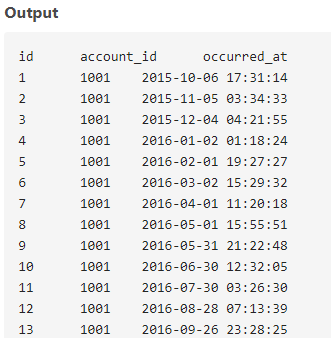
**Lab No 06**

**SQL for Data Analysis**

**Task#01 Run the following queries:**

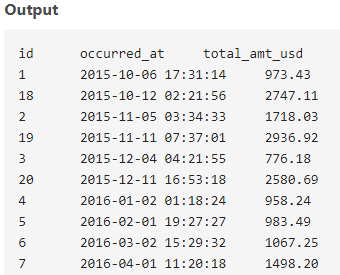
**-- Query 1: Select id, account\_id, and occurred\_at columns for all orders in the orders table.**

SELECT id, account\_id, occurred\_at FROM orders;



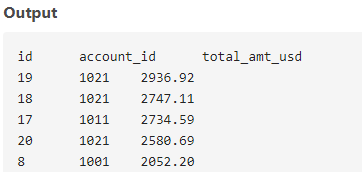
**-- Query 2: Return the 10 earliest orders with id, occurred\_at, and total\_amt\_usd.**

SELECT id, occurred\_at, total\_amt\_usd FROM orders ORDER BY occurred\_at LIMIT 10;



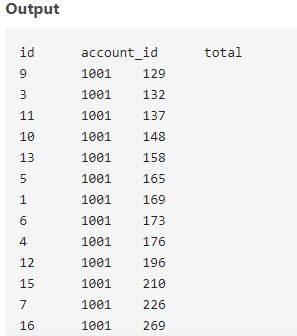
**-- Query 3: Return the top 5 orders with largest total\_amt\_usd and id, account\_id.**

SELECT id, account\_id, total\_amt\_usd FROM orders ORDER BY total\_amt\_usd DESC LIMIT 5;



**-- Query 4: Return the bottom 20 orders with least total and id, account\_id.**

SELECT id, account\_id, total FROM orders ORDER BY total LIMIT 20;



**-- Query 5: Return the top 5 rows from orders ordered by newest to oldest, with largest total\_amt\_usd for each date.**

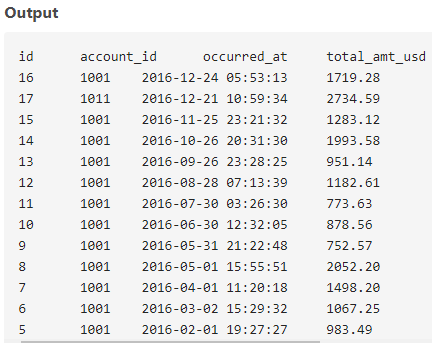
SELECT id, account\_id, occurred\_at, total\_amt\_usd FROM

(SELECT \*, ROW\_NUMBER() OVER (PARTITION BY DATE(occurred\_at)

ORDER BY total\_amt\_usd DESC)

AS row\_num FROM orders ) ranked

WHERE row\_num <= 5 ORDER BY occurred\_at DESC;



**-- Query 6: Return the top 10 rows from orders ordered by oldest to newest, with smallest total\_amt\_usd for each date.**

SELECT id, account\_id, occurred\_at, total\_amt\_usd

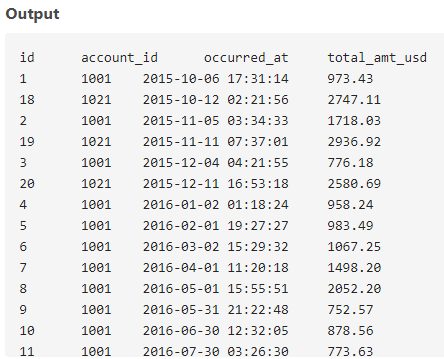
FROM ( SELECT \*, ROW\_NUMBER() OVER (PARTITION BY DATE(occurred\_at) ORDER BY total\_amt\_usd)

AS row\_num

FROM orders ) ranked

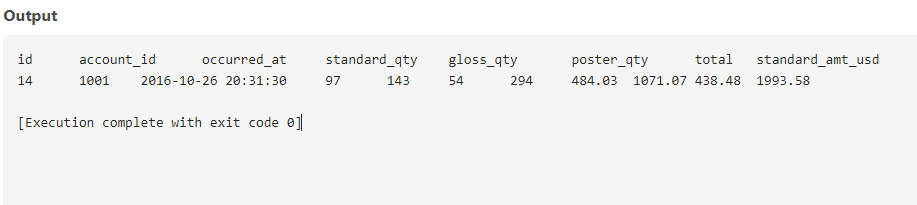
WHERE row\_num <= 10

ORDER BY occurred\_at;



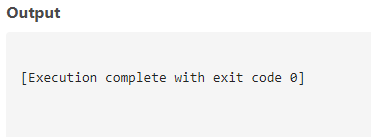
**-- Query 7: Pull the first 5 rows and all columns from the orders table with gloss\_amt\_usd >= 1000.**

SELECT \* FROM orders WHERE gloss\_amt\_usd >= 1000 LIMIT 5;



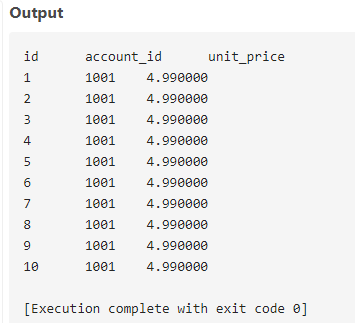
**-- Query 8: Pull the first 10 rows and all columns from the orders table with total\_amt\_usd < 500.**

SELECT \* FROM orders WHERE total\_amt\_usd < 500 LIMIT 10;



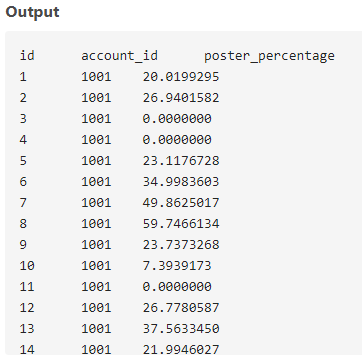
**-- Query 9: Create a column for unit price of standard paper for first 10 orders.**

SELECT id, account\_id, standard\_amt\_usd / standard\_qty AS unit\_price FROM orders LIMIT 10;



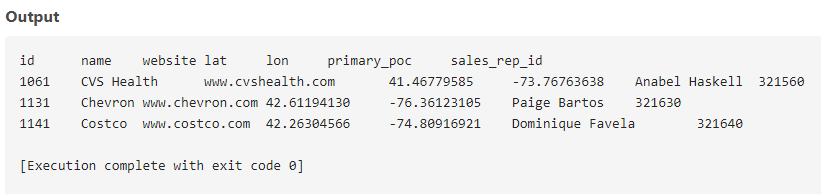
**-- Query 10: Find the percentage of revenue from poster paper for each order (with potential division workaround).**

SELECT id, account\_id, (poster\_amt\_usd \* 100.0) / NULLIF(total\_amt\_usd, 0) AS poster\_percentage FROM orders;



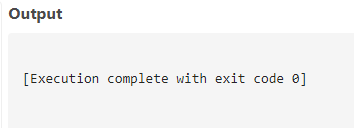
**-- Query 11: Use the accounts table to find all companies whose names start with 'C'.**

SELECT \* FROM accounts WHERE name LIKE 'C%';



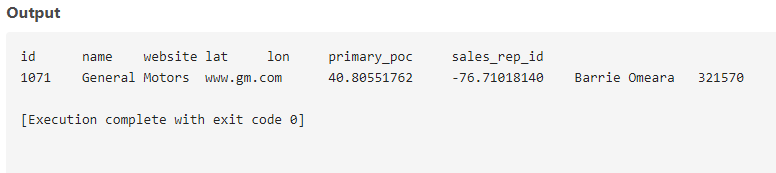
**-- Query 12: Use the accounts table to find all companies whose names contain the string 'one'.**

SELECT \* FROM accounts WHERE name LIKE '%one%';



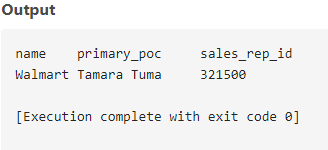
**-- Query 13: Use the accounts table to find all companies whose names end with 's'.**

SELECT \* FROM accounts WHERE name LIKE '%s';



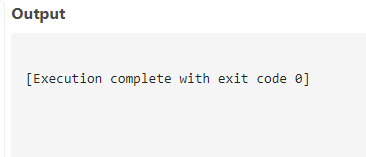
**-- Query 14: Use the accounts table to find the account name, primary\_poc, and sales\_rep\_id for Walmart, Target, and Nordstrom.**

SELECT name, primary\_poc, sales\_rep\_id FROM accounts WHERE name IN ('Walmart', 'Target', 'Nordstrom');



**-- Query 15: Use the web\_events table to find all information regarding individuals contacted via the channel of organic or adwords.**

SELECT \* FROM web\_events WHERE channel IN ('organic', 'adwords');

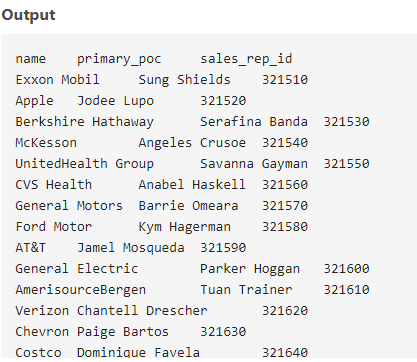


**-- Query 16 a: Use the accounts table to find account name, primary poc, and sales rep id for all stores except Walmart, Target, and Nordstrom.**

SELECT name, primary\_poc, sales\_rep\_id

FROM accounts

WHERE name NOT IN ('Walmart', 'Target', 'Nordstrom');

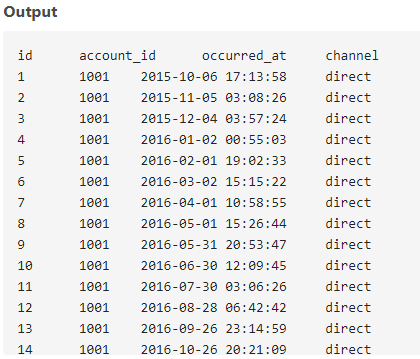


**-- Query 16 b: Use the web\_events table to find all information regarding individuals who were contacted via any method except organic or adwords.**

SELECT \*

FROM web\_events

WHERE channel NOT IN ('organic', 'adwords');



**-- Query 17 a: Use the accounts table to find all companies whose names do not start with 'C'.**

SELECT \*

FROM accounts

WHERE name NOT LIKE 'C%';

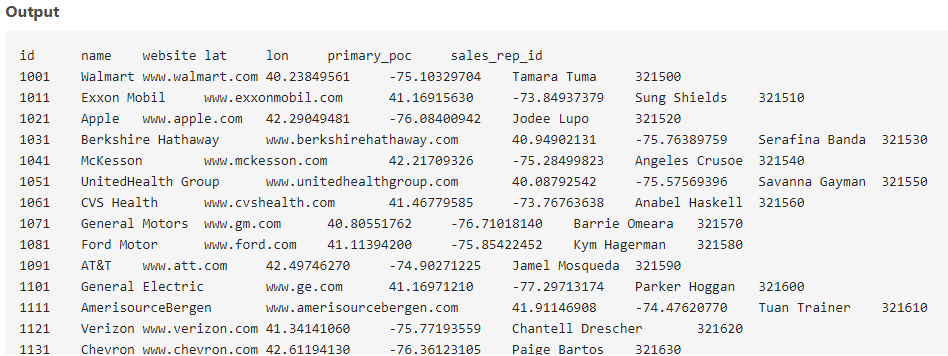


**-- Query 17 b: Use the accounts table to find all companies whose names do not contain the string 'one'.**

SELECT \*

FROM accounts

WHERE name NOT LIKE '%one%';

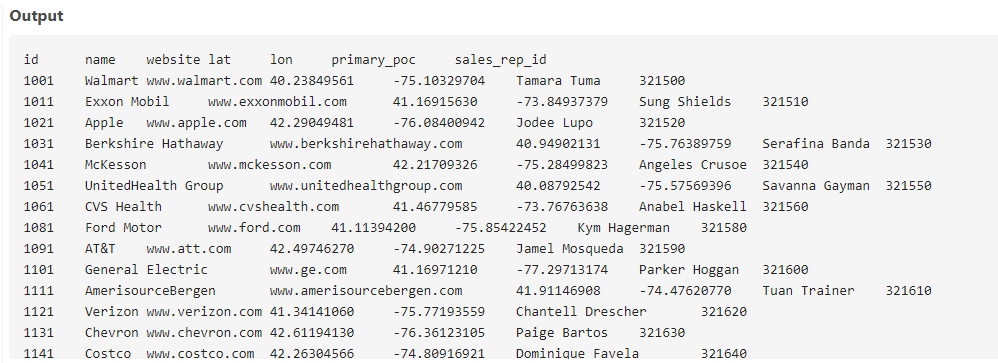


**-- Query 17 c: Use the accounts table to find all companies whose names do not end with 's'.**

SELECT \*

FROM accounts

WHERE name NOT LIKE '%s';

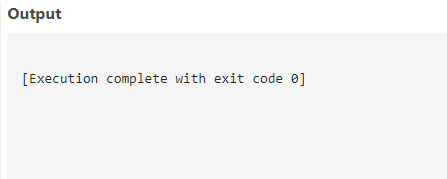


**-- Query 18: Write a query to return orders where standard\_qty > 1000, poster\_qty = 0, and gloss\_qty = 0.**

SELECT \*

FROM orders

WHERE standard\_qty > 1000 AND poster\_qty = 0 AND gloss\_qty = 0;

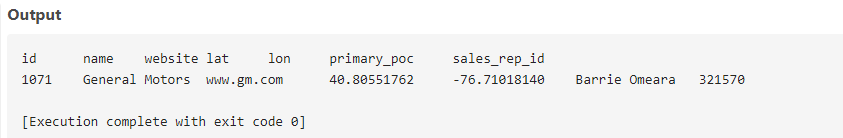


**-- Query 19: Use the accounts table to find companies whose names do not start with 'C' and end with 's'.**

SELECT \*

FROM accounts

WHERE name NOT LIKE 'C%' AND name LIKE '%s';



**-- Query 20: Use the web\_events table to find information about individuals contacted via organic or adwords in 2016.**

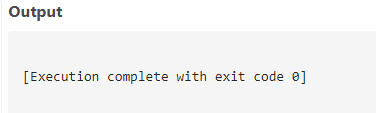
SELECT \*

FROM web\_events

WHERE channel IN ('organic', 'adwords')

AND DATE(occurred\_at) BETWEEN '2016-01-01' AND '2016-12-31'

ORDER BY occurred\_at DESC;

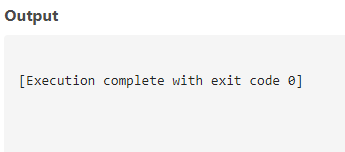


**-- Query 21: Return orders where standard\_qty > 1000, poster\_qty = 0, and gloss\_qty = 0.**

SELECT \*

FROM orders

WHERE standard\_qty > 1000 AND poster\_qty = 0 AND gloss\_qty = 0;

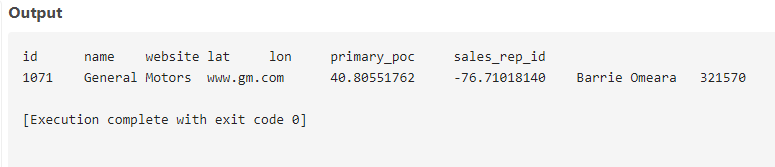


**-- Query 22: Use the accounts table to find companies whose names do not start with 'C' and end with 's'.**

SELECT \*

FROM accounts

WHERE name NOT LIKE 'C%' AND name LIKE '%s';



**-- Query 23: Use the web\_events table to find information about individuals contacted via organic or adwords in 2016.**

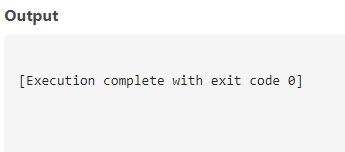
SELECT \*

FROM web\_events

WHERE channel IN ('organic', 'adwords')

AND DATE(occurred\_at) BETWEEN '2016-01-01' AND '2016-12-31'

ORDER BY occurred\_at DESC;



**[THE END]**