ANALYTICS FOR ADVENTUREWORKS

PREPARED FOR

EAE Data Management 2020

Francesc Casado

Bruno Garcia

PREPARED BY

Adrian Hagen

Jon Dale

Mohamed Ashmawy

Mostafa AbdElKhalek

Joseph Higaki

Table of Contents

Executive Summary

Adventure Works Cycles is a multinational bicycle manufacturer that records data from its operations in a database that is optimized for Online Transaction Processing (OLTP). The company transactions recorded are from the following business processes: Manufacturing, Sales, Purchasing, Product Management, Contact Management, and Human Resources. Some data discovery shows that the company has around 97 brands of different bikes, grouped into mountain bikes, road bikes, and touring bikes. They manufacture some components themselves, but also buy components, accessories, and clothing from outside vendors. They sell to both retail stores, which we have defined as resellers and individual customers.

The company, in order to improve business performance needs to obtain insights from its operations. Getting insights from an OLTP database has been proven challenging in the past, therefore Adventure Works Cycles will start to build datamarts that comply with an Online Analytical Processing (OLAP) database approach, so that Analytical solutions are built based on those datamarts. This project focuses on analyzing sales and purchasing, where certain business challenges are being dealt with. These challenges are presented in section 2 below.

1. Project Overview

Adventure Works Cycles will create Sales and Purchasing datamarts as they have been prioritized by the company's management to be the areas where any improvement from the analytics insights will have a large impact on financial key performance indicators.

2. Business Challenges

The company has business challenges and needs an analytics solution to provide insights on:

1. Analyze reseller sales vs direct consumer sales over specific holidays.
2. Create top 5 products sales ranking by seasons.
3. Create top 5 products profit ranking, quarter by quarter.
4. Create a top 5 Vendor ranking of rejected items on purchase orders.
5. See in a single view sold items vs purchased items, for each product category

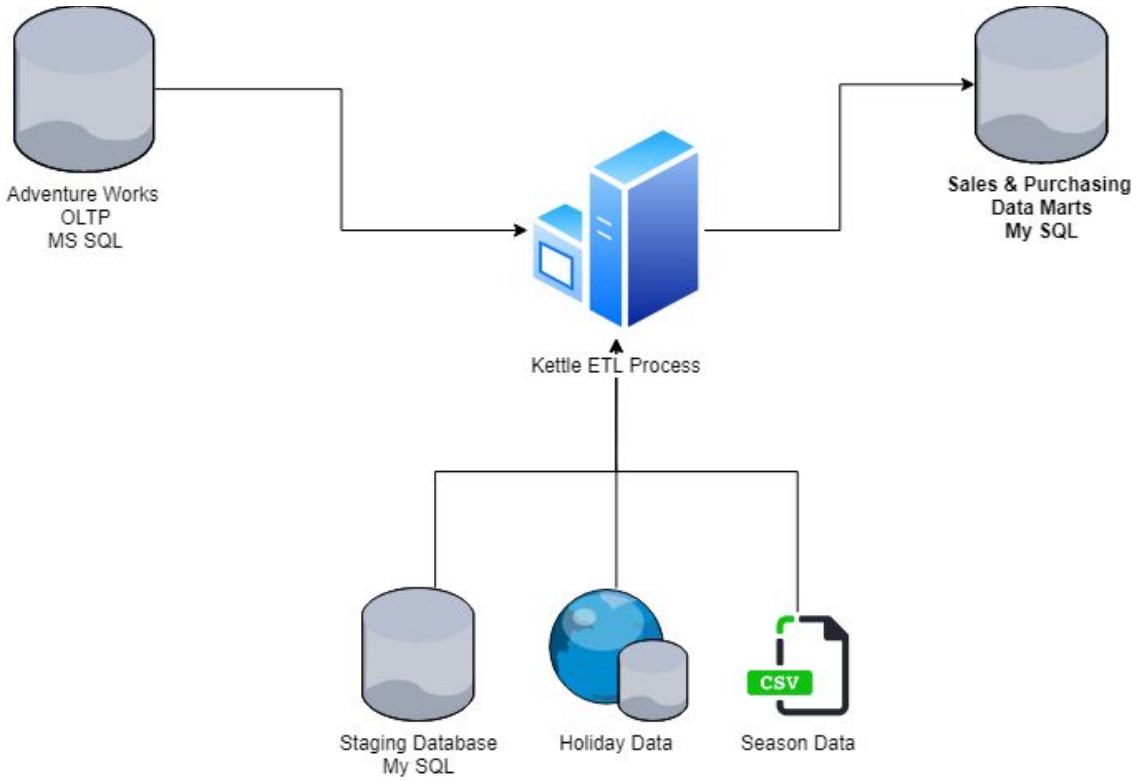
3. Technical Architecture

Kettle will be used to extract, transform and load the data into the data marts.

Main source of information is the Adventure Works Transactional database.

Additional data sources:

1. Staging database, used for calculations, with self generated data.
2. Open data for getting country holidays was used from https://date.nager.at/
3. A local csv for calculating seasons across dates

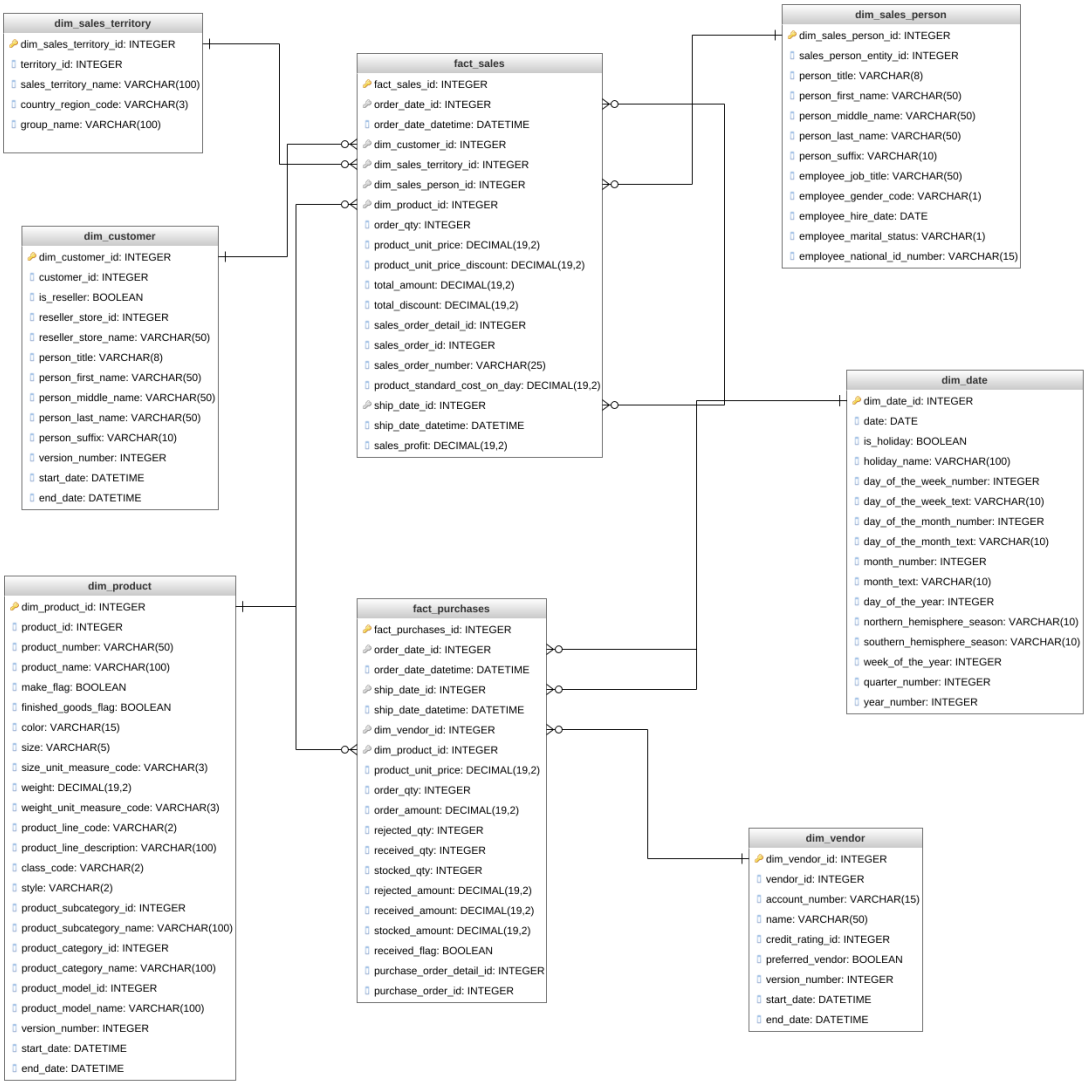


4. Business Entities for Analytical Model

From a theoretical standpoint our analytical model follows Kimball's definition of a data warehouse. It is the sum of the datamarts sales and purchasing, and as a result the data is organized by these two subjects for now. The Kimball approach allows for a more simplistic and flexible model, which can be extended if new business cases appear. Additionally, it is an integrated or unified view of the data from the different sources, where naming conventions and descriptions are consistent. Our generated date dimension allows for an historical view of the facts. Best practices based on the Kimball Rules are implemented such as ensuring every fact table has an associated date dimension table, the creation of surrogate keys, and for the dimensions we find it necessary we apply a slowly changing dimension type 2 structure.

Our fact tables sales and purchasing contain relevant quantitative data such as prices, quantity, sales, rejected- and received amounts. The dimensions contain descriptive data on sales territories, customers, products, sales persons, and date.

As the diagram below shows our model follows a star schema, resulting in a model that has fast query time and is simple to use for the end user.



In the following sections are detailed each of the tables from the Sales & Purchasing Data Marts.

5. Product

Product Dimension

Table Name	dim_product

The product dimension has been de-normalized to contain Product Line, Product Category, Subcategory and Product Model. The purpose is to be able to filter the facts on products, subcategory, or models. This dimension combined with other dimensions and facts will allow us to answer business challenges 2, 3 and 5: Create top 5 products sales ranking by seasons; create top 5 products profit ranking, quarter by quarter; see in a single view sold items vs purchased items for each product category.

Product Dimension is a slowly changing dimension type 2, where its history is kept by versioning each row and assigning an effective date range by the ETL process.

Product Data Dictionary

Column Name	Key?	Type	Description	Source
dim_product_id	Yes	INTEGER	This is the Surrogate key for the Product Dimension. This is particularly important because it is a SCD Type 2	Generated in the mySql DB using the add sequence feature in Kettle
product_id	No	INTEGER	This is the Product Primary Key of the Transactional database	Extracted from AdventureWorks2019 table Production.Product column ProductID
product_number	No	VARCHAR	Product number	Extracted from AdventureWorks2019 table Production.Product column ProductNumber
product_name	No	VARCHAR	Name of the product	Extracted from AdventureWorks2019 table Production.Product column Name
make_flag	No	BOOLEAN	0 = Product is purchased, 1 = Product is manufactured in-house. Default: 1	Extracted from AdventureWorks2019 table Production.Product column MakeFlag

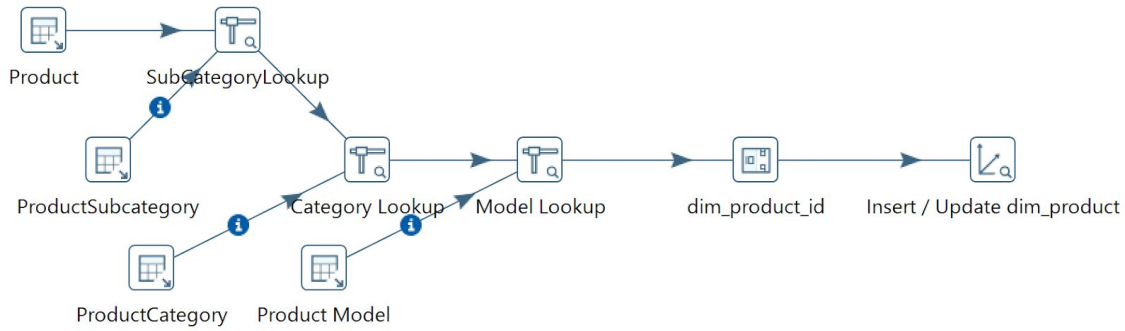
finished_goods_flag	No	BOOLEAN	0 = Product is not a salable item. 1 = Product is salable. Default: 1	Extracted from AdventureWorks2019 table Production.Product column FinishedGoodsFlag
color	No	VARCHAR	Color of product	Extracted from AdventureWorks2019 table Production.Product column Color
size	No	VARCHAR	Size of product	Extracted from AdventureWorks2019 table Production.Product column Size
size_unit_measure_code	No	VARCHAR	Size unit measure code of the product	Extracted from AdventureWorks2019 table Production.Product column SizeUnitMeasureCode
size_unit_measure_name	No	VARCHAR	Size unit measure name	Extracted from AdventureWorks2019 table Production.UnitMeasure column Name
weight	No	VARCHAR	Weight of the product	Extracted from AdventureWorks2019 table Production.Product column Weight
weight_unit_measure_code	No	VARCHAR	Weight unit measure code	Extracted from AdventureWorks2019 table Production.Product column WeightUnitMeasureCode
weight_unit_measure_name	No	VARCHAR	Weight unit measure name	Extracted from AdventureWorks2019 table Production.Product column WeightUnitMeasureCode
product_line_code	No	VARCHAR	Product line code, R = Road, M = Mountain, T = Touring, S = Standard	Extracted from AdventureWorks2019 table Production.Product column ProductLine
product_line_description	No	VARCHAR	Description of product line	Extracted from AdventureWorks2019 table Production.Product column ProductLine
class_code	No	VARCHAR	H = High, M = Medium, L = Low	Extracted from AdventureWorks2019 table

				Production.Product column Class
class_description	No	VARCHAR	Class description	Extracted from AdventureWorks2019 table
style	No	VARCHAR	W = Womens, M = Mens, U = Universal	Extracted from AdventureWorks2019 table
style_description	No	VARCHAR	Style description	Extracted from AdventureWorks2019 table
produt_subcategory_id	No	INTEGER	Subcategory ID, each product is divided in subcategories	Extracted from AdventureWorks2019 table Production.ProductSubcategory column ProductSubcategoryID
product_subcategory_name	No	VARCHAR	Name of the subcategory	Extracted from AdventureWorks2019 table Production.ProductSubcategory column Name
product_category_id	No	INTEGER	Product category ID	Extracted from AdventureWorks2019 table Production.ProductCategory column ProductCategoryID
product_category_name	No	VARCHAR	Name of the product cateogry	Extracted from AdventureWorks2019 table Production.ProductCategory column Name
product_model_id	No	INTEGER	Product model id	Extracted from AdventureWorks2019 table Production.ProductModel column ProductModelID
product_model_name	No	VARCHAR	Name of the product model	Extracted from AdventureWorks2019 table Production.ProductModel column Name
start_date	No	DATETIME	Start effective date for the product version	Calculated in Kettle, based on Adventureworks2019 changes
end_date	No	DATETIME	End effective date for the product version	Calculated in Kettle, based on Adventureworks2019 changes
version_number	No	INTEGER	Version number representing the sequence where a product has changed in history, after the	Calculated in Kettle, based on Adventureworks2019 changes

			dimension first load	

Product Transformation

Kettle transformation	dim_product.ktr



The Product transformation de-normalizes Product Subcategory Category and Model by doing lookups.

We are using a Dimension lookup/update component in Kettle to implement the Slowly Changing Dimension type 2 behaviour. This allows changes in the source system to be traced, where a new dimension row will be added for the changed record. This allows us to have the full history of values. When a value of an attribute changes, the current record will be closed. The new record with the changed data will become the current record. Each record will have a start- and end date allowing the user to identify the time periods the records are active or closed.

6. Customer

Customer Dimension

Table Name	dim_customer

The customer dimension has been de-normalized to contain information about individual customers and resellers in one table. The purpose of the table is to be able to filter facts such as sales on individual customers and resellers, or categorize sales to one of these two groups. This dimension allows us to answer business challenge 1: Analyze reseller sales vs direct consumer sales over specific holidays.

Customer Dimension is a slowly changing dimension type 2. Its history is kept by versioning each row and assigning an effective date range by the ETL process.

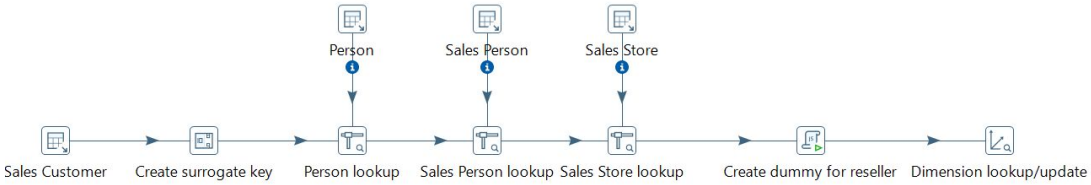
Customer Data Dictionary

Column Name	Key?	Type	Description	Source
dim_customer_id	Yes	INTEGER	Surrogate key	Generated in a staging database using autoincrement
customer_id	No	INTEGER	This is the customer primary key in the transactional database	Extracted from AdventureWorks2019 table Sales.Customer
is_reseller	No	BOOLEAN	Flag indicating if it is a reseller or individual. 1= reseller and 0 = no reseller (individual)	Created in Kettle by using a Modified JavaScript value, where the flag defines a reseller where store_id is not null and an individual as a customer where store_id is null.
reseller_store_id	No	INTEGER	ID for resellers	Extracted from AdventureWorks2019 table Sales.Store and column BusinessEntityID.
reseller_store	No	VARC	Name of reseller	Extracted from AdventureWorks2019

_name		HAR		table Sales.Store and column Name
person_title	No	VARCHAR	Title of person	Extracted from AdventureWorks2019 table Person.Person and column Title
person_first_name	No	VARCHAR	First name of person	Extracted from AdventureWorks2019 table Person.Person and column FirstName
person_middle_name	No	VARCHAR	Middle name of person	Extracted from AdventureWorks2019 table Person.Person and column MiddleName
person_last_name	No	VARCHAR	Last name of person	Extracted from AdventureWorks2019 table Person.Person and column LastName
person_suffix	No	VARCHAR	Suffix of person	Extracted from AdventureWorks2019 table Person.Person and column Suffix
version_number	No	INTEGER	Version number of the record	Generated if we need to insert a new record into the database
start_date	No	DATETIME	Start date of validity of record	Generated if we need to insert a new record into the database
end_date	No	DATETIME	End date of validity of record	Generated if we need to insert a new record into the database

Customer Transformation

Kettle transformation	dim_customer.ktr



Information about customers and resellers are acquired from the person.person table and sales.store table in the transactional database. To get the name of the reseller the business entityid was needed and was retrieved from the salesperson table in the transactional database by using territoryid. To be able to filter between reseller and individual customers a dummy for resellers was created using a java script. This script creates a dummy value equal to one if the resellerid is not null, and zero if it is null. A add sequence function was added to create the surrogate key.

We used a Dimension lookup/update component in Kettle to implement the Slowly Changing Dimension type 2 behaviour.

7. Sales Person

Sales Person Dimension

Table Name	dim_sales_person

The sales dimension has been de-normalized to contain all the info about the sales employees that can be used to identify the employees role and identity. This dimension does not resolve one of the presented business challenges, but it is likely to be useful in the future. For example, it allows the end-users to see which sales persons are the top performing measured by sales.

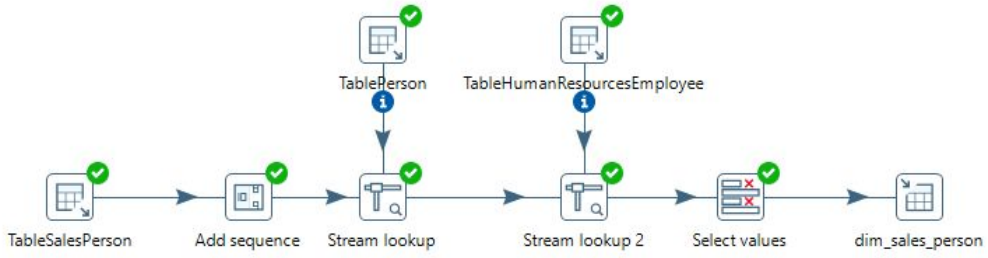
Sales Person Data Dictionary

Column Name	Key?	Type	Description	Source
dim_sales_person_id	Yes	Integer	Surrogate key	Generated in the mySql DB using the add sequence feature in kettle
sales_person_entity_id	No	Integer	A unique ID that identify the Sales employees	AdventureWorks SalesPerson table
person_title	No	String	Sales Employee Title	AdventureWorks Person table Joined with SalesPerson table using BusinessEntityID
person_first_name	No	String	Sales Employee FirstName	AdventureWorks Person table Joined with SalesPerson table using BusinessEntityID
person_middle_name	No	String	Sales Employee MiddleName	AdventureWorks Person table Joined with SalesPerson table using BusinessEntityID
person_last_name	No	String	Sales Employee LastName	AdventureWorks Person table Joined with SalesPerson table using BusinessEntityID
person_suffix	No	String	Sales Employee Suffix	AdventureWorks Person table Joined with SalesPerson table using BusinessEntityID
employee_job_title	No	String	Sales Employee JobTitle	AdventureWorks Person table Joined with SalesPerson table using BusinessEntityID
employee_gender_code	No	String	Sales Employee GenderCode	AdventureWorks Person table Joined with Employee table using BusinessEntityID
employee_hire_date	No	Date	Sales Employee HireDate	AdventureWorks Person table Joined with Employee table using BusinessEntityID

employee_marital_status	No	Char	Sales Employee MartialStatus	AdventureWorks Person table Joined with Employee table using BusinessEntityID
employee_national_id_number	No	Integer	Sales Employee NationalIDNumber	AdventureWorks Person table Joined with Employee table using BusinessEntityID

Sales Person Transformation

Kettle transformation	dim_sales_person.ktr



The SalesPerson transformation de-normalizes table person and table HrEmployee by doing lookups.

An add sequence function was added to create the surrogate key and the select values function to map the data to the right columns in the MySql database.

8. Sales Territory

Sales Territory Dimension

Table Name	dim_sales_territory

This dimension is responsible for collecting and representing the data about the sales territory, starting with the territory's geographic location and unique ID to the territory's name. This dimension is not related to a specific business challenge either, but could also be useful in the future. It allows the end users to answer questions such as which sales territory have the highest sales or quantity sold.

Sales Territory Data Dictionary

Column Name	Key?	Type	Description	Source
Sales_Territory_ID	Yes	Integer	Surrogate Key	Generated in the mySql DB using the add sequence feature in Kettle
territory_ID	No	Integer	Unique ID for every sales territory	Adventureworks sales territory table
Sales_Territory_Name	No	Varchar	Name of each specific sales territory	Adventureworks sales territory table
Country_Region_Code	No	Varchar	An abbreviation code to the region where the territory is	Adventureworks sales territory table
Country_Region_Name	No	Varchar	Name of the specific region where the territory is	Adventureworks sales territory table
Group_Name	No	Varchar	Geographic location where multiple country regions are	Adventureworks sales territory table

Sales Territory Transformation

Kettle transformation	dim_sales_territory.ktr

We used the “add sequence” to have an automated adding to sequence of the unique ID. Then we filtered the data we need with the select values to get the desired result “clean data”.

9. Vendor

Vendor Dimension

Table Name	dim_vendor

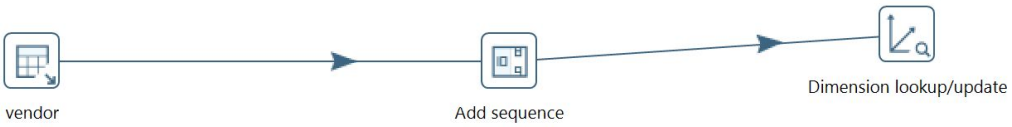
This dimension contains data about the vendors, and it's history is tracked using slow changing dimension type 2. This dimension allows us to answer business case 4, where we wish to know the top 5 vendors in terms of rejected items on purchased orders.

Vendor Data Dictionary

Column Name	Key?	Type	Description	Source
dim_vendor_id	Yes	Integer	Surrogate key	Generated by "add sequence" in Kettle if we need to insert a new record into the database
vendor_id	No	Integer	ID of vendor	Extracted from AdventureWorks vendor table using BusinessEntityID
account_number	No	Varchar	Vendor's account number	Extracted from AdventureWorks vendor table using AccountNumber
name	No	Varchar	Name of company	Extracted from AdventureWorks vendor table using Name
credit_rating_id	No	Integer	Credit rating: 1 = superior, 2 = excellent, 3 = above average, 4 = average, 5 = below average	Extracted from AdventureWorks vendor table using CreditRating
preferred_vendor	No	Boolean	This vendor is preferred over other vendors	AdventureWorks Vendor
version_number	No	Integer	Version number of the record	Generated if we need to insert a new record into the database
start_date	No	Datetime	Start date of validity of record	Generated if we need to insert a new record into the database
end_date	No	Datetime	End date of validity of record	Generated if we need to insert a new record into the database

Vendor Transformation

Kettle transformation	dim_vendor.ktr



This transformation features slow changing dimension type 2, where we have a start date and end date of the validity of the records, and a version number that tells us what version of the record it is.

The slow changing dimension is handled by the dimension lookup/update step in Kettle. This uses the surrogate key «dim_vendor_id» which is the primary key of the dimension. When a new version of the record is being inserted, the version number is incremented by 1, and the start date of the previous record is set to the start date of the new record to be inserted. This way, the old record has been invalidated, because it is not the latest version.

10. Date

Date Dimension

Table Name	dim_date

In the date dimension each record represents a single day so that analytics can be run as effectively as possible. The date dimension will hold information to whether it is a Spanish holiday or not, and which holiday name is.

Additionally, the date dimension contains useful values such as the sequential day of the year, the week number, quarter and northern and southern season.

The date dimension is relevant to answer business challenges 1-3, and in general is very important to allow an historical overview of the quantitative data in our fact tables.

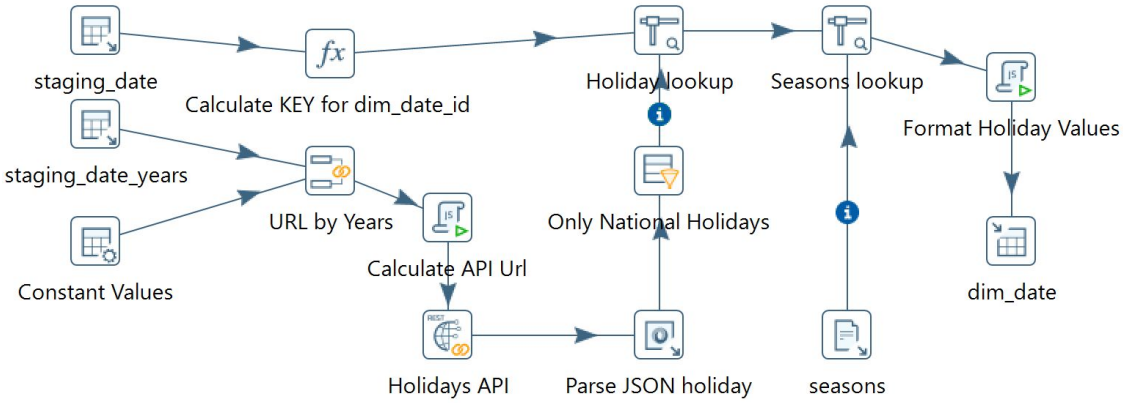
Date Data Dictionary

Column Name	Key?	Type	Description	Source
dim_date_id	Yes	Integer	This is the surrogate key It is an integer with the following format: 20201214	Generated in a staging database that lists days in a datarange YEAR * 10,000 + MONTH * 100 DAY
is_holiday	No	Integer	Flag indicating if it is a holiday or not	Holiday data is exrtracted via an from https://date.nager.at/api/v2/publi days/2020/es
holiday_name	No	VARCH/	National HOliday name Spain	Holiday data is exrtracted via an from https://date.nager.at/api/v2/publi days/2020/es

date	No	Date	Represents the day of th row, in a date database	Generated in a staging database that lists days in a datarange
day_of_the_week_n ber	No	integer	Week number (00 to 53)	Generated in a staging database that lists days in a datarange
day_of_the_week_t	No	VARCHA	Week english text. e.g. Monday, Tuesday	Generated in a staging database that lists days in a datarange
day_of_the_month_ ber	No	integer	Day of the month as a numeric value (01 to 31)	Generated in a staging database that lists days in a datarange
day_of_the_month_	No	VARCHA	Day of the month text. e 1st, 2nd	Generated in a staging database that lists days in a datarange
month_number	No	integer	Day of the month numbe to 31)	Generated in a staging database that lists days in a datarange
month_text	No	VARCHA	Month english text. e.g. January, February	Generated in a staging database that lists days in a datarange
day_of_the_year	No	integer	Day of the year (1 to 366	Generated in a staging database that lists days in a datarange
northern_hemisphe eason	No	VARCHA	Season text in english fo northern hemisphere tex (Winter, Autum, Summe Fall)	Calculated from a csv source
southern_hemisphe eason	No	VARCHA	Season text in english fo southern hemisphere tex (Winter, Autum, Summe Fall)	Calculated from a csv source
week_of_the_year	No	integer	Week where Monday is first day of the week (01 53)	Generated in a staging database that lists days in a datarange
quarter_number	No	integer	Quarter as a numeric va (1 - 4)	Generated in a staging database that lists days in a datarange
year_number	No	integer	Year as a numeric, 4-dig value	Generated in a staging database that lists days in a datarange

Date Transformation

Kettle transformation	dim_date.ktr



The date dimension uses an initial list of dates, that describe the minimum and maximum values the data mart transactions can have.

This is an initial setup and it's done in a staging database in MySQL

```
11
12 •    CALL fill_staging_date_table('1970-01-01','2050-12-31');
13
```

The staging database will contain sequential dates, with attribute values from calculations that are easily achieved in a database engine. These are:

- Day of the week number
- Day of the week text
- Day of the month number
- Day of the month text
- Month number
- Month text
- Day of the year
- Week of the year

- Quarter

Additional source: Holiday

Open data is used to retrieve holidays.

Nager Date is a public holiday database that has REST API endpoints to retrieve holidays by country and year.

```
GET /PublicHolidays/{Year}/{CountryCode}
```

In the kettle transformation, we extract holidays the following way:

- Within each year of the date dimension initial span.
- Call the Nager Date API for the year and Spain holidays
- Parse the JSON API response to obtain date and holiday name
- Filter only the nation-wide holidays
- Use them as lookup for processing each record (day), in the date dimension

Additional source: Seasons

The assumption is that seasons' date span does not change.

To demonstrate a different kind of data source, the seasons data has been set on a *.csv and is consumed by the transformation

northern_hemisphere	southern_hemisphere	month
Winter	Summer	12
Winter	Summer	1
Winter	Summer	2
Spring	Autumn	3
Spring	Autumn	4
Spring	Autumn	5
Summer	Winter	6
Summer	Winter	7
Summer	Winter	8
Autumn	Spring	9
Autumn	Spring	10
Autumn	Spring	11

11. Sales

Sales Fact

Table Name	fact_sales

The fact table sales stores sales measures at the Sales Order Detail level. Its main measures are the price, quantity and total amount for each product sold within Sales Orders. Sales Fact also stores the product standard cost by the time the order was issued. This table is important to make us able to deal with business challenges 1-3 and 5.

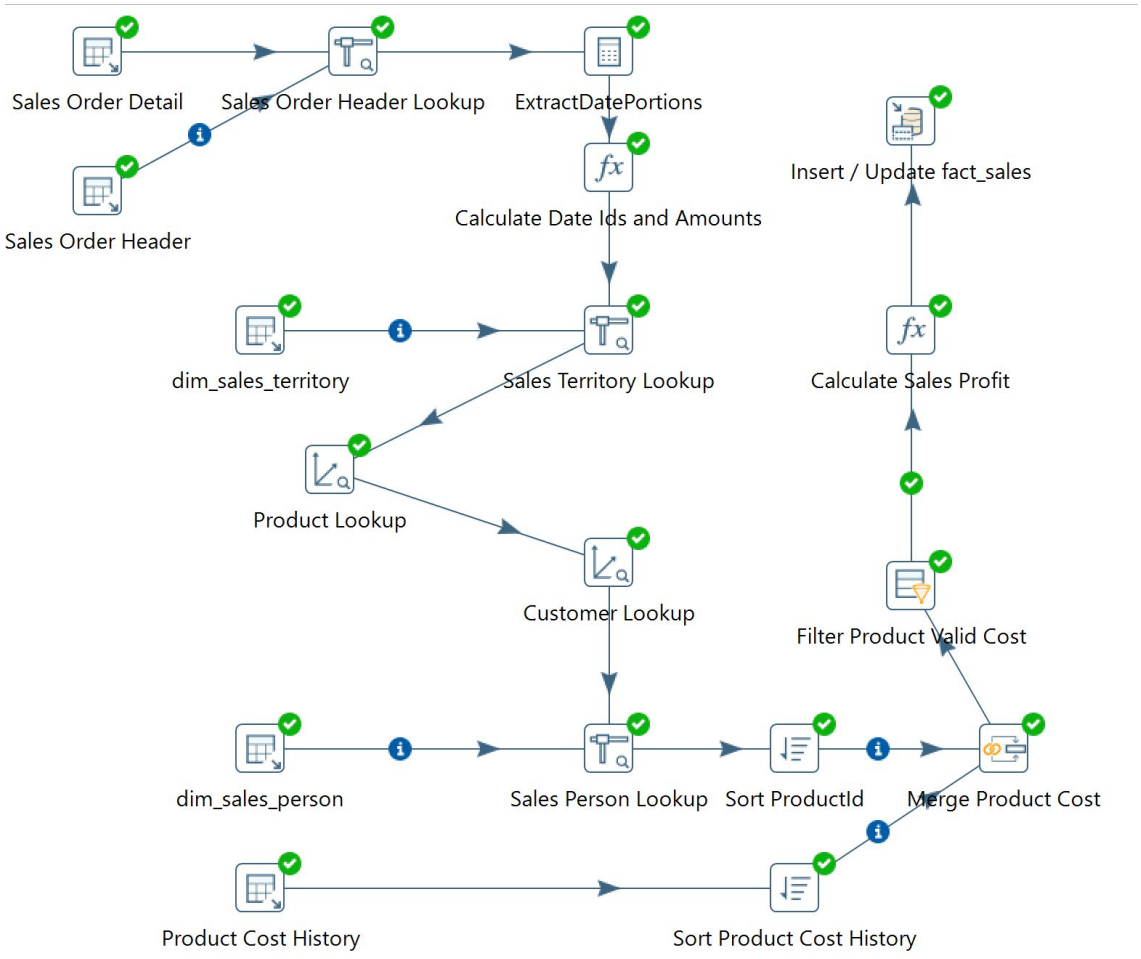
Sales Data Dictionary

Column Name	Key?	Type	Description	Source
fact_sales_id	Yes	Integer	Surrogate key	Generated by «add sequence» in Kettle.
order_date_id	No	Integer	ID of order date	Calculated by Kettle.
order_date_datetime	No	Datetime	The date of the order	Extracted from AdventureWorks SalesOrderHeader.
dim_customer_id	No	Integer	ID of the customer	Customer dimension.
dim_sales_territory_id	No	Integer	ID of the territory the customer is in	Sales territory dimension.
dim_sales_person_id	No	Integer	ID of the sales person	Sales person dimension.
dim_product_id	No	Integer	ID of the product	Product dimension.
order_qty	No	Integer	The order quantity	Extracted from AdventureWorks SalesOrderDetail.
product_unit_price	No	Decimal	The price per unit	Extracted from AdventureWorks SalesOrderDetail.
product_unit_price_discount	No	Decimal	Amount of discount of price per unit	Extracted from AdventureWorks SalesOrderDetail.
total_amount	No	Decimal	The total order amount	Extracted from AdventureWorks SalesOrderHeader.
total_discount	No	Decimal	The total discount amount	Extracted from AdventureWorks SalesOrderHeader.
sales_order_detail_id	No	Integer	Unique integer for each item sold	Extracted from AdventureWorks SalesOrderDetail.

sales_order_id	No	Integer	ID of the order	Extracted from AdventureWorks SalesOrderDetail.
sales_order_number	No	Varchar	The order number associated with an order	Extracted from AdventureWorks SalesOrderHeader.
product_standard_cost_on_day	No	Decimal	The standard cost of the product that day	Calculated by Kettle.
ship_date_id	No	Integer	ID of the order ship date	Extracted from AdventureWorks SalesOrderHeader.
ship_date_datetime	No	Datetime	The date when the order was shipped	Extracted from AdventureWorks SalesOrderHeader.
sales_profit	No	Integer	Calculated by subtracting Product Unit Price - Standard Product Cost, times the order quantity.	Product unit price and quantity extracted from AventureWorks SaleOrderheader. Calculated standard product cost from DW

Sales Transformation

Kettle transformation	fact_sales.ktr



Sales Fact table is based on Sales Order Detail. It lookups information from Sales Order Header. At this stage, order and ship dates are calculated to match Date Dimensions IDs.

This is an example of how the surrogate date dimension key is calculated, based on Order Date:

[order_date_year]*10000 + [order_date_month]*100 + [order_date_day]

Product Standard Cost is calculated by looking into the Product Cost History table and finding the correct cost date range, based on the Order Date.

Finally, Sales Profit is calculated by subtracting Product Unit Price - Standard Product Cost, times the order quantity.

12. Purchases

Purchases Fact

Table Name	fact_purchases

The purchase Fact contains all the info about the trading process of the product, how many was rejected, received, and stocked and what is the price of them. A boolean value exists to indicate if the product is delivered or not.

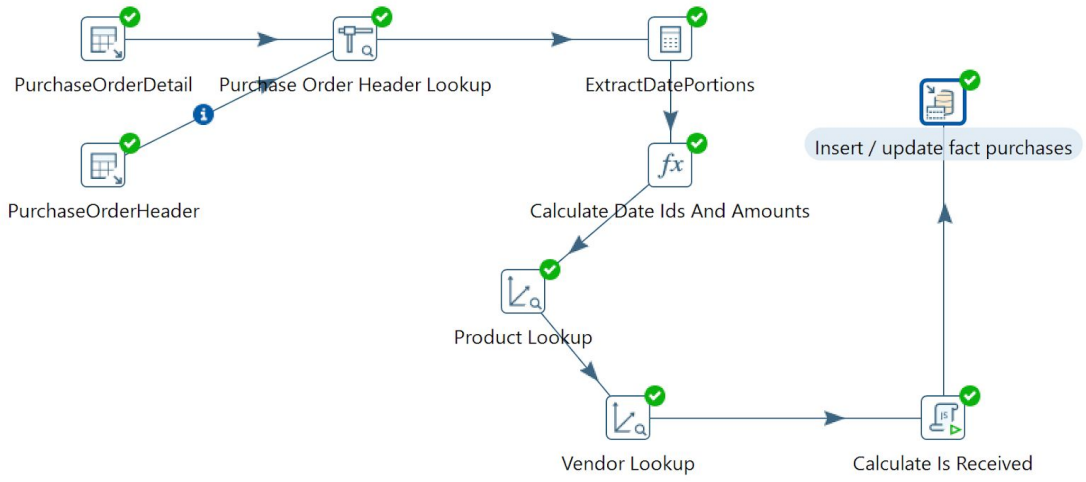
PurchasesData Dictionary

Column Name	Key?	Type	Description	Source
fact_purchases_id	Yes	Integer	Surrogate key	Generated in the mySql DB using the add sequence feature in kettle
dim_product_id	No	Integer	ID of the product	Foreign key from table dim_product MySql database
product_unit_price	No	Integer	Unit price of the product	AdventureWorks PurchaseOrderDetail
rejected_qty	No	Integer	Quantity of rejected items	AdventureWorks PurchaseOrderDetail
received_qty	No	Integer	Quantity of items received from the vendor	AdventureWorks PurchaseOrderDetail
stocked_qty	No	Integer	Quantity of accepted items into inventory	AdventureWorks PurchaseOrderDetail
product_standard_cost_on_day	No	Integer	The standard cost of the item that day	ated column matching the Standard Cost date i stHistory, that contains the Sales.SalesOrderHe
rejected_amount	No	Integer	Value of rejected items	Calculated column from [RejectedQty] * [UnitPrice]
received_amount	No	Integer	Value of received items	Calculated column from [ReceivedQty] * [UnitPrice]
stocked_amount	No	Integer	Value of items in inventory	Calculated column from [StockedQty] * [UnitPrice]
received_flag	No	Boolean	Flag indicating if the order is received or not,	Created column in Kettle by using a Modified JavaScript value

			1 = received and 0 = not received	
due_date_id	No	Integer	Date the order is due	Foriegn key for table dim_date converted using kettle to match the integer value in the column of the dim_date table
dim_vendor_id	No	Integer	ID of the vendor	Foreign key from table dim_vendor MySql database

Purchases Transformation

Kettle transformation	fact_purchases.ktr



The FactPurchase transformation de-normalizes tables orderDetail, Orderheader, dim_Product, and dim_vendor by doing lookups.

The calculator function was used to differentiate between year, month,and day for the orderDate and the shipDate.

The javaScript code was used to check if the order was received or not.

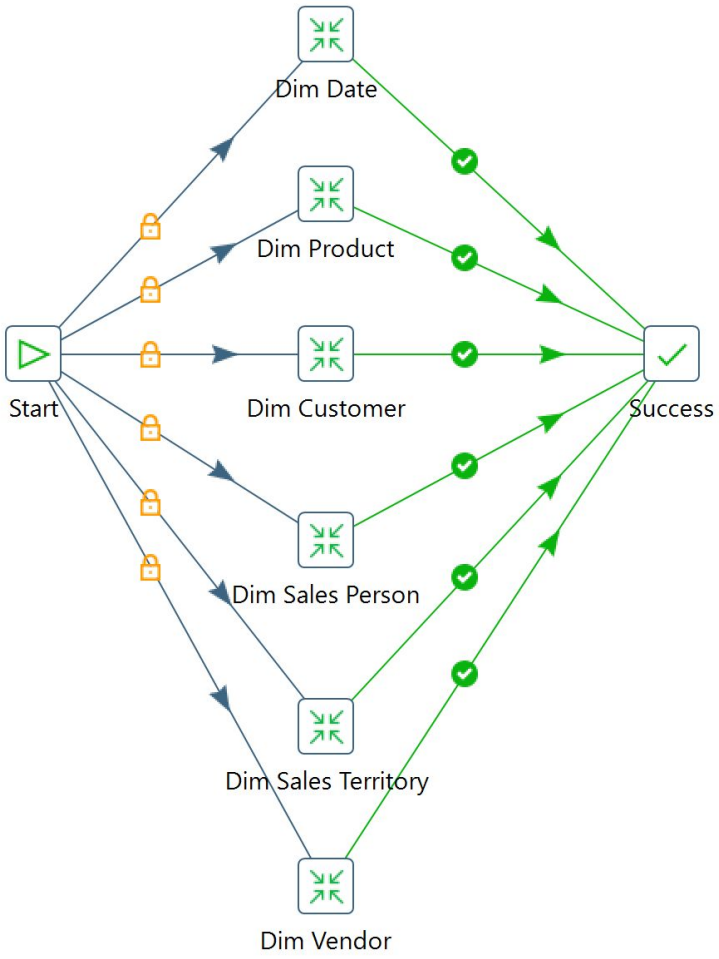
13. Job Scheduling

The ETL process has been splitted in two sections

Dimensions Job

Kettle job	DimensionsJob.kjb

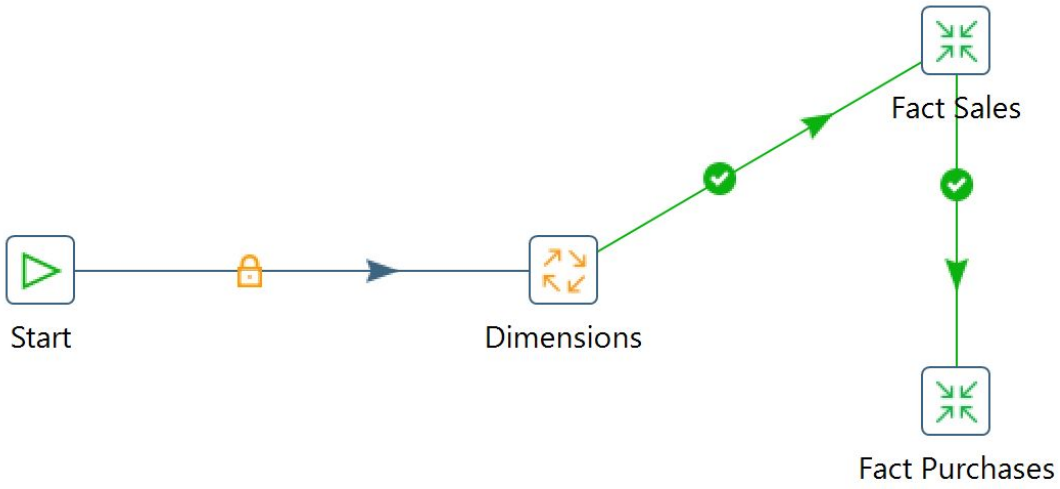
This Job launches the transformations for the dimension tables.



Facts Job

Kettle job	FactsJob.kjb

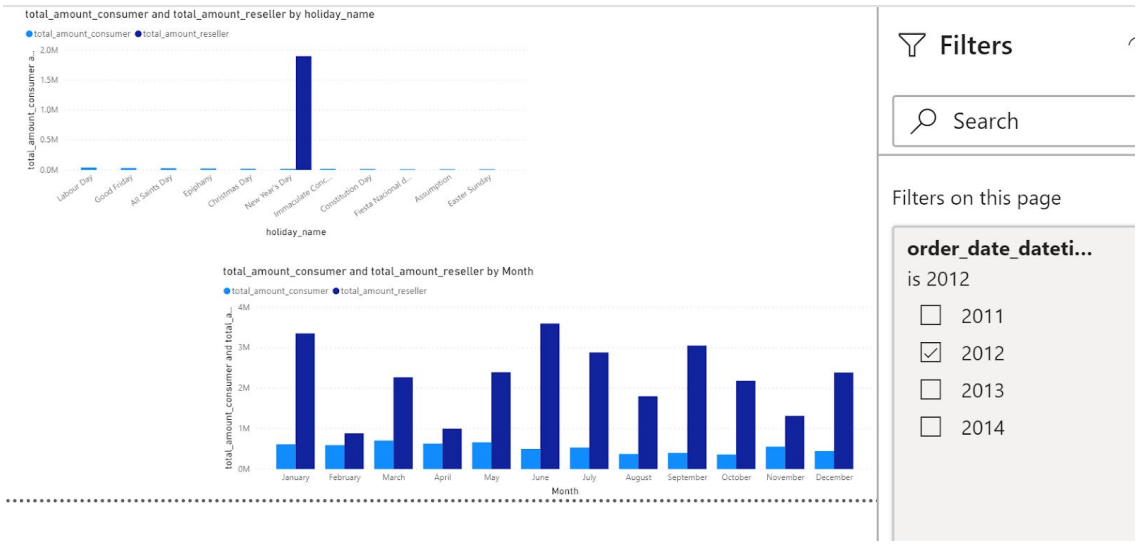
This Job launches the transformations for the fact tables. It has a dependency to the Dimensions Job, so that fact tables are always built with up-to-date dimension records



14. Conclusions

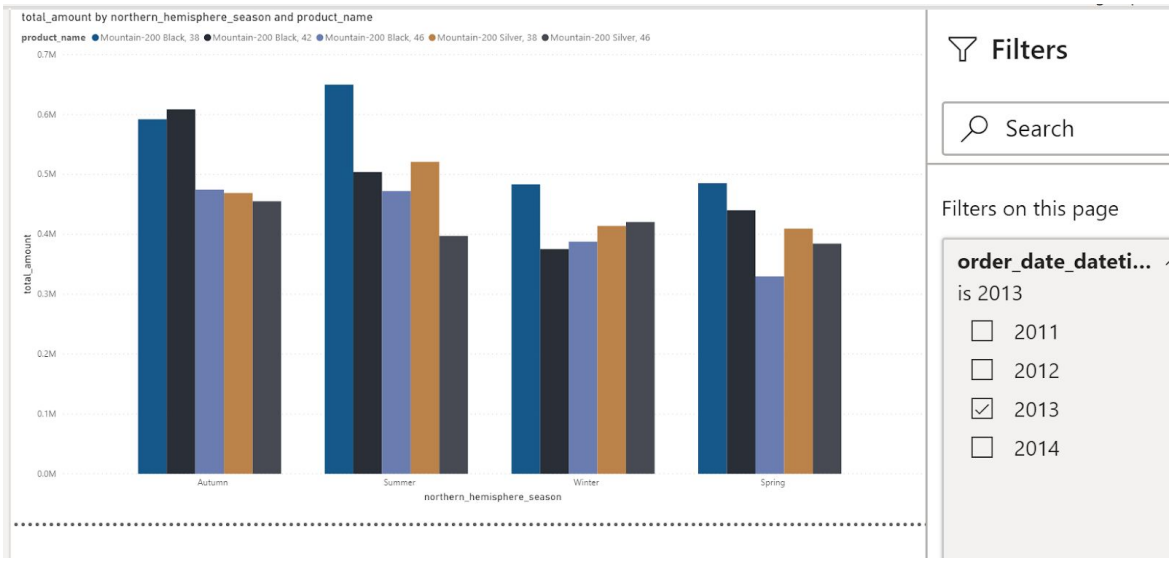
In this section we look at the answer to the business challenges.

1. Analyze reseller sales vs direct consumer sales over specific holiday days.



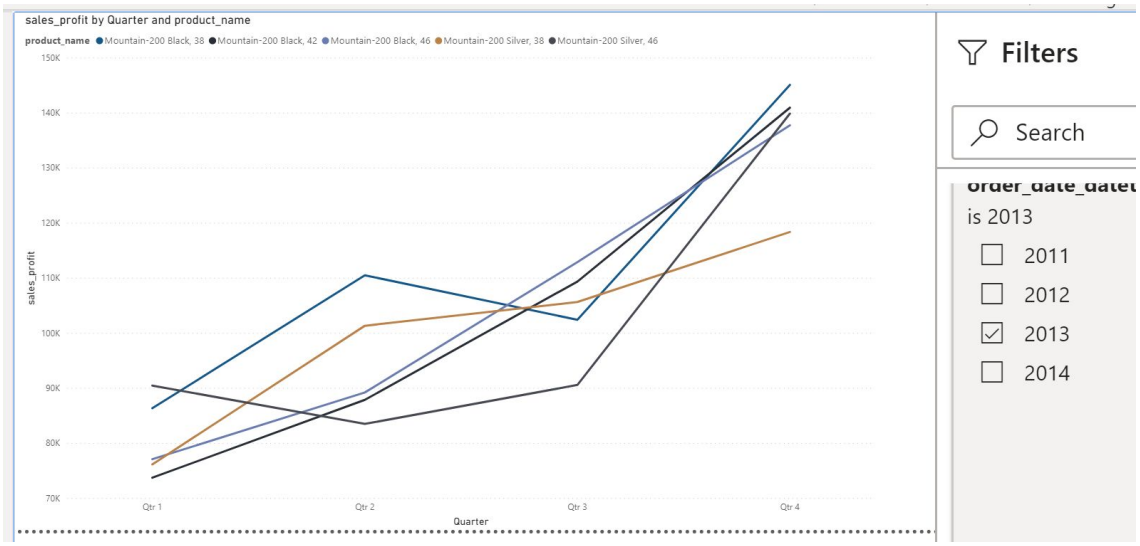
Looking specifically at reseller sales vs direct consumer sales over holidays we find the existence of sales to individual costumer across all holiday days. Regarding resellers the only sale during holiday days seem to be in the New Year's Day. This visualization is filtered on 2012 specifically. The second visualization looks at reseller sales and direct consumer sales month by month. One insight from this is seeing how resellers seem to be a more important segment in terms of sales for AdventureWorks.

2. Create top 5 products sales ranking by seasons.

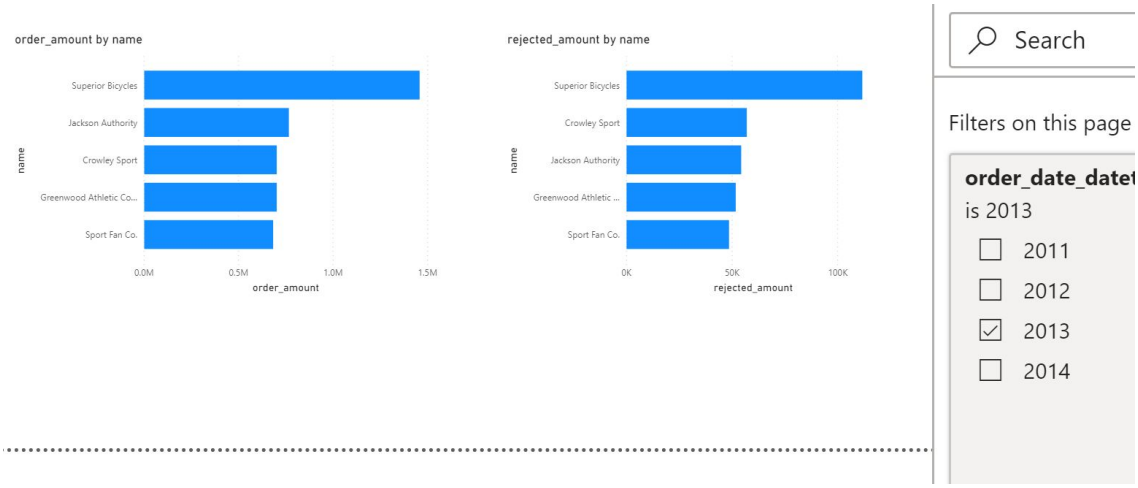


This visualization shows the top five product sales ranked by seasons. The most sales seem to be done around Autumn and Summer. The top products are all Mountain Bikes, but with different colors or sizes. This visualization looks at 2013 specifically.

3. Create top 5 products profit ranking, quarter by quarter



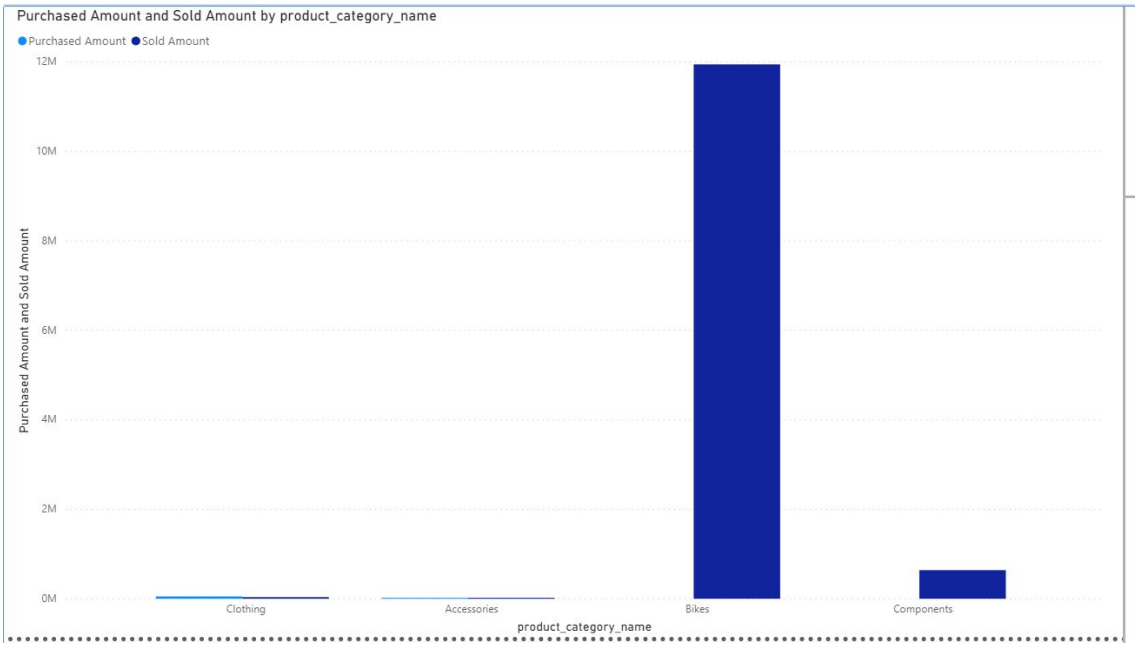
4. Create a top 5 Vendor ranking of rejected items on purchase orders.



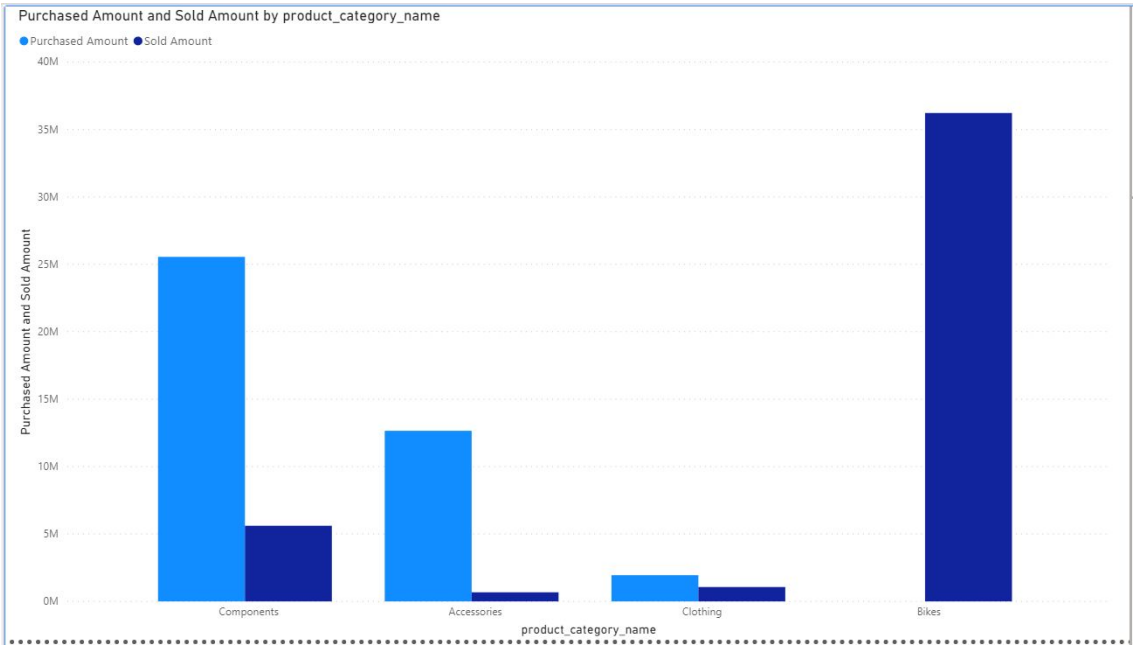
These two visualizations show the top 5 vendors in terms of order amount and rejected amount. Vendors who we order a lot of products from, also have a high amount of rejected products relative to other vendors.

5. See in a single view sold items vs purchased items, for each product category

In general these two visualizations show interesting information regarding which product categories AdventureWorks are earning and spending the most money on.



2011 Mostly bikes and components. Purchases might not be recorded in this year.



2013 Bikes is still the most sold category. Components, Accessories and Clothing purchases exceed the sales amount for these specific categories.

Appendix

1. Source Database

Database Engine	Microsoft SQL Server
Database Name	AdventureWorks2019

2. Staging Database

To setup the staging database execute DDL Script and then DML.

Database Engine	My SQL
Database Name	`staging_data_management_mmjja`
DDL Setup Script	`.\DB Scripts\Destination MySQL\ create.staging_data_management_mmjja.sql`
DML Setup Script	`.\DB Scripts\Destination MySQL\ init.staging_data_management_mmjja.sql`

3. Additional Data Source

This file should be in the same folder as the dim_date transformation.

Seasons csv file	`seasons_by_month.csv`

4. Destination Database

To setup the destination database execute DDL Script at My SQL Workbench.

Database Engine	My SQL
Database Name	`eae_data_management_mmjja`
DDL Setup Script	`.\DB Scripts\Destination MySQL\ create.eae_data_management_mmjja.sql`

5.ETL files

Customer transformation	`dim_customer.ktr`
Date transformation	`dim_date.ktr` Dependencies: ● `./seasons_by_month.csv` ● Connectivity to https://date.nager.at
Product Transformation	`dim_product.ktr`
Sales Person Transformation	`dim_sales_person.ktr`
Sales Territory Transformation	`dim_sales_territory.ktr`
Vendor transformation	`dim_vendor.ktr`
Purchases transformation	`fact_purchase.ktr`

Sales transformation	`fact_sales.ktr`
	
	When testing fact_sales transformation, consider restricting the Sales Order Detail input to these orders.
	These will reduce the data flow volume, for taking a sample of the entire sales.
	If taken into consideration the entire SalesOrderDetail table, it takes from 6 to 10 hours to finish. This is on a laptop running an Intel64 Family 6 Model 78 Stepping 3 GenuineIntel ~2492 Mhz, 16 Gb of RAM and SSD drive. It has Win10 home with both local MSSQL and MySQL
	
	`where SalesOrderID in (43659,43660,43661,43662,43663,43664,4,43666,43667,43668,51092)`
	
	We couldn't work on performance advantages that would expedite this process.

Dimension Tables Job	`DimensionsJob.kjb`

	Dependencies: • `./dim_*.ktr` (6 files above mentioned)
Fact Tables Job	`FactsJob.kjb` Dependencies: • `./DimensionsJob.kjb` • `./fact_purchase.ktr` • `./fact_sales.ktr`

6.Testing scenarios scripts

Enclosed there are also 4 test scenarios that need scripts to be executed at the source and destination databases.

Source MSSQL	`.\DB Scripts\Source MSSQL\transactional.testing.scenarios`
Destination MySQL	`.\DB Scripts\Destination MySQL\datamart.testing.scenarios.sql`