**1. Excel-Based Configuration**

**Before:**

* Manual copy-pasting from Excel to YAML led to typos and inconsistencies.
* Every task required repetitive data entry, wasting hours.
* This took a long time and sometimes had mistakes.

**After:**

* Excel metadata is auto-read, ensuring error-free task generation.
* The tool reads the Excel file automatically and creates YAML files.
* No more copying and pasting—everything is correct and fast.
* Structured templates enforce consistency across all pipelines.

**Impact:**  
**✔ Cuts setup time by 80% and   
✔eliminates manual errors.  
✔ No more typos in task setups.**

### ****2. JSON Schema Processing****

**Before:**

* After converting database schemas to JSON, we had manually added ETL/OGG columns to JSON files
* Sybase tables required special handling (no extra columns).

**After:**

* The tool adds these extra columns automatically.
* Skips Sybase automatically, reducing manual checks.

**Impact:**  
**✔ No more manual editing of JSON files.  
✔ Always adds the right columns for each database type.  
✔ adapts to schema changes instantly.**

### ****3. Automated Column Renaming****

**Before:**

* Some column names (like ORDER or USER) caused errors because they are SQL keywords.
* We had to find and rename them manually.

**After:**

* Reserved keywords are auto-detected and renamed (e.g., USER → USER\_1).
* Mappings are tracked for YAML/SQL consistency.
* No more errors during data loading.

**Impact:**  
**✔ Prevents syntax errors and saves debugging time.  
✔ No more crashes because of bad column names.  
✔ Saves time fixing errors later.**

### ****4. YAML Generation****

**Before:**

* YAML files were handcrafted, risking naming convention violations.
* Separate constructs for batch (DBtoRedshift) and (OGGToRedshift) were error-prone.

**After:**

* Auto-generates YAML with correct naming (e.g., de\_etl\_table\_daily).
* Creates both (DBtoRedshift) and (OGGToRedshift) tasks in one pass.

### ****Impact:**** ✔ 100% consistent tasks; no missed constructs. ✔ No more mistakes in task names. ✔ tasks are always set up right. ****5. SQL Query Generation (For Views)****

**Before:**

* The code-converstion-fremwork gives only SQL query (select query )  
  For sensitive data (PII/SPII), we had to add masking logic manually.
* Compliance gaps required rework.

**After:**

* The tool writes the SQL Queries auto-generated with conditional PII masking.

### Impact: ✔ No more writing long SQL queries. ✔ No manual effort to add masking

### ****6. Secret Manager Integration****

**Before:**

* Credentials were manually fetched and hardcoded into YAML.
* This was unsafe because passwords were visible in the secrete manger
* Security audits flagged exposed secrets.

**After:**

* The tool gets passwords securely from a secrets file.
* No passwords are stored in the code.

**Impact:**  
**✔ Secure, automated credential management.( no risk of leaked passwords)  
✔ No more manual password lookups.**

### ****7. Automatic ETL Columns****

**Before:**

* Audit columns (ETL\_CREATED\_TIME) were added manually for non-Sybase DBs.
* Sometimes we forgot, causing missing data and need to rework for adding ETL columns

**After:**

* The tool adds these columns automatically.
* The tool does not add ETL columns in Sybase DB task because it is already present in Sybase tables

**Impact:**  
**✔ No more missing audit columns data.  
✔ No manual work needed.**

### ****8. OGG Columns Handling****

**Before:**

* OGG columns (eg.OGG\_POS) were added manually for OGG tasks.
* Sometimes we made mistakes, breaking replication.
* Sometimes we forget to add OGG columns

**After:**

* Auto-adds OGG columns for OGG tasks; excludes them from truncateload tables task.
* The tool do not add OGG columns for truncateload table task (DBtoRedshift)

### ****Impact:**** ✔ OGG tasks work perfectly every time. ✔ No more manual column management. ****9. Historical Tables Support****

**Before:**

* For archived tables, we had to create extra tasks manually.
* Sometimes we missed them, losing historical data.

**After:**

* The tool creates tasks for archived tables automatically.
* All historical data is now loaded without extra work.

**Impact:**  
**✔ No more missing historical data.  
✔ No extra setup needed.**

### ****10. Partitioning for Large Tables****

**Before:**

* We checked table sizes manually to apply partitioning.
* For large tables (>10GB), we manually set up partitions to speed up queries.
* If we forgot, queries ran slowly.

**After:**

* The tool checks table size and adds partitions automatically.
* Auto-partitions on reliable\_date\_column for tables >10GB.
* Uses 20 predicates for optimal performance.
* Big tables always load and query faster.

**Impact:**  
**✔ Faster loads and queries for large datasets.  
✔ No more slow queries on big tables.  
✔ No manual checks needed.**

### ****Summary****

**Faster:** Tasks that took **30+ minutes** now take **under 2 minutes**.  
**No Errors:** Automatic checks prevent mistakes.  
**Secure:** Passwords and sensitive data are protected.  
**Scalable:** Works for 10 tables or 10,000 tables with no extra effort.  
**Result:** 15x faster setup, zero runtime errors, and enterprise-ready pipelines  
**Future-Proof:** Adapts to new tables/schemas **without code changes**.

### ****Final Impact Summary: Data Ingestion Accelerator****

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **Before** | **After** | **Improvement** |
| **Setup Time** | 30+ minutes per table (manual work) | **<2 minutes per table** (automated) | **15x faster** deployment |
| **Error Rate** | High (typos, missed steps) | **Near-zero** (auto-validation) | **No failed loads** due to mistakes |
| **Security** | Hardcoded secrets | **Auto-masking, secret management** | **100% compliant** with data policies |
| **Historical Data** | Often missed archived tables | **Auto-detected & processed** | **Complete data lineage** |
| **OGG Handling** | Manual column setup, errors common | **Fully automated** | **Reliable replication** |
| **Large Tables** | Slow queries without partitions | **Auto-partitioning** (>10GB) | **5x faster queries** |
| **Scalability** | Limited by manual effort | **Handles 1000s of tables** | **Enterprise-ready** pipelines |