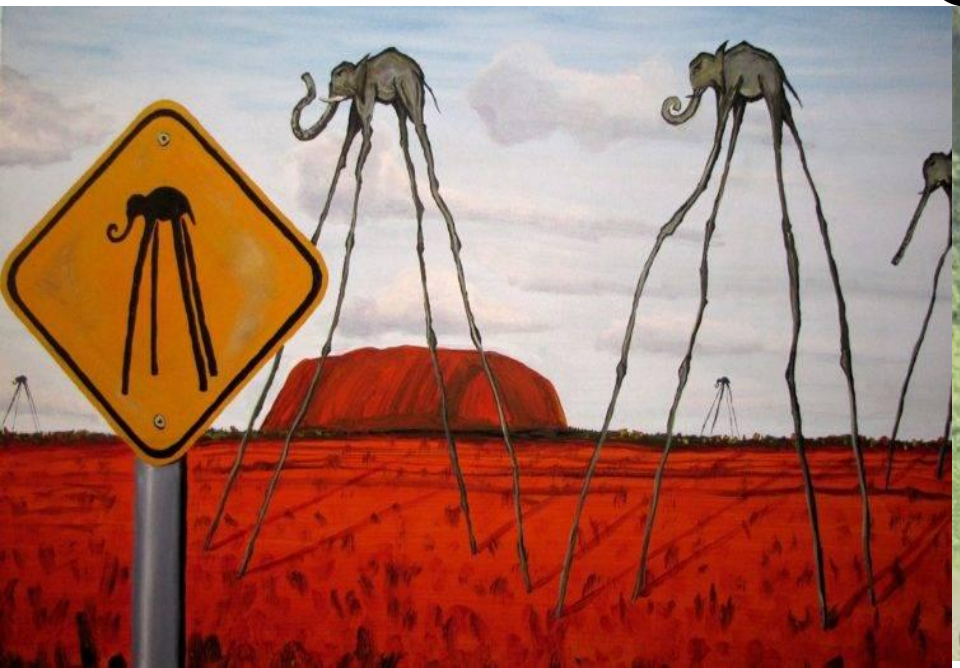



gifak.net

```python
from django.db import models


class User(models.Model):
    username = models.CharField(max_length=200, blank=False, null=False)


class Article(models.Model):
    author = models.ForeignKey(User, on_delete=models.CASCADE)
    text = models.TextField(blank=False, null=False)
    created_time = models.DateTimeField(blank=False, null=False)
    category = models.CharField(max_length=200)

    class Meta:
        indexes = [
            models.Index(fields=['created_time']),
            models.Index(fields=['category']),
        ]


class Comment(models.Model):
    author = models.ForeignKey(User, on_delete=models.CASCADE)
    article = models.ForeignKey(Article, on_delete=models.CASCADE)
    text = models.TextField(blank=False, null=False)
    created_time = models.DateTimeField(blank=False, null=False)

    class Meta:
        indexes = [
            models.Index(fields=['created_time']),
        ]
```

Explain queries

```
explain analyze
select * from myapp_comment;
```

Seq Scan on myapp_comment
(cost=0.00..20.20 **rows=1020** width=52)
(actual time=0.005..0.005 **rows=0** loops=1)

Insert 1000000 comments
Seq Scan on myapp_comment
(cost=0.00..21364.00 **rows=1000000** width=57)
(actual time=0.015..71.723 **rows=1000000** loops=1)

Delete comments and **Insert 1000000** new comments
Seq Scan on myapp_comment
(cost=0.00..22273.44 **rows=1090944** width=57)
(actual time=0.030..106.348 **rows=1000000** loops=1)

```
select * from pg_stat_user_tables
where relname = 'myapp_comment'
```

n_tup_ins	**2000000**
n_tup_del	**1000000**
n_live_tup	**1000000**
last_autoanalyze	**2018-10-07 22:15:29.706471+01**
autoanalyze_count	3

```
Select
  Most_common_vals,
  Most_common_freqs,
  correlation
from pg_stats
where tablename='myapp_article' and attname='category';
```

```
most_common_vals        {chess,physics,biology,mathematics}
most_common_freqs       [0.4593, 0.2934, 0.148367, 0.0989333]
correlation             0.334284
```

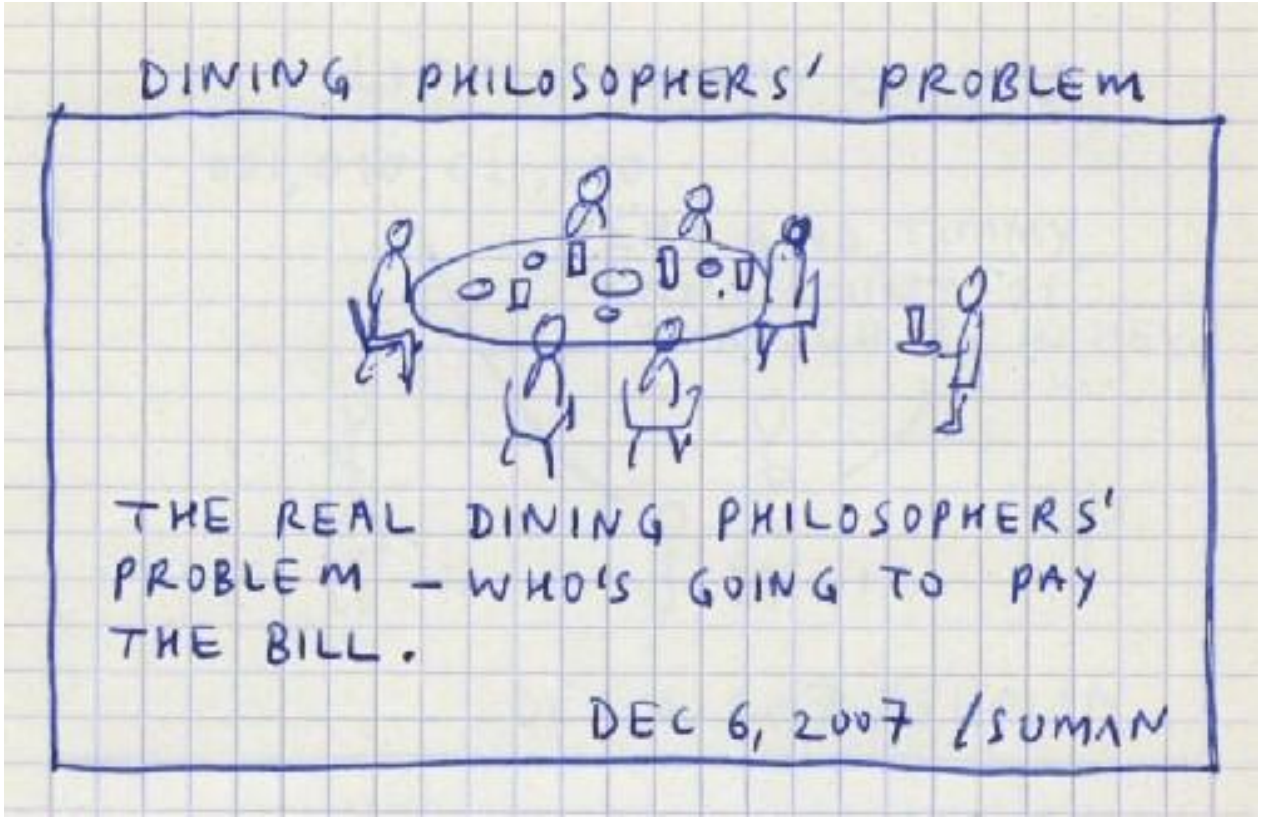
```
Select
   n_live_tup
from pg_stat_user_tables
where relname='myapp_article'
100000
```

Chess estimated rows: 100000 * 0.4593 = 45930

```
explain analyze
select * from myapp_article
where category='chess'
```

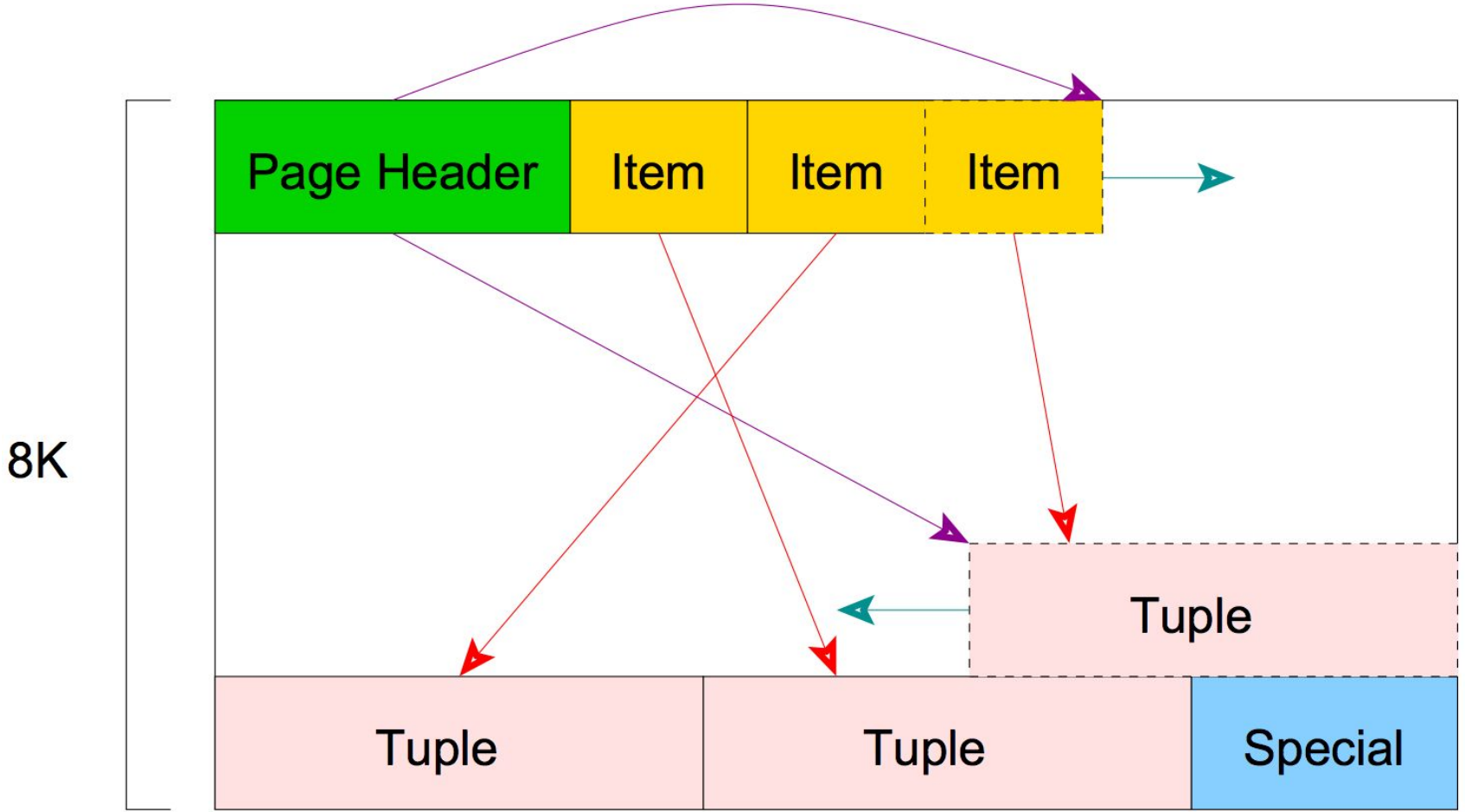
```
 Seq Scan on myapp_article

(cost=0.00..3720.00 rows=45930 width=60)

(actual time=7.184..28.974 rows=45821 loops=1)

 Filter: ((category)::text = 'chess'::text)

 Rows Removed by Filter: 54179
```

Locks - dining philosopher problem



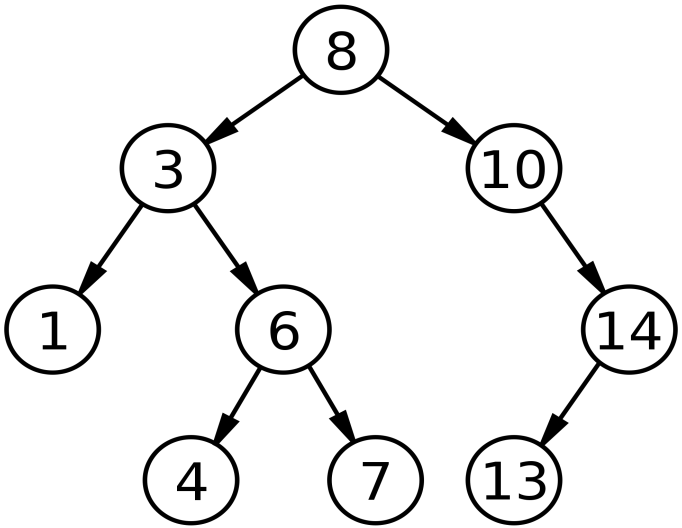
```
begin; delete from myapp_comment;
begin; alter table myapp_comment add column foobar integer;

Select
  granted, pid
from pg_locks
where granted=false;
False, 10240

select query from pg_stat_activity where pid = 10240;
alter table myapp_comment add column foobar integer

pg_blocking_pids
select pg_terminate_backend(<pid-here>);
```

https://gist.github.com/dxe4/1702c21544a37a2cb7b37640a4a956a7

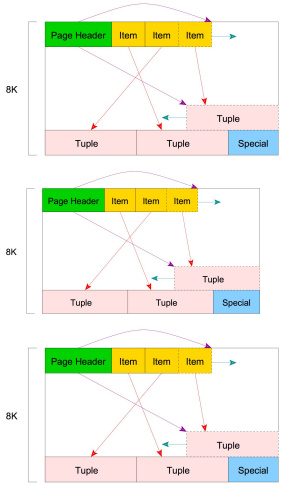

Page Header
Item
Item
Item
8K
Tuple
Tuple
Tuple
Special

Two of the Index types

Binary tree default



BRIN (block range index)

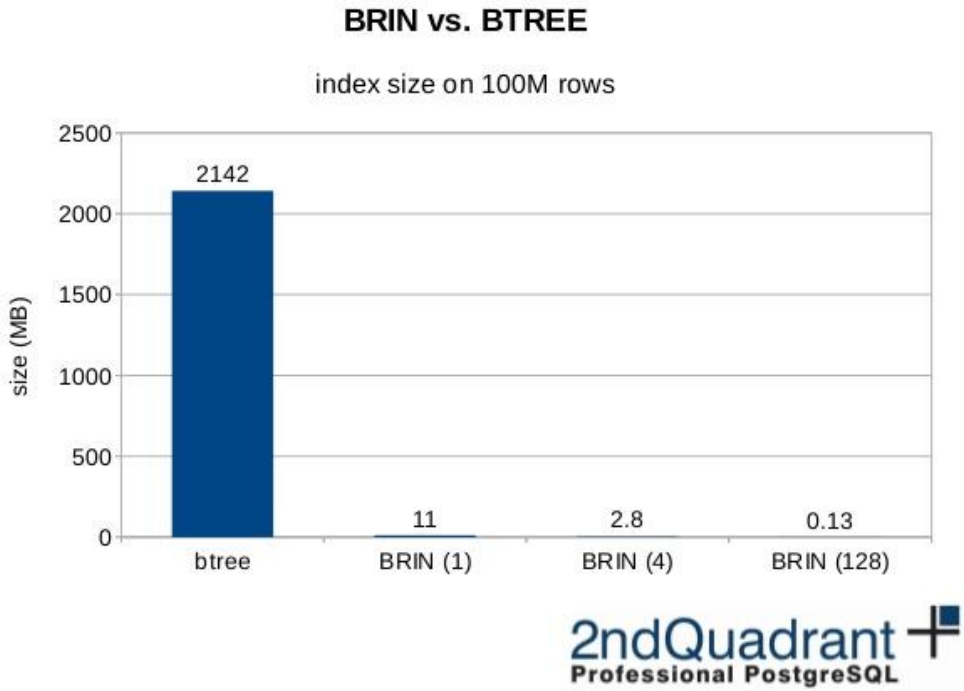


Block 1
Min value 1
Max value 10

Block 2
Min value 11
Max value 21

Block 3
Min value 22
Max value 33

Btree vs brin



2ndQuadrant
Professional PostgreSQL

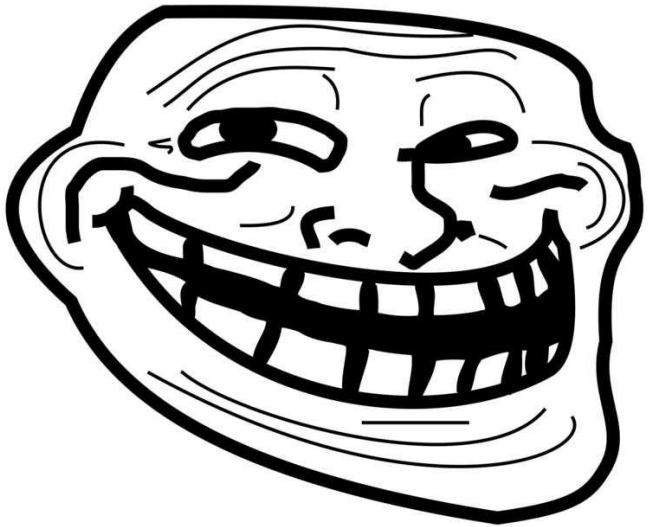
Query A: 2 seconds
Query B: 0.5 seconds

Query A with a filter: 35ms
Query B with a filter: 300ms

Query A
Sort Method: external merge Disk: 72344kB
Sort Method: quicksort Memory: 3408kB


WORKED FINE IN DEV
OPS PROBLEM NOW
memegenerator.net

Delete from table. Takes 10 minutes
Disable all triggers.
Delete from table. Takes 30 seconds



Vacuum and analyze

Vacuum deletes “dead tuples”
Vacuum empties data on disk that is not needed
Auto vacuum triggers by default as a routine task

Analyze creates statistics for tables
Auto analyze triggers by default as a routine task

Psql

Psql postgres://user:pass@host/db -qAt

\o file_name.txt
Select * from users;
\o

Man psql

Psql vs pgcli

```
psql (9.6.10)
Type "help" for help.

mydb=> begin
mydb-> ;
BEGIN
mydb=> select * from myapp_user limit 2;
 id  |                username
-----+---------------------------------------
 101 | 9e8732d6-cb0e-4240-bbd3-fdc6f2a3835a
 102 | b6674dd1-a480-46bb-b86c-f1b7a68de81f
(2 rows)

mydb=>
```

```
myuser@localhost:mydb> select * from myapp_user limit 2;
+------+--------------------------------------+
| id   | username                             |
|------+--------------------------------------|
| 101  | 9e8732d6-cb0e-4240-bbd3-fdc6f2a3835a |
| 102  | b6674dd1-a480-46bb-b86c-f1b7a68de81f |
+------+--------------------------------------+
SELECT 2
Time: 0.017s
myuser@localhost:mydb>

[F2] Smart Completion: ON  [F3] Multiline: OFF  [F4] Emacs-mode     Transaction
```

resources

Blocks and news:
https://blog.2ndquadrant.com/
https://www.citusdata.com/blog/
https://postgresweekly.com/

Stats on queries
Pg_stat_statements https://www.postgresql.org/docs/9.4/static/pgstatstatements.html
Auto_explain https://www.postgresql.org/docs/9.6/static/auto-explain.html

More postgres links https://github.com/dhamaniasad/awesome-postgres

Other
Pgcli https://www.pgcli.com/
Pgbench https://www.postgresql.org/docs/10/static/pgbench.html
Pgcompact https://github.com/grayhemp/pgtoolkit


QUESTIONS ?