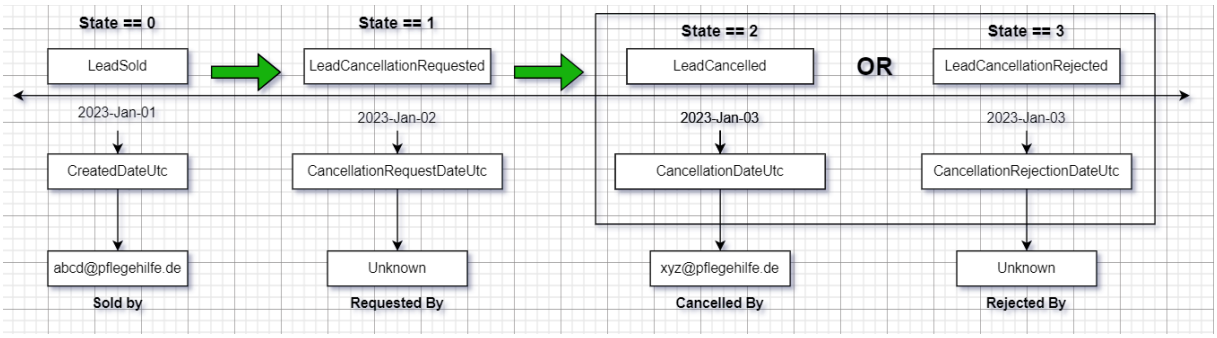
Data Engineer Task

1.0 Introduction to Lead Model:

To complete the task, it is very important to understand that what is a lead and its life cycle. A lead is a contact detail of a potential client sent to companies, clients, who are looking for a company which can provide care services or products such as Wheelchair, emergency button or 24-Hour care. The life cycle of lead has 4 stages.



Right now, we have 1 row for every lead, which has multiple columns for keeping the record of when did the state changed for the lead. We want to create a new table, which will have 1 row for every event happened to the lead in its life cycle. For example, if a lead is cancelled then it means total 3 events happened in the life of a lead. 1st event was the lead sold, 2nd it got requested for cancellation and 3rd it got cancelled. So, we want to insert 3 individual rows in a table which will be named as **LeadEvents** in Snowflake. You will get more details in **section 4.0**.

2.0 Introduction to Task:

Before calculating the events of the lead, you are required to create a data pipeline using the Azure Data Factory for the provided data in excel. There are 100 leads in the provided excel. Use SQL as the source and Snowflake as the destination, more details are available in **section 3.0.** After the data is moved using the pipeline. You need to create a python project where you will be reading the data from the Snowflake and apply the described transformations in **section 4.0**. After the transformation, you will have to save the transformed data into another table in Snowflake as mentioned above in **section 1.0**.

3.0 Extract & Load:

The data provided in excel, will be imported into a SQL table. You will have to provide the SQL scripts along with your submission that will help us create the same source database at our end while evaluation. Your script should create the following:

1. Database.
2. Table.
3. Query to insert excel data into SQL table.

After importing the data in SQL, create an Azure Data Factory pipeline to move the data from SQL to Snowflake. The pipeline should follow the below points:

1. The pipeline should be incremental.
2. Must be connected with GitHub, as in the end we will evaluate from your GitHub repo.
3. The pipelines should be easily extensible and maintainable.
4. The pipelines should send email on failure on any activity within the pipeline.
5. The pipelines should use linked services.
6. A trigger after 30 minutes should be attached with the pipelines.

The data in excel has following columns and below is the definition for each column in the file. The table name would be **CompanyLeads** in your source and destination database.

Column Name	Type	Definition
Id	UUID	It is the unique id for every lead or row in the table.
State	Int	It represents the current state of the lead. It can have values from 0 to 3
CreatedDateUtc	Datetime	It represents the sold date of the lead (State = 0)
CancellationRequestDateUtc	Datetime	It represents the date on which the lead was requested to be cancelled (State = 1).
CancellationDateUtc	Datetime	It represents the date on which the lead was marked as cancelled (State = 2)
CancellationRejectionDateUtc	Datetime	It represents the date on which the lead cancellation was marked as rejected (State = 3)
SoldEmployee	String	Email of the employee who sold the lead
CancelledEmployee	String	Email of the employee who cancelled the lead
UpdatedDateUtc	Datetime	It represents the last datetime on which the row was updated.

Note: You will also provide the scripts for Snowflake which will help to create the same architecture of Snowflake at our end. Your script should create the following at our end:

1. Warehouse instance.
2. Database.
3. Tables.

4.0 Transform:

After the data is moved to the Snowflake, we would like to transform the data as mentioned in the introduction part and for that you need to use Python. Where you will read the data from Snowflake **CompanyLeads** table. And perform the transformation on the whole data. The transformed data will be saved back into another table in Snowflake which will be

named as **LeadEvents.** For the python code, you can create another folder in your same data factory repo root directory. You can name that folder as **BIT**. In this folder you will create your python scripts for the transformations.

The task for the transformation is to calculate the events happened throughout the life cycle for every lead. A lead can have state 0 to 3 and the change in state is sequential means any lead having state 2 or 3 would must have state 0 and 1 as well in past then only it can reach to state 2 or 3.

The transformed table will have the following columns, and the calculation of columns is explained after the below table.

Column Name	Date Type	Definition	Sample Value
Id	UUID	It will be the unique value for every row in the table	DAA18DEB-5565-CA36-6229-08D62AD390B3
EventType	string	It is the type of event that happened to the lead on a specific time.	LeadSold, LeadRequestedCancellation, LeadCancelled, LeadCancellationRejected
EventEmployee	String	It is the email of the person who performed that event.	abcd@pflegehilfe.de
EventDate	DateTime	It is the datetime of the event when it happened	2018-10-08 09:23:48.1996100
LeadId	UUID	It is the unique ID of every lead moved from SQL	DAA18DEB-5565-CA36-6229-08D62AD390B3
UpdatedDateUtc	Datetime	It is the available datetime in UpdateDateUtc column in CompanyLeads table for the lead when your code was calculating the events for the lead	2018-10-08 09:23:48.1996100

4.1 Logics for Columns Calculation

- **EventType**
  - Would be equal to **LeadSold:** If State = 0
  - Would be equal to **LeadRequestedCancellation:** If State = 1
  - Would be equal to **LeadCancelled:** If State = 2
  - Would be equal to **LeadCancellationRejected:** If State = 3
- **Id**
  - It will be a simple GUID, generated using the python code for every event of the lead.
- **EventEmployee**
  - This will be the value of the column in CompanyLeads row, which represents the person who did that event. For example, the CancelledEmployee column represents that the lead was cancelled by this employee.

- **EventDate**
    - This will be the datetime of when the event happened. For example, if the current state of a lead is 0, means it is just sold yet. Then the **EventDate** would be equal to the **CreatedDateUtc** of the lead.

Note: The **LeadCancellationRequested** and **LeadCancellationRejected** events will always have **EventEmployee** as **“Unknown”**.

The python project should follow the following points:

- Good project and folder structure.
- Consistent naming convention
- Good comments
- Good exception handling.
- No secrets should be hard coded in code.
- Follow object-oriented principals.
- Use good design pattern for connection with outside services.
- Clean, Efficient, and simple logics that can be easily adapt change.

Best of Luck!