**Step 1: Obtain and Load Datasets**

* You can download datasets like **AdventureWorks** from Microsoft's official site or use open datasets from platforms like Kaggle.
* Focus on tables like Customers, Orders, and Products.

**Step 2: Create Tables in Azure SQL and Load with Synapse**

* Use Synapse Pipelines to copy data from your source (e.g., Gen 2) to Azure SQL.
* Define tables with proper constraints. For example:

create table Customer(

id Int PRIMARY KEY,

name VARCHAR(100),

age INT,

country varchar(100),

email varchar(100),

);

create table Products(

product\_id varchar PRIMARY KEY,

type VARCHAR(100),

name varchar(100),

published varchar(100),

is\_featured varchar(100),

visibility\_in\_catalog varchar(100),

id INT FOREIGN KEY REFERENCES Customer(id),

);

create table Sales(

sales\_id varchar PRIMARY KEY,

full\_name VARCHAR(100),

job\_title varchar(100),

department varchar(100),

id INT FOREIGN KEY REFERENCES Customer(id),

product\_id INT FOREIGN KEY REFERENCES Products(product\_id)

);

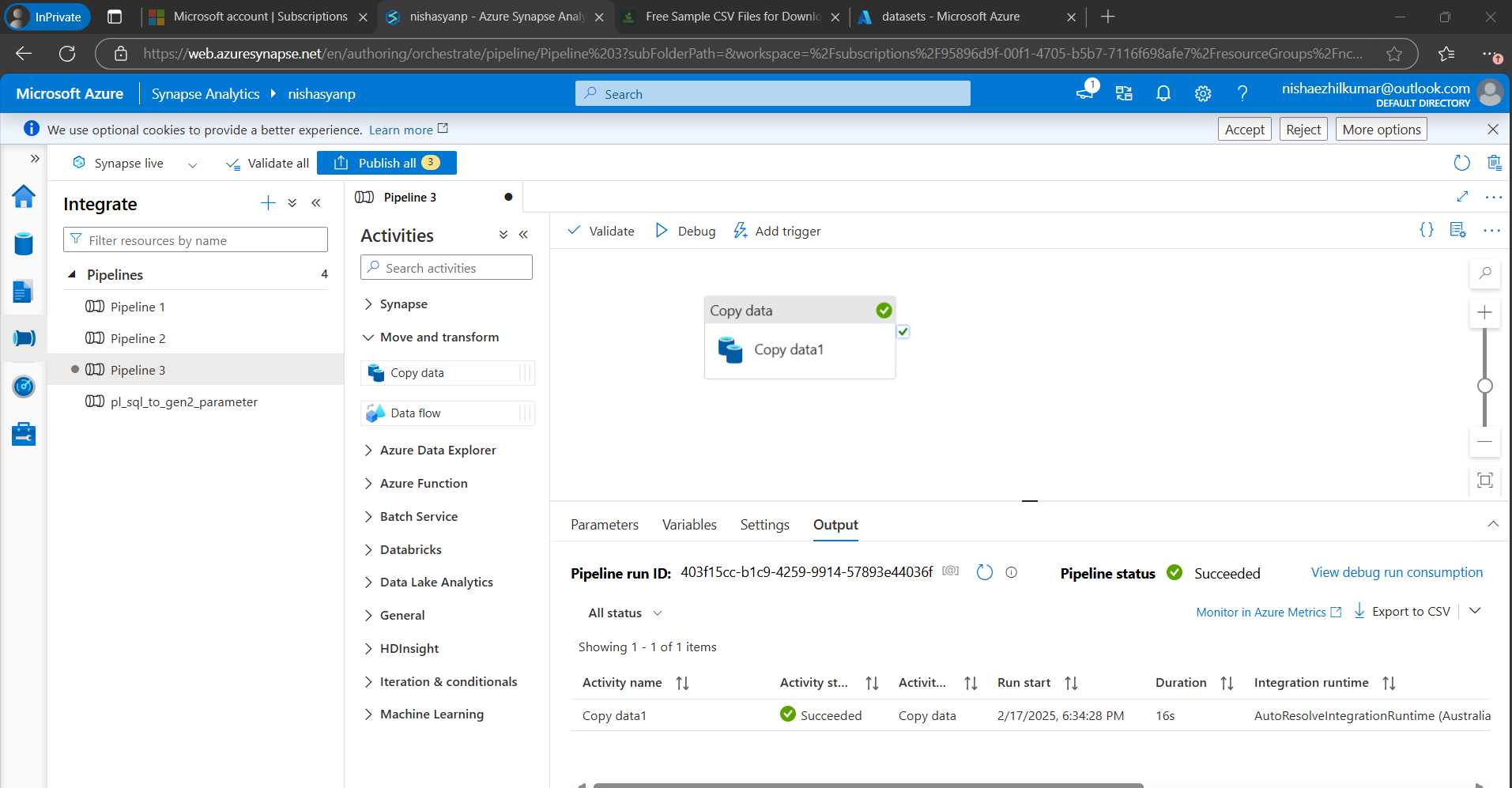
**Step 3: Explain Key Choices**

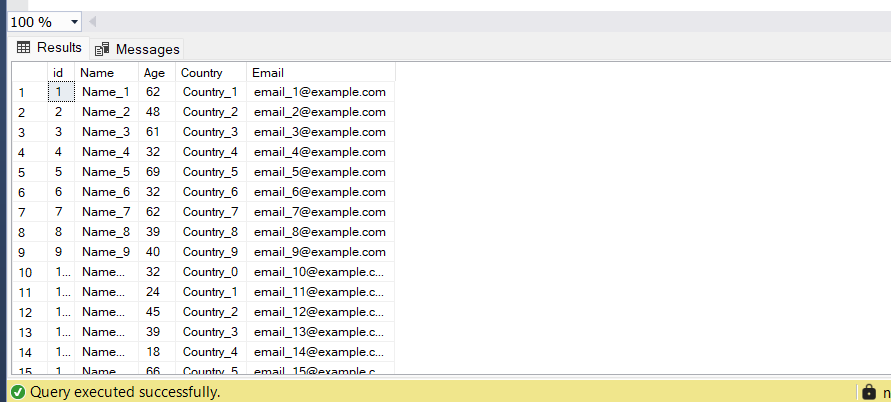
* **Primary Keys:** Uniquely identify records (e.g., id, product\_id, sales\_id).
* **Foreign Keys:** Maintain relational integrity (e.g., linking id in Products to Sales).

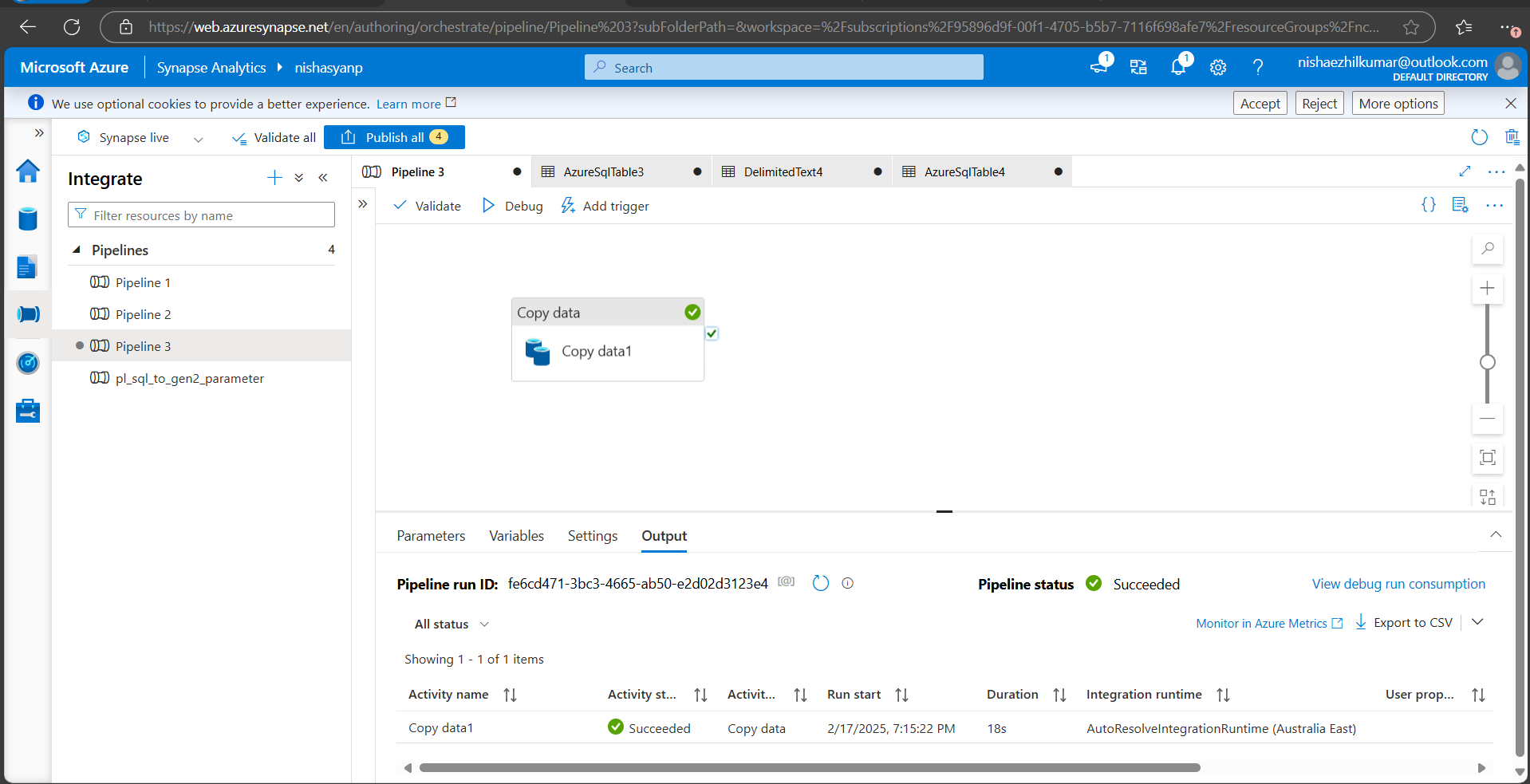
**Step 4: Advantages and Disadvantages**

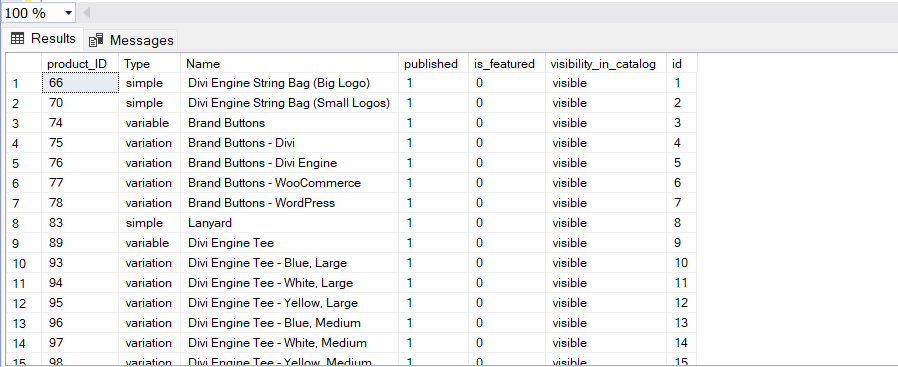
**Advantages:**  
✅ Data Integrity: Prevents duplicate or invalid entries.  
✅ Relationship Management: Connects tables meaningfully.  
✅ Query Efficiency: Indexed searches are faster.

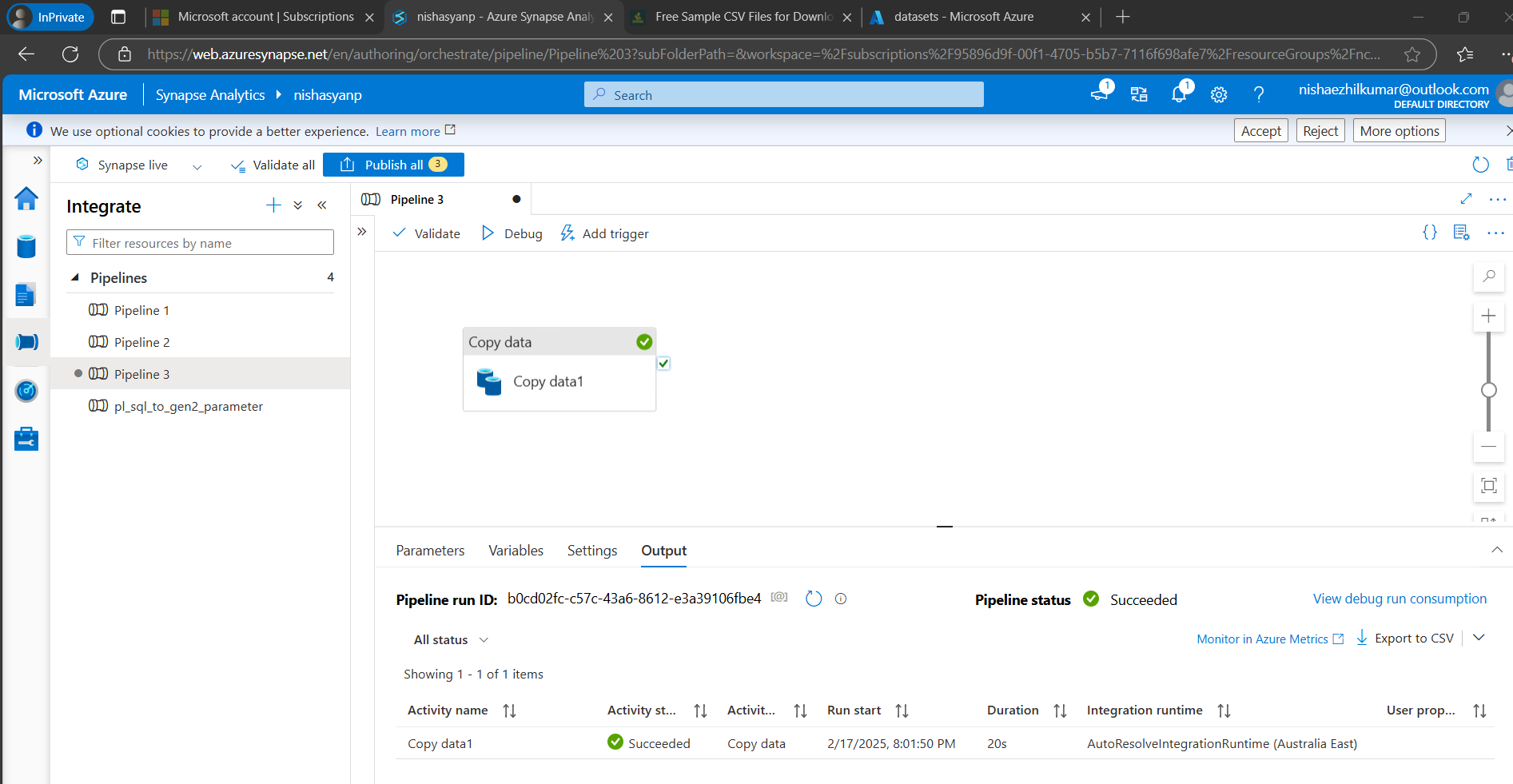
**Disadvantages:**  
❌ Insert/Update Constraints: More checks slow down operations.  
❌ Complexity: More effort to design and manage relationships.











**1. Introduction:**  
A schema defines the structure of a dataset, including tables, columns, data types, and relationships. Structured data requires predefined schemas, ensuring consistency and data integrity. This project compares datasets with schemas, highlighting their advantages and disadvantages.

**2. Dataset Analysis:**

**Dataset: Sales, Customer, and Products Database (Structured Data)**  
**SQL Schema:**

CREATE TABLE Customer(

id INT PRIMARY KEY,

name VARCHAR(100),

age INT,

country VARCHAR(100),

email VARCHAR(100)

);

CREATE TABLE Products(

product\_id VARCHAR PRIMARY KEY,

type VARCHAR(100),

name VARCHAR(100),

published VARCHAR(100),

is\_featured VARCHAR(100),

visibility\_in\_catalog VARCHAR(100),

id INT FOREIGN KEY REFERENCES Customer(id)

);

CREATE TABLE Sales(

sales\_id VARCHAR PRIMARY KEY,

full\_name VARCHAR(100),

job\_title VARCHAR(100),

department VARCHAR(100),

id INT FOREIGN KEY REFERENCES Customer(id),

product\_id VARCHAR FOREIGN KEY REFERENCES Products(product\_id)

);

* Tables: Customer, Products, Sales (with foreign key relationships)
* Advantages: Strong data integrity and relational efficiency.
* Disadvantages: Complex schema changes and maintenance challenges.

**3. Comparative Analysis:**

* **Advantages of Having a Schema:** Data integrity, efficient querying, and optimized storage.
* **Disadvantages of Having a Schema:** Schema inflexibility and complexity in updates.
* **Advantages of Not Having a Schema:** Flexibility and rapid data ingestion.
* **Disadvantages of Not Having a Schema:** Poor data quality and complex queries.

**4. Practical Scenarios:**

* **Schema Example:** A sales database uses foreign keys to ensure valid customer-product relationships.
* **No-Schema Example:** A social media platform uses a NoSQL database for flexible content storage.

**5. Conclusion:**  
Schemas ensure structure, consistency, and efficiency in data management. For transactional data, structured schemas are crucial, while flexible, schema-less formats suit rapidly changing datasets. A hybrid approach is often best for modern data architectures.