Below is **your complete, end-to-end ML Phase Master Plan**, fully aligned with the **final tables & columns** you shared.
This is a **professional, structured, and executable roadmap**—the same quality you'd give in a real project plan for a company.

It includes:

✔ SQL-based dataset extraction
✔ ML dataset design
✔ Full pipeline for recommender (content + collaborative → hybrid)
✔ Time-series forecasting
✔ Customer segmentation
✔ NLP + sentiment analysis
✔ Trend analysis & BI insights
✔ Model serving & integration plan
✔ What to build first vs later

🔥 ML MASTER PLAN (Start → End)

Tailored to Your Exact Tables & Columns

1️⃣ PHASE 0 — Confirm Data Foundations

(Already Done)

You now have:

✔ Customers (1.37M rows)

- Rich demographics
- Engagement fields
- First/Last name, email
- Signup_date (Great for churn & recency features)

✔ Articles (105k rows)

- Full metadata
- Price, stock
- Category structure
- Timestamps (created_at, last_updated)

✓ Transactions (31.7M rows)

- Time, customer, article
- Price
- Sales channel
- Enough volume for:
    - Collaborative filtering
    - Time-series forecasting
    - Trend analysis
    - RFM segmentation

✓ Reviews, Events, Orders, Order Items, Cart, Wishlist

Huge opportunity for:

- NLP
- Recommendation re-ranking
- Intent prediction
- Click → cart → purchase journey modeling

You are perfectly setup for ML.

2️⃣ PHASE 1 — ML Dataset Creation (Core ML Data Warehouse)

This is your **first real ML step.**

You will extract 6 main ML datasets:

📌 Dataset A — User-Item Interaction Matrix

From: **transactions**

Used For: **Collaborative Filtering**

Columns:

customer_id	article_id	purchase_count	total_spent	last_purchase_date	first_purchase_date	recency_days	time_weighted_score

SQL required:

- GROUP BY customer_id, article_id
- COUNT(*)
- SUM(price)
- MAX(t_dat)
- MIN(t_dat)

📌 Dataset B — Article Content Feature Dataset

From: **articles**, **categories**

Used For: **Content-Based Recommender**

Features include:

- product_name (TF-IDF)
- detail_desc (TF-IDF / word embeddings)
- product_type_name (OHE/embedding)
- garment_group_name
- section_name
- colour_group_name
- price
- stock
- category hierarchy embedding
- created_at age (article lifecycle)

📌 Dataset C — Customer Feature Dataset

From: **customers**, **transactions**, **orders**, **wishlist**, **cart**

Used For:

- **Customer segmentation**
- **Personalization layer for recommendations**

Features:

Demographics

- age
- signup_age = CURRENT_DATE – signup_date
- club_member_status
- fashion_news_frequency

Behavioral

- total_transactions
- total_amount
- avg basket size
- avg order value
- recency / frequency / monetary (RFM)
- session events
- wishlist size
- cart abandonment rate
- most engaged category
- most used sales_channel_id

📌 Dataset D — Time Series Dataset (For Forecasting)

Extract from **transactions**:

| article_id | date | daily_sales | daily_revenue | price | moving_avg | seasonality_index |

Group by daily/weekly:

```
SELECT article_id, t_dat::date AS date,
COUNT(*) AS sales,
SUM(price) AS revenue
FROM transactions
GROUP BY article_id, t_dat::date;
```

📌 Dataset E — Reviews Dataset

From: **reviews**

Used For: NLP

- Sentiment
- Topic modeling
- Review embeddings
- Toxicity detection
- Category sentiment index

Columns:

- rating
- review_text
- created_at
- article_id
- customer_id

📌 Dataset F — Customer Behavior Events Dataset

From: **events**

Used For:

- Clickstream analysis
- Conversion modeling
- Intent prediction
- Multi-step recommendation re-ranking

Columns:

- event_type (view / click / cart / buy / wishlist / session_start)
- created_at
- article_id
- session trajectory

3️⃣ PHASE 2 — Exploratory Data Analysis (EDA)

Goal: Understand behavior, trends, correlations.

Customers:

- Age distribution
- Membership statuses
- Fashion_news_frequency → purchase behavior
- RFM histograms
- Cluster tendencies

Articles:

- Price distribution
- Category distribution
- Article lifecycle
- Stock vs demand

Transactions:

- Monthly sales
- Seasonal peaks (Oct–Dec?)
- Weekday vs weekend sales
- Sales channel patterns

Events:

- Funnel (view → cart → wishlist → purchase)
- Drop-off points

Reviews:

- Rating distribution
- Sentiment over time
- Category sentiment differences

4️⃣ PHASE 3 — Feature Engineering

This is crucial for high model performance.

📌 For Collaborative Filtering

- time-decay weighting
- implicit feedback weighting
- normalize price
- long-tail filtering

📌 For Content-Based Recommender

- TF-IDF vectors
- BERT embeddings
- One-hot / embeddings for:
  - product group
  - garment group
  - color group
  - index group
- Price normalization
- Category tree embeddings (parent-child encoding)

📌 For Segmentation

- RFM score
- RFM percentile buckets
- genre preference encoding
- channel preference encoding
- event-derived behavioral features

📌 For Time Series

- Rolling windows
- Lag features
- Holiday flags
- Promotional event flags

- Price elasticity

📌 For NLP

- Clean text
- Remove stopwords
- Lemmatization
- Compute sentiment score
- Compute topic probability vector
- Generate review embedding using BERT

5️⃣ PHASE 4 — Model Building

Below is the recommended order:

🎯 MODEL 1: Customer Segmentation

Use:

- K-Means (baseline)
- GMM (advanced)
- HDBSCAN (robust clustering)

Output:

- 4–7 customer groups:
    - High value
    - Budget shoppers
    - Inactive customers
    - Trend followers
    - New users

Stored in DB table:

`customer_segments(customer_id, segment_id, updated_at)`

🎯 MODEL 2: Collaborative Filtering Recommender

Use:

- Implicit ALS (best for huge sparse data)
  or
- LightFM
  or
- Matrix Factorization (SGD/BPR)

Input:

user-item interaction matrix (Dataset A)

Output:

- User latent vectors
- Item latent vectors
- Top-N recommendations per user

Store in DB:

```
recommendations_cf
(customer_id, article_id, score)
```

🎯 MODEL 3: Content-Based Recommender

Inputs:

- TF-IDF/BERT embeddings
- Article metadata
- Category embeddings
- Price similarity
- Color/style similarity

Use:

- Cosine similarity
- Nearest neighbors

Store in DB:

```
recommendations_cb
(article_id, similar_article_id, similarity_score)
```

🎯 MODEL 4: Hybrid Recommender System

Combine:

```
final_score = α * CF_score + β * CB_score + γ * Popularity_score
```

And re-rank using:

- stock availability
- recent trends
- customer preferences

Produce the final master table:

```
recommendations_final(customer_id, article_id, final_score)
```

🎯 MODEL 5: Time-Series Forecasting

Purpose:

Predict future demand for:

- articles
- categories
- entire store

Models:

- Prophet (fast/easy)
- ARIMA
- LSTM (optional deep learning)

Predictions:

- future daily/weekly/monthly sales
- top upcoming products
- declining products
- seasonality trends

Store in DB:

```
article_forecast(article_id, date, predicted_sales)
```

🎯 MODEL 6: NLP / Sentiment Analysis

Tasks:

- Sentiment scoring using BERT
- Topic modeling (LDA)
- Keyword extraction
- Review embeddings
- Category-level sentiment

Outputs:

- sentiment_score 0–1
- topic distribution
- sentiment by category
- toxicity flag

Store in DB:

```
review_analysis(review_id, sentiment_score, topic_id, embedding)
```

🎯 MODEL 7: Trend Analysis & BI Insights

Uses all your data to generate:

- top trending products
- declining categories
- seasonal charts
- customer lifecycle KPIs
- inventory risk alerts
- price elasticity per article

Stored as:

```
trends(category_id, trend_score, growth_rate, updated_at)
```

6️⃣ PHASE 5 — Model Serving & Integration

The backend/frontend team will consume ML outputs via:

Option A — Store predictions in DB tables

Recommended for your current architecture.

Option B — Expose ML as REST API

Flask/FastAPI service:

```
/recommend/customer/{id}
/forecast/article/{id}
/segments/{id}
```

Option C — Microservice Container

Using Docker + PostgreSQL.

7️⃣ PHASE 6 — Monitoring & Retraining

Using:

- MLflow (experiment tracking)
- Airflow (scheduled retraining)
- EvidentlyAI (data drift, model drift, concept drift)

Schedule:

- Daily: Incremental transaction ingestion
- Weekly: Re-train CF model
- Monthly: Re-train CB/BERT embeddings
- Quarterly: Re-train segmentation

8️⃣ Recommended Execution Order

Here's EXACTLY the order you should follow:

✔ **Step 1 — Extract ML datasets (A–F)**

✔ **Step 2 — Perform EDA**

✔ **Step 3 — Customer Segmentation**

✔ **Step 4 — Collaborative Filtering**

✔ **Step 5 — Content-Based**

✔ **Step 6 — Hybrid Recommender**

✔ **Step 7 — Time Series Forecasting**

✔ **Step 8 — NLP Models**

✔ **Step 9 — Trend Analysis**

✔ **Step 10 — Model Serving**

✔ **Step 11 — Monitoring + Retraining Setup**

⭐ If you'd like, I can now:

→ **Generate all SQL queries for ML datasets**

→ **Create full ML folder structure**

→ **Create Jupyter notebook templates for each model**

→ **Write exact Python code for segmentation, CF, CB, hybrid, forecasting, NLP**

⟶ Create a full ML architecture diagram for your report

Just tell me:

👉 **"Generate ML folder + SQL + notebooks"**

or

👉 **"Start building model 1: Segmentation"**