Upskilling Track 1 : Basics

An upskilling path for basic skills for Data Engineer

A data engineer needs to have some basic skills, this document mentions all necessary skills required which act as base irrespective of nature of data engineering project.
It covers,

- SQL
- Linux Shell
- Python
- Git

SQL

SQL Language

Take understanding of following aspects of SQL (through various sources available over internet and from books available online),

- Data Definition Language
    - CREATE of database / schema
    - CREATE of table
    - INSERT data into table
    - Update a specific record in table
    - UPSERT into table
- Indexes on table,
    - Understand the purpose of index. (Why do we create table indexes?)
    - Understand constraints for an index (Eg. Query on index prefix makes use of index, query on non prefix fields do not make use of index ; Index can not have function call in it etc.)
    - How to create index?
    - Update / delete index
- Data fetch basics,
    - SELECT clause
    - WHERE clause
    - Subquery in FROM clause
    - IN clause used within WHERE
    - Subquery in IN clause
- Time related operations,
    - What is EPOCH timestamp?
    - How to convert a string date and time into EPOCH timestamp?
    - How to perform date add, sub, diff operations?
    - How to extract day, month, year, day of week from date?

- Aggregation
    - What is group by?
    - What are different aggregation operations?
    - What is purpose of 'having' clause?
- Joins
    - What is Inner Join?
    - What is Left / Right Outer Join?
    - What is Full outer join?
    - In case of join data explodes (count of records increase) if joining key is not unique in both of tables, why does that happen? What should be done in order to avoid that?
- Functions,
    - Date Functions (these are most widely used in practical use)
    - How to handle NULL values with COALESCE?
    - What is GROUP_CONCAT (or its equivalent in mysql)?

SQL learning resources

Refer to this internal document for further more details : SQL | Training Outline Document 2

Few links referred from above

- https://www.geeksforgeeks.org/sql-tutorial/
- https://www.tutorialspoint.com/sql/index.htm
- https://www.w3schools.com/sql/

SQL Assignment

Attached below are links to 2 files. Please download these files in your local file system.

- consumerInternetFile
- startUpFile

Check these two datasets and write SQL queries to answer the following questions.

- How many startups are there in Pune City?
- How many startups in Pune got their Seed/ Angel Funding?
- What is the total amount raised by startups in Pune City? Hint - use regex_replace to get rid of null
- What are the top 5 Industry_Vertical which has the highest number of startups in India?
- Find the top Investor(by amount) of each year.
- Bonus:
    - Find the top startup(by amount raised) from each city?
    - Which SubVertical had the highest growth(in number of startups) over the years?
    - Which SubVertical had the highest growth(in funding) over the years?

Linux Shell

- Linux shell script,
    - Go through all assignments mentioned here : https://medium.com/@sankad_19852/shell-scripting-exercises-5eb7220c2252
    - Above link contains shell scripting basics, go through each example above, understand shell script mentioned as solution and try it out.
    - You should get confidence on know how of shell scripting after going through above
- Important shell commands to understand in details,
    - Checking data size in a directory / file
        - du -sh <dir path>
    - Checking line count of a file
        - cat <filepath> | wc -l
    - Checking total disk space of a linux server
        - df -h
    - Checking total memory of a linux machine
        - cat /proc/meminfo
    - Checking total count of CPUs available on a linux server
        - cat /proc/cpuinfo
    - Check content of log file as it is getting appended
        - tail -f <log file path>
    - Find a file with given pattern in a directory
        - find <directory path> -iname <file pattern>
        - Eg. find . -iname *.java
    - AWK programming,
        - Used for delimiter separated files (CSV / TSV etc.)
        - How to get count of columns per line through AWK
        - How to print all values of only one column using AWK
    - Sort command
    - Uniq command
    - How to check which processes are running on given linux server
        - ps - aef
    - How to search for a value from output of a command
        - grep
        - Eg. ps -aef | grep -i kafka
        - [Above command is to search instances of kafka process running on linux server]
    - How to check if port of a remote machine is open or not
        - telnet <remote machine ip> <remote machine port>

        - Eg. telnet 10.1.101.23 8080
    - How to check whether a remote machine is reachable or not
        - ping <remote machine ip>
    - How to connect to a remote machine?
        - ssh command
    - How to copy data from current linux machine to other remote linux machine?
        - scp command
    -

- Linux terminal game (cover 15 steps from this)
    - Below is the link for a terminal based CTF game. Please read the instructions on how to play the game.
        - Bandit
        - If you're stuck on some level feel free to contact your mentors and discuss the solution.

Python

Install a Python IDE (Anyone will do)

- Visual Studio Code
- PyCharm

Note: If you're used to some other IDE that's also fine just make sure you have all the best practices set as per the next section.

Go through python best practices.

- https://data-flair.training/blogs/python-best-practices/
- pep8 is python standard: https://www.python.org/dev/peps/pep-0008/
- Any editor you choose will have a pep8 checker search on how to enable that for your editor. Eg: for visual studio: https://code.visualstudio.com/docs/python/linting#:~:text=Enable%20linters%23,name%20of%20the%20chosen%20linter.

Post going through best practices, perform the following assignment with best practices followed.

Assignment

Implement a file upload utility with the functionality expected below,

- Prerequisites,

- Install MySQL
    - Get an understanding of how to handle JSON in python.
    - Get an understanding of Pandas in python.
    - Get an understanding of MySQL connection in python.
    - Use 'argparse' for argument parsing.

- Name of utility : file_upload.py
- What is it supposed to do?
    - Whenever the above utility is executed with necessary arguments it will take a directory as an argument, and read all files from the directory, and upload them to the corresponding MySQL table specified in config.
- Arguments,
    - --source_dir <Source directory from which we need to read all files to upload to mysql>
    - --mysql_details <Path of a JSON file which contains all details of MySQL to which to connect to>
        - JSON file contents,
            - {
                - "mysql_ip": "127.0.0.1",
                - "port" : "3306",
                - "username" : "<appropriate username>",
                - "password" : "<password",
                - < any other field required for connection >
            - }
    - --destination_table <Name of table to upload this data to>
- Processing,
    - Create MySQL connection
        - Read JSON file and create MySQL conn object.
    - List files from <source_dir>
    - Iterate through these files one by one.
    - Read each csv file into pandas dataframe.
    - Load pandas dataframe to mysql.

GIT

Understanding git basics

Git Pro Book : https://git-scm.com/book/en/v2/

Go through first 3 chapters of this book,

- Chapter 2 : Gives details about commit and undo on local branch
- Chapter 3 : Gives details of working on branches

Basic GIT steps required to be performed frequently

Initialize (any project you want to clone from dataeaze repo)

1. Go to https://bitbucket.org, create your bitbucket account. Remember this account is to be dedicated for dataeaze.
2. Ask administrator to add you as a member of dataeaze team on bitbucket.
3. On your machine,
    a. Install git

Fork a repository

Prerequisite : Administrator of dataeaze bitbucket account should have added you in dataeaze team.
Go to your bitbucket account. You will see repositories owned by dataeaze.

1. Fork a repository of your interest. How to do that is described as,
    a. Click on repository link
    b. Click on '+' sign left hand side panel there is one 'ACTIONS' item named 'Fork'
    c. Click on Fork. It will take you through a form
    d. Click on 'Fork Repository'

Clone a repository to local machine

1. Create a local directory in home as : mkdir ~/dataeaze/
2. Goto ~/dataeaze/ as : cd ~/dataeaze/
3. Clone forked repository,
    a. Go to your bitbucket account home.
    b. You will see some repositories listed there with your username as prefix.
    c. Select repository which you just forked. This will be the one with your username as prefix of repository
    d. On right hand side top corner you will see a link as 'HTTPS'. Copy that link.
    e. On linux terminal in directory ~/dataeaze/ execute command : git clone <paste previously copied url>
    f. It will ask for your password : Enter your password and its done. It will clone complete repository from source location to your local machine.
4. Add new remote of upstream
    a. Go to 'dataeaze' repo of same project
    b. Copy URL of same
    c. Add new remote in local as,
        c.i. Git remote add upstream <dataeaze repo url>

Create a new branch for any task

1. Bring your local develop branch in sync with dataeaze develop by pull
    a. git checkout develop
    b. git pull upstream develop

2. Do not work on ‘develop’ branch. Always fork a separate branch to work.
3. Fork a new branch as: git checkout -b <branchName>
4. Create periodic meaningful commits. Have meaningful properly formatted commit messages in each.

Creating pull request

1. Add remote as upstream (This is one time step, not required to do it everytime). How to add remote is described below,
    a. Go to your bitbucket account. Click on ‘all’ repositories.
    b. Click on repository which you want to raise pull request against, but this time click on repository owned by dataeaze (ie. repository with url starting with dataeaze as prefix).
    c. Copy url of this repository which you will find on right hand side top corner.
    d. You need to add this repository as remote by command : git remote add upstream <paste url here>
    e. Verify this by executing command : git remote -v
    f. Bring your develop branch to latest updated state,
    g. git checkout develop
    h. git pull upstream develop (This way changes pushed by other developers come to your local repository)
2. Push branch on which you were working to your origin
    a. git checkout <yourBranch>
    b. git merge develop (This is necessary to bring your work from your branch on top of latest develop)
    c. git push origin <yourBranch> (This way your local branch is pushed to your remote repository hosted at bitbucket. Ideally you should do it daily, this preserves your code even at loss of your local machine)
3. Create pull request
    a. Go to your bitbucket account.
    b. Click on ‘Create pull request’
    c. Create a pull request between ‘yourusername’->’yourbranch’ and ‘dataeaze’->’develop’
4. Done

Address review comments

1. Switch to your branch corresponding to this pull request
2. Go through pull request on bitbucket.
3. Look for each comment, if any code change is required then do accordingly.
4. Create one commit with message as : Addressed review comments of PR #<pull request number>
5. Push your branch again to origin as : git push origin <your branch name>

Important Git Commands

How to check how many branches are there?
====
git branch -v

How to check our current branch
====
<branch marked with * is current branch>

How to check how many remotes are there?
====
git remote -v

How to add new remote?
====
git remote add <remote_name> <url>
Eg.
git remote add upstream <url>

How to fetch a remote branch to local
====
git fetch <remote_name> <branch_name>:<local_branch_name>
Eg.
git fetch origin aws_orc_optimizations:aws_orc_optimizations

How to switch branch?
====
git checkout <branch name to switch to>
Eg.
git checkout aws_orc_optimizations

How to fork a new branch from existing current branch?
====
git checkout -b <new_branch_name>

How to bring your local branch in sync with the remote branch? (ie. to pull all changes which have happened on remote to local branch)?
====

step 1 > Checkout to branch which we want to bring in sync
git checkout <branch name>
Eg. git checkout aws_orc_optimizations

step 2 > Pull branch from remote
git pull <remote name> <branch name>
Eg.
git pull origin aws_orc_optimizations

How to commit?
====
To check what is current status of code modifications
----
git status

What is format of output of git status?

There are three sections displayed :
- changes staged for commit : changes ready for commit, these are files on which git add is done.
- changes not staged for commit : These are modified files on which 'git add' is not yet done
- untracked files : These are new files in git directory, these are not yet tracked. If you want to track then need to perform 'git add'

To add a untracked / modified file
----
git add <file_path>

To remove an already added file
----
git rm <file_path>

How to check what have we changed in files to be committed?
====
git diff <file path>
Eg.

How to push branch to remote
====
git push origin <branch name>
git push origin feature/dtze2

How to check diff of current branch HEAD with 1 commit before?
====
git diff HEAD~1

How to change username and email in git commit
====
git config --global user.name "Your Username"
git config --global user.email "yourEmail@dataeaze.io"

Advanced : Changing previous commits

[These are advanced features, are required in order to fix]

Make sure to take backup by creating a backup branch before moving ahead with any of advanced fixes

How to reset current branch to some previous commit?

This delete all code changes after that commit

git reset --hard <commit id>

How to reset the latest commit?

git reset HEAD~1

This brings out all code changes from given commit to 'staging' area

How to do controlled changes in : commit message / commit author?

With use of 'rebase' command

1. Identify how many commits do you want to go back
    a. You can check previous commits as : git log
    b. You can check online log of all previous commits as : git log --oneline
2. Execute git rebase command for those many commits,
    a. git rebase HEAD~<count of commits to go back>
        a.i. Eg. git rebase HEAD~5
3. Once git rebase is done, it is required to specify what actions are to be performed on every commit
    a. There are commands like : pick / exec / delete etc.
    b. More details are mentioned here : https://git-scm.com/docs/git-rebase

Assignment

- **Create repo, develop feature, push and raise PR**
    - Create your own project on bitbucket
    - Clone it to local
    - Add your own repo as upstream
    - Make sure that project has two branches on bitbucket : master and develop (create and push those)
    - Create a feature/<featureName> branch
    - Push this feature/<featureName> branch to bitbucket
    - Create pull request against develop branch
    - **Evaluate,**
        - **Check point 1 to be reviewed from reviewer : PR created and assign it for review to reviewer**
- **How to change previous commits**
    - Create 4 commits in current branch with few changes per commit
    - Push these commits to current branch on remote
    - Previous change 1,

        - Create a new branch from current branch : feature/change_branch1
        - With git rebase command do changes in previous commits, change commit name and git message
    - Previous change 2
        - Create a new branch from current branch : feature/change_branch2
        - With git rebase : Merge any two of previous commits into a single commit.
    - **Evaluate,**
        - **Push all branches to remote repository**
        - **Get all three branches reviewed from reviewer**