# **Netezza Performance Server (NPS) Getting Started**

## Lab overview

This lab will introduce you to the Netezza Performance Server (NPS) Data Warehouse service. We will also refer to NPS as Netezza. You will learn how to access nzsql from the Netezza command line tool and perform these basic operations:

* Create a database
* Create a table
* Load a table
* Unload a table

These tasks provide an initial experience of interacting with the database engine. Other labs (outside of this workshop) are available to provide users experience with the advanced features of Netezza.

## Understanding Netezza Users

The NPS environment consists of two types of users - operating system users and database users. Operating system users can access the command line on the Netezza Linux host containers. Database users can access NPS to perform database-related tasks.

The Netezza Linux Container has the following default users:

|  |  |
| --- | --- |
| **Users** | **Capabilities** |
| root | Linux user with login permission to NPS  Has ownership to various NPS files in the installation directory  Usually not exposed to client users |
| nz | Linux user with login permission to NPS  Has ownership to various NPS files in the installation directory  Access to the Netezza command tools |
| admin | Database super-user (not a Linux user)  Full access to all database administration privileges and object |

## Connecting to the Netezza Performance Server Command Line

To access the Netezza host Linux container command line, you will use the Terminal application available on the Unified Desktop.

The Netezza host Linux container is running the service sshd which is configured to allow remote secure login. In our environment ssh is listening on port 22.

|  |  |
| --- | --- |
| nz | Note: In Cloud Pak for Data System with Netezza Performance Server, sshd is listening on port 51022. In Netezza on Cloud ssh is disabled as the cloud solution provides other mechanisms to access the command line tools. |

### **SSH to the Netezza Host Container using the Terminal**

1. Double-click the Terminal desktop icon.

A close up of a clock

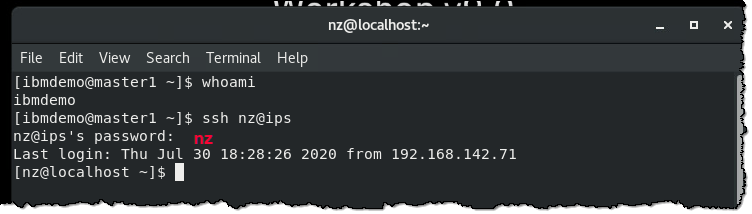
Description automatically generated

|  |  |
| --- | --- |
| A picture containing food, drawing, shirt  Description automatically generated  nz | Note: You can be user ibmdemo to ssh into node ips. |

1. Next, ssh into the Netezza Linux host container:

$ whoami

$ ssh nz@ips (password = nz)



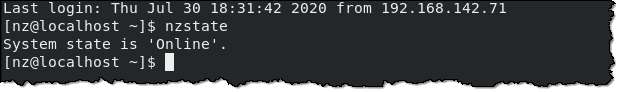
1. You are now logged into the Netezza host Linux container as user nz.

|  |  |
| --- | --- |
| A picture containing food, drawing, shirt  Description automatically generated  nz | Note: In this lab the root user is not needed on the Netezza host Linux container.  For reference, the root password is netezza. |

## Checking the state of NPS

1. Check the state of the Netezza engine by running nzstate.

$ nzstate



|  |  |
| --- | --- |
| A picture containing food, drawing, shirt  Description automatically generated  nz | If the Netezza system is offline (Stopped) then start Netezza with the nzstart command. |

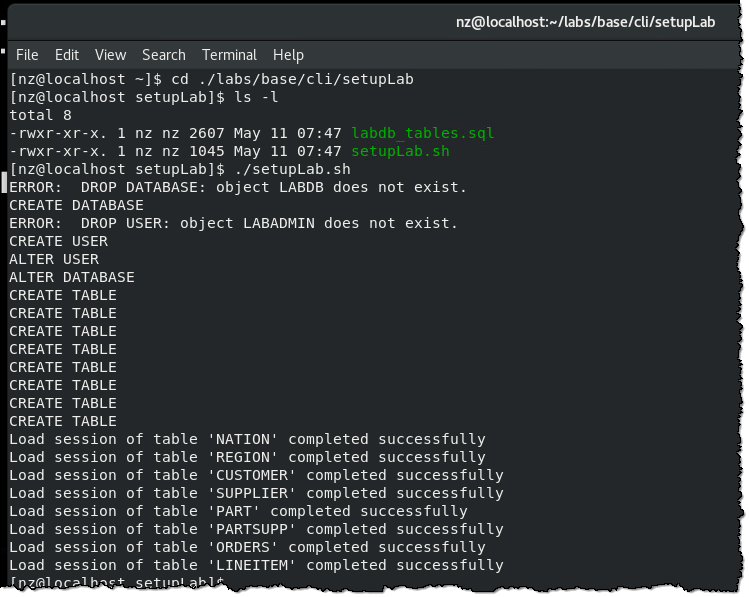
## Setting up the lab database

1. Execute the script to set up the database used in this lab:

$ cd ./labs/base/cli/setupLab/

$ ls -l

$ ./setupLab.sh



## Connecting to the Netezza system database using nzsql

Since you have not created any user or databases yet, you will connect to the default database as the default user, with the following credentials:

Database: system

Username: admin

Password: password

When issuing the nzsql command, the user supplies the user account, password and the database to connect to, using the example syntax below. Do not try to execute this command as it is just demonstrating the syntax:

|  |
| --- |
| **nzsql -d [db\_name] -u [user] -pw [password]** |

Alternatively, these values (db\_name, user, password) can be stored in the command shell and passed to the nzsql command when nzsql is issued without any arguments.

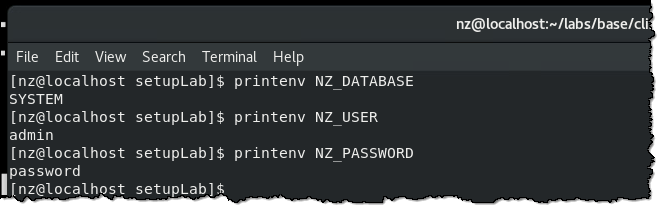
### **Verify the current database, user and password values**

1. Let’s verify the current database, user and password values stored in the command shell by issuing printenv NZ\_DATABASE, printenv NZ\_USER, and printenv NZ\_PASSWORD commands:

$ printenv NZ\_DATABASE

$ printenv NZ\_USER

$ printenv NZ\_PASSWORD

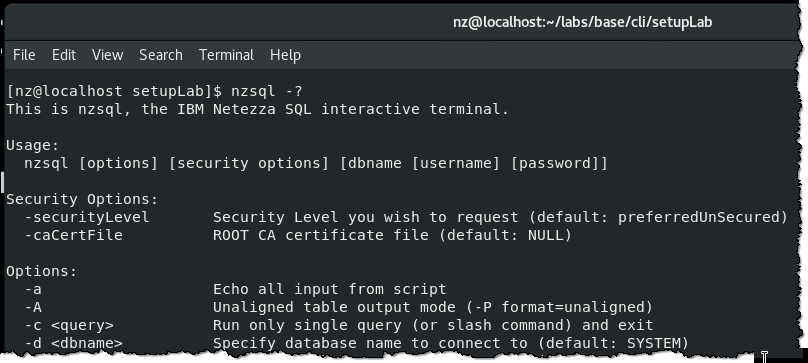


Since the current values correspond to our desired values, no modification is required. These environment variables are stored in the nz user’s .bashrc file.

### **Review nzsql options and using the console**

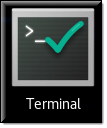
1. Let’s take a look at what options are available to start nzsql.

$ nzsql -?



1. The -? option will list the usage and all options for the nzsql command. In this next exercise, you will start nzsql without arguments.

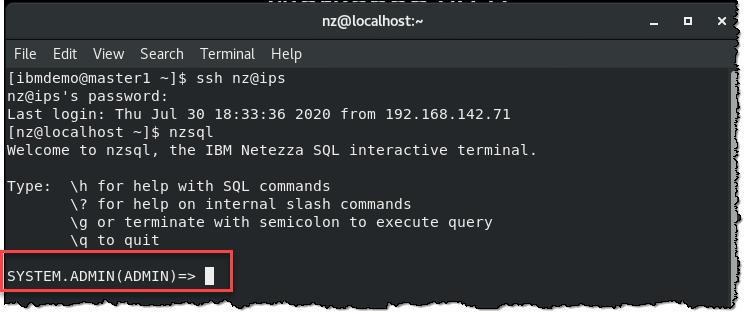
Open up another terminal window by double-clicking the Terminal desktop icon.



1. Next, ssh into the Netezza Linux host container:

$ ssh nz@ips (password = nz)

$ nzsql



This will bring up the nzsql prompt below that shows a connection to the system database as user admin:

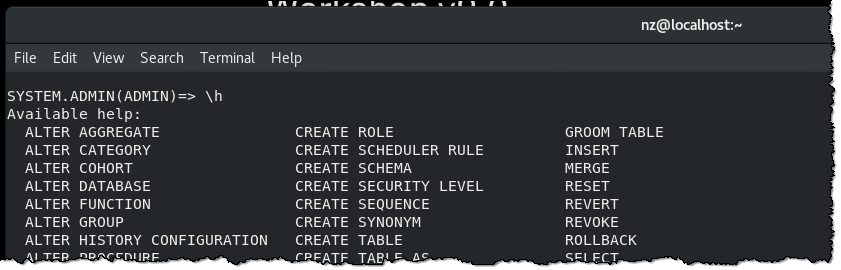
The nzsql console prompt format: DATABASE.SCHEMA(USER)

## Commonly used commands and SQL statements

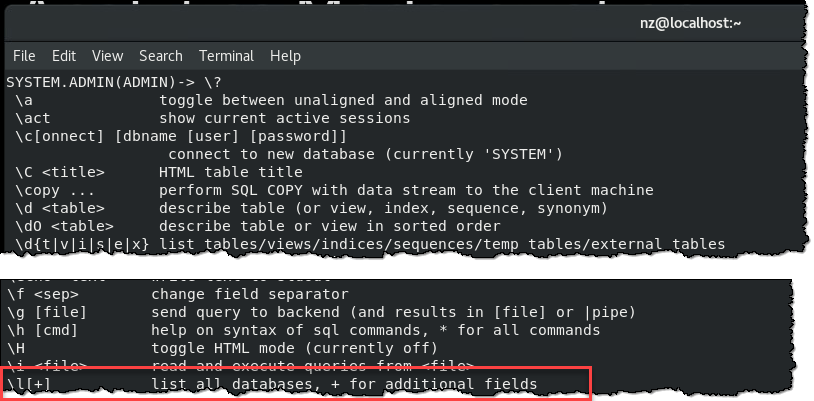
There are commonly used commands that start with \ which we will demonstrate in this section. First, you will run the two help commands to familiarize yourself with these handy commands. The \h command lists the available SQL commands, while the \? command is used to list the internal slash commands.

1. Execute these commands in the nzsql console:

SYSTEM.ADMIN(ADMIN)=> \h



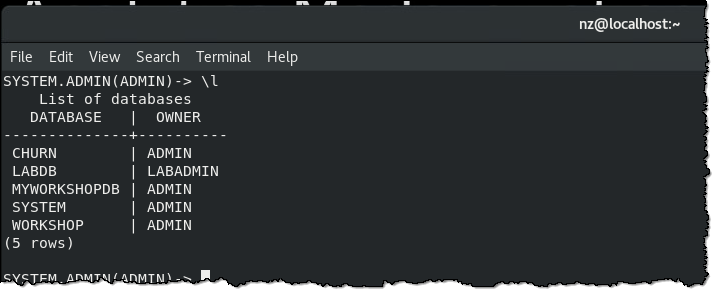
SYSTEM.ADMIN(ADMIN)=> \?



From the output of the \? command, you found the \l internal command you can use to find out all the databases.

1. Let’s look at the list of databases that have been created:

SYSTEM.ADMIN(ADMIN)=> \l

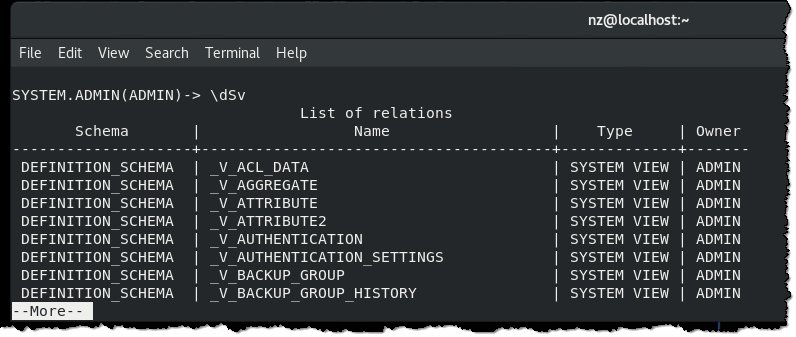


Secondly, you will use \dSv to find the system views within the system database.

|  |  |
| --- | --- |
| A picture containing food, drawing, shirt  Description automatically generated  nz | Note: There are system tables, but it is not recommended to directly access those tables as they can change from release to release and are restricted from the normal user. |

1. Let’s find the system views:

SYSTEM.ADMIN(ADMIN)=> \dSv



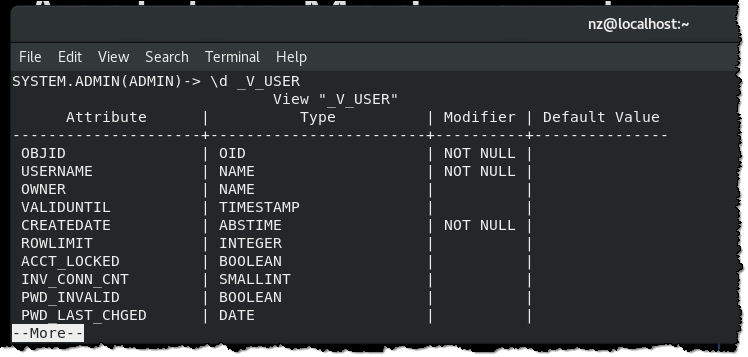
|  |  |
| --- | --- |
| A picture containing food, drawing, shirt  Description automatically generated  nz | Note: Press the space bar to scroll down the result set when you see --More-- on the screen. |

The list of the system views was truncated in the output above due to the length. Here are the primary views to investigate for DBAs/users new to Netezza:

|  |
| --- |
| **\_V\_USER**  **\_V\_SESSION\_V\_DATABASE**  **\_V\_SCHEMA**  **\_V\_TABLE**  **\_V\_VIEW**  **\_V\_ATTRIBUTE**  **\_V\_SEQUENCE**  **\_V\_SYNONYM**  **\_V\_FUNCTION**  **\_V\_AGGREGATE**  **\_V\_PROCEDURE** |

1. From the previous command, you can see that there is a user table called \_V\_USER. To find out what is stored in that table, you will use the describe command \d.

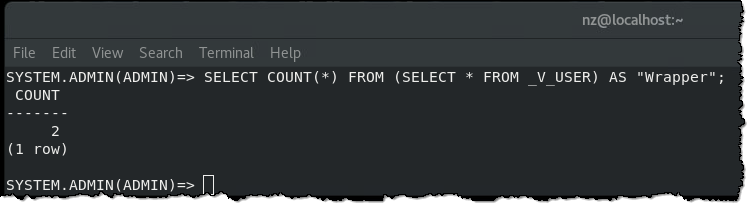
SYSTEM.ADMIN(ADMIN)=> \d \_V\_USER



This will return all the columns of the \_V\_USER system table.

1. Next, find the existing users stored in the table. In case too many rows are returned at once, you can first calculate the number of rows it contains by executing the following query:

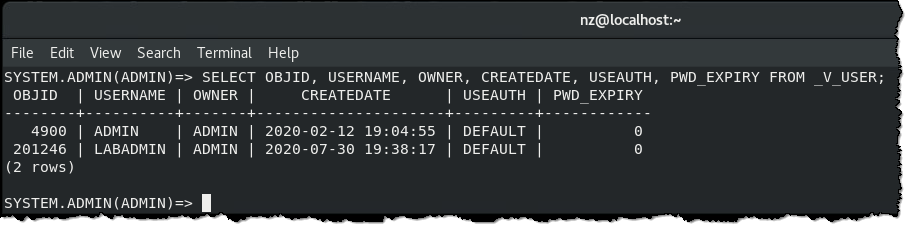
SYSTEM.ADMIN(ADMIN)=> SELECT COUNT(\*) FROM (SELECT \* FROM \_V\_USER) AS “Wrapper”;



1. The above query is essentially the same as SELECT COUNT (\*) FROM \_V\_USER. We have demonstrated the subselect syntax in case there is a complex query that needs to have the result set evaluated. The result should show the two entries in the user table.

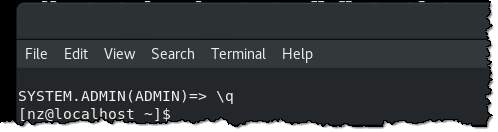
You can next enter the following query to list the usernames:

SYSTEM.ADMIN(ADMIN)=> SELECT OBJID, USERNAME, OWNER, CREATEDATE, USEAUTH, PWD\_EXPIRY FROM \_V\_USER;



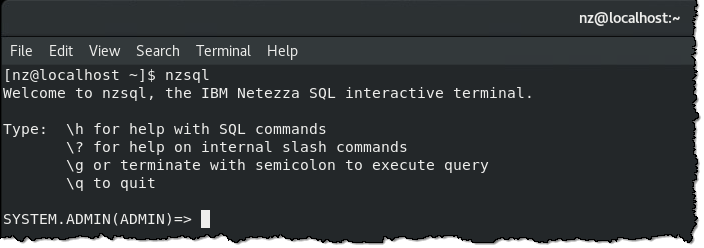
1. To exit nzsql, use the command \q to return to NPS.

SYSTEM.ADMIN(ADMIN)=> \q



1. Re-enter nzsql:

$ nzsql



You will be using the nzsql command line tool throughout the rest of this lab.

For a complete list of the nzsql internal slash options, see: <https://ibm.biz/nzsql-slash-options>.

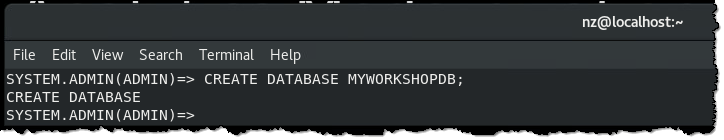
## Creating a database

A database in Netezza is a collection of entities such as schemas, tables, views, synonyms, and other objects.

The initial system consists of the SYSTEM database containing system tables and views. Initially, only the admin user can create databases, but the admin user can grant other users permission to create databases as described in security and access control. You cannot delete the system database. The admin user can also make another user the owner of a database, giving that user admin-like control over that database and its contents. Security is out of scope for this lab as you will perform all operations as the ADMIN database super-user.

1. The database creator becomes the default owner of the database. The owner can remove the database and all of its objects, even if other users own objects within the database.

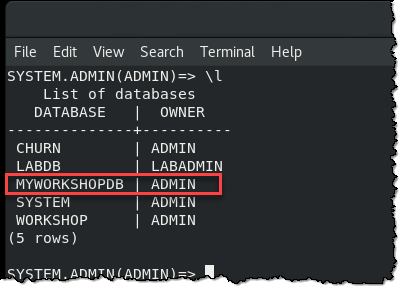
SYSTEM.ADMIN(ADMIN)=> CREATE DATABASE MYWORKSHOPDB;



|  |  |
| --- | --- |
| A picture containing food, drawing, shirt  Description automatically generated  nz | If database MYWORKSHOPDB already exists, you will get an error on the CREATE statement.  Simply drop it and then create it like this: |

1. List the databases in the Netezza system:

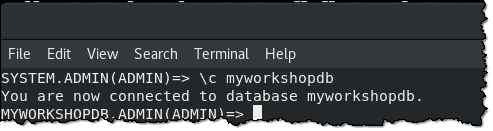
SYSTEM.ADMIN(ADMIN)=> \l



Notice the end of the \ commands do not require a terminator ; this is because the \ commands terminate automatically on a single line. SQL statements can span more than one line in nzsql.

1. Connect to the new database:

SYSTEM.ADMIN(ADMIN)=> \c myworkshopdb



## Creating a table

Now that you have a database, you can create a table to store your data. Tables are specific to a Netezza database and have one or more columns, each with attributes or data types.

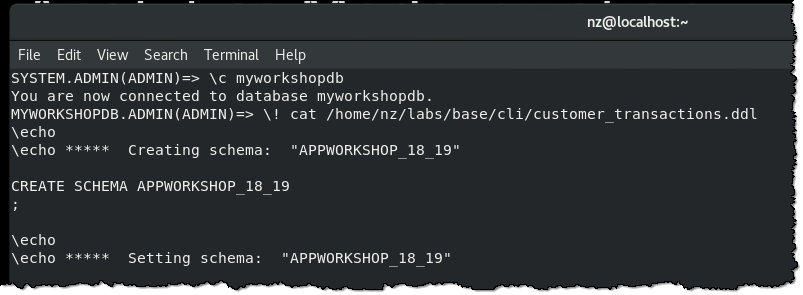
**Netezza supported data types:**

| **Data type** | **Example** | **Description** |
| --- | --- | --- |
| byteint smallint integer bigint | **120 0 256 1290985** | See [Integer data types](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_integer_data_types.html?view=kc). |
| numeric decimal | **-99.56 123.679** | See [Fixed-point data types](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_fixed_point_data_types.html?view=kc). |
| real  double precision | **-81293.35** | See [Floating-point data types](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_floating_point_data_types.html?view=kc). |
| char (*n*) | **salary** | See [Character strings](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_character_strings.html?view=kc). |
| varchar (*n*) | **this is a variable string** | See [Character strings](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_character_strings.html?view=kc). |
| boolean | **true** | This is an ASCII string that contains one of the following values:  True yes 1 t y  False no 0 f n |
| JSON |  | See [Document data types](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/dbuser/r_dbuser_document_data_types.html?view=kc) |
| date | **2002-02-04** | The date is an exact 4-byte data type. The system recognizes a range of dates that are composed of year, month, and day.  See [DateStyle option](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/r_load_datestyle.html?view=kc). |
| time | **01:59:45**  **23:00:01** | See [time data type](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_time.html?view=kc). |
| time with time zone | **01:15:33 -05** | See [Time with time zone (timetz) data type](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_time_zone.html?view=kc). |
| timestamp | **2002-02-04 01:15:33** | See [timestamp data type](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_timestamp.html?view=kc). |
| interval | **2 years 3 months 4 days** | See [Interval data types](https://www.ibm.com/support/knowledgecenter/SS5FPD_1.0.0/com.ibm.ips.doc/postgresql/load/c_load_interval_data_types.html?view=kc). |

1. To create our lab table, make sure you are connected to the database you created called MYWORKSHOPDB. A DDL file was prepared for you in /home/nz/labs/base/cli.

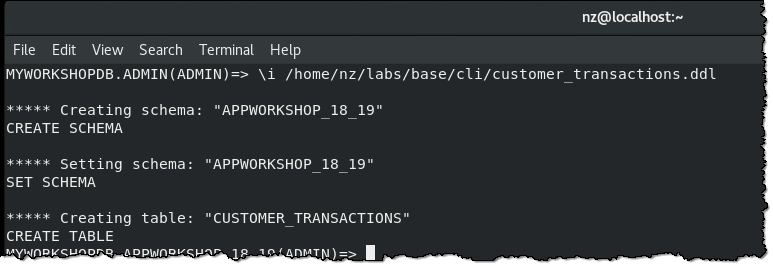
View the file while in nzsql by running an operating system command with \! option:

MYWORKSHOPDB.ADMIN(ADMIN)=> \! cat /home/nz/labs/base/cli/customer\_transactions.ddl



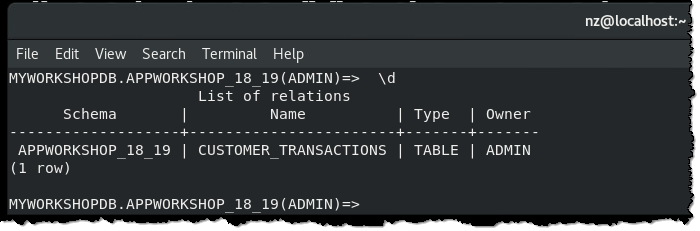
1. Now create the table with the \i option:

MYWORKSHOPDB.ADMIN(ADMIN)=> \i /home/nz/labs/base/cli/customer\_transactions.ddl



1. Check that the table exists: (Note: there is a space before the \d):

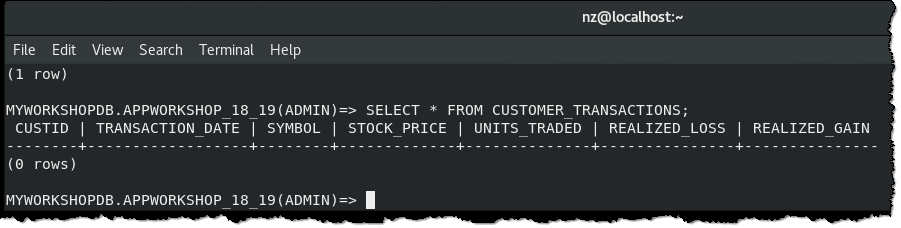
MYWORKSHOPDB.ADMIN(ADMIN)=> \d



Notice that the nzsql prompt schema changed, this was due to the SET SCHEMA command that was specified in customer\_transactions.ddl.

1. Select the table data:

MYWORKSHOPDB.APPWORKSHOP\_18\_19(ADMIN)=> SELECT \* FROM CUSTOMER\_TRANSACTIONS;



There is no data in the table yet because you just created it. In the next section, you will load some data.

## Loading data into a table

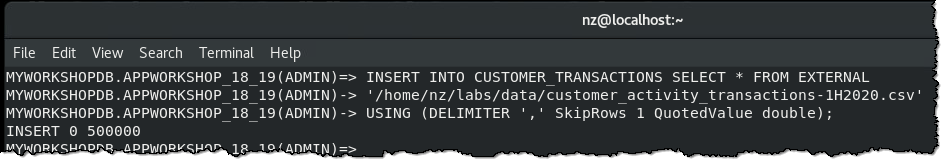
1. Load data into our CUSTOMER\_TRANSACTIONS table using an external table (a csv file.).

You will be using a three line command for doing this, terminated on the last line with the ; terminator:

MYWORKSHOPDB.APPWORKSHOP\_18\_19(ADMIN)=> INSERT INTO CUSTOMER\_TRANSACTIONS SELECT \* FROM EXTERNAL

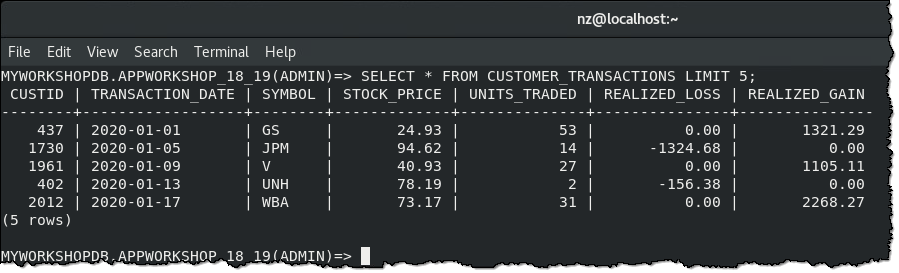
MYWORKSHOPDB.APPWORKSHOP\_18\_19(ADMIN)=> ‘/home/nz/labs/data/customer\_activity\_transactions-1H2020.csv’

MYWORKSHOPDB.APPWORKSHOP\_18\_19(ADMIN)=> USING (DELIMITER ‘,’ SkipRows 1 QuotedValue double);



1. Select 5 rows from the table CUSTOMER\_TRANSACTIONS:

MYWORKSHOPDB.APPWORKSHOP\_18\_19(ADMIN)=> SELECT \* FROM CUSTOMER\_TRANSACTIONS LIMIT 5;



## Lab conclusion

In this lab you learned how to use the nzsql tool to create a database, a table and load data into that table. These are the basic functions of the Netezza database.

**\*\* End of Lab 16 – Netezza Performance Server (NPS) Getting Started**

Lab by Dan Hancock, Edited by Burt Vialpando and Kent Rubin