Microsoft

Azure Data Factory

Visual Data Flow

Limited Preview December 2018

Visual Data Flow Authoring

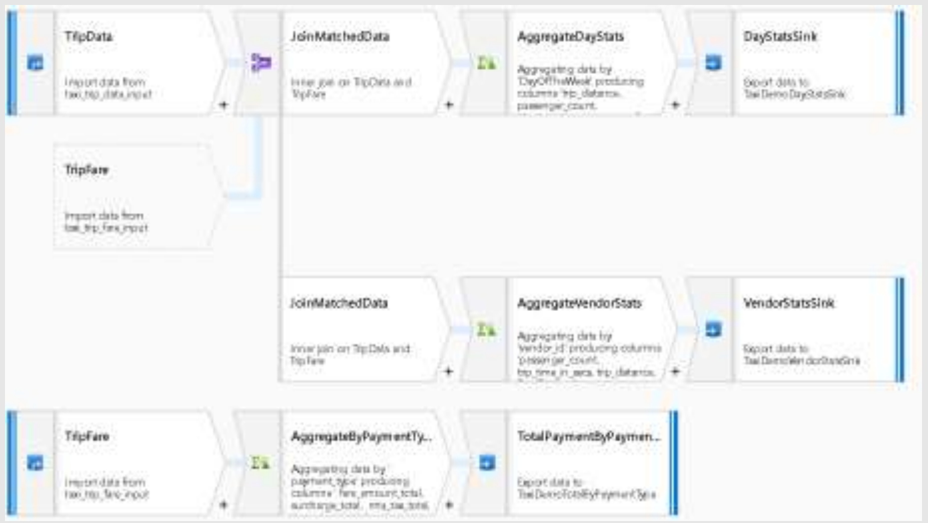
- Transform Data, At Scale, in the Cloud, Zero-Code
  - Cloud-first, scale-out ELT
  - Code-free dataflow pipelines
- Serverless scale-out transformation execution engine
- Maximum Productivity for Data Engineers
  - Does NOT require understanding of Spark / Scala / Python / Java
- Resilient Data Transformation Flows
  - Built for big data scenarios with unstructured data requirements
  - Operationalize with Data Factory scheduling, control flow and monitoring

Visual Data Flow Key Tenets

- Visual "Data Flow Builder" / "Data Mapping"
- Extensible through scripting and expressions
- Data Flow can be embedded into ISV / SaaS apps
  - Embed UI
  - Embed Parameterize Data Flows
- A graphical UI for building data transformation routines on Spark
- Built for resiliency and operationalized environments

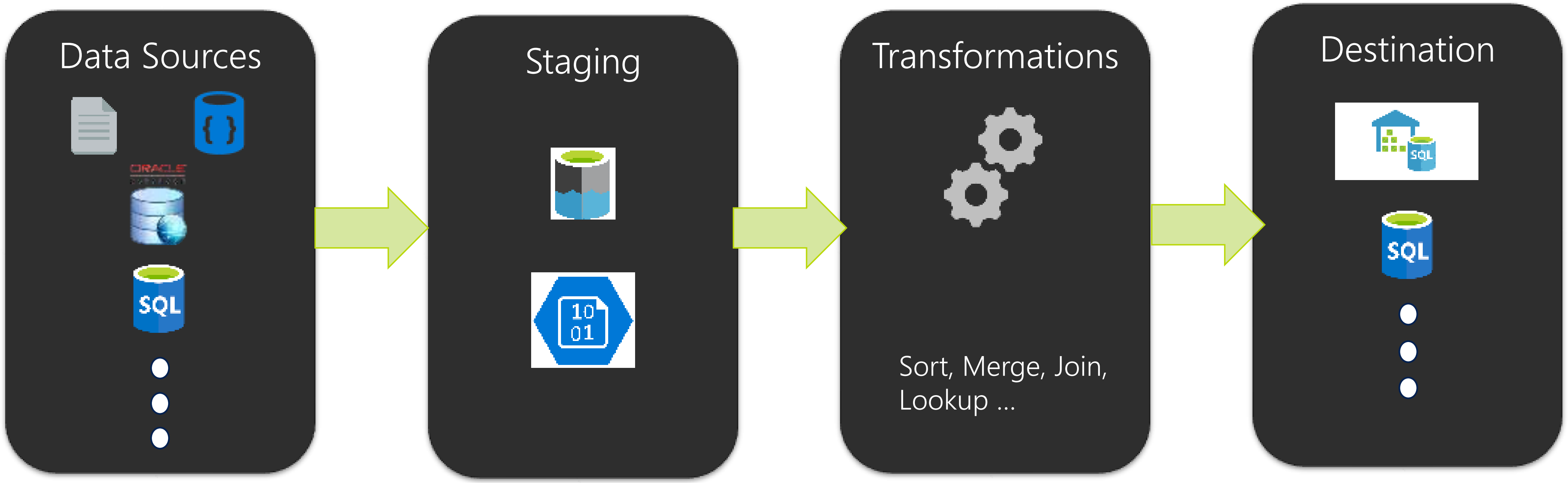
Code-free Data Transformation At Scale

- Does not require understanding of Spark, Big Data Execution Engines, Clusters, Scala, Python …
- Focus on building business logic and data transformation
  - Data cleansing
  - Aggregation
  - Data conversions
  - Data prep
  - Data exploration


TripData
Import data from taxi_trip_data_input
JoinMatchedData
Inner join on TripData and TripFare
AggregateDayStats
Aggregating data by 'DayOfTheWeek' producing columns 'trip_distance, passenger_count,
DayStatsSink
Export data to TaxiDemoDayStatsSink
TripFare
Import data from taxi_trip_fare_input
JoinMatchedData
Inner join on TripData and TripFare
AggregateVendorStats
Aggregating data by 'vendor_id' producing columns 'passenger_count, trip_time_in_secs, trip_distance,
VendorStatsSink
Export data to TaxiDemoVendorStatsSink
TripFare
Import data from taxi_trip_fare_input
AggregateByPaymentTy...
Aggregating data by 'payment_type' producing columns ' fare_amount_total, surcharge_total, mta_tax_total,
TotalPaymentByPaymen...
Export data to TaxiDemoTotalByPaymentType

… not …

ADF Data Flow Workstream



- Explicit user action
- User places data source(s) on design surface, from toolbox
- Select explicit sources

- Implicit/Explicit
- Data Lake staging area as default
- User does not need to configure this manually
- Advanced feature to set staging area options
- File Formats / Types (Parquet, JSON, txt, CSV …)

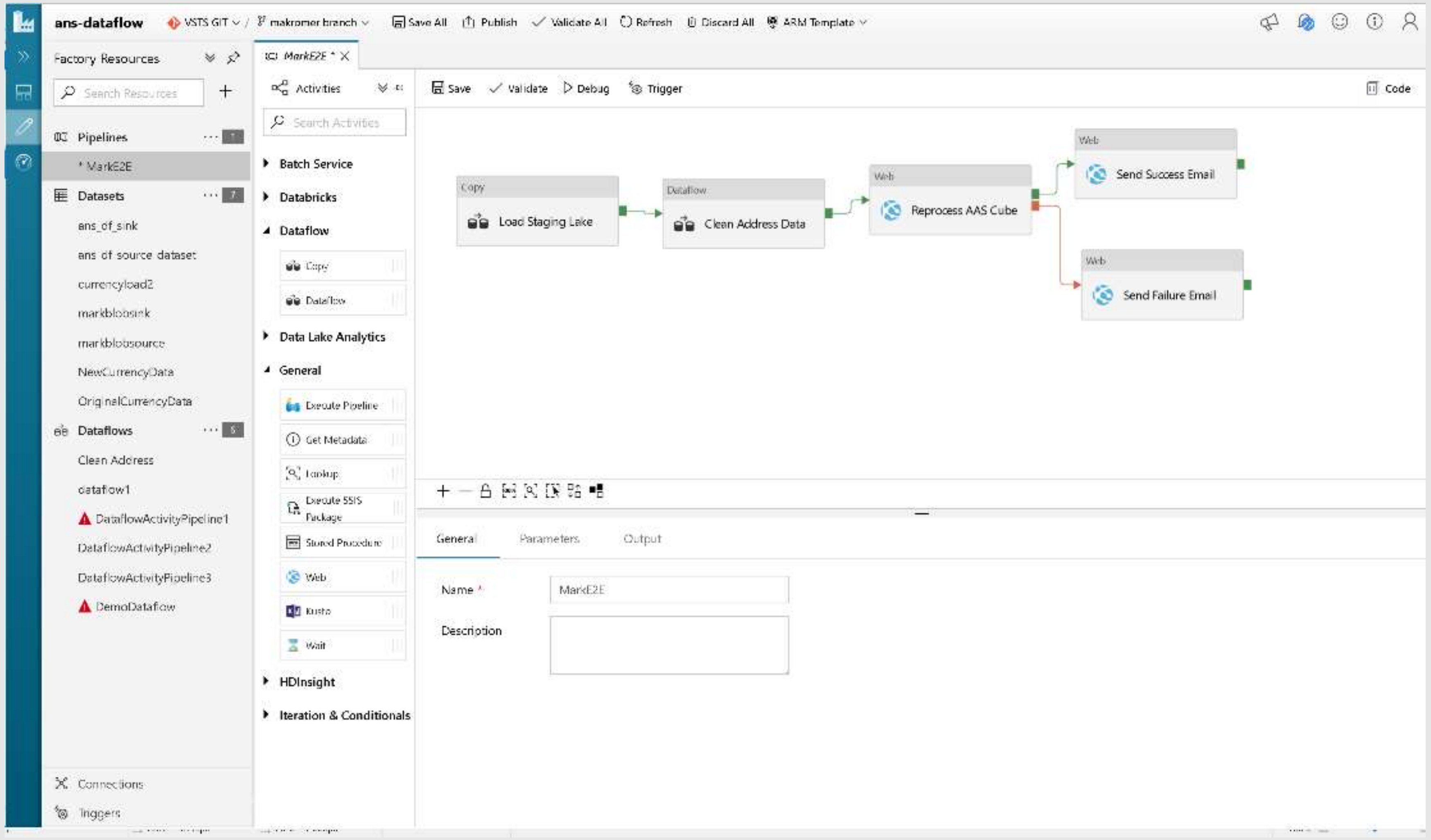
- Explicit user action
- User places transformations on design surface, from toolbox
- User must set properties for transformation steps and step connectors

- Explicit user action
- User chooses destination connector(s)
- User sets connector property options

Data Flow Limited Preview Support & SLAs

- Azure SLAs are NA for preview services (private or public preview) until GA of the service.
- Limited Preview Support
    - Handled directly with the Azure Engineering team via adfdataflowext@microsoft.com.
- Sign-up for ADF Data Flow service
    - http://aka.ms/dataflowpreview
    - Microsoft Azure must whitelist your subscription ID to turn on the feature for you
- Public Preview Support
    - Normal Azure customer service channels

ADF Pipeline Execution of a Data Flow Activity

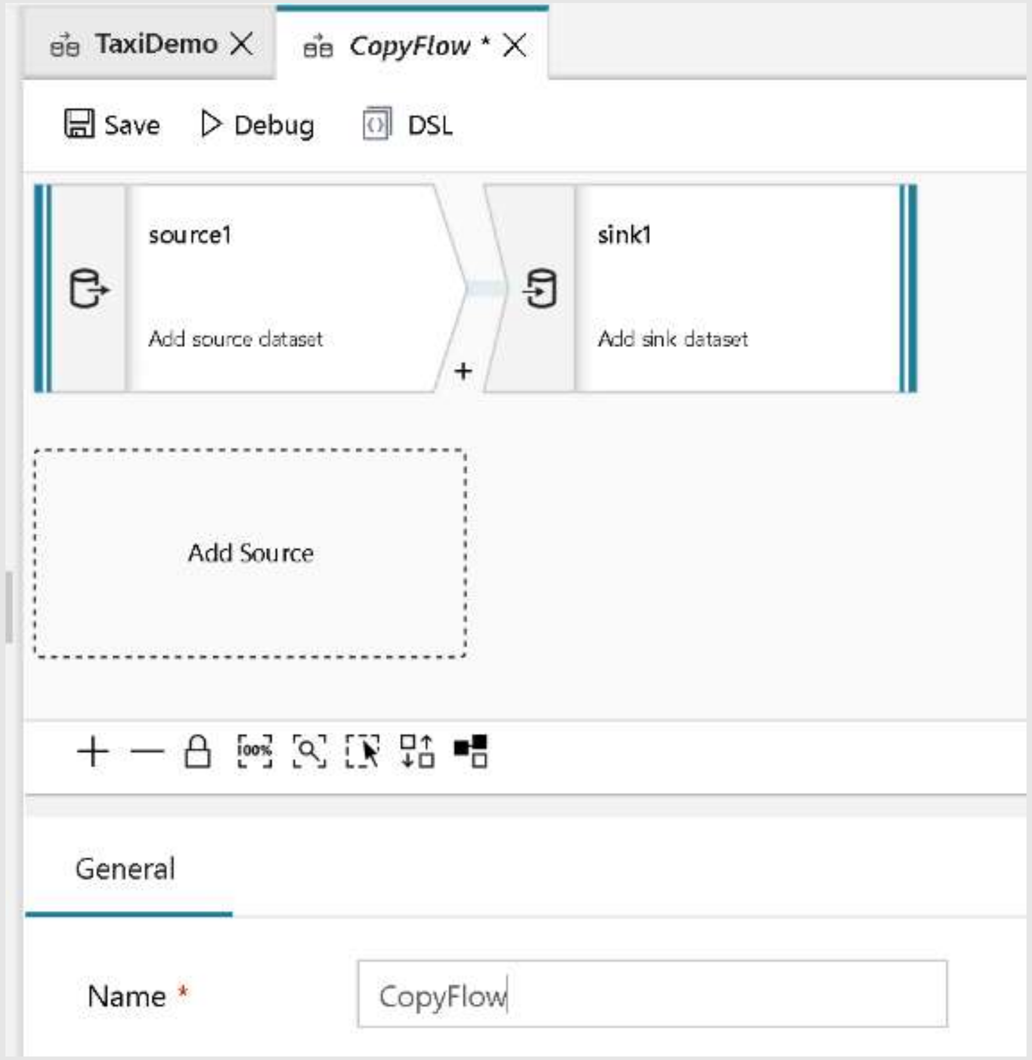


Name * MarkE2E

Description

- Design code-free ETL workflows
- Copy data from on-prem, other clouds and Azure
- Stage data for transformation
- Build visual data transformations
- Schedule triggers for your pipeline execution
- Monitor processes and configure alerts
- All within ADF

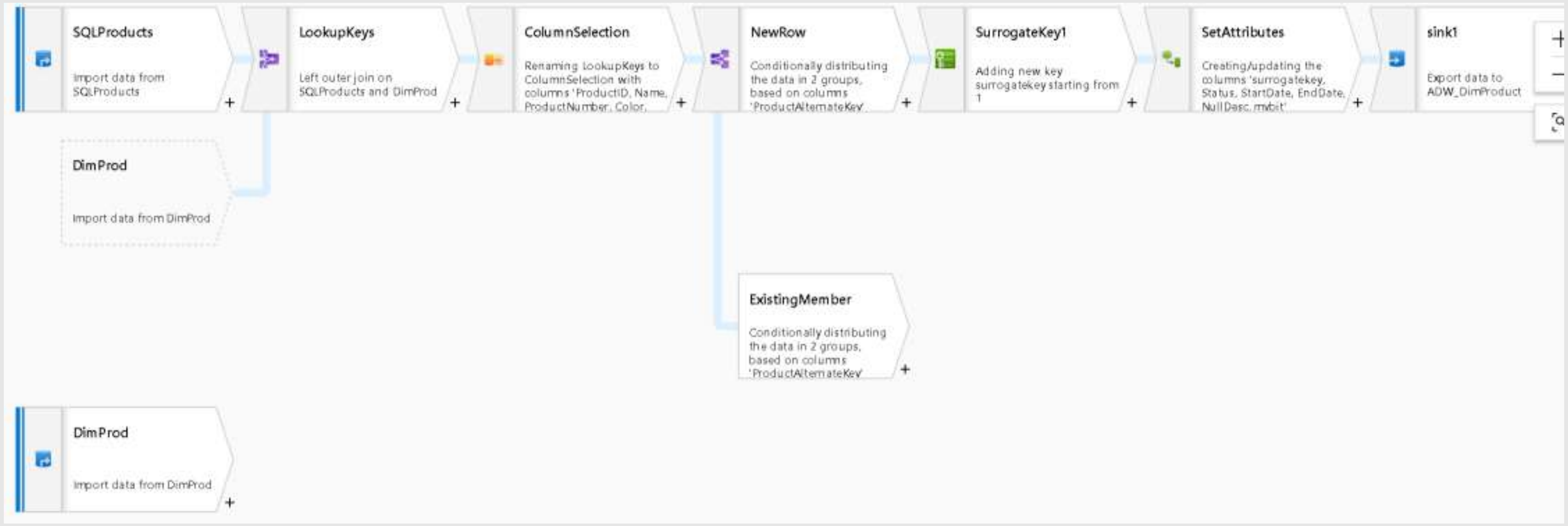
Simple Copy Flow



Name * CopyFlow

- ADF Data Flow is a guided construction process
- Begin by defining the Datasets for your Source and Sink
- Add Transformations to each node in your data flow
- Or simply copy from source to sink with no transformation
- Map columns and fields along the way

Slowly Changing Dimension Scenario



SQLProducts
Import data from SQLProducts

DimProd
Import data from DimProd

LookupKeys
Left outer join on SQLProducts and DimProd

ColumnSelection
Renaming LookupKeys to ColumnSelection with columns 'ProductID, Name, ProductNumber, Color,

NewRow
Conditionally distributing the data in 2 groups, based on columns 'ProductAlternateKey'

ExistingMember
Conditionally distributing the data in 2 groups, based on columns 'ProductAlternateKey'

SurrogateKey1
Adding new key surrogatekey starting from 1

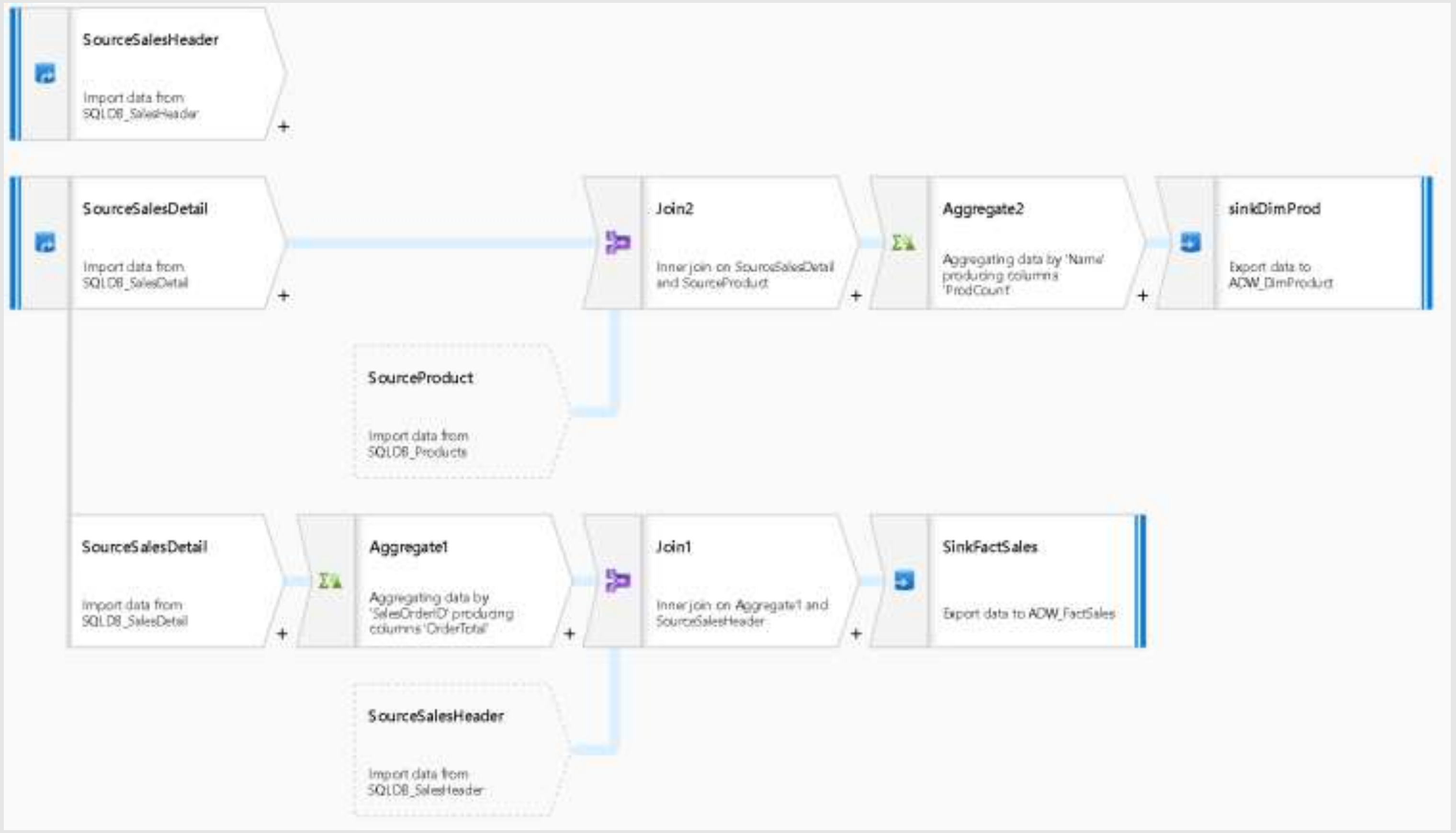
SetAttributes
Creating/updating the columns 'surrogatekey, Status, StartDate, EndDate, NullDesc, mybit'

sink1
Export data to ADW_DimProduct

DimProd
Import data from DimProd

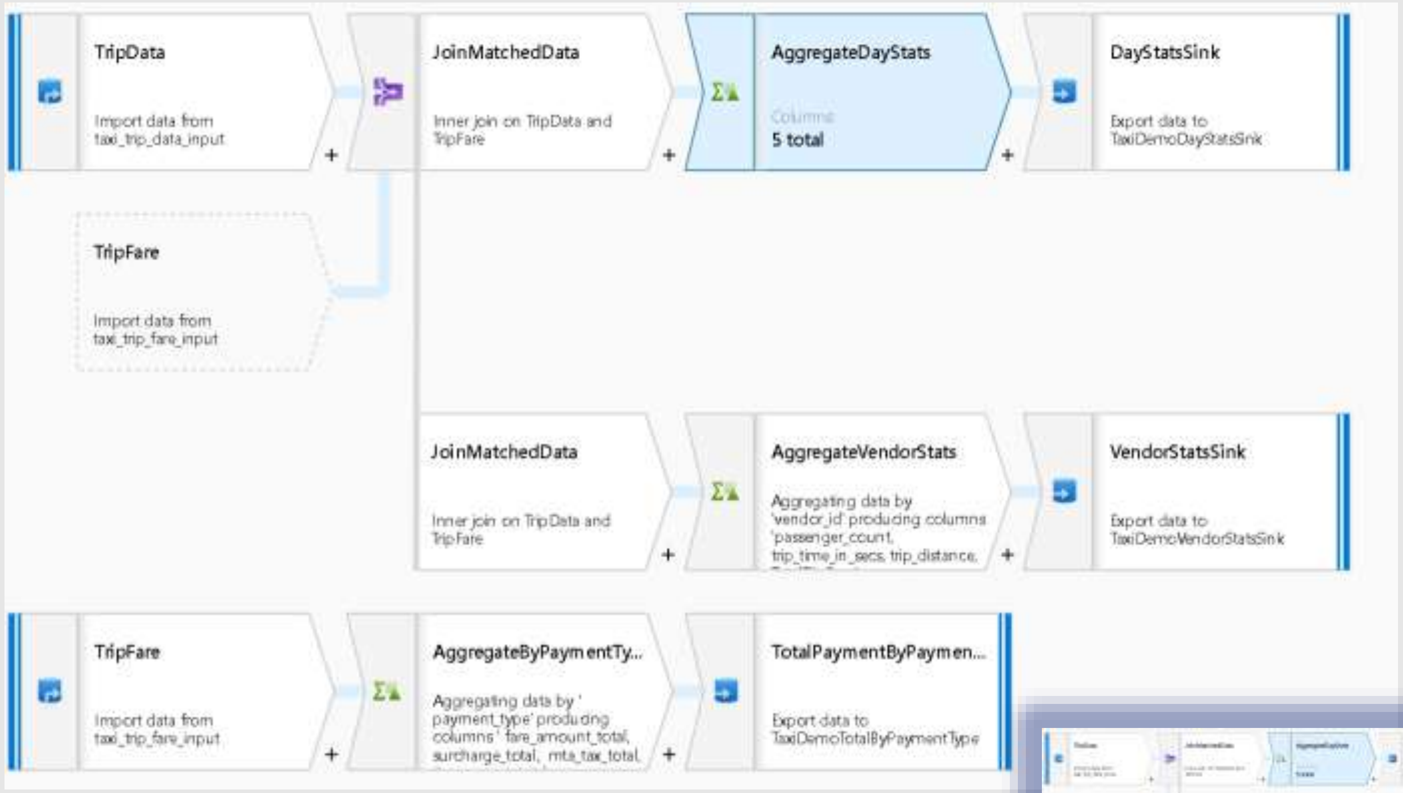
- Common DW pattern to manage changing attributes to dimension members
- Graphically build code-free SCD ETL pattern to load your data warehouse
- Connect directly to Azure SQL DB and Azure SQL DW
- Use Lookup, Surrogate Key, Derived Column and Select transforms

Load Star Schema DW Scenario

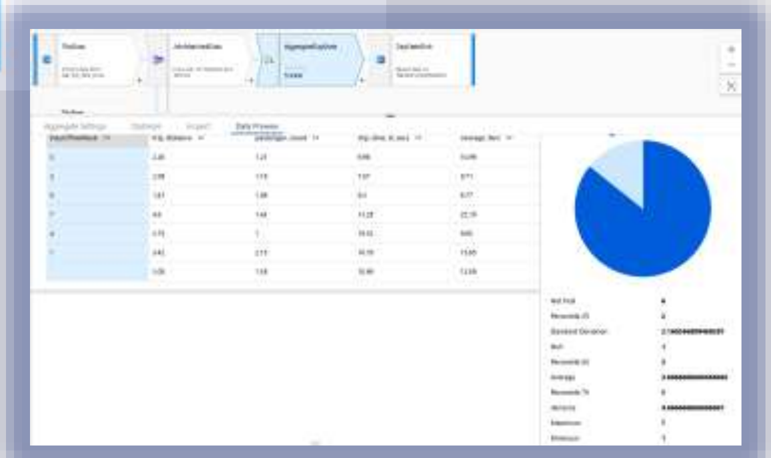

SourceSalesHeader
Import data from SQLDB_SalesHeader
SourceSalesDetail
Import data from SQLDB_SalesDetail
Join2
Inner join on SourceSalesDetail and SourceProduct
Aggregate2
Aggregating data by 'Name' producing columns 'ProdCount'
sinkDimProd
Export data to ADW_DimProduct
SourceProduct
Import data from SQLDB_Products
SourceSalesDetail
Import data from SQLDB_SalesDetail
Aggregate1
Aggregating data by 'SalesOrderID' producing columns 'OrderTotal'
Join1
Inner join on Aggregate1 and SourceSalesHeader
SinkFactSales
Export data to ADW_FactSales
SourceSalesHeader
Import data from SQLDB_SalesHeader

- Classic ETL pattern is easy to build in ADF's code-free Data Flow visual data transformation environment
- Add Aggregate transforms to produce calculations that you store in your analytical database schema
- Use Join transform to combine data from multiple data sources and data streams inside your data flow
- Land your data in your Lake folders or direct to Azure SQL DW

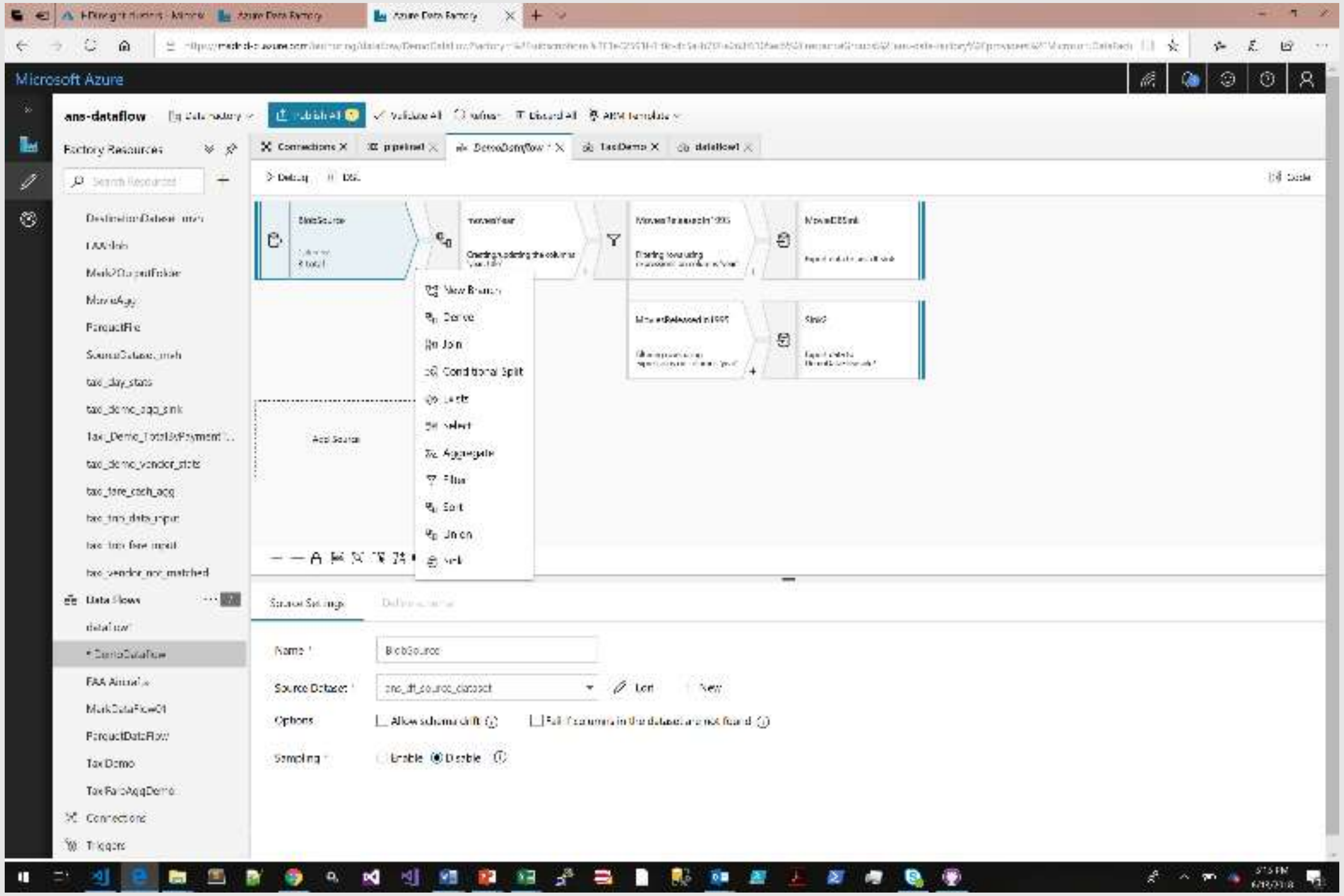
Data Lake Data Science Scenario


TripData
Import data from taxi_trip_data_input
JoinMatchedData
Inner join on TripData and TripFare
AggregateDayStats
Columns
5 total
DayStatsSink
Export data to TaxiDemoDayStatsSink
TripFare
Import data from taxi_trip_fare_input
JoinMatchedData
Inner join on TripData and TripFare
AggregateVendorStats
Aggregating data by 'vendor_id' producing columns 'passenger_count, trip_time_in_secs, trip_distance,
VendorStatsSink
Export data to TaxiDemoVendorStatsSink
TripFare
Import data from taxi_trip_fare_input
AggregateByPaymentTy...
Aggregating data by ' payment_type' producing columns ' fare_amount_total, surcharge_total, mta_tax_total,
TotalPaymentByPaymen...
Export data to TaxiDemoTotalByPaymentType

- ADF supports building visual data transformations against your data directly in Data Lake locations (i.e. Azure Blob Store, Azure Data Lake Store)
- Built-in handling of schema drift for frequent changes in data lake file formats, columns, and data types
- Perform data exploration and data profiling across your data lake in ADF Data Flow win interactive debug data preview



Build your logical data flows adding data transformations in a guided experience



Source Settings

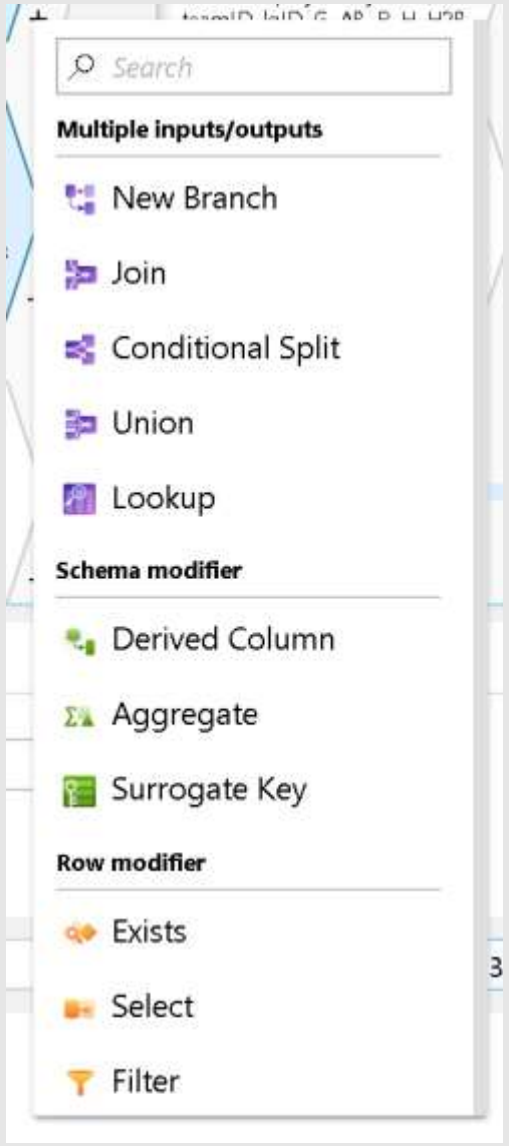
Name *: BlobSource

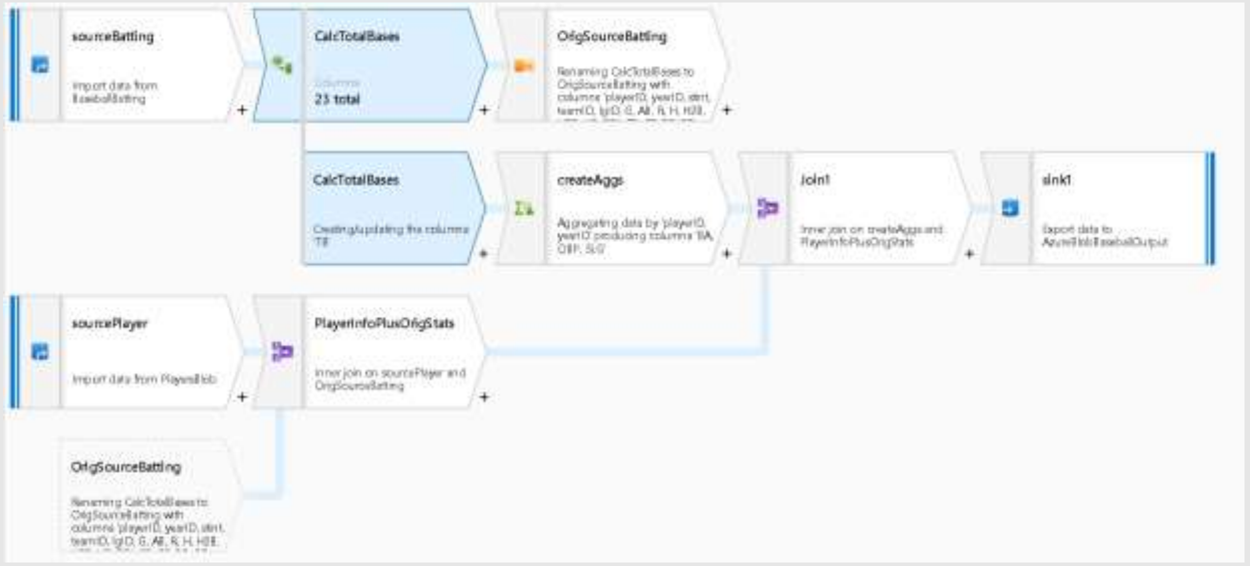
Source Dataset *: ans_df_source_dataset

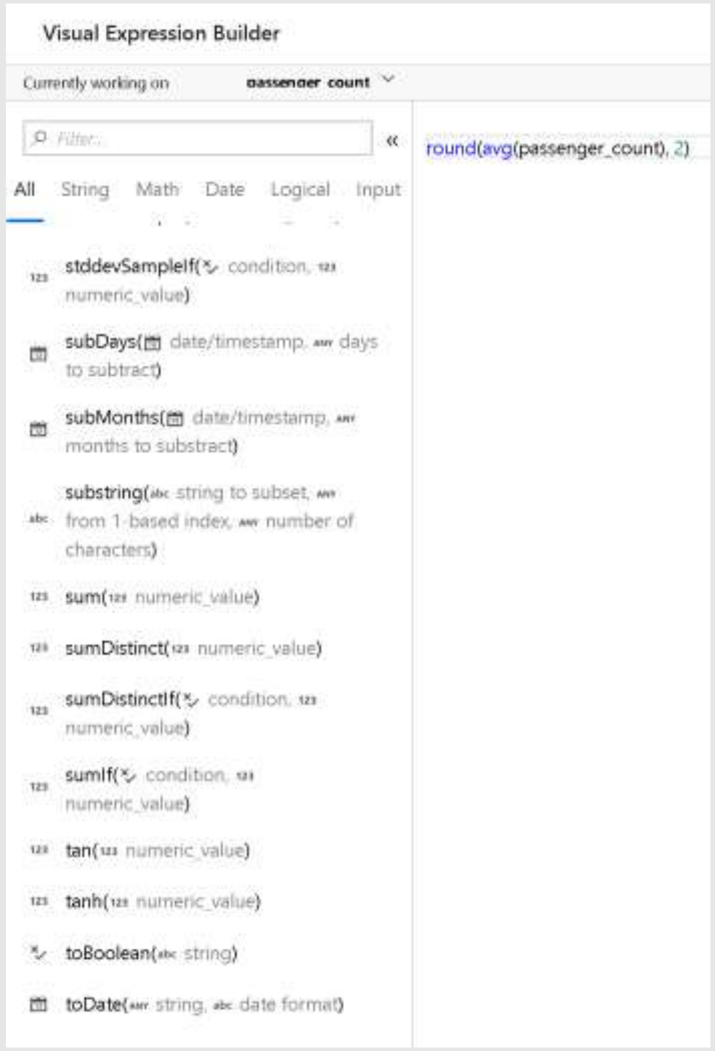
Options: ☐ Allow schema drift ☐ Fail if columns in the dataset are not found

Sampling *: Enable [✓ Disable]

Microsoft Azure Data Factory Continues to Extend Data Flow Library with a Rich Set of Transformations and Expression Functions




sourceBatting
Import data from BaseballBatting
CalcTotalBases
Columns
23 total
OrigSourceBatting
Renaming CalcTotalBases to OrigSourceBatting with columns 'playerID, yearID, stint, teamID, lgID, G, AB, R, H, H2B,
CalcTotalBases
Creating/updating the columns 'TB'
createAggs
Aggregating data by 'playerID, yearID' producing columns 'BA, OBP, SLG'
Join1
Inner join on createAggs and PlayerInfoPlusOrigStats
sink1
Export data to AzureBlobBaseballOutput
sourcePlayer
Import data from PlayersBlob
PlayerInfoPlusOrigStats
Inner join on sourcePlayer and OrigSourceBatting
OrigSourceBatting
Renaming CalcTotalBases to OrigSourceBatting with columns 'playerID, yearID, stint, teamID, lgID, G, AB, R, H, H2B,


Visual Expression Builder

Currently working on passenger_count

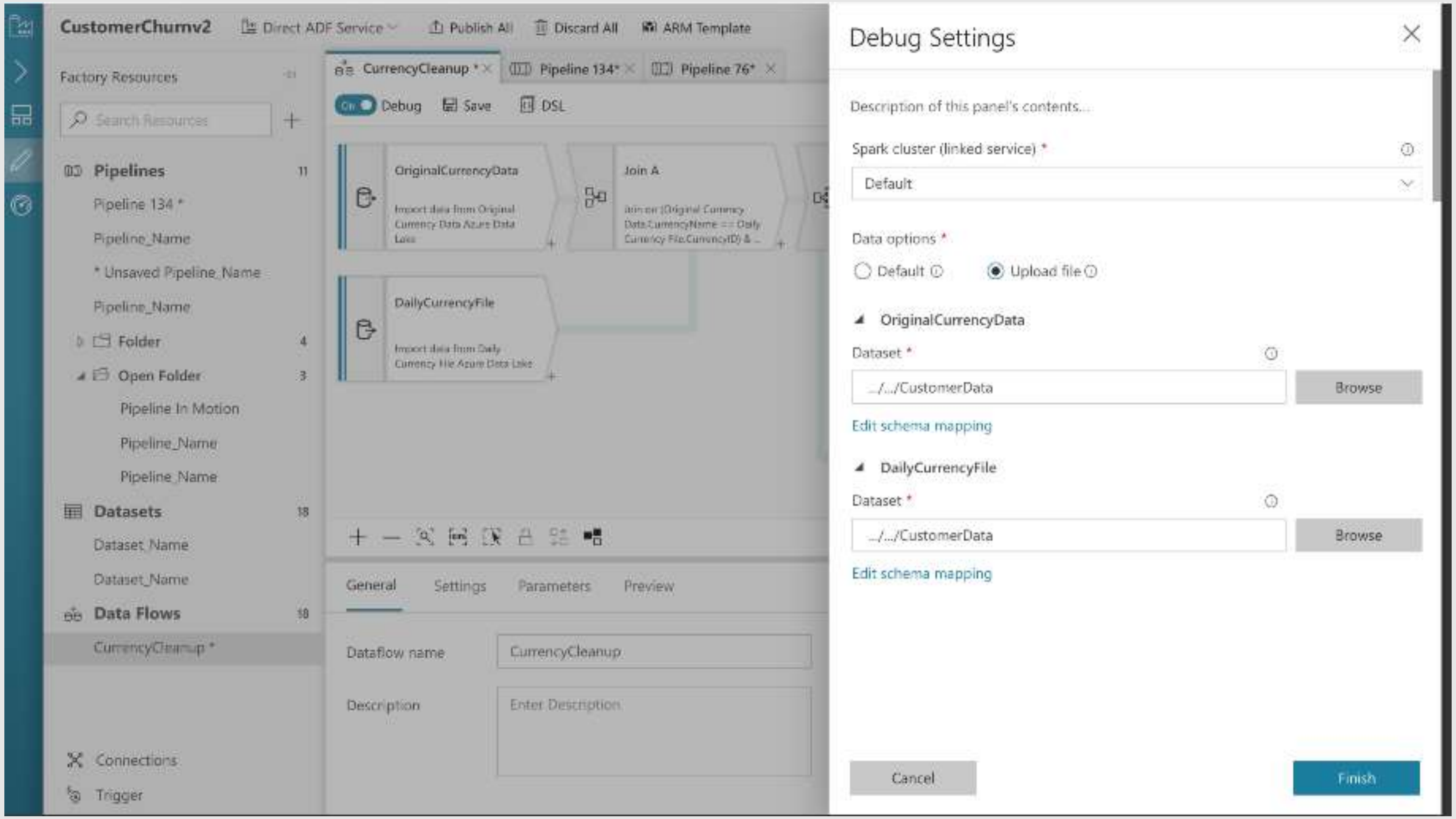
Filter...

All String Math Date Logical Input

```
round(avg(passenger_count), 2)
```

- stddevSampleIf(condition, numeric_value)
- subDays(date/timestamp, days to subtract)
- subMonths(date/timestamp, months to substract)
- substring(string to subset, from 1-based index, number of characters)
- sum(numeric_value)
- sumDistinct(numeric_value)
- sumDistinctIf(condition, numeric_value)
- sumIf(condition, numeric_value)
- tan(numeric_value)
- tanh(numeric_value)
- toBoolean(string)
- toDate(string, date format)

Switch to Debug Mode and select sample data to work with for debugging



CustomerChurnv2 | Direct ADF Service | Publish All | Discard All | ARM Template

Factory Resources

Search Resources

Pipelines 11
- Pipeline 134 *
- Pipeline_Name
- * Unsaved Pipeline_Name
- Pipeline_Name
- Folder 4
- Open Folder 3
  - Pipeline In Motion
  - Pipeline_Name
  - Pipeline_Name

Datasets 18
- Dataset_Name
- Dataset_Name

Data Flows 18
- CurrencyCleanup *

Connections

Trigger

CurrencyCleanup * | Pipeline 134* | Pipeline 76*

On Debug | Save | DSL

OriginalCurrencyData
Import data from Original Currency Data Azure Data Lake

Join A
Join on (Original Currency Data.CurrencyName == Daily Currency File.CurrencyID) & ...

DailyCurrencyFile
Import data from Daily Currency File Azure Data Lake

General | Settings | Parameters | Preview

Dataflow name: CurrencyCleanup

Description: Enter Description

Debug Settings

Description of this panel's contents...

Spark cluster (linked service) *: Default

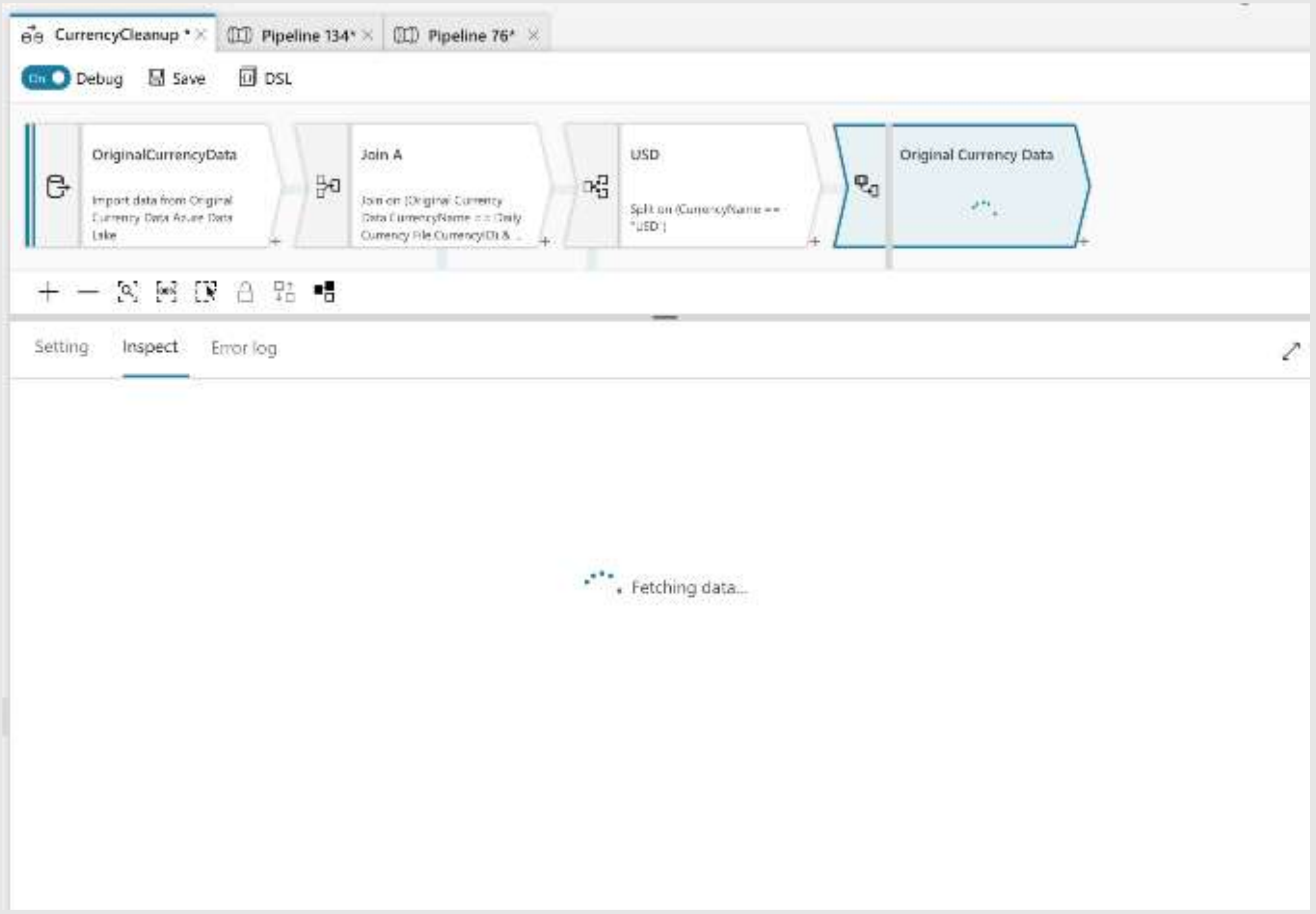
Data options *: Default [✓ Upload file]

OriginalCurrencyData
Dataset *: .../.../CustomerData | Browse
Edit schema mapping

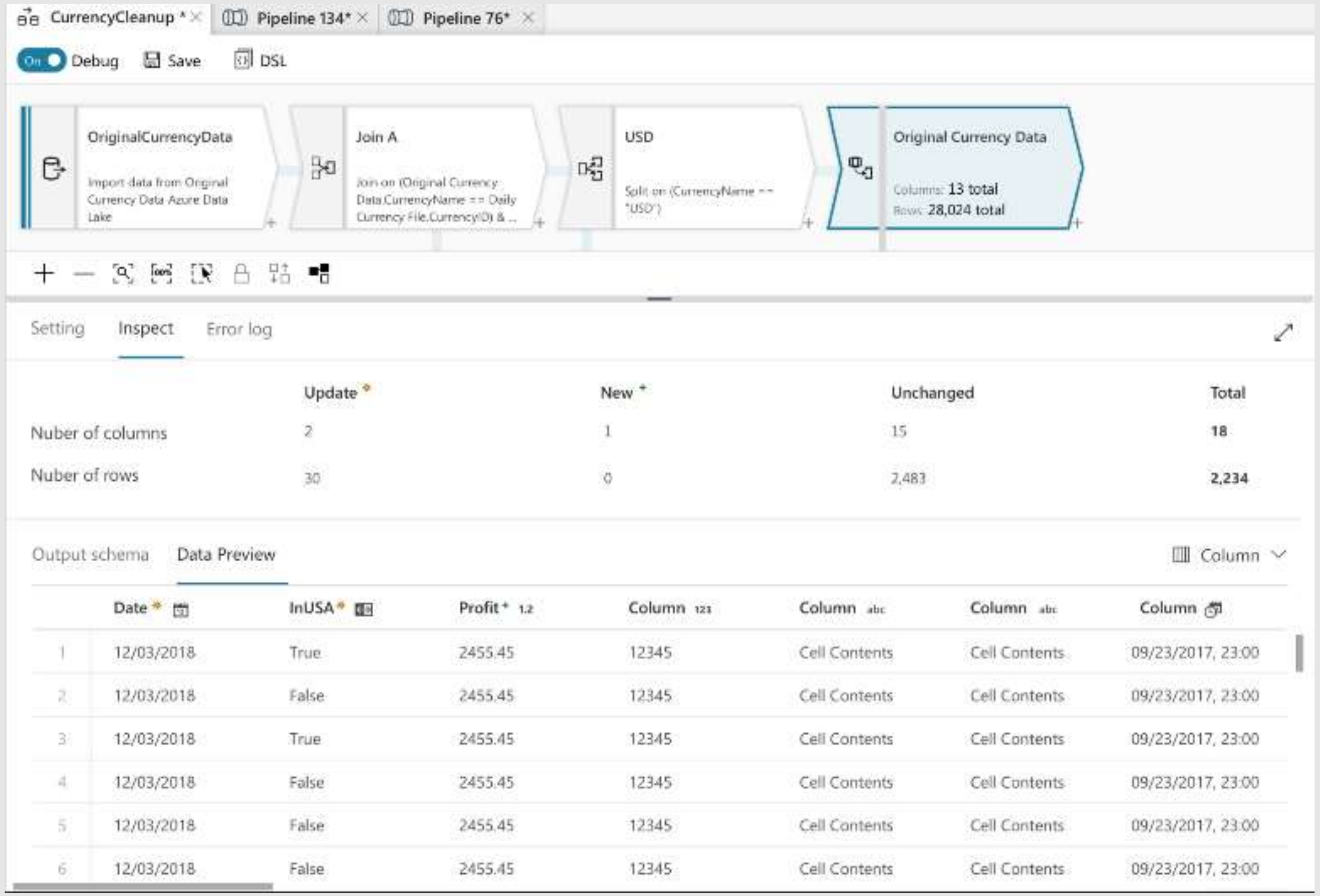
DailyCurrencyFile
Dataset *: .../.../CustomerData | Browse
Edit schema mapping

Cancel | Finish

Debug mode provides row-level context and visible results in inspector pane


CurrencyCleanup *
Pipeline 134*
Pipeline 76*
On
Debug
Save
DSL
OriginalCurrencyData
Import data from Original Currency Data Azure Data Lake
Join A
Join on (Original Currency Data.CurrencyName == Daily Currency File.CurrencyDt & ...
USD
Split on (CurrencyName == "USD")
Original Currency Data
Setting
Inspect
Error log
Fetching data...

Debug mode provides row-level context and visible results in inspector pane



OriginalCurrencyData
Import data from Original Currency Data Azure Data Lake

Join A
Join on (Original Currency Data.CurrencyName == Daily Currency File.CurrencyID) & ...

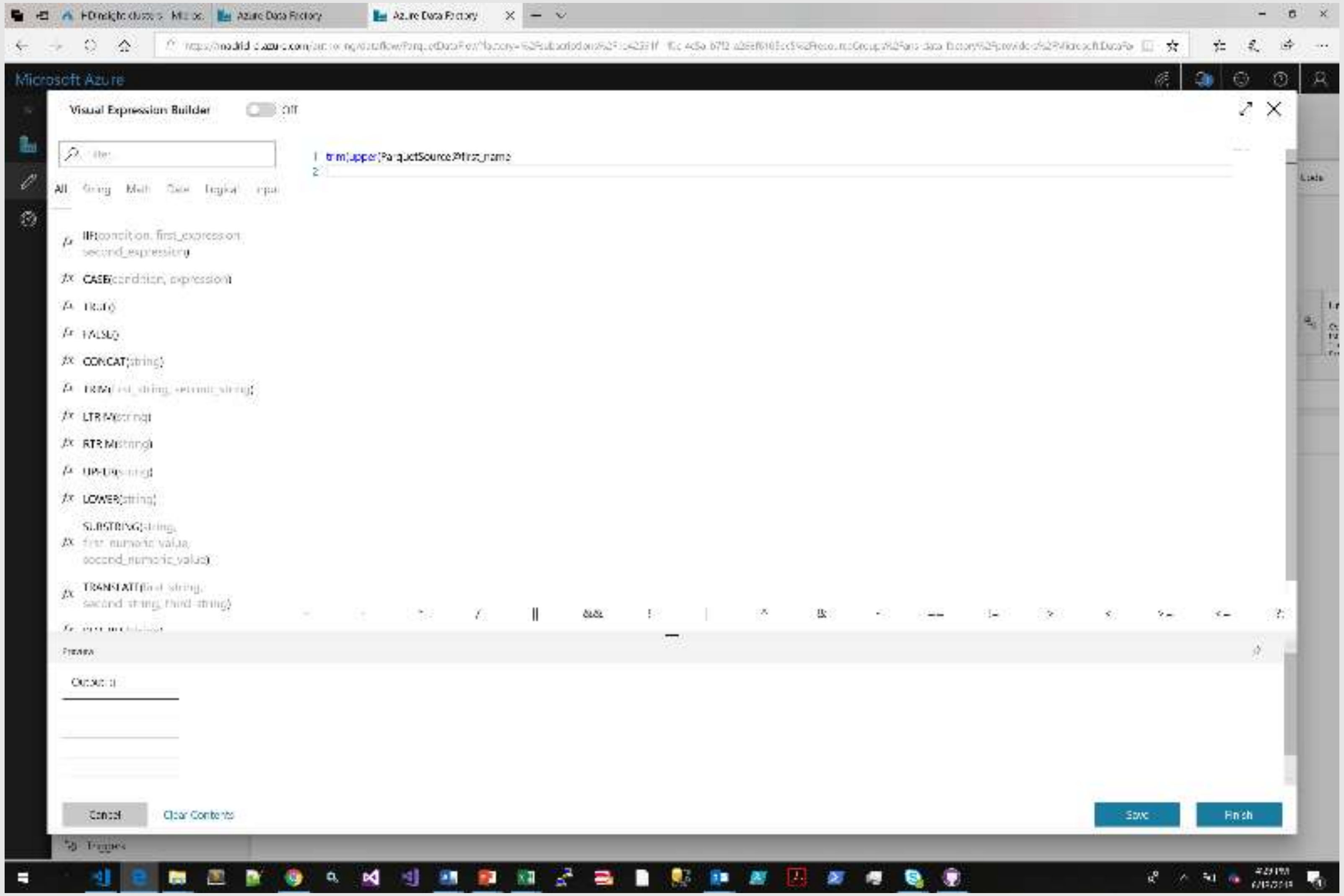
USD
Split on (CurrencyName == "USD")

Original Currency Data
Columns: 13 total
Rows: 28,024 total

	Update *	New *	Unchanged	Total
Nuber of columns	2	1	15	18
Nuber of rows	30	0	2,483	2,234

	Date *	InUSA *	Profit * 1.2	Column 123	Column abc	Column abc	Column
1	12/03/2018	True	2455.45	12345	Cell Contents	Cell Contents	09/23/2017, 23:00
2	12/03/2018	False	2455.45	12345	Cell Contents	Cell Contents	09/23/2017, 23:00
3	12/03/2018	True	2455.45	12345	Cell Contents	Cell Contents	09/23/2017, 23:00
4	12/03/2018	False	2455.45	12345	Cell Contents	Cell Contents	09/23/2017, 23:00
5	12/03/2018	False	2455.45	12345	Cell Contents	Cell Contents	09/23/2017, 23:00
6	12/03/2018	False	2455.45	12345	Cell Contents	Cell Contents	09/23/2017, 23:00

Interactive Expression Builder – Build data transform expressions, not Spark code



Microsoft Azure

Visual Expression Builder Off

```
trim(upper(ParquetSource@first_name
```

All String Math Date Logical

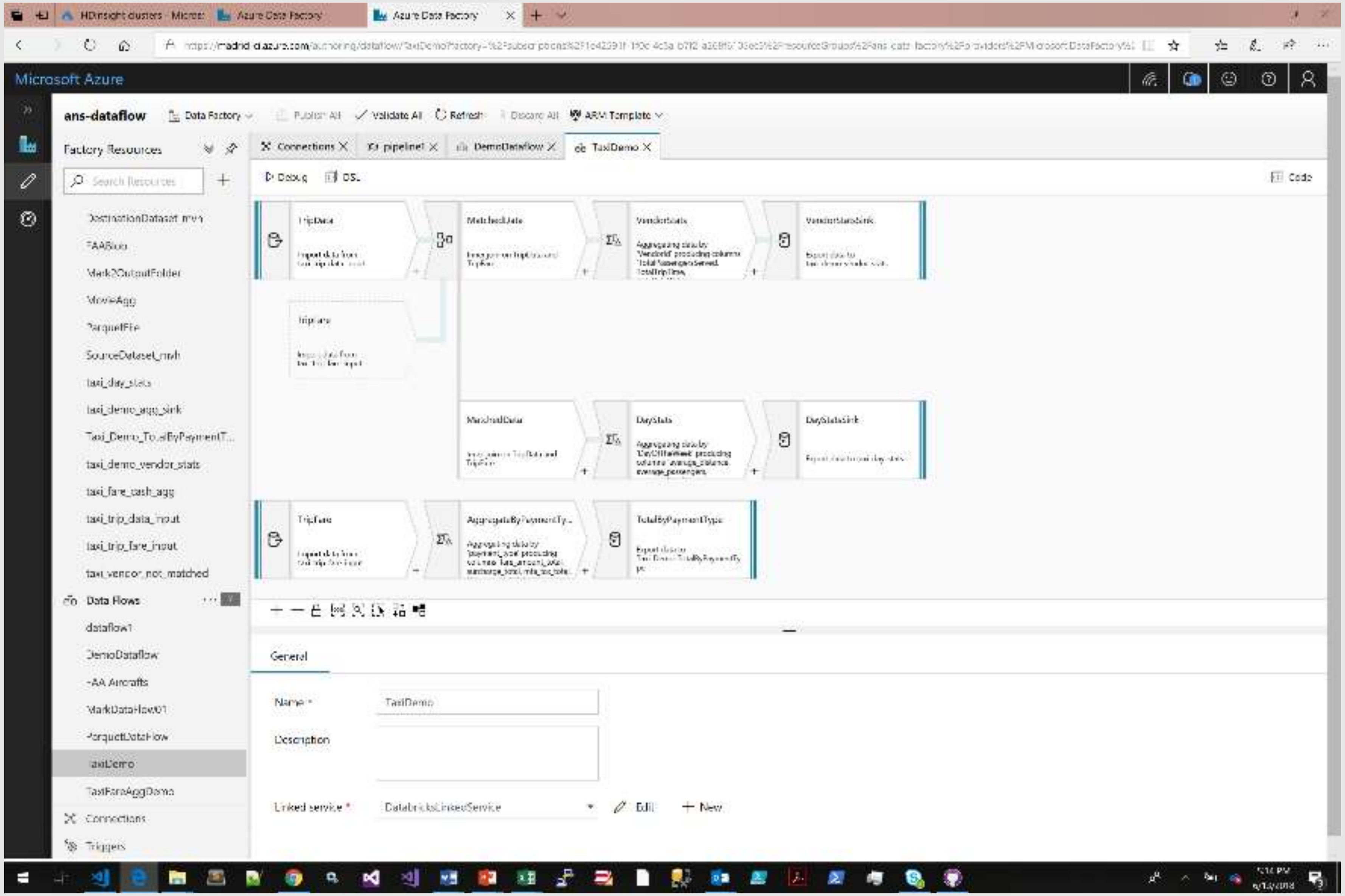
- IIF(condition, first_expression, second_expression)
- CASE(condition, expression)
- TRUE()
- FALSE()
- CONCAT(string)
- TRIM(first_string, second_string)
- LTRIM(string)
- RTRIM(string)
- UPPER(string)
- LOWER(string)
- SUBSTRING(string, first_numeric_value, second_numeric_value)
- TRANSLATE(first_string, second_string, third_string)

Preview

Output

Cancel Clear Contents Save Finish

Azure Data Factory Visual Data Flow

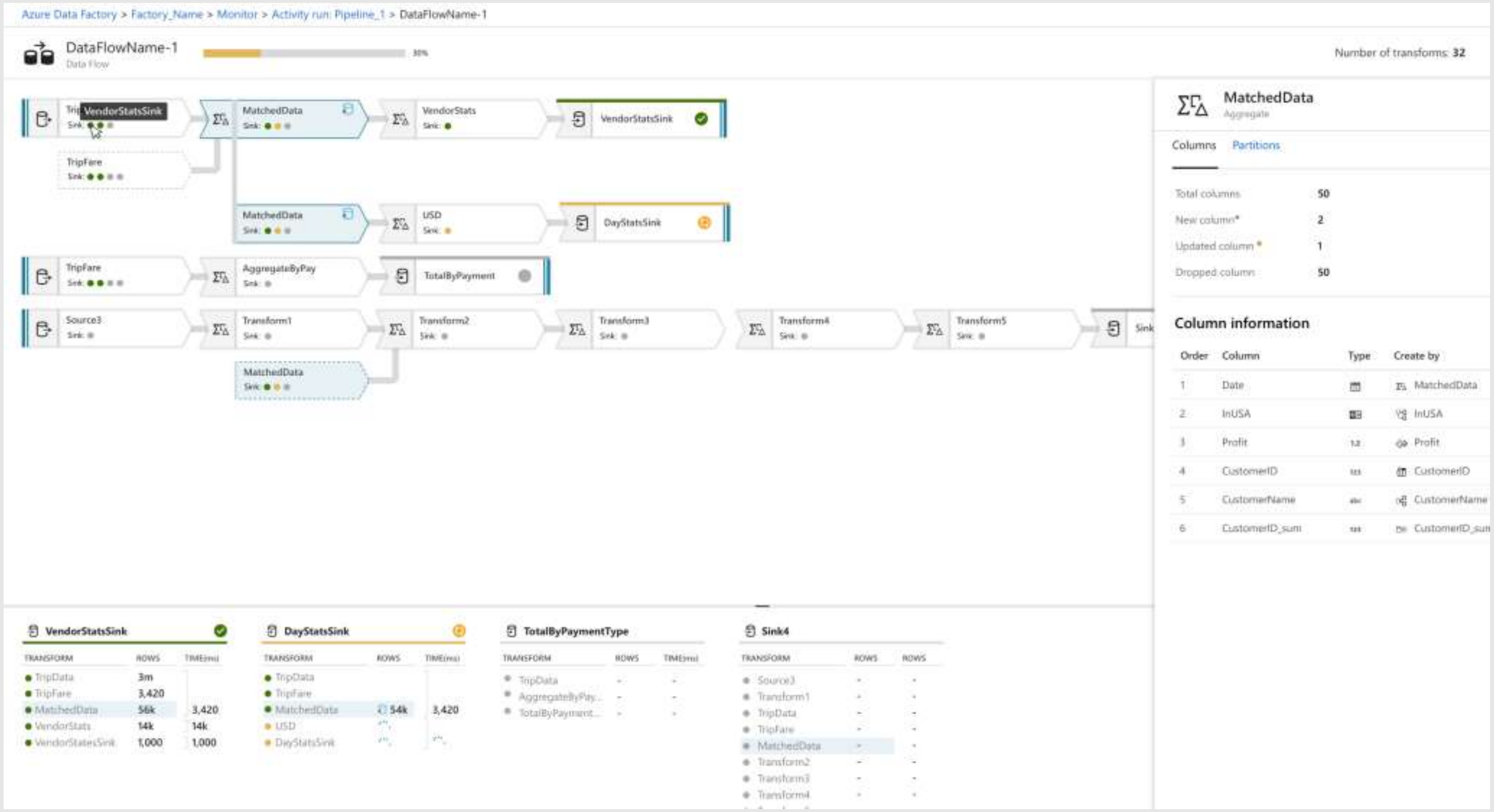


Name * TaxiDemo

Description

Linked service * DatabricksLinkedService Edit + New

Deep Monitoring Introspection of Data Transformations



Number of transforms: **32**

MatchedData
Aggregate

Columns | Partitions

Total columns	50
New column*	2
Updated column *	1
Dropped column	50

Column information

Order	Column	Type	Create by
1	Date		MatchedData
2	InUSA		InUSA
3	Profit	1.2	Profit
4	CustomerID	123	CustomerID
5	CustomerName	abc	CustomerName
6	CustomerID_sum	123	CustomerID_sum

VendorStatsSink ✓

TRANSFORM	ROWS	TIME(ms)
● TripData	3m	
● TripFare	3,420	
● MatchedData	56k	3,420
● VendorStats	14k	14k
● VendorStatesSink	1,000	1,000

DayStatsSink

TRANSFORM	ROWS	TIME(ms)
● TripData		
● TripFare		
● MatchedData	54k	3,420
● USD		
● DayStatsSink		

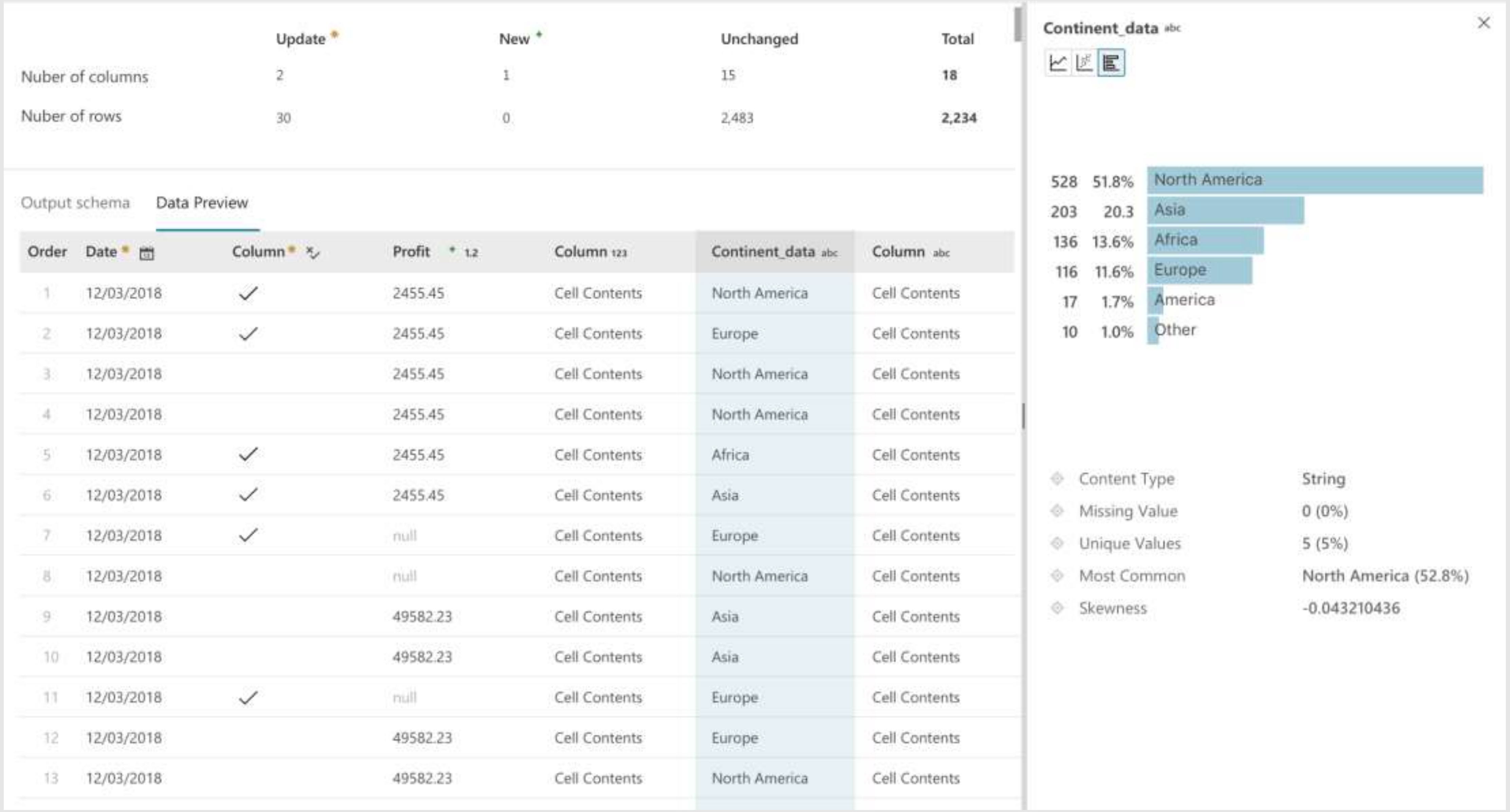
TotalByPaymentType

TRANSFORM	ROWS	TIME(ms)
● TripData	-	-
● AggregateByPay...	-	-
● TotalByPayment...	-	-

Sink4

TRANSFORM	ROWS	ROWS
● Source3	-	-
● Transform1	-	-
● TripData	-	-
● TripFare	-	-
● MatchedData	-	-
● Transform2	-	-
● Transform3	-	-
● Transform4	-	-

Debug Data Flows with Data Preview and Data Sampling



	Update *	New *	Unchanged	Total
Nuber of columns	2	1	15	**18**
Nuber of rows	30	0	2,483	**2,234**

Output schema   Data Preview

Order	Date *	Column *	Profit * 1.2	Column 123	Continent_data abc	Column abc
1	12/03/2018	✓	2455.45	Cell Contents	North America	Cell Contents
2	12/03/2018	✓	2455.45	Cell Contents	Europe	Cell Contents
3	12/03/2018		2455.45	Cell Contents	North America	Cell Contents
4	12/03/2018		2455.45	Cell Contents	North America	Cell Contents
5	12/03/2018	✓	2455.45	Cell Contents	Africa	Cell Contents
6	12/03/2018	✓	2455.45	Cell Contents	Asia	Cell Contents
7	12/03/2018	✓	null	Cell Contents	Europe	Cell Contents
8	12/03/2018		null	Cell Contents	North America	Cell Contents
9	12/03/2018		49582.23	Cell Contents	Asia	Cell Contents
10	12/03/2018		49582.23	Cell Contents	Asia	Cell Contents
11	12/03/2018	✓	null	Cell Contents	Europe	Cell Contents
12	12/03/2018		49582.23	Cell Contents	Europe	Cell Contents
13	12/03/2018		49582.23	Cell Contents	North America	Cell Contents

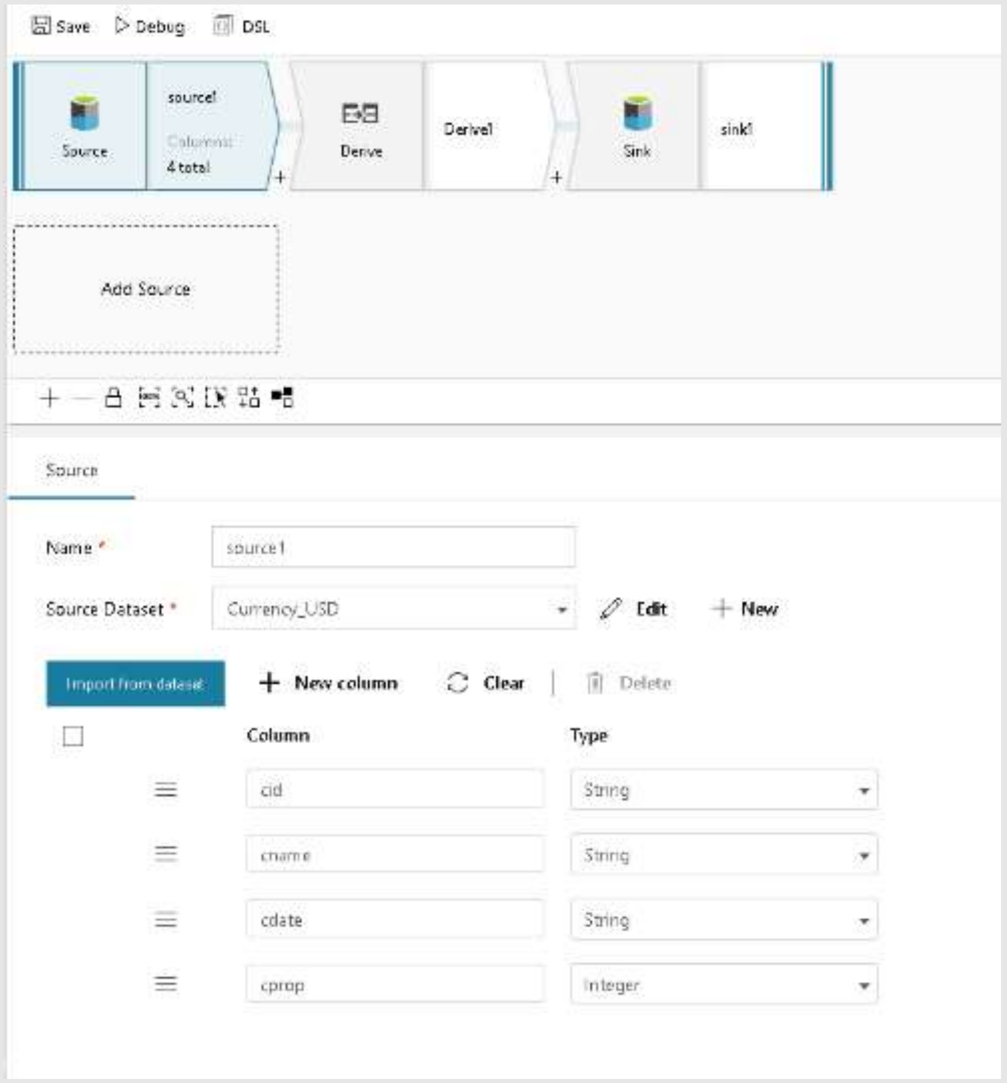
Continent_data abc

528	51.8%	North America
203	20.3	Asia
136	13.6%	Africa
116	11.6%	Europe
17	1.7%	America
10	1.0%	Other

Content Type	String
Missing Value	0 (0%)
Unique Values	5 (5%)
Most Common	North America (52.8%)
Skewness	-0.043210436

Build Resilient Data Flows with Schema Drift Handling

Data Engineer Defines Source will take ALL fields from the source file with flexible schema

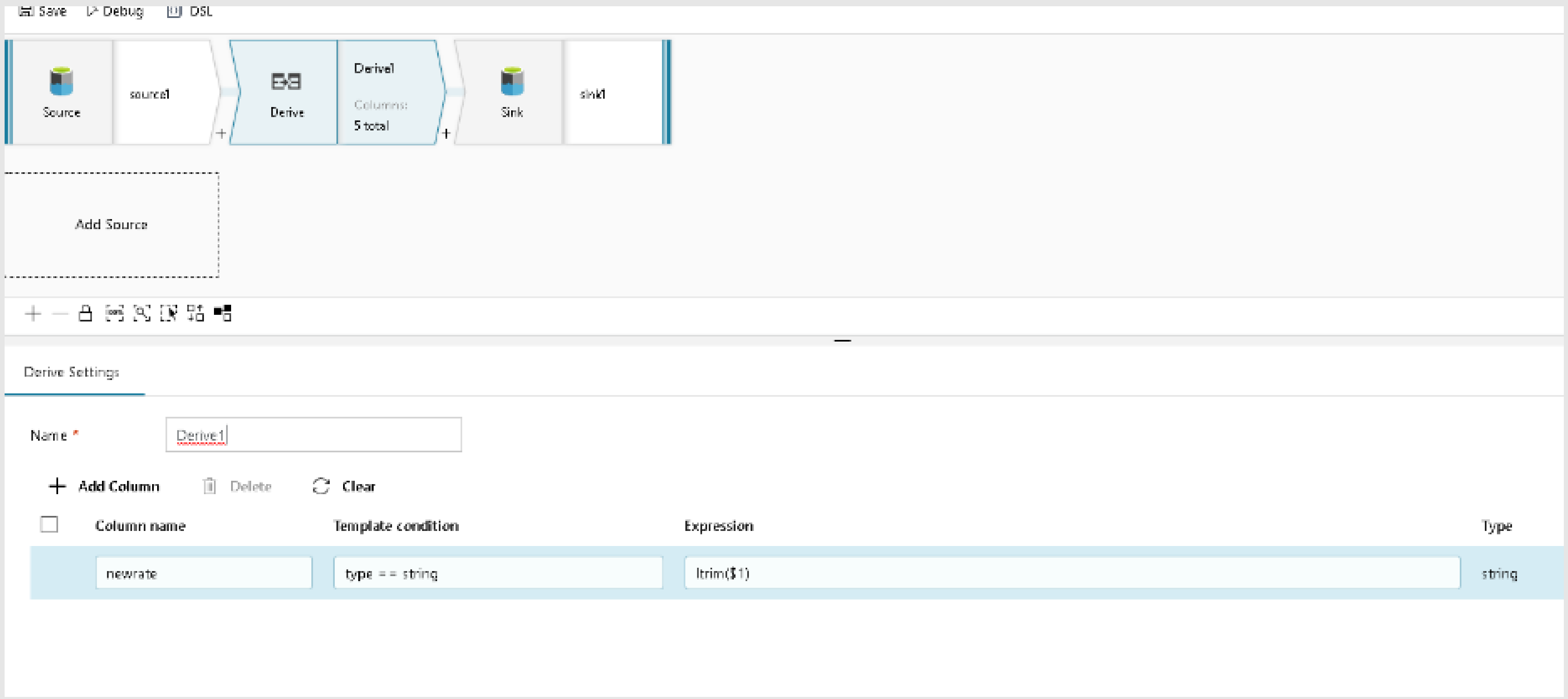


Name * source1

Source Dataset * Currency_USD Edit New

Column	Type
cid	String
cname	String
cdate	String
cprop	Integer

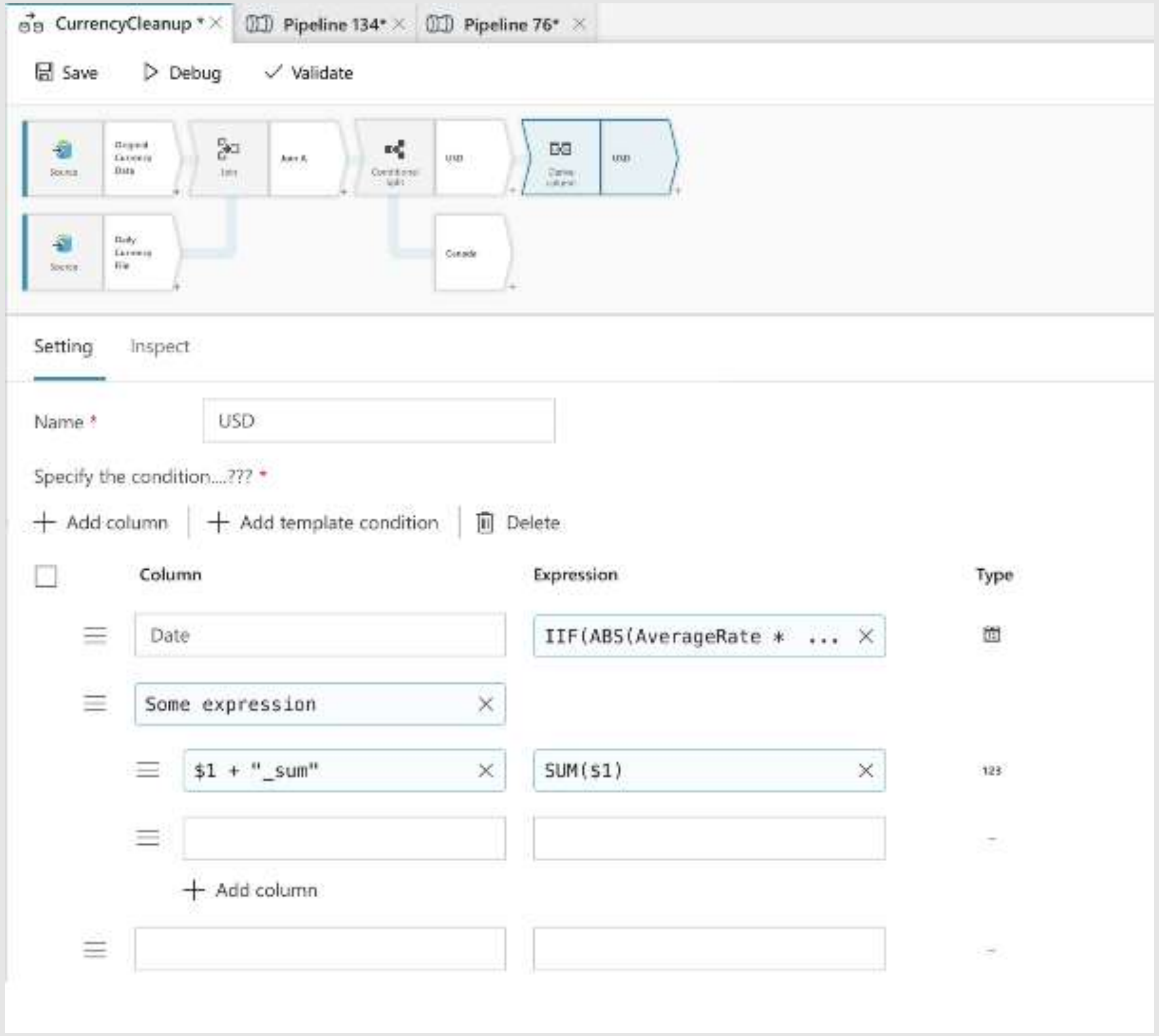
Data Engineer derives columns using template expression patterns based on name and type matching. No need to define static field names.



Name * Derive1

	Column name	Template condition	Expression	Type
	newrate	type == string	ltrim($1)	string

Data Engineer derives columns using template expression based on name and type matching. No need to define static field names.



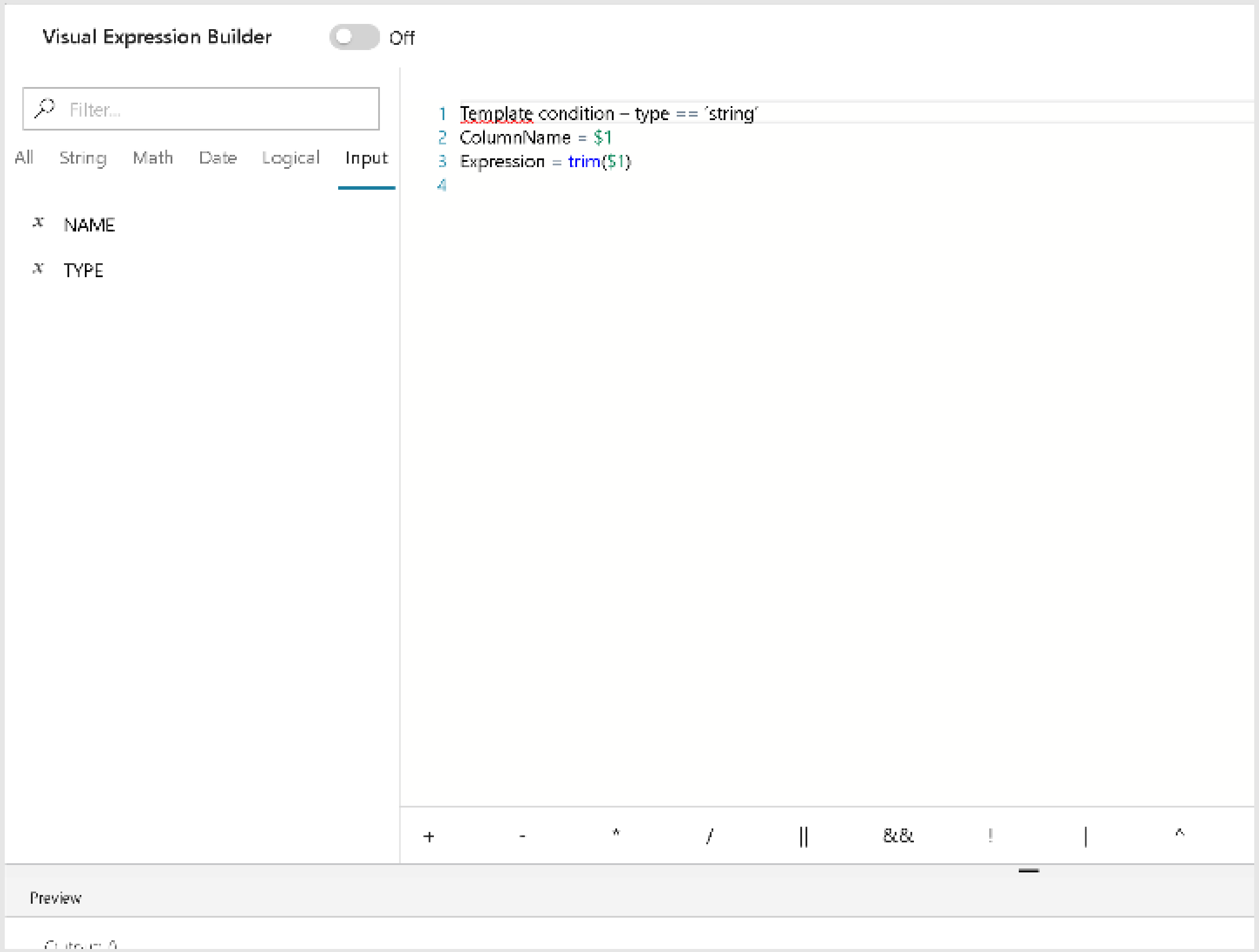
Name * USD

Specify the condition....??? *

+ Add column | + Add template condition | Delete

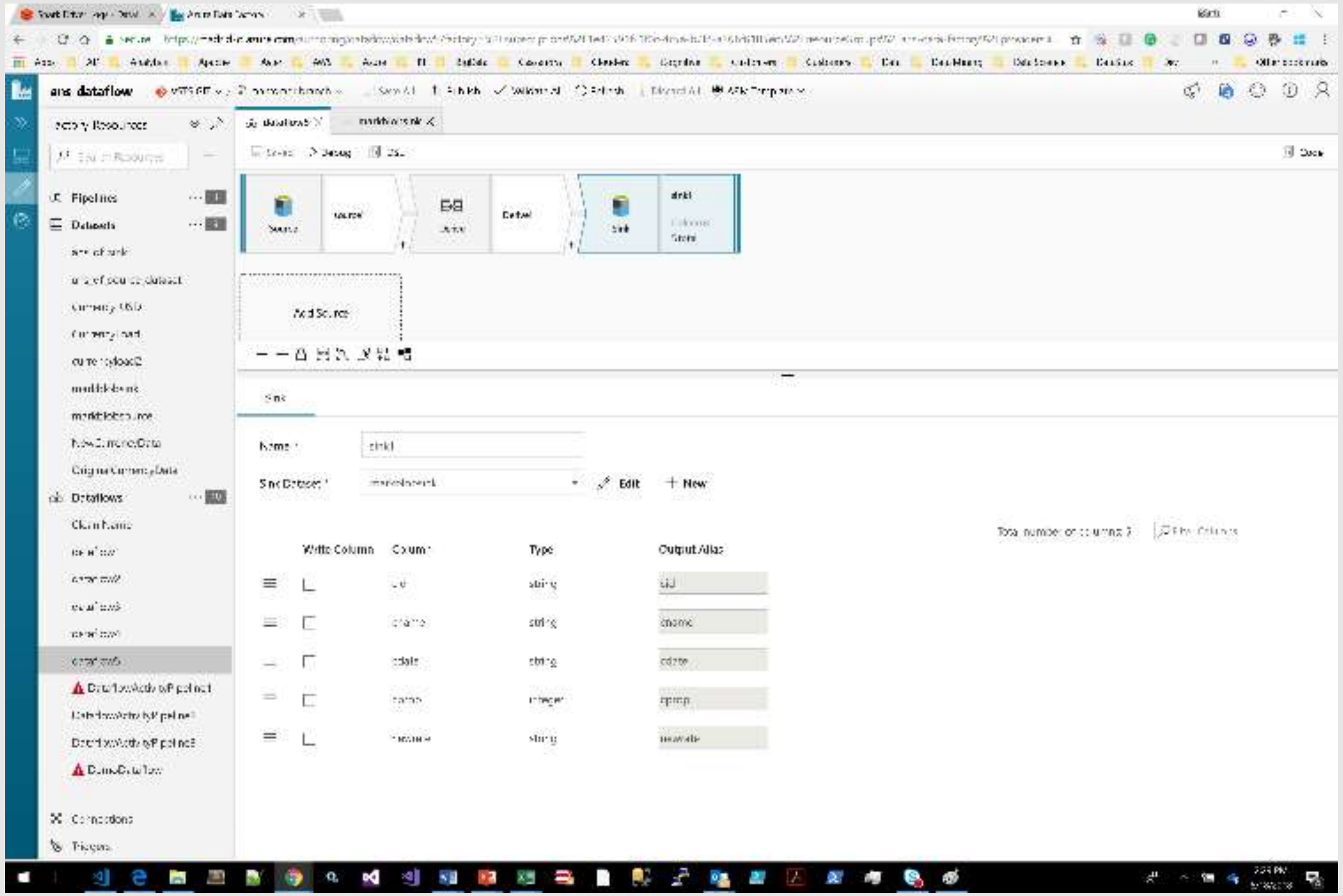
Column	Expression	Type
Date	IIF(ABS(AverageRate * ...	📅
Some expression		
$1 + "_sum"	SUM($1)	123
		-
+ Add column		
		-

Data Engineer derives columns using template expression based on name and type matching



```
Template condition – type == 'string'
ColumnName = $1
Expression = trim($1)
```

Sink all incoming fields along with new derived field



Sink

Name * sink1

Sink Dataset * [illegible] Edit + New

Total number of columns 7

Write Column	Column	Type	Output Alias
☐	cid	string	cid
☐	cname	string	cname
☐	cdate	string	cdate
☐	cpmp	integer	cpmp
☐	newvalue	string	newvalue