

Open in app ↗

Medium

 Search Write

# DuckDB Performance Problems with Inappropriate Pivoting Queries on Very Large Datasets



Josef Machytka

4 min read · Just now



The DuckDB documentation clearly states that this tool is designed for handling datasets fitting into memory. I fully understand that I'm pushing this tool beyond its limits with the scenarios described here. Therefore, this article is not a complaint against DuckDB — I love this tool and use it very often. Instead, I aim to share my findings, which might inspire further improvements.

In my previous article, "[How DuckDB Handles Data Not Fitting Into Memory](#)", I described DuckDB's behavior when working with a 100 GB CSV file on a machine with 64 GB RAM and 8 GB swap. DuckDB handled the imports perfectly, and since then it has become my go-to tool for data imports, saving me both time and manual effort.

Recently, while preparing to enhance my talk on DuckDB (see in my [speaker portfolio](#)), I revisited tests with very large datasets. Using ChatGPT, I generated a dataset simulating measurements from hypothetical probes. Here is the schema:

```
CREATE TABLE probe_measurements (  
  measurement_id BIGSERIAL PRIMARY KEY,  
  probe_id INTEGER NOT NULL,  
  measurement_time TIMESTAMP NOT NULL,  
  temperature NUMERIC(5,2) NOT NULL,  
  pressure NUMERIC(6,2) NOT NULL,  
  humidity NUMERIC(5,2) NOT NULL,  
  voltage NUMERIC(5,2) NOT NULL,  
  current NUMERIC(5,2) NOT NULL,  
  resistance NUMERIC(10,2) NOT NULL,  
  sample_rate INTEGER NOT NULL);
```

Using the generated Python code, I created a 100 GB CSV file, resulting in 1.42 billion records. As detailed in “[DuckDB Database File as a New Standard for Sharing Data](#)”, importing this data into PostgreSQL produced a 144 GB table, whereas DuckDB’s database file was a compact 27 GB. Most commands worked flawlessly, but issues arose when attempting pivot operations on this massive dataset.

## Pivoting Challenges

Creating pivot tables with many columns in plain SQL is challenging because it typically requires extensive manual input. However, as shown in “[Easy and Intelligent Pivot Tables with DuckDB](#)”, DuckDB makes it easier to refine pivot queries incrementally. Yet, dataset size matters. While smaller datasets allow iterative improvements without problems, very large datasets reveal some limitations.

For instance, the following badly designed pivot query executed on a smaller portion of my table worked reasonably well, but it produced hundreds of millions of rows and dozens of columns, showing me that I had to improve the query significantly:

```
PIVOT probe_measurements ON measurement_time::date  
USING AVG(temperature) AS avg_temperature;
```

The issue with this query is its lack of grouping and date boundaries, leading to as many rows as the table and as many columns as unique dates in the *measurement\_time* column. For meaningful analysis, a more focused query is necessary:

```
PIVOT (  
  SELECT probe_id, measurement_time, temperature  
  FROM probe_measurements  
  WHERE measurement_time BETWEEN '2024-12-01' AND '2024-12-30')  
ON measurement_time::date  
USING AVG(temperature) AS avg_temperature GROUP BY probe_id;
```

However, when testing that original unfocused pivot query on the massive *probe\_measurements* table (1.42 billion rows), DuckDB's memory usage skyrocketed. Despite setting *max\_memory/memory\_limit* of 50 GiB, memory usage quickly reached 98% of both RAM and swap. DuckDB then began creating large temporary files, consuming up to 85 GiB (the value of *max\_temp\_directory\_size* setting) before failing.

## Performance Observations

Here are the metrics and errors observed during the tests:

- Initial Run (85 GiB Limit):

```
Run Time (s): real 331.251 user 378.160763 sys 421.816367
```

```
Out of Memory Error: failed to offload data block of size 256.0 KiB (85.8 GiB/85.8 GiB)
This limit was set by the 'max_temp_directory_size' setting.
```

```
By default, this setting utilizes the available disk space on the drive where the database is located.
You can adjust this setting, by using (for example) PRAGMA max_temp_directory_size = 1024;
```

- Increased Limit (270 GB Disk Space):

Despite increasing *max\_temp\_directory\_size* to 1 TB, only 270 GB was available on disk. DuckDB eventually crashed after nearly 15 minutes with a “no space left on device” error:

```
Run Time (s): real 847.012 user 669.826647 sys 1220.951889
```

```
IO Error: Could not write file "test.duckdb.tmp/duckdb_temp_storage-8.tmp": No space left on device
```

## Performance of Properly Designed Query

On the other hand, when I used a properly designed pivot query, results were delivered amazingly quickly without exhausting any resources.

```
PIVOT (
  SELECT probe_id, measurement_time, temperature
  FROM probe_measurements
  WHERE measurement_time BETWEEN '2024-12-01' AND '2024-12-30'
  ON measurement_time:date
  USING AVG(temperature) AS avg_temperature GROUP BY probe_id;
100%
100%
```

probe_id	2024-12-18 avg_tem.	2024-12-19 avg_tem.	2024-12-20 avg_tem.	2024-12-21 avg_tem.	2024-12-27 avg_tem.	2024-12-28 avg_tem.	2024-12-29 avg_tem.	2024-12-30 avg_tem.
int64	double	double	double	double	double	double	double	double
1	-0.8933333333333331	0.008709499040937112	-0.0953097981531423	0.01928555229008652	-0.024082900616364426	-0.06485498315668406	0.012230606453306028	2.89
2	-1.8681481481481474	-0.011747858418378	-0.1913336817521093	-0.06358527768649375	-0.04171747826158102	-0.0284335947669743	-0.025157668080887	-31.82
3	0.5514285714285712	0.03776647082605746	0.27300376356349165	0.1404810777274826	0.011376852696737617	0.0502324666059877	0.08683195299201234	-3.990000000000001
4	0.15520325203252047	-0.08069218717719168	0.08583476862546609	-0.06139660214476116	0.19196071702861273	-0.08186630396370788	0.08936674468468202	19.36
5	0.5163888888888885	-0.04578049073644817	0.01953642221439854	-0.015513540808762	-0.013590022754125	-0.07238289422942779	-0.18353784242348998	-27.380000000000003
6	1.016548672566371	0.029071114059955196	0.08107466316960297	0.017795231646764302	0.11904802583941525	0.04725992074933865	0.08015156107911504	-10.3
7	0.07568807339449547	0.14391917325689826	0.055126131347262220	-0.09983962752190843	0.10227462051470873	-0.15890324113132445	-0.05464776857339331	17.95
8	2.834600000000001	0.1148238688388646	0.09522098292944779	0.06277199528672414	0.10036929933078622	0.13754822380761314	-0.05208853538214823	-19.71
9	1.5455056119775286	0.05855687778879753	-0.03765412237862671	0.04697267741617703	-0.06521116188527977	-0.003208910688739	-0.067795151830081	13.08
10	-0.6838000000000001	-0.28492896786129136	-0.08407264992227768	-0.055358886158886	0.06143462707866434	0.08977096262267186	0.07613119220133417	-10.3
11	-3.7764347826086957	-0.18416217933853829	0.07367137147592087	0.18342240982852617	0.0469199417019546	0.0842402263436545	0.07204998928117939	45.96
12	1.5206066666666666	0.2106927160444198	0.10893850718864263	-0.05575693574662155	0.06861218940920189	-0.007777932469061	0.039384196880641206	4.923333333333334
13	-0.447196261682432	-0.04059074910542247	0.05239493113863208	-0.0214055639940501	-0.03392269779570314	0.07835466119365161	-0.03261512269386271	-1.7650000000000006
14	-2.6013392857142858	0.07316688236305598	-0.050057994486842	-0.02237049309405826	0.039494545004054426	0.07728930817610073	-0.03379195482260548	5.0200000000000006
15	-2.2092391304347827	-0.005144966824155	-0.056626768137670	0.002132036763498929	-0.0502924418318254	0.05335375464928863	0.06826872289785106	20.44
16	-5.104600695452175	-0.3803382623814361	0.012928384872972334	-0.059370342858545	0.09010858405407832	0.03176683704763877	-0.08066881223794503	-13.65
17	4.781382978723406	0.4659882476708733	0.16312028213207262	-0.07717680773680778	0.03365820303722565	0.0891822381972314	0.0912665807993609	-17.515
18	0.09867256637168122	-0.002597352359666	0.038032066751196233	0.05646905733496645	-0.056157513598531	0.04528215540037215	-0.07462724619622213	8.267499999999998
19	2.0938851082829783	-0.011483227106448	0.02934949084170985	-0.1291783341520482	0.09808158720020972	-0.06237591444411059	-0.14364666666666666	7.306666666666666
20	0.6865853658536589	-0.1047552749549083	-0.02643816057244126	0.053324791695569904	-0.03488868099314119	0.004723957690249741	-0.09954826202597237	-17.515
901	6.274810181818182	-0.05899642646971017	-0.15701241460368753	-0.014281049483521	-0.005729976703744055	0.07811998462155151	0.08303317574310037	35.695
902	4.146236559139785	-0.006218066473466	-0.12709462668479813	0.07320853030892017	-0.1600774813251191	0.0847571846019098	0.06076915538762025	45.92
903	-2.2424489795918374	0.1292289729655479	-0.01578956372968345	-0.11036742167256543	0.09850752882287772	0.08381679949749958	0.1201148831511466	26.12
904	-4.8618918918918919	-0.014959834176213	-0.06395120195650242	0.10667740909953037	0.046130732866371114	0.10371431376077715	-0.1031907299622613	9.035
905	-1.8353846153846156	0.00329638458702291	0.09753330664989428	0.14356711686992685	-0.146125083888932	0.04264527854992811	0.045275777407355354	45.31
906	-4.5237272727272724	0.021925843797540274	0.018245244860026953	-0.018245244860026953	0.007001466575995972	0.05816651930445028	0.09480504012551043	8.267499999999998
907	0.9466936930693031	-0.17302650062401223	-0.123581527478735527	-0.04427029467290464	-0.0031278964522367	-0.057873365677165	0.05902411002970675	7.306666666666666
908	-0.969747706422023	-0.02859900578515256	-0.01257230155877223	0.024569387622198974	-0.04804684167350322	0.004980424344399889	-0.0681810149468637	-17.515
909	3.5601886792452837	0.153967848557685617	0.09489573405199327	-0.1713985185185183	0.0817664332483276	0.0672998507794916	-0.09539878056797271	6.27
910	-4.490662489638554	0.0126843873180816	0.1376593654106001	-0.1605113371668568	0.0382005065532511	-0.01058317805728	-0.024889568211442	2.5549999999999997
911	-2.127849462365992	-0.06413519898512372	0.03634113253189271	0.05266666992410176	0.1172017821684689	0.0043520666936853	-0.082490663653953	-46.48
912	-0.580091743119266	-0.0203666279019639	0.06159776005763601	-0.07686053072330868	-0.11079811859732074	0.07561719025622922	-0.002432837532352	-17.515
913	3.4088988254680972	-0.0358374650840987	-0.0327235366839524	-0.05926547933085	-0.005098547548169	-0.030956338743224	-0.10192112685241057	40.25
914	1.014887218845113	0.06102590214551422	-0.02018151491365789	0.09384264049610529	-0.09856344801167159	0.10671040536775777	0.0990104409360255	-4.031999999999998
915	-2.29923076923077	0.20774787867194425	0.04415988120099207	-0.0555594590360652	-0.0670810119571237	0.05968745899488238	0.011847770143864587	-10.93
916	1.4535245901693949	0.02414919136423966	0.04559683800920881	-0.13891605347874647	-0.00022091745526095	-0.032139952113877	0.04877205139074573	-10.93
917	-2.591138211382114	-0.08115191341321969	0.06748401691747802	-0.12257086855989407	-0.07479103742613288	0.09856108471698426	0.028141248435979503	-10.93
918	2.2799999999999994	-0.06084394631148976	0.1306441237113398	-0.13457757095103934	0.1202241752121354	-0.009107387029561	-0.12827528011262124	-10.93
919	1.0701010101010102	0.17764822066501787	0.01700966001344825	-0.1071036931238524	-0.11884668304668297	0.067664259352482	0.05315475995143244	-10.93
1000	-0.6461654135338342	0.02934094501109798	0.04084012120365214	-0.009584065728453	-0.05436323954044496	0.0339513640493399	-0.07553808902334414	-10.93

Result was delivered in amazing 15 seconds, and DuckDB was running calculations massively in parallel (machine has 20 cores).

Run Time (s): real 15.471 user 293.897610 sys 5.348931

## Summary

These findings suggest that DuckDB's pivot command could benefit from a pre-check to warn about potential output size issues. For large datasets, users should first analyze the data and carefully design queries with appropriate filters and groupings to avoid resource exhaustion. While these challenges highlight the tool's limitations, they also underline its amazing potential for handling large-scale data with properly optimized queries.





Image created by the author using DeepDreamGenerator

Duckdb

Pivoting

Pivot Tables

Data Analysis

**Written by Josef Machytka**[Edit profile](#)

47 Followers · 10 Following

I work as PostgreSQL specialist & database reliability engineer at NetApp Deutschland, Open Source Services division.

## No responses yet



What are your thoughts?

[Respond](#)

## Recommended from Medium





Carlyn Beccia



## Trump Fiddles While Musk Burns Down MAGA

Musk unleashes his anger on MAGA Republicans and vows "war."



5d ago



15.6K



316



Mark Manson



## 40 Life Lessons I Know at 40 (That I Wish I Knew at 20)

Today is my 40th birthday.

Sep 23, 2024



34K



781

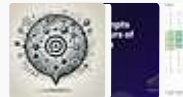


### Lists



#### Practical Guides to Machine Learning

10 stories · 2124 saves



#### ChatGPT prompts

51 stories · 2421 saves



#### Staff picks

791 stories · 1544 saves




#### Natural Language Processing

1882 stories · 1516 saves






 Karolina Kozmana

## Common side effects of not drinking

By rejecting alcohol, you reject something very human, an extra limb that we have...

Jan 21, 2024  53K  1516

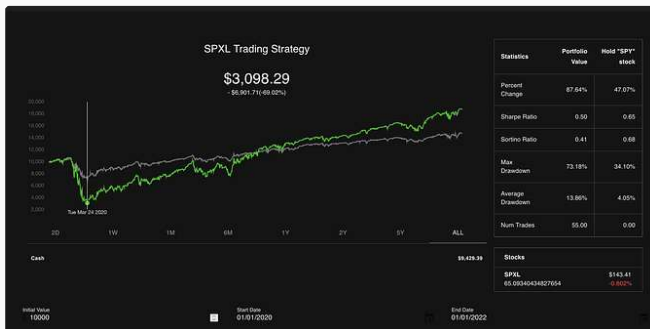



 Da Money Hacker

## 3 Legit Websites That You Can Use to Earn an Extra \$400/Day

Easy as pie...

Aug 17, 2024  3.8K  150



 In DataDrivenInvestor by Austin Starks

## I just tried OpenAI's updated o1 model. This technology will BREA...

All of my articles are 100% free to read! Non-members can read for free by checking out...

★ Dec 22, 2024  2.6K  68



 In Human Parts by Devon Price 

## Laziness Does Not Exist

Psychological research is clear: when people procrastinate, there's usually a good reason

★ Mar 23, 2018  337K  1971



See more recommendations