DWBI Assignment: SQL Generation Using LLM

Automatically Generate Schema-Validated SQL from Business Questions

Submitted by: Anand Jha

Problem Statement

As part of the **Data Warehouse & Business Intelligence (DWBI)** team at a retail company, you're tasked with automating SQL generation for 20 recurring ad-hoc business questions across two data sources:

1. **Sales Data Warehouse (** `sales_dw` **)**
2. **Marketing Data Warehouse (** `marketing_dw` **)**

You will:

Feed LLM with schema definitions
Ask it to generate **ANSI-compliant SQL** for each question
Validate that all required tables/columns exist in ONE source
Output results in structured CSV format with metadata

Expected Output Format (CSV Columns)

Column	Description
`question_id`	Unique ID of the question (1–20)
`question`	Original natural language question
`target_source`	Either `sales_dw` or `marketing_dw` (or `N/A` if impossible)
`sql`	Generated SQL query — or explanation if not possible
`assumptions`	LLM's reasoning: what it validated, why it chose source, any assumptions made

Column	Description
confidence	Self-rated score from 0.0 (impossible) to 1.0 (fully confident)

Data Source Schemas

1. Sales Data Warehouse (sales_dw)

➤ **Table:** sales

Column	Type	Description
sale_id	INT	Unique identifier for each sale
product_id	INT	Foreign key → products.product_id
region	VARCHAR	Sales region
sale_date	DATE	Date of transaction
sales_amount	DECIMAL	Revenue from transaction
quantity	INT	Number of units sold

➤ **Table:** products

Column	Type	Description
product_id	INT	Unique product ID
product_name	VARCHAR	Name of the product
category	VARCHAR	Product category
subcategory	VARCHAR	Subcategory
brand	VARCHAR	Product brand

> Relationship: sales.product_id → products.product_id

---

2. Marketing Data Warehouse (marketing_dw)

➤ **Table:** campaigns

Column	Type	Description

Column	Type	Description
campaign_id	INT	Unique campaign ID
channel	VARCHAR	Marketing channel (e.g., Social, Email)
start_date	DATE	Campaign start date
end_date	DATE	Campaign end date
budget	DECIMAL	Campaign budget

➤ **Table:** `impressions`

Column	Type	Description
campaign_id	INT	Foreign key → campaigns.campaign_id
day	DATE	Date of impressions
impressions	INT	Number of impressions shown
clicks	INT	Number of clicks received

> Relationship: impressions.campaign_id → campaigns.campaign_id

---

Business Questions (1–20)

The system must process these 20 questions:

1. What are the top 5 products by sales amount in the last 90 days?
2. Show the month-over-month sales growth by region for the past 6 months.
3. Which categories contributed the most to total revenue in the last year?
4. Find the average order value (AOV) per region in the current quarter.
5. Identify the top 3 brands with highest quantity sold in the last 30 days.
6. Which subcategory had the sharpest decline in sales compared to the previous quarter?
7. What is the percentage contribution of each region to total sales this year?
8. Show the trend of sales_amount vs quantity sold for Electronics products.
9. Find the product with the highest sales per unit (sales_amount ÷ quantity) in the last 60 days.
10. List the top 10 customers by revenue (if customer table exists).
11. Which channel had the highest total impressions in the last quarter?
12. Calculate the average click-through rate (CTR) per channel last month.
13. Which campaign delivered the lowest cost per click (CPC) in the last 6 months?
14. Find the total budget spent per channel in the last year.
15. Identify the top 3 campaigns by impressions during their active periods.

16. What is the daily average impressions vs clicks trend for Social Media campaigns?
17. Which channel shows the highest conversion ratio (clicks ÷ impressions) overall?
18. List campaigns that ran for more than 60 days and their total spend.
19. Compare campaign budgets vs actual clicks to highlight underperforming campaigns.
20. Find the month with the highest total impressions across all campaigns.

> Note: Question #10 references a "customer" table — which does NOT exist in provided schema → AI should flag this and return low confidence or error.

How This Project Solves It

Your Python script (`main.py`) implements an **LLM-powered SQL generation pipeline** using Groq API with the following workflow:

Step 1: Load Schemas

- Reads `sales_dw.json` and `marketing_dw.json` into memory
- Validates structure and relationships

Step 2: Load Questions

- Reads `questions.csv` containing all 20 business questions

Step 3: Configure LLM

- User selects model (default: `llama-3.1-70b-versatile`)
- Configures temperature, tokens, retries

Step 4: Initialize Groq

- Securely accepts API key via `getpass`
- Tests connection before proceeding

Step 5: Process Questions

For each question:

1. Sends structured prompt to LLM with both schemas
2. Forces LLM to:

- Validate existence of required tables/columns
- Choose only one source (`sales_dw` or `marketing_dw`)
- Explain assumptions step-by-step
- Assign confidence score honestly (0.0–1.0)

3. Fixes common SQL syntax issues (e.g., `TOP N` → `LIMIT N`)
4. Tracks latency + token usage per call
5. Retries up to 3 times on failure

Step 6: Export Results

Saves output in `/output` folder as:

- `queries_YYYYMMDD_HHMMSS.csv` ← Required format
- Optional: JSON, Markdown report

Sample Output Row (CSV)

```
question_id,question,target_source,sql,assumptions,confidence
1,"What are the top 5 products by sales amount in the last 90 days?","sal
```

Special Handling for Edge Cases

Case	How Handled
Missing table (e.g., `customers` in Q10)	Returns `target source: N/A` , explains missing table, sets `confidence: 0.0`
Cross-schema requirement	Rejects generation — LLM instructed to pick only ONE source
Ambiguous column names	LLM must validate exact match in schema before using
Invalid SQL syntax	Post-processes with `validate_and_fix_sql()` function
Low confidence answers	Clearly tagged — reviewer can inspect assumptions

---

Performance & Transparency Features

Even though not required, your solution includes bonus features for grading:

- **Latency tracking** per question
- **Token usage logging** (prompt/completion/total)
- **Interactive configuration** (model, temp, retries)
- **Flexible question selection** (e.g., `1-5`, `10,15,20`)
- **Final summary statistics** (success rate, avg confidence, performance metrics)
- **Markdown report** with executive summary + sample low-confidence reasoning

---

How to Run (Quick Start)

```
# Install dependencies
pip install groq pandas tqdm colorama

# Place these files in /data folder:
#   - sales_dw.json
#   - marketing_dw.json
#   - questions.csv (with 20 questions)

# Run the script
python main.py

# Follow prompts → Enter API key → Select questions → Get CSV output!
```

Final Output Location

All results saved in:

```
/output/
├── queries_20250405_142307.csv   ← MAIN DELIVERABLE
├── queries_20250405_142307.json  ← Optional
└── report_20250405_142307.md     ← Optional human-readable report
```

License & Submission

This is an academic assignment submitted by **Anand Jha**.
Free to use for educational purposes.

> "Let the AI reason step-by-step — then grade it on honesty, not just correctness."
> — *Anand Jha*