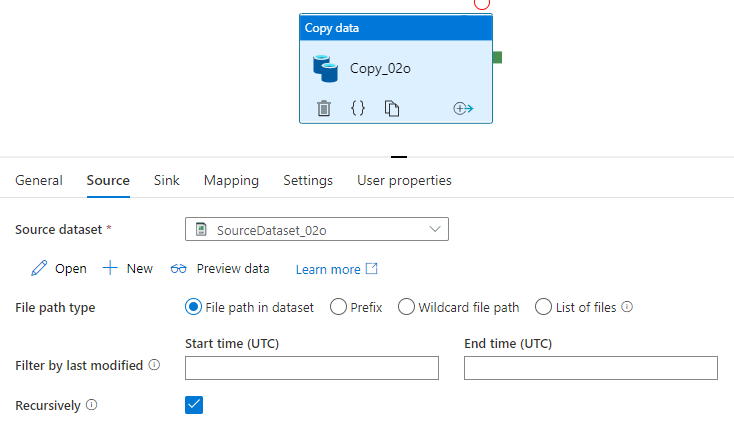
**SOLUTION OF ADF CURRICULUM SHEET:**

Ques:1: Copy data from blob storage to on-prem& Azure SQL Database, considering CSV/TXT(any delimiter) files are available on blob

Sol: Steps to do:

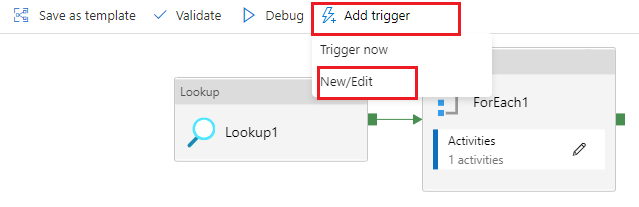
* Take a copy activity, set source dataset as blob and sink dataset as Azure SQL, copy the data from blob to Azure SQL
* Point the source file as .csv or .txt and in the sink side create a table and the data from blob will copy into table.

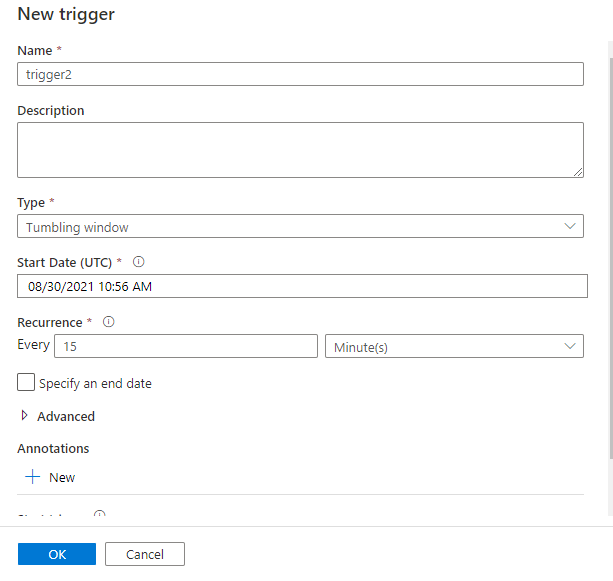


Ques:2: Configure tumbling window trigger on azure data factory over the same pipeline

Sol: Steps to do:

* Click add trigger and then new/edit
* Create a new trigger of type tumbling window, click Ok.

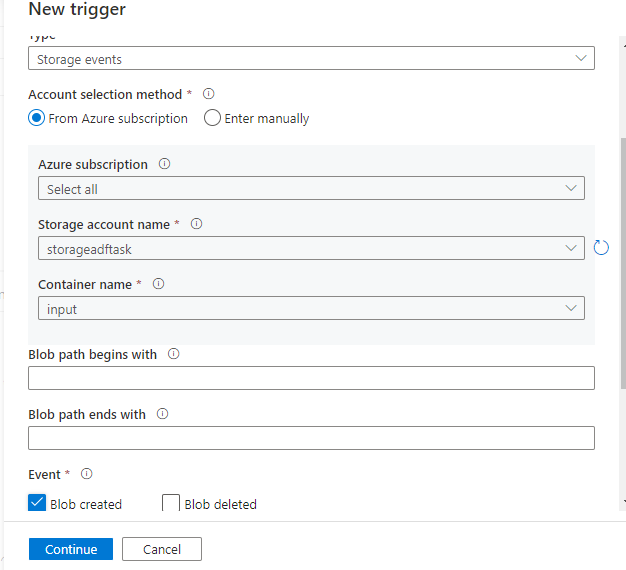




Ques:3: Configure event trigger on a blob storage that will copy newly added files to SQL

Sol: Steps to do:

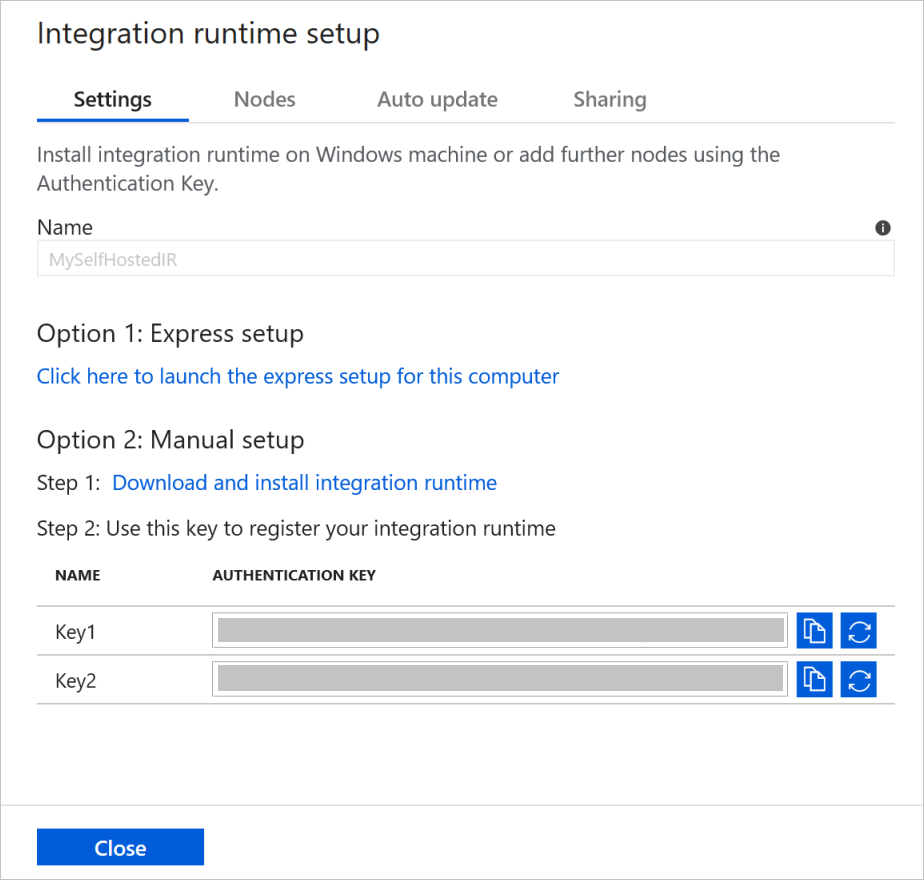
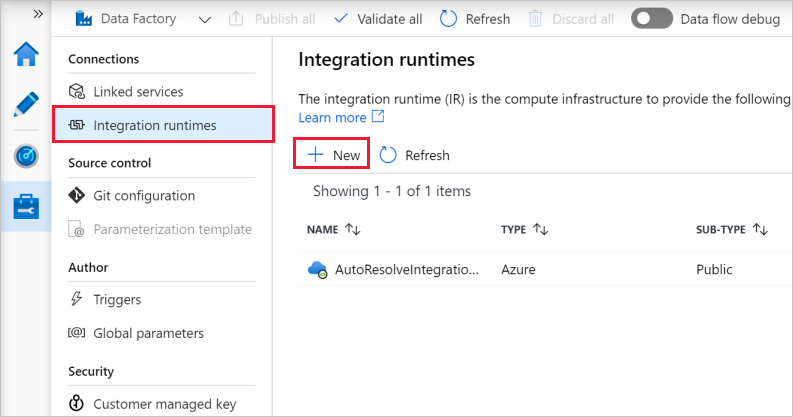
* Select a pipeline which will copy your data from blob and added into SQL, add trigger now/edit.
* Create a new trigger of type storage event, select your subscription id, storage account name, container name
* Select event as blob created and hit continue.

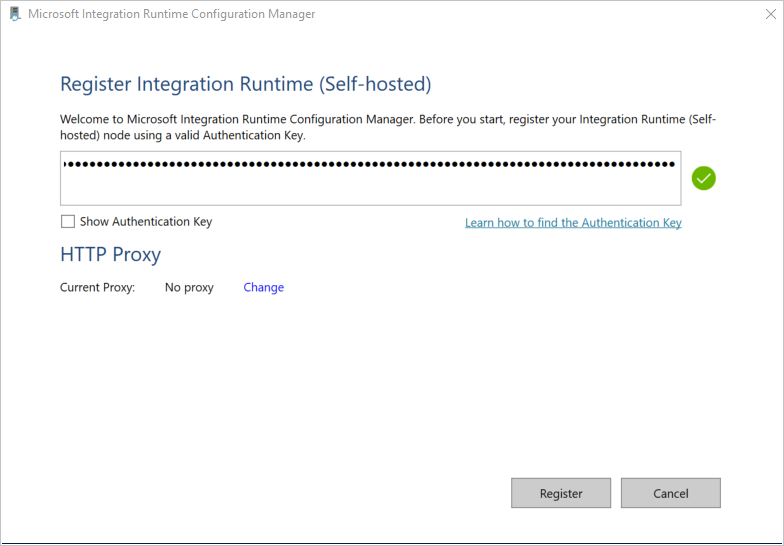


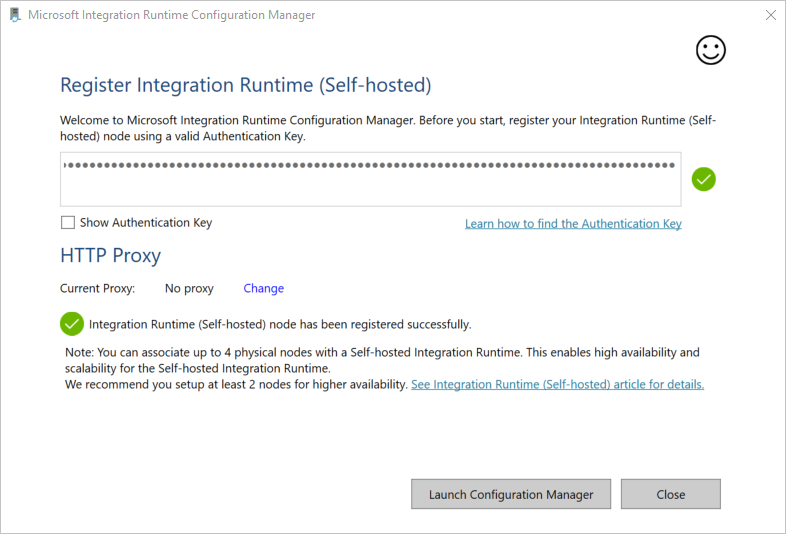
Ques:4: Configure linked self hosted IR.

Sol: Steps to do:

* On the home page of the Azure Data Factory UI, select the [Manage tab](https://docs.microsoft.com/en-us/azure/data-factory/author-management-hub) from the leftmost pane.
* Select **Integration runtimes** on the left pane, and then select **+New**.
* On the Integration runtime setup page, select Azure, Self-Hosted, and then select Continue.
* On the following page, select Self-Hosted to create a Self-Hosted IR, and then select Continue.
* Enter a name for your IR and select Create.
* On the Integration runtime setup page, select the link under Option 1 to open the express setup on your computer. Or follow the steps under Option 2 to set up manually. The following instructions are based on manual setup:
* Copy and paste the authentication key. Select Download and install integration runtime.
* Download the self-hosted integration runtime on a local Windows machine. Run the installer.
* On the Register Integration Runtime (Self-hosted) page, paste the key you saved earlier, and select Register.
* On the New Integration Runtime (Self-hosted) Node page, select Finish.
* After the self-hosted integration runtime is registered successfully, you see the following window:



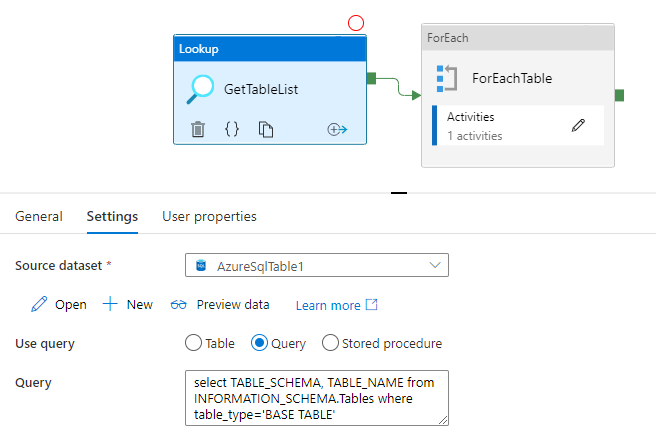


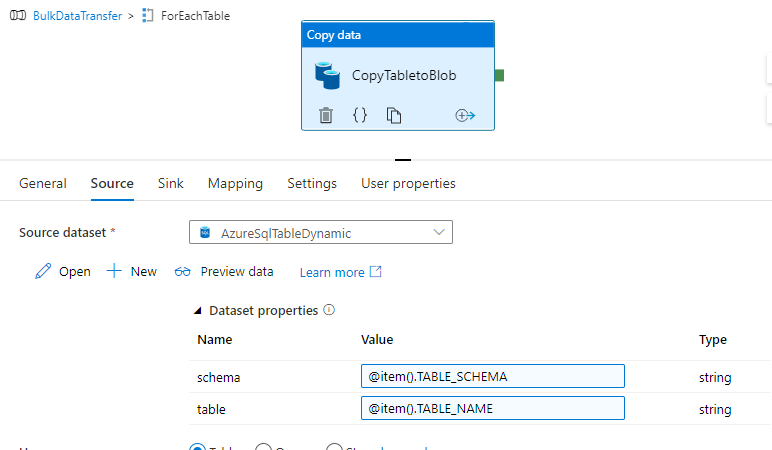


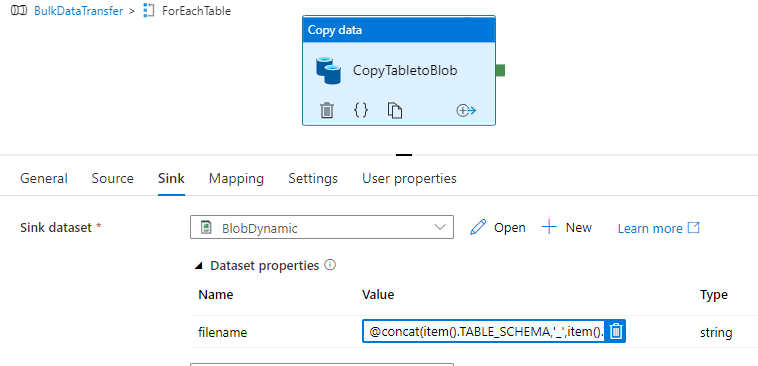
Ques:7: Copy multiple tables in bulk in SQL - Dynamic load for multiple datasets in a single pipeline

Sol: Steps to do:

* Take a lookup activity which will fetch multiple table whose table type is base table, the query- select TABLE\_SCHEMA, TABLE\_NAME from INFORMATION\_SCHEMA.Tables where table\_type='BASE TABLE'
* Attach the lookup with for-each activity and pass the output of lookup in the for-each activity.
* Inside for-each, take a copy activity which will copy from Azure SQL to Blob, pass the parameter for table schema and table name in the source dataset.
* In the sink side, pass the parameter for filename in the sink dataset and pass the query in the parameter- @concat(item().TABLE\_SCHEMA,'\_',item().TABLE\_NAME,'.csv')
* This will return filename as – ( tableSchema\_tableName.csv )



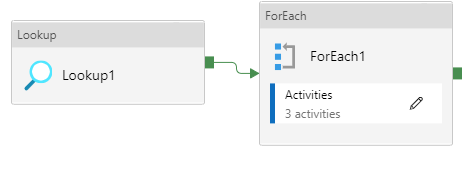


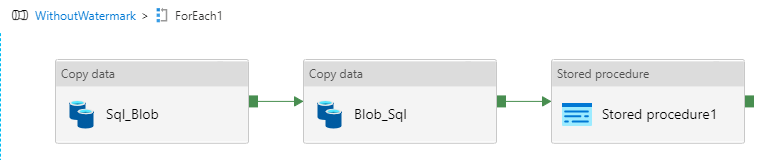


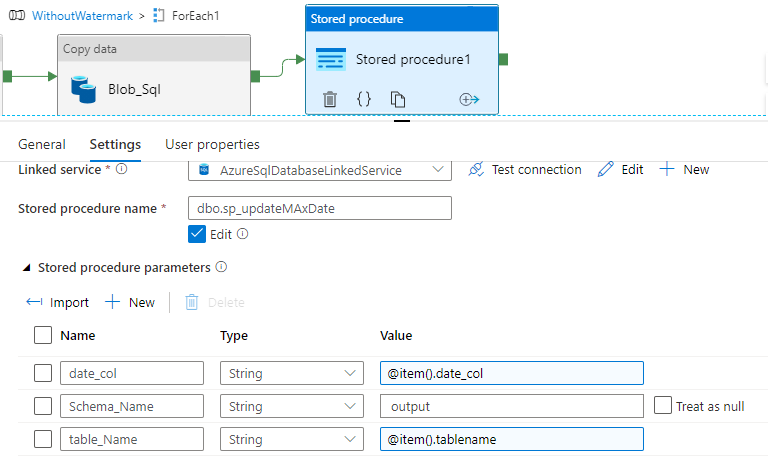
Ques:8: Copy data incrementally from multiple tables from Local SQL Server to Azure SQL Database.

Sol: Steps to do:

* Take a lookup activity which will point to the table contain, detail of table name, filename, sink folder path.
* Attach lookup with for-each and pass the output of lookup in for-each.
* Inside for-each activity, take 2copy activity, one will copy your data and store into blob and act as a stagging table for temporary storage. Another copy activity which will take your data from stagging and then recopy them into your table.
* Take stored procedure for update date in the table.



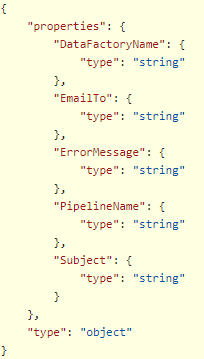
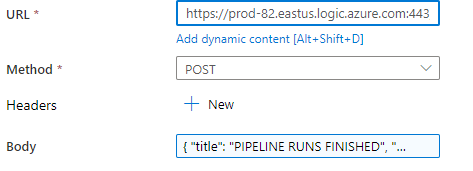


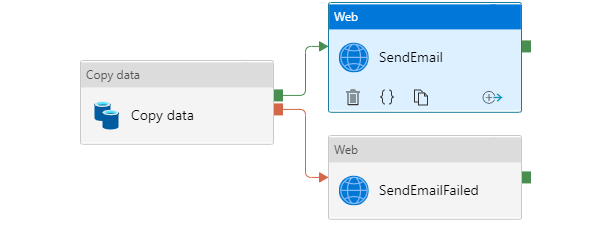


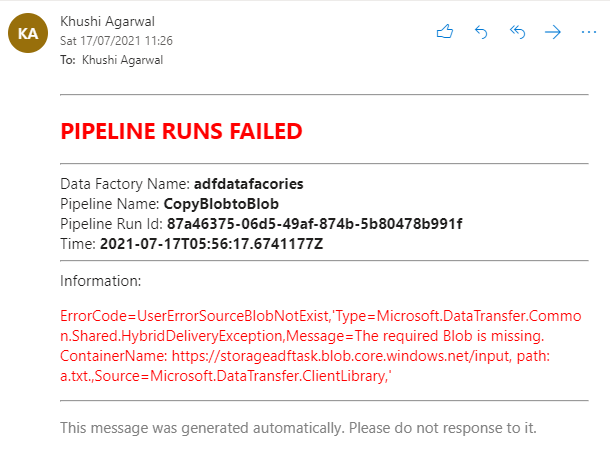
Ques:9: Use Logic App to send a custom response of failure/success of a pipeline to user via email.

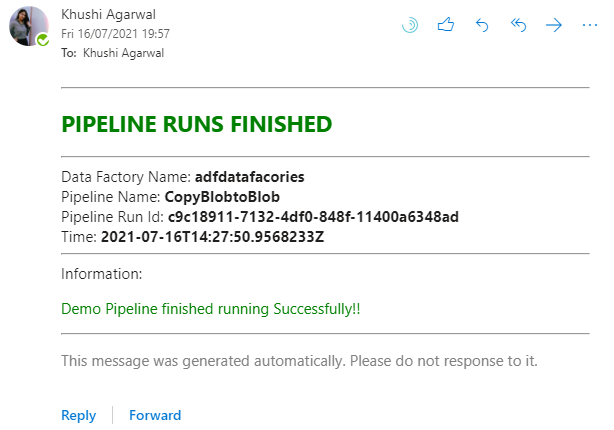
Sol: Steps to do:

* Create a logic app with azure portal.
* In Logic app Design, use “**When a HTTP request is received”** template.
* Create an API with your email account, where you want to receive your mail.
* Now use JSON schema written below and select POST method and save it, it contain a URL at the top of the box Copy that for later.
* Create a copy Pipeline that copy the data from Source to Sink.
* Create 2 Web Pipeline, one for Pipeline Fails and other of Succeed.
* Paste the copy URL and in the body, section write your message what you want to declare in the mail, as shown below.



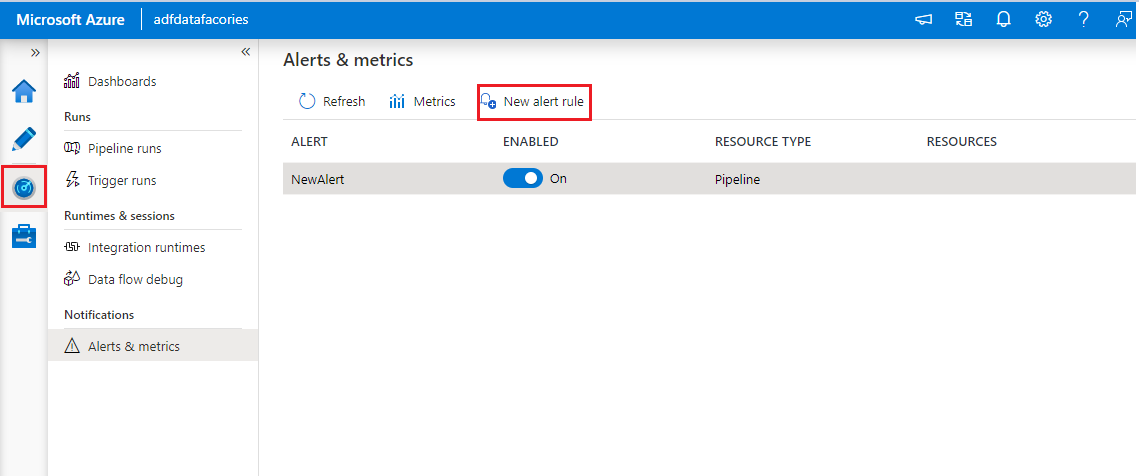


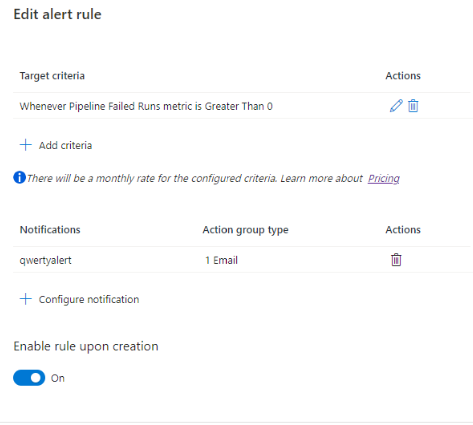


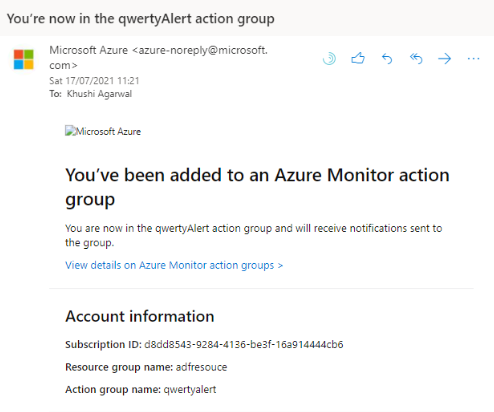
Ques:10: Use ADF Monitor alerts to send an email/messages/push notifications whenever a pipeline/activity fails/succeeds

Sol: Steps to do:

* To get started, simply navigate to the **Monitor** tab in your data factory, select **Alerts & Metrics**, and then select **New Alert Rule**.
* Select the target data factory metric for which you want to be alerted.
* Fill the details and enter the email id, number for SMS so that you receive notification whenever Pipeline response.
* And every time when pipeline response you will get notification via email, or SMS.



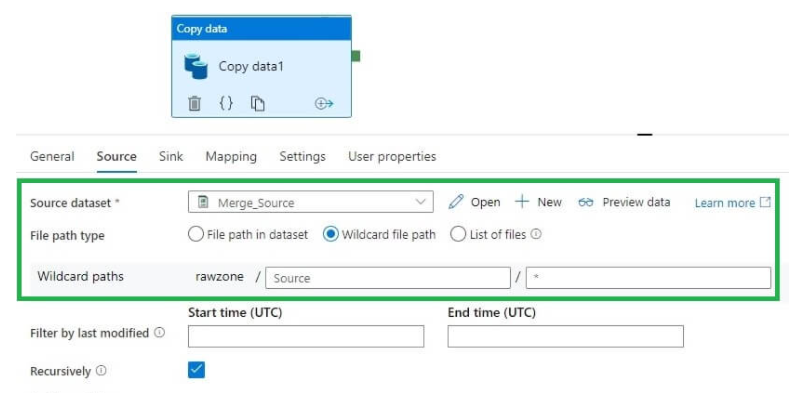


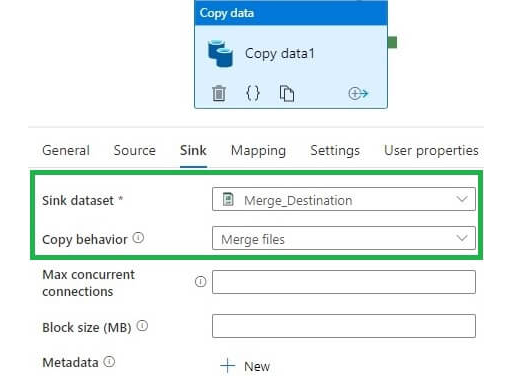


Ques:11: Merge 2 files in blob using azure data factory

Sol: Steps to do:

* Take a copy activity, add source, and sink dataset as a blob
* Select source folder and select file path type as a wildcard file and use asterisk sign (\*) path so that it will copy multiple tables.
* In the sink side, select copy behavior as a merge files.
* The files will be merged and a single file will be created contain, both the files data.

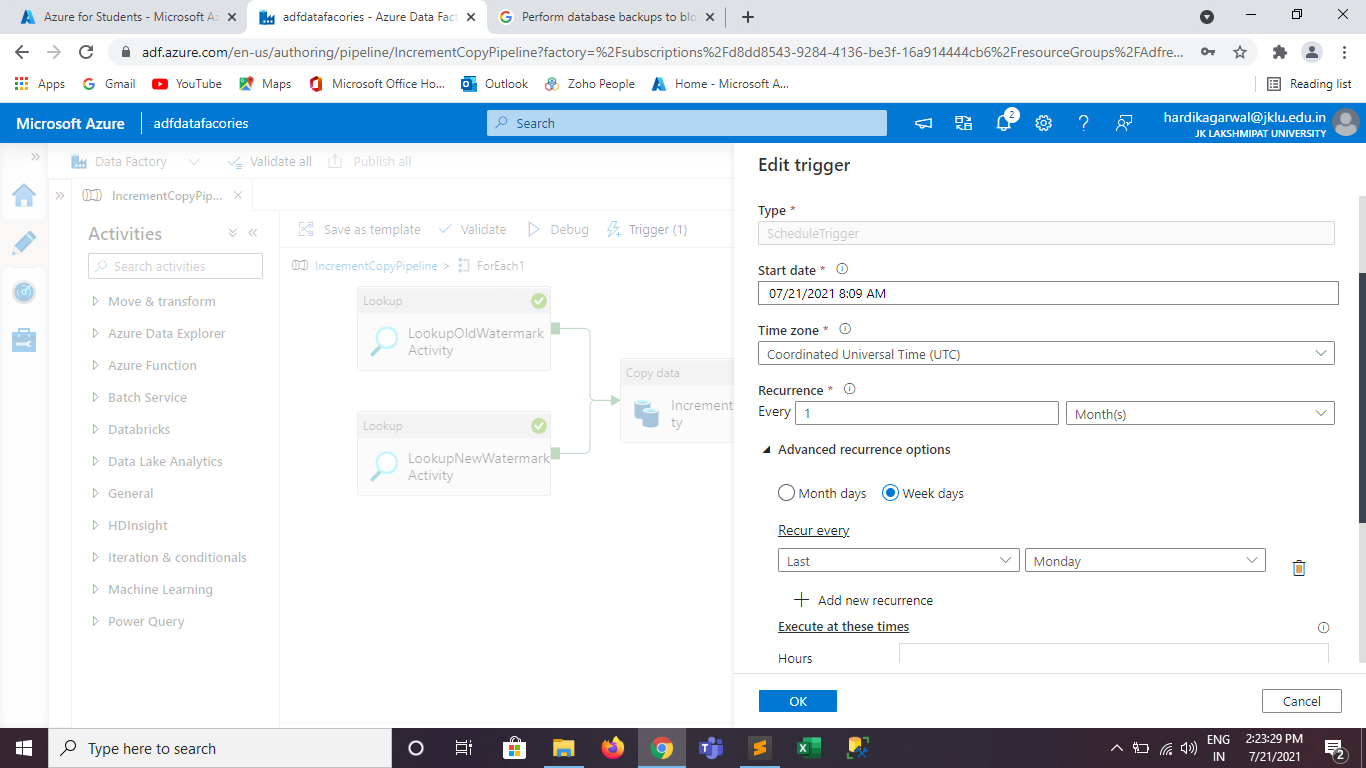




Ques:12: Trigger a stored procedure only on the last day of the month when it is Monday.

Sol: Steps to do:

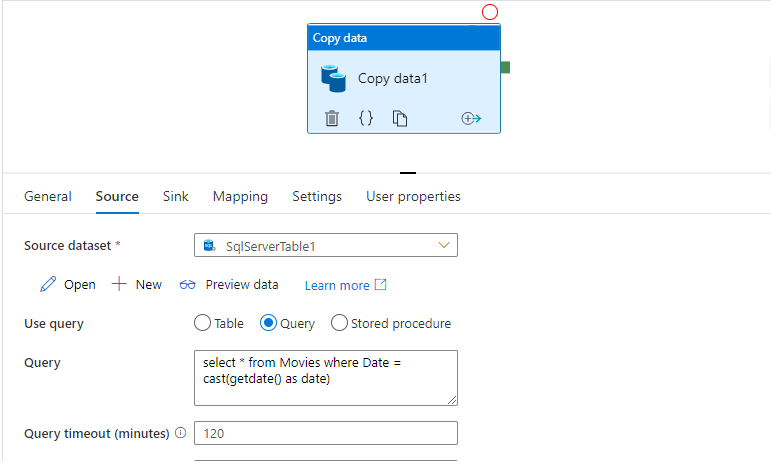
* Select trigger option, shows top of the activities in the pipeline.
* Select add/new trigger
* Select the Schedule trigger, set start date and time and select Recurrence for every 1st of month and set weekly recurrence for every month end Monday. (As shown in picture.)

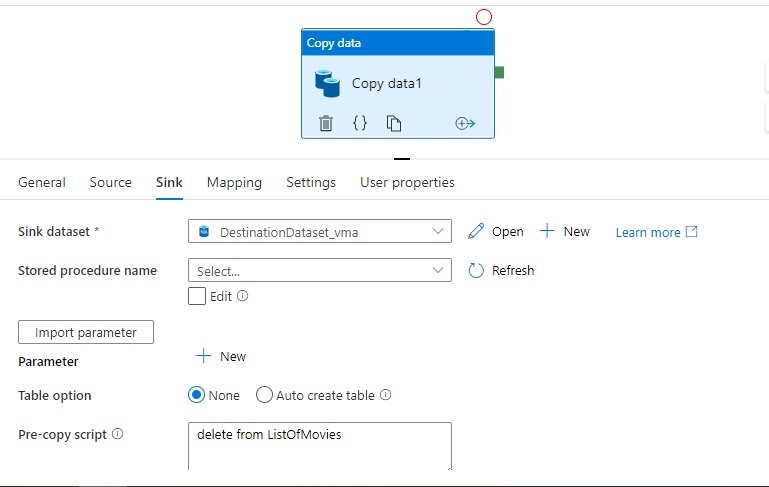


Ques: 14: Incremental load for every Hour. TIP - Create a pipeline and schedule its trigger to every hour which will fetch the data of the current day and in the pre-copy, script remove the data of the current day from the target and then load it.

Sol: Steps to do:

* In this take a copy activity, and in the source connect Dataset with local SQL.
* In the query section, writ a query which will fetch the only current date value.
* In the Sink, connect dataset as Azure SQL and in the pre-copy script, write query for delete the table, so that it avoids duplicity in the table.

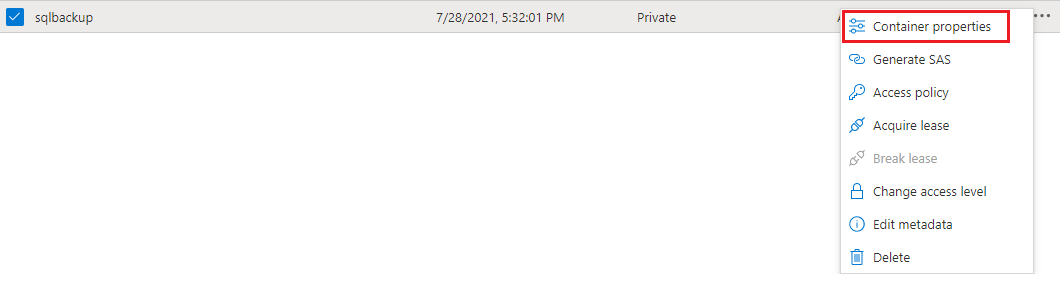


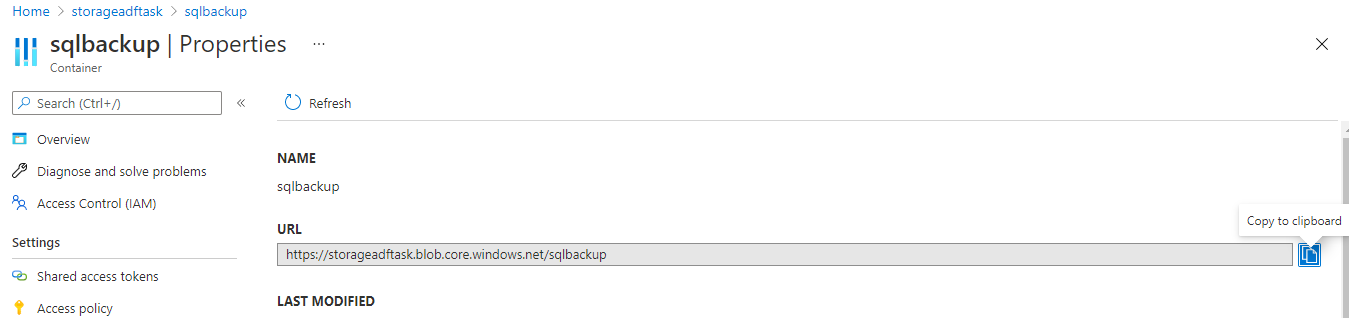


Ques:15: Perform database backups to blob and schedule them on every 1st date of month and name of the backup file should be like “backup\_052020.bak” here “052020” is a current month and current year when the backup is scheduled using ADF

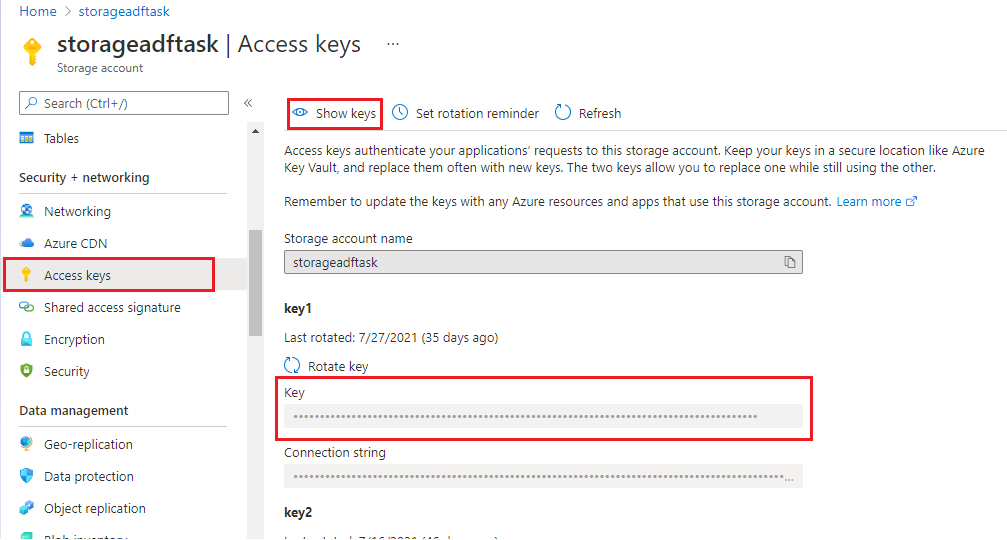
Sol: Steps to do:

* Open storage account, select container you want to backup in.
* Select container properties an copy the URL into clipboard.
* Open SSMS, login to local SQL.
* Write the 1st query in the SQL and create your credential like here backupdata and write the storage account name where you want to kept backup.
* In the container click Access key in the left side of panel then show keys to display the key and copy it and paste it in secret clause in SQL query. Execute the query.
* Write the next query for backup and pasted the URL copied from the Azure portal container in ‘to URL’ clause and mention the name of the backup as the month number and year and extension as (.bacpac).
* Mention the same credential name as in 1st query.
* Execute the backup query, it takes time to execute and after some time it created the backup of your database in Azure storage account mentioned container.









Ques: 16: Restore the database backup to a new database using azure data factory

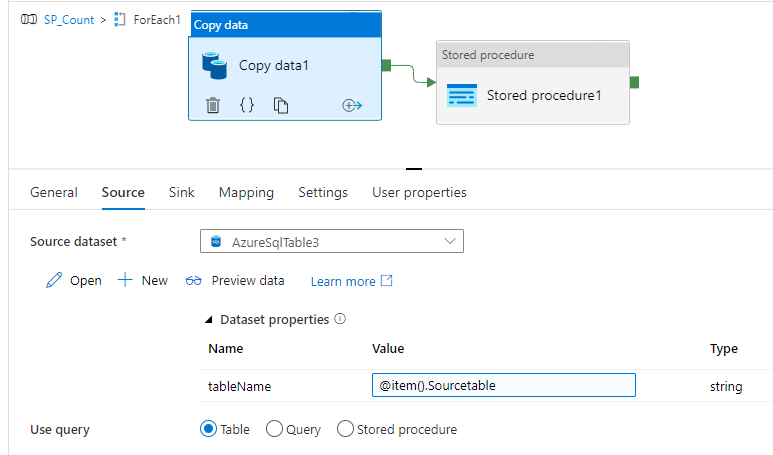
Sol: Steps to do:

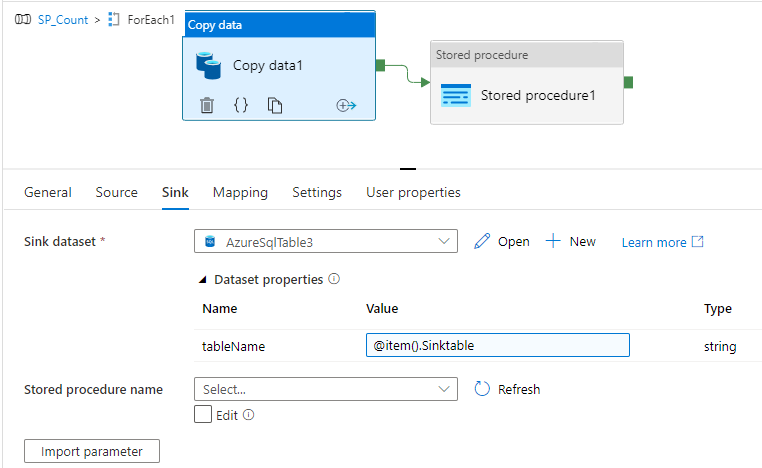
* Create a new database in the azure portal.
* Restore the backup file in the newly created database.

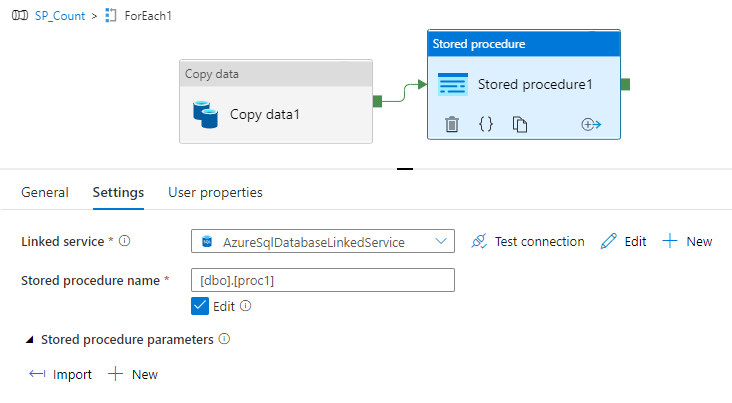
Ques:17: Create a data validation dynamic store procedure to validate table count from the source to destination

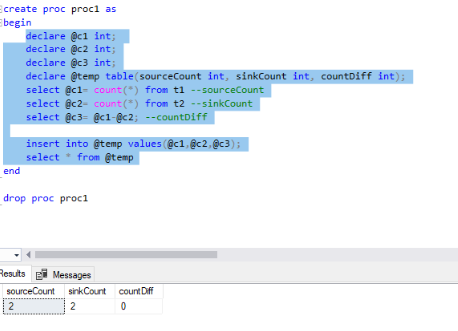
Sol: Steps to do:

* Take a lookup activity and pass the detail of source table name and sink table name in table.
* Attach the lookup with for-each and pass the output of lookup in for-each.
* Inside for-each, take a copy activity and one stored procedure activity.
* in the copy activity, select source dataset and pass the parameter of table name in it. It will fetch the value from lookup.
* In the sink side, pass the parameter in the dataset for sink table name and pass the output of for-each in it. It will fetch values from lookup.
* Pass the stored procedure in the stored procedure activity.
* The stored procedure contains the query as shown below. Which will get the number of counts from source table, sink table and count difference between then if any and stored in the temp table and the output of that as shown below.





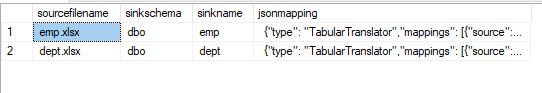


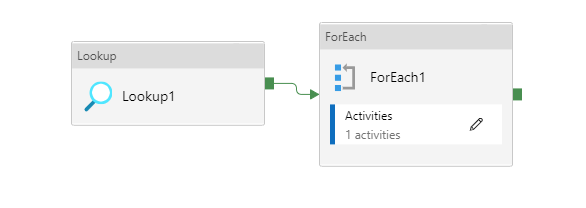


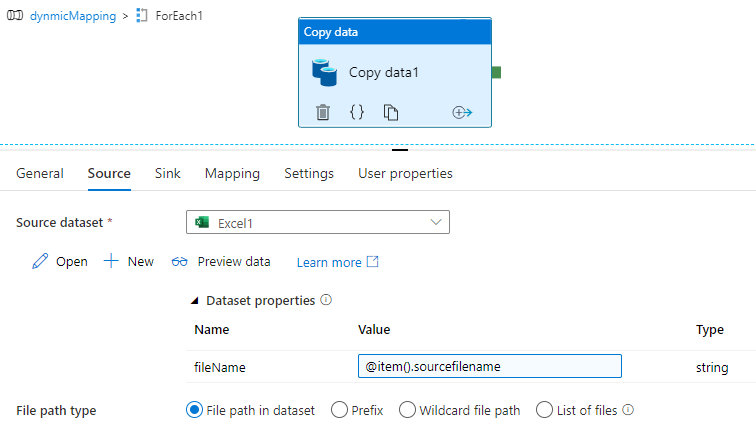
Ques:18: Implementation of dynamic mapping in Copy tables/flat files in ADF

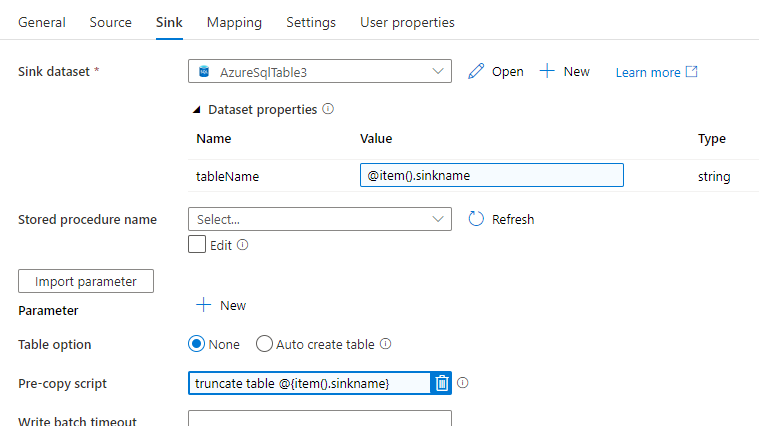
Sol: Steps to do:

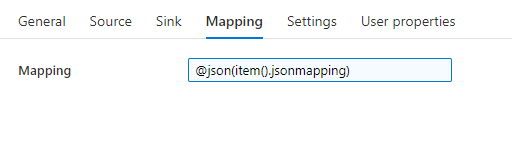
* Take a Lookup Activity which will point to your mapping table, contains table name, where the data should be taken from, and where the data should be into, including json mapping. (Shown in picture)
* Attached lookup activity with for-each activity and pass the output value of lookup in the for-each expression and in for-each take a copy activity which will copy your data from source to sink.
* Inside copy activity my source dataset is excel as my data is kept inside excel file, add a parameter in dataset for filename and pass it in the copy activity with the output of for-each activity (shown in picture).
* In the sink, as copying data from blob to azure SQL so dataset is also dataset, add parameter to dataset for table name pass it in the activity with the output of for-each activity. In the pre-copy script truncate the sink table so that every time whenever the new data load it removes the existing data. (Shown in picture)
* In the mapping column, add your json mapping in the form of json taken from the output of the for-each activity.







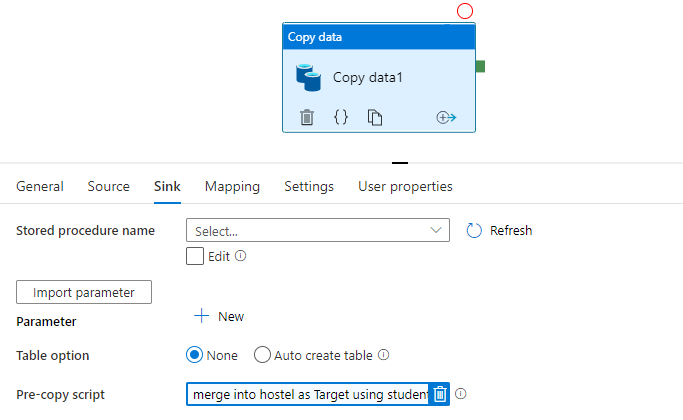


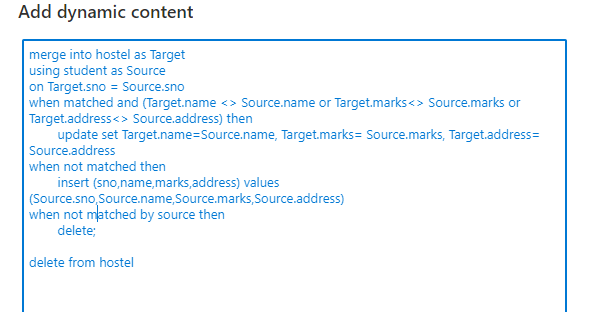


Ques:19: Dynamic Pipelines for Insert Update Multiple tables using ADF/SQL

Sol: Steps to do:

* Take a copy activity, select source and sink as a Azure SQL.
* Copy table data from source to sink.
* In pre-copy script in the sink side, write merge function/upsert logic to implement SCD type 1.

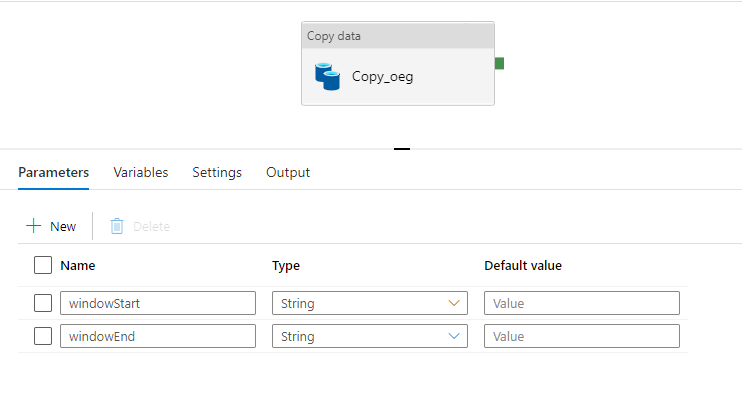


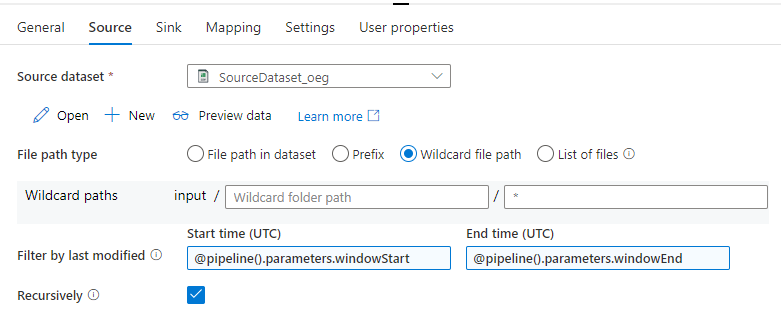


Ques:20: Perform Incremental load on a blob storage (source/sink - blob.. Batch file daily)

Sol: Steps to do:

* Set the parameter of the pipeline for window start and end.
* Select the copy activity, in the source select a blob dataset and set file path as wildcard, as we must transfer and copy multiple files.
* Set filter by last modified date and pass the parameter value in start time and end time.
* In the sink, add a blob dataset, where you want the data to be kept.
* Whenever your pipeline runs, it asked for the time for parameter set the time and execute it.

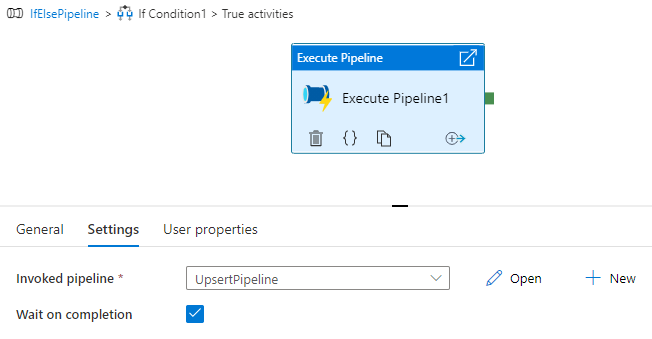
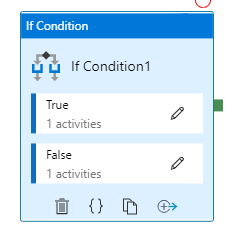




Ques:21: Link One pipeline to other pipeline with using dynamic parameters

Sol: Steps to do:

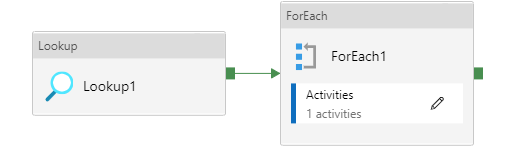
* Take if condition activity and in the condition write -@bool(equals(dayOfMonth(utcnow()),4)) which means the current month is equal to 4 or not.
* If condition matches it follow true activity, inside that it follow execute pipeline which is refer to another pipeline.
* In the else activity, it will execute a copy activity which will copy data from source to sink.

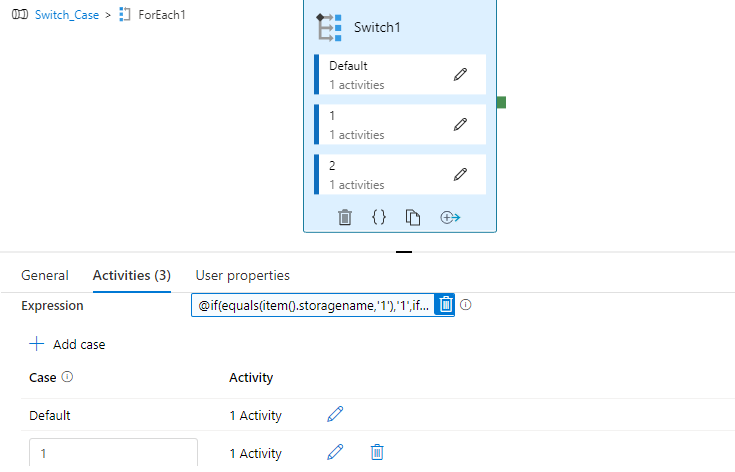


Ques:22: You have a 3 different storage account (so three linked services)  
Create a pipeline where you need to use to different blob storage according to the different requirement, might be some condition where storage accounts have to be dynamically switched.

Sol: Steps to do:

* Create 3 storage account and 3 linked service for that storage account.
* In the pipeline, take lookup activity which will point to the list of storage account.
* Attach lookup with for-each activity and pass the output of the lookup activity in the for-each expression in the setting tab.
* In the for-each activity add a switch activity and add 2 case and name its same way the lookup file named.
* In the expression or condition for the switch case used the if condition as given below.
* And inside case, a copy activity to copy data from the source to sink in the same storage account which case gets executed.
* Whenever your pipeline execute it will fetch 1st value from the lookup and pass the value via for-each and check the condition- if the value is equals to case name 1 or not if yes it executes case1 in which it will copy the data in the storage account 1. Same for everyone, lastly, if the condition hasn’t matched then it will go to default and execute the copy activity inside the default tab and copying the data in 3rd storge account.

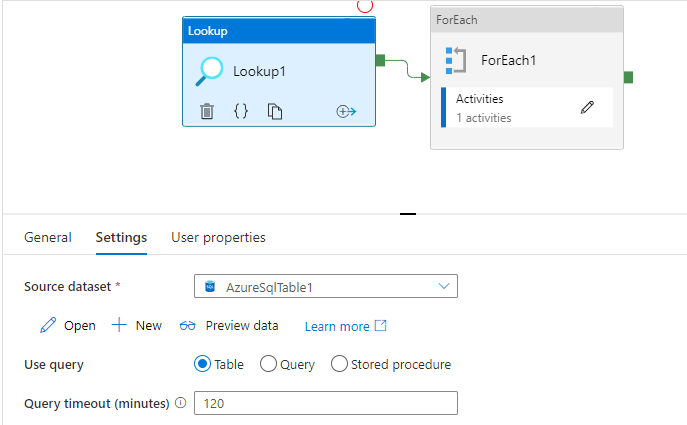


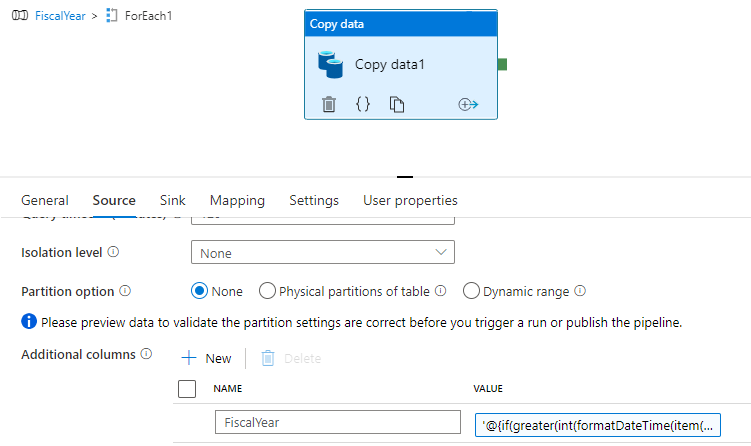


Ques:23: Write a function for the fiscal year. For example, "04=01=Y+1" when Month is 04 and year is current year then output is 01 and year will be current year+1 in ADF

Sol: Steps to do:

* Take a lookup activity which will point to a table, which contain sample dates on which fiscal year will be calculated.
* Attach the lookup activity to for-each activity the output of lookup in for-each setting.
* In for-each, take a copy activity it will copy data from source to sink, take an additional column in the sink table as fiscal year in which the year after calculated will be displayed them.
* In the source of copy activity add an ‘Additional Column’ which will add an imaginary column into source table and give value/ condition for calculating fiscal year.
* **@{if(greater(int(formatDateTime(item().SampleDate,'MM')),03),formatDateTime(utcnow(),'yyyy'),sub(int(formatDateTime(utcnow(),'yyyy')),1))}** - this condition will check if the given year is greater than 3 or not, if yes, then it will display the current year in an additional column in sink table, else deduct the current yar by 1.



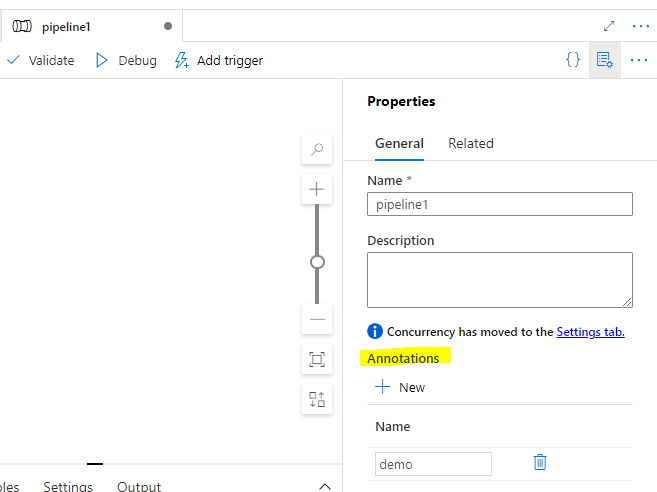


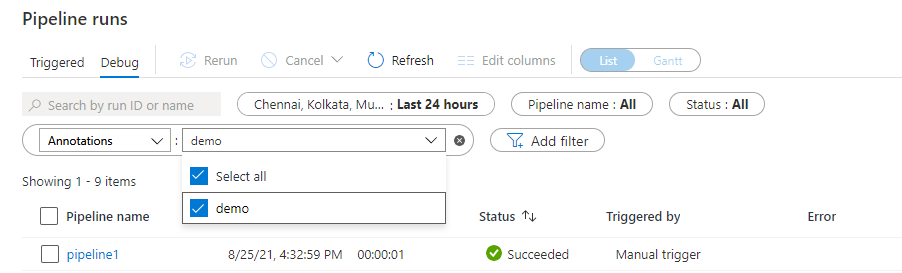
Ques:24: Write a function to get the output value in Annotations.

Sol: Annotation is just like tags, we used annotation to define pipeline, linked services, and trigger so that we can easily find specific pipeline out of loads of pipelines.

Steps to do:

* Take a pipeline, and in the properties, you can see annotation.
* Click new for annotation and write name.
* Whenever your pipelines run, in the monitor section, under pipeline runs use can easily search your pipeline by filter by – Annotation and then annotation name.

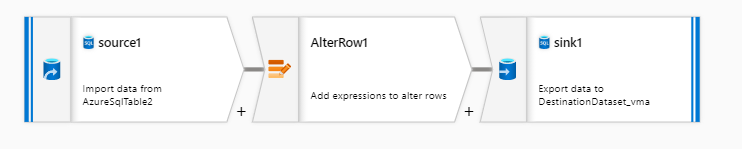


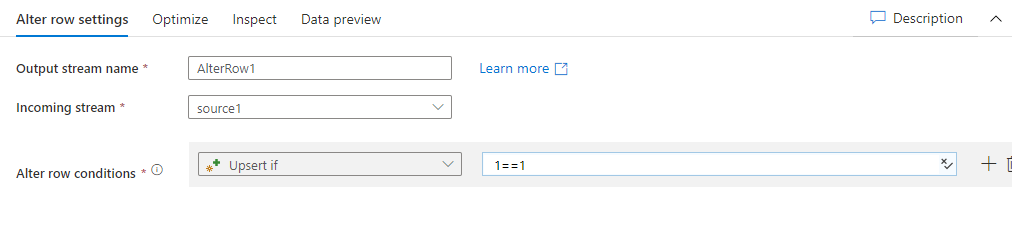


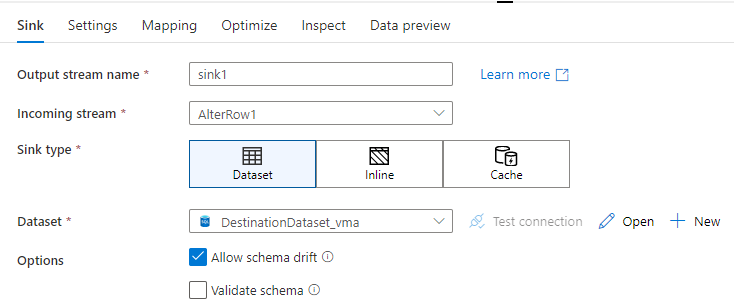
Ques:26: Implement SCD-1 logic using mapping data flows. (Source- blob, sink -SQL DB)

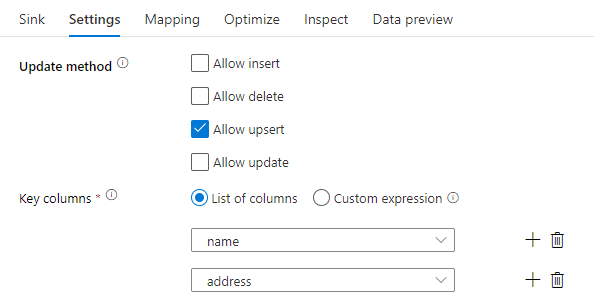
Sol: Steps to do:

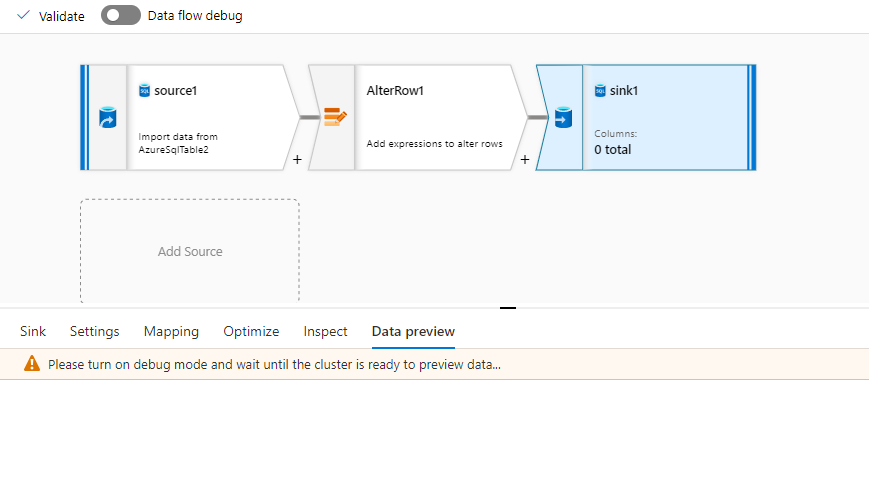
* Create a dataflow and add source.
* In source, select dataset as Azure SQL table.
* Add a branch of alter row by clicking on the + sign near source.
* Enter the name for alter row and add source name as the incoming stream and select alter row condition as a Upsert if and condition be 1==1. So that it will perform upsert logic in both the tables and check condition if it is true or not.
* In the optimize section under Alter row set partition by use current partition.
* Add sink just like you added alter row branch by clicking on + sign near alter row.
* In sink, add incoming stream as alter row and set dataset as Azure SQL table.
* In setting section under sink, select update method as Allow Upsert.
* Others set as default.
* To preview the data, turn on data flow debug mode.
* For proper usage, create a pipeline and add data flow add reference to the data flow and debug it or apply trigger for automatic use.







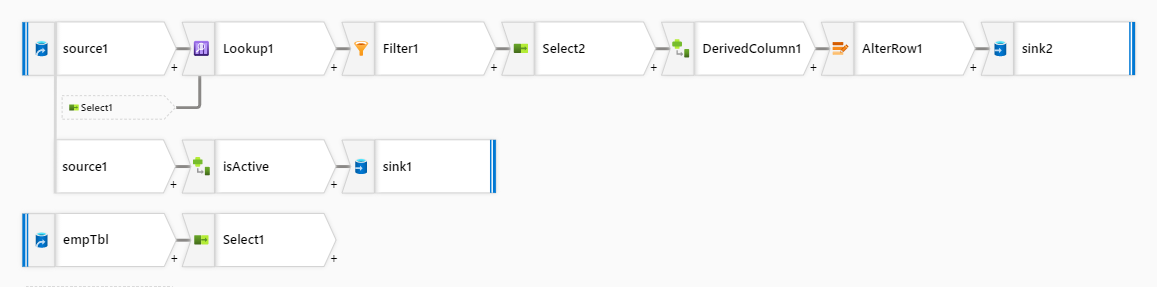


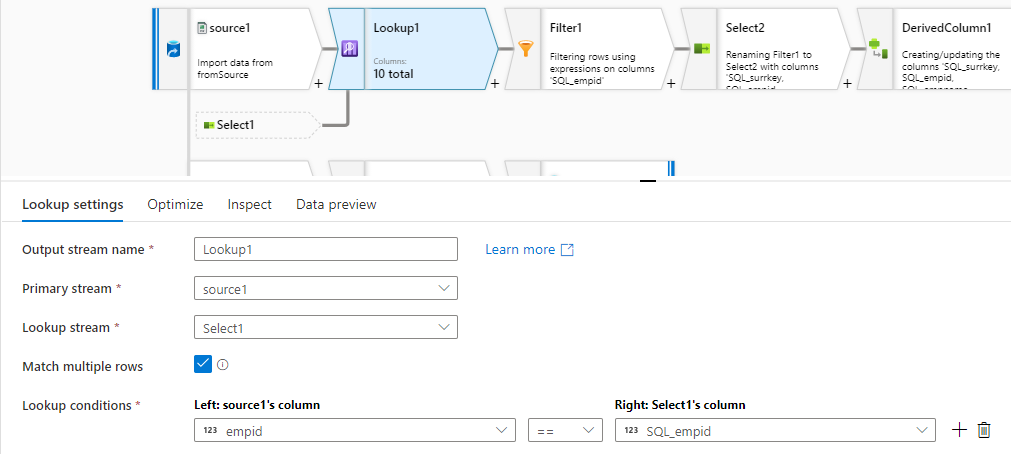


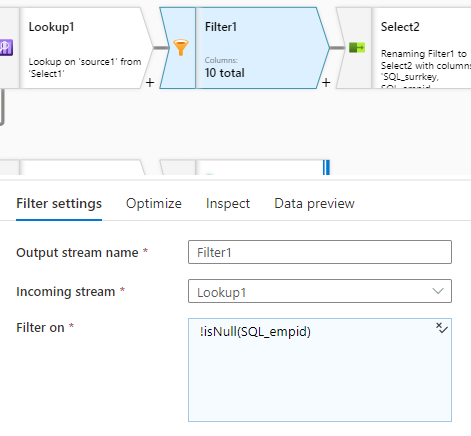
Ques:27: Implement SCD-2 logic using mapping data flows. (Source- blob, sink -SQL DB)

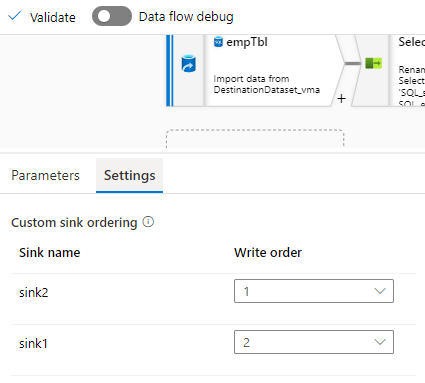
Sol: Steps to do:

* Create a dataflow and add source.
* Add a dataset as a Blob storage and pass the .csv file as a source file
* In the projection section under source edit the column datatype like for empid, salary as integer/number.
* You can inspect the data for correction.
* Add a lookup branch by clicking + sign near source. Add primary stream as source and lookup stream as select1 which is also a branch of lookup. Add a condition of lookup select a column for comparing.
* Add another branch as filter which will filter the row-wise data and add filter on condition.
* Add another branch as select2, incoming stream as a filter1.
* By adding another branch as derived column which will add another column in the sink table as is active shows the active status of the current row/ particular person.
* Add alter row, another branch attached to derived column. In the alter row branch add alter row condition as update if and apply condition as 1==1.
* Add sink which was incoming from alter row and set dataset as Azure SQL table.
* Import the source data for setting the is active row as 1 by default if any new record inserted. By adding a branch od derived column and then point it to sink as same as above.
* Take another different source as empTbl (here), which will import data from sink and add select branch to avoid any duplicity in the sink table.
* In the end, in the setting side of the dataflow not in any activity/branch set sink order which sink will execute first. Set sink2 as 1st execution because it will update the record based on condition and then sin1 will insert the record with additional column in sink side as is active as 1.
* To execute the dataflow, turn on the Data Flow Debug mode and for regular usage add a pipeline and add activity as data flow and refer the data flow and debug it for manual usage or apply trigger for automatic usage.





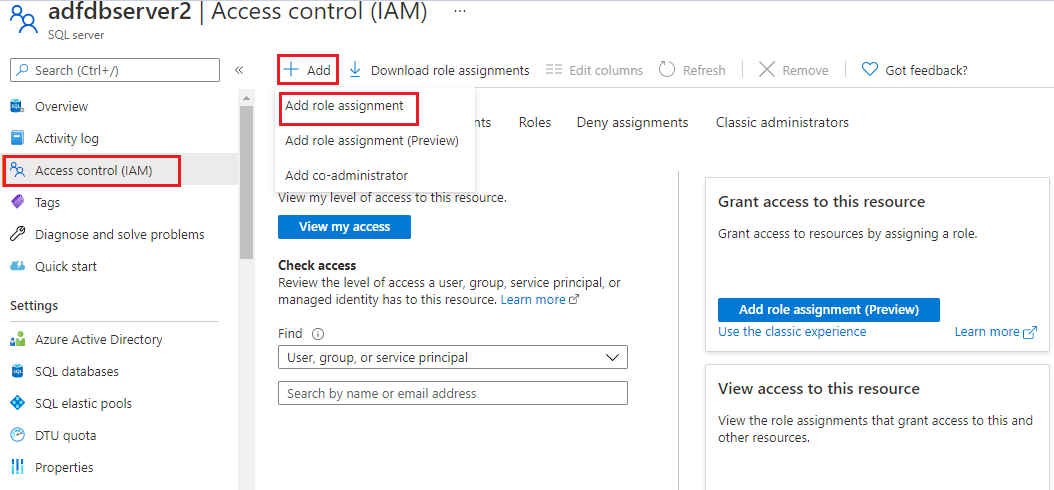


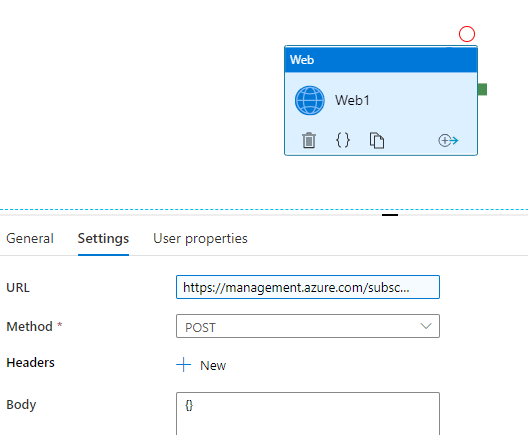


Ques:28: Schedule DW to PAUSE/RESUME daily using REST API's in ADF

Sol: Steps to do:

* Create a data warehouse account in the azure portal.
* After created account on data warehouse, click on server name and go to Access control (IAM) on the left side of the panel.
* Add a role assignment, select the role as a contributor and add member to your account.
* In the Azure Data Factory, take a pipeline and add web activity.
* In the URL section inside, past the URL from Microsoft Docs for pause/ resume and edit the subscription id, resource, database name and database server name.
* In the body section, keep it blank so {}.
* Select integration runtime and select managed identity as authentication.
* In the resource –‘ <https://management.azure.com/> ‘

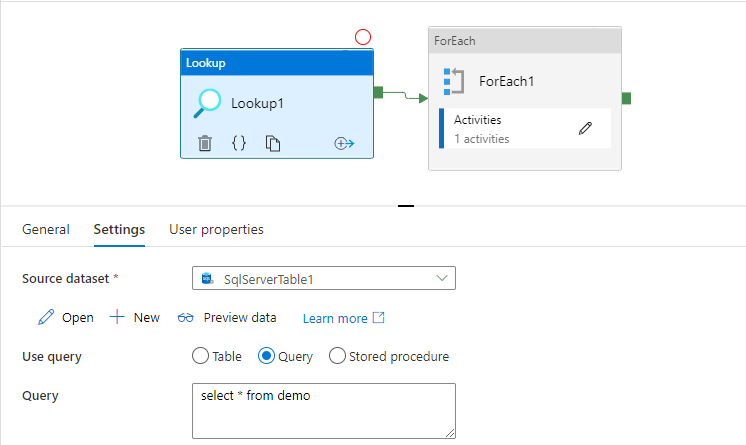


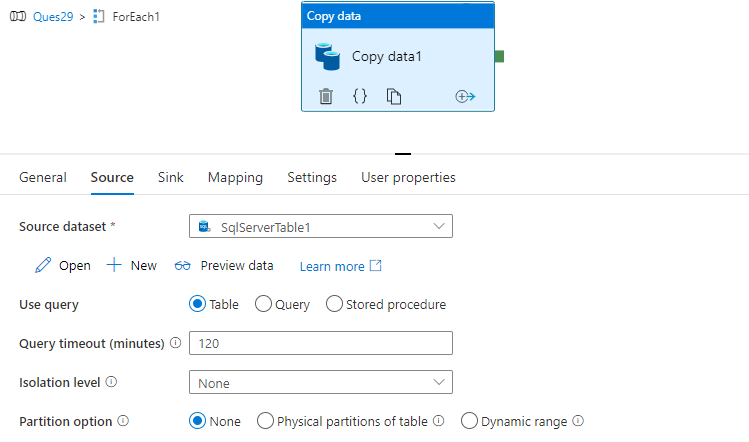


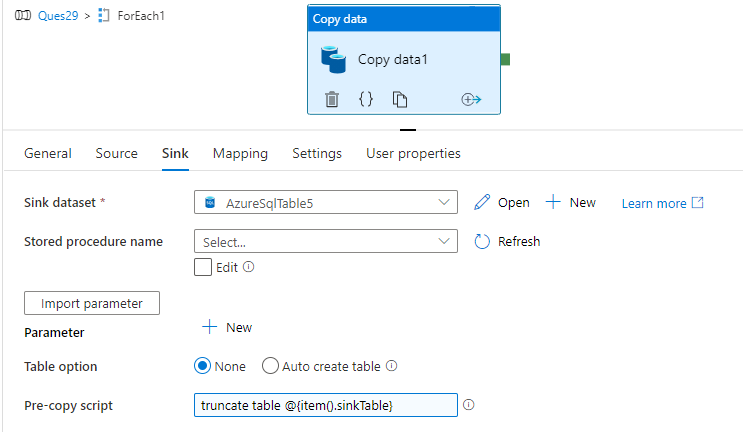
Ques:29: Copy a table from SQL DB into Azure SQL DB/DW. Create a new table in the sink side and remove the white spaces in the column headers then map the source columns with sink. Also, you do not have to copy all the columns. (e.g., If there are 10 columns in the source table then only copy data for 8 of them)

Sol: Steps to do:

* Take a lookup, connect local SQL in dataset, which will point to a table ‘demo’, which contain details of source table, sink table and json mapping for column of source and sink table which you want to copy.
* Attached lookup with for-each activity and pass the output of the same.
* In the for-each take copy activity, which will copy data from source to sink, connect dataset of source as local SQL.
* In the sink side, connect dataset as an Azure SQL, and write pre-copy script for truncate table to avoid duplicacy.
* In the mapping section, **@json(item().jsonMapping) ,**jsonMapping is the name of the column.



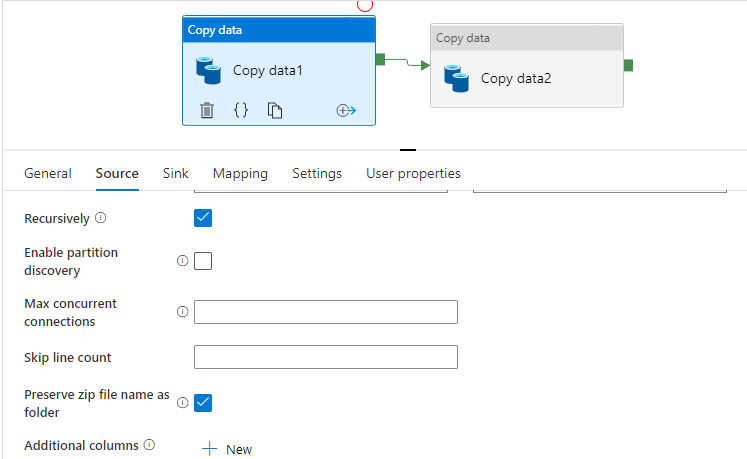


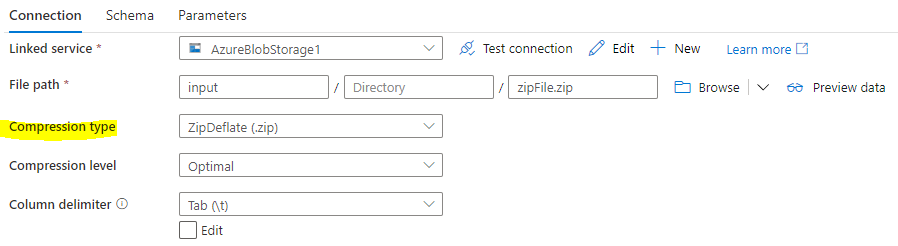


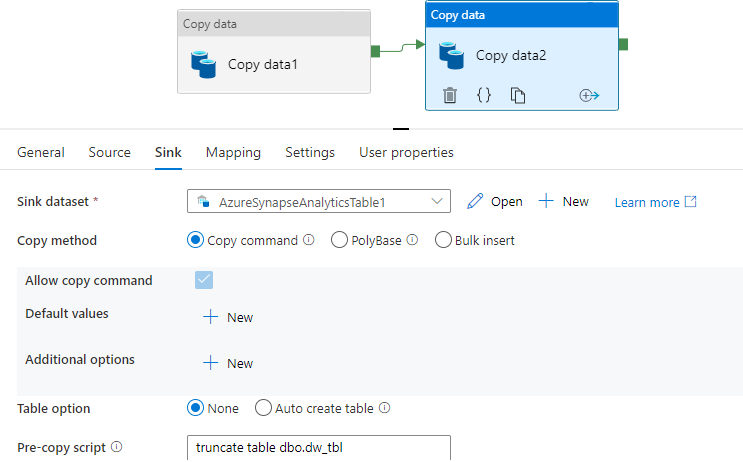
Ques:30: Copy a zipped text (tab delimited) file from blob storage to Azure Synapse table using data Factory

Sol: Steps to do:

* Create a .zip file and upload it into blob
* In the Azure Data Factory, take a copy activity which will unzip the .zip file. In the source dataset use compress type as file Zip Deflate.
* When you select compress as a Zip Deflate then tick on ‘preserve zip file name as folder’ option in the source side of first copy activity.
* In the sink side, select the dataset where you want to store the unzip file.
* Take another copy activity, the source will point to the sink dataset of first copy activity which will point to the unzip of the file and then copy it into data warehouse table.
* Use pre-copy script for truncating the table in the data warehouse to avoid duplicity.





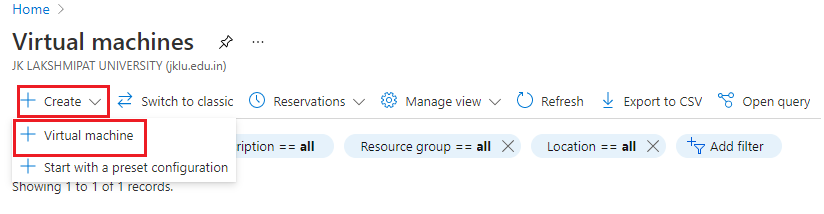


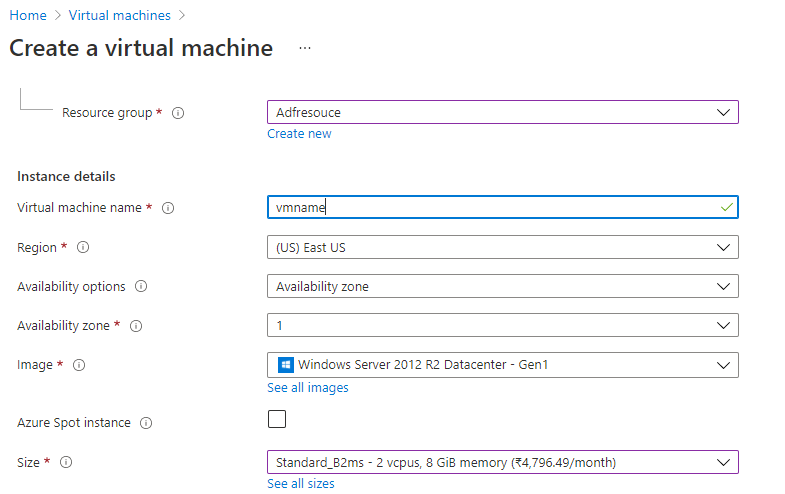
**EXTRA TASK:**

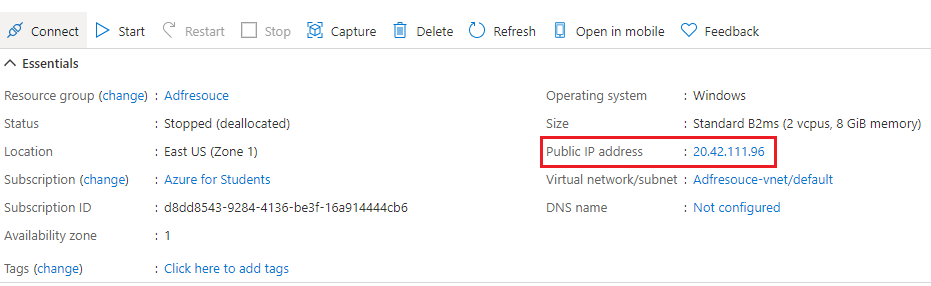
Task:1: R&D on how to alert rules on virtual machine when self-hosted IR goes offline.

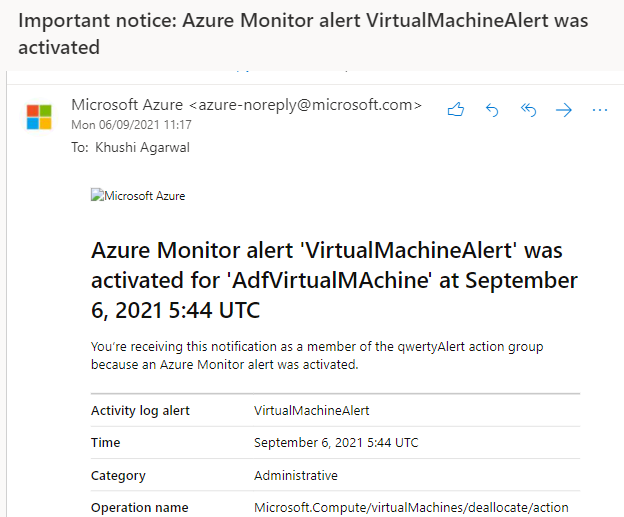
Sol: Steps to do:

* Create a virtual machine. Fill the details of resource group, subscription id and select image as windows 2012 and size as b2ms or can select from more.
* Select create and review and create the virtual machine, this will take some time to deploy.
* Copy the public IP address and press windows + R with your keyboard, write mstsc.
* This will open a window where you must paste the copied IP address.
* On the left side of the panel, under monitor section click alert.
* Create a new alert rule and set the condition as- ‘deallocate the virtual machine’.
* Set up the self-Hosted IR in the virtual machine computer.
* Whenever virtual machine stops it will send you the message, call, emails, or notification that virtual machine is set as power off.









Task:2: Data load from Rest API

Sol: Steps to do:

* Create App Registration in the Azure portal.
* In the Azure Data Factory, create a pipeline.
* Take a copy activity, select source dataset as Rest and enter the base URL (can copy from dummy Rest API) and select authentication type as Anonymous (as it is public) and select the method as Get.
* Under sink side of copy activity, select dataset as Azure SQL Database and select table as auto create. When pipeline executes it will copy your data from Rest API to SQL Database.Graphical user interface, text, application, email

  Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Task:3: Perform an incremental load, taking lookup at blob. Also delete the extra files.

Sol: Steps to do:

* Create a file contain details of updated date as created Date and table name and store this in blob storage.
* In ADF, take a lookup activity, which will point to the detail file which was stored at blob.
* Attach for-each activity and pass the output of the lookup in the for-each activity.
* Inside for-each, take 3 copy activities: 1. Which will copy data from SQL to BLOB. 2. Will copy data from BLOB to SQL (in different schema) and 3. Which will fetch max date from the table and store in the BLOB file. Here, in copy activities 1 and 2 Blob is like staging area.
* In the 1st copy activity, under source side select dataset as Azure SQL DB and select query for incremental load as- *select \* from input.@{item().tablename} where created\_date > '@{item().created\_date}'* ,select sink side as blob, and take 2 parameter of folder name and file name and set the value for folder name as- *@concat('input','/',item().tablename,'/',formatDateTime(utcnow(),'yyyy'),'/',formatDateTime(utcnow(),'MM'),'/',string(sub(int(formatDateTime(utcnow(),'dd')),1)))* and filename as- [*@concat(item().tablename,'\_',string(sub(int(formatDateTime(utcnow(),'dd')),1)),formatDateTime(utcnow(),'MM'),'.txt')*](mailto:as-@concat(item().tablename,'_',string(sub(int(formatDateTime(utcnow(),'dd')),1)),formatDateTime(utcnow(),'MM'),'.txt'))
* In the 2nd copy activity, the source will be the sink of the 1st copy activity with the same folder and file name and value. In the sink side select Azure SQL as a sink dataset and give the different schema.
* In the 3rd copy activity, take sink as Azure SQL and write query which will fetch max date from the table, the query- *@concat('select max(created\_date) as created\_date from output. ',item().tablename)* ,sink section will be blob and create a folder where the fetched values will be stored and filename will be- *@concat(item().tablename,'\_list.csv')* . Add an additional column for table name in the file so that it will merge easily with the lookup file and update lookup file. In Mapping side, under Type conversion section write Date Time format as- yyyy-MM-dd, which will convert the date from Date Time to date.
* Outside for-each, take a copy activity which will merge the fetched max date files into lookup file, so it will update lookup file date. Take source and sink dataset as Blob. In the source side, select file path as wildcard and give the file path where the fetched max date file stored and filename as- \*\_list.csv , In the sink side select file as lookup on which merge will perform and select copy behavior as Merge data.
* Take a Delete activity, which will delete the extra files which was the max date file. Select source as blob and file path as Wildcard file path, the value- \*\_list.csv and in logging setting, tick on enable logging and select logging account linked service as Blob and give the folder path where the deleted files detail will be stored.

*Waterfall chart

Description automatically generated with low confidence*

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Task:4: Copy data from Blob to Azure SQL. Fetch data from consideration detail file.

Sol: Steps to do:

* Take a copy activity which will structure the data, select source dataset as blob and set 5 in skip line count which will skip 5 lines from the top. Select sink as blob.
* Take lookup activity, select source dataset as Azure SQL which will point to the table detail in the SQL table that contain table name, category, target, and json mapping for each category in a table.
* Take for-each activity which will attach to lookup activity and pass the output of lookup in the for-each setting.
* Inside for-each, take a stored procedure, which will delete the data according to current fiscal year, so that the table does not contain any duplicity. In stored procedure activity, select import parameter pass the value of table name in the tbl parameter and for year parameter pass the value as- '@{if(greater(int(formatDateTime(utcnow(),'MM')),03),formatDateTime(utcnow(),'yyyy'),sub(int(formatDateTime(utcnow(),'yyyy')),1))}'
* The stored procedure created as-

create or alter proc spproc (@tbl varchar(30), @year varchar(20)) as exec ( 'delete from '+@tbl+' Where date >= concat('+@year+',''-03-31'')')

* Create the tables in the Azure SQL so that the data will copy into the tables.
* Take a copy activity, select source dataset as blob and add additional column for category and target and take the value of both from lookup.
* Select sink dataset as Azure SQL and take parameter of tablename which will take tablename from lookup table via for-each and copy the data into the table.
* Under mapping section, add dynamic contain and pass the value of jsonmapping from lookup table into mapping section, also convert it into json format.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface

Description automatically generated

Task:5: Create a table contain date column and set indexing on date and datatype of date column must be Date Time and fetch the data according to date and in execution plan be seek.

Sol: Indexing is a way of sorting several records on multiple fields. Creating an index on a field in a table creates another data structure which holds the field value, and a pointer to the record it relates to. This index structure is then sorted, allowing Binary Searches to be performed on it.

There are 2 types of indexes: Clustered and Non-clustered index.

Clustered index can be on multiple columns but there is only one clustered index on a table. Clustered index is faster as compared to non-clustered index. It is in ordered structured and no need for additional disk space.

Non-Clustered index- There can be multiple non-clustered indexes in a table. It is slower than Clustered index. The non-clustered index is in unordered structured and required additional space.

Index seek- retrieve selected rows (on where condition). Index scan- retrieve all rows from the table. It will decrease the query performance. In the program always perform index seek.

Key lookups are on non-clustered index, when we select \* then it will give scan and key lookup. to avoid this use, select Col name then will give seek and no key lookup.

The lookups refer to how many times the query required data to be pulled from the clustered index or the heap (does not have a clustered index). Lookups are also something you want to try to avoid.

Indexing is the way to get an unordered table into an order that will maximize the query’s efficiency while searching.

Steps to do:

* Create a table which contain date column which is of datatype Date Time, name, and age.
* Insert the values into table and use different time for same date.
* Create clustered index on date. Which will order your data.
* Then search the data according to date and check if execution plan is in seek or not, if not then do it seek.

Table

Description automatically generatedText, letter

Description automatically generated

Task:6: Perform Dynamic Aliasing.

Sol: Steps to do:

* Create a table which contain id, name, amount, and date. Date will be kept of different month. This is our main table. (DAlias)
* Create another table which is final table (AliasDemo), which contain id, name, OctAmt, NovAmt, DecAmt. The value for the amounts will be based on the month of date column.
* Create a temporary table as @tmp in which contain id and distinct names of the main table. The count of this table will be stored in a variable @x.
* With loops and equation insert the value as required in the final table.
* When we view the final table then we get the output as required.

Table

Description automatically generated Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated with medium confidence

Task:7: