24/11/2025

HANDLEBAR HAVEN DATA WAREHOUSE

ETL Implementation and Business Intelligence Analysis

GEORGY ONISHI

BSAN7206: Assignment 2

Word Count: 3,774

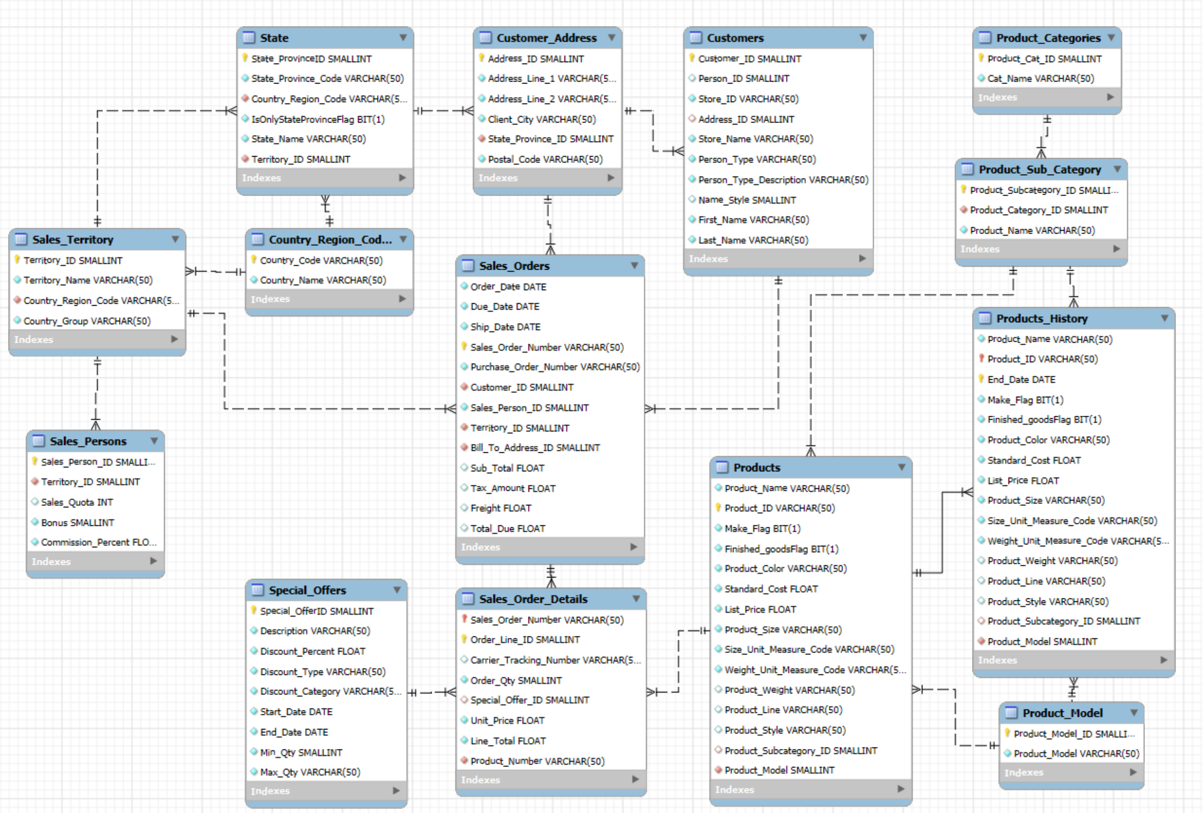
Contents

1. Introduction

Handlebar Haven (HH) is a bicycle manufacturer and retailer serving a worldwide customer base across United States, Canada, Australia, the United Kingdom, France, and Germany. The company manufactures and sells several categories of bicycles, and also re-sells accessories, clothing, and components sourced from external suppliers. With a customer base spanning 700 retail stores and over 18,000 individual customers, HH operates through both business-to-business sales as well as direct-to-consumer.

The company has an operational database comprised of multiple interconnected systems capturing customer, product, transaction, supplier, and other information, with over 31,000 transactional records spanning from a period of three years between 2021~2024, providing a rich source for analytical insights.

Handlebar Haven's Operational Database Schema

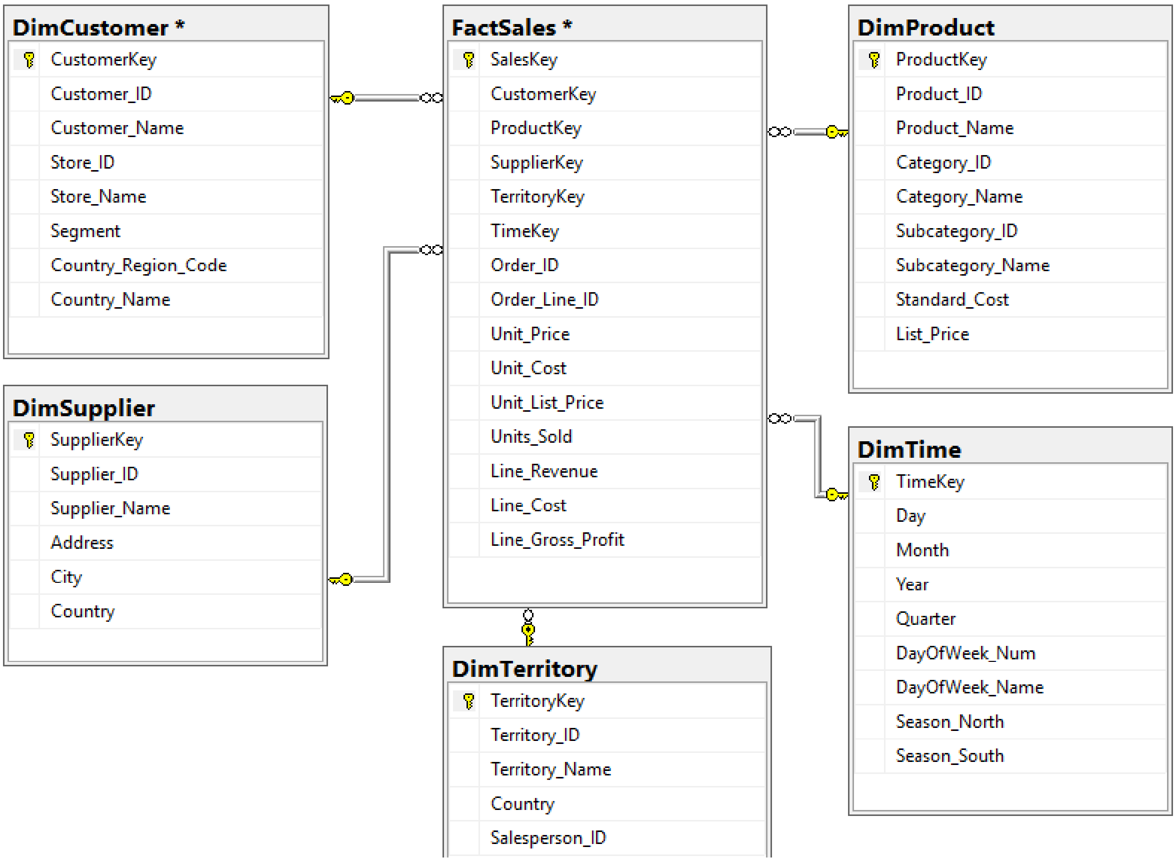


HH has experienced strong growth and consistent profitability in recent years. Management is now exploring opportunities for further expansion. As such they are seeking evidenced-based analysis to answer questions around identifying key customers, most profitable products, sales territories, time periods, as well as identifying low-performing product categories and sub-categories for replacement.

This report covers our implementation of a comprehensive business analytics solution designed for HH. We present our dimensional data warehouse design that transforms fragmented operational data into an integrated analytical platform. We explain our complete ETL (Extract, Transform, Load) process, from initial data extraction through final dimensional model population, ensuring data quality and business rule implementation throughout. We leverage the resulting data warehouse to provide data-driven insights and strategic recommendations addressing Handlebar Haven's five key business questions, enabling informed decisions that support their continued growth and market expansion.

2. Dimensional Model Design

HH requires a dimensional model to analyse customer profitability, product performance, territory results, seasonal/time-based trends, and low-value product categories. We have designed the following model to meet these analytic requirements:



In designing our dimensional model for HH we followed the four-step methodology of Kimball & Ross (2013).

2.1 Select business process: Our schema models HH's sales process, which we capture in the transactional fact table FactSales. FactSales enables analysis of unit sales, revenues, and margins at the order line level.

2.2 Declare data grain: FactSales has a grain of one row per sales order line. We omitted non-essential details (such as detailed address fields, freight, or cosmetic product desceriptors) to keep the model lean and focused on the insights required by HH. This captures sales at their most detailed level so results can be aggregated by the appropriate dimension.

2.3 Identify the dimensions: We selected DimDate, DimCustomer, DimProduct and DimTerritory, which together provide the slicing and hierarchies needed for HH's key business questions. A promotion dimension was omitted, as the business case did not require promotion-level analysis.

2.4 Identify the fact measure: FactSales includes business metrics which support profitability analysis by product, customer, territory, and time period; these metrics are covered in further detail in section 3.4 of this report.

3. Data Integration and ETL

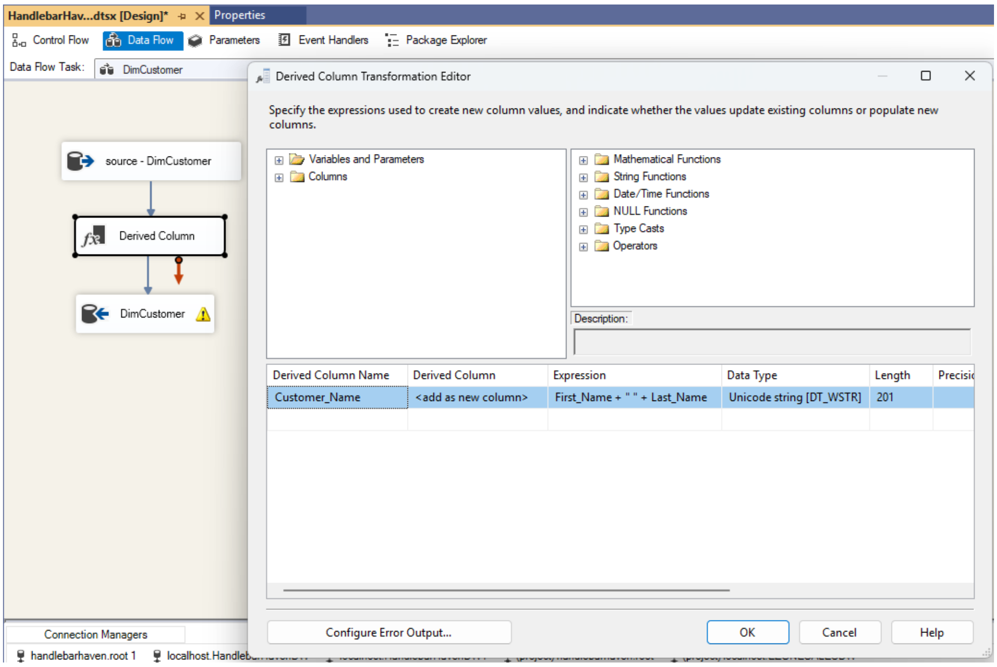
3.1 Overview of the ETL Process

I used SSIS (SQL Server Integration Services) to integrate data from HH's transactional sales system into a data warehouse based on the proposed dimensional model, following ETL methodology:

1. **Database preparation (Pre-ETL):** Based on my dimensional model, I used CREATE TABLE scripts to set up the data warehouse and target tables to be loaded in later steps. The complete schema DDL scripts can be viewed at Appendix D.2.
2. **Extraction:** I employed SQL queries with denormalised JOINs directly against the source database. I used LEFT JOINs with COALESCE functions to handle incomplete data, as when 62 customers that did not have address details were assigned 'Unknown' values rather than omitted. The complete SQL queries that I used can be viewed at Appendix D.1.
3. **Transformation:** I used SSIS Derived Column transformations to calculate and generate surrogate keys and temporal attributes, standardise data formats, concatenate attributes, etc., ensuring that my tables and attributes corresponded to the final data warehouse design.
4. **Loading sequence:** I loaded the five dimension tables in parallel, followed by the fact table, into my data warehouse in SSMS (SQL Server Management Studio); this sequence ensured that SSIS Lookup transformations would correctly link the business keys to their respective surrogate keys.

3.2 SSIS Transformations

3.2.1 DimCustomer

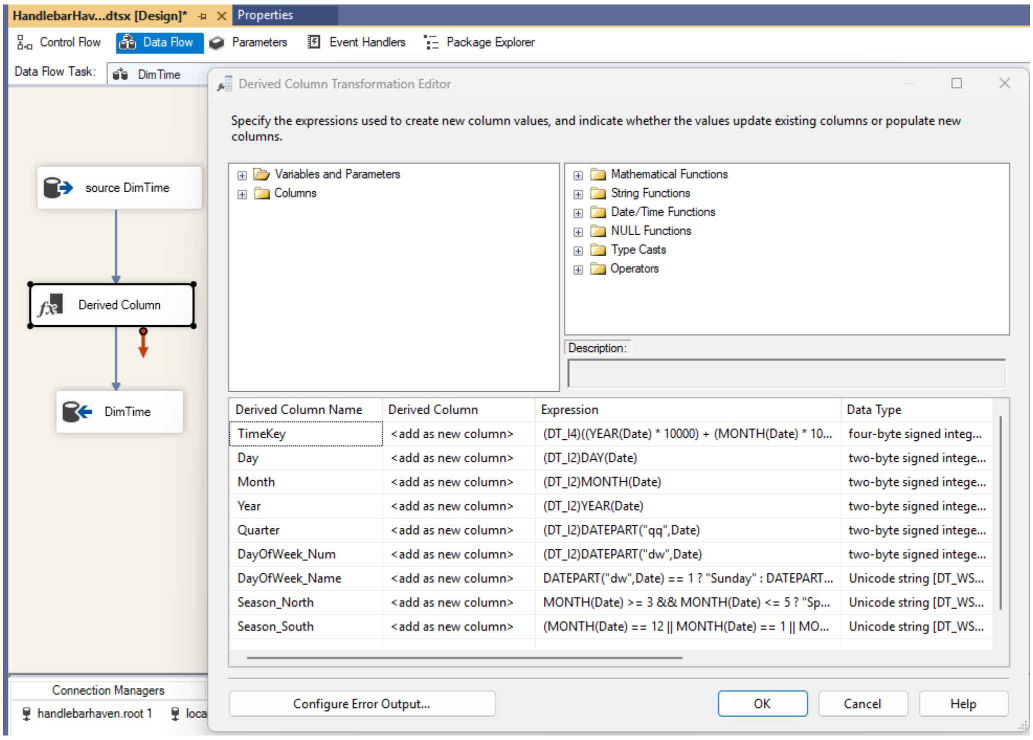


Specify the expressions used to create new column values, and indicate whether the values update existing columns or populate new columns.

Derived Column Name	Derived Column	Expression	Data Type	Length	Precisi
Customer_Name	<add as new column>	First_Name + " " + Last_Name	Unicode string [DT_WSTR]	201	

Attribute	Type	Transformation	Notes
Customer_Name	Derived column	First_Name + " " + Last_Name	• New attribute containing customer full name – for simplicity, in case of duplicates CustomerKey can be referenced

3.2.2 DimTime



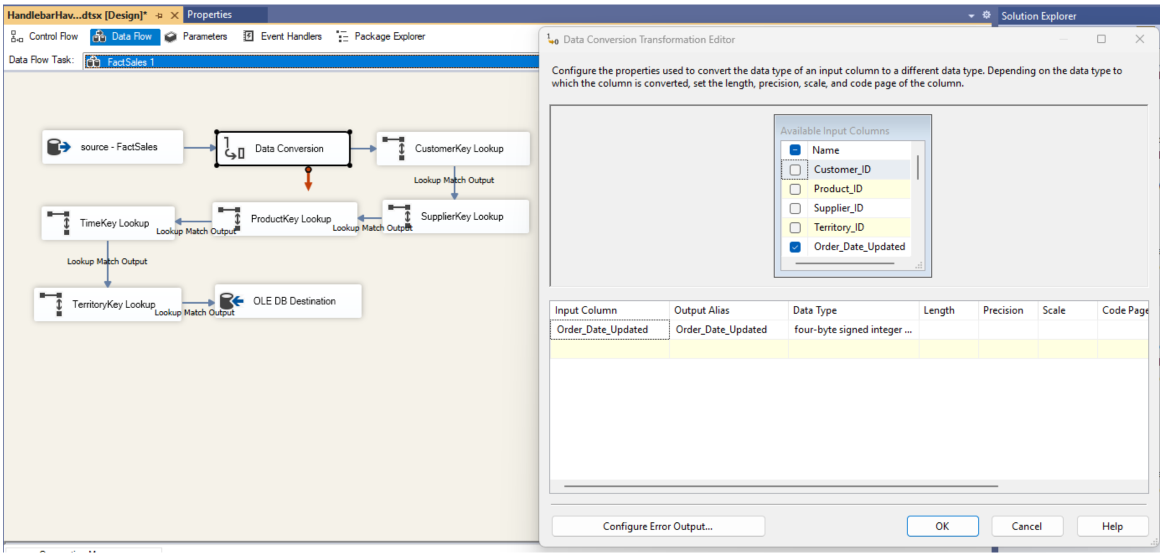
Specify the expressions used to create new column values, and indicate whether the values update existing columns or populate new columns.

Derived Column Name	Derived Column	Expression	Data Type
TimeKey	<add as new column>	(DT_I4)((YEAR(Date) * 10000) + (MONTH(Date) * 10...	four-byte signed integ...
Day	<add as new column>	(DT_I2)DAY(Date)	two-byte signed intege...
Month	<add as new column>	(DT_I2)MONTH(Date)	two-byte signed intege...
Year	<add as new column>	(DT_I2)YEAR(Date)	two-byte signed intege...
Quarter	<add as new column>	(DT_I2)DATEPART("qq",Date)	two-byte signed intege...
DayOfWeek_Num	<add as new column>	(DT_I2)DATEPART("dw",Date)	two-byte signed intege...
DayOfWeek_Name	<add as new column>	DATEPART("dw",Date) == 1 ? "Sunday" : DATEPART...	Unicode string [DT_WS...
Season_North	<add as new column>	MONTH(Date) >= 3 && MONTH(Date) <= 5 ? "Sp...	Unicode string [DT_WS...
Season_South	<add as new column>	(MONTH(Date) == 12 || MONTH(Date) == 1 || MO...	Unicode string [DT_WS...

Attribute	Type	Transformation	Notes
TimeKey	Derived column	(DT_I4)((YEAR(Date) * 10000) + (MONTH(Date) * 100) + DAY(Date))	• TimeKey is the 8-digit integer for yyyymmdd; easily converted to DATE format
Day	Derived column	(DT_I2)DAY(Date)	• Derived Day, Month, Year, and Quarter to enable temporal analyses at different grains
Month	Derived column	(DT_I2)MONTH(Date)	
Year	Derived column	(DT_I2)YEAR(Date)	
Quarter	Derived column	(DT_I2)DATEPART("qq",Date)	
DayOfWeek_Num	Derived column	(DT_I2)(DATEPART("dw",Date) == 1 ? 7 : DATEPART("dw",Date) - 1)	• Derived day of week as both a number and string in separate columns; shows descriptive info while also allowing quantitative day of week analyses • Monday = 1 (NOT Sunday = 1) so that weekend days are not split at 1 and 7
DayOfWeek_Name	Derived column	DATEPART("dw",Date) == 2 ? "Monday" : DATEPART("dw",Date) == 3 ? "Tuesday" : DATEPART("dw",Date) == 4 ? "Wednesday" : DATEPART("dw",Date) == 5 ? "Thursday" : DATEPART("dw",Date) == 6 ? "Friday" : DATEPART("dw",Date) == 7 ? "Saturday" : "Sunday"	
Season_North	Derived column	MONTH(Date) >= 3 && MONTH(Date) <= 5 ? "Spring" : MONTH(Date) >= 6	• Separate seasons attribute required for

		&& MONTH(Date) <= 8 ? "Summer" : MONTH(Date) >= 9 && MONTH(Date) <= 11 ? "Autumn" : "Winter"	Northern and Southern hemispheres, given large customer base in Australia • CASE statement querying enables assigning seasons to regions obtain seasonal analyses
Season_South	Derived column	(MONTH(Date) == 12 || MONTH(Date) == 1 || MONTH(Date) == 2) ? "Summer" : (MONTH(Date) >= 3 && MONTH(Date) <= 5) ? "Autumn" : (MONTH(Date) >= 6 && MONTH(Date) <= 8) ? "Winter" : "Spring"	

3.2.3 *FactSales*



Configure the properties used to convert the data type of an input column to a different data type. Depending on the data type to which the column is converted, set the length, precision, scale, and code page of the column.

Available Input Columns

☐	Name
☐	Customer_ID
☐	Product_ID
☐	Supplier_ID
☐	Territory_ID
☑	Order_Date_Updated

Input Column	Output Alias	Data Type	Length	Precision	Scale	Code Page
Order_Date_Updated	Order_Date_Updated	four-byte signed integer ...				

Attribute	Type	Transformation	Notes
Order_Date	Data conversion	Convert to DT_I4 four-byte integer	• Attributes need to match format to perform a lookup operation; this data conversion was made so Order_Date matches TimeKey format
CustomerKey	Lookup	Lookup DimCustomer using Customer_ID to retrieve CustomerKey surrogate key	• Links fact records to customer dimension via surrogate key
SupplierKey	Lookup	Lookup DimSupplier using Supplier_ID to retrieve SupplierKey surrogate key	• Links fact records to supplier dimension via surrogate key
ProductKey	Lookup	Lookup DimProduct using Product_ID to retrieve ProductKey surrogate key	• Links fact records to product dimension via surrogate key
TimeKey	Lookup	Lookup DimTime using converted Order_Date (yyyymmdd format) to retrieve TimeKey surrogate key	• Links fact records to time dimension; requires date format conversion for matching

TerritoryKey	Lookup	Lookup DimTerritory using Territory_ID to retrieve TerritoryKey surrogate key	- Links fact records to territory dimension via surrogate key

3.3 Data Sourcing and Mapping

3.3.1 DimCustomer

DimCustomer Attribute	Source Database Attribute	Source DB Table	Notes
CustomerKey	[Auto-generated]	N/A	Surrogate key generated during ETL
Customer_ID	Customer_ID	customers	Natural key from operational system, maintains traceability to source
Customer_Name	First_Name + Last_Name	customers	Concatenated full name for simplified reporting and analysis
Store_ID	Store_ID	customers	Identifies retail store customers for B2B analysis; NULL for individual consumers
Store_Name	Store_Name	customers	Retail store business name for B2B customer identification
Segment	Person_Type	customers	Customer classification (Individual Consumer vs Store) for segmentation analysis
Country_Region_Code	Country_Region_Code	country_region_codes	Country codes for geographic analysis and territory assignment
Country_Name	Country_Name	sales_territory	Full country names for user-friendly geographic reporting

3.3.2 DimSupplier

DimSupplier Attribute	Source Database Attribute	Source DB Table	Notes
SupplierKey	[Auto-generated]	N/A	Surrogate key generated during ETL process for dimensional integrity
Supplier_ID	Supplier_ID	supplier	Natural key from operational system, maintains traceability to source
Supplier_Name	Supplier_Name	supplier	Company name for supplier identification and reporting
Address	Address	supplier	Street address for supplier location analysis
City	City	supplier	City location for geographic analysis
Country	Country	supplier	Country location for geographic analysis

3.3.3 DimProduct

DimProduct Attribute	Source Database Attribute	Source DB Table	Notes
ProductKey	[Auto-generated]	N/A	Surrogate key generated during ETL process for dimensional integrity
Product_ID	Product_ID	products	Natural key from operational system; maintains traceability to source

Product_Name	Product_Name	products	Product name for identification and reporting
Category_ID	product_cat_ID	product_categories (via Product_Subcategory_ID lookup)	Category identifier obtained through subcategory relationship
Category_Name	cat_name	product_categories (via Product_Subcategory_ID lookup)	Category name for product classification and analysis
Subcategory_ID	product_subcategory_ID	product_sub_category	Subcategory identifier from products table via join
Subcategory_Name	product_name	product_sub_category	Subcategory name for detailed product classification
Standard_Cost	Standard_Cost	products	Product cost for margin and profitability calculations
List_Price	List_Price	products	Manufacturer suggested retail price for pricing analysis

3.3.4 *DimTime*

DimTime Attribute	Source Database Attribute	Source DB Table	Notes
TimeKey	[Derived from Order_Date]	sales_orders	Calculated as YYYYMMDD integer format from Order_Date for efficient lookup operations
Day	[Derived from Order_Date]	sales_orders	Extracted day component using DAY() function
Month	[Derived from Order_Date]	sales_orders	Extracted month component using MONTH() function
Year	[Derived from Order_Date]	sales_orders	Extracted year component using YEAR() function
Quarter	[Derived from Order_Date]	sales_orders	Calculated using DATEPART("qq") function for quarterly analysis
DayOfWeek_Num	[Derived from Order_Date]	sales_orders	Calculated day of week number with Monday=1 for business logic alignment
DayOfWeek_Name	[Derived from Order_Date]	sales_orders	Descriptive day name derived from DATEPART("dw") with conditional logic
Season_North	[Derived from Order_Date]	sales_orders	Northern hemisphere seasons calculated from month values for US/European customers
Season_South	[Derived from Order_Date]	sales_orders	Southern hemisphere seasons calculated from month values for Australian customers

3.3.5 *DimTerritory*

DimTerritory Attribute	Source Database Attribute	Source Database Table	Notes
TerritoryKey	[Auto-generated]	N/A	Surrogate key generated during ETL process for dimensional integrity

Territory_ID	Territory_ID	sales_territory	Natural key from operational system; maintains traceability to source
Territory_Name	Territory_Name	sales_territory	Territory name for geographic identification and reporting
Country	Country_Name	country_region_codes	Full country name through region code relationship for user-friendly geographic analysis
Salesperson_ID	Sales_Person_ID	sales_persons	Links territory to assigned salesperson; LEFT JOIN handles territories without assigned sales staff

3.3.6 FactSales

FactSales Attribute	Source Database Attribute	Source Database Table	Notes
SalesKey	[Auto-generated]		Surrogate primary key
CustomerKey	[Auto-generated]		Foreign key
ProductKey	[Auto-generated]		Foreign key
SupplierKey	[Auto-generated]		Foreign key
TerritoryKey	[Auto-generated]		Foreign key
TimeKey	Order_Date (converted to YYYYMMDD)	sales_orders	Foreign key requiring lookup to DimTime
Order_ID	Sales_Order_Number	sales_order_details	Business key for order identification and traceability
Order_Line_ID	Order_Line_ID	sales_order_details	Line item identifier within each order
Unit_Price	Unit_Price	sales_order_details	Actual selling price per unit for the transaction
Unit_Cost	Standard_Cost	products	Product standard cost for margin calculations
Unit_List_Price	List_Price	products	Manufacturer suggested retail price for pricing analysis
Units_Sold	Order_Qty	sales_order_details	Quantity of units sold in the transaction
Line_Revenue	Line_Total	sales_order_details	Total revenue for the order line (Unit_Price x Units_Sold)
Line_Cost	[Calculated]	products, sales_order_details	Calculated as Standard_Cost x Order_Qty for profitability analysis
Line_Gross_Profit	[Calculated]	sales_order_details, products	Calculated as (Unit_Price - Standard_Cost) x Order_Qty for profitability analysis

3.4 Metric Calculations

The following FactSales attributes are sourced from the source database, in the sales_order_details and products tables:

1. Unit_Price,
2. Unit_Cost,

3. Unit_List_Price,
4. Units_Sold

From these measures we derived the following additional business metrics:

5. Line_Revenue = Unit_Price × Units_Sold
6. Line_Cost = Unit_Cost × Units_Sold
7. Line_Gross_Profit = (Unit_Price - Unit_Cost) × Units_Sold
8. (Average_profit_margin was not included as a FactSales attribute, but will be calculated during the analytical querying phase via Line_Gross_Profit / Line_Revenue * 100)

These transformations were developed during the dimensional model design phase and then implemented at the 'Transform' step of the ETL process.

We selected these metrics to respond to HH's business questions, which are asking about profitability in conjunction with other parameters such as Seasons, Territories, and Product Categories. HH's source operational database could be queried to generate these insights, but only through long and complex queries with many tabular joins. Our dimensional model design enables us to efficiently query the database to generate the requested insights.

4. Business Insights and Recommendations

I completed this section by running SQL queries on the completed data warehouse to return tables with the correct attributes and grain that correspond to each business question. These tables were then read into Power BI for data visualisation that enabled exploration and analysis that generated insights and thereby responded to the questions.

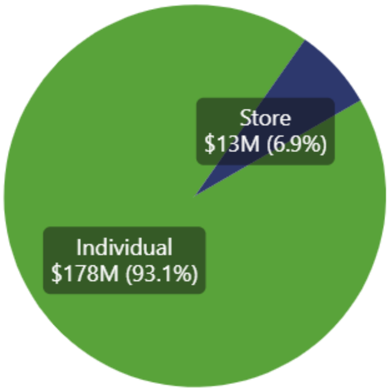
4.1 Key Customers

Figure 1: Top 25 Customers by Revenue (All Time)

	Client	Segment	Country	Units Sold	Total Revenue	Gross Profit	Profit Margin (%)
1	Bethany Chande	Individual	Australia	270	$312,795	$99,272	31.7
2	Cynthia Malhotra	Individual	Australia	440	$311,523	$128,059	41.1
3	Calvi Xu	Individual	US	286	$286,795	$60,712	21.2
4	Javier Hernandez	Individual	UK	396	$249,532	$69,199	27.7
5	Steve Gao	Individual	France	185	$234,864	$92,109	39.2
6	Leonard Sha	Individual	US	201	$224,219	$66,889	29.8
7	Jessie Yang	Individual	US	181	$221,416	$75,598	34.1
8	Richard Wood	Individual	Australia	274	$214,991	$76,669	35.7
9	Dalto Roberts	Individual	US	343	$214,673	$58,592	27.3
10	Brando Smith	Individual	US	235	$207,071	$45,125	21.8
11	Tamara Nath	Individual	Australia	189	$205,243	$84,909	41.4
12	Mario She	Individual	Germany	181	$201,096	$64,786	32.2
13	Garrett James	Individual	US	219	$200,395	$84,656	42.2
14	Natasha Ruiz	Individual	Canada	167	$199,575	$67,196	33.7
15	Kristy Hernandez	Individual	Canada	317	$195,267	$65,966	33.8
16	Efficient Cycling	Store	France	142	$194,800	$86,963	44.6
17	Meredith Prasad	Individual	US	286	$193,907	$83,933	43.3
18	Jorge Liu	Individual	US	336	$192,419	$73,165	38
19	Kately Wright	Individual	US	188	$189,512	$64,966	34.3
20	Marcus King	Individual	France	117	$186,424	$63,020	33.8
21	Morga Phillips	Individual	US	269	$185,925	$71,575	38.5
22	Stephanie Ward	Individual	US	192	$185,824	$73,998	39.8
23	Sheena Lal	Individual	France	186	$183,080	$57,108	31.2
24	Janet Gill	Individual	France	149	$177,812	$68,641	38.6
25	Alexandria Gonzales	Individual	Australia	179	$177,806	$75,955	42.7

B2C Dominance: Direct-to-consumer sales comprise 93% of revenue versus 7% B2B. Only one B2B customer (Efficient Cycling, France) ranks in the top 25, highlighting retail channel dependence.

Figure 2: Revenue Share by Client Segment

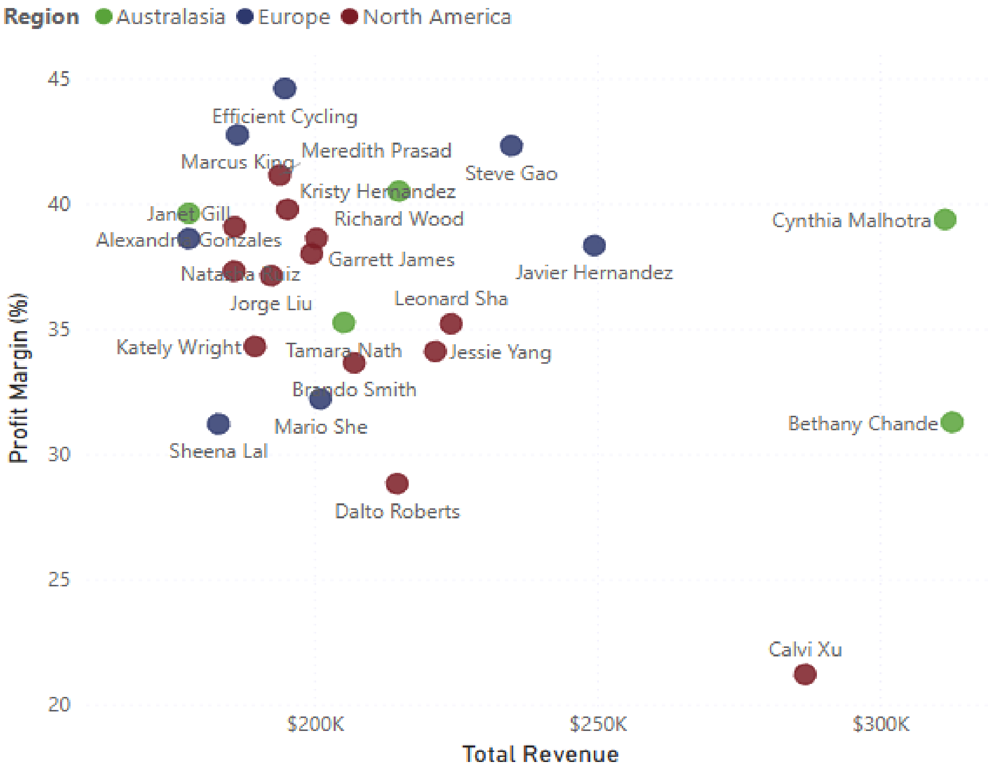


Customer Profitability Patterns:

a) **High-margin outlier:** Efficient Cycling delivers exceptional margins (44.6%) – warrants targeted B2B development

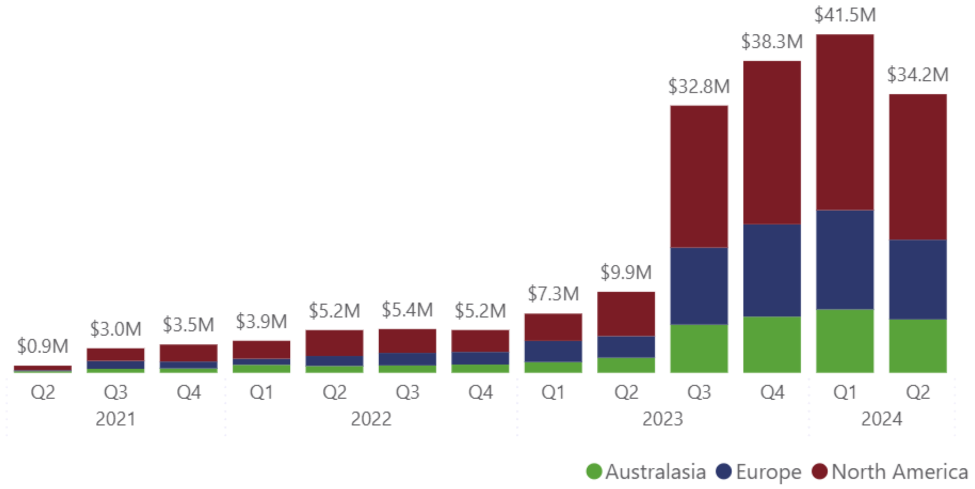
b) **Low-margin concerns:** Calvin Xu and Brando Smith generate high revenue but poor margins – investigate discount purchasing behaviour or product mix

Figure 3: Top 25 Clients, Total Revenue vs Profit Margin



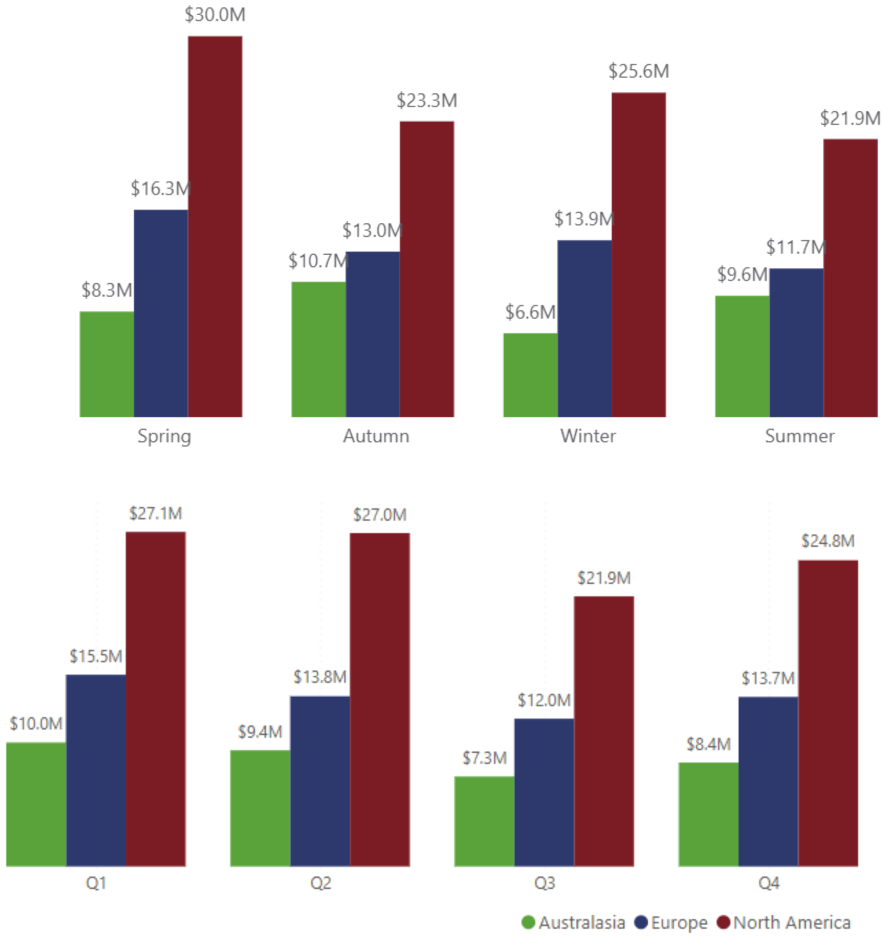
Revenue Growth Timeline: Figure 4 shows a dramatic Q2~Q3 2023 surge from $9.9M to $32.8M (287% growth), sustaining at $30M+ quarterly through Q1-2024. However, Q1~Q2-2024 decline of 17% aligns with territorial profit drops – suggests systemic issue.

Figure 4: Total Revenue over time (Q2-2021~Q2-2024)



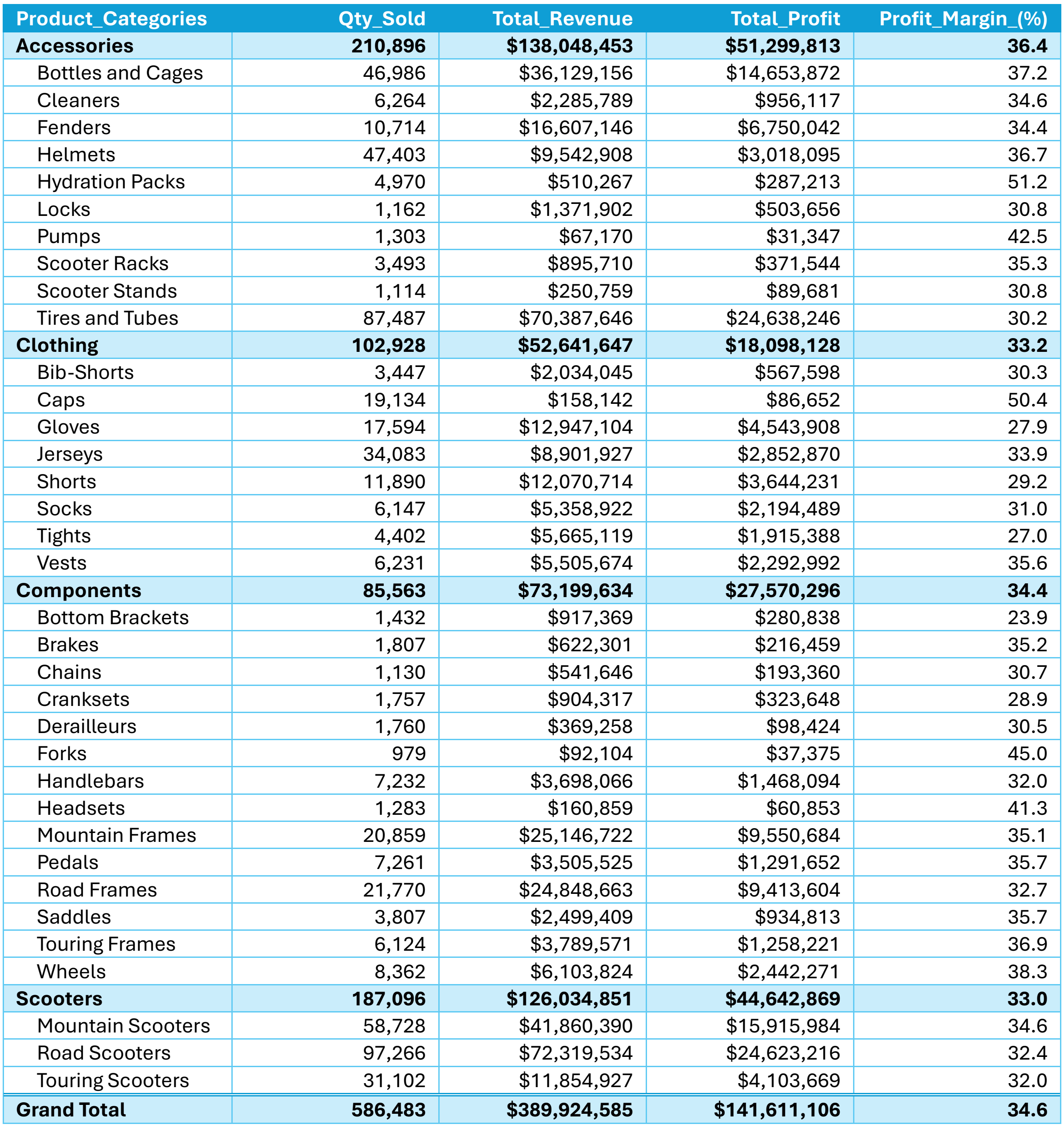
Non-Seasonal Sales Cycles: Figure 5 shows peak performance in March~May ($40M), strong October~February ($35-38M), with June~September trough (~$24M). Pattern consistency across regions indicates business/fiscal drivers rather than weather dependency.

Figure 5: Total Revenue by Season & Quarter & Region



4.2 Most profitable products

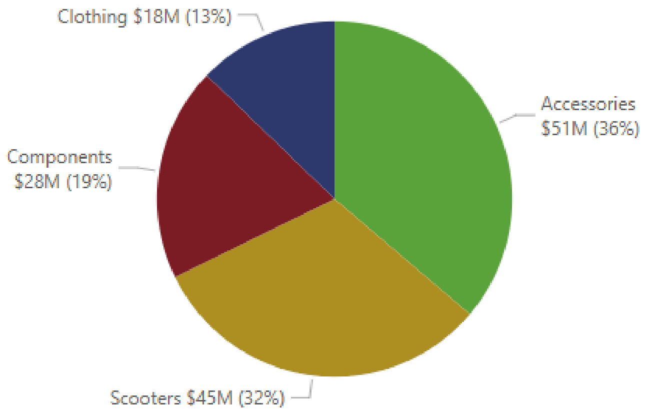
Figure 6: Product Performance by Category and Subcategory



Product_Categories	Qty_Sold	Total_Revenue	Total_Profit	Profit_Margin_(%)
Accessories	**210,896**	**$138,048,453**	**$51,299,813**	**36.4**
Bottles and Cages	46,986	$36,129,156	$14,653,872	37.2
Cleaners	6,264	$2,285,789	$956,117	34.6
Fenders	10,714	$16,607,146	$6,750,042	34.4
Helmets	47,403	$9,542,908	$3,018,095	36.7
Hydration Packs	4,970	$510,267	$287,213	51.2
Locks	1,162	$1,371,902	$503,656	30.8
Pumps	1,303	$67,170	$31,347	42.5
Scooter Racks	3,493	$895,710	$371,544	35.3
Scooter Stands	1,114	$250,759	$89,681	30.8
Tires and Tubes	87,487	$70,387,646	$24,638,246	30.2
Clothing	**102,928**	**$52,641,647**	**$18,098,128**	**33.2**
Bib-Shorts	3,447	$2,034,045	$567,598	30.3
Caps	19,134	$158,142	$86,652	50.4
Gloves	17,594	$12,947,104	$4,543,908	27.9
Jerseys	34,083	$8,901,927	$2,852,870	33.9
Shorts	11,890	$12,070,714	$3,644,231	29.2
Socks	6,147	$5,358,922	$2,194,489	31.0
Tights	4,402	$5,665,119	$1,915,388	27.0
Vests	6,231	$5,505,674	$2,292,992	35.6
Components	**85,563**	**$73,199,634**	**$27,570,296**	**34.4**
Bottom Brackets	1,432	$917,369	$280,838	23.9
Brakes	1,807	$622,301	$216,459	35.2
Chains	1,130	$541,646	$193,360	30.7
Cranksets	1,757	$904,317	$323,648	28.9
Derailleurs	1,760	$369,258	$98,424	30.5
Forks	979	$92,104	$37,375	45.0
Handlebars	7,232	$3,698,066	$1,468,094	32.0
Headsets	1,283	$160,859	$60,853	41.3
Mountain Frames	20,859	$25,146,722	$9,550,684	35.1
Pedals	7,261	$3,505,525	$1,291,652	35.7
Road Frames	21,770	$24,848,663	$9,413,604	32.7
Saddles	3,807	$2,499,409	$934,813	35.7
Touring Frames	6,124	$3,789,571	$1,258,221	36.9
Wheels	8,362	$6,103,824	$2,442,271	38.3
Scooters	**187,096**	**$126,034,851**	**$44,642,869**	**33.0**
Mountain Scooters	58,728	$41,860,390	$15,915,984	34.6
Road Scooters	97,266	$72,319,534	$24,623,216	32.4
Touring Scooters	31,102	$11,854,927	$4,103,669	32.0
Grand Total	**586,483**	**$389,924,585**	**$141,611,106**	**34.6**

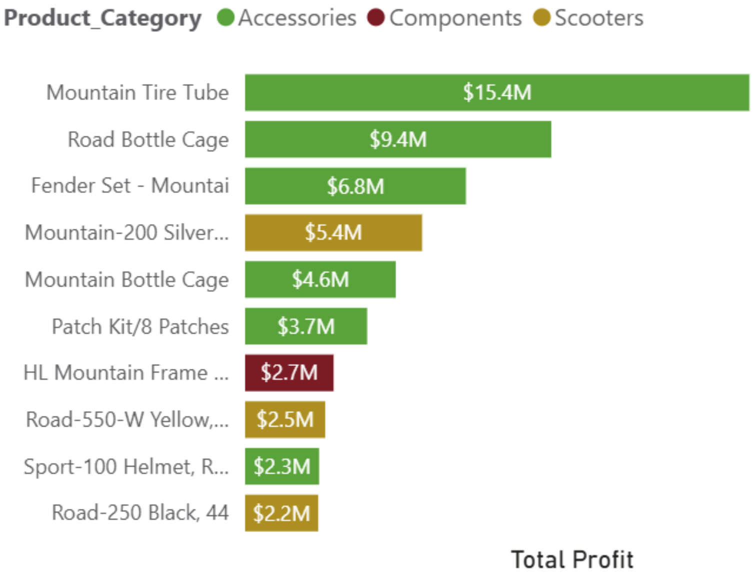
Category Dominance: Accessories lead profitability at $51M, closely followed by Scooters ($45M). Components and Clothing contribute 32% combined – a significant secondary revenue stream.

Figure 7: Gross Profit by Product Category



Product Concentration: Top 10 products are predominantly accessories and scooters, confirming category-level patterns.

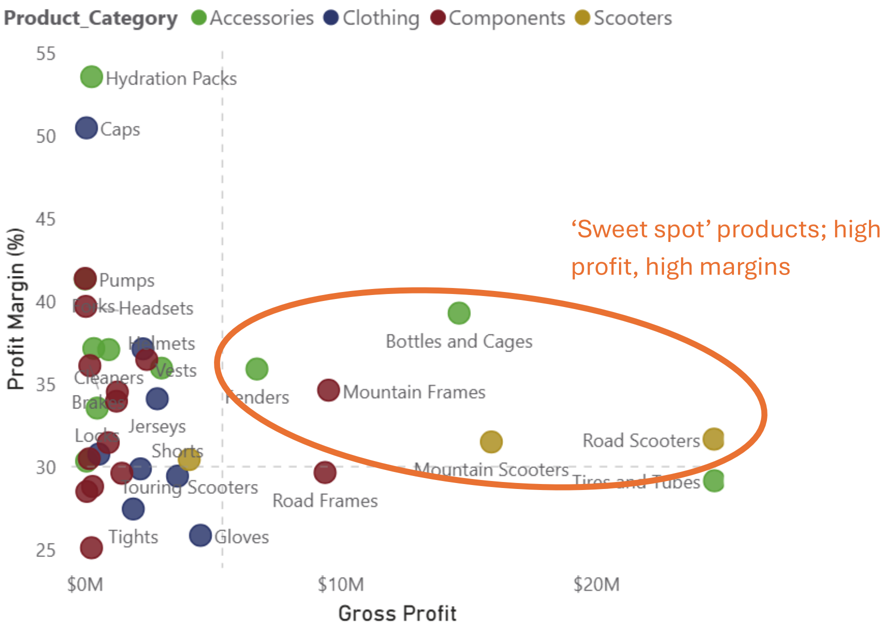
Figure 8: Top 10 Products by Gross Profit



Margin-Volume Trade-offs: High-margin products like Hydration Packs (51.2%) and Caps (50.4%) generate minimal absolute profit due to low volumes (Figure 9). Optimal growth targets are "sweet spot" products combining substantial profit (>$500K) with healthy margins (>30%):

a) Bottles and Cages: $14.7M profit, 37% margin
b) Road Scooters: $24.6M profit, 32% margin
c) Mountain Scooters: $15.9M profit, 35% margin
d) Mountain Frames: $9.6M profit, 35% margin
e) Fenders: $6.8M profit, 34% margin

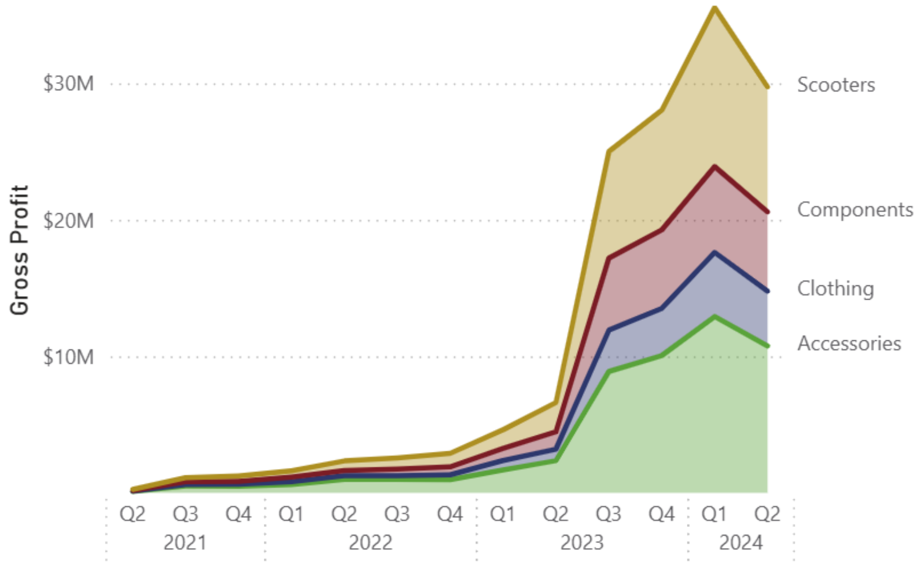
Figure 9: Top 25 Products, Revenue vs Margin



'Sweet spot' products; high profit, high margins

Growth Inflection: Q2-Q3 2023 marked a significant profit surge across all categories, with elevated performance sustaining through Q1 2024 peak. Category profit shares remained stable despite absolute growth.

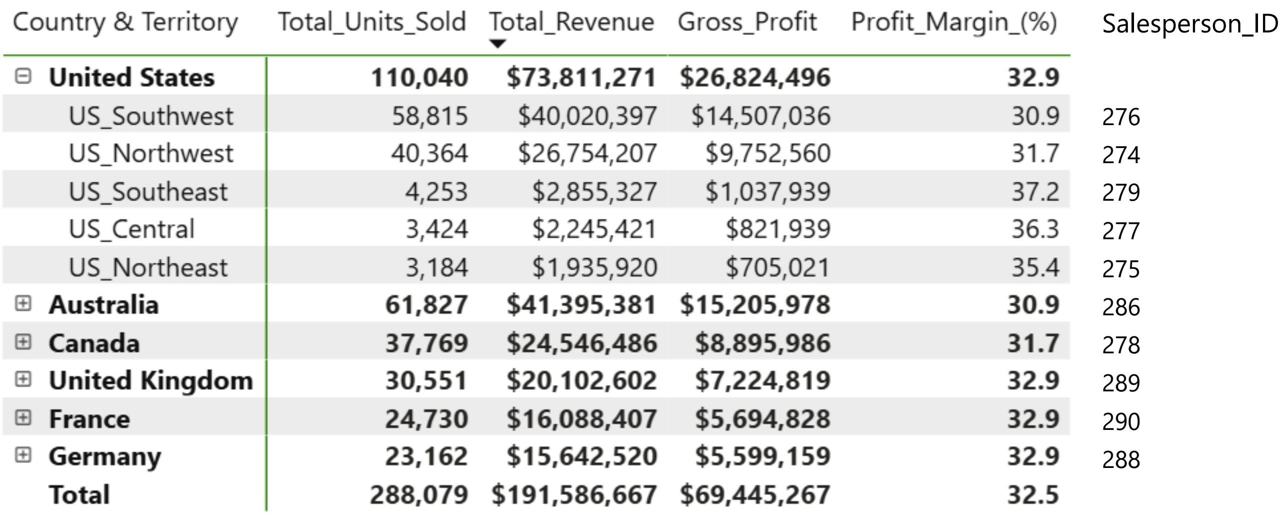
Figure 10: Gross Profit by Product Category (2021-24)



4.3 Most profitable sales territories

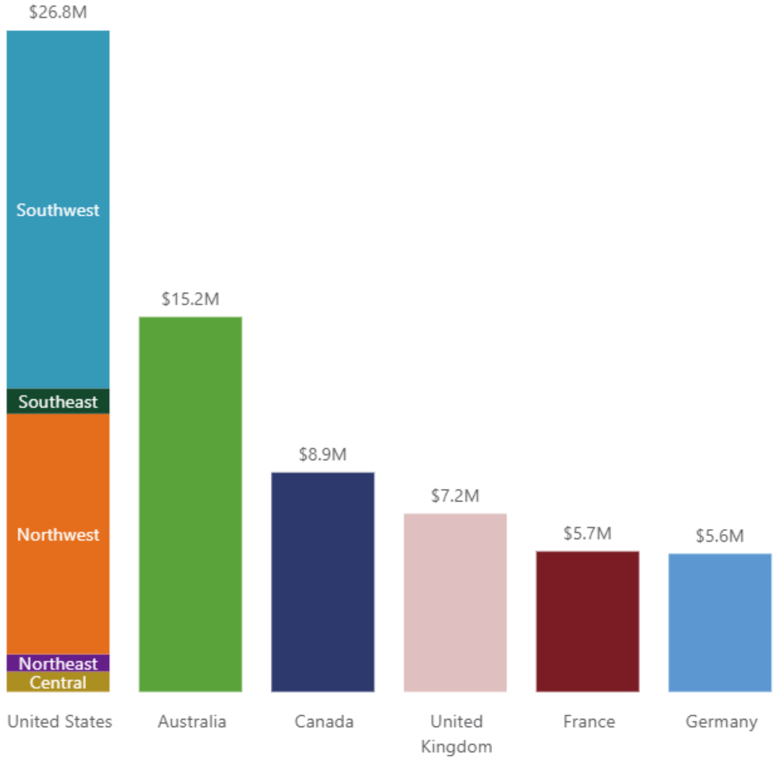
Territory Performance (2021-24): US territories dominate across all metrics (revenue, profit, margins), with the West Coast region leading globally. Australia emerges as the secondary high-value market.

Figure 11: Sales Territories Table



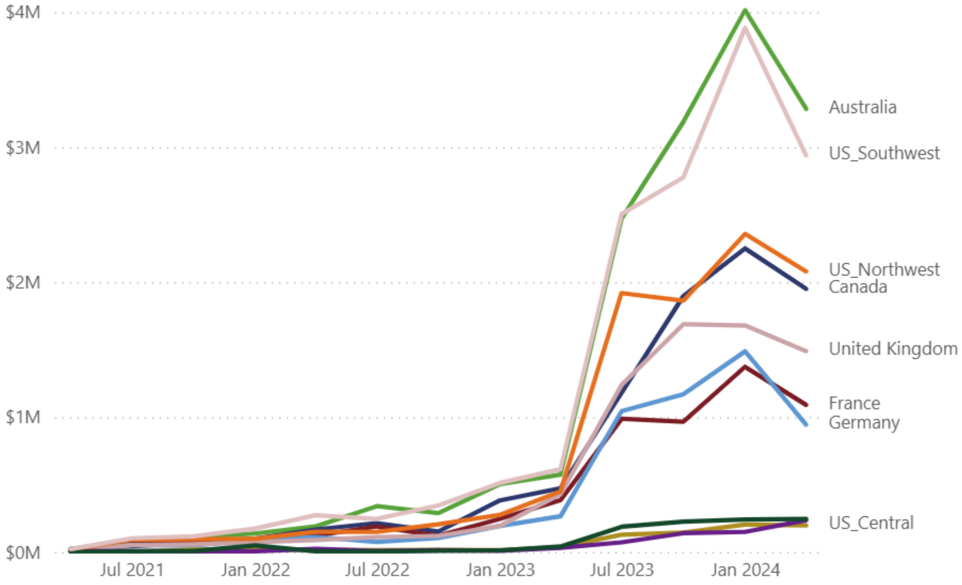
Country & Territory	Total_Units_Sold	Total_Revenue	Gross_Profit	Profit_Margin_(%)	Salesperson_ID
⊟ **United States**	**110,040**	**$73,811,271**	**$26,824,496**	**32.9**	
US_Southwest	58,815	$40,020,397	$14,507,036	30.9	276
US_Northwest	40,364	$26,754,207	$9,752,560	31.7	274
US_Southeast	4,253	$2,855,327	$1,037,939	37.2	279
US_Central	3,424	$2,245,421	$821,939	36.3	277
US_Northeast	3,184	$1,935,920	$705,021	35.4	275
⊞ **Australia**	**61,827**	**$41,395,381**	**$15,205,978**	**30.9**	286
⊞ **Canada**	**37,769**	**$24,546,486**	**$8,895,986**	**31.7**	278
⊞ **United Kingdom**	**30,551**	**$20,102,602**	**$7,224,819**	**32.9**	289
⊞ **France**	**24,730**	**$16,088,407**	**$5,694,828**	**32.9**	290
⊞ **Germany**	**23,162**	**$15,642,520**	**$5,599,159**	**32.9**	288
Total	**288,079**	**$191,586,667**	**$69,445,267**	**32.5**	

Figure 12: Gross Profit by Country & Territory



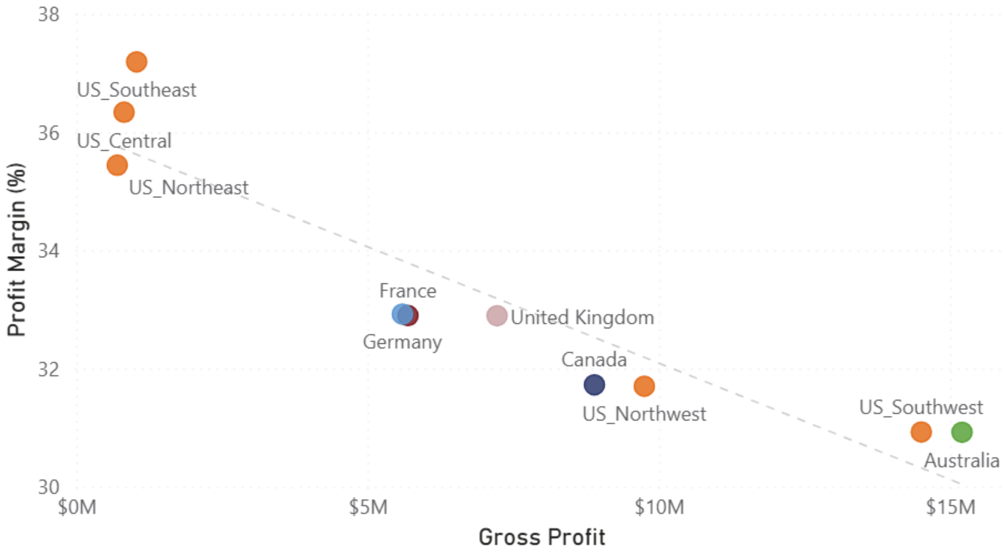
Growth Trajectories: All territories began from comparable profit baselines but diverged significantly, indicating varying market development rates and competitive positioning (Figure 13). On the other hand universal quarter-on-quarter profit drops across all territories from Q1 to Q2 2024. This synchronised decline suggests a company-wide issue rather than regional factors – requires immediate investigation and continued monitoring.

Figure 13: Quarterly Gross Profit (2021-24)



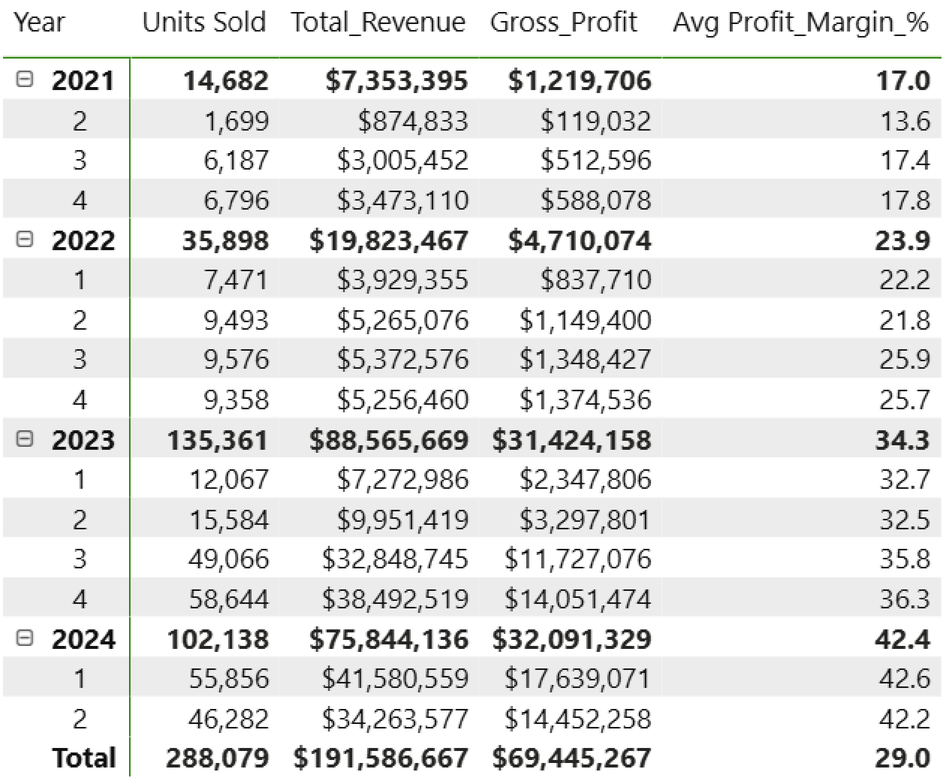
Profit-Margin Trade-off: Strong negative correlation exists between territorial gross profit and profit margins. High-volume territories operate on thinner margins, likely due to competitive pricing pressures in mature markets or bulk discount strategies.

Figure 14: Gross Profit vs Profit Margin (%)



4.4 Most profitable time periods

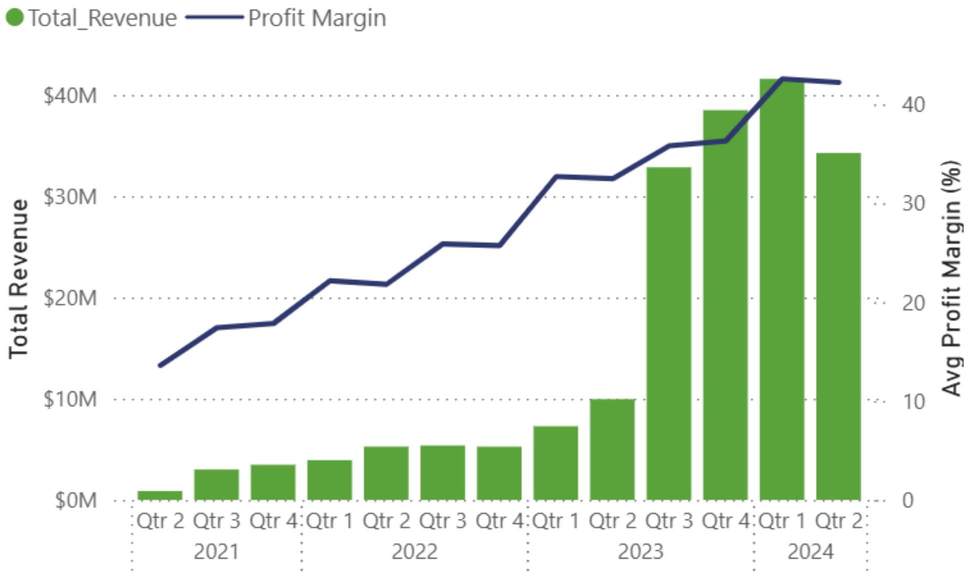
Figure 15: Sales metrics by Year and Quarter



Year	Units Sold	Total_Revenue	Gross_Profit	Avg Profit_Margin_%
⊟ **2021**	**14,682**	**$7,353,395**	**$1,219,706**	**17.0**
2	1,699	$874,833	$119,032	13.6
3	6,187	$3,005,452	$512,596	17.4
4	6,796	$3,473,110	$588,078	17.8
⊟ **2022**	**35,898**	**$19,823,467**	**$4,710,074**	**23.9**
1	7,471	$3,929,355	$837,710	22.2
2	9,493	$5,265,076	$1,149,400	21.8
3	9,576	$5,372,576	$1,348,427	25.9
4	9,358	$5,256,460	$1,374,536	25.7
⊟ **2023**	**135,361**	**$88,565,669**	**$31,424,158**	**34.3**
1	12,067	$7,272,986	$2,347,806	32.7
2	15,584	$9,951,419	$3,297,801	32.5
3	49,066	$32,848,745	$11,727,076	35.8
4	58,644	$38,492,519	$14,051,474	36.3
⊟ **2024**	**102,138**	**$75,844,136**	**$32,091,329**	**42.4**
1	55,856	$41,580,559	$17,639,071	42.6
2	46,282	$34,263,577	$14,452,258	42.2
Total	**288,079**	**$191,586,667**	**$69,445,267**	**29.0**

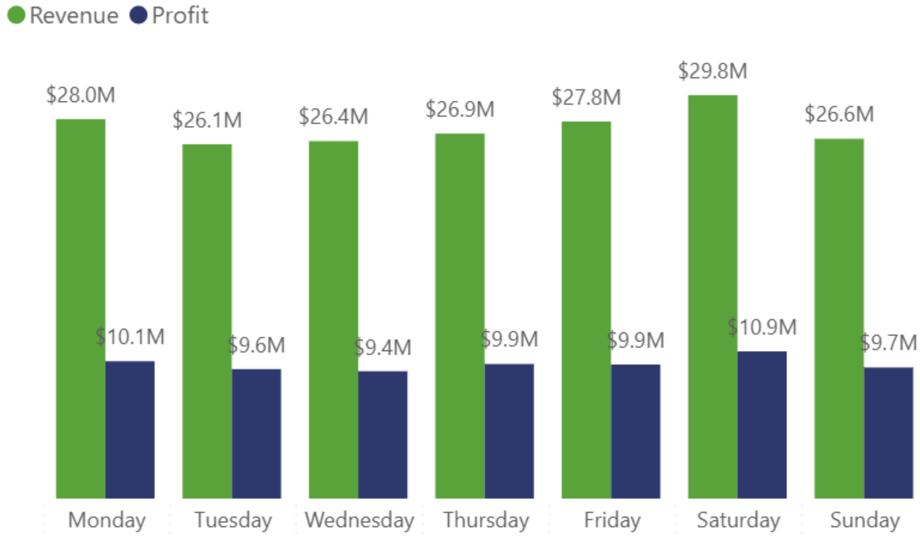
Margin Expansion with Scale: Revenue growth from $875K (Q2-2021) to $34M (Q2-2024) coincided with gradual and significant profit margin improvement from 13.6% to 42.2%. This suggests HH is achieving operational leverage including through fixed costs efficiency.

Figure 16: Quarterly revenue & profit margins (2021-24)



Weekly Sales Cycle: Strong weekend-weekstart performance with Saturday leading and Monday following. Mid-week weakness on Tuesday and Wednesday creates a 13% revenue gap between peak and trough days.

Figure 17: Total Revenue and Profit by Day of Week



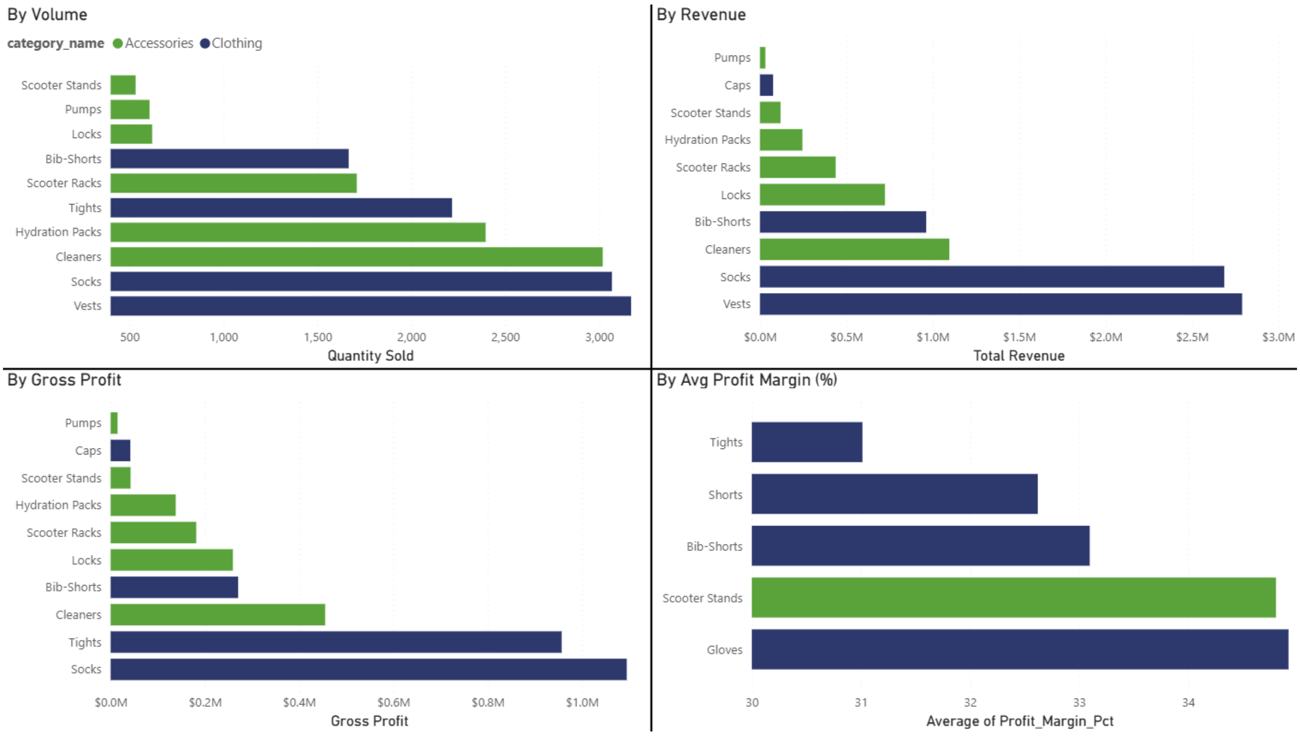
Recommendations:

- Schedule major product launches and campaigns for Friday-Monday window
- Deploy mid-week promotional campaigns to flatten demand curves

4.5 Categories and sub-categories for replacement

Multi-Metric Underperformers: Across Accessories and Clothing categories, **scooter stands**, **caps**, **pumps**, **locks**, **bib-shorts**, and **hydration packs** consistently underperform on volume, revenue, and profit metrics. Lowest-margin items include **tights**, **shorts**, **bib-shorts**, **scooter stands**, and **gloves**.

Figure 18: Lowest Performing Subcategories (Clothing and Accessories)



Profit-Margin Matrix Analysis: Plotting subcategories by gross profit vs margin percentage reveals distinct replacement candidates. Caps and hydration packs, despite low absolute profits ($41K and

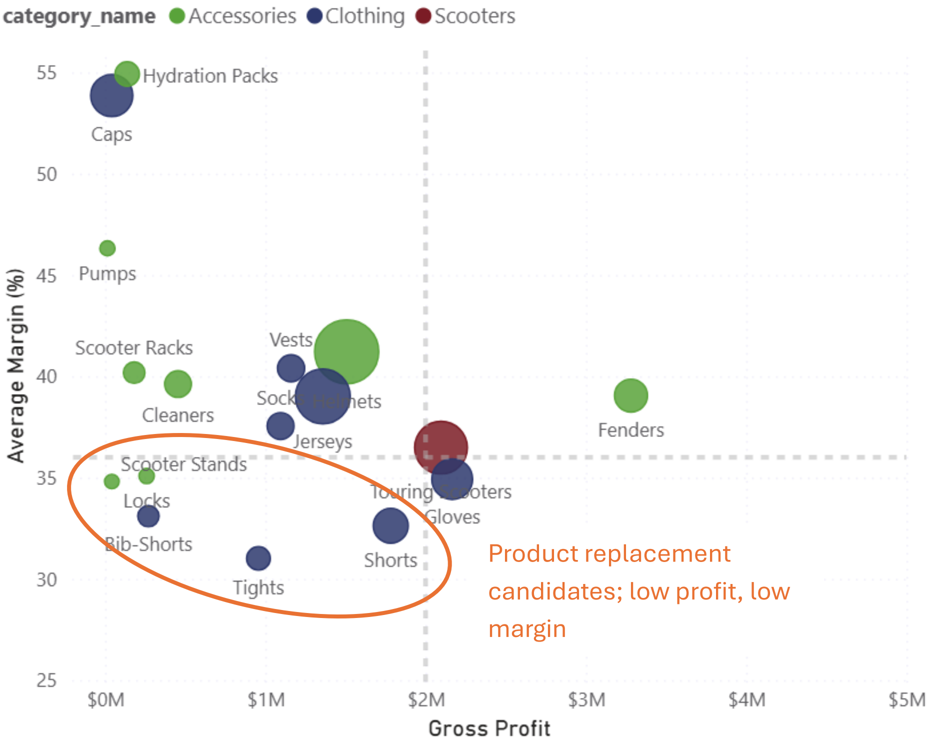
$138K respectively), warrant retention due to exceptional margins (50%+) and high volume turnover potential.

On the other hand subcategories falling below $2M profit and 36% margin thresholds as shown in Figure 19 include:

a) bib-shorts ($270K profit, 33.1% margin)
b) locks ($259K profit, 35% margin)
c) tights ($257K profit, 31% margin)
d) scooter stands ($42K profit, 34.8% margin)

We exclude the Components subcategory from replacement consideration – maintaining comprehensive replacement parts inventory (bottom brackets, cranksets) drives customer retention and cross-selling opportunities.

Figure 19: Least Profitable Subcategories, Gross Profit vs Margin (%)



Product replacement candidates; low profit, low margin

Figure 20: Time series of gross profits from 'clothing' category products

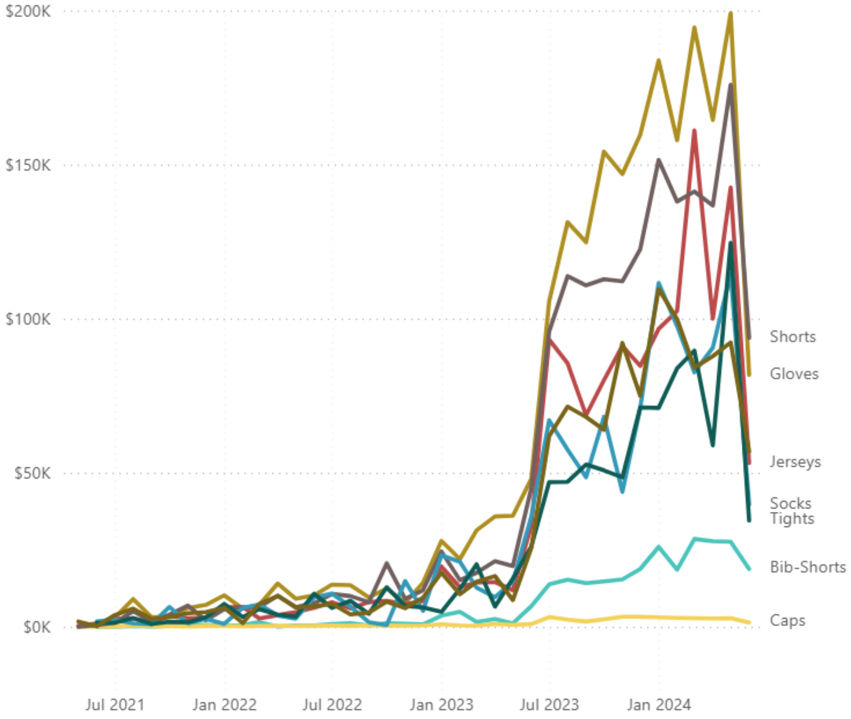
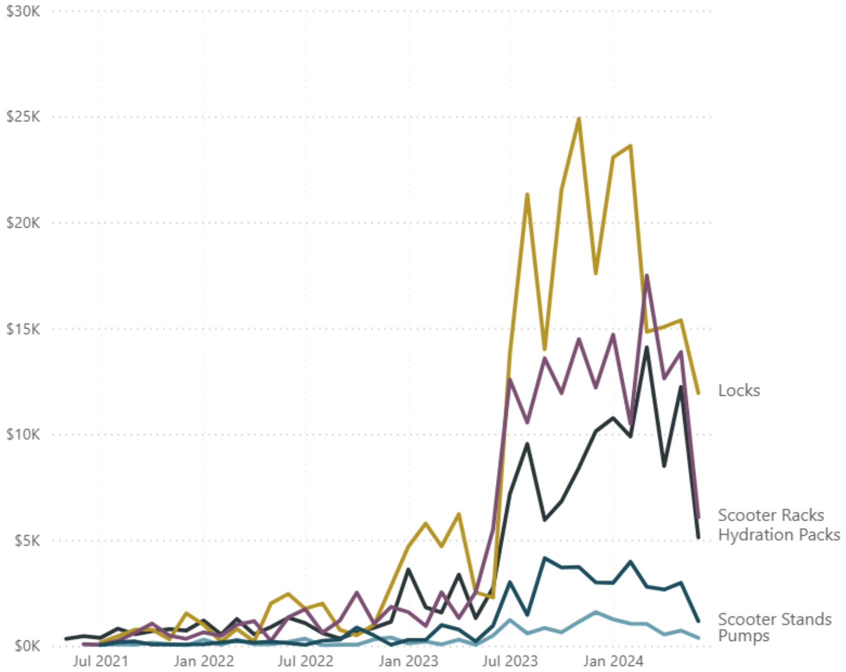


Figure 21: Time series of gross profits from 'accessories' category products



4.6 Consolidated Recommendations

1. **Customer Strategy:**
   a) Develop targeted B2B sales programme leveraging Efficient Cycling's 44.6% margin as proof of concept for store partnerships
   b) Investigate discount structures for high-revenue, low-margin customers (Calvin Xu, Brando Smith) to improve profitability without sacrificing volume

c) Implement customer retention initiatives in North American and European markets to address Q2 2024 revenue decline

2. **Product Portfolio Optimisation:**
   a) Prioritise inventory and marketing investment in "sweet spot" products combining high profit and healthy margins: Bottles and Cages ($14.7M, 37%), Road Scooters ($24.6M, 32%), Mountain Scooters ($15.9M, 35%), Mountain Frames ($9.6M, 35%), and Fenders ($6.8M, 34%)
   b) Replace bib-shorts with expanded shorts range based on persistent underperformance versus top clothing performer; Replace scooter stands with enhanced scooter racks offering given consistent market preference
   c) Retain caps and hydration packs despite low absolute profits due to exceptional margins (50%+) and turnover potential
3. **Territorial Growth:**
   a) Replicate West Coast US market development strategies in underperforming territories to accelerate growth trajectories
   b) Review pricing strategies in high-volume territories to balance margin compression (negative correlation observed) against market share gains
4. **Promotional Timing:**
   a) Launch major product campaigns and releases during spring season (March-May peak: $40M revenue)
   b) Schedule key promotional activities for Friday-Monday window to capitalise on weekend sales peaks
   c) Deploy mid-week promotional campaigns (Tuesday-Wednesday) to flatten 13% demand gap between peak and trough days
5. **Urgent Investigation:** Prioritise root cause analysis of universal Q1-Q2 2024 profit decline across all territories – synchronised drop suggests company-wide pricing, cost, or operational issue requiring immediate remediation

5. Conclusion

This project successfully implemented a comprehensive dimensional data warehouse solution for HH, transforming fragmented operational data into an integrated analytical platform capable of supporting strategic decision-making.

The star schema design enables efficient analysis of customer profitability, product performance, territorial results, and time-based trends. The ETL process using SSIS extracted data from HH’s operational MySQL database, applying necessary transformations and loading 121,000 transactional records spanning 2021-2024 into a SQL Server data warehouse.

The analytical insights derived from this data warehouse revealed several critical findings:

- B2C dominance with 93% revenue share suggests untapped B2B growth potential, particularly given Efficient Cycling's exceptional 44.6% margin;
- The identification of "sweet spot" products (Bottles and Cages, Road/Mountain Scooters, and Mountain Frames) combining substantial profit with healthy margins provides clear direction for inventory and marketing prioritisation;
- The universal Q1-Q2 2024 profit decline across all territories signals a company-wide issue requiring immediate management attention;
- Spring peak and weekend sales patterns offer tactical opportunities for campaign timing optimisation.

The dimensional model's design directly addresses HH's five key business questions whilst maintaining flexibility for future analytical requirements. By implementing best practices in dimensional modelling, this solution provides a robust foundation for ongoing business intelligence activities supporting Handlebar Haven's continued growth and market expansion.

6. Statement on AI Usage

Artificial intelligence tools (ChatGPT) were used selectively to support the development of this report. AI assistance was used to:

- Troubleshoot specific SSIS technical issues when encountered during implementation
- Validate dimensional modelling approaches and SQL query logic
- Review and refine report structure and documentation clarity

All design decisions, ETL implementation, data analysis, business insights, and final documentation were completed independently by the author. AI outputs were treated as supplementary guidance and critically evaluated for accuracy and alignment with course objectives.

7. References

Elmasri, R. and Navathe, S. (2017) *Fundamentals of database systems*. 7th edn. Boston: Pearson.

Kimball, R. and Ross, M. (2013) *The data warehouse toolkit: the definitive guide to dimensional modeling*. 3rd edn. Indianapolis: Wiley.

Appendix A: Technical Environment and Lessons Learned

A.1 Technical Architecture

The data warehouse implementation used the following technical stack:

- **Source System**: MySQL 8.0 (handlebarhaven operational database)
- **ETL Tool**: SQL Server Integration Services (SSIS) in Visual Studio 2022
- **Target System**: SQL Server 2019 (HandlebarHavenDW data warehouse)
- **Analysis Tools**: SQL Server Management Studio (SSMS), Power BI Desktop
- **Connections**: ADO.NET Source for MySQL extraction, OLE DB Destination for SQL Server loading

A.2 Implementation Challenges and Resolutions

Challenge 1: Fact Table Duplication (819,804 vs 121,000 Expected Rows)

Problem: The initial FactSales implementation produced 819,804 rows, nearly seven times the expected 121,000 transactions.

Root Cause: The source query joined to products_history, which is a Slowly Changing Dimension Type 2 table. This table contains multiple historical price versions per Product_ID with different End_Dates. When joined without filtering, this created a product with approximately 26 historical price versions matching each transaction record.

Resolution: I modified the source query to join directly to the products table instead of products_history, retrieving only current pricing. This eliminated the duplication while maintaining accurate cost and margin calculations.

Challenge 2: Customer Dimension Lookup Failures (62 Missing Customers)

Problem: FactSales loading failed with 62 lookup errors. These customers appeared in sales_orders but were missing from DimCustomer.

Root Cause: The DimCustomer source query used INNER JOINs through the customer_address, state, and country_region_codes hierarchy. Customers with incomplete address data were excluded entirely, even though they had valid transaction history in the system.

Resolution: I converted the INNER JOINs to LEFT JOINs and used COALESCE functions to assign 'Unknown' values where geography attributes were missing. This increased DimCustomer from 19,758 to 19,820 rows, capturing all customers while flagging incomplete data that the operational team could potentially remediate.

Challenge 3: Data Type Mismatches in Lookup Transformations

Problem: TimeKey lookup transformations failed despite TimeKey existing in both the source and dimension tables.

Root Cause: MySQL's CAST(DATE_FORMAT(Order_Date, '%Y%m%d') AS UNSIGNED) function produces a BIGINT (8-byte unsigned integer), while SQL Server's DimTime.TimeKey was defined as INT (4-byte signed integer). SSIS lookup transformations require exact data type matching between source and target fields.

Resolution: I added a Data Conversion transformation component before the TimeKey lookup, explicitly converting the BIGINT to DT_I4 (four-byte signed integer) to match the SQL Server INT type.

Challenge 4: Inverted Seasonal Logic in DimTime

Problem: Initial seasonal assignments showed January in the USA as "Summer" instead of "Winter". The Season_North and Season_South columns had been swapped.

Root Cause: The SSIS Derived Column transformation formulas for Season_North and Season_South were accidentally inverted during implementation.

Resolution: I applied a post-load correction using SQL UPDATE statements with temporary columns to swap the values without triggering MySQL's safe update mode restrictions. A verification query comparing Australia (territory_code='AU') against other territories confirmed the proper hemisphere assignment.

A.3 Technical Lessons Learned

SSIS Best Practices:

- Use OLE DB Destination (not ADO.NET Destination) for SQL Server targets. OLE DB provides better bulk loading performance and proper handling of square bracket identifiers.
- Create target tables manually in SSMS before configuring SSIS destinations. The SSIS "New..." button generates DDL immediately rather than during package execution, which complicates iterative development.
- Use "SQL command" mode (not "Table or view" mode) for MySQL sources. This avoids SSIS auto-generation of problematic double-quote syntax.
- Configure Lookup transformations with "Full cache" mode and "Fail component" error handling during development. This enables rapid identification of dimension/fact referential integrity issues.

Data Quality Strategies:

- Apply DISTINCT in dimension source queries to prevent duplicate surrogate keys.
- Use LEFT JOINs with COALESCE for incomplete hierarchical data. This captures all relevant records while flagging data quality issues for potential resolution.
- CAST dates explicitly to DATE type to strip timestamps that can cause lookup mismatches.
- TRUNCATE target tables before reloading during iterative development to prevent accumulation of duplicate records across test runs.

Cross-Platform Data Type Management:

- MySQL UNSIGNED integers do not directly map to SQL Server signed integers and require explicit Data Conversion transformations.
- SSIS string literals default to Unicode (DT_WSTR/NVARCHAR). Dimension columns should use NVARCHAR unless specifically requiring VARCHAR for compatibility reasons.
- Currency and financial calculations require DECIMAL(19,4) format, never FLOAT or REAL, due to precision loss that affects margin calculations.

Source Table Selection:

- Use current operational tables (products) rather than historical audit tables (products_history) for fact table extraction, unless specifically implementing Slowly Changing Dimension logic.
- Validate expected row counts before and after ETL. Significant discrepancies indicate join issues or data quality problems requiring investigation.

Appendix B: Data Dictionary

FactSales			
Attribute Name	**Description**	**Source Table/Field**	**Transformation**
SalesKey	Primary surrogate key for fact table	N/A	Auto-generated IDENTITY(1,1)
CustomerKey	Foreign key to DimCustomer	sales_orders/Customer_ID	Lookup transformation using Customer_ID
ProductKey	Foreign key to DimProduct	products/Product_ID (via sales_order_details/Product_Number)	Lookup transformation using Product_ID
SupplierKey	Foreign key to DimSupplier	products/Supplier_ID	Lookup transformation using Supplier_ID
TerritoryKey	Foreign key to DimTerritory	sales_territory/Territory_ID (via sales_orders)	Lookup transformation using Territory_ID
TimeKey	Foreign key to DimTime	sales_orders/Order_Date	Convert to YYYYMMDD integer format, lookup to DimTime
Order_ID	Business key for order identification	sales_order_details/Sales_Order_Number	Direct mapping
Order_Line_ID	Line item identifier within order	sales_order_details/Order_Line_ID	Direct mapping
Unit_Price	Actual selling price per unit	sales_order_details/Unit_Price	Direct mapping
Unit_Cost	Standard cost per unit	products/Standard_Cost	Direct mapping
Unit_List_Price	Manufacturer suggested retail price	products/List_Price	Direct mapping
Units_Sold	Quantity of units sold	sales_order_details/Order_Qty	Direct mapping
Line_Revenue	Total revenue for order line	sales_order_details/Line_Total	Direct mapping
Line_Cost	Total cost for order line	Calculated	Standard_Cost × Order_Qty
Line_Gross_Profit	Gross profit for order line	Calculated	(Unit_Price - Standard_Cost) × Order_Qty
DimCustomer			
CustomerKey	Surrogate primary key	Auto-generated	IDENTITY(1,1)
Customer_ID	Business key for customer	customers.Customer_ID	
Customer_Name	Full customer name	customers.First_Name, customers.Last_Name	Concatenated with space
Segment	Customer type classification	customers.Person_Type	

Country_Region_Code	ISO country code	state.Country_Region_Code	LEFT JOIN via customer_address and state
Country_Name	Full country name	country_region_codes.Country_Name	LEFT JOIN via state and country_region_codes
DimProduct			
ProductKey	Surrogate primary key	Auto-generated	IDENTITY(1,1)
Product_ID	Business key for product	products.Product_ID	
Product_Name	Product name	products.Product_Name	
Category_ID	Product category identifier	product_categories.Product_Cat_ID	INNER JOIN via product_sub_category
Category_Name	Product category name	product_categories.Cat_Name	INNER JOIN via product_sub_category
Subcategory_ID	Product subcategory identifier	product_sub_category.Product_Subcategory_ID	INNER JOIN from products
Subcategory_Name	Product subcategory name	product_sub_category.Product_Name	INNER JOIN from products
Standard_Cost	Manufacturing cost per unit	products.Standard_Cost	
List_Price	Recommended retail price	products.List_Price	
DimSupplier			
SupplierKey	Surrogate primary key	Auto-generated	IDENTITY(1,1)
Supplier_ID	Business key for supplier	suppliers.Supplier_ID	
Supplier_Name	Supplier company name	suppliers.Supplier_Name	
Address	Supplier street address	suppliers.Address	
City	Supplier city location	suppliers.City	
Country	Supplier country	suppliers.Country	
DimTerritory			
TerritoryKey	Surrogate primary key	Auto-generated	IDENTITY(1,1)
Territory_ID	Business key for sales territory	sales_territory.Territory_ID	
Territory_Name	Sales territory name	sales_territory.Territory_Name	
Country	Territory country	sales_territory.Country_Region_Code	Via country_region_codes lookup
Salesperson_ID	Assigned salesperson	sales_territory.Territory_ID	Via sales_persons lookup
DimTime			
TimeKey	Date in YYYYMMDD format (PK)	sales_orders.Order_Date	CAST(FORMAT(Order_Date, 'yyyyMMdd') AS INT)
Day	Day of month (1-31)	sales_orders.Order_Date	DAY(Order_Date)

Month	Month number (1-12)	sales_orders.Order_Date	MONTH(Order_Date)
Year	Four-digit year	sales_orders.Order_Date	YEAR(Order_Date)
Quarter	Calendar quarter (1-4)	sales_orders.Order_Date	Derived from Month ((Month-1)/3)+1
DayOfWeek_Num	Day of week number (1-7)	sales_orders.Order_Date	DATEPART(WEEKDAY, Order_Date)
DayOfWeek_Name	Day name (Monday-Sunday)	sales_orders.Order_Date	DATENAME(WEEKDAY, Order_Date)
Season_Northern	Season for Northern Hemisphere	sales_orders.Order_Date	Derived: Dec-Feb=Winter, Mar-May=Spring, Jun-Aug=Summer, Sep-Nov=Autumn
Season_Southern	Season for Southern Hemisphere	sales_orders.Order_Date	Derived: Dec-Feb=Summer, Mar-May=Autumn, Jun-Aug=Winter, Sep-Nov=Spring

Appendix C: SSIS Transformations

Refer to the ZIP file submitted with this report.

Appendix D: SQL queries deployed in this project

D.1 Source database extraction queries

I used these queries at the EXTRACT stage to source and perform basic transformations on attributes from the HH database in MySQL, ready for further transformations in SSIS.

```
-- 1. DIMCUSTOMER SOURCE QUERY
SELECT
    c.Customer_ID,
    c.First_Name,
    c.Last_Name,
    c.Store_ID,
    COALESCE(c.Store_Name, 'Unknown') AS Store_Name,
    c.Person_Type AS Segment,
    COALESCE(s.Country_Region_Code, 'Unknown') AS Country_Region_Code,
    COALESCE(crc.Country_Name, 'Unknown') AS Country_Name
FROM
    customers c
LEFT JOIN
    customer_address ca
    ON c.Address_ID = ca.Address_ID
LEFT JOIN
    state s
    ON ca.State_Province_ID = s.State_ProvinceID
LEFT JOIN
    country_region_codes crc
    ON s.Country_Region_Code = crc.Country_Code
```

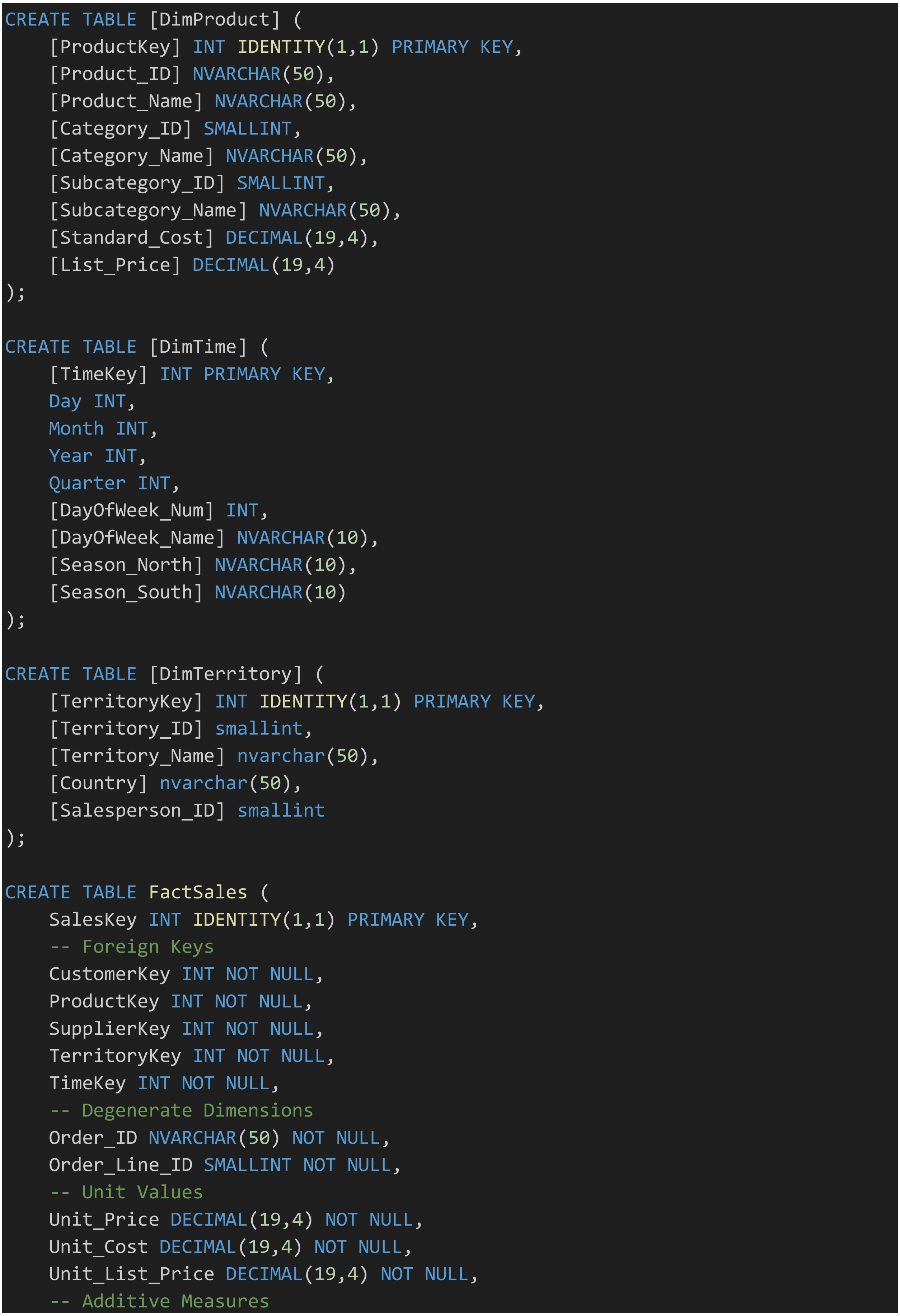
```sql
ORDER BY c.Customer_ID ASC;

-- 2. DIMSUPPLIER SOURCE QUERY
SELECT
    Supplier_ID,
    Supplier_Name,
    Address,
    City,
    Country
FROM suppliers

-- 3. DIMPRODUCT SOURCE QUERY
SELECT
    p.Product_ID,
    p.Supplier_ID,
    p.Product_Name,
    pc.product_cat_ID AS 'Category_ID',
    pc.cat_name AS 'Category_name',
    psc.product_subcategory_ID AS 'Subcategory_ID',
    psc.product_name AS 'Subcategory_name',
    p.Standard_Cost,
    p.List_Price
FROM
    products p
INNER JOIN  product_sub_category psc
    ON p.product_subcategory_ID = psc.product_subcategory_ID
INNER JOIN  product_categories pc
    ON psc.product_category_ID = pc.product_cat_ID;

-- 4. DIMTIME SOURCE QUERY
SELECT DISTINCT CAST(Order_Date AS DATE) AS Date
FROM sales_orders
WHERE Order_Date IS NOT NULL

-- 5. DIMTERRITORY SOURCE QUERY
SELECT
    st.Territory_ID,
    st.Territory_Name,
    crc.Country_Name AS Country,
    sp.Sales_Person_ID AS Salesperson_ID
FROM sales_territory st
INNER JOIN country_region_codes crc
    ON st.Country_Region_Code = crc.Country_Code
LEFT JOIN sales_persons sp
    ON st.Territory_ID = sp.Territory_ID
ORDER BY Territory_ID;

-- 6. FACTSALES SOURCE QUERY
```

```
SELECT
    so.Customer_ID,
    p.Product_ID,
    p.Supplier_ID,
    st.Territory_ID,
    CAST(DATE_FORMAT(so.Order_Date, '%Y%m%d') AS UNSIGNED) AS
Order_Date_Updated,
    sod.Sales_Order_Number AS Order_ID,
    sod.Order_Line_ID,
    sod.Unit_Price,
    p.Standard_Cost AS Unit_Cost,
    p.List_Price AS Unit_List_Price,
    sod.Order_Qty AS Units_Sold,
    (p.Standard_Cost * sod.Order_Qty) AS Line_Cost,
    sod.Line_Total AS Line_Revenue,
    (sod.Unit_Price - p.Standard_Cost) * sod.Order_Qty AS Line_Gross_Profit
FROM
    sales_order_details sod
INNER JOIN products p
    ON sod.Product_Number = p.Product_ID
INNER JOIN sales_orders so
    ON sod.Sales_Order_Number = so.Sales_Order_Number
INNER JOIN sales_territory st
    ON st.Territory_ID = so.Territory_ID;
```

D.2 Schema DDL commands

I used these commands to create the schema and set the attribute domains for the destination data warehouse.

```
CREATE TABLE [DimCustomer] (
    [CustomerKey] INT IDENTITY(1,1) PRIMARY KEY,
    [Customer_ID] SMALLINT,
    [Customer_Name] NVARCHAR(100),
    [Store_ID] SMALLINT,
    [Store_Name] NVARCHAR(100),
    [Segment] NVARCHAR(50),
    [Country_Region_Code] NVARCHAR(50),
    [Country_Name] NVARCHAR(50)
);

CREATE TABLE [DimSupplier] (
    [SupplierKey] INT IDENTITY(1,1) PRIMARY KEY,
    [Supplier_ID] INT NOT NULL,
    [Supplier_Name] NVARCHAR(100),
    [Address] NVARCHAR(200),
    [City] NVARCHAR(100),
    [Country] NVARCHAR(100)
);
```



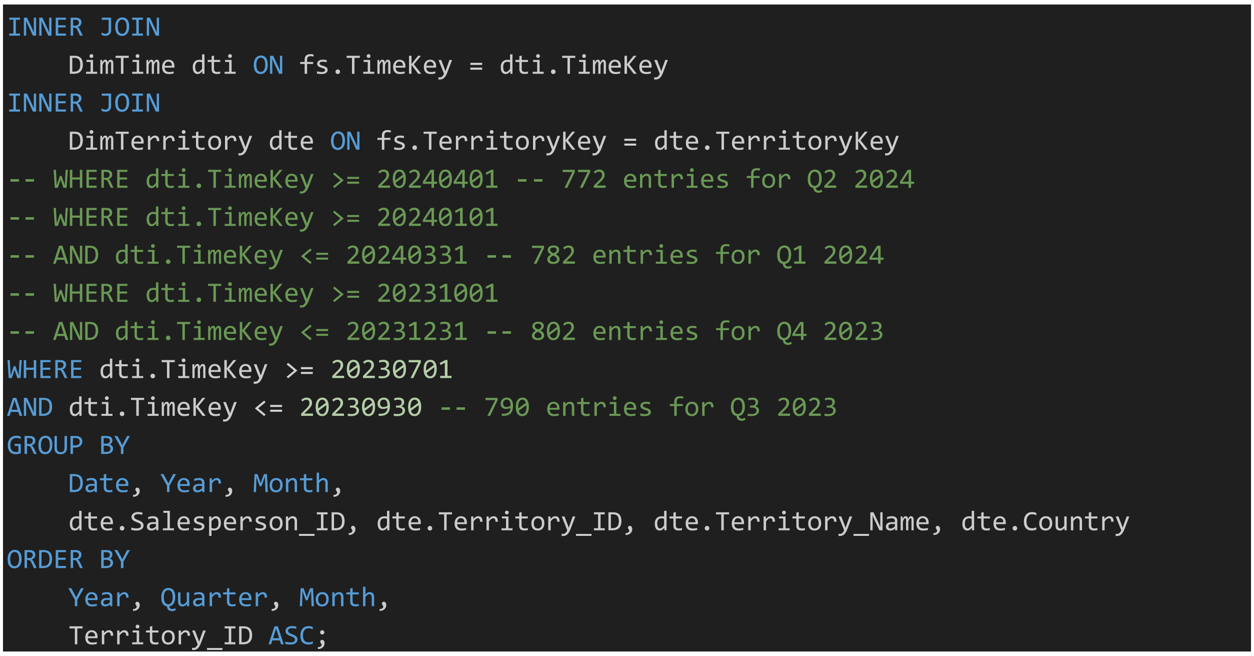
```sql
CREATE TABLE [DimProduct] (
    [ProductKey] INT IDENTITY(1,1) PRIMARY KEY,
    [Product_ID] NVARCHAR(50),
    [Product_Name] NVARCHAR(50),
    [Category_ID] SMALLINT,
    [Category_Name] NVARCHAR(50),
    [Subcategory_ID] SMALLINT,
    [Subcategory_Name] NVARCHAR(50),
    [Standard_Cost] DECIMAL(19,4),
    [List_Price] DECIMAL(19,4)
);

CREATE TABLE [DimTime] (
    [TimeKey] INT PRIMARY KEY,
    Day INT,
    Month INT,
    Year INT,
    Quarter INT,
    [DayOfWeek_Num] INT,
    [DayOfWeek_Name] NVARCHAR(10),
    [Season_North] NVARCHAR(10),
    [Season_South] NVARCHAR(10)
);

CREATE TABLE [DimTerritory] (
    [TerritoryKey] INT IDENTITY(1,1) PRIMARY KEY,
    [Territory_ID] smallint,
    [Territory_Name] nvarchar(50),
    [Country] nvarchar(50),
    [Salesperson_ID] smallint
);

CREATE TABLE FactSales (
    SalesKey INT IDENTITY(1,1) PRIMARY KEY,
    -- Foreign Keys
    CustomerKey INT NOT NULL,
    ProductKey INT NOT NULL,
    SupplierKey INT NOT NULL,
    TerritoryKey INT NOT NULL,
    TimeKey INT NOT NULL,
    -- Degenerate Dimensions
    Order_ID NVARCHAR(50) NOT NULL,
    Order_Line_ID SMALLINT NOT NULL,
    -- Unit Values
    Unit_Price DECIMAL(19,4) NOT NULL,
    Unit_Cost DECIMAL(19,4) NOT NULL,
    Unit_List_Price DECIMAL(19,4) NOT NULL,
    -- Additive Measures
    Units_Sold SMALLINT NOT NULL,
    Line_Revenue DECIMAL(19,4) NOT NULL,
    Line_Cost DECIMAL(19,4) NOT NULL,
    Line_Gross_Profit DECIMAL(19,4) NOT NULL
);
```

D.3 SQL Queries for business analysis

I used these queries to generate tables with the relevant attributes and at the appropriate grain to answer each of the business questions posed by the client.

```
SELECT * FROM handlebarhaven_dw.factsales;

-- QUESTION 1 KEY CUSTOMERS: ALL TIME STATS
SELECT
    dc.Customer_ID,
    dc.Customer_Name,
    dc.Store_ID,
    dc.Store_Name,
    dc.Segment,
    dc.Country_Name,
    SUM(units_sold) AS Total_Units_Sold,
    ROUND(SUM(fs.Line_Revenue), 2) AS Total_Revenue,
    ROUND(SUM(fs.Line_Cost), 2) AS Total_Cost,
    ROUND(SUM(fs.Line_Gross_Profit), 2) AS Gross_Profit,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Revenue)) * 100, 1) AS
Profit_Margin_Pct,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Cost)) * 100, 1) AS
Markup_Pct
FROM
    factsales fs
INNER JOIN
    dimcustomer dc
    ON fs.CustomerKey = dc.CustomerKey
GROUP BY
    Customer_ID, Customer_Name, Store_ID, Store_Name, Segment, Country_Name
ORDER BY
    Gross_Profit DESC;

-- QUESTION 1 KEY CUSTOMERS: STATS BY YEAR, MONTH, SEASON, QUARTER
SELECT
    dti.TimeKey AS Date,
    dti.Year,
    CASE
        WHEN dc.Country_Region_Code = 'AU' THEN dti.Season_South
        ELSE dti.Season_North
    END AS Season,
    dti.Quarter,
    dti.Month,
    dc.Customer_ID,
    dc.Customer_Name,
    dc.Store_ID,
    dc.Store_Name,
    dc.Segment,
    dc.Country_Name,
    SUM(fs.Units_Sold) AS Total_Units_Sold,
    ROUND(SUM(fs.Line_Revenue), 2) AS Total_Revenue,
    ROUND(SUM(fs.Line_Cost), 2) AS Total_Cost,
    ROUND(SUM(fs.Line_Gross_Profit), 2) AS Gross_Profit,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Revenue)) * 100, 1) AS
Profit_Margin_Pct,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Cost)) * 100, 1) AS
Markup_Pct
FROM
    FactSales fs
INNER JOIN
    DimCustomer dc ON fs.CustomerKey = dc.CustomerKey
INNER JOIN
    DimTime dti ON fs.TimeKey = dti.TimeKey
GROUP BY
    Date, Year, Season, Quarter, Month,
    Customer_ID, Customer_Name, Store_ID, Store_Name, Segment, Country_Name
ORDER BY
    Year, Quarter, Month, Gross_Profit DESC;

    -- VALIDATION QUERY
    SELECT
    dti.Month,
    CASE
        WHEN dc.Country_Region_Code = 'AU' THEN dti.Season_South
        ELSE dti.Season_North
    END AS Season,
    dc.Country_Region_Code,
    SUM(fs.Line_Revenue) AS Total_Revenue
FROM FactSales fs
INNER JOIN DimCustomer dc ON fs.CustomerKey = dc.CustomerKey
INNER JOIN DimTime dti ON fs.TimeKey = dti.TimeKey
GROUP BY Month, Season, Country_Region_Code
ORDER BY Month;

-- QUESTION 2 & 5 PRODUCT & CATEGORY PROFITABILITY
SELECT
    dti.TimeKey AS Date,
    dti.Year,
    CASE
        WHEN dte.territory_name = 'Australia' THEN dti.season_south
        ELSE dti.season_north
    END AS Season,
    dti.Quarter,
    dti.Month,
    dp.product_name,
    dp.category_name,
    dp.subcategory_name,
    SUM(units_sold) AS Total_Units_Sold,
    ROUND(SUM(fs.Line_Revenue), 2) AS Total_Revenue,
    ROUND(SUM(fs.Line_Gross_Profit), 2) AS Gross_Profit,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Revenue)) * 100, 1) AS
Profit_Margin_Pct
FROM
    factsales fs
INNER JOIN
    dimproduct dp
    ON fs.productkey = dp.productkey
INNER JOIN
    dimtime dti
    ON fs.timekey = dti.timekey
INNER JOIN
    dimterritory dte
    ON fs.territorykey = dte.territorykey
GROUP BY
    Season, Quarter, Month, Year, Date,
    dp.product_name, dp.category_name, dp.subcategory_name
ORDER BY
    Gross_Profit;

-- Q2 & Q5, ALL TIME (remove temporal attributes)
SELECT
    dp.product_name,
    dp.category_name,
    dp.subcategory_name,
    SUM(units_sold) AS Total_Units_Sold,
    ROUND(SUM(fs.Line_Revenue), 2) AS Total_Revenue,
    ROUND(SUM(fs.Line_Gross_Profit), 2) AS Gross_Profit,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Revenue)) * 100, 1) AS
Profit_Margin_Pct
FROM
    factsales fs
INNER JOIN
    dimproduct dp
    ON fs.productkey = dp.productkey
INNER JOIN
    dimterritory dte
    ON fs.territorykey = dte.territorykey
GROUP BY
    dp.product_name, dp.category_name, dp.subcategory_name
```

```
ORDER BY
    Gross_Profit;

SELECT * FROM handlebarhaven_dw.dimterritory;

-- ~~~QUESTION 3 SALES TERRITORY PROFITABILITY~~~
SELECT
--     dti.TimeKey AS Date,
--     dti.Year,
--     dti.Month,
    dte.Territory_ID,
    dte.Territory_Name,
    dte.Country,
    dte.Salesperson_ID,
    SUM(fs.Units_Sold) AS Total_Units_Sold,
    ROUND(SUM(fs.Line_Revenue), 2) AS Total_Revenue,
    ROUND(SUM(fs.Line_Gross_Profit), 2) AS Gross_Profit,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Revenue)) * 100, 1) AS
Profit_Margin_Pct
FROM
    FactSales fs
INNER JOIN
    DimTime dti ON fs.TimeKey = dti.TimeKey
INNER JOIN
    DimTerritory dte ON fs.TerritoryKey = dte.TerritoryKey
GROUP BY
--     Date, Year, Month,
    dte.Salesperson_ID, dte.Territory_ID, dte.Territory_Name, dte.Country
ORDER BY
--     Year, Quarter, Month,
    Territory_ID ASC;

    -- VERYFIYING THAT 2024-Q2 HAS NORMAL NUMBER OF SALES ENTRIES..
SELECT
    dti.TimeKey AS Date,
    dti.Year,
    dti.Month,
    dte.Territory_ID,
    dte.Territory_Name,
    dte.Country,
    dte.Salesperson_ID,
    SUM(fs.Units_Sold) AS Total_Units_Sold,
    ROUND(SUM(fs.Line_Revenue), 2) AS Total_Revenue,
    ROUND(SUM(fs.Line_Gross_Profit), 2) AS Gross_Profit,
    ROUND((SUM(fs.Line_Gross_Profit) / SUM(fs.Line_Revenue)) * 100, 1) AS
Profit_Margin_Pct
FROM
    FactSales fs
```



```
INNER JOIN
    DimTime dti ON fs.TimeKey = dti.TimeKey
INNER JOIN
    DimTerritory dte ON fs.TerritoryKey = dte.TerritoryKey
-- WHERE dti.TimeKey >= 20240401 -- 772 entries for Q2 2024
-- WHERE dti.TimeKey >= 20240101
-- AND dti.TimeKey <= 20240331 -- 782 entries for Q1 2024
-- WHERE dti.TimeKey >= 20231001
-- AND dti.TimeKey <= 20231231 -- 802 entries for Q4 2023
WHERE dti.TimeKey >= 20230701
AND dti.TimeKey <= 20230930 -- 790 entries for Q3 2023
GROUP BY
    Date, Year, Month,
    dte.Salesperson_ID, dte.Territory_ID, dte.Territory_Name, dte.Country
ORDER BY
    Year, Quarter, Month,
    Territory_ID ASC;
```