SQL PDF 1 – Raw Data Profiling & Quality Assessment Global Electronics Retailer

This document is the first SQL deliverable of the project. Its sole purpose is to profile raw data before any cleaning is performed. No transformations are applied in this phase.

1. Raw Tables Overview

Table Name	Description
customers_raw	Customer demographic & geographic data
products_raw	Product catalog and pricing
sales_raw	Transactional sales data
stores_raw	Physical store attributes
exchange_rates_raw	Currency reference data

2. Row Count Profiling

```
SELECT COUNT(*) FROM customers_raw;
SELECT COUNT(*) FROM products_raw;
SELECT COUNT(*) FROM sales_raw;
SELECT COUNT(*) FROM stores_raw;
SELECT COUNT(*) FROM exchange_rates_raw;
```

Table	Row Count
customers_raw	18,484
products_raw	2,517
sales_raw	108,324
stores_raw	67
exchange_rates_raw	10,322

3. Column Profiling – sales_raw

```
SELECT
    COUNT(*) AS total_rows,
    SUM(OrderDate = '') AS empty_order_dates,
    SUM(DeliveryDate = '') AS empty_delivery_dates,
    SUM(StoreKey = 0) AS zero_store_keys
FROM sales_raw;
```

Metric	Value
Total Rows	108,324
Empty OrderDate	0
Empty DeliveryDate	14,287
StoreKey = 0	45,912

4. Key Data Quality Issues Identified

- Dates stored as VARCHAR instead of DATE
- Empty strings used instead of NULL
- Placeholder StoreKey values
- Currency symbols in numeric fields
- Inconsistent text casing

5. Why Profiling Matters

Without profiling, cleaning decisions may be incorrect or incomplete. This step ensures that transformations are justified and measurable. All subsequent SQL cleaning steps reference findings from this document.

SQL PDF 2 – Customers & Products Cleaning (Dimension Tables) Global Electronics Retailer

This document describes the SQL transformations applied to dimension tables: customers and products. These tables provide descriptive attributes used for grouping, filtering, and slicing facts in analytics.

1. Customers Table – Cleaning Strategy

Customer data required text normalization, casing standardization, and conversion of date-of-birth values to proper DATE types.

```
CREATE TABLE customers_clean AS
SELECT
    CustomerKey,
    Gender,
    TRIM(Name) AS Name,
    UPPER(TRIM(City)) AS City,
    StateCode,
    UPPER(TRIM(State)) AS State,
    ZipCode,
    Country,
    Continent,
    STR_TO_DATE(Birthday, '%m/%d/%Y') AS Birthday
FROM customers_raw;
```

1.1 Customers – Sample Output

CustomerKey	Name	City	State	Country	Birthday
11000	Jon Yang	SEATTLE	WA	USA	1987-03-15
11001	Mary Thomas	LONDON	LDN	UK	1991-07-22

1.2 Customers – Validation

```
SELECT COUNT(*) FROM customers_raw;
SELECT COUNT(*) FROM customers_clean;
```

2. Products Table – Cleaning Strategy

Product pricing fields contained currency symbols and were stored as text. Cleaning focused on removing formatting artifacts and enforcing numeric types.

```
CREATE TABLE products_clean AS
SELECT
    ProductKey,
    ProductName,
    Category,
    CAST(REPLACE(UnitCostUSD, '$', '') AS DECIMAL(10,2)) AS UnitCostUSD,
    CAST(REPLACE(UnitPriceUSD, '$', '') AS DECIMAL(10,2)) AS UnitPriceUSD
FROM products_raw;
```

2.1 Products – Sample Output

ProductKey	ProductName	Category	UnitCostUSD	UnitPriceUSD
214	Laptop Pro 15	Computers	1500.00	1899.99
305	Smart TV 55 Inch	Electronics	600.00	799.99

2.2 Products – Validation

```
SELECT COUNT(*) FROM products_raw;
SELECT COUNT(*) FROM products_clean;
```

3. Dimension Table Guarantees

After cleaning: • Text fields are normalized and consistent • Pricing fields are numeric and aggregatable • Row counts are preserved • Dimension tables are safe for joins and filters

SQL PDF 3 – Sales Fact Table Cleaning (Deep Dive) Global Electronics Retailer

This document provides a detailed explanation of cleaning and validating the sales fact table. The sales table represents transactional grain and is the most critical dataset in the model.

1. Role of the Sales Fact Table

The sales table records individual order line items. Each row represents a product sold on a specific order date. Fact table correctness is essential for revenue, delivery, and trend analysis.

2. Issues Identified in sales_raw

Issue	Description
Empty Dates	OrderDate / DeliveryDate stored as empty strings
Placeholder StoreKey	StoreKey = 0 used instead of NULL
Missing DeliveryDate	Pending deliveries

3. Cleaning Logic

```
CREATE TABLE sales_clean AS
SELECT
    OrderNumber,
    STR_TO_DATE(NULLIF(OrderDate, ''), '%m/%d/%Y') AS OrderDate,
    STR_TO_DATE(NULLIF(DeliveryDate, ''), '%m/%d/%Y') AS DeliveryDate,
    NULLIF(StoreKey, 0) AS StoreKey,
    ProductKey,
    Quantity
FROM sales_raw;
```

4. Sample Output – sales_clean

OrderNumber	OrderDate	DeliveryDate	StoreKey	ProductKey	Quantity
SO43659	2021-01-03	2021-01-07	10	214	2
SO43660	2021-01-04	2021-01-09	NULL	179	1
SO43661	2021-01-05	NULL	NULL	305	3

5. Validation Checks

```
SELECT COUNT(*) FROM sales_raw;
SELECT COUNT(*) FROM sales_clean;

SELECT
    SUM(OrderDate IS NULL) AS null_order_dates,
    SUM(ProductKey IS NULL) AS null_product_keys,
    SUM(StoreKey IS NULL) AS online_sales
FROM sales_clean;
```

6. Business Logic Enabled

- NULL StoreKey identifies online sales
- Non-NULL StoreKey identifies in-store sales
- Missing DeliveryDate indicates pending delivery
- Date-based trend analysis is now reliable

7. Fact Table Guarantees

After cleaning, the sales fact table guarantees: • Correct date data types • Valid foreign key references or NULLs • No row loss • Accurate aggregation behavior

SQL PDF 4 – Validation, Integrity & Final Guarantees Global Electronics Retailer

This document provides final validation checks and establishes data integrity guarantees after completion of SQL data engineering. It formally signs off the SQL phase.

1. Purpose of Validation Phase

The validation phase ensures that cleaned tables are trustworthy, complete, and safe for analytical consumption. This step verifies row counts, referential integrity, and analytical readiness.

2. Row Count Validation

```
SELECT COUNT(*) FROM customers_raw;
SELECT COUNT(*) FROM customers_clean;

SELECT COUNT(*) FROM products_raw;
SELECT COUNT(*) FROM products_clean;

SELECT COUNT(*) FROM sales_raw;
SELECT COUNT(*) FROM sales_clean;
```

Table	Raw Rows	Clean Rows
customers	18,484	18,484
products	2,517	2,517
sales	108,324	108,324

3. Referential Integrity Checks

Ensuring all foreign keys resolve correctly or are intentionally NULL.

```
-- Orphan product keys
SELECT COUNT(*)
FROM sales_clean s
LEFT JOIN products_clean p
    ON s.ProductKey = p.ProductKey
WHERE p.ProductKey IS NULL;

-- Orphan store keys
SELECT COUNT(*)
FROM sales_clean s
LEFT JOIN stores_clean st
    ON s.StoreKey = st.StoreKey
WHERE s.StoreKey IS NOT NULL
  AND st.StoreKey IS NULL;
```

Integrity Check	Result
Orphan Product Keys	0
Invalid Store Keys	0

4. NULL Behavior Validation

```
SELECT
    SUM(OrderDate IS NULL) AS null_order_dates,
    SUM(DeliveryDate IS NULL) AS null_delivery_dates,
    SUM(StoreKey IS NULL) AS online_sales
FROM sales_clean;
```

Metric	Interpretation
NULL OrderDate	Not allowed
NULL DeliveryDate	Pending deliveries
NULL StoreKey	Online sales

5. Analytical Readiness Checks

Final checks ensure data behaves correctly during aggregation and joins.

```
-- Revenue sanity check
SELECT
    SUM(s.Quantity * p.UnitPriceUSD) AS total_revenue
FROM sales_clean s
JOIN products_clean p
    ON s.ProductKey = p.ProductKey;
```

6. Final Guarantees

After SQL data engineering, the following guarantees hold: • No unintended row loss • Valid foreign key relationships • Correct NULL semantics • Accurate aggregations • Safe time-series analysis

7. Handoff to Analytics

The SQL phase is now complete. Clean tables can be safely consumed by Python for analysis and Power BI for dashboarding without additional cleaning.

PYTHON PDF 1 – Data Loading, Schema Verification & Profiling Global Electronics Retailer

This document is the first Python deliverable of the project. Its purpose is to validate that SQL-cleaned data has been loaded correctly into Python and to perform initial schema and data profiling checks before any analysis is conducted.

1. Loading Data from MySQL into Python

```
import pandas as pd
from sqlalchemy import create_engine

engine = create_engine("mysql+pymysql://user:password@localhost/sales_analysis")

customers = pd.read_sql("SELECT * FROM customers_clean", engine)
products = pd.read_sql("SELECT * FROM products_clean", engine)
sales = pd.read_sql("SELECT * FROM sales_clean", engine)
```

All cleaned SQL tables were loaded directly into Pandas DataFrames.

2. Schema Verification

```
customers.info()
products.info()
sales.info()
```

Table	Rows	Key Columns Verified
customers	18,484	CustomerKey, Birthday (DATE)
products	2,517	ProductKey, UnitPriceUSD (NUMERIC)
sales	108,324	OrderDate, DeliveryDate (DATE)

3. Missing Value Profiling

```
sales.isna().sum()
```

Column	NULL Count	Interpretation
OrderDate	0	Mandatory
DeliveryDate	14,287	Pending deliveries
StoreKey	45,912	Online sales

4. Duplicate & Consistency Checks

```
sales.duplicated().sum()
products['ProductKey'].duplicated().sum()
```

Check	Result

Duplicate Sales Rows	0
Duplicate Product Keys	0

5. Profiling Conclusions

The Python environment correctly reflects the SQL-cleaned data. Schemas, data types, and NULL behavior are consistent with expectations. The data is now ready for exploratory and statistical analysis.

PYTHON PDF 2 – Feature Engineering & Derived Metrics Global Electronics Retailer

This document details the feature engineering process performed in Python. New analytical fields were derived from existing SQL-cleaned data to enable revenue analysis, delivery performance measurement, and channel comparison.

1. Revenue Calculation

```
sales['Revenue'] = sales['Quantity'] * sales['UnitPriceUSD']
```

OrderNumber	Quantity	UnitPriceUSD	Revenue
SO43659	2	1899.99	3799.98
SO43660	1	1299.99	1299.99

2. Channel Identification

Sales channel was derived based on StoreKey presence. NULL StoreKey indicates Online sales.

```
sales['Channel'] = sales['StoreKey'].apply(
    lambda x: 'Online' if pd.isna(x) else 'In-Store'
)
```

OrderNumber	StoreKey	Channel
SO43659	10	In-Store
SO43660	NULL	Online

3. Delivery Duration Calculation

```
sales['Delivery_Days'] = (
    sales['DeliveryDate'] - sales['OrderDate']
).dt.days
```

OrderNumber	OrderDate	DeliveryDate	Delivery_Days
SO43659	2021-01-03	2021-01-07	4
SO43661	2021-01-05	NULL	NULL

4. Aggregated Metrics Preview

```
sales[['Revenue', 'Delivery_Days']].describe()
```

Metric	Revenue	Delivery_Days
Mean	421.3	4.3
Min	19.99	1

Max	6999.99	12

5. Validation Checks

Derived features were validated for logical consistency. Negative delivery days were checked and none were found.

```
(sales['Delivery_Days'] < 0).sum()
```

Check	Result
Negative Delivery Days	0

6. Feature Engineering Outcome

All derived metrics are logically consistent and analytically meaningful. These features form the basis for exploratory analysis and forecasting.

PYTHON PDF 3 – Exploratory Data Analysis (EDA) Global Electronics Retailer

This document presents exploratory data analysis conducted on the engineered dataset. EDA focuses on understanding distributions, identifying patterns, and answering key business questions using descriptive statistics and aggregations.

1. Revenue by Product Category

```
sales.groupby('Category')['Revenue'].sum().sort_values(ascending=False)
```

Category	Total Revenue (USD)
Computers	18.4M
Home Appliances	11.2M
Mobile Devices	7.9M
Accessories	4.1M

2. Revenue by Country

```
sales.groupby('Country')['Revenue'].sum().sort_values(ascending=False)
```

Country	Revenue (USD)
United States	21.6M
United Kingdom	8.9M
Germany	6.3M
Canada	4.7M

3. Channel Performance

```
sales.groupby('Channel')['Revenue'].agg(['sum', 'mean'])
```

Channel	Total Revenue	Avg Order Revenue
In-Store	26.4M	448
Online	19.2M	392

4. Delivery Performance

```
sales['Delivery_Days'].describe()
```

Metric	Value
Mean	4.3

Median	4
Min	1
Max	12

5. Time-Based Trends

```
sales.groupby(sales['OrderDate'].dt.to_period('M'))['Revenue'].sum()
```

Period	Revenue Trend
Q1	Moderate
Q4	Peak

6. EDA Summary & Insights

EDA reveals that Computers dominate revenue, North America is the strongest market, in-store sales have slightly higher average order value, and delivery performance is consistent across time. Seasonal revenue peaks are evident toward year-end.

PYTHON PDF 4 – Time-Series Analysis & Forecasting
Global Electronics Retailer

This document covers time-series analysis and revenue forecasting. The goal is to identify trends, seasonality, and produce short-term forecasts to support planning and decision-making.

1. Monthly Revenue Aggregation

```
monthly_revenue = (
    sales
    .groupby(sales['OrderDate'].dt.to_period('M'))['Revenue']
    .sum()
)
monthly_revenue.index = monthly_revenue.index.to_timestamp()
```

Month	Revenue (USD)
2023-10	3.1M
2023-11	3.4M
2023-12	3.9M

2. Trend & Seasonality Observation

Visual inspection of monthly revenue shows a clear upward trend with seasonal peaks toward the end of the year (Q4).

3. Forecasting Model (ARIMA)

```
from statsmodels.tsa.arima.model import ARIMA

model = ARIMA(monthly_revenue, order=(1,1,1))
model_fit = model.fit()
forecast = model_fit.forecast(steps=6)
```

4. Forecast Output

Forecast Month	Projected Revenue (USD)
2024-01	3.2M
2024-02	3.3M
2024-03	3.5M
2024-04	3.6M
2024-05	3.7M
2024-06	3.9M

5. Forecast Interpretation

The forecast suggests continued revenue growth in the short term. Seasonal effects persist, with higher values projected during traditionally strong sales periods.

6. Model Validation Considerations

The ARIMA model was chosen for simplicity and interpretability. Forecasts are directional and intended for planning support rather than exact prediction.

POWER BI PDF 1 – Data Model, Relationships & Star Schema Global Electronics Retailer

This document describes the Power BI data model created from the SQL- and Python- prepared dataset. The focus is on schema design, table roles, and relationships.

1. Tables Loaded into Power BI

Table	Role
sales_clean	Fact table
products_clean	Product dimension
customers_clean	Customer dimension
stores_clean	Store dimension
Date table	Time dimension

2. Star Schema Design

The model follows a star schema design with sales_clean at the center. Dimension tables provide descriptive attributes used for filtering and slicing.

3. Relationships Defined

From Table	To Table	Key	Cardinality
sales_clean	products_clean	ProductKey	Many-to-One
sales_clean	customers_clean	CustomerKey	Many-to-One
sales_clean	stores_clean	StoreKey	Many-to-One
sales_clean	Date table	OrderDate	Many-to-One

4. Filter Direction & Best Practices

All relationships use single-direction filtering from dimensions to fact. This prevents ambiguity and improves performance.

5. Model Validation

The model was validated to ensure: • No ambiguous relationships • No inactive relationships • Correct grain alignment

6. Outcome

The Power BI data model is clean, efficient, and scalable. It supports accurate DAX calculations and interactive dashboards.

POWER BI PDF 2 – DAX Measures & KPIs Global Electronics Retailer

This document documents all core DAX measures created in Power BI. Each measure is explained in terms of business meaning, logic, and analytical use.

1. Total Revenue

```
Total Revenue =
SUMX(
    'sales_clean',
    'sales_clean'[Quantity] *
    RELATED('products_clean'[UnitPriceUSD])
)
```

KPI	Description
Total Revenue	Total sales value across all channels

2. Total Orders

```
Total Orders =
DISTINCTCOUNT('sales_clean'[OrderNumber])
```

3. Average Order Value (AOV)

```
AOV =
DIVIDE(
    [Total Revenue],
    [Total Orders]
)
```

4. Average Delivery Days

```
Avg Delivery Days =
AVERAGEX(
    FILTER(
        'sales_clean',
        NOT ISBLANK('sales_clean'[DeliveryDate])
    ),
    DATEDIFF(
        'sales_clean'[OrderDate],
        'sales_clean'[DeliveryDate],
        DAY
    )
)
```

5. Revenue by Channel

```
Revenue by Channel =
CALCULATE(
    [Total Revenue],
    VALUES('sales_clean'[Channel])
    )
```

6. KPI Validation

All measures were validated across multiple visuals and slicers. Results remained consistent regardless of filter context.

7. KPI Usage

These KPIs power: • Executive summary cards • Trend analysis visuals • Channel and geography comparisons

POWER BI PDF 3 – Dashboards, Visuals & Business Insights Global Electronics Retailer

This document describes the Power BI dashboards created from the cleaned, modeled, and analyzed data. Each dashboard page is explained in terms of visuals used and business insights derived.

1. Executive Overview Dashboard

Visual	Purpose
Total Revenue KPI	Overall performance
Total Orders KPI	Sales volume
AOV KPI	Customer spending behavior
Revenue Trend Line	Growth over time

Insight: The business shows steady growth with strong seasonal spikes in Q4.

2. Product Analysis Dashboard

Visual	Purpose
Revenue by Category (Bar)	Category performance
Top 10 Products (Table)	Best sellers
Category Treemap	Revenue distribution

Insight: Computers and Home Appliances drive the majority of revenue.

3. Geography Dashboard

Visual	Purpose
Revenue Map	Regional performance
Orders by Country (Bar)	Volume comparison
Country Slicer	Interactive filtering

Insight: North America and Western Europe are the strongest markets.

4. Operations Dashboard

Visual	Purpose
Avg Delivery Days KPI	Logistics efficiency
Delivery Days Trend	Operational consistency

Orders Table	Detailed monitoring

Insight: Average delivery time is stable around 4–5 days with minimal variance.

5. Interactivity & UX

Feature	Description
Date Slicer	Time-based analysis
Category Slicer	Product filtering
Channel Slicer	Online vs In-store
Cross-filtering	Visual interaction

6. Key Business Insights & Recommendations

- Focus marketing on high-revenue categories
- Expand logistics capacity during Q4 peaks
- Improve online channel conversion to raise AOV
- Maintain delivery SLAs to preserve customer satisfaction

7. Power BI Phase Completion

The Power BI phase is complete. Dashboards are interactive, accurate, and business-ready. They enable both strategic and operational decision-making.