# 

# 

# 

# 

# ***Project Documentation***

***On***

***OTT Platform Advertisement Analytics***

# 

# **CONTENT**

| S.No | Sub S.No | Title of Content |
| --- | --- | --- |
| 1. |  | Case Study: OTT Platform Advertisement Analytics |
|  | 1.1 | Executive Summary |
|  | 1.2 | business Problem |
| 2. |  | Introduction |
| 3. |  | Problem Statement |
|  | 3.1 | Objective |
|  | 3.2 | Challenges |
|  | 3.3 | Business Requirements |
| 4. |  | Tools and Requirements |
|  | 4.1 | Software and Tools |
|  | 4.2 | Hardware Requirements |
|  | 4.3 | Knowledge / Skill Requirements |
| 5. |  | Implementation |
|  | 5.1 | Data Sources |
|  | 5.2 | Azure Cloud Setup (Prerequisites) |
|  | 5.2.1 | Create Azure Account (and Resource Group) |
|  | 5.2.2 | Create Azure SQL Server |
|  | 5.2.3 | Create Azure SQL Database |
|  | 5.2.4 | Configure Azure SQL Database Firewall Rules |
|  | 5.2.5 | Create Azure Storage Account (for Blob Storage) |
|  | 5.3 | Azure Data Factory - ADF |
|  | 5.3.1 | Create Azure Data Factory |
|  | 5.3.2 | Orchestration Steps and Pipeline Architecture — design pattern |
|  | 5.3.3 | Publish and Trigger the Orchestration Pipeline |
|  | 5.3.4 | Monitor the Orchestration Pipeline Run |
| 6. |  | Data Storage (Azure SQL Database) |
|  | 6.1 | Schema Design and Normalization |
| 7. |  | Data Modeling & Analytics (Power BI) |
|  | 7.1 | Data Modeling (Power BI) |
|  | 7.1.1 | Connect Power BI to Azure SQL Database |
|  | 7.1.2 | Build Relationships in Power BI |
|  | 7.2 | Power BI Report Design |
| 8. |  | Key Learnings |
| 9. |  | Future Enhancements |
| 10. |  | Conclusion |

# 

## 

## **1. Case Study: OTT Platform Advertisement Analytics**

## **1.1 Executive Summary**

This case study demonstrates how to build an advertisement analytics platform for an OTT (Over The Top) streaming service using Azure's free tier services. The solution tracks user engagement, ad performance, and viewing patterns to optimize advertisement placement and revenue generation.

The end-to-end solution involves extracting data from SQL tables, storing it in Azure Blob Storage, transforming it via Azure Data Factory (ADF) pipelines, storing the cleaned data in Azure SQL Database, and analyzing it using Power BI. This setup provides actionable insights for decision-makers, enhances user experience, and drives business growth through informed advertisement strategies.

**1.2 Business Problem**

StreamFlix, a growing OTT platform, needs to:

* Track advertisement performance across different shows and user segments
* Analyze user engagement with various ad types
* Optimize ad placement based on viewing patterns
* Generate revenue reports for advertisers
* Store and process large volumes of streaming and advertisement data

## **2. Introduction**

OTT platforms have become a primary medium for streaming video content. Revenue generation in OTT platforms largely depends on advertisements. StreamFlix, although expanding rapidly, lacked a structured approach to monitor ad performance and optimize placements.

The project leverages **Azure Cloud services** and **Power BI** to implement a robust analytics solution that tracks ad impressions, clicks, and user engagement metrics. It provides an integrated approach to analyze large-scale streaming and advertisement data to enhance revenue and improve the user experience.

**3. Problem Statement**

#### **3.1 Objective**

The main objectives of the project are:

* Build an **end-to-end analytics platform** for OTT advertisements.
* **Ingest** data into a central storage.
* **Transform** and **store** the data efficiently in Azure SQL Database.
* Use **Power BI** to visualize user engagement, ad performance, and revenue reports.
* Provide actionable insights to optimize ad placement and increase revenue.

#### **3.2 Challenges**

* Ensuring **data quality** and consistency across multiple tables.
* Building an **efficient ETL pipeline** using ADF.
* Creating insightful **Power BI dashboards**.

#### **3.3 Business Requirements**

* Accurate reporting of ad performance metrics.
* Insights into user behavior and engagement patterns.
* Optimization recommendations for ad placement.
* Scalable solution with minimal operational cost.
* Generate revenue reports for advertisers

## **4. Tools and Requirements**

### **4.1 Software and Tools**

For this project, we mainly used Azure services along with some supporting tools:

| **Tool/Software** | **Purpose** | **Notes** |
| --- | --- | --- |
| **Azure Account** | Cloud platform to host and manage services | Free Tier |
| **Azure Blob Storage** | Stores raw SQL data files | - |
| **Azure Data Factory (ADF)** | Builds the ETL pipeline, handles data movement and transformation | Latest |
| **Azure SQL Database** | Central place to store structured data | Latest |
| **Power BI Desktop** | For data modeling and building dashboards | Latest |
| **Power BI Service** | To publish and share dashboards online | Optional |
| **SQL Server Management Studio (SSMS)** | Used for querying and checking SQL databases | Latest |
| **Python (Optional)** | For extra data transformation or ML analytics | 3.x |
| **Excel (Optional)** | For quick checks and small data inspections | Latest |

### **4.2 Hardware Requirements**

Since the project involves running ETL pipelines and handling dashboards, below are the minimum and recommended specs:

| Components | Minimum | Recommended |
| --- | --- | --- |
| Ram | 8 GB | 16 GB |
| Processor | Intel i5 | Intel i7 |
| Storage | 256 GB HDD | 512 GB SSD |
| Internet | 10 Mbps | 50 Mbps |

### **4.3 Knowledge / Skill Requirements**

To work on this project smoothly, the following skills are useful:

* Basics of SQL and working with databases.
* Familiarity with Azure services like Blob Storage, ADF, and SQL DB.
* Understanding of ETL pipelines and how data transformation works.
* Knowledge of Power BI for analytics and dashboarding.
* (Optional) Python skills for advanced analytics or machine learning.

## **5. Implementation**

### **5.1 Data Sources**

The project uses 7 datasets which is in csv format:

1. **Users Dataset:** Contains user profile and subscription info (user\_id, age, gender, subscription\_type, country, registration\_date etc).
2. **Content Dataset:** Metadata of shows/movies including (content\_id, title, genre, duration\_minutes, release\_date etc).
3. **Advertisement Dataset:** Advertisement metadata (ad\_id, ad\_type, advertiser\_name, campaign\_name, cpm\_rate etc).
4. **Ad Impressions Dataset:** Ad impression events with timestamps and content mapping like (impression\_id, ad\_id, content\_id, timestamp, impression\_cost).
5. **Ad Clicks Dataset:** Ad click events linked to users and ads like (click\_id, ad\_id, user\_id, click\_time).
6. **User Session Dataset:**Captures user viewing sessions including watch time, content, and ad interactions like (session\_id, user\_id, content\_id, start\_time, end\_time, watch\_time\_seconds, ads\_shown, ads\_interacted).
7. **Revenue Dataset:** Stores details of ad revenue generated across campaigns, ad types, and time periods it includes (date, ad\_id, campaign\_id, revenue).

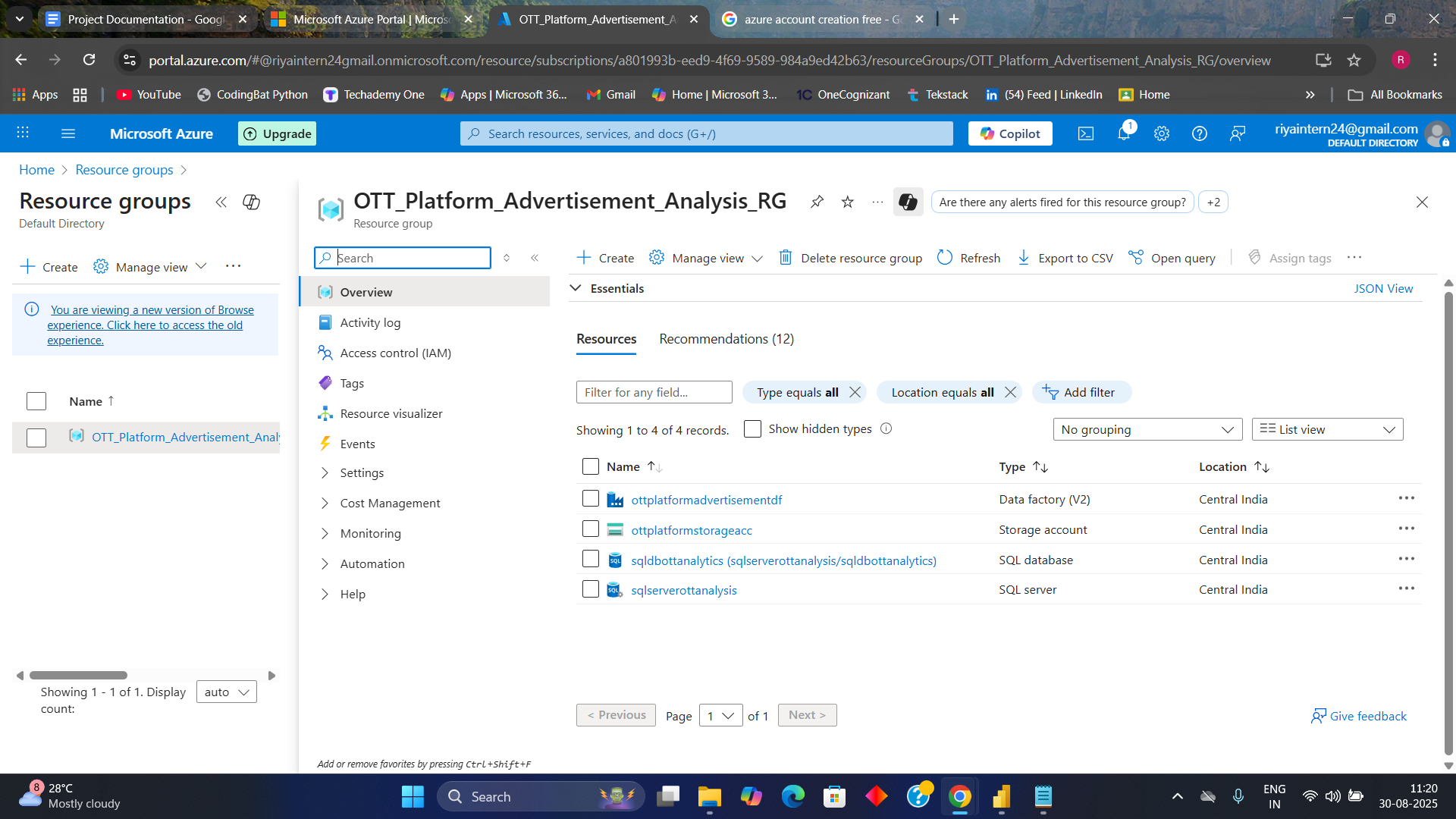
### 

### **5.2 Azure Cloud Setup (Prerequisites)**

### **5.2.1 Create Azure Account (and Resource Group)**

1. Sign in or create a free Azure account (<https://portal.azure.com/>).
2. Create a **Resource Group**:  
   * Azure Portal → Resource groups → + Create → give a name (e.g., rg-streamflix-analytics) → choose region.
3. Use the resource group as the logical container for all resources.

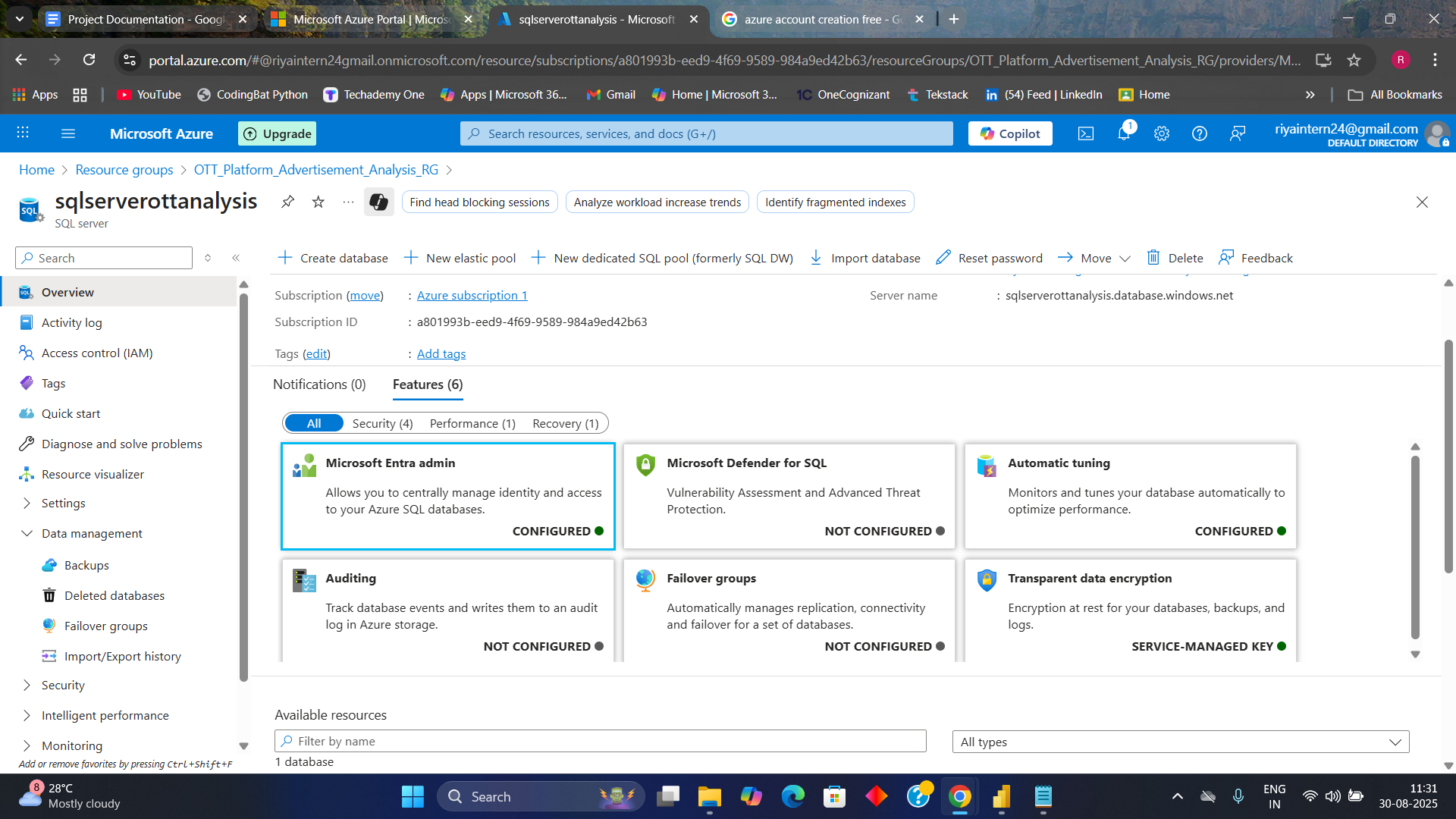
**Snapshots: Creating Resource Group**



### **5.2.2 Create Azure SQL Server**

1. In Azure Portal → **SQL servers** → + Add.
2. Enter server name (e.g., streamflix-sqlsvr), admin username & password, select resource group and region.
3. Note the **server fully qualified name** (streamflix-sqlsvr.database.windows.net) — As it will need later.

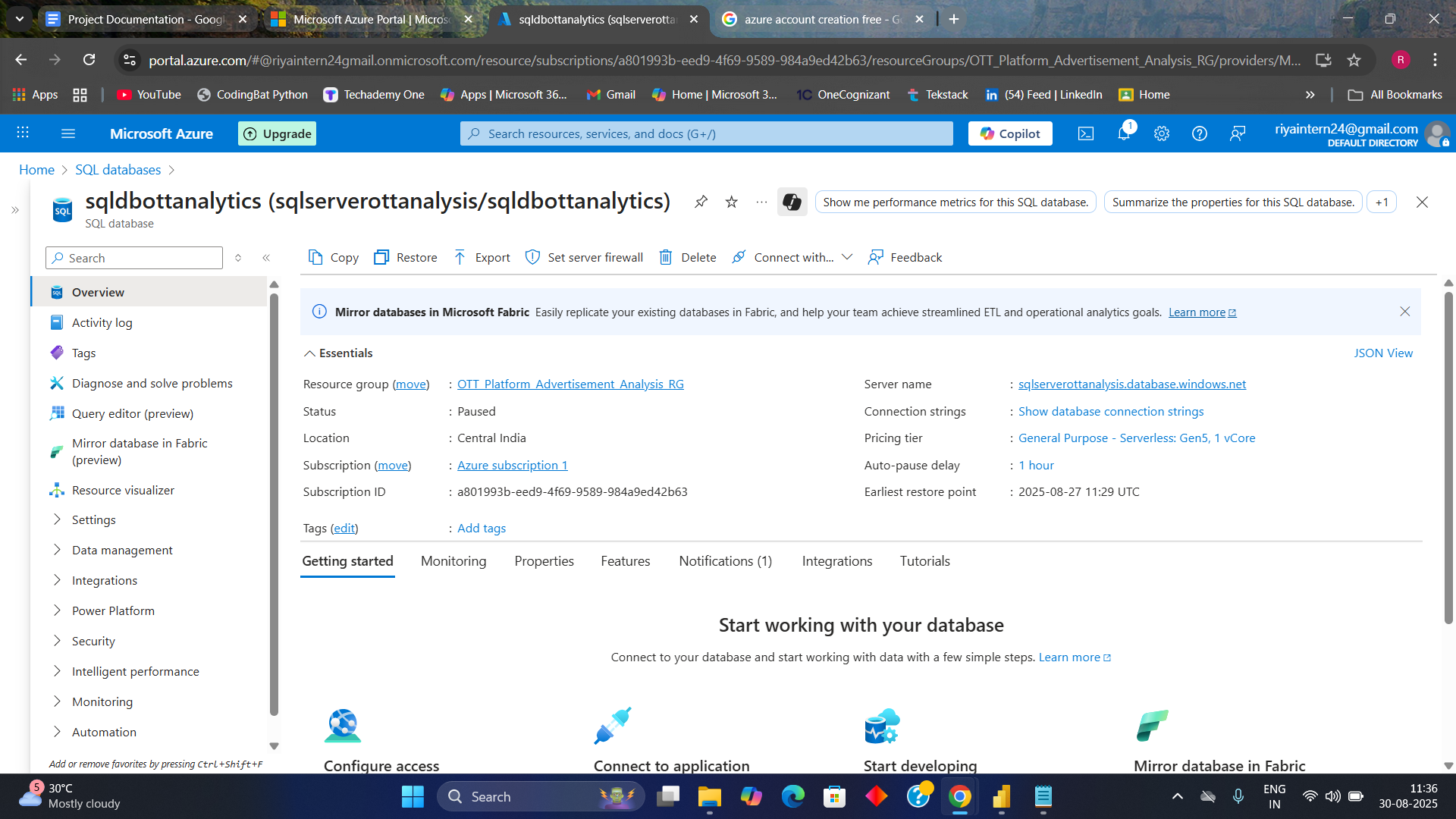
**Snapshot: Creating SQL Server**



### **5.2.3 Create Azure SQL Database**

1. Azure Portal → **SQL databases** → + Add.
2. Choose the server that created previously, database name (e.g., streamflix\_analytics).
3. Select compute tier — for free tier/testing choose Basic/Serverless or minimal DTU. For production, use vCore as needed.
4. Create **schema** later with SSMS or Azure Data Studio

**Snapshot: Creating Azure SQL Database**



### **5.2.4 Configure Azure SQL Database Firewall Rules**

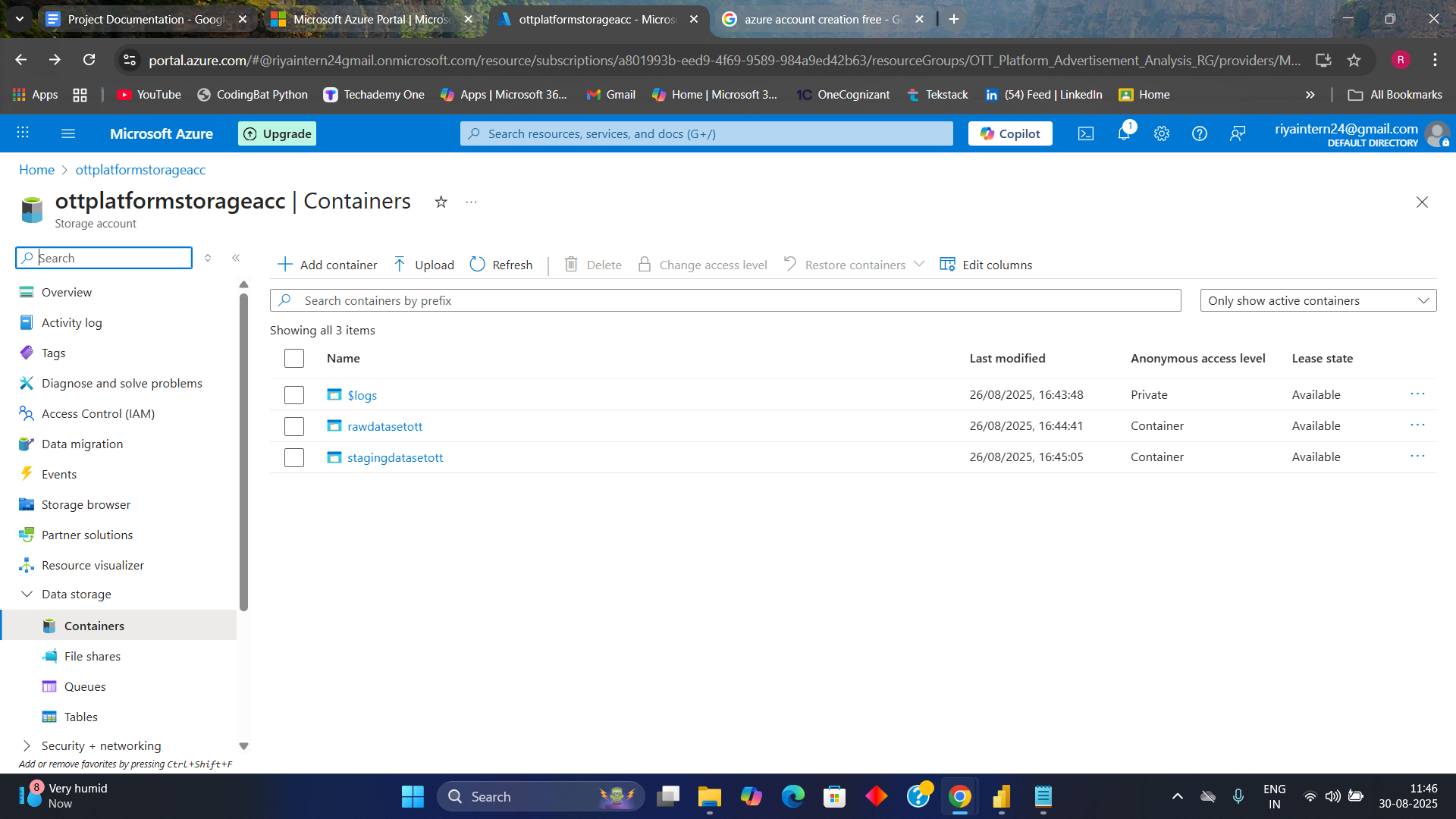
1. Azure Portal → your SQL server → **Networking** (Firewall rules).
2. Add client IP (current machine) to allow connections.
3. Power BI Service will connect, add Power BI service IP ranges.
4. For automation, enable Allow Azure services and resources to access this server.

### **5.2.5 Create Azure Storage Account (for Blob Storage)**

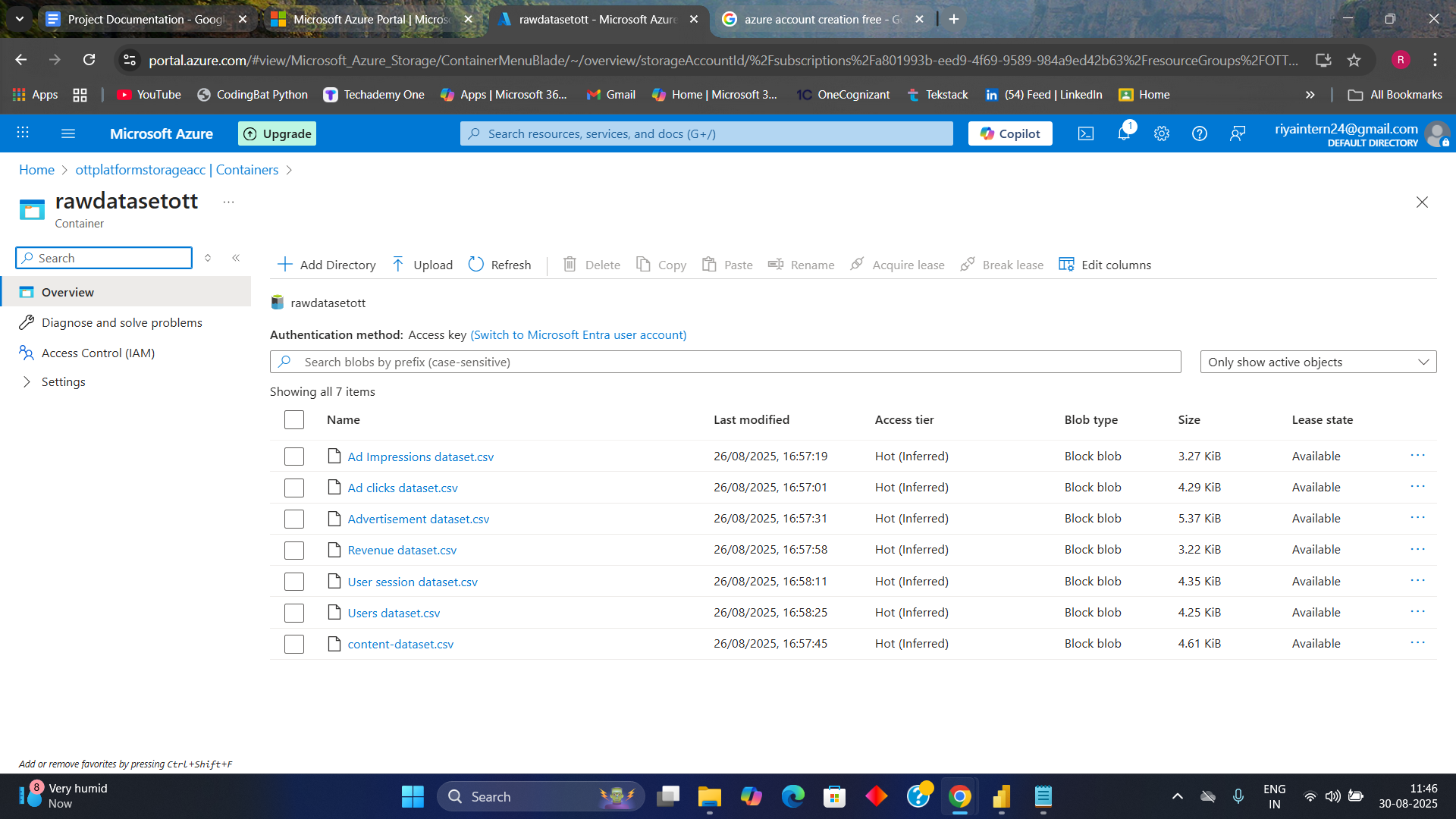
1. Azure Portal → **Storage accounts** → + Create.
2. Fill name (e.g., streamflixstorage), select resource group and region, choose performance & redundancy (standard LRS).
3. After creation → Storage account → **Containers** → + Container (e.g., rawdataset).

4. Upload a csv files into raw/Users/ etc.

**Snapshots: Creating Storage Account**



**Snapshot : Uploading Raw CSV Datasets**



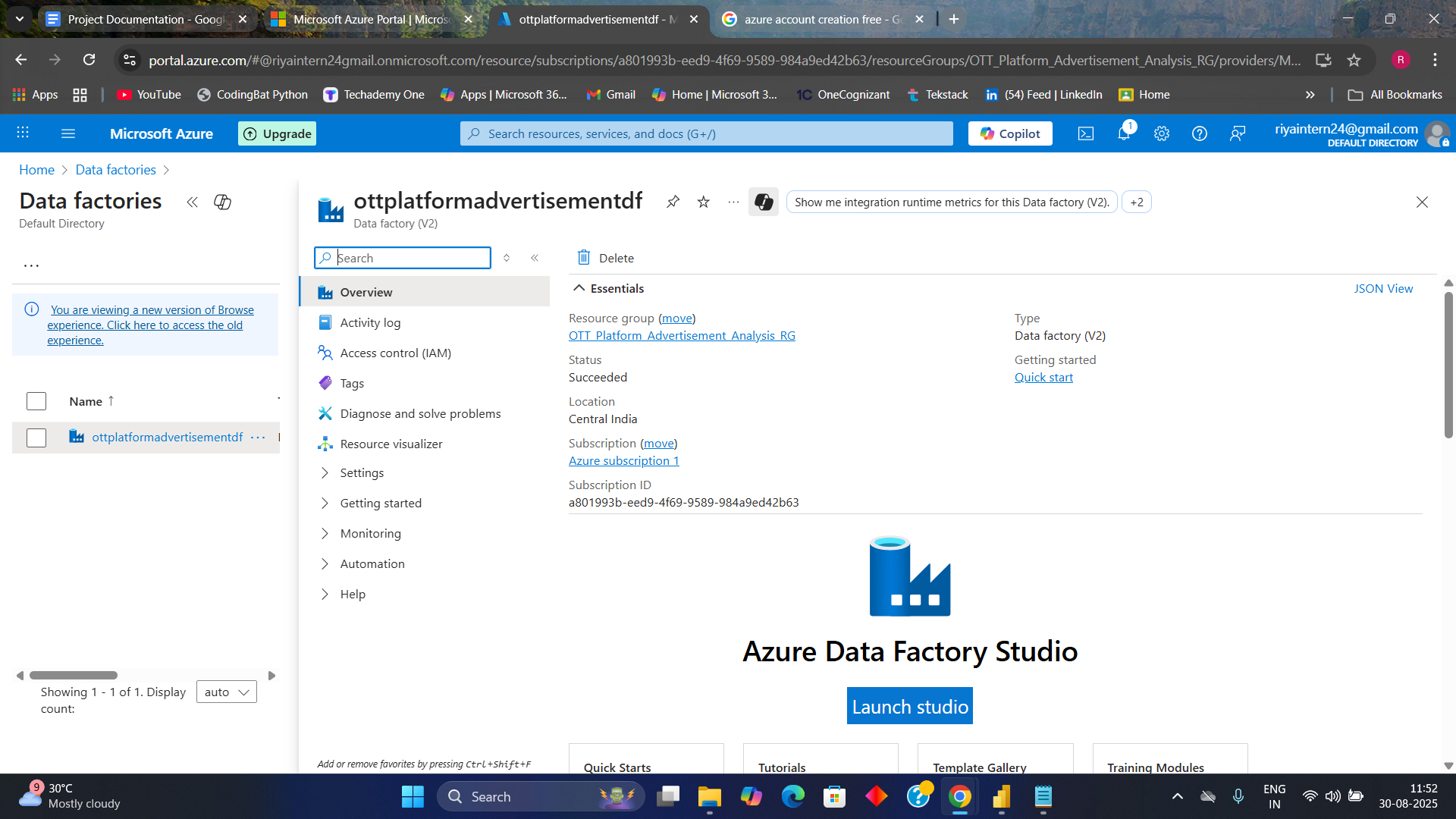
### 

### **5.3 Azure Data Factory - ADF**

#### **5.3.1 Create Azure Data Factory**

1. Azure Portal → **Data factories** → + Create.
2. Name it (df-streamflix), select resource group & region.
3. After creation open **ADF Studio** (Author & Monitor)

.**Snapshot: Creating Azure Data Factory**



### **5.3.2 Orchestration Steps and Pipeline Architecture - design pattern** The data processing for the OTT Ads Analytics System is designed using three distinct pipelines in Azure Data Factory (ADF). This modular approach ensures better separation of concerns, scalability, and simplified monitoring.

The process involves the following pipelines and their roles:

### **Create Datasets in ADF**

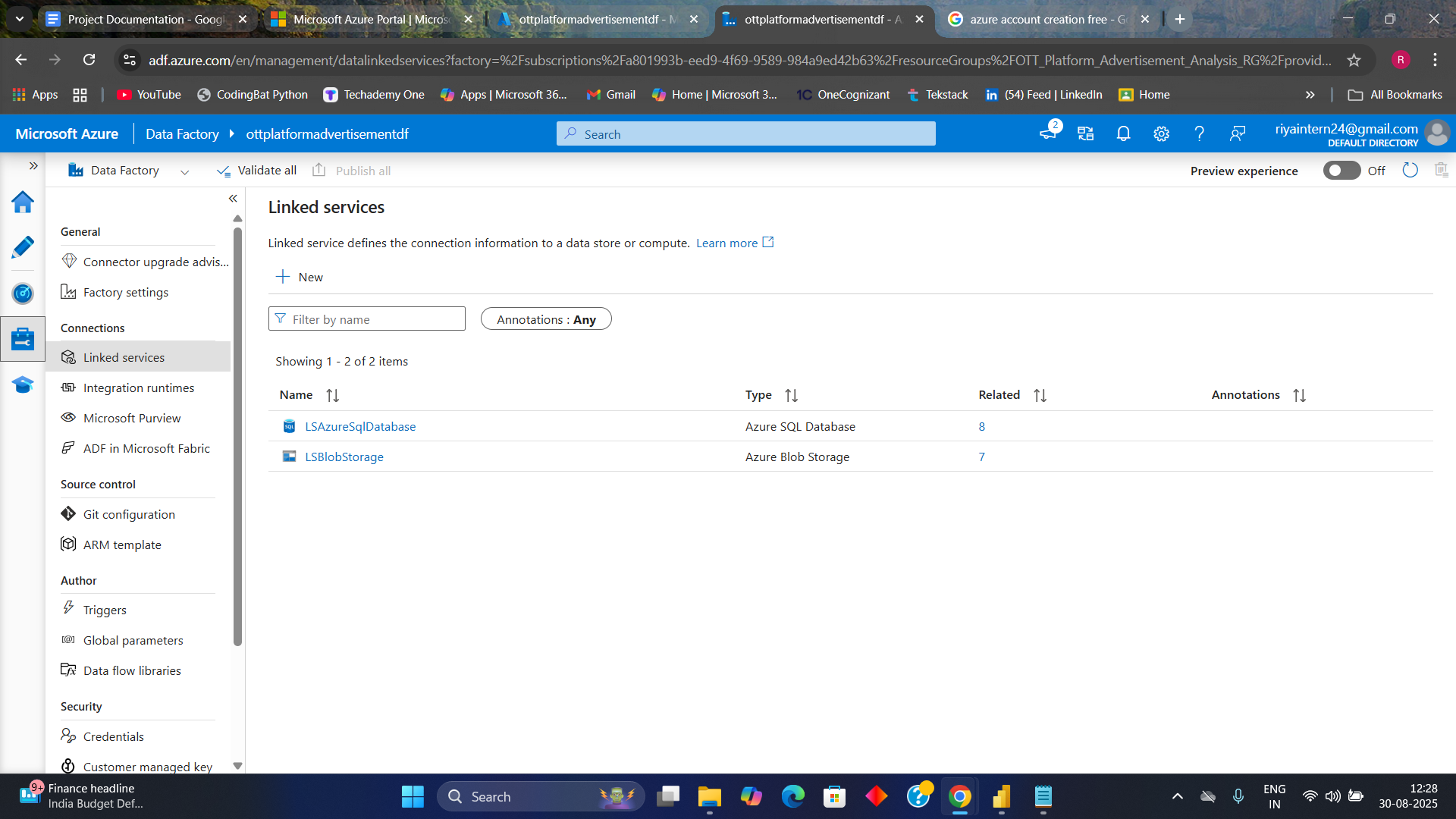
1. In ADF Studio → Author → Datasets → + New.
2. Create **source datasets** (Blob CSV/Parquet) for each entity: Users, Content, Ads, Impressions, Clicks, Revenue, Sessions.
3. For each dataset, configure schema preview and sample file path.

#### 

### **2. Create Linked Services in ADF**

1. In ADF Studio → Manage → Linked Services → + New.
2. Create Linked Service to **Azure Blob Storage**: choose authentication (Managed Identity, Account Key or SAS). Test connection.
3. Create Linked Service to **Azure SQL Database**: server name, DB name, admin user or managed identity. Test connection.

**Snapshot : Creating Linked Services**



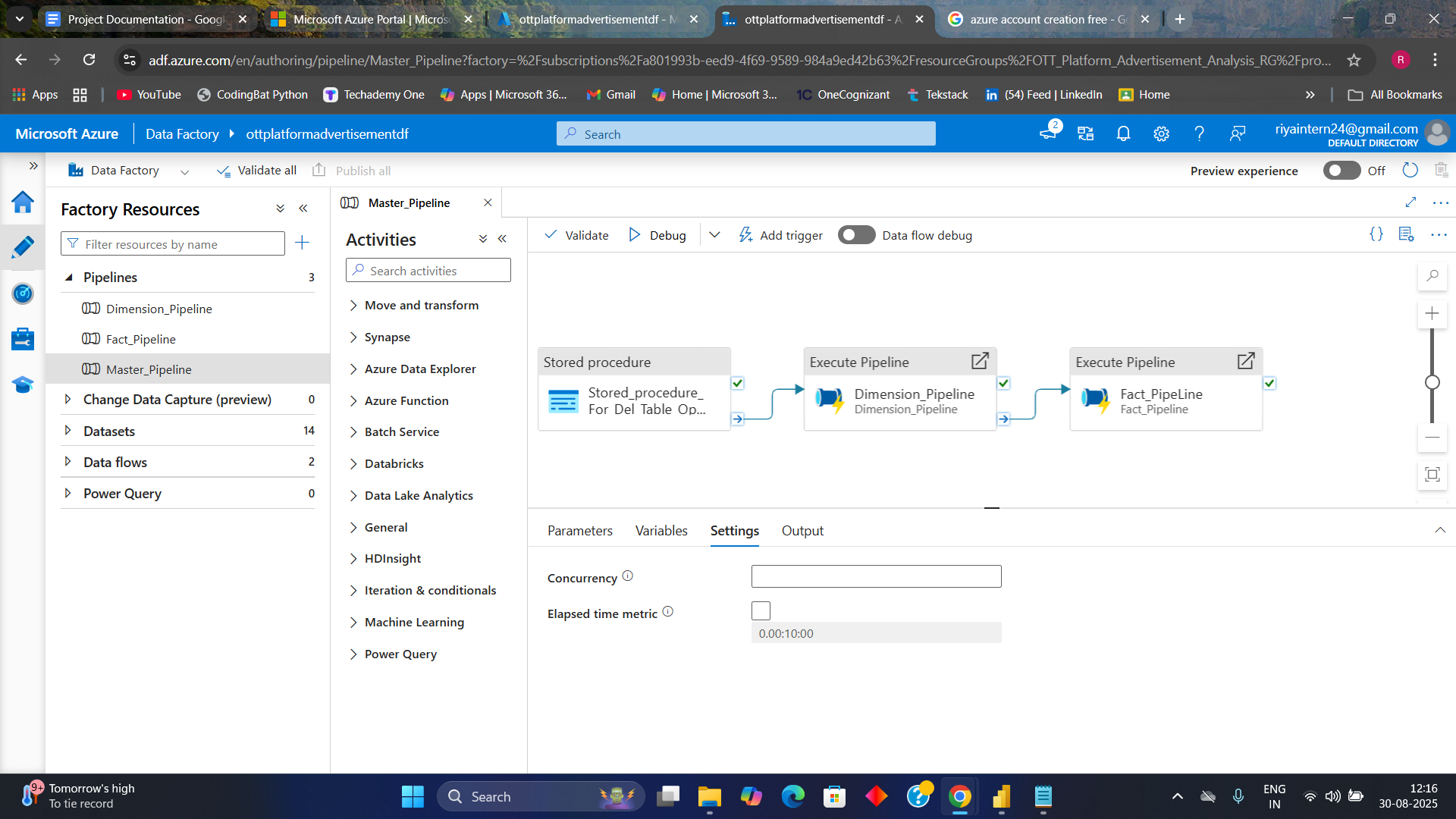
**3. Orchestration Pipeline**

#### **1. Master Pipeline (Master\_Pipeline)**

#### **Purpose**: This is the master pipeline that **coordinates and manages the execution flow**. It ensures that Dimension data is loaded before Fact data, maintaining referential integrity in the SQL database.

1. **Activities**:  
   1. Contains **Stored Procedure** to delete all the already existing data in sql dband **Execute Pipeline** activities to trigger the Dimension and Fact pipelines.
   2. Handles execution order, retries in case of failure, and notification logic.
   3. Ensures the pipeline run history is tracked for monitoring purposes.
2. **Flow**:  
   1. Start →Execute Stored Procedure
   2. Once successful Execute **Dimension\_Pipeline**
   3. Once successful → Execute **Fact\_Pipeline**
   4. On completion → log to monitoring

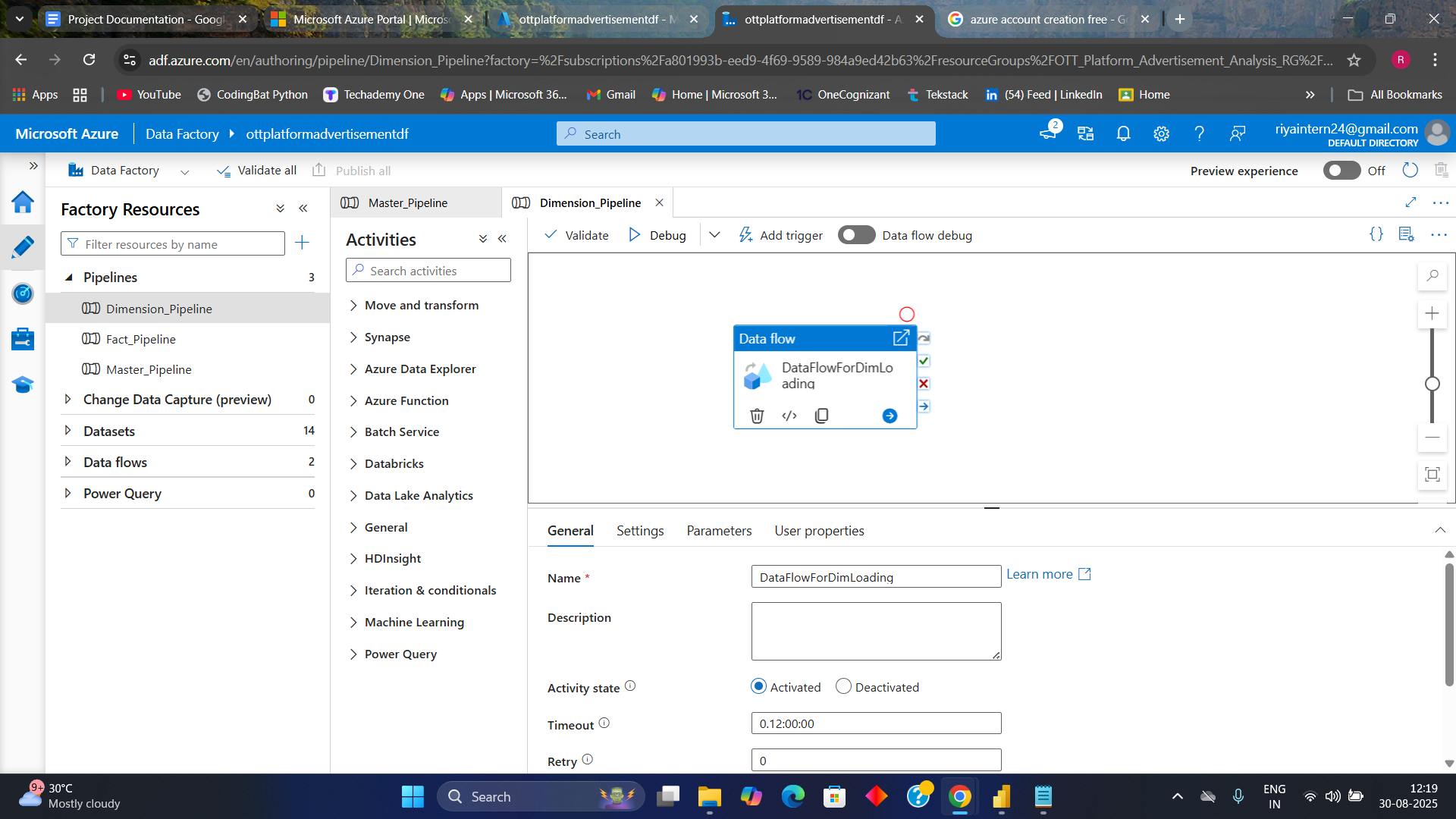
**Snapshot: Master Pipeline**



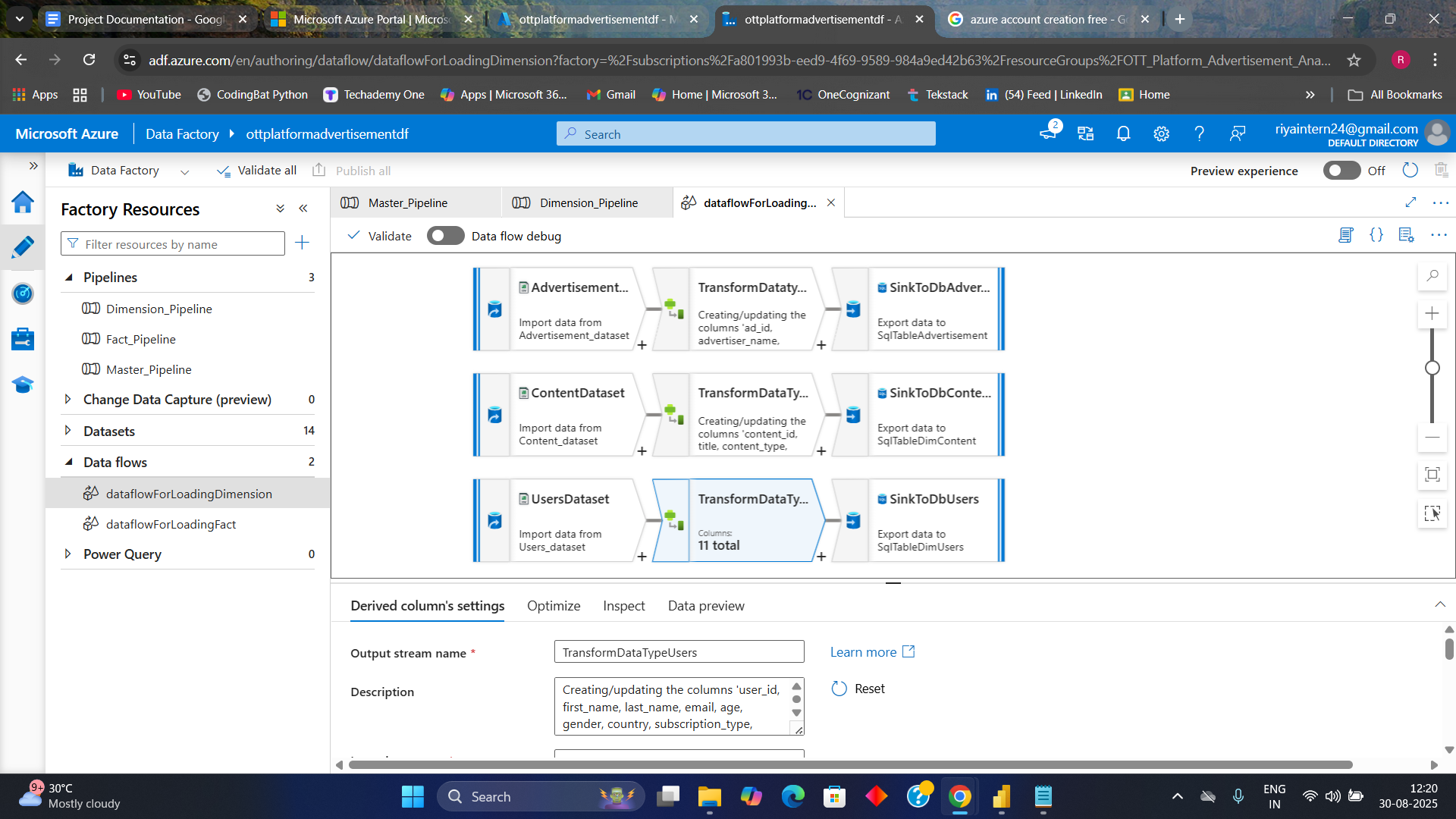
#### **2. Dimension Loading Pipeline (Dimension\_Pipeline)**

* **Purpose**:  
  This pipeline focuses on transforming and loading all **Dimension tables** that provide reference context for reporting (e.g., Users, Content, Advertisements, Sessions).
* **Activities**:  
  + Contains **1 Data Flow activity** to process dimension source data (CSV/Blob).
  + Performs **data cleansing** (removing nulls, handling invalid values).
  + Applies **schema mapping** and data type conversions.
  + Loads the output into SQL Database **Dimension Tables** (e.g., dim\_users, dim\_content, dim\_advertisement).
* **Flow Example**:  
   Raw Files (Blob Storage) → Data Flow (Transformation + Cleaning) → SQL Sink (Dimension Tables)

**Snapshots: Dimension Pipeline**



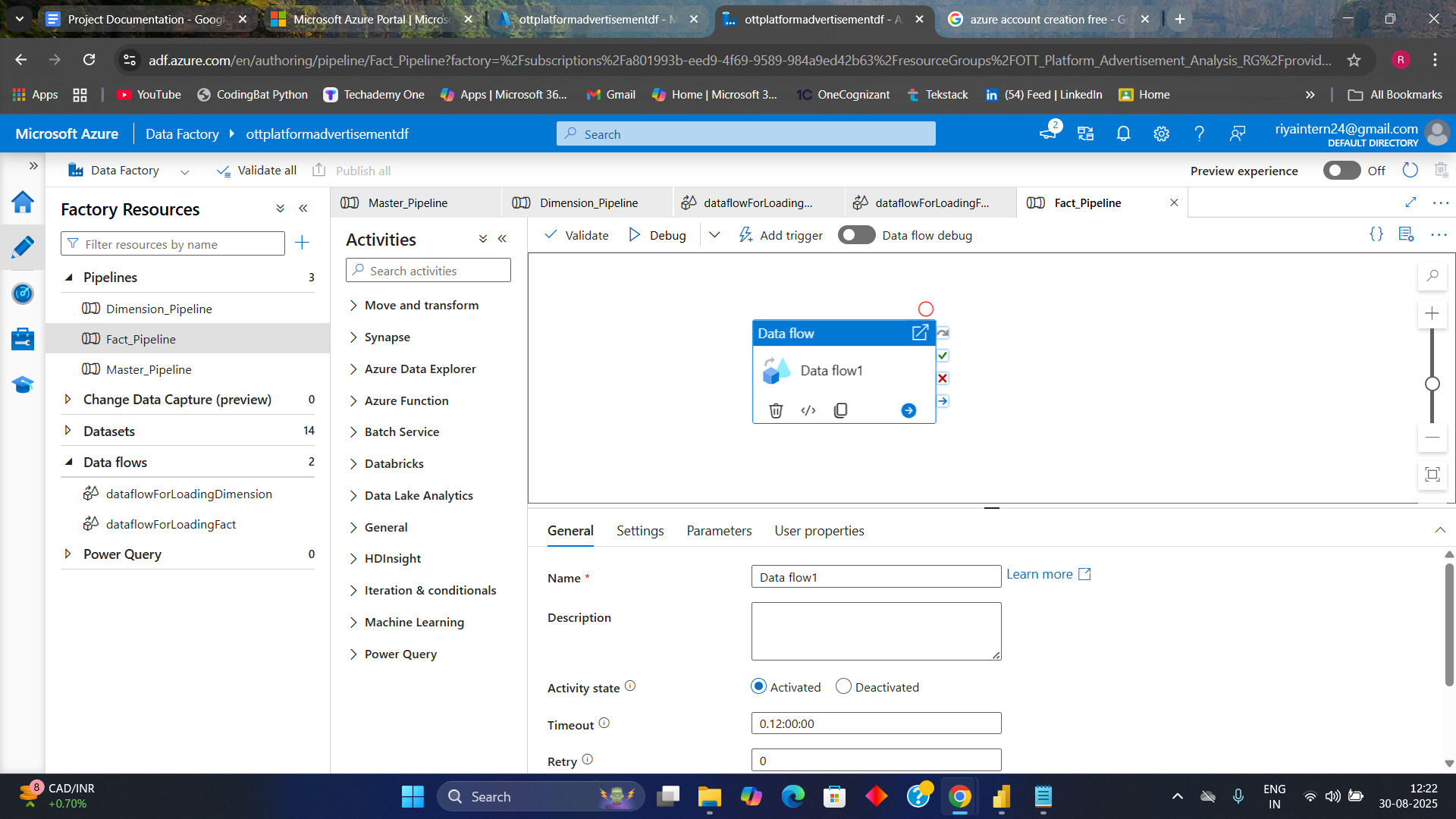
**Snapshot : Data Flow For loading Data into Dimension Tables**



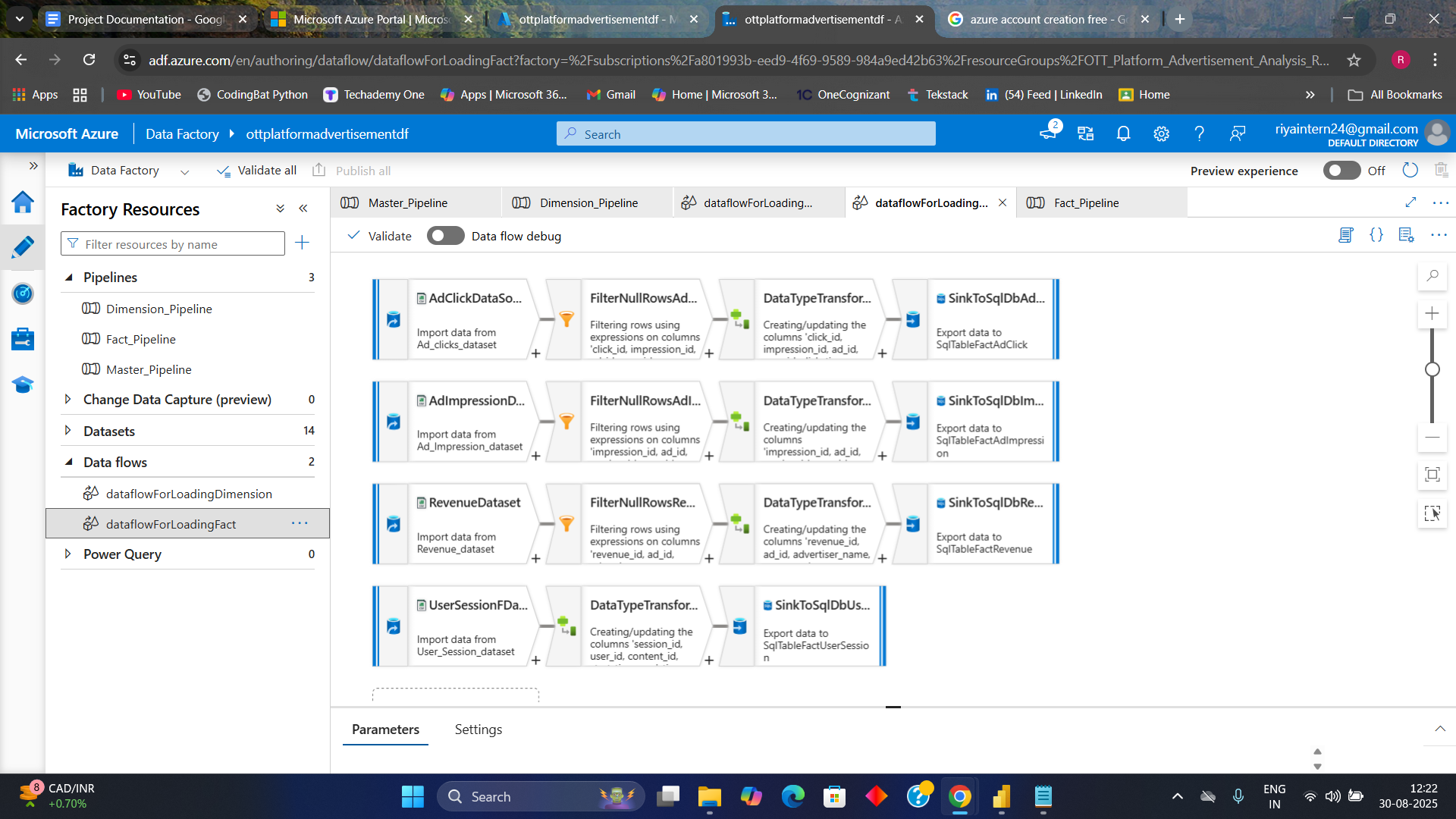
#### **3. Fact Loading Pipeline (Fact\_Pipeline)**

* **Purpose:**This pipeline is responsible for handling Fact data such as Ad Impressions, Clicks, Revenue, and User Engagement. These tables are linked to Dimension tables for analytics.
* **Activities:**
  + Contains 1 Data Flow activity that processes Fact-level events.
  + Filters raw fact data (removing incomplete sessions, invalid clicks).
  + Performs joins/lookups with Dimension tables (to ensure proper foreign key relationships).
  + Loads processed data into Fact Tables in SQL DB (fact\_ad\_impressions, fact\_clicks, fact\_revenue etc).
* Flow Example:  
  Raw Fact Data (Blob Storage) → Data Flow (Transformation + Joins with Dimensions) → SQL Sink (Fact Tables)

**Snapshot : Creating Fact Pipeline**



**Snapshot : Data Flow For loading Data into Fact Tables**

****

### **5.3.3 Publish and Trigger the Orchestration Pipeline**

* Validate pipelines (Author → Validate). Fix errors.
* Click **Publish All** to deploy.
* Create Trigger: Author → Triggers → + New → choose Schedule Trigger (daily/hourly) or Tumbling Window (for partitioned data).
* Attach trigger to Master\_Pipeline and start it.
* Test: Run pipeline manually first (Debug/Trigger Now) to confirm end-to-end behavior.

### **5.3.4 Monitor the Orchestration Pipeline Run**

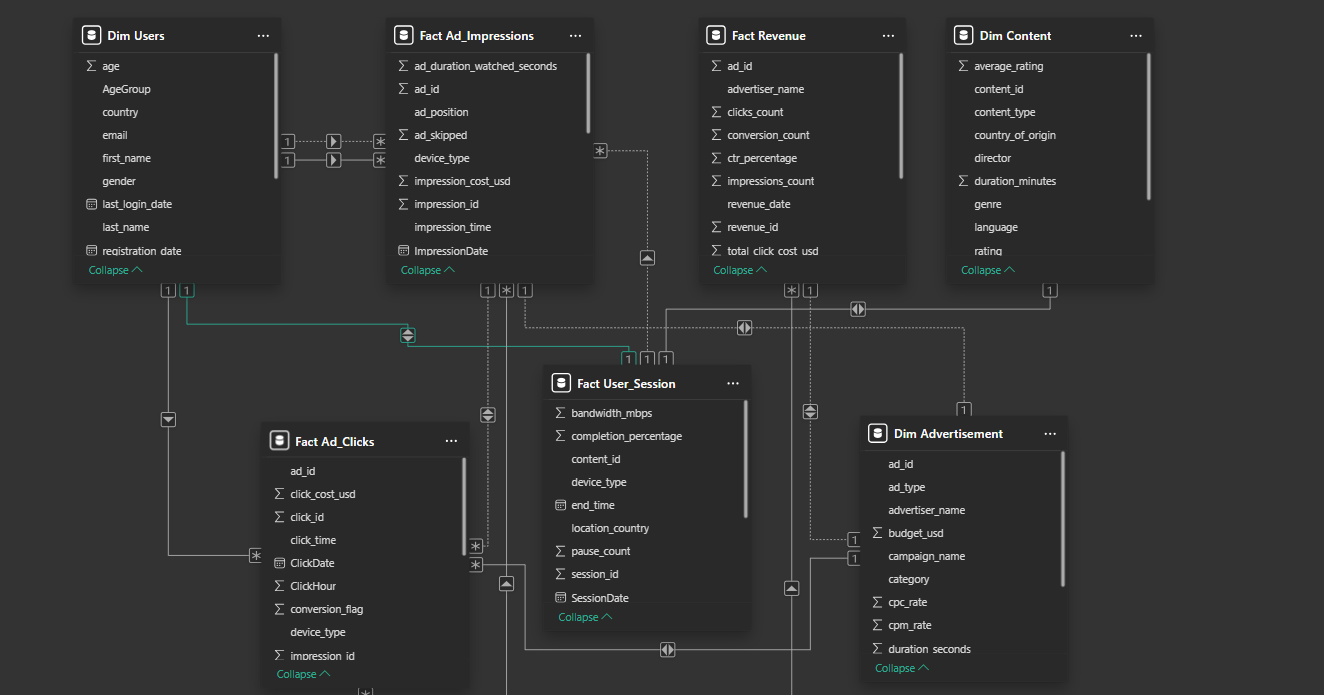
1. Use ADF Monitor (Author & Monitor) → Pipeline runs to see success/failure, duration, and activity details.
2. Click an individual run to view detailed activity logs and error messages.
3. For production: integrate with **Log Analytics** and for failures or high run times.
4. Maintain run history and store logs in Blob or Log Analytics for audit.

## **6. Data Storage (Azure SQL Database) —**

### **6.1 Schema Design and Normalization**

1. **Choose schema**: Star schema — Dimension tables (Dim\_Users, Dim\_Content, Dim\_Advertisement) and Fact tables (Fact\_Ad\_Impressions, Fact\_Ad\_Clicks, Fact\_Revenue, Fact\_Sessions).
2. **Design example tables** (columns):  
   * Dim\_Users(user\_id PK, age, gender, subscription\_type, registration\_date)
   * Dim\_Content(content\_id PK, title, genre, duration\_minutes, release\_year)
   * Fact\_Ad\_Impressions(impression\_id PK, ad\_id FK, content\_id FK, impression\_time, device, region)
3. **Primary & Foreign Keys:** define primary keys in dimension tables and reference them as foreign keys in fact tables to maintain referential integrity and enable efficient star schema joins.
4. **Normalization**: keep dimension attributes atomic (no repeated groups) and use lookup tables for categorical values (country, device).

**Snapshot: Data Model**



**7. Data Modeling & Analytics (Power BI)**

## **7.1 Data Modeling (Power BI)**

### **7.1.1 Connect Power BI to Azure SQL Database**

1. **Get connection details**: server name (servername.database.windows.net) and database name.
2. Open **Power BI Desktop** → Home → Get Data → Azure → Azure SQL Database.
3. Enter server and DB details; choose authentication method (Basic SQL Auth or Azure AD).
4. **Choose connection mode**:  
   * **Import** — data is imported into Power BI (faster visuals, supports full modeling).
   * **DirectQuery** — live queries to Azure SQL (saves memory, but query performance depends on SQL).

### **7.1.2 Build Relationships in Power BI**

1. In Power BI Desktop → **Model View**, To see loaded tables.
2. **Create relationships** by dragging a field from one table to another: e.g., Fact\_Ad\_Impressions.ad\_id → Dim\_Advertisement.ad\_id.
3. Set **Cardinality** correctly:  
   * Fact → Dimension should be **Many-to-One** (Many in fact, One in dimension).
4. Set **Cross filter direction**: default single direction is safest; use **Both** only when required (careful: can cause ambiguous filters and performance issues).
5. **Mark date table**: create a Date dimension table and mark it as the date table (Modeling → Mark as Date Table) so time intelligence DAX works correctly.
6. **Hide unnecessary columns** (IDs, audit columns) from report view to keep fields list clean.
7. **Validate relationships** by creating simple visuals to confirm filters/aggregations behave as expected.

## **7.2 Power BI Report Design**

To enable business stakeholders and advertisers to derive insights from the OTT platform’s advertisement data, three interactive dashboards were created in **Power BI**. Each page is designed to focus on a specific analytical goal, providing visibility into ad performance, user engagement, placement efficiency, and revenue.

### **📊 Page 1: Advertisement Performance by Show & User Segments**

**Purpose:** This page provides insights into how advertisements perform across different shows and user demographics.

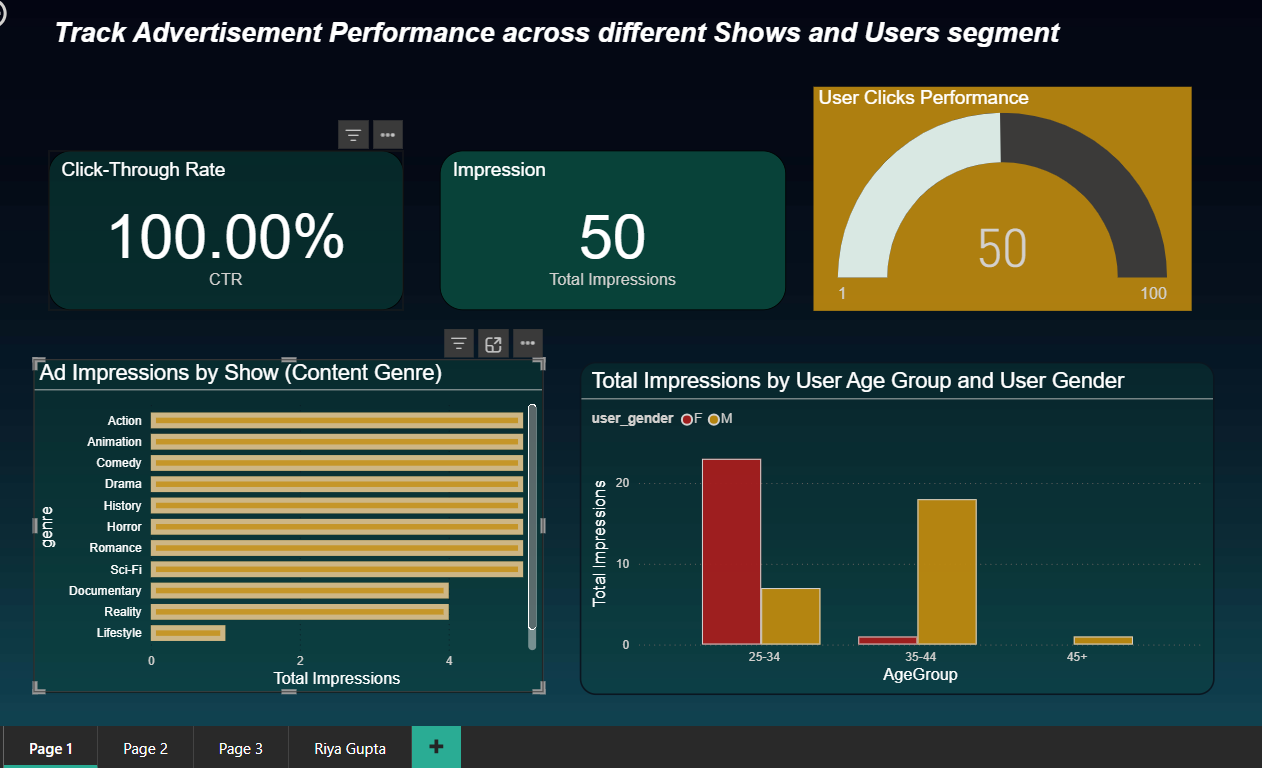
**Key Visuals & Metrics:**

* **KPI Cards:** Displays overall Click-Through Rate (CTR), total impressions, and total clicks.
* **Ad Impressions by Show (Content Genre):** Horizontal bar chart showing impressions distribution across genres like Action, Comedy, Drama, etc.
* **Total Impressions by User Age Group & Gender:** Clustered bar chart highlighting how different age groups and genders interact with ads.
* **Gauge Chart (Clicks):** Quick visualization of total clicks achieved against the target scale.

**Business Value:**

* Helps advertisers understand which shows deliver the highest engagement.
* Enables segmentation to tailor ad campaigns to specific audience groups.

**Snapshot: Advertisement Performance by Show & User Segments**



### **📊 Page 2: User Engagement & Ad Placement Optimization**

**Purpose:** This page analyzes user engagement with different ad types and evaluates ad placement (pre-roll, mid-roll, end-roll) to maximize effectiveness.

**Key Visuals & Metrics:**

* **CTR by Ad Type and Device:** Clustered bar chart comparing CTR performance of Video vs. Banner ads across devices (Laptop, Mobile, Smart TV, Tablet).
* **Avg. Ad Watch Seconds by Ad Type:** Bar chart showing watch time for Video and Banner ads, helping to measure ad stickiness.
* **CTR by Ad Position:** Bar chart comparing CTR for Pre-roll and Mid-roll placements.
* **Total Ad Skips by Ad Position:** Chart highlighting skip behavior, identifying which ad placement leads to higher skips.

**Business Value:**

* Identifies optimal ad positions with minimal skips and higher CTR.
* Supports better placement strategies to improve ROI.

### **Snapshot : User Engagement & Ad Placement Optimization**



### **📊 Page 3: Advertiser Revenue & Campaign Reporting**

**Purpose:** This page focuses on financial performance, providing insights into revenue generation and advertiser campaign effectiveness.

**Net Revenue by Advertiser (Bar/Column Chart)**

* Shows which advertisers bring in the most revenue.
* Example: Apple and American Express contribute a larger share compared to others.

**Total Revenue (Card/KPI Visual)**

* Displays the overall revenue generated from all ads combined.
* Example: Total revenue crosses **40K**.

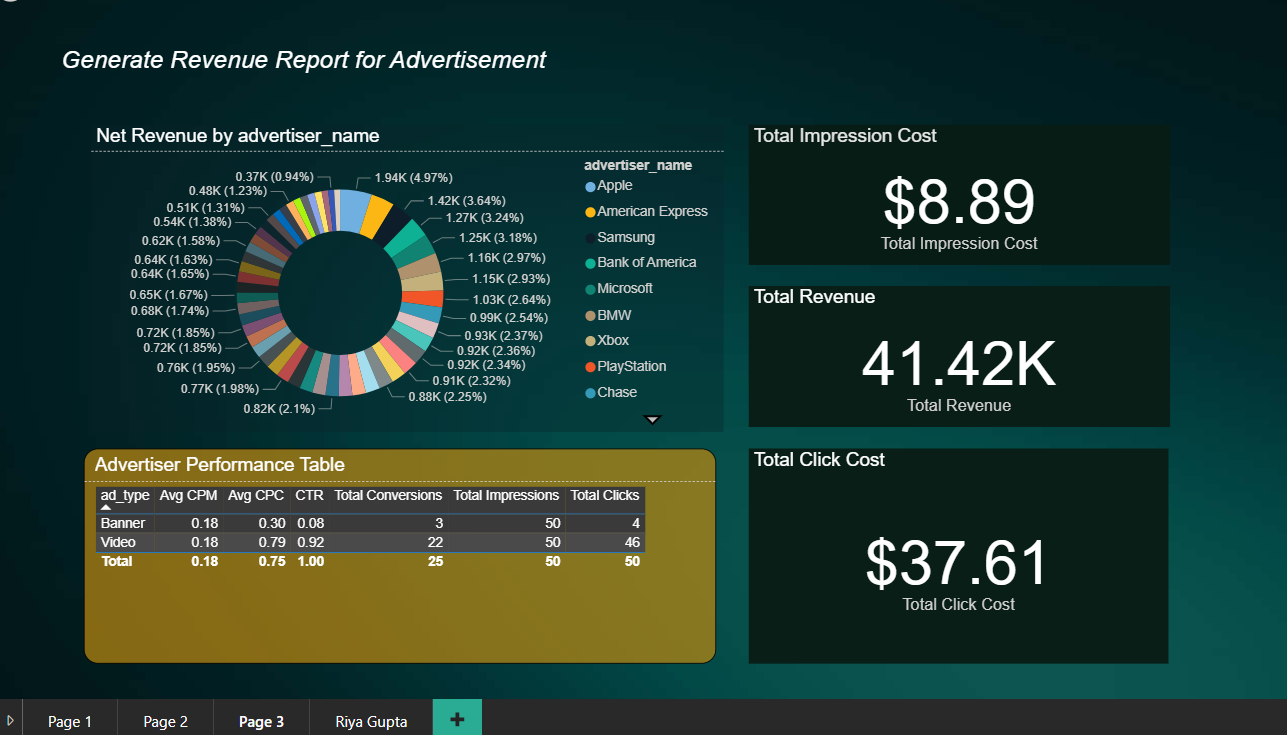
**Impression Cost & Click Cost (Cards)**

* Impression cost shows how much it costs to display ads.
* Click cost highlights expense when users actually engage.
* Example: Impression cost is very low compared to click cost.

**Ad Type Performance (Clustered Bar or Donut Chart)**

* The donut chart represents the **revenue share by each advertiser**, showing which advertiser contributes the most to overall earnings.

**Snapshot: Advertiser Revenue & Campaign Reporting**



## **8. Key Learnings**

During the course of this project, several important technical and business learnings were gained:

1. **Data Ingestion & Orchestration**
   * Learned how to design **pipelines in Azure Data Factory (ADF)** for structured ingestion, transformation, and loading of large-scale data.
   * Understood the importance of separating **raw, staging, and final layers** for data quality and governance.
2. **Data Transformation & Modeling**
   * Hands-on experience in building **data flows** for cleansing and filtering raw advertisement datasets.
   * Designed a **star schema** model with Fact and Dimension tables, applying **Primary Key and Foreign Key constraints** for consistency.
3. **Database & Storage Concepts**
   * Gained practical knowledge in **AzureSQLDatabase** for storing fact/dimension tables and writing queries.
   * Explored schema design patterns and ER diagram representation for data warehouse modeling.
4. **Analytics & Visualization**
   * Developed **interactive dashboards in Power BI**, focusing on a performance, user engagement, placement optimization, and advertiser revenue.
   * Understood how to define KPIs such as **CTR, impressions, clicks, revenue, and skips**, and represent them visually.
5. **Cloud Integration & Monitoring**
   * Learned to orchestrate pipelines using a **Master Orchestration pipeline** with retries, parameters, and logging.
   * Understood the role of **monitoring tools like Log Analytics & ADF run history** in ensuring pipeline reliability.

## **9. Future Enhancements**

This project can be extended with additional features to improve scalability and insights:

1. **Real-time Data Processing**
   * Integrate **Azure Event Hub or Kafka** with **Stream Analytics** to process live ad impressions and clicks instead of batch ingestion.
2. **Advanced Analytics with Machine Learning**
   * Implement **predictive models** (e.g., which ad placement is most likely to result in engagement).
   * Use **recommendation models** to suggest the best ad type for specific user segments.
3. **Integration with External APIs**
   * Pull **advertiser spend data** or external campaign metrics to provide a holistic 360° reporting view.
4. **Role-Based Access in Power BI**
   * Enable **Row-Level Security (RLS)** in Power BI dashboards to provide advertiser-specific reports securely.
5. **Automated Alerts & Notifications**
   * Implement email/Teams alerts when pipelines fail or when CTR/revenue drops below thresholds.
6. **Enhanced Data Quality Framework**
   * Add data validation checks at ingestion and transformation stages to reduce incorrect or missing values

## **10. Conclusion**

This project successfully demonstrated how **Azure Data Factory, Azure SQL Database, and Power BI** can be integrated to build an **end-to-end data analytics solution** for OTT advertisement performance.

* We ingested and transformed raw data into a structured star schema with fact and dimension tables.
* Pipelines were designed for loading, orchestration, and monitoring, ensuring data quality and reliability.
* Power BI dashboards provided insights into ad performance, user engagement, placement optimization, and advertiser revenue, enabling data-driven decision-making.