

Распределение данных по нескольким БД

(шардинг и партицирование)

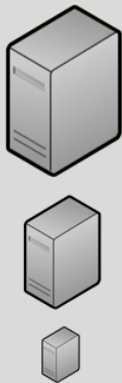
Докладчик:

Гриднев Лев Владимирович

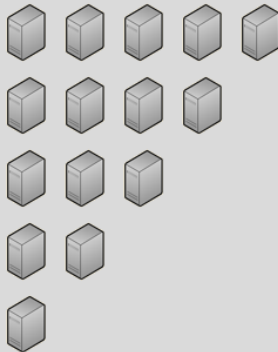
Санкт-Петербург, 2017

Немного о масштабировании

Vertical



Horizontal

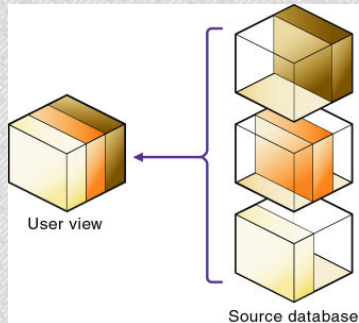
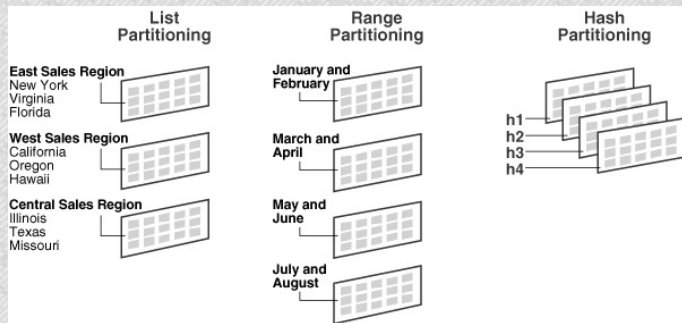


Горизонтальное масштабирование

1. Партицирование (partitioning)
2. Шардинг (sharding)
3. Репликация (replication)

Партицирование

Партицирование - это разбиение таблиц, содержащих большое количество записей, на логические части по неким выбранным принципам.



Шардинг

Шардинг - разбиение данных на группы (по определённому критерию) и хранение их на разных серверах (шардах).

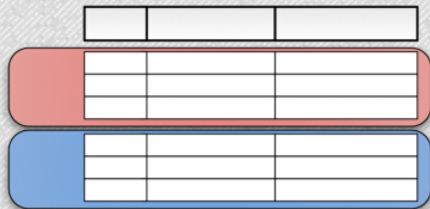


| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Vertical



| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |



| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |

| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |

Horizontal

Практическая часть

PostgreSQL (создание таблиц)

```
create table news_old (  
    id bigint not null,  
    category_id int not null,  
    author character varying  
not null,  
    title character varying  
)
```

```
create table news (  
    id bigint not null,  
    category_id int not null,  
    author character varying  
not null,  
    title character varying  
)
```

PostgreSQL (наследование с ограничением)

```
create table news_1 (check (category_id=1)) inherits (news)
```

Строгое значение

```
check (author = 'Денис')
```

Список значений

```
check (author in ('Денис', 'Антон', 'Олег'))
```

Диапазон значений

```
check (rate>100 and rate<=200)
```


PostgreSQL (создание правила)

```
create rule news_insert_to_1 as on insert to news  
where (category_id=1)  
do instead insert into news_1 values (new.*)
```

PostgreSQL (результаты партицирования)

```
opdb=# explain analyze select * from news_old;
```

QUERY PLAN

```
Seq Scan on news_old (cost=0.00..1980.00 rows=100000 width=47) (actual time=0.014..6.691 rows=100000 loops=1)
Planning time: 0.528 ms
Execution time: 8.037 ms
(3 строки)
```

```
opdb=# explain analyze select * from news;
```

QUERY PLAN

```
Append (cost=0.00..1984.00 rows=100001 width=47) (actual time=0.012..8.047 rows=100000 loops=1)
-> Seq Scan on news (cost=0.00..0.00 rows=1 width=76) (actual time=0.006..0.006 rows=0 loops=1)
-> Seq Scan on news_1 (cost=0.00..165.34 rows=8334 width=47) (actual time=0.007..0.587 rows=8334 loops=1)
-> Seq Scan on news_2 (cost=0.00..161.34 rows=8334 width=45) (actual time=0.008..0.520 rows=8334 loops=1)
-> Seq Scan on news_3 (cost=0.00..165.34 rows=8334 width=46) (actual time=0.008..0.541 rows=8334 loops=1)
-> Seq Scan on news_4 (cost=0.00..169.34 rows=8334 width=50) (actual time=0.007..0.529 rows=8334 loops=1)
-> Seq Scan on news_5 (cost=0.00..165.33 rows=8333 width=47) (actual time=0.008..0.527 rows=8333 loops=1)
-> Seq Scan on news_6 (cost=0.00..161.33 rows=8333 width=45) (actual time=0.009..0.475 rows=8333 loops=1)
-> Seq Scan on news_7 (cost=0.00..165.33 rows=8333 width=46) (actual time=0.008..0.516 rows=8333 loops=1)
-> Seq Scan on news_8 (cost=0.00..169.33 rows=8333 width=50) (actual time=0.007..0.514 rows=8333 loops=1)
-> Seq Scan on news_9 (cost=0.00..165.33 rows=8333 width=46) (actual time=0.009..0.483 rows=8333 loops=1)
-> Seq Scan on news_10 (cost=0.00..161.33 rows=8333 width=44) (actual time=0.009..0.440 rows=8333 loops=1)
-> Seq Scan on news_11 (cost=0.00..165.33 rows=8333 width=45) (actual time=0.010..0.458 rows=8333 loops=1)
-> Seq Scan on news_12 (cost=0.00..169.33 rows=8333 width=50) (actual time=0.007..0.446 rows=8333 loops=1)
Planning time: 3.567 ms
Execution time: 9.404 ms
(16 строк)
```

PostgreSQL (результаты партицирования)

```
opdb=# explain analyze select * from news_old where category_id=2;  
QUERY PLAN
```

```
-----  
Seq Scan on news_old (cost=0.00..2230.00 rows=8290 width=47) (actual time=0.013..8.364 rows=8334 loops=1)  
  Filter: (category_id = 2)  
    Rows Removed by Filter: 91666  
Planning time: 0.089 ms  
Execution time: 8.507 ms  
(5 строк)
```

```
opdb=# explain analyze select * from news where category_id=2;  
QUERY PLAN
```

```
-----  
Append (cost=0.00..182.18 rows=8335 width=45) (actual time=0.013..0.983 rows=8334 loops=1)  
  -> Seq Scan on news (cost=0.00..0.00 rows=1 width=76) (actual time=0.006..0.006 rows=0 loops=1)  
    Filter: (category_id = 2)  
  -> Seq Scan on news_2 (cost=0.00..182.18 rows=8334 width=45) (actual time=0.007..0.795 rows=8334 loops=1)  
    Filter: (category_id = 2)  
Planning time: 0.704 ms  
Execution time: 1.120 ms  
(7 строк)
```

PostgreSQL (результаты шардинга)

```
opdb=# explain analyze select * from news_old;
```

QUERY PLAN

```
-----  
Seq Scan on news_old (cost=0.00..1979.97 rows=99997 width=47) (actual time=0.012..7.293 rows=100000 loops=1)  
Planning time: 0.535 ms  
Execution time: 8.738 ms  
(3 строки)
```

```
opdb=# explain analyze select * from news;
```

QUERY PLAN

```
-----  
Append (cost=100.00..269.14 rows=1638 width=76) (actual time=8.193..5437.309 rows=100000 loops=1)  
  -> Foreign Scan on news_1 (cost=100.00..134.57 rows=819 width=76) (actual time=8.193..2697.683 rows=50000 loops=1)  
  -> Foreign Scan on news_2 (cost=100.00..134.57 rows=819 width=76) (actual time=7.393..2737.287 rows=50000 loops=1)  
Planning time: 27.275 ms  
Execution time: 5485.485 ms  
(5 строк)
```

PostgreSQL (результаты шардинга)

```
opdb=# explain analyze select * from news_old where category_id=2;  
QUERY PLAN
```

```
-----  
Seq Scan on news_old (cost=0.00..2229.96 rows=49895 width=47) (actual time=0.014..9.007 rows=50000 loops=1)  
  Filter: (category_id = 2)  
  Rows Removed by Filter: 50000  
Planning time: 0.080 ms  
Execution time: 9.778 ms  
(5 строк)
```

```
opdb=# explain analyze select * from news where category_id=2;  
QUERY PLAN
```

```
-----  
Append (cost=100.00..240.64 rows=8 width=76) (actual time=26.800..2699.548 rows=50000 loops=1)  
  -> Foreign Scan on news_1 (cost=100.00..120.32 rows=4 width=76) (actual time=3.576..3.576 rows=0 loops=1)  
  -> Foreign Scan on news_2 (cost=100.00..120.32 rows=4 width=76) (actual time=23.223..2694.796 rows=50000 loops=1)  
Planning time: 0.104 ms  
Execution time: 2705.820 ms  
(5 строк)
```

Источники информации

1. Все что нужно знать о секционировании (Часть 1)
2. MongoDB от теории к практике. Руководство по установке кластера mongoDB
3. Горизонтальное масштабирование базы данных реального проекта с помощью SQL Azure Federations
4. Горизонтальное масштабирование серверов баз данных для OLTP-систем, или что есть на рынке
5. Масштабирование базы данных через шардирование и партиционирование
6. Масштабирование баз данных — партиционирование, репликация и шардинг
7. Как устроена MySQL-репликация
8. Андрей Куманяев: Как мы сбежали от PostgreSQL или когда реляционная БД не справляется
9. Алексей Зиновьев - Выбор NoSQL базы данных: "Не в свои сани не садись"
10. Введение в NoSQL базы данных часть 1 и часть 2