

Objective

This document serves as a guide to understanding the Annie's Magic Numbers use case. It outlines the initial approaches, exploratory data analysis, and diagrams, providing a structured overview of the process.

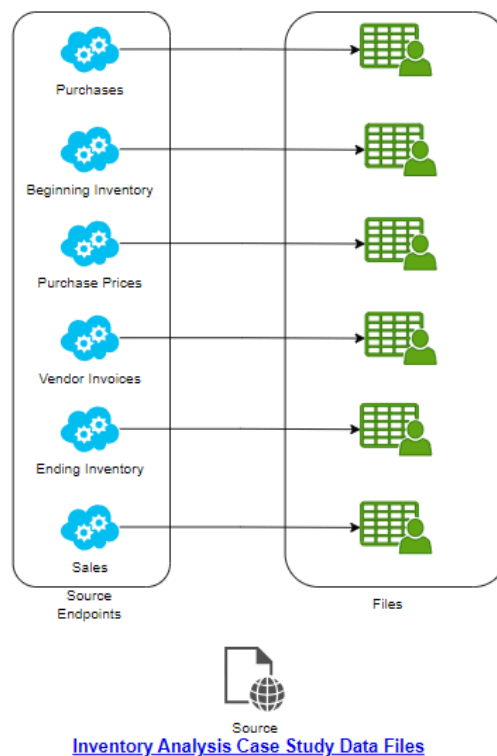
Data Sources

Location of the data

The data sources come from an endpoint available on the [Data analytics case study data files](#) page. Each endpoint provides CSV files that must be ingested to extract the information required by stakeholders.

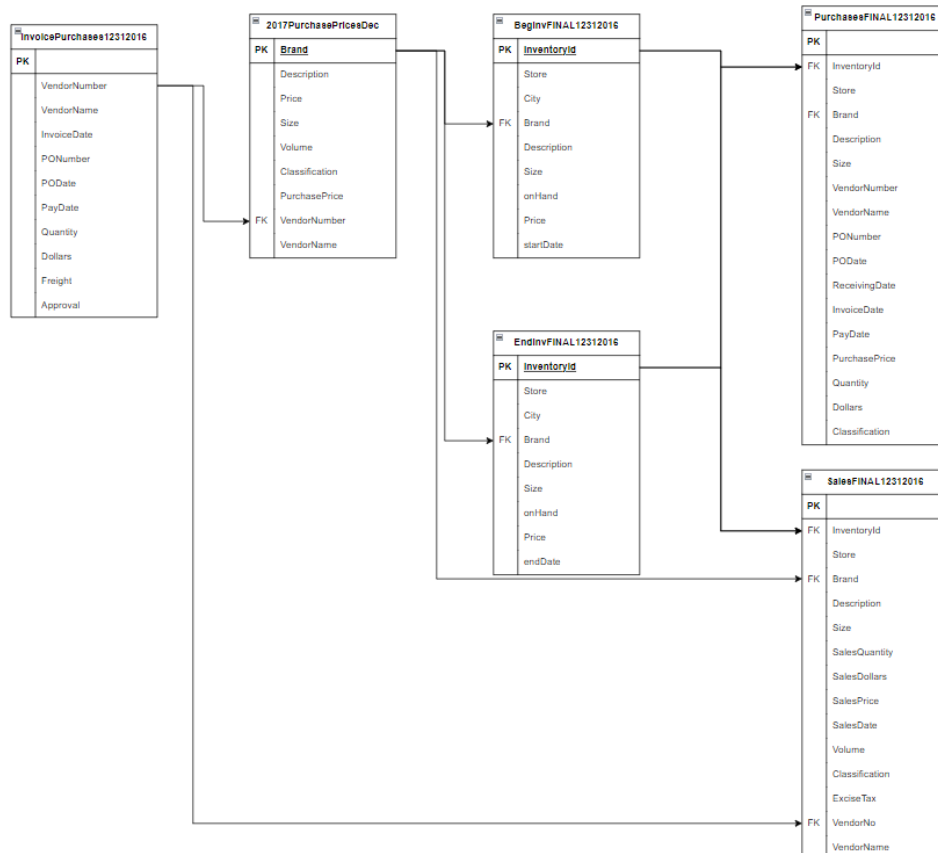
Data Relationships

In the initial analysis of the source, we identified data extracted from a transactional system. Some of these datasets appear to be related, as illustrated in the following diagram:



Some tables do not have a primary key, so it will be necessary to create one in the Data Warehouse.

During the initial analysis, we identified that some data is extracted from a transactional source. Based on this, we can infer potential relationships between datasets, as illustrated in the following diagram:



Calculations required

The calculation requested by the stakeholders are the following: **profits** and **margins**

Profit

$$\text{Profit} = \text{Total Revenue} - \text{Total Cost}$$

Margin

$$\text{Profit Margin} = \left(\frac{\text{Profit}}{\text{Total Revenue}} \right) \times 100$$

Exploratory Data Analysis

To perform the initial exploratory data analysis (EDA), we manually loaded the information into the Databricks Unity Catalog (Filestore). This allowed us to implement the first version of the profit and margin calculations.

The primary tables used for this analysis are:

- Sales
- Purchase

The SQL implementation is as follows:

```
%sql
SELECT
    s.InventoryId,
    s.Store,
    s.SalesDescription,
    s.SalesPrice,
    p.PurchasePrice,
    (s.SalesPrice - p.PurchasePrice) AS Profit,
    ((s.SalesPrice - p.PurchasePrice) / s.SalesPrice) * 100 AS ProfitMargin
FROM
    Sales s
LEFT JOIN
    PurchasePrices p
    ON s.Brand = p.Brand
LIMIT 5;
```

The complete exploratory data analysis is available in the following Databricks notebook: [AMN - Databricks Notebook](#)

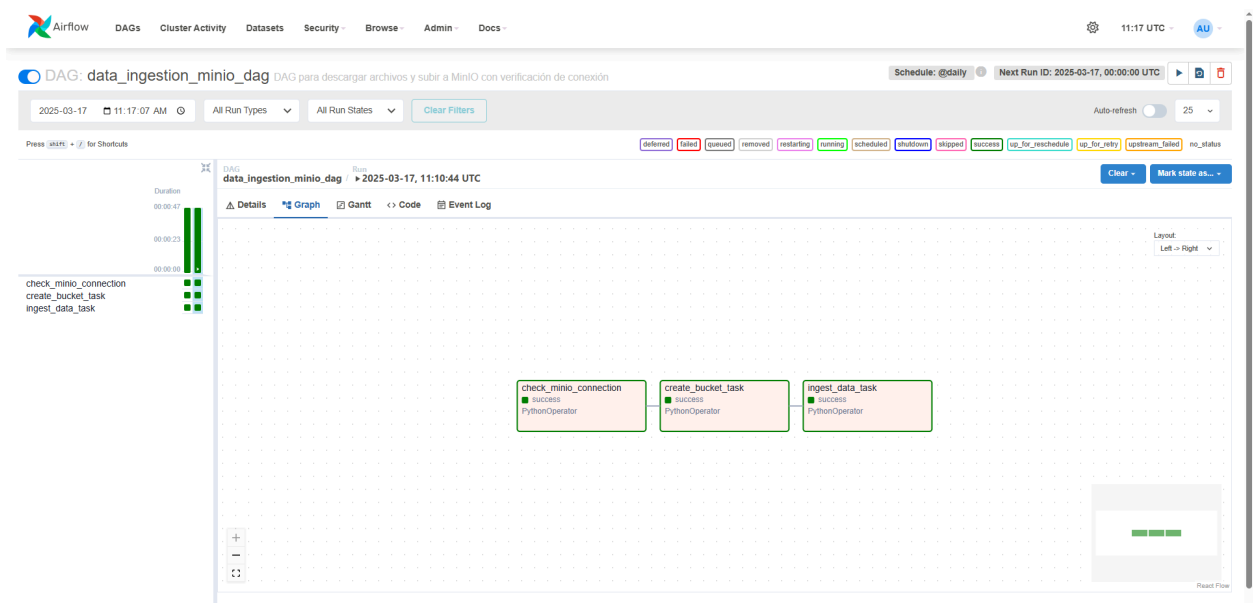
Brands not profitable

```
SELECT
    s.Brand,
    s.SalesDescription,
    (s.SalesPrice - p.PurchasePrice) AS Profit
FROM
    Sales s
LEFT JOIN
    PurchasePrices p
ON s.Brand = p.Brand
WHERE
    (s.SalesPrice - p.PurchasePrice) < 0
```

Steps

Ingest Data from Sources to S3 (Minio)

The movement of the data is made in a dag which moves from the endpoints to the data lake as .zip



Location of the objects

MINIO
OBJECT STORE
2017 LICENSE

User

Object Browser

Access Keys

Documentation

Administrator

Buckets

Policies

Identity

Monitoring

Events

Configuration

License

Object Browser

Search Start typing to filter objects in the bucket

Settings

Help

Dark Mode

rawdata

Created on: Mon, Mar 17 2025 08:10:49 (GMT-3) Access: PRIVATE 195.8 MiB - 6 Objects

Rewind

Refresh

Upload

rawdata

Create new path

Name	Last Modified	Size
2017PurchasePricesDec.csv.zip	Today, 08:10	270.3 KiB
BegInvFINAL12312016.csv.zip	Today, 08:10	3.9 MiB
EndInvFINAL12312016.csv.zip	Today, 08:11	4.2 MiB
PurchasesFINAL12312016.csv.zip	Today, 08:10	43.8 MiB
SalesFINAL12312016.csv.zip	Today, 08:11	143.5 MiB
VendorInvoices12312016.csv.zip	Today, 08:10	120.0 KiB

Transforming the data taking only the information we need

Using Implementing a job in spark the information is taken from the data-lake and load into a relational database in postgres

Loading data

DAG: data_transformation_dag DAG para transformar datos con Spark y cargarlos en PostgreSQL

Schedule: @daily Next Run ID: 2025-03-17, 00:00:00 UTC

2025-03-17 11:17:09 AM All Run Types All Run States Clear Filters

Auto-refresh 25

Press [Shift] + [J] for Shortcuts

deferred failed queued removed restarting running scheduled shutdown skipped success up_for_reschedule up_for_retry upstream_failed no_status

Clear Mark state as...

data_transformation_dag 2025-03-17, 11:15:59 UTC

Details Graph Gantt Code Event Log

Layout: Left -> Right

download_and_load_to_spark PythonOperator

load_to_postgres PythonOperator

React Flow

Spark Jobs

Spark Master at spark://f991cdfec9e:7077

URL: spark://f991cdfec9e:7077
Alive Workers: 1
Cores in use: 8 Total: 0 Used
Memory in use: 6.6 GB Total: 0.0 B Used
Resources in use:
Applications: 0 Running, 21 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker id	Address	State	Cores	Memory	Resources
worker-20250317110900-172.20.0.6-33801	172.20.0.6:33801	ALIVE	8 (0 Used)	6.6 GB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Completed Applications (21)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20250317111809-2032	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:18:09	airflow	FINISHED	1 s
app-20250317111823-2019	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:18:23	airflow	FINISHED	1 s
app-20250317111757-2016	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:17:57	airflow	FINISHED	00 s
app-20250317111721-2017	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:17:21	airflow	FINISHED	1 s
app-20250317111645-2016	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:16:45	airflow	FINISHED	00 s
app-20250317111619-2015	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:16:19	airflow	FINISHED	3 s
app-20250317111619-2014	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:16:19	airflow	FINISHED	3 s
app-20250317111608-2012	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:16:08	airflow	FINISHED	3 s
app-20250317111608-2011	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:16:08	airflow	FINISHED	3 s
app-20250317111610-2013	Data Transformation with Spark	0	1024.0 MiB		2025/03/17 11:16:10	airflow	FINISHED	1 s
app-20250317111530-2010	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:15:30	airflow	FINISHED	1 s
app-20250317111453-2009	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:14:53	airflow	FINISHED	1 s
app-20250317111417-2006	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:14:17	airflow	FINISHED	1 s
app-20250317111341-2007	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:13:41	airflow	FINISHED	1 s
app-20250317111305-2006	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:13:05	airflow	FINISHED	00 s
app-20250317111229-2005	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:12:29	airflow	FINISHED	00 s
app-20250317111163-2004	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:11:63	airflow	FINISHED	1 s
app-20250317111116-2003	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:11:16	airflow	FINISHED	3 s
app-20250317111029-2002	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:10:29	airflow	FINISHED	10 s
app-20250317110953-2001	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:09:53	airflow	FINISHED	1 s
app-20250317110917-2000	Data Transformation with Spark	8	1024.0 MiB		2025/03/17 11:09:17	airflow	FINISHED	2 s

Destination in the warehouse

amn_datawarehouse 17.4 (Debian 17.4-1.pgdg120+2)

amn_datawarehouse

public

Query

Tables (2)

purchase_prices

columns

inventoryid text

store integer

brand integer

description text

size text

vendornumber integer

vendorname text

ponumber integer

podate date

receivingdate date

invoicedate date

paydate date

purchaseprice double precision

quantity integer

dollars double precision

classification integer

Index

Triggers

sales

columns

inventoryid text

store integer

brand integer

description text

size text

salesquantity integer

salesdollars double precision

salesprice double precision

salesdate date

volume double precision

classification integer

excisetax double precision

vendorno integer

vendorname text

Index

Triggers