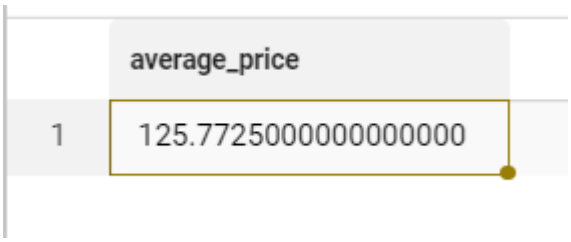
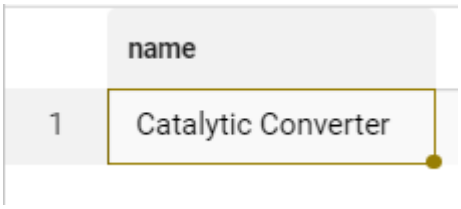
RevRoll:

1. **What's the average price of a part?**
   SELECT AVG(price) AS average_price
   FROM parts;



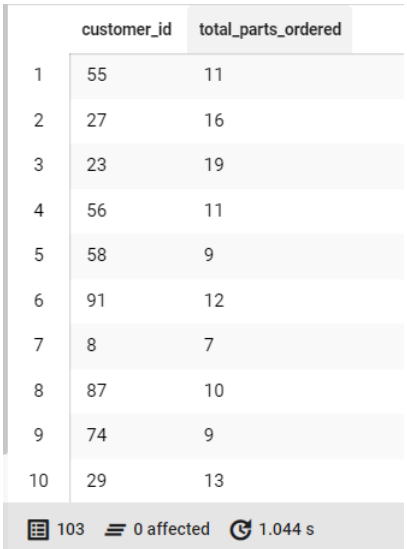
	average_price
1	125.7725000000000000

2. **What's the name of the most expensive part?**
   SELECT name
   FROM parts
   ORDER BY price DESC
   LIMIT 1;



	name
1	Catalytic Converter

3. **How many parts did every customer order?**
   SELECT customer_id, SUM(quantity) AS total_parts_ordered
   FROM orders
   GROUP BY customer_id;



	customer_id	total_parts_ordered
1	55	11
2	27	16
3	23	19
4	56	11
5	58	9
6	91	12
7	8	7
8	87	10
9	74	9
10	29	13

103 0 affected 1.044 s

4. **What is the average time to install a derby?**
   SELECT AVG(installer_one_time + installer_two_time) AS average_install_time
   FROM install_derby;

	average_install_time
1	76.6444444444444444

5. **Find the top 5 installers with the highest average installation time.**

```
SELECT i.installer_id, i.name, AVG(installer_one_time + installer_two_time) AS
average_install_time
FROM installers  i
JOIN install_derby id
ON i.installer_id = id.installer_one_id OR i.installer_id = id.installer_two_id
GROUP BY i.installer_id, i.name
ORDER BY average_install_time DESC
LIMIT 5;
```

	installer_id	name	average_install_time
1	5	Clint Clutch	85.7777777777777778
2	7	Mikey Muffler	85.0000000000000000
3	11	Diesel	79.1111111111111111
4	3	Bolt	78.6666666666666667
5	9	Axle	78.2222222222222222

6. **Which installers have participated in at least 1 derby?**

```
SELECT DISTINCT
   i.name AS name
FROM
   installers i
INNER JOIN
   install_derby ins ON i.installer_id = ins.installer_one_id
   OR
   i.installer_id = ins.installer_two_id
WHERE
   ins.installer_one_time IS NOT NULL OR ins.installer_two_time IS NOT NULL;
```

	name
1	Diesel
2	Mikey Muffler
3	Axle
4	Sparky
5	Clint Clutch
6	Gearhead Gus
7	Turbo
8	Bolt
9	Dollar Bill Brakes
10	Piston Pete

10 0 affected 0.998 s

7. **Report the **most expensive** part in each order.**
Only include installed orders. In case of a tie, report all parts with the maximum price.
Order by order_id and limit the output to 5 rows.

```
WITH OrderPartPrices AS (
  SELECT
     o.order_id,
     p.part_id,
     p.price,
     RANK() OVER (PARTITION BY o.order_id ORDER BY p.price DESC) AS
price_rank
  FROM
     orders o
  INNER JOIN
     parts p ON o.part_id = p.part_id
  INNER JOIN
     installs i ON o.order_id = i.order_id
)
SELECT
  order_id,
  part_id
FROM
  OrderPartPrices
WHERE
  price_rank = 1
ORDER BY
  order_id
Limit 5;
```

	order_id	part_id
1	1	8
2	6	19
3	9	14
4	9	14
5	10	17

8. **Find the installers who have completed installations for at least four consecutive days.**
Include the `installer_id`, start date of the consecutive installations period and the end date of the consecutive installations period. Return the result table ordered by `installer_id` in ascending order.

```
WITH CTE1 AS(
   SELECT
      *,
      LAG(install_date) OVER(PARTITION BY installer_id ORDER BY install_date)
AS pre_install_date
   FROM
      installs
   ORDER BY
      installer_id, install_date
),
CTE2 AS(
   SELECT
      installer_id,
                              install_date,
  coalesce(pre_install_date,install_date) as pre_install_date
      ,install_date - INTERVAL '1 day' * (ROW_NUMBER() OVER(PARTITION BY
installer_id ORDER BY install_date)) AS GRP,
  (ROW_NUMBER() OVER(PARTITION BY installer_id ORDER BY install_date)) AS
rn
   FROM CTE1
WHERE install_date - pre_install_date = 1
)
SELECT
   installer_id,
   MIN(pre_install_date) AS consecutive_start,
   MAX(install_date) AS consecutive_end
FROM
   CTE2
GROUP BY
   installer_id, GRP
HAVING COUNT(GRP) > 2;
```

	installer_id	consecutive_start	consecutive_end
1	3	2023-03-28	2023-03-31
2	4	2023-08-15	2023-08-18
3	7	2023-12-20	2023-12-23

9. **Installers receive performance based year end bonuses. Bonuses are calculated by taking 10% of the total value of parts installed by the installer. Calculate the bonus earned by each installer rounded to a whole number. Sort the result by bonus in increasing order.**

```
WITH names AS(
    SELECT name, i.installer_id
    FROM installers i
),
orders AS(
    SELECT o.order_id, ins.installer_id, SUM(p.price * quantity) AS total_value
    FROM orders o
    INNER JOIN installs ins ON o.order_id = ins.order_id
    INNER JOIN parts p ON o.part_id = p.part_id
    GROUP BY o.order_id, ins.installer_id
),
bonuses AS(
    SELECT n.name, n.installer_id, SUM(o.total_value) AS total_value, CAST(0.1 *
SUM(o.total_value) AS INTEGER) AS bonus
    FROM names n
    INNER JOIN orders o ON n.installer_id = o.installer_id
    GROUP BY n.name, n.installer_id
)

SELECT name, bonus
FROM bonuses
ORDER BY bonus;
```

	name	bonus
1	Axle	792
2	Piston Pete	867
3	Turbo	959
4	Bolt	1018
5	Dollar Bill Brakes	1028
6	Gearhead Gus	1134
7	Sparky	1134
8	Mikey Muffler	1159
9	Clint Clutch	1184
10	Willy the Wrench	1247

11 0 affected 0.965 s

10. **RevRoll encourages healthy competition. The company holds a “Install Derby” where installers face off to see who can change a part the fastest in a tournament style contest.**

    Derby points are awarded as follows:

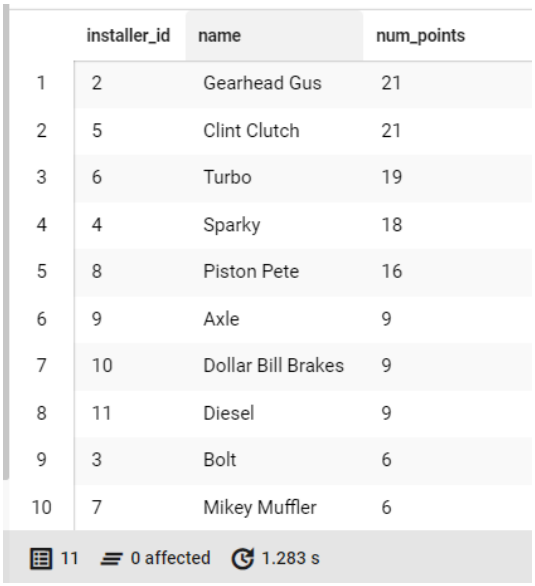
    - An installer receives three points if they win a match (i.e., Took less time to install the part).
    - An installer receives one point if they draw a match (i.e., Took the same amount of time as their opponent).
    - An installer receives no points if they lose a match (i.e., Took more time to install the part).

    We need to calculate the scores of all installers after all matches. Return the result table ordered by `num_points` in decreasing order.
    In case of a tie, order the records by `installer_id` in increasing order.

```
with installer1_outcomes as
(
select
   installer_one_id as installer_id,
   case when installer_one_time < installer_two_time then 3
   when installer_one_time = installer_two_time then 1
   else 0 end as points
   from install_derby
),
installer2_outcomes as
(
select
   installer_two_id as installer_id,
   case when installer_two_time < installer_one_time then 3
   when installer_one_time = installer_two_time then 1
   else 0 end as points
   from install_derby

),
combo as
(
select
   *
from
   installer1_outcomes
union all
select
   *
from
   installer2_outcomes
)
select
   i.installer_id,
   i.name,
```

```
    coalesce(sum(c.points),0) as num_points
from
    installers i
left join
    combo c
on
    i.installer_id = c.installer_id
group by
    i.installer_id,
    i.name
order by
    num_points desc,
    installer_id;
```



	installer_id	name	num_points
1	2	Gearhead Gus	21
2	5	Clint Clutch	21
3	6	Turbo	19
4	4	Sparky	18
5	8	Piston Pete	16
6	9	Axle	9
7	10	Dollar Bill Brakes	9
8	11	Diesel	9
9	3	Bolt	6
10	7	Mikey Muffler	6

11 0 affected 1.283 s

11. **<u>find the fastest install time with its corresponding `derby_id` for each installer. In case of a tie, you should find the install with the smallest `derby_id`.</u>**

```
WITH Install AS (
    SELECT derby_id, installer_id, install_time
    FROM (
        SELECT derby_id, installer_one_id AS installer_id, installer_one_time AS
install_time FROM install_derby
        UNION ALL
        SELECT derby_id, installer_two_id AS installer_id, installer_two_time AS
install_time FROM install_derby
    ) AS unpivoted
),
RankedInstall AS (
    SELECT
        derby_id,
```

```
        installer_id,
        install_time,
        ROW_NUMBER() OVER (PARTITION BY installer_id ORDER BY install_time,
derby_id) AS row_num
    FROM
        Install
)
SELECT
    derby_id,
    installer_id,
    install_time
FROM
    RankedInstall
WHERE
    row_num = 1
ORDER BY
    installer_id;
```

	derby_id	installer_id	install_time
1	12	2	22
2	26	3	23
3	32	4	20
4	28	5	21
5	45	6	26
6	48	7	34
7	46	8	24
8	25	9	23
9	33	10	25
10	34	11	22

10 0 affected 0.984 s