ISYS90086-Data Warehousing Assignment 2 Report

Overhill Winery Data Warehouse Implementation

Chaitra Chandra Mouli <Student Id: 977313>
Daniel Gil <Student Id: 905923>

Contents

1 Executive Summary

Overhill winery is a medium sized winery located on the Bellarine peninsula, 150 km away from Melbourne. Currently they have three established markets : locally in Melbourne (Victoria), interstate in Australia and internationally in the United Kingdom. The winery has two separate information systems to manage production and sales, that is difficult to manage because of different database management systems.
With expanding markets and business along with increasing demands, the winery is moving towards data driven decision making for the business and implementing the Data Warehouse for achieving the business goals.
The Data Warehouse designed previously is implemented with provided source data and design changes are made to accommodate the data given and store the information in the Data Warehouse tables using the Extract Transform Load (ETL) process.
Data from various or multiple sources are integrated and extracted to gather information necessary to transform and store in the Data Warehouse.
The growing business and expanding markets are also handled in the Data Warehouse by implementing Slowly Changing Dimensions (SCDs) of various Types of history handling.
Data is transformed using different techniques such as, data cleansing, formatting, splitting, merging, computing, and history handling, depending on the source data and the target Data Warehouse table.
The extracted and transformed data is then loaded/inserted into the Data Warehouse tables, various Dimension Tables or the Fact tables, depending on the transformations and data source.
While implementing the ETL process for the Data Warehouse with the provided source data, various issues and challenges faced regarding data inconsistencies, missing values, data quality issues, data format changes and other problems are handled and consistent and quality data is loaded and stored into the Data Warehouse, with minimum data loss. Additional information for technical and business users are described in a metadata repository in order to facilitate the execution process and generate queries in the data warehouse.
The various aspects of the processes involved above are described in detail below in this report.

2 Design of the ETL Process

2.1 Data Quality

Decision support systems that use data from various external sources, the quality of the data obtained for the decision making process is not the best, since the data is very different from the original data source and this impacts the understanding of the data context.
The process of improving the quality of the data that is used for better decision making process is quite hard and complex since obtaining better quality data is difficult. Instead, the decision making process can be improved based on the quality of the available quality of data to provide decision makers with information regarding the quality of data and adapting the decision making process and strategy. [1]

2.2 Data Extraction, Transformation and Loading (ETL)

- **Extraction**
  The extraction of data from various sources in specific time periods that can be daily, weekly, monthly, quarterly, seasonally or annually. The design of the ETL process considers data extracted in 2018 as an initial load and subsequent incremental loads using data from 2019.
  The data from various source systems are analysed for missing values, ascertained for correct values and other initial analysis during the extraction and before transformation.

- **Transformation**
  Transforming the data into well organised and defined data structures by selecting, splitting or joining, converting, formatting, summarising, enriching, decoding, mapping and cleansing the data extracted.
  The duplicate records are removed and keys are restructured and all the units and measurements are standardised to consolidate the data into the Data warehouse.

- **Loading**
  Loading the data into the Data Warehouse once the transformation of the data is complete. Loads can be initial load (populating the Data Warehouse for the first time) or incremental load (applying ongoing updates periodically to the Data Warehouse). In addition, history is carefully preserved using techniques for slowly and rapid changing dimensions to ensure analysis completeness.

The ETL process follows a sequential execution of transformation involving all the necessary conformed dimensions and the fact tables.
The transformations are executed through initial and incremental jobs and they are explained in detail in the coming sections on metadata.

2.3 Market Dimension Transformation

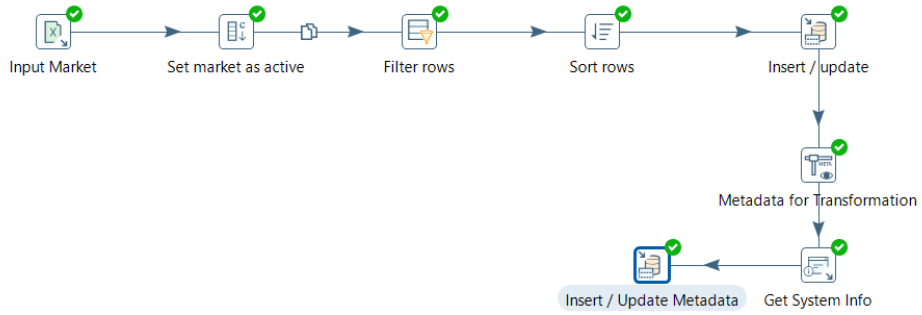


Figure 1: Market Transformation

The Market transformation involves the following steps:

- Data is extracted from Market excel input file.
- A constant flag *isActive* is set to 1 against each market.
- The data is filtered for null and missing values.
- Data is sorted according to the market key and name.
- Slowly Changing Dimension of Type 1 is implemented to update the changes in market, for example, if the market description changes or is set as inactive, then the correspondent value in the dimension is also updated. Also the data is loaded/inserted into the DimMarkets Table in the Data Warehouse.
- Metadata for the transformation is extracted.

2.4 Time Dimension Transformation

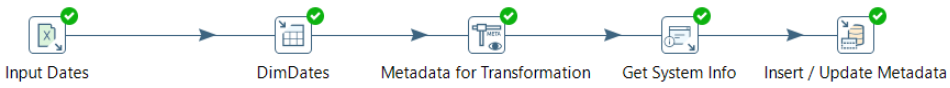


Figure 2: Time Transformation

The Time transformation involves the following steps:

- Data is extracted from DimDates excel input file.
- The data is loaded/inserted into the DimDates Table in the Data Warehouse.
- Metadata for the transformation is extracted.

2.5 Agent Dimension Transformation

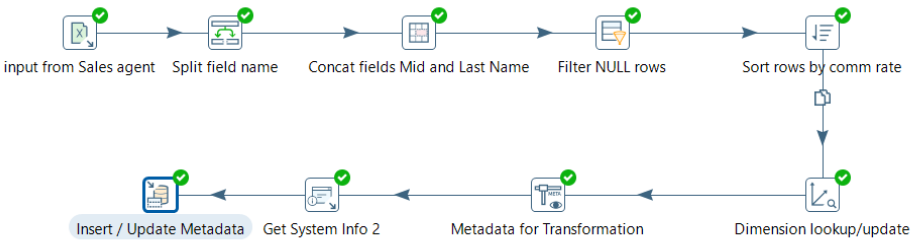


Figure 3: Agent Transformation

The Agent transformation involves the following steps:

- Data is extracted from SalesAgent excel input file.
- The AgentName is split into first, middle and last names.
- The middle and last names are concatenated into LastName field.
- The data is filtered for null and missing values.
- Data is sorted according to the commission rates.
- Slowly Changing Dimension of Type 2 is implemented to update the changes in market with Agents start and finish dates and versions. Also the data is loaded/inserted into the DimAgents Table in the Data Warehouse.
- Metadata for the transformation is extracted.

2.6 Production History Dimension Transformation

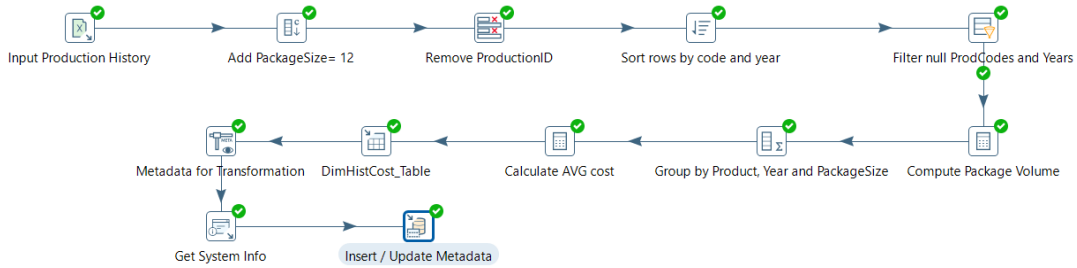


Figure 4: Production History Transformation

The Production History transformation involves the following steps:

- Data is extracted from ProductionHistory excel input file.
- The PackageSize of dozen is added per product.
- The ProductionID is removed from the data.
- The data is sorted according to ProductCode and ProductionYear.
- The data is filtered for null and missing values.
- The Package Volume is computed as PackageVolume = Volume / PackageSize.
- The data is grouped by Product, Year and the size.
- The AverageCost is computed as AverageCost = Cost per package / PackageVolume.
- The data is loaded/inserted into the DimHistCost Table in the Data Warehouse.

2.7 Production Dimension Transformation

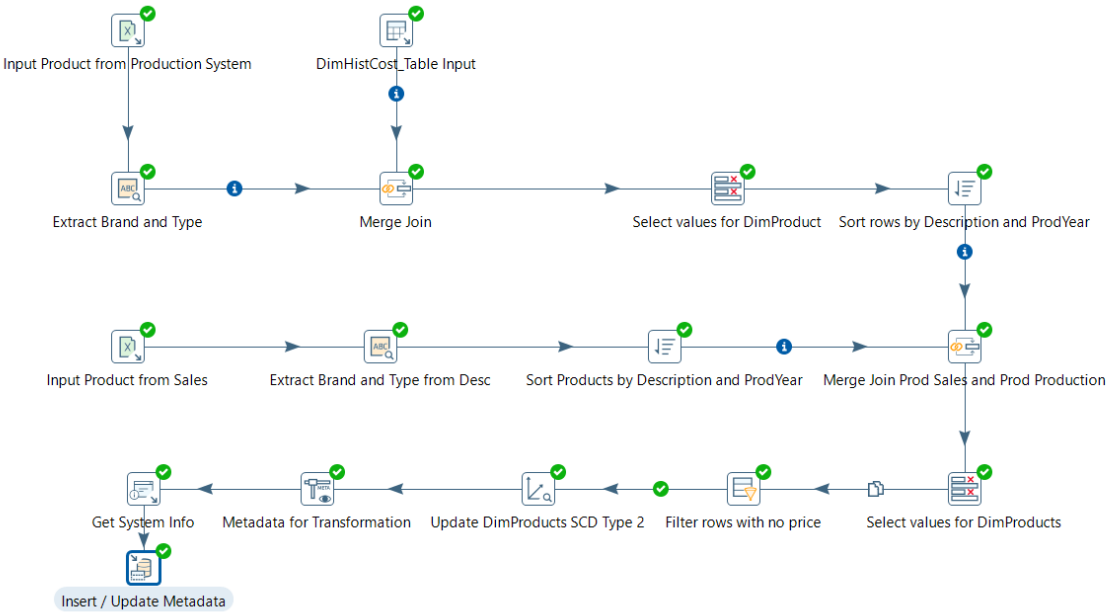


Figure 5: Production Transformation

The Production transformation involves the following steps:

- Data is extracted from Product excel input file from the Production System.
- The Brand and Type of product is extracted from Description.
- The data from DimHistCost table is extracted.
- Both the input data is merged (INNER JOIN) according to ProductCode.
- Fields important for the product table are only selected.
- The data is sorted according to production year and description.
- Data is extracted from Product input file from Sales System.
- The Brand and Type of product is extracted from Description.
- The data is sorted according to production year and description.
- Both the processed data of Products from sales and production system are merged (FULL OUTER JOIN) based on description, group and production year.
- Fields important for the product table are only selected.
- The data is filtered for null and missing values.
- Slowly Changing Dimension of Type 2 is implemented to update the changes in Product prices with Product Price start and finish dates and versions. Also the data is loaded/inserted into the DimProducts Table in the Data Warehouse.
- Metadata for the transformation is extracted.

2.8 Initial Customer Dimension Transformation

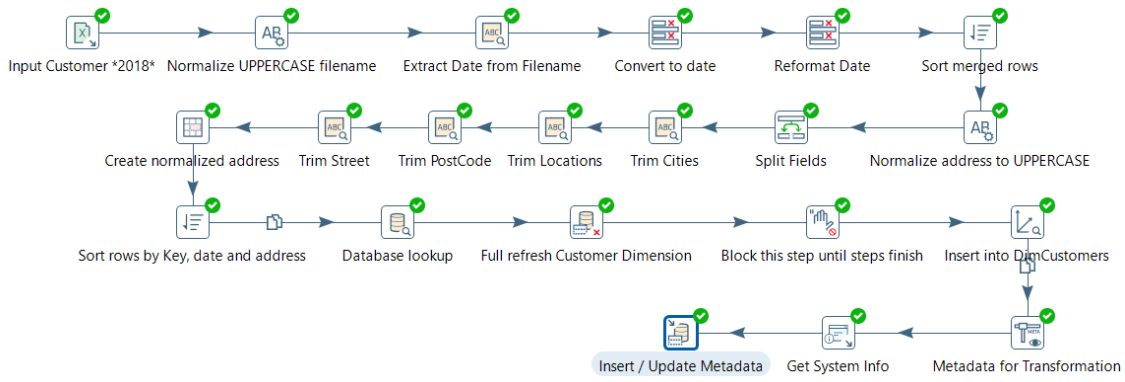


Figure 6: Initial Customer Transformation

The Initial Customer transformation involves the following steps:

- Data is extracted from Customer Jan2018 excel input file. A regular expression in the data extraction captures Customer file in 2018. As a consequence, one (1) file is extracted.
- The filename is normalised to uppercase.
- The date from the filename is extracted.
- The date extracted is converted to date format and re-formatted.
- The data is sorted.
- The Address is normalised to uppercase.
- Address is split into Street, Location, City and Postcode.
- The Cities, Postcode, Location and Street data fields are trimmed.
- The address is normalised and sorted according to dates, key and address.
- The city field is selected, sorted and loaded/inserted into DimCities Table in the Data Warehouse.
- The location is sorted and unique locations are loaded/inserted into DimLocations Table in the Data Warehouse.
- A location lookup is done from the DimLocations table and sorted Customer data and Customer Dimension is refreshed and waited until all the loads are complete .
- The Customer data is loaded/inserted into the DimCustomers Table in the Data Warehouse.
- Metadata for the transformation is extracted.

2.9 Incremental Customer Dimension Transformation

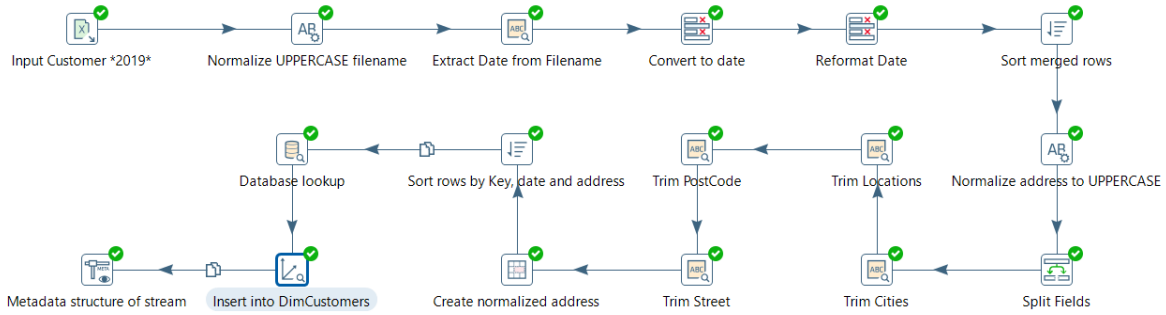


Figure 7: Incremental Customer Transformation

The Incremental Customer transformation involves the following steps:

- Data is extracted from all Customer 2019 excel input files. A regular expression in the extraction consider files in the data directory only for 2019. As a consequence, two (2) files are extracted.
- The filename is normalised to uppercase.
- The date from the filename is extracted.
- The date extracted is converted to date format and re-formatted.
- The data is sorted.
- The Address is normalised to uppercase.
- Address is split into Street, Location, City and Postcode.
- The Cities, Postcode, Location and Street data fields are trimmed.
- The address is normalised and sorted according to dates, key and address.
- A database lookup is done from the DimLocations table and the Customer LocationID is returned for the matching location and postcode .
- The Customer data is loaded/inserted into the DimCustomers Table in the Data Warehouse using SCD Type 2 to update the address.
- Metadata for the transformation is extracted.

2.10 Transaction Sales Fact Transformation

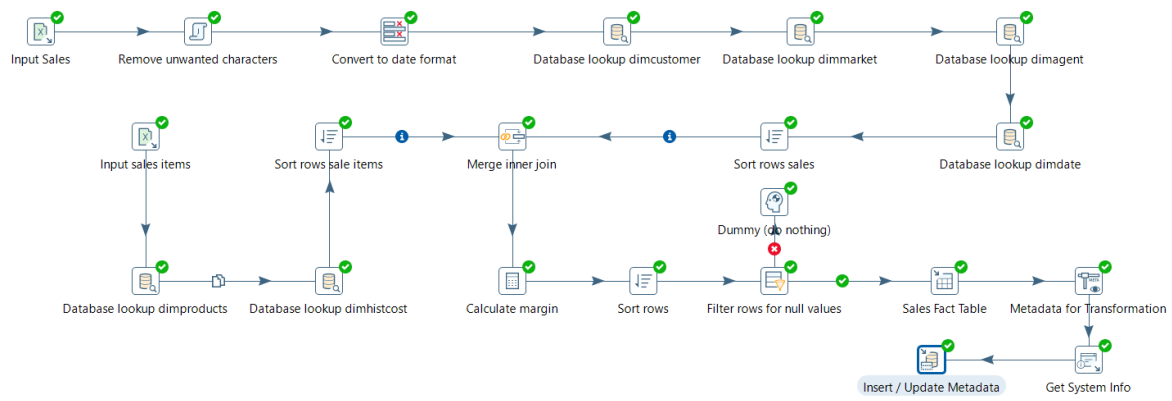
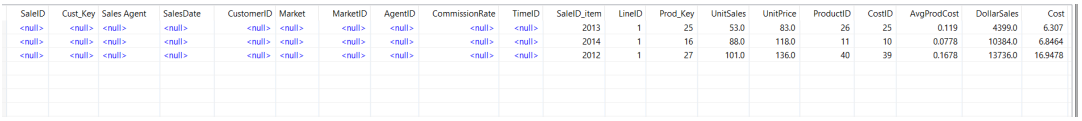


Figure 8: Transaction Sales Transformation

The Transaction Grain Sales transformation involves the following steps:

- Data is extracted from the Sales excel input file.
- Unwanted characters from the date filed is removed and data is cleansed.
- The date is converted into Date type where dates with incorrect format and values are handled.
- A database lookup is done with the DimCustomers table to extract the Customer details of the sales.
- A database lookup is done with the DimMarket table to extract the Market details of the sales.
- A database lookup is done with the DimAgents table to extract the Agent details of the sales.
- A database lookup is done with the DimDates table to extract the Time details of the sales.
- The data is sorted.
- Data is extracted from the SaleItems excel input file.
- A database lookup is done with the DimProducts table to extract the Product details of the sales.
- A database lookup is done with the DimHistCost table to extract the Cost details of the sales.
- The data is sorted.
- Both sorted data from Sales and SaleItems are merged (FULL OUTER JOIN) on SalesID.
- The CommissionAmount, GrossMargin and NetMargin is calculated.

- All the collected and computed data is sorted on SalesID.
- The data is filtered for null and missing values. The missing values are collected and the user can view the report from the transformation as shown in figure 9. It is planned for future releases to add this data in the metadata repository for audit purposes.
- The not null and correct data is loaded/inserted into the TxGrainSalesFact Table in the Data Warehouse.
- Metadata for the transformation is extracted.



SaleID	Cust_Key	Sales Agent	SalesDate	CustomerID	Market	MarketID	AgentID	CommissionRate	TimeID	SaleID_item	LineID	Prod_Key	UnitSales	UnitPrice	ProductID	CostID	AvgProdCost	DollarSales	Cost
<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	2013	1	25	53.0	83.0	26	25	0.119	4399.0	6.307
<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	2014	1	16	88.0	118.0	11	10	0.0778	10384.0	6.8464
<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	<null>	2012	1	27	101.0	136.0	40	39	0.1678	13736.0	16.9478

Figure 9: The table of rejected data from the filter for missing and null values.

3 Design of the Data Warehouse

3.1 Dimension Table Changes

- Naming convention was adjusted to CamelCase in all data warehouse.
- Set composite PK for dimensions with mini dimensions and fact tables.
- Country Dimension Table is removed from the Data Warehouse design.
- **Time Dimension Table:**
  Naming convention has been changed to meet CamelCase and the naming convention given by DimDates.
  Added new date, day, month, week fields for managing the data from the source files.
  Fixed relationship direction between date dimension and the fact tables.
- **Product Dimension Table:**
  Added product code to product dimension.
  Changed Slowly Changing Dimension (SCD) to type 2 in Price - Product Dimension as the source data does not contain date information about the price.
- **History Cost Dimension Table:**
  Added product code to the production history cost rapidly changing dimension.
- **Market Dimension Table:**
  Added MarketKey to Market table to identify the Markets from the source table.
  Added isActive an indicator implementing Slowly Changing Dimension (SCD) Type 1 for changing markets.
- **Agent Dimension Table:**
  Added AgentKey to Agent table to identify the Agents from the source table.
  Added version an indicator implementing Slowly Changing Dimension (SCD) Type 2 for the changing commission rates and Agents along with AgentStartDate and AgentFinishDate.
  The AgentName is split into two fields, the AgentFirstName and AgentLastName from the source table.
- **Customer Dimension Table:**
  Added CustKey to Customer table to identify the Customers from the source table.
  Added version an indicator implementing Slowly Changing Dimension (SCD) Type 2 for changing customer locations and customers along with ValidFromDate and ValidToDate.
- **City Dimension Table**
  The countryID is removed from the table since no data is available from source customer table.

3.2 Fact Table Changes

- Added transactions table as fact table
- Added GrossMargin and NetMargin computed from the Sales and Cost.

- Added LineID for the Transactions fact table and primary key is composed now of SalesID and LineID.

4 Appendix 1: Data Dictionary

4.1 Updates

The table 1 shows the changes implemented for the transaction sales table.

Table 1: Transaction Grain Sales Fact Table

Attribute	Description	Source
SalesID	Unique identifier for sales orders in the table	Saleid attribute in the Sales source file
LineID	identifier for the line in transaction	lineid obtained from SalesItem file
ProductID	Unique identifier for the product table	Surrogate key in product dimension table
MarketID	Unique identifier for the market table	Surrogate key in market dimension table
CustomerID	Unique identifier for the customer table	Surrogate key in customer dimension table
AgentID	Unique identifier for the agent table	Surrogate key in agent dimension table
TimeID	Unique identifier for the time table	Surrogate key in time dimension table
UnitSales	total quantity of products that are sold every transaction	Quantity attribute for each product sold for every transaction obtained from SaleItem file
UnitsCost	units price for product production every transaction	price attribute for each product sold for every transaction from SaleItem file
GrossMargin	Gross margin amount of sales of product that are sold every transaction	Margin amount calculated for the product sold every transaction. GrossMargin = DollarSales - Cost
NetMargin	Net margin of sales of product that are sold every transaction	Margin amount calculated for the products sold every transaction. NetMargin = GrossMargin - CommissionAmount
CommissionAmount	commission amount earned by agents after sales of product every Transaction	commission amount attribute calculated from sales price and commission rate for every agent's product sales every transaction. CommissionAmount = CommissionRate * DollarSales

Table 2: Data Dictionary of Transaction Sales Fact Table from the data warehouse

4.2 Metadata

ETL are used to load data from the operational systems to Overhill data warehouse in order to support decision making. It is necessary a mechanism to describe the data in order to increase visibility, understanding and usability through all levels in the organization. The data model presented section 3, the dictionary shown in this section, the source system file structure and data generated during ETL execution represents key elements of metadata:

- **Technical Metadata** Metadata needs to be available for query all the time, we have provided Metadata schema to capture operational metadata generated during the ETL process which is executed in jobs described in table 3 for the initial load into the data warehouse, in table 4 for scheduled incremental updates and in table **??** to schedule updates in transaction sales fact table according to defined Initial load and Incremental Strategies.
  As part of the procedure, foreign key checks must be disabled in the database connection **set foreign_key_checks = 0** and scripts for data warehouse setup and metadata repository setup are provided:
  - **Data Warehouse:** *dw_overhill.sql*
  - **Metadata Repository:** *metadata.sql*

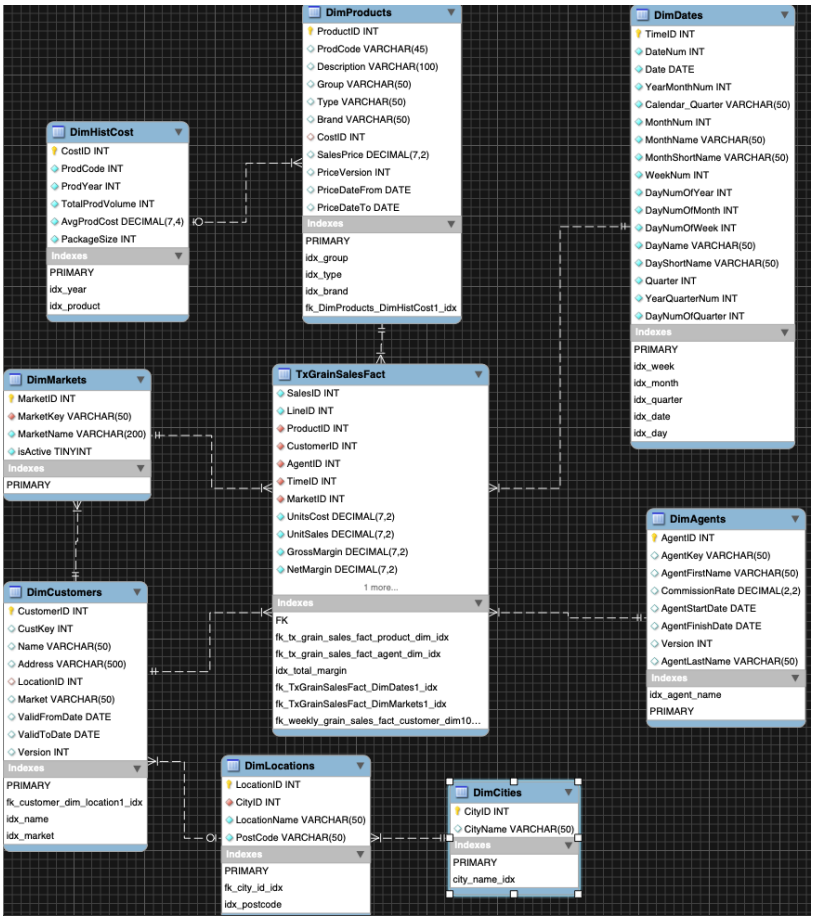


Figure 10: Overhill Data Warehouse Design

The figure 10 shows the model for the data warehouse that generates the sql file.

The initial load shown in figure 11 includes transformations for dimensions Markets, Dates, Agents, Customers, Locations and Products.

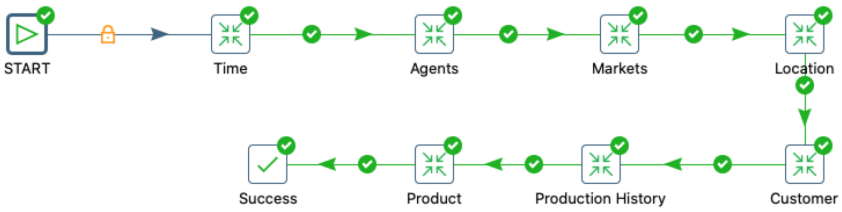


Figure 11: Initial Load Job Sequence of Transformations

Table 3: Initial Load Job Metadata

Job name	InitialLoadJob
Job Purpose	Execute the initial load corresponding to 2018 data
Source Files	DimDates.xlsx Product.xlsx (Production System) Product.xlsx (Sales System) ProductionHistory.xlsx Sales Agent.xlsx Customer.*2018.*.xlsx Market.xlsx
Target Tables	DimAgents DimCities DimCustomers DimDates DimHistCost DimLocations DimMarkets DimProducts TxGrainSalesFact
Rejected Data	Shown and logged in transformation
Pre Processes	None
Post Processes	IncrementalLoadJob
Frequency	No scheduling

On the other hand, the job shown in figure 12 includes transformations for updating Locations and Customers scheduled every week.

Finally, the Sales job shown in figure to update fact table is also updated every week using the Sales transformation.

Figure 12: Incremental Locations and Customers Job

Table 4: Incremental Load Job Metadata

Job name	IncrementalLoadJob
Job Purpose	Execute the incremental load corresponding to 2019 data
Source Files	Customer.*2019.*.xlsx
Target Tables	DimCities DimLocations DimCustomers
Rejected Data	Shown and logged in transformation
Pre Processes	InitialLoadJob
Post Processes	OverhillSalesJob
Frequency	Weekly

Figure 13: Sales Job

Table 5: Sales data load Job Metadata

Job name	OverhillSalesJob
Job Purpose	Execute the initial load for transactions
Source Files	Sales.xlsx SalesItem.xlsx
Target Tables	TxGrainSalesFact
Rejected Data	Shown and logged in transformation
Pre Processes	IncrementalLoadJob
Post Processes	None
Frequency	Weekly

- **Operational Metadata** Each transformation in a job generates metadata regarding the execution results. The figure 14 shows the schema where metadata is stored for each transformation execution.

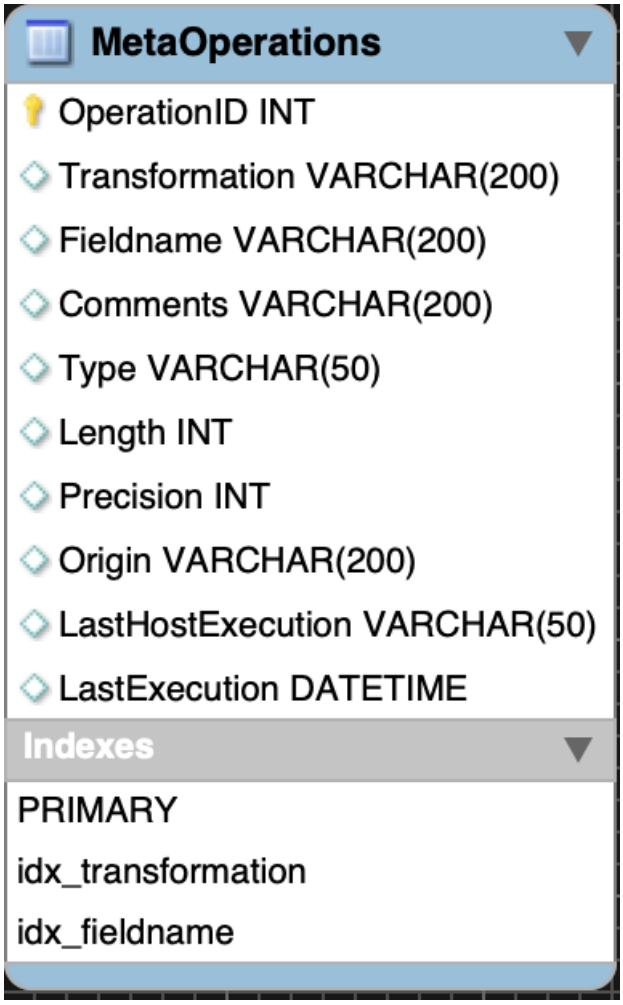


Figure 14: Operational Metadata schema

For example, the user can query the database using the SQL code 1 to pull data about the execution.

```
1    SELECT * FROM MetaOperations
2    WHERE Transformation LIKE '%sales%'
3
```

Listing 1: SQL code (MySQL) Operational Metadata for sales transformation

The table 6 shows the results for query 1 where users can identify in metadata: Fieldnames, Transformation logic for extracted data, host executing the job, Origin of data, data types and timestamp from execution.

- **Business Metadata** Examples of queries are shown in table 7. The user can use the dimensions to explain the data in fact table.

Table 6: Metadata operations

Transformation	Fieldname	Comments	Type	Length	Precision	Origin	LastHostExecution	LastExecution
transaction_sales_transformation	SaleID		Integer	-1	0	Input Sales	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	Cust_Key		Number	-1	-1	Input Sales	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	Sales Agent		String	-1	-1	Input Sales	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	SalesDate		Date	-1	-1	Convert to date format	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	CustomerID		Integer	-1	0	Database lookup dimcustomer	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	Market		String	-1	-1	Database lookup dimcustomer	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	MarketID		Integer	-1	0	Database lookup dimmarket	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	AgentID		Integer	-1	0	Database lookup dimagent	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	CommissionRate	CommissionRate	Number	2	2		192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	TimeID	TimeID	Integer	9	0		192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	SaleID_item		Integer	-1	0	Input sales items	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	LineID		Integer	-1	0	Input sales items	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	Prod_Key		Integer	-1	0	Input sales items	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	UnitSales		Number	-1	-1	Input sales items	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	UnitPrice		Number	-1	-1	Input sales items	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	ProductID		Integer	-1	0	Database lookup dimproducts	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	CostID		Integer	-1	0	Database lookup dimproducts	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	AvgProdCost		Number	-1	-1	Database lookup dimhistcost	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	DollarSales	MULTIPLY	Number	-1	-1	Calculate margin	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	Cost	MULTIPLY	Number	-1	-1	Calculate margin	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	GrossMargin	SUBTRACT	Number	-1	-1	Calculate margin	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	CommissionAmount	MULTIPLY	Number	-1	-1	Calculate margin	192.168.0.4	2020-02-06 22:47:20
transaction_sales_transformation	NetMargin	SUBTRACT	Number	-1	-1	Calculate margin	192.168.0.4	2020-02-06 22:47:20

Table 7: Query Examples

Description	SQL
Most profitable customers	*SELECT Name, floor(sum(UnitSales)) as TotalUnitSales,floor(sum(NetMargin)) as TotalProfitfrom* *FROM TxGrainSalesFact* *INNER JOIN DimCustomers DC on TxGrainSalesFact.CustomerID = DC.CustomerID* *GROUP BY Name* *ORDER BY TotalProfit DESC*
Most profitable markets	*SELECT DM.MarketKey, floor(sum(UnitSales)) as TotalUnitSales, floor(sum(NetMargin)) as TotalProfit* *FROM TxGrainSalesFact* *INNER JOIN DimMarkets DM on TxGrainSalesFact.MarketID = DM.MarketID* *GROUP BY DM.MarketKey* *ORDER BY TotalProfit DESC*

5 Appendix 3: Work breakdown

All the work for this project was a collaborative effort and all members produced or review the content and outputs, in particular, the work was broken down in tasks with specific owners to ensure completeness shown in table 8:

Table 8: Work breakdown

Task	Chaitra Mouli (977313)	Daniel Gil (905923)
Summary	80%	20%
Design analysis and changes	50%	50%
Time Transformation	40%	60%
Agent and Market Transformations	60%	40%
Location and Customer Transformations	40%	60%
Product and Production History Transformations	40%	60%
Transaction Sales Transformation	60%	40%
Data Dictionary Changes Documentation	80%	20%
Metadata	30%	70%

References

[1] R. Price and G. Shanks, *Data Quality and Decision Making*. Springer Berlin Heidelberg, pp. 65–82. [Online]. Available: http://link.springer.com/10.1007/978-3-540-48713-5_4