**Interview Test**

**Requirement**

You will require a github.com account.

**Task 1**

**Please feel free to use any language for this task.**

It is relatively common to work with large datasets within the QA team that are often too large to easily manipulate by hand using tools such as Excel. This data often needs to be crosschecked against remote systems to ensure that data processing is being handled correctly.

Another common aspect of our work involves interrogating web APIs for information. The problem laid out below combines both of these aspects of our day-to-day work.

We have a large csv file (2017.csv) that lists dogs licensed in the US during 2017. The data is split into the following 7 sections:

LicenseType, Breed, Color, DogName, OwnerZip, ExpYear, ValidDate

1. Read the .csv file and extract the Breeds provided. Normalize the breed names by removing all whitespacing and making them all lowercase. Lastly create a list of unique breeds without duplicates.

Here I’m considering list\_dogs as table using sql function to get the data for the following questions

Solution: select distinct (lower(trim(replace(breed,’ ‘,’’))))as breed from list\_dogs

Explaination: for above query I’m using inside the parenthesis replace the white spaces with breed field and remove it with those spaces using trim function and convert the breed name into lower cases using lower function in uniquely identifies using the distinct function.

1. Create a list of number of licenses by LicenseType of each unique breed.

Solution:

select lower(trim(replace(breed,’ ‘,’’)))as breed,licensetype, count(\*) from list\_dogs

group by lower(trim(replace(breed,’ ‘,’’)))as breed,licensetype

Explanation: Here in this query I’m taking with the same breed logic along with count of licensetype of each unique breed

1. Find out the top 5 popular name of dogs and create a list of these names along with count of dogs having these names.

Solution:

Select dogname,count(\*) as cnt from list\_dogs

group by dogname order by count(\*) desc limit 5

Explanation: Here by using count function to count the names of dog which is top most 5 popular using limit function.

**Bonus**

Create a method which takes date range as input and return the details of licences issues during that date.

**Notes**

Please write proper comments explaining each method

**Task 2**

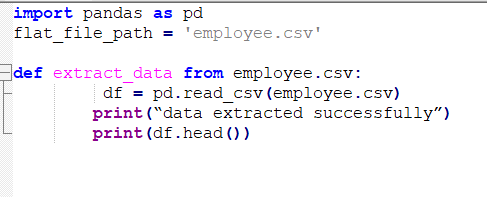
**Enhanced ETL Script (Python with Pandas and SQLAlchemy)**

Assume the flat file has columns id, name, date\_of\_birth, salary, and department\_id. The new database has two tables: employees and departments. The departments table should be populated with unique department IDs and names.

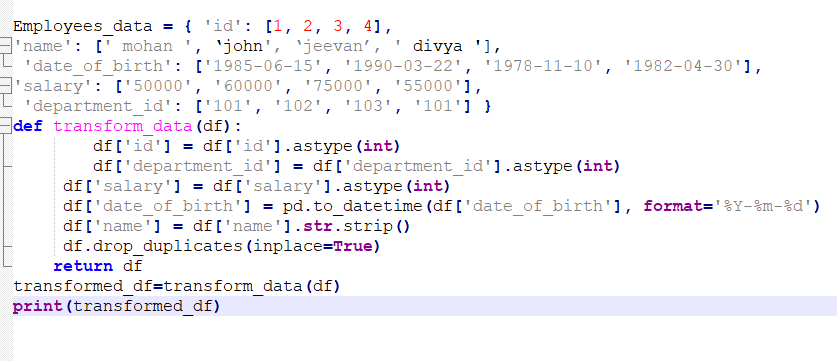
Write a Python script to:

1. Extract data from the flat file.

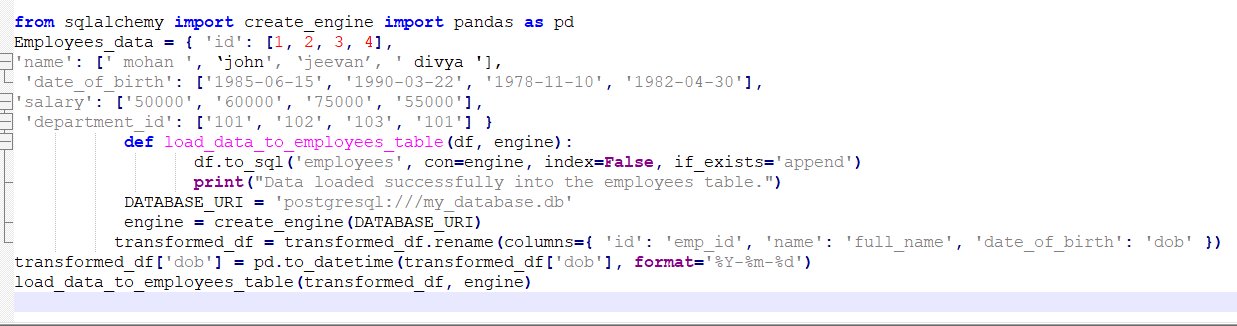
Solution:



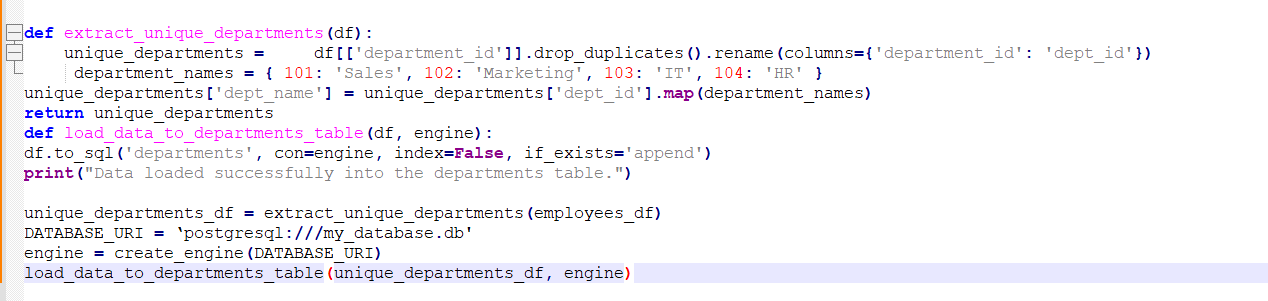
1. Transform the data, including data type conversions and trimming spaces.



1. Load the data into the employee’s table.



1. Populate the departments table with unique department IDs and names.



**SQL Validations**

**Task 3**

**From Flat file**

Assume a flat file has columns id, name, date\_of\_birth, salary, and department\_id, and the new database has two tables: employees with columns emp\_id, full\_name, dob, salary, and department\_id, and departments with columns dept\_id and dept\_name.

Write a SQL query to validate that the data loaded into the new database matches the data from the flat files, including a join with the departments table to ensure department names are correctly associated.

WITH flat\_file\_data AS (

SELECT \*

FROM EXTERNAL (

'path/to/desktop/employee\_data.csv'

FIELDS TERMINATED BY ','

OPTIONALLY ENCLOSED BY '"'

(id, name, date\_of\_birth, salary, department\_id)

)

)

SELECT

f.id AS flat\_file\_id,

f.name AS flat\_file\_name,

f.date\_of\_birth AS flat\_file\_dob,

f.salary AS flat\_file\_salary,

f.department\_id AS flat\_file\_department\_id,

d.dept\_name AS flat\_file\_dept\_name,

e.emp\_id AS database\_emp\_id,

e.full\_name AS database\_full\_name,

e.dob AS database\_dob,

e.salary AS database\_salary,

e.department\_id AS database\_department\_id,

dep.dept\_name AS database\_dept\_name

FROM

flat\_file\_data f

LEFT JOIN

employees e ON f.id = e.emp\_id

LEFT JOIN

departments dep ON e.department\_id = dep.dept\_id

LEFT JOIN

departments d ON f.department\_id = d.dept\_id

WHERE

(f.name != e.full\_name OR

f.date\_of\_birth != e.dob OR

f.salary != e.salary OR

f.department\_id != e.department\_id OR

d.dept\_name != dep.dept\_name);

**Task 4**

You have two tables in your database:

1. employees:
   * emp\_id (Primary Key)
   * full\_name
   * dob
   * salary
   * department\_id (Foreign Key)
2. departments:
   * dept\_id (Primary Key)
   * dept\_name
3. projects:
   * project\_id (Primary Key)
   * project\_name
   * department\_id (Foreign Key)
4. employee\_projects:
   * emp\_id (Foreign Key)
   * project\_id (Foreign Key)

**Queries:**

1. **Find Departments with Average Salary Greater than a Certain Amount**
   * **Objective:** Identify departments where the average salary of employees is greater than $50,000.

Solution:- WITH cte as(

SELECT department\_id, avg(sal)FROM employees GROUP BY department\_id HAVING avg(sal)>50000)

SELECT cte.department\_id,d.dname FROM cte INNER JOIN departments d

ON cte. department\_id =d. department\_id

1. **List Employees with Salaries Above the Department Average and Working on More Than One Project**
   * **Objective**: List employees whose salary is above the average salary of their respective departments and who are working on more than one project.

with cte\_1 as(

select e.\*,p.\*,row\_number()over(partition by p.department\_id order by project\_id asc)as rn from employees e inner join projects p

on e.department\_id=p.department\_id),

cte\_2 as(

SELECT department\_id,avg(salary)AS avg\_salary FROM cte\_1 GROUP BY department\_id)

select cte\_1.full\_name,cte\_1.salary,cte\_2.avg\_salary

from cte\_1 inner join cte\_2

on cte\_1.department\_id=cte\_2.department\_id

where cte\_1.rn>1 and cte\_1.salary>cte\_2.avg\_salary

1. **Find Employees with the Highest Salary in Each Department and List Their Projects**
   * Objective: List the employees who have the highest salary in each department and the projects they are working on.

with cte\_1 as(

select e.\*,dense\_rank()over(partition by e.department\_id order by e.salary desc)as rnk from employees e)

select p.project\_id,p.project\_name,cte\_1.full\_name,cte\_1.rnk as high\_salary,cte\_1.deptno

from projects p inner join cte\_1

on cte\_1.department\_id=p.department\_id

where cte\_1.rnk=1

**Task 5**

**Data Quality and Performance Testing:**

* + Describe how you would perform data quality checks to ensure that the migrated data is accurate and complete.

Solutions:-

1.Source and Target record count check:-Using sql queries apply the transformation logic to verify the count in source and target.

2.Duplicate check in Target:- To verify the duplicate records by using primary key column in the target table.

3.Null data check in the target:- To verify the null data in the target using primary key column

4.Full data validation:-To verify the data comparision between source and target using minus query in sql

5.DDL Checks:- To verify the data type,data length in a target

6.Error Reporting and Issue Resolution:- Log any discrepancies found during validation, detailing the exact row or field where the issue occurred

* + Outline a strategy for performance testing of the ETL process. Include considerations for handling large volumes of data and ensuring the ETL process completes within acceptable time frames.

Solution:-

1. **Define Performance Objectives:- Mainly focus on the kpi includes**

**ETL completion time: How long does the etl process take for a given volume of data?**

**Resource Utilization: how efficiently does the etl process use CPU,memory,disk I/O and network bandwidth?**

**Scalability: how well does the ETL process handle increasing volumes of data?**

**Data throughput: Number of records processed per second.**

**Concurrency:** The ability to handle multiple ETL jobs running in parallel without performance degradation

2. **Create a Test Environment:** Create a test environment that closely mirrors the production environment in terms of hardware, software, network configurations, and data characteristics. Replicate the ETL tools, servers, and data pipelines as they exist in production.

3. **Data Volume Considerations: data volume handling based on the data size like small,medium,large for validating the etl process and functionality of the process.**

**4. Test Scenarios:** Test with different volumes of data and observe the impact on extraction time. Test different extraction formats (CSV, JSON, XML, etc.) to evaluate which is optimal for performance.

1. **Monitor System Performance: CPU, Memory, Disk I/O, and Network usage**: Observe how the ETL process impacts system resources and identify potential bottlenecks

1. **Stress Testing and Scalability:** Simulate a peak load by running the ETL process with a data volume that exceeds the expected production workload. Test running multiple ETL jobs concurrently and observe whether they compete for resources, causing performance degradation
2. **Optimization Techniques:** Partition large tables by date or some other meaningful key to improve performance for large data loads and queries.Test whether using incremental loading (only loading new or changed data) improves performance compared to full data loads.
3. **End-to-End ETL Testing:**The ETL process completes within the acceptable time frame, even under peak load.There are no errors or failures during the process, and data integrity is maintained.
4. **Validation of Data Quality and Integrity :** Ensure that the number of rows in the source system matches the number of rows in the target system after the ETL process.

Check that all business rules have been correctly applied during the ETL process.

10. **Performance Tuning** : Optimize SQL queries used in the extraction and transformation phases.Tune memory and CPU settings for the ETL tool.Implement caching for frequently accessed lookup data.Optimize the use of indexes in the target database.