Then let’s also assume that I now have a year’s worth of data stored and I want to search this table for a particular day’s average temperature. I could do a simple sequential scan of all rows in the table looking for every timestamp between a maximum and minimum, however this is not very efficient if I am looking for a result from just 1/365th of the data set.

I could speed this up by adding a simple BTREE index (the default type) to a column on this table, however it will be reasonably large for what I am doing.

This is where the BRIN index is a great idea.

And now for the hands-on part

Test setup:

CREATE DATABASE sensors;  
  
CREATE TABLE temperature\_log (log\_id serial, sensor\_id int,  
log\_timestamp timestamp without time zone, temperature int);  
  
INSERT INTO temperature\_log(sensor\_id,log\_timestamp,temperature)  
VALUES (1,generate\_series('2016-01-01'::timestamp,'2016-12-31'::timestamp,'1 second'),  
        round(random()\*100)::int);

This will create 31536001 rows of sensor test data.

I'll now retrieve the average temperature value for the 4th April 2016 and here are the results:

EXPLAIN ANALYZE SELECT AVG(temperature) FROM temperature\_log  
WHERE log\_timestamp>='2016-04-04' AND log\_timestamp<'2016-04-05';  
QUERY PLAN --------------------------------------------------  
  
Aggregate (cost=674124.07..674124.08 rows=1 width=4)  
(actual time=3180.288..31 80.289 rows=1 loops=1)  
  
-> Seq Scan on temperature\_log (cost=0.00..673907.00 rows=86826 width=4)  
       (actual time=629.297..3138.002 rows=86400 loops=1)  
  
Filter: ((log\_timestamp >= '2016-04-04 00:00:00'::timestamp without time zone)  
        AND (log\_timestamp < '2016-04-05 00:00:00'::timestamp without time zone) )  
  
Rows Removed by Filter: 31449601  
  
Planning time: 0.047 ms  
  
Execution time: 3180.448 ms  
  
(6 rows)

I will get the result in 3180 milliseconds with a straightforward sequential scan and filter of the whole table.

When I add a BTREE index and execute the same query as I did earlier for the same day’s average temperature, I will see:

CREATE INDEX idx\_temperature\_log\_log\_timestamp ON temperature\_log  
USING btree (log\_timestamp);  
  
vacuum analyze;  
  
EXPLAIN ANALYZE SELECT AVG(temperature) FROM temperature\_log  
WHERE log\_timestamp>='2016-04-04' AND log\_timestamp<'2016-04-05';  
  
QUERY PLAN --------------------------------------------------  
  
Aggregate (cost=3467.15..3467.16 rows=1 width=4)  
          (actual time=103.333..103.333 rows=1 loops=1)  
  
-> Index Scan using idx\_temperature\_log\_log\_timestamp on temperature\_log  
       (cost=0.56..3250.08 rows=86826 width=4)  
       (actual time=0.019..55.902 rows=86400 loop s=1)  
  
Index Cond: ((log\_timestamp >= '2016-04-04 00:00:00'::timestamp without time zone)  
             AND (log\_timestamp < '2016-04-05 00:00:00'::timestamp without time zone))  
  
Planning time: 0.059 ms  
  
Execution time: 103.476 ms  
  
(5 rows)

So this is much quicker, returning the result in 103 milliseconds.

With a BRIN index and the same query as earlier for the same day’s average temperature, I see:

DROP INDEX idx\_temperature\_log\_log\_timestamp;  
  
CREATE INDEX idx\_temperature\_log\_log\_timestamp ON temperature\_log  
USING BRIN (log\_timestamp) WITH (pages\_per\_range = 128);  
  
vacuum analyse;  
  
EXPLAIN ANALYZE SELECT AVG(temperature) FROM temperature\_log  
WHERE log\_timestamp>='2016-04-04' AND log\_timestamp<'2016-04-05';

My execution plan looks like:

QUERY PLAN -----------------------------------------  
  
Aggregate (cost=161148.90..161148.91 rows=1 width=4)  
(actual time=96.843..96.8 44 rows=1 loops=1)  
  
-> Bitmap Heap Scan on temperature\_log  
       (cost=935.12..160929.60 rows=87719 width=4)  
       (actual time=1.123..53.489 rows=86400 loops=1)  
  
Recheck Cond: ((log\_timestamp >= '2016-04-04 00:00:00'::timestamp without time zone)  
                AND (log\_timestamp < '2016-04-05 00:00:00'::timestamp without time zone))  
  
Rows Removed by Index Recheck: 14080  
  
Heap Blocks: lossy=640  
  
-> Bitmap Index Scan on idx\_temperature\_log\_log\_timestamp  
         (cost=0.00. .913.19 rows=87719 width=0)  
         (actual time=0.638..0.638 rows=6400 loops=1)  
  
Index Cond: ((log\_timestamp >= '2016-04-04 00:00:00'::timestamp without time zone)  
             AND (log\_timestamp < '2016-04-05 00:00:00'::timestamp without time zone))  
  
Planning time: 0.064 ms  
  
Execution time: 97.023 ms  
  
(9 rows)

For the BRIN index I get a result in 97 milliseconds. This is slightly quicker than using a BTREE index for this particular test but let’s take a look at the index sizes.

This query will return the index size of our new indexes:

SELECT  
nspname AS schema\_name,  
relname AS index\_name,  
round(100 \* pg\_relation\_size(indexrelid) / pg\_relation\_size(indrelid)) / 100 AS index\_ratio,  
pg\_size\_pretty(pg\_relation\_size(indexrelid)) AS index\_size,  
pg\_size\_pretty(pg\_relation\_size(indrelid)) AS table\_size  
  
FROM  
pg\_index I  
  
LEFT JOIN  
pg\_class C  
  
ON  
(C.oid = I.indexrelid)  
  
LEFT JOIN  
pg\_namespace N  
  
ON  
(N.oid = C.relnamespace)  
  
WHERE  
C.relkind = 'i' AND  
pg\_relation\_size(indrelid) > 0 AND  
relname='idx\_temperature\_log\_log\_timestamp'  
  
ORDER BY  
pg\_relation\_size(indexrelid) DESC, index\_ratio DESC;

The BRIN index is 72KB in size, The BTREE index is 676MB index size.