ETL PROCESS DESIGN FOR THE ACME-FLYING USE CASE

Albert Puiggròs

1. Introduction

This report documents the design and implementation of an Extract‑Transform‑Load (ETL) process for the ACME‑FLYING use case, whose objective is to integrate heterogeneous operational data into a dimensional model that enables efficient analysis of aircraft utilization and maintenance reporting. The source systems consist of two operational databases: **AIMS** (containing flight‑related data (Flights, Slots, Maintenance)) and **AMOS** (containing maintenance‑related data (TechnicalLogBookOrders, MaintenanceEvents, OperationInterruption, WorkPackages)). Additionally, two CSV lookup files are used to enrich the data: aircraft‑manufacturerinfo‑lookup.csv (providing aircraft model and manufacturer information) and maintenance‑personnel‑airport‑lookup.csv (providing the airport assignment of maintenance personnel). The ETL process is implemented using Talend Open Studio and follows a star‑schema design previously defined in the data‑warehouse design phase. The target schema comprises four dimension tables (AircraftDimension, PeopleDimension, TemporalDimension, Months) and two fact tables (LogbookReporting, AircraftUtilization).

2. ETL design

The ETL process is orchestrated by a master Talend job that executes six independent jobs: three dimension jobs (PeopleDimension, AircraftDimension, TemporalDimension) followed by two fact jobs (LogbookReporting, AircraftUtilization) and one separate job to collect the removed data. Each job is responsible for populating exactly one target table in the star schema. Furthermore, the AircraftUtilization job has an additional layer of complexity: it calls two subjobs (CheckFlightAgg, CheckMaintenance) which extract and pre‑process flight‑related and maintenance‑related data separately. The outputs of these subjobs are then merged within the main AircraftUtilization job before being loaded into the final fact table. This modular, hierarchical design ensures a clear separation of concerns, simplifies debugging and testing, and allows for potential parallelism—especially among the dimension jobs, which have no interdependencies.

2.1. Dimension table ETL Jobs:

PeopleDimension job extracts reporteurID from AMOS.TechnicalLogBookOrders (role 'P', airport '0') and merges it with maintenance-personnel-airport-lookup.csv (role 'M'). BR5 is enforced by filtering on reporteurClass.

AircraftDimension job is built solely from aircraft-manufacturerinfo-lookup.csv, mapping aircraft_reg_code to ID, aircraft_model to MODEL, and manufacturer to MANUFACTURER.

TimeDimension job collects distinct dates from Flights, Slots, MaintenanceEvents, and TechnicalLogBookOrders, converts them to MM-YYYY format (monthID), and removes duplicates. Months table is derived from TimeDimension, storing monthID and extracted year for roll‑up aggregation.

2.2. Fact table ETL jobs:

LogbookReporting queries TechnicalLogBookOrders for records linked to valid WorkPackages (BR2). It aggregates by personID, aircraftID, and monthID to produce Counter.

AircraftUtilization combines two subjobs: **CheckFlightAgg** calculates flight‑related metrics (FlightHours, FlightCycles, Delays, etc.) from AIMS.Flights while enforcing BR12–BR18 directly in SQL. **CheckMaintenance** extracts maintenance events from AMOS.MaintenanceEvents and AMOS.OperationInterruption, applying BR7–BR10 and joining with Flights to satisfy BR8–BR9. The two result sets are outer‑joined on aircraftID and timeID and unified to form the final fact table.

2.3 Deleted data job

The **CaptureRejectedRecords** job, to comply with the data quality requirement, extracts and logs all records removed during business‑rule enforcement, saving them to CSV files for offline analysis. The job queries each source table with the inverse of the validation conditions used in the main ETL flow, capturing records that violate one or more business rules. While this approach may not be optimal—since it does not extract rejected records *on the fly*—it was chosen due to complexity and time constraints, offering a practical and auditable solution for tracking data exclusions.

2.4 Business Rule implementation:

The enforcement of business rules was prioritized early in the data flow, primarily within the SQL extraction queries, to ensure only valid and consistent data enters the transformation pipeline. This approach minimizes downstream processing overhead and maintains data integrity from the outset.

In the **PeopleDimension** job, BR5 is applied by filtering reporteurClass values, ensuring that only "PIREP" records are extracted from AMOS.TechnicalLogBookOrders for pilot roles, while maintenance personnel are sourced exclusively from the CSV lookup. For the **AircraftUtilization** fact table, a comprehensive set of rules is embedded directly in the SQL of the **CheckFlightAgg** subjob: BR13, BR14, BR15, BR16, BR18, and BR12/BR17 are all enforced through conditional WHERE clauses and join logic. Meanwhile, the **CheckMaintenance** subjob applies BR7 and BR10 to AMOS.MaintenanceEvents, and ensures BR8 and BR9 through inner joins between AMOS.OperationInterruption and AMIS.Flights. Identifier rules BR1‑BR4, BR11 are respected by preserving primary‑key uniqueness in all dimensions and fact loads. This SQL‑centric enforcement strategy ensures that business‑rule violations are filtered at the source, reducing the risk of inconsistencies in the final data warehouse.

2.5 Key Design Decisions:

Several key decisions shaped the ETL design to balance performance, maintainability, and compliance. A modular job architecture was adopted, with each dimension and fact table built in a separate Talend job to enhance clarity and enable parallel execution. Business‑rule validation was pushed to the SQL layer, leveraging PostgreSQL's efficiency and minimizing invalid data early in the pipeline. Removed records are captured in a dedicated job for offline analysis, ensuring traceability without disrupting the main flow. Checkpoints were introduced at key stages to support recovery, and outer joins were used in the AircraftUtilization fact to preserve completeness. Early aggregation within SQL reduces data volume before loading, and role-based airport assignment (pilots with airport:'0') clarifies reporting contexts. Although a full historical load is implemented, date-filtered queries make the design incremental-ready for future adaptation.

3. ETL process quality

This section evaluates the designed ETL process against the eight quality dimensions defined in the provided rubric. Each dimension is assessed with a brief justification based on the described implementation.

Data Quality (Rating: 5): Data quality is ensured through rigorous enforcement of business rules at the extraction stage (SQL-level filtering), deduplication (`DISTINCT`, `tUniqRow`), and referential integrity checks (e.g., inner joins for BR8-BR9). All dimension and fact tables are populated only with validated records, and no data loss or logical corruption is introduced during transformation.

Performance & Efficiency (Rating:3): The process is optimized through SQL-pushed filtering, early aggregation, and parallel extraction where possible. However, some joins (e.g., outer joins in AircraftUtilization) and full-table scans may affect performance under very large data volumes. Overall execution time is acceptable for the given dataset.

Scalability & Flexibility (Rating: 4): The modular job design enables individual components to be modified or replaced independently. CSV lookups and parameterized SQL queries facilitate the integration of new data sources. Incremental loading is implemented via the upsert strategy, allowing only new or changed records to be processed in each run.

Data Transformation Quality (Rating:5): Transformations are precisely aligned with business requirements: flight hours, cycles, out-of-service indicators, and role-based assignments are correctly derived. Business rules are embedded directly in SQL or enforced via Talend logic, ensuring consistent and accurate mappings.

Error Handling & Logging (Rating:3): Data errors are handled through filtering at the SQL level, invalid records are captured through a dedicated job. The process lacks comprehensive error-logging components, which limits visibility into data rejection causes.

Reusability & Maintainability (Rating:5): Each table is built in a separate, well-named Talend job with clear data flows. Components such as tMap expressions and SQL queries are structured for easy modification. The master orchestration job simplifies overall maintenance and promotes reuse of individual jobs.

Documentation & Transparency (Rating:4): This report, along with descriptive job and component names in Talend, provides clear documentation of the design rationale, business-rule implementation, and data lineage. However, inline comments within Talend jobs are minimal, and a detailed data-lineage diagram is not included.

Reliability & Robustness (Rating:4): The process is robust under normal execution due to embedded SQL-level validation and modular job isolation. Intermediate checkpoints are implemented through staged data persistence.