**Data Flow Transformations**

To access the list of transforms in Data Factory, one needs to have an instance of it created using which we can create data pipelines. We can create Data Flows which can be used as a part of the data pipeline. In the Data Flow graph, once you have added one or multiple data sources, then the next logical step is to add one or more transformations to it.

There are different types of transformations, but we are going to use few transformations for this project.

The transformations are as follows.

**Source Transformation:**

* A source transformation configures your data source for the data flow. When you design data flows, your first step is always configuring a source transformation. To add a source, select the Add Source box in the data flow canvas.
* Every data flow requires at least one source transformation, but you can add as many sources as necessary to complete your data transformations. You can join those sources together with a join, lookup, or a union transformation.
* Each source transformation is associated with exactly one dataset or linked service. The dataset defines the shape and location of the data you want to write to or read from. If you use a file-based dataset, you can use wildcards and file lists in your source to work with more than one file at a time.

**Select Transformation:**

* The select transformation is used to rename, drop, or reorder columns. This transformation doesn't alter row data, but chooses which columns are propagated downstream.
* In a select transformation, users can specify fixed mappings, use patterns to do rule-based mapping, or enable auto mapping. Fixed and rule-based mappings can both be used within the same select transformation. If a column doesn't match one of the defined mappings, it will be dropped.

**Lookup Transformation:**

* The lookup transformation is used to reference data from another source in a data flow stream. The lookup transformation appends columns from matched data to your source data.
* A lookup transformation is like a left outer join. All rows from the primary stream will exist in the output stream with additional columns from the lookup stream.

**Conditional Transformation:**

* The conditional split transformation routes data rows to different streams based on matching conditions. The conditional split transformation is like a CASE decision structure in a programming language. The transformation evaluates expressions, and based on the results, directs the data row to the specified stream.

**Derived Column Transformation:**

* The derived column transformation is used to generate new columns in your data flow or to modify existing fields.

**Aggregate Transformation:**

* The Aggregate transformation defines aggregations of columns in your data streams. Using the Expression Builder, you can define different types of aggregations such as SUM, MIN, MAX, and COUNT grouped by existing or computed columns.

There are two types in Aggregate Transformation where we can change the settings that are required

1. Group by
2. Aggregate Columns

**Group by:**

* Select an existing column or create a new computed column to use as a group by clause for your aggregation.

**Aggregate Columns:**

* Go to the Aggregates tab to build aggregation expressions. You can either overwrite an existing column with an aggregation or create a new field with a new name.

**JOIN Transformation:**

* This is the first transform you would find in the list when you click on the + sign in the graph to add transformations. Typically, when you have data from one or more data sources, there is a need to bind this data into a common stream and for such use-cases, this transform can be used. Shown below are the different types of joins that are supported along with the option to specify join conditions with different operators.
* The join transformation is used to combine data from two sources or streams in a mapping data flow. The output stream will include all columns from both sources matched based on a join condition.

There are Five different types of joins.

**Join types:**

Mapping data flows currently supports five different join types.

1. Full Outer Join
2. Inner Join
3. Left Outer Join
4. Right Outer Join
5. Custom Cross Join

**Full Outer Join:**

* Full outer join outputs all columns and rows from both sides with NULL values for columns that aren't matched.

**Inner Join:**

* Inner join only outputs rows that have matching values in both tables.

**Left Outer Join:**

* Left outer join returns all rows from the left stream and matched records from the right stream. If a row from the left stream has no match, the output columns from the right stream are set to NULL. The output will be the rows returned by an inner join plus the unmatched rows from the left stream.

**Right Outer Join:**

* Right outer join returns all rows from the right stream and matched records from the left stream. If a row from the right stream has no match, the output columns from the left stream are set to NULL. The output will be the rows returned by an inner join plus the unmatched rows from the right stream.

**Custom Cross join:**

* Cross join outputs the cross product of the two streams based upon a condition. If you're using a condition that isn't equality, specify a custom expression as your cross-join condition. The output stream will be all rows that meet the join condition.
* You can use this join type for non-equi joins and OR conditions.

**Pivot Transformation:**

* The pivot transformation is used to create multiple columns from the unique row values of a single column. Pivot is an aggregation transformation where you select group by columns and generate pivot columns using aggregate functions.

**Sort Transformation:**

* The Sort transformation allows you to sort the incoming rows on the current data stream. You can choose individual columns and sort them in ascending or descending order.