How to extract data and load using Azure Data Factory
Step by Step of Loading Data from Azure SQL DB Source to Azure SQL DB Destination

For the present demo, we will need an Azure Account in the Portal.Azure.com Site. Inside the Account, we will create SQL SERVER with Adventure Works DB database and use it as source as well as Destination. These steps will first create DB Server and Corresponding Database in Azure.

In the given Azure Portal Subscription, we will create the DB with given subscription.
Connect to the given SQL SERVER with the created credentials

Create a View vSource-Product_SalesOrderDetail with the below screenshot for the Data Source as well as DestinationTarget for the Data Destination

Now after the Source and Destination Defined, we will use ADF to take Data from the View and Load the Destination Table.

Now to Create a Pipeline in Azure Data Factory to Extract the data from Data Source and Load in to Destination
After Clicking the Azure Data Factory,
After Clicking the Azure Data Factory, Click Author and Deploy

Step 1 Now Click the New Linked Service and Click Deploy

```javascript
{
    "name": "SQLAzureLinkedService",
    "properties": {
        "hubName": "sampleazuredatafactory1_hub",
        "type": "AzureSqlDatabase",
        "typeProperties": {
            "connectionString": "Data Source=adventureworksserver1.database.windows.net;Initial Catalog=SampleAdvWorksDB;Integrated Security=False;User ID=advuser;Password=**********;Connect Timeout=30;Encrypt=True"
        }
    }
}
```

Now follow below steps in the same order
Step 2
Now Create an Input Dataset

```json
{
    "name": "InputDataset-View",
    "properties": {
        "structure": [
            {
                "name": "SalesOrderID",
                "type": "Int32"
            },
            {
                "name": "SalesOrderDetailID",
                "type": "Int32"
            },
            {
                "name": "OrderQty",
                "type": "Int16"
            },
            {
                "name": "ProductName",
                "type": "String"
            },
            {
                "name": "ProductNumber",
                "type": "String"
            },
            {
                "name": "ProductColor",
                "type": "String"
            },
            {
                "name": "UnitPrice",
                "type": "Decimal"
            },
            {
                "name": "UnitPriceDiscount",
                "type": "Decimal"
            },
            {
                "name": "LineTotal",
                "type": "Decimal"
            }
        ],
        "published": false,
        "type": "AzureSqlTable",
        "linkedServiceName": "SQLAzureLinkedService",
        "typeProperties": {
            "structure": [
                {
                    "name": "SalesOrderID",
                    "type": "Int32"
                },
                {
                    "name": "SalesOrderDetailID",
                    "type": "Int32"
                },
                {
                    "name": "OrderQty",
                    "type": "Int16"
                },
                {
                    "name": "ProductName",
                    "type": "String"
                },
                {
                    "name": "ProductNumber",
                    "type": "String"
                },
                {
                    "name": "ProductColor",
                    "type": "String"
                },
                {
                    "name": "UnitPrice",
                    "type": "Decimal"
                },
                {
                    "name": "UnitPriceDiscount",
                    "type": "Decimal"
                },
                {
                    "name": "LineTotal",
                    "type": "Decimal"
                }
            ],
            "published": false,
            "type": "AzureSqlTable",
            "linkedServiceName": "SQLAzureLinkedService",
            "typeProperties": {
            
```}

Step 3
Now Create the Output Dataset

```json
{
    "name": "OutputDataset-Table",
    "properties": {
        "structure": [
            {
                "name": "SalesOrderID",
                "type": "Int32"
            },
            {
                "name": "SalesOrderDetailID",
                "type": "Int32"
            },
            {
                "name": "OrderQty",
                "type": "Int16"
            },
            {
                "name": "ProductName",
                "type": "String"
            },
            {
                "name": "ProductNumber",
                "type": "String"
            },
            {
                "name": "ProductColor",
                "type": "String"
            },
            {
                "name": "UnitPrice",
                "type": "Decimal"
            },
            {
                "name": "UnitPriceDiscount",
                "type": "Decimal"
            },
            {
                "name": "LineTotal",
                "type": "Decimal"
            }
        ],
        "published": false,
        "type": "AzureSqlTable",
        "linkedServiceName": "SQLAzureLinkedService",
        "typeProperties": {
```
| tableName: "[SalesLT].[vSource-Product_SalesOrderDetail]" |
| "availability": { |
| "frequency": "Day", |
| "interval": 1 |
| } |
| "external": true, |
| "policy": {} |
| } |
| tableName: "[dbo].[DestinationTarget]" |
| "availability": { |
| "frequency": "Day", |
| "interval": 1 |
| } |
| "external": false, |
| "policy": {} |
| } |
Step 4

Now create the Pipeline

{
  "name": "CopyPipeline",
  "properties": {
    "activities": [
      {
        "type": "Copy",
        "typeProperties": {
          "source": {
            "type": "SqlSource",
            "sqlReaderQuery": "select * from [SalesLT].[vSource-Product_SalesOrderDetail]"
          },
          "sink": {
            "type": "SqlSink",
            "writeBatchSize": 0,
            "writeBatchTimeout": "00:00:00"
          },
          "translator": {
            "type": "TabularTranslator",
            "columnMappings": "SalesOrderID:SalesOrderID,SalesOrderDetailID:SalesOrderDetailID,OrderQty:OrderQty,ProductName:ProductName,ProductNumber:ProductNumber,ProductColor:ProductColor,UnitPrice:UnitPrice,UnitPriceDiscount:UnitPriceDiscount,LineTotal:LineTotal"
          }
        }
      },
      "inputs": [
        {
          "name": "InputDataset-View"
        }
      ],
      "outputs": [
        {
          "name": "OutputDataset-Table"
        }
      ],
      "policy": {
        "timeout": "1.00:00:00",
        "concurrency": 1,
        "executionPriorityOrder": "NewestFirst",
        "style": "StartOfInterval",
        "retry": 3,
        "longRetry": 0,
        "longRetryInterval": "00:00:00"
      }
    }
  }
}
"scheduler": {
    "frequency": "Day",
    "interval": 1
},
"name": "Activity-0-[SalesLT][vSource-Product_SalesOrderDetail]->[dbo][DestinationTarget]"
},
"start": "2017-01-17T19:59:22.854Z",
"end": "2099-12-30T18:30:00Z",
"isPaused": false,
"hubName": "sampleazuredatafactory1_hub",
"pipelineMode": "Scheduled"
}

Check the Row Count before the Execution

```
SELECT GETDATE(), COUNT(*) from [dbo].[DestinationTarget]
```

![Check the Row Count before the Execution](image)

Check the Row Count after the Execution

```
SELECT GETDATE(), COUNT(*) from [dbo].[DestinationTarget]
```

![Check the Row Count after the Execution](image)
Click the Monitor and Manage the Pipeline to verify the same

If you want to check a SECOND WAY, then Click Copy Data (PREVIEW), as per the data Wizard, Click Next in the Properties Pane
Select Source as Azure SQL Database
Now create a Connection from the Azure SQL Database.
Select the Corresponding View

Select tables from which to copy the data (or) use a custom query

**EXISTING TABLES**

- [ ] [SalesLT].ProductModelProductDescription
- [ ] [SalesLT].SalesOrderDetail
- [ ] [SalesLT].SalesOrderHeader
- [ ] [SalesLT].ProductModel
- [ ] [SalesLT].ProductAndDescription
- [ ] [SalesLT].ProductModelCatalogDescription
- [ ] [SalesLT].Source-Product_SalesOrderDetail

**USE QUERY**

Filter by table name

Show views

Previous  Next

Create the Destination Connection

Specify the destination data store for the copy task. You can use an existing data store connection (or) specify a new data store. Click HERE to suggest new copy destinations or give comments.

FROM EXISTING CONNECTIONS  CONNECT TO A DATA STORE

Azure Blob Storage  Azure Data Lake Store  Azure DocumentDB  Azure SQL Database
Now Confirm the Connection and Dataset

Copy Data (SampleAzureDataFactory1)

1. Properties
   Recurring copy

2. Source
   Azure SQL Database

3. Destination
   Connection
   Dataset

4. Performance
   Parallel copy

5. Summary

Connection name (required)
Destination-SQLAzure-hic

Server / database selection method (required)
From Azure subscriptions

Azure subscription (required)
Select all

Server name (required)
adventureworksserver1

Database name (required)
SampleAdvWorksDB

User name (required)
advuser

Password (required)
************

Previous  Next
Confirm the Schema Mapping

Table mapping
For each table you have selected to copy in the source data store, select a corresponding table in the destination data store or specify the stored procedure to run at the destination.

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>[SalesLT]Source-Product_SalesOrderDetail</td>
<td>[dbo].[DestinationTarget]</td>
</tr>
</tbody>
</table>

Schema mapping
Choose how source and destination columns are mapped.

Source: [SalesLT]Source-Product_SalesOrderDetail

Destination: [dbo].[DestinationTarget]

[SalesOrderID (Int32)] ➔ [dbo].[DestinationTarget].SalesOrderID (Int32) required

OrderQty (Int16) ➔ [dbo].[DestinationTarget].OrderQty (Int16) required

ProductName (String) ➔ [dbo].[DestinationTarget].ProductName (String) required

Repeatability settings
Method: None
Take the Default Values in the Performance Section and Click Next and Click finish