Chapter 7

Averages

Contents

1 The Trouble with Averages

- A common difficulty of working with data is being precise when asking a question. For example, consider that you have a table which contains information from a bank. Each row contains monthly information about a bank customer, including their balance, if they own a home and some other demographic information. We could, seemingly, answer the following questions:
  - What is the average bank account size for people with more than $2,500 in their account?
  - What is the average bank account size for people who own their own home?

  However, these questions *aren't* that well-defined since bank customers move through time and their characteristics change. What happens to a customer who sells their home in the middle of the year – Do you include them in the second question above? What if a person's bank balance changes over time – do you include them in the first question?

- Let's consider a specific example:

  > "What is the average number of rows per plaza with a vehicles cash per hour greater than 500?"

  In this case, there are a number of different ways that we can answer the question:

  1. **Any Row:** We only include those rows which have a vehicles cash per hour greater than 1,000:

```
select avg( ct ) as avgct
from
    (select
        count(1) as ct
    from
        cls.mta
    where vehiclescash > 500
    group by plaza ) as innerQ;


  avgct
-------
  22432
```

  2. **One Time:** We include all information from a plaza if it ever crosses the boundary:

```
select avg( ct ) as avgct
from
    (select
        max( vehiclescash) as maxcash
        , count(1) as ct
        , plaza
    from
        cls.mta
    group by plaza ) as innerQ
where maxcash > 500;


  avgct
-------
 116573
```

3. **Always:** We include all counties which always have a vehiclescash greater than 500:

```
select avg( ct ) as avgct
from
    (select
        min( vehiclescash) as mincash
        , count(1) as ct
        , plaza
    from
        cls.mta
    group by plaza ) as innerQ
where mincash > 500;


avgct
-------
```

Note that this does not return anything since no plaza fulfills this critera

4. **On Average:** We can include all counties which have, on average, vehicles cash greater than 1,000:

```
select avg( ct ) as avgct
from
    (select
        avg( vehiclescash) as avgcash
        , count(1) as ct
        , plaza
    from
        cls.mta
    group by plaza ) as innerQ
where avgcash > 500;


  avgct
-------
  92232
```

The three previous queries are all equally correct interpretations of the question above. Since the question did not adequately define the terms used reasonable people can come to different answers. The moral of the story is that SQL requires a level of precision not generally found when discussing data. Be careful!

2 HAVING

- If we just want to return the total number of rows for each county that fulfills the previous three conditions? To do this we can use the HAVING clause, which works like a WHERE clause, but is evaluated *after* the GROUP BY. It uses the same column groupings as the GROUP BY clause.

  The HAVING clause is written after the GROUP BY, but before LIMIT, if there is a LIMIT.

  Let's return the raw data from some of the examples above:

  1. **One Time:** We include all information from a plaza if it ever crosses the boundary:

```
select
    max( vehiclescash) as maxcash
    , count(1) as ct
    , plaza
from
    cls.mta
group by plaza
having max(vehiclescash) > 500;


  maxcash      ct    plaza
---------  ------  -------
     1352  122976        1
     1040  122976        2
     1594  122976        3
     1368  120624        4
      674  122976        5
[...]
```

2. **Always:** We include all counties which always have a vehiclescash greater than 500:

```
select
    plaza
    , count(1) as ct
from
    cls.mta
group by plaza
having min(vehiclescash) > 500;


plaza    ct
-------  ----
```

Note that in this example we did not explicitly include the MIN in the SELECT statement.

3. **On Average:** We can include all counties which have, on average, vehicles cash greater than 1,000:

```
select
    plaza,
    count(1) as ct
from
    cls.mta
group by plaza
having avg(vehiclescash) > 500;


  plaza      ct
-------  ------
      3  122976
     11   61488
```

- In each of the examples above, only those counties which fulfill the aggregation criteria set forth in the HAVING clause are returned.

3 COALESCE and NVL

- There is a special CASE statement that is frequently used to handle null values, called COALESCE.[1] COALESCE returns the first non-Null value it encounters. Consider the following example data:

Table 7.1: Table with missing values: tab_missing

SID	phone1	phone2
1	(111) 123 4567	(222) 123 4567
2		(333) 123 4567
3	(444) 123 4567	
4		

We could run the following queries on this:

[1] In Oracle the statement is NVL.

```
select
    coalesce( phone1, phone2) as phone, SID
from
    tab_missing;

    phone       |  SID
----------------+------
(111) 123 4567 | 1
(333) 123 4567 | 2
(444) 123 4567 | 3
               | 4
```

You can see that for each row the query returns the first non-null value it finds. For the fourth SID, however, all values are Null and a Null is returned.