc8d991e0:/tmp/HANADumpViewer> python hanadumpviewer.py -df indexserver_ls80010.oom.trc,indexserver_mo.rtedump.trc
File /tmp/hanadumpviewer_output/indexserver_ls80010.oom.trc.dot was created
File /tmp/hanadumpviewer_output/indexserver_mo.rtedump.trc.dot was created
```

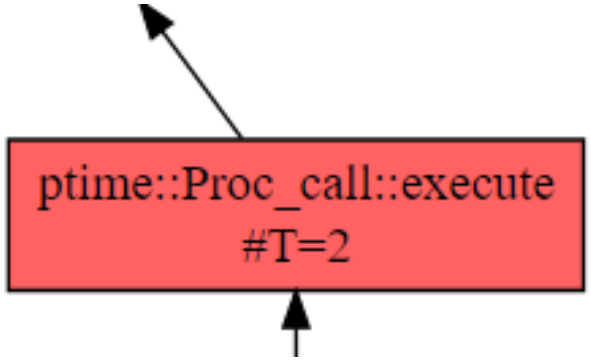
Example:
Here 3 crashdump indexserver .trc files are collected from “cdtrace” and for each a .dot file is created:

```
mo-fc8d991e0:/tmp/HANADumpViewer> python hanadumpviewer.py -nd 3 -dt crashdump
File /tmp/hanadumpviewer_output/indexserver_mo-fc8d991e0.30003.crashdump.20170227-042318.006632.trc.dot was created
File /tmp/hanadumpviewer_output/indexserver_mo-fc8d991e0.30003.crashdump.20170321-114146.044877.trc.dot was created
File /tmp/hanadumpviewer_output/indexserver_mo-fc8d991e0.30003.crashdump.20170408-102430.001820.trc.dot was created
```

HANADumpViewer – Dot Files

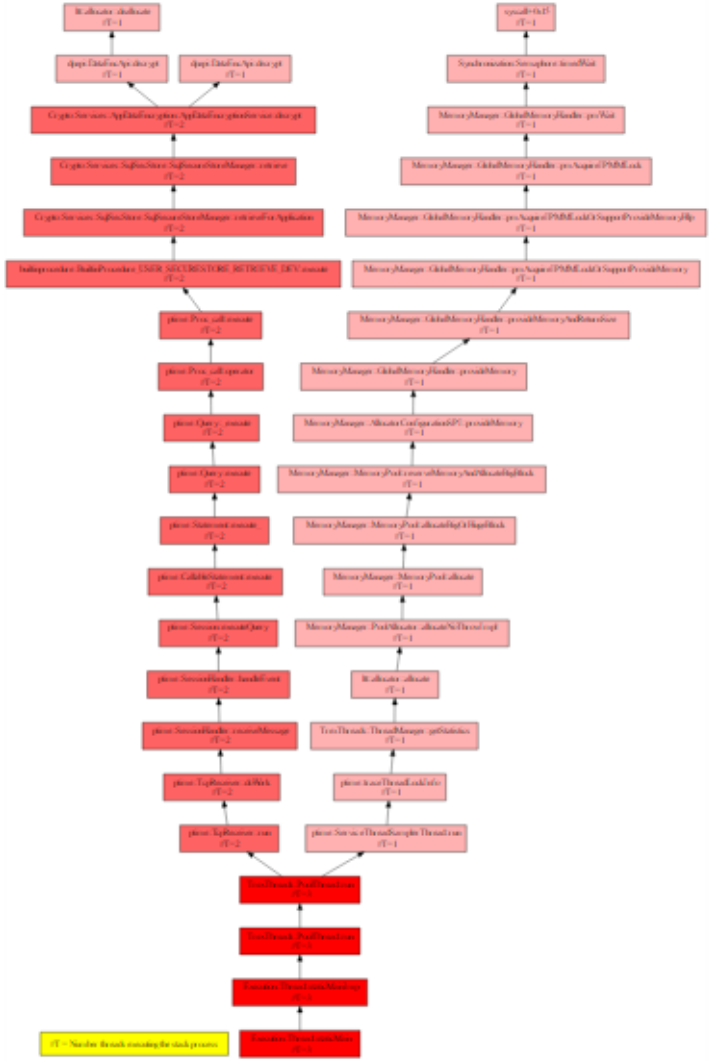
The .dot files can be viewed in e.g. http://www.webgraphviz.com

Each stack component is shown in its own box:



The red color scale and the #T label shows how many threads that executed that particular stack process, as is also explained in the legend box:

#T = Number threads executing the stack process



HANADumpViewer – Plot Threads

One can also specify to add the thread definitions in the .dot files

Flag	Unit	Details	Explanation	Default
-pt	true/ false	plot threads	Show the thread definitions as additional components	false

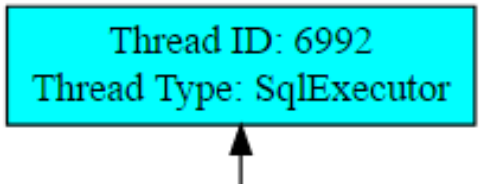
Example:
Here .dot files with thread definitions are created from 3 crashdump indexserver .trc files from “cdtrace”:

```
mo-fc8d991e0:/tmp/HANADumpViewer> python hanadumpviewer.py -nd 3 -dt crashdump -pt true
File /tmp/hanadumpviewer_output/indexserver_mo-fc8d991e0.30003.crashdump.20170227-042318.006632.trc.dot was created
File /tmp/hanadumpviewer_output/indexserver_mo-fc8d991e0.30003.crashdump.20170321-114146.044877.trc.dot was created
File /tmp/hanadumpviewer_output/indexserver_mo-fc8d991e0.30003.crashdump.20170408-102430.001820.trc.dot was created
```

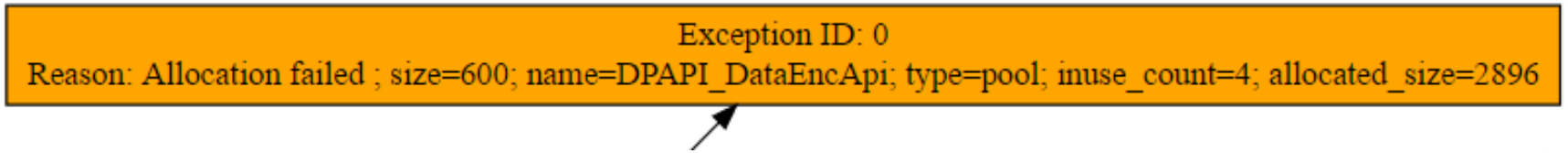
HANADumpViewer – Threads

There are two type of threads;
Normal threads and **Exception threads**

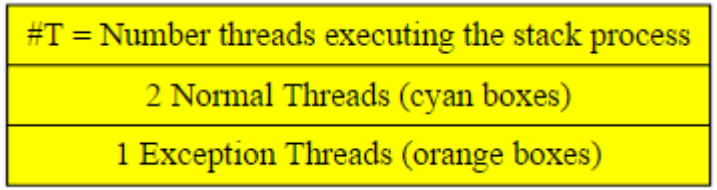
Normal threads are shown in cyan colored boxes and have a thread id and thread type

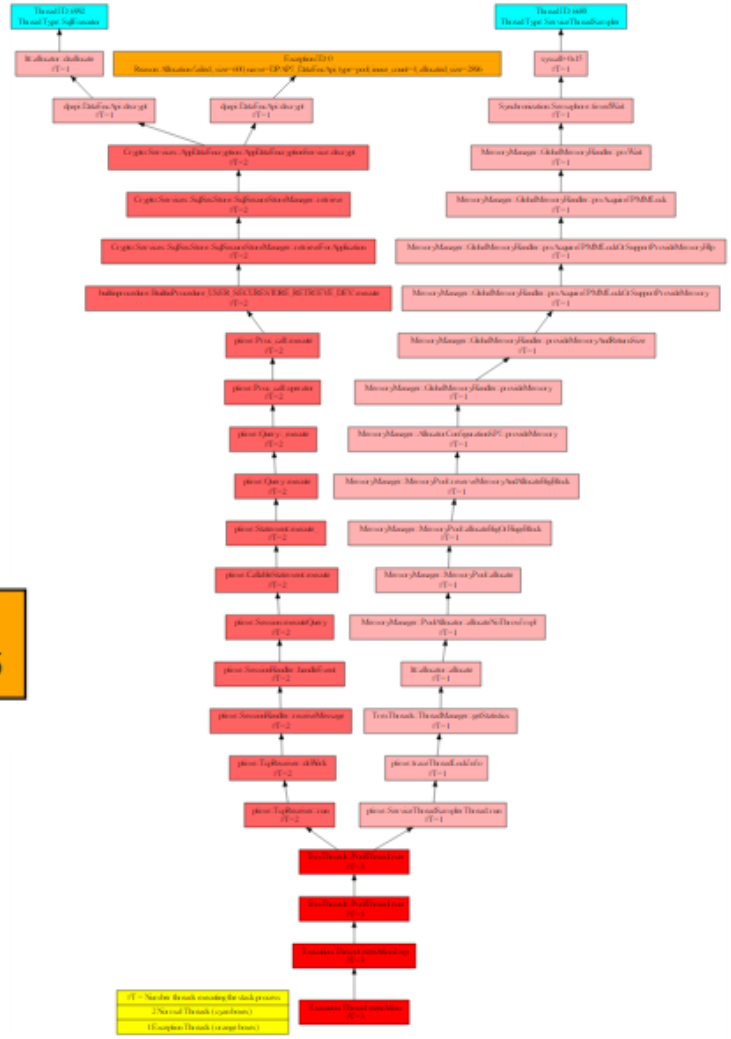


Exception threads are shown in orange colored boxes, their id are simply their order number in the .trc file and they have a reason for failure:



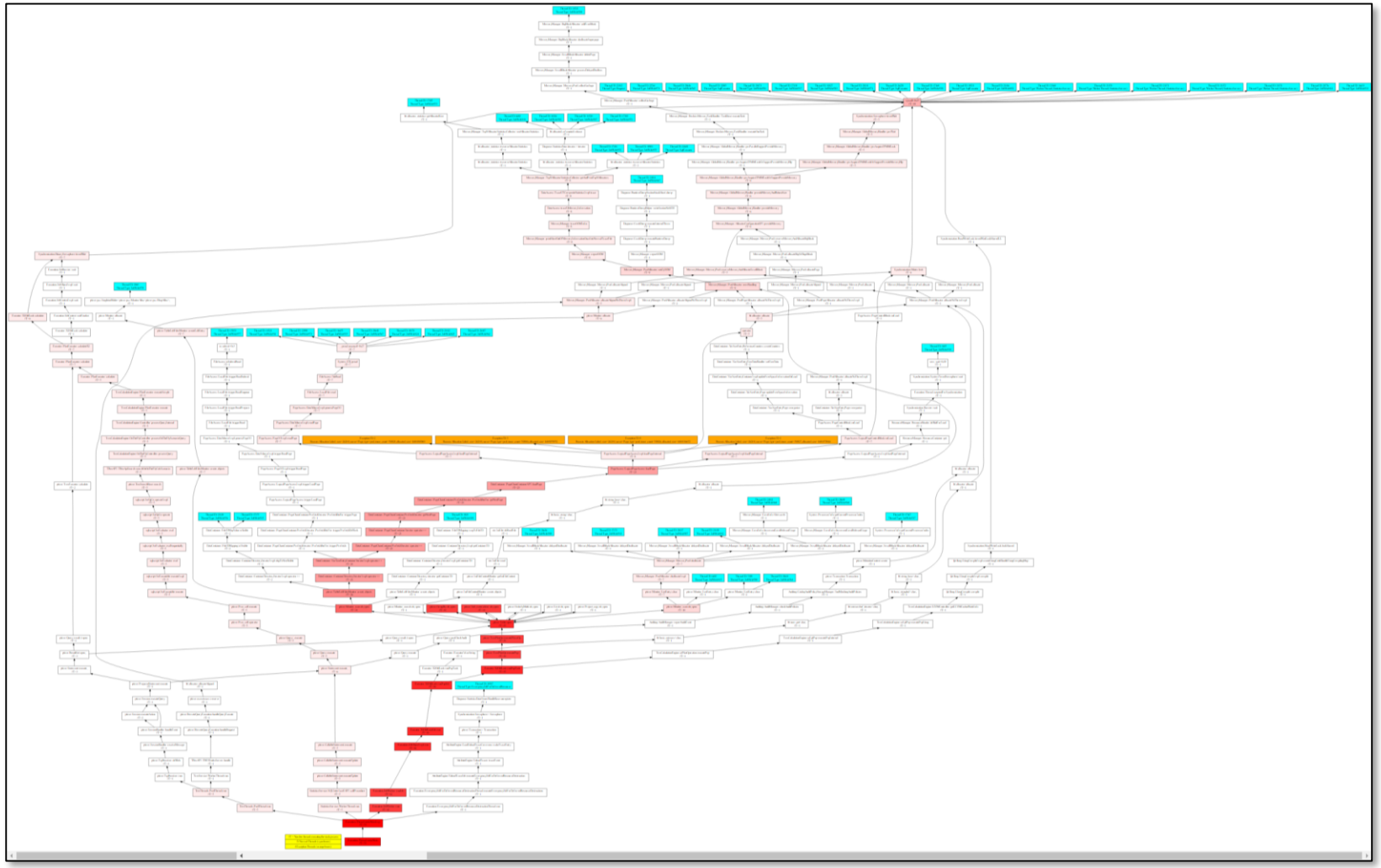
The Legend is showing number of threads:





HANADumpViewer – Large Results

The result from the HANA Dump Viewer can be rather extensive:



HANADumpViewer – Wait Graph

The section [INDEXMANAGER_WAITGRAPH] consists of a .dot graph

With the flag –mw this .dot graph is saved in a dedicated file:

Flag	Unit	Details	Explanation	Default
-mw	true/ false	make wait graph	Creates a file with the indexserver wait graph which is simply the content of the section [INDEXSERVER_WAITGRAPH]	false

Example:

```
oqladm@ls80010:/tmp/HANADumpViewer> rm -r ../hanadumpviewer_output/
oqladm@ls80010:/tmp/HANADumpViewer> python hanadumpviewer.py -md false -mw true -df rtedump.txt
oqladm@ls80010:/tmp/HANADumpViewer> ls ../hanadumpviewer_output/
indexmanager_waitgraph_rtedump_txt.dot
```

The content of this file can be viewed with e.g. webgraphviz.com:

HANADumpViewer – Views for Excel

All system views, e.g. the M_ views, are dumped in the dump files

With the flag –mv these files are saved in a dedicated .csv files:

Flag	Unit	Details	Explanation	Default
-mv	true/ false	make view files	Creates a .csv file for for each system view, e.g. the M_ views, in a VIEW folder in the output directory	false

Example:

```
oqladm@ls80010:/tmp/HANADumpViewer> rm -r ../hanadumpviewer_output/
oqladm@ls80010:/tmp/HANADumpViewer> python hanadumpviewer.py -md false -mv true -df rtedump.txt
oqladm@ls80010:/tmp/HANADumpViewer> ls ../hanadumpviewer_output/VIEWS_rtedump_txt/
DATABASE_STATISTICS.csv                    M_DEV_REP_LOGSENDER.csv                    M_LAZY_
EMERGENCY_JOBS.csv                         M_DEV_REP_TRANSACTIONS.csv                 M_LOG_B
M_BLOCKED_TRANSACTIONS_.csv                M_DEV_REP_TRANSLOGREPLAYER.csv             M_LOG_R
```

These files can be opened with e.g. Excel:

CONNECTION_ID	OBJECT_HANDLE	TRANSACTION_ID	STAR
-350604	1,59575E+13	823	1494
-350724	1,67648E+13	559	1494
-350232	1,58932E+13	745	1494
-350049	1,64568E+13	687	1494
-350384	1,62905E+13	564	1494