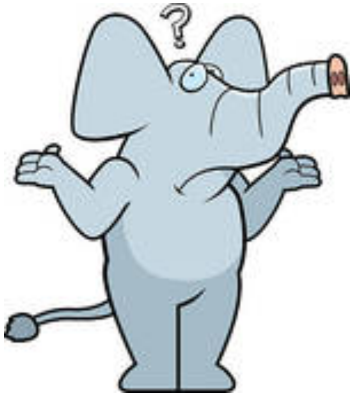
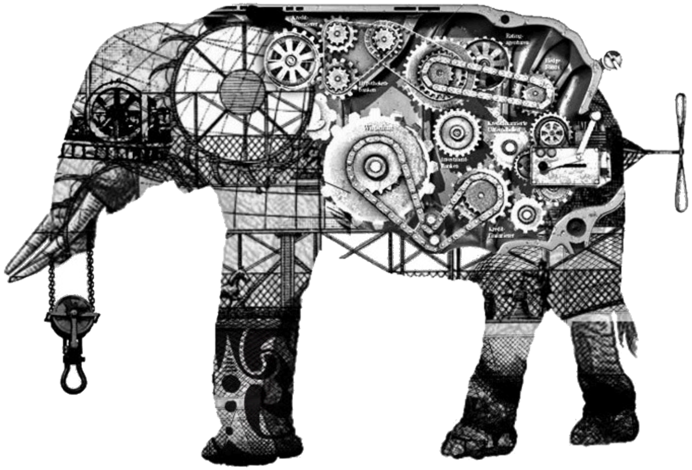
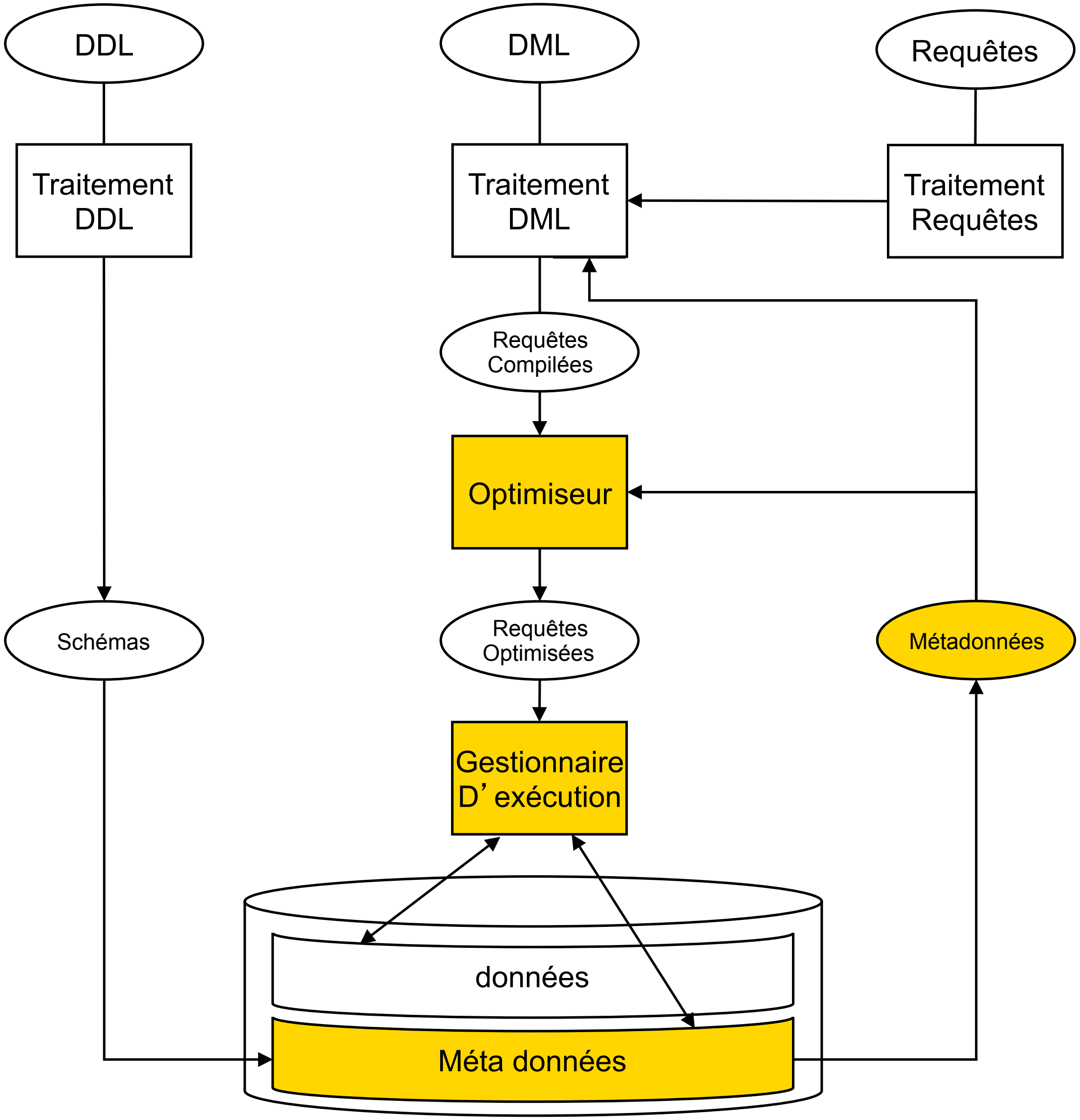


Conception Avancée de Bases de Données

pg_stats



(C) Diego Mazzeo



D’après C.J DATE

DDL : langage de définition des données; DML : langage de manipulation des données

PostgreSQL's statistics collector

- PostgreSQL's statistics collector is a subsystem that supports collection and reporting of information about server activity.

- The collector can count accesses to tables and indexes in both disk-block and individual-row terms.

Postgresql documentation

- http://www.postgresql.org/docs/9.2/static/view-pg-stats.html

- The view pg_stats provides access to the information stored in the pg_statistic catalog.

- This view allows access only to rows of pg_statistic that correspond to tables the user has permission to read, and therefore it is safe to allow public read access to this view.

pg_statistic

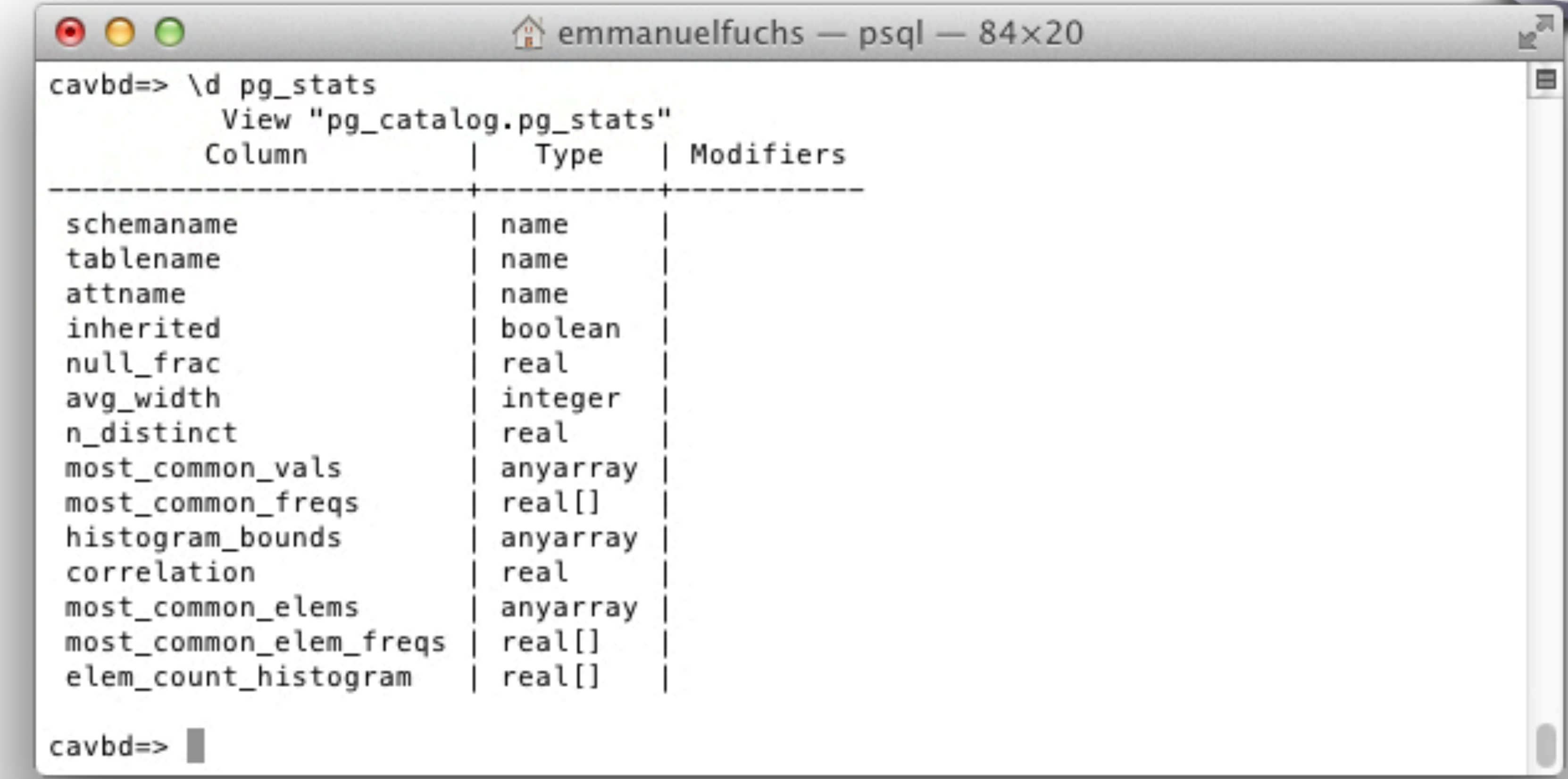
- The catalog pg_statistic stores statistical data about the contents of the database.
- Entries are created by ANALYZE and subsequently used by the query planner.
- There is one entry for each table column that has been analyzed.
- Note that all the statistical data is inherently approximate, even assuming that it is up-to-date.

pgstat_stat_interval

- Each individual server process transmits new block and row access counts to the collector just before going idle;
- So a query or transaction still in progress does not affect the displayed totals.
- Also, the collector itself emits a new report at most once per pgstat_stat_interval milliseconds (500 by default). So the displayed information lags behind actual activity.
- Current-query information is reported to the collector immediately, but is still subject to the pgstat_stat_interval delay before it becomes visible.

http://www.postgresql.org/docs/8.0/interactive/monitoring-stats.html

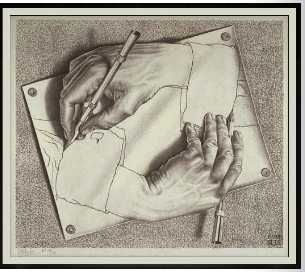
pg_stats

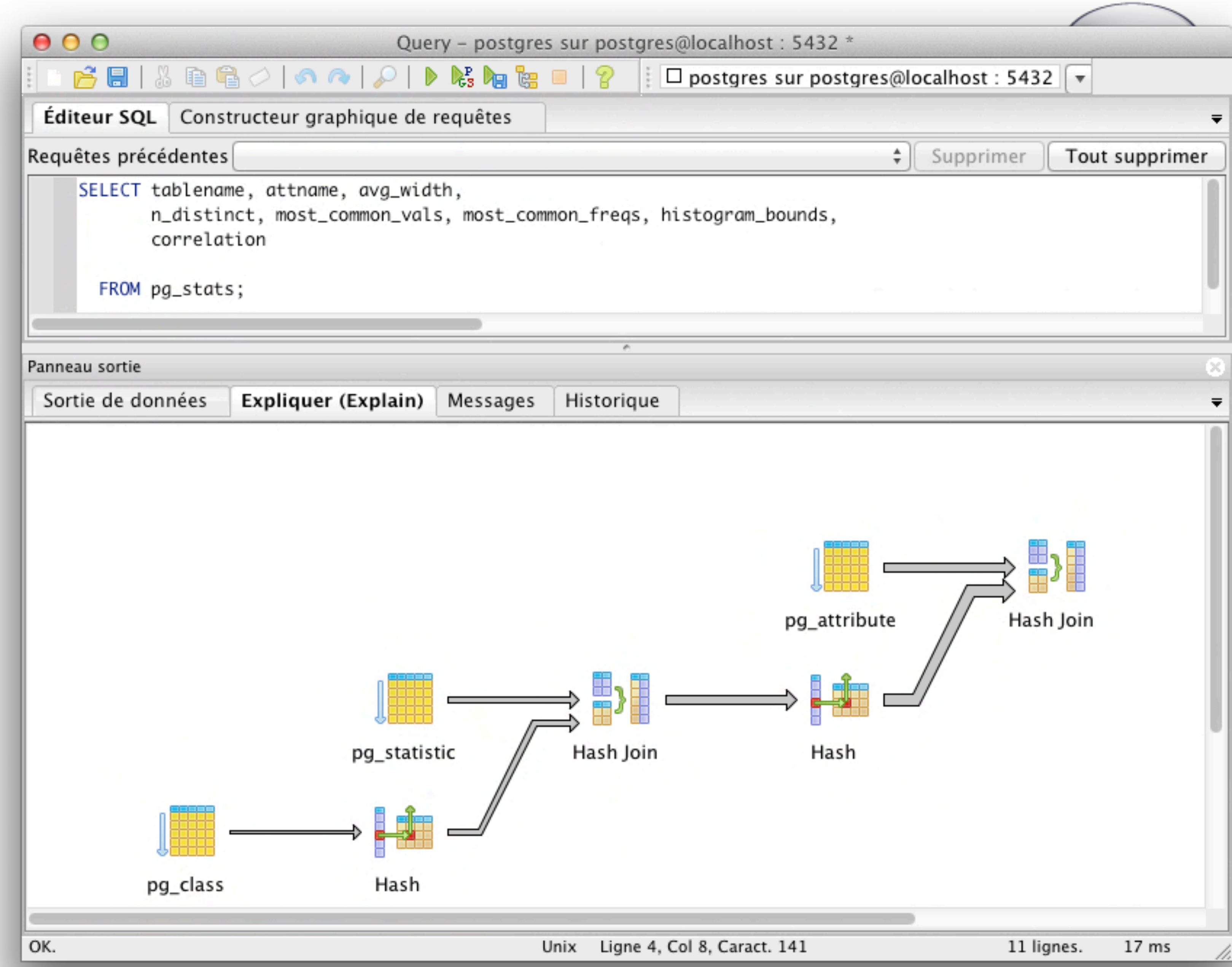


```
cavbd=> \d pg_stats
          View "pg_catalog.pg_stats"
         Column         |   Type   | Modifiers
------------------------+----------+-----------
 schemaname             | name     |
 tablename              | name     |
 attname                | name     |
 inherited              | boolean  |
 null_frac              | real     |
 avg_width              | integer  |
 n_distinct             | real     |
 most_common_vals       | anyarray |
 most_common_freqs      | real[]   |
 histogram_bounds       | anyarray |
 correlation            | real     |
 most_common_elems      | anyarray |
 most_common_elem_freqs | real[]   |
 elem_count_histogram   | real[]   |

cavbd=> 
```

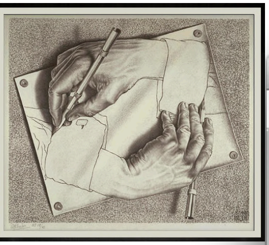
pg_stats view select

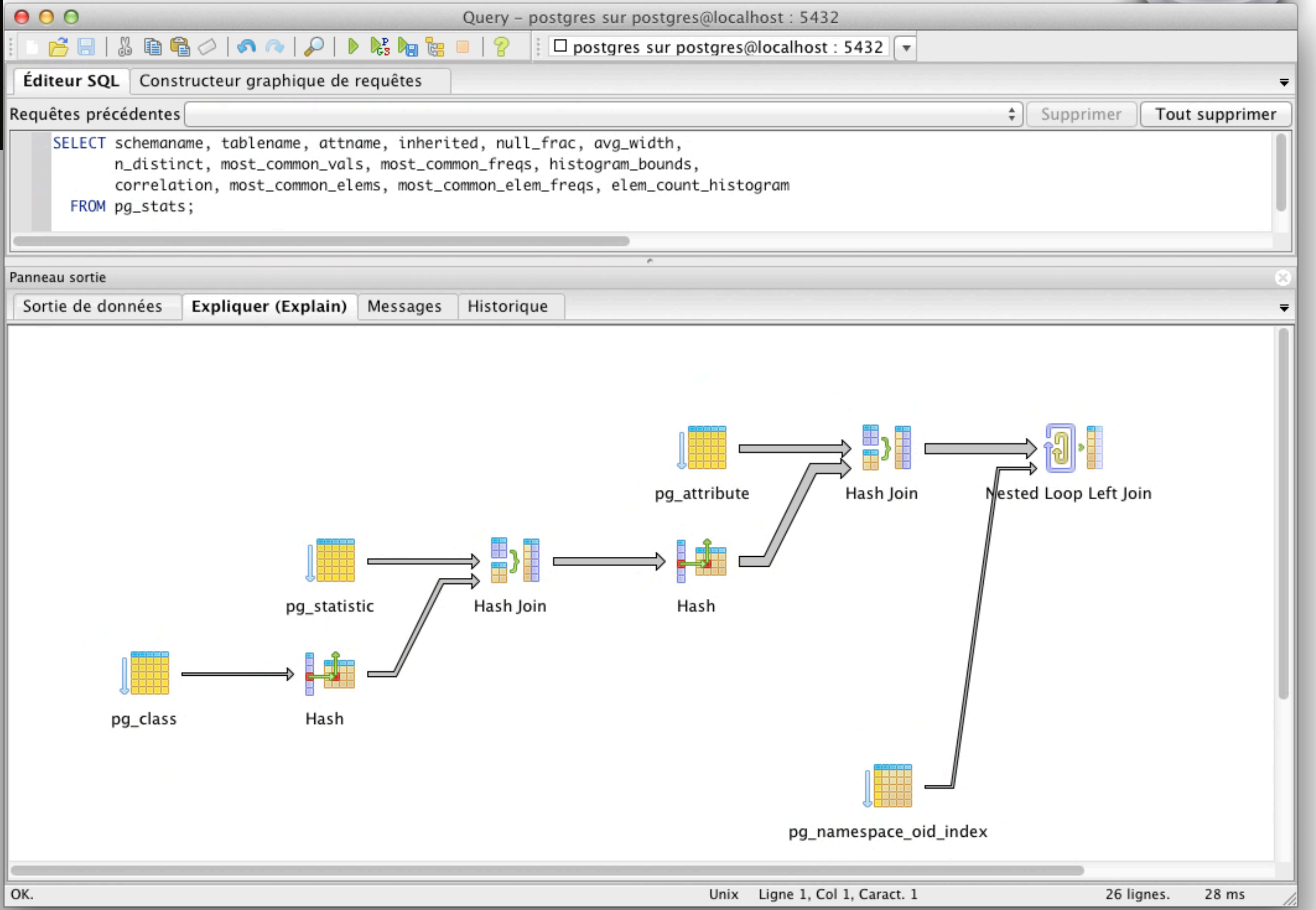
```
SELECT schemaname, tablename, attname, inherited, null_frac, avg_width,
          n_distinct, most_common_vals, most_common_freqs, histogram_bounds,
          correlation, most_common_elems, most_common_elem_freqs,
          elem_count_histogram

 FROM pg_stats;
```



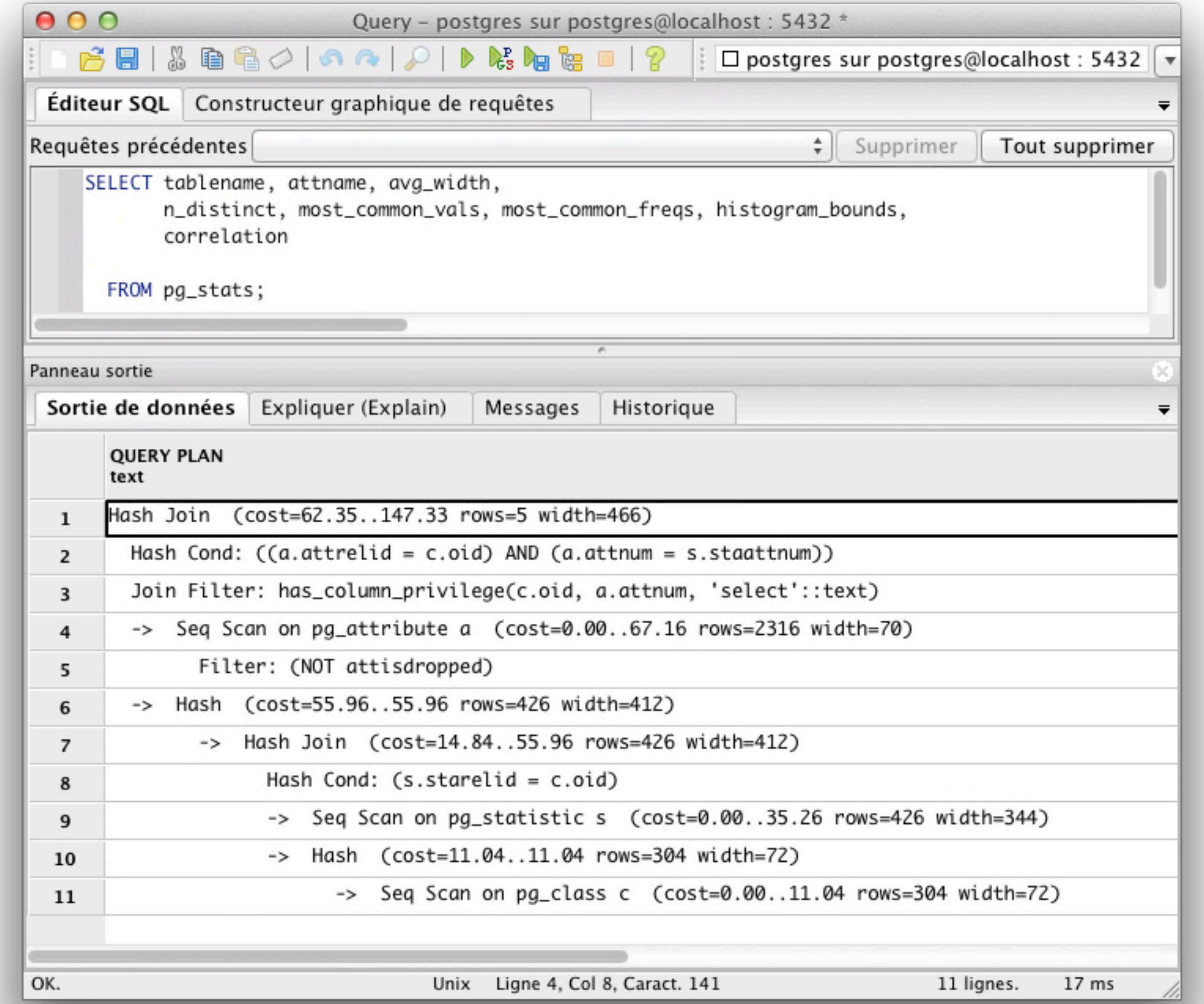

Query – postgres sur postgres@localhost : 5432 *
□ postgres sur postgres@localhost : 5432
Éditeur SQL
Constructeur graphique de requêtes
Requêtes précédentes
Supprimer
Tout supprimer

```
SELECT tablename, attname, avg_width,
       n_distinct, most_common_vals, most_common_freqs, histogram_bounds,
       correlation

  FROM pg_stats;
```





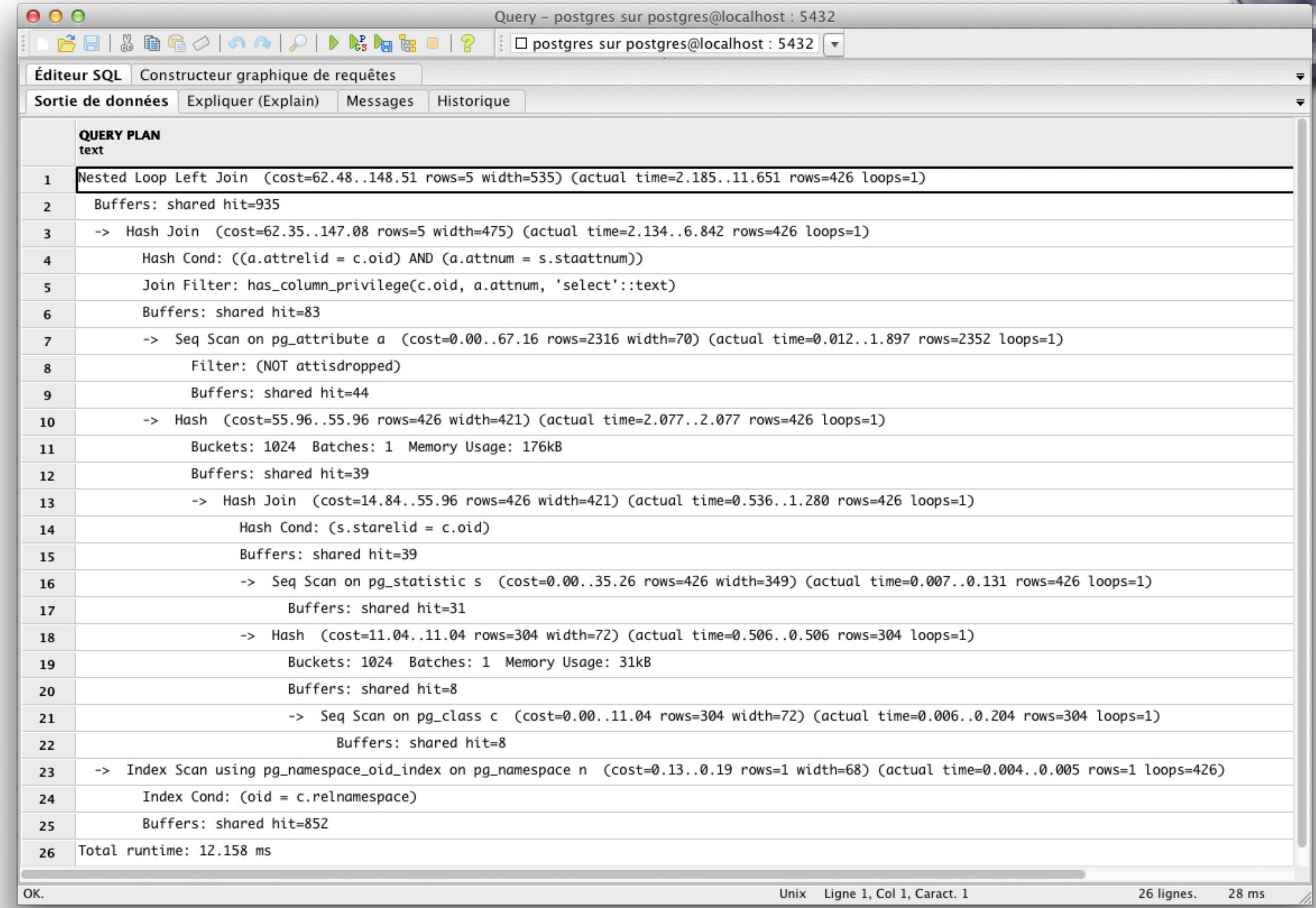
```sql
SELECT schemaname, tablename, attname, inherited, null_frac, avg_width,
       n_distinct, most_common_vals, most_common_freqs, histogram_bounds,
       correlation, most_common_elems, most_common_elem_freqs, elem_count_histogram
  FROM pg_stats;
```


Query – postgres sur postgres@localhost : 5432 *
☐ postgres sur postgres@localhost : 5432
Éditeur SQL
Constructeur graphique de requêtes
Requêtes précédentes
Supprimer
Tout supprimer
Panneau sortie
Sortie de données
Expliquer (Explain)
Messages
Historique

```sql
SELECT tablename, attname, avg_width,
       n_distinct, most_common_vals, most_common_freqs, histogram_bounds,
       correlation

  FROM pg_stats;
```

```
     QUERY PLAN
     text
1    Hash Join  (cost=62.35..147.33 rows=5 width=466)
2      Hash Cond: ((a.attrelid = c.oid) AND (a.attnum = s.staattnum))
3      Join Filter: has_column_privilege(c.oid, a.attnum, 'select'::text)
4      ->  Seq Scan on pg_attribute a  (cost=0.00..67.16 rows=2316 width=70)
5            Filter: (NOT attisdropped)
6      ->  Hash  (cost=55.96..55.96 rows=426 width=412)
7            ->  Hash Join  (cost=14.84..55.96 rows=426 width=412)
8                  Hash Cond: (s.starelid = c.oid)
9                  ->  Seq Scan on pg_statistic s  (cost=0.00..35.26 rows=426 width=344)
10                 ->  Hash  (cost=11.04..11.04 rows=304 width=72)
11                       ->  Seq Scan on pg_class c  (cost=0.00..11.04 rows=304 width=72)
```

```
OK.          Unix    Ligne 4, Col 8, Caract. 141          11 lignes.     17 ms
```

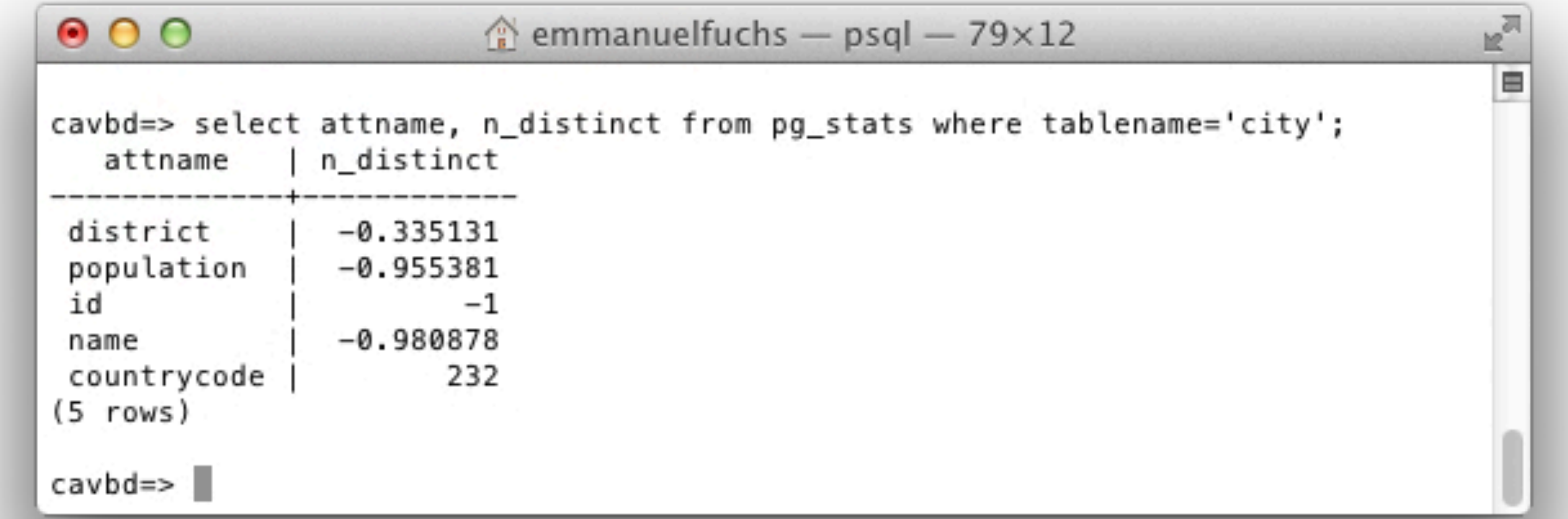

Query – postgres sur postgres@localhost : 5432
☐ postgres sur postgres@localhost : 5432
Éditeur SQL
Constructeur graphique de requêtes
Sortie de données
Expliquer (Explain)
Messages
Historique
QUERY PLAN
text
OK.
Unix
Ligne 1, Col 1, Caract. 1
26 lignes.
28 ms

```
Nested Loop Left Join  (cost=62.48..148.51 rows=5 width=535) (actual time=2.185..11.651 rows=426 loops=1)
  Buffers: shared hit=935
  ->  Hash Join  (cost=62.35..147.08 rows=5 width=475) (actual time=2.134..6.842 rows=426 loops=1)
        Hash Cond: ((a.attrelid = c.oid) AND (a.attnum = s.staattnum))
        Join Filter: has_column_privilege(c.oid, a.attnum, 'select'::text)
        Buffers: shared hit=83
        ->  Seq Scan on pg_attribute a  (cost=0.00..67.16 rows=2316 width=70) (actual time=0.012..1.897 rows=2352 loops=1)
              Filter: (NOT attisdropped)
              Buffers: shared hit=44
        ->  Hash  (cost=55.96..55.96 rows=426 width=421) (actual time=2.077..2.077 rows=426 loops=1)
              Buckets: 1024  Batches: 1  Memory Usage: 176kB
              Buffers: shared hit=39
              ->  Hash Join  (cost=14.84..55.96 rows=426 width=421) (actual time=0.536..1.280 rows=426 loops=1)
                    Hash Cond: (s.starelid = c.oid)
                    Buffers: shared hit=39
                    ->  Seq Scan on pg_statistic s  (cost=0.00..35.26 rows=426 width=349) (actual time=0.007..0.131 rows=426 loops=1)
                          Buffers: shared hit=31
                    ->  Hash  (cost=11.04..11.04 rows=304 width=72) (actual time=0.506..0.506 rows=304 loops=1)
                          Buckets: 1024  Batches: 1  Memory Usage: 31kB
                          Buffers: shared hit=8
                          ->  Seq Scan on pg_class c  (cost=0.00..11.04 rows=304 width=72) (actual time=0.006..0.204 rows=304 loops=1)
                                Buffers: shared hit=8
  ->  Index Scan using pg_namespace_oid_index on pg_namespace n  (cost=0.13..0.19 rows=1 width=68) (actual time=0.004..0.005 rows=1 loops=426)
        Index Cond: (oid = c.relnamespace)
        Buffers: shared hit=852
Total runtime: 12.158 ms
```

n_distinct

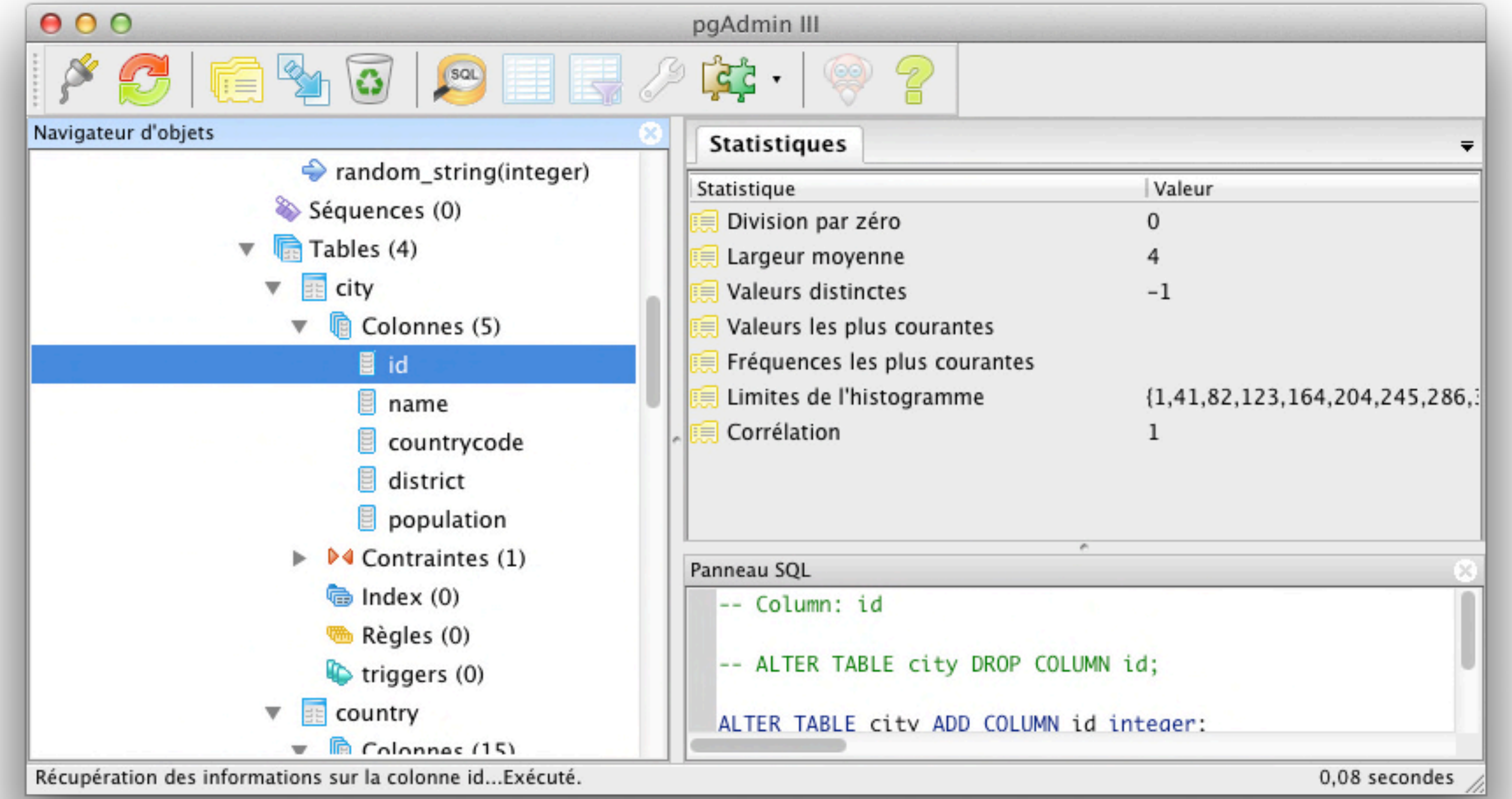
- If greater than zero,
  - the estimated number of distinct values in the column.
- If less than zero,
  - the negative of the number of distinct values divided by the number of rows.
    - The negated form is used when ANALYZE believes that the number of distinct values is likely to increase as the table grows;
    - The positive form is used when the column seems to have a fixed number of possible values.
    - -1 indicates a unique column in which the number of distinct values is the same as the number of rows.

Pgstats : n_distinct



```
cavbd=> select attname, n_distinct from pg_stats where tablename='city';
   attname    | n_distinct
--------------+------------
 district     |  -0.335131
 population   |  -0.955381
 id           |         -1
 name         |  -0.980878
 countrycode  |        232
(5 rows)

cavbd=> 
```

Pgadmin III city.id statistics

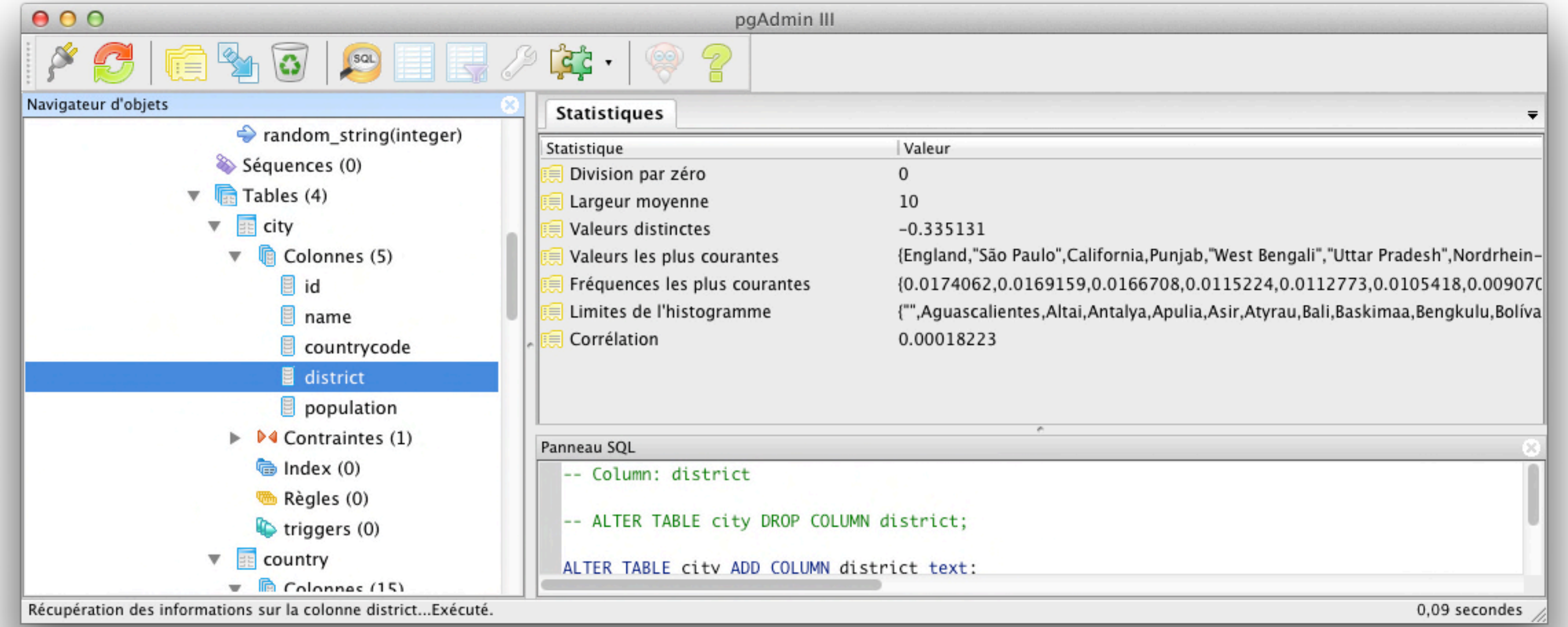


Statistique	Valeur
Division par zéro	0
Largeur moyenne	4
Valeurs distinctes	-1
Valeurs les plus courantes	
Fréquences les plus courantes	
Limites de l'histogramme	{1,41,82,123,164,204,245,286,:
Corrélation	1

```
-- Column: id

-- ALTER TABLE city DROP COLUMN id;

ALTER TABLE city ADD COLUMN id integer;
```

Récupération des informations sur la colonne id...Exécuté. 0,08 secondes

Pgadmin III city.district statistics



Navigateur d'objets

- random_string(integer)
- Séquences (0)
- Tables (4)
  - city
    - Colonnes (5)
      - id
      - name
      - countrycode
      - district
      - population
    - Contraintes (1)
    - Index (0)
    - Règles (0)
    - triggers (0)
  - country
    - Colonnes (15)

Statistiques

Statistique	Valeur
Division par zéro	0
Largeur moyenne	10
Valeurs distinctes	-0.335131
Valeurs les plus courantes	{England,"São Paulo",California,Punjab,"West Bengali","Uttar Pradesh",Nordrhein-
Fréquences les plus courantes	{0.0174062,0.0169159,0.0166708,0.0115224,0.0112773,0.0105418,0.00907
Limites de l'histogramme	{"",Aguascalientes,Altai,Antalya,Apulia,Asir,Atyrau,Bali,Baskimaa,Bengkulu,Bolíva
Corrélation	0.00018223

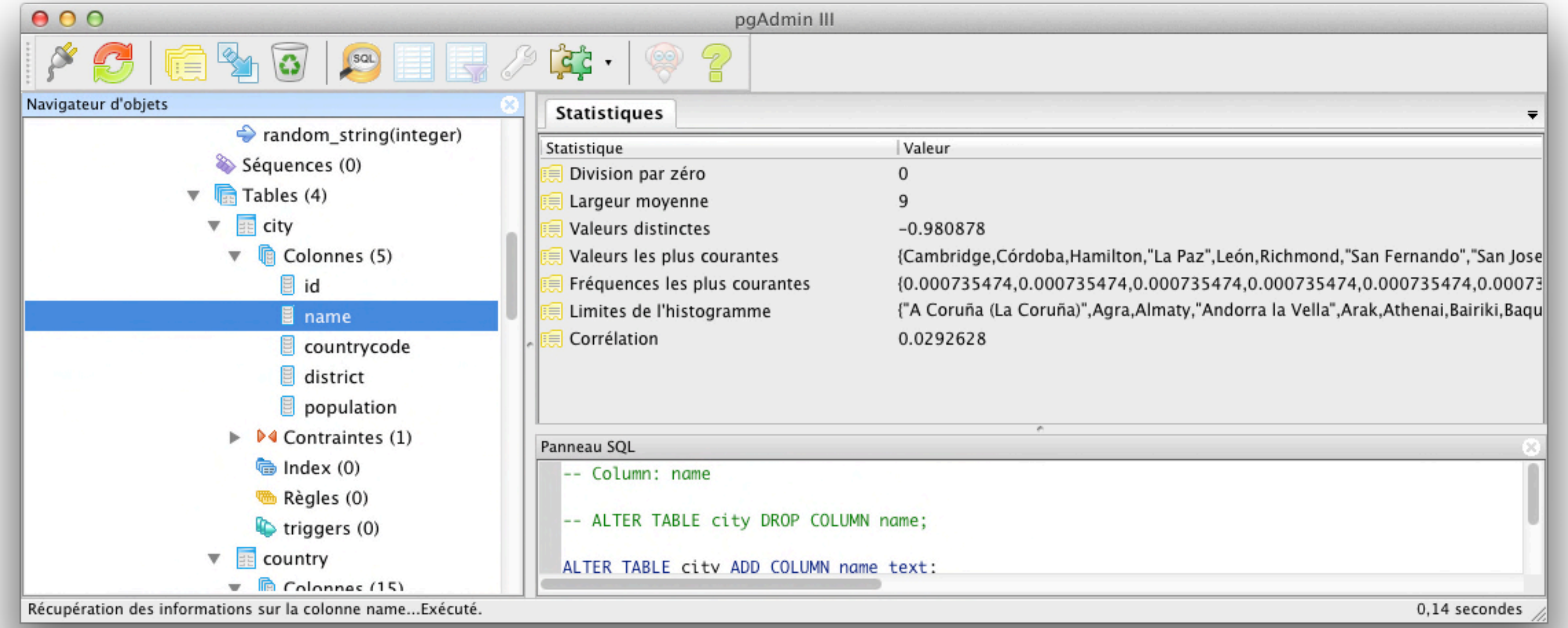
Panneau SQL

```
-- Column: district

-- ALTER TABLE city DROP COLUMN district;

ALTER TABLE city ADD COLUMN district text;
```

Récupération des informations sur la colonne district...Exécuté.

0,09 secondes

Pgadmin III city.name statistics



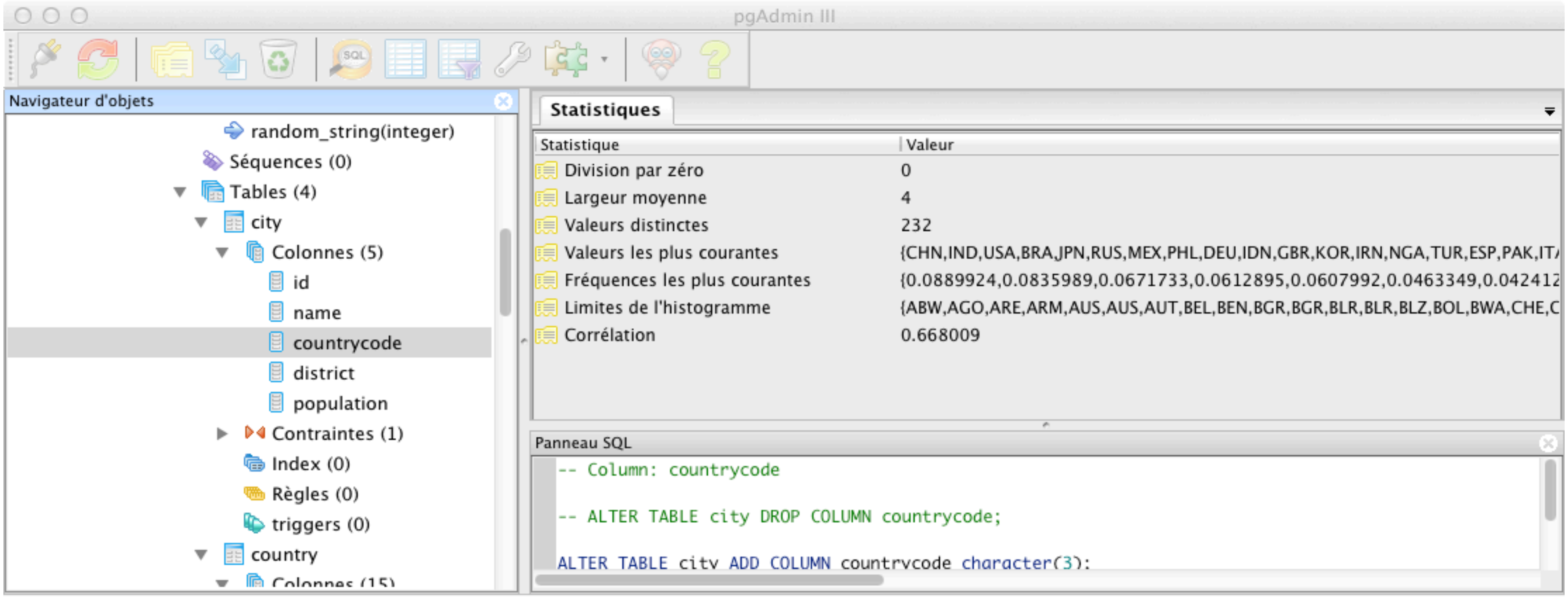
Statistique	Valeur
Division par zéro	0
Largeur moyenne	9
Valeurs distinctes	-0.980878
Valeurs les plus courantes	{Cambridge,Córdoba,Hamilton,"La Paz",León,Richmond,"San Fernando","San Jose
Fréquences les plus courantes	{0.000735474,0.000735474,0.000735474,0.000735474,0.000735474,0.00073
Limites de l'histogramme	{"A Coruña (La Coruña)",Agra,Almaty,"Andorra la Vella",Arak,Athenai,Bairiki,Baqu
Corrélation	0.0292628

```
-- Column: name

-- ALTER TABLE city DROP COLUMN name;

ALTER TABLE city ADD COLUMN name text;
```

Récupération des informations sur la colonne name...Exécuté.

0,14 secondes

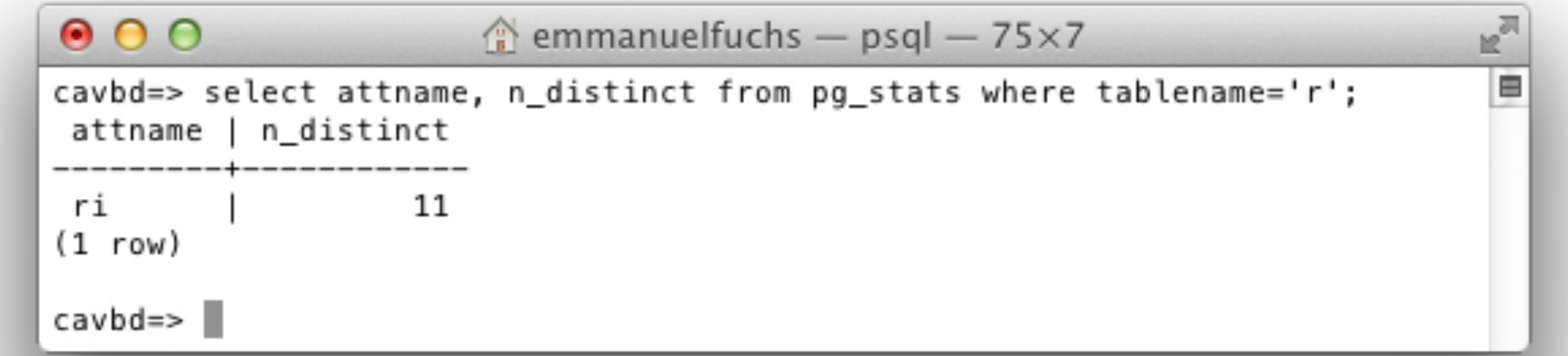
Pgadmin III city.countrycode statistics



Statistique	Valeur
Division par zéro	0
Largeur moyenne	4
Valeurs distinctes	232
Valeurs les plus courantes	{CHN,IND,USA,BRA,JPN,RUS,MEX,PHL,DEU,IDN,GBR,KOR,IRN,NGA,TUR,ESP,PAK,IT
Fréquences les plus courantes	{0.0889924,0.0835989,0.0671733,0.0612895,0.0607992,0.0463349,0.042412
Limites de l'histogramme	{ABW,AGO,ARE,ARM,AUS,AUS,AUT,BEL,BEN,BGR,BGR,BLR,BLR,BLZ,BOL,BWA,CHE,C
Corrélation	0.668009

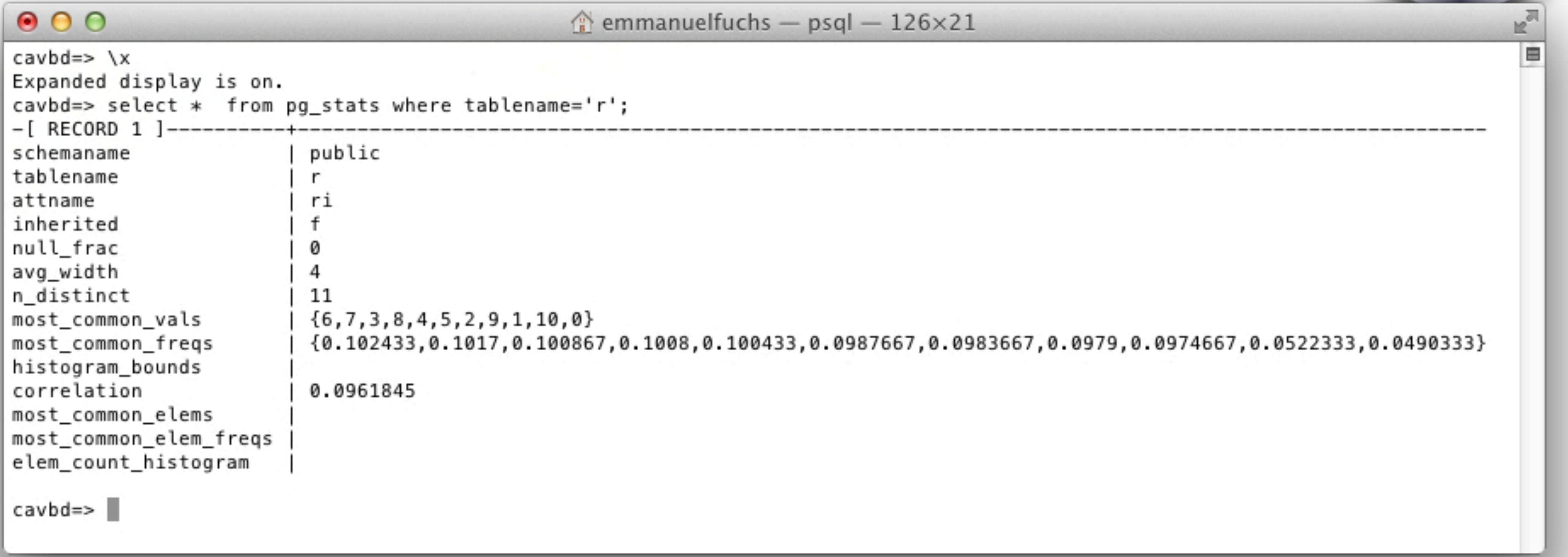
```
-- Column: countrycode

-- ALTER TABLE city DROP COLUMN countrycode;

ALTER TABLE city ADD COLUMN countrycode character(3);
```

n_distinct r = 11



```
cavbd=> select attname, n_distinct from pg_stats where tablename='r';
 attname | n_distinct 
---------+------------
 ri      |         11
(1 row)

cavbd=> 
```

n_distinct r = 11

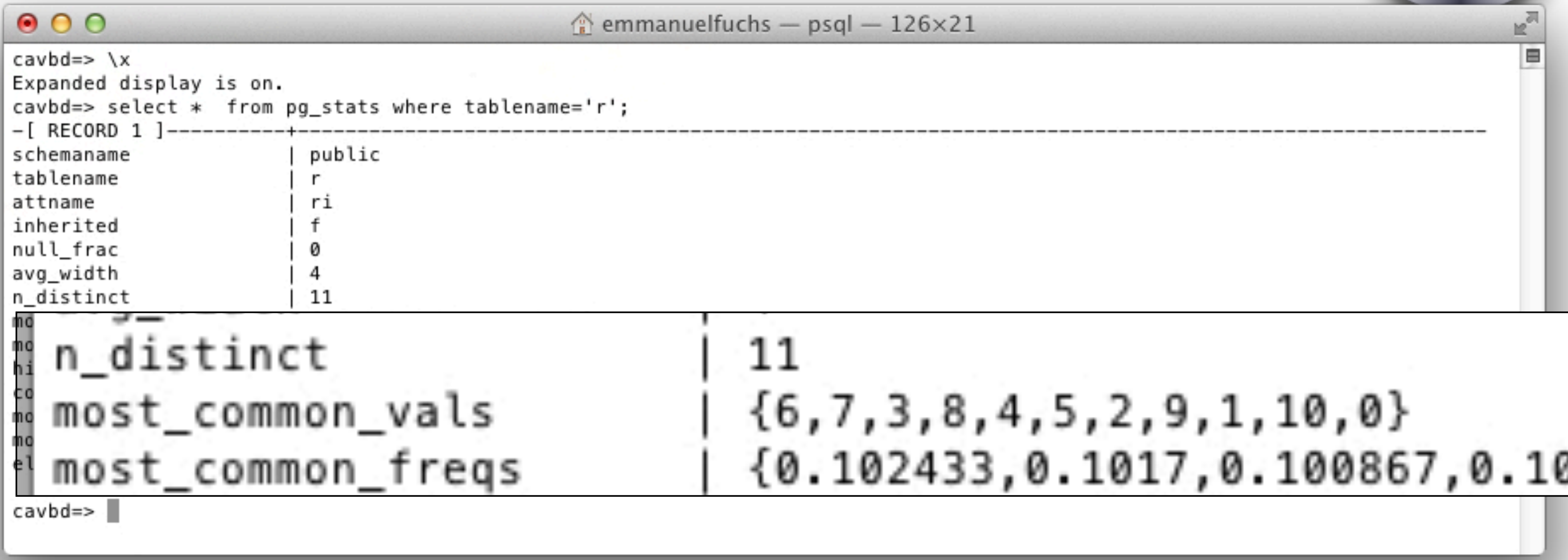


```
cavbd=> \x
Expanded display is on.
cavbd=> select *  from pg_stats where tablename='r';
-[ RECORD 1 ]----------+----------------------------------------------------------------------------------------------------
schemaname             | public
tablename              | r
attname                | ri
inherited              | f
null_frac              | 0
avg_width              | 4
n_distinct             | 11
most_common_vals       | {6,7,3,8,4,5,2,9,1,10,0}
most_common_freqs      | {0.102433,0.1017,0.100867,0.1008,0.100433,0.0987667,0.0983667,0.0979,0.0974667,0.0522333,0.0490333}
histogram_bounds       |
correlation            | 0.0961845
most_common_elems      |
most_common_elem_freqs |
elem_count_histogram   |

cavbd=> 
```

most_common_freqs

- A list of the frequencies of the most common values, i.e., number of occurrences of each divided by total number of rows.

most_common_value



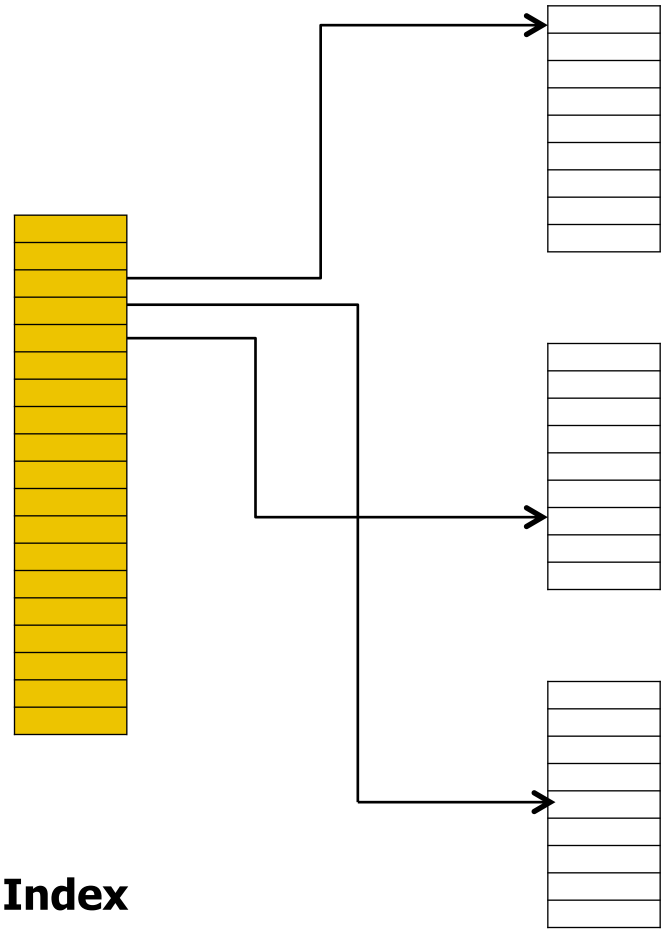
```
cavbd=> \x
Expanded display is on.
cavbd=> select *  from pg_stats where tablename='r';
-[ RECORD 1 ]----------+----------------------------------------------------------------------------------
schemaname             | public
tablename              | r
attname                | ri
inherited              | f
null_frac              | 0
avg_width              | 4
n_distinct             | 11
mo
mo
hi
co
mo
mo
el
cavbd=> 
```

```
n_distinct             | 11
most_common_vals       | {6,7,3,8,4,5,2,9,1,10,0}
most_common_freqs      | {0.102433,0.1017,0.100867,0.10
```

correlation

- Statistical correlation between physical row ordering and logical ordering of the column values.
- This ranges from -1 to +1.
- When the value is near -1 or +1, an index scan on the column will be estimated to be cheaper than when it is near zero, due to reduction of random access to the disk.
- This column is NULL if the column data type does not have a < operator.

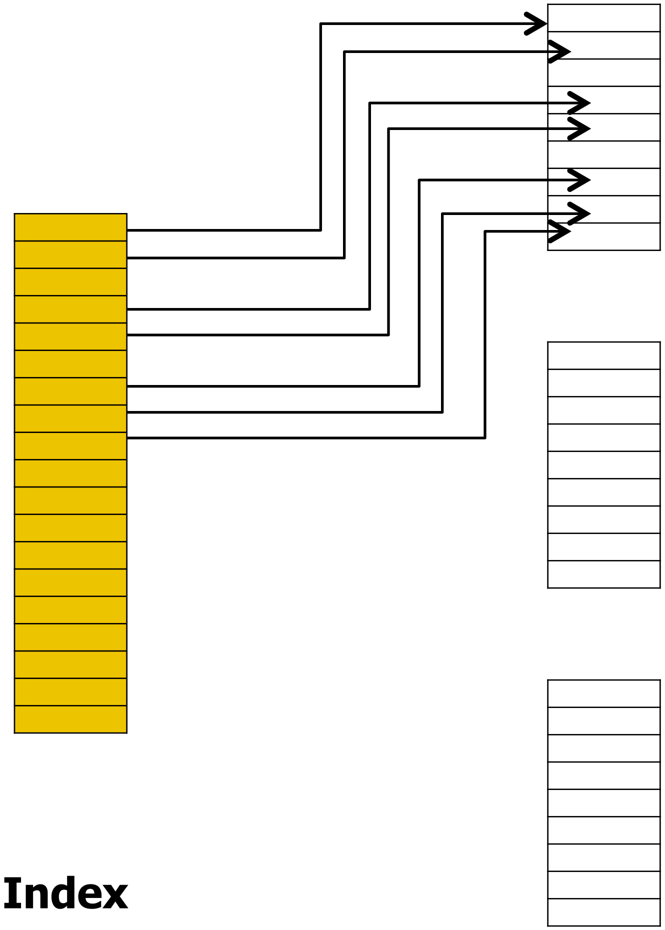
Index and disk blocks correlation



0

Disk Blocks

Index and disk blocks correlation



1

Index

Disk Blocks