

НАСТРОЙКА РЕПЛИКАЦИИ .....	2
ЧТЕНИЕ СО СЛЕЙВОВ .....	5
ПОТЕРЯ МАСТЕРА ПРИ АСИНХРОННОЙ РЕПЛИКАЦИИ .....	7
ПОТЕРЯ МАСТЕРА ПРИ СИНХРОННОЙ РЕПЛИКАЦИИ.....	9
ВЫВОД .....	11

## НАСТРОЙКА РЕПЛИКАЦИИ

В качестве БД – **Postgres 14**. Нагрузочное тестирование - **Jmeter 5.5**

Бекенд - [github.com/jackc/pgx/v5/pgxpool](https://github.com/jackc/pgx/v5/pgxpool)

Будем делать master - slave асинхронную репликацию. Один мастер и два слейва, для удобства балансировки (50/50 между слейвами), восстановления при потере мастера и интеграции будем использовать **pgpool**.

Воспользуемся образами от bitnami т.к. это позволит удобно разворачивать систему. Отличная документация <https://hub.docker.com/r/bitnami/pgpool>

Ниже представлен фрагмент docker-compose с конфигурацией.

```
pg-0:
  image: bitnami/postgresql-repmgr:14
  ports:
    - 5432
  volumes:
    - pg_0_data:/bitnami/postgresql
  environment:
    - POSTGRESQL_POSTGRES_PASSWORD=adminpassword
    - POSTGRESQL_USERNAME=customuser
    - POSTGRESQL_PASSWORD=custompassword
    - POSTGRESQL_DATABASE=default-db
    - REPMGR_PASSWORD=repmgrpassword
    - REPMGR_PRIMARY_HOST=pg-0
    - REPMGR_PARTNER_NODES=pg-0,pg-1,pg-2
    - REPMGR_NODE_NAME=pg-0
    - REPMGR_NODE_NETWORK_NAME=pg-0
  healthcheck:
    test: [ "CMD-SHELL", "pg_isready -U postgres -d default-db" ]
    interval: 10s
    timeout: 5s
    retries: 5
    start_period: 10s
pg-1:
  image: bitnami/postgresql-repmgr:14
  ports:
    - 5432
  volumes:
    - pg_1_data:/bitnami/postgresql
  environment:
    - POSTGRESQL_POSTGRES_PASSWORD=adminpassword
    - POSTGRESQL_USERNAME=customuser
    - POSTGRESQL_PASSWORD=custompassword
    - POSTGRESQL_DATABASE=default-db
    - REPMGR_PASSWORD=repmgrpassword
    - REPMGR_PRIMARY_HOST=pg-0
    - REPMGR_PARTNER_NODES=pg-0,pg-1,pg-2
    - REPMGR_NODE_NAME=pg-1
    - REPMGR_NODE_NETWORK_NAME=pg-1
  healthcheck:
    test: [ "CMD-SHELL", "pg_isready -U postgres -d default-db" ]
    interval: 10s
```

```

    timeout: 5s
    retries: 5
    start_period: 10s
pg-2:
  image: bitnami/postgresql-repmgr:14
  ports:
    - 5432
  volumes:
    - pg_2_data:/bitnami/postgresql
  environment:
    - POSTGRESQL_POSTGRES_PASSWORD=adminpassword
    - POSTGRESQL_USERNAME=customuser
    - POSTGRESQL_PASSWORD=custompassword
    - POSTGRESQL_DATABASE=default-db
    - REPMGR_PASSWORD=repmgrpassword
    - REPMGR_PRIMARY_HOST=pg-0
    - REPMGR_PARTNER_NODES=pg-0,pg-1,pg-2
    - REPMGR_NODE_NAME=pg-2
    - REPMGR_NODE_NETWORK_NAME=pg-2
  healthcheck:
    test: [ "CMD-SHELL", "pg_isready -U postgres -d default-db" ]
    interval: 10s
    timeout: 5s
    retries: 5
    start_period: 10s
pgpool:
  image: bitnami/pgpool:latest
  ports:
    - 5432:5432
  environment:
    - PGPOOL_BACKEND_NODES=0:pg-0:5432:0:pg-
0:ALWAYS_PRIMARY|DISALLOW_TO_FAILOVER,1:pg-1:5432:1:pg-
1:DISALLOW_TO_FAILOVER,2:pg-2:5432:1:pg-2:DISALLOW_TO_FAILOVER
    - PGPOOL_SR_CHECK_USER=customuser
    - PGPOOL_SR_CHECK_PASSWORD=custompassword
    - PGPOOL_ENABLE_LDAP=no
    - PGPOOL_POSTGRES_USERNAME=postgres
    - PGPOOL_POSTGRES_PASSWORD=adminpassword
    - PGPOOL_ADMIN_USERNAME=admin
    - PGPOOL_ADMIN_PASSWORD=adminpassword
  depends_on:
    pg-0:
      condition: service_healthy
    pg-1:
      condition: service_healthy
    pg-2:
      condition: service_healthy
  healthcheck:
    test: [ "CMD", "/opt/bitnami/scripts/pgpool/healthcheck.sh" ]
    interval: 10s
    timeout: 5s
    retries: 5

```

Берем именно postgresql-repmgr тк хотим использовать стратегию failover из коробки. Будем делать потоковую репликацию(Streaming Replication), по-умолчанию она асинхронная.

Рассмотрим флаги для нод:

- 1) ALWAYS\_PRIMARY – флаг означает, что данная нода(БД) является постоянным мастером.
- 2) DISALLOW\_TO\_FAILOVER – флаг означает, что аварийное(failover) переключение не доступно для ноды
- 3) ALLOW\_TO\_FAILOVER – флаг означает, что аварийное переключение доступно

С помощью этих флагов будем управлять поведением системы при потере мастера.

Смотрим статус нод: `show pool_nodes;`

```
PGPASSWORD=adminpassword psql -U postgres -h localhost -p 5432 -d postgres -c "show pool_nodes;"
node_id | hostname | port | status | pg_status | lb_weight | role | pg_role | select_cnt | load_balance_node | replication_delay | replication_state | repli
cation_sync_state | last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
0       | pg-0     | 5432 | up     | up        | 0.000000 | primary | primary | 4           | false              | 0                |                    |
2023-02-06 17:42:18
1       | pg-1     | 5432 | up     | up        | 0.500000 | standby | standby | 0           | true               | 0                |                    |
2023-02-06 17:42:18
2       | pg-2     | 5432 | up     | up        | 0.500000 | standby | standby | 6           | false              | 0                |                    |
2023-02-06 17:42:18
(3 rows)
```

Смотрим тип репликации слейвов

```
SELECT pid,username,application_name,state, sync_state FROM
pg_stat_replication;
```

```
a.m.tsitulskiy@macbook-C02FRB6AMD6R highload-architect % make show-repl
PGPASSWORD=adminpassword psql -U postgres -h localhost -p 5432 -d postgres -c "SELECT pid,username,application_name,state, sync_state FROM pg_stat_replication;"
pid | username | application_name | state | sync_state
-----+-----+-----+-----+-----
370 | repmgr  | pg-2             | streaming | async
371 | repmgr  | pg-1             | streaming | async
(2 rows)
```

Рассмотрим потерю мастера при асинхронной и синхронной репликации, стратегию failover – включим, тк без нее очевидно будет потеря данных.

ЧТЕНИЕ СО СЛЕЙВОВ

Запустим скрипт для сбора docker stats, что бы потом визуализировать

Запустим нагрузочный тест для чтения из прошлого ДЗ

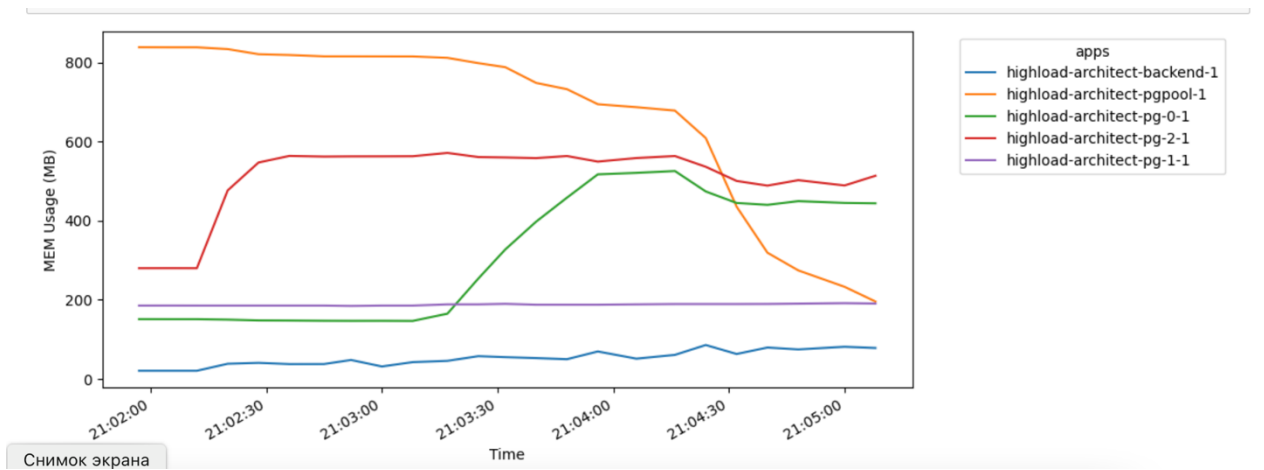
```
a.m.tsitulskiy@macbook-C02FRB6AMD6R highload-architect % make run-jmeter
jmeter -n -t ./HTTP_Request_1.jmx -l ./results.csv -e -o ./report
WARNING: package sun.awt.X11 not in java.desktop
Creating summariser <summary>
Created the tree successfully using ./HTTP_Request_1.jmx
Starting standalone test @ 2023 Feb 6 21:02:13 MSK (167570653325)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary + 556 in 00:00:17 = 33.5/s Avg: 29 Min: 3 Max: 1144 Err: 0 (0.00%) Active: 1 Started: 1 Finished: 0
summary + 2421 in 00:00:30 = 80.7/s Avg: 12 Min: 3 Max: 351 Err: 0 (0.00%) Active: 1 Started: 1 Finished: 0
summary + 2977 in 00:00:47 = 63.9/s Avg: 15 Min: 3 Max: 1144 Err: 0 (0.00%)
summary + 3356 in 00:00:30 = 111.9/s Avg: 51 Min: 3 Max: 3506 Err: 0 (0.00%) Active: 10 Started: 11 Finished: 1
summary + 6333 in 00:01:17 = 82.7/s Avg: 34 Min: 3 Max: 3506 Err: 0 (0.00%)
summary + 3582 in 00:00:30 = 119.3/s Avg: 83 Min: 6 Max: 3289 Err: 0 (0.00%) Active: 10 Started: 11 Finished: 1
summary + 9915 in 00:01:47 = 93.0/s Avg: 52 Min: 3 Max: 3506 Err: 0 (0.00%)
summary + 4044 in 00:00:30 = 134.8/s Avg: 422 Min: 8 Max: 1912 Err: 0 (0.00%) Active: 100 Started: 111 Finished: 11
summary + 13959 in 00:02:17 = 102.2/s Avg: 159 Min: 3 Max: 3506 Err: 0 (0.00%)
summary + 3392 in 00:00:30 = 113.0/s Avg: 827 Min: 357 Max: 4660 Err: 0 (0.00%) Active: 100 Started: 111 Finished: 11
summary + 17351 in 00:02:47 = 104.1/s Avg: 290 Min: 3 Max: 4660 Err: 0 (0.00%)
summary + 1799 in 00:00:34 = 52.2/s Avg: 920 Min: 310 Max: 21760 Err: 0 (0.00%) Active: 1085 Started: 1111 Finished: 26
summary + 19150 in 00:03:21 = 95.2/s Avg: 349 Min: 3 Max: 21760 Err: 0 (0.00%)
Apache/2.4.18 [Ubuntu] Error: 130
```

Видим, что запросы шли на слейвы

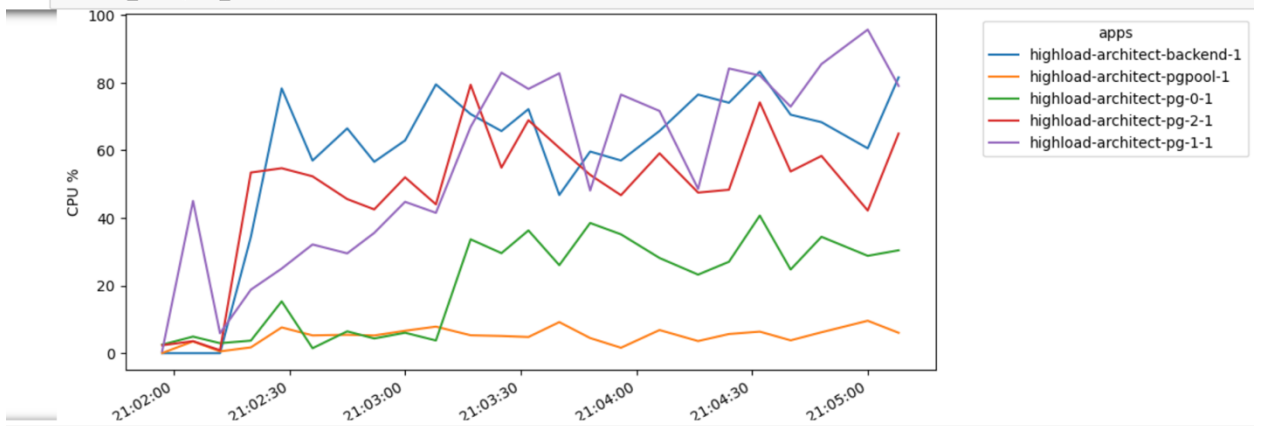
```
node_id | hostname | port | status | pg_status | lb_weight | role | pg_role | select_cnt | load_balance_node | replication_delay | replication_state | r
cation_sync_state | last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
0 | pg-0 | 5432 | up | up | 0.000000 | primary | primary | 80 | false | 0 | | 
| 2023-02-06 17:42:18
1 | pg-1 | 5432 | up | up | 0.500000 | standby | standby | 2607 | false | 0 | | 
| 2023-02-06 17:42:18
2 | pg-2 | 5432 | up | up | 0.500000 | standby | standby | 16730 | true | 0 | | 
| 2023-02-06 17:42:18
(3 rows)
```

В среднем такое потребление ресурсов было во время нагрузки на чтение

	CPU %	MEM %	MEM Usage	PID	NET INPUT	NET OUTPUT	BLOCK INPUT	BLOCK OUTPUT
NAME								
highload-architect-backend-1	57.83	1.29	50.97	11.50	404.17	393.43	2.08	0.00
highload-architect-pg-0-1	5.10	17.20	676.91	17.92	791.58	1860.00	34.96	3230.00
highload-architect-pg-1-1	20.35	7.76	305.57	11.88	876.88	45.49	176.34	3389.58
highload-architect-pg-2-1	48.47	12.89	507.53	11.92	897.67	364.57	205.71	3477.92
highload-architect-pgpool-1	55.59	4.75	186.85	38.21	1042.38	1354.71	2.21	1.67



[5]: ds.plot\_category\_all()



## ПОТЕРЯ МАСТЕРА ПРИ АСИНХРОННОЙ РЕПЛИКАЦИИ

Запустим скрипт для сбора docker stats, что бы потом визуализировать.

Запустим нагрузку на запись, остановим pg-0 и посмотрим % ошибок

```
a.m.tsitulskiy@macbook-C02FRB6AMD6R highload-architect % make run-jmeter-w
jmeter -n -t ./HTTP_write.jmx -l ./results-w.csv -e -o ./report-w
WARNING: package sun.awt.X11 not in java.desktop
Creating summariser <summary>
Created the tree successfully using ./HTTP_write.jmx
Starting standalone test @ 2023 Feb 6 21:35:37 MSK (1675708537793)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary + 284 in 00:00:22 = 12.8/s Avg: 77 Min: 60 Max: 336 Err: 0 (0.00%) Active: 1 Started: 1 Finished: 0
summary + 464 in 00:00:30 = 15.5/s Avg: 64 Min: 60 Max: 93 Err: 0 (0.00%) Active: 1 Started: 1 Finished: 0
summary = 748 in 00:00:52 = 14.3/s Avg: 69 Min: 60 Max: 336 Err: 0 (0.00%)
summary + 66 in 00:00:31 = 2.2/s Avg: 3055 Min: 61 Max: 6512 Err: 44 (66.67%) Active: 11 Started: 11 Finished: 0
summary = 814 in 00:01:23 = 9.8/s Avg: 311 Min: 60 Max: 6512 Err: 44 (5.41%)
summary + 72 in 00:00:30 = 2.4/s Avg: 4211 Min: 3333 Max: 5182 Err: 72 (100.00%) Active: 11 Started: 11 Finished: 0
summary = 886 in 00:01:53 = 7.9/s Avg: 628 Min: 60 Max: 6512 Err: 116 (13.09%)
summary + 1137 in 00:00:30 = 38.4/s Avg: 1949 Min: 99 Max: 62792 Err: 12 (1.06%) Active: 100 Started: 111 Finished: 11
summary = 2023 in 00:02:22 = 14.2/s Avg: 1371 Min: 60 Max: 62792 Err: 128 (6.33%)
summary + 1648 in 00:00:30 = 55.0/s Avg: 1828 Min: 1575 Max: 2248 Err: 0 (0.00%) Active: 100 Started: 111 Finished: 11
summary = 3671 in 00:02:52 = 21.3/s Avg: 1576 Min: 60 Max: 62792 Err: 128 (3.49%)
summary + 473 in 00:00:30 = 15.7/s Avg: 2371 Min: 1568 Max: 22230 Err: 1 (0.21%) Active: 1065 Started: 1111 Finished: 46
summary = 4144 in 00:03:22 = 20.5/s Avg: 1667 Min: 60 Max: 62792 Err: 129 (3.11%)
^Cmake: *** [run-jmeter-w] Error 130
```

Видим, что failover отработал

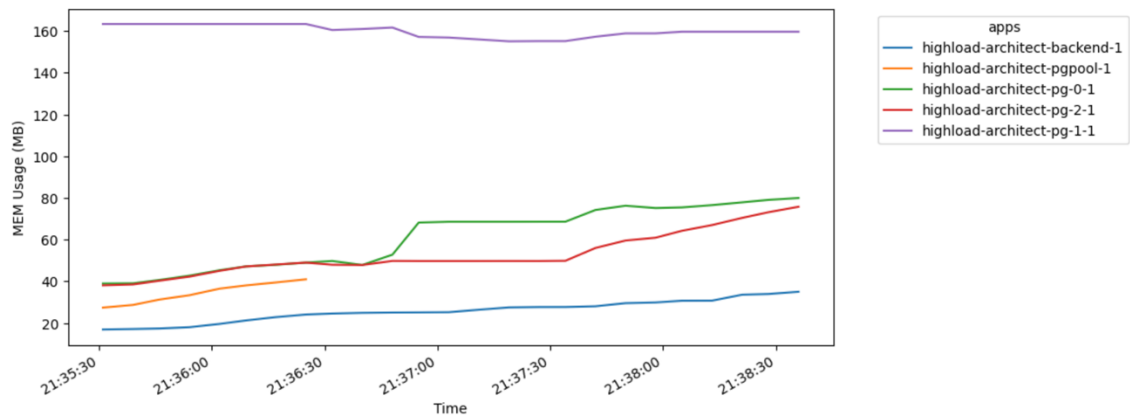
```
(END)...skipping...
node_id | hostname | port | status | pg_status | lb_weight | role | pg_role | select_cnt | load_balance_node | replication_delay
cation_sync_state | last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
0 | pg-0 | 5432 | down | down | 0.000000 | standby | unknown | 4 | false | 0
| 2023-02-06 18:37:32
1 | pg-1 | 5432 | up | up | 0.500000 | primary | primary | 5 | true | 0
| 2023-02-06 18:37:32
2 | pg-2 | 5432 | up | up | 0.500000 | standby | standby | 1 | false | 0
| 2023-02-06 18:33:07
(3 rows)
```

Среднее потребление ресурсов

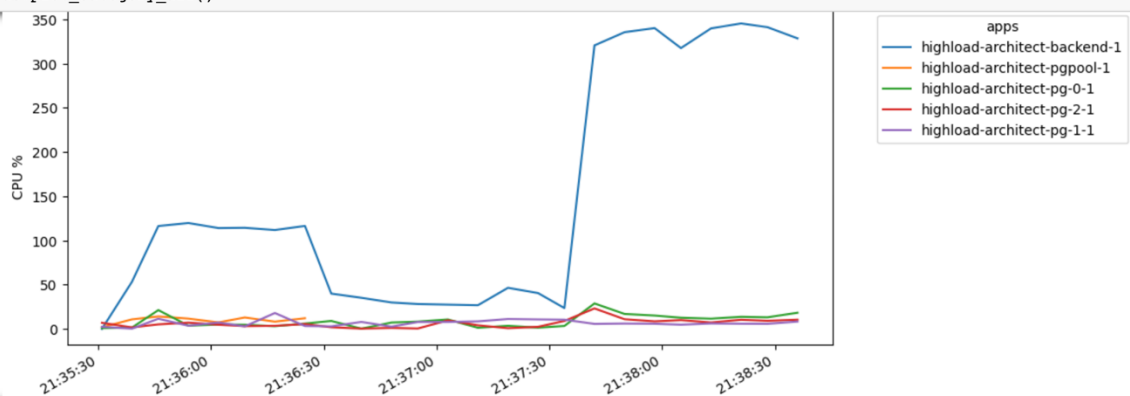
```
[4]: ds.mean_for_apps()
: [4]:
```

	CPU %	MEM %	MEM Usage	PID	NET INPUT	NET OUTPUT	BLOCK INPUT	BLOCK OUTPUT
NAME								
highload-architect-backend-1	148.50	0.65	25.73	9.16	0.84	0.98	1.11	0.00
highload-architect-pg-0-1	9.87	0.88	34.48	16.00	2.38	1157.50	0.05	58.06
highload-architect-pg-1-1	8.86	1.55	61.09	13.64	583.48	18.77	0.93	252.28
highload-architect-pg-2-1	6.29	1.34	52.80	10.60	600.64	1.60	0.71	255.00
highload-architect-pgpool-1	6.71	4.06	159.98	38.08	1.22	1.57	0.95	1.67

```
[5]: ds.plot_category('MEM Usage')
```



```
[6]: ds.plot_category_all()
```



Если просто убить мастера и не восстанавливать, то  
На pg-0 мастер, которого потеряли: 1000810 записей  
На pg-1 слейв, ставший мастером: 1004123 записей  
На pg-2 слейв: 1000810 записей

В случае если мастер просто умер(docker-compose stop), а потом  
восстановился(docker-compose start)

На pg-0 мастер, которого потеряли: 1004123 записей  
На pg-1 слейв, ставший мастером: 1004123 записей  
На pg-2 слейв: 1004123 записей



# ПОТЕРЯ МАСТЕРА ПРИ СИНХРОННОЙ РЕПЛИКАЦИИ

## Устанавливаем синхронную репликацию на pg-0

- POSTGRESQL\_SYNCHRONOUS\_COMMIT\_MODE=on
- POSTGRESQL\_NUM\_SYNCHRONOUS\_REPLICAS=2

```
a.m.tsitulskiy@macbook-C02FRB6AMD6R highload-architect % make show-repl
PGPASSWORD=adminpassword psql -U postgres -h localhost -p 5432 -d postgres -c "SELECT pid,username,application_name,sta
pid | username | application_name | state | sync_state
-----+-----+-----+-----+-----+
375 | repmgr | pg-2 | streaming | sync
376 | repmgr | pg-1 | streaming | sync
(2 rows)
```

## Действуем аналогично

```
a.m.tsitulskiy@macbook-C02FRB6AMD6R highload-architect % make run-jmeter-w
jmeter -n -t ./HTTP_write.jmx -l ./results-w.csv -e -o ./report-w
WARNING: package sun.awt.X11 not in java.desktop
Creating summariser <summary>
Created the tree successfully using ./HTTP_write.jmx
Starting standalone test @ 2023 Feb 6 22:53:24 MSK (1675713204825)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary + 42 in 00:00:05 = 8.0/s Avg: 121 Min: 63 Max: 847 Err: 0 (0.00%) Active: 1 Started: 1 Finished: 0
summary + 389 in 00:00:30 = 13.0/s Avg: 76 Min: 61 Max: 424 Err: 0 (0.00%) Active: 1 Started: 1 Finished: 0
summary + 431 in 00:00:35 = 12.3/s Avg: 80 Min: 61 Max: 847 Err: 0 (0.00%)
summary + 183 in 00:00:31 = 6.0/s Avg: 163 Min: 62 Max: 5404 Err: 6 (3.28%) Active: 11 Started: 11 Finished: 0
summary = 614 in 00:01:06 = 9.3/s Avg: 105 Min: 61 Max: 5404 Err: 6 (0.98%)
summary + 70 in 00:00:30 = 2.3/s Avg: 4504 Min: 3330 Max: 8078 Err: 70 (100.00%) Active: 11 Started: 11 Finished: 0
summary = 684 in 00:01:36 = 7.1/s Avg: 555 Min: 61 Max: 8078 Err: 76 (11.11%)
summary + 1336 in 00:00:29 = 45.4/s Avg: 490 Min: 89 Max: 51788 Err: 12 (0.90%) Active: 100 Started: 111 Finished: 11
summary = 2020 in 00:02:05 = 16.1/s Avg: 512 Min: 61 Max: 51788 Err: 88 (4.36%)
summary + 1473 in 00:00:30 = 49.1/s Avg: 2052 Min: 1570 Max: 4543 Err: 0 (0.00%) Active: 100 Started: 111 Finished: 11
summary = 3493 in 00:02:35 = 22.5/s Avg: 1161 Min: 61 Max: 51788 Err: 88 (2.52%)
summary + 1307 in 00:00:30 = 43.3/s Avg: 1953 Min: 1548 Max: 6716 Err: 0 (0.00%) Active: 1087 Started: 1111 Finished: 24
summary = 4800 in 00:03:05 = 25.9/s Avg: 1377 Min: 61 Max: 51788 Err: 88 (1.83%)
^Cmake: *** [run-jmeter-w] Error 130
```

## Failover – отработал

```
a.m.tsitulskiy@macbook-C02FRB6AMD6R highload-architect % make show-nodes
PGPASSWORD=adminpassword psql -U postgres -h localhost -p 5432 -d postgres -c "show pool_nodes;"
node_id | hostname | port | status | pg_status | lb_weight | role | pg_role | select_cnt | load_balance_node | replication_delay | re
plication_sync_state | last_status_change
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
0 | pg-0 | 5432 | down | down | 0.000000 | standby | unknown | 82 | false | 0 |
| 2023-02-06 19:55:01
1 | pg-1 | 5432 | up | up | 0.500000 | primary | primary | 5 | true | 0 |
| 2023-02-06 19:55:01
2 | pg-2 | 5432 | up | up | 0.500000 | standby | standby | 15 | false | 0 |
| 2023-02-06 19:38:27
(3 rows)
(END)
```

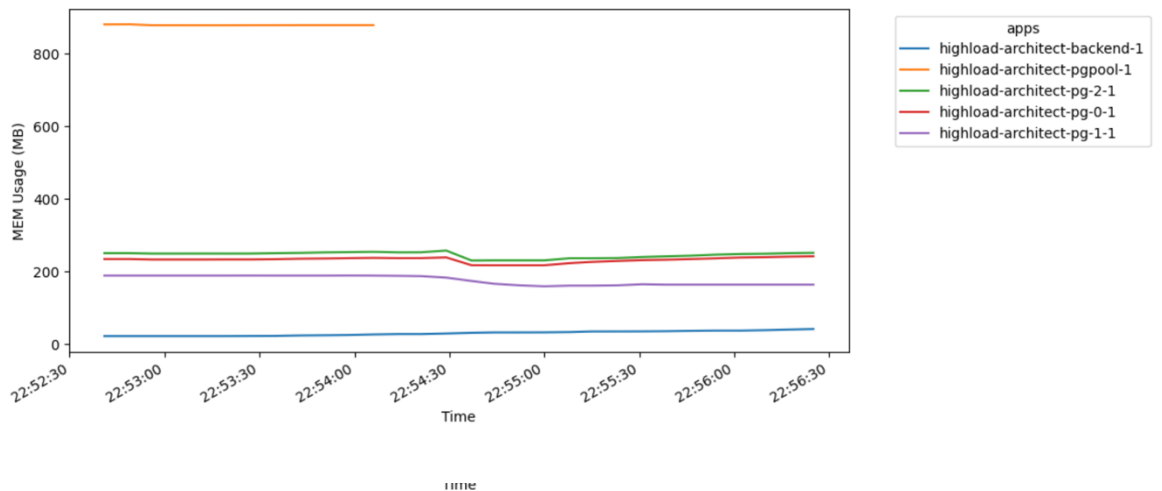
## Потребление ресурсов

```
In [4]: ds.mean_for_apps()
```

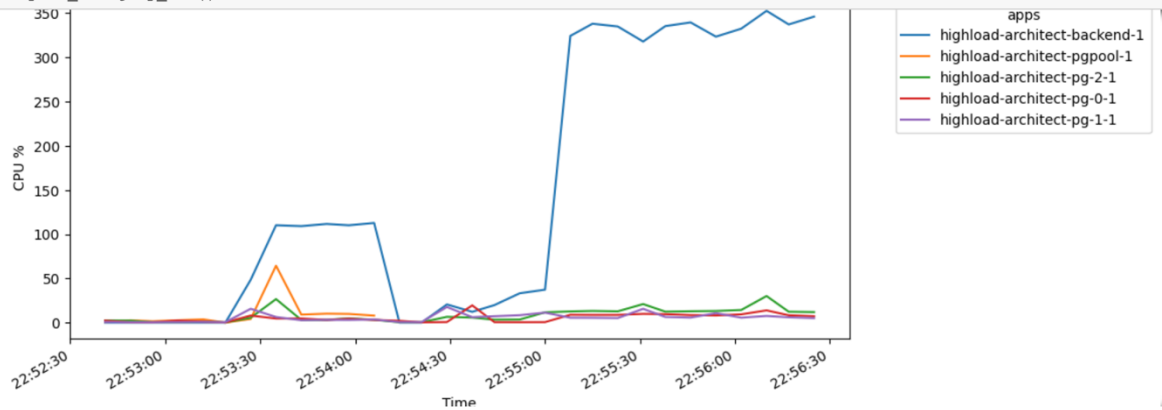
```
Out[4]:
```

	CPU %	MEM %	MEM Usage	PID	NET INPUT	NET OUTPUT	BLOCK INPUT	BLOCK OUTPUT
NAME								
highload-architect-backend-1	146.86	0.74	29.19	10.20	0.84	0.97	8.71	0.00
highload-architect-pg-0-1	9.75	22.32	878.50	16.17	781.42	1834.17	21.40	3085.00
highload-architect-pg-1-1	8.38	6.23	245.31	13.67	860.80	16.58	21.63	3144.00
highload-architect-pg-2-1	5.34	5.89	231.74	11.03	891.60	4.61	23.93	3203.67
highload-architect-pgpool-1	5.63	4.46	175.44	38.33	648.63	939.43	4.67	1.67

```
In [5]: ds.plot_category('MEM Usage')
```



```
n [6]: ds.plot_category_all()
```



Если мастер умер, а потом восстановился, то

На pg-0 мастер, которого потеряли: 1004801 записей

На pg-1 слейв, ставший мастером: 1004801 записей

На pg-2 слейв: 1004801 записей

## **ВЫВОД**

Благодаря failover и pgpool, в случае временной потери мастера, данные в любом случае засинкаются, когда мастер восстановится.

Если сначала умрет мастер, потом умрет весь кластер, то данные будут рассинхронизированы.

По latency видно, что синхронная репликация немного дольше, чем асинхронная.