

SQL Code

```
1  CREATE database RetailDB_3;
2  USE RetailDB_3;
3  -- =====
4  -- CREATE DATABASE
5  -- =====
6  CREATE DATABASE IF NOT EXISTS RetailDB_3;
7  USE RetailDB_3;
8
9  -- =====
10 -- 1. Customerq
11 -- =====
12 CREATE TABLE Customerq (
13     customer_id INT AUTO_INCREMENT PRIMARY KEY,
14     name VARCHAR(100),
15     email VARCHAR(100),
16     city VARCHAR(50),
17     signup_date DATE
18 );
19
20 -- =====
21 -- 2. Suppliersq
22 -- =====
23 CREATE TABLE Suppliersq (
24     supplier_id INT AUTO_INCREMENT PRIMARY KEY,
25     supplier_name VARCHAR(100),
26     contact_email VARCHAR(100),
27     city VARCHAR(50)
28 );
29
30 -- =====
31 -- 3. Shippersq
32 -- =====
33 CREATE TABLE Shippersq (
34     shipper_id INT AUTO_INCREMENT PRIMARY KEY,
35     shipper_name VARCHAR(100),
36     contact VARCHAR(100)
37 );
38
39 -- =====
40 -- 4. Payment_Methodsq
41 -- =====
42 CREATE TABLE Payment_Methodsq (
43     payment_id INT AUTO_INCREMENT PRIMARY KEY,
44     payment_type VARCHAR(50) UNIQUE
45 );
46 DROP TABLE Productsq;
47 SET FOREIGN_KEY_CHECKS = 0;
48 DROP TABLE IF EXISTS Productsq;
49
50 -- =====
51 -- 5. Productsq
52 -- =====
```

```

53  CREATE TABLE Productsq (
54      product_id INT,
55      product_name VARCHAR(100),
56      category VARCHAR(50),
57      price DECIMAL(10,2),
58      stock_qty INT,
59      supplier_id INT
60  );
61
62  -- =====
63  -- 6. Orderq
64  -- =====
65  CREATE TABLE Orderq (
66      order_id INT AUTO_INCREMENT PRIMARY KEY,
67      customer_id INT,
68      order_date DATE,
69      payment_id INT,
70      shipper_id INT,
71      FOREIGN KEY (customer_id) REFERENCES Customerq(customer_id),
72      FOREIGN KEY (payment_id) REFERENCES Payment_Methodsq(payment_id),
73      FOREIGN KEY (shipper_id) REFERENCES Shippersq(shipper_id)
74  );
75  SET FOREIGN_KEY_CHECKS = 1;
76  -- =====
77  -- 7. Order_Itemq
78  -- =====
79  CREATE TABLE Order_Itemq (
80      order_item_id INT AUTO_INCREMENT PRIMARY KEY,
81      order_id INT,
82      product_id INT,
83      quantity INT,
84      price_each DECIMAL(10,2)
85  );
86
87
88  -- =====
89  -- Q1 Total revenue by shipper
90  -- =====
91  SELECT sh.shipper_name,
92         SUM(oi.quantity * oi.price_each) AS total_revenue
93  FROM Orderq o
94  JOIN Shippersq sh ON o.shipper_id = sh.shipper_id
95  JOIN Order_Itemq oi ON o.order_id = oi.order_id
96  GROUP BY sh.shipper_name;
97
98  -- =====
99  -- Q2 Top 5 highest spending customers
100 -- =====
101 SELECT c.name,
102         SUM(oi.quantity * oi.price_each) AS total_spent
103 FROM Customerq c
104 JOIN Orderq o ON c.customer_id = o.customer_id
105 JOIN Order_Itemq oi ON o.order_id = oi.order_id
106 GROUP BY c.name
107 ORDER BY total_spent DESC

```

```

108    LIMIT 5;
109
110    -- =====
111    -- Q3 Categories avg price > 8000
112    -- =====
113    SELECT category, AVG(price) AS avg_price
114        FROM Productsq
115        GROUP BY category
116        HAVING AVG(price) > 8000;
117
118    -- =====
119    -- Q4 Orders per city
120    -- =====
121    SELECT c.city, COUNT(o.order_id) AS total_orders
122        FROM Customerq c
123        JOIN Orderq o ON c.customer_id = o.customer_id
124        GROUP BY c.city
125        ORDER BY total_orders DESC;
126
127    -- =====
128    -- Q5 Suppliers with >1 category
129    -- =====
130    SELECT s.supplier_name
131        FROM Suppliersq s
132        JOIN Productsq p ON s.supplier_id = p.supplier_id
133        GROUP BY s.supplier_name
134        HAVING COUNT(DISTINCT p.category) > 1;
135
136    -- =====
137    -- Q6 Items per order
138    -- =====
139    SELECT o.order_id, COUNT(oi.order_item_id) AS item_count
140        FROM Orderq o
141        LEFT JOIN Order_Itemq oi ON o.order_id = oi.order_id
142        GROUP BY o.order_id;
143
144    -- =====
145    -- Q7 Customers spent > average
146    -- =====
147    SELECT c.name
148        FROM Customerq c
149        JOIN Orderq o ON c.customer_id = o.customer_id
150        JOIN Order_Itemq oi ON o.order_id = oi.order_id
151        GROUP BY c.name
152        HAVING SUM(oi.quantity * oi.price_each) >
153            (SELECT AVG(total_spent)
154                FROM (
155                    SELECT SUM(quantity * price_each) AS total_spent
156                        FROM Order_Itemq
157                        GROUP BY order_id
158                ) t);
159
160    -- =====
161    -- Q8 Products > category avg price
162    -- =====

```

```

163  SELECT *
164  FROM Productsq p
165  WHERE price >
166      (SELECT AVG(price)
167       FROM Productsq
168       WHERE category = p.category);
169
170  -- =====
171  -- Q9 Orders > 50000
172  -- =====
173  SELECT DISTINCT c.name
174  FROM Customerq c
175  JOIN Orderq o ON c.customer_id = o.customer_id
176  JOIN Order_Itemq oi ON o.order_id = oi.order_id
177  GROUP BY c.name, o.order_id
178  HAVING SUM(oi.quantity * oi.price_each) > 50000;
179
180  -- =====
181  -- Q10 Customers > avg orders
182  -- =====
183  SELECT customer_id
184  FROM Orderq
185  GROUP BY customer_id
186  HAVING COUNT(order_id) >
187      (SELECT AVG(cnt)
188       FROM (
189           SELECT COUNT(order_id) AS cnt
190           FROM Orderq
191           GROUP BY customer_id
192       ) t);
193
194  -- =====
195  -- Q11 Most expensive product
196  -- =====
197  SELECT *
198  FROM Productsq
199  WHERE price = (SELECT MAX(price) FROM Productsq);
200
201  -- =====
202  -- Q12 Rank customers by spending
203  -- =====
204  SELECT c.name,
205      SUM(oi.quantity * oi.price_each) AS total_spent,
206      RANK() OVER (ORDER BY SUM(oi.quantity * oi.price_each) DESC) AS rank_no
207  FROM Customerq c
208  JOIN Orderq o ON c.customer_id = o.customer_id
209  JOIN Order_Itemq oi ON o.order_id = oi.order_id
210  GROUP BY c.name;
211
212  -- =====
213  -- Q13 Cumulative sales by date
214  -- =====
215  SELECT o.order_date,
216      SUM(oi.quantity * oi.price_each) AS daily_sales,
217      SUM(SUM(oi.quantity * oi.price_each))

```

```

218      OVER (ORDER BY o.order_date) AS cumulative_sales
219  FROM Orderq o
220  JOIN Order_Itemq oi ON o.order_id = oi.order_id
221  GROUP BY o.order_date;
222
223  -- =====
224  -- Q14 Order count + percentage
225  -- =====
226  SELECT customer_id,
227      COUNT(order_id) AS order_count,
228      ROUND(
229          COUNT(order_id) * 100.0 /
230          SUM(COUNT(order_id)) OVER (), 2
231      ) AS percentage_contribution
232  FROM Orderq
233  GROUP BY customer_id;
234
235  -- =====
236  -- Q15 Most recent order per customer
237  -- =====
238  SELECT *
239  FROM (
240      SELECT o.*,
241          ROW_NUMBER() OVER
242              (PARTITION BY customer_id ORDER BY order_date DESC) rn
243      FROM Orderq o
244  ) t
245  WHERE rn = 1;
246
247  -- =====
248  -- Q16 Product sales rank per category
249  -- =====
250  SELECT p.product_name,
251      p.category,
252      SUM(oi.quantity) AS total_qty,
253      RANK() OVER (
254          PARTITION BY p.category
255          ORDER BY SUM(oi.quantity) DESC
256      ) AS rank_in_category
257  FROM Productsq p
258  JOIN Order_Itemq oi ON p.product_id = oi.product_id
259  GROUP BY p.product_name, p.category;
260
261  -- =====
262  -- Q17 Price category CASE
263  -- =====
264  SELECT product_name,
265      CASE
266          WHEN price > 60000 THEN 'High'
267          WHEN price BETWEEN 10000 AND 60000 THEN 'Medium'
268          ELSE 'Low'
269      END AS price_category
270  FROM Productsq;
271
272  -- =====

```

```

273 -- Q18 Top 3 customers (CTE)
274 -- =====
275 WITH cust_spend AS (
276     SELECT c.name,
277             SUM(oi.quantity * oi.price_each) AS total_spent
278     FROM Customerq c
279     JOIN Orderq o ON c.customer_id = o.customer_id
280     JOIN Order_Itemq oi ON o.order_id = oi.order_id
281     GROUP BY c.name
282 )
283     SELECT *
284     FROM cust_spend
285     ORDER BY total_spent DESC
286     LIMIT 3;
287
288 -- =====
289 -- Q19 Customer loyalty
290 -- =====
291 WITH order_count AS (
292     SELECT customer_id, COUNT(order_id) AS cnt
293     FROM Orderq
294     GROUP BY customer_id
295 )
296     SELECT customer_id,
297             CASE
298                 WHEN cnt >= 5 THEN 'High Loyalty'
299                 WHEN cnt >= 2 THEN 'Medium Loyalty'
300                 ELSE 'Low Loyalty'
301             END AS loyalty_status
302     FROM order_count;
303
304 -- =====
305 -- Q20 Monthly growth %
306 -- =====
307 WITH monthly_sales AS (
308     SELECT DATE_FORMAT(order_date, '%Y-%m') AS month,
309             SUM(oi.quantity * oi.price_each) AS revenue
310     FROM Orderq o
311     JOIN Order_Itemq oi ON o.order_id = oi.order_id
312     GROUP BY month
313 )
314     SELECT month,
315             revenue,
316             ROUND(
317                 (revenue - LAG(revenue) OVER (ORDER BY month)) * 100 /
318                 LAG(revenue) OVER (ORDER BY month), 2
319             ) AS growth_percent
320     FROM monthly_sales;
321
322 -- =====
323 -- Q21 Top 2 customers per city
324 -- =====
325 WITH city_sales AS (
326     SELECT c.city, c.name,
327             SUM(oi.quantity * oi.price_each) AS total_spent

```

```

328     FROM Customerq c
329     JOIN Orderq o ON c.customer_id = o.customer_id
330     JOIN Order_Itemq oi ON o.order_id = oi.order_id
331     GROUP BY c.city, c.name
332   )
333   SELECT *
334   FROM (
335     SELECT *,
336           RANK() OVER (PARTITION BY city ORDER BY total_spent DESC) rnk
337     FROM city_sales
338   ) t
339   WHERE rnk <= 2;
340
341 -- =====
342 -- Q22 Top 3 cities by revenue + shipper
343 -- =====
344   SELECT c.city, sh.shipper_name,
345         SUM(oi.quantity * oi.price_each) AS revenue
346   FROM Customerq c
347   JOIN Orderq o ON c.customer_id = o.customer_id
348   JOIN Shippersq sh ON o.shipper_id = sh.shipper_id
349   JOIN Order_Itemq oi ON o.order_id = oi.order_id
350   GROUP BY c.city, sh.shipper_name
351   ORDER BY revenue DESC
352   LIMIT 3;
353
354 -- =====
355 -- Q23 Order full details
356 -- =====
357   SELECT o.order_id, c.name, p.product_name,
358         s.supplier_name, sh.shipper_name
359   FROM Orderq o
360   JOIN Customerq c ON o.customer_id = c.customer_id
361   JOIN Order_Itemq oi ON o.order_id = oi.order_id
362   JOIN Productsq p ON oi.product_id = p.product_id
363   JOIN Suppliersq s ON p.supplier_id = s.supplier_id
364   JOIN Shippersq sh ON o.shipper_id = sh.shipper_id;
365
366 -- =====
367 -- Q24 Sales per supplier
368 -- =====
369   SELECT s.supplier_name,
370         SUM(oi.quantity * oi.price_each) AS total_sales,
371         AVG(oi.quantity * oi.price_each) AS avg_order_value
372   FROM Suppliersq s
373   JOIN Productsq p ON s.supplier_id = p.supplier_id
374   JOIN Order_Itemq oi ON p.product_id = oi.product_id
375   GROUP BY s.supplier_name;
376
377 -- =====
=====
378 -- Q25 Categories >30% revenue
379 -- =====
380 WITH cat_sales AS (
381   SELECT p.category,

```

```
382           SUM(oi.quantity * oi.price_each) AS revenue
383     FROM Productsq p
384   JOIN Order_Itemq oi ON p.product_id = oi.product_id
385   GROUP BY p.category
386 ),
387 total_rev AS (
388   SELECT SUM(revenue) AS total FROM cat_sales
389 )
390 SELECT cs.category
391 FROM cat_sales cs, total_rev tr
392 WHERE cs.revenue > 0.3 * tr.total;
```