**Pg\_prewarm Extension**

The pg\_prewarm module provides a convenient way to load relation data into PostgreSQL buffer cache. Prewarming can be performed manually using the pg\_prewarm function, or can be performed automatically by including pg\_prewarm in [shared\_preload\_libraries](https://www.postgresql.org/docs/current/runtime-config-client.html#GUC-SHARED-PRELOAD-LIBRARIES).

 In case of auto prewarm, system will run a background worker which periodically records the contents of shared buffers in a file called autoprewarm.blocks and will be using 2 background workers, reload those same blocks after a restart.

**Prerequisites:**

Contrib module needs to be installed in Linux for pg\_prewarm extension.

**Manual Prewarm:**

1. CREATE EXTENSION pg\_prewarm;
2. Check how many blocks of table available in pg\_buffercache.

SELECT count(\*)

FROM pg\_buffercache

WHERE relfilenode = pg\_relation\_filenode('labs'::regclass);

1. Check the tables in buffer cache.

SELECT c.relname,

count(\*) blocks,

round( 100.0 \* 8192 \* count(\*) / pg\_table\_size(c.oid) ) "% of rel",

round( 100.0 \* 8192 \* count(\*) FILTER (WHERE b.usagecount > 3) / pg\_table\_size(c.oid) ) "% hot"

FROM pg\_buffercache b

JOIN pg\_class c ON pg\_relation\_filenode(c.oid) = b.relfilenode

WHERE b.reldatabase IN (

0, (SELECT oid FROM pg\_database WHERE datname = current\_database())

)

AND b.usagecount is not null

GROUP BY c.relname, c.oid

ORDER BY 2 DESC

LIMIT 10;

1. Number of pages used by table.

SELECT oid::regclass AS tbl, relpages

FROM pg\_class

WHERE relname = 'labs';

1. Call Pg\_prewarm extension and load the table.

SELECT \* FROM pg\_prewarm('labs');

1. Check the table is in memory:

SELECT c.relname,

count(\*) blocks,

round( 100.0 \* 8192 \* count(\*) / pg\_table\_size(c.oid) ) "% of rel",

round( 100.0 \* 8192 \* count(\*) FILTER (WHERE b.usagecount > 3) / pg\_table\_size(c.oid) ) "% hot"

FROM pg\_buffercache b

JOIN pg\_class c ON pg\_relation\_filenode(c.oid) = b.relfilenode

WHERE b.reldatabase IN (

0, (SELECT oid FROM pg\_database WHERE datname = current\_database())

)

AND b.usagecount is not null

GROUP BY c.relname, c.oid

ORDER BY 2 DESC

LIMIT 10;

1. Restart postgresql.
2. Check the table is in memory: (The table wont be their in memory)

SELECT c.relname,

count(\*) blocks,

round( 100.0 \* 8192 \* count(\*) / pg\_table\_size(c.oid) ) "% of rel",

round( 100.0 \* 8192 \* count(\*) FILTER (WHERE b.usagecount > 3) / pg\_table\_size(c.oid) ) "% hot"

FROM pg\_buffercache b

JOIN pg\_class c ON pg\_relation\_filenode(c.oid) = b.relfilenode

WHERE b.reldatabase IN (

0, (SELECT oid FROM pg\_database WHERE datname = current\_database())

)

AND b.usagecount is not null

GROUP BY c.relname, c.oid

ORDER BY 2 DESC

LIMIT 10;

**Auto Prewarm:**

1. CREATE EXTENSION pg\_prewarm;
2. ALTER SYSTEM SET shared\_preload\_libraries = 'pg\_prewarm';
3. Check how many blocks of table available in pg\_buffercache.

SELECT count(\*)

FROM pg\_buffercache

WHERE relfilenode = pg\_relation\_filenode('labs'::regclass);

1. Check the table is present in buffer cache.

SELECT c.relname,

count(\*) blocks,

round( 100.0 \* 8192 \* count(\*) / pg\_table\_size(c.oid) ) "% of rel",

round( 100.0 \* 8192 \* count(\*) FILTER (WHERE b.usagecount > 3) / pg\_table\_size(c.oid) ) "% hot"

FROM pg\_buffercache b

JOIN pg\_class c ON pg\_relation\_filenode(c.oid) = b.relfilenode

WHERE b.reldatabase IN (

0, (SELECT oid FROM pg\_database WHERE datname = current\_database())

)

AND b.usagecount is not null

GROUP BY c.relname, c.oid

ORDER BY 2 DESC

LIMIT 10;

1. Number of pages used by table.

SELECT oid::regclass AS tbl, relpages

FROM pg\_class

WHERE relname = 'labs';

1. Call Pg\_prewarm extension and load the table.

SELECT \* FROM pg\_prewarm('labs');

1. Check the table is in memory:

SELECT c.relname,

count(\*) blocks,

round( 100.0 \* 8192 \* count(\*) / pg\_table\_size(c.oid) ) "% of rel",

round( 100.0 \* 8192 \* count(\*) FILTER (WHERE b.usagecount > 3) / pg\_table\_size(c.oid) ) "% hot"

FROM pg\_buffercache b

JOIN pg\_class c ON pg\_relation\_filenode(c.oid) = b.relfilenode

WHERE b.reldatabase IN (

0, (SELECT oid FROM pg\_database WHERE datname = current\_database())

)

AND b.usagecount is not null

GROUP BY c.relname, c.oid

ORDER BY 2 DESC

LIMIT 10;

1. Restart postgresql.
2. Check the table is in memory: (The table should be in their memory)

SELECT c.relname,

count(\*) blocks,

round( 100.0 \* 8192 \* count(\*) / pg\_table\_size(c.oid) ) "% of rel",

round( 100.0 \* 8192 \* count(\*) FILTER (WHERE b.usagecount > 3) / pg\_table\_size(c.oid) ) "% hot"

FROM pg\_buffercache b

JOIN pg\_class c ON pg\_relation\_filenode(c.oid) = b.relfilenode

WHERE b.reldatabase IN (

0, (SELECT oid FROM pg\_database WHERE datname = current\_database())

)

AND b.usagecount is not null

GROUP BY c.relname, c.oid

ORDER BY 2 DESC

LIMIT 10;

1. If you want to know how much percentage of the buffer the table is using:

SELECT

c.relname,

pg\_size\_pretty(count(\*) \* 8192) as buffered,

round(100.0 \* count(\*) /

(SELECT setting FROM pg\_settings

WHERE name='shared\_buffers')::integer,1)

AS buffers\_percent,

round(100.0 \* count(\*) \* 8192 /

pg\_table\_size(c.oid),1)

AS percent\_of\_relation

FROM pg\_class c

INNER JOIN pg\_buffercache b

ON b.relfilenode = c.relfilenode

INNER JOIN pg\_database d

ON (b.reldatabase = d.oid AND d.datname = current\_database())

GROUP BY c.oid,c.relname

ORDER BY 3 DESC LIMIT 10;