

A Short Story about Using Data Factory

A “little bit” about why first...

“Well begun is half done.”

Before the well begun – define the problem

- Customers do not always know their own problems well, though they feel something is wrong.

Understand the Situation

Problem: We don't any insight.

Similar as going to doctor, before treatment, an important question is:

- What is the root cause to the problem?



Business Wishes – which exactly “insights” are missing and how much they are wanted

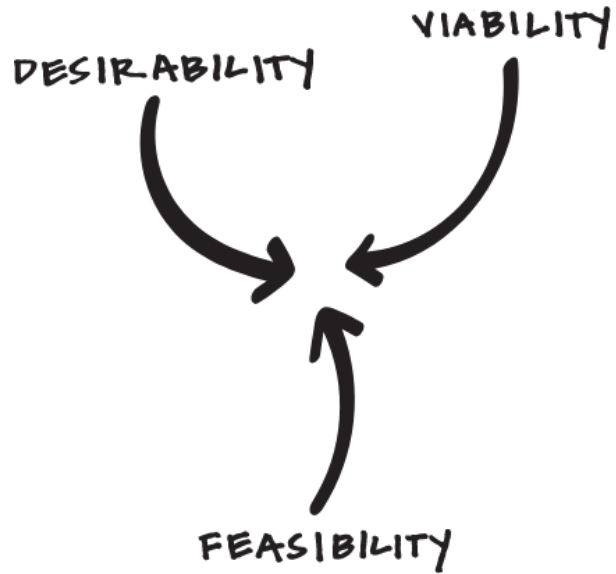
			Business value								
Wishes			Reputation Impacts (0 -5)	Commercial impacts (0 - 5)	Value for prodcut development	Necessary for continuing daily work (0-5)	Manpower Saved (0-5)	Employee satification	Overall Score	Weights	
Number of li			4	5	1	5	1	5	180	Reputation Impacts	10
What is "nor			1	3	3	2	0	2	87	Commercial Impacts	10
Usages of Te			3	3	1	3	0	5	129	Value for prodcut development	5
Usage of bod			1	3	5	2	2	4	123	Necessary for continuing daily work	8
Segementati			1	5	4	2	0	3	120	Manpower Saved	5
Churn analys			3	5	5	5	3	5	200	Employee satification	8
CRM automa			2	5	0	5	5	5	175		
CRM automa			1	3	4	0	0	4	92		
Generates a			1	4	3	4	4	4	149		
Calculate the			2	4	4	3	0	3	128		
Automate ca			0	4	0	4	5	4	129		
Clustering documents based on links and usage separately, and compare			2	2	5	1	0	3	97		

Where the Data is – Tech. Feasibility

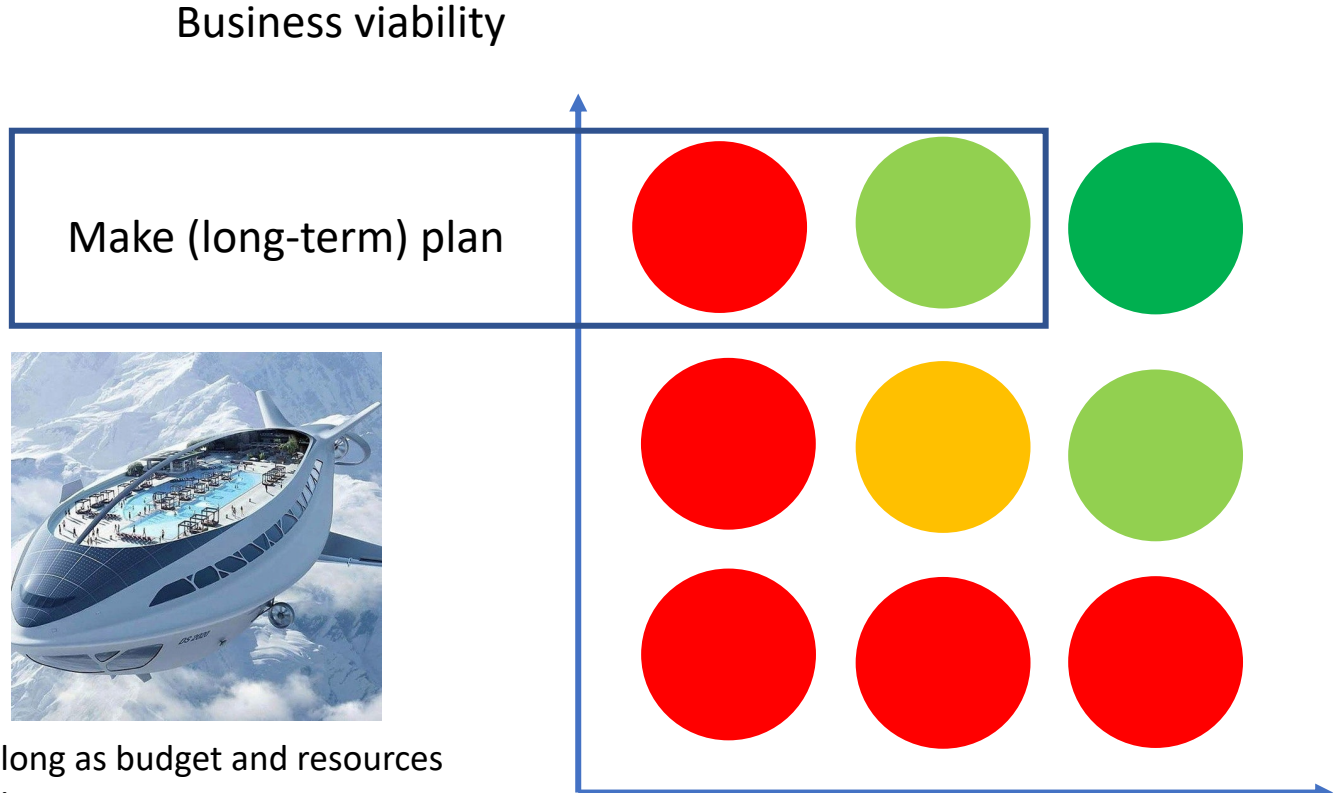


- Figured out how data flow between systems
- Read every column in the backend database

Project Selection



The intersection where design thinking lives
IDEO



as long as budget and resources
fit it...

Technical feasibility

Make multiple selection matrices as needed

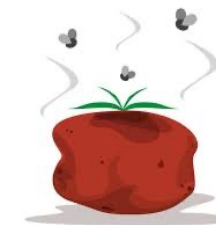
- I was doing similar thing here

Now come back to the topic: the root cause was identified during the exercises too.

We don't any insight because:

- ~~• We don't have any data.~~
- Data are in different systems - but no one collected, connected, and **stored** them

- There is no fridge or freezer, materials (data) are rotten and casted away



Common Practices in Industries



Systems living in present



serving **daily business**



Data Warehouse



Systems collect and
keeps history from
systems living in
present



serving **analytics**

*"It is almost impossible to have a sense of
vision without a sense of history. " – Kofi Annan*

Didn't exist (but now we have both)

What is a Data Warehouse



In **computing**, a **data warehouse** (DW or DWH), also known as an **enterprise data warehouse (EDW)**, is a system used for **reporting** and **data analysis** and is considered a core component of **business intelligence**.^[1] Data warehouses are central **repositories** of integrated data from one or more disparate sources. They store current and historical data in one single place^[2] that are used for creating analytical reports for workers throughout the enterprise.^[3] This is beneficial for companies as it enables them to interrogate and draw insights from their data and make decisions.^[4]



Sure, imagine a data warehouse as a giant, organized library for information that a business uses to make important decisions. Think of it like a magic storage space where all the data from different parts of the company gets neatly organized and kept in one place.

In a regular library, books are arranged by categories, like fiction, history, science, and so on. In a data warehouse, information from different areas of the business, like sales, customer info, inventory, and more, is stored in a way that makes it easy to find and use.

It actually sounds more like core layer of DWH

Current Data Warehouse Structure

- Data marts
 - denormalized, ready for being consumed



- Core Layer
 - organized library as mentioned by ChatGPT
 - normalized, single source reference



- Staging Layer
 - tables imported from different systems, as they are
 - extracted information from one single event/streaming source (e.g. event hub)



What is Data Lake

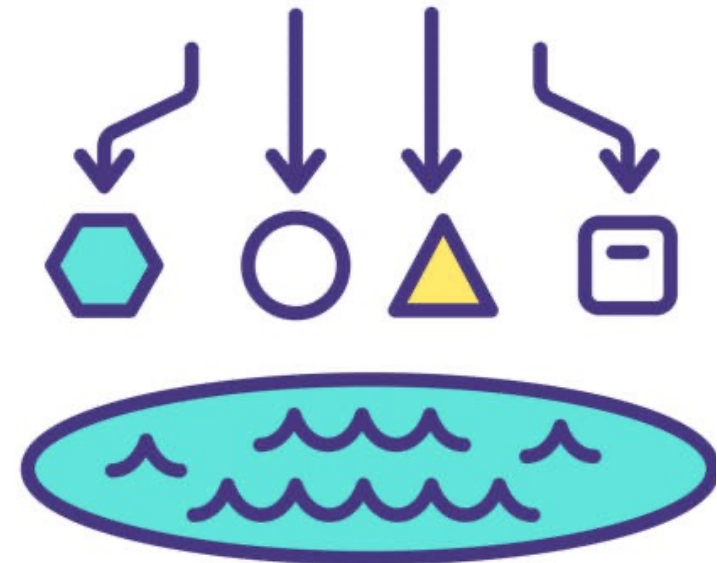
Data Lake:

Imagine a data lake as a vast, unstructured reservoir where the business stores all kinds of data, from raw files to structured information, without much organization. It's like a big lake where data from various sources, like customer logs, website clicks, social media mentions, and more, are all dumped in one place. This makes it easy to throw data into the lake without worrying about organizing it first.

4. Use Cases:

- Data Warehouse: Best for structured data analysis, reporting, and business intelligence, where the data is already well-understood.
- Data Lake: Great for storing large volumes of raw data that might have future use, especially for big data analytics, machine learning, and exploratory analysis.

In essence, a data warehouse is like a carefully curated library, whereas a data lake is more like a catch-all storage lake for all kinds of data, which can be useful when you're dealing with large, diverse, and unstructured datasets, but it requires more processing and structuring when you want to extract meaningful insights.



Databricks



- Azure Databricks is used to process data in Data Lake and transfer them into DHW
- All needed are:
 - Ensure interpretation from data to real world events is correct
 - Resource

The screenshot shows the Databricks web interface. On the left is a dark sidebar with navigation options like 'Workspace', 'Recents', 'Data', 'Workflows', 'Compute', 'SQL', 'SQL Editor', 'Queries', 'Dashboards', 'Alerts', 'Query History', 'SQL Warehouses', 'Data Engineering', 'Job Runs', 'Data Ingestion', 'Delta Live Tables', 'Machine Learning', 'Experiments', 'Feature Store', 'Models', 'Serving', 'Marketplace', 'Partner Connect', 'Disable new UI', and 'Provide feedback'. The main area displays a notebook titled 'Incremental Load: Event Hub ID Pairs Extraction' in 'master' mode, using 'Python'. The notebook content includes:
Cmd 1: A Python script for credential setup and date calculation.

```
1 # Credential
2 app_id = dbutils.secrets.get(scope="KeyVault", key="databricks-app-id")
3 datalake = dbutils.secrets.get(scope="KeyVault", key="databricks-mounted-datalake")
4 [redacted] = dbutils.secrets.get(scope="KeyVault", key="[redacted]")
5 # Load packages
6 from StorageOperation_functions import *
7 # Create today's date
8 from datetime import date
9 today = date.today()
10 # Yesterday date
11 from datetime import timedelta
12 yesterday = today - timedelta(days = 1)
13 #Pandas
14 import pandas as pd
15 #For processing json
16 import json
17 #connection string to staging database
18 postgresql_staging_url = dbutils.secrets.get(scope="KeyVault", key="adb-psql-staging-connection-string")
```

Command took 0.80 seconds -- by peiqing.zhang@gyldendal.no at 17/06/2023, 16:43:38 on Peiqing Zhang's Personal Compute Cluster

Cmd 2: A script to mount the storage.

```
1 KeyVault = 'KeyVault'
2 SecretId = "databricks-app-secret"
3 tenant_id = [redacted]
4 source = 'abfss://eventhub@' + datalake + '.dfs.core.windows.net/'
5 mnt = '/mnt/eventhub'
6 mount(tenant_id, app_id, source, mnt, KeyVault, SecretId)
```

Command took 0.28 seconds -- by peiqing.zhang@gyldendal.no at 17/06/2023, 16:43:43 on Peiqing Zhang's Personal Compute Cluster

Cmd 3: A section header 'Load Data from Yesterday'.Cmd 4: A script to read data from the staging area.

```
1 avro_df = spark.read.format("avro").load(['/mnt/eventhub/' + [redacted] + '/0/' + str(yesterday.year) + '/' + str(yesterday.month).zfill(2) + '/' + str(yesterday.day).zfill(2) + '/' + '*' + '*' + '*'])
```


Eliminate the Root Cause

to build a data warehouse and a data lake
and logistics, and other necessary supporting facilities



Data Warehouse



i.e. a platform to host them, and pipelines to transfer data

Deliver some work to build confidence and trust (directly profits will be a plus).

Status

- Kitchen is built up, fridge and freezer is there, part of the logistics is built
- Continue with building logistics
- Continue with organizing storage
- By the way making salads (PowerBI Dashboards)



Principles

Data Insight Platform

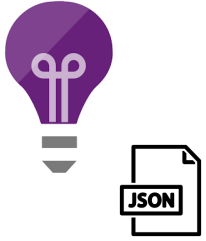


Owned by Peiqing Zhang ...

Last updated: Mar 02, 2023 • 2 min read •  3 people viewed

Principles:

1. Cloud based
2. Flexible - possible to replace building blocks with new ones later
3. Low cost based on not sacrificing security and robustness
4. Serverless as possible
5. Implement IaC (Infrastructure as Code)



EventHub

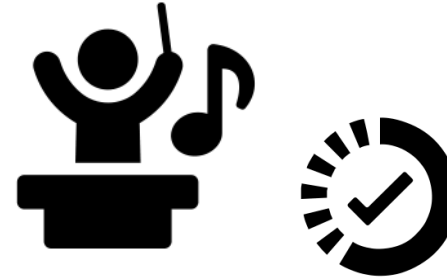


Microsoft
Dynamics 365



SQL server on Premise

Data Sources



- **Connect**
- **Process and Transfer**
- **Orchestration and Automation**



Data Warehouse

Postgres SQL –
the cheapest, Prod
upgrade to 4
cores when
summer student
was here

再窮不能窮教育，
再苦不能苦孩子。

*Though poor, we could not be poor
on education; though bitter, we
could not be bitter to the children.*



Data Lake

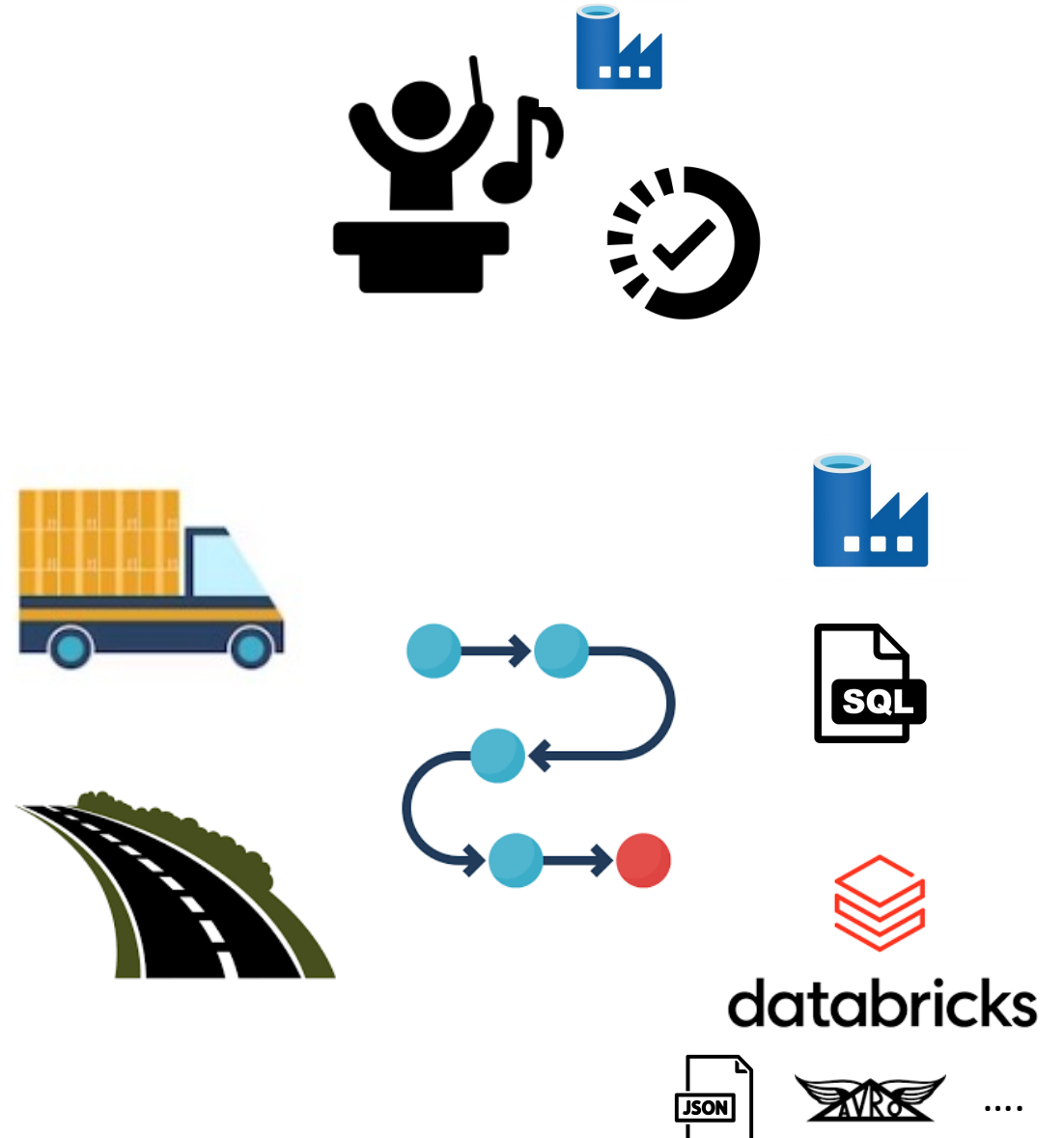
Storage account
Datalake Gen2.

Data Storage

Needs Identified

- Able to connect to different data sources, especially ERP, CRM, on-premise.
- Tool to process and transfer unstructured/**semi-structured** data to unstructured/semi-structured/**structured data** (bold is what is being executed now)
- Tool to process and transfer structured data to structured data.
- Tool for automation and orchestration

The second point is covered by Databricks (Python notebooks in Databricks, called by Data Factory), rest are covered by Data Factory.



Reasoning the Choice on Data Factory

- Allows scheduled trigger (only a daily load is needed for now)
- Provides connections to ERP, CRM, on-premises.
- Able to orchestrate other Azure resources which can be used for handling data, e.g. Databricks, Azure Functions etc.
- Pay as you go

Take a Tour in Azure Data Factory

The screenshot displays the Azure Data Factory Studio interface. On the left, a sidebar contains navigation links under 'Settings' (Networking, Managed identities, Properties, Locks) and 'Getting started' (Quick start). The main area features the 'Azure Data Factory Studio' logo and a 'Launch studio' button. Below this, the 'Data Factory Release Pipeline' section is active, showing tabs for 'Releases', 'Deployments', and 'Analytics'. The 'Releases' tab is selected, displaying a table with columns for 'Releases', 'Created', and 'Stages'. A single release, 'Release-33', is listed with a unique ID and a creation timestamp. The 'Stages' column shows two stages, 'QA' and 'Prod', both marked as successful with green checkmarks.

Releases	Created	Stages
Release-33 00d9c2a5 adf_pub...	18/07/2023, 13:15:32	QA Prod

CI/CD pipeline is in use already:

Linked services: like roads



They are created by Bicep template (IaC)

General

Factory settings

Connections

Linked services

Manage

Integration runtimes

Microsoft Purview

Source control

Git configuration

ARM template

Author

Triggers

Global parameters

Data flow libraries

Security

Credentials

Customer managed key

Outbound rules

Managed private endpoints

Workflow orchestration manager

Apache Airflow

Linked services

Linked service defines the connection information to a data store or compute. [Learn more](#)

+ New

Filter by name

Annotations : Any

Showing 1 - 9 of 9 items

Name	Type	Related
AzureDatabricks	Azure Databricks	2
AzureDataLakeStorage	Azure Data Lake Storage Gen2	24
AzureKeyVault	Azure Key Vault	7
AzurePostgreSqlCore	Azure Database for PostgreSQL	4
AzurePostgreSqlData...	Azure Database for PostgreSQL	0
AzurePostgreSqlStaging	Azure Database for PostgreSQL	5
	SQL server	9
	SQL server	4
	SQL server	5

Integration Runtimes: like trucks



Integration runtimes						
The integration runtime (IR) is the compute infrastructure to provide the following data integration capabilities across different network environment. Learn more						
+ New Refresh						
<input type="text" value="Filter by name"/>						
Showing 1 - 3 of 3 items						
Name	Type	Sub-type	Status	Related	Region	Version
AutoResolveIntegrationRun...	Azure	Public	Running	0	Auto Resolve	---
integrationRuntime	Azure	Managed Virtual Network	Running	4	West Europe	---
<div></div>	Self-Hosted	Linked	Running (Limited)	3	---	---

- Auto Resolve Integration Runtime: trucks which only allowed to be run in public area
- Integration Runtime Connected to Managed Virtual Network: trucks getting access to private territory via a private road and portal (managed private endpoint)
- Self-hosted Integration Runtime: trucks provided by the food supplier (e.g. on-premise SQL-sever)

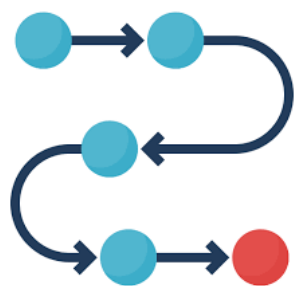
You don't own the truck teams 24/7, but you call them as taxis (pay as you go)

Managed private endpoints					
Managed private endpoint uses a private IP address from within Managed Virtual Network to connect to an Azure resource or your own private link service. Connections using managed private endpoints listed below provide access to Azure resources or private link services. Learn more					
+ New Refresh					
<input type="text" value="Search to filter items..."/>					
Showing 1 - 1 of 1 items					
Name	Provisioning ...	Approval state	VNet name	Possible Link...	Linked resource ID
AzurePostgreSqlPrivateEndpoint	Succeeded	Approved	default	0	<div></div>

Data flow: food handling



Factory Resources	
Filter resources by name	+
Pipelines	2
Change Data Capture (preview)	0
Datasets	36
Data flows	5
updateCoreLicenseOwner	
updateCoreLicenses	
updateCoreProduct	
updateCoreUsage	
updateStagingIDPairs	
Power Query	0
Templates	0



I use it to arrange the fridge

updateCoreLicense... X

Saved Validate Data flow debug

stagingLicenseUsers

Columns: 26 total

+

coreLicenseOwner

Add sink dataset

Add Source

Source settings Source options Projection Optimize Inspect Data preview

Input

Table Query Stored procedure

Query * ⓘ

with company_data_history as (
select ll.company as tibet_company_uuid,

Incremental column ⓘ

Batch size ⓘ

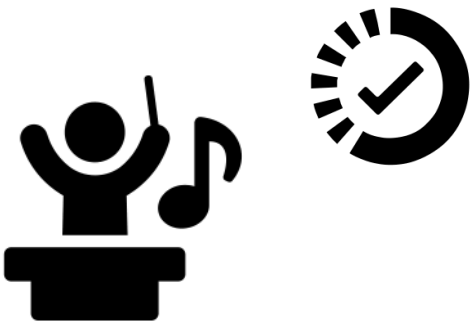
Isolation level ⓘ

Read uncommitted

Pipelines and its triggers: Orchestration and Automation

- which first, and when

Pipelines can call other Azure resources which can be used for handling data.



The screenshot displays the Azure Data Factory (ADF) user interface. On the left, the 'Factory Resources' pane shows a tree view with categories like Pipelines, Data flows, Power Query, and Templates. The 'Pipelines' section is expanded, showing a list of pipelines including 'load_lawesome_ba...' and 'scheduled_daily_data_load'. The main canvas shows a pipeline named 'scheduled_daily_data...' with a sequence of activities: 'Execute Pipeline' (containing a 'Copy' activity), followed by three 'Notebook' activities, and finally a 'Data flow' activity labeled 'Staging ID Pairs Updates'. The interface includes various toolbars for actions like 'Validate', 'Debug', and 'Data flow debug'. At the bottom, there are tabs for 'Parameters', 'Variables', 'Settings', and 'Output', with a '+ New' button under the Parameters tab.

Click around and play with it. 😊