Tutorial: Collaborative Data Science with Interactive Workstations and Databases

Contents

Introduction

This hands-on tutorial presents design patterns for collaborative data science using the featured precisionFDA pfda-ttyd and pfda-jupyterLab interactive workstation apps, and precisionFDA Databases. Through the development of the tutorial assets, precisionFDA's powerful capabilities for secure sharing and analysis of FISMA-Moderate authorized data are clearly demonstrated, and users will be empowered to develop their own collaborative regulatory data science use cases.

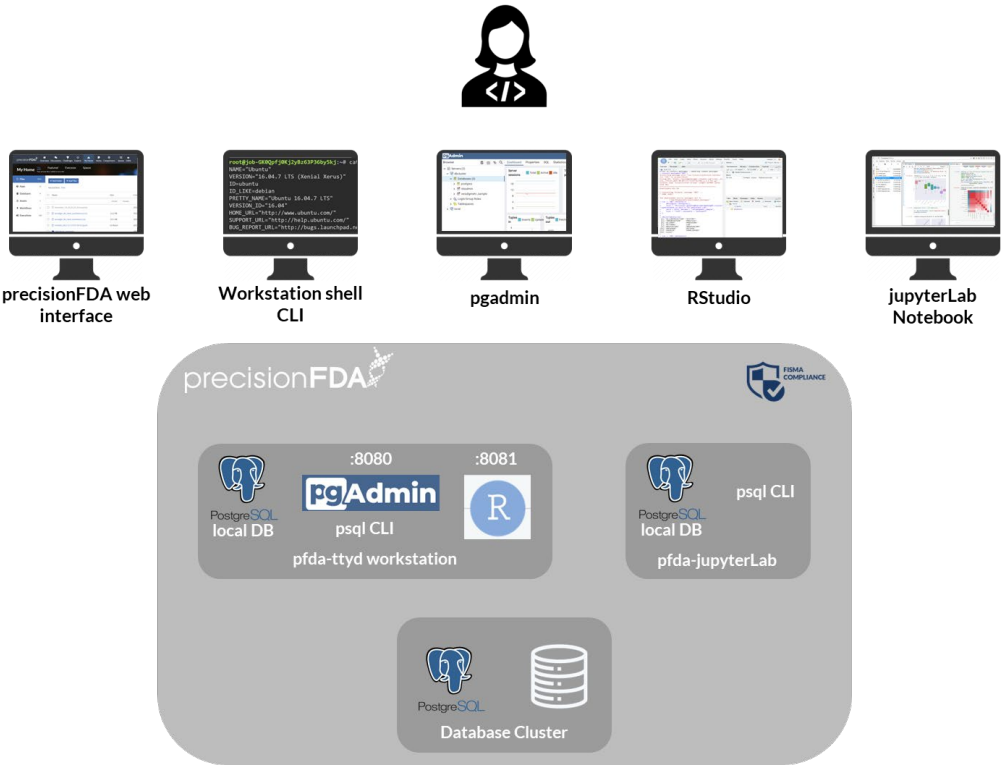
As always, keep in mind that all of these workstations, notebooks, and databases are strictly within the sole provenance of the user that launched them, and that in compliance with precisionFDA's FISMA authorization, the ability to deliver multi-user web services or databases is specifically not supported on precisionFDA.

Users can however use the power of the cloud to efficiently achieve their collaborative data science and bioinformatics objectives, and use the regulatory-grade platform to share the tools and results with full chain of provenance tracking. For cross-cutting analysis across FDA datasets, users will need to bring their data into the FDA's Intelligent Data Lifecycle Ecosystem (FiDLE).

Learning Objectives

Through this hands-on tutorial you will:

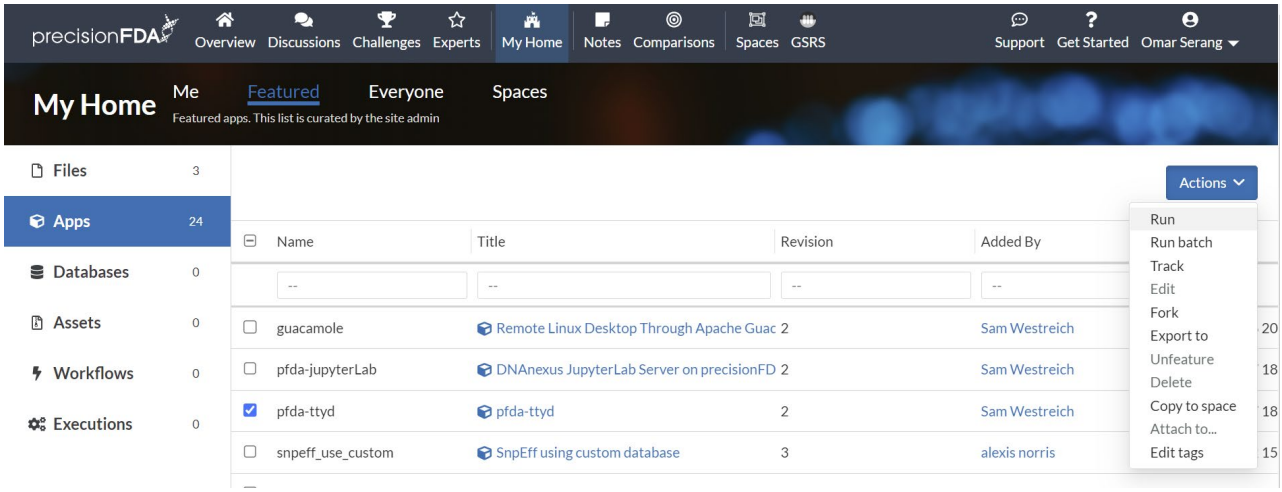
- Use the precisionFDA command line utility (pfda) to programmatically transfer files to and from precisionFDA and workstations and notebooks.
- Configure ttyd workstations to present web services on ports 8080 and 8081 for secure browser-based access with a rich UI.
- Launch a data analysis ttyd workstation with a local PostgreSQL database server, psql command line database client, pgadmin GUI database client, and RStudio configured with PostgreSQL access.
- Launch a precisionFDA Database cluster and access it from the data analysis workstation using psql and pgadmin to configure and install a database on the cluster from DDL and delimited data files.
- Use pgadmin to backup the cluster database to a precisionFDA file.
- Use pgadmin to restore the database backup to the data analysis workstation local database.
- Access the cluster and the workstation local databases from RStudio.
- Launch a jupyterLab workstation with a local PostgreSQL database server, and psql command line database client, and an example Python database analysis notebook.
- Use pgadmin to restore the database backup to the jupyterLab workstation local database.
- Access the cluster and the workstation local databases from a Python notebook.

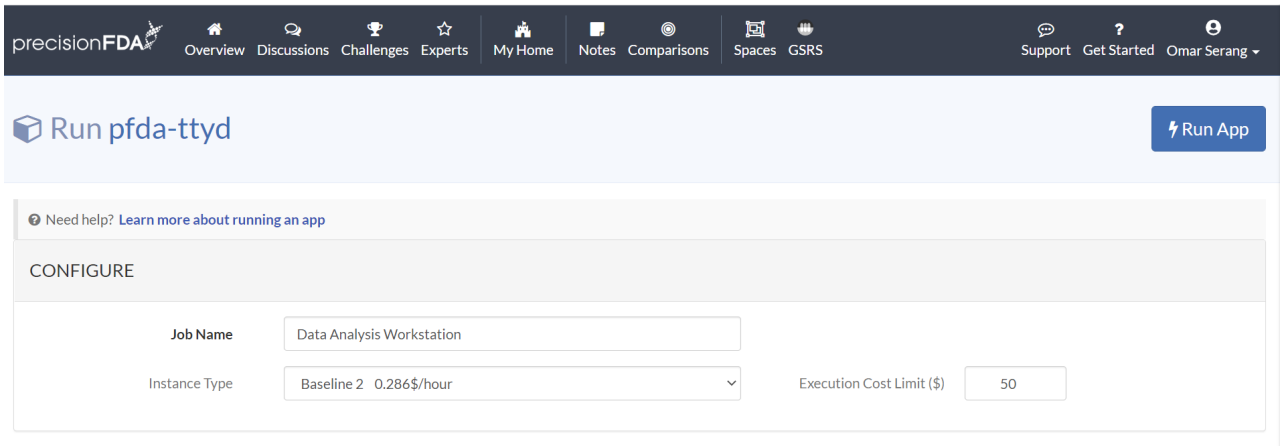


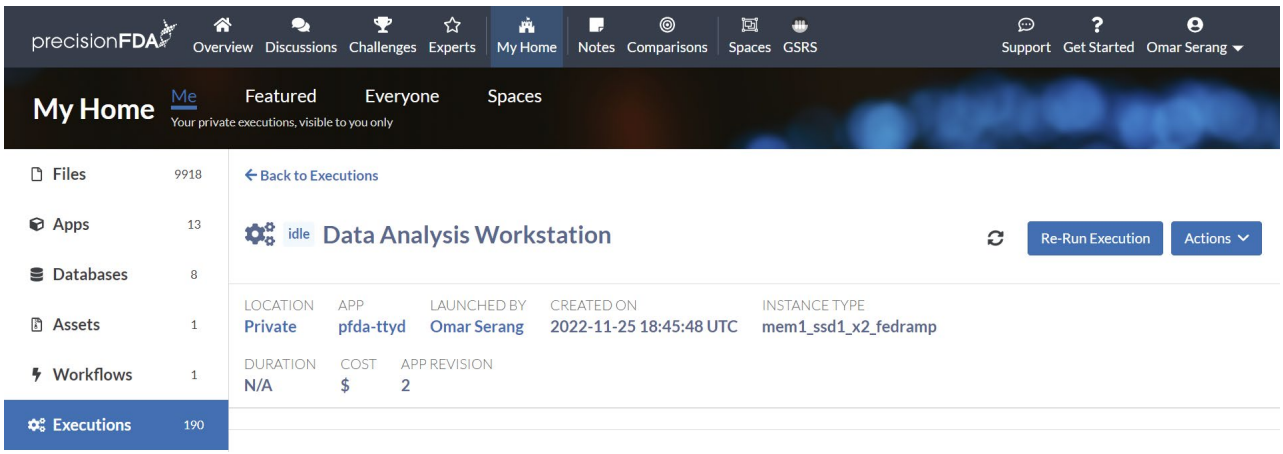
Build Data Analysis Workstation

Run the pfda-ttyd Featured App

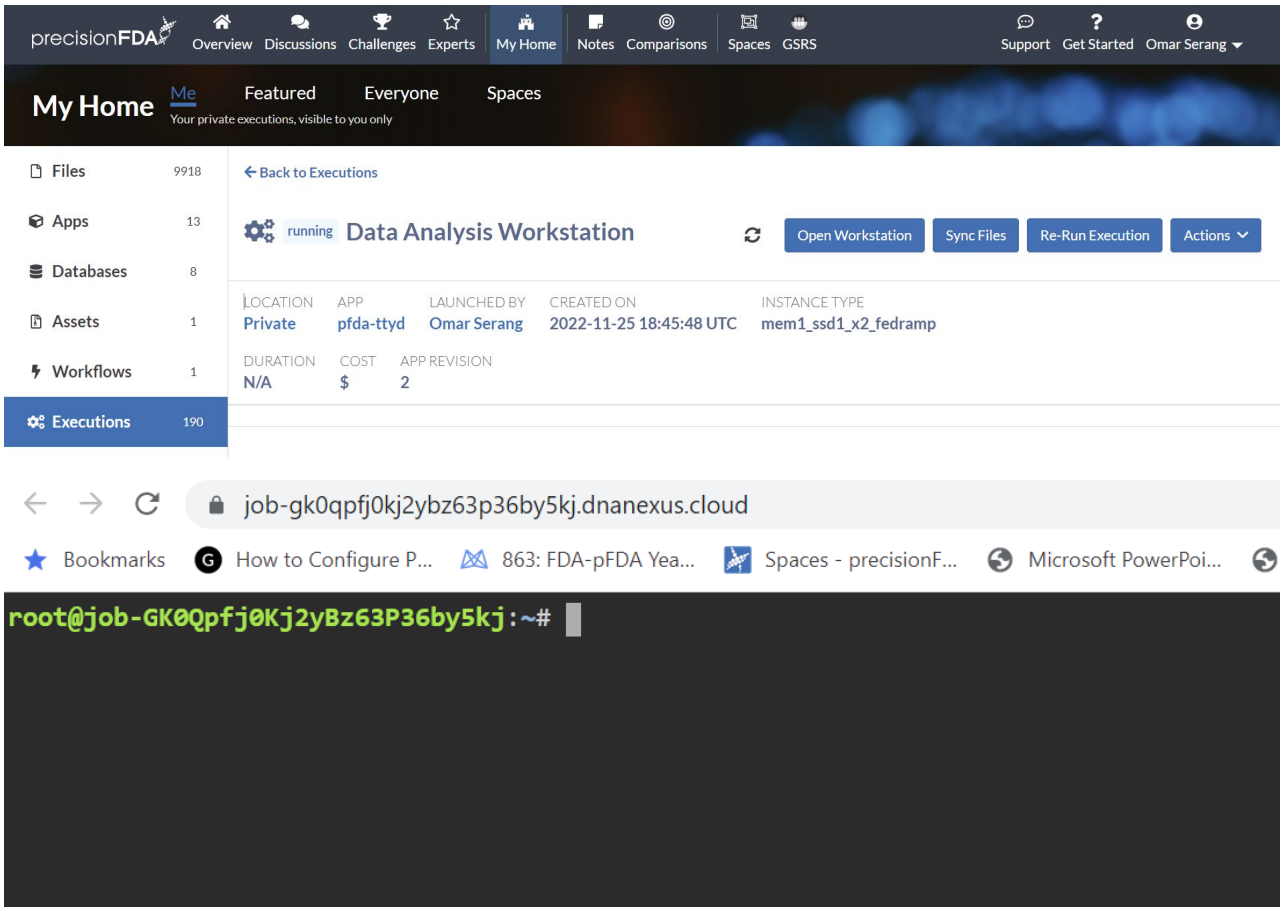
Using the smallest instance type, run the Data Analysis Workstation job.







Refresh the execution status using the button until the job is running and open the workstation.

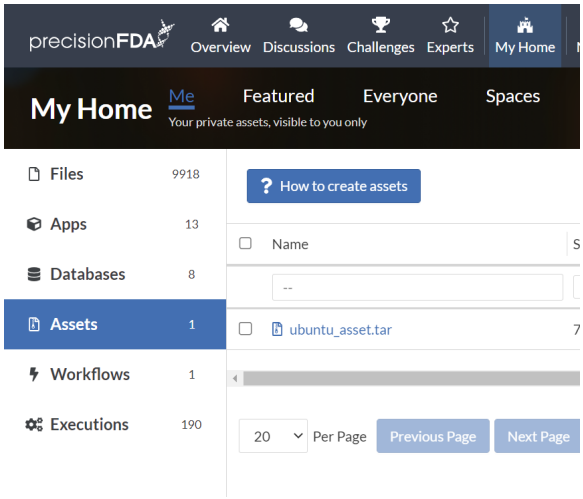


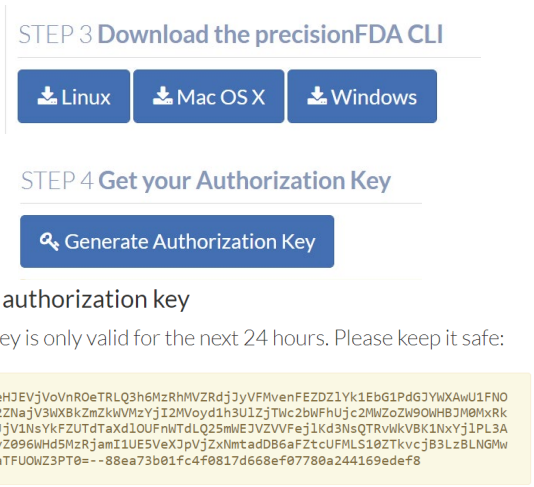
Use dx-get-timeout and dx-set-timeout to view and set the workstation application time-to-live after which it will self-terminate.

```
dx-set-timeout 1d
dx-get-timeout
0 days 23 hours 59 minutes 56 seconds
```

Download and Install the pfda CLI

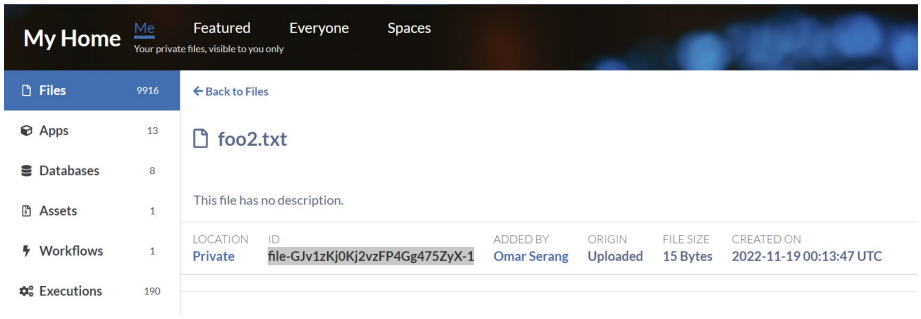
Under My Home Assets, click on the How to create assets button to find links to the precisionFDA CLI, and the button to generate the temporary authorization key that you'll use with the CLI.






```
# Install pfda CLI
wget https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.2.tar.gz
tar xf pfda-linux-2.2.tar.gz
mv pfda /usr/bin/
pfda --version
```

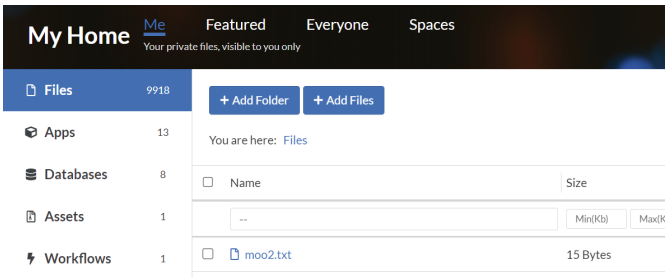
Copy a file ID and retrieve an authorization key to and download a file from precisionFDA to the workstation local FS.




```
key="…."

pfda download -key $key -file-id file-GJv1zKj0Kj2vzFP4Gg475ZyX-1
```

Upload a file from the workstation local filesystem to precisionFDA (note the key is cached).

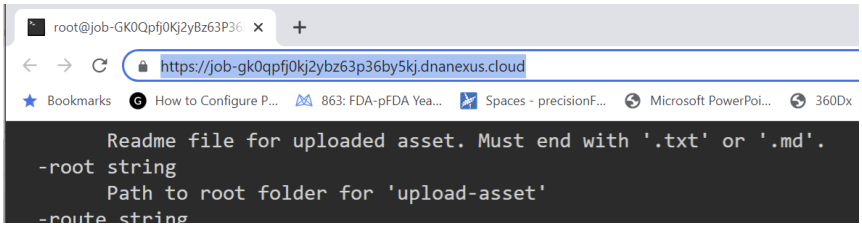
```
mv foo2.txt moo2.txt
pfda upload-file -file moo2.txt
```

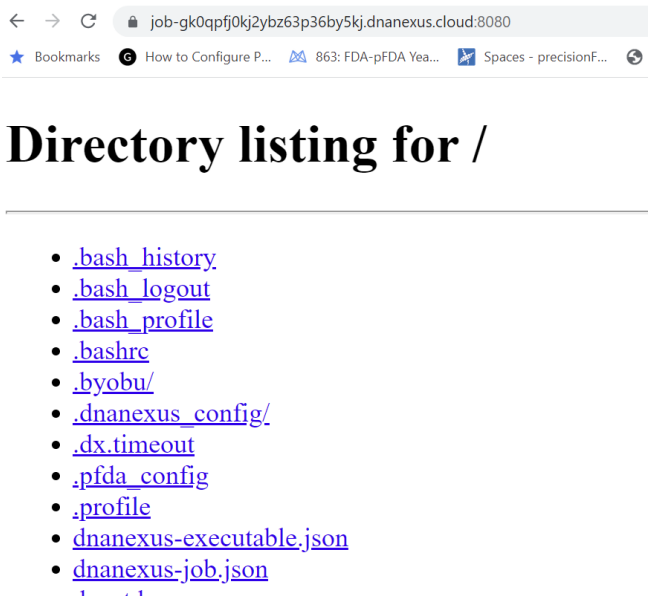


Present Test Web Servers on Ports 8080 and 8081

To see how a ttyd workstation can present web services to an external browser, start up test web servers on ports 8080 and 8081 and using the workstation job's URL, access the web services from your web browser (e.g. https://job-gk0qpfj0kj2ybz63p36by5kj.dnanexus.cloud:8080).

```
python3 -m http.server 8080 &
python3 -m http.server 8081 &
```





Kill the http.server jobs to free up ports 8080 and 8081 for pgadmin and RStudio (e.g. kill 1937).

Deploy Local PostgreSQL DB Server and CLI

Deploy a local PostgreSQL DB server on the Data Analysis workstation. Map the postgres port from the container to the workstation (host) OS.

```
# Install and start a postgreSQL server (and psql CLI)
```

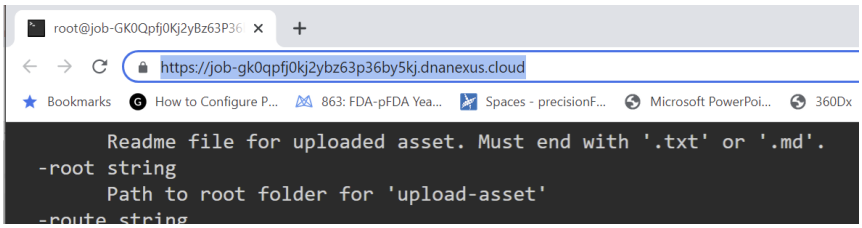
```
docker run --name postgres -e POSTGRES_PASSWORD=password -p 5432:5432 -d
postgres:13.4-buster

# Install postgres client
apt update
apt install postgresql-client -y

# Connect to local postgres db
PGPASSWORD="password" psql -h localhost -U postgres
```

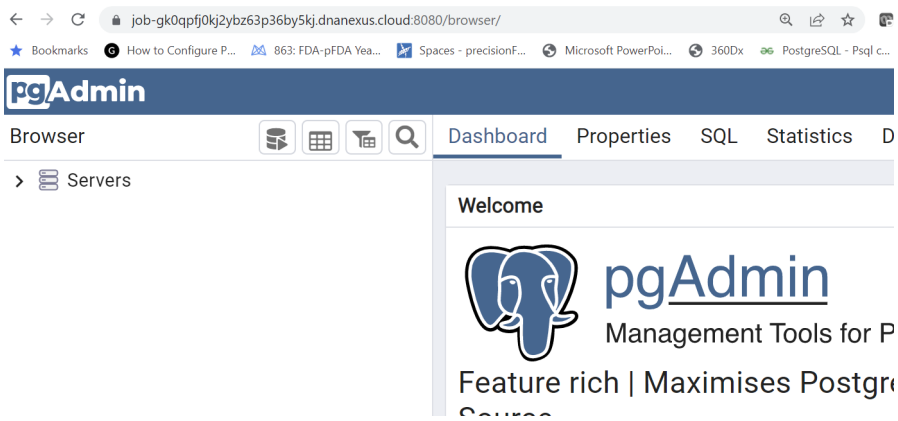
Deploy pgadmin and Connect to Local DB

pgadmin4 is deployed in a Docker container mapping the pgadmin web service port 80 to workstation port 8080. A directory is created on the workstation with the appropriate ownership to enable database backup files created in pgadmin to be copied from the container to the workstation.

```
# Create and configure host directory for backup files from pgadmin
mkdir /home/dnanexus/db_backups
sudo chown -R 5050:5050 db_backups/
sudo chmod ugo+w db_backups/

# Run pgadmin
docker run --name pgadmin -it -v
/home/dnanexus/db_backups:/home/dnanexus/db_backups -p 8080:80 -e
'PGADMIN_DEFAULT_EMAIL=user@domain.com' -e
'PGADMIN_DEFAULT_PASSWORD=password' -d dpage/pgadmin4
```

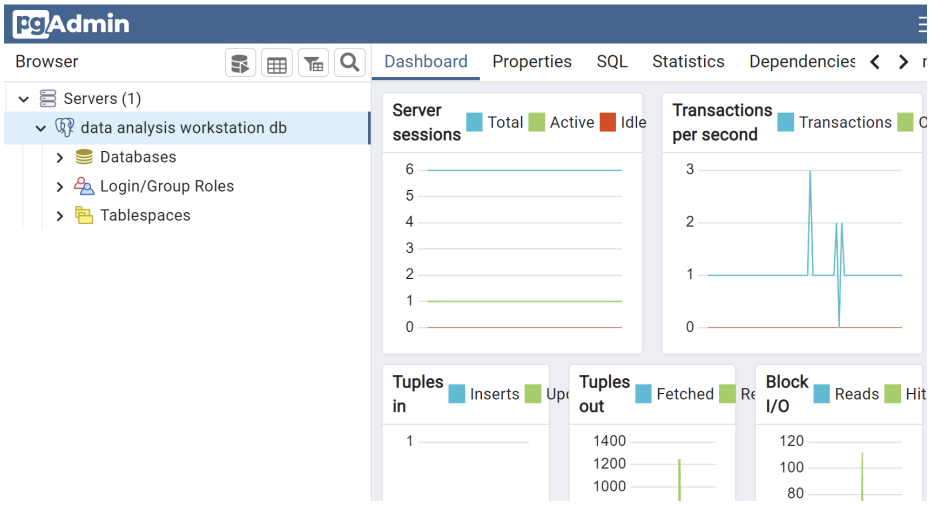
Access the pgadmin web service from your web browser (e.g. https://job-gk0qpfj0kj2ybz63p36by5kj.dnanexus.cloud:8080) with the specified credentials (user@domain.com, password).





To connect pgadmin in the container to the postgres database server port on the host, first obtain the docker0 interface IP address. This will be used in place of localhost in pgadmin (since localhost in pgadmin refers to the container local host). Add the workstation local database as a new server (data analysis workstation db) using the docker0 address (user *postgres*, password *password*).

```
ip addr show docker0

2: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UP group default
    link/ether 02:42:cd:c8:f1:0e brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
```

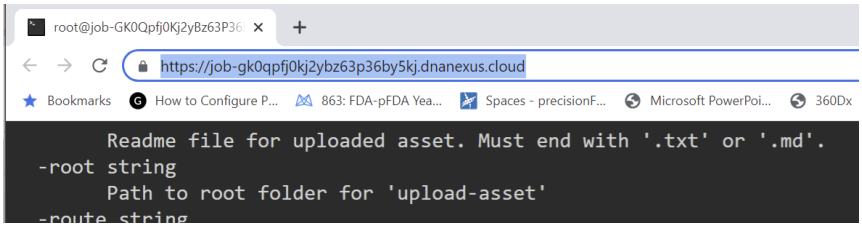


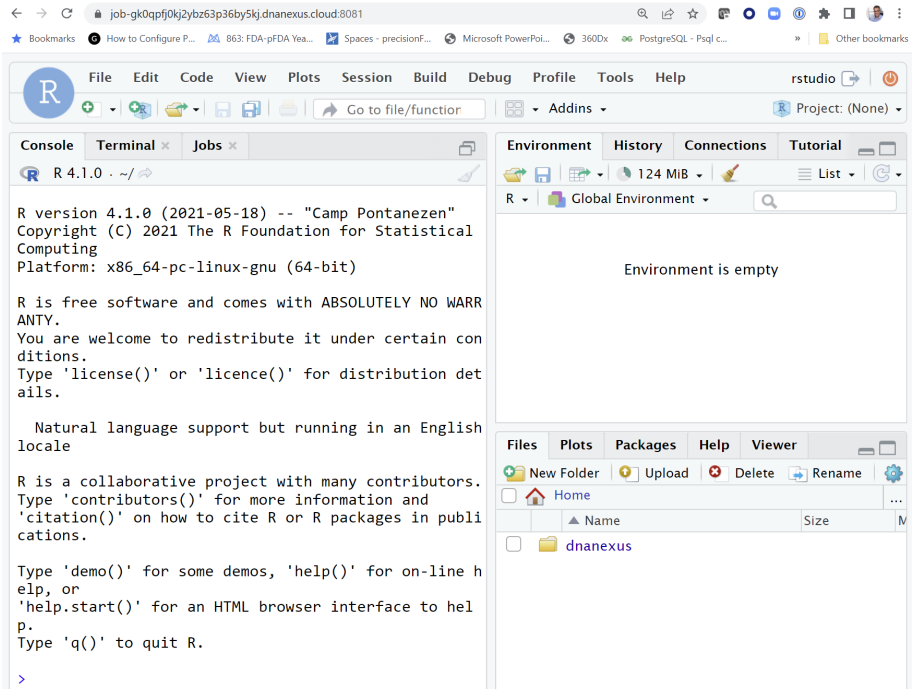
Deploy RStudio

RStudio is deployed in a Docker container mapping the RStudio web service port 8787 to host port 8081. A directory is created on the workstation with the appropriate ownership to enable database backup files created in pgadmin to be copied from the container to the workstation.

```
# Deploy RStudio
docker pull rocker/rstudio:4.1.0
docker run --name rstudio -p 8081:8787 -v
/home/dnanexus:/home/rstudio/dnanexus -e DISABLE_AUTH=true -d
rocker/rstudio:4.1.0
```

Access the RStudio web service from your web browser (e.g. https://job-gk0qpfj0kj2ybz63p36by5kj.dnanexus.cloud:8081).





Install the RPostgres packages and test access to the workstation local database. In the R Studio console:

```
# Install RPostgres
install.packages("RPostgres")

# Connect to local database
library(DBI)
con <- DBI::dbConnect(
    RPostgres::Postgres(),
    host = "172.17.0.1",
      port = 5432, dbname = "postgres",
    user = "postgres", password = "password"
)
# List the tables in db postgres
dbListTables(con)
```

Since there are no tables in the postgres database, the response is **`character(0)`**. Let's add two tables using psql and run the same R query. In psql on the data analysis workstation:

```
CREATE TABLE public."PATIENT" (
    patient_id bigint NOT NULL,
    name character varying,
    gender character varying,
    zip character varying,
    country character varying,
    created_date date
);

CREATE TABLE public."OBSERVATION" (
    observation_id bigint NOT NULL,
```

```
    patient_id bigint,
    observation_name character varying,
    loinc character varying,
    created_date date
);
```

The query in the RStudio console now shows the two new tables.

```
dbListTables(con)
[1] "PATIENT"     "OBSERVATION"
```

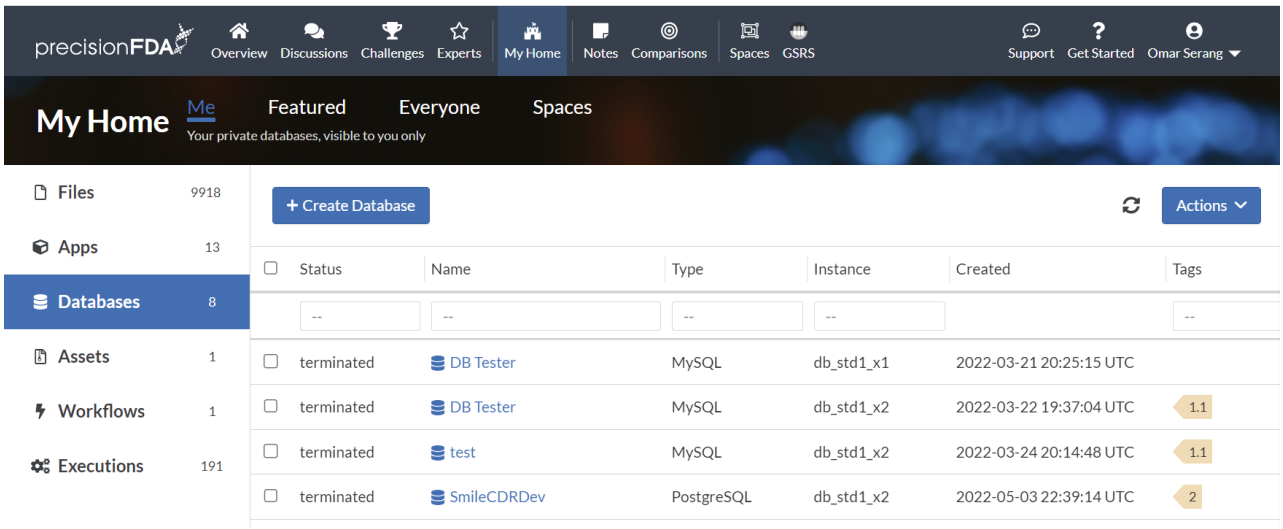
Let's drop the tables since we will be populating this database from a backup a later stage in this tutorial. In psql on the data analysis workstation:

```
DROP TABLE public."PATIENT"
DROP TABLE public."OBSERVATION"
```

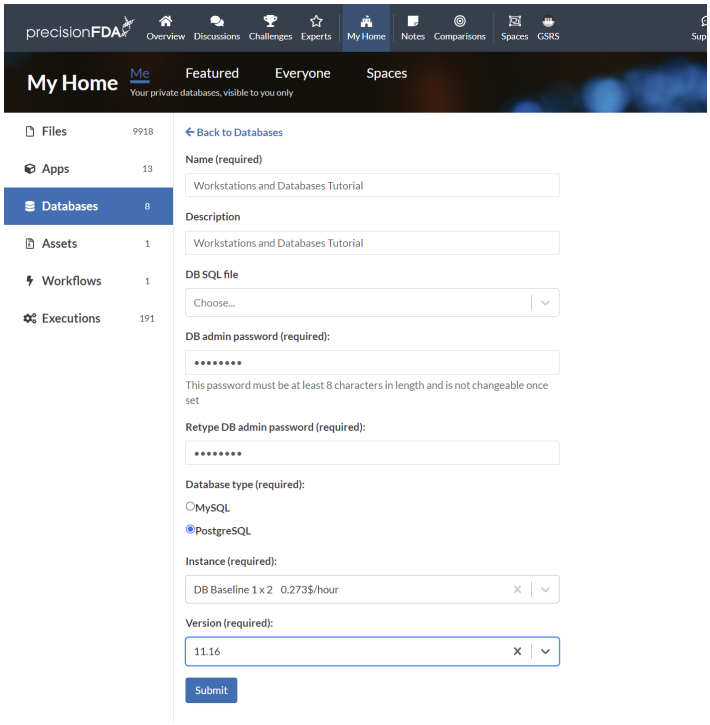
Deploy a precisionFDA Database Cluster

Create the Database

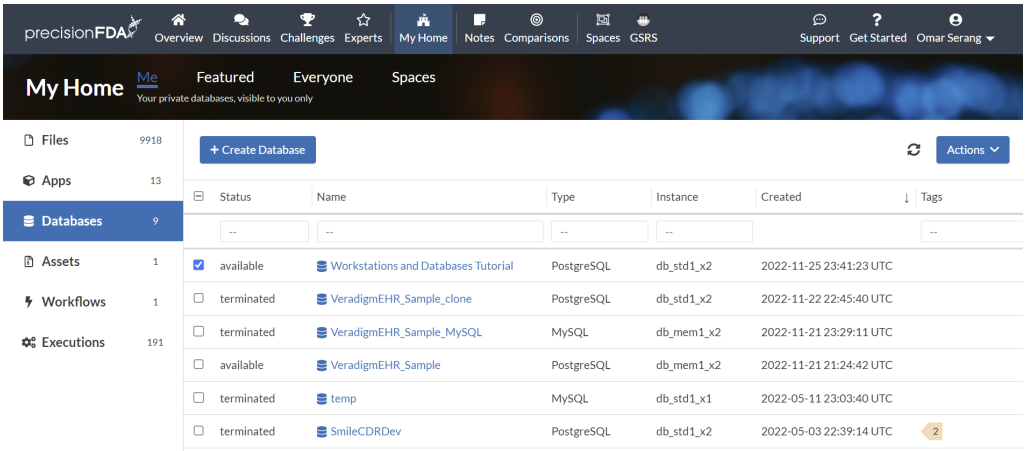
Select the Databases tab in My Home and click the Create Database button.



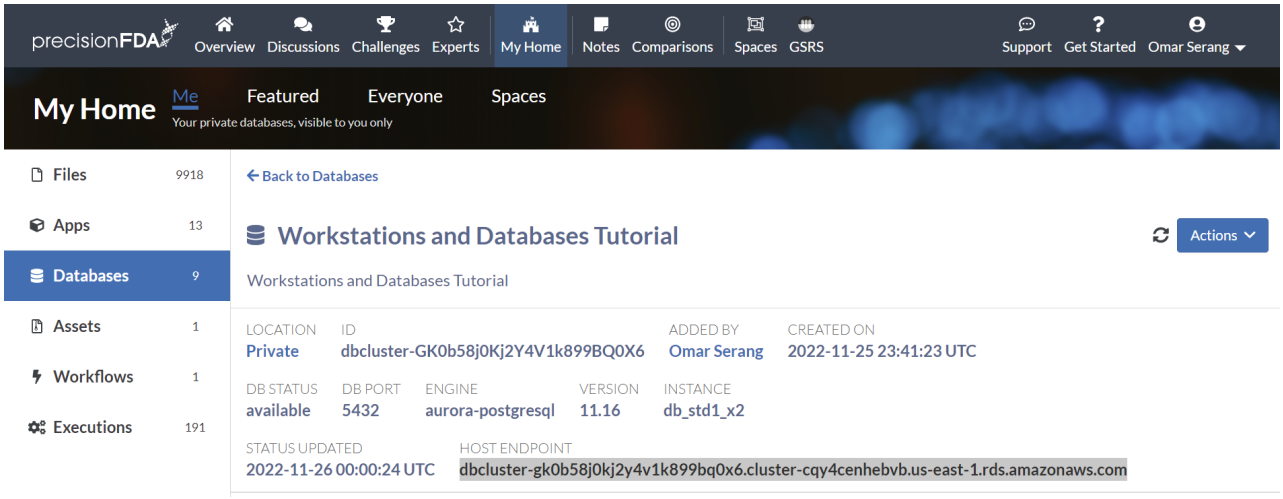
Create a “Workstations and Databases Tutorial” database, “password”, PostgreSQL 11.16 on the smallest available database instance type, and click the Submit button.



Refresh the database status using the  button until the database is available.

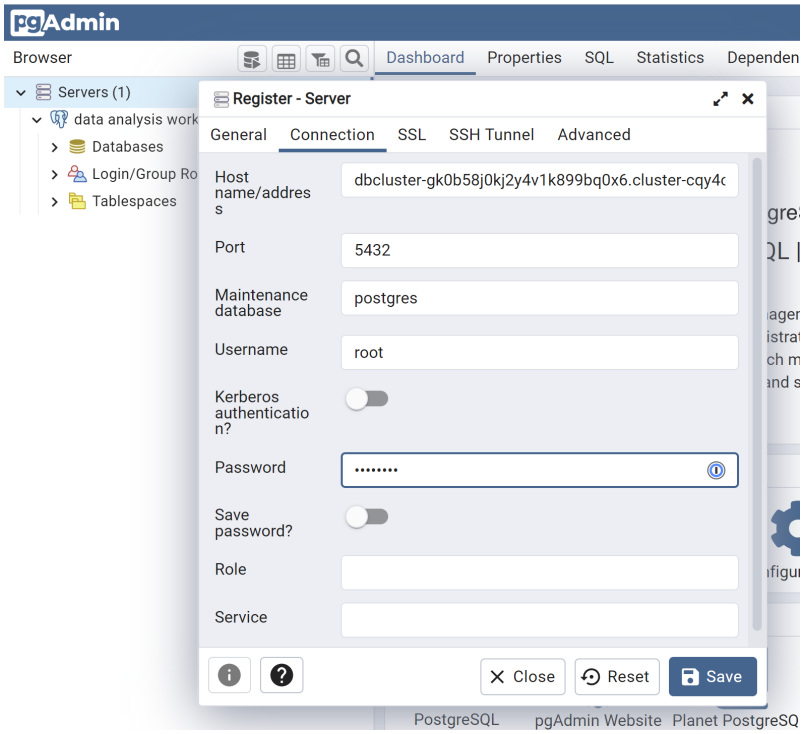


Click on the Workstations and Databases Tutorial database to open the detail page and copy the host endpoint URL.

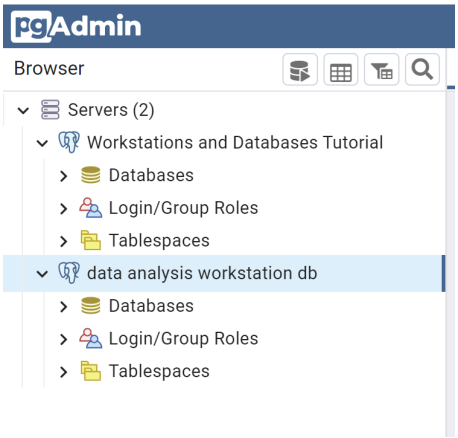


Connect to the cluster DB from pgadmin

In the pgadmin web service, add a new server for the Workstations and Databases Tutorial DB cluster using the host endpoint, user root, and the password specified when the database was created.



Note that we now have connections to both the local database on the data analysis workstation, and the cluster database.



Create a new database and tables

Connect to the cluster database from psql in the data analysis workstation shell.

```
psql --host=dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebvb.us-east-1.rds.amazonaws.com --username=root -d postgres
```

Using psql, create a new database.

```
-- Database: workstations_and_databases_tutorial_db
CREATE DATABASE workstations_and_databases_tutorial_db
    WITH
    OWNER = root
    ENCODING = 'UTF8'
    CONNECTION LIMIT = -1
    IS_TEMPLATE = False;
```

Connect to the new database and create two tables.

```
\c workstations_and_databases_tutorial_db;

psql (9.5.25, server 11.16)
WARNING: psql major version 9.5, server major version 11.
         Some psql features might not work.
SSL connection (protocol: TLSv1.2, cipher: ECDHE-RSA-AES128-GCM-SHA256, bits: 128, compression: off)
You are now connected to database "workstations_and_databases_tutorial_db" as user "root".
workstations_and_databases_tutorial_db=>

CREATE TABLE public."PATIENT" (
    patient_id bigint NOT NULL,
    name character varying,
    gender character varying,
    zip character varying,
    country character varying,
    created_date date
```

```
);

CREATE TABLE public."OBSERVATION" (
    observation_id bigint NOT NULL,
    patient_id bigint,
    observation_name character varying,
    loinc character varying,
    created_date date
);

\dt
          List of relations
 Schema |    Name     | Type  | Owner
--------+-------------+-------+-------
 public | OBSERVATION | table | root
 public | PATIENT     | table | root
(2 rows)
```

Load the cluster database from delimited text files

Although the workflow illustrated here may seem over-engineered for loading two data files, the techniques presented here were used to reliably and efficiently transfer tens of thousands of files and 15+ TB of data to precisionFDA.

In the data analysis workstation shell, create a datafiles directory

```
mkdir datafiles
```

Create and upload delimited data files

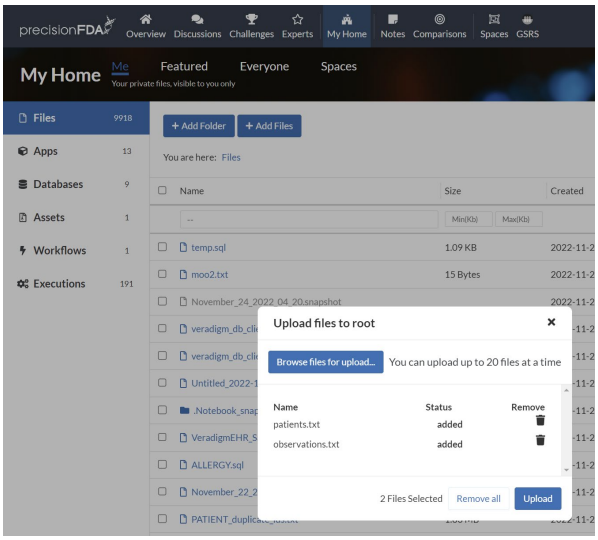
On your local client (i.e. laptop), create file **`patients.txt`** with the following content:

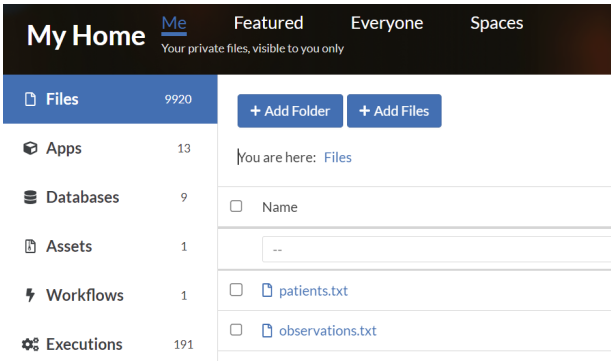
```
12345|Fred Foobar|M|94040|USA|2022-10-25
12346|Mary Merry|F|94040|USA|2022-09-24
12347|Barney Rubble|M|94040|USA|2022-08-23
```

Create file **`observations.txt`** with the following content:

```
9870|12345|Annual check up|66678-4|2022-11-01
9871|12345|Emergency|LG32756-5|2022-11-02
9872|12346|Clinic visit|66678-4|2022-11-03
9873|12347|Lab results|74418-5|2022-11-04
9874|12347|Post-op checkup|65375-8|2022-11-05
```

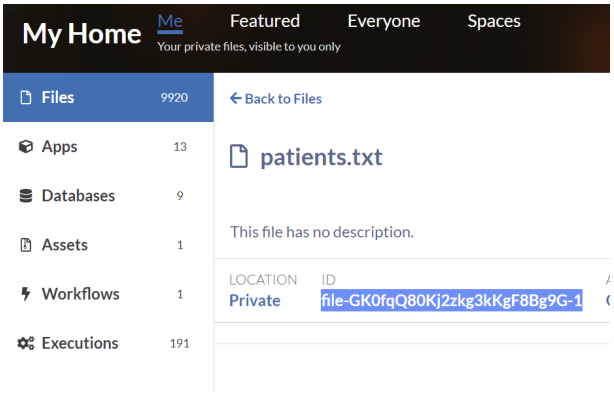
In My Home / Files use the Add Files button to upload the two files to your private area.





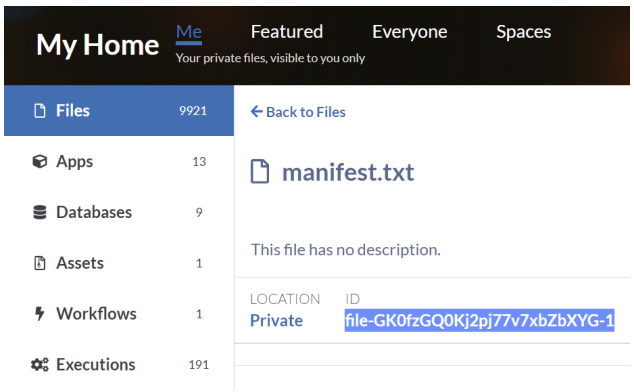
Create and upload a manifest of data file IDs

Click into patients.txt and observations.txt details pages and copy their file IDs into a file named manifest.txt file on your local client.



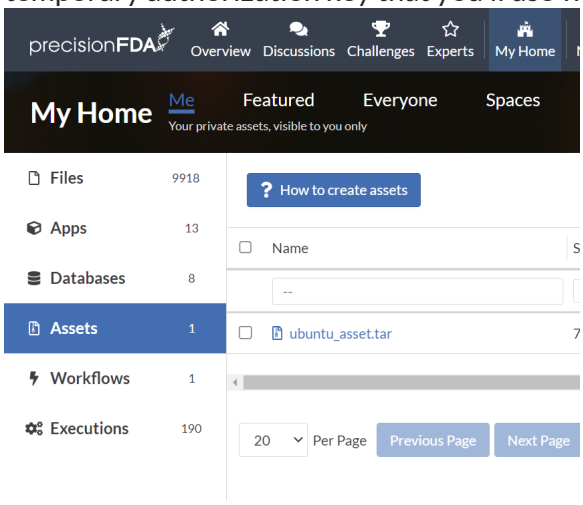

```
-- manifest.txt
file-GK0fqQ80Kj2zkg3kKgF8Bg9G-1
file-GK0fqGQ0Kj2gBZjxF24493PY-1
```

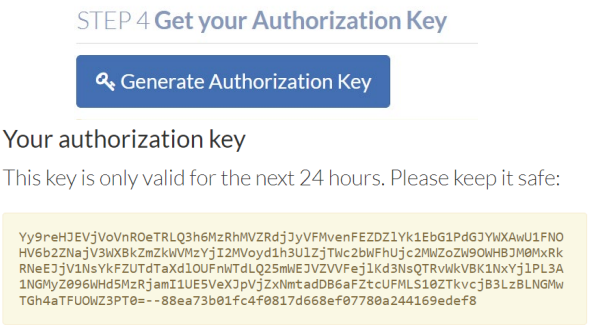
Use the Add Files button to upload the `manifest.txt` file to your private area. Click into the details for the uploaded file and copy the file ID.



Download the files in the manifest to the Data Analysis Workstation

Under My Home Assets, click on the How to create assets button to find the button to generate the temporary authorization key that you'll use with the CLI.





Using pfda CLI in the data analysis workstation shell, download the manifest.txt file to the workstation filesystem.

`key="..."`

```
pfda download -key $key -file-id file-GK0fzGQ0Kj2pj77v7xbZbXYG-1

ls -l
-rw-r--r-- 1 root root 66 Nov 26 01:58 manifest.txt
```

Iterate through manifest and download data files

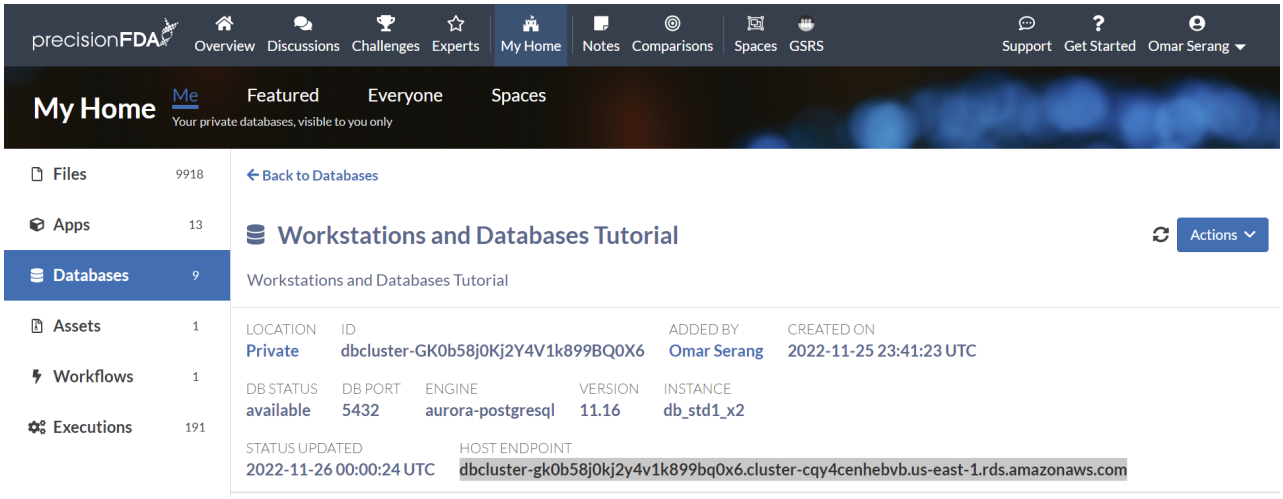
In the data analysis workstation shell install and run dos2unix on the manifest.txt file to ensure there are no cross-OS end-of-line issues.

```
cd ~/datafiles
apt install dos2unix
dos2unix manifest.txt

for FILE in $(cat manifest.txt); do pfda download -key $key -file-id
$FILE; done

ls
manifest.txt  observations.txt  patients.txt
```

Copy the data into the cluster DB tables

Connect to the workstations_and_databases_tutorial_db cluster database

Using the database host endpoint, connect to the workstations_and_databases_tutorial_db cluster database using psql on the data analysis workstation:




```
psql --host=dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebvb.us-
east-1.rds.amazonaws.com --username=root -d
workstations_and_databases_tutorial_db

workstations_and_databases_tutorial_db=>
```

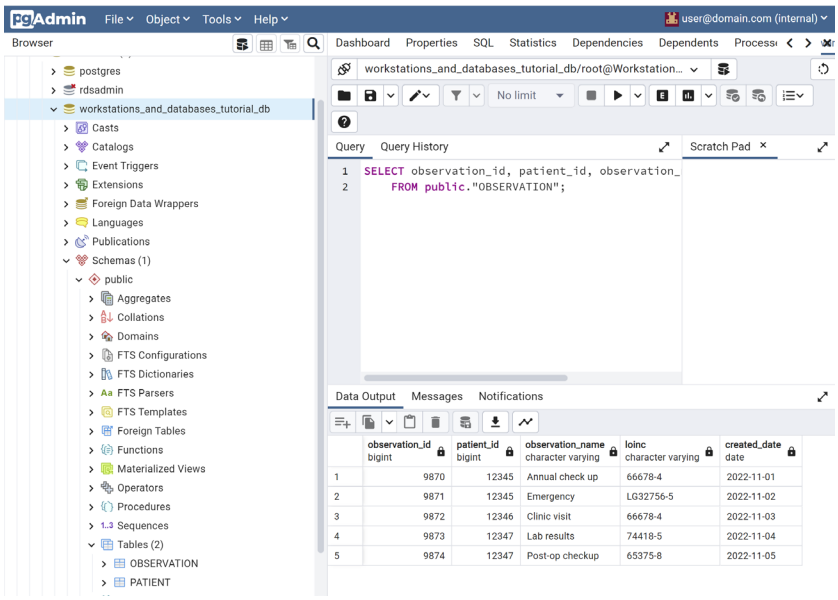
Copy the patients and observations data into the cluster DB

In psql:

```
\copy public."PATIENT" from '/home/dnanexus/datafiles/patients.txt'
delimiter '|' NULL ''
```

```
\copy public."OBSERVATION" from
'/home/dnanexus/datafiles/observations.txt' delimiter '|' NULL ''

select * from public."PATIENT";
patient_id |     name      | gender |  zip  | country | created_date
-----------+---------------+--------+-------+---------+--------------
      12345 | Fred Foobar   | M      | 94040 | USA     | 2022-10-25
      12346 | Mary Merry    | F      | 94040 | USA     | 2022-09-24
      12347 | Barney Rubble | M      | 94040 | USA     | 2022-08-23

select * from public."OBSERVATION";
observation_id | patient_id | observation_name |   loinc   |created_date
---------------+------------+------------------+-----------+-----------
          9870 |      12345 | Annual check up  | 66678-4   | 2022-11-01
          9871 |      12345 | Emergency        | LG32756-5 | 2022-11-02
          9872 |      12346 | Clinic visit     | 66678-4   | 2022-11-03
          9873 |      12347 | Lab results      | 74418-5   | 2022-11-04
          9874 |      12347 | Post-op checkup  | 65375-8   | 2022-11-05
```

Observe the new tables and data in the pgadmin Workstations and Databases Tutorial server connection.



Connect RStudio to the cluster DB

In the RStudio console:

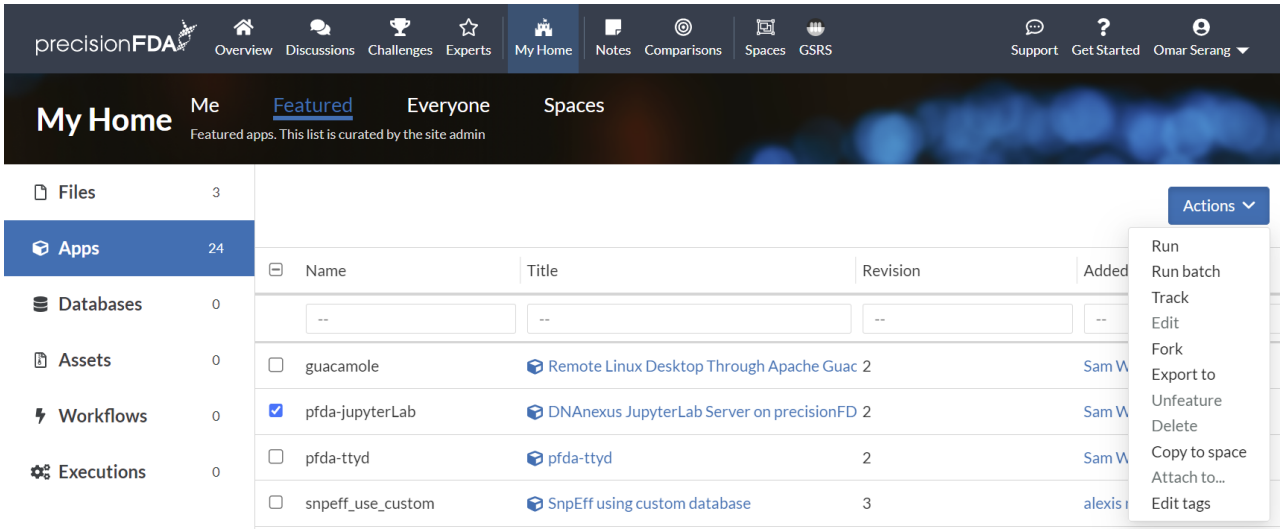
```
library(DBI)
con <- DBI::dbConnect(
    RPostgres::Postgres(),
    host = "dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebvb.us-
east-1.rds.amazonaws.com",
     port = 5432, dbname = "workstations_and_databases_tutorial_db",
    user = "root", password = "password"
```

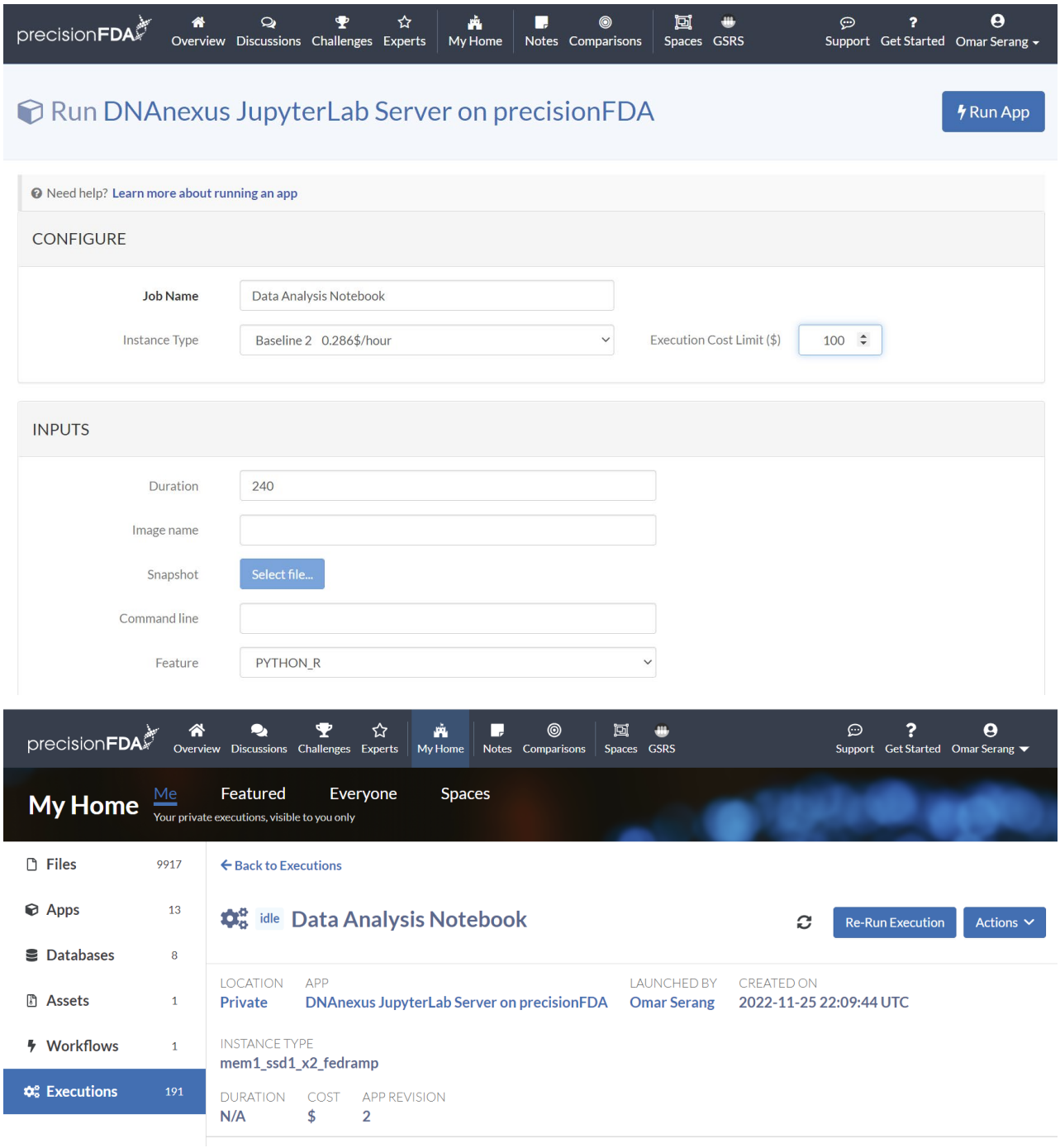
```
)
dbListTables(con)
[1] "OBSERVATION" "PATIENT"
```

Build Data Analysis Notebook

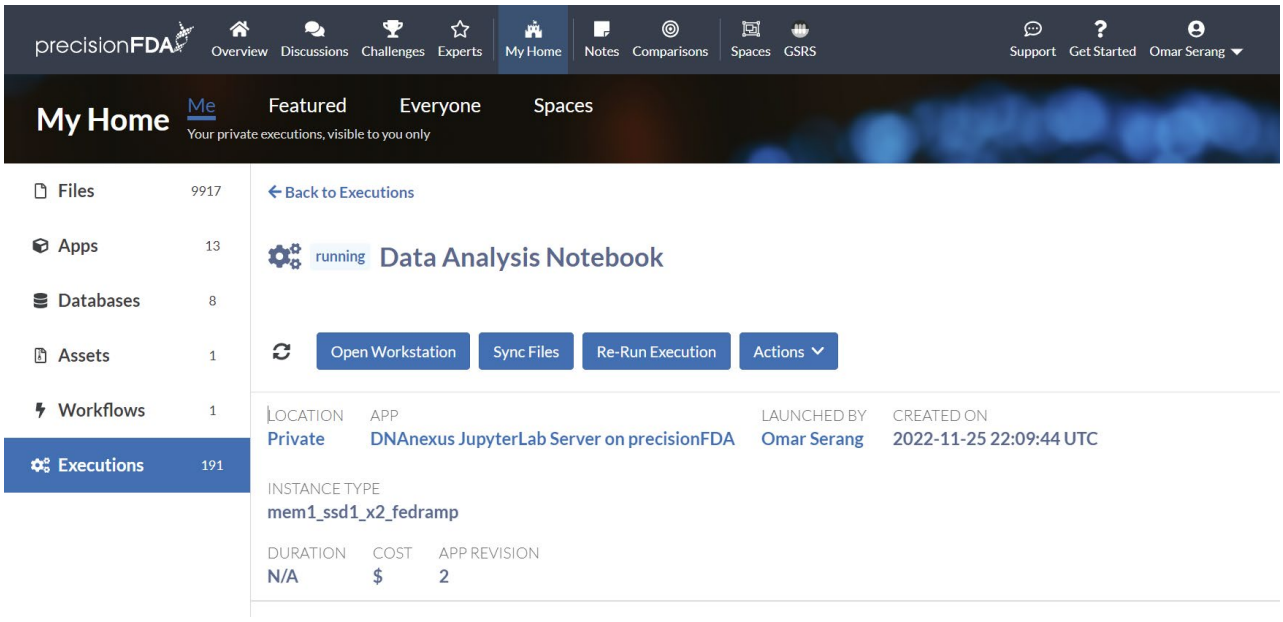
Run the pfda-jupyterLab Featured App

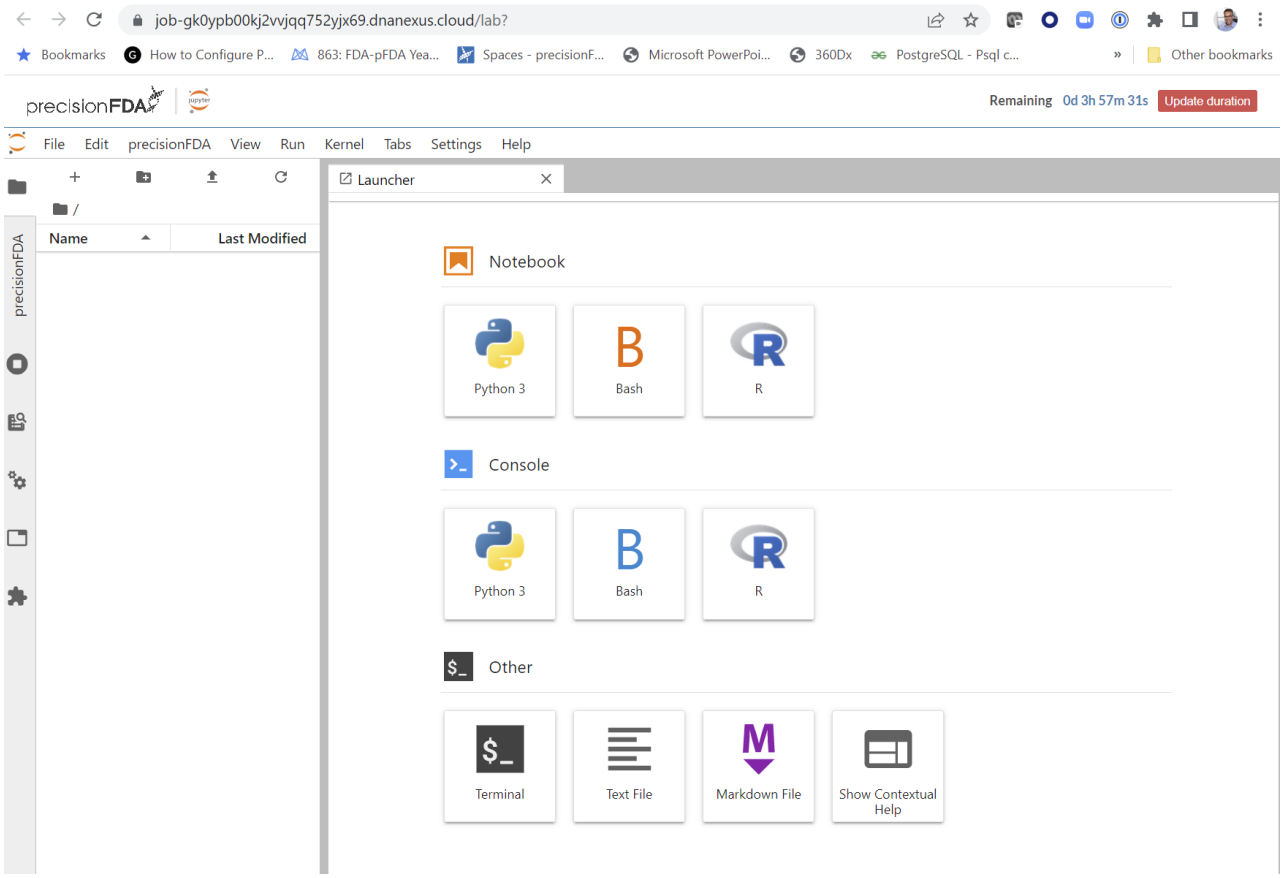
Using the smallest instance type, run the Data Analysis Notebook job specifying PYTHON_R.



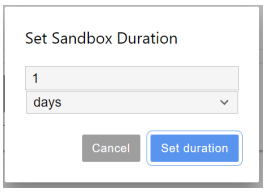


Refresh the execution status using the button until the job is running and open the workstation. It may take a few minutes after the job is running for the notebook to open.


precisionFDA
Overview
Discussions
Challenges
Experts
My Home
Notes
Comparisons
Spaces
GSRS
Support
Get Started
Omar Serang
My Home
Me
Featured
Everyone
Spaces
Your private executions, visible to you only
Files 9917
Apps 13
Databases 8
Assets 1
Workflows 1
Executions 191
Back to Executions
running
Data Analysis Notebook
Open Workstation
Sync Files
Re-Run Execution
Actions
LOCATION
Private
APP
DNAnexus JupyterLab Server on precisionFDA
LAUNCHED BY
Omar Serang
CREATED ON
2022-11-25 22:09:44 UTC
INSTANCE TYPE
mem1_ssd1_x2_fedramp
DURATION
N/A
COST
$
APP REVISION
2

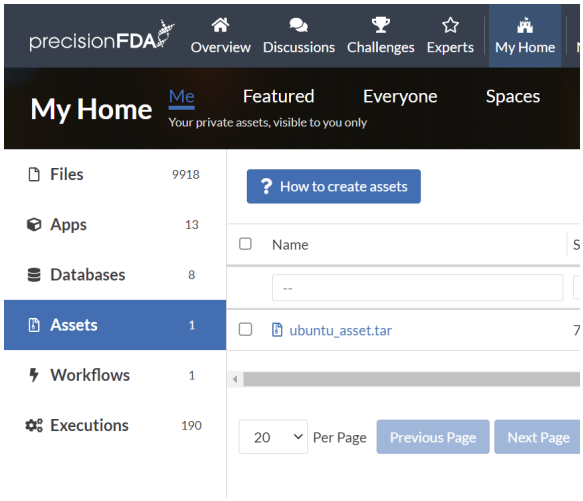

job-gk0ypb00kj2vvjqq752yjx69.dnanexus.cloud/lab?
Bookmarks
How to Configure P...
863: FDA-pFDA Yea...
Spaces - precisionF...
Microsoft PowerPoi...
360Dx
PostgreSQL - Psql c...
Other bookmarks
precisionFDA
Remaining 0d 3h 57m 31s
Update duration
File Edit precisionFDA View Run Kernel Tabs Settings Help
Launcher
Name
Last Modified
Notebook
Python 3
Bash
R
Console
Python 3
Bash
R
Other
Terminal
Text File
Markdown File
Show Contextual Help

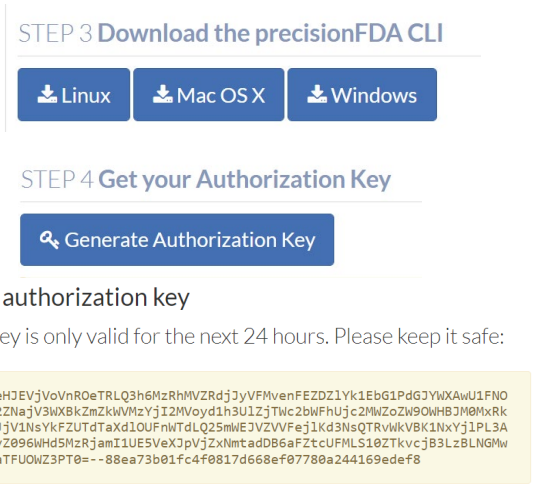
Adjust the remaining time-to-live for the notebook using the Update duration button.



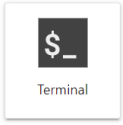
Download and Install the pfda CLI

Under My Home Assets, click on the How to create assets button to find links to the precisionFDA CLI, and the button to generate the temporary authorization key that you'll use with the CLI.



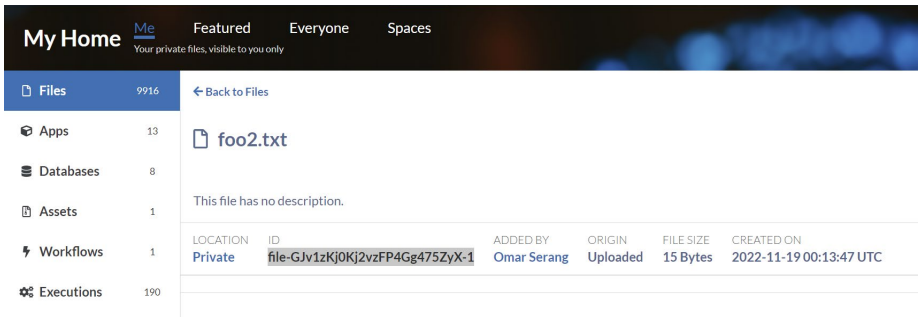


Open a Terminal in the Data Analysis notebook.




```
-- Install pfda CLI
wget https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.1.2.tar.gz
tar xf pfda-linux-2.1.2.tar.gz
mv pfda /usr/bin/
pfda –version
```

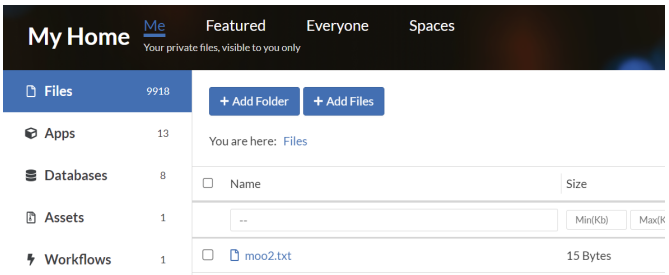
Copy a file ID and retrieve an authorization key to and download a file from precisionFDA to the workstation local FS.




```
pfda download -key
Mk5VTENlTS83R2I1U3dXQkRnWEhzamJvVVFrTVZrOHA4STI4OTM0MitRWnNqZWVBSVRndlBi
cG1IUU9PeStjbTBLRXUzNW5rMmMrMjV6bGVhSnVTUlhDd2dEOVhRdUZvdmE1a29pcHdWWS92
RGNyN1ljTlZtdnNjbE15RXVyVnl1Zkd3UUVxODZpYzNsWi9JWVVBcEw3VE5uaXdMSTdYNHNW
VFJpZGJYdXlVa2hsRFFnR2dDc1JISzhuYWxla2JXLS1zVjRhSVBCWFdaRXBFWnBsMXNtSXB3
PT0=--e19f53de7644d63dd3898717896a88bd0a383db6 -file-id file-
GJv1zKj0Kj2vzFP4Gg475ZyX-1
```

Upload a file from the workstation local filesystem to precisionFDA (note the key is cached).

```
mv foo2.txt moo2.txt
pfda upload-file -file moo2.txt
```



Deploy Local PostgreSQL DB Server

Deploy a local PostgreSQL DB server on the Data Analysis workstation. Map the postgres port from the container to the workstation (host) OS. In the notebook terminal:

```
sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -
cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc |
apt-key add -
apt-get update
apt-get -y install postgresql
```

Configure postgres to enable password-free local login. Find pg_hba.conf in /etc/postgresql/ and configure with permissive permissions.

```
find /etc/postgresql -name pg_hba.conf | xargs sed -i 's/peer/trust/'
find /etc/postgresql -name pg_hba.conf | xargs sed -i 's/md5/trust/'
```

Start the local PostgreSQL DB server on the Data Analysis notebook.

```
/etc/init.d/postgresql start
 * Starting PostgreSQL 15 database server
[ OK ]
```

```
/etc/init.d/postgresql status
15/main (port 5432): online

psql -U postgres -h 127.0.0.1
      psql (15.1 (Ubuntu 15.1-1.pgdg18.04+1))
      SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384,
      compression: off)
      Type "help" for help.

      postgres=#
```

Create a Table with some data in the Local DB

In psql in the notebook terminal create a table, then copy two records from stdin into the table display them.

```
CREATE TABLE public."PATIENT" (
    patient_id int NOT NULL,
    name character varying,
    gender character varying,
    zip character varying,
    country character varying,
    created_date date
);

COPY public."PATIENT" (patient_id, name, gender, zip, country,
created_date) from stdin;
```

Add these two records when prompted from the above COPY, and terminate with the \. record.

```
12345 foo   m     94040 usa   2022-11-25
54321 bar   m     94040 usa   2022-11-25
\.

select * from public."PATIENT";
 patient_id | name | gender |  zip  | country | created_date
------------+------+--------+-------+---------+--------------
      12345 | foo  | m      | 94040 | usa     | 2022-11-25
      54321 | bar  | m      | 94040 | usa     | 2022-11-25
(2 rows)
```

Create a Notebook and Connect to the Local DB

In the notebook terminal, install the psycopg2 binary.

```
pip install psycopg2-binary
```

Open a Python 3 notebook.

And enter the following code:

```
import psycopg2
conn = psycopg2.connect("dbname='postgres' user='postgres'
host='127.0.0.1'")
cur = conn.cursor()
cur.execute('SELECT * FROM public."PATIENT" limit 10')
# fetch results
rows = cur.fetchall()
# iterate through results
for row in rows:
    print ("   ", row)
```

Connect to the Cluster DB

Open a Python 3 notebook.

And enter the following code:

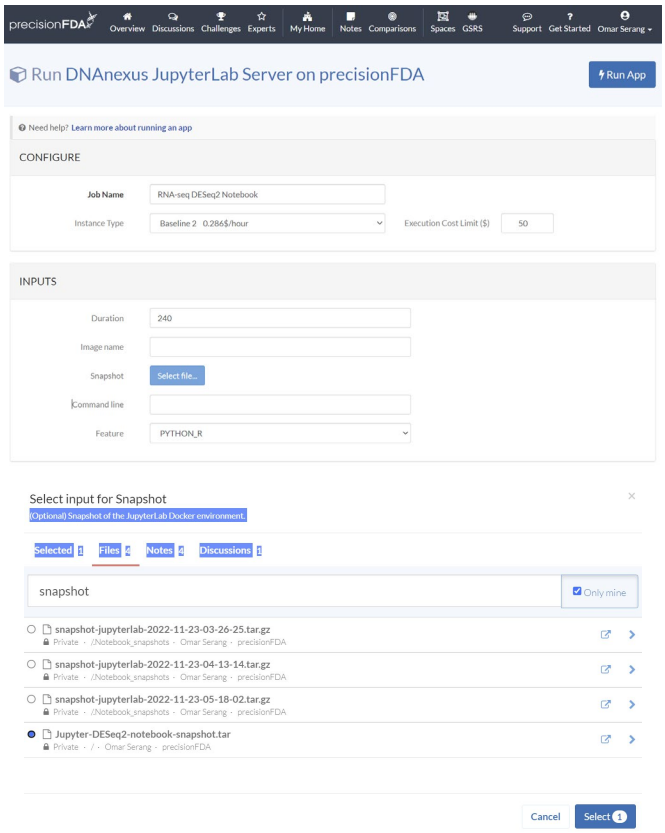
```
import psycopg2

conn = psycopg2.connect("dbname='workstations_and_databases_tutorial_db'
user='root' host='dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-
cqy4cenhebvb.us-east-1.rds.amazonaws.com' password='password'")

cur = conn.cursor()
cur.execute('SELECT * FROM public."PATIENT" limit 10')
# fetch results
rows = cur.fetchall()
# iterate through results
for row in rows:
    print ("PATIENT", row[0], row[1], row[2])

cur.execute('SELECT * FROM public."OBSERVATION" limit 10')
```

```
# fetch results
rows = cur.fetchall()
# iterate through results
for row in rows:
    print ("OBSERVATION", row[0], row[1], row[2])
```

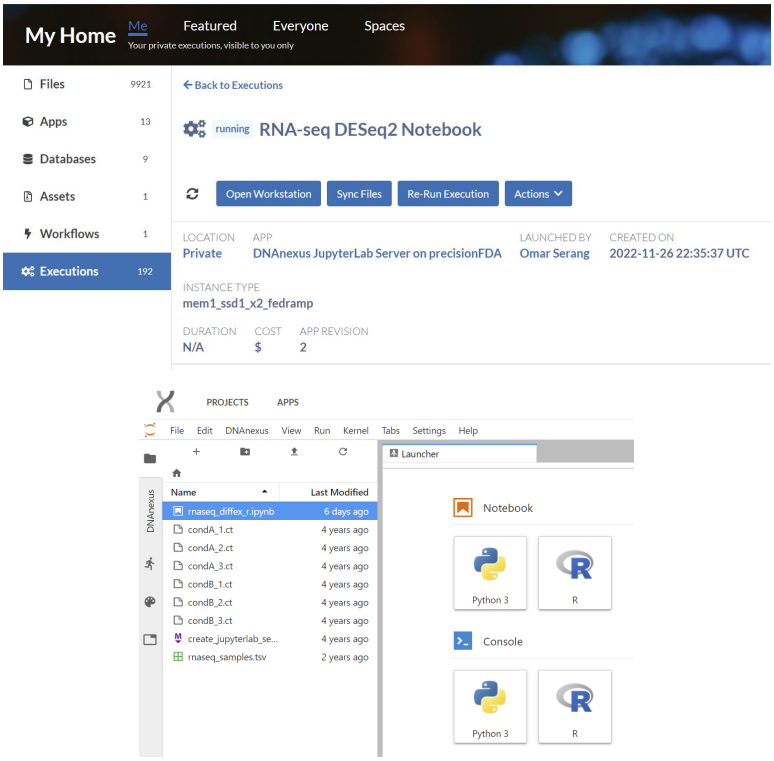
```
PATIENT 12345 Fred Foobar M
PATIENT 12346 Mary Merry F
PATIENT 12347 Barney Rubble M
OBSERVATION 9870 12345 Annual check up
OBSERVATION 9871 12345 Emergency
OBSERVATION 9872 12346 Clinic visit
OBSERVATION 9873 12347 Lab results
OBSERVATION 9874 12347 Post-op checkup
```

Load a Complete Notebook from a Snapshot

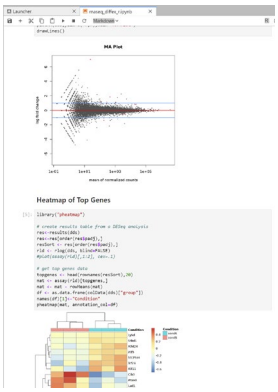
Using My Home / Applications, run the featured pfda-jupyterLab app on the smallest instance type, providing the *Jupyter-DESeq2-notebook-snapshot.tar* file as input. This snapshot contains a complete RNA-seq DESeq2 quantification JupyterLab workbook with R package, notebook, input file and sample sheet all included.



Once the app is running, click the open workstation button to access a rich visual and interactive analysis environment.



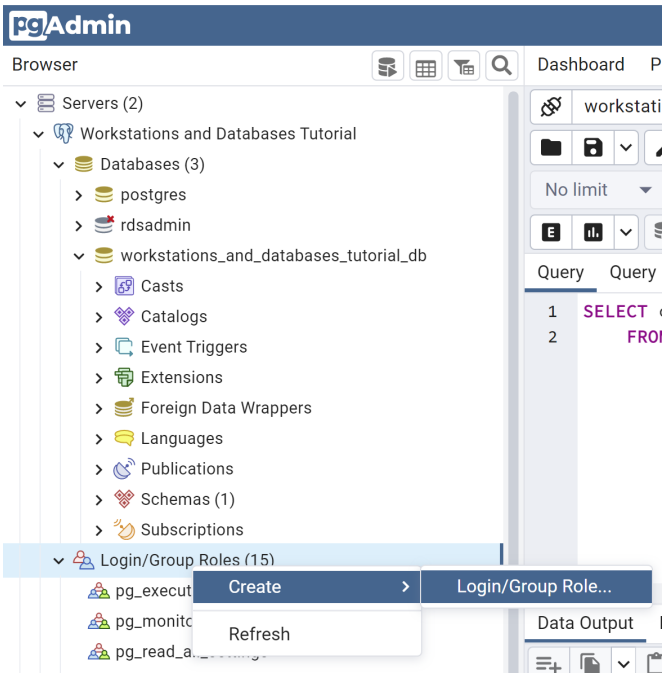
Open the *rnaseq_diffex_r* notebook to explore the data.

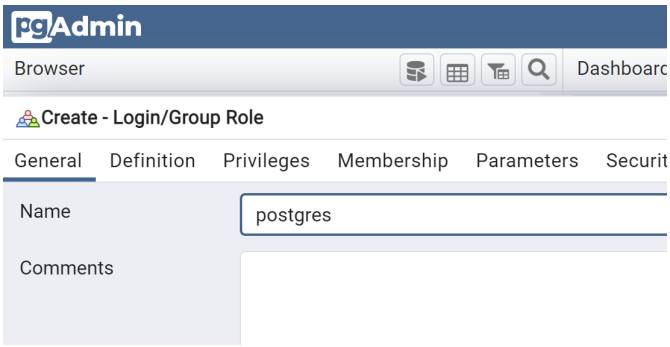


Backup the cluster DB and restore it to local DBs

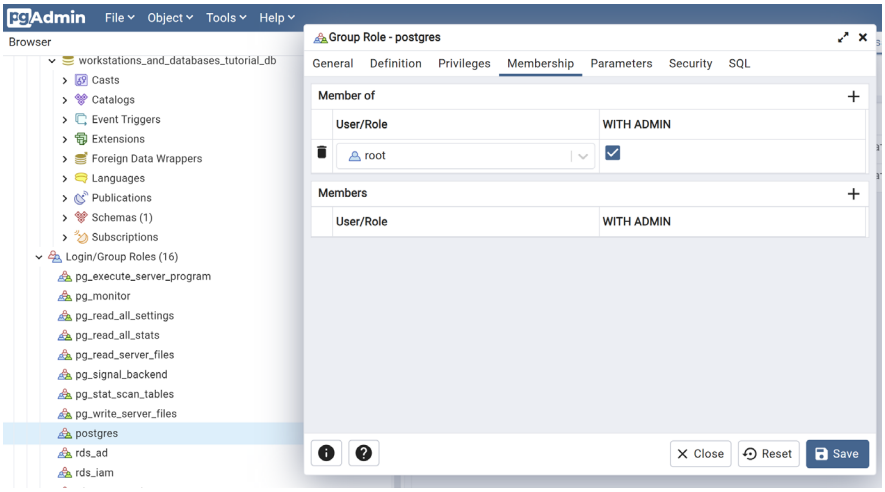
Add a postgres role to the cluster DB

Right-click Login/Group Roles in the Workstations and Databases Tutorial server connection in pgadmin and add a postgres role.



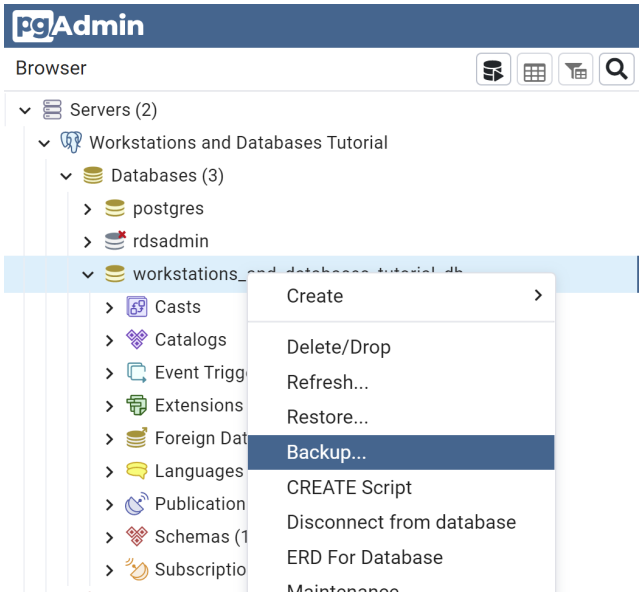


Right-click the postgres role and select properties, and add the to the root group with admin privileges in the Membership tab.

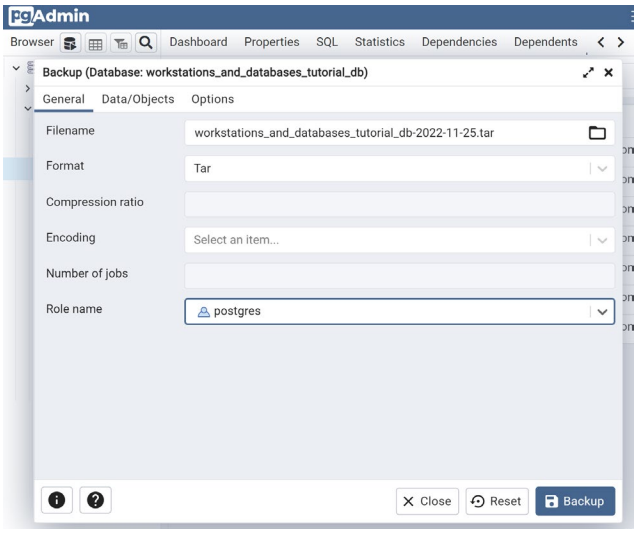


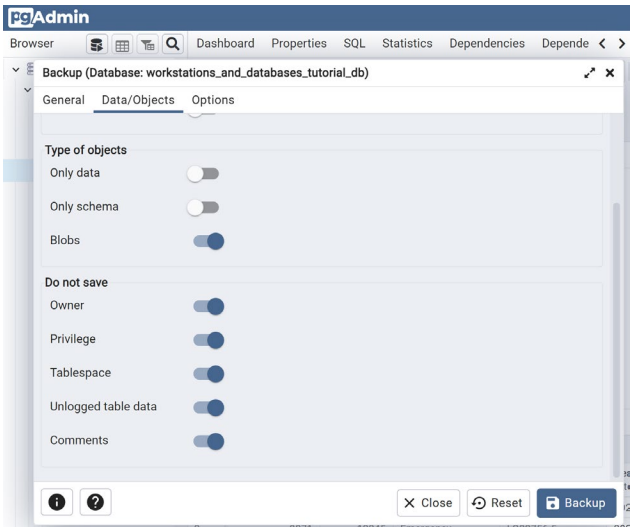
Backup the cluster DB using pgadmin

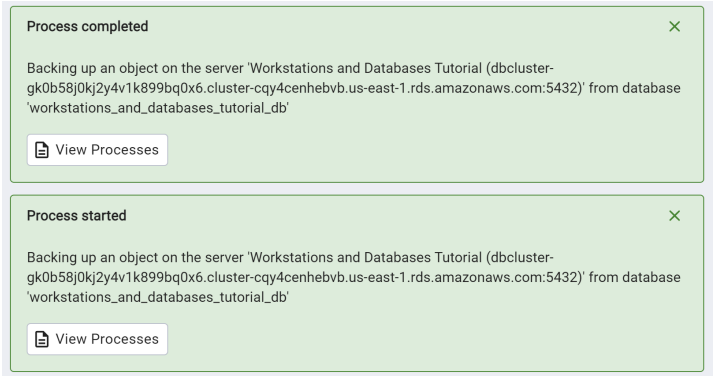
Select the workstations_and_databases_tutorial_db database in the Workstations and Databases Tutorial server connection in pgadmin and right-click to backup the database.



Specify a backup filename (e.g. workstations_and_databases_tutorial_db-2022-11-25.tar), format as Tar, assign role name *postgres* and set all the Data/Objects Do not save options..







Copy the backup file from the pgadmin container to the workstation filesystem

Since pgadmin is running in a Docker container on the data analysis workstation, we are going to have to connect to the pgadmin container shell and copy the backup file to the mount point shared by the container and the workstation (i.e. /home/dnanexus/db_backups). On the data analysis workstation:

Connect to the shell in the pgadmin container.

```
docker exec -it pgadmin sh
/pgadmin4 $
```

Copy the backup file from the pgadmin backup directory to the container-host shared volume.

```
ls /var/lib/pgadmin/storage/user_domain.com
workstations_and_databases_tutorial_db-2022-11-25

cp
/var/lib/pgadmin/storage/user_domain.com/workstations_and_databases_tuto
rial_db-2022-11-25.tar /home/dnanexus/db_backups
```

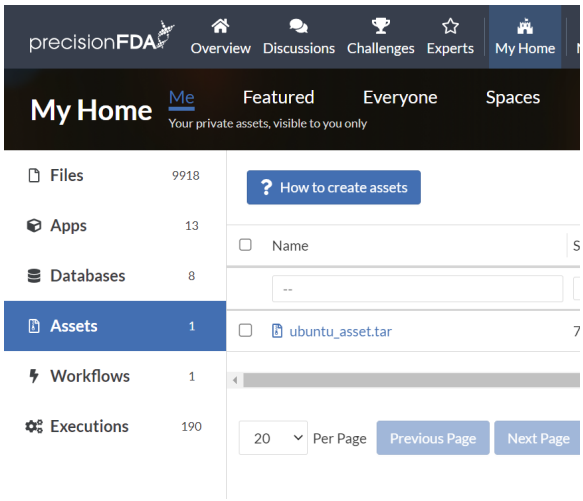
Control-D to exit the container shell and verify the presence of the backup file on the workstation in the container-host shared mount point.

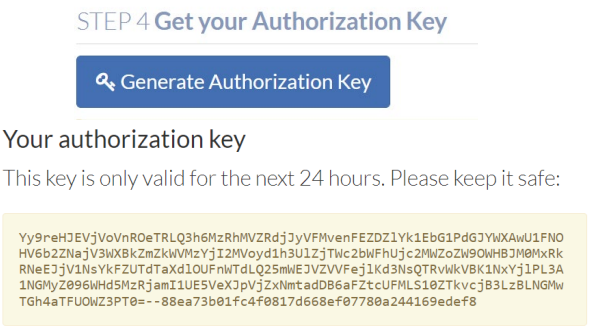
```
ls db_backups/
```

`workstations_and_databases_tutorial_db-2022-11-25`

Upload the backup file to precisionFDA

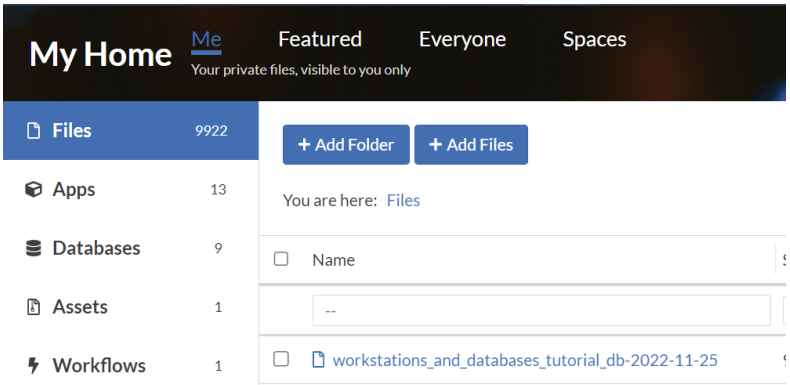
Under My Home Assets, click on the How to create assets button to find the button to generate the temporary authorization key that you'll use with the CLI.





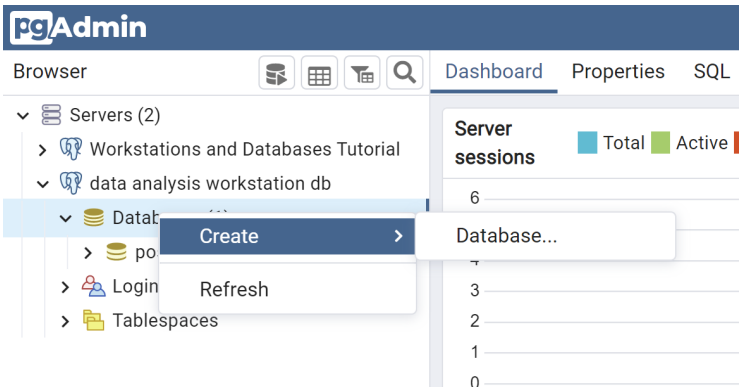
On the data analysis workstation shell:

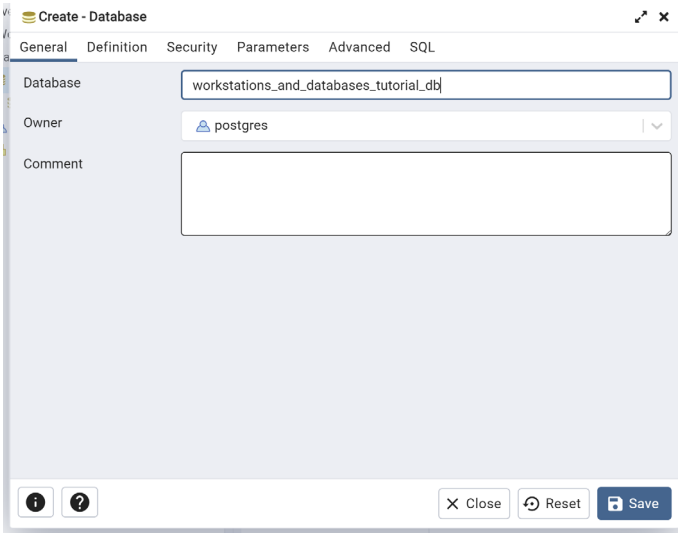
```
key="..."

pfda upload-file -key $key -file
~/db_backups/workstations_and_databases_tutorial_db-2022-11-25.tar
```



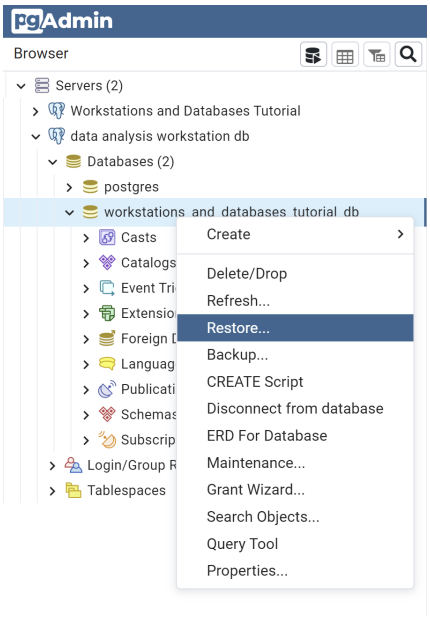
Restore the backup to the data analysis workstation local DB

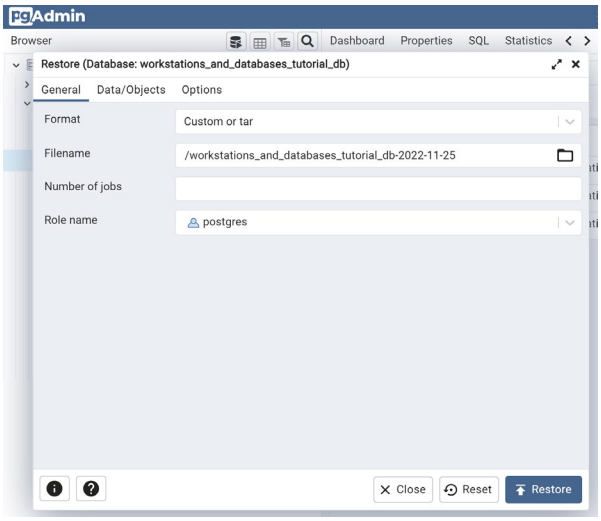
Using the pgadmin connection to the data analysis workstation db, create a new database *workstations_and_databases_tutorial_db*, owner *postgres*.

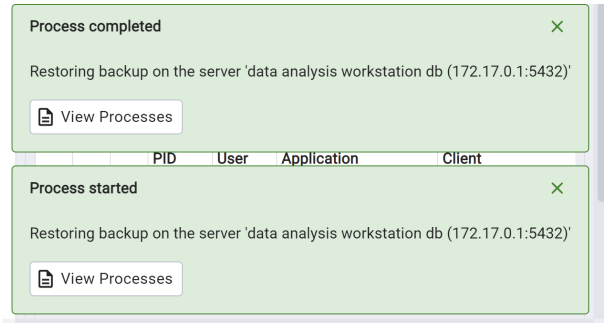




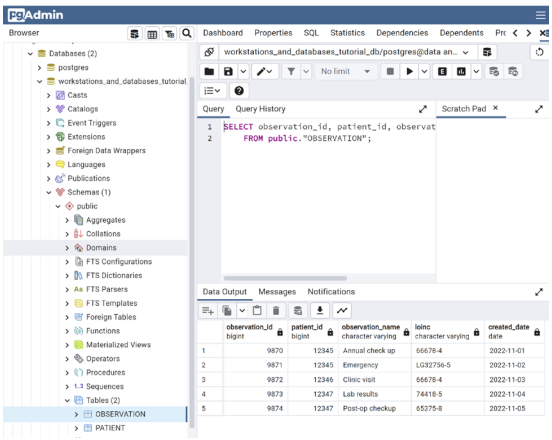
Right-click on the new database on the data analysis and workstation db server connection and restore the backup to the local server (from the file in the pgadmin container), using custom or tar format, and the postgres role name.





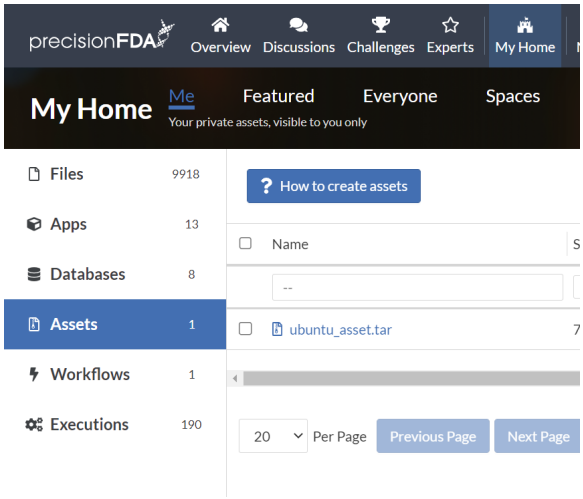


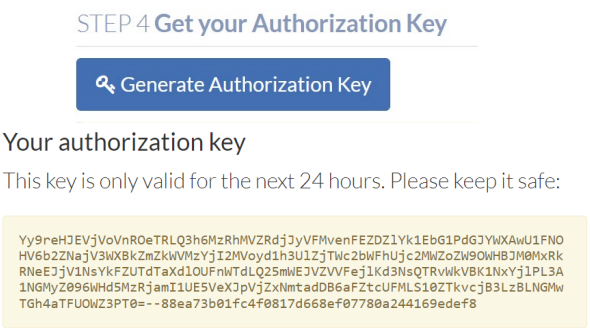
Select the contents of the restored PATIENT and OBSERVATION tables.



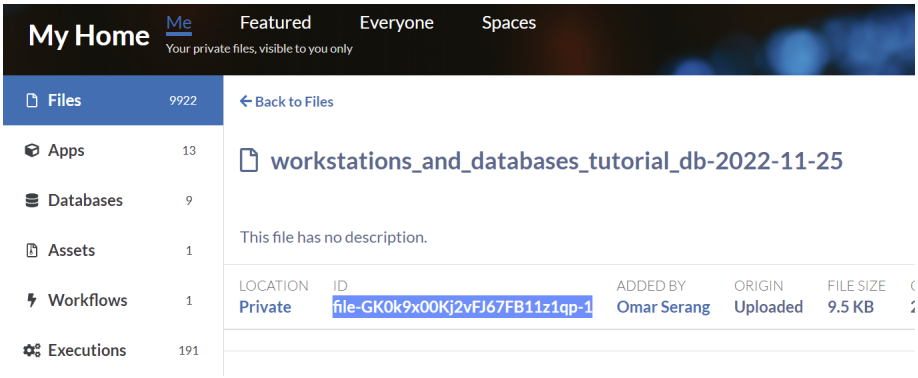
Restore the backup to the data analysis notebook local DB

Under My Home Assets, click on the How to create assets button to find the button to generate the temporary authorization key that you'll use with the CLI.





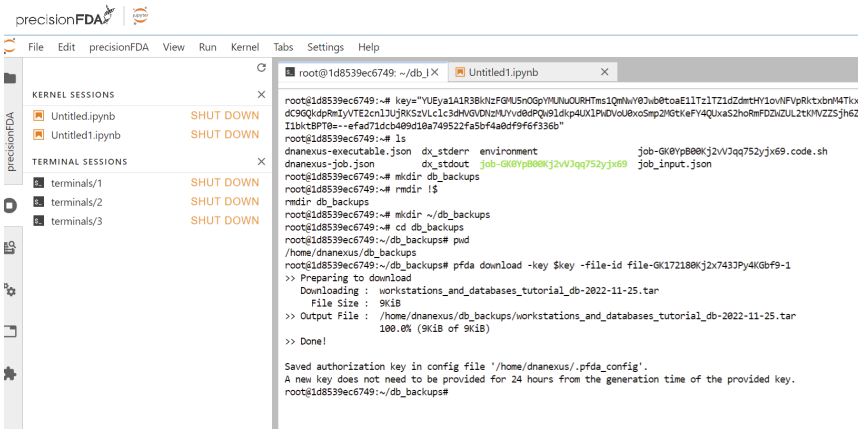
Click into the detail page for the backup file and copy the file ID.



In a terminal window in the data analysis jupyterLab notebook, download the backup file using its file ID as copied in the step above:

```
mkdir ~/db_backups
cd db_backups
```

```
key="..."
pfda download -key $key -file-id file-GK172180Kj2x743JPy4KGbf9-1
```



In psql connected to the local host, create a new database *workstations_and_databases_tutorial_db*, and a new user *root*.

```
psql -U postgres -h 127.0.0.1
psql (15.1 (Ubuntu 15.1-1.pgdg18.04+1))
postgres=#

CREATE USER root;

CREATE DATABASE workstations_and_databases_tutorial_db
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    CONNECTION LIMIT = -1
    IS_TEMPLATE = False;
```

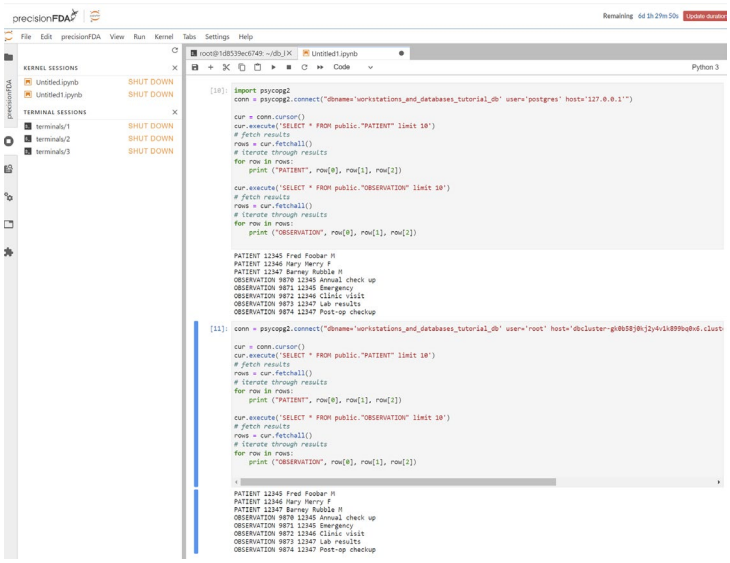
Ctrl-D to exit psql and use restore the database from the backup file.

```
pg_restore --dbname=workstations_and_databases_tutorial_db --verbose
~/db_backups/workstations_and_databases_tutorial_db-2022-11-25.tar -U
postgres
```

You can ignore the errors associated with the *root* role not existing and use the Python notebook to select the contents from the restored database. We can observe the same results from newly restored database as from the cluster database that was the backup source. In a notebook Python code block:

```
import psycopg2
conn = psycopg2.connect("dbname='workstations_and_databases_tutorial_db'
user='postgres' host='127.0.0.1'")

cur = conn.cursor()
cur.execute('SELECT * FROM public."PATIENT" limit 10')
# fetch results
rows = cur.fetchall()
# iterate through results
for row in rows:
```

```
    print ("PATIENT", row[0], row[1], row[2])

cur.execute('SELECT * FROM public."OBSERVATION" limit 10')
# fetch results
rows = cur.fetchall()
# iterate through results
for row in rows:
    print ("OBSERVATION", row[0], row[1], row[2])
```



Snapshot, Terminate, and Restore Workstations

In keeping with good cloud usage practice, we will snapshot and terminate the workstations, preserving their entire state as built out through this tutorial. Additionally since we've backed up the database to a precisionFDA file, we can safely terminate the cluster database as well.

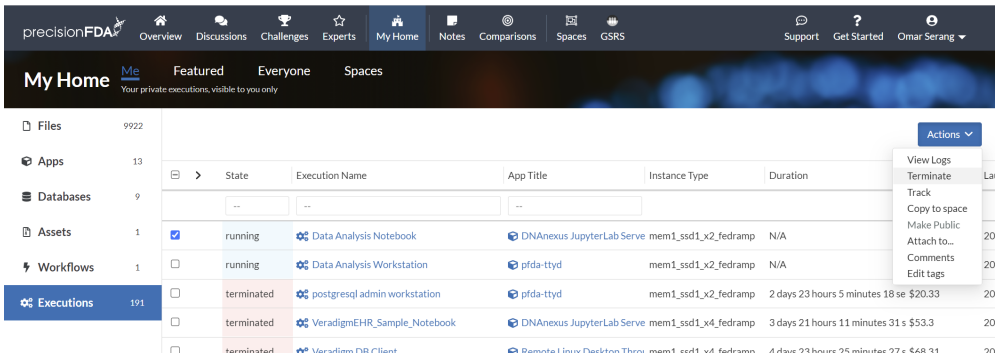
Stop the Docker Containers and Snapshot Data Analysis Workstation

Using the data analysis workstation shell, create a snapshot of the workstation in you My Home files area.

```
Docker stop
dx-create-snapshot

dx ls -al *snapshot
```

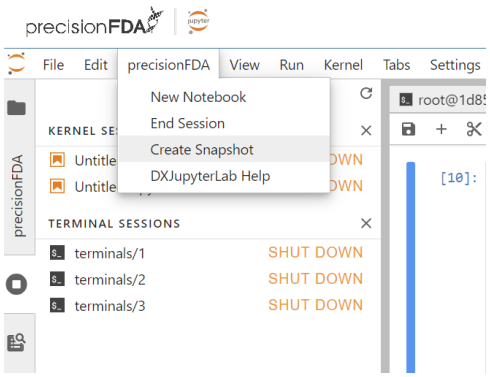
Terminate the Workstation

In My Home / Executions, select (one at a time unfortunately) the Data Analysis Workstation and Data Analysis Notebook executions and select Terminate under the Action dropdown menu.



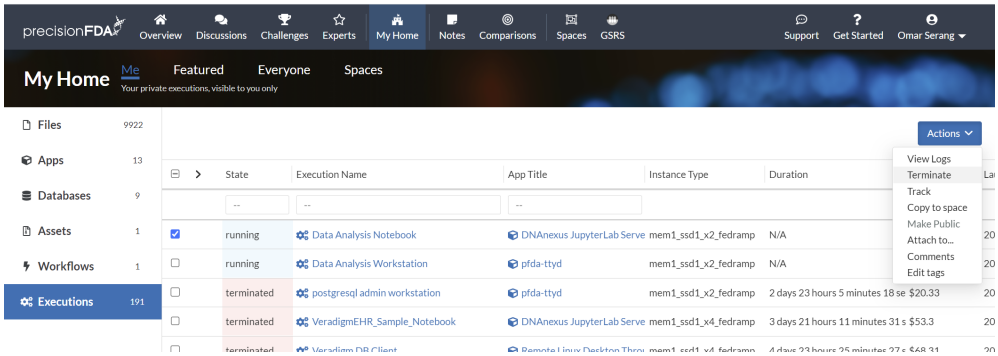
Snapshot and Terminate the Data Analysis Notebook

Select Create Snapshot in the precisionFDA menu in the jupyterLabs interface.



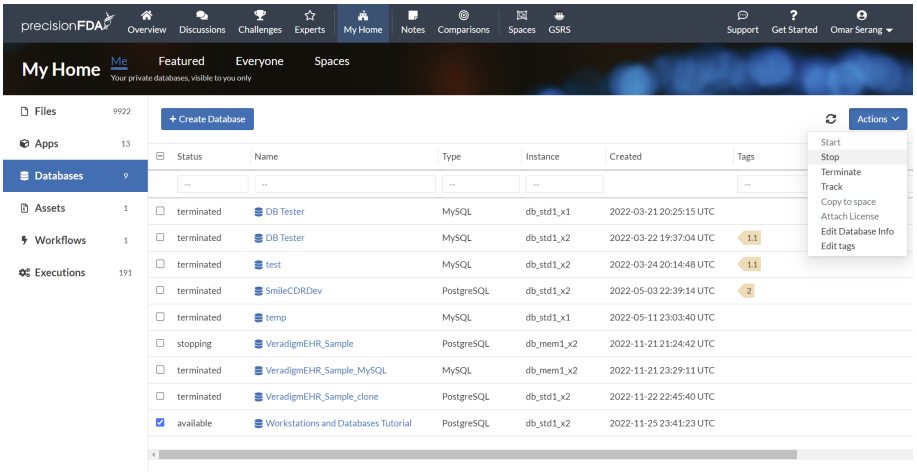
Terminate the Workstation and Notebook and Database Cluster

In My Home / Executions, select (one at a time unfortunately) the Data Analysis Workstation and Data Analysis Notebook executions and select Terminate under the Action dropdown menu.



Stop or Terminate the Database Cluster

In My Home / Databases, select the database for action and either Stop or Terminate the database using the Action dropdown menu. If your data is already stored on precisionFDA and can be readily reconstituted into a new database, then select Terminate. If your database is a work in progress and you'd like to keep it intact while not using it overnight, or the weekend, then select Stop.



Restore Workstation and Notebook from Snapshots