
AZURE PROJECT

NTT DATA SERVICES

By

SHANMUGA ARTHI DHARMARAJ

Guided By

MALLIKARJUNA GANDHAMSETTY

Table of Contents:

Problem Statement 1:.....	5
Solution :	6
Screenshot 1.1: Resource Group, Data Factory, Storage Account – Blob Storage and Azure Data Lake Storage, SQL Database and SQL Server – resources created details.....	6
Screenshot 1.2.a: Creating DB in SSMS.....	7
Screenshot 1.2.b: Populating Table using insert query in SSMS.....	7
Screenshot 1.2.c: Verifying the data inserted in Tables.....	8
Screenshot 1.3.a: Displays both Pipelines, Copy Activity and Source Dataset	9
Screenshot 1.3.b: Displays Sink Dataset	9
Screenshot 1.4.a: Pipeline Activity Details	10
Screenshot 1.4.b: Pipeline Debug Details	10
Screenshot 1.4.c: Pipeline Manual Trigger Details	11
Screenshot 1.5: Output Verification.....	11
Files:	11
Problem Statement 2:.....	12
Solution :	12
Screenshot 2.1: Displays Properties Tab of Copy Data Tool.....	12
Screenshot 2.2.a: Displays Source Tab of Copy Data Tool	13
Screenshot 2.2.b.: Displays Preview Data for all the Tables of SalesLT.....	13
Screenshot 2.3: Displays Sink Data Linked Service, Server Name, Dedicatedpool.....	14
Screenshot 2.4.a: Displays Table Mapping Source to Sink	14
Screenshot 2.4.b: Displays Column Mapping Source to Sink	15
Screenshot 2.5: Displays Settings Page, Staging, Linked Service, PolyBase setting	15
Screenshot 2.6: Displays Summary Page.....	16
Screenshot 2.7: Displays Summary Page.....	16
Screenshot 2.8: Displays Pipeline Monitoring.....	17
Screenshot 2.9: Displays Pipeline Debug Details	17
Screenshot 2.10: Output Verification.....	18
Problem Statement 3:.....	19
Solution:	19
TextFile :	19
Screenshot 3.1: Displays Text File in Blob Storage inside Container	20
Query:	20
Screenshot 3.2.a: Displays both Pipelines, Copy Activity and Source Dataset	21

Screenshot 3.2.b: Displays Sink Dataset.....	21
Screenshot 3.3.a: Pipeline Activity Detail	22
Screenshot 3.3.b: Pipeline Debug Detail.....	22
Screenshot 3.3.c: Pipeline Manual Trigger detail	23
Screenshot 3.4: Text File copied from Azure Blob Storage to Azure SQL Database	24
Screenshot 3.5: Output Verification.....	24
Problem Statement 4:	25
Solution :	25
Screenshot 4.1.a: Displays the Semicolon Delimited Files in Blob Storage.....	25
Screenshot 4.1.b: Displays the Comma Delimited Files in Blob Storage.....	26
Screenshot 4.2.a: Displays the Source Dataset	27
Screenshot 4.2.b: Displays the Sink Dataset.....	27
Screenshot 4.3.a: Displays the Pipeline, Lookup Activity Details.....	28
Query:	28
Screenshot 4.3.b: Displays the Preview Data of the Lookup Activity	29
Screenshot 4.3.c: Displays the Source Dataset Copy Activity within ForEach Activity	30
Screenshot 4.3.d: Displays the Sink Dataset Copy Activity within ForEach Activity	30
Screenshot 4.4: Pipeline Debug Details	31
Screenshot 4.5.a: Output Verification I.....	31
Screenshot 4.5.b: Output Verification II	32
Screenshot 4.5.c: Output Verification III	32
Screenshot 4.5.d: Output Verification IV	33
Problem Statement 5:	34
Solution:	34
Screenshot 5.1.a: Displays table creation in SSMS	34
Screenshot 5.1.b: Displays Table Creation in SSMS:.....	35
Screenshot 5.2.a: Displays both pipeline, Old Water Mark LookUp Activity and Watermark Dataset.....	35
Screenshot 5.2.b: Displays verifying the Data Source Table in Azure SQL Database	36
Screenshot 5.2.c: Displays Linked Services for Watermark Dataset.....	36
Screenshot 5.2.d: Displays both pipeline, New Water Mark LookUp Activity with Query	37
Query:	37
Screenshot 5.2.e: Displays Incremental Copy Activity and Source Dataset Details	37
Query:	37
Screenshot 5.2.f: Displays Sink Dataset Details	38

Screenshot 5.3.a: Pipeline Debug Details	38
Screenshot 5.3.b: Pipeline Activity Details.....	39
Screenshot 5.3.c: Pipeline Debug Detail II	39
Screenshot 5.3.d: Pipeline Trigger Details.....	40
Screenshot 5.4: New records insertion in Data Source Table	40
Screenshot 5.5: Output Verification: Files generated after every run.....	41
Files:	41
Problem Statement 6:	42
Solution :	42
Screenshot 6.1: Creating DB in Azure SQL Database.....	42
Screenshot 6.2.a: Creating Person Table.....	43
Screenshot 6.2.b: Creating Course Table	43
Screenshot 6.2.c: Creating Student Table	44
Screenshot 6.2.d: Creating Credit Table	44
Screenshot 6.3.a: Populating Person Table	45
Screenshot 6.3.b: Populating Course Table	45
Screenshot 6.3.c: Populating Student Table.....	46
Screenshot 6.3.d: Populating Credit Table.....	46
Screenshot 6.4: Output Verification: Query to extract the student details.....	47
Files:	47
Problem Statement 7:	48
Solution :	48
Screenshot 7.1: Creating DB in Azure SQL Database.....	48
Screenshot 7.1.a: Creating Person Table.....	49
Screenshot 7.1.b: Creating ContactDetailType Table	49
Screenshot 7.1.c: Creating Address Table	50
Screenshot 7.1.d: Creating ContactDetail Table	50
Screenshot 7.2.a: Populating Person, ContactDetailType, Address Table.....	51
Screenshot 7.2.b: Populating Table ContactDetail	51
Screenshot 7.3: Output Verification: Query to extract the Person details	52
Files:	52
Problem Statement 8:	53
Solution :	53
Screenshot 8.1.a: Datasets for PersonInfoDb.....	53
Person - Schema:.....	53

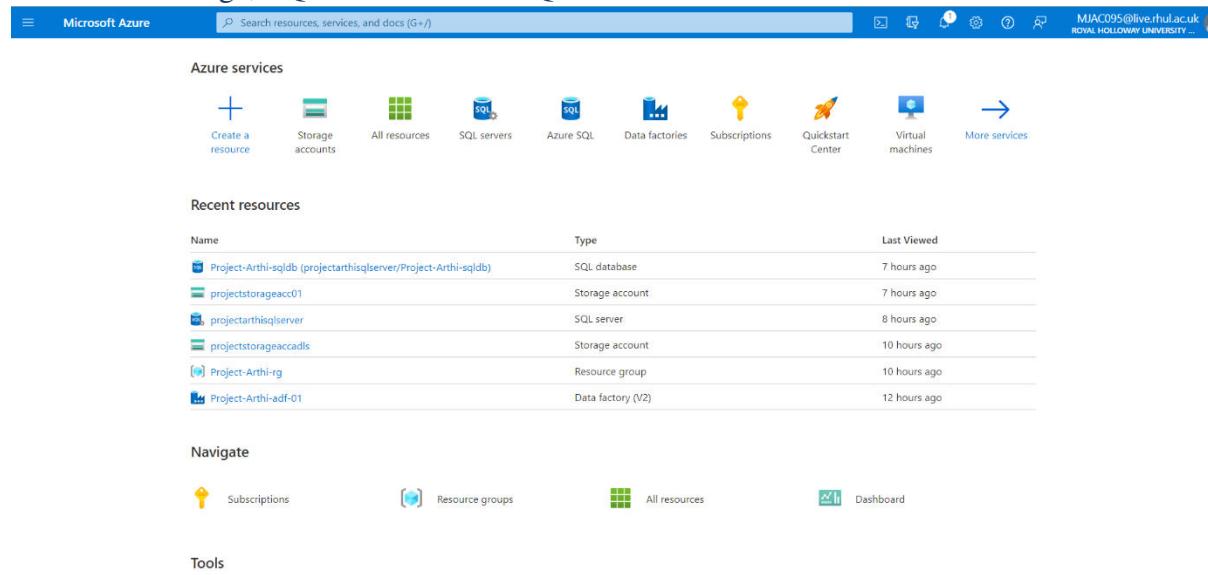
Address - Schema:	54
ContactDetailType - Schema:	55
ContactDetail - Schema:	56
Screenshot 8.1.b: Query extracting People full contact info.....	57
Query:	57
Screenshot 8.1.c: Output in JSON FORMAT.....	58
Sample Output:	58
Files:	59

Problem Statement 1: Ingest data from a single Table in SQL Server database to Azure Blob storage

Solution :

- Created a Resource group, named as: Project-Arthi-rg
- Created a Blob Storage Account, named as: projectstorageacc01
- Created a data factory, named as: Project-Arthi-adf-01
- Created a self-hosted integration runtime named as: IR-1

Screenshot 1.1: Resource Group, Data Factory, Storage Account – Blob Storage and Azure Data Lake Storage, SQL Database and SQL Server – resources created details



Name	Type	Last Viewed
Project-Arthi-sqldb (projectarthisqlserver/Project-Arthi-sqldb)	SQL database	7 hours ago
projectstorageacc01	Storage account	7 hours ago
projectarthisqlserver	SQL server	8 hours ago
projectstorageacc01	Storage account	10 hours ago
Project-Arthi-rg	Resource group	10 hours ago
Project-Arthi-adf-01	Data factory (V2)	12 hours ago

- Create SQL Server and Azure Storage linked services: In SQL Server Management System, created own Database named **Store** containing 7 Tables namely **dbo.Category, dbo.Customer, dbo.Outlet, dbo.Product, dbo.Product_Category, dbo.Purchase, dbo.Purchaseitem, dbo.Stock.**
- Created database and populated the table using insert query values in each table.

Screenshot 1.2.a: Creating DB in SSMS

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure for 'localhost.Store (sa (70))'. The 'Tables' node under 'BikeStores' is expanded, showing various table definitions. The 'Customer' table definition is highlighted:

```
1 --> 1. Create tables
2
3 --Outlet(_outlet_name_, address_street, address_city).
4
5 CREATE TABLE Outlet(
6     outlet_name varchar(50) NOT NULL UNIQUE,
7     address_street varchar(50),
8     address_city varchar(50),
9     primary key(outlet_name));
10
11 --Customer(_customer_nr_, customer_name, address_street, address_city).
12
13 CREATE TABLE Customer(
14     customer_nr varchar(10) NOT NULL UNIQUE,
15     customer_name varchar(50),
16     address_street varchar(50),
17     address_city varchar(50),
18     primary key(customer_nr));
19
```

The status bar at the bottom indicates 'localhost (15.0 RTM) sa (70) Store 00:00:00 0 rows'.

Screenshot 1.2.b: Populating Table using insert query in SSMS

Screenshot 1.2.c: Verifying the data inserted in Tables

```

Object Explorer
Connect ... SQL Server 15.0.2009 - sa ...
localhost (SQL Server 15.0.2009 - sa)
Databases
System Databases
Database Snapshots
BikeStores
practicedb
Store
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.Category
dbo.Customer
dbo.Outlet
dbo.Product
dbo.Product_Category
dbo.Purchase
dbo.PurchaseDetail
dbo.Stock
Views
External Resources
Synonyms
Programmability
Service Broker
Storage
Security
Security
Server Objects
Replication
PolyBase
Always On High Availability
Management
Integration Services Catalogs
SQL Server Agent (Agent XPs disabled)
...
Ready

SQLCreateQuery1.sql ..._out.Store (sa (70)) StoreSQLQuery.sql ..._host.Store (sa (72)) SQLQuery3.sql - localhost.Store (sa (71))*
1 select * from Outlet
2
3 select * from Category

Results Messages
outlet_name address_street address_city
1 Ads Nawziel Mount Street Woodside
2 Mentfield Crescent Brooklyn
3 Natural Valley Kingsley Road Pittsfield
4 Red Outlet Senator Brooklyn
5 Sklep Awziel High Street Rye

category_id description
1 1 Toe Boot
2 2 Block Heel
3 3 Brogues
4 4 Ivory Embroidered Tie Front Top
5 5 Broderie Yoke Cotton Top
6 6 Blue MidRise Denim Shorts
7 7 Knee Length Shift Dress
8 8 Multicolored Tie Dye Maxi
9 9 Tunic Dress Top

Query executed successfully
localhost (15.0 RTM) sa (71) Store 00:00:00 20 rows
Ln 2 Col 1 Ch 1 INS

```

Sample: **dbo.Outlet**

-- 1. Create Tables

--Outlet(_outlet_name_, address_street, address_city).

CREATE TABLE Outlet(

```

outlet_name varchar(50) NOT NULL UNIQUE,
address_street varchar(50),
address_city varchar(50),
primary key(outlet_name));

```

--2. Insert Tables

--Outlet(_outlet_name_, address_street, address_city).

INSERT INTO Outlet VALUES ('Sklep Awziel', 'High Street', 'Rye'),

('Mentfield', 'Crescent', 'Brooklyn'),

('Natural Valley', 'kingsley Road',

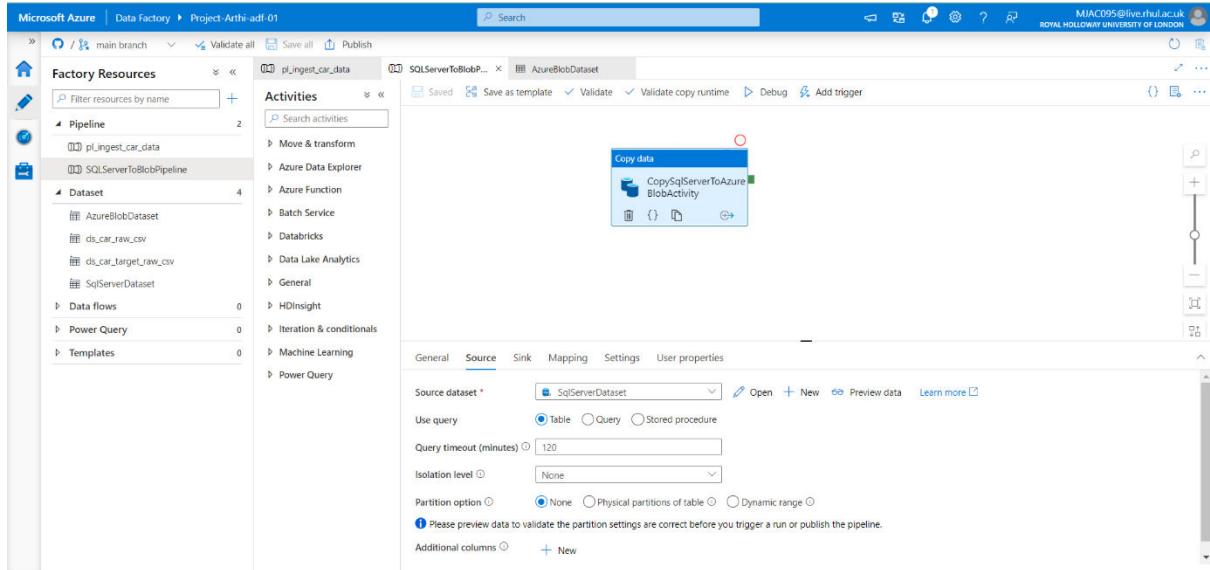
'Pittsfield'),

('Red Outlet', 'Senator', 'Brooklyn'),

('Ads Nawziel', 'Mount Street', 'Woodside');

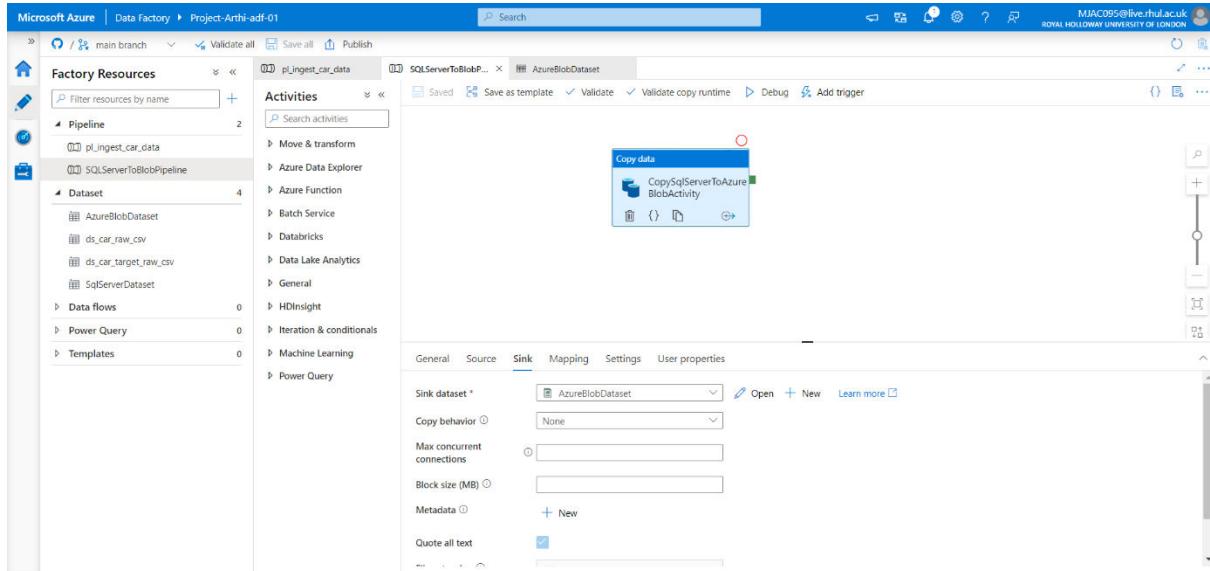
- Created SQL Server and Azure Blob datasets named as **projectstorageacc01** as showed in Screenshot 1.
- Created a pipeline named **SQLServerToBlobPipeline** with a copy activity **“CopySqlServerToAzureBlobActivity”** to move the data from on-premises to blob storage. Then set up Source dataset: **SqlServerDataset**.

Screenshot 1.3.a: Displays both Pipelines, Copy Activity and Source Dataset



- Sink dataset as: **AzureBlobDataset**, created file path as **myblob / Directory: fromonprem/ File: @CONCAT(pipeline().RunId, '.txt')**, to rename file as PipelineRunID.txt.

Screenshot 1.3.b: Displays Sink Dataset



- If the file section adds dynamic content not given user, it will automatically get the filename from the copied data.
- Started a pipeline run, debugged, pipeline validation done. Published All

Screenshot 1.4.a: Pipeline Activity Details

The screenshot shows the 'Details' view for a pipeline run. It displays a summary of the activity: 'SQL server' to 'Azure Blob Storage' with a status of 'Succeeded'. Key metrics include 288 bytes read, 5 rows read, and 235 bytes written. The copy duration was 00:00:05 with a throughput of 57.344 bytes/s. The activity was triggered at 10/3/21, 10:58:09 PM. The data was transferred via a queue, with a total duration of 00:00:01. The transfer process involved reading from source (00:00:00) and writing to sink (00:00:00). Data consistency verification was not verified. A satisfaction rating of 5 stars is shown.

Screenshot 1.4.b: Pipeline Debug Details

The screenshot shows the 'Pipeline runs' page in Microsoft Azure Data Factory. It lists three runs under the 'Triggered' tab. The runs are:

Pipeline name	Run start	Run end	Duration	Status	Triggered by	Error	Run ID
SQLServerToBlobPipeline	10/3/21, 10:58:07 PM	10/3/21, 10:58:16 PM	00:00:08	Succeeded	Manual trigger		622bd2ba-9138-4a6b-b883...
SQLServerToBlobPipeline	10/3/21, 10:57:27 PM	10/3/21, 10:57:38 PM	00:00:10	Succeeded	Manual trigger		9aeb4ae0-0ac1-4374-81b7...
pl_Ingest_car_data	10/3/21, 7:52:47 PM	10/3/21, 7:53:03 PM	00:00:15	Succeeded	Manual trigger		5f3573fc-6e23-46d5-a302...

- Added trigger by clicking on trigger now
- Monitoring the relative pipeline SQLServerToBlobPipeline

Screenshot 1.4.c: Pipeline Manual Trigger Details

The screenshot shows the Microsoft Azure Data Factory interface. The left sidebar has a 'Pipeline runs' section selected. The main area displays a table titled 'Pipeline runs' with one item listed:

Pipeline name	Run start	Run end	Duration	Triggered by	Status	Error	Run	Parameters	Annotat
SQLServerToBlobPipeline	10/4/21, 5:01:36 AM	10/4/21, 5:02:00 AM	00:00:24	Manual trigger	Succeeded				Original

- Output verified: Table copied to Azure blob storage as .txt file.

Screenshot 1.5: Output Verification

The screenshot shows the Microsoft Azure Storage Explorer interface. The left sidebar has 'Storage Explorer (preview)' selected. The main area shows a blob container named 'myblob' containing a single file named 'c1ab545b-7176-4cf6-b155-6929dd79946.txt'. The file details are as follows:

NAME	ACCESS TIER	ACCESS TIER LAST MODIFIED	LAST MODIFIED	BLOB TYPE	CONTENT TYPE	SIZE	STATUS
c1ab545b-7176-4cf6-b155-6929dd79946.txt	Hot (inferred)	10/3/2021, 11:12:33 PM	10/3/2021, 11:12:33 PM	Block Blob	application/octet-stream	235 B	Active

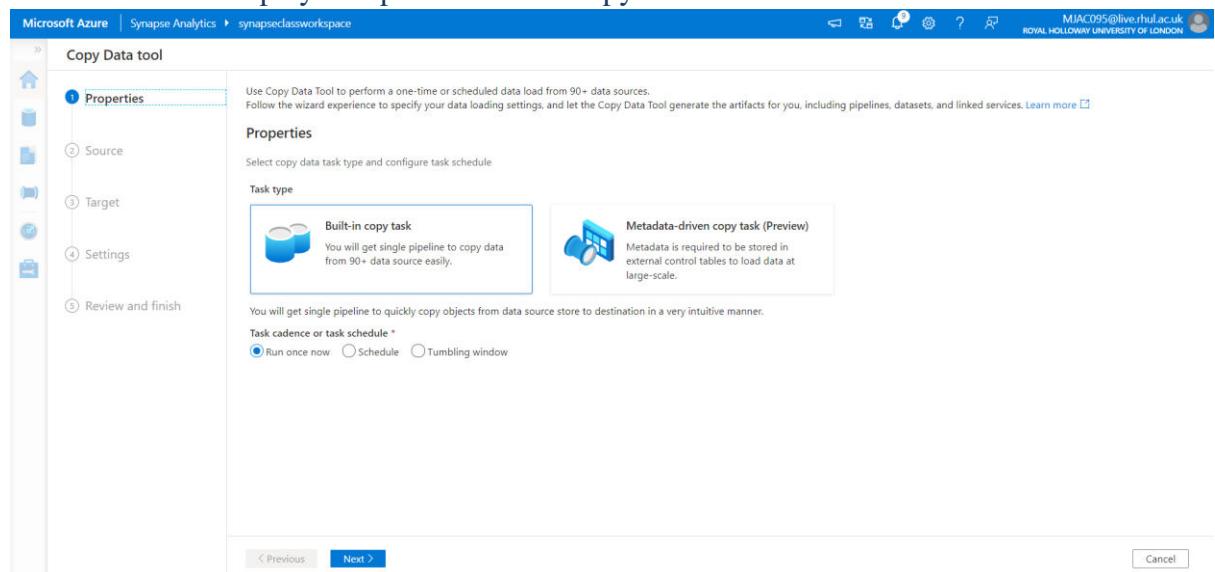
Files: Output text files for this Problem in Solution1 Folder.

Problem Statement 2: Ingest data from multiple Tables in SQL Server database to Azure Synapse Analytics

Solution :

- Created a Resource group, named as: Project-Arthi-rg
- Created a Blob Storage Account, named as: projectstorageacc01
- Created Azure Synapse Workspace named as: synapseclassworkspace
- In the Synapse Workspace, click Ingest:
 - Copy Data Tool is Displayed
 - Chose Built-in copy Task

Screenshot 2.1: Displays Properties Tab of Copy Data Tool



- In the Source Tab,
 - Choose Source Type: Azure SQL Database
 - Choose Connection: AzureSqlDatabase
 - Source Tables: chose existing Tables,
 - entered “SalesLT”
 - Select check box SelectAll

Screenshot 2.2.a: Displays Source Tab of Copy Data Tool

Source data store

Specify the source data store for the copy task. You can use an existing data store connection or specify a new data store.

Source type: Azure SQL Database

Connection: AzureSqlDatabase

Integration runtime: AutoResolveIntegrationRuntime

Source tables: Existing tables

AddressID	AddressLine1	AddressLine2	City	StateProvince	CountryRegion	PostalCode	rowguid
9	8713 Yosemite Ct.		Bothell	Washington	United States	98011	268af621-76d7-4c78-9441-144fd139821a
11	1318 Lasalle Street		Bothell	Washington	United States	98011	981b3303-ac3a-49c7-9f96-fb670785b269
25	9178 Jumping St.		Dallas	Texas	United States	75201	c8df3bd9-48f0-4654-a8dd-14a67a84d3c6

Screenshot 2.2.b.: Displays Preview Data for all the Tables of SalesLT

Preview data

Linked service: AzureSqlDatabase

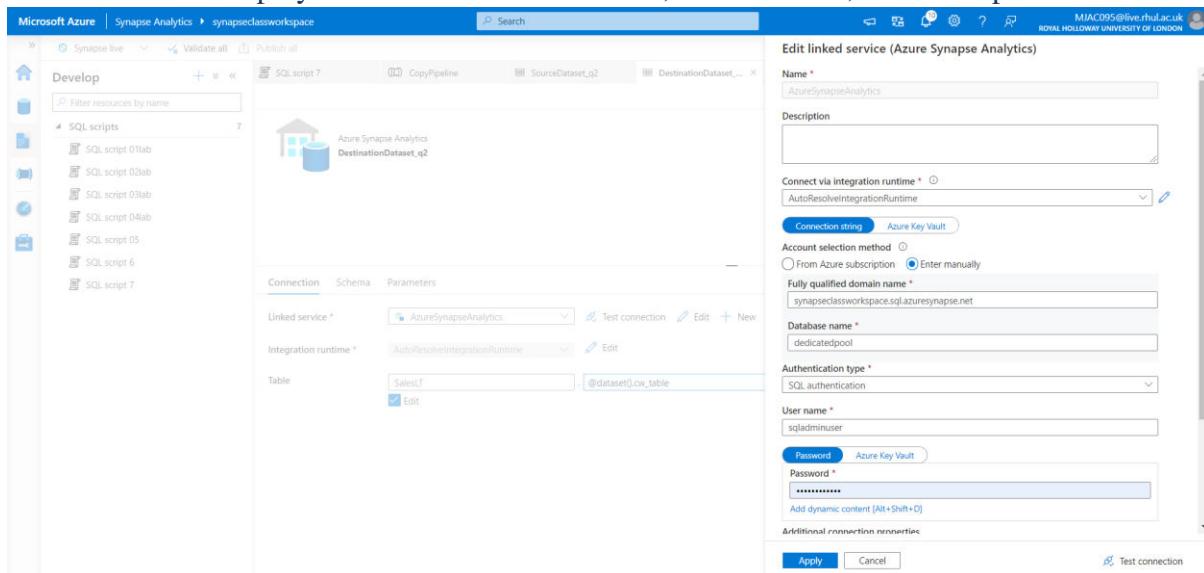
Object: SalesLT.Address

AddressID	AddressLine1	AddressLine2	City	StateProvince	CountryRegion	PostalCode	rowguid
9	8713 Yosemite Ct.		Bothell	Washington	United States	98011	268af621-76d7-4c78-9441-144fd139821a
11	1318 Lasalle Street		Bothell	Washington	United States	98011	981b3303-ac3a-49c7-9f96-fb670785b269
25	9178 Jumping St.		Dallas	Texas	United States	75201	c8df3bd9-48f0-4654-a8dd-14a67a84d3c6

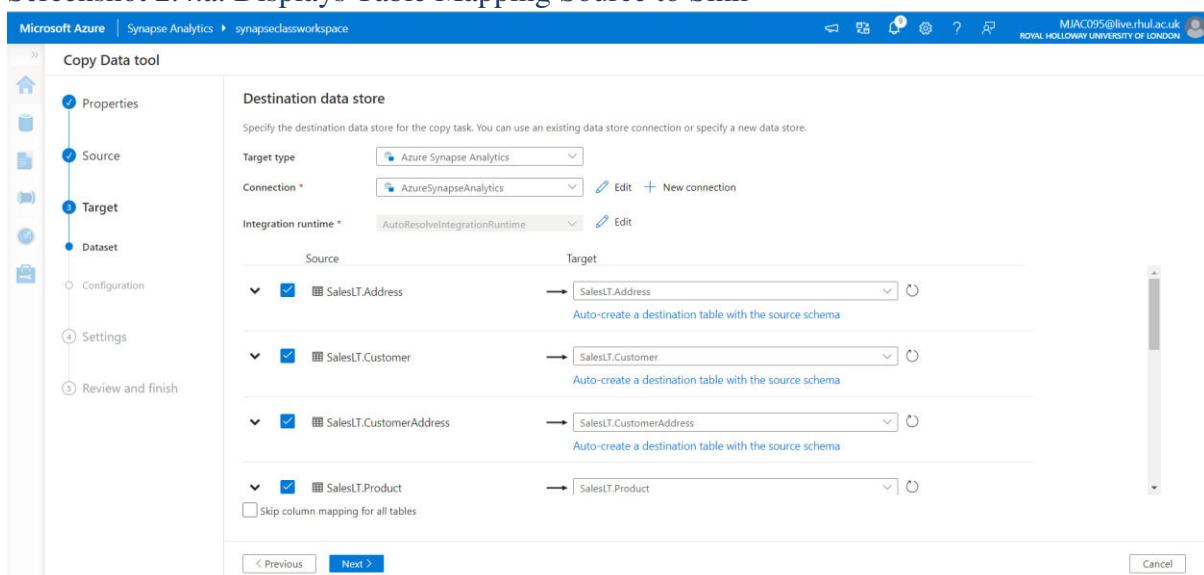
- In the Target Tab,
 - Choose Source Type: Azure Synapse Analytics
 - Choose Connection: AzureSynapseAnalytics
 - Choose server name: synapseclassworkspace.sql.azuresynapse.net
 - Database: dedicatedpool

- In the Destination Data Store:
 - Table mapping should be done,
 - Column mapping has to be done,
 - Deselect Type conversion.

Screenshot 2.3: Displays Sink Data Linked Service, Server Name, Dedicatedpool



Screenshot 2.4.a: Displays Table Mapping Source to Sink



Screenshot 2.4.b: Displays Column Mapping Source to Sink

The screenshot shows the 'Column mapping' step of the Copy Data tool. On the left, a navigation pane lists steps: Properties, Source, Target, Dataset, Configuration, Settings, and Review and finish. The 'Target' step is selected. The main area displays 'Table mappings (10)' for SalesLT.Address, SalesLT.Customer, SalesLT.CustomerAddress, SalesLT.Product, SalesLT.ProductCategory, and SalesLT.ProductCategory. Below this, 'Column mappings' are listed with columns for Source, Type, Destination, and Type. The mappings show AddressID, AddressLine1, AddressLine2, City, StateProvince, CountryRegion, and PostalCode being mapped to their respective columns in the Azure Synapse Analytics sink.

Source	Type	Destination	Type
AddressID	int	AddressID	int
AddressLine1	nvarchar	AddressLine1	nvarchar
AddressLine2	nvarchar	AddressLine2	nvarchar
City	nvarchar	City	nvarchar
StateProvince	nvarchar	StateProvince	nvarchar
CountryRegion	nvarchar	CountryRegion	nvarchar
PostalCode	nvarchar	PostalCode	nvarchar

- The storage is used for staging the data before it loads into Azure Synapse Analytics by using PolyBase.
- After the copy is complete, the temporary data in Azure Blob Storage is automatically cleaned up.

Screenshot 2.5: Displays Settings Page, Staging, Linked Service, PolyBase setting

The screenshot shows the 'Settings' step of the Copy Data tool. The left navigation pane shows 'Properties', 'Source', 'Target', 'Settings', and 'Review and finish'. The 'Settings' step is selected. The main area contains various configuration options under 'Settings'. Under 'Staging settings', 'Enable staging' is checked. Under 'Storage Path', 'Allow PolyBase' is checked. Under 'Copy method', 'PolyBase' is selected. Other options like 'Copy command' and 'Bulk insert' are available but not selected.

Screenshot 2.6: Displays Summary Page

Summary

You are running pipeline to copy data from Azure SQL Database to Azure Synapse Analytics.

Properties

Task name: CopyPipeline

Task description:

Source

Connection name: AzureSQLDatabase
Dataset name: SourceDatasetq2

Table name: items

Target

Connection name: AzureSynapseAnalytics
Dataset name: DestinationDatasetq2

Table name: items

Copy settings

Timeout: 7.00:00:00
Retry: 0
Retry interval: 30
Secure output: false

Save | Cancel

Cancel

- Now, we have created the required datasets and pipeline settings,

Screenshot 2.7: Displays Summary Page

Copy Data tool

Properties
Source
Target
Settings
Review and finish
Review
Deployment

Azure SQL Database → Azure Blob Storage → Azure Synapse Analytics

Deployment complete

Validate copy runtime environment ✓

Deployment step Status

- > Creating datasets Succeeded ✓
- > Creating pipelines Succeeded ✓
- > Running pipelines Succeeded ✓

Datasets and pipelines have been created. You can now monitor and edit the copy pipelines or click finish to close Copy Data Tool.

Finish | Edit pipeline | Monitor

Screenshot 2.8: Displays Pipeline Monitoring

The screenshot shows the Microsoft Azure Synapse Analytics Pipeline runs monitoring interface. The pipeline name is 'CopyPipeline'. One run was triggered manually and completed successfully. The run details are as follows:

Pipeline name	Run start	Run end	Duration	Triggered by	Status	Error	Run
CopyPipeline	10/12/21, 10:36:06 AM	10/12/21, 10:36:38 AM	00:00:31	Manual trigger	Succeeded		Original

- In the below Screenshot below, we could see all the Tables are copied into the Tables,

Screenshot 2.9: Displays Pipeline Debug Details

The screenshot shows the Microsoft Azure Synapse Analytics Pipeline debug details for the 'CopyPipeline' activity. There are 11 activity runs listed, all of which were successful (Status: Succeeded). The activity types include 'Copy data' and 'ForEach'. The table below summarizes the activity runs:

Activity name	Activity type	Run start	Duration	Status	Error	Log	Integration runtime	User proper...	Destination	Source	Run
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:19	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.Address	SalesLT.Address	c70c	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:24	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.SalesOrderHeader	SalesLT.SalesOrderHeader	d31	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:24	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.SalesOrderDetail	SalesLT.SalesOrderDetail	044f	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:25	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.ProductDescription	SalesLT.ProductDescription	f468	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:23	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.Customer	SalesLT.Customer	28e1	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:19	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.Product	SalesLT.Product	eb3	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:27	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.CustomerAddress	SalesLT.CustomerAddress	a0f1	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:23	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.ProductCategory	SalesLT.ProductCategory	2074	
Copy_sm0	Copy data	10/12/21, 10:36:10 AM	00:00:25	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.ProductModelPrc	SalesLT.ProductModelPrc	5d8c	
ForEach_sm0	ForEach	10/12/21, 10:36:09 AM	00:00:30	Succeeded			DefaultIntegrationRuntime (UK South)DefaultInteg	SalesLT.ProductModel	SalesLT.ProductModel	334c	

- Output verified: Using select query in any one of the tables and view the records

Screenshot 2.10: Output Verification

The screenshot shows the Microsoft Azure Synapse Analytics workspace interface. The left sidebar displays the 'Data' section with 'Workspace' and 'Linked' options, and a 'Databases' tree view containing several tables under the 'dedicatedpool (SQL)' database. The main area shows a 'SQL script 7' tab with the following query:

```
1 select * from [SalesLT].[Address]
```

The results are displayed in a table format with the following columns: AddressID, AddressLine1, AddressLine2, City, StateProvince, CountryRegion, PostalCode, rowguid, and ModifiedDate. The table contains 10 rows of data, each representing an address entry from the SalesLT.Address table. The data includes various addresses such as '8713 Yosemite Ct.' and 'Ward Parkway Center' across different cities like Bothell, Kansas City, and Downey.

AddressID	AddressLine1	AddressLine2	City	StateProvince	CountryRegion	PostalCode	rowguid	ModifiedDate
9	8713 Yosemite Ct.	(NULL)	Bothell	Washington	United States	98011	268af621-76d7...	2006-07-01T00...
634	Ward Parkway Center	(NULL)	Kansas City	Missouri	United States	64106	3f2aa425-7d02...	2006-09-01T00...
1042	Stonewood Mall	(NULL)	Downey	California	United States	90241	d1d79211-aec3...	2005-08-01T00...
634	Ward Parkway Center	(NULL)	Kansas City	Missouri	United States	64106	3f2aa425-7d02...	2006-09-01T00...
1042	Stonewood Mall	(NULL)	Downey	California	United States	90241	d1d79211-aec3...	2005-08-01T00...
9	8713 Yosemite Ct.	(NULL)	Bothell	Washington	United States	98011	268af621-76d7...	2006-07-01T00...
634	Ward Parkway Center	(NULL)	Kansas City	Missouri	United States	64106	3f2aa425-7d02...	2006-09-01T00...
1042	Stonewood Mall	(NULL)	Downey	California	United States	90241	d1d79211-aec3...	2005-08-01T00...
9	8713 Yosemite Ct.	(NULL)	Bothell	Washington	United States	98011	268af621-76d7...	2006-07-01T00...
634	Ward Parkway Center	(NULL)	Kansas City	Missouri	United States	64106	3f2aa425-7d02...	2006-09-01T00...
1042	Stonewood Mall	(NULL)	Downey	California	United States	90241	d1d79211-aec3...	2005-08-01T00...
9	8713 Yosemite Ct.	(NULL)	Bothell	Washington	United States	98011	268af621-76d7...	2006-07-01T00...
634	Ward Parkway Center	(NULL)	Kansas City	Missouri	United States	64106	3f2aa425-7d02...	2006-09-01T00...

At the bottom of the results pane, a message indicates: '000000 Query executed successfully.'

Problem Statement 3: Load data from Azure Blob storage to a database in Azure SQL Database by using Azure Data Factory

Solution:

- Created a Resource group, named as: Project-Arthi-rg
- Created a Blob Storage Account, named as: projectstorageacc01
- Created a data factory, named as: Project-Arthi-adf-01
- Created a self-hosted integration runtime named as: IR-1

as shown in Screenshot1.1.

- Created a Blob Storage and SQL table as **Source**

Created a Outlet.txt file and uploaded in **myblob – container / input – folder / Outlet.txt**

TextFile :

Outlet.txt

```
outlet_name,address_street,address_city
Sklep Awziel,High Street,Rye
Mentfield,Crescent,Brooklyn
Natural Valley,kingsley Road,Pittsfield
Red Outlet,Senator,Brooklyn
Ads Nawziel,Mount Street,Woodside
```

Screenshot 3.1: Displays Text File in Blob Storage inside Container

The screenshot shows the Azure Storage Explorer interface. On the left, there's a sidebar with various Azure services like Overview, Activity log, Tags, etc. The main area shows a tree view of 'BLOB CONTAINERS' with 'myblob' selected. Inside 'myblob', there's a single file named 'outlet.txt'. A table below the file list provides detailed statistics: NAME (outlet.txt), ACCESS TIER (Hot inferred), ACCESS TIER LAST MODIFIED (10/4/2021, 7:19:34 PM), LAST MODIFIED (10/4/2021, 7:19:34 PM), BLOB TYPE (Block Blob), CONTENT TYPE (text/plain), SIZE (203 B), STATUS (Active), REMAINING DAYS (not applicable), DELETED TIME (not applicable), and LEASE STATE (not applicable). The status bar at the bottom says 'Showing 1 to 1 of 1 cached items'.

- Created SQL table as **Sink**, with the following SQL script: **dbo.outlets**

Query: **dbo.outlets**

```
CREATE TABLE dbo.outlets(
    ID int IDENTITY(1,1) NOT NULL,
    outlet_name varchar(50) NOT NULL,
    address_street varchar(50),
    address_city varchar(50),
    primary key(ID));
```

GO

```
CREATE CLUSTERED INDEX IX_outlet_ID ON dbo.outlets (ID);
```

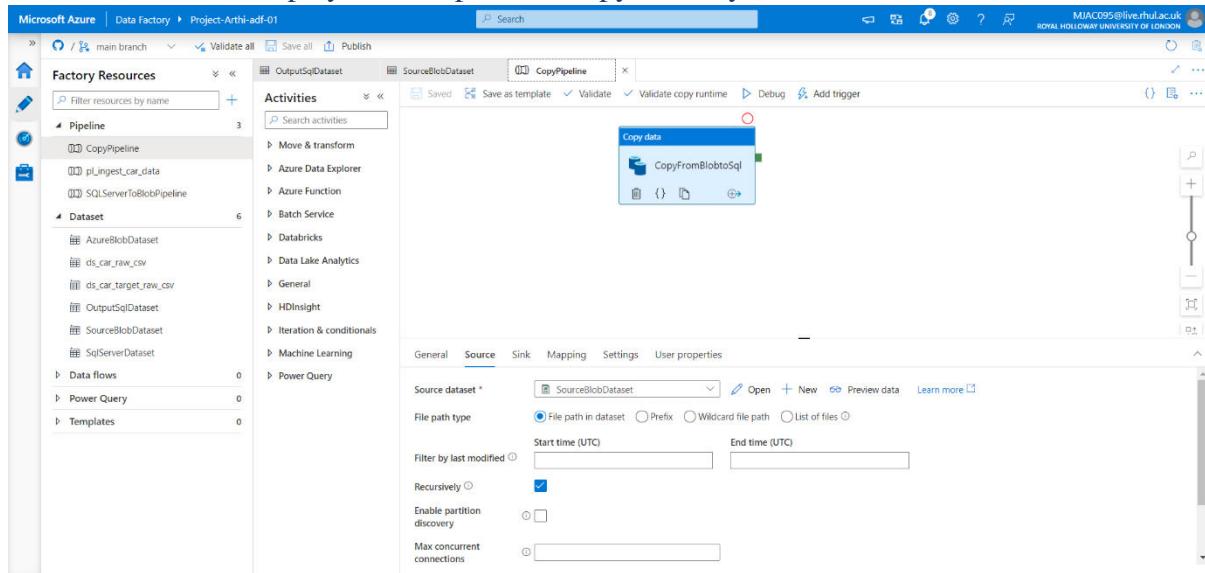
- In case of if any attribute is given unique, it creates integration violation, so we drop and truncate table and create a new one using above SQL script.

```
DROP TABLE dbo.outlet1
```

```
TRUNCATE TABLE dbo.outlet1
```

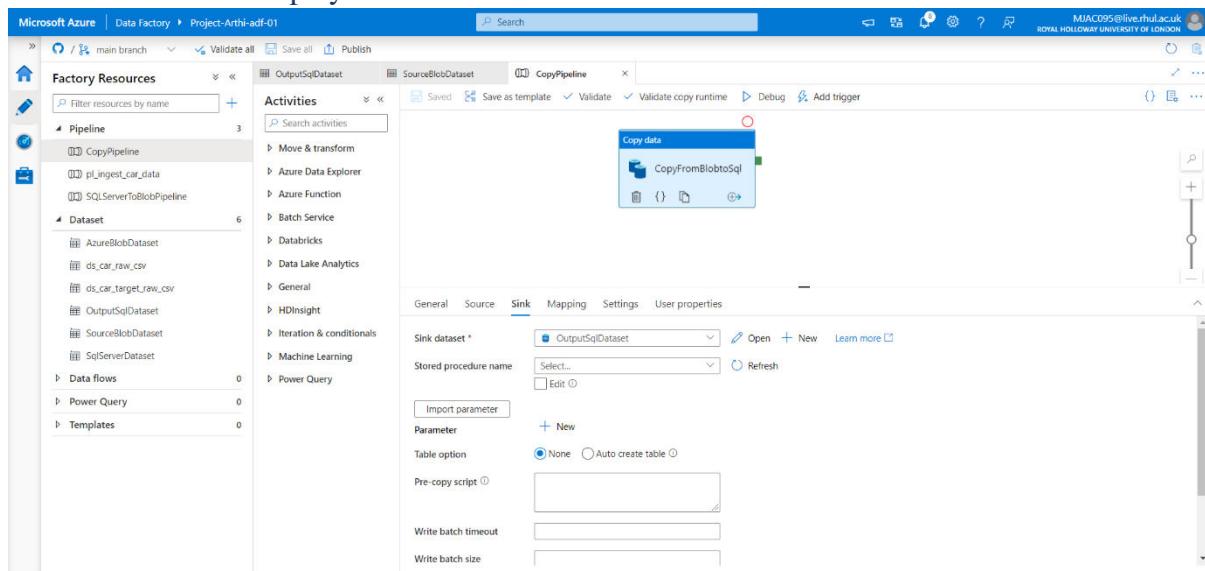
- Created a pipeline named CopyPipeline with a copy activity “ CopyFromBlobToSql” to move the data from blob storage to Azure SQL database. Then set up Source dataset: SourceBlobDataset with linked service AzureStorageLinkedService

Screenshot 3.2.a: Displays both Pipelines, Copy Activity and Source Dataset



- Sink dataset as: OutputSqlDataset (Azure SQL database) , with linked service AzureSqlDatabaseLinkedService

Screenshot 3.2.b: Displays Sink Dataset



- Started a pipeline run, debugged, pipeline validation done. Published All

Screenshot 3.3.a: Pipeline Activity Detail

The screenshot shows the 'Details' view for a pipeline activity. The activity is a 'Copy' operation named 'Copy_d1'. It has a single step: 'Azure Blob Storage' to 'Azure SQL Database'. The status is 'Succeeded'. Key metrics shown include:

- Azure Blob Storage:**
 - Data read: 203 bytes
 - Files read: 1
 - Rows read: 5
 - Peak connections: 1
- Azure SQL Database:**
 - Data written: 288 bytes
 - Rows written: 5
 - Peak connections: 2
- Copy duration:** 00:00:11
- Throughput:** 18.432 bytes/s

Under the 'Copy duration' section, it details the process flow:

- Azure Blob Storage → Azure SQL Database:**
 - Start time:** 10/4/21, 7:51:59 PM
 - Used parallel copies:** 1
 - Duration:** 00:00:11
- Queue:** Listing source (00:00:00), Reading from source (00:00:00), Writing to sink (00:00:00)
- Transfer:** 00:00:06

At the bottom, it says 'Data consistency verification' is 'Not verified'.

Screenshot 3.3.b: Pipeline Debug Detail

The screenshot shows the 'Pipeline runs' page. The sidebar navigation includes 'Runs', 'Pipeline runs', 'Trigger runs', 'Runtimes & sessions', 'Integration runtimes', 'Data flow debug', 'Notifications', and 'Alerts & metrics'. The main area displays a table of pipeline runs:

Pipeline name	Run start	Run end	Duration	Status	Triggered by	Error	Run ID
CopyPipeline	10/5/21, 8:28:57 AM	10/5/21, 8:29:21 AM	00:00:24	Succeeded	Manual trigger		3d5cc3fb-b5cb-4713-abef-1...
CopyPipeline	10/4/21, 8:44:15 PM	10/4/21, 8:44:36 PM	00:00:21	Succeeded	Manual trigger		9bf32f6-21e3-4fa4-b3f7-6...
CopyPipeline	10/4/21, 8:39:01 PM	10/4/21, 8:39:13 PM	00:00:12	Succeeded	Manual trigger		14251a9b-9499-4857-9127...
CopyPipeline	10/4/21, 8:29:06 PM	10/4/21, 8:29:18 PM	00:00:12	Succeeded	Manual trigger		4d1dc229-d880-475b-b7e1...
CopyPipeline	10/4/21, 8:24:41 PM	10/4/21, 8:25:00 PM	00:00:19	Succeeded	Manual trigger		faa1aa08-c00f-4a33-83c6-0...
SQLServerToBlobPipeline	10/4/21, 8:18:56 PM	10/4/21, 8:19:07 PM	00:00:11	Succeeded	Manual trigger		d1c35718-0449-4112-a1a5...
CopyPipeline	10/4/21, 7:59:23 PM	10/4/21, 7:59:41 PM	00:00:18	Failed	Manual trigger		9ba21bfff-a78c-4621-93ce-cb...
CopyPipeline	10/4/21, 7:51:56 PM	10/4/21, 7:52:12 PM	00:00:15	Succeeded	Manual trigger		8d303e49-0003-4090-0754...

- Added trigger by clicking on trigger now
- Monitoring the relative pipeline CopyPipeline

Screenshot 3.3.c: Pipeline Manual Trigger detail

The screenshot shows the 'Pipeline runs' page in Microsoft Azure Data Factory. The left sidebar includes 'Runs', 'Pipeline runs' (selected), 'Trigger runs', 'Runtimes & sessions', 'Integration runtimes', 'Data flow debug', 'Notifications', and 'Alerts & metrics'. The main area displays a table of pipeline runs:

Pipeline name	Run start	Run end	Duration	Triggered by	Status	Error	Run	Parameters	Annotat
CopyPipeline	10/5/21, 8:32:11 AM	10/5/21, 8:32:29 AM	00:00:17	Manual trigger	Succeeded		Original		
CopyPipeline	10/4/21, 8:45:06 PM	10/4/21, 8:45:16 PM	00:00:10	Manual trigger	Succeeded		Original		
CopyPipeline	10/4/21, 8:39:51 PM	10/4/21, 8:40:14 PM	00:00:23	Manual trigger	Failed		Original		
CopyPipeline	10/4/21, 8:29:42 PM	10/4/21, 8:29:51 PM	00:00:09	Manual trigger	Failed		Original		
CopyPipeline	10/4/21, 8:25:39 PM	10/4/21, 8:25:53 PM	00:00:14	Manual trigger	Failed		Original		
CopyPipeline	10/4/21, 7:58:13 PM	10/4/21, 7:58:32 PM	00:00:19	Manual trigger	Failed		Original		

- Output verified: Text File copied from Azure blob storage to a Table format in Azure SQL database.
- Verified using SQL SCRIPT:

```
SELECT * FROM [dbo].[outlets]
```

- When the copy activity performed for first time, text file 5 rows copied into table as first row as header and next 5 rows as 5 records.
- When we triggered, text is copied into table, therefore next 5 records are created in the table with ID 6-10.

Screenshot 3.4: Text File copied from Azure Blob Storage to Azure SQL Database

The screenshot shows the Microsoft Azure SQL Database Query Editor interface. The top navigation bar includes 'Microsoft Azure', 'Search resources, services, and docs (G+)', and a user account 'MJA095@live.rhul.ac.uk ROYAL HOLLOWAY UNIVERSITY...'. The main title is 'Project-Arthi-sqldb (projectarthisqlserver/Project-Arthi-sqldb) | Query editor (preview)'.

The left sidebar displays a tree view of database objects: 'Tables' (selected), 'dbo.outlet', 'Views', and 'Stored Procedures'. The main area shows 'Query 1' with the following code:

```
1 select * from [dbo].[outlets]
```

The 'Results' tab is selected, showing the output of the query:

ID	outlet_name	address_street	address_city
1	Sklep Awziel	High Street	Rye
2	Mentfield	Crescent	Brooklyn
3	Natural Valley	kingsley Road	Pittsfield
4	Red Outlet	Senator	Brooklyn

A status message at the bottom indicates 'Query succeeded | 0s'.

Screenshot 3.5: Output Verification

This screenshot is identical to Screenshot 3.4, showing the Microsoft Azure SQL Database Query Editor interface. The top navigation bar, sidebar object tree, and the contents of 'Query 1' are the same. The 'Results' tab displays the same data from the 'outlets' table:

ID	outlet_name	address_street	address_city
1	Sklep Awziel	High Street	Rye
2	Mentfield	Crescent	Brooklyn
3	Natural Valley	kingsley Road	Pittsfield
4	Red Outlet	Senator	Brooklyn

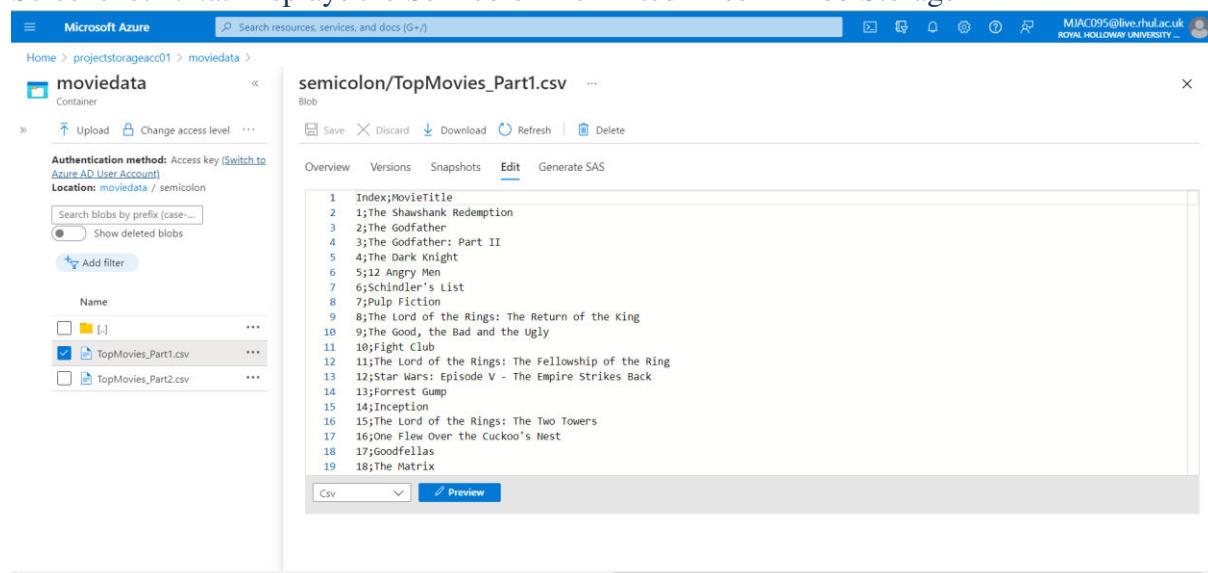
A status message at the bottom indicates 'Query succeeded | 0s'.

Problem Statement 4: How to Load Multiple Files in Parallel in Azure Data Factory

Solution :

- Created a Resource group, named as: Project-Arthi-rg
- Created a Blob Storage Account, named as: projectstorageacc01
- Created a data factory, named as: Project-Arthi-adf-01
- Created a self-hosted integration runtime named as: IR-1 as shown in Screenshot 1.
- My Source file contains 3 Top movie name files, 2 semicolon Delimited with 100 values, 1 comma delimited with 50 values.
- These files uploaded in the Blob storage account
 - projectstorageacc01
 - /Container: moviedata
 - /Folder: semicolon (2 files)
 - comma (1 files)

Screenshot 4.1.a: Displays the Semicolon Delimited Files in Blob Storage



The screenshot shows the Microsoft Azure Blob Storage interface. The left sidebar shows the 'moviedata' container under 'projectstorageacc01'. The main area displays three files in the 'semicolon' folder:

- TopMovies_Part1.csv**: A semicolon-delimited file containing 100 movie entries, starting with "Index;MovieTitle" and ending with "18;The Matrix".
- TopMovies_Part2.csv**: A semicolon-delimited file containing 50 movie entries, starting with "1;The Shawshank Redemption" and ending with "25;The Godfather".
- TopMovies_Part3.csv**: A comma-delimited file containing 50 movie entries, starting with "1;The Godfather" and ending with "25;The Shawshank Redemption".

The 'Edit' tab is selected for the 'TopMovies_Part1.csv' file, showing its contents in a preview pane.

Screenshot 4.1.b: Displays the Comma Delimited Files in Blob Storage

The screenshot shows the Microsoft Azure Blob Storage interface. On the left, there's a sidebar for the 'moviedata' container with options like Overview, Diagnosis and solve problems, Access Control (IAM), Properties, and Metadata. The main area shows a list of blobs in the 'comma' folder. One blob, 'TopMovies_Part3.csv', is selected and its preview is displayed. The preview shows a CSV file with the following data:

	Index	MovieTitle
1	281	12 Years a Slave
2	282	Ben-Hur
3	283	Persona
4	284	The Grand Budapest Hotel
5	285	Million Dollar Baby
6	286	Amores Perros
7	287	The Message
8	288	The Princess Bride
9	289	Jurassic Park
10	290	Hachi: A Dog's Tale
11	291	Nausicaä of the Valley of the Wind
12	292	Udaan
13	293	Memories of Murder
14	294	Stalker
15	295	Touch of Evil
16	296	The Grapes of Wrath
17	297	The Truman Show
18	298	Before Sunrise

- Metadata for these files are stored in the
 - Azure SQL Database named: Project-Arthi-sqldb,
 - Azure SQL Server named as: projectarthisqlserver
 - Schema for the dbo.Metadata_ADF is stored in a table as key-value pair.
 - Insert queries of the Metadat_ADF, and Destination Table queries is attached with project in the Solution 4 folder.
- Created 2 datasets. One for blob storage and one for SQL Server.
- Source Dataset named as: ds_movies_dynamic_csv
 - Created Linked Service: ls_movies_dynamic_q4
 - Need to add Parameters as FolderName and Delimiter symbol
 - Set the file path:
 - movidata/@dataset()Foldername/File
 - Set the Column delimiter:
 - check edit check box,
 - set: @dataset().DelimiterSymbol

Screenshot 4.2.a: Displays the Source Dataset

The screenshot shows the 'Factory Resources' blade in Microsoft Azure Data Factory. A dataset named 'ds_movies_dynamic_csv' is selected. The 'Connection' tab is active, showing the following details:

- Linked service:** ls_movies_dynamic_q4 (selected)
- Integration runtime:** IR-1
- File path:** moviedata/@dataset().FolderName/file (with a dropdown for FileName)
- Compression type:** None
- Column delimiter:** @dataset().DelimiterSymbol
- Row delimiter:** Default (\r\n, or \n\r)
- Encoding:** Default(UTF-8)
- Escape character:** Backslash (\)
- Quote character:** Double quote (")
- First row as header:** checked

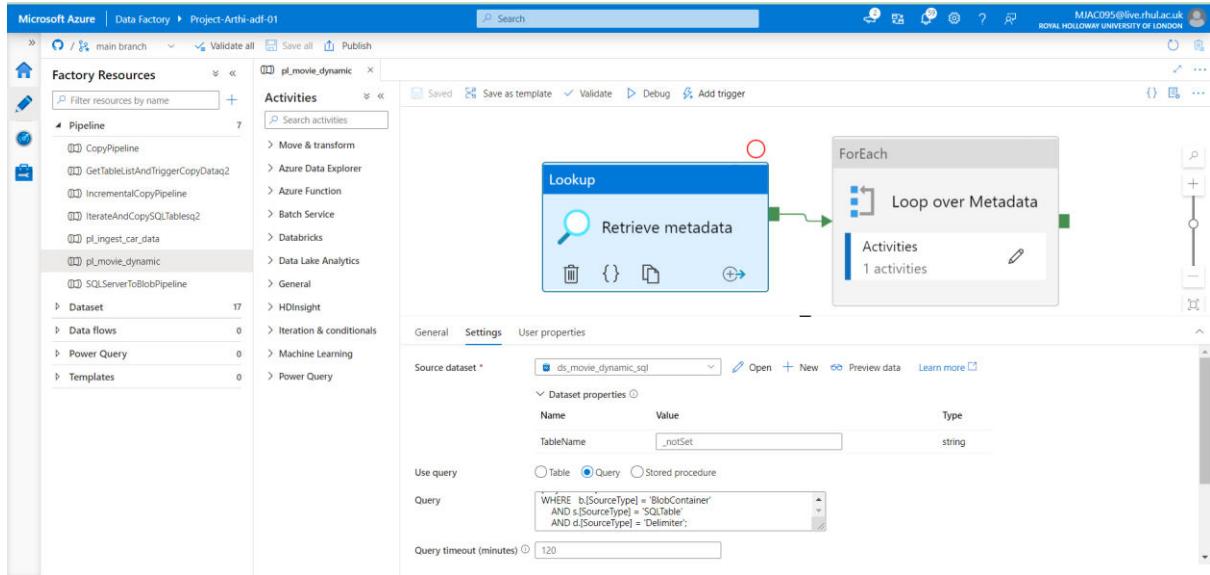
- Sink Dataset named as: ds_movies_dynamic_sql
 - Created linked Service: SQLtest
 - Need to add Parameters as TableName
 - Set theTable:
 - dbo/@dataset()TableName

Screenshot 4.2.b: Displays the Sink Dataset

The screenshot shows the 'Factory Resources' blade in Microsoft Azure Data Factory. A dataset named 'ds_movies_dynamic_sql' is selected. The 'Connection' tab is active, showing the following details:

- Linked service:** sqltest (selected)
- Integration runtime:** IR-1
- Table:** dbo/@dataset().TableName (with a dropdown for TableName)
- Edit:** checkbox checked

Screenshot 4.3.a: Displays the Pipeline, Lookup Activity Details



- We connect to the source dataset, set tablename as “_notSet”
- We Use query and deselect the First row only checkbox
- With the following query, we can retrieve the metadata from SQL Server:

Query:

```

SELECT
    b.[ObjectName]
    ,FolderName = b.[ObjectValue]
    ,SQLTable = s.[ObjectValue]
    ,Delimiter = d.[ObjectValue]
FROM [dbo].[Metadata_ADF] b
JOIN [dbo].[Metadata_ADF] s ON b.[ObjectName] = s.[ObjectName]
JOIN [dbo].[Metadata_ADF] d ON b.[ObjectName] = d.[ObjectName]
WHERE b.[SourceType] = 'BlobContainer'
    AND s.[SourceType] = 'SQLTable'
    AND d.[SourceType] = 'Delimiter';

```

Screenshot 4.3.b: Displays the Preview Data of the Lookup Activity

ObjectName	FolderName	SQLTable	Delimiter
Databricks	semicolon	topmovies_semicolon	;
Data Lake A	comma	topmovies_comma	,

- Next we add ForEach Activity:
 - In Setting tab: set items as : “ @activity('Retrieve Metadata').output.value ” as dynamic content
 - In Activities tab : Add Copy Activity : Copy Blob to SQL
 - Set Source dataset : ds_movies_dynamic_csv
 - Fill the dataset Properties as @(item().FolderName),
@(item().Delimiter)
 - Choose WildcardPath, set wildcard filename as “ *.csv”
 - Set Sink dataset: ds_movies_dynamic_sql
 - Fill the dataset Properties as @(item().TableName)
 - Table Option: None
 - Pre-Copy script:

`TRUNCATE TABLE dbo.@{item().SQLTable}`

Screenshot 4.3.c: Displays the Source Dataset Copy Activity within ForEach Activity

The screenshot shows the Microsoft Azure Data Factory Studio interface. A pipeline named 'pf_movie_dynamic' is open, showing a 'Loop over Metadata' activity. Inside this loop, there is a 'Copy data' activity with the source set to 'Copy Blob to SQL'. The 'Source' tab is selected, showing the configuration for the source dataset:

- Source dataset:** ds_movies_dynamic_csv
- Dataset properties:**

Name	Value	Type
FolderName	@{item().FolderName}	string
DelimiterSymbol	@{item().Delimiter}	string
- File path type:** Wildcard file path
- Wildcard paths:** moviedata/*.*csv
- Filter by last modified:** Start time (UTC) and End time (UTC) fields are empty.
- Recursively:** Checked
- Enable partition discovery:** Unchecked

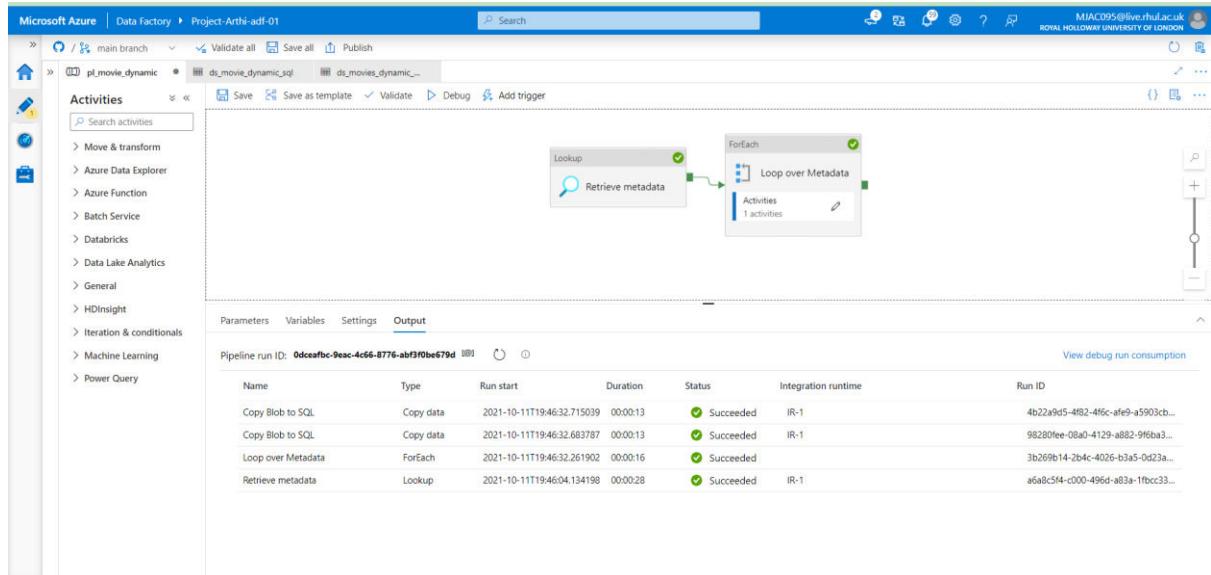
Screenshot 4.3.d: Displays the Sink Dataset Copy Activity within ForEach Activity

The screenshot shows the Microsoft Azure Data Factory Studio interface. The same pipeline and loop structure are visible. The 'Sink' tab is selected for the 'Copy data' activity, showing the configuration for the sink dataset:

- Sink dataset:** ds_movie_dynamic_sql
- Dataset properties:**

Name	Value	Type
TableName	@{item().SQLTable}	string
- Stored procedure name:** Select... (dropdown menu)
- Import parameter:** New
- Parameter:** None (radio button selected)
- Table option:** Auto create table (radio button)
- Pre-copy script:** TRUNCATE TABLE dbo.@{item().SQLTa...}
- Write batch timeout:** (empty input field)

Screenshot 4.4: Pipeline Debug Details



- Output verified: Using SQL Query in the Azure SQL Database, checking the total number of rows in each table and viewing the records.

Screenshot 4.5.a: Output Verification I

```

1 select count(*) from [dbo].[topmovies_comma]
2

```

Results

50

Query succeeded | 0s

Screenshot 4.5.b: Output Verification II

The screenshot shows the Microsoft Azure portal interface for a SQL database named "Project-Arthi-sqldb". The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The main area displays a query editor with the following details:

- Query 4** is selected.
- The query entered is: `select * from [dbo].[topmovies_comma]`.
- The results pane shows the following data:

Index	MovieTitle
201	12 Years a Slave
202	Ben-Hur
203	Persona
204	The Grand Budapest Hotel

- A message at the bottom of the results pane says "Query succeeded | 0s".

Screenshot 4.5.c: Output Verification III

The screenshot shows the Microsoft Azure portal interface for the same SQL database "Project-Arthi-sqldb". The left sidebar is identical to the previous screenshot. The main area displays a query editor with the following details:

- Query 2** is selected.
- The query entered is: `select count(*) from [dbo].[topmovies_semicolon]`.
- The results pane shows the following data:

Count
200

- A message at the bottom of the results pane says "Query succeeded | 0s".

Screenshot 4.5.d: Output Verification IV

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'Project-Arathi-sqldb'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and several preview features like Power Platform, Power BI, Power Apps, and Power Automate. The main area is titled 'Query editor (preview)' and displays a list of queries (Query 1 to Query 6). The current query, 'Query 5', is selected and contains the following SQL code:

```
1 select * from [dbo].[topmovies_semicolon]
```

The results pane shows the output of the query:

Index	MovieTitle
101	L.A. Confidential
102	For a Few Dollars More
103	3 Idiots
104	Rashomon

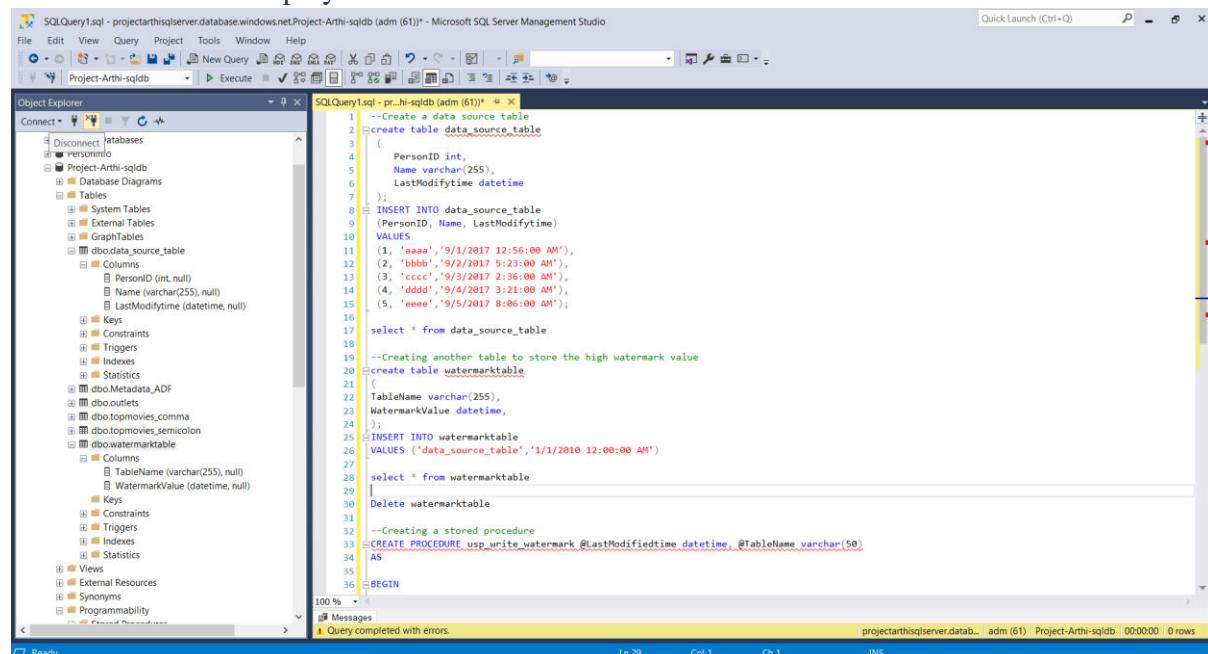
A message at the bottom of the results pane states 'Query succeeded | 0s'.

Problem Statement 5: Incrementally load data from Azure SQL Database to Azure Blob storage using the Azure portal

Solution:

- Created a Resource group, named as: Project-Arthi-rg
- Created a data factory, named as: Project-Arthi-adf-01
- Created a self-hosted integration runtime named as: IR-1, as shown in Screenshot1.1.
- Created 2 tables named **data_source_table** as the data source store, **watermarktable** to store the watermark value and a stored procedure named **usp_write_watermark**.

Screenshot 5.1.a: Displays table creation in SSMS



```
--Create a data source table
CREATE TABLE data_source_table
(
    PersonID int,
    Name varchar(255),
    LastModifytime datetime
);

--INSERT INTO data_source_table
--(PersonID, Name, LastModifytime)
VALUES
(1, 'aaaa', '9/3/2017 12:56:00 AM'),
(2, 'bbbb', '9/3/2017 5:23:00 AM'),
(3, 'cccc', '9/3/2017 2:36:00 AM'),
(4, 'dddd', '9/4/2017 3:21:00 AM'),
(5, 'eeee', '9/5/2017 8:06:00 AM');

select * from data_source_table

--Creating another table to store the high watermark value
CREATE TABLE watermarktable
(
    TableName varchar(255),
    WatermarkValue datetime
);

--INSERT INTO watermarktable
VALUES ('data_source_table', '1/1/2010 12:00:00 AM');

select * from watermarktable

Delete watermarktable

--Creating a stored procedure
CREATE PROCEDURE usp_write_watermark @LastModifiedtime datetime, @TableName varchar(50)
AS
BEGIN
```

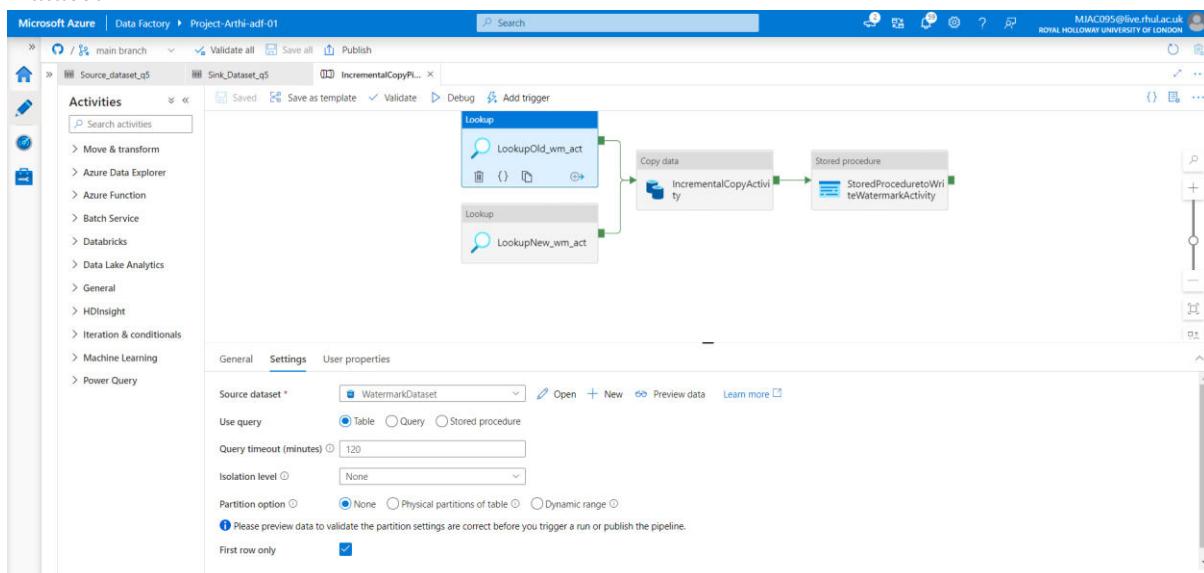
Screenshot 5.1.b: Displays Table Creation in SSMS:

The screenshot shows the Microsoft SQL Server Management Studio interface. The title bar reads "SQLQuery1.sql - projectarthisqldb.database.windows.net\Project-Arthi-sqldb (adm (61)) - Microsoft SQL Server Management Studio". The left pane is the Object Explorer, displaying a tree view of database objects including Tables, Keys, Constraints, Triggers, Indexes, Statistics, Views, External Resources, Synonyms, Programmability (Stored Procedures, System Stored Procedures, dbms_procs_watermark), Functions, Database Triggers, Assemblies, and Types (System Data Types, User-Defined Data Types, User-Defined Table Types, User-Defined Types, XML Schema Collections). The right pane is the SQL Query Editor, showing the following T-SQL script:

```
16 select * from data_source_table
17
18 --Creating another table to store the high watermark value
19 create table watermarktable
20 (
21     TableName varchar(255),
22     WatermarkValue datetime,
23 );
24
25 INSERT INTO watermarktable
26 VALUES ('data_source_table','1/1/2010 12:00:00 AM')
27
28 select * from watermarktable
29
30 Delete watermarktable
31
32 --Creating a stored procedure
33 CREATE PROCEDURE usp_write_watermark @LastModifiedtime datetime, @TableName varchar(50)
34 AS
35
36 BEGIN
37
38 UPDATE watermarktable
39 SET [WatermarkValue] = @LastModifiedtime
40 WHERE [TableName] = @TableName
41
42 END
```

- Created a pipeline named IncrementalCopyPipeline with a Lookup activity “LookupOld_wm_act” to hold the current watermark data from **watermarktable**. Then set up Source dataset: SourceDataset_q5 with linked service “ls_azsqldb_q5” with data_source_table.

Screenshot 5.2.a: Displays both pipeline, Old Water Mark LookUp Activity and Watermark Dataset



Screenshot 5.2.b: Displays verifying the Data Source Table in Azure SQL Database

The screenshot shows the Microsoft Azure Query editor interface. The top navigation bar includes 'Microsoft Azure', a search bar, and user information 'MUAC095@live.rhul.ac.uk ROYAL HOLLOWAY UNIVERSITY OF LONDON'. The main title is 'Project-Arthi-sqlDb (projectarthisqlserver/Project-Arthi-sqlDb) | Query editor (preview)'.

The left sidebar shows the database structure for 'Project-Arthi-sqlDb (adm)': Tables (dbo.data_source_table, dbo.Metadata_ADF, dbo.outlets, dbo.topmovies_comma, dbo.topmovies_semicolon, dbo.watermarktable), Views, and Stored Procedures.

The central area is titled 'Query 1' with the following SQL code:

```
1 select * from [dbo].[data_source_table]
```

The results pane displays the following data:

PersonID	Name	LastModifystime
1	aaaa	2017-09-01T00:56:00.0000000
2	bbbb	2017-09-02T05:23:00.0000000
3	cccc	2017-09-03T02:36:00.0000000
4	dddd	2017-09-04T03:21:00.0000000
5	eeee	2017-09-05T08:06:00.0000000

At the bottom, a message indicates 'Query succeeded | 0s'.

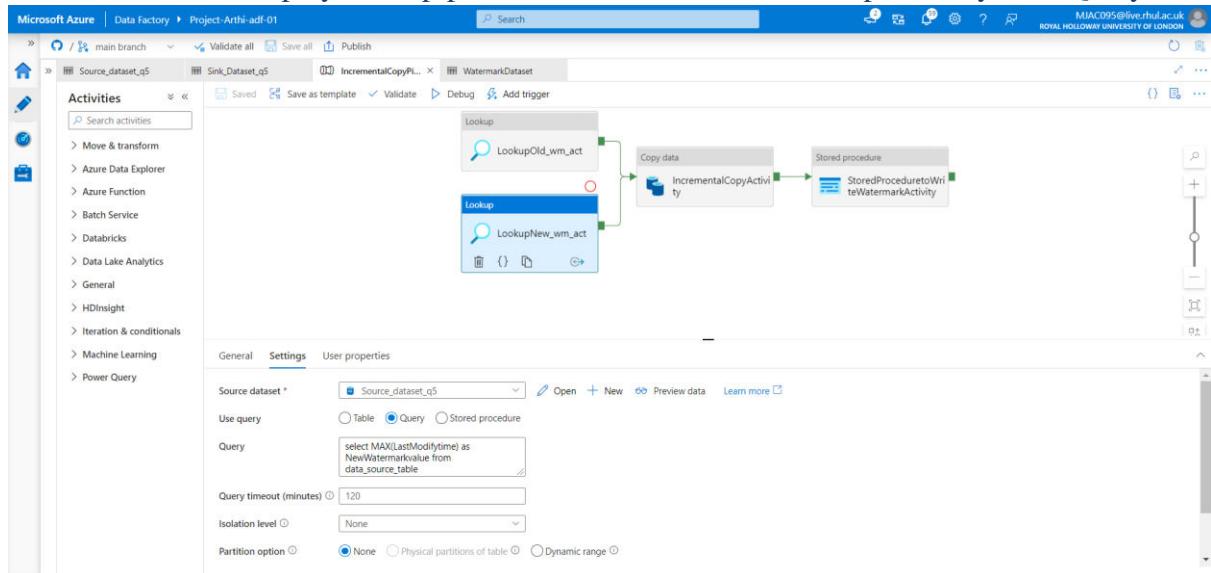
Screenshot 5.2.c: Displays Linked Services for Watermark Dataset

The screenshot shows the Microsoft Data Factory interface. The top navigation bar includes 'Microsoft Azure', a search bar, and user information 'MUAC095@live.rhul.ac.uk ROYAL HOLLOWAY UNIVERSITY OF LONDON'. The main title is 'Project-Arthi-adf-01'.

The left sidebar shows 'Factory Resources' under 'Dataset': asynapseSqlDWDSq2, AzureBlockDataset, AzureSqlDBDSq2, ds_car_raw_csv, ds_car_target_raw_csv, ds_movie_dynamic_sql, ds_movies_dynamic_csv, OutputSqlDataset, pbm8_ContactDetail, pbm8_ContractDetailType, Pbm8_Address, Pbm8_Person, SinkDataset5, SourceBlobDataset, SourceDataset5, SqlServerDataset, and watermarkDataset.

The right pane shows the configuration for the 'watermarkDataset' dataset. It has tabs for 'Connection', 'Schema', and 'Parameters'. Under 'Connection', it is set to 'Linked service': 'ls_q5_azsqlDb' (selected), 'Test connection' (successful), 'Integration runtime': 'IR-1', and 'Table': 'dbo.watermarktable'.

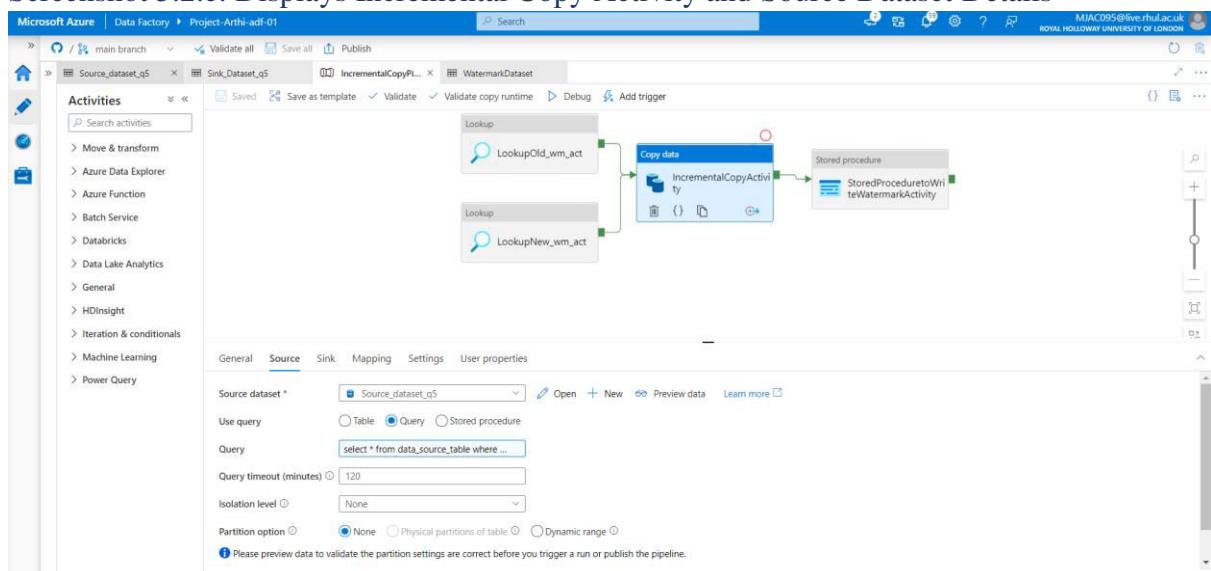
Screenshot 5.2.d: Displays both pipeline, New Water Mark LookUp Activity with Query



Query:

```
select MAX(LastModifytime) as NewWatermarkvalue from data_source_table
```

Screenshot 5.2.e: Displays Incremental Copy Activity and Source Dataset Details



Query:

```
select * from data_source_table where LastModifytime > '@{activity('LookupOldWaterMarkActivity').output.firstRow.WatermarkValue}' and LastModifytime <= '@{activity('LookupNewWaterMarkActivity').output.firstRow.NewWatermarkvalue}'
```

Screenshot 5.2.f: Displays Sink Dataset Details

- Azure Data Factory automatically creates the output folder **incrementalcopy** if it does not exist. In the Screenshot, we have browsed the **File path** to navigate to a folder in a **blob container**. In the File path field, add the dynamic content :

`@CONCAT ('Incremental-', pipeline ().RunId, '.txt')`

Screenshot 5.3.a: Pipeline Debug Details

Name	Type	Run start	Duration	Status	Integration runtime	Run ID
StoredProceduretoWriteWatermarkA	Stored procedure	2021-10-11T16:28:23.509768	00:00:05	Succeeded	IR-1	492a46cf-0fd-4b87-8a6d-15f609...
IncrementalCopyActivity	Copy data	2021-10-11T16:28:11.691503	00:00:10	Succeeded	IR-1	1466b22d-788-471f-84bd-3ce1ca...
LookupNew_wm_act	Lookup	2021-10-11T16:27:42.731615	00:00:14	Succeeded	IR-1	a63136d6-11b4-4c98-9a21-5a681...
LookupOld_wm_act	Lookup	2021-10-11T16:27:42.669115	00:00:28	Succeeded	IR-1	dac76ea2-e2fc-488d-ab70-f5c353a...

Screenshot 5.3.b: Pipeline Activity Details

The screenshot shows the 'Details' view for a pipeline activity. The activity is a 'Copy' operation from 'Azure SQL Database' to 'Azure Blob Storage'. The status is 'Succeeded'. Key metrics shown include:

- Azure SQL Database:**
 - Data read: 52 bytes
 - Rows read: 2
 - Peak connections: 1
- Azure Blob Storage:**
 - Data written: 112 bytes
 - Files written: 1
 - Rows written: 2
 - Peak connections: 1
- Copy duration:** 00:00:09
- Throughput:** 10.24 bytes/s
- Details:**
 - Queue:** Working duration 00:00:05, Total duration 00:00:05. Sub-tasks: Time to first byte 00:00:00, Reading from source 00:00:00, Writing to sink 00:00:00.
 - Transfer:** Working duration 00:00:01, Total duration 00:00:01. Sub-tasks: Time to first byte 00:00:00, Reading from source 00:00:00, Writing to sink 00:00:01.
- Data consistency verification:** Not verified.

At the bottom, there's a satisfaction survey and a link to 'View debug run consumption'.

- Added trigger by clicking on trigger now

Screenshot 5.3.c: Pipeline Debug Detail II

The screenshot shows the 'Pipeline runs' page. The left sidebar has a 'Runs' section with 'Pipeline runs' selected. The main area displays a table of recent runs:

Pipeline name	Run start	Run end	Duration	Status	Triggered by	Error	Run ID
IncrementalCopyPipeline	10/11/21, 5:27:41 PM	10/11/21, 5:28:29 PM	00:00:47	Succeeded	Manual trigger		a2ec67fb-ac05-4934-beef-0...
IncrementalCopyPipeline	10/11/21, 5:15:54 PM	10/11/21, 5:15:16 PM	00:01:21	Succeeded	Manual trigger		ef11c5d4-087c-4313-a0f5-5...
IncrementalCopyPipeline	10/11/21, 3:50:20 PM	10/11/21, 3:50:53 PM	00:00:32	Succeeded	Manual trigger		318c796f-38be-4947-b96c-0...
IncrementalCopyPipeline	10/11/21, 3:44:58 PM	10/11/21, 3:50:00 PM	00:05:02	Cancelled	Manual trigger		163055a-b39c-4406-b9eb-...
IncrementalCopyPipeline	10/11/21, 3:36:56 PM	10/11/21, 3:37:50 PM	00:00:54	Succeeded	Manual trigger		569c4bcf-9156-4f16-8696-3...
IncrementalCopyPipeline	10/11/21, 3:10:07 PM	10/11/21, 3:10:39 PM	00:00:32	Succeeded	Manual trigger		f9158a38-e494-4b97-8a09-5...
IncrementalCopyPipeline	10/11/21, 2:54:41 PM	10/11/21, 2:55:21 PM	00:00:39	Succeeded	Manual trigger		407321ba-4712-40f5-905c...
IncrementalCopyPipeline	10/11/21, 2:15:55 PM	10/11/21, 2:16:26 PM	00:00:30	Succeeded	Manual trigger		b105ec4f-1293-4930-9848...
IncrementalCopyPipeline	10/11/21, 9:22:01 AM	10/11/21, 9:22:46 AM	00:00:45	Succeeded	Manual trigger		785711c4-228e-42fe-8876-ff...
IncrementalCopyPipeline	10/11/21, 9:09:23 AM	10/11/21, 9:10:00 AM	00:00:36	Succeeded	Manual trigger		78534c2b-dc26-4040-b999...
IncrementalCopyPipeline	10/11/21, 9:02:19 AM	10/11/21, 9:03:42 AM	00:01:23	Succeeded	Manual trigger		0c0bd22b-992c-426d-b469...
IncrementalCopyPipeline	10/11/21, 8:49:22 AM	10/11/21, 8:49:41 AM	00:00:18	Failed	Manual trigger		9803b599-29d9-449d-9eeb...

Screenshot 5.3.d: Pipeline Trigger Details

The screenshot shows the 'Pipeline runs' section in the Microsoft Azure Data Factory interface. The left sidebar includes options like Dashboards, Runs, Pipeline runs (selected), Trigger runs, Runtimes & sessions, Integration runtimes, Data flow debug, Notifications, and Alerts & metrics. The main area displays a table of pipeline runs:

Pipeline name	Run start	Run end	Duration	Triggered by	Status	Error	Run	Parameters	Annotate
IncrementalCopyPipeline	10/11/21, 5:50:53 PM	10/11/21, 5:51:36 PM	00:00:43	Manual trigger	Succeeded		Original		
IncrementalCopyPipeline	10/11/21, 5:45:25 PM	10/11/21, 5:45:25 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:44:49 PM	10/11/21, 5:44:49 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:42:40 PM	10/11/21, 5:42:40 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:40:28 PM	10/11/21, 5:40:28 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:40:00 PM	10/11/21, 5:40:00 PM	00:00:00	Manual trigger	Failed		Original		
> IncrementalCopyPipeline	10/11/21, 5:31:06 PM	10/11/21, 5:31:06 PM	00:00:00	Manual trigger	Failed		Rerun (Latest)		
IncrementalCopyPipeline	10/11/21, 5:21:58 PM	10/11/21, 5:21:58 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:19:28 PM	10/11/21, 5:19:28 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:15:24 PM	10/11/21, 5:15:24 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:13:38 PM	10/11/21, 5:13:38 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 5:11:52 PM	10/11/21, 5:11:52 PM	00:00:00	Manual trigger	Failed		Original		
IncrementalCopyPipeline	10/11/21, 2:19:22 PM	10/11/21, 2:21:39 PM	00:02:17	Manual trigger	Succeeded		Original		

- Output verified: Table values in Azure SQL database copied to Text File in Azure blob storage
- When the copy activity performed for first time, text file 5 rows copied into text file as first row as header and next 5 records as 5 rows.
- New watermark is captured from the table using the query.
- New values 2 records are inserted into the table
- When we triggered, records are copied into new text file, therefore updated 2 records are created in the table with index value 6,7.
- Again, New watermark is captured from the table using the query.

Screenshot 5.4: New records insertion in Data Source Table

The screenshot shows the Microsoft Azure SQL Database Query editor. The top navigation bar includes 'Microsoft Azure', 'Search resources, services, and docs (G+)', and user information. The main area shows a query editor for the 'Project-Arthi-sqldb' database, specifically for the 'Project-Arthi-sqldb' schema. The query editor has tabs for 'Query 1', 'Query 2', 'Query 3', and 'Query 4' (selected). The code pane contains the following T-SQL script:

```

1  INSERT INTO data_source_table
2  VALUES (6, 'newdata', '9/6/2017 2:23:00 AM')
3
4  INSERT INTO data_source_table
5  VALUES (7, 'newdata', '9/7/2017 9:01:00 AM')

```

The results pane shows the output of the query:

Query succeeded: Affected rows: 2

The bottom status bar indicates 'Query succeeded | 0s'.

Screenshot 5.5: Output Verification: Files generated after every run

The screenshot shows the Microsoft Azure Storage Explorer interface. The left sidebar lists blob containers: 'adftagedpolybasetempdata', 'data' (which is selected), 'moviedata', and 'myblob'. The main pane displays a table of blobs under the 'data' container. The table has columns: NAME, ACCESS TIER, ACCESS TIER LAST MODIFIED, LAST MODIFIED, BLOB TYPE, CONTENT TYPE, SIZE, STATUS, REMAINING DAYS, and DELETED TIME. There are four entries:

NAME	ACCESS TIER	ACCESS TIER LAST MODIFIED	LAST MODIFIED	BLOB TYPE	CONTENT TYPE	SIZE	STATUS	REMAINING DAYS	DELETED TIME
Incremental-318c796f-38be-4947-b96c-09eda28dfe0fb.txt	Hot (inferred)	10/11/2021, 3:03:39 PM	10/11/2021, 3:03:39 PM	Block Blob	application/octet-stream	39 B	Active		
Incremental-e1f1c5d4-0071-4313-a055-31efb635a093.txt	Hot (inferred)	10/11/2021, 5:14:52 PM	10/11/2021, 5:14:52 PM	Block Blob	application/octet-stream	220 B	Active		
Incremental-a2ec67fb-ac05-4934-beef-0df5fc6767f5.txt	Hot (inferred)	10/11/2021, 5:28:20 PM	10/11/2021, 5:28:20 PM	Block Blob	application/octet-stream	112 B	Active		
Incremental-378ad153-f566-43d6-bb34-05d124c8bed.txt	Hot (inferred)	10/11/2021, 5:51:21 PM	10/11/2021, 5:51:21 PM	Block Blob	application/octet-stream	302 B	Active		

Showing 1 to 4 of 4 cached items

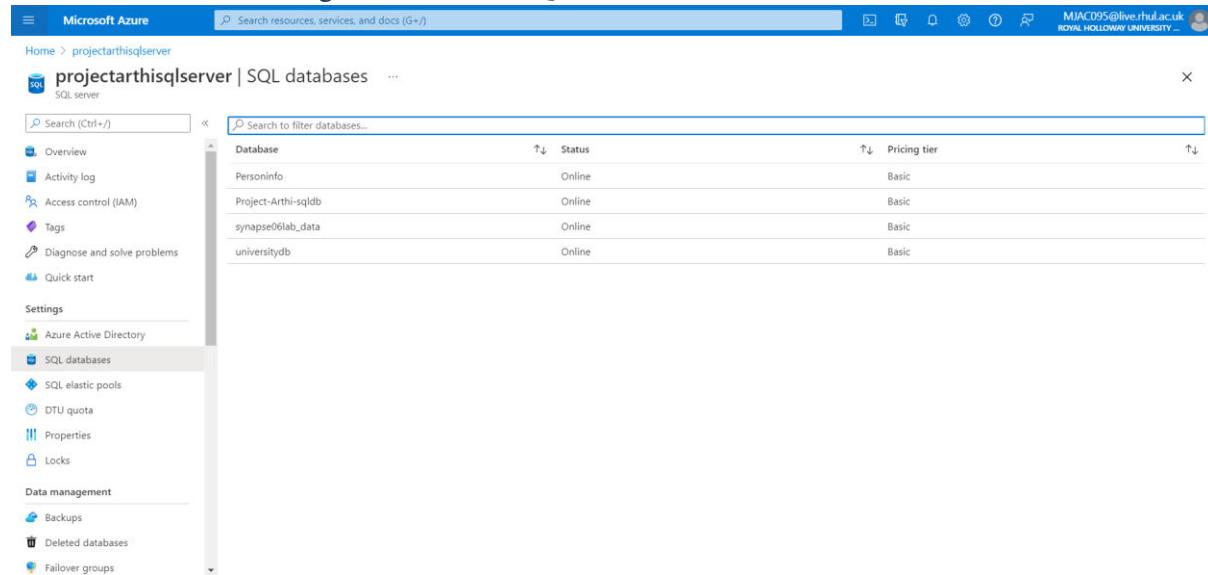
Files: attached queries and output text files for this Problem in Solution5 Folder.

**Problem Statement 6: Design a Relational Database Model for the below scenario
in Azure SQL Database**

Solution :

- Created a Azure SQL Database named as: **universitydb** containing 4 Tables namely **dbo.Person, dbo.Student, dbo.Course, dbo.Credit**
- Created tables and relationship among them as given in the problem
- Created database and populated the table using insert query in each table using random values

Screenshot 6.1: Creating DB in Azure SQL Database



The screenshot shows the Microsoft Azure portal interface for managing SQL databases. The left sidebar shows navigation options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Quick start, Settings (with Azure Active Directory and SQL databases selected), Data management (with Backups, Deleted databases, and Failover groups), and Monitoring (with DTU quota, Properties, and Locks). The main content area displays a table of databases:

Database	Status	Pricing tier
Personinfo	Online	Basic
Project-Arthi-sqldb	Online	Basic
synapse06lab_data	Online	Basic
universitydb	Online	Basic

Screenshot 6.2.a: Creating Person Table

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'universitydb'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, Query editor (preview), Power Platform, Settings, Data management, and Replicates. The main area displays the 'universitydb (adm)' object explorer, which shows a 'Tables' section with 'dbo.Person'. The 'dbo.Person' table is expanded, showing columns: PersonId (PK, varchar, not null), FirstName (varchar, null), MiddleInitial (varchar, null), LastName (varchar, null), and DateOfBirth (date, null). A query editor window titled 'Query 1' is open, containing the following T-SQL code:

```
--Person(_PersonId_,_FirstName,_MiddleInitial,_LastName,_DateOfBirth).
CREATE TABLE dbo.Person (
    PersonId varchar(255) NOT NULL PRIMARY KEY,
    FirstName varchar(255),
    MiddleInitial varchar(255),
    LastName varchar(255),
    DateOfBirth DATE
);
```

The 'Messages' tab at the bottom of the query editor shows the message: 'Query succeeded: Affected rows: 0'.

Screenshot 6.2.b: Creating Course Table

The screenshot shows the Microsoft Azure portal interface for the same 'universitydb' database. The left sidebar is identical to the previous screenshot. The main area displays the 'universitydb (adm)' object explorer, showing a 'Tables' section with 'dbo.Course'. The 'dbo.Course' table is expanded, showing columns: CourseId (PK, varchar, not null), Name (varchar, null), and Teacher (varchar, null). A query editor window titled 'Query 2' is open, containing the following T-SQL code:

```
--Course(_CourseId_, _Name, _Teacher).
CREATE TABLE dbo.Course (
    CourseId varchar(255) NOT NULL PRIMARY KEY,
    Name varchar(255),
    Teacher varchar(255)
);
```

The 'Messages' tab at the bottom of the query editor shows the message: 'Query succeeded: Affected rows: 0'.

Screenshot 6.2.c: Creating Student Table

The screenshot shows the Microsoft Azure portal interface for a database named 'universitydb'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, Query editor (preview), Power Platform, Power BI (preview), Power Apps (preview), Power Automate (preview), Settings, Compute + storage, Connection strings, Properties, Locks, and Data management. The main area displays the 'universitydb (adm)' object explorer, which shows tables like dbo.Course, dbo.Person, and dbo.Student. The 'dbo.Student' table is expanded, showing columns: StudentId (PK, varchar, not null), PersonId (varchar, null), and Email (varchar, null). A query editor window titled 'Query 3' is open, containing the following T-SQL code:

```
1 --Student_(StudentId ,PersonId,Email).
2 --The attribute PersonId references the PK of Person.
3 CREATE TABLE dbo.Student (
4     StudentId varchar(255) NOT NULL PRIMARY KEY,
5     PersonId varchar(255) FOREIGN KEY references dbo.Person,
6     Email varchar(255)
7 );
8
9 );
```

The 'Messages' tab shows the message: 'Query succeeded: Affected rows: 0'. The status bar at the bottom indicates 'Query succeeded | 0s'.

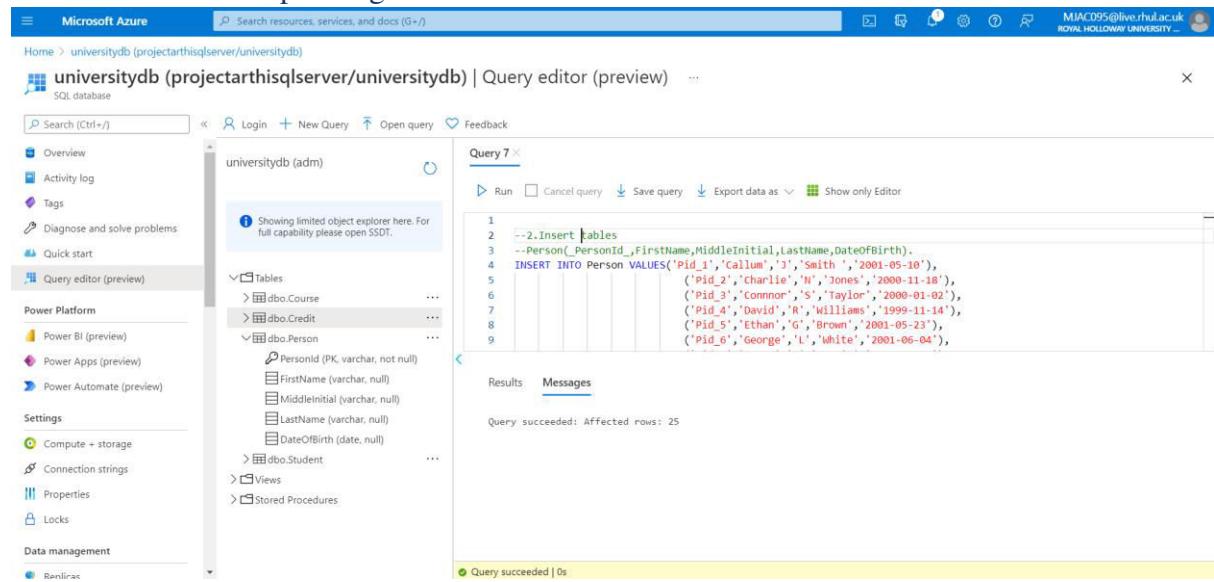
Screenshot 6.2.d: Creating Credit Table

The screenshot shows the Microsoft Azure portal interface for the same 'universitydb' database. The left sidebar and object explorer are identical to the previous screenshot. The 'dbo.Credit' table is expanded, showing columns: StudentId (PK, varchar, not null), CourseId (PK varchar, not null), Grade (varchar, null), and Attempt (varchar, null). A query editor window titled 'Query 4' is open, containing the following T-SQL code:

```
3 --The attribute StudentId references the PK of Student.
4 --The attribute CourseId references the PK of Course.
5 CREATE TABLE dbo.Credit (
6     StudentId varchar(255) FOREIGN KEY references dbo.Student,
7     CourseId varchar(255) FOREIGN KEY references dbo.Course,
8     Grade varchar(255),
9     Attempt varchar(255),
10    PRIMARY KEY(StudentId, courseId)
11 );
```

The 'Messages' tab shows the message: 'Query succeeded: Affected rows: 0'. The status bar at the bottom indicates 'Query succeeded | 0s'.

Screenshot 6.3.a: Populating Person Table



The screenshot shows the Microsoft Azure portal interface for a SQL database named 'universitydb'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, Query editor (preview), Power BI (preview), Power Apps (preview), Power Automate (preview), Compute + storage, Connection strings, Properties, Locks, and Data management. The 'Query editor (preview)' link is currently selected.

The main area displays a query editor titled 'Query 7' with the following T-SQL script:

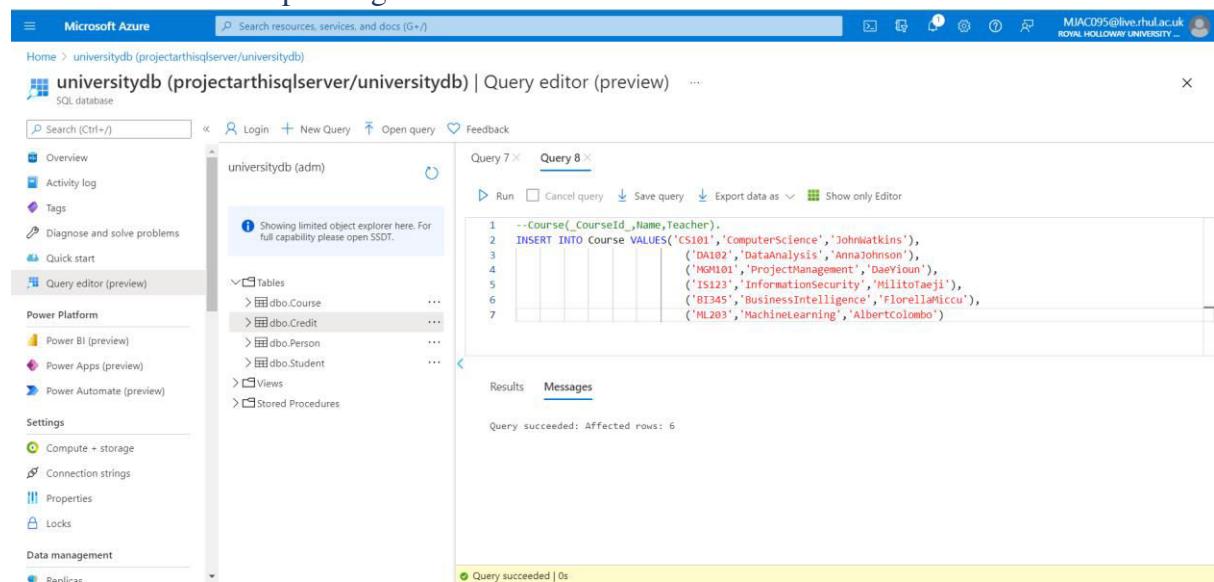
```

1 --2.Insert tables
2 --Person(_PersonId_,FirstName,MiddleInitial,LastName,DateOfBirth),
3 INSERT INTO Person VALUES('Pid_1','Callum','J','Smith ','2001-05-10'),
4 ('Pid_2','charlie','W','Jones ','2000-11-18'),
5 ('Pid_3','Connor','S','Taylor ','2000-01-02'),
6 ('Pid_4','David','R','Williams ','1999-11-14'),
7 ('Pid_5','Ethan','G','Brown ','2001-05-23'),
8 ('Pid_6','George','L','White ','2001-06-04'),
9

```

The results pane shows a green status bar indicating 'Query succeeded | 0s'.

Screenshot 6.3.b: Populating Course Table



The screenshot shows the Microsoft Azure portal interface for a SQL database named 'universitydb'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, Query editor (preview), Power BI (preview), Power Apps (preview), Power Automate (preview), Compute + storage, Connection strings, Properties, Locks, and Data management. The 'Query editor (preview)' link is currently selected.

The main area displays a query editor titled 'Query 8' with the following T-SQL script:

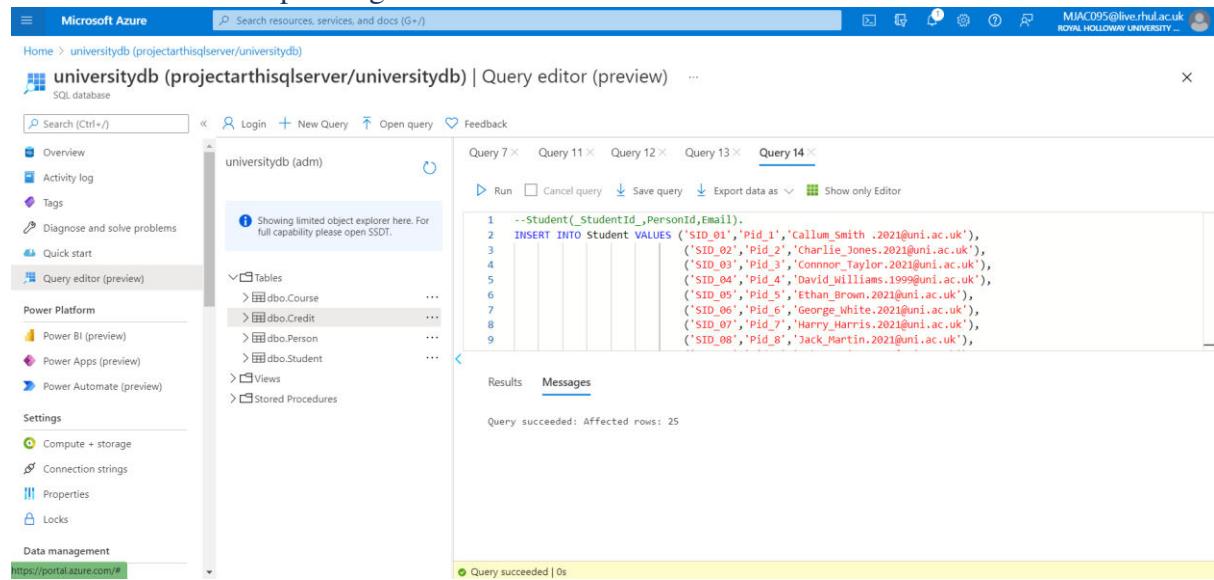
```

1 --Course(_CourseId_,Name,Teacher),
2 INSERT INTO Course VALUES('CS101','ComputerScience','JohnWatkins'),
3 ('DA102','DataAnalysis','AnnaJohnson'),
4 ('MG101','ProjectManagement','DaeYoun'),
5 ('IS123','InformationSecurity','Miltotaeji'),
6 ('BI345','BusinessIntelligence','FlorellaPiccu'),
7 ('ML203','MachineLearning','AlbertColombo')

```

The results pane shows a green status bar indicating 'Query succeeded | 0s'.

Screenshot 6.3.c: Populating Student Table



The screenshot shows the Microsoft Azure portal interface for a SQL database named 'universitydb'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, Query editor (preview), Power Platform, Power BI (preview), Power Apps (preview), Power Automate (preview), Settings, Compute + storage, Connection strings, Properties, and Locks. The main area displays the 'universitydb (adm)' object explorer, which lists Tables (dbo.Course, dbo.Credit, dbo.Person, dbo.Student), Views, and Stored Procedures. A message box indicates: 'Showing limited object explorer here. For full capability please open SSDT.' Below the object explorer, the query editor shows a list of queries numbered 7 through 14. Query 14 is selected and contains the following T-SQL code:

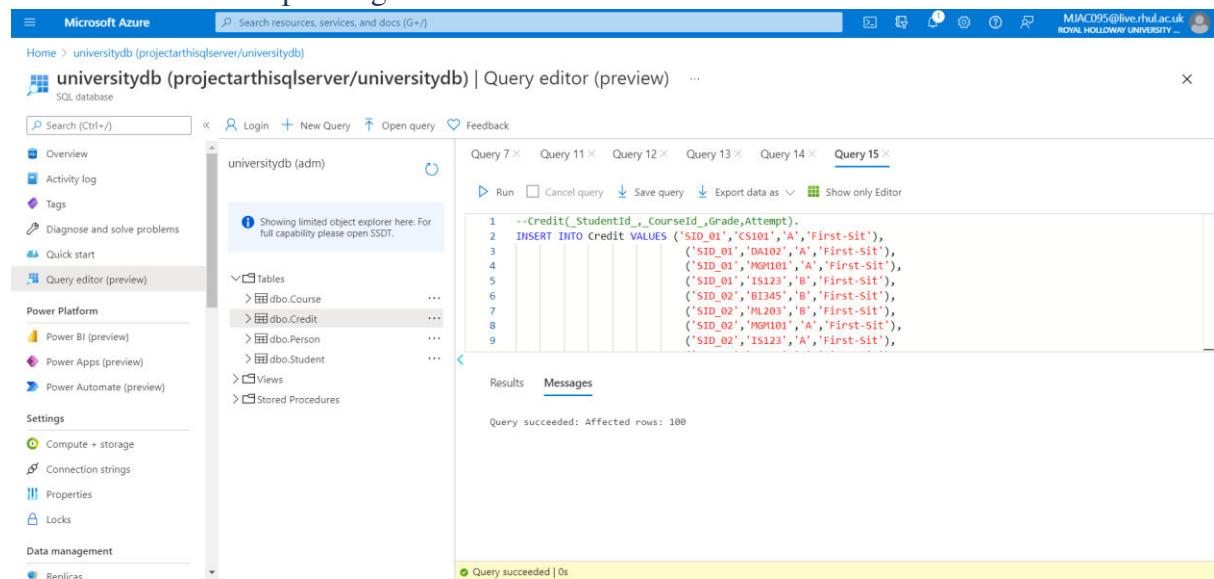
```

1 --Student(_StudentId_,PersonId,Email).
2 INSERT INTO Student VALUES ('SID_01','Pid_1','Callum_Smith .2021@uni.ac.uk'),
3 ('SID_02','Pid_2','Charlie_Jones.2021@uni.ac.uk'),
4 ('SID_03','Pid_3','Connor_Taylor.2021@uni.ac.uk'),
5 ('SID_04','Pid_4','David_Williams.1999@uni.ac.uk'),
6 ('SID_05','Pid_5','Ethan_Brown.2021@uni.ac.uk'),
7 ('SID_06','Pid_6','George_White.2021@uni.ac.uk'),
8 ('SID_07','Pid_7','Harry_Harris.2021@uni.ac.uk'),
9 ('SID_08','Pid_8','Jack_Martin.2021@uni.ac.uk'),

```

The results pane shows 'Query succeeded: Affected rows: 25'.

Screenshot 6.3.d: Populating Credit Table



The screenshot shows the Microsoft Azure portal interface for a SQL database named 'universitydb'. The left sidebar contains the same navigation links as in screenshot 6.3.c. The main area displays the 'universitydb (adm)' object explorer, listing Tables (dbo.Course, dbo.Credit, dbo.Person, dbo.Student), Views, and Stored Procedures. A message box indicates: 'Showing limited object explorer here. For full capability please open SSDT.' Below the object explorer, the query editor shows a list of queries numbered 7 through 15. Query 15 is selected and contains the following T-SQL code:

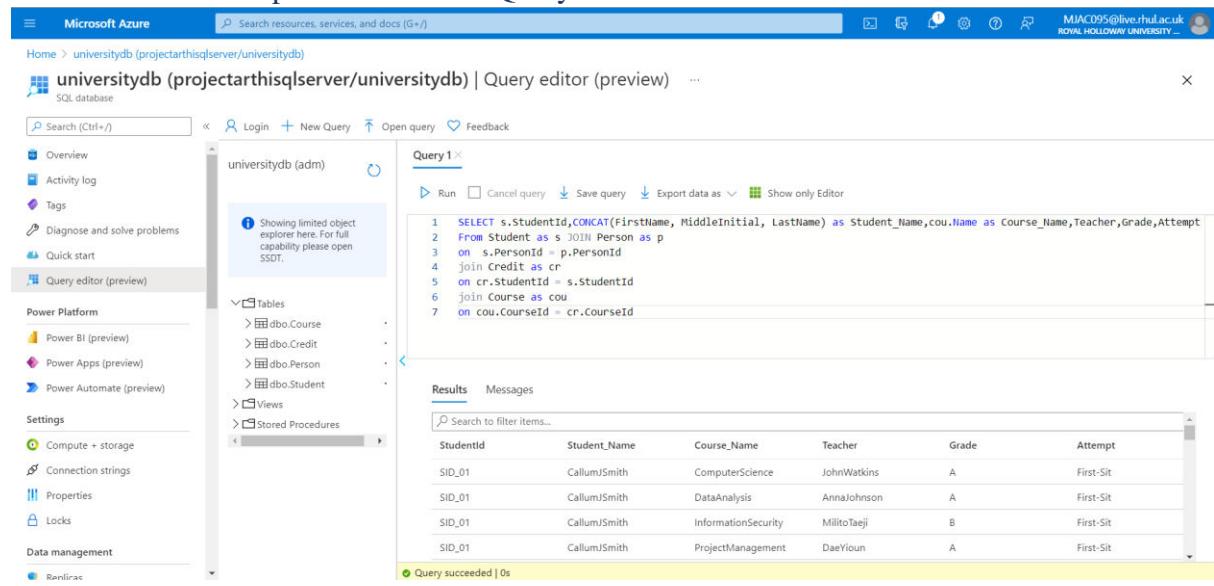
```

1 --Credit(_StudentId_,_CourseId_,Grade,Attempt).
2 INSERT INTO Credit VALUES ('SID_01','CS101','A','First-sit'),
3 ('SID_01','DA102','A','First-sit'),
4 ('SID_01','MGR101','A','First-sit'),
5 ('SID_01','IS123','B','First-sit'),
6 ('SID_02','B1345','B','First-sit'),
7 ('SID_02','ML203','B','First-sit'),
8 ('SID_02','MGM101','A','First-sit'),
9 ('SID_02','IS123','A','First-sit'),

```

The results pane shows 'Query succeeded: Affected rows: 100'.

Screenshot 6.4: Output Verification: Query to extract the student details



The screenshot shows the Microsoft Azure portal interface for a SQL database named 'universitydb'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and various Power Platform and Settings options. The main area is titled 'universitydb (admin) | Query editor (preview)'.

The 'Query 1' tab is active, displaying the following T-SQL code:

```

1 SELECT s.StudentId,CONCAT(FirstName, MiddleInitial, LastName) as Student_Name,cou.Name as Course_Name,Teacher,Grade,Attempt
2 From Student as s JOIN Person as p
3 on s.PersonId = p.PersonId
4 Join Credit as cr
5 on cr.StudentId = s.StudentId
6 join Course as cou
7 on cou.CourseId = cr.CourseId

```

The 'Results' tab is selected, showing a table with the following data:

StudentId	Student_Name	Course_Name	Teacher	Grade	Attempt
SID_01	CallumSmith	ComputerScience	JohnWatkins	A	First-Sit
SID_01	CallumSmith	DataAnalysis	AninaJohnson	A	First-Sit
SID_01	CallumSmith	InformationSecurity	MilitoTaeji	B	First-Sit
SID_01	CallumSmith	ProjectManagement	DaeYioun	A	First-Sit

A green status bar at the bottom indicates 'Query succeeded | 0s'.

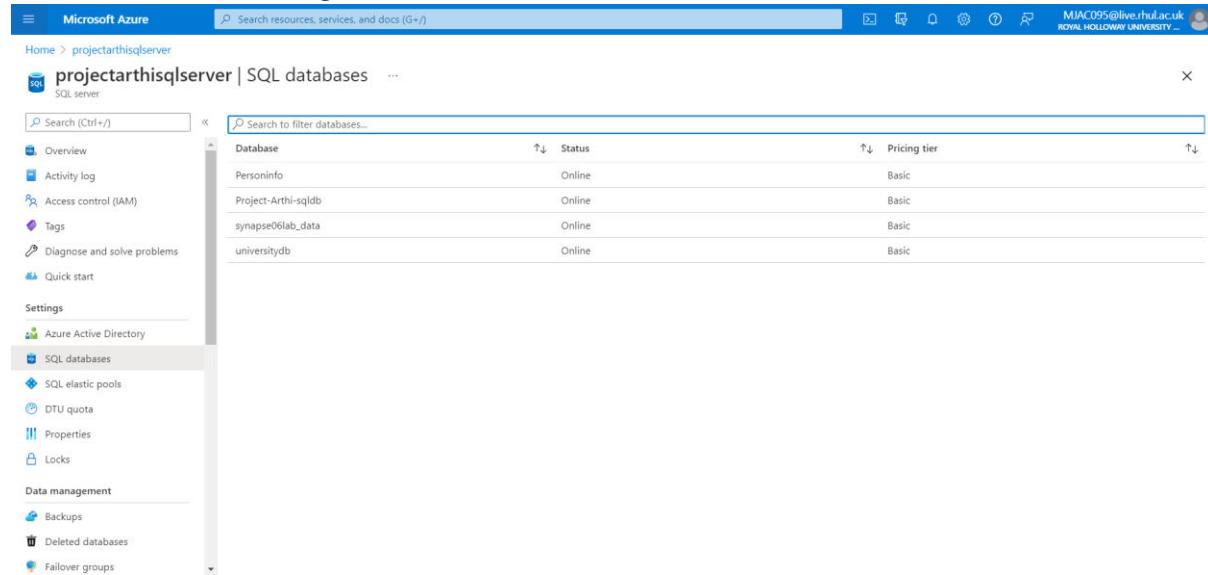
Files: attached create table, insert values, and extracting queries sql file of this Problem is in Solution6 Folder.

**Problem Statement 7: Design a Relational Database Model for the below scenario
in Azure SQL Database**

Solution :

- Created an Azure SQL Database named as: **Personinfo** containing 4 Tables namely **dbo.Person, dbo.Address, dbo.ContactDetail, dbo.ContactDetailType**
- Created tables and relationship among them as given in the problem
- Created database and populated the table using insert query in each table using random values.

Screenshot 7.1: Creating DB in Azure SQL Database



Database	Status	Pricing tier
Personinfo	Online	Basic
Project-Arathi-sqldb	Online	Basic
synapse06lab_data	Online	Basic
universitydb	Online	Basic

Screenshot 7.1.a: Creating Person Table

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'Personinfo'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The main area displays the 'Personinfo (adm)' database object explorer, which shows a 'Tables' node expanded, revealing the 'dbo.Person' table. The table has three columns: PersonId (PK, int, not null), FirstName (varchar, null), and LastName (varchar, null). A query editor window titled 'Query 1' is open, containing the following T-SQL code:

```
--Person( _PersonId ,_FirstName,_LastName ).  
CREATE TABLE dbo.Person (  
    PersonId int NOT NULL PRIMARY KEY,  
    FirstName varchar(255) ,  
    LastName varchar(255)  
);
```

The results pane below the query editor shows the message "Query succeeded: Affected rows: 0".

Screenshot 7.1.b: Creating ContactDetailType Table

The screenshot shows the Microsoft Azure portal interface for the same 'Personinfo' database. The left sidebar and object explorer are identical to the previous screenshot. The 'Query editor (preview)' window is now active, showing 'Query 2' selected. It contains the following T-SQL code:

```
--ContactDetailType( _CDT_Id ,_Detail ).  
CREATE TABLE dbo.ContactDetailType (  
    TypeId int NOT NULL PRIMARY KEY,  
    Detail varchar(255)  
);
```

The results pane shows the message "Query succeeded | 0s".

Screenshot 7.1.c: Creating Address Table

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'Personinfo'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The main area displays the 'Personinfo (adm)' database object explorer, which shows tables like 'dbo.ContactDetailType' and 'dbo.Address'. The 'Query editor (preview)' tab is active, showing five tabs: Query 1, Query 2, Query 3 (selected), Query 4, and Query 5. The code in Query 3 is:

```
1 --Address(_AddressId_,PersonId,Line1,Line2,City,State,Zip,Country,TypeId).
2 --The attribute PersonId references the PK of Person.
3 --The attribute TypeId references the PK of ContactDetailType.
4 CREATE TABLE dbo.Address (
5     AddressId int NOT NULL PRIMARY KEY,
6     PersonId int FOREIGN KEY references dbo.Person,
7     Line1 varchar(255),
8     Line2 varchar(255),
9     City varchar(255),
```

The 'Messages' tab at the bottom indicates 'Query succeeded | 0s'.

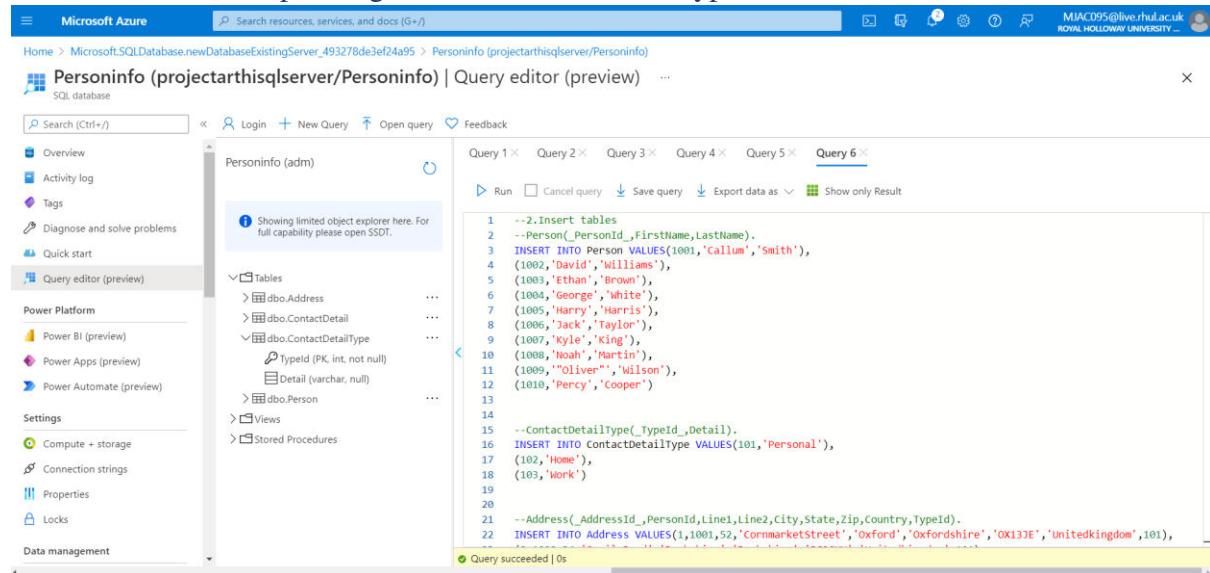
Screenshot 7.1.d: Creating ContactDetail Table

The screenshot shows the Microsoft Azure portal interface for the same 'Personinfo' database. The left sidebar and object explorer are identical to the previous screenshot. The 'Query editor (preview)' tab is active, showing five tabs: Query 1, Query 2, Query 3, Query 4 (selected), and Query 5. The code in Query 4 is:

```
1 --ContactDetail(_ContactId_, PersonId, Detail,TypeId).
2 --The attribute PersonId references the PK of Person.
3 --The attribute TypeId references the PK of ContactDetailType.
4 CREATE TABLE dbo.ContactDetail (
5     ContactId int NOT NULL PRIMARY KEY,
6     PersonId int FOREIGN KEY references dbo.Person,
7     TypeId int FOREIGN KEY references dbo.ContactDetailType,
8     Detail varchar(255)
9 );
```

The 'Messages' tab at the bottom indicates 'Query succeeded | 0s'.

Screenshot 7.2.a: Populating Person, ContactDetailType, Address Table



```

--2.Insert tables
--Person(_PersonId_,FirstName,LastName).
INSERT INTO Person VALUES(1001,'Callum','Smith'),
(1002,'David','Williams'),
(1003,'Ethan','Brown'),
(1004,'George','White'),
(1005,'Harry','Harris'),
(1006,'Jack','Taylor'),
(1007,'Kyle','King'),
(1008,'Noah','Martin'),
(1009,'Oliver','Wilson'),
(1010,'Percy','Cooper')

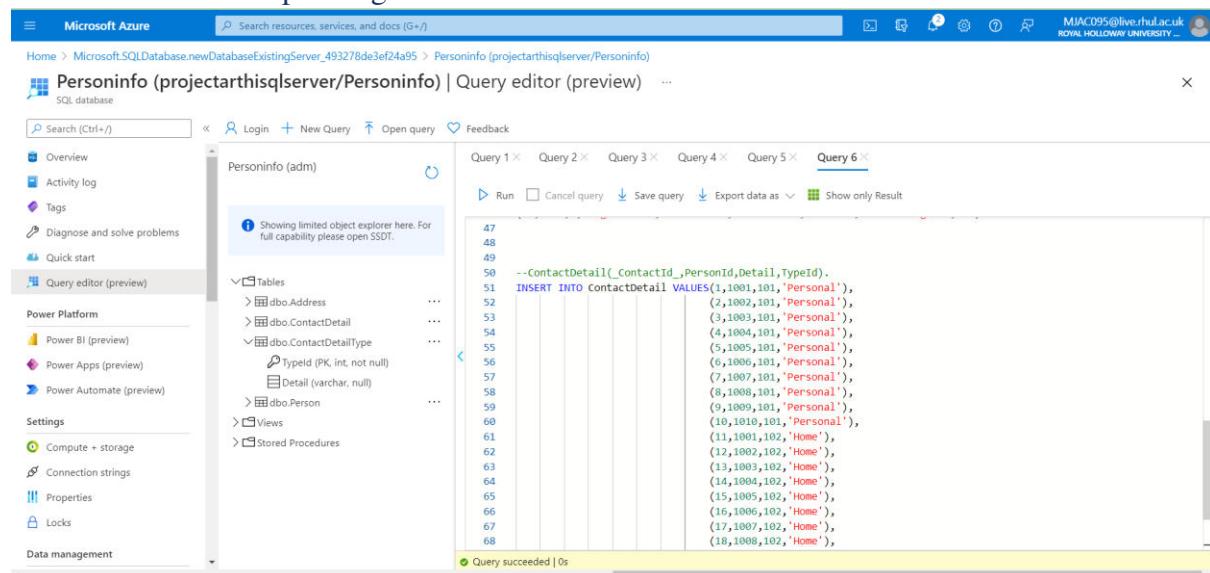
--ContactDetailType(_TypeId_,Detail).
INSERT INTO ContactDetailType VALUES(101,'Personal'),
(102,'Home'),
(103,'Work')

--Address(_AddressId_,PersonId,Line1,Line2,City,State,Zip,Country,TypeId).
INSERT INTO Address VALUES(1,1001,52,'Cormarketstreet','oxford','oxfordshire','OX13QE','Unitedkingdom',101),
(2,1002,53,'Highgate','London','Greaterlondon','N17 4AA','Unitedkingdom',102),
(3,1003,54,'Bakerstreet','London','Greaterlondon','W1U 1EE','Unitedkingdom',103),
(4,1004,101,'Personal'),
(5,1005,101,'Personal'),
(6,1006,101,'Personal'),
(7,1007,101,'Personal'),
(8,1008,101,'Personal'),
(9,1009,101,'Personal'),
(10,1010,101,'Personal'),
(11,1001,102,'Home'),
(12,1002,102,'Home'),
(13,1003,102,'Home'),
(14,1004,102,'Home'),
(15,1005,102,'Home'),
(16,1006,102,'Home'),
(17,1007,102,'Home'),
(18,1008,102,'Home')

```

Query succeeded | 0s

Screenshot 7.2.b: Populating Table ContactDetail



```

--ContactDetail(_ContactId_,PersonId,Detail,TypeId).
INSERT INTO ContactDetail VALUES(1,1001,101,'Personal'),
(2,1002,101,'Personal'),
(3,1003,101,'Personal'),
(4,1004,101,'Personal'),
(5,1005,101,'Personal'),
(6,1006,101,'Personal'),
(7,1007,101,'Personal'),
(8,1008,101,'Personal'),
(9,1009,101,'Personal'),
(10,1010,101,'Personal'),
(11,1001,102,'Home'),
(12,1002,102,'Home'),
(13,1003,102,'Home'),
(14,1004,102,'Home'),
(15,1005,102,'Home'),
(16,1006,102,'Home'),
(17,1007,102,'Home'),
(18,1008,102,'Home')

```

Query succeeded | 0s

Screenshot 7.3: Output Verification: Query to extract the Person details

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'Personinfo'. The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The main area displays a query editor titled 'Query 1' with the following SQL code:

```
1 SELECT distinct(p.PersonId),FirstName, LastName, City,ContactId,cd.TypeId, Detail
2 FROM Person as p Join Address as ad
3 on ad.PersonId = p.PersonId
4 join ContactDetail as cd
5 on cd.PersonId = p.PersonId
6 order by p.PersonId
```

The results pane shows a table with the following data:

PersonId	FirstName	LastName	City	ContactId	TypeId	Detail
1001	Callum	Smith	Oxford	1	101	Personal
1001	Callum	Smith	Oxford	11	102	Home
1002	David	Williams	Berkshire	2	101	Personal
1002	David	Williams	Berkshire	12	102	Home

The status bar at the bottom indicates 'Query succeeded | 0s'.

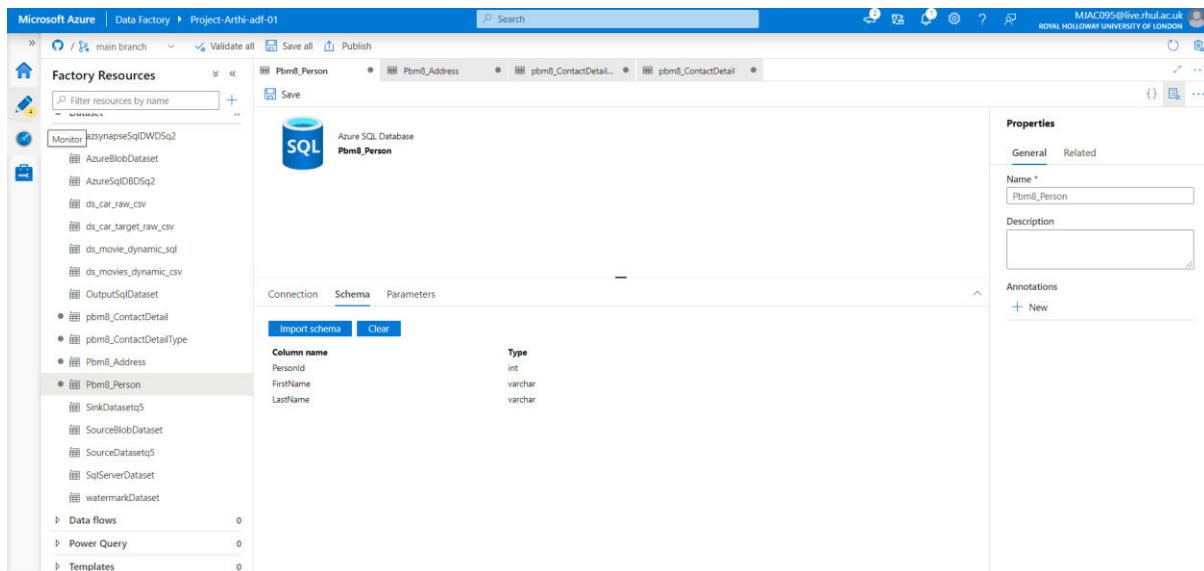
Files: attached create table, insert values, and extracting queries sql file with this Project in Solution7 Folder.

Problem Statement 8: Design a Non-Relational Database Model equivalent for a Relational Model

Solution :

- Created an Azure SQL Database named as: **Personinfo** containing 4 Tables namely **dbo.Person, dbo.Address, dbo.ContactDetail, dbo.ContactDetailType**
- Created tables and relationship among them as given in the problem
- Created database and populated the table using insert query in each table using random values
- Created a dataset for each table in Azure Data factory to have table in JSON schema

Screenshot 8.1.a: Datasets for PersonInfoDb



Person - Schema:

```
{  
  "name": "Pbm8_Person",  
  "properties": {  
    "linkedServiceName": {  
      "referenceName": "ls_pbm8",  
      "type": "LinkedServiceReference"  
    },  
    "annotations": [],  
    "type": "AzureSqlTable",  
    "schema": [  
      {"name": "PersonId", "type": "int"},  
      {"name": "FirstName", "type": "varchar"},  
      {"name": "LastName", "type": "varchar"}  
    ]  
  }  
}
```

```

{
    "name": "PersonId",
    "type": "int",
    "precision": 10
},
{
    "name": "FirstName",
    "type": "varchar"
},
{
    "name": "LastName",
    "type": "varchar"
}
],
"typeProperties": {
    "schema": "dbo",
    "table": "Person"
}
}
}
}

```

Address - Schema:

```

{
    "name": "Pbm8_Address",
    "properties": {
        "linkedServiceName": {
            "referenceName": "ls_pbm8",
            "type": "LinkedServiceReference"
        },
        "annotations": [],
        "type": "AzureSqlTable",
        "schema": [
            {
                "name": "AddressId",
                "type": "int",
                "precision": 10
            },
            {
                "name": "PersonId",
                "type": "int",
                "precision": 10
            },
            {
                "name": "Line1",
                "type": "varchar"
            }
        ]
    }
}

```

```

        "name": "Line2",
        "type": "varchar"
    },
    {
        "name": "City",
        "type": "varchar"
    },
    {
        "name": "State",
        "type": "varchar"
    },
    {
        "name": "Zip",
        "type": "varchar"
    },
    {
        "name": "Country",
        "type": "varchar"
    },
    {
        "name": "TypeId",
        "type": "int",
        "precision": 10
    }
],
"typeProperties": {
    "schema": "dbo",
    "table": "Address"
}
}
}

```

ContactDetailType - Schema:

```

{
    "name": "pbm8_ContactDetailType",
    "properties": {
        "linkedServiceName": {
            "referenceName": "ls_pbm8",
            "type": "LinkedServiceReference"
        },
        "annotations": [],
        "type": "AzureSqlTable",
        "schema": [
            {
                "name": "TypeId",
                "type": "int",
                "precision": 10
            }
        ]
    }
}

```

```

        },
        {
            "name": "Detail",
            "type": "varchar"
        }
    ],
    "typeProperties": {
        "schema": "dbo",
        "table": "ContactDetailType"
    }
}
}
}

```

ContactDetail - Schema:

```

{
    "name": "pbm8_ContactDetail",
    "properties": {
        "linkedServiceName": {
            "referenceName": "ls_pbm8",
            "type": "LinkedServiceReference"
        },
        "annotations": [],
        "type": "AzureSqlTable",
        "schema": [
            {
                "name": "ContactId",
                "type": "int",
                "precision": 10
            },
            {
                "name": "PersonId",
                "type": "int",
                "precision": 10
            },
            {
                "name": "TypeId",
                "type": "int",
                "precision": 10
            },
            {
                "name": "Detail",
                "type": "varchar"
            }
        ],
        "key": {
            "name": "ContactId"
        }
    }
}

```

```

    "typeProperties": {
        "schema": "dbo",
        "table": "ContactDetail"
    }
}
}
}

```

Screenshot 8.1.b: Query extracting People full contact info

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'Personinfo'. The left sidebar includes links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and Query editor (preview). The main area displays a query editor with two tabs: 'Query 1' and 'Query 2'. The 'Query 1' tab contains the following T-SQL code:

```

1 --Person(_PersonId_,FirstName,LastName).
2 --Address(_AddressId_,PersonId,Line1,Line2,City,State,Zip,Country,TypeId).
3 --ContactDetail(_ContactId_, PersonId, Detail,TypeId).
4
5
6 SELECT DISTINCT(Person.PersonId) as PersonId, FirstName, LastName,
7 | AddressId,Line1,Line2,City,State,Zip,Country,ContactId,ContactDetail.TypeId,Detail
8 FROM Person Join Address
9 on Address.PersonId = Person.PersonId
10 join ContactDetail
11 on ContactDetail.PersonId = Person.PersonId
12 order by Person.PersonId
13 FOR JSON AUTO

```

The status bar at the bottom indicates 'Query succeeded | 0s'.

Query:

```

--Person(_PersonId_,FirstName,LastName).
--Address(_AddressId_,PersonId,Line1,Line2,City,State,Zip,Country,TypeId).
--ContactDetail(_ContactId_, PersonId, Detail,TypeId).

```

```

SELECT DISTINCT(Person.PersonId) as PersonId, FirstName, LastName,
AddressId,Line1,Line2,City,State,Zip,Country,ContactId,ContactDetail.TypeId,Detail
FROM Person Join Address
on Address.PersonId = Person.PersonId
join ContactDetail
on ContactDetail.PersonId = Person.PersonId
order by Person.PersonId
FOR JSON AUTO

```

Screenshot 8.1.c: Output in JSON FORMAT

The screenshot shows the Microsoft Azure portal interface for a SQL database named 'Personinfo'. The left sidebar includes links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, and the current 'Query editor (preview)'. The main area displays a query titled 'Query 1' with the results shown in 'Formatted JSON' format. The JSON output represents a list of two address records, each containing a person's details, their address, and a contact detail record. The contact detail record includes a contact ID, type ID, and detail type.

```
[{"Person.PersonId": 1001, "Person.FirstName": "Callum", "Person.LastName": "Smith", "Address": [{"AddressId": 1, "Line1": "52", "Line2": "CornmarketStreet", "City": "Oxford", "State": "Oxfordshire", "Zip": "OX13JE", "Country": "Unitedkingdom", "ContactDetail": [{"ContactId": 1, "TypeId": 101, "Detail": "Personal"}]}], {"AddressId": 11, "Line1": "52", "Line2": "CornmarketStreet", "City": "Oxford", "State": "Oxfordshire", "Zip": "OX13JE", "Country": "Unitedkingdom", "ContactDetail": [{"ContactId": 1, "TypeId": 101, "Detail": "Personal"}]}]
```

Sample Output:

```
[  
 {  
 "Person.PersonId": 1001,  
 "Person.FirstName": "Callum",  
 "Person.LastName": "Smith",  
 "Address": [  
 {  
 "AddressId": 1,  
 "Line1": "52",  
 "Line2": "CornmarketStreet",  
 "City": "Oxford",  
 "State": "Oxfordshire",  
 "Zip": "OX13JE",  
 "Country": "Unitedkingdom",  
 "ContactDetail": [  
 {  
 "ContactId": 1,  
 "TypeId": 101,  
 "Detail": "Personal"  
 }  
 ]  
 },  
 {  
 "AddressId": 11,  
 "Line1": "52",  
 "Line2": "CornmarketStreet",  
 "City": "Oxford",  
 "State": "Oxfordshire",  
 "Zip": "OX13JE",  
 "Country": "Unitedkingdom",  
 "ContactDetail": [  
 {  
 "ContactId": 1,  
 "TypeId": 101,  
 "Detail": "Personal"  
 }  
 ]  
 }
```

```
        },
        {
            "AddressId": 1,
            "Line1": "52",
            "Line2": "CornmarketStreet",
            "City": "Oxford",
            "State": "Oxfordshire",
            "Zip": "OX13JE",
            "Country": "Unitedkingdom",
            "ContactDetail": [
                {
                    "ContactId": 11,
                    "TypeId": 102,
                    "Detail": "Home"
                }
            ]
        },
        {
            "AddressId": 11,
            "Line1": "52",
            "Line2": "CornmarketStreet",
            "City": "Oxford",
            "State": "Oxfordshire",
            "Zip": "OX13JE",
            "Country": "Unitedkingdom",
            "ContactDetail": [
                {
                    "ContactId": 11,
                    "TypeId": 102,
                    "Detail": "Home"
                }
            ]
        }
    ],
},
```

Files: attached Complete output with this Project in Solution8 Folder.