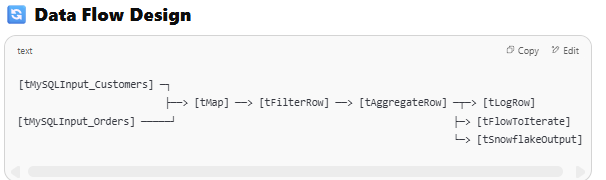
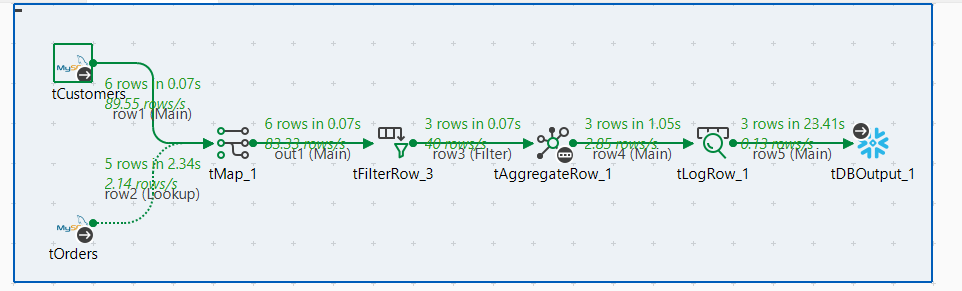
Exercise\_3: Average\_1

**Goal:**  
Extract customer and order data from **MySQL**, join and transform it with tMap, filter high-value orders using tFilterRow, aggregate total order amount per customer using tAggregateRow, count processed rows via tFlowToIterate, preview with tLogRow, and load into **Snowflake** using tSnowflakeOutput.





**✅ Components Used**

1. tMySQLInput (Customer Data)
2. tMySQLInput (Order Data)
3. tMap (Join + Transform)
4. tFilterRow (Filter high-value orders)
5. tAggregateRow (Aggregate total amount per customer)
6. tSnowflakeOutput (Final Load)
7. tLogRow (Preview Output)
8. tFlowToIterate (Count records)

**🧾 MySQL Sample Tables**

**📌 customers**

sql

CopyEdit

CREATE TABLE customers (

customer\_id INT PRIMARY KEY,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

email VARCHAR(100)

);

**📌 orders**

sql

CopyEdit

CREATE TABLE orders (

order\_id INT PRIMARY KEY,

customer\_id INT,

amount DECIMAL(10,2),

order\_date DATE

);

**🏗️ Talend Job Construction Step-by-Step**

**🔹 STEP 1: tMySQLInput – Customer Data**

**➤ Configuration:**

* DB Type: MySQL
* Query:

sql

CopyEdit

SELECT customer\_id, first\_name, last\_name, email FROM customers;

**🔹 STEP 2: tMySQLInput – Order Data**

**➤ Configuration:**

* Same DB settings
* Query:

sql

CopyEdit

SELECT order\_id, customer\_id, amount, order\_date FROM orders;

**🔹 STEP 3: tMap – Join + Transform**

**➤ Input Connections:**

* Connect tMySQLInput\_Customers and tMySQLInput\_Orders to tMap.

**➤ Join Setup:**

* Join orders.customer\_id == customers.customer\_id

**➤ Output:**

* Output table name: joined\_data
* Columns & expressions:
  + customer\_id → customers.customer\_id
  + order\_id → orders.order\_id
  + full\_name → customers.first\_name + " " + customers.last\_name
  + email → customers.email
  + amount → orders.amount
  + order\_date → orders.order\_date

**🔹 STEP 4: tFilterRow – High Value Orders**

**➤ Input: tMap output**

**➤ Condition:**

* amount > 300.00

**➤ Output:**

* Only high-value transactions passed to next component

**🔹 STEP 5: tAggregateRow – Total Order Amount Per Customer**

**➤ Group By:**

* customer\_id
* full\_name
* email

**➤ Operations:**

| **Output Column** | **Function** | **Input Column** |
| --- | --- | --- |
| total\_amount | Sum | amount |

**🔹 STEP 6: tLogRow – Preview Output**

**➤ Settings:**

* Mode: *Table*

**➤ Input: Connect from tAggregateRow**

**🔹 STEP 7: tFlowToIterate – Count Rows**

**➤ Purpose:**

Track total records processed

**➤ Configuration:**

* Key: recordCount
* Value: "1" (Each row increments counter)

**🔹 STEP 8: tSnowflakeOutput – Load to Snowflake**

Requires Snowflake JDBC Driver (install manually in Trial version)

**➤ Connection:**

* Account: <your\_account>.snowflakecomputing.com
* Warehouse: COMPUTE\_WH
* Database: YOUR\_DB
* Schema: PUBLIC
* Table: AGG\_CUSTOMER\_ORDERS

**➤ Schema Mapping:**

* customer\_id
* full\_name
* email
* total\_amount

**🔄 Data Flow Design**