P4 练习一_基本SQL

1. 尝试编写自己的查询，以便为 orders 表中的所有订单选择 id、account_id 和 occurred_at 列。
   Answer: `SELECT id, account_id, occurred_at FROM orders;`

2. 编写一个查询，将响应限制在前 15 行，和包括 web_events 表中的 occurred_at、account_id 和 channel 字段，我们来试一下。
   Answer: `SELECT occurred_at, account_id, channel FROM web_events LIMIT 15;`

3. ORDER BY 练习
   1) 编写查询，返回 orders 表的前 10 个订单。包含 id、occurred_at 和 total_amt_usd。
   Answer: `SELECT id, occurred_at, total_amt_usd FROM orders ORDER BY occurred_at LIMIT 10;`
   2) 编写一个查询，基于 total_amt_usd 返回前 5 个最高的 订单 (orders 表)。包括 id、account_id 和 total_amt_usd。
   Answer: `SELECT id, account_id, total_amt_usd FROM orders ORDER BY total_amt_usd DESC LIMIT 5;`
   3) 编写一个查询，基于 total 返回前 20 个最低订单 (orders 表)。包括 id、account_id 和 total。
   Answer: `SELECT id, account_id, total FROM orders ORDER BY total LIMIT 20;`
   4) 编写一个查询，返回按从最新到最早排序的订单中的前 5 行，但需首先列出每个日期的最大 total_amt_usd。
   Answer: `SELECT * FROM orders ORDER BY occurred_at DESC, total_amt_usd DESC LIMIT 5;`
   5)编写一个查询，返回按从最早到最新排序的 订单 中的前 10 行，但需首先列出每个日期的最小 total_amt_usd。
   Answer: `SELECT * FROM orders ORDER BY occurred_at, total_amt_usd LIMIT 10;`

4. WHERE 练习
   1) 从订单表提取出大于或等于 1000 的 gloss_amt_usd 美元数额的前五行数据（包含所有列）。
   Answer: `SELECT * FROM orders WHERE gloss_amt_usd >= 1000 LIMIT 5;`
   2) 从订单 表提取出小于 500 的 total_amt_usd 美元数额的前十行数据（包含所有列）。
   Answer: `SELECT * FROM orders WHERE total_amt_usd < 500 LIMIT 10;`
   3) WHERE 和非数字数据一起使用
   Answer: `SELECT name, website, primary_poc FROM accounts WHERE name = 'Exxon Mobil';`

5. 算术运算符
   下面是提供解决方案的查询。你会注意到，由于除以零，所以出现了一个错误。在这节课结束之前，我们将具体学习 CASE 语句， 了解关于此错误的原因。一个简单（但不完全准确）的解决方案是将给分母加1。
   1) **Answer:** `SELECT id, account_id, standard_amt_usd/standard_qty AS unit_price FROM orders LIMIT 10;`
   2)**Answer:** `SELECT id, account_id, poster_amt_usd/(standard_amt_usd + gloss_amt_usd + poster_amt_usd) AS post_per FROM orders;`

6. LIKE 练习
   使用 accounts (客户) 表查找
   1) 所有以 'C' 开头公司名。
   Answer: `SELECT name FROM accounts WHERE name LIKE 'C%';`
   2)名称中包含字符串 'one' 的所有公司名。
   Answer: `SELECT name FROM accounts WHERE name LIKE '%one%';`
   3)所有以 's' 结尾的公司名。
   Answer: `SELECT name FROM accounts WHERE name LIKE '%s';`

7. IN 练习
   1) 使用 客户 表查找 Walmart、Target 和 Nordstrom 的 name (客户名称), primary_poc (主要零售店), and sales_rep_id (销售代表 id)。
   Answer: `SELECT name, primary_poc, sales_rep_id FROM accounts WHERE name IN ('Walmart', 'Target', 'Nordstrom');`
   2) 使用 web_events 表查找有关通过 organic 或 adwords 联系的所有个人信息。
   Answer: `SELECT * FROM web_events WHERE channel IN ('organic', 'adwords');`

8. NOT 练习
   1) 使用客户表查找除 Walmart、Target 和 Nordstrom 之外的所有商店的客户名称、主要零售店和销售代表 id。
   Answer: `SELECT name, primary_poc, sales_rep_id FROM accounts WHERE name NOT IN ('Walmart','Target','Nordstrom');`

2) 使用 web_events 表查找除通过任何方法联系的个人的所有信息，除了使用 organic 或 adwords 方法。 使用客户表查找：

Answer: `SELECT * FROM web_events WHERE channel NOT IN ('organic', 'adwords');`

3) 所有不以 'C' 开头的公司名。

Answer: `SELECT name FROM accounts WHERE name NOT LIKE 'C%';`

4) 所有名称中不包含字符串 'one' 的公司名。

Answer: `SELECT name FROM accounts WHERE name NOT LIKE '%one%';`

5) 所有不以 's' 结尾的公司名。

Answer: `SELECT name FROM accounts WHERE name NOT LIKE '%s';`

9. AND 和 BETWEEN 练习

1) 编写一个查询，返回所有订单，其中 standard_qty 超过 1000，poster_qty 是 0，gloss_qty 也是 0。

Answer: `SELECT * FROM orders WHERE standard_qty > 1000 AND poster_qty = 0 AND gloss_qty = 0;`

2) 使用客户表查找所有不以 'C' 开始但以 's' 结尾的公司名。

Answer: `SELECT name FROM accounts WHERE name NOT LIKE 'C%' AND name like '%s';`

3) 使用 web_events 表查找通过 organic 或 adwords 联系，并在 2016 年的任何时间开通帐户的个人全部信息，并按照从最新到最旧的顺序排列。

NOTE: 你可能会觉得对日期数据使用 BETWEEN 不太好理解。对于单纯的日期数据而言 (只包含年月日不包含时间的数据，如'2016-12-31'), 其默认时间为日期当天的 00:00:00，而 BETWEEN 通常是不包含端点在内的，这就是为什么将右边的时间点设为'2017-01-01'的原因。

Answer: `SELECT * FROM web_events WHERE channel IN ('organic','adwords') AND occurred_at BETWEEN '2016-01-01' AND '2017-01-01' ORDER BY occurred_at DESC;`

10. OR 练习

1) 查找 订单 (orders) id 的列表，其中 gloss_qty 或 poster_qty 大于 4000。只在结果表中包含 id 字段。

Answer: `SELECT id FROM orders WHERE gloss_qty > 4000 OR poster_qty > 4000;`

2) 编写一个查询，返回订单 (orders) 的列表，其中标准数量 (standard_qty)为零，光泽度 (gloss_qty) 或海报数量 (poster_qty)超过 1000。

Answer: `SELECT * FROM orders WHERE standard_qty = 0 AND (gloss_qty > 1000 OR poster_qty > 1000);`

3) 查找以 'C' 或 'W' 开头的所有公司名 (company names)，主要联系人 (primary contact) 包含 'ana' 或 'Ana'，但不包含 'eana'。

Answer: `SELECT * FROM accounts WHERE (name LIKE 'C%' OR name LIKE 'W%') AND ((primary_poc LIKE %ana% OR primary_poc LIKE '%Ana%') AND primary_poc NOT LIKE '%eana%');`

P4 练习二_join练习

1. 尝试获取 accounts 表格中的所有数据，以及 orders 表格中的所有数据。
   Answer: `SELECT orders.*, accounts.* FROM accounts JOIN orders ON accounts.id = orders.account_id;`

2. 尝试从 orders 表格中获取 standard_qty、gloss_qty 和 poster_qty，并从 accounts 表格中获取 website和 primary_poc。
   Answer: `SELECT orders.standard_qty, orders.gloss_qty, orders.poster_qty, accounts.website, accounts.primary_poc FROM orders JOIN accounts ON orders.account_id = account_id;`

3. 为与客户名称 Walmart 相关的所有 web_events 创建一个表格。表格应该包含三列：primary_poc、事件时间和每个事件的渠道。此外，你可以选择添加第四列，确保仅选中了 Walmart 事件。
   Answer: `SELECT a.primary_poc, w.occurred_at, w.channel, a.name FROM web_events w JOIN accounts a ON w.account_id = a.id WHERE a.name = 'Walmart;'`

4. 为每个 sales_rep（销售代表）对应的 region（区域）以及相关的 accounts（客户）创建一个表格，最终表格应该包含三列：区域 name（名称）、销售代表 name（名称），以及客户 name（名称）。根据客户名称按字母顺序 (A-Z) 排序。
   Answer: `SELECT r.name region, s.name rep, a.name account FROM sales_reps s JOIN region r ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id ORDER BY a.name`

5. 提供每个 order（订单）的每个区域 name（名称），以及客户 name（名称）和订单的 unit price（单价）(total_amt_usd/total)。最终表格应该包含三列：region name（区域名称）、account name（客户名称）和 unit price（单价）。少数几个客户的总订单数为 0，因此我除以的是 (total + 0.01) 以确保没有除以 0。
   Answer: `SELECT r.name region, a.name account, o.total_amt_usd/(o.total + 0.01) unit_price FROM region r JOIN sales_reps s ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id;`

6. INNER JOIN 问题
   Answer: `SELECT c.countryid, c.countryName, s.stateName FROM Country c JOIN State s ON c.countryid = s.countryid;`

7. LEFT JOIN 问题
   Answer: `SELECT c.countryid, c.countryName, s.stateName FROM Country c LEFT JOIN State s ON c.countryid = s.countryid;`

8. 最后KEFT JOIN 注意事项
   Answer: `SELECT c.countryid, c.countryName, s.stateName FROM State s LEFT JOIN Country c ON c.countryid = s.countryid;`

9. 为每个销售代表对应的区域以及相关的客户创建一个表格，这次仅针对 Midwest 区域。最终表格应该包含三列：区域名称、销售代表姓名，以及客户名称。根据客户名称按字母顺序 (A-Z) 排序。
   Answer: `SELECT r.name region, s.name rep, a.name account FROM sales_reps s JOIN region r ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id WHERE r.name = 'Midwest' ORDER BY a.name;`

10. 为每个销售代表对应的区域以及相关的客户创建一个表格，这次仅针对 Midwest 区域，并且销售代表的名字以 S 开头。最终表格应该包含三列：区域名称、销售代表姓名，以及客户名称。根据客户名称按字母顺序 (A-Z) 排序。
    Answer: `SELECT r.name region, s.name rep, a.name account FROM sales_reps s JOIN region r ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id WHERE r.name = 'Midwest' AND s.name LIKE 'S%' ORDER BY a.name;`

11. 为每个销售代表对应的区域以及相关的客户创建一个表格，这次仅针对 Midwest 区域，并且销售代表的姓以 K 开头。最终表格应该包含三列：区域名称、销售代表姓名，以及客户名称。根据客户名称按字母顺序 (A-Z) 排序。
    Answer: `SELECT r.name region, s.name rep, a.name account FROM sales_reps s JOIN region r ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id WHERE r.name = 'Midwest' AND s.name LIKE *% K%* ORDER BY a.name;`

12. 提供每个订单的每个区域的名称，以及客户名称和所支付的单价 (total_amt_usd/total)。但是，只针对标准订单数量超过 100 的情况提供结果。最终表格应该包含三列：区域名称、客户名称和单价。为了避免除以 0 个订单，这里可以在分母上加上 0.01，即：(total_amt_usd/(total+0.01))。
    Answer: `SELECT r.name region, a.name account, o.total_amt_usd/(o.total + 0.01) unit_price FROM region r JOIN sales_reps s ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id WHERE o.standard_qty > 100;`

13. 提供每个订单的每个区域的名称，以及客户名称和所支付的单价 (total_amt_usd/total)。但是，只针对标准订单数量超过 100 且广告纸数量超过 50 的情况提供结果。最终表格应该包含三列：区域名称、客户名称和单价。按照最低的单价在最之前排序。为了避免除以 0 个订单，这里可以在分母上加上 0.01，即：(total_amt_usd/(total+0.01))。
    Answer: `SELECT r.name region, a.name account, o.total_amt_usd/(o.total + 0.01) unit_price FROM region r JOIN sales_reps s ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id WHERE o.standard_qty > 100 AND o.poster_qty > 50 ORDER BY unit_price;`

14. 提供每个订单的每个区域的名称，以及客户名称和所支付的单价 (total_amt_usd/total)。但是，只针对标准订单数量超过 100 且广告纸数量超过 50 的情况提供结果。最终表格应该包含三列：区域名称、客户名称和单价。按照最高的单价在最之前排序。为了避免除以 0 个订单，这里可以在分母上加上 0.01，即：(total_amt_usd/(total+0.01))。
    Answer: `SELECT r.name region, a.name account, o.total_amt_usd/(o.total + 0.01) unit_price FROM region r JOIN sales_reps s ON s.region_id = r.id JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id WHERE o.standard_qty > 100 AND o.poster_qty > 50 ORDER BY unit_price DESC;`

15. account id 为 1001 的客户使用了哪些不同的渠道。最终表格应该包含 2 列：客户名称和不同的渠道。你可以尝试使用 SELECT DISTINCT 使结果仅显示唯一的值。
    Answer: `SELECT DISTINCT a.name, w.channel FROM accounts a JOIN web_events w ON a.id = w.account_id WHERE a.id = '1001';`

16. 找出发生在 2015 年的所有订单。最终表格应该包含 4 列：occurred_at、account name、order total 和order total_amt_usd。
    Answer: `SELECT w.occurred_at, a.name, o.total, o.total_amt_usd FROM accounts a JOIN orders o ON o.account_id = a.id JOIN web_events w ON a.id = w.account_id WHERE w.occurred_at BETWEEN *01 01-2015' AND '01-01-2016' ORDER BY w.occurred_at DESC;`

练习三_SQL聚合

1. 算出 orders 表格中的 poster_qty 纸张总订单量。
   Answer: `SELECT SUM(poster_qty) AS total_poster_sales FROM orders;`

2. 算出 orders 表格中 standard_qty 纸张的总订单量。
   Answer: `SELECT SUM(standard_qty) AS total_standard_sales FROM orders;`

3. 根据 orders 表格中的 total_amt_usd 得出总销售额。
   Answer: `SELECT SUM(total_amt_usd) AS total_dollar_sales FROM orders;`

4. 算出 orders 表格中每个订单在 standard 和 gloss 纸张上消费的数额。结果应该是表格中每个订单的金额。
   Answer: `SELECT standard_qty + gloss_qty AS total_standard_gloss FROM orders;`

5. 每个订单的 price/standard_qty 纸张各不相同。我想得出 orders 表格中每个销售机会的这一比例。
   Answer: `SELECT SUM(standard_amt_usd)/SUM(standard_qty) AS standard_price_per_unit FROM orders;`

6. 最早的订单下于何时？
   Answer: `SELECT MIN(occurred_at) FROM orders;`

7. 尝试执行和第一个问题一样的查询，但是不使用聚合函数。
   Answer: `SELECT occurred_at FROM orders ORDER BY occurred_at LIMIT 1;`

8. 最近的 web_event 发生在什么时候？
   Answer: `SELECT MAX(occurred_at) FROM web_events;`

9. 尝试以另一种方式执行上个问题的查询，不使用聚合函数。
   Answer: `SELECT occurred_at FROM web_events ORDER BY occurred_at DESC LIMIT 1;`

10. 算出每个订单在每种纸张上消费的平均 (AVERAGE) 金额，以及每个订单针对每种纸张购买的平均数量。最终答案应该有 6 个值，每个纸张类型平均销量对应一个值，以及平均数量对应一个值。
    Answer: `SELECT AVG(standard_qty) mean_standard, AVG(gloss_qty) mean_gloss, AVG(poster_qty) mean_poster, AVG(standard_amt_usd) mean_standard_usd, AVG(gloss_amt_usd) mean_gloss_usd, AVG(poster_amt_usd) mean_poster_usd FROM orders;`

11. 我相信你都渴望知道在所有订单上消费的中值 total_usd 是多少？虽然这一概念已经超出我们的范围。注意，这比我们到目前为止介绍的基本内容要高深一点，但是我们可以按照以下方式对答案进行硬编码。
    Answer: `SELECT * FROM (SELECT total_amt_usd FROM orders ORDER BY total_amt_usd LIMIT 3457) AS Table1 ORDER BY total_amt_usd DESC LIMIT 2;`

12. 哪个客户（按照名称）下的订单最早？你的答案应该包含订单的客户名称和日期。
    Answer: `SELECT a.name, o.occurred_at FROM accounts a JOIN orders o ON a.id = o.account_id ORDER BY occurred_at LIMIT 1;`

13. 算出每个客户的总销售额（单位是美元）。答案应该包括两列：每个公司的订单总销售额（单位是美元）以及公司名称。
    Answer: `SELECT a.name, SUM(total_amt_usd) total_sales FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY a.name;`

14. 最近的 web_event 是通过哪个渠道发生的，与此 web_event 相关的客户是哪个？你的查询应该仅返回三个值：日期、渠道和客户名称。
    Answer: `SELECT w.occurred_at, w.channel, a.name FROM web_event w JOIN accounts a ON w.account_id = a.id ORDER BY w.occurred_at DESC LIMIT 1`

15. 算出 web_events 中每种渠道的次数。最终表格应该有两列：渠道和渠道的使用次数。
    Answer: `SELECT w.channel, COUNT(*) FROM web_events w JOIN accounts a ON a.id = w.account_id GROUP BY w.channel`

16. 与最早的 web_event 相关的主要联系人是谁？
    Answer: `SELECT a.primary_poc FROM web_events w JOIN accounts a ON a.id = w.account_id ORDER BY w.occurred_at LIMIT 1;`

17. 每个客户所下的最小订单是什么（以总金额（美元）为准）。答案只需两列：客户名称和总金额（美元）。从最小金额到最大金额排序。
    Answer: `SELECT a.name, MIN(total_amt_usd) smallest_order FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.name ORDER BY smallest_order;`

18. 算出每个区域的销售代表人数。最早表格应该包含两列：区域和 sales_reps 数量。从最少到最多的代表人数排序。
    Answer: `SELECT r.name, COUNT(*) num_reps FROM region r JOIN sales_reps s ON r.id = s. region_id GROUP BY r. name ORDER BY num_reps;`

19. 对于每个客户，确定他们在订单中购买的每种纸张的平均数额。结果应该有四列：客户名称一列，每种纸张类型的平均数额一列。
    Answer: `SELECT a.name, AVG(o.standard_qty) avg_stand, AVG(gloss_qty) avg_gloss, AVG(poster_qty) avg_poster FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.name;`

20. 对于每个客户，确定在每个订单中针对每个纸张类型的平均消费数额。结果应该有四列：客户名称一列，每种纸张类型的平均消费数额一列。
    Answer: `SELECT a.name, AVG(o.standard_amt_usd) avg_stand, AVG(gloss_amt_usd) avg_gloss, AVG(poster_amt_usd) avg_poster FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.name;`

21. 确定在 web_events 表格中每个销售代表使用特定渠道的次数。最终表格应该有三列：销售代表的名称、渠道和发生次数。按照最高的发生次数在最上面对表格排序。
    Answer: `SELECT s.name, w.channel, COUNT(*) num_events FROM accounts a JOIN web_events w ON a.id = w.account_id JOIN sates_reps s ON s.id = a.sales_rep_id GROUP BY s.name, w.channel ORDER BY num_events DESC;`

22. 确定在 web_events 表格中针对每个地区特定渠道的使用次数。最终表格应该有三列：区域名称、渠道和发生次数。按照最高的发生次数在最上面对表格排序。
    Answer: `SELECT r.name, w.channel, COUNT(*) num_events FROM accounts a JOIN web events w ON a.id = w.account_id JOIN sales_reps s ON s.id = a.sales_rep_id JOIN region r ON r.id = s.region_id GROUP BY r.name, w. channel ORDER BY num_events DESC;`

23. 使用 DISTINCT 检查是否有任何客户与多个区域相关联？
    Answer: 下面的两个查询产生了相同的行数（351行），因此我们知道每个客户仅与一个区域相关联。如果每个客户与多个区域相关联，则第一个查询返回的行数应该比第二个查询的多。
    `SELECT DISTINCT a.id, r.id, a.name, r.name FROM accounts a JOIN sales_reps s ON s.id = a.sales_rep_id JOIN region r ON r.id = s.region_id;`
    and
    `SELECT DISTINCT id, name FROM accounts;`

24. 有没有销售代表要处理多个客户？
    Answer: 实际上， 所有销售代表都要处理多个客户。 销售代表处理的最少客户数量是 3 个。 有 50 个销售代表，他们都有多个客户。 在第二个查询中使用 DISTINCT 确保包含了第一个查询中的所有销售代表。
    `SELECT s.id, s.name, COUNT(*) num_accounts FROM accounts a JOIN sales_reps s ON s.id = a.sales_rep_id GROUP BY s.id, s.name ORDER BY num_accounts;`
    and
    `SELECT DISTINCT id, name FROM sa"les_reps;`

25. 有多少位销售代表需要管理超过 5 个客户？
    Answer:
    `SELECT s.id, s.name, COUNT(*) num_accounts FROM accounts a JOIN sales_reps s ON s.id = a.sales_rep_id GROUP BY s.id, s.name HAVING COUNT(*) > 5 ORDER BY num_accounts;`
    实际上，我们可以使用SUBQUERY获得这一结果，如下所示。其他查询也可以使用这一逻辑，下面就不显示了。
    `SELECT COUNT(*) num_reps_above5 FROM(SELECT s.id, s.name, COUNT(*) num_accounts FROM accounts a JOIN sales_reps s ON s.id = a.sales_rep_id GROUP BY s.id, s.name HAVING COUNT(*) > 5 ORDER BY num_accounts) AS Tablel;`

26. 有多少个客户具有超过 20 个订单？
Answer: `SELECT a.id, a.name, COUNT(*) num_orders FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.id, a.name HAVING COUNT(*) > 20 ORDER BY num_orders;`

27. 哪个客户的订单最多？
Answer: `SELECT a.id, a.name, COUNT(*) num_orders FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY num_orders DESC LIMIT 1;`

28. 有多少个客户在所有订单上消费的总额超过了 30,000 美元？
Answer: `SELECT a.id, a.name, SUM(o.total_amt_usd) total_spent FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.id, a.name HAVING SUM(o.total_amt_usd) > 30000 ORDER BY total_spent;`

29. 有多少个客户在所有订单上消费的总额不到 1,000 美元？
Answer: `SELECT a.id, a.name, SUM(o.total_amt_usd) total_spent FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.id, a.name HAVING SUM(o.total_amt_usd) < 1000 ORDER BY total_spent;`

30. 哪个客户消费的最多？
Answer: `SELECT a.id, a.name, SUM(o.total_amt_usd) total_spent FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY total_spent DESC LIMIT 1;`

31. 哪个客户消费的最少？
Answer: `SELECT a.id, a.name, SUM(o.total_amt_usd) total_spent FROM accounts a JOIN orders o ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY total_spent LIMIT 1;`

32. 哪个客户使用 facebook 作为与消费者沟通的渠道超过 6 次？
Answer: `SELECT a.id, a.name, w.channel, COUNT(*) use_of_channel FROM accounts a JOIN web_events w ON a.id = w.account_id GROUP BY a.id, a.name, w.channel HAVING COUNT(*) > 6 AND w.channel = 'facebook' ORDER BY use_of_channel;`

33. 哪个客户使用 facebook 作为沟通渠道的次数最多？
Answer: `SELECT a.id, a.name, w.channel, COUNT(*) use_of_channel FROM accounts a JOIN web_events w ON a.id = w.account_id WHERE w.channel = 'facebook' GROUP BY a.id, a.name, w.channel`

34. 哪个渠道是客户最常用的渠道？
Answer: `SELECT a.id, a.name, w.channel, COUNT(*) use_of_channel FROM accounts a JOIN web events w ON a.id= w.account id GROUP BY a.id, a.name, w.channel ORDER BY use of channel DESC LIHIT 10;`

35. Parch & Posey 在哪一年的总销售额最高？数据集中的所有年份保持均匀分布吗？
Answer: `SELECT DATE_PART('year', occurred_at) ord_year, SUM(total_amt_usd) total_spent FROM orders GROUP BY 1 ORDER BY 2 DESC;`

36. Parch & Posey 在哪一个月的总销售额最高？数据集中的所有月份保持均匀分布吗？
Answer: `SELECT DATE_PART('month', occurred_at) ord_month, SUM(total_amt_usd) total_spent FROM orders WHERE occurred at BETWEEN '2014-01-01' AND '2017-01-01' GROUP BY 1 ORDER BY 2 DESC;`

37. Parch & Posey 在哪一年的总订单量最多？数据集中的所有年份保持均匀分布吗？
Answer: `SELECT DATE_PART('year', occurred_at) ord_year, COUNT(*) total_sales FROM orders GROUP BY 1 ORDER BY 2 DESC;`

38. Parch & Posey 在哪一个月的总订单量最多？数据集中的所有月份保均匀分布吗？
Answer: `SELECT DATE_PART('month', occurred_at) ord_month, COUNT(*) total_sales FROM orders WHERE occurred at BETWEEN '2014-01-01' AND '2017-01-01' GROUP BY 1 ORDER BY 2 DESC;`

39. Walmart 在哪一年的哪一个月在铜版纸上的消费最多？
Answer: `SELECT DATE_TRUNC('month', o.occurred_at) ord_date, SUM(o.gloss_amt_usd) tot_spent FROM orders o JOIN accounts a ON a.id = o.account_id WHERE a.name = 'Walmart' GROUP BY 1 ORDER BY 2 DESC LIMIT 1;`

40. 我们想要根据相关的消费量了解三组不同的客户。最高的一组是终身价值（所有订单的总销售额）大于200,000 美元的客户。第二组是在 200,000 到 100,000 美元之间的客户。最低的一组是低于 under100,000 美元的客户。请提供一个表格，其中包含

与每个客户相关的级别。你应该提供客户的名称、所有订单的总销售额和级别。消费最高的客户列在最上面。

Answer: `SELECT a.name, SUM(total_amt_usd) total_spent, CASE WHEN SUM(total_amt_usd) > 200000 THEN 'top' WHEN SUM(total_amt_usd) > 100000 THEN 'middle' ELSE 'low' END AS customer_level FROM orders o JOIN accounts a ON o.accounted = a.id GROUP BY a.name ORDER BY 2 DESC`

41. 现在我们想要执行和第一个问题相似的计算过程，但是我们想要获取在 2016 年和 2017 年客户的总消费数额。级别和上一个问题保持一样。消费最高的客户列在最上面。

    Answer: `SELECT a.name, SUM(total_amt_usd) total_spent, CASE WHEN SUM(total_amt_usd) > 200000 THEN 'top' WHEN SUM(total_amt_usd) > 100000 THEN 'middle' ELSE 'low' END AS customer_level FROM orders o JOIN accounts a ON o.accounted = a.id WHERE occurred.at > '2015-12-31' GROUP BY 1 ORDER BY 2 DESC;`

42. 我们想要找出绩效最高的销售代表，也就是有超过 200 个订单的销售代表。创建一个包含以下列的表格：销售代表名称、订单总量和标为 top 或 not 的列（取决于是否拥有超过 200 个订单）。销售量最高的销售代表列在最上面。

    Answer: `SELECT s.name, COUNT(*) num_ords,CASE WHEN COUNT(*) > 200 THEN 'top' ELSE 'not' END AS sales_rep_level FROM orders o JOIN accounts a ON o.accounted = a.id JOIN sales_reps s ON s.id = a.sales_rep_id GROUP BY s.name ORDER BY 2 DESC;`

43. 之前的问题没有考虑中间水平的销售代表或销售额。管理层决定也要看看这些数据。我们想要找出绩效很高的销售代表，也就是有超过 200 个订单或总销售额超过 750000 美元的销售代表。中间级别是指有超过 150 个订单或销售额超过 500000 美元的销售代表。创建一个包含以下列的表格：销售代表名称、总订单量、所有订单的总销售额，以及标为 top、middle 或 low 的列（取决于上述条件）。在最终表格中将销售额最高的销售代表列在最上面。根据上述标准，你可能会见到几个表现很差的销售代表！

    Answer: `SELECT s.name, COUNT(*), SUM(o.total_amt_usd) total_spent, CASE WHEN COUNT(*) > 200 OR SUM(o.total_amt_usd) > 750000 THEN 'top' WHEN C0UNT(*) > 150 OR SUM(o.total_amt_usd) > 500000 THEN 'middle' ELSE 'low' END AS sales_rep_level FROM orders o JOIN accounts a ON o.accounted = a.id JOIN sales_reps s ON s.id = a.sales_rep_id GROUP BY s.name ORDER BY 3 DESC;`

练习四_SQL子查询

一、你的首个子查询

1. 首先，我们需要按照日期和渠道分组。然后按事件数(第三列)排序，这样可以快速得出第一个问题的答案。
   Answer: `SELECT DATE_TRUNC('day', occurred_at) AS day, channel, COUNT(*) as events FROM web_events GROUP BY 1,2 ORDER BY 3 DESC;`

2. 可以看出，要获得这一结果，提供了整个原始表格。查询的附加部分包括 *，并且我们需要为表格设置别名。此外，是在SELECT语句中(而不是FROM)中提供表格。
   Answer: `SELECT * FROM (SELECT DATE_TRUNC('day', occurred_at) AS day, channel, C0UNT(*) as events FROM web_events GROUP BY 1,2 ORDER BY 3 DESC) sub;`

3. 最后，我们在以下语句中能够获得显示每个渠道一天的平均事件数的表格。
   Answer: `SELECT channel, AVG(events) AS average_events FROM (SELECT DATE_TRUNC('day' occurred_at) AS day, channel, C0UNT(*) as events FROM web_events GROUP BY 1,2) sub GROUP BY channel ORDER BY 2 DESC;`

二、 练习：关于子查询的更多内容

4. 以下是从orders表格中获取第一个订单的年/月信息的查询。
   Answer: `SELECT DATE_TRUNC('month', MIN(occurred_at)) FROM orders;`

5. 然后，为了获取每个订单的平均值，我们可以在一个查询中执行所有的任务。但是为了便于阅读，我在下面提供了两个查询，单独执行每一步。
   Answer:
   `SELECT AVG(standard_qty) avg_std, AVG(gloss_qty) avg_gls, AVG(poster_qty) avg_pst FROM orders WHERE DATE_TRUNC('month', occurred_at) = (SELECT DATE_TRUNC('month', MIN(occurred_at)) FROM orders);`
   `SELECT SUM(total_amt_usd) FROM orders WHERE DATE_TRUNC('month', occurred_at) = (SELECT DATE_TRUNC('month', MIN(occurred_at)) FROM orders);`

三、 爱上子查询

6. 提供每个区域拥有最高销售额 (total_amt_usd) 的销售代表的姓名。
   Answer:
   首先，我要算出与每个销售代表徊关的总销售额（total_amt_usd），并旦要得出他们所在的区域。 以下查询提供了这一信息。
   `SELECT s.name rep_name, r.name region_name, SUM（o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.aceount id = a.id JOIN region r ON r.id = s.region_id GROUP BY 1,2 ORDER BY 3 DESC;`
   接着， 得出每个区域的最高销售额， 然后使用该信息从最终结果中获取这些行。
   `SELECT region_name, MAX(total_amt) total_amt FROM(SELECT s.name rep_name, r.name region_name, SUM(o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account id = a.id JOIN region r ON r.id = s.region_id GROUP BY 1, 2) inner1 GROUP BY 1;`
   本质上，这是两个表格的连接，其中区域和销售额相匹配。
   `SELECT tl.rep_name, tl.region_name, tl.total_amt FROM(SELECT s.name rep_name, r.name region_name, SUM(o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account id = a.id JOIN region r ON r.id = s.region_id GROUP BY 1,2 ORDER BY 3 DESC) tl JOIN (SELECT region_name, MAX(total_amt) total_amt FROM(SELECT s.name rep_name, r.name region_name, SUM(o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id JOIN region r ON r.id = s. region_id GROUP BY 1, 2 inner 1 GROUP BY 1) t2 ON t1.region_name = t2.region_name AND t1.total_amt = t2.total_amt;`

7. 对于具有最高销售额 (total_amt_usd) 的区域，总共下了多少个订单（total count orders）？
   Answer:
   我写的第一个查询是获取每个区域的total_amt_usd。
   `SELECT r.name region_name, SUM（o.total_amt_usd) total_amt FROM sates_reps s JOIN accounts a ON a.sales_rep_id = s.id`

```
JOIN orders o ON o.account id = a.id JOIN region r ON r.id = s.region_id GROUP BY r.name;
```
然后， 我们仅从该表格中获取销售额最高的区域。 可以通过两种方法来获取， 一种是使用子查询后的最大值，另一种是按降序排序，然后获取最高值。
```
SELECT MAX(total_amt) FROM (SELECT r.name region_name, SUM(o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sates_rep_id = s.id JOIN orders o ON o.account id = a.id JOIN region r ON r.id = s.region_id GROUP BY r.name) sub;
```
最终, 我们要获取具有该区域销售额的总订单量:
```
SELECT r.name, SUM(o.total) total_orders FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id JOIN region r ON r.id = s.region_id GROUP BY r.name HAVING SUM(o.total_amt_usd) = (SELECT MAX(total_amt) FROM (SELECT r.name region_name, SUM(o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id JOIN region r ON r.id = s.region_id GROUP BY r.name) sub;
```
结果就是 Northeast， 总订单为 1230378 个。

8. 对于购买标准纸张数量 (standard_qty) 最多的客户（在作为客户的整个时期内），有多少客户的购买总数依然更多？
   Answer:
   首先， 我们要得出购买标准纸张数量（standard_qty）最多的客户。 以下查询获取了该客户，以及总消费：
   ```
   SELECT a.name account_name, SUM(o.standard_qty) total_std, SUM（o.total) total FROM accounts a JOIN orders o ON o.account_id = a.id GROUP BY 1 ORDER BY 2 DESC LIMIT 1;
   ```
   现在， 我将使用上述信息获取总消费更高的所有客户：
   ```
   SELECT a.name FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY 1 HAVING SUM(o.total) > (SELECT total FROM (SELECT a.name act_name, SUM(o.standard_qty) total_std, SUM(o.total) total FROM account a JOIN orders o ON o.account_id = a.id GROUP BY 1 ORDER BY 2 DESC LIMIT 1) sub);
   ```
   上述查询列出了具有更多订单的客户列表。我们还可以使用另一个简单的子查询获取数量。
   ```
   SELECT COUNT(*) FROM (SELECT a.name FROM orders o JOIN accounts a ON a.id = o.accountid GROUP BY 1 HAVING SUM(o.total) > (SELECT total FROM (SELECT a.name act_name, SUM(o.standard_qty) tot_std, SUM(o.total) total FROM accounts a JOIN orders o ON o.account_id = a.id GROUP BY 1 ORDER BY 2 DESC LIMIT 1) inner_tab) counter_tab;
   ```

9. 对于（在作为客户的整个时期内）总消费 (total_amt_usd) 最多的客户，他们在每个渠道上有多少w eb_events？
   Answer:
   我们首先需要获取在整个客户时期内消费最多的客户。
   ```
   SELECT a.id, a.name, SUM(o.total_amt_usd) tot_spent FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY 3 DESC LIMIT 1;
   ```
   现在，我们要获取该企业(可以使用id进行匹配)在每个渠道上的事件数。
   ```
   SELECT a.name, w.channel, COUNT(*) FROM accounts a JOIN web_events w ON a.id = w.account_id AND a.id = (SELECT id FROM (SELECT a.id, a.name, SUM(o.total_amt_usd) tot_spent FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY 3 DESC LIMIT 1) inner_table) GROUP BY 1, 2 ORDER BY 3 DESC;
   ```

10. 对于总消费前十名的客户，他们的平均终身消费 (total_amt_usd) 是多少?
    Answer:
    现在，我们需要找出总消费(total_amt_usd)在前十名的客户。
    ```
    SELECT a.id, a.name, SUM(o.total_amt_usd) tot_spent FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY 3 DESC LIMIT 10;
    ```
    现在计算这十个客户的平均消费。
    ```
    SELECT AVG(tot_spent) FROM (SELECT a.id, a.name, SUM(o.total_amt_usd) tot_spent FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY 3 DESC LIMIT 10) temp;
    ```

11. 比所有客户的平均消费高的企业平均终身消费 (total_amt_usd) 是多少？
    Answer:
    首先计算出所有客户的总消费(total_amt_usd) 平均值：
    ```
    SELECT AVG(o.totat_amt_usd) avg_all FROM orders o JOIN accounts a ON a.id = o.account_id;
    ```
    然后，只获取高于这一平均值的客户。
    ```
    SELECT o.account_id, AVG(o.total_amt_usd) FROM orders o GROUP BY 1 HAVING AVG(o.total_amt_usd) > (SELECT AVG(o.total_amt_usd) avg_all FROM orders o JOIN accounts a ON a.id = o.account_id);
    ```
    最后，算出这些值的平均值。
    ```
    SELECT AVG(avg_amt) FROM (SELECT o.account_id, AVG(o.total_amt_usd) avg_amt FROM orders o GROUP BY 1 HAVING AVG(o.total_amt_usd) > (SELECT AVG(o.total_amt_usd) avg_all FROM orders o JOIN accounts a ON a.id = o.account_id)) temp_table;
    ```

四、练习：WITH与子查询

12. 提供每个区域拥有最高销售额 (total_amt_usd) 的销售代表的姓名。
Answer: `WITH tl AS (SELECT s.name rep_name, r.name region_name, SUM（o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id JOIN region r ON r.id = s.region_id GROUP BY 1,2 ORDER BY 3 DESC), t2 AS (SELECT region_name, MAX(total_amt) total_amt FROM tl GROUP BY 1) SELECT tl.rep_name, tl.region_name, tl.total_amt FROM tl JOIN t2 ON tl.region_name = t2.region_name AND tl.total_amt = t2.total_amt;`

13. 对于具有最高销售额 (total_amt_usd) 的区域，总共下了多少个订单？
Answer: `WITH t1.AS (SELECT r.name region_name, SUM(o.total_amt_usd) total_amt FROM sales_reps s JOIN accounts a ON a.sales_rep_id = s.id JOIN orders o ON o.account_id = a.id JOIN region r ON r.id = s.region_id GROUP BY r.name), t2 AS SELECT MAX(total_amt) FROM t1) SELECT r.name, SUM(o.total) total_orders FROM sales_reps s JOIN accounts a ON a.sa1es_rep_id = s.id JOIN orders o ON o.account_id = a.id JOIN region r ON r.id = s.region_id GROUP BY r.name HAVING SUM(o.total_amt_usd) = (SELECT * FROM t2);`

14. 对于购买标准纸张数量 (standard_qty) 最多的客户（在作为客户的整个时期内），有多少客户的购买总数（total）比该用户的购买总数（total）更多？
Answer: `WITH t1 AS SELECT a.name account_name, SUM(o.standard_qty) total_std, SUM(o.total) total FROM accounts a JOIN orders o ON o.account_id = a.id GROUP BY 1 ORDER BY 2 DESC LIMIT 1), t2 AS (SELECT a.name FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY 1 HAVING SUM(o.total) > (SELECT total FROM t1)) SELECT COUNT(*) FROM t2;`

15. 对于（在作为客户的整个时期内）总消费 (total_amt_usd) 最多的客户，他们在每个渠道上有多少web_events？
Answer: `WITH t1 AS SELECT a.id, a.name, SUM(o.total_amt_usd) tot_spent FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY 3 DESC LIMIT 1) SELECT a.name, w.channel, COUNT(*) FROM accounts a JOIN web_events w ON a.id = w.account_id AND a.id = (SELECT id FROM t1) GROUP BY 1, 2 ORDER BY 3 DESC;`

16. 对于总消费前十名的客户，他们的平均终身消费 (total_amt_usd) 是多少?
Answer: `WITH tl AS (SELECT a.id, a.name, SUM(o.total_amt_usd) tot_spent FROM orders o JOIN accounts a ON a.id = o.account_id GROUP BY a.id, a.name ORDER BY 3 DESC LIMIT 10) SELECT AVG(tot_spent) FROM tl;`

17. 比所有客户的平均消费高的企业平均终身消费 (total_amt_usd) 是多少？
Answer: `WITH tl AS SELECT AVG(o.total_amt_usd) avg_all FROM orders o JOIN accounts a ON a.id = o.account_id), t2 AS SELECT o.account_id, AVG(o.total_amt_usd) avg_amt FROM orders o GROUP BY 1 HAVING AVG(o.total_amt_usd) > (SELECT * FROM tl)) SELECT AVG(avg_amt) FROM t2;`