SQLAdria - Vodice, June 2025

Guillaume Arnould / Diego Cardalliaguet

Contents

IBM SQL Data Insights for Db2 v13 Installation Lab

The purpose of this repository is to guide the participant though a worked example of how to setup and use SQL Data Insights. The hands on lab consists of a Windows Image (with 3270 emulator and putty) and a Linux on Z image running a z/OS guest.

The HOL was originally written in github using markdown. If you prefer to access these instructions in a web browser, open the repository at the link below.

LINK TO THE INSTALLATION GUIDE_

From there you can download the PDF version of this guide.

1. SQL DI Deployment Overview and Planning

This section explains the context of the SQLDI Hands on Learning

1. The components and dependences of an SQLDI Instance
2. The HOL Environment that will be used for the Setup Lab

1.1 The Components and Dependencies of an SQLDI Instance

The first step is to understand all the components that are needed to build an SQLDI environment, their dependencies, and how they interact with each other. Study the diagram below, and read the notes that follow.

The SQLDI Server is a set of services running in USS, which interact with artefacts in Db2 z/OS.

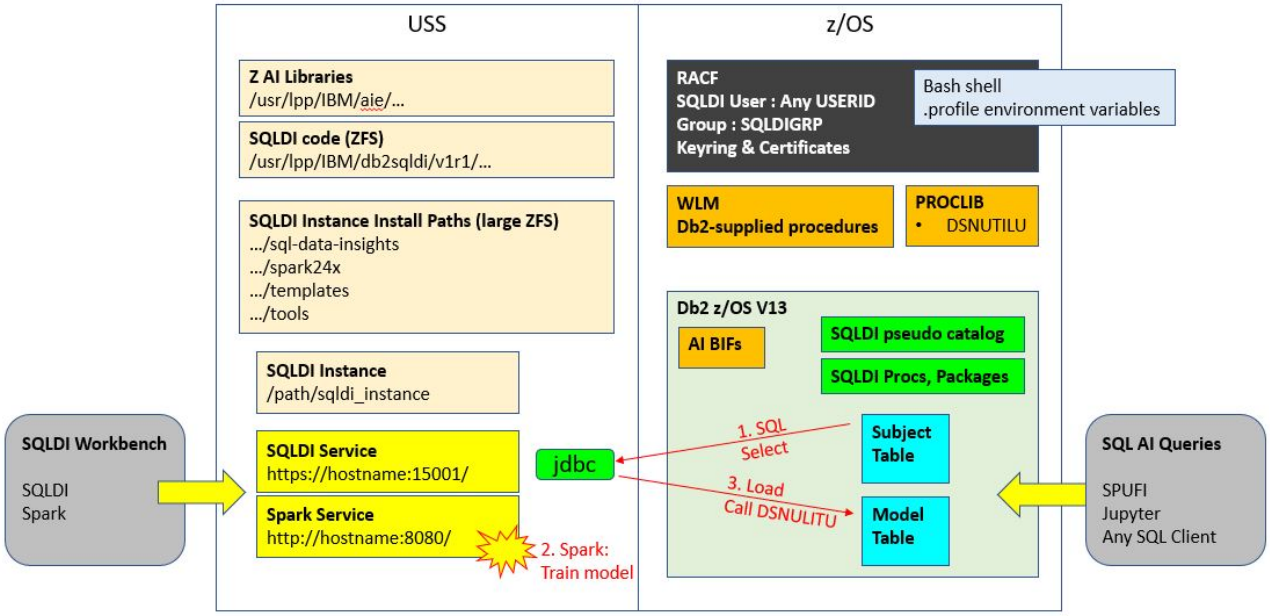


Figure 1: SQLDI Components

Left Hand Side: The SQLDI Server instance SQL Data Insights is a very self-contained package that installs easily into USS. It includes it's own copy of Spark, db2 jdbc driver, model training services, web server for the SQLDI browser interface, as well as tools like the bash shell. All you need to do is install it from the correct SMP/E FMID that comes with Db2 v13 (5698-DB2, FMID: HDBDD18).

After installation, you must create an SQLDI instance. This is dependent on a number of USS and z/OS pre-requsuites being set correctly.

- The USS environment must reflect the correct PATH and LIBPATH variables for the various Z AI libraries, the deep learning compiler etc. . .
- The script to create an SQLDI instance must be executed by the SQLDI instance userid, which must have the correct RACF properties
- It must reference a RACF keyring containing the authentication certificate to access z/OS
- It must also specify the network ports that SQLDI and Spark will use.
- It must also specify a USS paths with enough space to deploy the SQLDI instance.

The script to create an SQLDI instance is easy to invoke. The hard work is the careful planning and provisioning of the environment that it needs.

Right Hand Side: The Db2 z/OS V13 subsystem, with several artifacts to support the SQLDI Server instance You need

- a standard Db2 V13 system, with the DB2-provided procedures and their WLM environments and associated PROCLIB members.
- The SQLDI pseudo catalog for the SQLDI server to store metadata
- Several new SQLDI stored proedures, Three UDFs and corresponding packages to be bound.
- A new WLM environment for the UDFs to execute in, and create a PROCLIB member to invoke them.

The notes above are a summary of the key pre-requisite considerations for SQLDI deployment. A comprehensive list of requiremnents is published in the SQLDI Knowledge Center for Db2 13.

1.2 The HOL Environment that will be used for the Setup Lab

The Hands on Learning lab is hosted in a virtualised environment accessed via the Cloud using ZVA. Booking requests can be made by IBMers, so that the environment will not be freely ready after the labs you are running today.

The diagram below illustrates the nature of ZVA, and how to access it. Documented here: ZVA_System_Access..

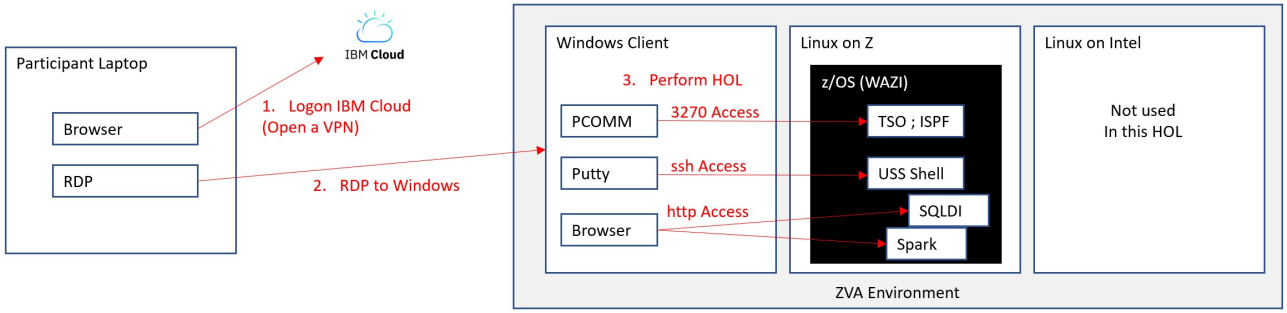


Figure 2: zva

The rest of this document contains the instructions for you to install and configure SQLDI on the ZVA image.

You will NEED to Know The following

Userids and Passwords in z/OS.

The z/OS userids and passwords that you will be using once you have accessed the ZVA or ZTrial system are

- IBMUSER (password SYS1) is a high privilege z/OS userid with Db2 Access
- AIDBADM (password aidbadm) is the userid that will be used as the SQLDI instance owner.

TCPIP hostnames.

Do **NOT** attempt to use TCPIP addresses during this HOL. The z/OS TCPIP stack has not been customised during the ZVA provisioning process. You must use hostnames, which have been setup.

From **Windows** you should point the applications (PCOMM and putty) at the z/OS system using hostname wg31.

From **USS** (where SQLDI runs) you should define your SQLDI and Spark instances to be located at hostname wg31.washington.ibm.com.

TASK

Let's check that all the components needed in USS are in place.

You will need to login as **ibmuser** into the PuTTY terminal.

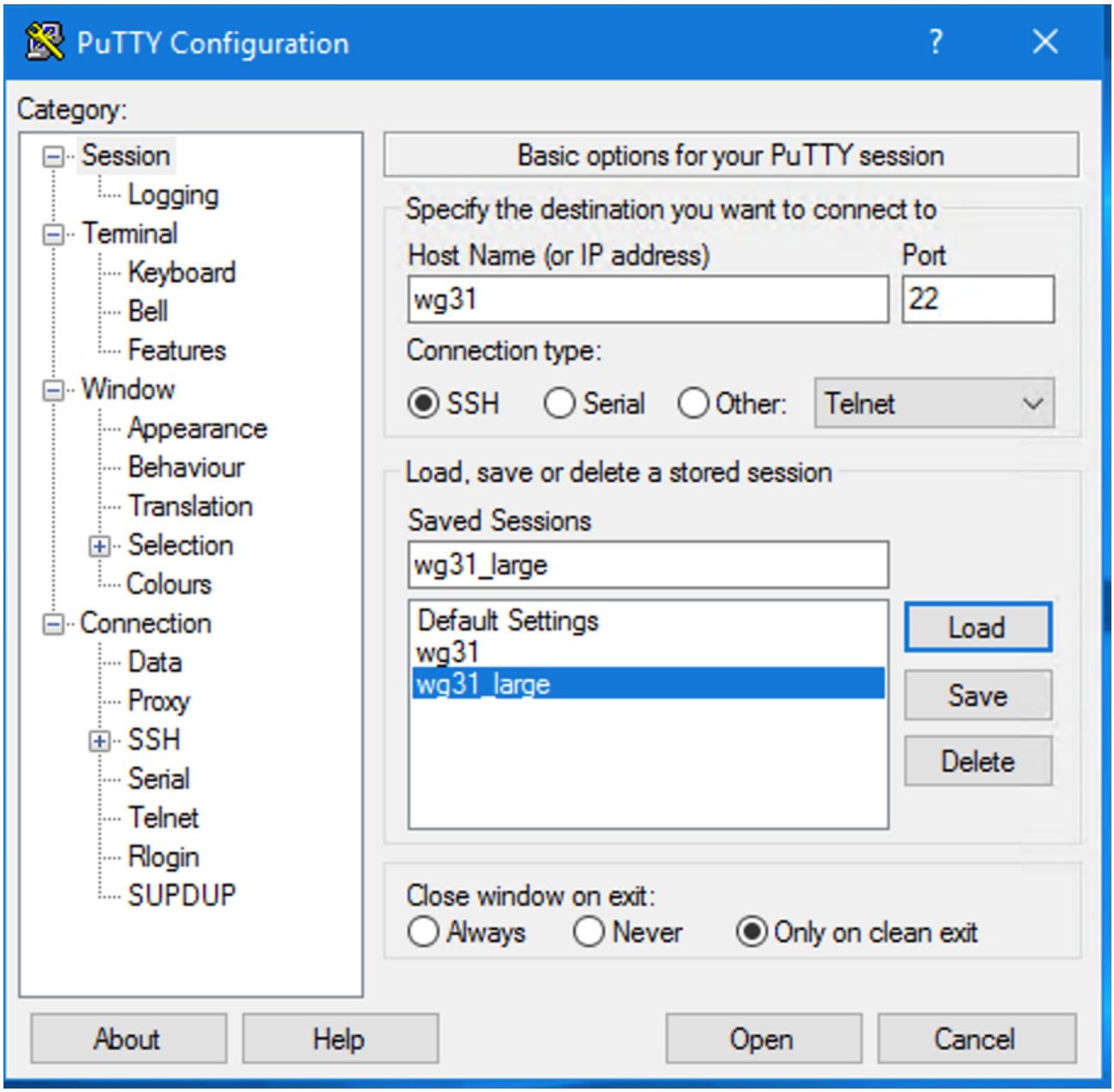


Figure 3: putty_wg31

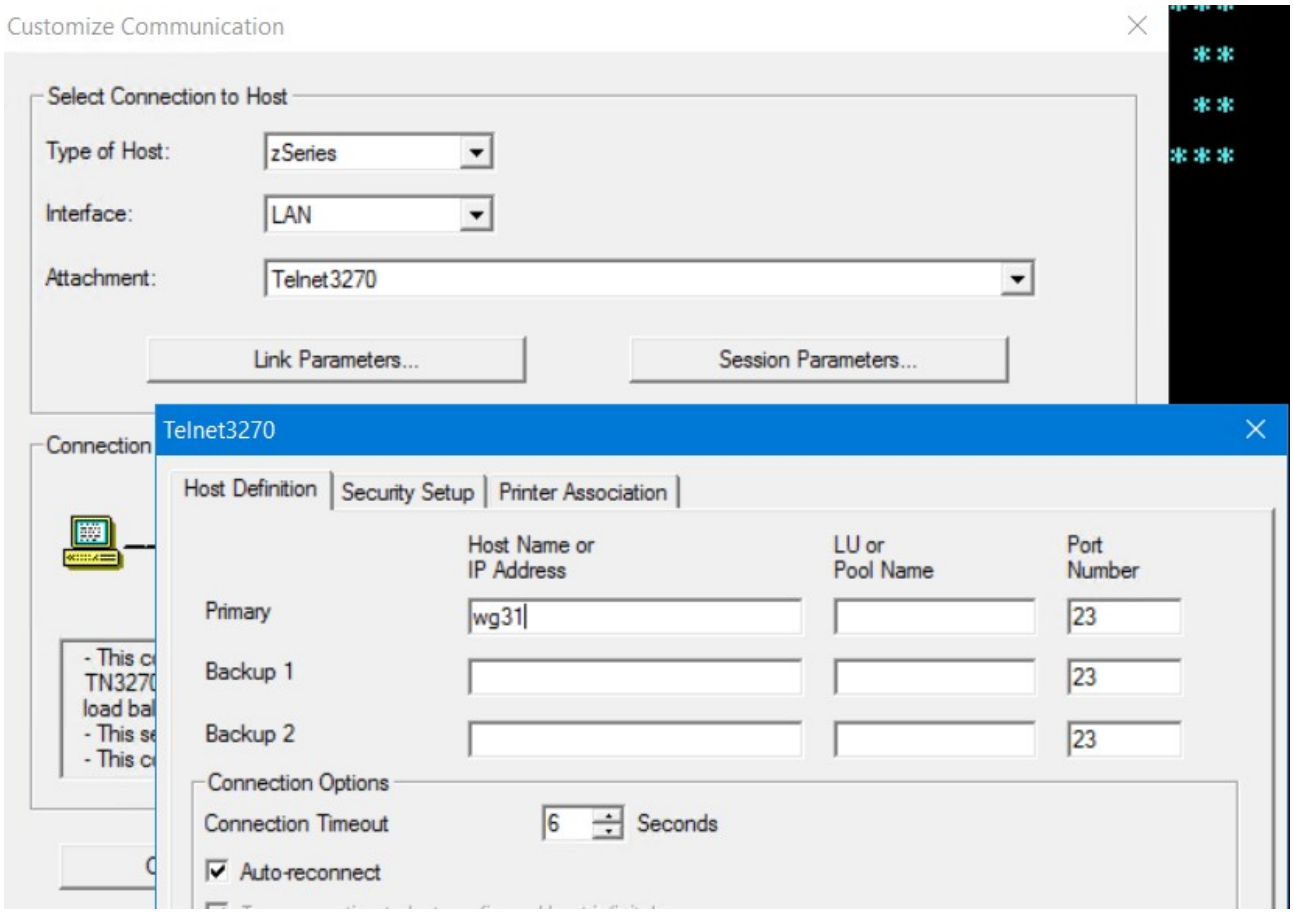


Figure 4: pcomm_wg31

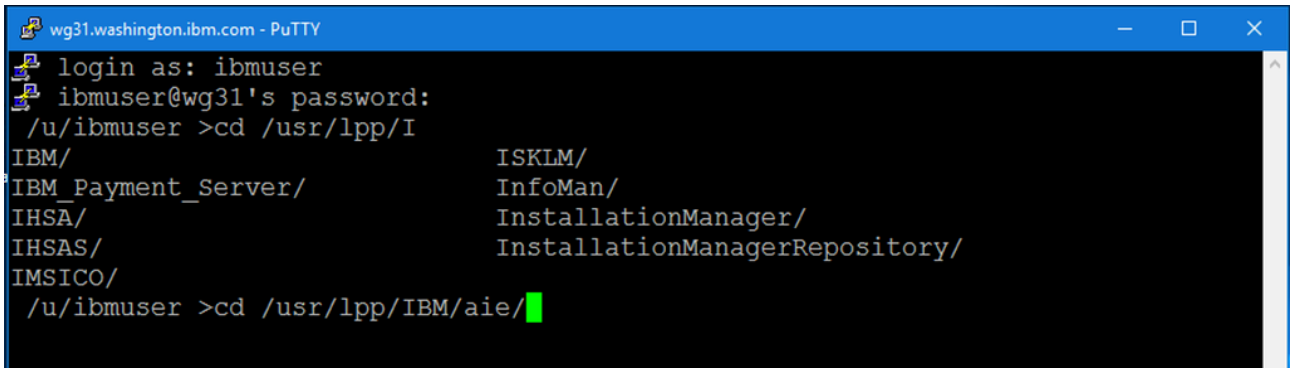


Figure 5: Login using PuTTY

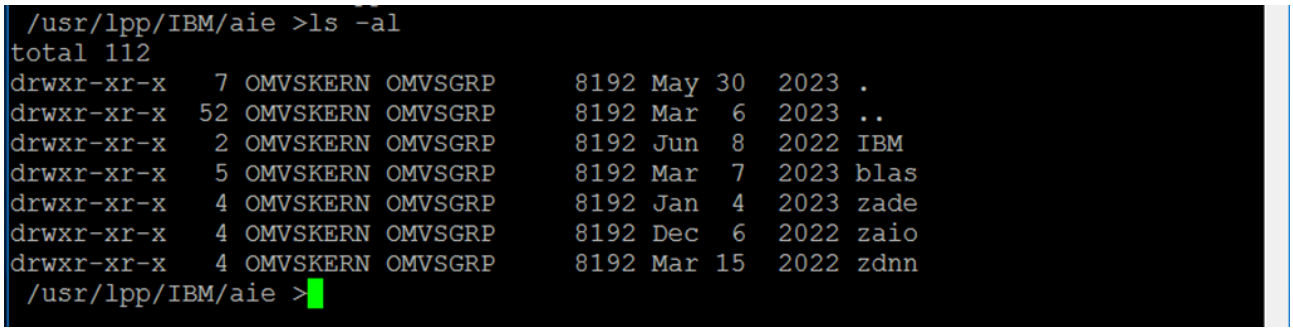


Figure 6: List path with libraries

```
 /usr/lpp/IBM/aie >cd /usr/lpp/IBM/db2sqldi/v1r1
 /usr/lpp/IBM/db2sqldi/v1r1 >ls -al
total 176
drwxr-xr-x   7 OMVSKERN SYS1        8192 May 10  2023 .
drwxr-xr-x   3 OMVSKERN OMVSGRP     8192 Mar  6  2023 ..
drwxr-xr-x   2 OMVSKERN SYS1        8192 Mar  6  2023 IBM
-rw-r--r--   2 OMVSKERN SYS1       17940 Mar  6  2023 NOTICE
-rw-r--r--   2 OMVSKERN SYS1         203 May 10  2023 README
drwxr-xr-x  13 OMVSKERN SYS1        8192 May 19 08:20 spark24x
drwxr-xr-x   6 OMVSKERN SYS1        8192 Jan 27  2023 sql-data-insights
drwxr-xr-x   3 OMVSKERN SYS1        8192 Jan  9  2023 templates
drwxr-xr-x   3 OMVSKERN SYS1        8192 Jan  9  2023 tools
 /usr/lpp/IBM/db2sqldi/v1r1 >
```

Figure 7: Verify the SQLDI libraries are mounted

2. Prepare the SQLDI Administration Userid and Group

There are two parts to this task. The first pertaining to RACF profiles, the second pertaining to USS environment variables.

In a nutshell, you need to setup the following: 1. A RACF userid with an omvs segment to be the SQLDI instance owner. 2. which has generous CPU and Memory limits to reflect the fact that model training might take some time. 3. which is a member of a RACF group named SQLDIGRP. 4. and has USS environment settings that include PATH and LIBPATH values to link to all the Z AI libraries and the Deep Learning Compiler.

Note: Db2 permissions The SQLDI instance owner itself does not need Db2 permissions. The userid that logs onto SQLDI via the Web UI will need to be a member of SQLDIGRP **and** will also need Db2 privileges.

We will use this logistical planning matter as a basis for problem determination later on on the HOL.

2.1 RACF User Profiles

Decide on a userid that will be the SQLDI owner within USS.

You will create **AIDBADM** user:

TASK

The JCL that was used to define RACF userid is found in `IBMUSER.SDISETUP(SDIUSCRT)` . It is standard RACF user profile, with a TSO signon and an omvs id.

Submit the JCL:

TASK

If you want to make additional RACF userids able to operate SQLDI, those users would also need similar customisation as the following steps for AIDBADM.

With the TSO command `tso lu aidbadm omvs` you can display the RACF user profile, or you can go using the panels:

- ISPF main panel

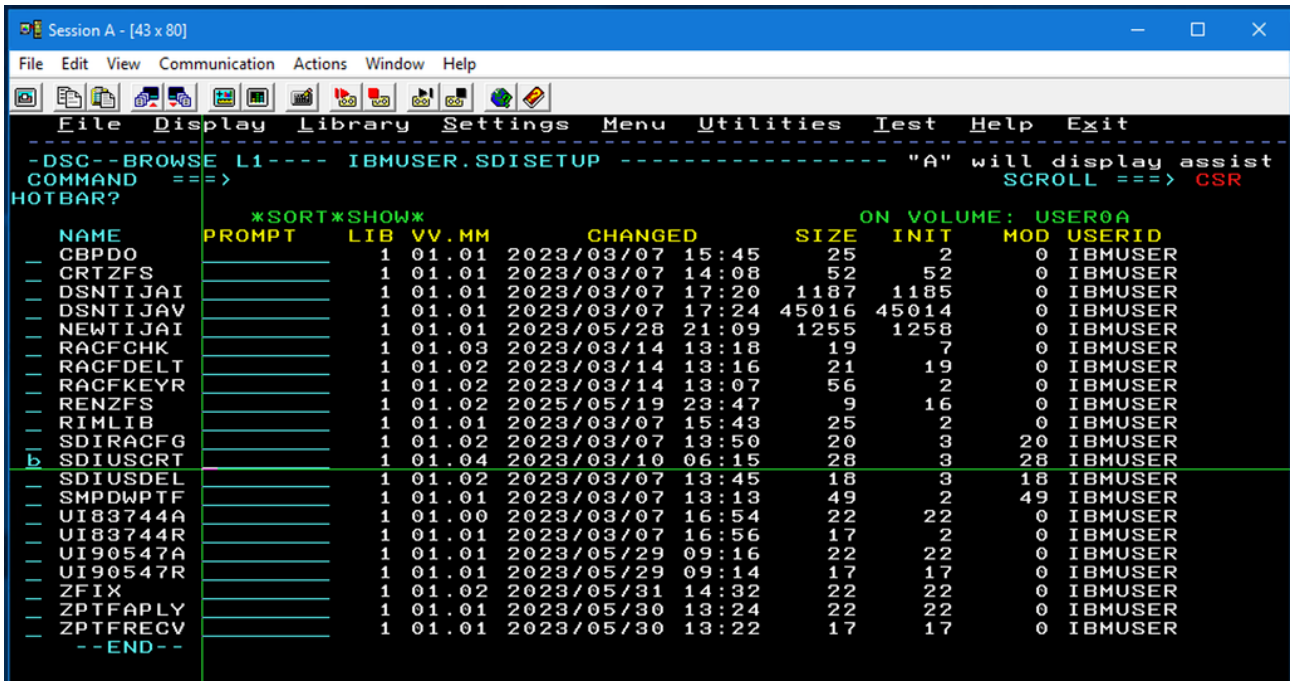


Figure 8: IBMUSER.SDISETUP

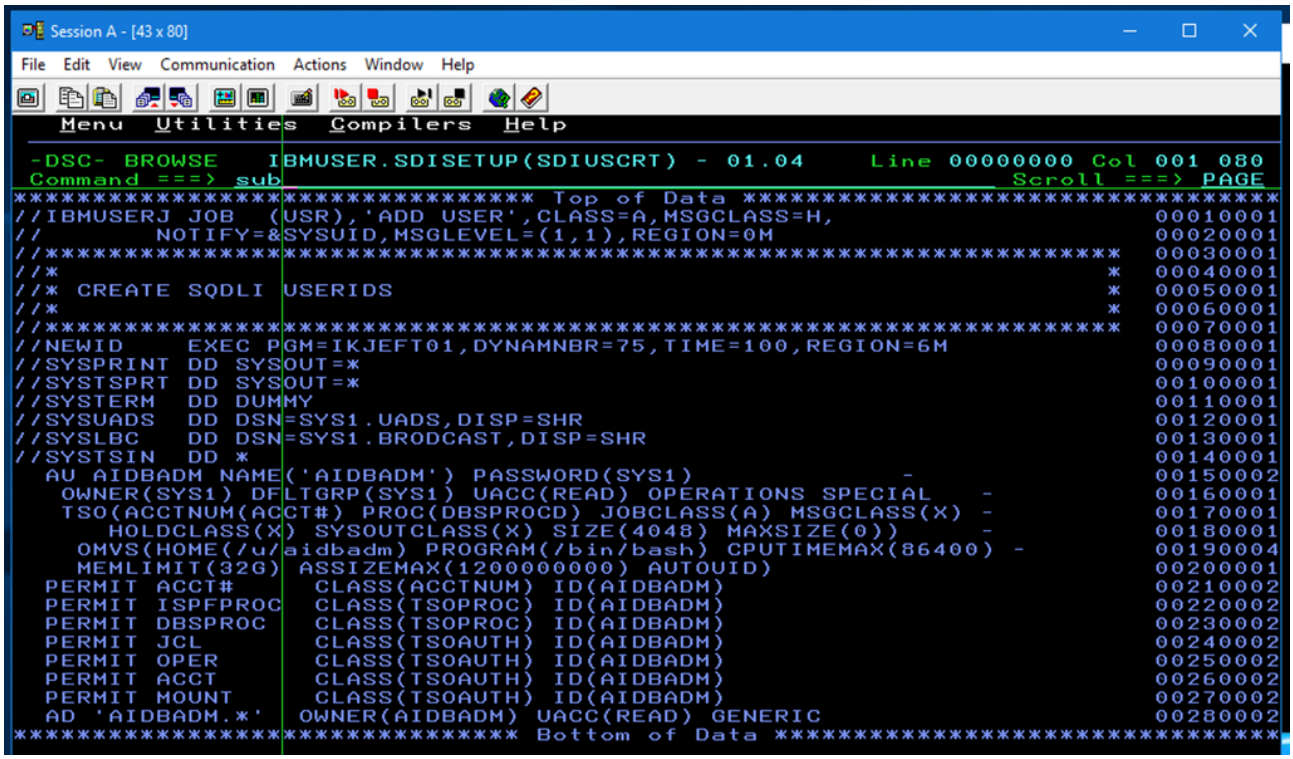
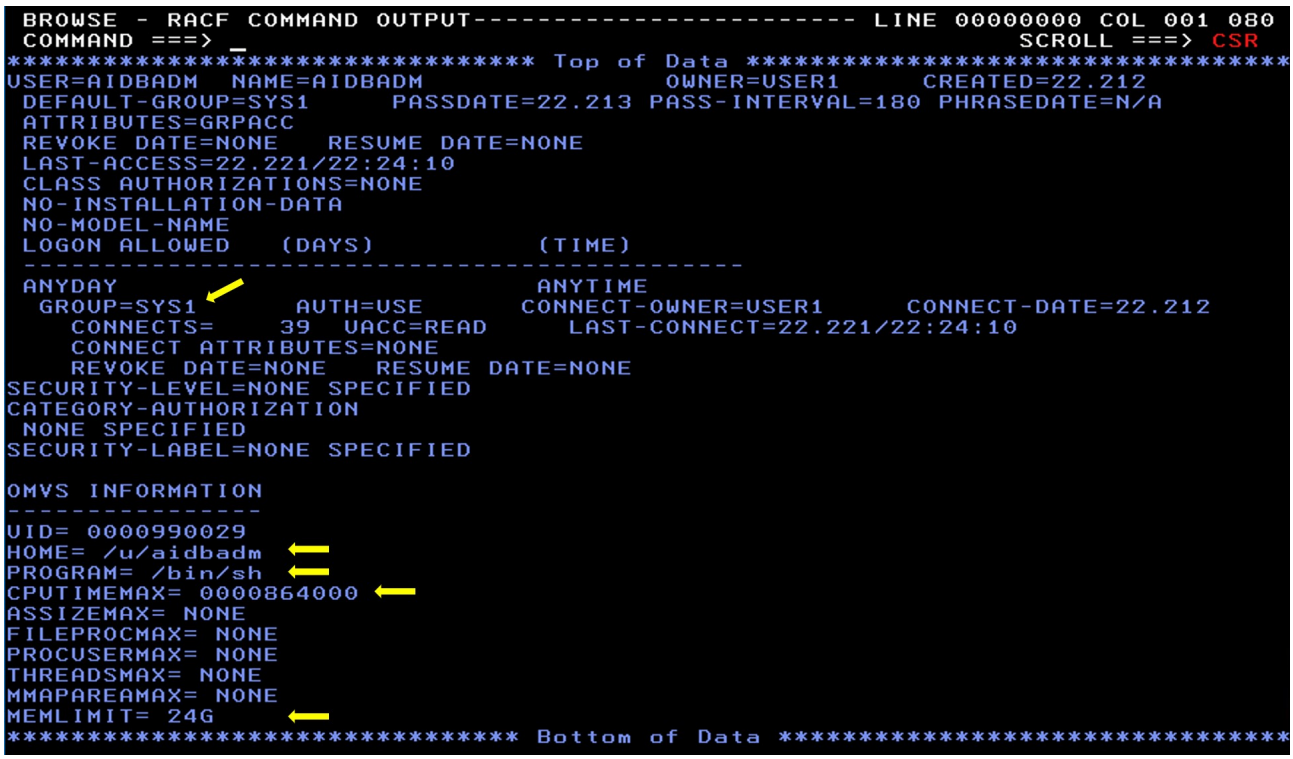


Figure 9: Submit member `SDIUSCRT`

- m.3 (for RACF)
- 4 (for user profiles)
- D - AIDBADM (to display the user profile for AIDBADM)
- s - OMVS (to include the omvs segment details 0

If the lab has been reset correctly, AIDBADM will be a member of the RACF Group 'SYS1', and will have an omvs segment with various omvs properties set.



Check the RACF profiles for user AIDBADM.

TASK

For each of the users, they need an OMVS segment with certain properties specified. We want to see the following values set for the SQLDI user.

- CPUTIMEMAX 864000 (to avoid the risk of timeouts during long model training tasks)
- MEMLIMIT 4GB minimum (because SQLDI and Spark need sufficient memory)
- PROGRAM /bin/sh (or change it to /bin/bash if you prefer that as a default)
- HOME /u/aidbadm (to follow the standard convention for the home directory of a user)

If the OMVS properties needs to be amended, go to RACF User Profiles (ISPF M.3.4) and select "2" to change the user profile of AIDBADM

```
EDIT       AIZ.AIDB0211.HOLFILES(ADDUSER) - 01.02          Columns 00001 00072
                      RACF - USER PROFILE SERVICES          PROFILE DISPLAYED
OPTION ===> 2_

SELECT ONE OF THE FOLLOWING:

         1     ADD           Add a user profile
         2     CHANGE        Change a user profile
         3     DELETE        Delete a user profile
         4     PASSWORD      Change your own password and related information
         5     AUDIT         Monitor user activity (Auditors only)

    D or 8     DISPLAY       Display profile contents
    S or 9     SEARCH        Search the RACF data base for profiles

ENTER THE FOLLOWING INFORMATION:

    USER      ===> AIDBADM     Userid
```

Specify that you want to change optional features

```
                          RACF - CHANGE USER AIDBADM
COMMAND ===>

    OWNER              _________  Userid or group name
    USER NAME          ____________________
    DEFAULT GROUP      _________  Group name

    _   Change PASSWORD related information
    s   Add or Change OPTIONAL information

TO ASSIGN A USER ATTRIBUTE, ENTER YES, TO CANCEL, ENTER NO

    ___   GROUP ACCESS            ___   SPECIAL
    ___   ADSP                    ___   OPERATIONS
    ___   OIDCARD                 ___   AUDITOR
    ___   NO-PASSWORD             ___   ROAUDIT
                                  ___   RESTRICTED

CHANGE OR DELETE THE MODEL PROFILE USED FOR USER DATA SETS (OPTIONAL):

    NEW MODEL  ____________________________________________________
    DELETE     ___  YES if no model is to be used
```

Select omvs

```
                          RACF - CHANGE USER AIDBADM
COMMAND ===>

To add or change the following information, enter any character.

    _ CLASS AUTHORITY                    _ KERB PARAMETERS
    _ INSTALLATION DATA                  _ LDAP PROXY PARAMETERS
    _ SECURITY LEVEL or CATEGORIES       _ ENTERPRISE IDENTITY MAPPING
    _ SECURITY LABEL                     _ CSDATA PARAMETERS
    _ LOGON RESTRICTIONS                 _ MFA PARAMETERS
    _ NATIONAL LANGUAGES
    _ DFP PARAMETERS
    _ TSO PARAMETERS
    _ OPERPARM PARAMETERS
    _ CICS PARAMETERS
    _ WORK ATTRIBUTES
    s OMVS PARAMETERS
    _ NETVIEW PARAMETERS
    _ DCE PARAMETERS
    _ OVM PARAMETERS
    _ LNOTES PARAMETERS
    _ NDS PARAMETERS
```

Edit CPUMAXTIME and MEMLIMIT to meet the crieria

```
                        RACF - CHANGE USER AIDBADM
                        OMVS PARAMETERS
COMMAND ===>

 Delete ALL OMVS information       (NOOMVS)  ___     Enter YES to DELETE

    -- OR --

 Choose to CHANGE or DELETE, then press ENTER.
                                                                  More:    -

 Specify CPU Time                   (CPUTIMEMAX)  86400_____   7 - 2147483647
 Delete  CPU Time                 (NOCPUTIMEMAX)  _            Enter any character

 Specify Address Space Size         (ASSIZEMAX)  ___________   10485760 -
                                                               2147483647
 Delete  Address Space Size       (NOASSIZEMAX)  _            Enter any character

 Specify Files Per Process        (FILEPROCMAX)  _______      3 - 524287
 Delete  Files Per Process      (NOFILEPROCMAX)  _            Enter any character

 Specify Processes Per UID        (PROCUSERMAX)  ______       3 - 32767
 Delete  Processes Per UID      (NOPROCUSERMAX)  _            Enter any character

 Specify Threads Per Process      (THREADSMAX)  _______      0 - 100000
 Delete  Threads Per Process    (NOTHREADSMAX)  _            Enter any character

 Specify Memory Map Area Size     (MMAPAREAMAX)  _________    1 - 16777216
 Delete  Memory Map Area Size   (NOMMAPAREAMAX)  _            Enter any character

 Specify Non-Shared Memory Size      (MEMLIMIT)  24_______  G Multiplier
 Delete  Non-Shared Memory Size    (NOMEMLIMIT)  _            Enter any character

 Specify Shared Memory Size          (SHMEMMAX)  ________  _ Multiplier
 Delete  Shared Memory Size        (NOSHMEMMAX)  _            Enter any character
```

And verify the changes

```
 BROWSE - RACF COMMAND OUTPUT------------------------ LINE 00000000 COL 001 080
 COMMAND ===> _                                        SCROLL ===> CSR
******************************** Top of Data ********************************
USER=AIDBADM  NAME=AIDBADM                OWNER=USER1     CREATED=22.212
 DEFAULT-GROUP=SYS1     PASSDATE=00.000 PASS-INTERVAL=180 PHRASEDATE=N/A
 ATTRIBUTES=NONE
 REVOKE DATE=NONE   RESUME DATE=NONE
 LAST-ACCESS=UNKNOWN
 CLASS AUTHORIZATIONS=NONE
 NO-INSTALLATION-DATA
 NO-MODEL-NAME
 LOGON ALLOWED    (DAYS)          (TIME)
 ---------------------------------------------
 ANYDAY                           ANYTIME
  GROUP=SYS1       AUTH=USE      CONNECT-OWNER=USER1     CONNECT-DATE=22.212
    CONNECTS=     00  UACC=READ      LAST-CONNECT=UNKNOWN
    CONNECT ATTRIBUTES=NONE
    REVOKE DATE=NONE   RESUME DATE=NONE
SECURITY-LEVEL=NONE SPECIFIED
CATEGORY-AUTHORIZATION
 NONE SPECIFIED
SECURITY-LABEL=NONE SPECIFIED

OMVS INFORMATION
----------------
UID= 0000990029
HOME= /u/aidbadm
PROGRAM= /bin/sh
CPUTIMEMAX= 0000086400
ASSIZEMAX= NONE
FILEPROCMAX= NONE
PROCUSERMAX= NONE
THREADSMAX= NONE
MMAPAREAMAX= NONE
MEMLIMIT= 24G
****************************** Bottom of Data ******************************
```

2.2 RACF Group Profiles

A RACF Group profile with the specific name "SQLDIGRP" is required for SQLDI, and userids that invoke SQLDI must be added into that group. You need to create the "SQLDIGRP" group and connect user "AIDBADM" to it. You can do this in any one of the following ways:

1. using the RACF Panels (ISPF M.3)
2. Using TSO commands below from ISPF Option 6.

3. Customising and Submitting the Job in `IBMUSER.SDISETUP(SDIRACFG)` illustrated below

If you choose the third option, this is the JCL that you must customize and submit.

```
//IBMUSERJ  JOB (FB3),'IBMUSER',NOTIFY=&SYSUID,
// MSGCLASS=H,CLASS=A,MSGLEVEL=(1,1),
//          REGION=0M,COND=(4,LT)
//S1        EXEC PGM=IKJEFT01

//SYSTSPRT DD SYSOUT=*

//SYSPRINT DD SYSOUT=*

//SYSTSIN  DD *

ADDGROUP SQLDIGRP OMVS(AUTOGID) OWNER(IBMUSER)

CONNECT (AIDBADM) GROUP (SQLDIGRP) OWNER(IBMUSER)

CONNECT (IBMUSER) GROUP (SQLDIGRP) OWNER(IBMUSER)

SETROPTS RACLIST(FACILITY) REFRESH

/*
 ↪
```

And verify that aidbadm is now a member of group SQLDIGRP

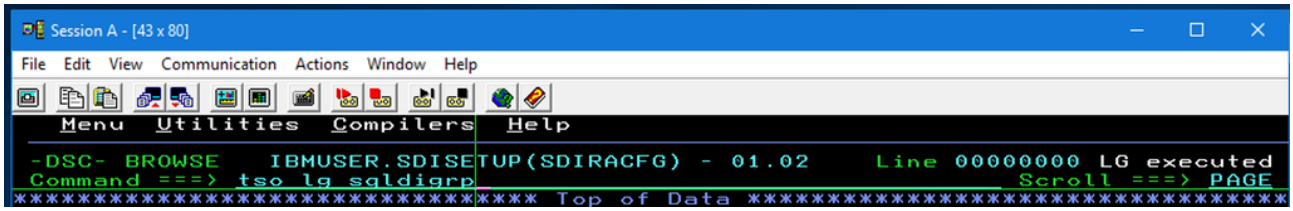


Figure 10: Command

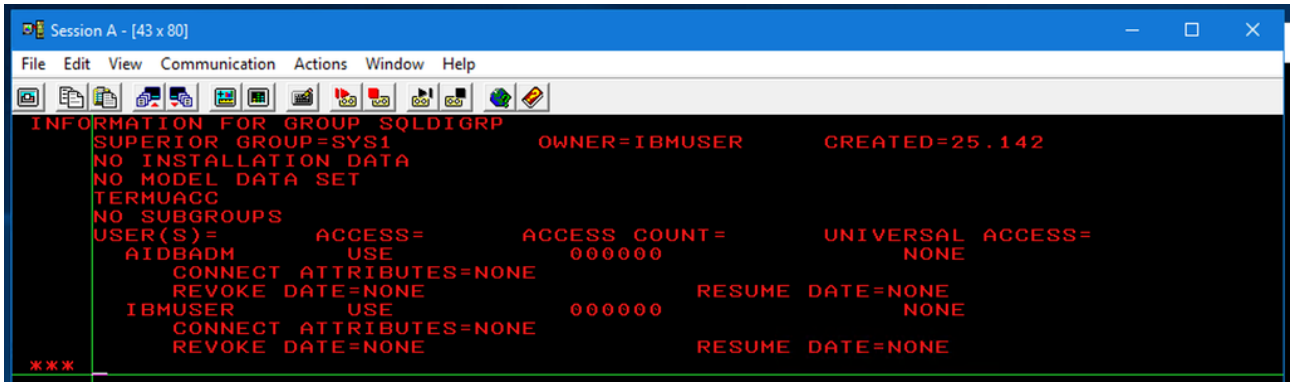


Figure 11: Command result

2.3 USS Environment Variables

The RACF user profile for AIDBADM has been checked for having an omvs segment, and for the required properties to run SQLDI.

The environment variables for a userid operating in USS are a mixture of environment variables set at a system level, and environment variables set for a specific useric using the `.profile`.

The **aidbadm** user needs to define PATH and LIBPATH environment variables so that all the required executables can be invoked at runtime.

TASK

Open a terminal session into USS (e.g. using putty) and **logon as ibmuser**. You should find yourself in the home directory for the ibmuser user.

Now, list all the files in your home directory with the `ls -al` command. (Files beginning with `.` are hidden unless you specify `-al`.)

You first need to copy the .profile.aidbadm file prepared for you in the aidbadm user home directory.

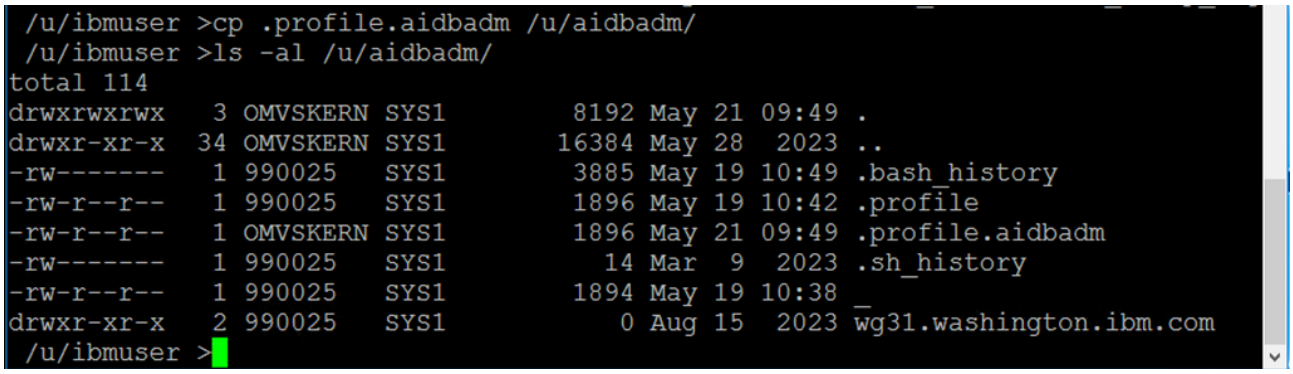


Figure 12: aidbadm user path in USS

(**Note:** Contents may differ from your screen)

```
 /u/ibmuser >ls -al
total 192
drwxr-xr-x   3 AIDBADM  SYS1        8192 Jul 28 02:20 .
drwxr-xr-x  40 OMVSKERN SYS1       16384 Jan 25  2022 ..
-rw-------   1 AIDBADM  SYS1        6312 Jul 27 06:35 .bash_history
-rwxr-xr-x   1 AIDBADM  SYS1        2891 Jul 26 23:58 .profile
-rwxr-xr-x   1 AIDBADM  SYS1        2891 Jul 26 23:58 .profile.aidbadm
-rw-------   1 AIDBADM  SYS1        1654 Jul 28 02:20 .sh_history
```

You then copy the .profile.aidbadm as .profile in aidbadm user home directory and change ownership of the new file to aidbadm user.

```
 /u/ibmuser >cp .profile.aidbadm /u/aidbadm/.profile
 /u/ibmuser >chown aidbadm /u/aidbadm/.profile
```

You can easily view the contents of the new .profile with the `cat` command as follows: (**NOTE:** Contents in your profile may differ form this output)

```
 /u/ibmuser >cd /u/aidbadm
 /u/aidbadm >cat .profile
export HOST=$(uname -n)
export PS1=' ${PWD} >'
export NET_IP=`host $HOST | grep addresses | awk ' { print \$5 } ' `
export LANG=En_US
```

```
export TERM=xterm
set -o vi
export _BPXK_AUTOCVT=ON
export _BPX_SHAREAS=NO
_BPXK_AUTOCVT=ON

# PATH -
PATH=".:${HOME}/bin:/usr/sbin:/usr/bin:${PATH}:/usr/local/bin:/usr/lpp/ldap/sbin:/usr/lp

# Add BASH to PATH
export PATH=/u/user1/tools/bash-4.3.48-2/bin:${PATH}

# use latest java version
if [ -r /usr/lpp/java/J8.0_64 ]
then
  export JAVA_HOME=${JAVA_HOME:-/usr/lpp/java/J8.0_64}
  export PATH="${PATH}:${JAVA_HOME}/bin"
  #Needed by jaydebeapi to find libj9a2e.so
  export
   ↪  LIBPATH=$LIBPATH:${JAVA_HOME}/lib/s390x:${JAVA_HOME}/lib/s390x/classic
fi

if [ -z "$IBM_JAVA_OPTIONS" ]; then
  export IBM_JAVA_OPTIONS="-Dfile.encoding=UTF-8"
else
  if [[ ! "$IBM_JAVA_OPTIONS" == *"-Dfile.encoding=UTF-8"* ]]; then
      export IBM_JAVA_OPTIONS=$IBM_JAVA_OPTIONS:-Dfile.encoding=UTF-8
  fi
fi

if [ -r .envfile ]
then
  echo "execute ENVIRONMENT .envfile "
  . .envfile
fi
```

Some required variables (like `JAVA_HOME`) are already specified, but none of the required SQLDI library paths are defined.

Even though we haven't yet installed the AI libraries and the SQLDI libraries, this HOL is structured to keep all the user profile settings together, and we know exactly what the paths will be.

If you are comfortable with the vi editor, then you can edit the `.profile` inside USS. Most of us would prefer to use the ISPF editor, as shown below.

Open ISPF edit (Option 2) and open the **`/u/aidbadm/.profile`** USS file.

```
# JAVA
export JAVA_HOME=/usr/lpp/java/J8.0_64
export PATH=$PATH:/apps/zospt/bin:/usr/lpp/java/J8.0_64/bin
# ZOAU REQUIREMENTS
```

```
export _BPXK_AUTOCVT=ON
export ZOAU_HOME=/usr/lpp/IBM/zoautil
export PATH=${ZOAU_HOME}/bin:$PATH
# ZOAU MAN PAGE REQS (OPTIONAL)
export MANPATH=${ZOAU_HOME}/docs/%L:$MANPATH
export CLASSPATH=${ZOAU_HOME}/lib/*:${CLASSPATH}
export LIBPATH=${ZOAU_HOME}/lib:${LIBPATH}
# IBM Python - Ansible supported
export PATH=/usr/lpp/IBM/cyp/v3r9/pyz/bin:$PATH
export PYTHONPATH=/usr/lpp/IBM/cyp/v3r9/pyz
export PYTHONPATH=${PYTHONPATH}:${ZOAU_HOME}/lib
# Rocket Ported Git
export _CEE_RUNOPTS='FILETAG(AUTOCVT,AUTOTAG) POSIX(ON)'
export PATH=/usr/lpp/Rocket/rsusr/ported/bin:$PATH
# SQLDI Setup
export SQLDI_INSTALL_DIR=/usr/lpp/IBM/db2sqldi/v1r1
export ZADE_INSTALL_DIR=/usr/lpp/IBM/aie/zade
export ZAIE_INSTALL_DIR=/usr/lpp/IBM/aie
export BLAS_INSTALL_DIR=/usr/lpp/cbclib
export SPARK_HOME=$SQLDI_INSTALL_DIR/spark24x
# SQLDI PATH
PATH=/bin:$PATH
PATH=$SQLDI_INSTALL_DIR/sql-data-insights/bin:$PATH
PATH=$SQLDI_INSTALL_DIR/tools/bin:$PATH
PATH=$ZADE_INSTALL_DIR/bin:$PATH
PATH=$PATH:$JAVA_HOME/bin
export PATH=$PATH
# SQLDI LIBPATH
LIBPATH=/lib:/usr/lib
LIBPATH=$LIBPATH:$JAVA_HOME/bin/classic
LIBPATH=$LIBPATH:$JAVA_HOME/bin/j9vm
LIBPATH=$LIBPATH:$JAVA_HOME/lib/s390x
LIBPATH=$LIBPATH:$SPARK_HOME/lib
LIBPATH=$BLAS_INSTALL_DIR/lib:$LIBPATH
LIBPATH=$ZAIE_INSTALL_DIR/zade/lib:$LIBPATH
LIBPATH=$ZAIE_INSTALL_DIR/zdnn/lib:$LIBPATH
LIBPATH=$ZAIE_INSTALL_DIR/zaio/lib:$LIBPATH
export LIBPATH=$LIBPATH
# SQLDI OTHER
export IBM_JAVA_OPTIONS="-Dfile.encoding=UTF-8"
export _BPXK_AUTOCVT=ON
export _BPX_SHAREAS=NO
export _ENCODE_FILE_NEW=ISO8859-1
export _ENCODE_FILE_EXISTING=UNTAGGED
export _CEE_RUNOPTS="FILETAG(AUTOCVT,AUTOTAG) POSIX(ON)"
export TERM=xterm
alias vi1='vi -W filecodeset=utf-8'
alias vi2='vi -W filecodeset=iso8859-1'
alias ll='ls -ltcpa'
export PS1=' ${PWD} >'
```

Save the file. Then lets check whether the `.profile` is being invoked when the aidbadm user logs on.

TASK

Open a new putty session, and logon as aidbadm. Then type the command **env** to see all the current environment variables.

Logging on will invoke the `.profile` script. Note that if you are already logged on within USS you need to invoke the `.profile` again to reflect your changes.

```
 /u/aidbadm >env
MAIL=/usr/mail/AIDBADM
ZAIE_INSTALL_DIR=/u/aidb0020/aie
HOST=S0W1
PATH=/u/aidb0020/aie/zade/bin:/u/aidb0020/tools/bin:/u/aidb0020/sql-data-insights/bin:/bin:/u/user1/t
ools/bash-4.3.48-2/bin:.:/u/aidbadm/bin:/usr/sbin:/usr/bin:/bin:.:/usr/sbin:/usr/local/bin:/usr/lpp/l
dap/sbin:/usr/lpp/NFS:/usr/lpp/java/J8.0_64/bin:/usr/lpp/java/J8.0_64/bin
SQLDI_INSTALL_DIR=/u/aidb0020
_BPXK_AUTOCVT=ON
SHELL=/bin/sh
IBM_JAVA_OPTIONS=-Dfile.encoding=UTF-8
_ENCODE_FILE_NEW=ISO8859-1
PS1= ${PWD} >
_CEE_RUNOPTS=FILETAG(AUTOCVT,AUTOTAG) POSIX(ON)
ZADE_INSTALL_DIR=/u/aidb0020/aie/zade
_=/bin/env
STEPLIB=none
LOGNAME=AIDBADM
LANG=En_US
LIBPATH=/u/aidb0020/aie/zaio/lib:/u/aidb0020/aie/zdnn/lib:/u/aidb0020/aie/zade/lib:/u/aidb0020/aie/bl
as/lib:/lib:/usr/lib:/usr/lpp/java/J8.0_64/bin/classic:/usr/lpp/java/J8.0_64/bin/j9vm:/usr/lpp/java/J
8.0_64/lib/s390x:/u/aidb0020/spark24x/lib
BLAS_INSTALL_DIR=/u/aidb0020/aie/blas
TERM=xterm
_BPX_SHAREAS=NO
HOME=/u/aidbadm
_ENCODE_FILE_EXISTING=UNTAGGED
SPARK_HOME=/u/aidb0020/spark24x
NET_IP=
JAVA_HOME=/usr/lpp/java/J8.0_64
TZ=CST6CDT
MANPATH=/usr/man/%L
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lib/nls/msg/%L/%N.cat
 /u/aidbadm >
```

Figure 13: env

You might find it easier to check individual environment settings with an **echo** command. For example

```
 /u/aidbadm >
 /u/aidbadm >echo $PATH
/u/aidb0020/aie/zade/bin:/u/aidb0020/tools/bin:/u/aidb0020/sql-data-insights/bin:/bin:/u/user1/tools/
bash-4.3.48-2/bin:.:/u/aidbadm/bin:/usr/sbin:/usr/bin:/bin:.:/usr/sbin:/usr/local/bin:/usr/lpp/ldap/s
bin:/usr/lpp/NFS:/usr/lpp/java/J8.0_64/bin:/usr/lpp/java/J8.0_64/bin
 /u/aidbadm >
```

Figure 14: echo

TASK

Some users with non-english keyboards at the Europe and AP Technical Academy had great difficulty in editing the .profile file using ISPF. If you encounter similar difficulties, please don't

waste your time attempting to overcome such challenges. Instead, you could replace the existing `.profile` file with a pre-customised file.

1. Log on to putty as user **ibmuser**.
2. `cd /u/ibmuser` and then list the files `ls -la`

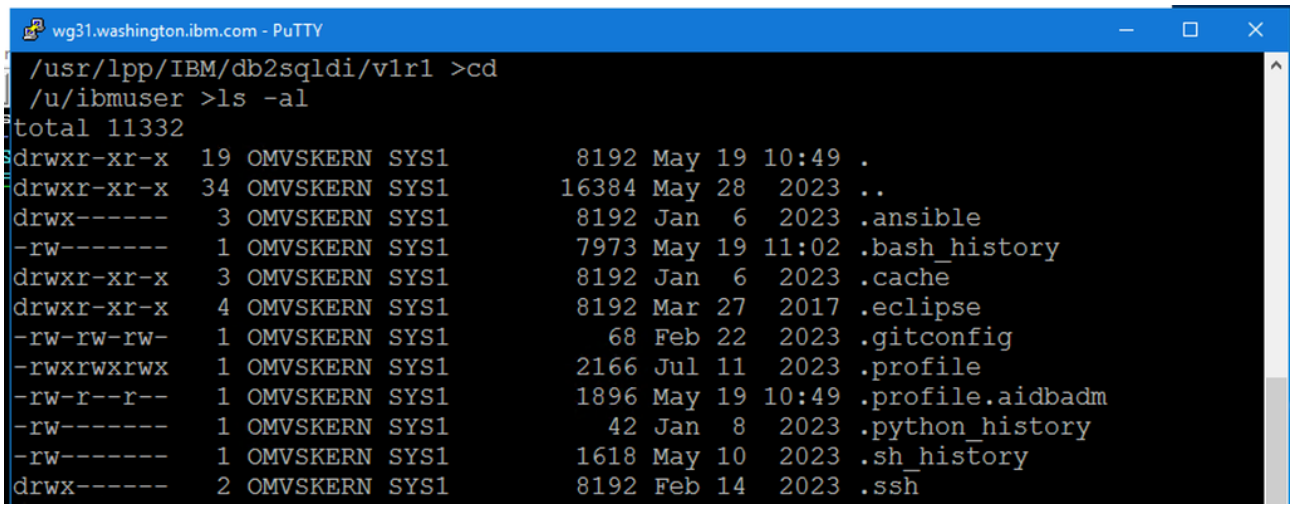


Figure 15: ibmuser path

3. `cp .profile.aidbadm /u/aidbadm/.profile`

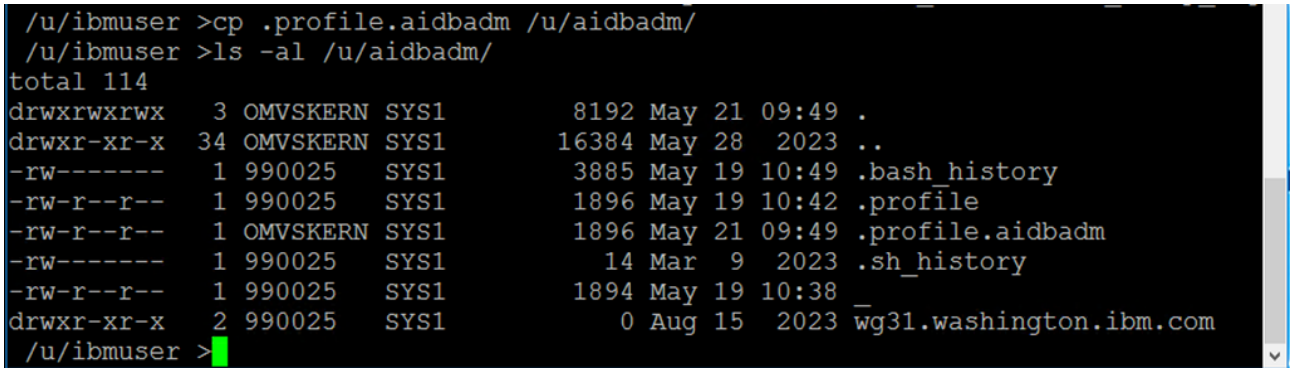


Figure 16: ibmuser path

4. `ls -al /u/aidbadm`

!!! Alternative Task : !!! Edit the `.profile` and check that the following environment variables are set correctly. Type them if needed.

- PATH
- LIBPATH
- ZAIE_INSTALL_DIR
- SQLDI_INSTALL_DIR
- ZADE_INSTALL_DIR
- BLAS_INSTALL_DIR
- JAVA_HOME
- SPARK_HOME

3. Prepare a large ZFS for SQLDI_HOME

The requirements for the zFS are that it will support an `SQLDI_HOME` path over over 100GB (for a realistic small system). The script to create the SQLDI instance checks the ZFS and fails in it is less than 4GB.

TASK

You will need to go to your TSO session and search for `IBMUSER.SDISETUP(CRTZFS)`

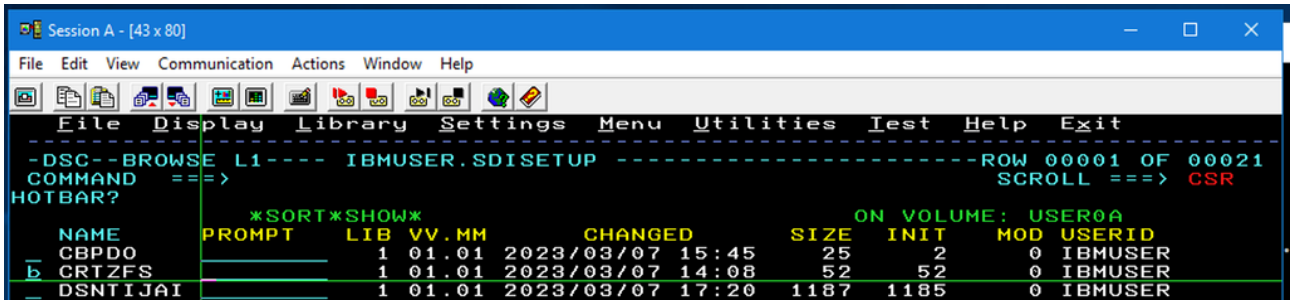


Figure 17: IBMUSER.SDISETUP(CRTZFS)

And then submit the JCL.

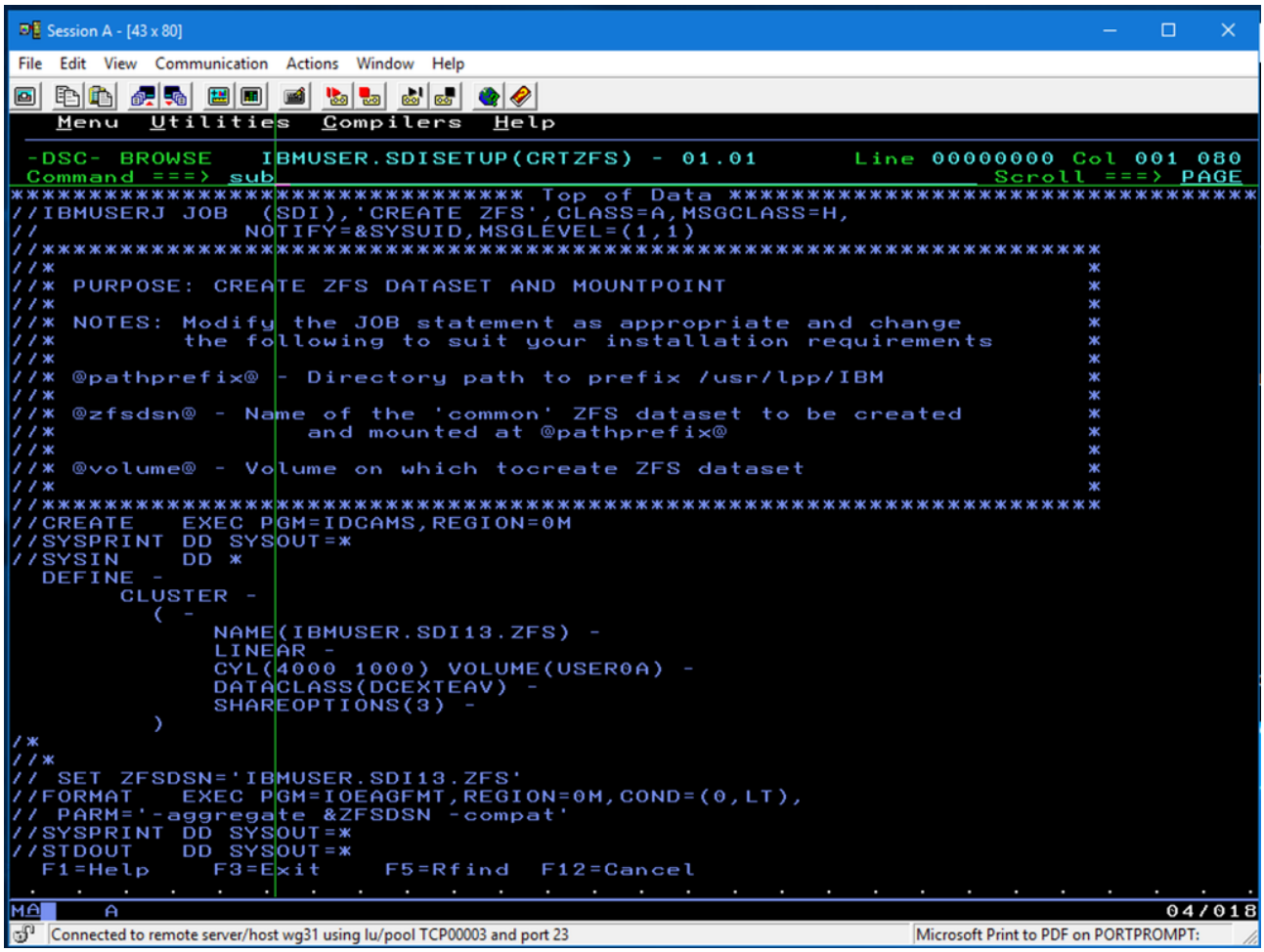


Figure 18: JCL for zFS generation

Wait for the correct execution and result for the JCL

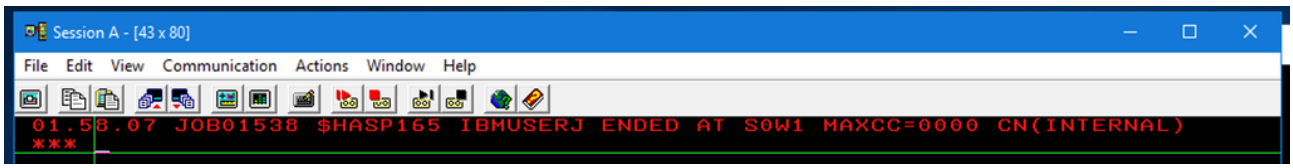


Figure 19: Successful execution

You can check the size of the ZFS in KB with the following command in USS, it has been mounted from the creation JCL. Use the VT100 terminal connected to USS. Command `df -k /u/sqldi13`:

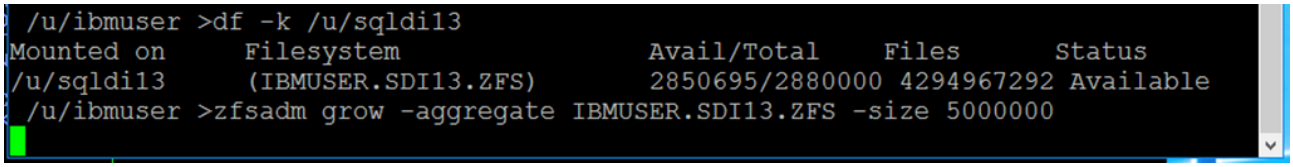


Figure 20: Check size of the zFS

Now grow the zFS to ensure that it is over 4GB in size, use the command `zfsadm grow -aggregate IBMUSER.SDI13.ZFS -size 5000000`:

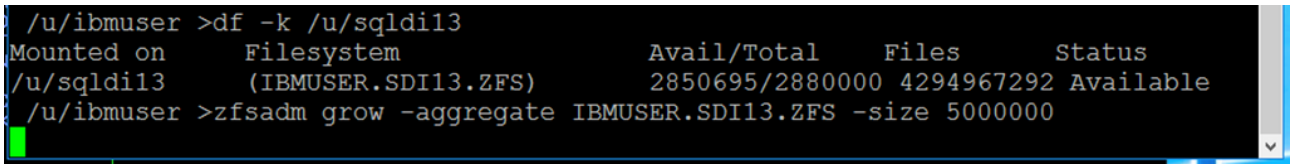


Figure 21: Grow command for zFS

Change `/u/sqldi13` ownership to AIDBADM user. Command: `chown AIDBADM /u/sqldi13`.

Check that it worked. Command: `ls -latr /u/sqldi13/`

---

4. Create SQLDI Pseudo-catalog

TASK

You will need to go to your TSO session and search for `IBMUSER.SDISETUP(DSNTIJAI)`, and execute the JCL.

and wait for the correct return code.

---

5. Prepare a Certificate and Keyring for SQLDI

Authentication for the SQLDI server is achieved by referencing a certificate in a RACF keyring during the SQLDI instance creation.

Additionally, we could setup network encryption rules using RACF certificates and PAGENT rules. z/OS uses Application-Transparent TLS (AT-TLS), so there would be no differnce in the SQLDI setup steps. Encryption is outside the scope of this HOL.

```
 /u/ibmuser >ls -altr /u/sqldi13/
total 32
drwxr-xr-x   34 OMVSKERN SYS1       16384 May 28  2023 ..
drwxr-xr-x    2 AIDBADM  SYS1           0 May 21 09:58 .
 /u/ibmuser >
```

Figure 22: Check ownership change

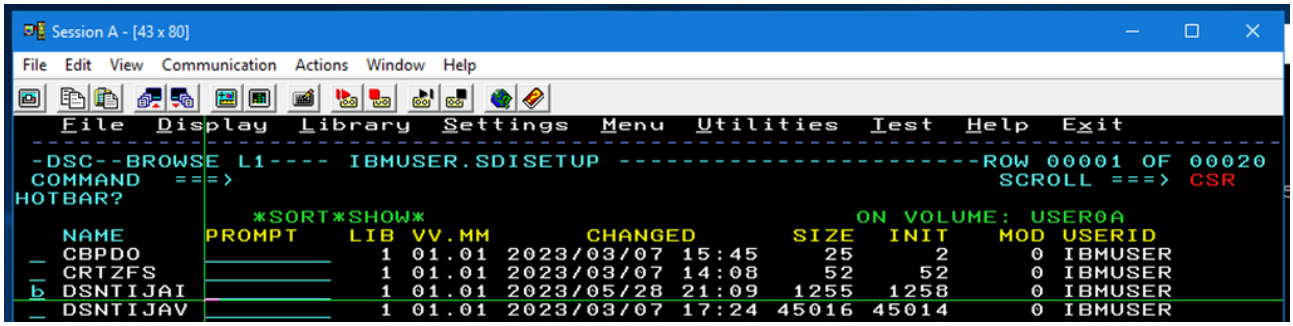


Figure 23: Create pseudo-catalog

A JCL job for creating a keyring containing a self-signed certificate is provided in `IBMUSER.SDISETUP(RACFKEYR)`. Logon to TSO, review and submit this job to create the RACF artefacts. The steps performed by this job are

1. create a keyring
2. create a certificate authority (identified by label WMLZCACert)
3. create a certificate (identified by label `WMLZCert_WMLZID`) and signed by the CA above
4. connect both the user certificate and the CA certificate to the keyring
5. grant permission to list the keyring to aidbadm (and any other user that might want to list it)
6. perform a RACF refresh

TASK

Go to dataset `IBMUSER.SDISETUP(RACFKEYR)`

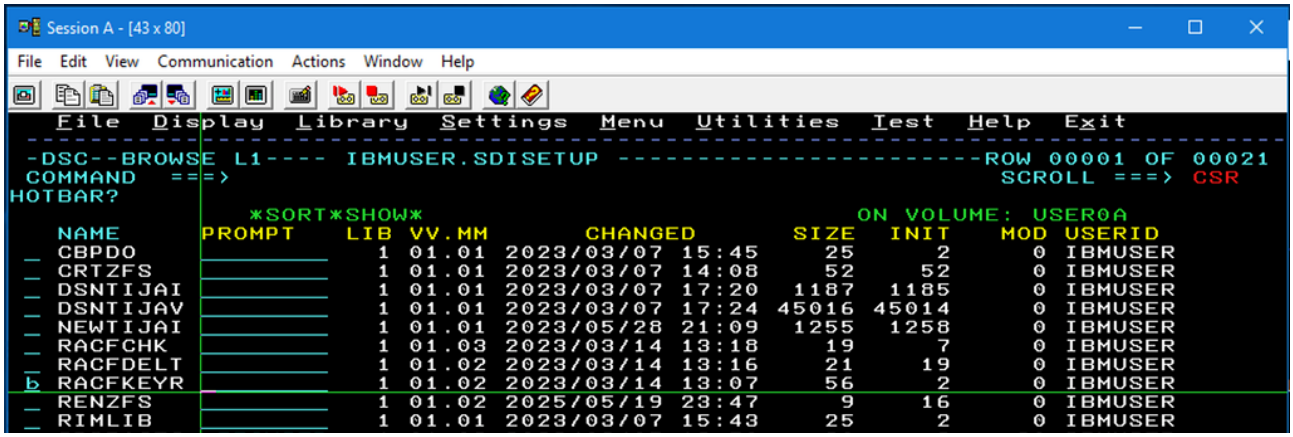


Figure 24: RACFKEYR member

And then submit the job:

You can check the status of the RACF objects by submitting `IBMUSER.SDISETUP(RACFCHK)`:

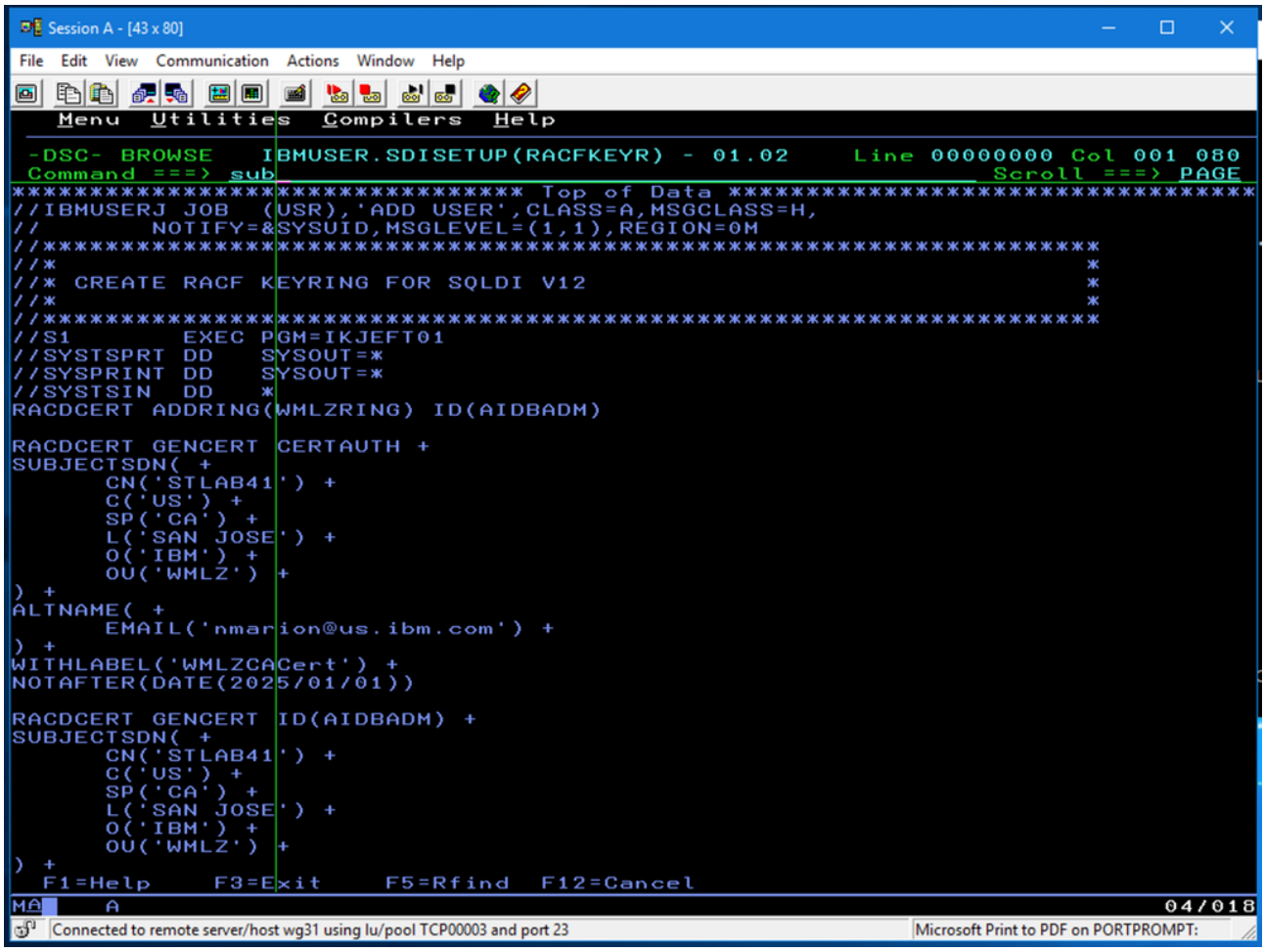


Figure 25: RACFKEYR execution

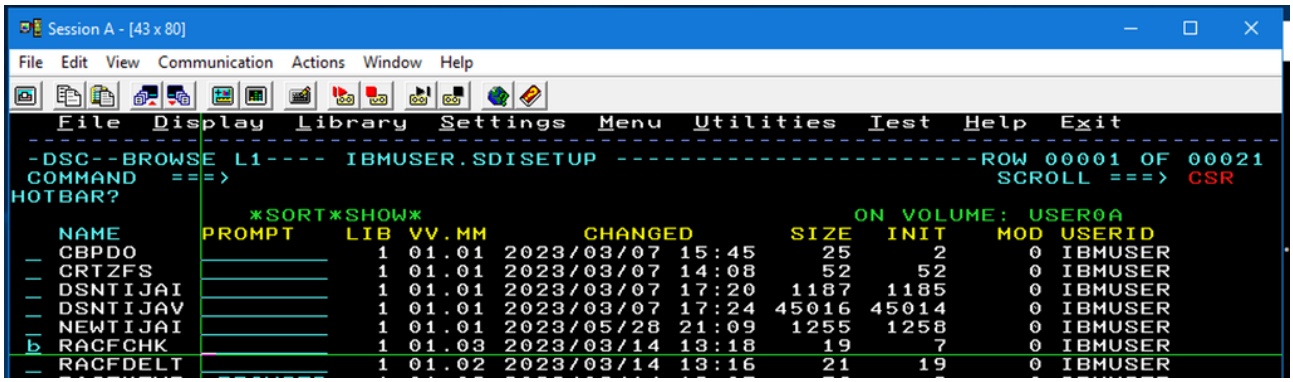


Figure 26: RACFCHK member

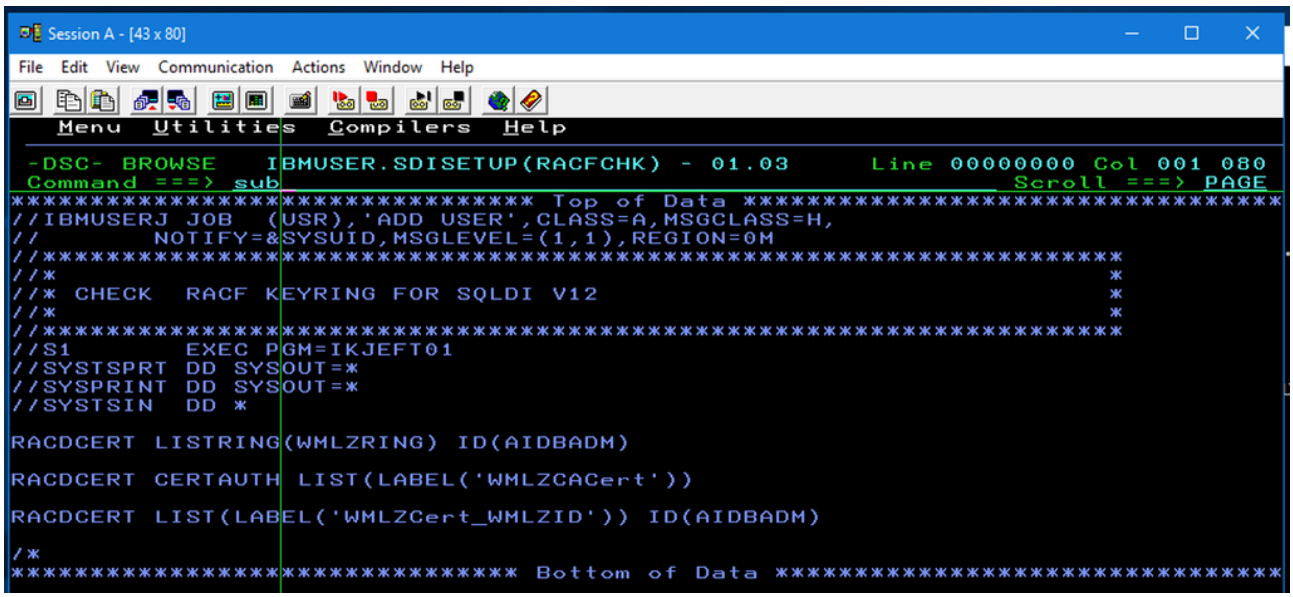


Figure 27: RACFCHK member

The output of the job should look like this:

---

6. Prepare network ports

SQLDI makes use of several TCPIP ports for communication between the various Spark and SQLDI components. You can control the values of all of these ports during the SQLDI instance create process if you need to.

For this HOL environment, all the default ports are free, meaning that you should not suffer port conflicts. However, in a customer environment you should communicate with the z/OS network administrator to check port availability. Commands like `NETSTAT` are available in USS and TSO to check reserved ports.

TASK

Check whether any of the default ports are already assigned. If they are you will need to choose a different free port when you create the SQLDI Server.

The default ports used by SQLDI are documented here SQLDI Pre-Requisites

- SQLDI Web UI on `15001`
- z/OS Spark Master on `7077`
- z/OS Spark Master REST API on `6066`
- z/OS Spark Master UI on `8080`
- z/OS Spark Worker UI on `8081`
- Other Spark ports can be system assigned or manually defined

---

7. Create the SQLDI Server Instance

The installation of SQLDI has placed a a script file (sqldi.sh) in `/u/aidbadm/sql-data-insights/bin`

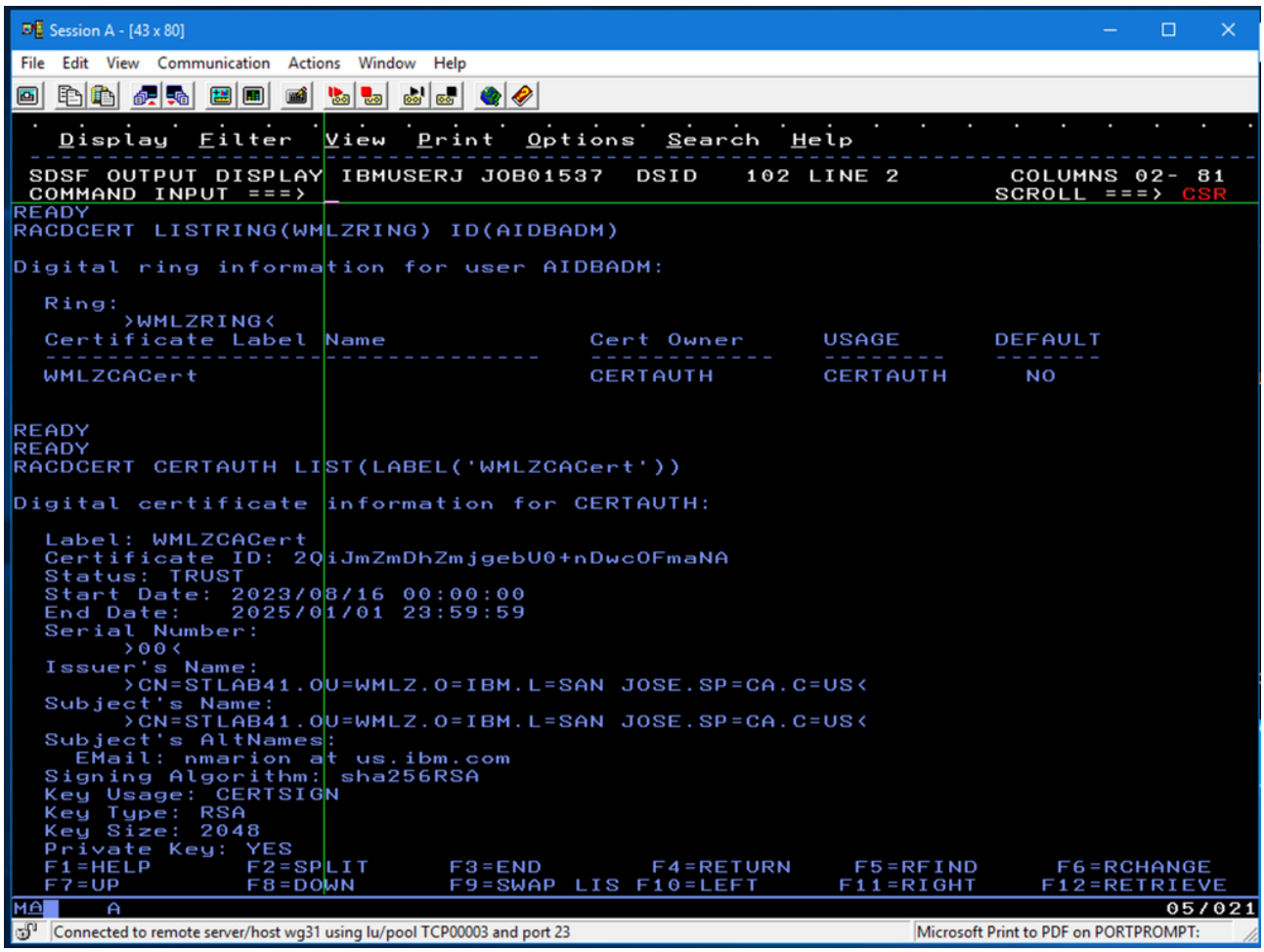


Figure 28: RACFCHK output

Figure 29: Command to see the ports

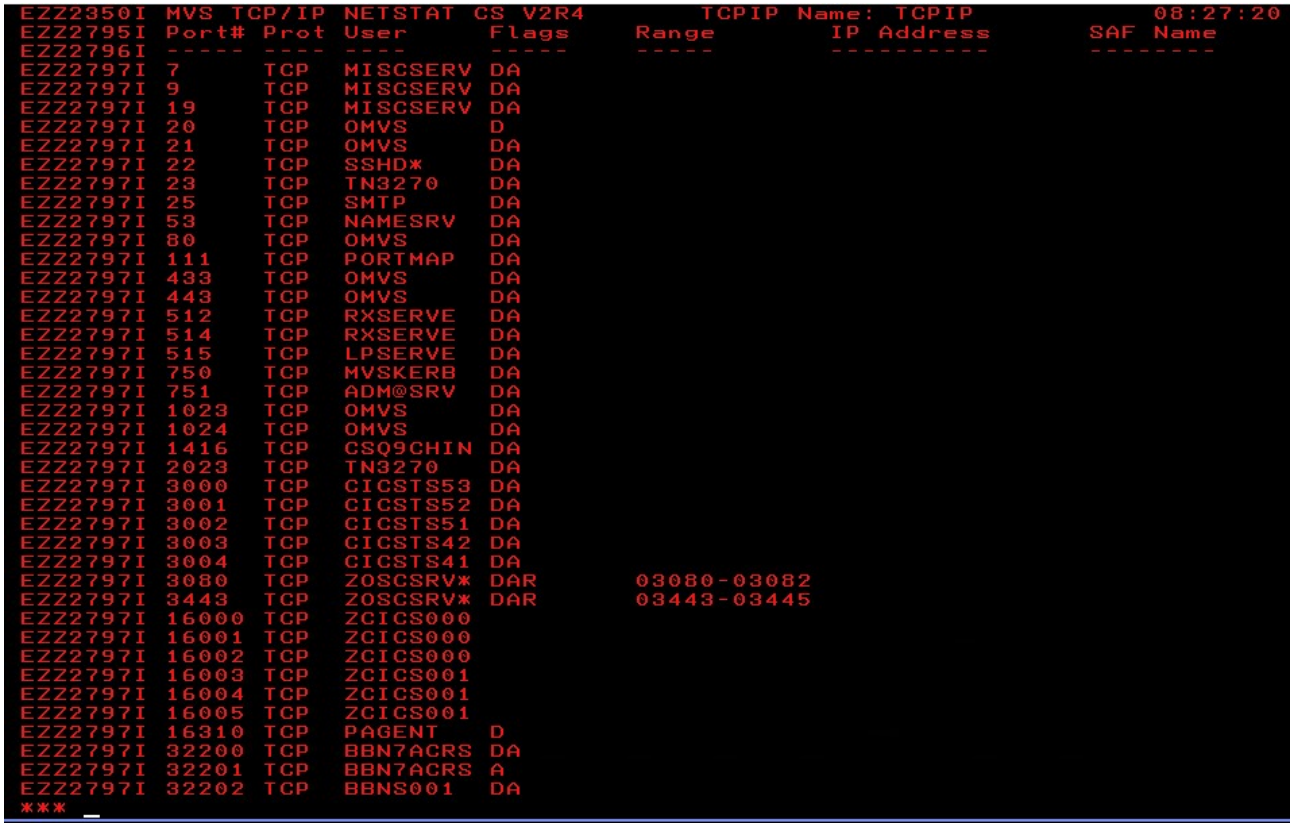


Figure 30: Ports

Assuming that you setup the `PATH` variable correctly (to include `/u/aidbadm/sql-data-insights/bin`) then `sqldi.sh` can be invoked from any path.

Open a putty session to USS, logon as **aidbadm**, and just type command `sqldi.sh` in order to get the command parameters returned to you

```
 /u/aidbadm >sqldi.sh

This script installs, starts, and stops SQL Data Insights. Before running the
 ↪  script, make sure
that you allocate a minimum of 4GB disk space to your zFS file system and meet
 ↪  other system requirements.
In case of an error, resolve the error and then rerun the script.

Usage:
  sqldi.sh [action] [-Xms <value>] [-Xmx <value>]

Action:
  create              Installs the SQL Data Insights application.
  start               Starts the SQL Data Insights application.
  stop                Stops the SQL Data Insights application.
  start_spark         Starts the embedded Spark cluster.
  stop_spark          Stops the embedded Spark cluster.

JVM Options:
  -Xms ''             Specifies the initial memory allocation for JVM in the
   ↪  format of [0-9]*[M,G],
```

```
                              e.g. 512M (Optional).
  -Xmx ''             Specifies the maximum memory allocation for JVM in the
  ↪  format of [0-9]*[M,G],
                              e.g. 1G (Optional).

Examples:
  ./sqldi.sh create
  ./sqldi.sh create -Xms 512M -Xmx 1024M
  ./sqldi.sh start
  ./sqldi.sh stop
  ./sqldi.sh start_spark
  ./sqldi.sh stop_spark

 /u/aidbadm >
```

You should use the bash shell for SQLDI work. This was installed to /u/aidbadm/tools/ when you installed the SQLDI package, and was added to your path when you edited the **.profile** file, so you can enter the bash shell by simply typing **bash** inside your putty USS shell.

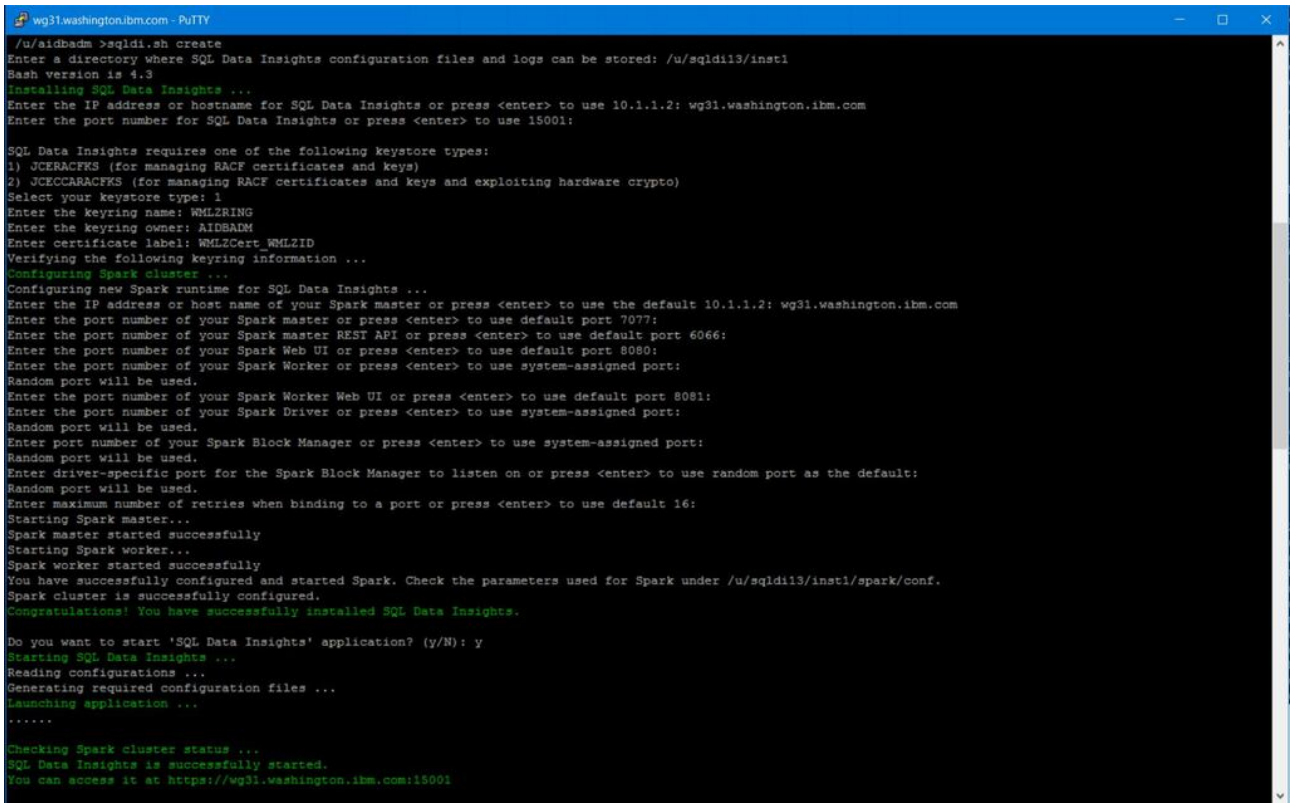


Figure 31: bash

You are now ready to create the SQLDI instance because

1. you know the default ports are available
2. you know the path where you want to install the instance (/u/sqldi13)
3. you know the name of the RACF keyring and the certificate to reference

---

TASK

Notes on TCPIP Addressing to Use. When running the `sqldi.sh` script to create the instance in this lab, you should **always** use the hostname **`wg31.washington.ibm.com`**, and never use the TCPIP Address.

In a customer environment it would be fine to use the TCPIP Address. But the cloning process used to provision the z/OS images has not customised TCPIP address in the z/OS TCPIP stack. The hosts file in the Windows client image has been edited so that both `wg31` and `wg31.washington.ibm.com` point at the actual z/OS image. And within the z/OS USS environment `wg31.washington.ibm.com` points at the actual z/OS image.

Do not be tempted to use the shortened hostname alias in windows (wg31) because this is not defined in USS.

1. Use `wg31.washington.ibm.com` for the **sqldi.sh create** script.
2. Use `wg31.washington.ibm.com` to access it from Windows later on in the HOL.

Please also note that some of the screenshots in this workbook may point to an actual IP address. Please ignore this, and use `hostname wg31.washington.ibm.com` consistently.

Invoke the `sqldi.sh` Script Execute the script, fill in the variables requested and wait until completion.

When you invoke the **sqldi.sh create** script, the dialog should look like the screenshot below. User prompts and responses have been highlighted with yellow arrows.

Note, there are several examples of informational messages not being retrieved from a missing message catalog. At time of writing this intrumentation problem has not been resolved, but the script still succeeds.

You will be prompted for many decisions, as follows.

```
Enter a directory where SQL Data Insights configuration files and logs can be
↪  stored: /u/sqldi13
>>> specify a path underneat /u/aidbadm ( the big ZFS mountpoint)

Enter the IP address or hostname for SQL Data Insights or press <enter> to use
↪  10.1.1.2:
>>> We're using wg31.washington.ibm.com  !!!!!!!!!!!!!!!

Enter the port number for SQL Data Insights or press <enter> to use 15001:
>>> Accept the default port

SQL Data Insights requires one of the following keystore types:
1) JCERACFKS (for managing RACF certificates and keys)
2) JCECCARACFKS (for managing RACF certificates and keys and exploiting
↪  hardware crypto)
Select your keystore type: 1
>>> The keystore type is 1

Enter the keyring name: WMLZRING
>>> Enter the name of the Keyring you created

Enter the keyring owner: AIDBADM
```

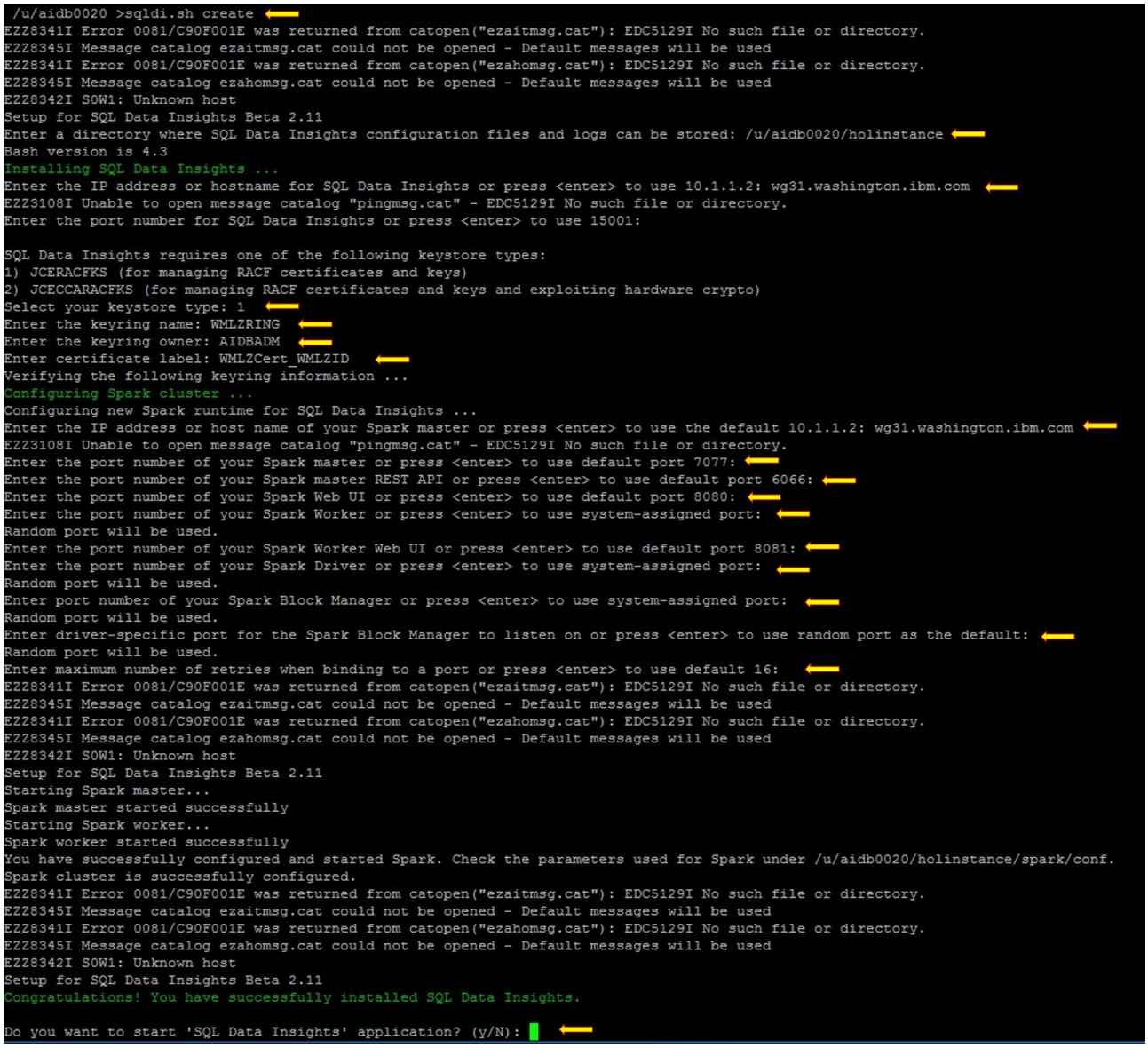


Figure 32: sqldicreate

```
>>> Enter the name of the keyring owner

Enter certificate label: WMLZCert_WMLZID
>>> Enter the label of the Certificate you created. (The user certificate, NOT
 ↪  the CA certificate)

Enter the IP address or host name of your Spark master or press <enter> to use
 ↪  the default 10.1.1.2:
>>> Use wg31.washington.ibm.com !!!!!!!!!!!!!!!

>>> And then specify your chosen ports.

You have successfully configured and started Spark. Check the parameters used
 ↪  for Spark under /u/aidbadm/holinstance/spark/conf.
>>> Remember this location

Do you want to start 'SQL Data Insights' application? (y/N):
>>> No
```

TASK

Run the `sqldi.sh create` from **AIDBADM** user (not IBMUSER). But when prompted enter **N** to avoid starting the server.

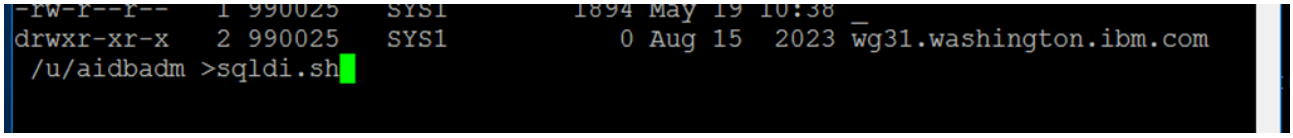


Figure 33: Create instance script

This is the example output in our test system.

Take a moment to review some updates that the `sqldi.sh create` script added to .profile

```
000087 # Generated by SQL Data Insights installation script
000088 export SQLDI_HOME=/u/aidbadm/holinstance
000089
000090
000091 export SPARK_CONF_DIR=/u/aidbadm/holinstance/spark/conf
000092 export SPARK_LOCAL_IP=10.1.1.2
000093 export SPARK_MASTER_PORT=7077
000094
000095 # aliases for SQL DI lifecycle management.
000096 alias start_sqldi="/u/aidbadm/sql-data-insights/bin/sqldi.sh start"
000097 alias stop_sqldi="/u/aidbadm/sql-data-insights/bin/sqldi.sh stop"
000098
000099 alias start_spark="/u/aidbadm/sql-data-insights/bin/sqldi.sh start_spark"
000100 alias stop_spark="/u/aidbadm/sql-data-insights/bin/sqldi.sh stop_spark"
000101
****** ***************************** Bottom of Data ****************************
```

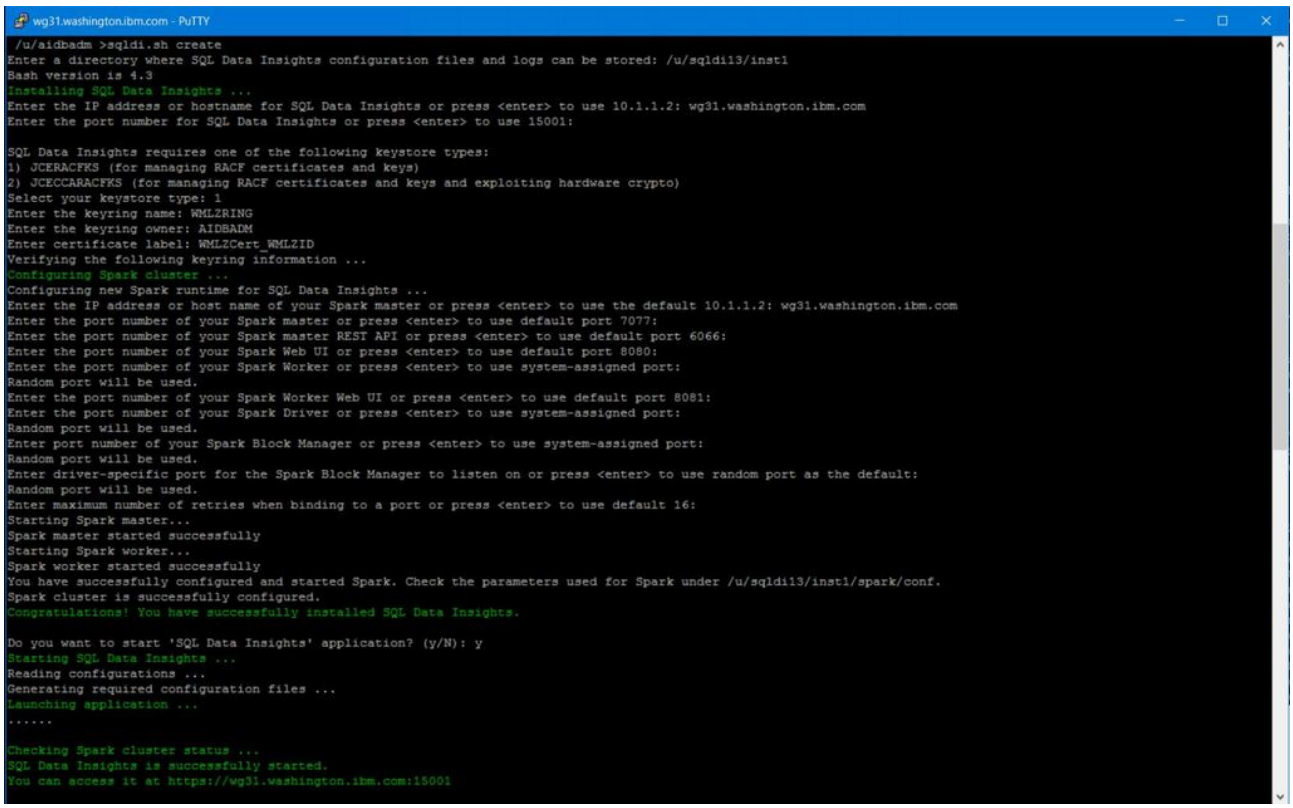


Figure 34: Example input/output

List the processes running under user AIDBADM, by using command `ps -ef`.

You should see the spark Master and Worker nodes using the ports you specified.

And Check the Spark Server by opening your browser on the Spark Web UI port `http://wg31.washington.ibm.com:8080/`. You should see a display like the screenshot below.

At this point, no training jobs will be executing, but you will use this Web UI to check on Spark progress later on.

The sqldi.sh script will automatically check whether spark is running, and start it if it is not running. As you get are gaining familiarity with SQLDI, it is good practice to start Spark and SQLDI separately.

TASK

The procedure to start SQLDI would be

1. `sqldi.sh start_spark`
2. check spark is up and running
3. `sqldi.sh start`

Likewise, the procedure to stop SQLDI would be

1. `sqldi.sh stop`
2. `sqldi.sh stop_spark`

Assuming spark is started, lets start SQLDI itself with the command `sqldi.sh start`

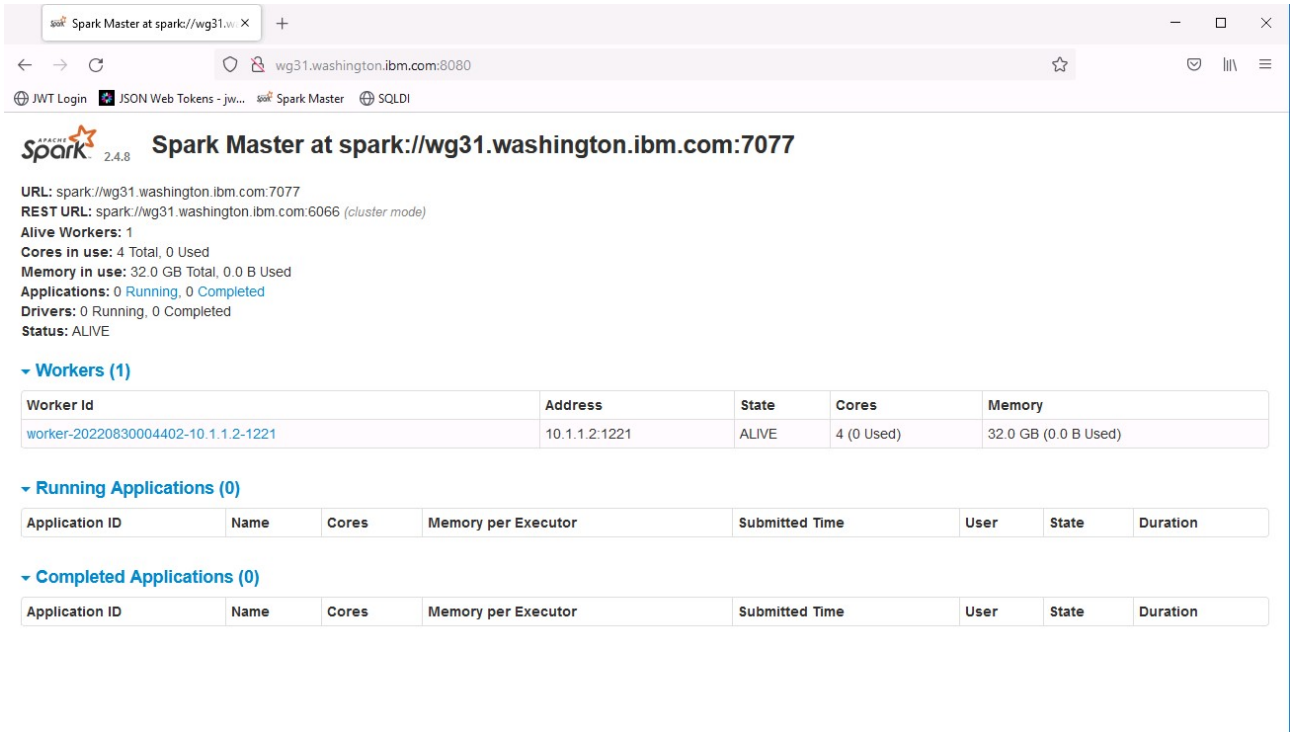


Figure 35: Spark UI

And Check the SQLDI Server by opening your browser on the SQLDI listener port `https://wg31.washington.ibm.com:15001/`.

Be sure to specify the URL exactly. Browsers will normally figure out whether an IP address is http or https , but this one doesn't.

Security Notes:

1. the first time you open this port there will be a privacy error.
2. The browser indicates that the session is "not secure", which is expected because the browser does not know about the CA that signed the certificate.
3. Just click proceed to accept the certificate.

Now you could logon to to the SQLDI Server. You would need to use a RACF userid that

1. has appropriate privileges to connect to Db2
2. has access the Db2 tables that you wish to use for SQL Data Insights
3. has access all the Db2 artefacts that support SQLDI
4. is a member of RACF Group SQLDIGRP

But you need to create the necessary Db2 artefacts first.

8 Create some SQLDI artefacts

The necessary functions required to run AI queries and to expploit SQLDI are installed with the **SQLDI FMID HDBDD18**. During this installation process there are just two steps to be run in order to complete SQLDI functions.

One of them is the pseudo-catalog that was already created in step 4.

The other one is a table sample that you can use to test SQLDI.

8.1 Create CHURN table for testing

TASK

Create the `CHURN` table using JCL `IBMUSER.SDISETUP(DSNTIJAV)`.

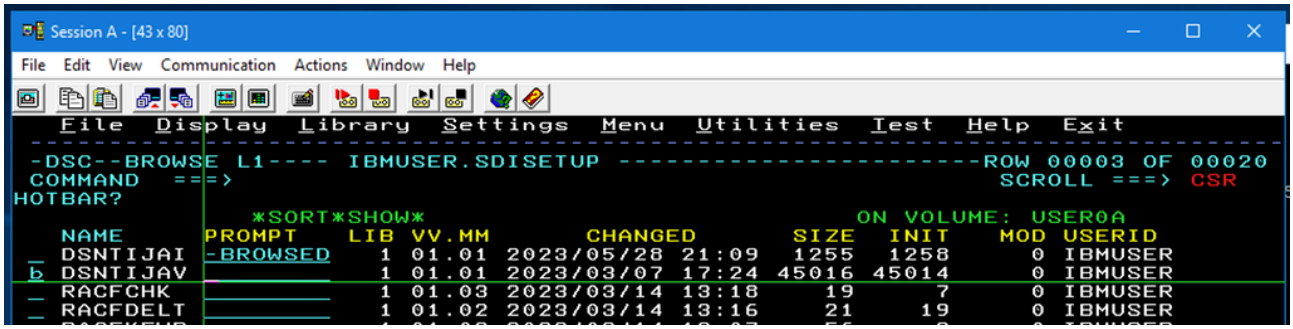


Figure 36: Member DSNTIJAV

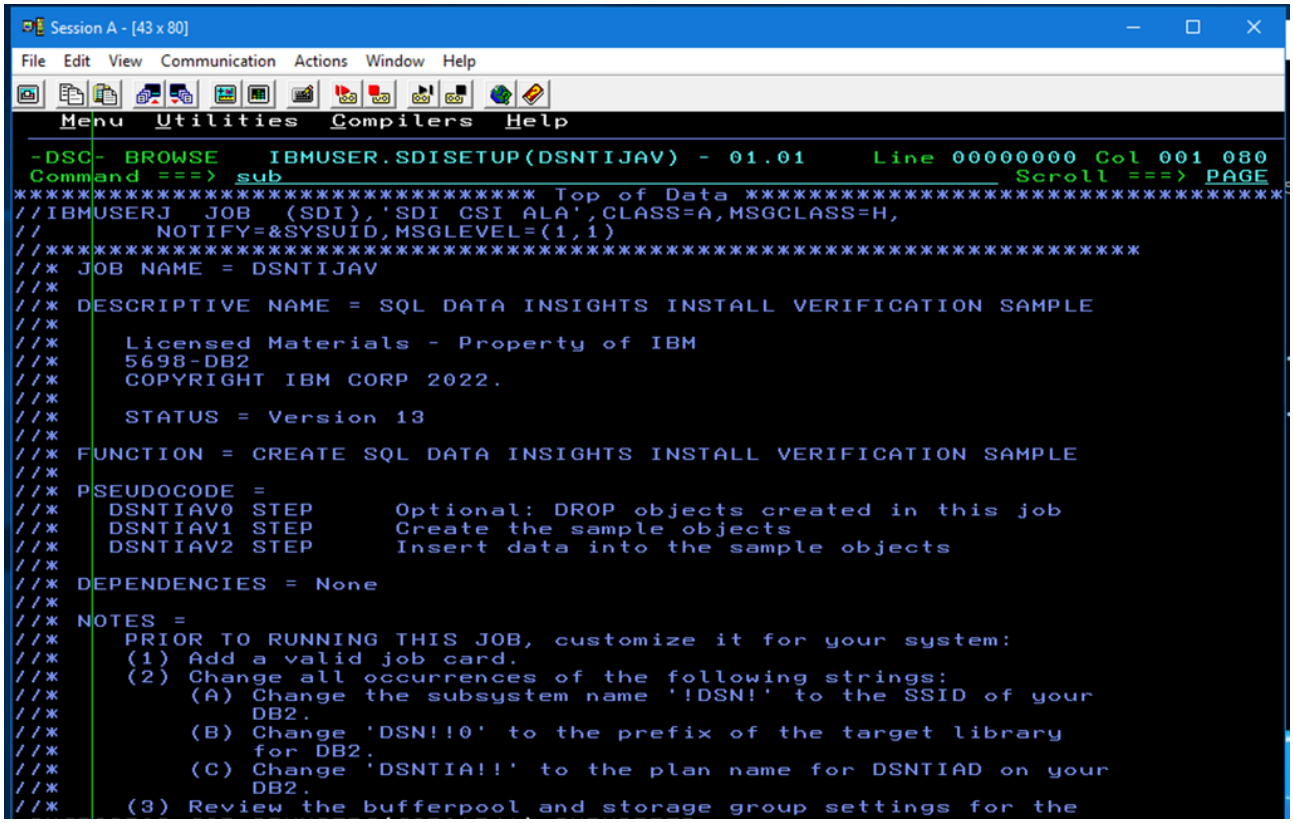


Figure 37: DSNTIJAV submit

The contents of the CHURN table can be viewed in Db2 Admin Tool (ISPF Option m.16).

Access the System Catalog Browser, use a Database Mask of `DSNAI*` and use a `b` command to browse the table if you are curious.

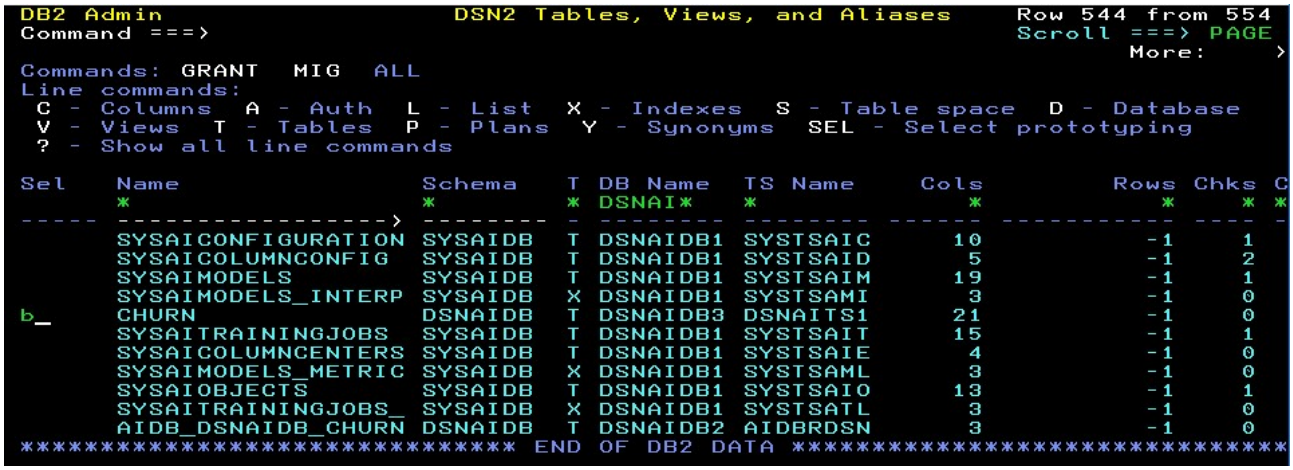


Figure 38: Access to CHURN table

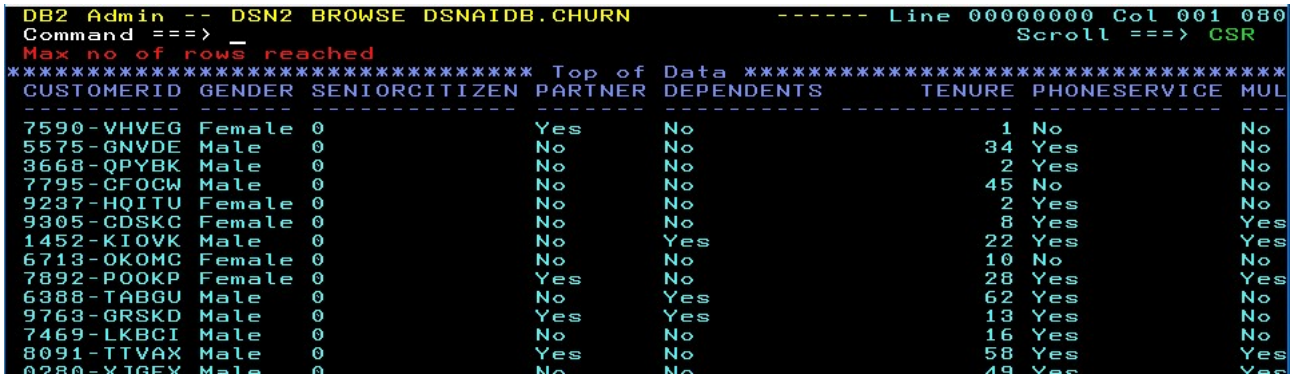


Figure 39: Browsing CHURN table

9 Test the installation with the IVP

Perform each of the tasks in the screenshot below to run an IVP test of SQLDI.

9.1 Get Connected to the SQLDI Server

Open the Chrome browser at `https://wg31.washington.ibm.com:15001` and logon with **IBMUSER/SYS1** (despite what you see in the image, YOUR USER IS NOT **USER1**).

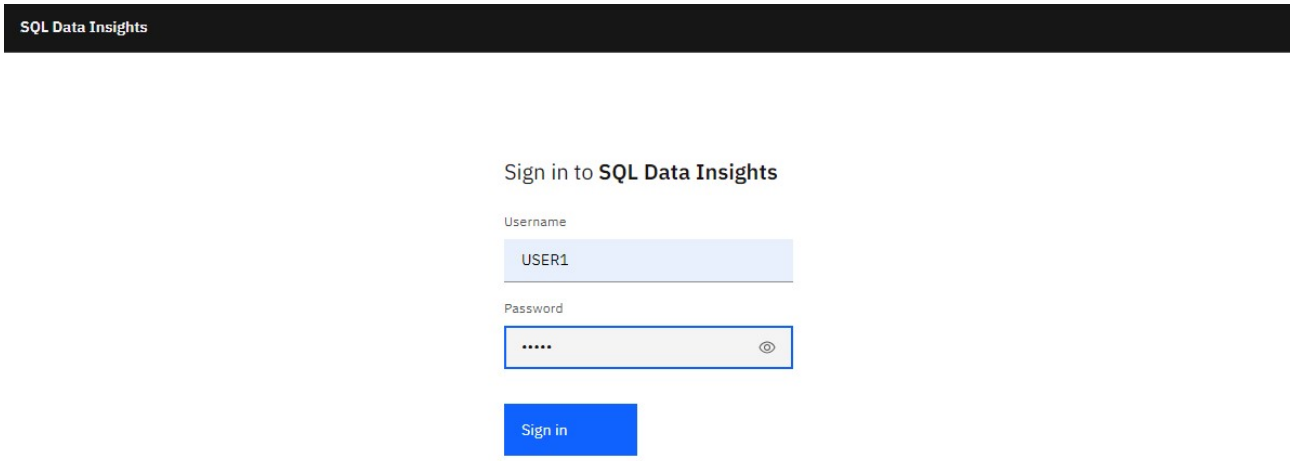


Figure 40: SQLDI Console

Oops . . . did you get an error like the one below ?

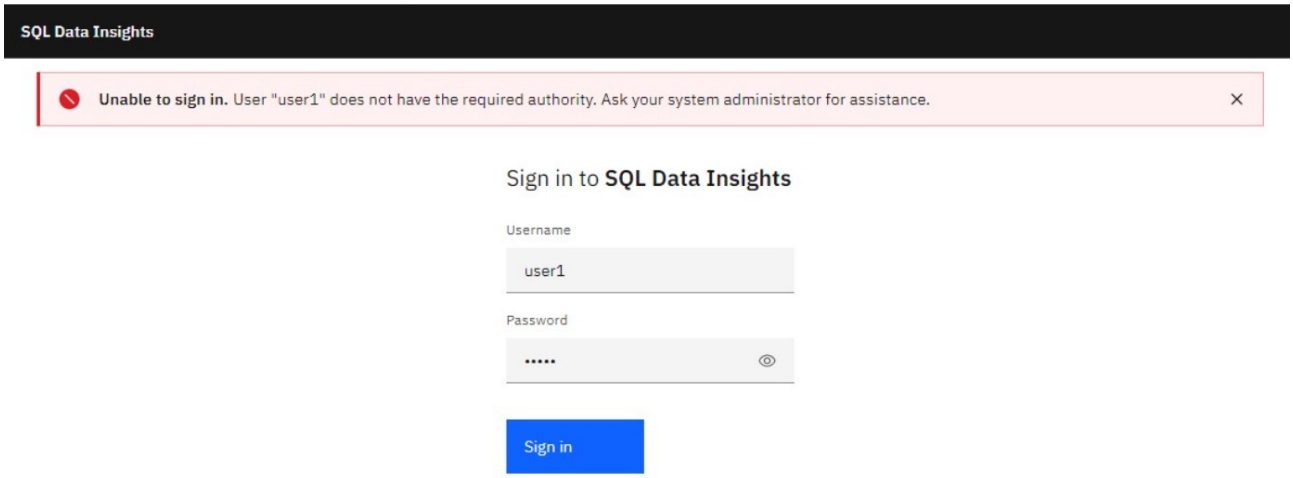


Figure 41: signon_error01

We know that IBMUSER has access to Db2 because we used it to perform the setup of all the Db2 objects required for SQLDI.

So we need to get a more detailed error message.

SQLDI and Spark each write logfiles in USS. The SQLDI Server writes it's logs to the the logs directory in the instance. Check the SQLDI Server log for additional information

In this case, we didn't get a lot more helpful diagnostic information, but it serves to illustrate the fact that much of the SQLDI diagnostic data will be surfaced in the USS environment.

```
 /u/aidb0020/holinstance/logs >ls -al
total 48
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug 15 23:17 .
drwxr-xr-x  10 AIDBADM  SYS1        8192 Aug 15 23:12 ..
-rw-r--r--   1 AIDBADM  SYS1         441 Aug 15 23:38 sql-data-insights_2022-08-15.0.log
 /u/aidb0020/holinstance/logs >cat sql-data-insights_2022-08-15.0.log
[2022-08-15 23:38:15.450] [application-akka.actor.default-dispatcher-44] [ERROR] [managements.UserManagement:49] -- User user1 can
not login SQL DI with error: {"errnoMsg":"unauthorized user","success":"false"}
[2022-08-15 23:38:15.453] [application-akka.actor.default-dispatcher-44] [ERROR] [controllers.LoginController:58] -- Failed to log
in, User "user1" does not have the required authority. Ask your system administrator for assistance.
 /u/aidb0020/holinstance/logs >
```

Figure 42: signon_error02

Section 11 provides guidance on how to increase the level of diagnostic information by editing the deploy.cfg file.

For now, just accept that the missing authority was membership of the RACF Group SQLDIGRP.

TASK

Edit the RACF job from earlier, or issue the following command from TSO option 6 to rectify the problem.

```
CONNECT (IBMUSER) GROUP(SQLDIGRP) OWNER(IBMUSER)
```

Now you should be abble to logon to the SQLDI Web UI with IBMUSER.

9.2 Define a Db2 Subsystem Connection

Observe there are currently no Db2 systems Connections. Press the "Add Connection" button.

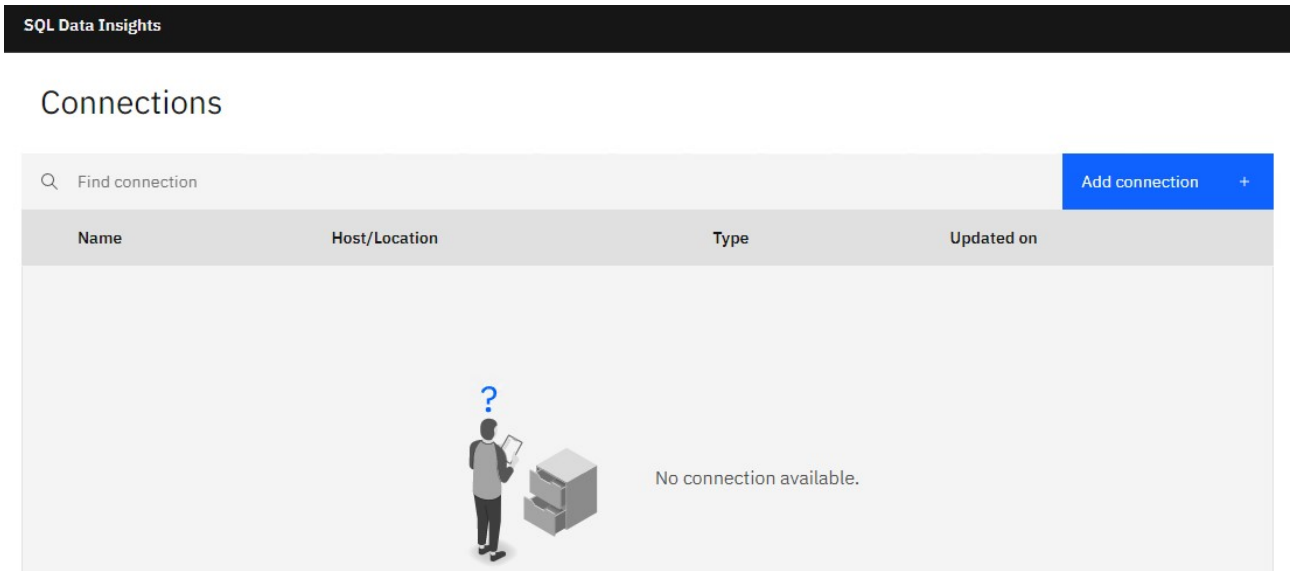


Figure 43: Connections

Fill in the details for the DBDG subsystem

- Name: `DBDG`
- Hostname: `wg31.washington.ibm.com`
- Port: `5045` (secport: `5046`)
- Location Name: `DALLASD`

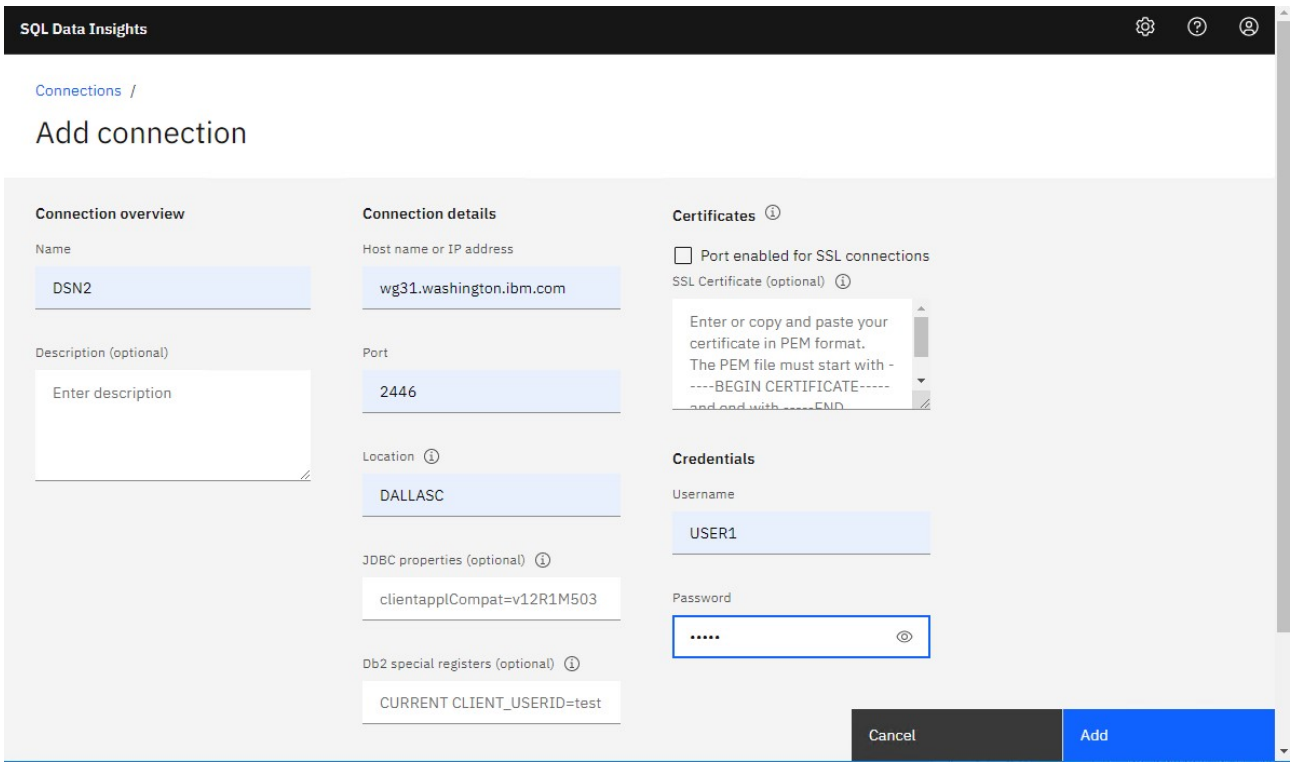


Figure 44: Add connection

- User/Password: `IBMUSER/SYS1`

See the newly defined connection, and Press the "**Connect**" button

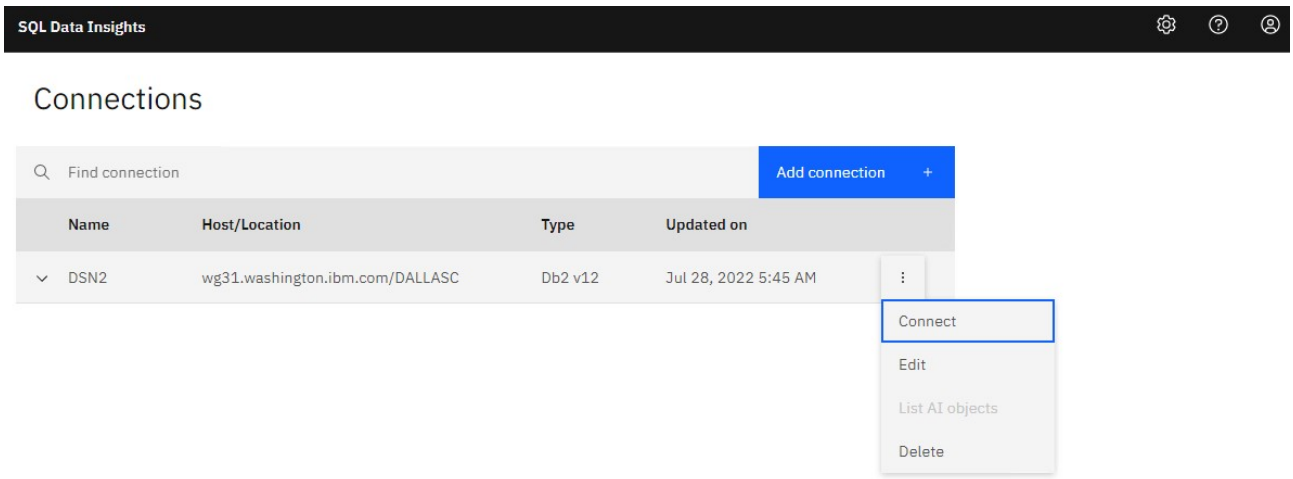


Figure 45: Connections

9.3 Work with "AI-Enabled" Tables

Once connected, select "**List AI Objects**" and observe that there are no AI-Enbled objects. Press the "**Add Object**" button.

Select the Table Schema "**DSNNAIDB**" and press the **Search Icon** to the right hand side of the window.

Select the Table `DSNAIDB.CHURN`

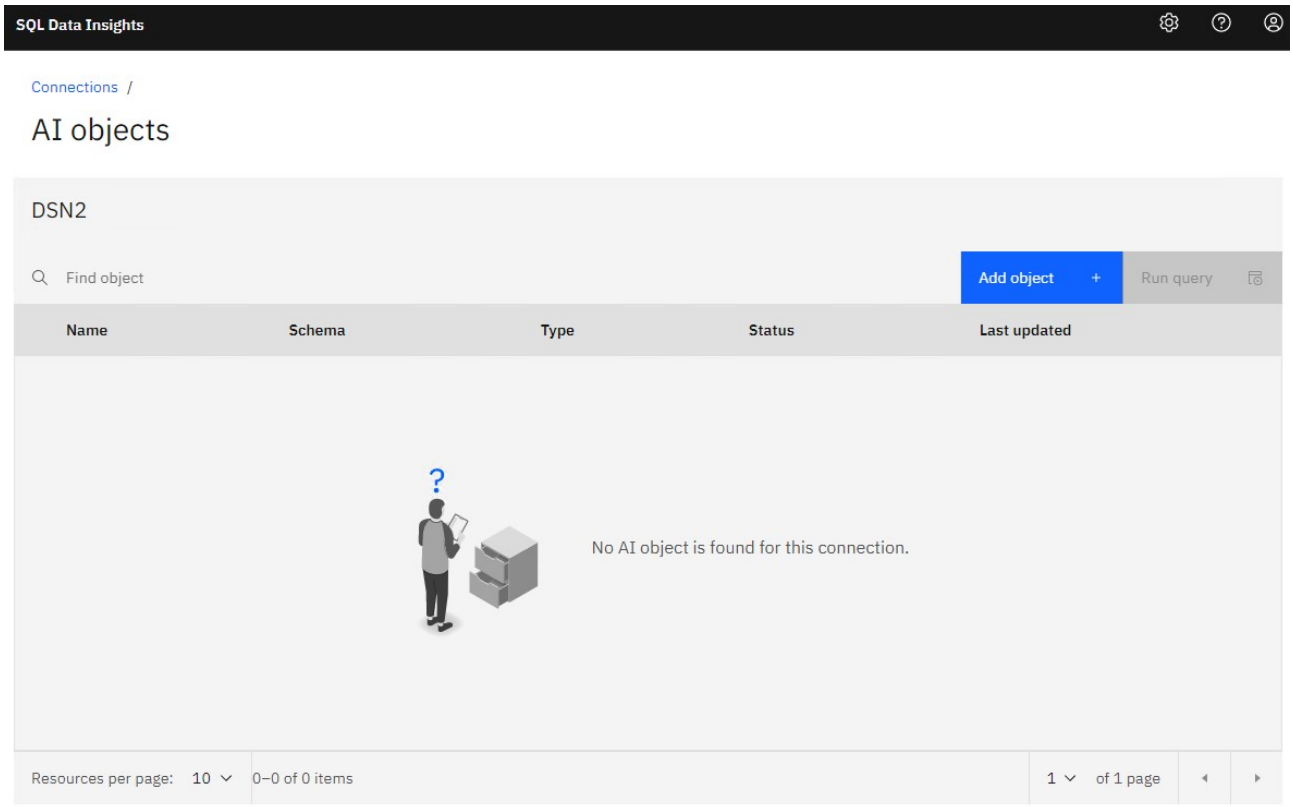


Figure 46: AI objects

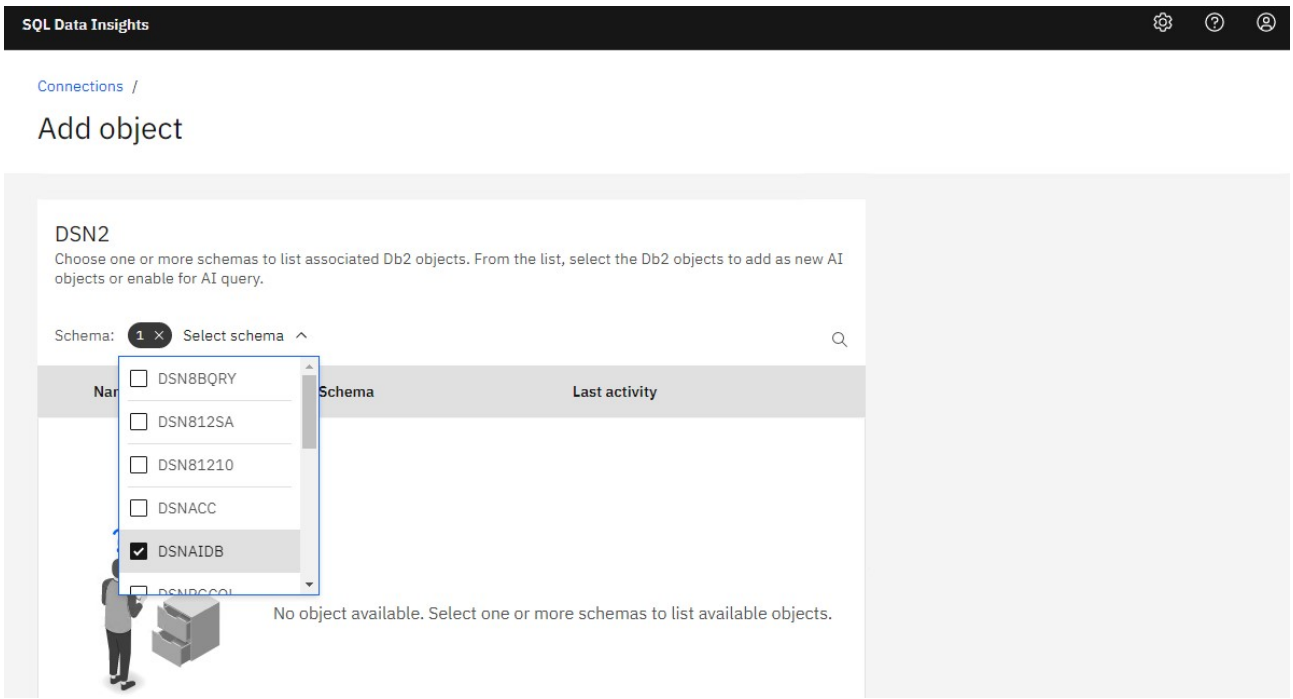


Figure 47: Add object

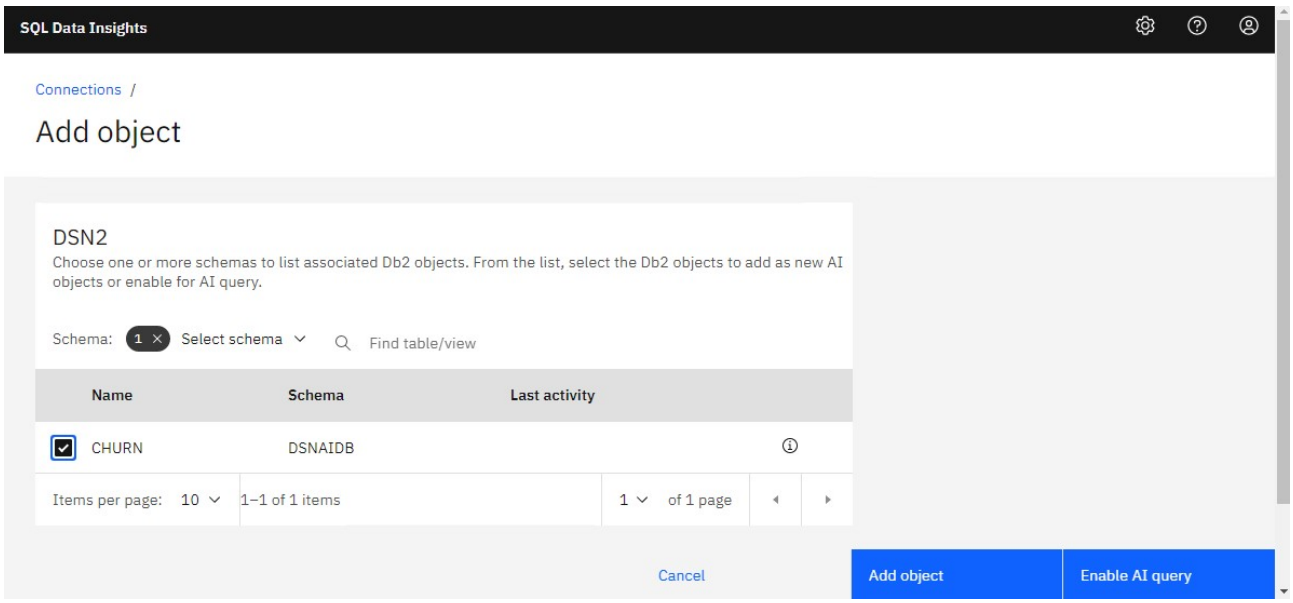


Figure 48: Add object

Select all the columns. (Note that SQLDI allows you to overwrite it's default choice of whether a column is Categorical or Numeric.)

Push the "**Enable AI**" button.

Note the caution that column selections cannot be changed after the model is build, and Push "**Enable AI**"

9.4 Be patient during Model Training and Observe Progress

Wait a few seconds to see the Browser showing the "**Enabling**" status.

Select the **Expansion** button to the left to get a more detailed view. If the model training fails for any reason, the SQL Error code will be visible here.

The image below is an example of a model training job that failed.

In this case a clear Db2 error code was returned, indicating that the WLM environment for DSNUTILU had not been correctly setup.

Sometimes the error messages are less clear.

In such cases you need to look at the SQLDI and Spark execution logs to discover what went wrong. Step 10 covers problem determination steps.

Training a model can take a while, especially if you are not running on a Z16 with a Telum AIU.

Open a new browser tab and access `wg31.washington.ibm.com:8080` to see what the Spark environment is doing.

Once the Running Application has finished, if you pop back to the SQLDI portal, you should see the Table "Enabled" for AI. Be aware that the automated refresh of this URL is several minutes, so you may choose to refresh the browser manually.

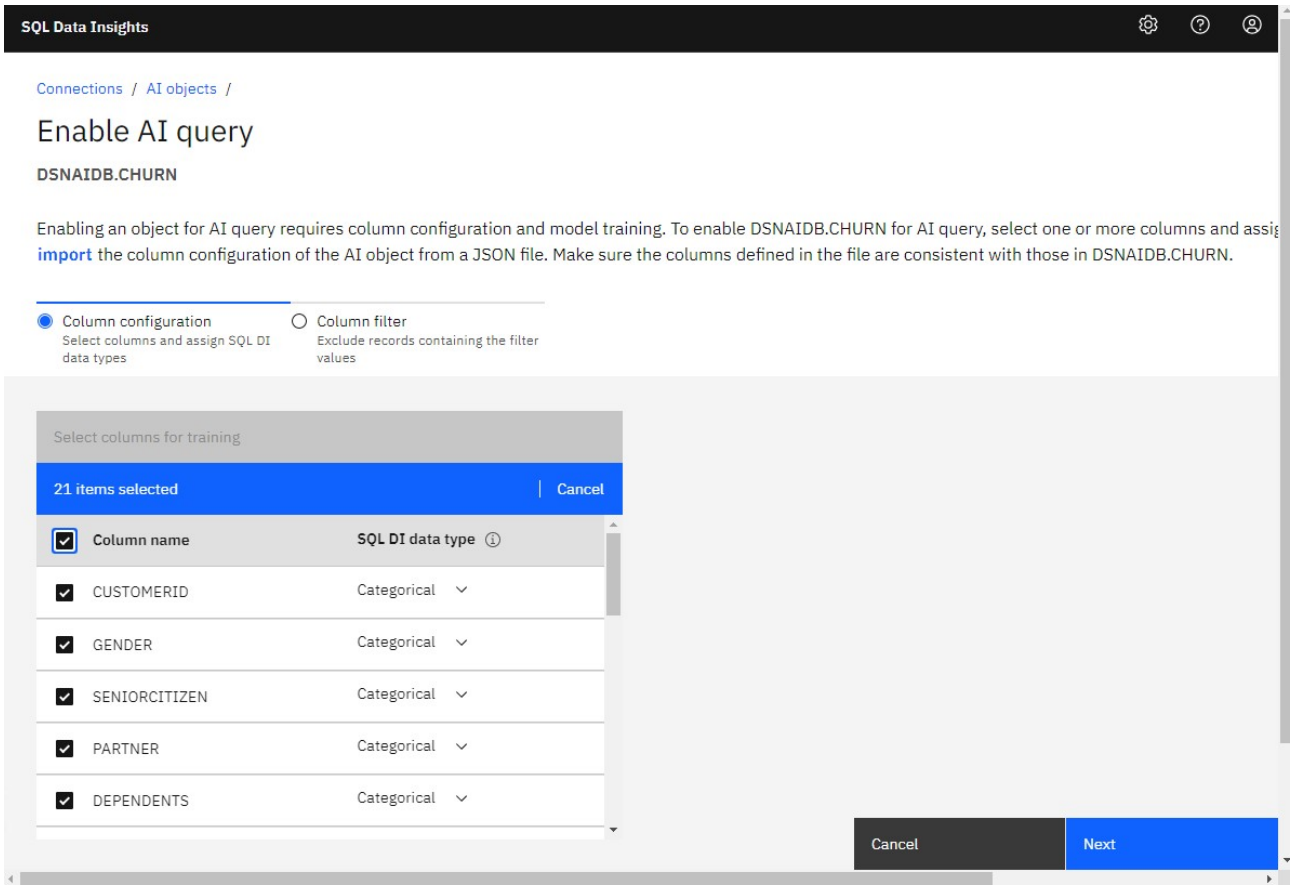


Figure 49: Enable AI query

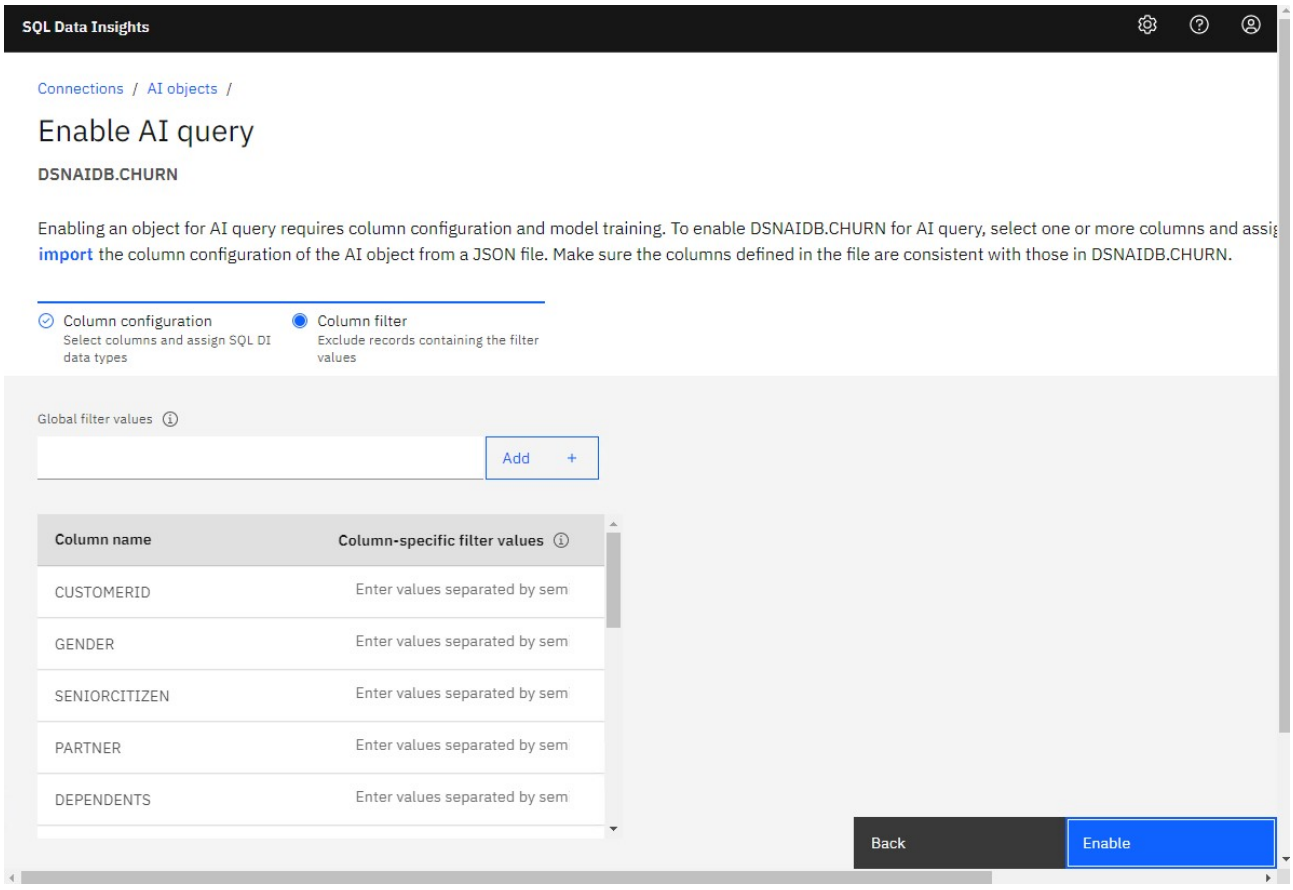


Figure 50: Enable AI query

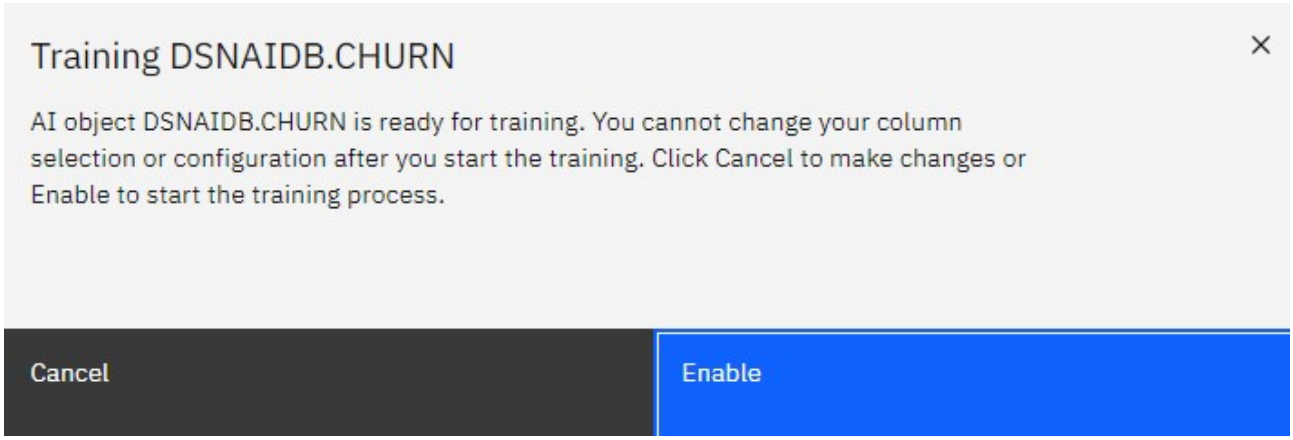


Figure 51: Enable training

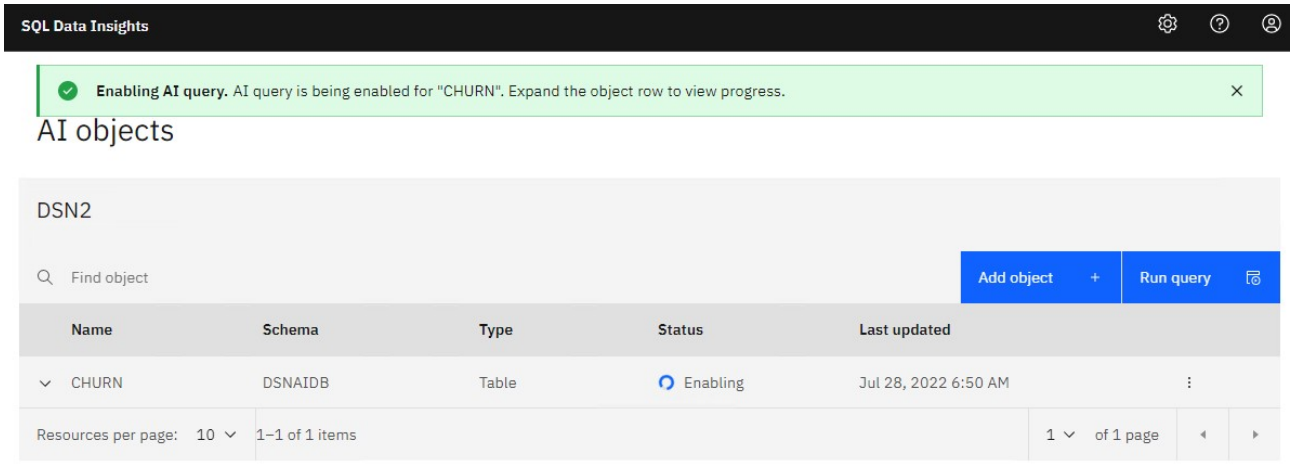


Figure 52: AI Objects

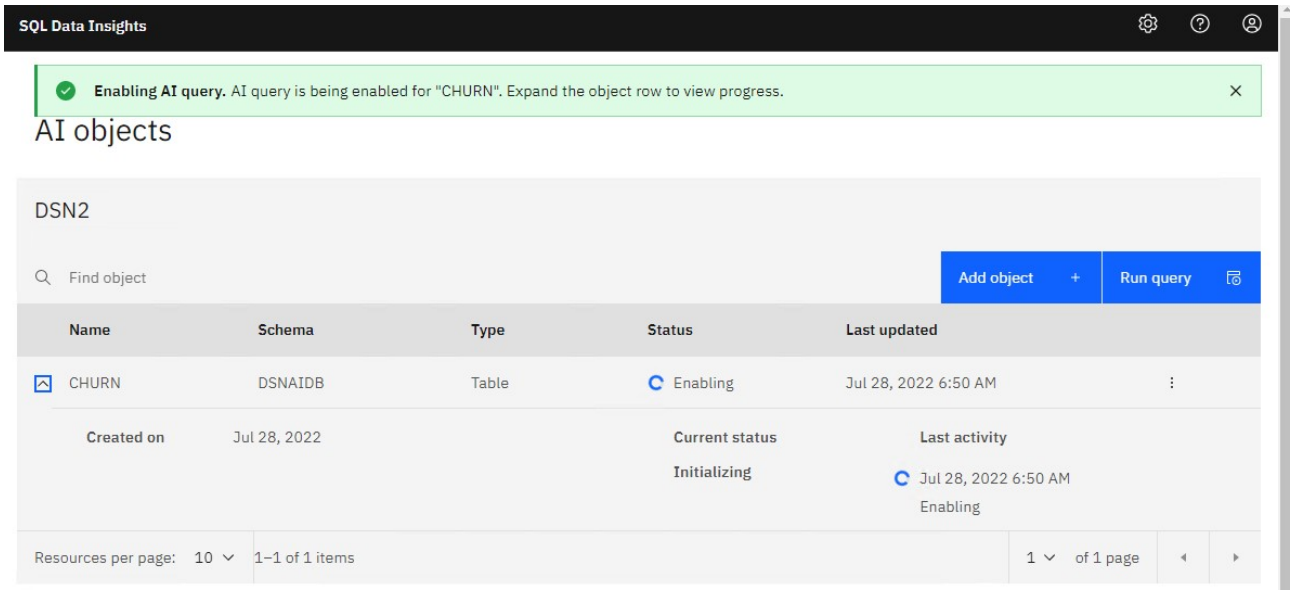


Figure 53: AI Objects

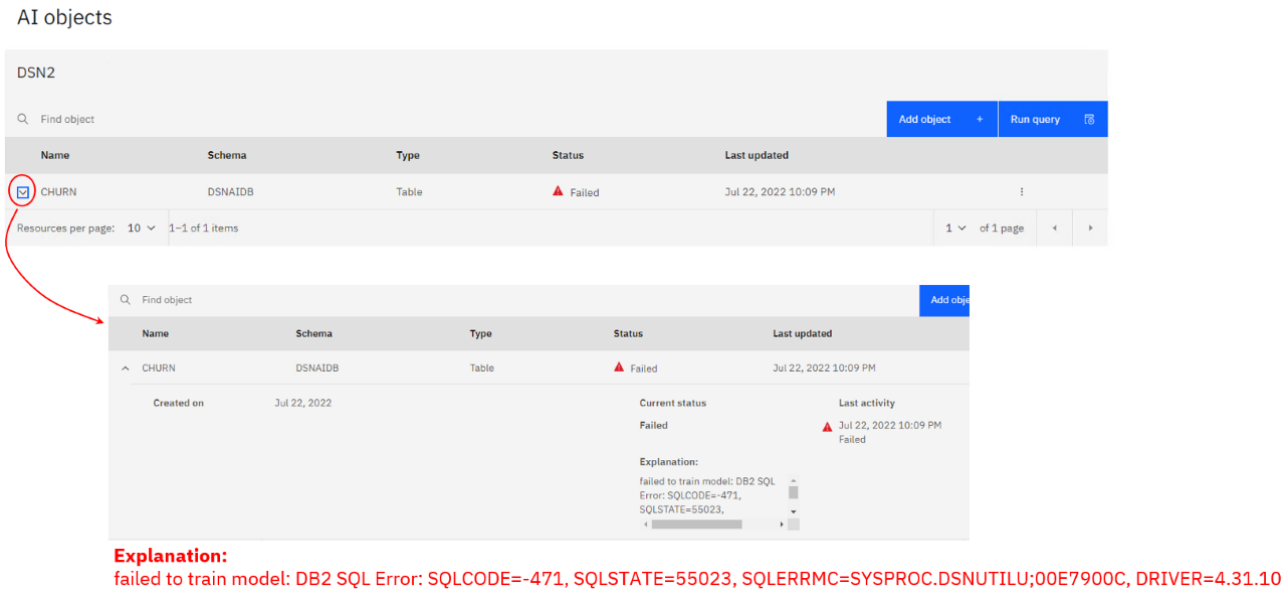


Figure 54: SQLDI_dropdown

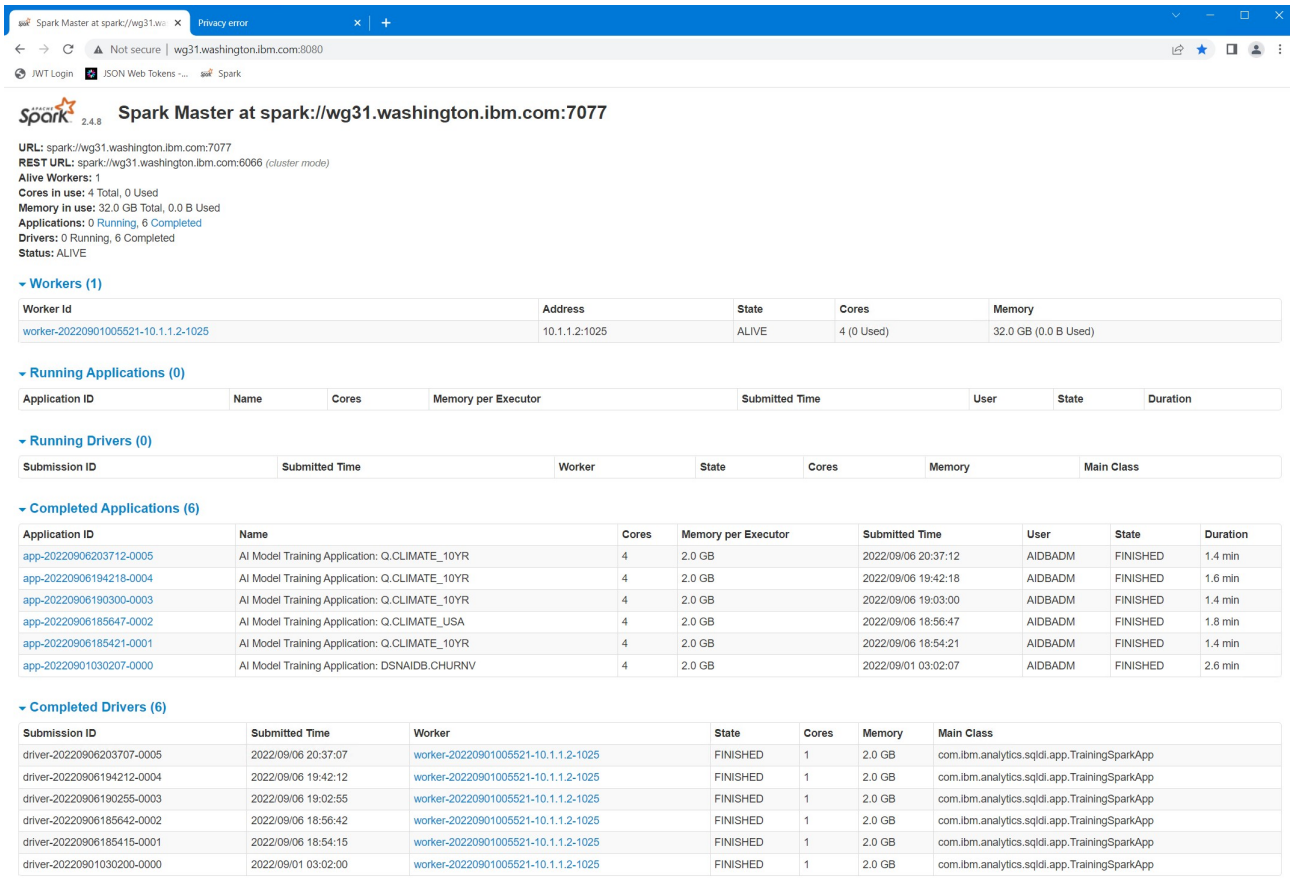


Figure 55: Spark master

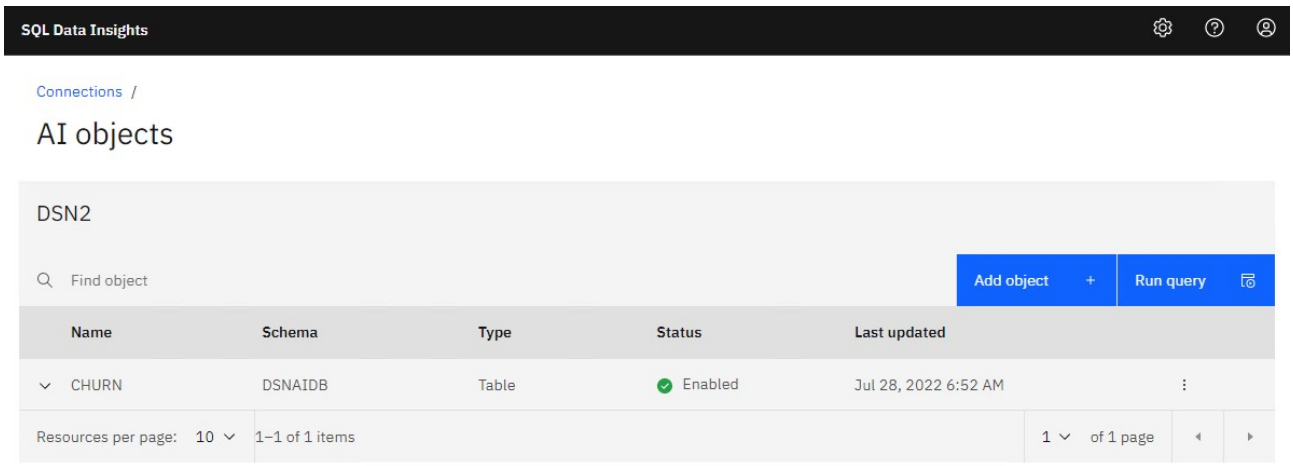


Figure 56: AI objects

9.5 Use SQLDI Dashboards

Note the available actions by pressing the elipses button on the right hand side. (DISABLE, ANALYZE DATA etc...)

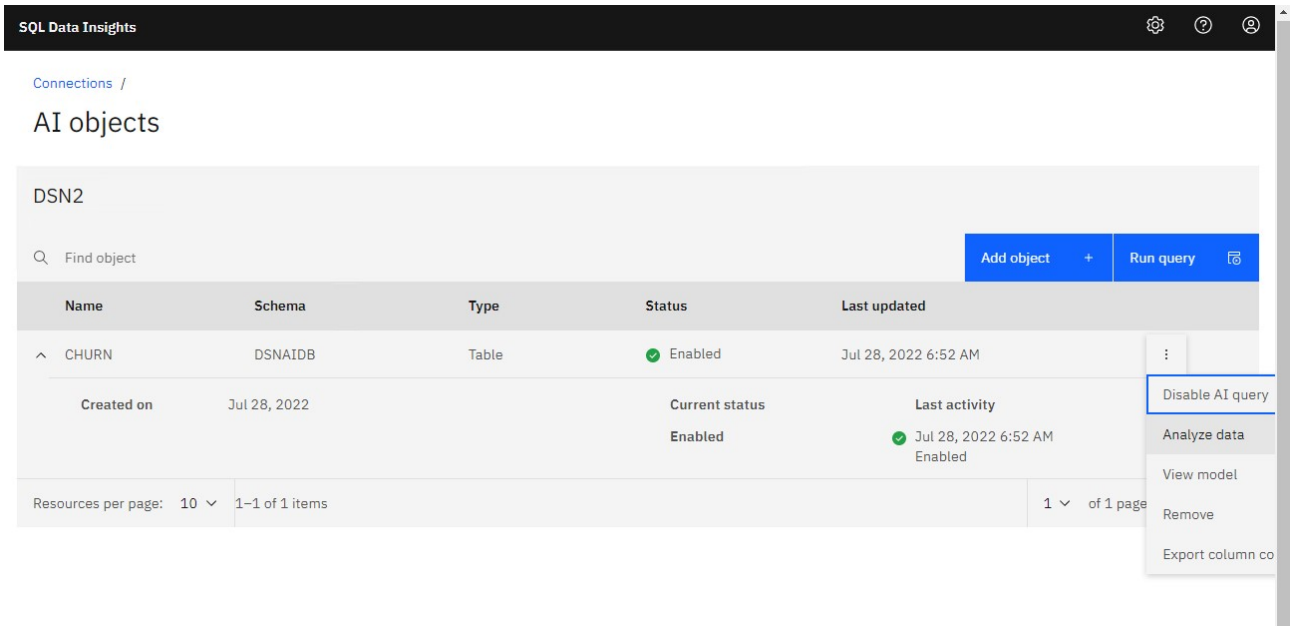


Figure 57: AI objects

Select Analyze Data, and you will see the first of 3 tabs with information. This first tab shows the object details.

The second tab shows data statistics. These are essential data points for a data scientist performing data wrangling.

The third tab shows the column influence of the model that SQLDI has established. This immediately tells the data scientist which columns are likely to be required when building scoring models.

NOTE. The column influence dashboard needs to be modified, because the influencing columns are dwarfed by the discriminator column (the unique key value). A future update should see

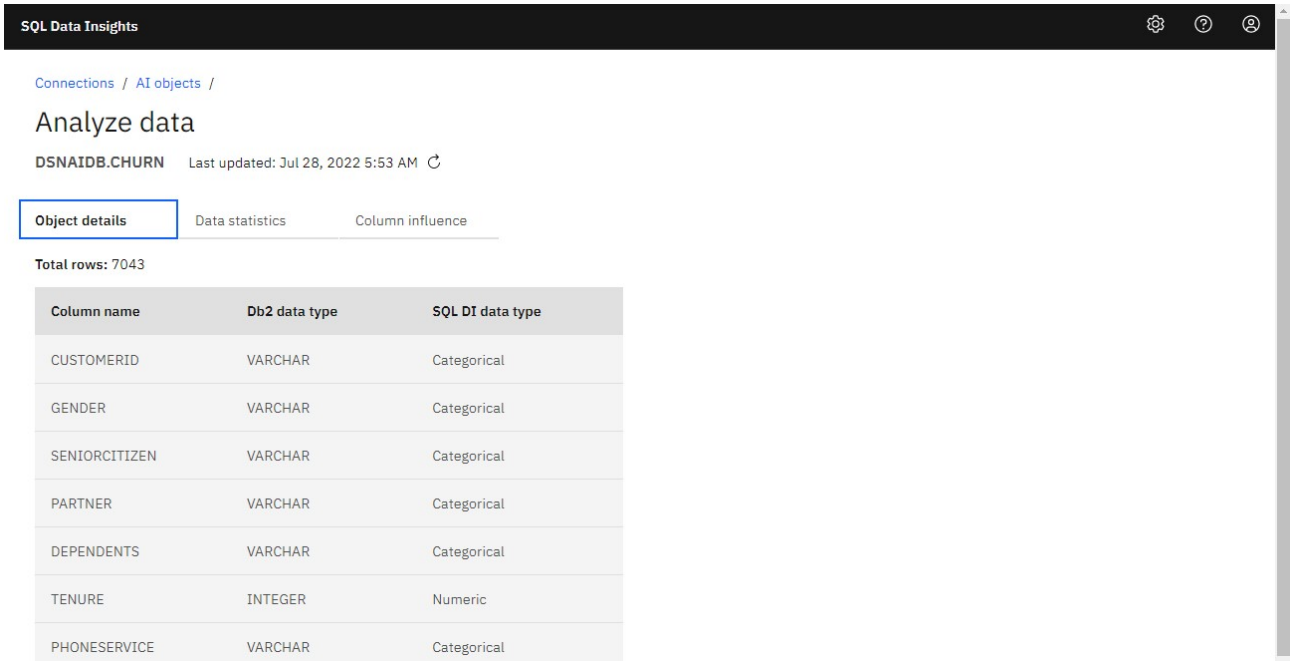


Figure 58: Analyze data

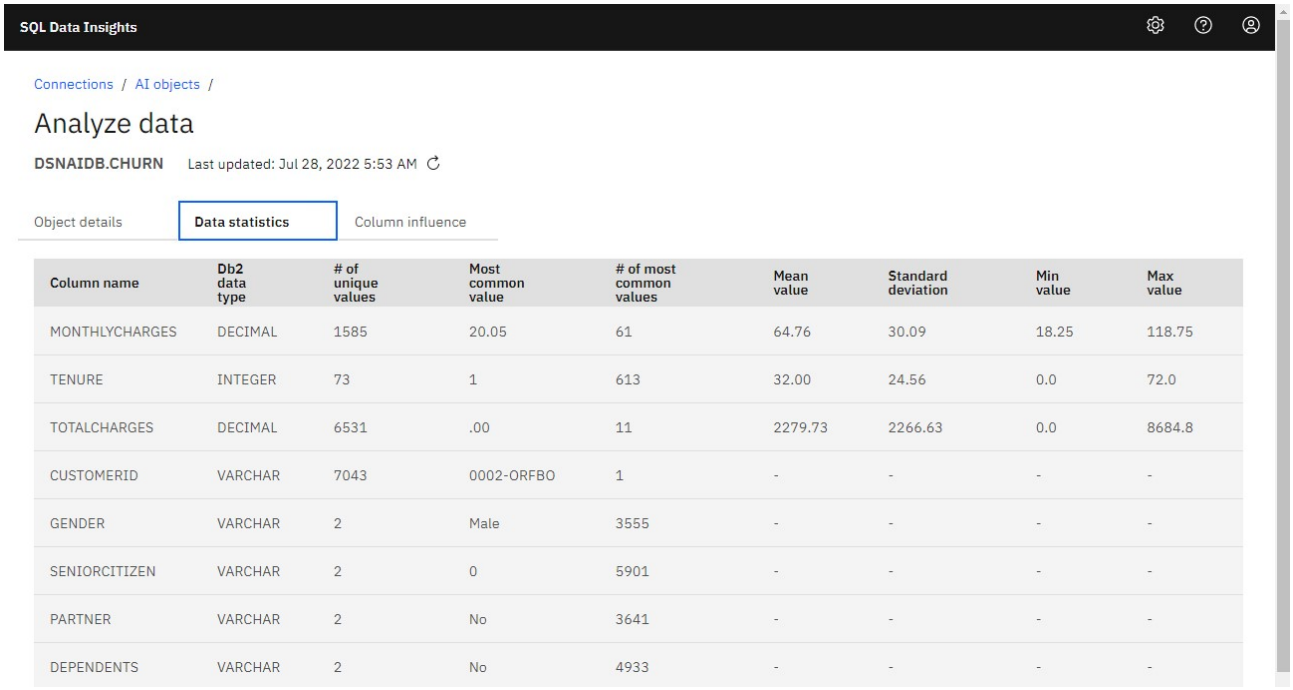


Figure 59: Analyze data

changes to this chart.

Figure 60: Analyze Data

9.6 Write AI queries

The next thing to do in the IVP test is to run an AI Query. Select "**Run Query**" button for the selected AI-Enables Table.

Use the Query Type Pulldown to obtain an SQL template that you will need to edit. Lets start with a "**Semantic Similarity**" query.

Note. These are static templates for the kind of query that SQLDI supports. they do not reflect the selected table at all. They are nothing more than a template for editing.

This is the SQL template that is provided. It is based on the CHURN Table in a different schema. You need to edit the schema to make the query run.

However, before editing the SQL, press "**Run**" to get a feel for what happens when an AI SQL query fails.

`SQLCODE -443` with a return code of +100 against AIOBJECTS and AIMODEL seems strange at first. What this actually means is that the UDF could not find the model table in the pseudo catalog. No surprise really because the table has been incorrectly identified.

Correct the error by overtyping the schema in the UDF call to "DSNAIDB". Run the Query.

```
SELECT * FROM
   (SELECT C.*,
       SYSFUN.AI_SIMILARITY('CUSTOMERID', CUSTOMERID,
       'CUSTOMERID', '3668-QPYBK', 'DSNAIDB', 'CHURN') AS SIMILARITY
    FROM DSNAIDB.CHURN C
    WHERE CUSTOMERID <> '3668-QPYBK')
  WHERE SIMILARITY > 0.5
  ORDER BY SIMILARITY DESC
  FETCH FIRST 20 ROWS ONLY;
```

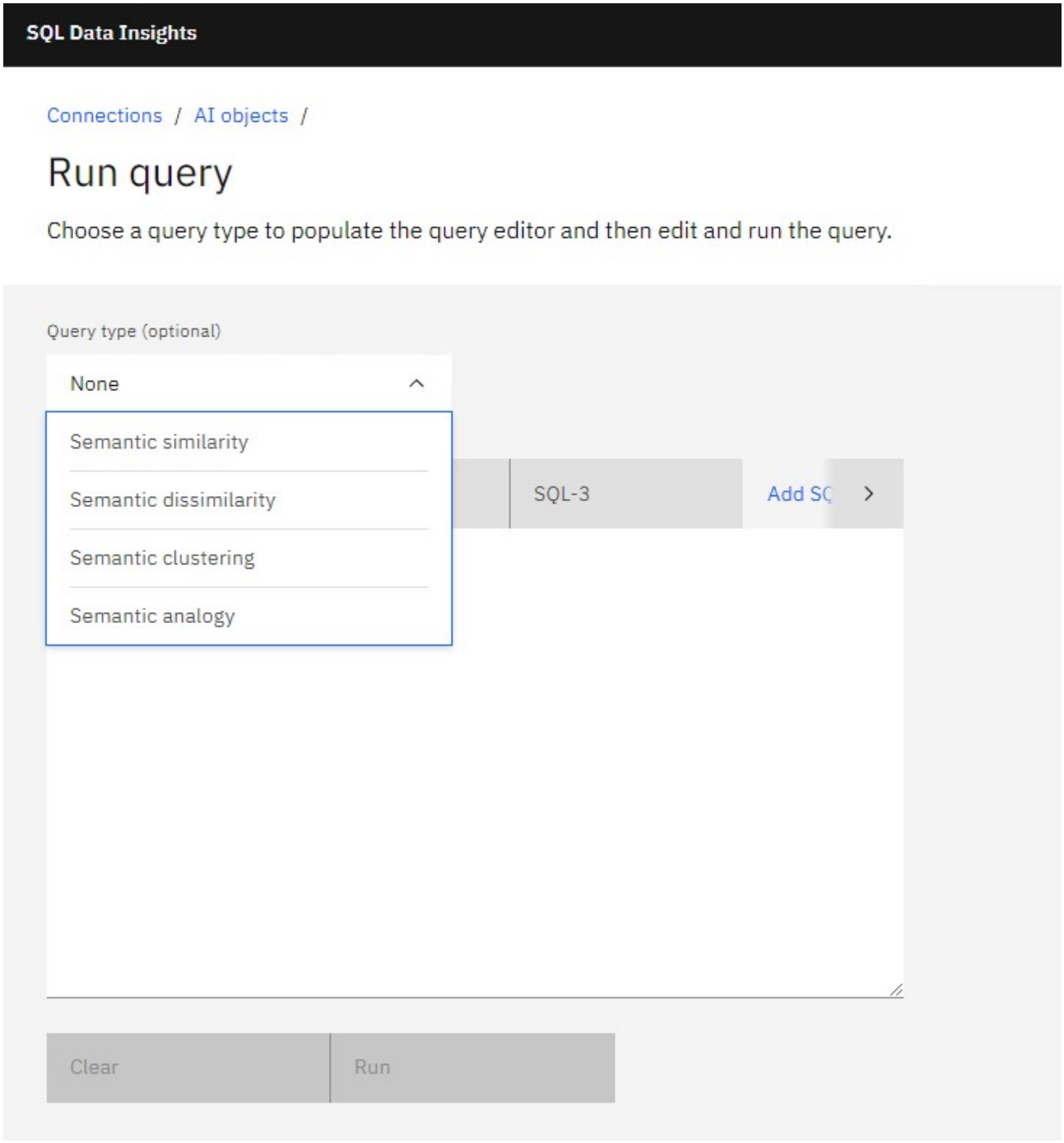


Figure 61: Run query

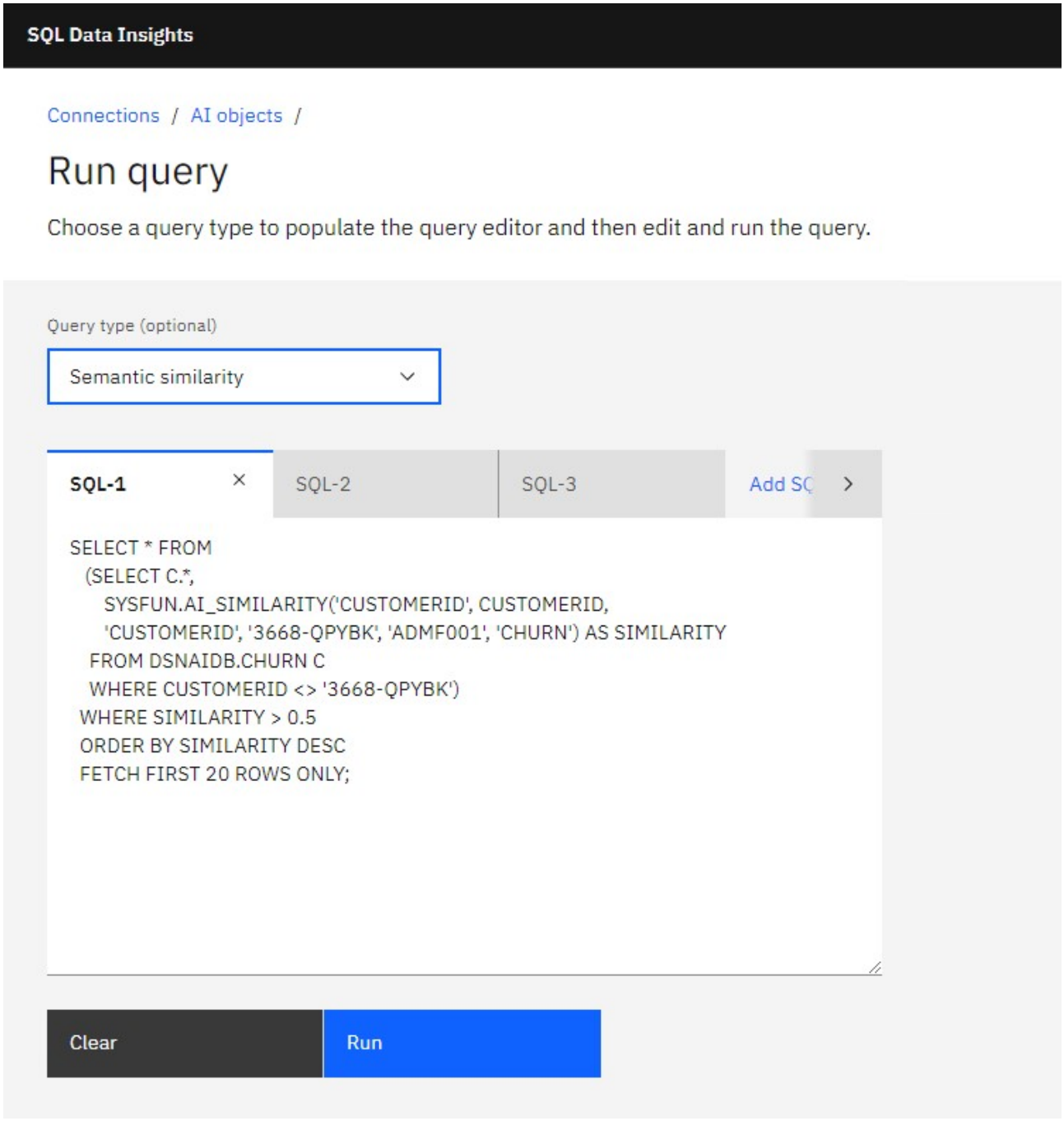


Figure 62: Run query

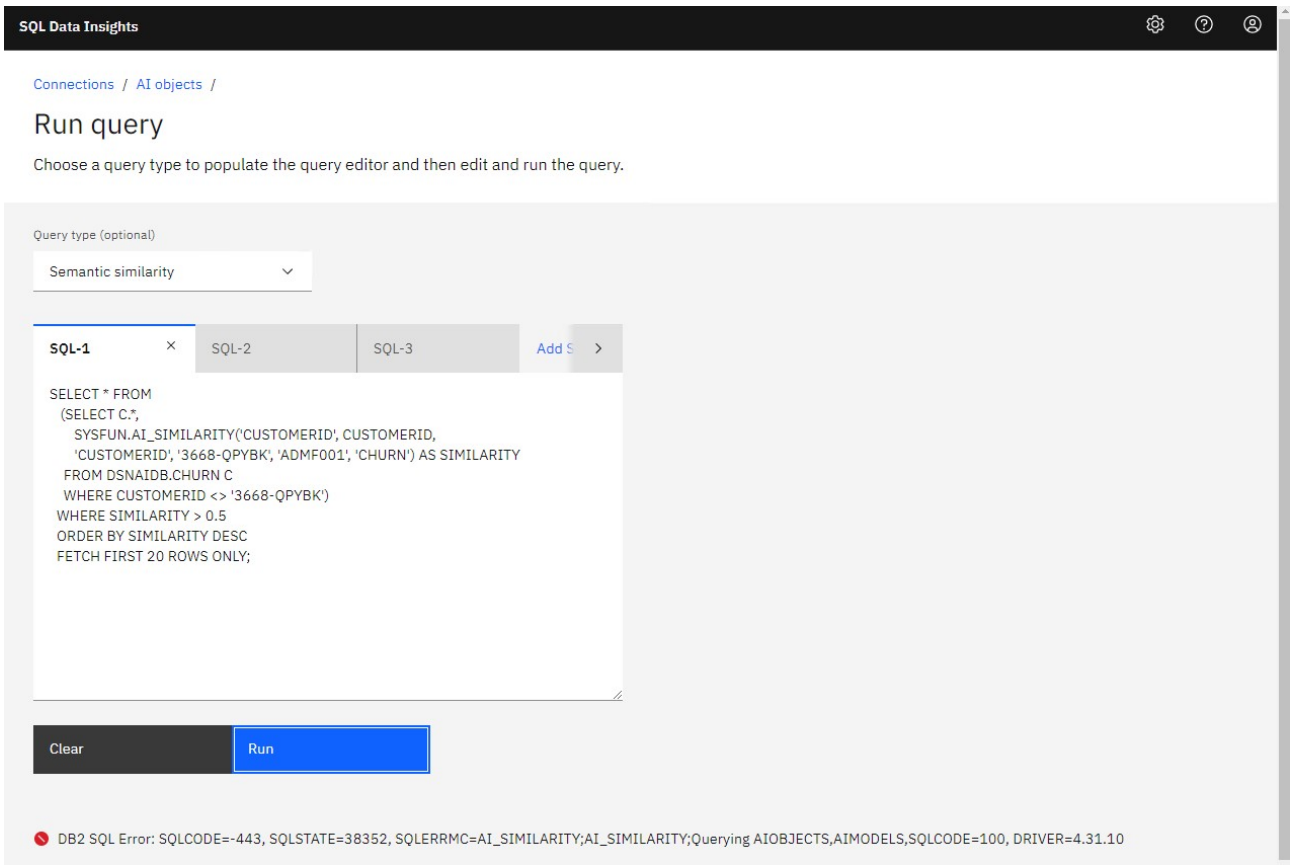


Figure 63: Run query

Scroll down to see the results.

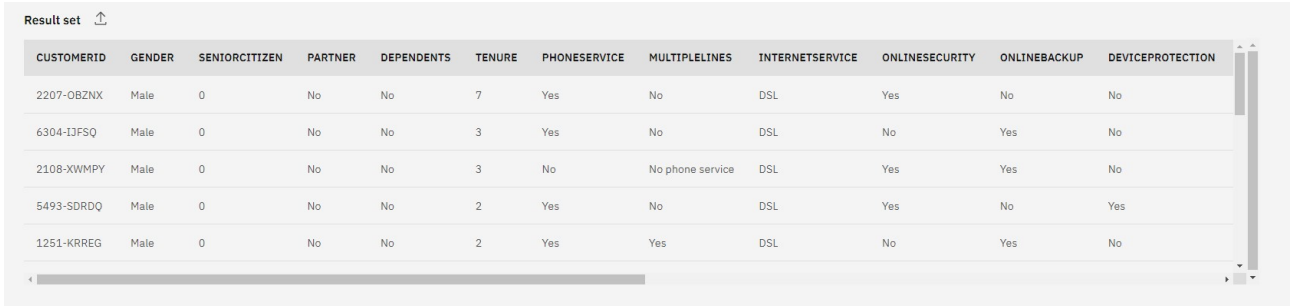


Figure 64: Run result

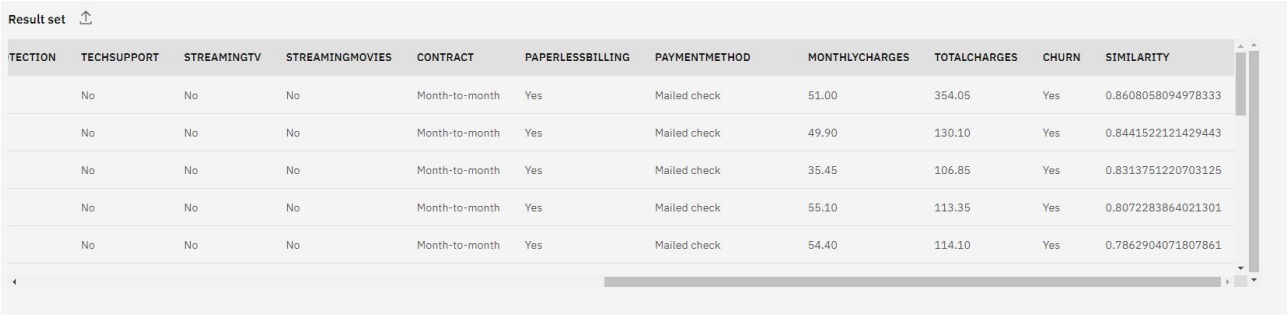
Scroll to the right to retrieve the similarity score. This is the mode's similarity rating against customer_id `3668-QPYBK` for each row in the result set.

You can experiment with the other SQL templates to get an understanding of the purpose of each of the AI functions. Then you will be ready to write your own AI queries.

10. Problem Determination Steps

Hopefully you completed your Hands on Learning workshop first time, before you had any reason to review this section on problem determination steps. Even if you did, please take a moment to review this section, so that you know where to look for execution logs in future.

This section covers the followinng topics



Result set

TECTION	TECHSUPPORT	STREAMINGTV	STREAMINGMOVIES	CONTRACT	PAPERLESSBILLING	PAYMENTMETHOD	MONTHLYCHARGES	TOTALCHARGES	CHURN	SIMILARITY
	No	No	No	Month-to-month	Yes	Mailed check	51.00	354.05	Yes	0.8608058094978333
	No	No	No	Month-to-month	Yes	Mailed check	49.90	130.10	Yes	0.8441522121429443
	No	No	No	Month-to-month	Yes	Mailed check	35.45	106.85	Yes	0.8313751220703125
	No	No	No	Month-to-month	Yes	Mailed check	55.10	113.35	Yes	0.8072283864021301
	No	No	No	Month-to-month	Yes	Mailed check	54.40	114.10	Yes	0.7862904071807861

Figure 65: Run result

1. Where to find execution logs
2. How to increase the amount of diagnostic information that is available
3. An insight into the SQLDI workflow that happens after you press the blue "Enable AI" button

10.1 Where to find execution logs.

When you create an SQLDI instance (holinstance) at path /u/aidbadm/holinstance the following subdirectories are created.

```
 /u/aidbadm/holinstance >ls -al
total 178
drwxr-xr-x  10 AIDBADM  SYS1        8192 Aug  8 04:14 .
drwxrwxrwx  10 990027   SYS1        8192 Aug  8 04:17 ..
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:37 conf
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  7 17:08 db
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  7 16:58 db2jcc.driver
-rw-r--r--   1 AIDBADM  SYS1         442 Aug  9 21:32 deploy.cfg
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  7 16:58 diag
-rw-r--r--   1 AIDBADM  SYS1           5 Aug  8 04:14 fred.cat
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:37 logs
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:37 pid
drwxr-xr-x   7 AIDBADM  SYS1        8192 Aug  7 16:59 spark
drwxr-xr-x   4 AIDBADM  SYS1        8192 Aug  7 16:58 temp
```

The `/u/aidbadm/holinstance/logs` path is where SQLDI logs are written.

Inside that directory you will find a new SQLDI log file written every day. Each day's log will be appended until midnight passes, whereupon a new log file will be created for further messages.

```
 /u/aidbadm/holinstance/logs >ls -al
total 80
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:37 .
drwxr-xr-x  10 AIDBADM  SYS1        8192 Aug  8 04:14 ..
-rw-r--r--   1 AIDBADM  SYS1           0 Aug  7 17:08
 ↪  sql-data-insights_2022-08-07.0.log
-rw-r--r--   1 AIDBADM  SYS1         663 Aug  8 05:26
 ↪  sql-data-insights_2022-08-08.0.log
-rw-r--r--   1 AIDBADM  SYS1       10420 Aug  9 21:40
 ↪  sql-data-insights_2022-08-09.0.log
```

These log files might be quite small, because by default they only get written when errors occur. The next section explains how to expand the level of logging if you need it.

The bulk of the work performed by SQLDI is executed in the spark environment. The spark directory contains 5 further subdirectories.

```
 /u/aidbadm/holinstance/spark >ls -al
total 112
drwxr-xr-x   7 AIDBADM  SYS1        8192 Aug  7 16:59 .
drwxr-xr-x  10 AIDBADM  SYS1        8192 Aug  8 04:14 ..
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  7 16:59 conf
drwxrwxr-x   9 AIDBADM  SYS1        8192 Aug  9 21:42 local
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:33 log
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:33 pid
drwxr-xr-x  16 AIDBADM  SYS1        8192 Aug  9 21:40 worker
```

The conf directory contains some configuration files, including the spark-env.sh script. Some of the parameters in this script were set directly during the instance creation dialog.

Other parameters (memory, number of cores, paths etc...) are set automatically by the instance creation script. They include the directories

```
 /u/aidbadm/holinstance/spark/conf >cat spark-env.sh

SPARK_MASTER_HOST=10.1.1.2
SPARK_LOCAL_IP=10.1.1.2
SPARK_MASTER_PORT=7077
SPARK_MASTER_WEBUI_PORT=8080
SPARK_WORKER_WEBUI_PORT=8081
SPARK_DAEMON_MEMORY=1G
SPARK_WORKER_INSTANCES=1
SPARK_LOG_DIR=/u/aidbadm/monday/spark/log
SPARK_LOCAL_DIRS=/u/aidbadm/monday/spark/local
SPARK_WORKER_DIR=/u/aidbadm/monday/spark/worker
SPARK_PID_DIR=/u/aidbadm/monday/spark/pid
SPARK_WORKER_MEMORY=32G
SPARK_WORKER_CORES=4
```

The log subdirectory contains the most helpful spark output logs. The worker logs are likely to be the most informative source of information about the work that has been performed.

```
 /u/aidbadm/holinstance/spark/log >ls -al
total 352
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:33 .
drwxr-xr-x   7 AIDBADM  SYS1        8192 Aug  7 16:59 ..
-rw-r--r--   1 AIDBADM  SYS1        3911 Aug  9 21:42
 ↪  spark-AIDBADM-org.apache.spark.deploy.master.Master-1-wg31.washington.ibm.com.out
-rw-r--r--   1 AIDBADM  SYS1       12640 Aug  8 06:06
 ↪  spark-AIDBADM-org.apache.spark.deploy.master.Master-1-wg31.washington.ibm.com.out.1
-rw-r--r--   1 AIDBADM  SYS1       16464 Aug  9 21:42
 ↪  spark-AIDBADM-org.apache.spark.deploy.worker.Worker-1-wg31.washington.ibm.com.out
-rw-r--r--   1 AIDBADM  SYS1       87693 Aug  8 06:06
 ↪  spark-AIDBADM-org.apache.spark.deploy.worker.Worker-1-wg31.washington.ibm.com.out.1
```

```
-rw-r--r--   1 AIDBADM  SYS1         170 Aug  7 16:59
↪  spark-configurator-master-stdout.log
```

10.2 How to increase the amount of diagnostic information that is available

Review /u/aidbadm/holinstance/deploy.cfg

```
VERSION="1.0.0.0"

SERVICE_HOST=10.1.1.2
SERVICE_PORT=15001

# keyring path and label
KEYSTORE_TYPE=JCERACFKS
KEYSTORE_PATH=safkeyring://QUlEQkFETS9XTUxaUklORw==
CERTIFICATE_LABEL=WMLZCert_WMLZID

# overall log level for SQL DI application and Spark job
log4j_level=ERROR

# minimum JVM heap size for SQL DI application
Xms=-Xms512M
# maximum JVM heap size for SQL DI application
Xmx=-Xmx2048M

# ALL, NONE, ON_FAILURE
KEEP_TRAINING_FILES=ON_FAILURE
```

SQLDI uses log4j for logging (a current version of log4j that is not affected to the infamous security exposure).

By default, SQLDI is configured to log ERRORS only.

You can google the org.apache.log4j.Level levels and select a different level (such as OFF, INFO, DEBUG, TRACE etc..).

10.3 An insight into the SQLDI workflow that happens after you press the blue "Enable AI" button

Another interesting value in the deploy.cfg file is `KEEP_TRAINING_FILES=ON_FAILURE`.

When you "Enable AI" for a table or view, the high-level sequence of steps performed by SQLDI is

1. Connect to Db2 vis JDBC T4 driver.
2. Gather Db2 catalog metadata about the object for which model training is to be performed.
3. Perform a SELECT against all the in-scope columns to bring the result set down to USS.
4. Start a Spark application to perform model training (initialize, pre-process data, train model).
5. Save the outputs of the model training are written as temporary files to a temporary directory.
6. Load the model table with these outputs (call a LOAD utility via SYSPROC.DSNUTILU)

If the training succeeds, and the model table is loaded, all these temporary datasets are deleted. This is a good default, because the load datasets can be quite large.

if the training fails (or if you set KEEP_TRAINING_FILES=ALL), the temporary datasets will be left in a temporary USS directory as shown below.

```
 /u/aidbadm/holinstance/temp/training/DSNAIDB_AIDB_DSNAIDB_CHURN_1660099214960
 ↪  >ls -al
total 52448
drwxr-xr-x   2 AIDBADM  SYS1        8192 Aug  9 21:42 .
drwxr-xr-x   7 AIDBADM  SYS1        8192 Aug  9 21:40 ..
-rw-r--r--   1 AIDBADM  SYS1    23350848 Aug  9 21:42
 ↪  DSNAIDB_AIDB_DSNAIDB_CHURN-320-10-5-normal-db2zos_zload.bin
-rw-r--r--   1 AIDBADM  SYS1     3034542 Aug  9 21:40
 ↪  DSNAIDB_AIDB_DSNAIDB_CHURN.txt
-rw-r--r--   1 AIDBADM  SYS1       62718 Aug  9 21:40 MONTHLYCHARGES.csv
-rw-r--r--   1 AIDBADM  SYS1         125 Aug  9 21:40
 ↪  MONTHLYCHARGES_output_minimums
-rw-r--r--   1 AIDBADM  SYS1       33361 Aug  9 21:40 TENURE.csv
-rw-r--r--   1 AIDBADM  SYS1         150 Aug  9 21:40 TENURE_output_minimums
-rw-r--r--   1 AIDBADM  SYS1       67932 Aug  9 21:40 TOTALCHARGES.csv
-rw-r--r--   1 AIDBADM  SYS1         125 Aug  9 21:40
 ↪  TOTALCHARGES_output_minimums
-rw-r--r--   1 AIDBADM  SYS1         592 Aug  9 21:42 load_emp_delim_ctl
-rw-r--r--   1 AIDBADM  SYS1      177619 Aug  9 21:40 vocab.txt
```

Inspecting the contents of these datasets (where possible) gives further insight into the workings of SQLDI.

The binary load dataset and the load control statement are fairly obvious. The .csv and minimums files contain additional information that is also stored in the model table, and is the source of some of the SQLDI analysis reports available in the SQLDI Web UI.

That concludes the Setup and Basic IVP.