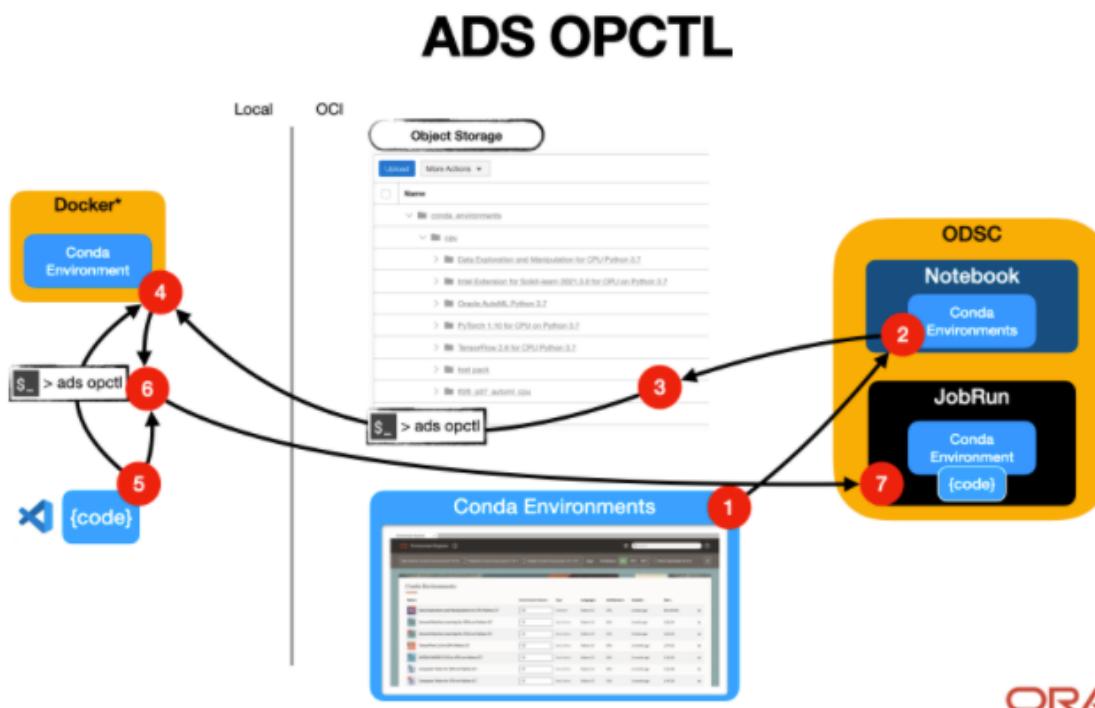


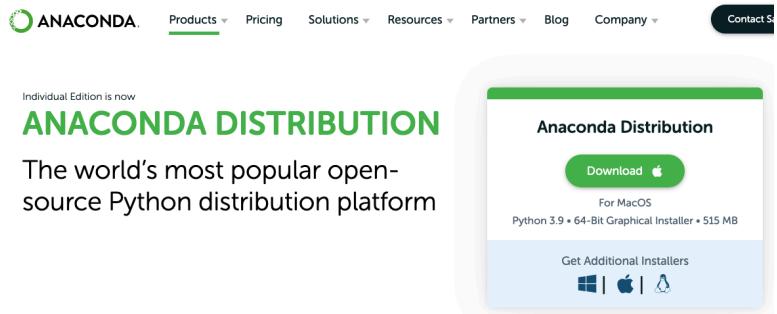
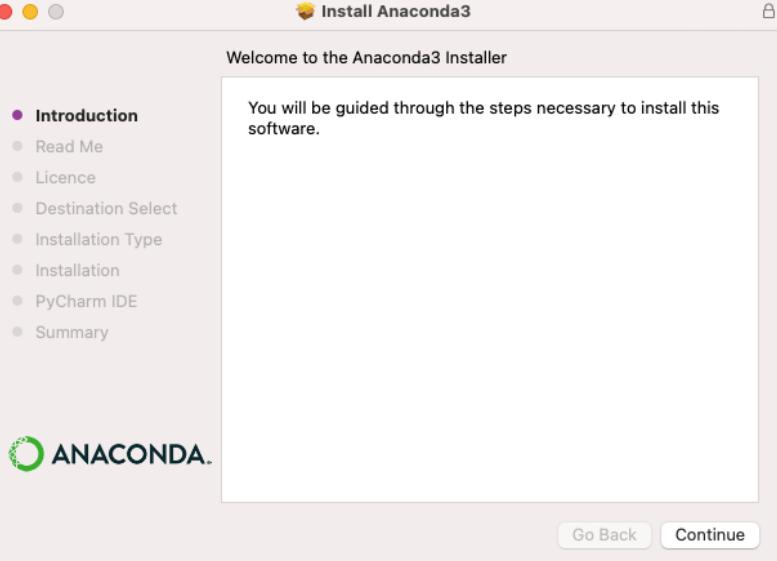
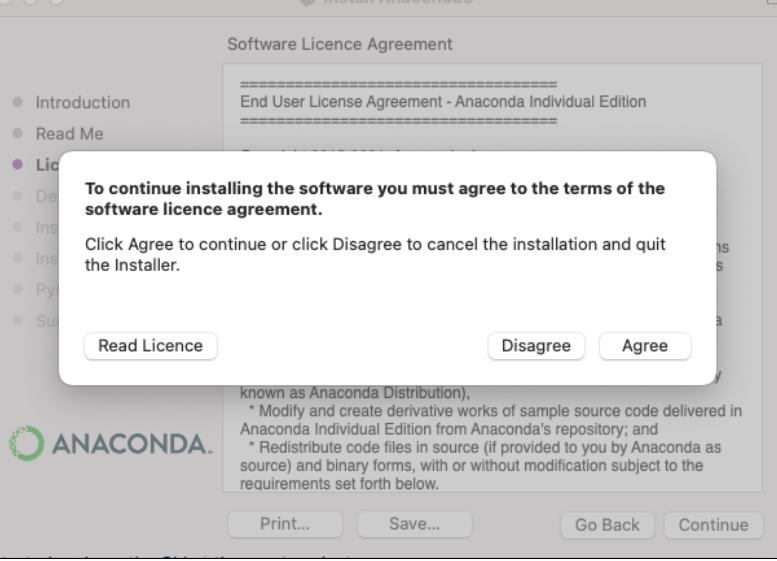
ADS OPTCL Jobs

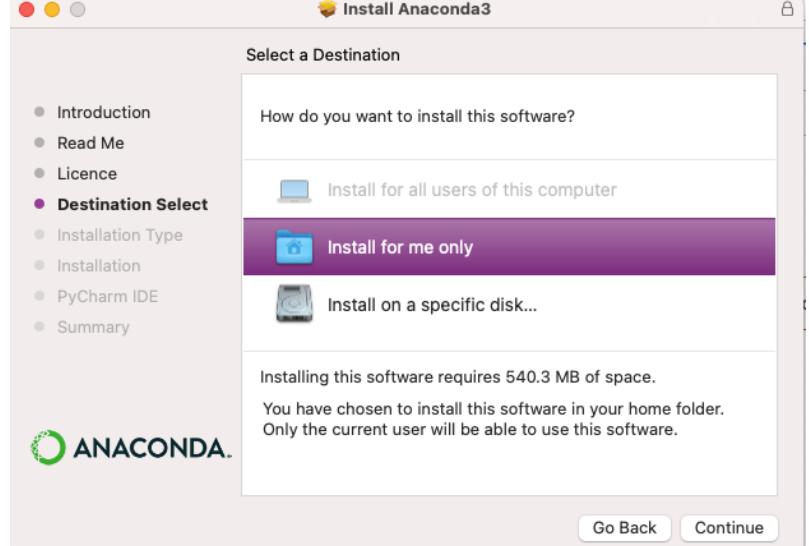
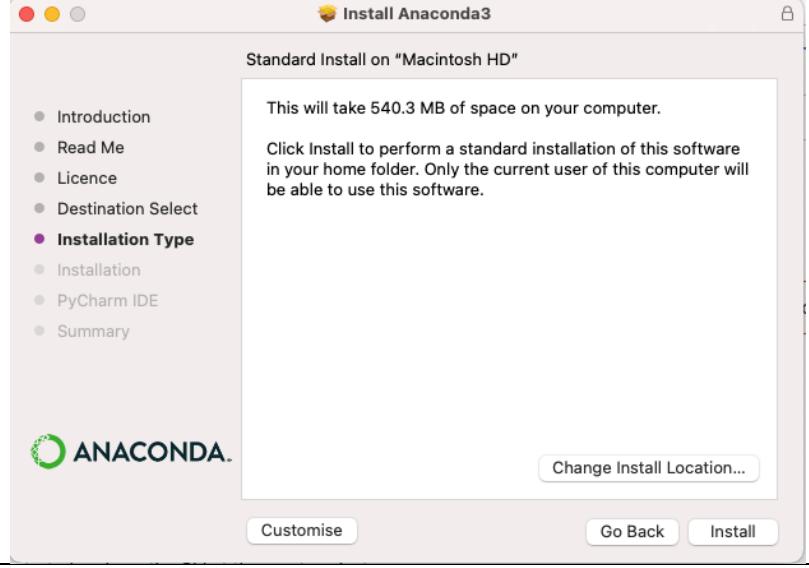
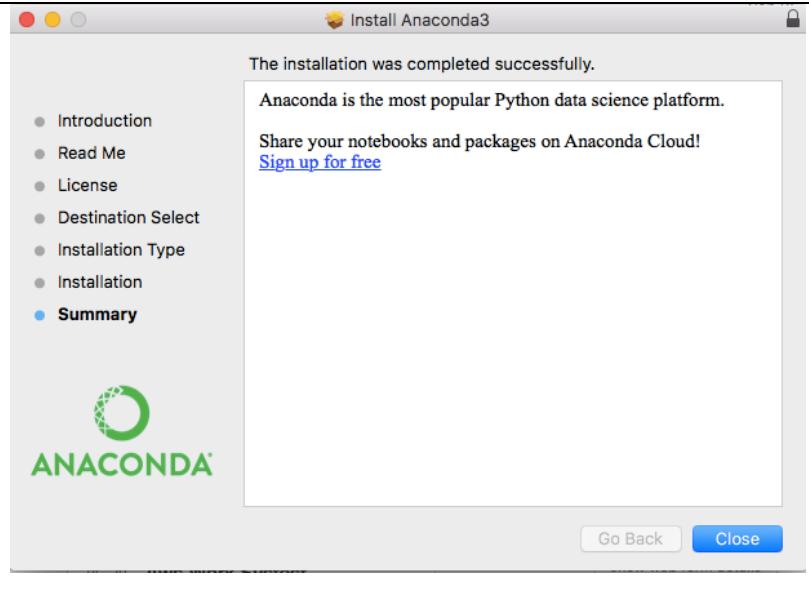
ADS Operator Control is a new CLI that allows to run Python Code on OCI essentially from everywhere, as well as test run code against ODSC Conda Environments outside of OCI. The main capabilities of the feature are:

- Allows to test run code against the ODSC Conda Packages outside OCI
- Provides the ability to submit code to OCI Data Science Jobs with a single command line essentially from everywhere
- Streamlines the AI/ML End-to-End development process based on Oracle Data Science Cloud Service
- Helps manage machine learning environment dependencies and reproducibility

Note – This guide will cover implementation using MacOS.



Pre-Requisite 1 – Install Conda	
Visit the Anaconda Website and download the Installer for your OS - https://www.anaconda.com/products/distribution	
Run the .pkg installer from your downloads folder. Double Click on File. Click Continue on Introduction. Click Continue on Read Me.	
Accept Licence Agreement.	

<p>Select Install for me only.</p> <p>Click Continue.</p>	 <p>The screenshot shows the 'Select a Destination' step of the Anaconda3 installation wizard. On the left, a sidebar lists steps: Introduction, Read Me, Licence, Destination Select (which is highlighted in purple), Installation Type, Installation, PyCharm IDE, and Summary. The main area is titled 'How do you want to install this software?' and contains three options: 'Install for all users of this computer' (disabled), 'Install for me only' (selected, highlighted in purple), and 'Install on a specific disk...'. Below this, a note states: 'Installing this software requires 540.3 MB of space. You have chosen to install this software in your home folder. Only the current user will be able to use this software.' At the bottom are 'Go Back' and 'Continue' buttons.</p>
<p>Click Install.</p>	 <p>The screenshot shows the 'Standard Install on "Macintosh HD"' step of the Anaconda3 installation wizard. On the left, a sidebar lists steps: Introduction, Read Me, Licence, Destination Select, Installation Type (which is highlighted in purple), Installation, PyCharm IDE, and Summary. The main area contains a note: 'This will take 540.3 MB of space on your computer. Click Install to perform a standard installation of this software in your home folder. Only the current user of this computer will be able to use this software.' Below this are 'Change Install Location...', 'Customise', 'Go Back', and 'Install' buttons.</p>
<p>You can ignore the installation of PyCharm IDE and Skip.</p>	
<p>Once Installed, Click Close.</p>	 <p>The screenshot shows the 'Summary' step of the Anaconda3 installation wizard. On the left, a sidebar lists steps: Introduction, Read Me, License, Destination Select, Installation Type, Installation, and Summary (which is highlighted in blue). The main area displays a message: 'The installation was completed successfully. Anaconda is the most popular Python data science platform. Share your notebooks and packages on Anaconda Cloud! Sign up for free'. At the bottom are 'Go Back' and 'Close' buttons.</p>

You can verify installation of Anaconda, by opening a Terminal and Running:

conda --version

```
dhcp-10-175-187-93:~ isyed$ conda --version
conda 4.10.3
dhcp-10-175-187-93:~ isyed$ █
```

Visit the Docker Desktop Download Page and download the installer relevant to your OS:
<https://www.docker.com/products/docker-desktop/>

Docker Desktop

Install Docker Desktop – the fastest way to containerize applications.

Mac with Intel Chip

Mac with Apple Chip

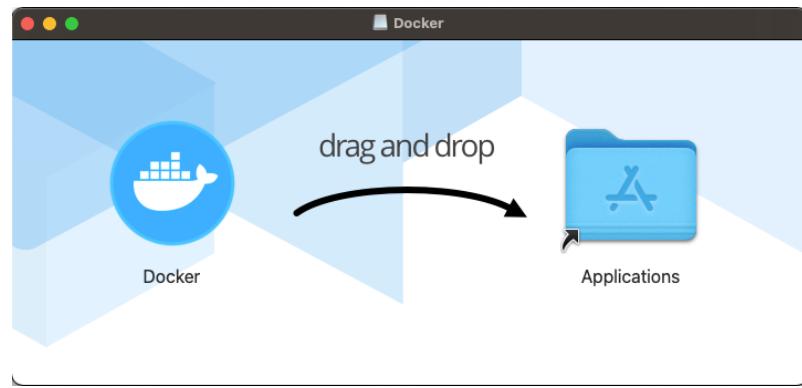
MOST COMMON

Also available for Windows and Linux

Once downloaded, double click on the **Docker.dmg** file downloaded.

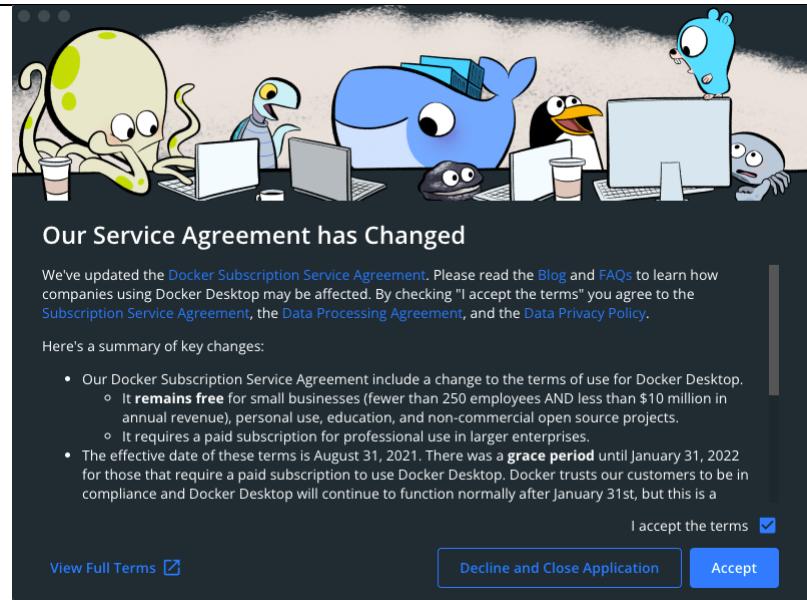
You can **Drag and Drop** the Docker Application Icon into your Applications Folder.

This may take a couple minutes to execute the copy.



Once copied across you can launch Docker.

You will be asked to Accept the new Service Agreement.

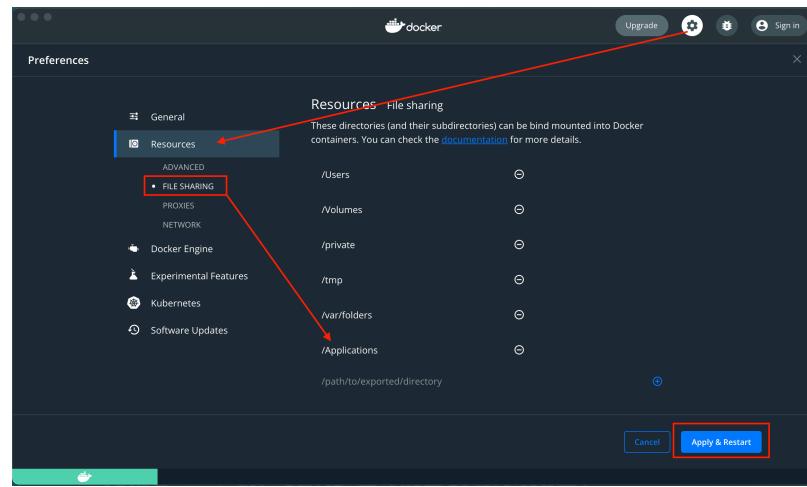


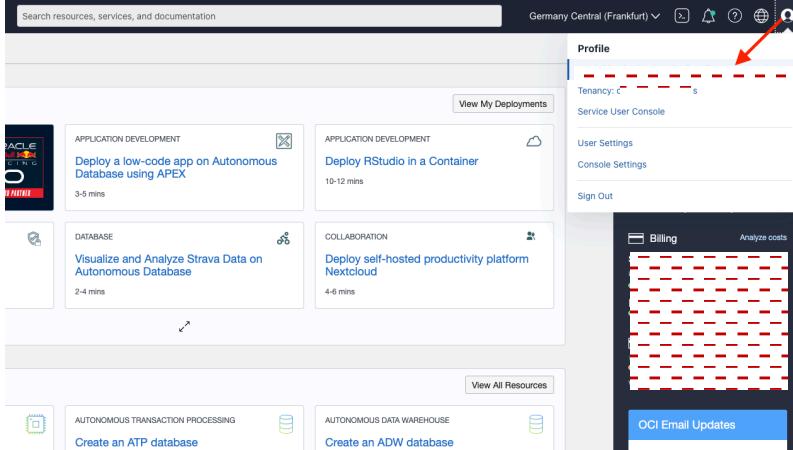
Creating, installing, and publishing conda packs involves mounting folders to docker images. Please make sure that you give docker desktop access to file sharing of the parent folder for conda.

Visit Settings > Resources > File Sharing

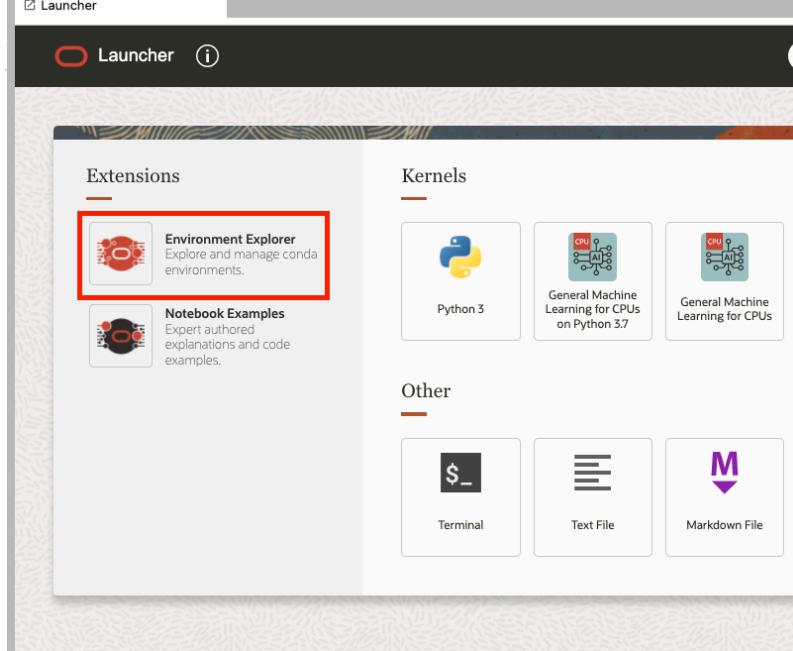
Add your parent folder for conda in my case /Applications.

Click Apply & Restart.



Pre-Requisite 3 – Generate API Signing Key					
<p>Login to the OCI Console.</p> <p>Visit your Profile in the top right corner and select your username.</p>					
<p>Scroll down and select the API Keys Tab.</p> <p>Click on Add API Key.</p>	<p>Auth tokens: Yes View Configuration file</p> <table border="1"> <thead> <tr> <th>Resources</th> <th>API Keys</th> </tr> </thead> <tbody> <tr> <td> Groups API Keys → Auth Tokens Customer Secret Keys Database Passwords OAuth 2.0 Client Credentials SMTP Credentials </td> <td> Add API Key → Groups Fingerprint </td> </tr> </tbody> </table>	Resources	API Keys	Groups API Keys → Auth Tokens Customer Secret Keys Database Passwords OAuth 2.0 Client Credentials SMTP Credentials	Add API Key → Groups Fingerprint
Resources	API Keys				
Groups API Keys → Auth Tokens Customer Secret Keys Database Passwords OAuth 2.0 Client Credentials SMTP Credentials	Add API Key → Groups Fingerprint				
<p>Select 'Generate API Key Pair'</p> <p>Download both the Private and Public Key and keep them safe.</p> <p>Click on Add.</p>	<p>Add API Key</p> <p>Note: An API key is an RSA key pair in PEM format used for signing API requests. You can generate the key pair here and download the private key. If you already have a key pair, you can choose to upload or paste your public key file instead. Learn more</p> <p><input checked="" type="radio"/> Generate API Key Pair <input type="radio"/> Choose Public Key File <input type="radio"/> Paste Public Key</p> <p>Public Key</p> <p>i Download the private key. It will not be shown again. After you download it, change the file permissions so only you can view it.</p> <p>Download Private Key Download Public Key</p> <p>Add Cancel</p>				

<p>Copy and paste the content of the Configuration File Preview into a local file named 'config'.</p> <p>Make sure to update the key_file path to point to your private key file.</p> <p>It is recommended to store both the config and private key file into the <code>~/.oci</code> directory.</p> <p>These will be used to authenticate against OCI when downloading condas locally and submitting Data Science Jobs.</p>	<h3>Configuration File Preview</h3> <p>Help</p> <p>Note: This configuration file snippet includes the basic authentication information you'll need to use the SDK, CLI, or other OCI developer tool. Paste the contents of the text box into your <code>~/.oci/config</code> file and update the <code>key_file</code> parameter with the file path to your private key. If you already have a Default profile in your config profile, you'll need to perform some additional steps. Learn more</p> <p>Select API Key Fingerprint <input type="text" value="-----"/></p> <p>Configuration File Preview Read-Only</p> <pre>[DEFAULT] user=- fingerprint=- tenant=- region=- key_file=-----oci-privatekey.pem</pre> <p>Paste the contents of the text box into your <code>~/.oci/config</code> file. Copy</p> <p>Close</p> <pre>dhcp-10-175-160-223:.oci isyed\$ pwd /Users/isyed/.oci dhcp-10-175-160-223:.oci isyed\$ ls -la total 16 drwxr-xr-x 4 isyed staff 128 6 Apr 10:49 . drwxr-xr-x+ 62 isyed staff 1984 5 Apr 12:57 .. -rw-r--r--@ 1 isyed staff 331 6 Apr 10:49 config -rw-r--r--@ 1 isyed staff 1726 6 Apr 10:48 -----oci-privatekey.pem dhcp-10-175-160-223:.oci isyed\$</pre>
--	---

<h3>Publishing a OCI Data Science Conda to Object Storage</h3>	
<p>If we want to install our ADS Conda environments locally we must first publish them to the OCI Object Storage in order to then later download them locally.</p> <p>Open up your OCI Data Science Notebook Session and Navigate to the Launcher.</p> <p>Select 'Environment Explorer'.</p>	

Navigate to a conda environment you would like to make available locally and *copy* the install command to first install it within the OCI Data Science Service.

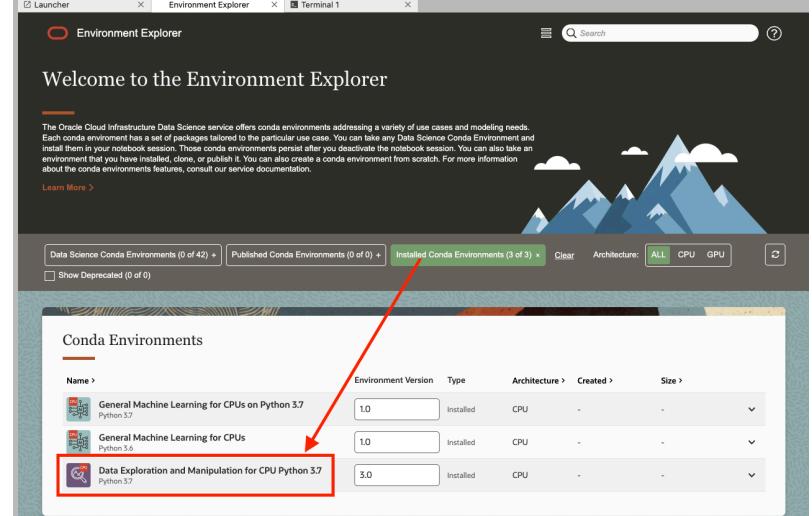
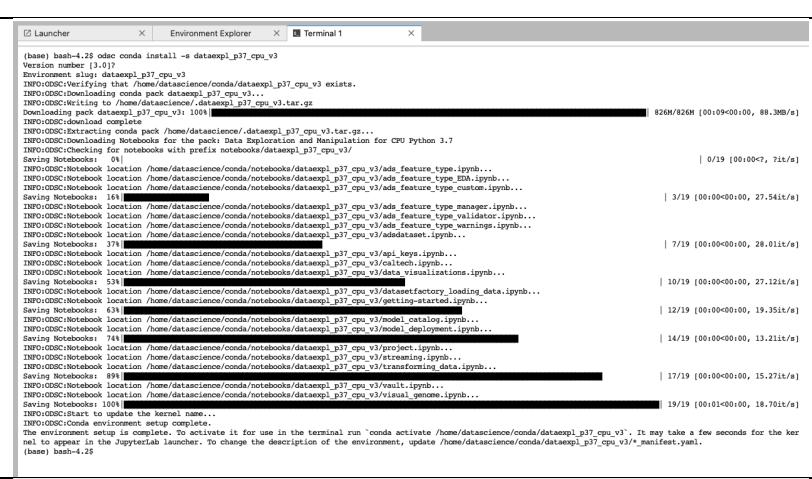
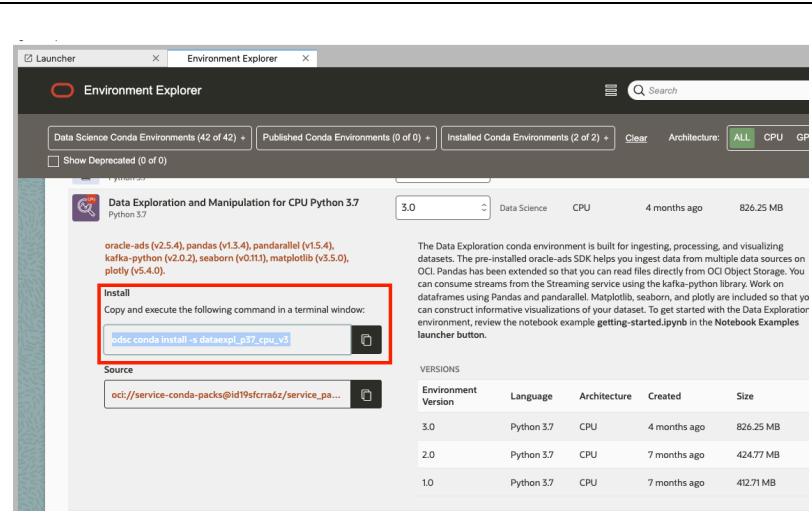
In my case I chose the '*Data Exploration and Manipulation for CPU Python 3.7*' conda.

Open a Terminal window from the Launcher within the OCI Data Science Platform, **paste and run** the command from early.

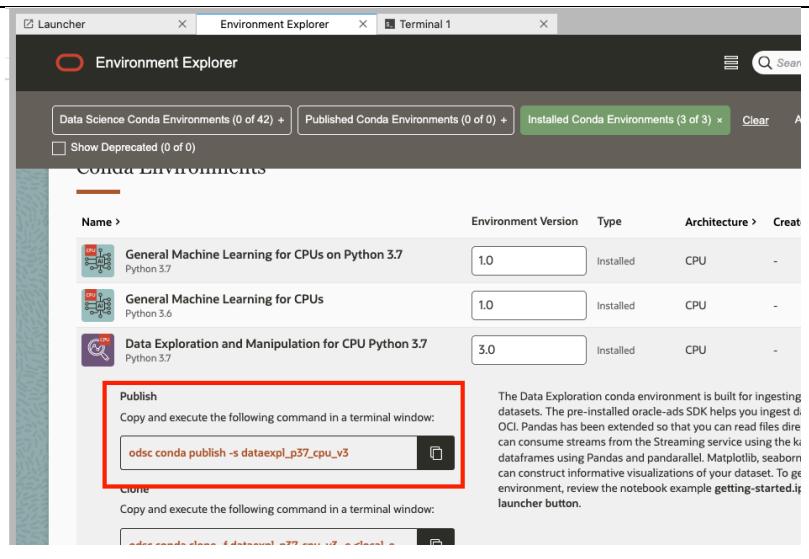
```
odsc conda install -s  
dataexpl_p37_cpu_v3
```

This may take a bit of time
to install all packages.

Once installed, navigate back to the environment explorer, and **click** on the '***Installed Conda Environments***' Tab, you should now see your new conda listed.



Select your newly installed conda and *copy* the ***Publish Command*** which we will use to push the conda environment to our Object Storage.



Before we publish the conda we must first initialise our Object Storage within the Data Science Service.

Within your Terminal in the Data Science Service **enter** and run the following:

```
odsc conda init -b  
<bucket_name> -n [REDACTED]  
<bucket_name_space>
```

Once initialised, we can *paste* our *publish conda command* into the Terminal Window.

```
odsc conda publish -s  
dataexpl p37 cpu v3
```

name.

This might take some time
to save the content to OCI
Object Storage



```
(base) bash-4.2$ odsc conda publish -a dataexpl_p37_cpu_v3
INFO:ODSC:Using environment information from /home/dsci/science/conda/dataexpl_p37_cpu_v3/data_Exploration_and_Manipulation_for_CPU_Python_3.7_manifest.yaml.
INFO:ODSC:Overwriting manifest file at /home/dsci/science/conda/dataexpl_p37_cpu_v3/data_Exploration_and_Manipulation_for_CPU_Python_3.7_manifest.yaml with latest dependency information
dataexpl/lib/python3.6/site-packages/conda/pack/core/pypy57;UserWarning:
Conda-managed packages were found without entries in the package cache. This
is usually due to 'conda clean -e' being unaware of symlinks or copied
packages. Checkacted packages:
  - lib_imperium-linpack-2.11.1... https://conda.acmeka.org/conda-forge/linux-64/lib_imperium-64-2.36.1-he41c9c9_2.tar.bz2
  - liblbb-2.70.1... https://conda.acmeka.org/conda-forge/linux-64/liblbb-2.70.1-h1274984_0.tar.bz2
  - pately-0.5.2... https://conda.acmeka.org/conda-forge/search/pately-0.5-pyhd8ed1ab_0.tar.bz2
  - pately-0.5.2... https://conda.acmeka.org/conda-forge/search/pately-0.5-pyhd8ed1ab_0.tar.bz2
  - ide4-3.1... https://conda.acmeka.org/conda-forge/search/ide4-3.1-pyhd8ed1ab_0.tar.bz2
  - pytz-2021.3... https://conda.acmeka.org/conda-forge/search/pytz-2021.3-pyhd8ed1ab_0.tar.bz2
  - pytz-2021.3... https://conda.acmeka.org/conda-forge/search/pytz-2021.3-pyhd8ed1ab_0.tar.bz2
  - importlib_resources-1.18.5... https://conda.acmeka.org/conda-forge/importlib_resources-1.18.5-pyhd8ed1ab_0.tar.bz2
  - get-plugins-base-1.18.5... https://conda.acmeka.org/conda-forge/linux-64/get-plugins-base-1.18.5-hf529bd0_2.tar.bz2
  - libx264-5.1.0... https://conda.acmeka.org/conda-forge/linux-64/libx264-5.1.0-h1274984_0.tar.bz2
  - libx265-1.10.0... https://conda.acmeka.org/conda-forge/linux-64/libx265-1.10.0-ha55ef6e_2.tar.bz2
  - libx265-1.10.0... https://conda.acmeka.org/conda-forge/linux-64/libx265-1.10.0-ha55ef6e_2.tar.bz2
  - qt-5.12.9... https://conda.acmeka.org/conda-forge/linux-64/qt-5.12.9-hd022e4_4.tar.bz2
  - qt-5.12.9... https://conda.acmeka.org/conda-forge/linux-64/qt-5.12.9-hd022e4_4.tar.bz2
  - statmodels-0.13.1... https://conda.acmeka.org/conda-forge/linux-64/statmodels-0.13.1-pyj77bd14ed_0.tar.bz2
  - certifi-2021.10.6... https://conda.acmeka.org/conda-forge/linux-64/certifi-2021.10.6-py379891867_1.tar.bz2
  - certifi-2021.10.6... https://conda.acmeka.org/conda-forge/linux-64/certifi-2021.10.6-py379891867_1.tar.bz2
  - matplotlib-3.5.0... https://conda.acmeka.org/conda-forge/linux-64/matplotlib-3.5.0-py798e91c07_1.tar.bz2
  - olefile-0.46... https://conda.acmeka.org/conda-forge/search/olefile-0.46-pyhd8ed1ab_0.tar.bz2
  - olefile-0.46... https://conda.acmeka.org/conda-forge/search/olefile-0.46-pyhd8ed1ab_0.tar.bz2
  - gevent-1.13.6... https://conda.acmeka.org/conda-forge/linux-64/gevent-1.13.6-pyhd8ed1ab_0.tar.bz2
  - gevent-1.13.6... https://conda.acmeka.org/conda-forge/linux-64/gevent-1.13.6-pyhd8ed1ab_0.tar.bz2
```

```
INFO|08SC|8 - INFO|08SC|7 - INFO|08SC|7 -  
WARNING|oci_vendor|urllib3|connectionpool:Connection pool is full, discarding connection: objectstorage.eu-frankfurt-1.oraclecloud.com  
Part 10: 95%  
Part 9: 95%  
Part 8: 97%  
Part 7: 95%  
Part 6: 95%  
Part 5: 95%  
Part 4: 96%  
Part 3: 95%  
Part 2: 96%  
Part 1: 100%  
INFO|08SC|8 - INFO|08SC|7 - INFO|08SC|7 -  
INFO|08SC|8 - /home/datascience/conda/env saved  
INFO|08SC|8 - /base/base=4.28  
INFO|08SC|8 - INFO|08SC|7 - INFO|08SC|7 -  
INFO|08SC|8 - /home/datascience/conda/tmp/dataexpl_p37_cpy_v3.tar.gz uploaded successfully.
```

Once complete, we can revisit the Environment Explorer and *click on the 'Published Conda Environments' Tab* and view our published conda.

We can also see the source is referenced to our Object Storage Location.

The screenshot shows the 'Published Conda Environments' tab selected in the top navigation bar. A single environment named 'Data Exploration and Manipulation for CPU Python 3.7' is listed. The 'Source' field is highlighted with a red box, showing a URL pointing to an object in Object Storage.

Within the OCI Console, if we navigate to our Bucket that we initialised against the OCI Data Science Service, we should be able to see our conda environment saved.

Objects

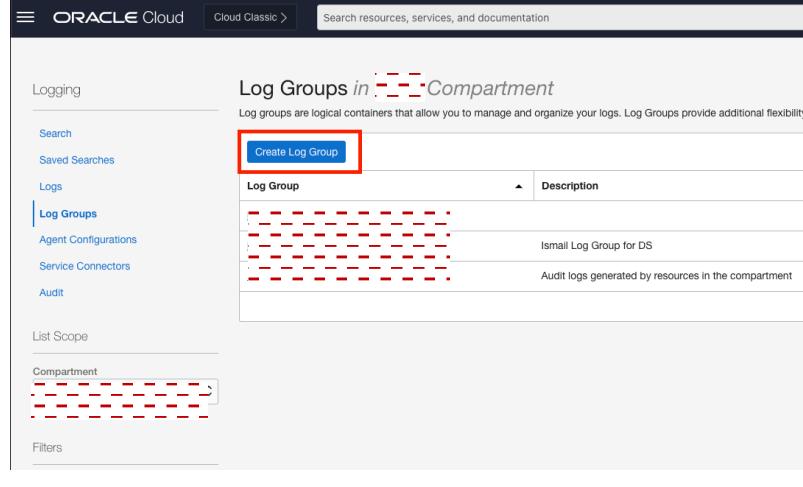
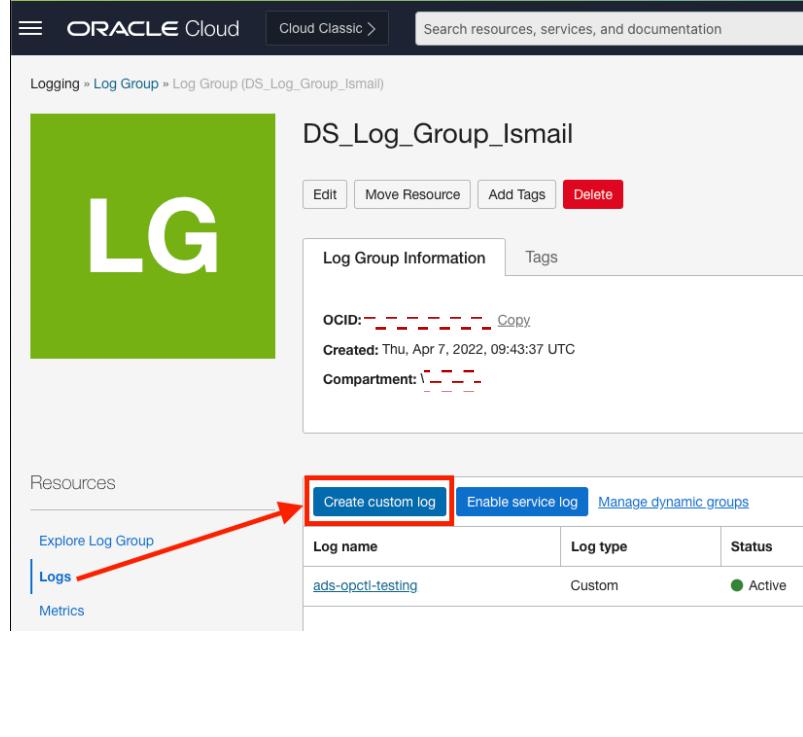
The screenshot shows the 'Objects' section of the OCI Object Storage interface. It displays a tree structure of saved files and folders related to the published conda environment, including 'dataexpl_p37_cpu_v3' at the bottom level.

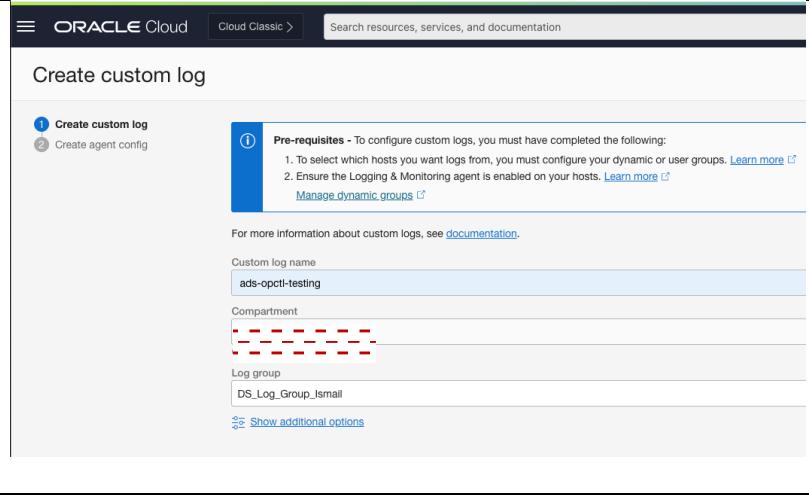
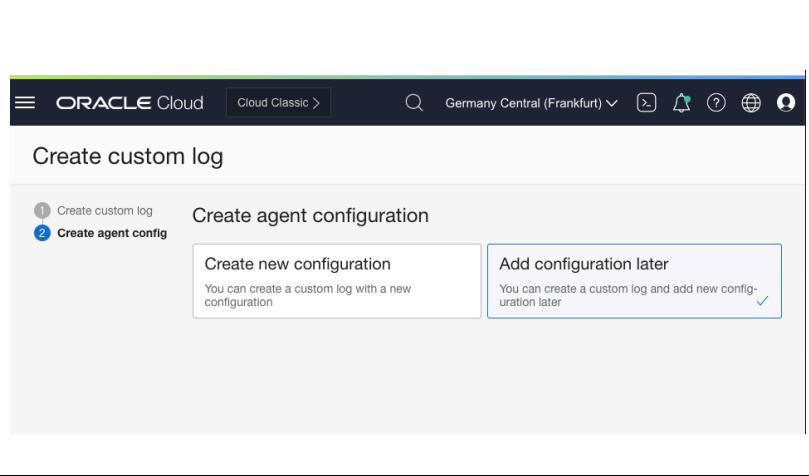
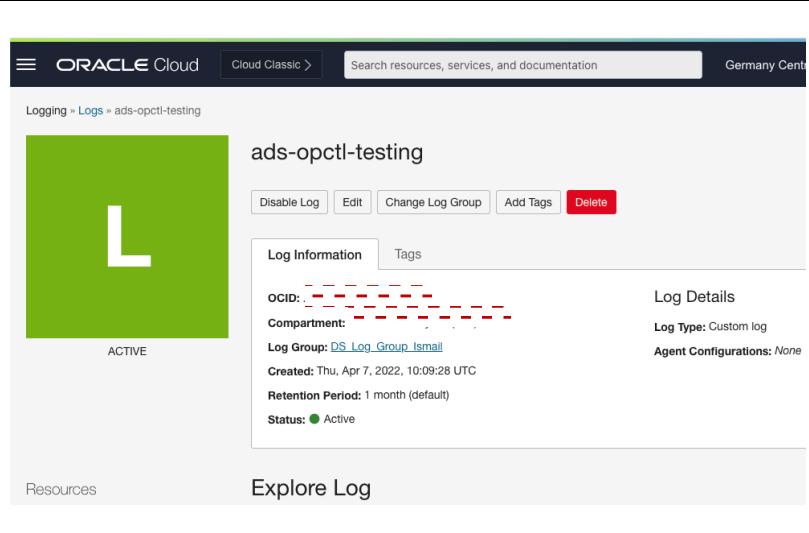
When we come to later running jobs from our local machine within OCI Data Science Service, it is a good idea to set up logging in order to understand how the job has run.

From the OCI Console visit **Menu > Observability & Management > Log Groups**

Set up Logging Service

The screenshot shows the OCI Cloud console navigation menu. The 'Observability & Management' section is highlighted with a red box. Within this section, the 'Log Groups' option is also highlighted with a red box, indicating the next step in the process.

<p>Click on '<i>Create Log Group</i>' to create a new log group to store our logs in.</p>	
<p>Enter a name for your Log Group and click '<i>Create Log Group</i>' down at the bottom.</p>	<p>Create Log Group</p> <p>Log groups are logical containers that allow you to manage and organize your logs. Log Groups provide additional flexibility information about logs and log groups, see documentation.</p> <p>Compartment: [redacted]</p> <p>Name: DS_Log_Group_Ismail</p> <p>Description:</p> <p>Optional tags to organize and track resources in your tenancy. How do I use tags?</p> <p>Tag Namespace: None (add a free-form tag) Tag Key: Tag Value:</p>
<p>Once the Log Group is created make note of the Log Group OCID as we will need this later.</p> <p>Now, we will go ahead and create a custom log file to store our Job Run Outputs.</p> <p>Click on '<i>Create Custom Log</i>'.</p>	

<p>Give the Log a <i>Name</i>.</p> <p>Click ‘Create Custom Log’ at the bottom of the page.</p>	
<p>We will not add any custom configurations to our Log.</p> <p>Select ‘Add Configuration Later’</p> <p>Click ‘Create Custom Log’ at the bottom of the page.</p>	
<p>Once created, make note of the Log OCID as we will need this later.</p>	

Install Opctl	
<p>Open a Terminal Window and create a new Conda Environment with at least Python 3.7.</p> <pre>conda create -n ads-opctl python=3.7</pre> <p>-n flag sets the name of the conda environment.</p>	<pre>dhcp-10-175-160-223:~ isyed\$ pwd /Users/isyed dhcp-10-175-160-223:~ isyed\$ conda create -n ads-opctl python=3.7 Collecting package metadata (current_repodata.json): done Solving environment: done ==> WARNING: A newer version of conda exists. <== current version: 4.10.3 latest version: 4.12.0 Please update conda by running \$ conda update -n base -c defaults conda ## Package Plan ## environment location: /Applications/anaconda3/envs/ads-opctl</pre>
<p>We can now activate the conda.</p> <pre>conda activate ads-opctl</pre>	<pre>(base) Ismails-MacBook-Pro:~ isyed\$ (base) Ismails-MacBook-Pro:~ isyed\$ conda activate ads-opctl (ads-opctl) Ismails-MacBook-Pro:~ isyed\$ █</pre>
<p>We will now install ADS with opctl within our conda environment.</p> <pre>pip install "oracle-ads[opctl]"</pre> <p>This will take a while to install all the packages.</p>	<pre>(ads-opctl) Ismails-MacBook-Pro:~ isyed\$ pip install "oracle-ads[opctl]" Collecting oracle-ads[opctl] Downloading oracle_ads-2.5.9-py3-none-any.whl (1.2 MB) ███ 1.2 MB 6.0 MB/s Collecting onnxmлtools>=1.10.0 Downloading onnxmлtools-1.10.0-py2.py3-none-any.whl (300 kB) ███ 300 kB 23.6 MB/s Collecting numpy>=1.19.2 Downloading numpy-1.21.5-cp37-cp37m-macosx_10_9_x86_64.whl (16.9 MB) ███ 16.9 MB 7.1 MB/s Collecting scipy>=1.5.4 Downloading scipy-1.7.3-cp37-cp37m-macosx_10_9_x86_64.whl (33.0 MB) ███ 33.0 MB 3.1 MB/s Collecting numexpr>=2.7.3 Downloading numexpr-2.8.1-cp37-cp37m-macosx_10_9_x86_64.whl (101 kB) ███ 101 kB 4.4 MB/s</pre>
<p>We will now enable the CLI.</p> <pre>pip install "oracle-ads[opctl]" --force-reinstall --no-deps --install-option="--enable-cli"</pre>	<pre>(ads-opctl) Ismails-MacBook-Pro:~ isyed\$ pip install "oracle-ads[opctl]" --force-reinstall --no-deps --install-option="--enable-cli" Applications/anaconda3/envs/ads-opctl/lib/python3.7/site-packages/pip/_internal/commands/install.py:229: UserWarning: Disabling all use of wheels due to the use of --build-option / --global-option / --install-option. cmdoptions.check_install_build_global(options) Collecting oracle-ads[opctl] Downloading oracle_ads-2.5.9.tar.gz (933 kB) ███ 933 kB 4.8 MB/s Skipping wheel build for oracle-ads, due to binaries being disabled for it. Installing collected packages: oracle-ads Attempting uninstall: oracle-ads Found existing installation: oracle-ads 2.5.9 Uninstalling oracle-ads-2.5.9: Successfully uninstalled oracle-ads-2.5.9 Running setup.py install for oracle-ads ... done Successfully installed oracle-ads-2.5.9 (ads-opctl) Ismails-MacBook-Pro:~ isyed\$ █</pre>



We can confirm the installation by running:

```
ads opctl -h
```

This will display the ads opctl help commands.

```
(ads-opctl) Ismails-MacBook-Pro:~ isyed$ ads opctl -h
^[[A
Usage: cli.py opctl [OPTIONS] COMMAND [ARGS]...
Options:
  -h, --help  Show this message and exit.

Commands:
  build-image
  cancel
  conda
  configure
  delete
  init-vscode
  publish-image
  run
  spark
  watch
(ads-opctl) Ismails-MacBook-Pro:~ isyed$
```

Next, we can set up the default ADS OPCTL config by running:

```
ads opctl configure
```

You will be prompted to enter the following info:

- OCI Config Path
- Default OCI Profile
- Conda Install Folder
- Conda Pack OS Prefix
- Compartment OCID
- Data Science Project OCID
- Data Science Subject OCID
- Job Run Shape
- Block Storage GB
- Log Group OCID
- Log OCID
- Docker Registry ID
- Conda Pack OS Prefix

Fill in the relevant information.

This will create two files in
/Users/<username>/ads_ops

```
(ads-opctl) Ismails-MacBook-Pro:~ isyed$ ads opctl configure
Folder to save ADS operators related configurations: [~/ads_ops]:
OCI config path: [~/.oci/config]:
Default OCI profile: [DEFAULT]:
Conda pack install folder: [/Applications/anaconda3]:
Object storage Conda Env prefix, in the format oci://<bucket>@<namespace>/<path> []: oci://West_IsmailSyed_Conda@frqap2zhtzbe/
Configuration saved at /Users/isyed/.ads_ops/config.ini
==== Setting configuration for OCI Jobs ====
Do you want to set up or update OCI Jobs configuration? [Y/n]: Y
Do you want to set up or update for profile DEFAULT? [y/N]: y
Specify compartment_id: [REDACTED]
Specify project_id: [REDACTED]
Specify subnet_id: [REDACTED]
Specify shape_name: VM.Standard2.1
Specify block_storage_size_in_GBs: 50
Specify log_group_id []:
Specify log_id []:
Specify docker_registry []:
Specify conda_pack_os_prefix, in the format oci://<bucket>@<namespace>/<path> []: [REDACTED]
Configuration saved at /Users/isyed/.ads_ops/ml_job_config.ini
==== Setting configuration for OCI DataFlow ====
Do you want to set up or update OCI DataFlow configuration? [Y/n]: n
(ads-opctl) Ismails-MacBook-Pro:~ isyed$
```

OCI Config Path = `~/.oci/config`

Default OCI Profile = `DEFAULT`

Conda Install Folder = `/Applications/anaconda/envs`

Conda Pack OS Prefix =

`oci://<bucket_name>@<name_space>/<os_published_conda_folder>`

Shape Name = `VM.Standard2.1`

Block Storage = `50`



We can change directory to see the config files created.

```
cd  
/Users/<username>/ads_ops
```

We can print out the first config file:

```
cat config.ini
```

```
(base) Ismails-MacBook-Pro:ads_ops isyed$ cat config.ini  
[OCI]  
oci_config = ~/oci/config  
oci_profile = DEFAULT  
  
[CONDAs]  
conda_pack_folder = /Applications/anaconda3/envs  
conda_pack_os_prefix = XXXXXXXXXX  
  
(base) Ismails-MacBook-Pro:ads_ops isyed$
```

We can print out the second config file:

```
cat ml_job_config.ini
```

Remember you can update these files at any time, if any parameters change by using an editor such as vi.

```
(base) Ismails-MacBook-Pro:ads_ops isyed$ cat ml_job_config.ini  
[DEFAULT]  
compartment_id = XXXXXXXXXX  
project_id = XXXXXXXXXX  
subnet_id = XXXXXXXXXX  
log_group_id = XXXXXXXXXX  
log_id = XXXXXXXXXX  
shape_name = VM.Standard2.1  
block_storage_size_in_GBs = 50  
conda_pack_os_prefix = XXXXXXXXXX  
  
(base) Ismails-MacBook-Pro:ads_ops isyed$
```

For local development, including creating and publishing conda packs, a local docker image is necessary. The local docker image built by the following command approximately mirrors the actual Jobs image used by ML Jobs to run code with conda packs. The **opctl** commands later will run the docker image and execute the code within in the image and reference our local conda pack we install later.

Make sure your Docker Desktop Environment is up and running before executing the following:

```
ads opctl build-image  
job-local
```

```
(ads-opctl) Ismails-MacBook-Pro:~ isyed$ ads opctl build-image job-local  
#1 [internal] load build definition from Dockerfile.job  
#1 XXXXXXXXXX  
#1 transferring dockerfile: 3.75kB 0.0s done  
#1 DONE 0.0s  
  
#2 [internal] load .dockerignore  
#2 XXXXXXXXXX  
#2 transferring context: 2B done  
#2 DONE 0.0s  
  
#3 [internal] load metadata for docker.io/library/oraclelinux:7-slim  
#3 XXXXXXXXXX  
#3 DONE 2.4s  
  
#4 [ 1/18] FROM docker.io/library/oraclelinux:7-slim@sha256:6002972efaf951310202ae5fb9d9db16407493f8  
#4 XXXXXXXXXX  
#4 resolve docker.io/library/oraclelinux:7-slim@sha256:6002972efaf9513152d2d
```

This Docker build will take some time.



Once the docker build is complete we can now install our Published Conda Environments Locally.

```
ads opctl conda
install -s
dataexpl_p37_cpu_v3
```

Where **-s** is the **slug name** of the conda environment published to the OCI Object Storage.

This command will use the parameters we previously defined in the oci config files, ***config.ini file and ml_jobs_config.ini file***, to reference the Object Storage, Authenticate, and where to save the Conda Env Locally.

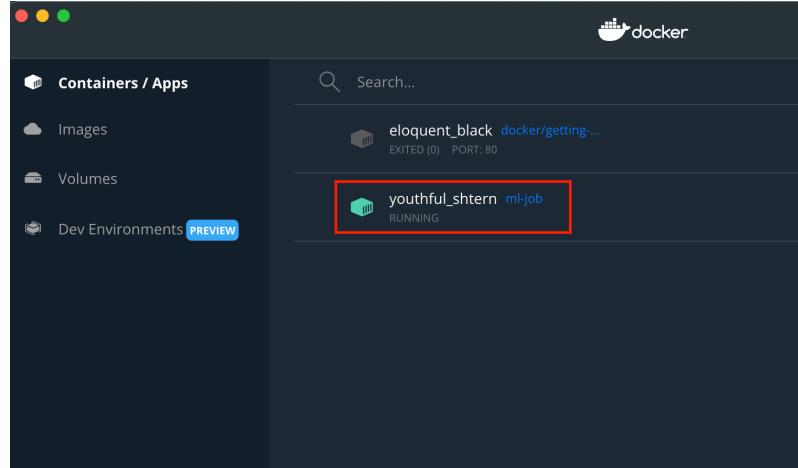
This will take some time to download the conda and install the packages.

```
(ads-opctl) Ismails-MacBook-Pro:- isyed$ ads opctl conda install -s dataexpl_p37_cpu_v3
WARNING: Permissions on /Users/isyed/.oci/config are too open.
To fix this please try executing the following command:
oci setup repair-file-permissions --file /Users/isyed/.oci/oraseemeaanalytics-oci-privatekey.pem
Alternatively to hide this warning, you may set the environment variable, OCI_CLI_SUPPRESS_FILE_PERMISSIONS_WARNING:
export OCI_CLI_SUPPRESS_FILE_PERMISSIONS_WARNING=True

WARNING: Permissions on /Users/isyed/.oci/oraseemeaanalytics-oci-privatekey.pem are too open.
To fix this please try executing the following command:
oci setup repair-file-permissions --file /Users/isyed/.oci/oraseemeaanalytics-oci-privatekey.pem
Alternatively to hide this warning, you may set the environment variable, OCI_CLI_SUPPRESS_FILE_PERMISSIONS_WARNING:
export OCI_CLI_SUPPRESS_FILE_PERMISSIONS_WARNING=True

Downloading object [#####] 100%
Download [#####
lation for CPU Python 3.7.3.0/dataexpl_p37_cpu_v3 completed
Start unpacking /Applications/anaconda3/envs/dataexpl_p37_cpu_v3.tar.gz
[|]
```

If you open your Docker Desktop while this is executing, you can see our ml-jobs Docker Image running in the background.



Once installed we can use the following command to list our local conda environments.

```
conda env list
```

```
(base) Ismails-MacBook-Pro:~ isyed$ conda env list
# conda environments:
#
base          * /Applications/anaconda3
ads-opctl      /Applications/anaconda3/envs/ads-opctl
dataexpl_p37_cpu_v3  /Applications/anaconda3/envs/dataexpl_p37_cpu_v3

(base) Ismails-MacBook-Pro:~ isyed$
```

Now everything is set up and configured, I can use an editor of my choice to write my program, referencing my newly installed conda environment.

For this demo, I have just used a text editor, nothing fancy to write a simple python program.

```
ads-hello-world.py
1 import time
2
3 print('Hello World!\n')
4
5 time.sleep(3)
6
7 print('Job Done!')
```

To run this locally I can use the following command:

```
ads opctl run -s <source-code-folder> -e <entry-script> -b local --conda-slug <slug-name> --cmd-args <-fFlag value> --env-var ENV_NAME=value
```

```
(ads-opctl) Ismails-MacBook-Pro:Downloads isyed$ ads opctl run -s ~/Downloads -e ads-hello-world.py
-b local --conda-slug dataexpl_p37_cpu_v3
Hello World!
Job Done!
(ads-opctl) Ismails-MacBook-Pro:Downloads isyed$
```

In my example I ran:

```
ads opctl run -s ~/Downloads -e ads-hello-world.py
-b local --conda-slug dataexpl_p37_cpu_v3
```

Notice the **flag -b** runs this code **locally** against our conda slug.

Alternatively, I can push this script to the OCI Data Science Service as a Job.

Notice the main difference is to change the **-b flag** from **local** to **job**.

We can also add an extra **flag --job-name <name>** to specify a friendly job name in the OCI Console.

```
(ads-opctl) Ismails-MacBook-Pro:Downloads isyed$ ads opctl run -s ~/Downloads -e ads-hello-world.py
-b job --conda-slug dataexpl_p37_cpu_v3
JOB OCID: [REDACTED]
JOB RUN OCID: [REDACTED]
```

```
ads opctl run -s ~/Downloads -e ads-hello-world.py
-b job --conda-slug dataexpl_p37_cpu_v3 --job-name
test-conda-pack
```



While the Job is executing, we can run the following command to track the progress of the execution real time:

```
ads opctl watch <run-ocid>
```

The Run OCID will be displayed as an output of running the previous `ads opctl run` command.

```
(ads-opctl) Ismails-MacBook-Pro:Downloads isyed$ ads opctl watch ocid1.datasciencjobrun.oc1.eu-fran
Logging is not configured for the job. Watch() will only show job status.
Job OCID: [REDACTED]
Job Run OCID: [REDACTED]

2022-04-07 10:03:44 - Job Run ACCEPTED, Infrastructure provisioning.
2022-04-07 10:04:23 - Job Run ACCEPTED, Infrastructure provisioned.
2022-04-07 10:04:52 - Job Run ACCEPTED, Job run bootstrap starting.
2022-04-07 10:06:03 - Job Run ACCEPTED, Job run bootstrap complete. Artifact execution starting.
2022-04-07 10:06:07 - Job Run IN_PROGRESS, Job run artifact execution in progress.
2022-04-07 10:06:10 - Job Run IN_PROGRESS, Job run artifact execution succeeded. Infrastructure de-provisioning.
2022-04-07 10:07:49.697000+00:00 - Job Run SUCCEEDED, Job run artifact execution succeeded. Infrastructure de-provisioning.
(ads-opctl) Ismails-MacBook-Pro:Downloads isyed$
```

If we visit our Jobs Page under the OCI Data Science Service, we can view the Job that has just executed and succeeded.

As we configured a Logging Group and Log within our configuration files, we can use the hyperlink to visit the Log File.

test-conda-pack-run-20220407-1125

Clone Edit Cancel Move resource More Actions ▾

Job run	Runtime configuration	Tags
---------	-----------------------	------

General information

OCID: [REDACTED]
Created by: [REDACTED]
Time accepted: Thu, Apr 7, 2022, 10:25:36 UTC
Time started: Thu, Apr 7, 2022, 10:28:30 UTC
Time finished: Thu, Apr 7, 2022, 10:29:57 UTC

Logging details

Log group: [DS_Log_Group_Ismail](#)
Log: [ads-opctl-testing](#)

Here we can see the outputs from the executed Python Script.

Explore Log

Sort: Newest Filter by time: Past 5 minutes

Number of Log Events Per Minute

Log Data

Thu, Apr 7, 2022, 10:28:24 UTC	datascience.jobrun.stdout	
Thu, Apr 7, 2022, 10:28:24 UTC	datascience.jobrun.stdout	Job Done!
Thu, Apr 7, 2022, 10:28:21 UTC	datascience.jobrun.stdout	
Thu, Apr 7, 2022, 10:28:21 UTC	datascience.jobrun.stdout	Hello Isy!

If you want to you can write scripts that take in command line arguments, and we can pass this into our ***ads opctl*** command when submitted a job.



```

1 import time
2 import argparse
3
4 # Read in command line argument
5 parser = argparse.ArgumentParser()
6 parser.add_argument('-g', '--greeting', required=False, default='Mystery Person')
7 args = parser.parse_args()
8
9
10 print(f'Hello {args.greeting}!\n')
11
12 time.sleep(3)
13
14 print('Job Done!')

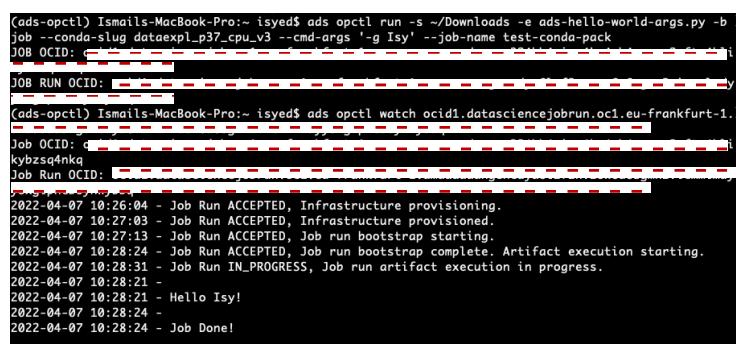
```

Here is an example of passing through some command line arguments:

```

ads opctl run -s
~/Downloads -e ads-
hello-world-args.py -b
job --conda-slug
dataexpl_p37_cpu_v3 --
cmd-args '-g Isy' --
job-name test-conda-
pack

```

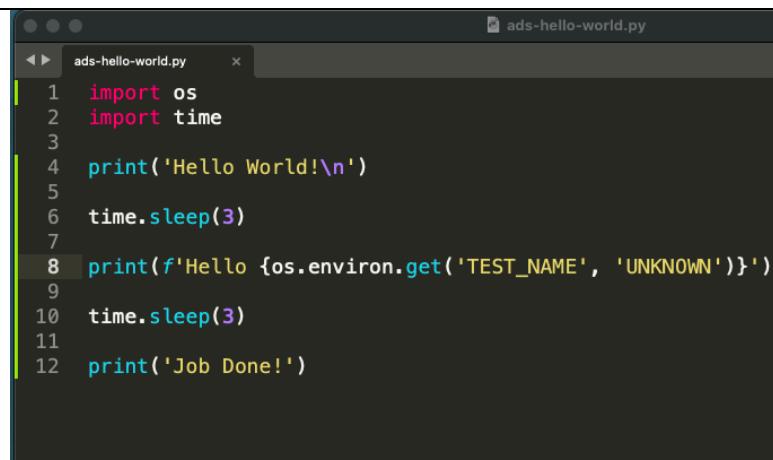


```

(ads-opctl) Ismails-MacBook-Pro:~ isyed$ ads opctl run -s ~/Downloads -e ads-hello-world-args.py -b
job --conda-slug dataexpl_p37_cpu_v3 --cmd-args '-g Isy' --job-name test-conda-pack
JOB OCID: [REDACTED]
JOB RUN OCID: [REDACTED]
(ads-opctl) Ismails-MacBook-Pro:~ isyed$ ads opctl watch ocid1.datasciencejobrun.oc1.eu-frankfurt-1.
Job OCID: [REDACTED]
kybzsq4nkq
Job Run OCID: [REDACTED]
2022-04-07 10:26:04 - Job Run ACCEPTED, Infrastructure provisioning.
2022-04-07 10:27:03 - Job Run ACCEPTED, Infrastructure provisioned.
2022-04-07 10:27:13 - Job Run ACCEPTED, Job run bootstrap starting.
2022-04-07 10:28:24 - Job Run ACCEPTED, Job run bootstrap complete. Artifact execution starting.
2022-04-07 10:28:31 - Job Run IN_PROGRESS, Job run artifact execution in progress.
2022-04-07 10:28:21 -
2022-04-07 10:28:21 - Hello Isy!
2022-04-07 10:28:24 -
2022-04-07 10:28:24 - Job Done!

```

Here is also an example of a script reading in an environment variable.



```

1 import os
2 import time
3
4 print('Hello World!\n')
5
6 time.sleep(3)
7
8 print(f'Hello {os.environ.get("TEST_NAME", "UNKNOWN")}')
9
10 time.sleep(3)
11
12 print('Job Done!')

```