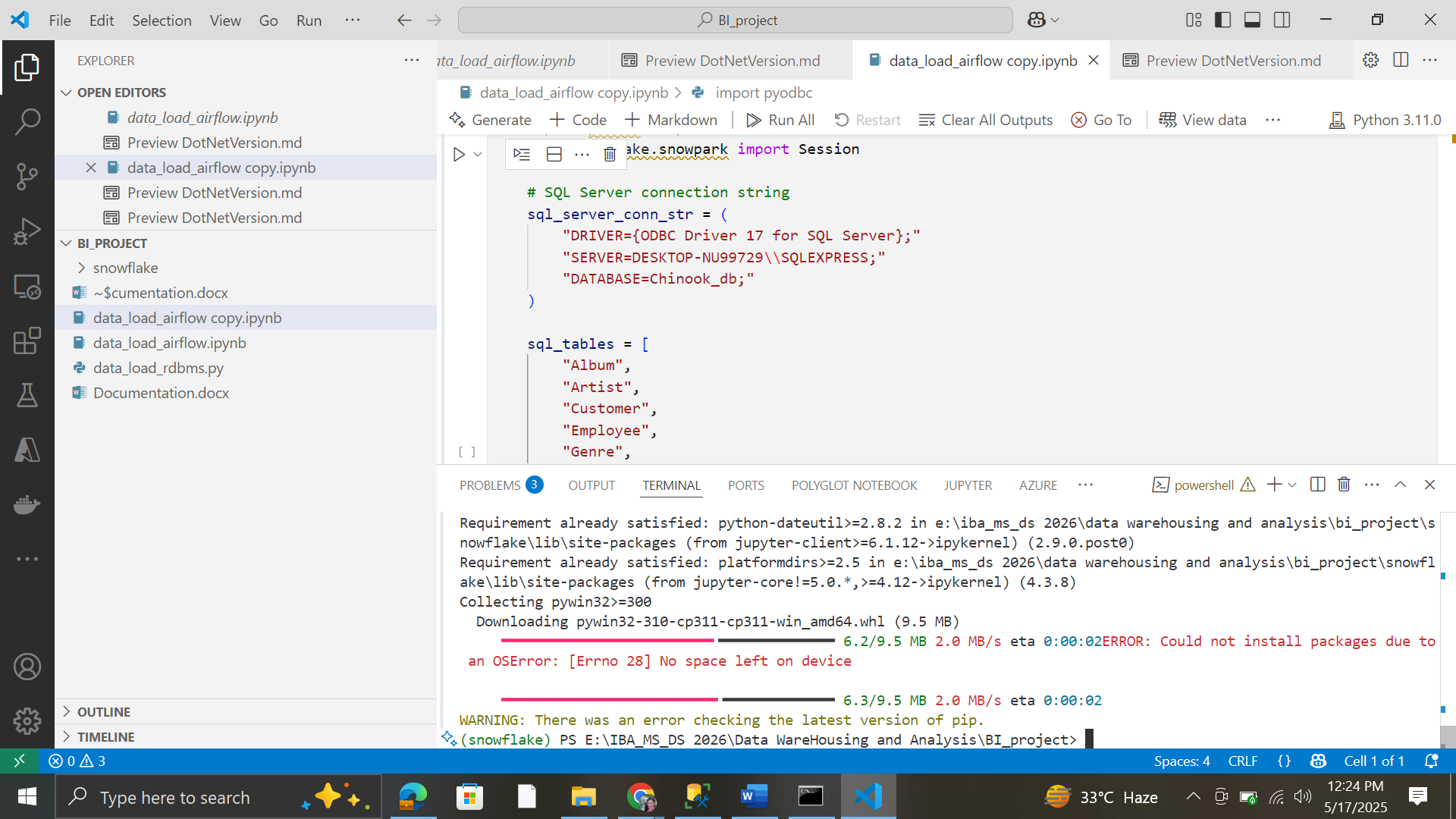
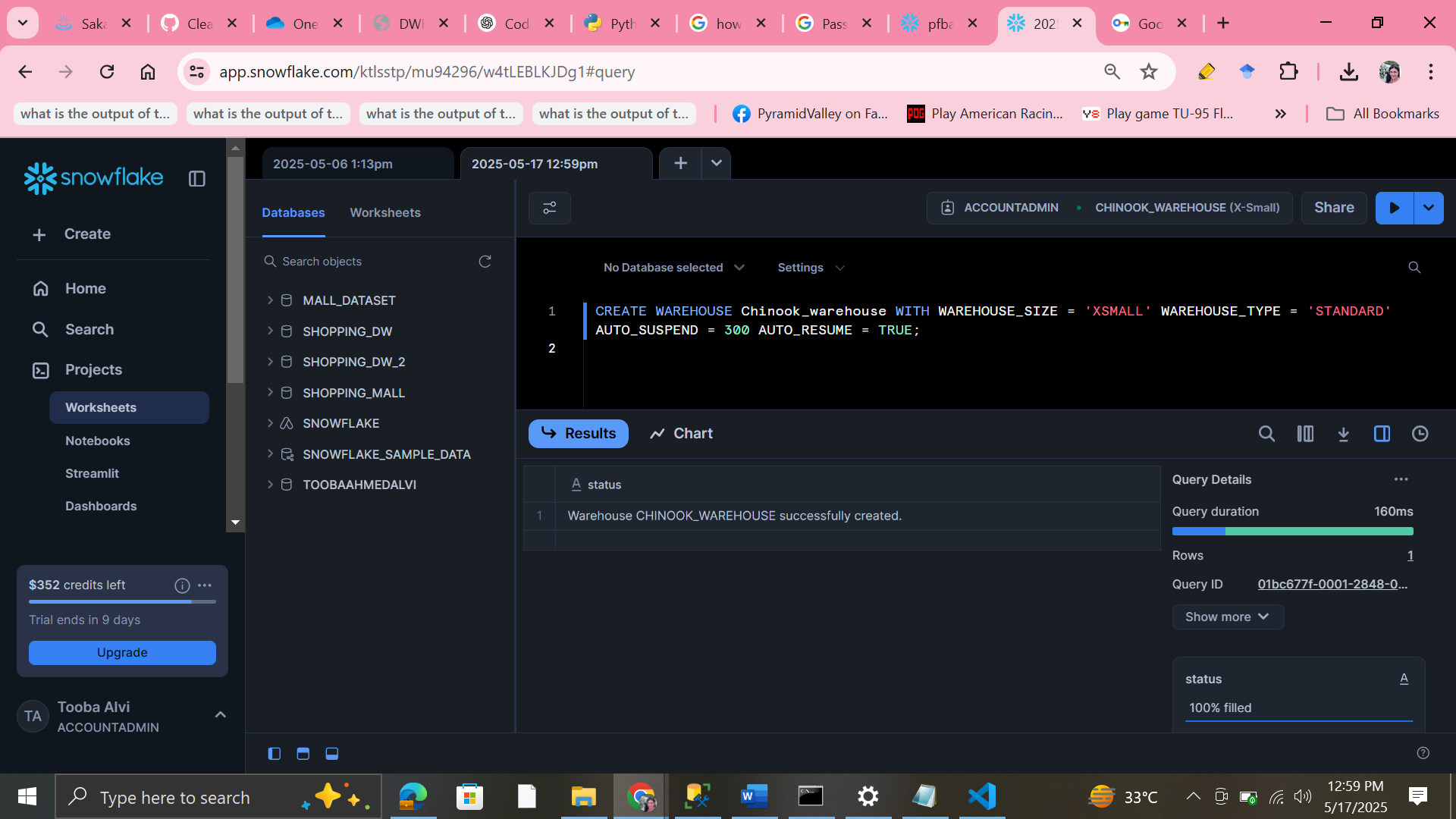


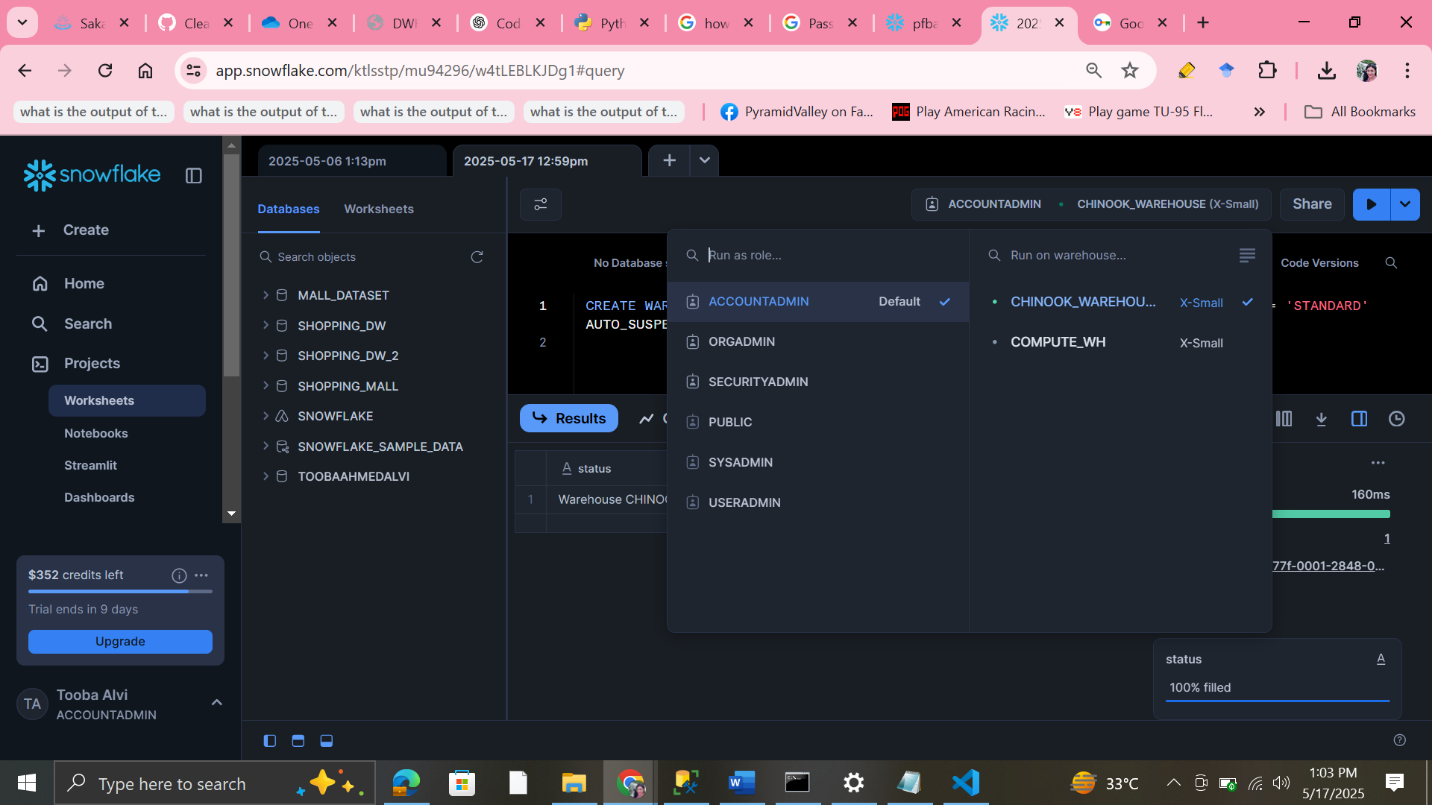
* 1. pyodbc — Connect to SQL Server
  2. pandas — Data manipulation
  3. snowflake-snowpark-python — Snowflake Python API
  4. python-dotenv — Load credentials from .env file

A screenshot of a computer

AI-generated content may be incorrect.









**What is RBAC?**

RBAC is a method to restrict system access to authorized users based on their roles.

In Snowflake, roles define **what actions a user can perform and what objects they can access** (databases, schemas, tables, warehouses, etc.).

**Key Concepts in Snowflake RBAC:**

* **Roles:** Logical collections of privileges.
* **Users:** Assigned one or more roles.
* **Privileges:** Permissions to perform actions on objects.
* **Role Hierarchy:** Roles can inherit privileges from other roles.

We are logged in as ACCOUNTADMIN (full access).

Our active warehouse for query execution is CHINOOK\_WAREHOUSE.

We can switch warehouses if you want to run queries on a different computer resource.

Creating a separate **cleaning stage** in Snowflake can help organize your pipeline better by clearly separating raw data ingestion from data cleaning/transformation.

**How we have structured it:**

1. **DATA\_INGESTION\_STAGE (Raw Landing Zone):**
   * This stage holds the raw, unprocessed data exactly as it arrives from source systems.
   * No changes or validations here; just a landing area.
2. **DATA\_CLEANING\_STAGE (Cleaning Zone):**
   * Create a new internal stage or schema where you load data from the raw ingestion stage after initial cleaning steps.
   * This is where you apply cleaning: remove duplicates, fix formatting issues, handle missing/nulls, apply validations, etc.
   * You can store cleaned intermediate tables here before final loading.
3. **FINAL\_SCHEMA (Production/Analytics Schema):**
   * After cleaning and transformation, load the final, clean data into production-ready tables here.
   * These tables are ready for consumption by BI tools or further processing.

**Benefits of adding a cleaning stage:**

* **Modularity:** Clear separation of concerns, easier to debug and maintain.
* **Data Quality:** You can isolate data quality checks and fixes in one place.
* **Auditing:** Easier to track which stage the data is in, and track changes over time.
* **Performance:** Transformation jobs can focus on smaller cleaned datasets, improving efficiency.

# STEP 1) CREATING INGESTION STAGE

