Database Settings

MySQL Host

localhost

MySQL User

root

Password





Database

retail_dws

Groq Settings

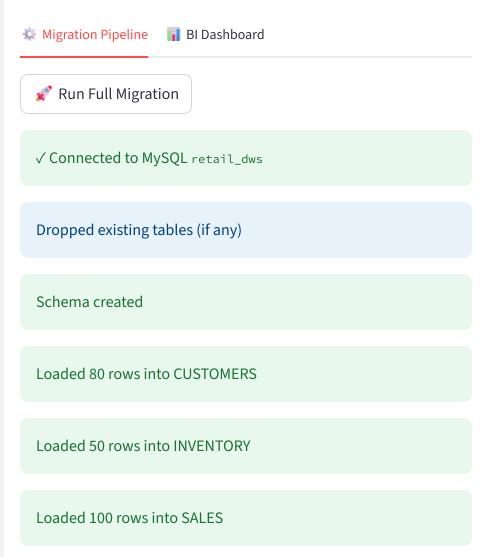
Groq API Key



Groq Model

llama-3.3-70b-versatile

GenAl-Assisted Migration Dashboard



Validation Results



```
"query":
   "-- 1. Count rows in CUSTOMERS, INVENTORY,
   SALES
   SELECT
      (SELECT COUNT(*) FROM CUSTOMERS) AS
   customers_count,
      (SELECT COUNT(*) FROM INVENTORY) AS
   inventory_count,
      (SELECT COUNT(*) FROM SALES) AS
   sales_count"
   ▼"result":[
      ▼ 0 : [
         0:55
        1:50
         2:60
      ]
   ]
}
▼1:{
   "query":
   "-- 2. Verify every SALES.customer_id exists
   in CUSTOMERS
   SELECT
      S.customer_id
  FROM
      SALES S
   LEFT JOIN
      CUSTOMERS C ON S.customer_id =
  C.customer_id
  WHERE
      C.customer_id IS NULL"
   "result": []
}
▼2:{
```

```
"query":
   "-- 3. Verify every SALES.product_id exists
  in INVENTORY
  SELECT
      S.product_id
   FROM
      SALES S
  LEFT JOIN
       INVENTORY I ON S.product_id =
  I.product_id
  WHERE
       I.product_id IS NULL"
   "result": []
▼3:{
   "query":
   "-- 4. Total of SALES.total_amount
  SELECT
       SUM(total_amount) AS total_sales
   FROM
      SALES"
   ▼"result":[
      ▼ 0 : [
        0:35544.29027366638
      ]
   ]
}
```

Translated PL/SQL

```
Converting Oracle PL/SQL to MySQL Stored Procedures

### Procedure to Get Monthly Sales

In MySQL, we don't need to specify the `OUT` paramet

```sql

DELIMITER //
```

```
CREATE PROCEDURE GetMonthlySales(IN p_month INT, IN
BEGIN
 SELECT DATE_FORMAT(sale_date, '%Y-%m') AS sale_m
 SUM(total_amount) AS total_sales
 FROM SALES
 WHERE MONTH(sale_date) = p_month
 AND YEAR(sale date) = p year
 GROUP BY DATE FORMAT(sale date, '%Y-%m');
END //
DELIMITER ;
Function to Check Reorder Point for Inventory
In MySQL, we can use a `FUNCTION` to return a `BOOLE
```sql
DELIMITER //
CREATE FUNCTION NeedReorder(p_product_id INT) RETURN
BEGIN
    DECLARE qty INT;
    SELECT quantity_in_stock INTO qty
    FROM INVENTORY
    WHERE product_id = p_product_id;
    IF qty < 100 THEN
       RETURN 1;
    ELSE
       RETURN 0;
    END IF;
END //
DELIMITER;
### Sample Business Query: Get Top 5 Customers by To
MySQL uses the `LIMIT` clause to limit the number of
```sql
SELECT c.customer_name, SUM(s.total_amount) AS total
```

```
FROM SALES s
JOIN CUSTOMERS c ON s.customer_id = c.customer_id
GROUP BY c.customer_name
ORDER BY total_purchase DESC
LIMIT 5;
. . .
Note: The above MySQL queries assume that the table
Example Use Cases
To call the `GetMonthlySales` procedure:
```sql
CALL GetMonthlySales(6, 2022);
To call the `NeedReorder` function:
```sql
SELECT NeedReorder(123) AS reorder_needed;
To execute the sample business query:
```sql
SELECT c.customer_name, SUM(s.total_amount) AS total
FROM SALES s
JOIN CUSTOMERS c ON s.customer_id = c.customer_id
GROUP BY c.customer_name
ORDER BY total_purchase DESC
LIMIT 5;
. . .
```

Generated BI Queries

```
```sql
-- Monthly Sales Trend
SELECT
 YEAR(order_date) AS year,
 MONTH(order_date) AS month,
```

```
SUM(total_amount) AS total_sales
FROM
 orders
GROUP BY
 YEAR(order_date),
 MONTH(order_date)
ORDER BY
 year,
 month;
-- Top 5 Customers
SELECT
 customer_name,
 SUM(total_amount) AS total_spent
FROM
 orders
GROUP BY
 customer_name
ORDER BY
 total_spent DESC
LIMIT 5;
-- Low Stock
SELECT
 product_name,
 quantity
FROM
 products
WHERE
 quantity < 100;
```

Report saved: output/migration\_report\_20250921\_2304.md