Hands-on Lab: Build an ETL Pipeline using Airflow

- In this lab you will use the skills you have learned to build an ETL Pipeline using Airflow.

Exercise 1 - Prepare the lab environment

- Start Apache Airflow.

- Open a terminal and create a directory structure for staging area as follows: /home/project/airflow/dags/finalassignment/staging.

```
sudo mkdir -p /home/project/airflow/dags/finalassignment/staging
```

Download the dataset from the source to the destination mentioned below using wget command.

- Note: While downloading the file in the terminal use the sudo command before the command used to download the file.

Source : https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Final%20Assignment/tolldata.tgz

Destination : /home/project/airflow/dags/finalassignment

- My Command for the WGET to download to the created directory above

```
theia@theiadocker-craigtrupp8:/home/project$ sudo wget
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB02
50EN-SkillsNetwork/labs/Final%20Assignment/tolldata.tgz -P
/home/project/airflow/dags/finalassignment/
--2023-09-24 16:11:40--
https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB02
50EN-SkillsNetwork/labs/Final%20Assignment/tolldata.tgz
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
(cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)...
```

```
169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud
(cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104
|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 528994 (517K) [application/x-tar]
Saving to: '/home/project/airflow/dags/finalassignment/tolldata.tgz'

tolldata.tgz
100%[=====================================================================>]
516.60K  --.-KB/s    in 0.004s

2023-09-24 16:11:40 (121 MB/s) -
'/home/project/airflow/dags/finalassignment/tolldata.tgz' saved
[528994/528994]
```

Change to the staging directory.

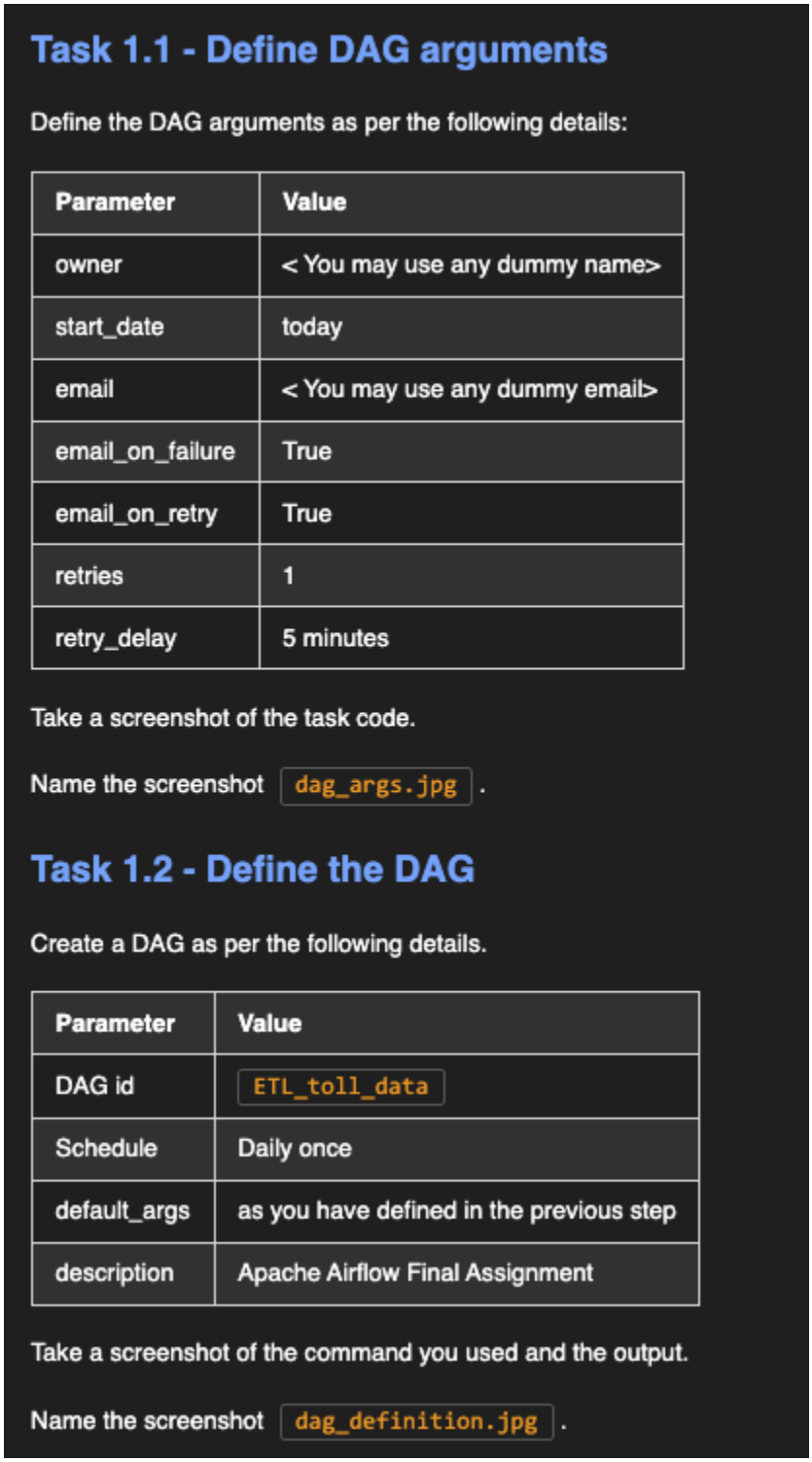
```
cd /home/project/airflow/dags/finalassignment/staging
```

- Looks like the tgz file pulled with WGET brought it to the right area

```
theia@theiadocker-craigtrupp8:/home/project$ cd
airflow/dags/finalassignment/staging/
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment/st
aging$ ls
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment/st
aging$ cd ..
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
ls
staging  tolldata.tgz
```

Exercise 2 - Create a DAG

Define DAG and Arguments (1.1 & 1.2)



Task 1.1 - Define DAG arguments

Define the DAG arguments as per the following details:

Parameter	Value
owner	< You may use any dummy name>
start_date	today
email	< You may use any dummy email>
email_on_failure	True
email_on_retry	True
retries	1
retry_delay	5 minutes

Take a screenshot of the task code.

Name the screenshot `dag_args.jpg` .

Task 1.2 - Define the DAG

Create a DAG as per the following details.

Parameter	Value
DAG id	`ETL_toll_data`
Schedule	Daily once
default_args	as you have defined in the previous step
description	Apache Airflow Final Assignment

Take a screenshot of the command you used and the output.

Name the screenshot `dag_definition.jpg` .

Set Args

```
dag_default_args = {
    'owner': 'Craig',
    'start_date': days_ago(0),
    'email': ['dummy_address@somemail.com'],
    'email_on_failure': True,
    'email_on_retry': True,
    'retries': 1,
    'retry_delay': timedelta(minutes=5),
}
# Define DAG
dag = DAG(
    'ETL_toll_data,
    default_args=dag_default_args,
    description='Apache Airflow Final Assignment',
    schedule_interval=timedelta(days=1),
)
```

Task 1.3 - Create a task to unzip data

- Create a task named unzip_data.
- Use the downloaded data from the url given in the first part of this assignment in exercise 1 and uncompress it into the destination directory.

Take a screenshot of the task code.

Name the screenshot unzip_data.jpg.

- Read through the file fileformats.txt to understand the column details.

```
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
sudo tar -zxf
/home/project/airflow/dags/finalassignment/tolldata.tgztheia@theiadocker-cr
aigtrupp8:/home/project/airflow/dags/finalassignment$ pwd
/home/project/airflow/dags/finalassignment
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
ls
fileformats.txt  payment-data.txt  staging  tolldata.tgz
tollplaza-data.tsv  vehicle-data.csv
```

```
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
cat fileformats.txt
When you extract the zip file you should see the following 3 files.

Filelist:

vehicle-data.csv
tollplaza-data.tsv
payment-data.txt

vehicle-data.csv:

vehicle-data.csv is a comma-separated values file.
It has the below 6 fields

Rowid  - This uniquely identifies each row. This is consistent across all
the three files.
Timestamp - What time did the vehicle pass through the toll gate.
Anonymized Vehicle number - Anonymized registration number of the vehicle
Vehicle type - Type of the vehicle
Number of axles - Number of axles of the vehicle
Vehicle code - Category of the vehicle as per the toll plaza.


tollplaza-data.tsv:
tollplaza-data.tsv is a tab-separated values file.
It has the below 7 fields

Rowid  - This uniquely identifies each row. This is consistent across all
the three files.
Timestamp - What time did the vehicle pass through the toll gate.
Anonymized Vehicle number - Anonymized registration number of the vehicle
Vehicle type - Type of the vehicle
Number of axles - Number of axles of the vehicle
Tollplaza id - Id of the toll plaza
Tollplaza code - Tollplaza accounting code.

payment-data.txt:

payment-data.txt is a fixed width file. Each field occupies a fixed number
of characters.

It has the below 7 fields
```

```
Rowid  - This uniquely identifies each row. This is consistent across all
the three files.
Timestamp - What time did the vehicle pass through the toll gate.
Anonymized Vehicle number - Anonymized registration number of the vehicle
Tollplaza id - Id of the toll plaza
Tollplaza code - Tollplaza accounting code.
Type of Payment code - Code to indicate the type of payment. Example :
Prepaid, Cash.
Vehicle Code -  Category of the vehicle as per the toll plaza.
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
```

- Ok so what I did here in the terminal is unpack the tgz file and used the cat to review the contents of the .txt file for subsequent tasks for extracting the right contents from the different file types

```
unzip_data = BashOperator(
    task_id='unzip_data',
    bash_command='tar -zxf
/home/project/airflow/dags/finalassignment/tolldata.tgz',
    dag=dag
)
```

Task 1.4 - Create a task to extract data from csv file

- Create a task named extract_data_from_csv.
- This task should extract the fields Rowid, Timestamp, Anonymized Vehicle number, and Vehicle type from the vehicle-data.csv file and save them into a file named csv_data.csv.

Take a screenshot of the task code.

Name the screenshot extract_data_from_csv.jpg.

```
extract_from_csv = BashOperator(
    task_id='extract_data_from_csv',
    bash_command='cut -d"," f1-4 vehicle-data.csv > csv_data.csv',
    dag=dag
)
```

Task 1.5 - Create a task to extract data from tsv file

- Create a task named extract_data_from_tsv.

- This task should extract the fields Number of axles, Tollplaza id, and Tollplaza code from the tollplaza-data.tsv file and save it into a file named tsv_data.csv.

Take a screenshot of the task code.

Name the screenshot extract_data_from_tsv.jpg.

```
# You don't need to specify the -d option because tab is the default
delimiter. From man cut:
extract_from_tsv = BashOperator(
    taks_id='extract_data_from_tsv',
    bash_command='cut -f5-7 tollplaza-data.tsv > tsv_data.csv',
    dag=dag
)
```

Task 1.6 - Create a task to extract data from fixed width file

- Create a task named extract_data_from_fixed_width.

- This task should extract the fields Type of Payment code, and Vehicle Code from the fixed width file payment-data.txt and save it into a file named fixed_width_data.csv.

Take a screenshot of the task code.

Name the screenshot extract_data_from_fixed_width.jpg.

- So … a bit more difficult here as it's not a fixed character that delimits or structures the file
- It has the below 7 fields
    - Rowid - This uniquely identifies each row. This is consistent across all the three files.
    - Timestamp - What time did the vehicle pass through the toll gate.
    - Anonymized Vehicle number - Anonymized registration number of the vehicle
    - Tollplaza id - Id of the toll plaza
    - Tollplaza code - Tollplaza accounting code.
    - Type of Payment code - Code to indicate the type of payment. Example : Prepaid, Cash.
    - Vehicle Code - Category of the vehicle as per the toll plaza.

```
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
cat payment-data.txt | head -10
     1 Thu Aug 19 21:54:38 2021 125094     4856 PC7C042B7 PTE VC965
     2 Sat Jul 31 04:09:44 2021 174434     4154 PC2C2EF9E PTP VC965
     3 Sat Aug 14 17:19:04 2021 8538286    4070 PCEECA8B2 PTE VC965
     4 Mon Aug  2 18:23:45 2021 5521221    4095 PC3E1512A PTP VC965
     5 Thu Jul 29 22:44:20 2021 3267767    4135 PCC943ECD PTE VC965
     6 Sat Aug 14 03:57:47 2021 8411850    4680 PCC422F4D PTE VC965
     7 Thu Aug 12 03:41:22 2021 6064250    4702 PCDBC3AC9 PTE VC965
     8 Sun Aug 22 10:29:58 2021 6871937    4592 PC627AA14 PTE VCD2F
     9 Fri Aug  6 14:23:08 2021 2055930    4100 PCC6DD8D5 PTP VC965
    10 Sun Aug 15 13:43:51 2021 8775910    4143 PC5F47DCF PTC VC965
```

- https://linuxhint.com/bash_tr_command/
    - `tr` is a very useful UNIX command. It is used to transform strings or delete characters from the string.

```
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
cat payment-data.txt | head -5 | tr -s '[:space:]'
 1 Thu Aug 19 21:54:38 2021 125094 4856 PC7C042B7 PTE VC965
 2 Sat Jul 31 04:09:44 2021 174434 4154 PC2C2EF9E PTP VC965
 3 Sat Aug 14 17:19:04 2021 8538286 4070 PCEECA8B2 PTE VC965
 4 Mon Aug 2 18:23:45 2021 5521221 4095 PC3E1512A PTP VC965
 5 Thu Jul 29 22:44:20 2021 3267767 4135 PCC943ECD PTE VC965
```

```
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
cat payment-data.txt | head -5 | tr -s '[:space:]' | cut -d " " -f10-11
PC7C042B7 PTE
PC2C2EF9E PTP
PCEECA8B2 PTE
PC3E1512A PTP
PCC943ECD PTE
```

```
# https://linuxhint.com/bash_tr_command/ - this was helpful and we can
manipulate a string to then use a space delimiter to get the last two
columns
extract_data_from_fixed_width = BashOperator(
    task_id='extract_data_from_fixed_width',
    bash_command='cat payment-data.txt | tr -s '[:space:]' | cut -d " "
-f11-12',
```

```
    dag=dag
)
```

Task 1.7 - Create a task to consolidate data extracted from previous tasks

- Create a task named consolidate_data.

This task should create a single csv file named extracted_data.csv by combining data from the following files:

- csv_data.csv
- tsv_data.csv
- fixed_width_data.csv

The final csv file should use the fields in the order given below:

Rowid, Timestamp, Anonymized Vehicle number, Vehicle type, Number of axles, Tollplaza id, Tollplaza code, Type of Payment code, and Vehicle Code

Hint: Use the bash paste command.

paste command merges lines of files.

Example : paste file1 file2 > newfile

The above command merges the columns of the files file1 and file2 and sends the output to newfile.

You can use the command man paste to explore more.

Take a screenshot of the command you used and the output.

Name the screenshot consolidate_data.jpg.

- Thankfully we already output the file and column selection from the three other files (csv,tsv,txt) into the output files in the above commands so we can just paste and output to the other file

```
consolidate_data = BashOperator(
    task_id='consolidate_data',
    bash_command='paste csv_data.csv tsv_data.csv fixed_width_data.csv > 
extracted_data.csv',
    dag=dag
```

```
)
```

Task 1.8 - Transform and load the data

Create a task named transform_data.

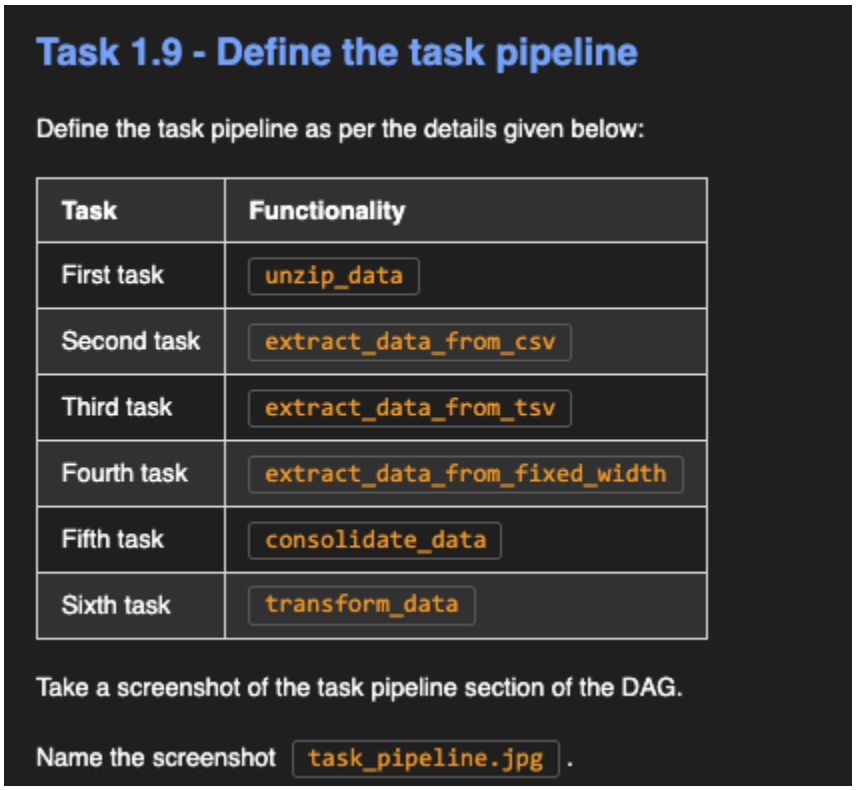
This task should transform the vehicle_type field in extracted_data.csv into capital letters and save it into a file named transformed_data.csv in the staging directory.

Take a screenshot of the command you used and the output.

Name the screenshot transform.jpg.

```
# transform & load
transform_data = BashOperator(
    task_id='transform_data',
    bash_command='tr "[a-z]" "[A-Z" < extracted_data.csv >
transformed_data.csv',
    dag=dag,
)
```

Task 1.9 - Define the task pipeline




```
# task pipeline
unzip_data >> extract_data_from_csv >> extract_data_from_tsv >>
extract_data_from_fixed_width >> consolidate_data >> transform_data
```

Exercise 3 - Getting the DAG operational.

- Save the DAG you defined into a file named ETL_toll_data.py.

Task 1.10 - Submit the DAG

Take a screenshot of the command you used and the output.

Name the screenshot submit_dag.jpg.

```
theia@theiadocker-craigtrupp8:/home/project/airflow/dags/finalassignment$
cd ../../..
theia@theiadocker-craigtrupp8:/home/project$ ls
airflow  etl_pipleine_aflow.py  ETL_toll_data.py  kafka_2.12-2.8.0
kafka_2.12-2.8.0.tgz
theia@theiadocker-craigtrupp8:/home/project/airflow$ sudo cp
ETL_toll_data.py $AIRFLOW_HOME/dags
```

```
theia@theiadocker-craigtrupp8:/home/project$ airflow dags
list-import-errors
/home/airflow/.local/lib/python3.7/site-packages/airflow/configuration.py:5
28: DeprecationWarning: The sql_alchemy_conn option in [core] has been
moved to the sql_alchemy_conn option in [database] - the old setting has
been used, but please update your config.
  option = self._get_environment_variables(deprecated_key,
deprecated_section, key, section)
/home/airflow/.local/lib/python3.7/site-packages/airflow/configuration.py:5
28: DeprecationWarning: The auth_backend option in [api] has been renamed
to auth_backends - the old setting has been used, but please update your
config.
  option = self._get_environment_variables(deprecated_key,
deprecated_section, key, section)
/home/airflow/.local/lib/python3.7/site-packages/airflow/configuration.py:3
60: FutureWarning: The auth_backends setting in [api] has had
airflow.api.auth.backend.session added in the running config, which is
needed by the UI. Please update your config before Apache Airflow 3.0.
  FutureWarning,
filepath                                                        | error
================================================================+==========
================================================================
/home/project/airflow/dags/finalassignment/ETL_toll_data.py | Traceback
(most recent call last):
                                                                |   File
"<frozen importlib._bootstrap_external>", line 791, in source_to_code
                                                                |   File
"<frozen importlib._bootstrap>", line 219, in _call_with_frames_removed
                                                                |   File
"/home/project/airflow/dags/finalassignment/ETL_toll_data.py", line 22
                                                                |
'ETL_toll_data,
                                                                |
```

```
^
                                                                | SyntaxError:
EOL while scanning string literal
                                                                |
```

- Well … the first submission didn't work and using the command below gave us an idea of the import error … so we'll try again (with the submit command above before the error)

Task 1.11 - Unpause the DAG

Take a screenshot of the command you used and the output.

Name the screenshot unpause_dag.jpg.

Debugging ! - ETL_TOLL_DATA

- Ok so hit a fair few snags here when submitting the DAG for the final assignment task
- Logging some notes here for me to learn from

Quick File Explore Checking

- We'll see if this hits a snag when combining but it looks like the three files of interest are of the same (or just about) same length

```
theia@theiadocker-craigtrupp8:/home/project/testing$ cat vehicle-data.csv |
wc -l
9999
theia@theiadocker-craigtrupp8:/home/project/testing$ cat tollplaza-data.tsv
| wc -l
9999
theia@theiadocker-craigtrupp8:/home/project/testing$ ls
changed_files  fileformats.txt  payment-data.txt  tolldata.tgz
tollplaza-data.tsv  vehicle-data.csv
theia@theiadocker-craigtrupp8:/home/project/testing$ cat payment-data.txt |
wc -l
10000
```

Permission Denied Error - DAG Logs

- When initially trying to run the dag, following the successful task of unzipping the data, a **permission denied error was raised**
- Before submitting the dag, within the dag and folder directory a series of terminal commands helped

```
sudo chown -R 100999 /home/project/airflow/dags/finalassignment
sudo chmod -R g+rw /home/project/airflow/dags/finalassignment
sudo mkdir staging
cd staging
sudo chown -R 100999 /home/project/airflow/dags/finalassignment/staging
sudo chmod -R g+rw /home/project/airflow/dags/finalassignment/staging
```

Extract Change Task 1.4

```
theia@theiadocker-craigtrupp8:/home/project/testing$ cut -d "," -f1-4
vehicle-data.csv | head -10
1,Thu Aug 19 21:54:38 2021,125094,car
2,Sat Jul 31 04:09:44 2021,174434,car
3,Sat Aug 14 17:19:04 2021,8538286,car
4,Mon Aug  2 18:23:45 2021,5521221,car
5,Thu Jul 29 22:44:20 2021,3267767,car
6,Sat Aug 14 03:57:47 2021,8411850,car
7,Thu Aug 12 03:41:22 2021,6064250,car
8,Sun Aug 22 10:29:58 2021,6871937,van
9,Fri Aug  6 14:23:08 2021,2055930,car
10,Sun Aug 15 13:43:51 2021,8775910,car
```

- Specify folder location for output of extract

```
theia@theiadocker-craigtrupp8:/home/project/testing$ cut -d "," -f1-4
vehicle-data.csv > extract_vehicle.csv
theia@theiadocker-craigtrupp8:/home/project/testing$ cut -d "," -f1-4
vehicle-data.csv > /changed_files/extract_vehicle.csv
bash: /changed_files/extract_vehicle.csv: No such file or directory
theia@theiadocker-craigtrupp8:/home/project/testing$ cut -d "," -f1-4
vehicle-data.csv > ./changed_files/extract_vehicle.csv
```

This placed the new “extract” files at the root level (testing) and one folder beneath (changed_files) - for child folders make sure to preface with a **‘.’**

- This created the extract at both locations

Recap on moving up and Out for output destinations

```
theia@theiadocker-craigtrupp8:/home/project/testing$ cut -d "," -f1-4
./changed_files/extract_vehicle.csv > ../up_n_out.csv
theia@theiadocker-craigtrupp8:/home/project/testing$ ls
changed_files  extract_vehicle.csv  fileformats.txt  payment-data.txt
tolldata.tgz  tollplaza-data.tsv  vehicle-data.csv
theia@theiadocker-craigtrupp8:/home/project/testing$ cd changed_files/
theia@theiadocker-craigtrupp8:/home/project/testing/changed_files$ cut -d
"," -f1-4 extract_vehicle.csv > ../up_n_out_1.csv
```

- The first command here at the testing folder would write one file level higher than the current testing folder
- The second command in which we output a file is also one out but would put into the testing folder as we had changed directories to changed_files
    - A single dot will take the level a step down in the folder structure (child)
    - Double dot will look up one level from the current child director (parent)

Unable to Progress (9/27) - Unzip Task not Creating Files Referenced in Subsequent Tasks

```
[2023-09-28, 00:00:25 UTC] {subprocess.py:74} INFO - Running command:
['bash', '-c', 'sudo tar -xvzf
/home/project/***/dags/finalassignment/tolldata.tgz']
[2023-09-28, 00:00:25 UTC] {subprocess.py:85} INFO - Output:
[2023-09-28, 00:00:25 UTC] {subprocess.py:92} INFO - sudo: effective uid is
not 0, is /usr/bin/sudo on a file system with the 'nosuid' option set or an
NFS file system without root privileges?
[2023-09-28, 00:00:25 UTC] {subprocess.py:96} INFO - Command exited with
return code 1
```