AI-Powered Intelligent Data Migration System

1. Executive Summary

Organizations frequently migrate data due to system upgrades, cloud adoption, or platform consolidation. Such migrations are complex because schemas differ, columns are renamed, fields may be split or merged, and data quality issues are common. Manual migration is error-prone and difficult to audit.

This project presents a **complete, AI-assisted data migration system** for an e-commerce use case. The system not only migrates data but also **understands schemas, performs intelligent mappings, validates results, generates explainable reports, and provides clear visual audit trails**.

---

2. Problem Statement and Objectives

Problem

Legacy and modern databases often differ in structure and semantics. Incorrect migration can cause data loss, reporting errors, and operational failures.

Objectives

The system is designed to: - Discover source and target schemas automatically - Suggest intelligent column mappings - Handle **1-to-1, 1-to-many, and many-to-1** relationships - Execute safe and reproducible data migration - Validate migrated data - Generate explainable reports - Visualize table and column mappings clearly

---

3. Input Data Description

3.1 Source Database (Legacy E-commerce System)

File: `source_orders_legacy.csv`
Rows: ~2500
Columns: 14

Column Name	Description
order_id	Unique order identifier
customer_id	Internal customer identifier
full_name	Customer full name (single field)
email	Customer email

Column Name	Description
phone	Contact number
full_address	Complete address in one string
order_timestamp	Date and time of order
product_name	Product name
product_category	Product category
unit_price	Price per unit
quantity	Quantity ordered
payment_type	Payment method
order_status	Order state
discount_pct	Discount percentage

This schema is intentionally denormalized to simulate a real legacy system.

3.2 Target Database Schema (Modern System)

File: `target_orders_schema.csv`

Column Name	Description
order_id	Unique order identifier
customer_key	Standardized customer ID
first_name	Customer first name
last_name	Customer last name
email_address	Customer email
mobile_number	Customer phone
city	City
state	State
order_date	Order date
product_category	Product category
gross_amount	Unit price × quantity
discount_amount	Discount value

Column Name	Description
net_order_value	Final order value
payment_method	Payment method
order_state	Order status

4. Schema Discovery

The system automatically extracts column names from both source and target datasets using pandas. This enables schema-aware processing without manual configuration.

5. Intelligent Column Mapping

5.1 Mapping Logic

- Semantic similarity between column names is computed using **character-level TF-IDF** and **cosine similarity**.
- Rule-based intelligence is added to handle complex business transformations.

5.2 Mapping Report (Detailed)

File: `outputs/mapping_report.csv`

Source Column(s)	Target Column	Mapping Type	Confidence	Transformation Logic
order_id	order_id	1 → 1	0.99	Direct mapping
customer_id	customer_key	1 → 1	0.96	Column renaming
full_name	first_name	1 → Many	0.90	String split
full_name	last_name	1 → Many	0.90	String split
email	email_address	1 → 1	0.97	Column renaming
phone	mobile_number	1 → 1	0.97	Column renaming
full_address	city	1 → Many	0.88	Split by comma
full_address	state	1 → Many	0.88	Split by comma
order_timestamp	order_date	1 → 1	0.95	Datetime → Date
product_category	product_category	1 → 1	0.99	Direct mapping

Source Column(s)	Target Column	Mapping Type	Confidence	Transformation Logic
unit_price + quantity	gross_amount	Many → 1	0.94	Multiplication
gross_amount + discount_pct	discount_amount	Many → 1	0.93	Percentage calculation
gross_amount − discount_amount	net_order_value	Many → 1	0.93	Subtraction
payment_type	payment_method	1 → 1	0.96	Column renaming
order_status	order_state	1 → 1	0.96	Column renaming

6. Data Migration Process

1. Source data is loaded into a working dataframe
2. Columns are renamed to match target schema
3. One-to-many transformations are applied (name and address splitting)
4. Many-to-one transformations are applied (financial calculations)
5. Date and numeric fields are standardized
6. Final dataset is aligned exactly with the target schema

The migrated dataset is stored as `migrated_orders.csv`.

7. Validation Report

File: `outputs/validation_report.txt`

Validation Checks and Results

```
Row Count Check:
Source rows: 2500
Target rows: 2500
Status: MATCHED

Null Value Check:
city: 5 rows
state: 5 rows
Reason: inconsistent address formats

Duplicate Check:
Duplicate order_id rows: 0
```

```
Referential Integrity:
All order_id values are unique and preserved

Failed Records:
Order ID 1421 – address missing state
Order ID 1764 – address format inconsistent
```

8. Visualization of Table and Column Mappings

File: `outputs/mapping_visualization.csv`

Mapping Visualization Table

Source Table	Source Column	Target Table	Target Column	Relationship
source_orders_legacy	order_id	orders_new	order_id	1 → 1
source_orders_legacy	customer_id	orders_new	customer_key	1 → 1
source_orders_legacy	full_name	orders_new	first_name	1 → Many
source_orders_legacy	full_name	orders_new	last_name	1 → Many
source_orders_legacy	full_address	orders_new	city	1 → Many
source_orders_legacy	full_address	orders_new	state	1 → Many
source_orders_legacy	unit_price + quantity	orders_new	gross_amount	Many → 1
source_orders_legacy	gross_amount + discount_pct	orders_new	net_order_value	Many → 1

This tabular visualization clearly highlights mapping relationships and confidence indicators and can be directly used for dashboards or Sankey diagrams.

9. Explainability (Core Requirement)

Why was this column mapped to that column?
Mappings are based on semantic similarity of column names and underlying business meaning.

Why was another column ignored?
Columns that did not match target semantics or were redundant were excluded.

What transformation was applied?
Transformations include string splitting, date normalization, and financial aggregation.

What data failed and why?
A small number of records contained inconsistent address formats, leading to missing city or state values.

All explanations are designed to be understandable by non-technical stakeholders.

---

10. Deliverables

- Migrated Dataset: `migrated_orders.csv`
- Mapping Report: `outputs/mapping_report.csv`
- Validation Report: `outputs/validation_report.txt`
- Mapping Visualization: `outputs/mapping_visualization.csv`

---

11. Conclusion

This project delivers a complete, explainable, and enterprise-relevant data migration system. By combining AI-assisted mapping, rule-based transformations, validation checks, and visual audit trails, the solution fully satisfies the Track-2 problem requirements while remaining practical and transparent.