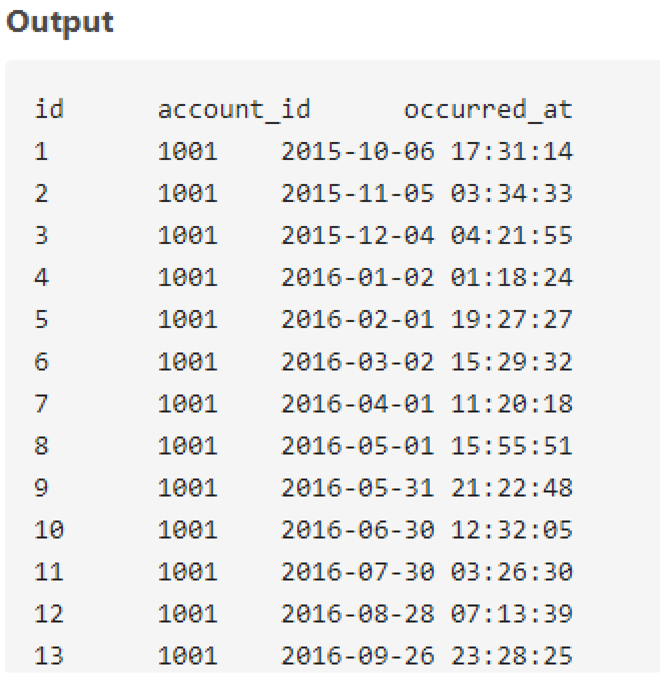
Lab No 06

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;

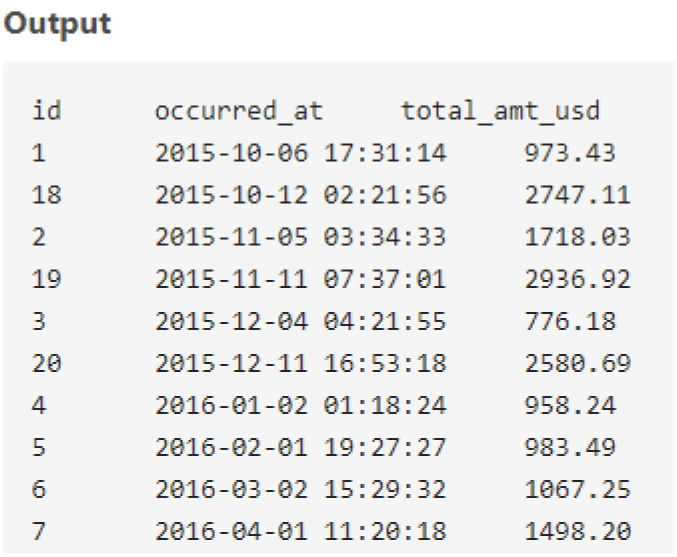


Output

```
id      account_id      occurred_at
1       1001    2015-10-06 17:31:14
2       1001    2015-11-05 03:34:33
3       1001    2015-12-04 04:21:55
4       1001    2016-01-02 01:18:24
5       1001    2016-02-01 19:27:27
6       1001    2016-03-02 15:29:32
7       1001    2016-04-01 11:20:18
8       1001    2016-05-01 15:55:51
9       1001    2016-05-31 21:22:48
10      1001    2016-06-30 12:32:05
11      1001    2016-07-30 03:26:30
12      1001    2016-08-28 07:13:39
13      1001    2016-09-26 23:28:25
```

-- 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;
```

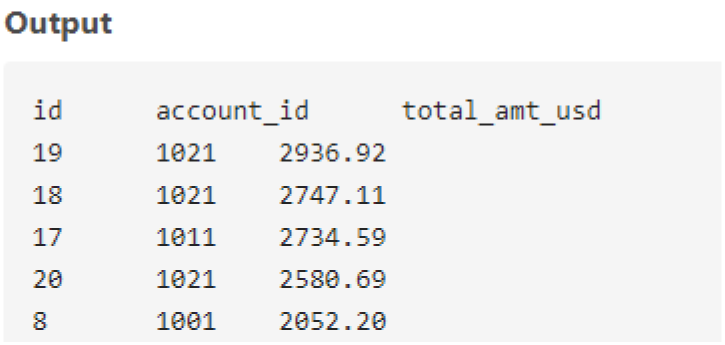


Output

```
id      occurred_at      total_amt_usd
1       2015-10-06 17:31:14      973.43
18      2015-10-12 02:21:56      2747.11
2       2015-11-05 03:34:33      1718.03
19      2015-11-11 07:37:01      2936.92
3       2015-12-04 04:21:55      776.18
20      2015-12-11 16:53:18      2580.69
4       2016-01-02 01:18:24      958.24
5       2016-02-01 19:27:27      983.49
6       2016-03-02 15:29:32      1067.25
7       2016-04-01 11:20:18      1498.20
```

-- 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;

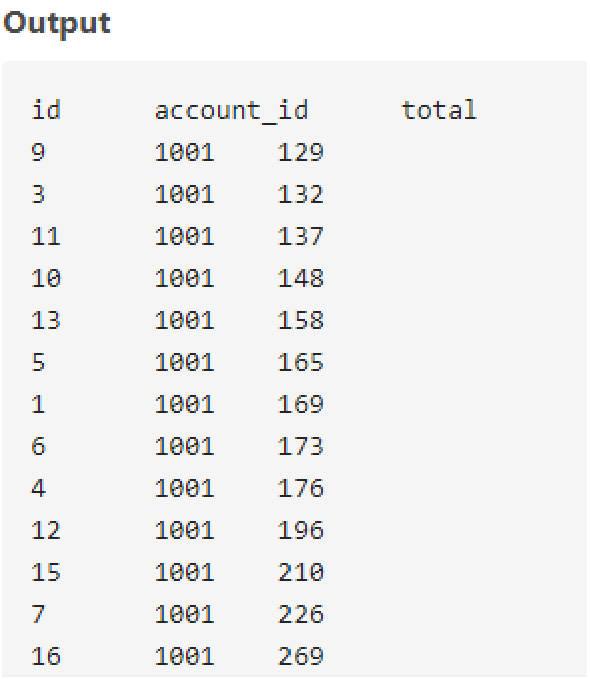


Output

```
id      account_id      total_amt_usd
19      1021    2936.92
18      1021    2747.11
17      1011    2734.59
20      1021    2580.69
8       1001    2052.20
```

-- 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;

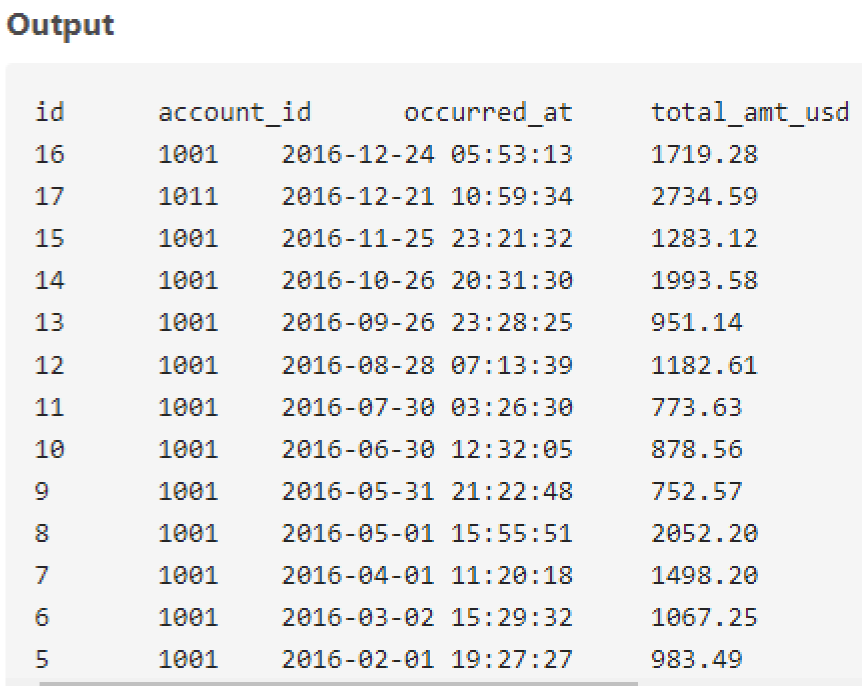


Output

```
id      account_id      total
9       1001    129
3       1001    132
11      1001    137
10      1001    148
13      1001    158
5       1001    165
1       1001    169
6       1001    173
4       1001    176
12      1001    196
15      1001    210
7       1001    226
16      1001    269
```

-- 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;
```

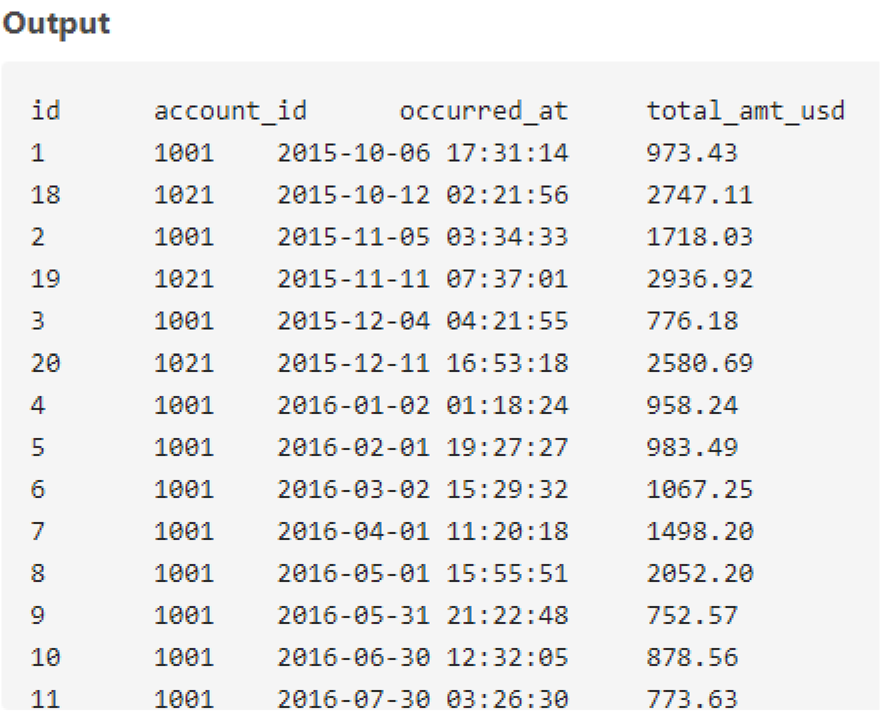


Output

```
id      account_id      occurred_at       total_amt_usd
16      1001     2016-12-24 05:53:13      1719.28
17      1011     2016-12-21 10:59:34      2734.59
15      1001     2016-11-25 23:21:32      1283.12
14      1001     2016-10-26 20:31:30      1993.58
13      1001     2016-09-26 23:28:25      951.14
12      1001     2016-08-28 07:13:39      1182.61
11      1001     2016-07-30 03:26:30      773.63
10      1001     2016-06-30 12:32:05      878.56
9       1001     2016-05-31 21:22:48      752.57
8       1001     2016-05-01 15:55:51      2052.20
7       1001     2016-04-01 11:20:18      1498.20
6       1001     2016-03-02 15:29:32      1067.25
5       1001     2016-02-01 19:27:27      983.49
```

-- 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;
```

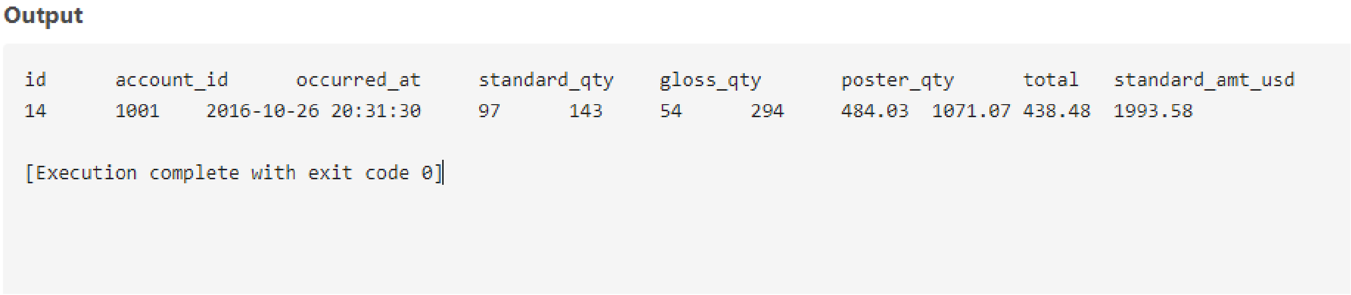


Output

```
id      account_id      occurred_at         total_amt_usd
1       1001    2015-10-06 17:31:14     973.43
18      1021    2015-10-12 02:21:56     2747.11
2       1001    2015-11-05 03:34:33     1718.03
19      1021    2015-11-11 07:37:01     2936.92
3       1001    2015-12-04 04:21:55     776.18
20      1021    2015-12-11 16:53:18     2580.69
4       1001    2016-01-02 01:18:24     958.24
5       1001    2016-02-01 19:27:27     983.49
6       1001    2016-03-02 15:29:32     1067.25
7       1001    2016-04-01 11:20:18     1498.20
8       1001    2016-05-01 15:55:51     2052.20
9       1001    2016-05-31 21:22:48     752.57
10      1001    2016-06-30 12:32:05     878.56
11      1001    2016-07-30 03:26:30     773.63
```

-- 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;

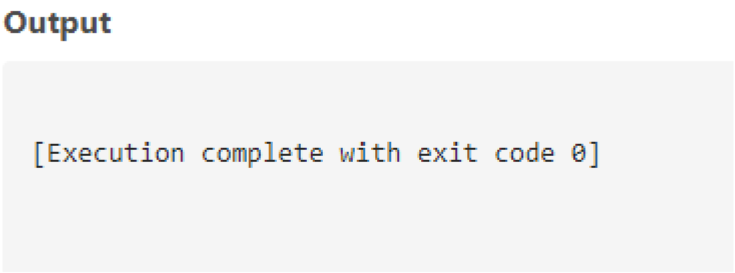


Output

```
id      account_id      occurred_at     standard_qty    gloss_qty       poster_qty      total   standard_amt_usd
14      1001    2016-10-26 20:31:30     97      143     54      294     484.03  1071.07 438.48  1993.58

[Execution complete with exit code 0]
```

-- 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;
```

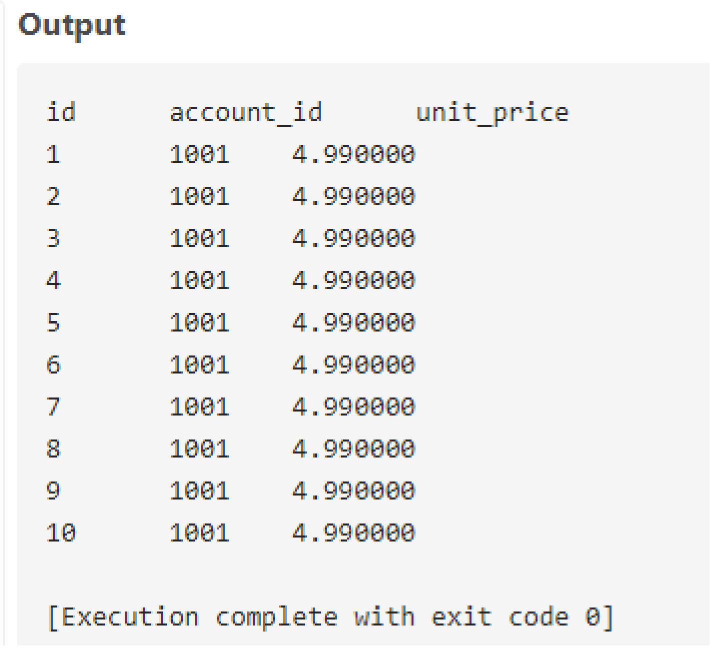


Output

```
[Execution complete with exit code 0]
```

-- 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;

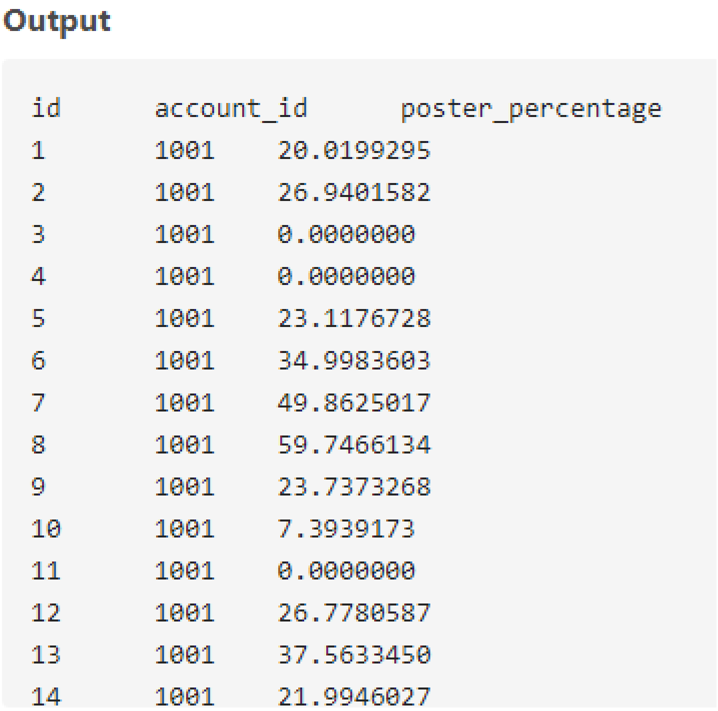


Output

```
id      account_id      unit_price
1       1001    4.990000
2       1001    4.990000
3       1001    4.990000
4       1001    4.990000
5       1001    4.990000
6       1001    4.990000
7       1001    4.990000
8       1001    4.990000
9       1001    4.990000
10      1001    4.990000

[Execution complete with exit code 0]
```

-- 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;

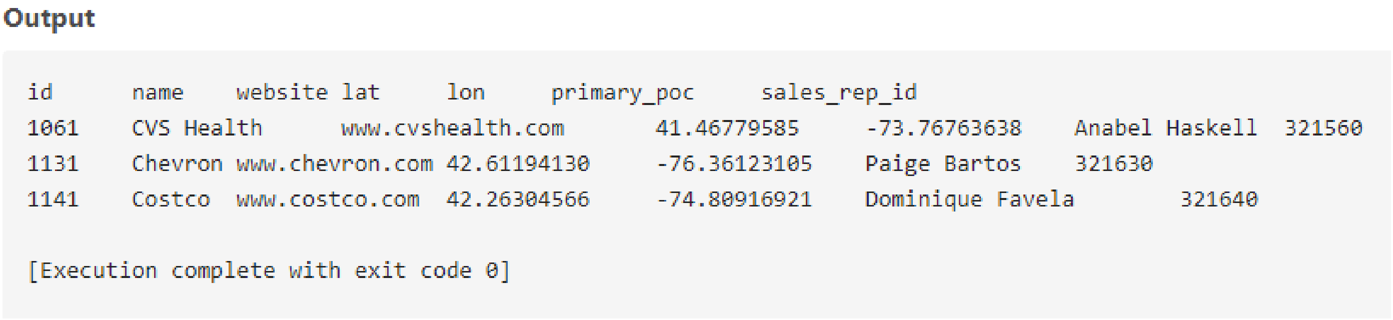


Output

```
id      account_id      poster_percentage
1       1001    20.0199295
2       1001    26.9401582
3       1001    0.0000000
4       1001    0.0000000
5       1001    23.1176728
6       1001    34.9983603
7       1001    49.8625017
8       1001    59.7466134
9       1001    23.7373268
10      1001    7.3939173
11      1001    0.0000000
12      1001    26.7780587
13      1001    37.5633450
14      1001    21.9946027
```

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

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

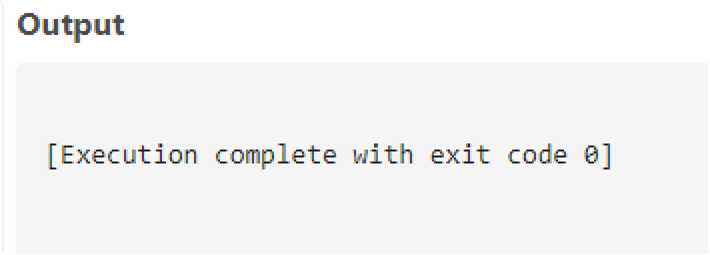


Output

```
id      name    website lat     lon     primary_poc     sales_rep_id
1061    CVS Health      www.cvshealth.com       41.46779585     -73.76763638    Anabel Haskell  321560
1131    Chevron www.chevron.com 42.61194130     -76.36123105    Paige Bartos    321630
1141    Costco  www.costco.com  42.26304566     -74.80916921    Dominique Favela        321640

[Execution complete with exit code 0]
```

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

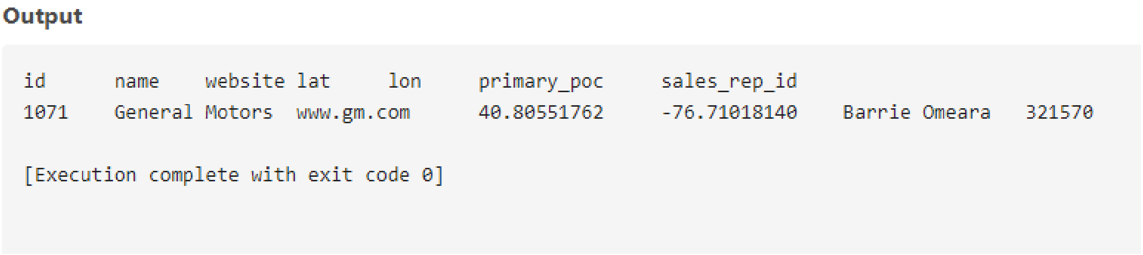
```sql
SELECT * FROM accounts WHERE name LIKE '%one%';
```



```
[Execution complete with exit code 0]
```

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

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

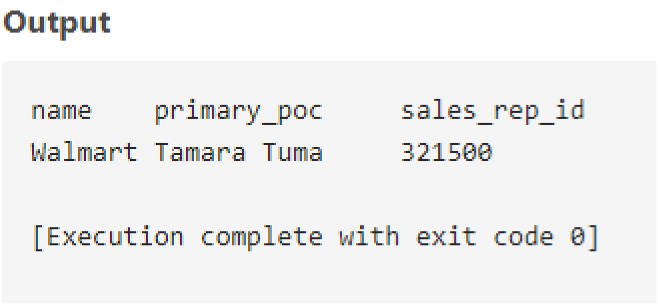


Output

```
id      name    website lat     lon     primary_poc     sales_rep_id
1071    General Motors  www.gm.com      40.80551762     -76.71018140    Barrie Omeara   321570

[Execution complete with exit code 0]
```

-- 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');

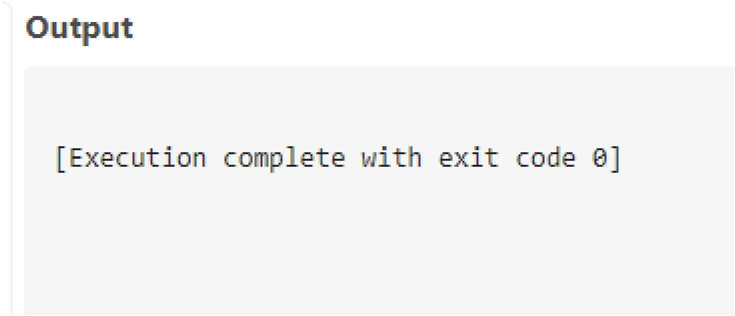


Output

```
name    primary_poc     sales_rep_id
Walmart Tamara Tuma     321500

[Execution complete with exit code 0]
```

-- 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');
```



Output

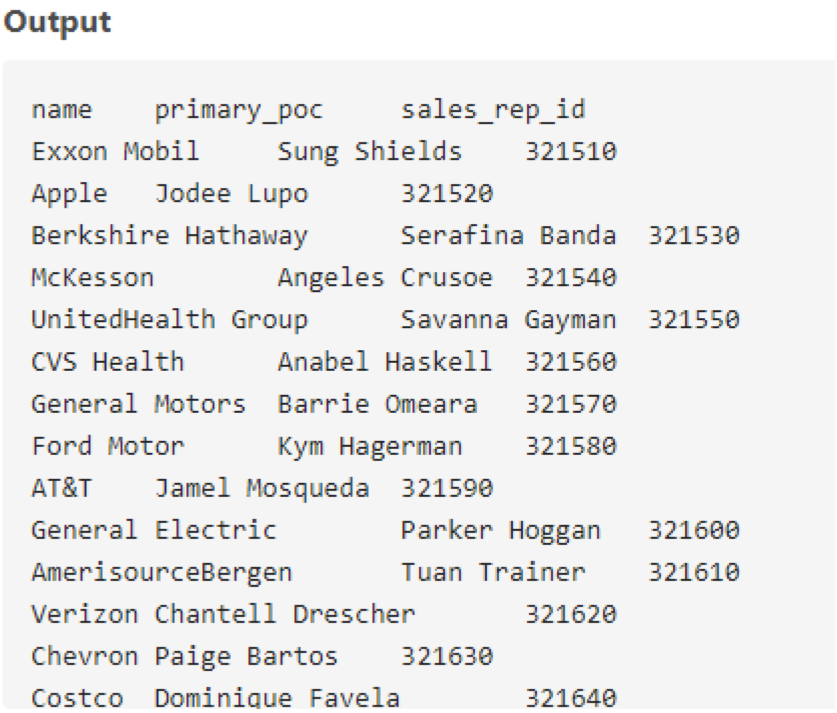
```
[Execution complete with exit code 0]
```

-- 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');



Output

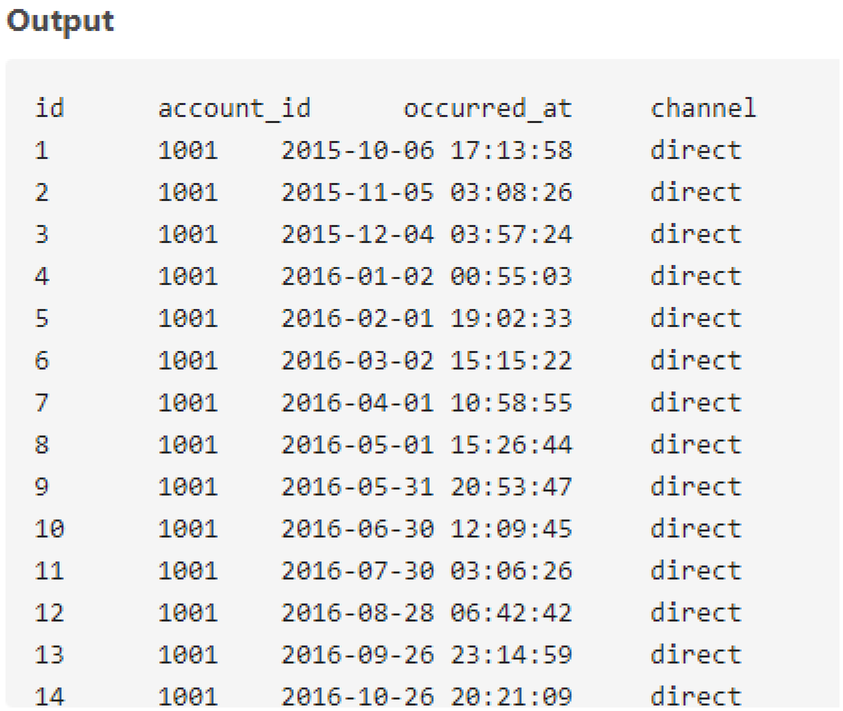
```
name    primary_poc     sales_rep_id
Exxon Mobil     Sung Shields    321510
Apple   Jodee Lupo      321520
Berkshire Hathaway      Serafina Banda  321530
McKesson        Angeles Crusoe  321540
UnitedHealth Group      Savanna Gayman  321550
CVS Health      Anabel Haskell  321560
General Motors  Barrie Omeara   321570
Ford Motor      Kym Hagerman    321580
AT&T    Jamel Mosqueda  321590
General Electric        Parker Hoggan   321600
AmerisourceBergen       Tuan Trainer    321610
Verizon Chantell Drescher       321620
Chevron Paige Bartos    321630
Costco  Dominique Favela        321640
```

-- **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');

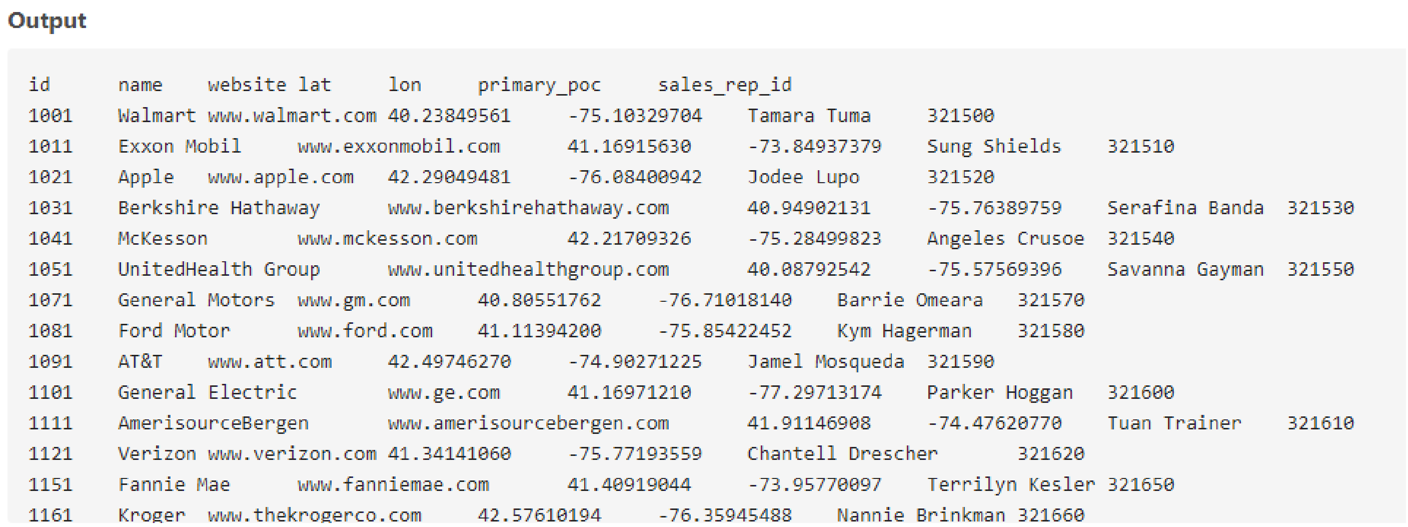


Output

```
id      account_id      occurred_at         channel
1       1001     2015-10-06 17:13:58     direct
2       1001     2015-11-05 03:08:26     direct
3       1001     2015-12-04 03:57:24     direct
4       1001     2016-01-02 00:55:03     direct
5       1001     2016-02-01 19:02:33     direct
6       1001     2016-03-02 15:15:22     direct
7       1001     2016-04-01 10:58:55     direct
8       1001     2016-05-01 15:26:44     direct
9       1001     2016-05-31 20:53:47     direct
10      1001     2016-06-30 12:09:45     direct
11      1001     2016-07-30 03:06:26     direct
12      1001     2016-08-28 06:42:42     direct
13      1001     2016-09-26 23:14:59     direct
14      1001     2016-10-26 20:21:09     direct
```

-- 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%';
```



Output

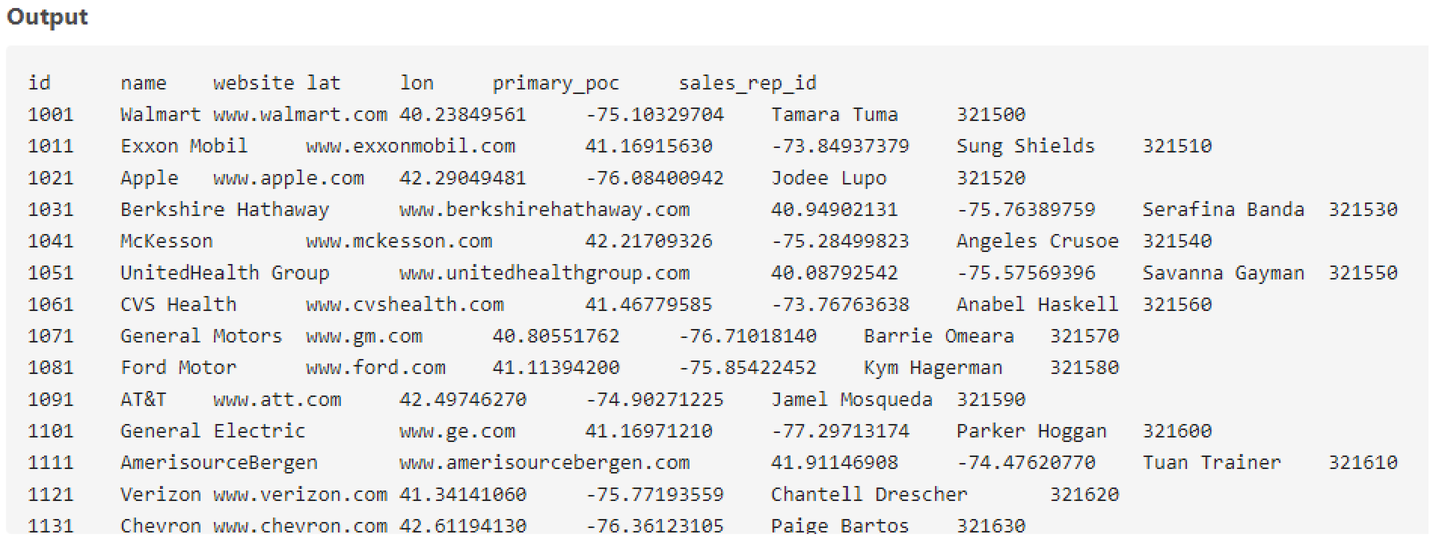
```
id      name    website lat     lon     primary_poc     sales_rep_id
1001    Walmart www.walmart.com 40.23849561     -75.10329704    Tamara Tuma     321500
1011    Exxon Mobil     www.exxonmobil.com      41.16915630     -73.84937379    Sung Shields    321510
1021    Apple   www.apple.com   42.29049481     -76.08400942    Jodee Lupo      321520
1031    Berkshire Hathaway      www.berkshirehathaway.com       40.94902131     -75.76389759    Serafina Banda  321530
1041    McKesson        www.mckesson.com        42.21709326     -75.28499823    Angeles Crusoe  321540
1051    UnitedHealth Group      www.unitedhealthgroup.com       40.08792542     -75.57569396    Savanna Gayman  321550
1071    General Motors  www.gm.com      40.80551762     -76.71018140    Barrie Omeara   321570
1081    Ford Motor      www.ford.com    41.11394200     -75.85422452    Kym Hagerman    321580
1091    AT&T    www.att.com     42.49746270     -74.90271225    Jamel Mosqueda  321590
1101    General Electric        www.ge.com      41.16971210     -77.29713174    Parker Hoggan   321600
1111    AmerisourceBergen       www.amerisourcebergen.com       41.91146908     -74.47620770    Tuan Trainer    321610
1121    Verizon www.verizon.com 41.34141060     -75.77193559    Chantell Drescher       321620
1151    Fannie Mae      www.fanniemae.com       41.40919044     -73.95770097    Terrilyn Kesler 321650
1161    Kroger  www.thekrogerco.com     42.57610194     -76.35945488    Nannie Brinkman 321660
```

-- 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%';

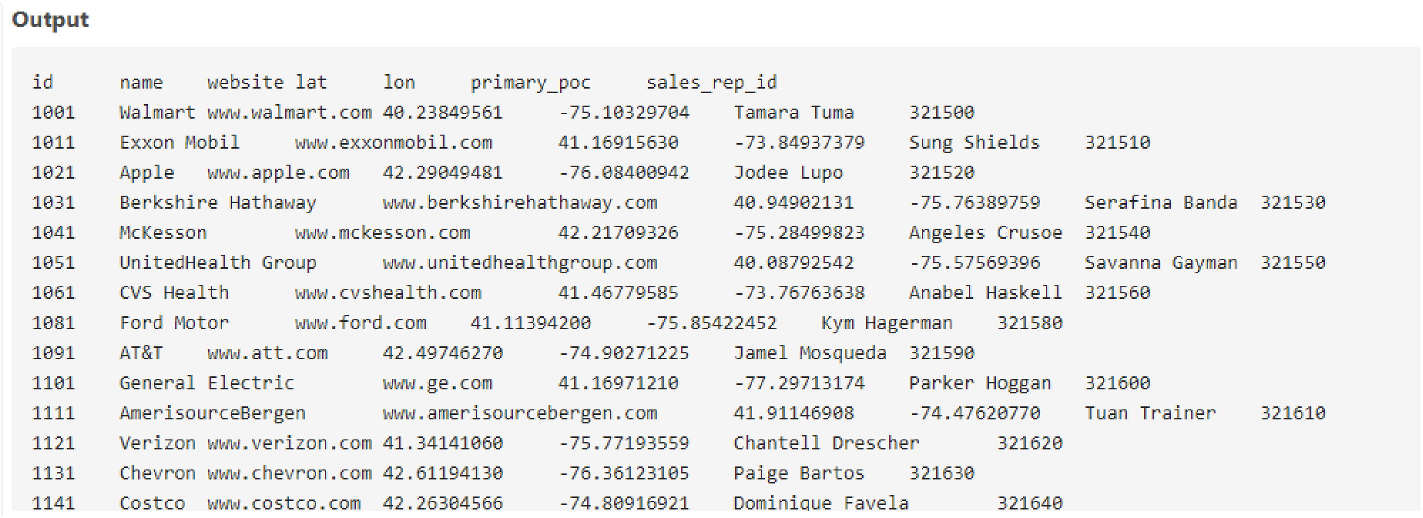


Output

```
id	name	website	lat	lon	primary_poc	sales_rep_id
1001	Walmart	www.walmart.com	40.23849561	-75.10329704	Tamara Tuma	321500
1011	Exxon Mobil	www.exxonmobil.com	41.16915630	-73.84937379	Sung Shields	321510
1021	Apple	www.apple.com	42.29049481	-76.08400942	Jodee Lupo	321520
1031	Berkshire Hathaway	www.berkshirehathaway.com	40.94902131	-75.76389759	Serafina Banda	321530
1041	McKesson	www.mckesson.com	42.21709326	-75.28499823	Angeles Crusoe	321540
1051	UnitedHealth Group	www.unitedhealthgroup.com	40.08792542	-75.57569396	Savanna Gayman	321550
1061	CVS Health	www.cvshealth.com	41.46779585	-73.76763638	Anabel Haskell	321560
1071	General Motors	www.gm.com	40.80551762	-76.71018140	Barrie Omeara	321570
1081	Ford Motor	www.ford.com	41.11394200	-75.85422452	Kym Hagerman	321580
1091	AT&T	www.att.com	42.49746270	-74.90271225	Jamel Mosqueda	321590
1101	General Electric	www.ge.com	41.16971210	-77.29713174	Parker Hoggan	321600
1111	AmerisourceBergen	www.amerisourcebergen.com	41.91146908	-74.47620770	Tuan Trainer	321610
1121	Verizon	www.verizon.com	41.34141060	-75.77193559	Chantell Drescher	321620
1131	Chevron	www.chevron.com	42.61194130	-76.36123105	Paige Bartos	321630
```

-- 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';
```

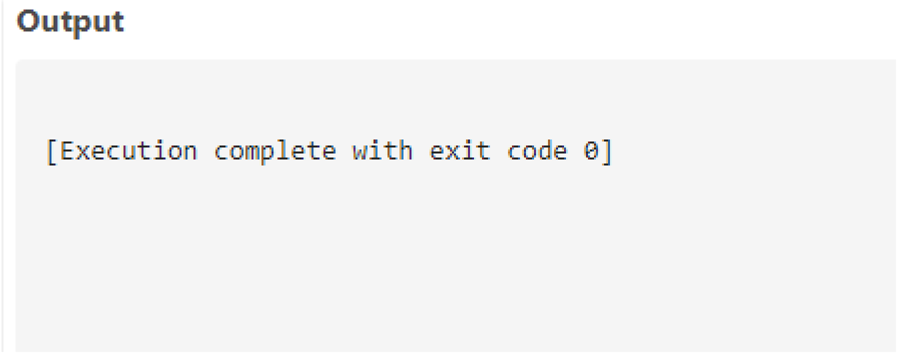


Output

```
id      name    website lat     lon     primary_poc     sales_rep_id
1001    Walmart www.walmart.com 40.23849561     -75.10329704    Tamara Tuma     321500
1011    Exxon Mobil     www.exxonmobil.com      41.16915630     -73.84937379    Sung Shields    321510
1021    Apple   www.apple.com   42.29049481     -76.08400942    Jodee Lupo      321520
1031    Berkshire Hathaway      www.berkshirehathaway.com       40.94902131     -75.76389759    Serafina Banda  321530
1041    McKesson        www.mckesson.com        42.21709326     -75.28499823    Angeles Crusoe  321540
1051    UnitedHealth Group      www.unitedhealthgroup.com       40.08792542     -75.57569396    Savanna Gayman  321550
1061    CVS Health      www.cvshealth.com       41.46779585     -73.76763638    Anabel Haskell  321560
1081    Ford Motor      www.ford.com    41.11394200     -75.85422452    Kym Hagerman    321580
1091    AT&T    www.att.com     42.49746270     -74.90271225    Jamel Mosqueda  321590
1101    General Electric        www.ge.com      41.16971210     -77.29713174    Parker Hoggan   321600
1111    AmerisourceBergen       www.amerisourcebergen.com       41.91146908     -74.47620770    Tuan Trainer    321610
1121    Verizon www.verizon.com 41.34141060     -75.77193559    Chantell Drescher       321620
1131    Chevron www.chevron.com 42.61194130     -76.36123105    Paige Bartos    321630
1141    Costco  www.costco.com  42.26304566     -74.80916921    Dominique Favela        321640
```

-- 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;
```

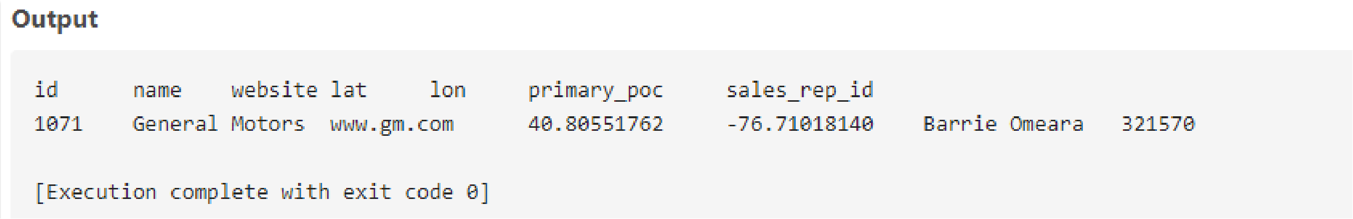


Output

```
[Execution complete with exit code 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';
```

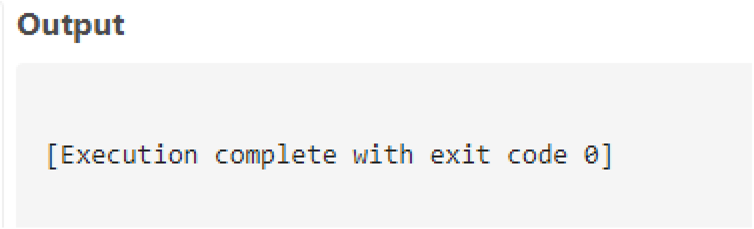


Output

```
id      name    website lat     lon     primary_poc     sales_rep_id
1071    General Motors  www.gm.com      40.80551762     -76.71018140    Barrie Omeara   321570

[Execution complete with exit code 0]
```

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

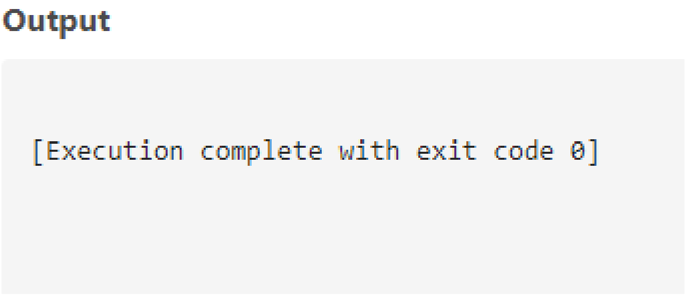
```
SELECT *

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;
```



```
[Execution complete with exit code 0]
```

-- 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;
```



Output

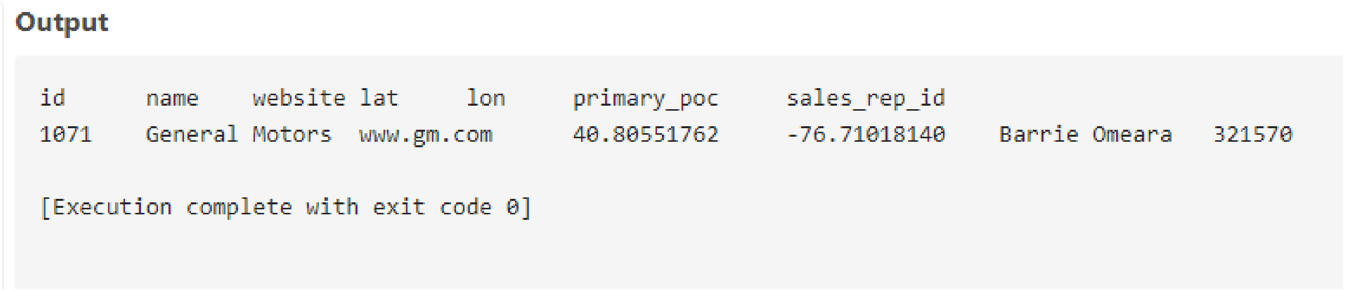
```
[Execution complete with exit code 0]
```

-- 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';

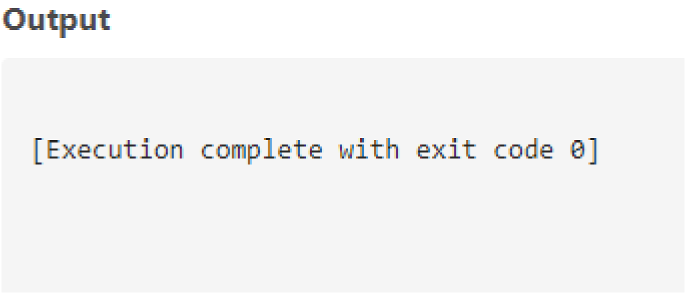


Output

```
id      name    website lat     lon     primary_poc     sales_rep_id
1071    General Motors  www.gm.com      40.80551762     -76.71018140    Barrie Omeara   321570

[Execution complete with exit code 0]
```

-- 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;
```



Output

```
[Execution complete with exit code 0]
```

[THE END]