```
csc343h-dianeh=> select * from runnymede;
 name  | age | grade
-------+-----+-------
 diane |     |     8
 will  |     |     8
 cate  |     |     1
 tom   |     |
 micah |     |     1
 grace |     |     2
(6 rows)

-- When we aggregate on a column, nulls in that column are ignored.
-- But count(*) counts tuples, and it includes every one, regardless of any nulls.

csc343h-dianeh=> select min(grade), max(grade), sum(grade), avg(grade), count(grade), count(*)
csc343h-dianeh-> from runnymede ;

 min | max | sum |        avg         | count | count
-----+-----+-----+--------------------+-------+-------
   1 |   8 |  20 | 4.0000000000000000 |     5 |     6
(1 row)

-- What if every value is null in the column we're aggregating on?
-- postgreSQL reports null when it hasn't a clue (i.e., for min, max, sum and avg).
-- count(*) can still give the same answer as before.
-- count(age) gives 0 because none are non-null.

csc343h-dianeh=> select min(age), max(age), sum(age), avg(age), count(age), count(*)
csc343h-dianeh-> from runnymede;

 min | max | sum | avg | count | count
-----+-----+-----+-----+-------+-------
     |     |     |     |     0 |     6
(1 row)

-- Remember that select is the last thing done.  Here we prune rows *before*
-- computing the aggregations.

csc343h-dianeh=> select min(grade), max(grade), sum(grade), avg(grade), count(grade), count(*)
csc343h-dianeh-> from runnymede
csc343h-dianeh-> where name > 'cate';

 min | max | sum |        avg         | count | count
-----+-----+-----+--------------------+-------+-------
   1 |   8 |  19 | 4.7500000000000000 |     4 |     5
(1 row)
```