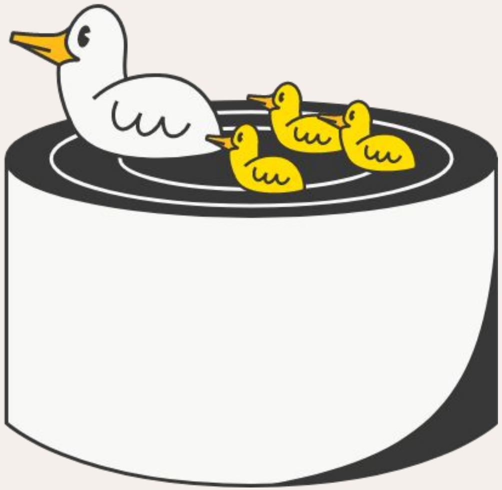
MotherDuck

AI & Machine Learning at MotherDuck



Who is MotherDuck…

MotherDuck

Founded in : May 2022

General Availability : June 2024

Employees: ~55

Locations: Seattle (HQ), SF, NYC, Amsterdam

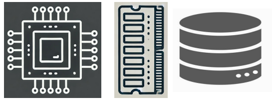
Funding: $100M (Series B)

DuckDB Labs Partnership

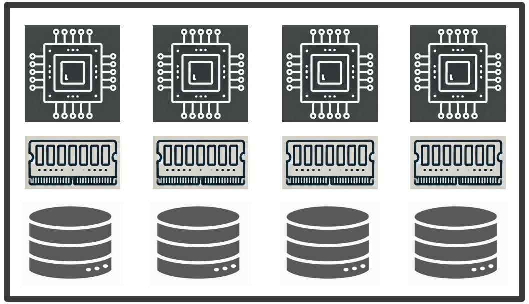
Amsterdam Offsite 2024

DESIGNING SYSTEMS FOR THE POST-BIG DATA WORLD

Laptop



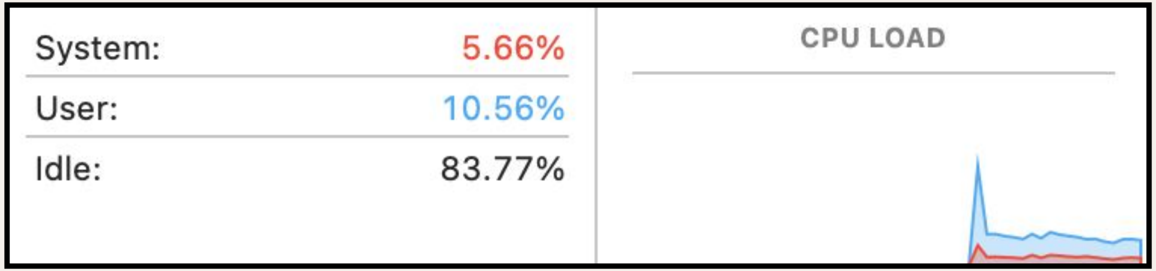
Cloud / Single Node



- Leverage Local Compute and Storage
- Leverage Cloud for Scale up and Collaboration
- Avoid the big data tax

Object Store

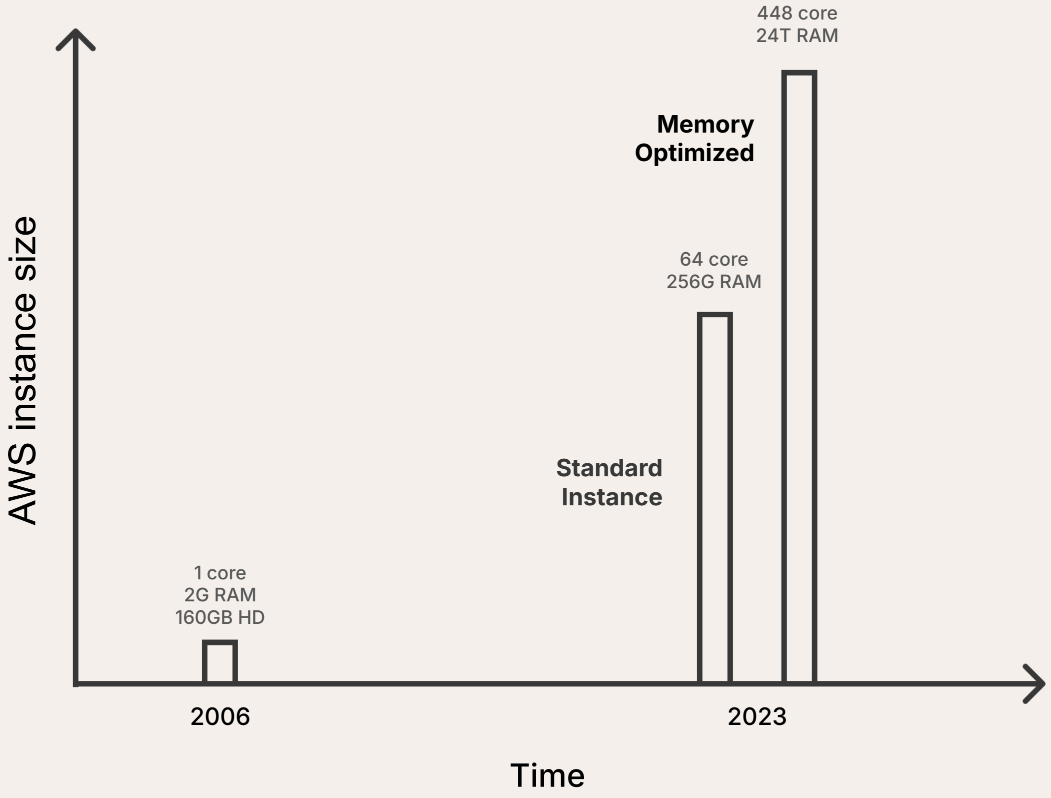
LOCAL COMPUTE MOSTLY JUST SITS IDLE



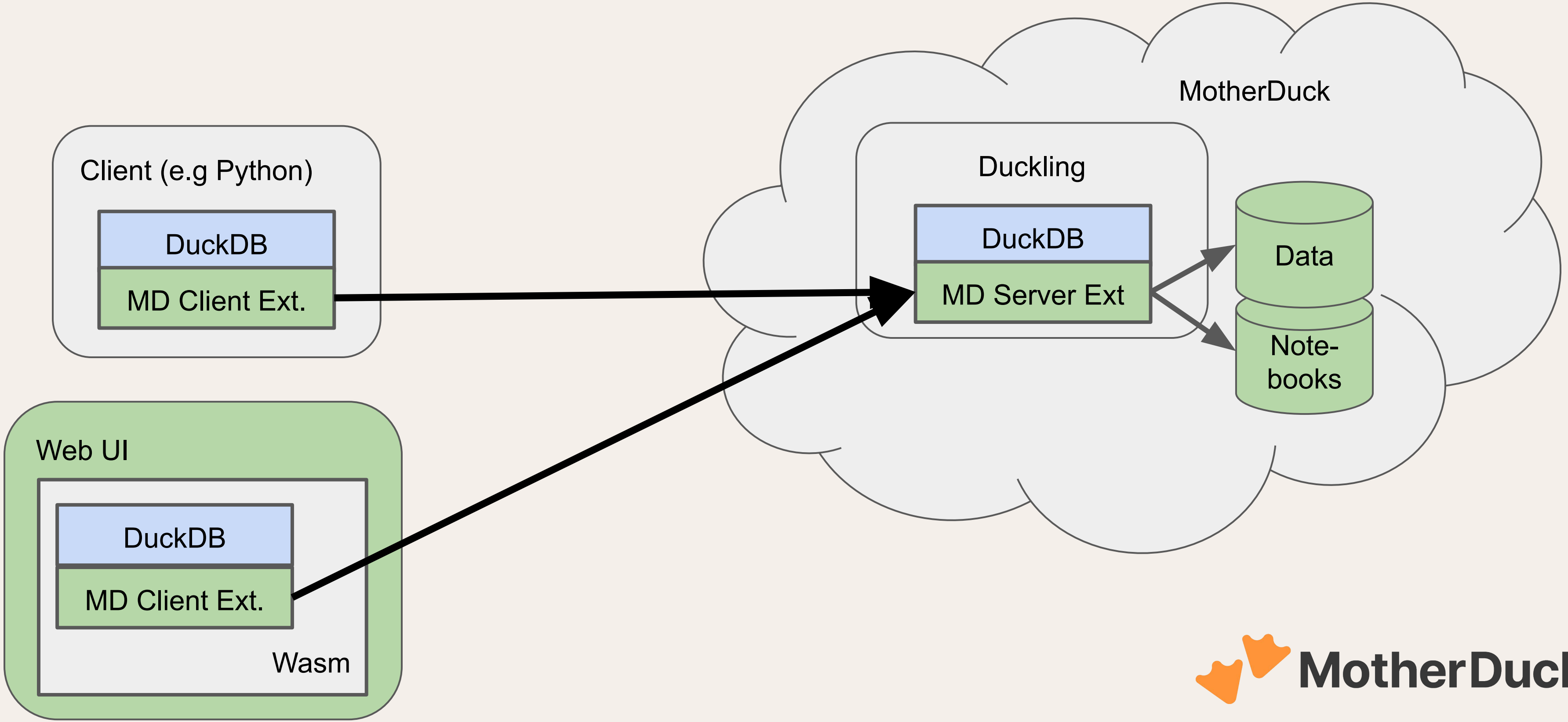
DISTRIBUTED COMPUTE IS STILL PAINFUL

2 worker nodes processing bottles

SINGLE NODE CLOUD COMPUTE IS REALLY POWERFUL



DuckDB Extension for Scale Up & Collaboration



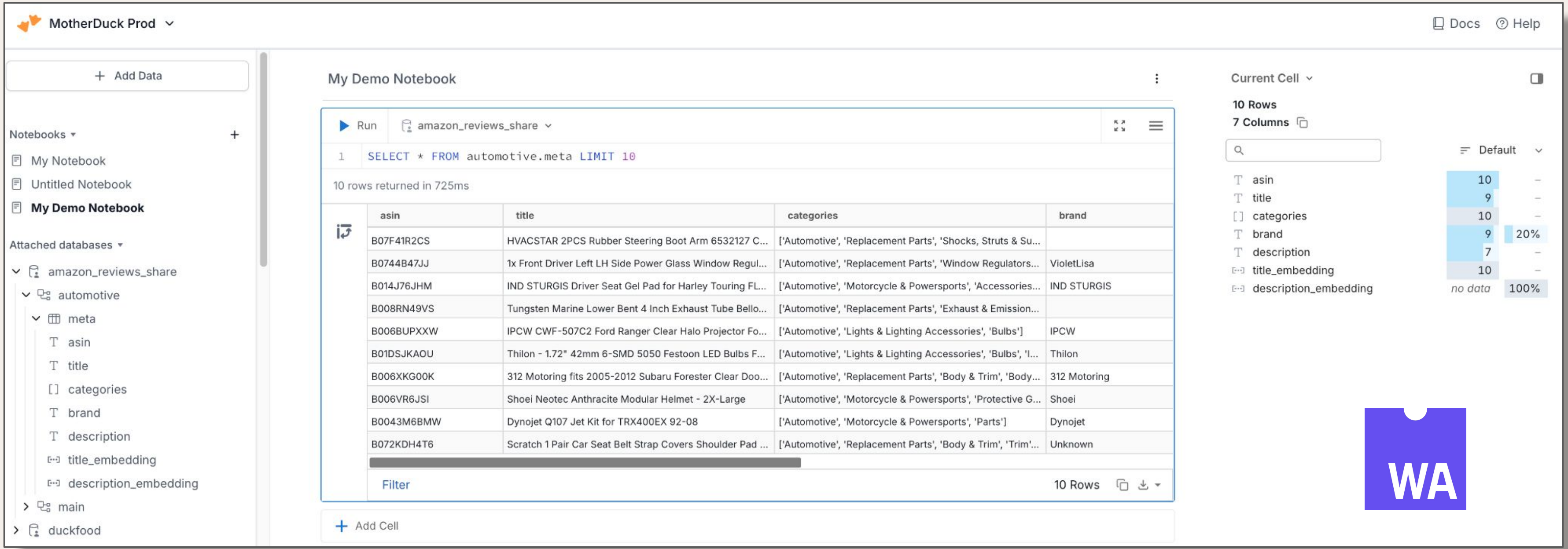
MotherDuck

Using MotherDuck - As simple as..

```
duckdb md:

v1.1.1 af39bd0dcf
Enter ".help" for usage hints.
D SHOW ALL DATABASES;
┌──────────────────────┬─────────────┬──────────────────┬───────────────────────────────────────────────────────────────┐
│        alias         │ is_attached │       type       │                     fully_qualified_name                      │
│       varchar        │   boolean   │     varchar      │                           varchar                             │
├──────────────────────┼─────────────┼──────────────────┼───────────────────────────────────────────────────────────────┤
│ amazon_reviews_share │ true        │ motherduck share │ md:_share/amazon_reviews_share/f09b8455-74d7-45ab-ad4b-2041da392ab0 │
│ duckfood             │ true        │ motherduck share │ md:_share/duckfood/faf5982e-9978-41b0-8eb1-7613d08a1b42       │
│ local_duck           │ true        │ duckdb           │                                                               │
│ my_db                │ true        │ motherduck       │ md:my_db                                                      │
├──────────────────────┴─────────────┴──────────────────┴───────────────────────────────────────────────────────────────┤
│ 4 rows                                                                                                      4 columns │
└────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────┘
```

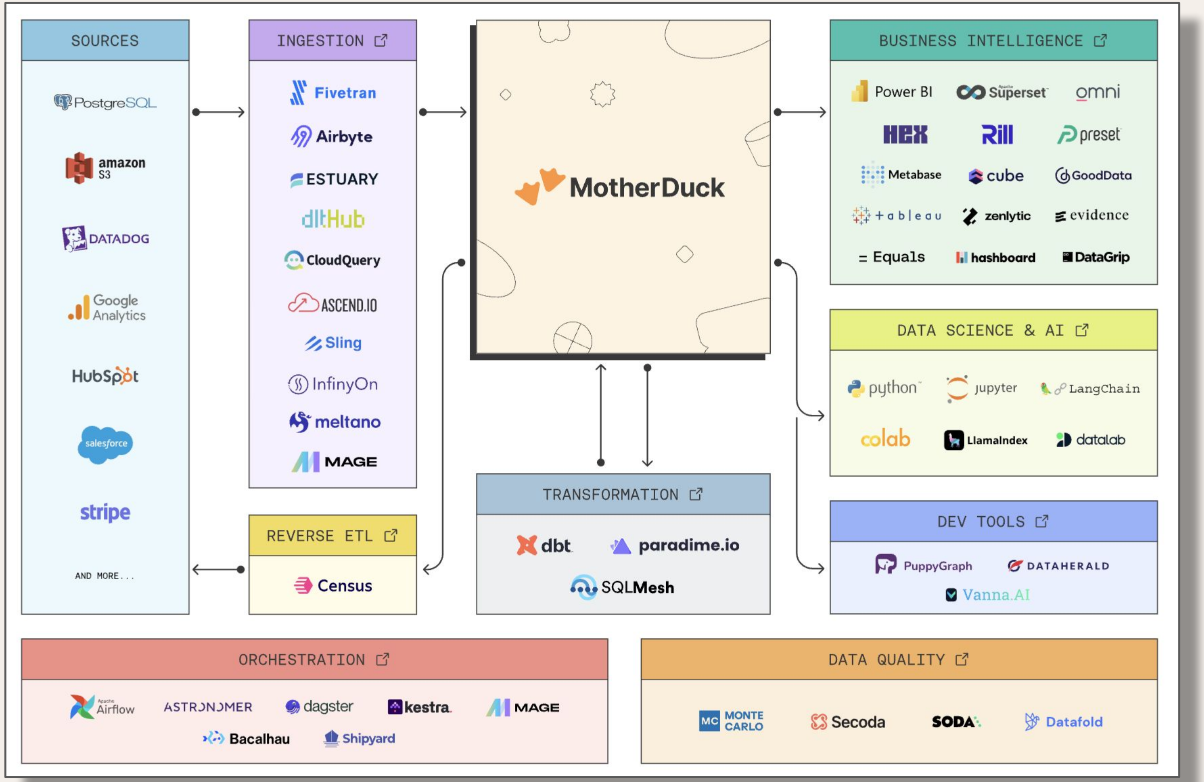
MotherDuck

Web UI - app.motherduck.com

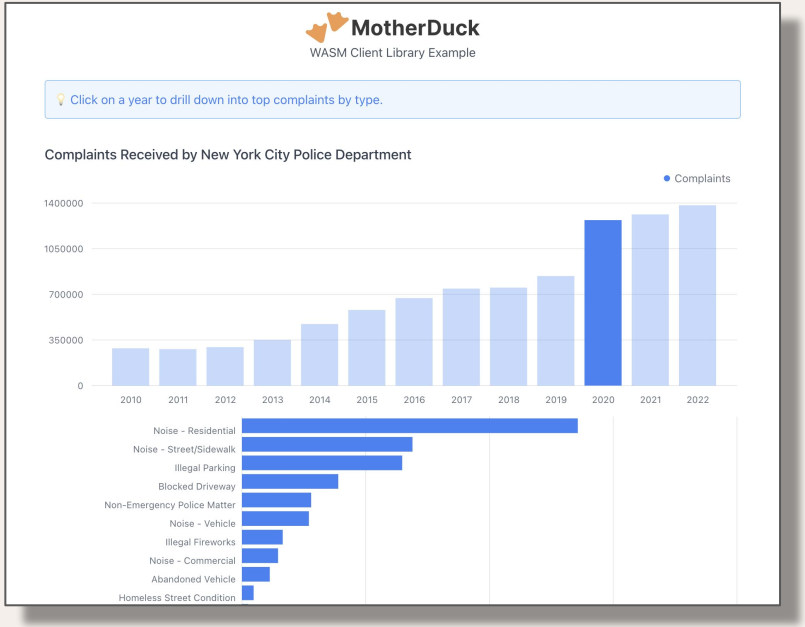


MotherDuck

And there is more…
Partner Ecosystem



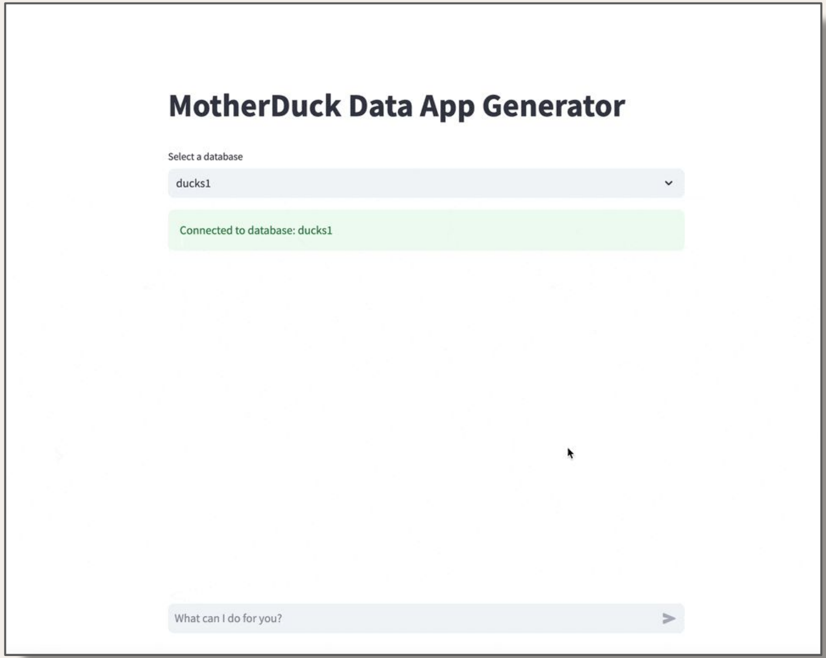
And there is more…
WASM SDK for Low-Latency Data Apps



https://www.npmjs.com/package/@motherduck/wasm-client

https://github.com/motherduckdb/wasm-client/tree/main

And there is more…
AI



https://github.com/motherduckdb/wasm-client/tree/main/data-app-generator

MotherDuck

NL to SQL

Text Processing with LLMs in SQL

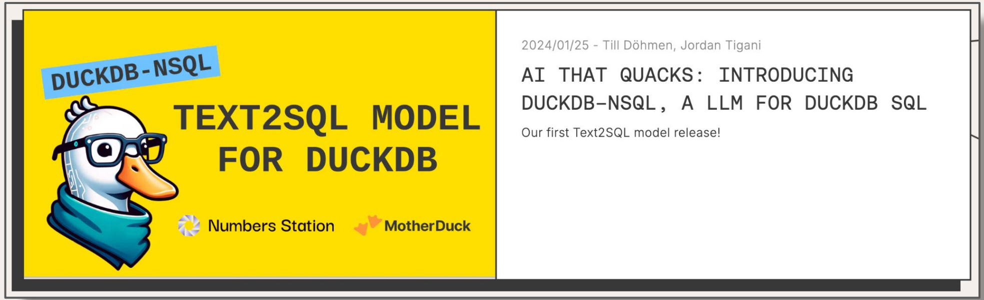
Embeddings & Vector Search

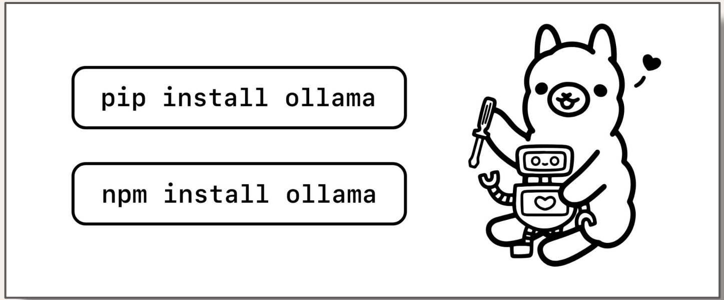
Classical Machine Learning

Let's Start With A Quiz

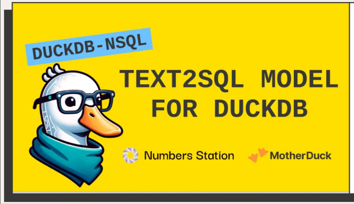
- get all columns ending with _amount from taxi table
- get a 10% reservoir table sample of rideshare table
- show summary statistics of rideshare table

```
doehmen@Tills-MacBook-Pro mono % ollama run duckdb-nsql
>>> get all columns ending with _amount from taxi table
 SELECT COLUMNS('.*_amount') FROM taxi;

>>> get a 10% reservoir table sample of rideshare table
 SELECT * FROM rideshare TABLESAMPLE RESERVOIR(10%);

>>> show summary statistics of rideshare table
 SUMMARIZE rideshare;
```





https://huggingface.co/spaces/motherduckdb/DuckDB-NSQL-7B



2024/01/25 - Till Döhmen, Jordan Tigani

AI THAT QUACKS: INTRODUCING DUCKDB-NSQL, A LLM FOR DUCKDB SQL

Our first Text2SQL model release!

motherduckdb/**DuckDB-NSQL-7B-v0.1** like 79

Text Generation · Transformers · Safetensors · llama · text-generation-inference · Inference Endpoints

Model card · Files and versions · Community 3 · Settings

DuckDB-NSQL-7B

Model Description

NSQL is a family of autoregressive open-source large foundation models (FMs) designed specifically for SQL generation tasks.

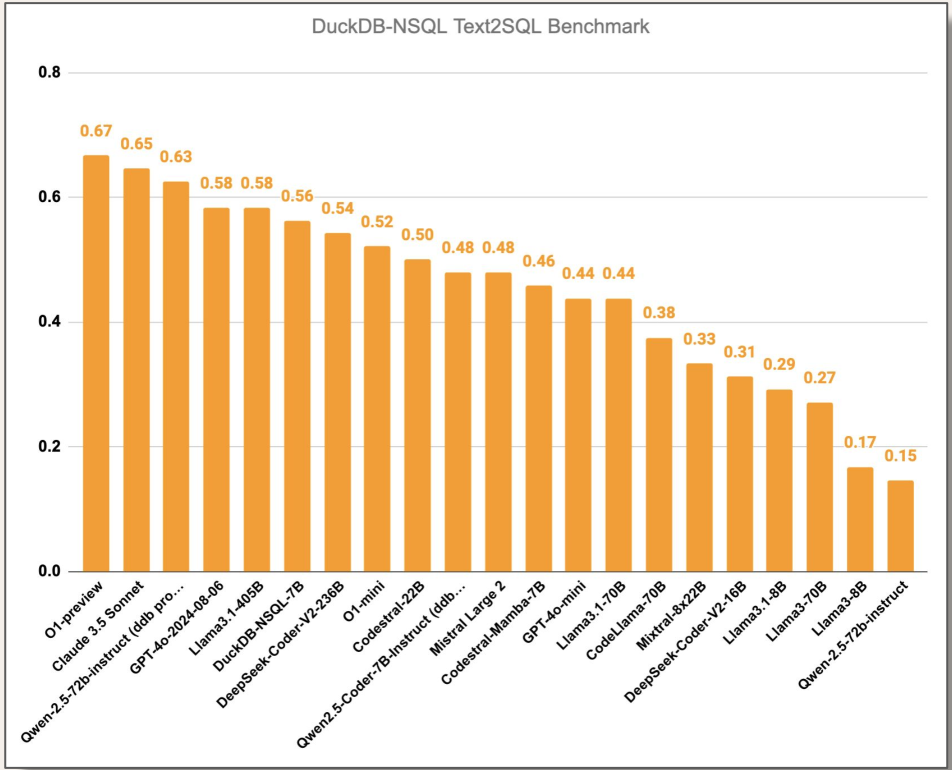
In this repository we are introducing a new member of NSQL, DuckDB-NSQL. It's based on Meta's original Llama-2 7B model and further pre-trained on a dataset of general SQL queries and then fine-tuned on a dataset composed of DuckDB text-to-SQL pairs.

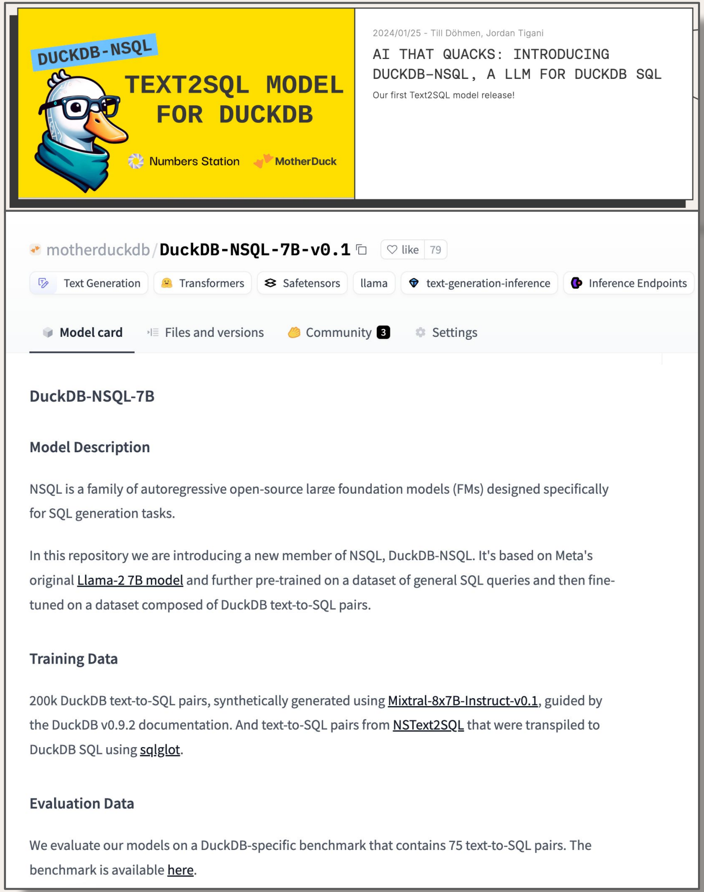
Training Data

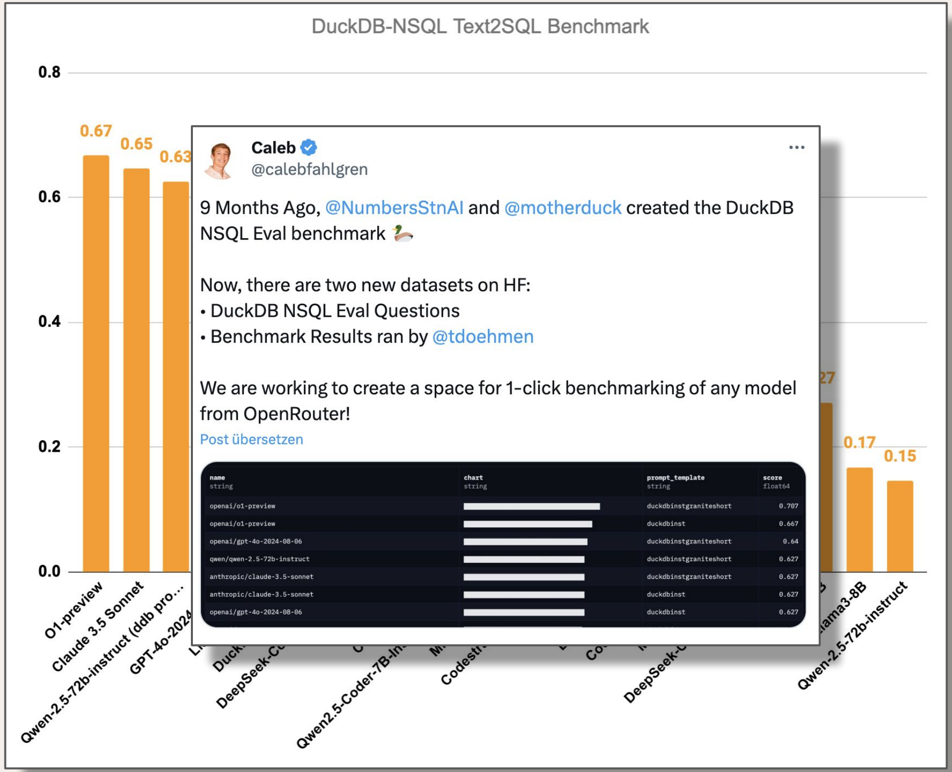
200k DuckDB text-to-SQL pairs, synthetically generated using Mixtral-8x7B-Instruct-v0.1, guided by the DuckDB v0.9.2 documentation. And text-to-SQL pairs from NSText2SQL that were transpiled to DuckDB SQL using sqlglot.

Evaluation Data

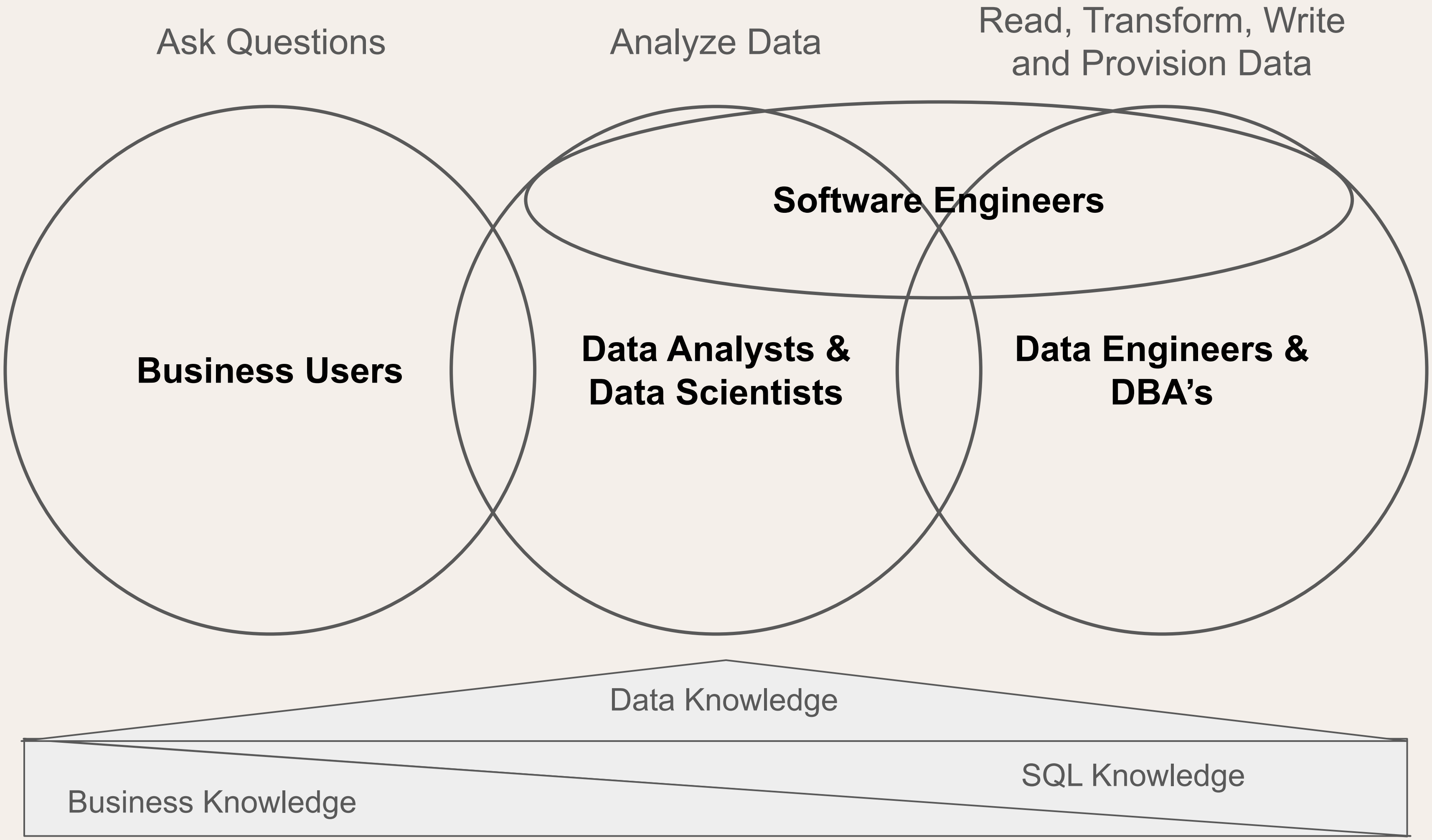
We evaluate our models on a DuckDB-specific benchmark that contains 75 text-to-SQL pairs. The benchmark is available here.

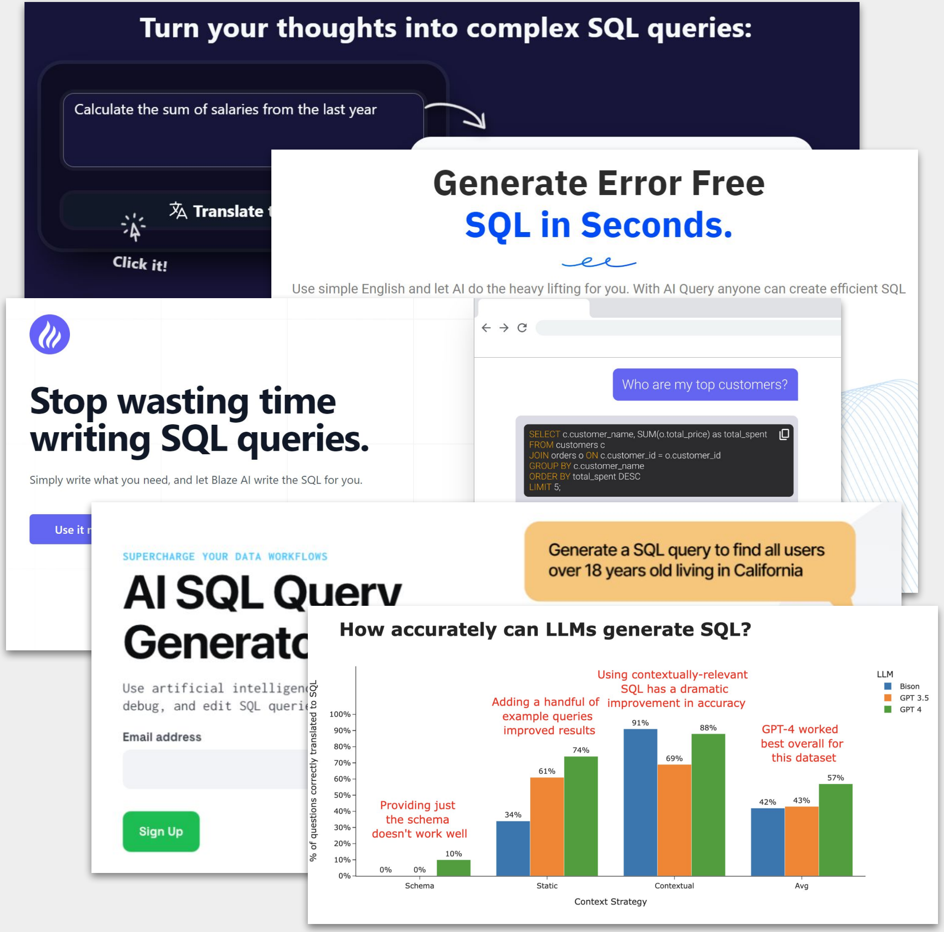






https://huggingface.co/sql-console


Ask Questions
Analyze Data
Read, Transform, Write and Provision Data
Software Engineers
Business Users
Data Analysts & Data Scientists
Data Engineers & DBA’s
Data Knowledge
SQL Knowledge
Business Knowledge



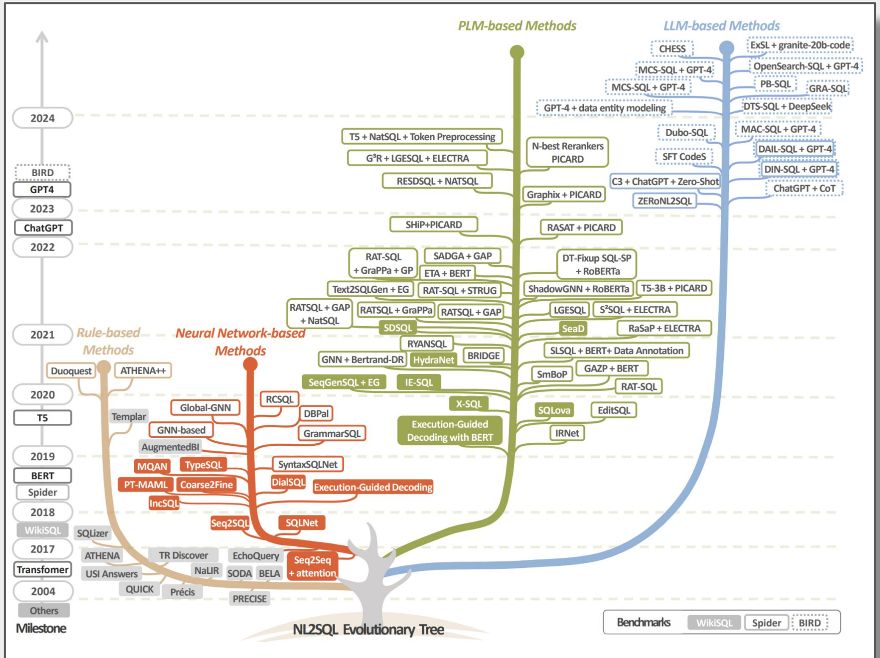


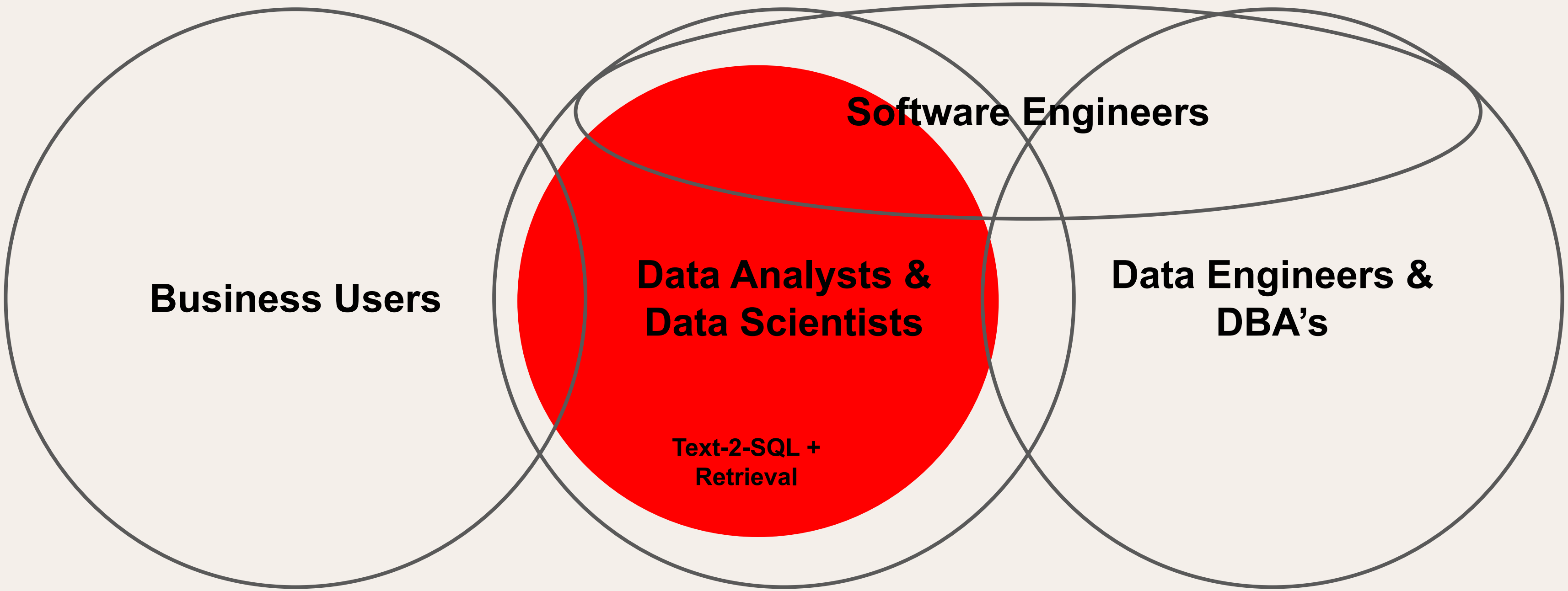
Figure 1: An Overview of NL2SQL Methods.

The Dawn of Natural Language to SQL: Are We Fully Ready? Boyan Li, Yuyu Luo, Chengliang Chai, Guoliang Li, and Nan Tang, VLDB 2024

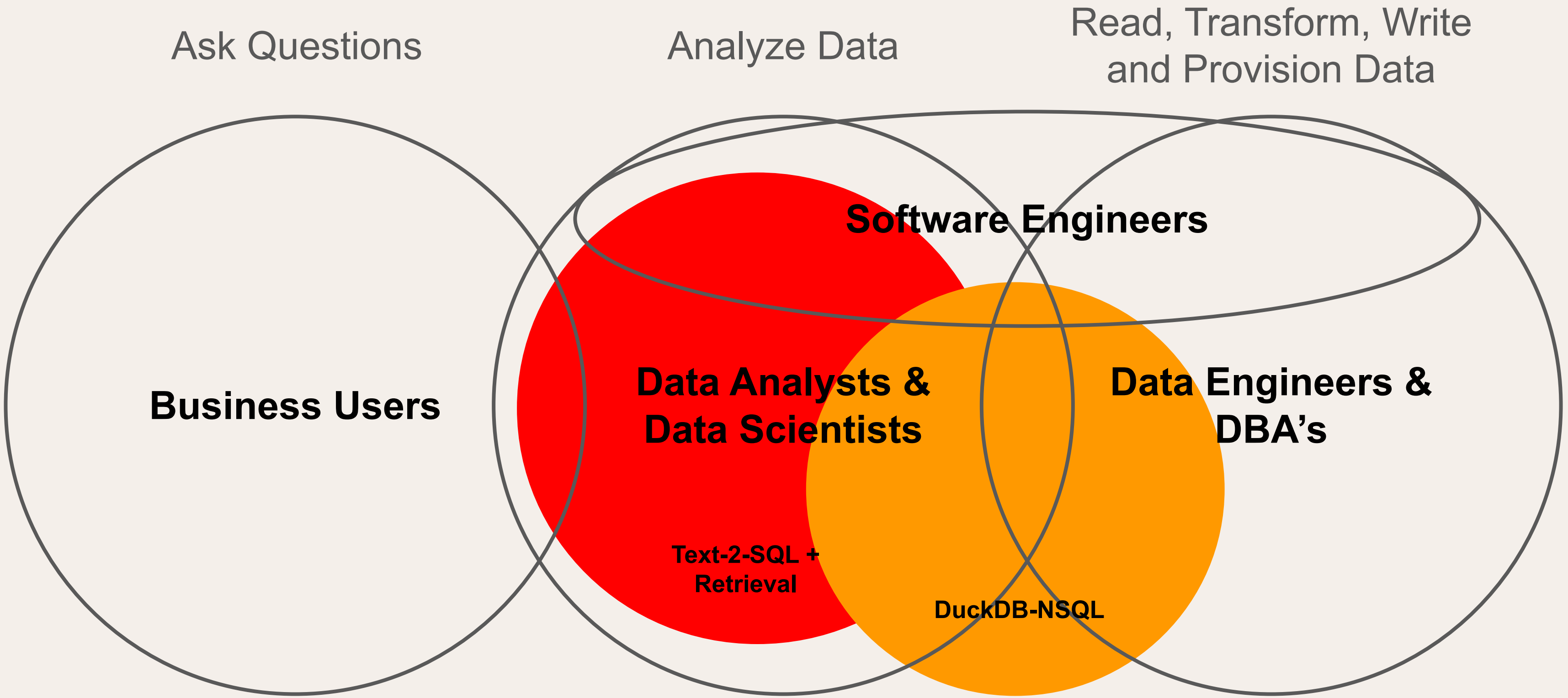
Ask Questions

Analyze Data

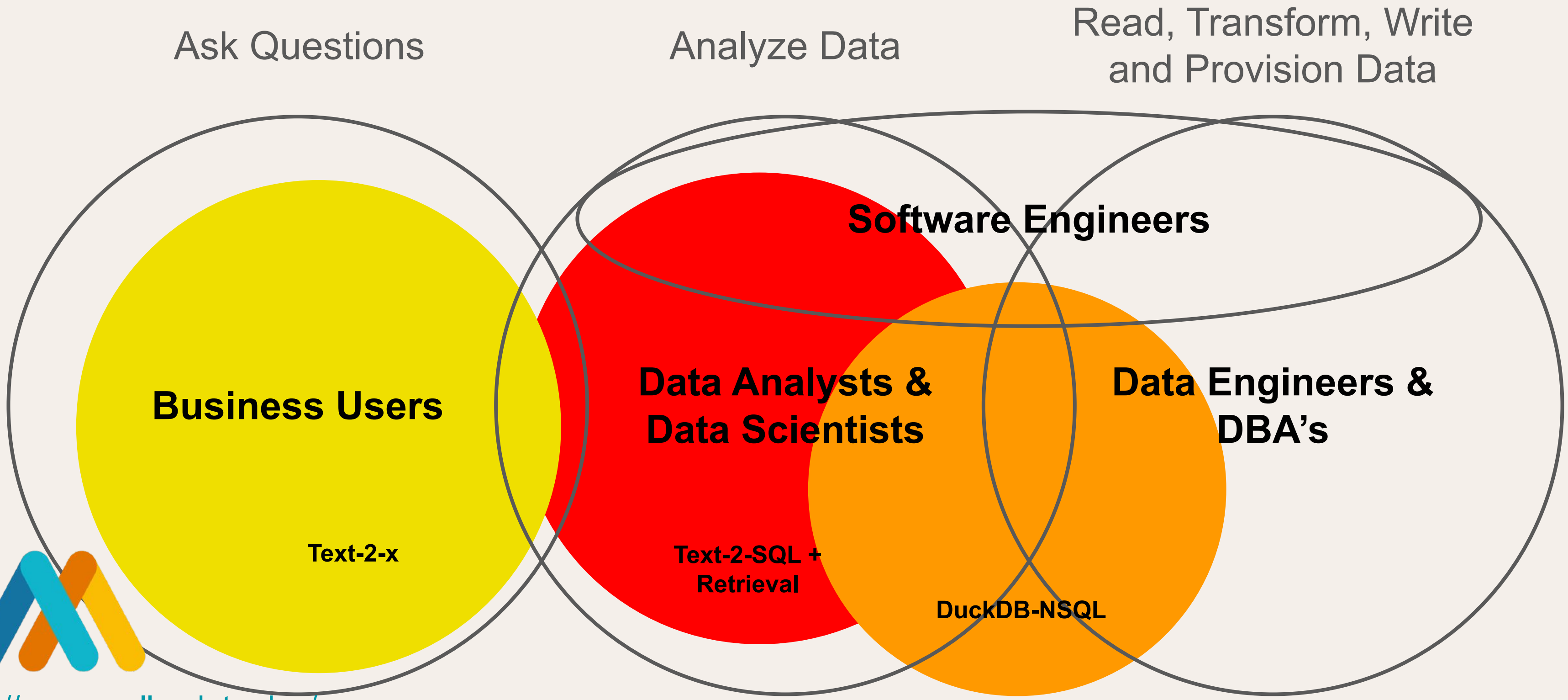
Read, Transform, Write and Provision Data



- Drafts for Analytical Queries
- Requires Data & SQL Knowledge for Verification



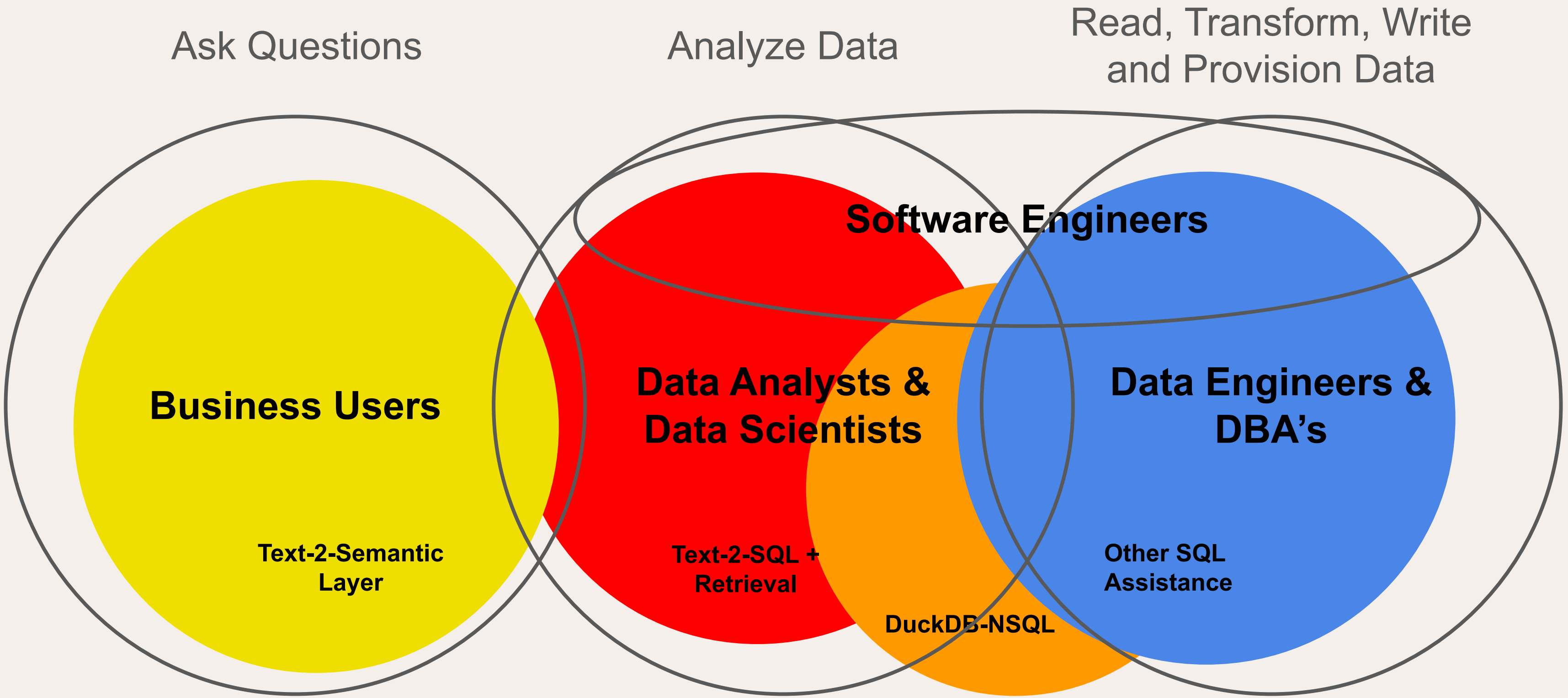
- Simple DuckDB SQL snippets for any type of statements
- Saves round trip to docs
- Not suitable for complex analytical queries



https://www.malloydata.dev/

- Regular NL2SQL not a fit
- Built-in Semantic Correctness
- Requires making tribal data knowledge explicit (lot of work!)

- Drafts for Analytical Queries
- Requires Data & SQL Knowledge for Verification

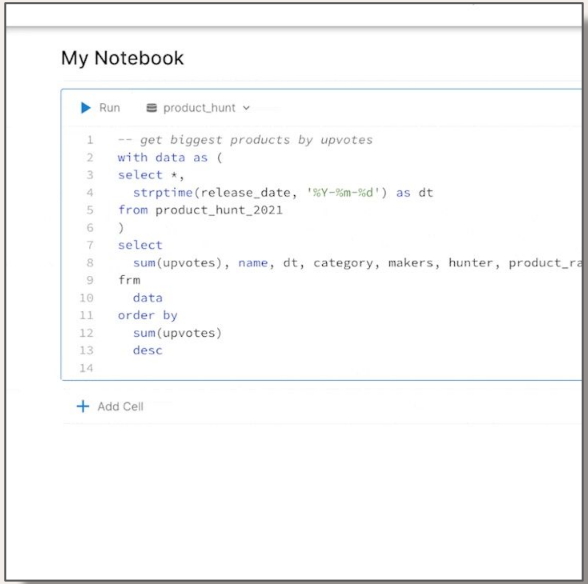


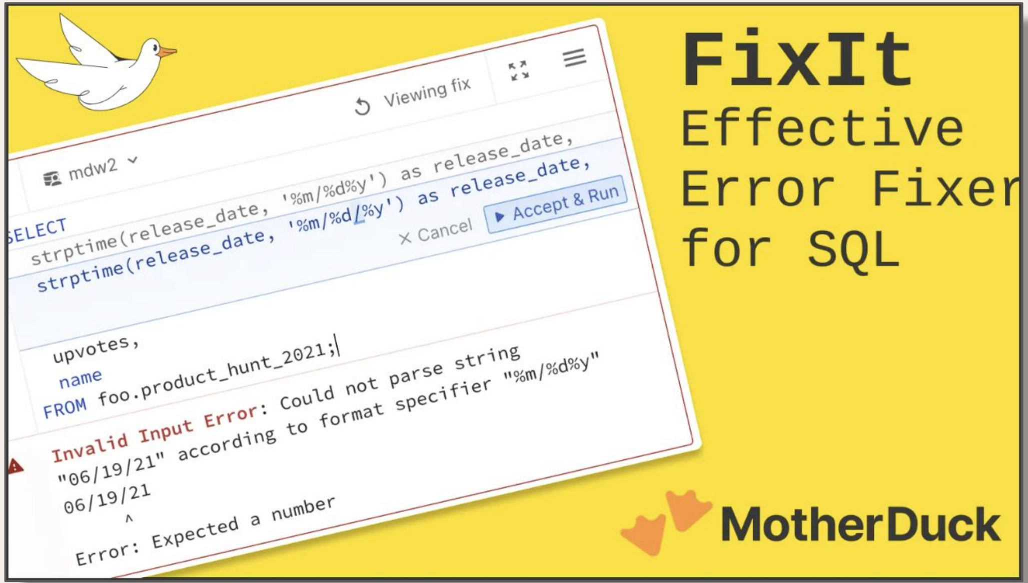
- Regular NL2SQL not a fit
- Built-in Semantic Correctness
- Requires making tribal data knowledge explicit (lot of work!)

- Drafts for Analytical Queries
- Requires Data & SQL Knowledge for Verification

- Focus on Dev. Experience
- Support for DDL / DML / ETL-Tasks

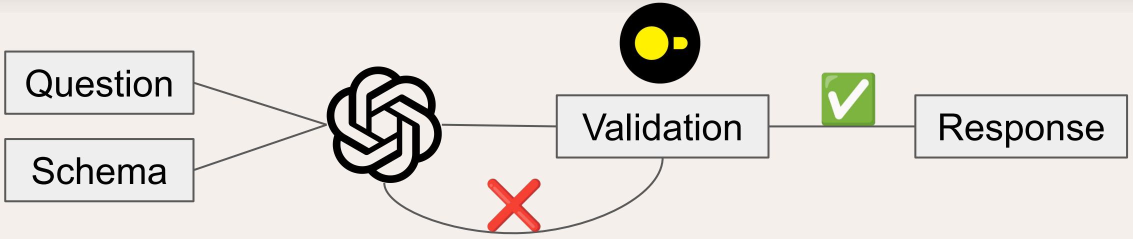
SQL Assistance in MotherDuck





SQL Assistance in MotherDuck

```
-- generate SQL
CALL prompt_sql('what are the top domains being shared on hacker_news?');

-- explain SQL
CALL prompt_explain('
SELECT COUNT(*) as domain_count,
SUBSTRING(SPLIT_PART(url, ''//'', 2), 1, POSITION(''/'' IN SPLIT_PART(url, ''//'', 2)) - 1) as domain
FROM hn.hacker_news
WHERE url IS NOT NULL GROUP BY domain ORDER BY domain_count DESC LIMIT 10;
');

-- fix SQL query
CALL prompt_fixup('SEELECT COUNT(*) as domain_count FROM hn.hackers');
```



MotherDuck

NL to SQL

Text Processing with LLMs in SQL

Embeddings & Vector Search

Classical Machine Learning

50% of database columns in the real world are strings!

Table 7: Column Data Type Distribution: *Shows the proportion of columns that use a particular data type and the proportion of columns marked as Predicate Columns by Redshift.*

Data Type	Stored Columns			Predicate Columns		
	Fleet	TPC-H	TPC-DS	Fleet	TPC-H	TPC-DS
varchar	52.1%	21.3%	11.2%	53.8%	15.6%	9.8%
numeric(P, S)	10.2%	14.8%	18.8%	7.0%	11.1%	8.8%
integer	9.1%	19.7%	44.5%	11.6%	22.2%	60.3%
bigint	7.0%	11.5%	-	9.4%	15.6%	-
timestamp w/o tz	6.2%	-	-	5.8%	-	-
double	4.5%	-	-	2.2%	-	-
boolean	3.9%	-	-	1.5%	-	-
date	2.2%	6.5%	2.6%	3.2%	8.9%	0.5%
smallint	2.1%	-	-	2.3%	-	-
char(N)	1.7%	26.2%	22.8%	2.4%	26.7%	20.6%
float	0.4%	-	-	0.2%	-	-
timestamp w/ tz	0.4%	-	-	0.4%	-	-

Why TPC Is Not Enough: An Analysis of the Amazon Redshift Fleet, van Renen et al., AWS, VLDB 2024

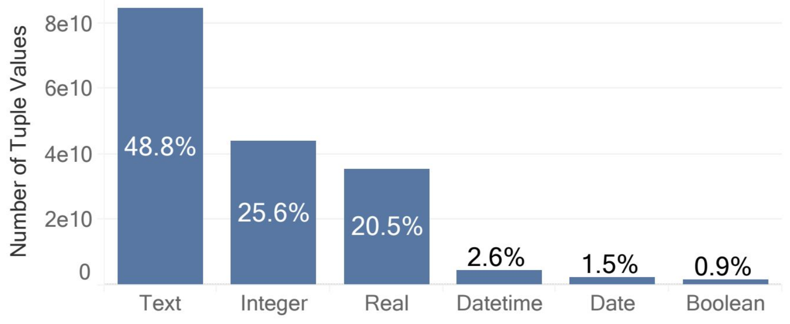
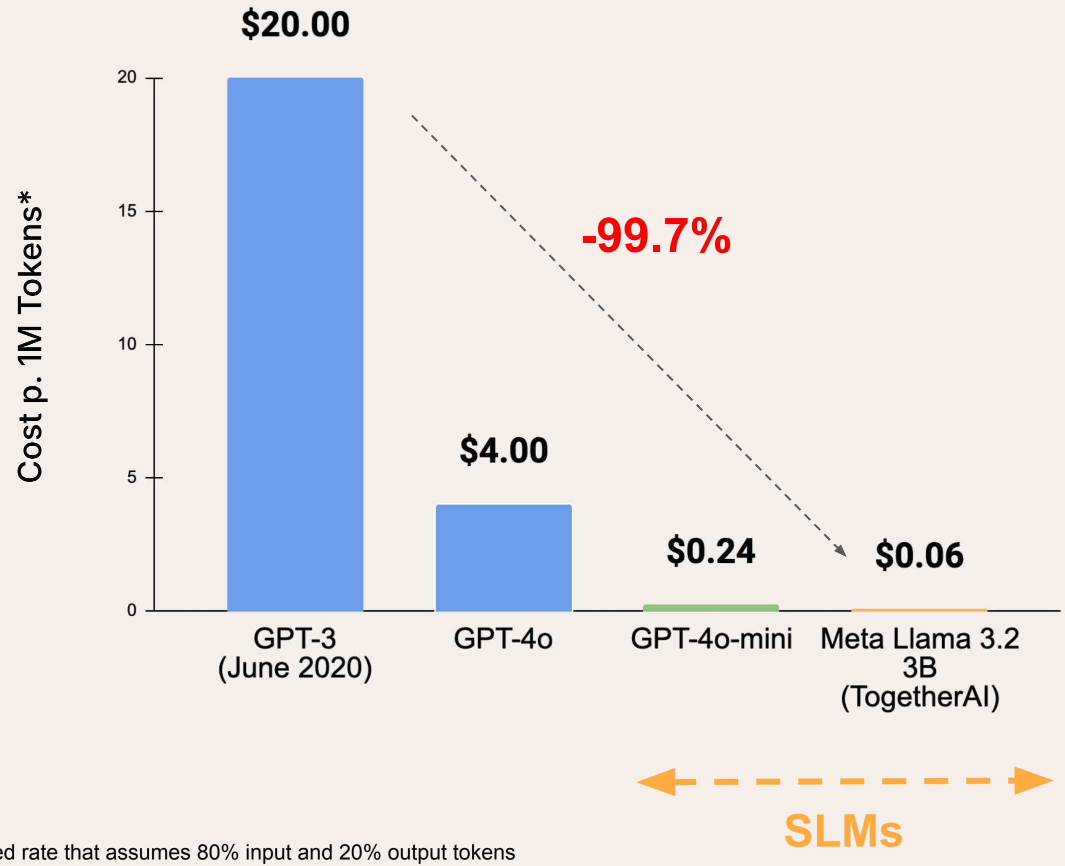


Figure 1: Distribution of data types used

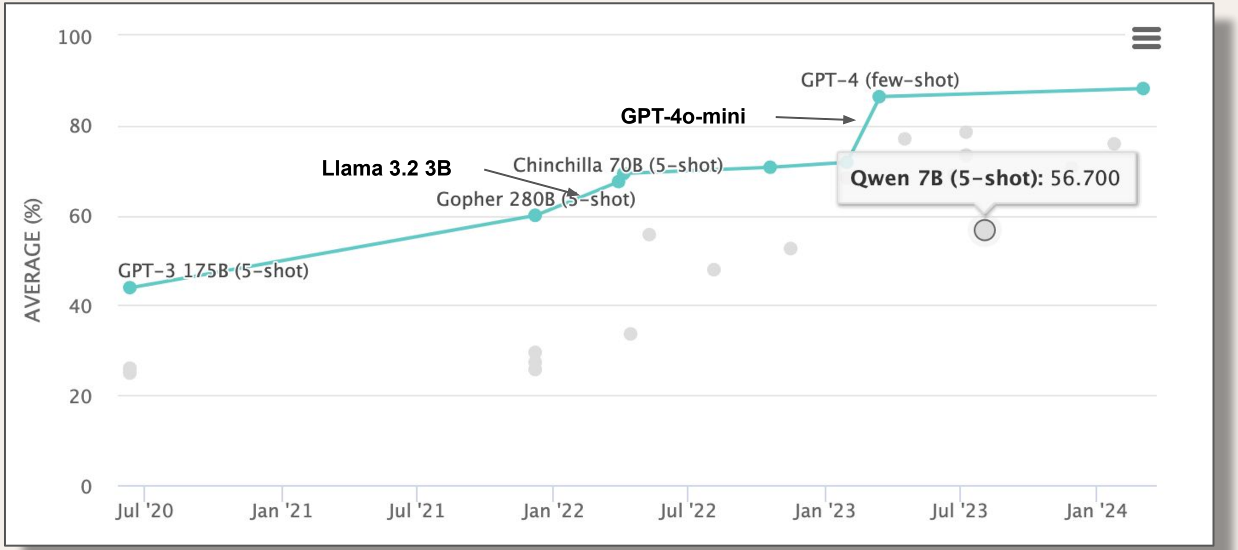
Get Real: How Benchmarks Fail to Represent the Real World, Vogelsgesang et al., Tableau Software, DBTest '18

Costs per Token have fallen dramatically in the past years.



* blended rate that assumes 80% input and 20% output tokens

Small Language Models have become powerful.



https://paperswithcode.com/sota/multi-task-language-understanding-on-mmlu

Prompt language models in SQL.

```sql
SELECT prompt('Write a poem about ducks', ‘gpt-4o’) AS response;
```

Text Summarization

```sql
SELECT by, text, timestamp,
       prompt('summarize the comment in 5 words: ' || text) AS summary
FROM hacker_news.hn
```

by	text	timestamp	summary
yrgulation	> In a word, gardening. It's very fulfilling.<p>M...	2022-08-22 18:37:49	Gardening brings fulfillment and joy.
paulmd	I actually don't know what you mean by that, but, ...	2022-08-22 18:37:46	Insurance policies generally favor insurers.
taylodl	59% of Americans are correct, but damn! Have they rea...	2022-08-22 18:37:49	Americans profit, worry about fairness.
0x457	Wrong or not, the point is that many websites that decla...	2022-08-22 18:37:33	XHTML validation issues frustrated developers.
nopehnnope	Some problems with that:<p>* The US had a single nati...	2022-08-22 18:37:37	EU diversity complicates potential federation.
jahewson	I dunno. I can imagine any of those points being the sub...	2022-08-22 18:37:27	Civil suits may target artists' responsibility.
idlehand	It took billions of years to get to that point, too. Comple...	2022-08-15 19:36:48	Complex life emerged during Cambrian explosion.
NeverFade	So we know there is racial discrimination against Asians...	2022-10-09 14:57:21	Racial discrimination against Asians acknowledged.
Comevius	Tegmark and Musk are both dumb people posing as int...	2022-10-09 14:57:21	Tegmark and Musk seek attention.

Structured Data Extraction

```sql
SELECT by, text, timestamp,
prompt(text,
  struct:={topic: 'VARCHAR', sentiment: 'INTEGER', technologies: 'VARCHAR[]'},
  struct_descr:={topic: 'topic of the comment, single word',
                 sentiment: 'sentiment of the post on a scale from 1 (neg) to 5 (pos)',
                 technologies: 'technologies mentioned in the comment'}) as my_output
FROM hn.hacker_news
LIMIT 100
```

Structured Data Extraction

by	text	timestamp	my_output
yrgulation	> In a word, gardening. It's very fulfilling.<p>M...	2022-08-22 18:37:49	{'topic': 'gardening', 'sentiment': 5, 'technologies': ['"au...
paulmd	I actually don't know what you mean by that, but, ...	2022-08-22 18:37:46	{'topic': 'insurance', 'sentiment': 3, 'technologies': []}
taylodl	59% of Americans are correct, but damn! Have they rea...	2022-08-22 18:37:49	{'topic': 'forgiveness', 'sentiment': 2, 'technologies': []}
0x457	Wrong or not, the point is that many websites that decla...	2022-08-22 18:37:33	{'topic': 'web development', 'sentiment': 4, 'technologie...
nopehnnope	Some problems with that:<p>* The US had a single nati...	2022-08-22 18:37:37	{'topic': 'politics', 'sentiment': 2, 'technologies': []}
jahewson	I dunno. I can imagine any of those points being the sub...	2022-08-22 18:37:27	{'topic': 'lawsuit', 'sentiment': 3, 'technologies': []}
idlehand	It took billions of years to get to that point, too. Comple...	2022-08-15 19:36:48	{'topic': 'evolution', 'sentiment': 4, 'technologies': []}
NeverFade	So we know there is racial discrimination against Asians...	2022-10-09 14:57:21	{'topic': 'discrimination', 'sentiment': 2, 'technologies': []}
Comevius	Tegmark and Musk are both dumb people posing as int...	2022-10-09 14:57:21	{'topic': 'politics', 'sentiment': 2, 'technologies': []}
dragontamer	<a href="https://twitter.com/elonmu...	2022-04-25 19:00:15	{'topic': 'COVID19', 'sentiment': 2, 'technologies': []}
pasquinelli	you'll get older and things will happen to you. at s...	2022-10-09 14:57:01	{'topic': 'life', 'sentiment': 2, 'technologies': []}
kop316	heh, right now I have had an issue where in Stripe chec...	2022-10-09 14:57:16	{'topic': 'payment', 'sentiment': 2, 'technologies': ['"Stri...
cdiamand	This assumes that the human mind can continue to exp...	2022-10-09 14:57:17	{'topic': 'cognition', 'sentiment': 3, 'technologies': []}
O_______O	From Glassdoor for related employer:<p><a href="https...	2022-10-09 14:56:46	{'topic': 'workplace', 'sentiment': 2, 'technologies': ['"W...
rchaud	This is a US lawsuit, filed by a non-Indian Infosys emplo...	2022-10-09 14:56:49	{'topic': 'lawsuit', 'sentiment': 3, 'technologies': []}

Filter

100 Rows

{} my_output

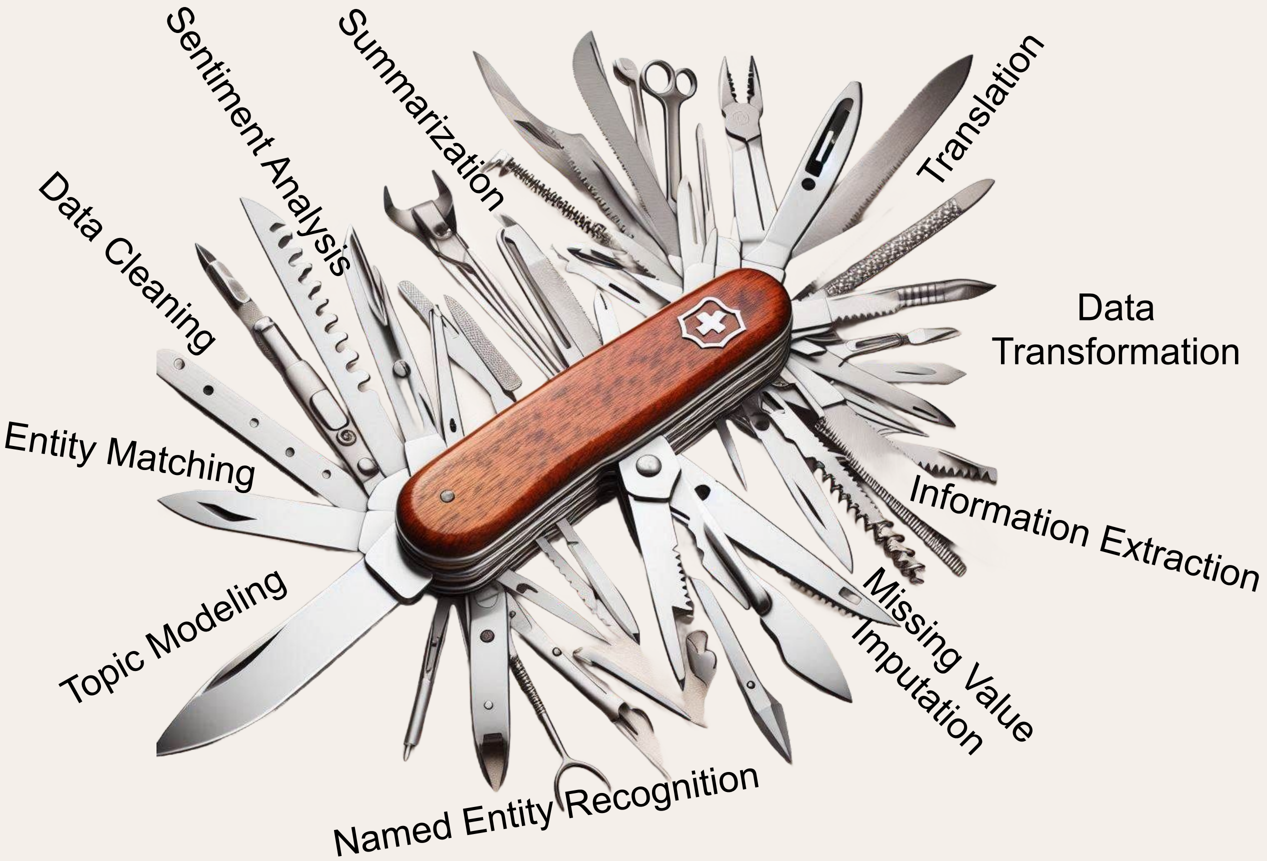
{} 3 keys
'topic' "web development"
'sentiment' 4
'technologies' [] 6 items
0 ""XHTML""
1 ""JavaScript""
2 ""CSS""
3 ""Rust""
4 ""Yew""
5 ""JSX""

Structured Data Extraction

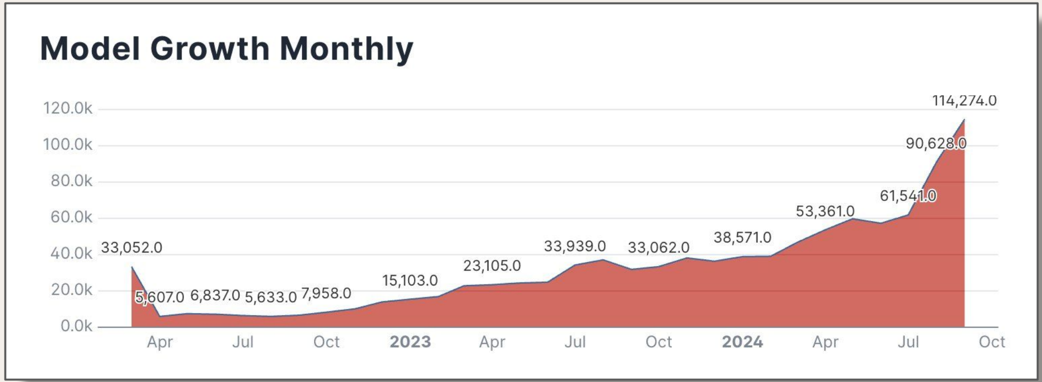
```sql
SELECT by, text, timestamp, my_output.* FROM my_struct_hn_table
```

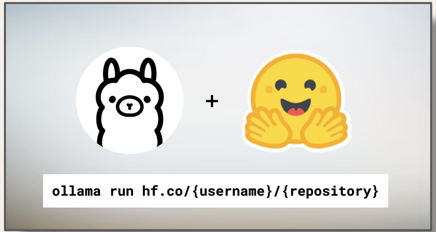
by	text	timestamp	topic	sentiment	technologies
yrgulation	> In a word, gardening. It's very fulfilling.<p>M...	2022-08-22 18:37:49	gardening	5	['"automation"', '"water pumps"', '"grou
paulmd	I actually don't know what you mean by that, but, ...	2022-08-22 18:37:46	insurance	3	[]
taylodl	59% of Americans are correct, but damn! Have they rea...	2022-08-22 18:37:49	finance	2	[]
0x457	Wrong or not, the point is that many websites that decla...	2022-08-22 18:37:33	web development	4	['"XHTML"', '"JavaScript"', '"CSS"', '"Ru
nopehnnope	Some problems with that:<p>* The US had a single nati...	2022-08-22 18:37:37	politics	2	[]
jahewson	I dunno. I can imagine any of those points being the sub...	2022-08-22 18:37:27	lawsuit	3	[]
idlehand	It took billions of years to get to that point, too. Comple...	2022-08-15 19:36:48	evolution	4	[]
NeverFade	So we know there is racial discrimination against Asians...	2022-10-09 14:57:21	discrimination	2	[]
Comevius	Tegmark and Musk are both dumb people posing as int...	2022-10-09 14:57:21	politics	2	[]
dragontamer	<a href="https://twitter.com/elonmu...	2022-04-25 19:00:15	COVID19	2	[]
pasquinelli	you'll get older and things will happen to you. at s...	2022-10-09 14:57:01	aging	2	[]

Pre-Trained Language Models are the swiss army knife of NLP



Open Models & Local Inference are on the rise.

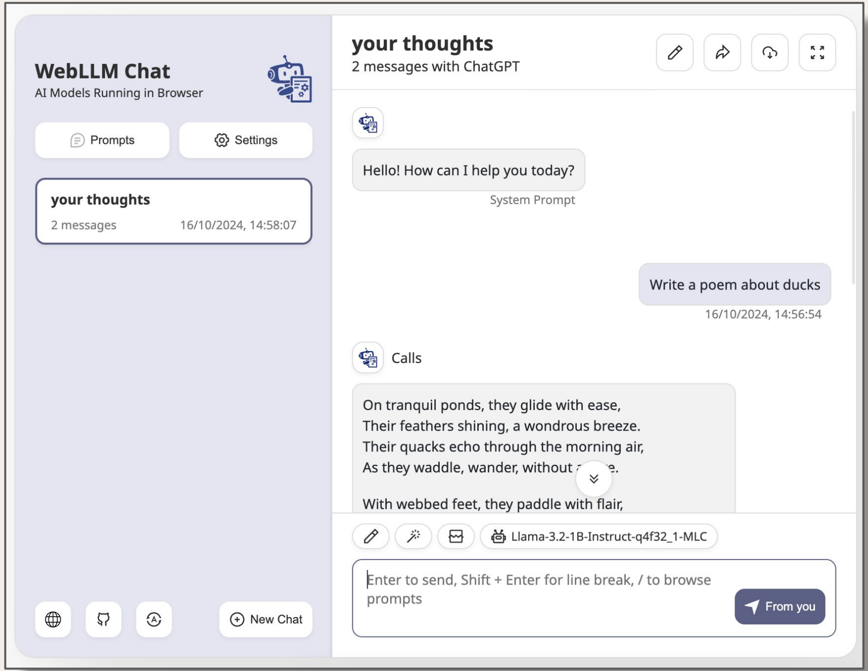





```
pip install ollama
```

```
npm install ollama
```

Open Models & Local Inference are on the rise.

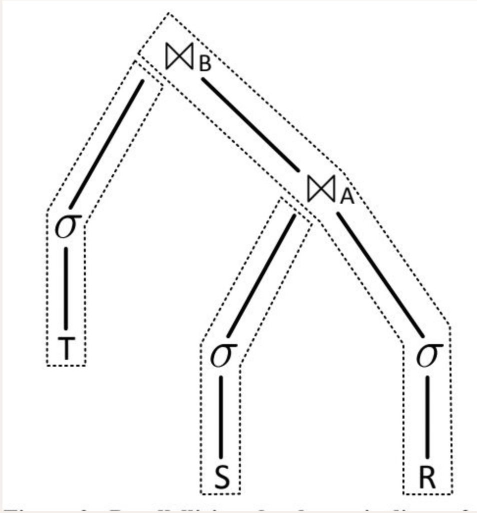


WebGPU

https://webllm.mlc.ai/

Dual Execution

```
select *
from T,S,R
where T.id=S.id AND S.id=R.id
```

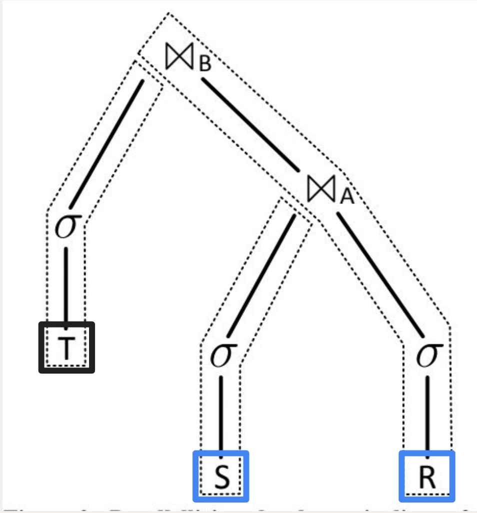


This example is taken from the [Morsel-Driven Parallelism](#) , which DuckDB is based on.

Dual Execution

```
select *
from T,S,R
where T.id=S.id AND S.id=R.id
```

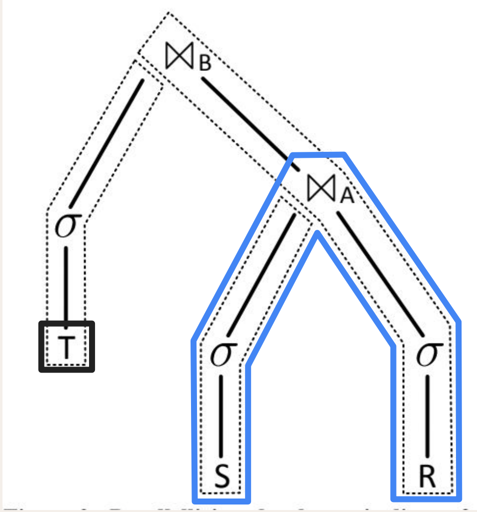
Local
Remote



This example is taken from the [Morsel-Driven Parallelism](#) , which DuckDB is based on.

Dual Execution

```
select *
from T,S,R
where T.id=S.id AND S.id=R.id
```



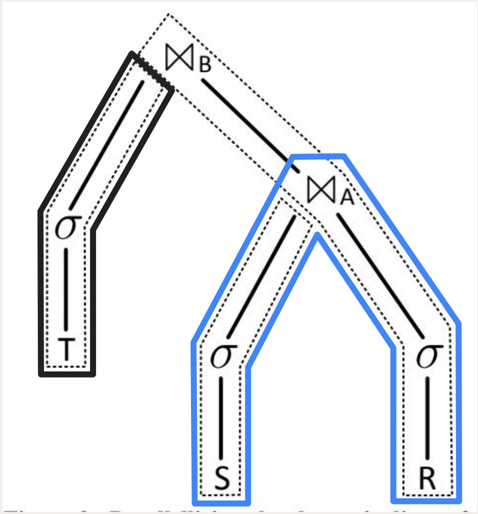
Local
Remote

This example is taken from the [Morsel-Driven Parallelism] , which DuckDB is based on.

Dual Execution

```
select *
from T,S,R
where T.id=S.id AND S.id=R.id
```

Local
Remote

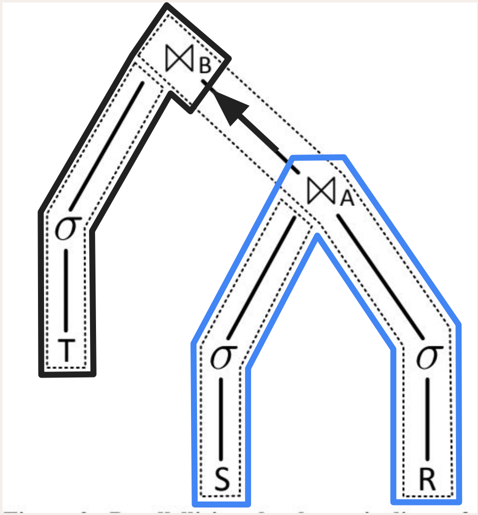


This example is taken from the [Morsel-Driven Parallelism](), which DuckDB is based on.

Dual Execution

$S \bowtie R$ is small, T is **large**

```
select *
from T,S,R
where T.id=S.id AND S.id=R.id
```

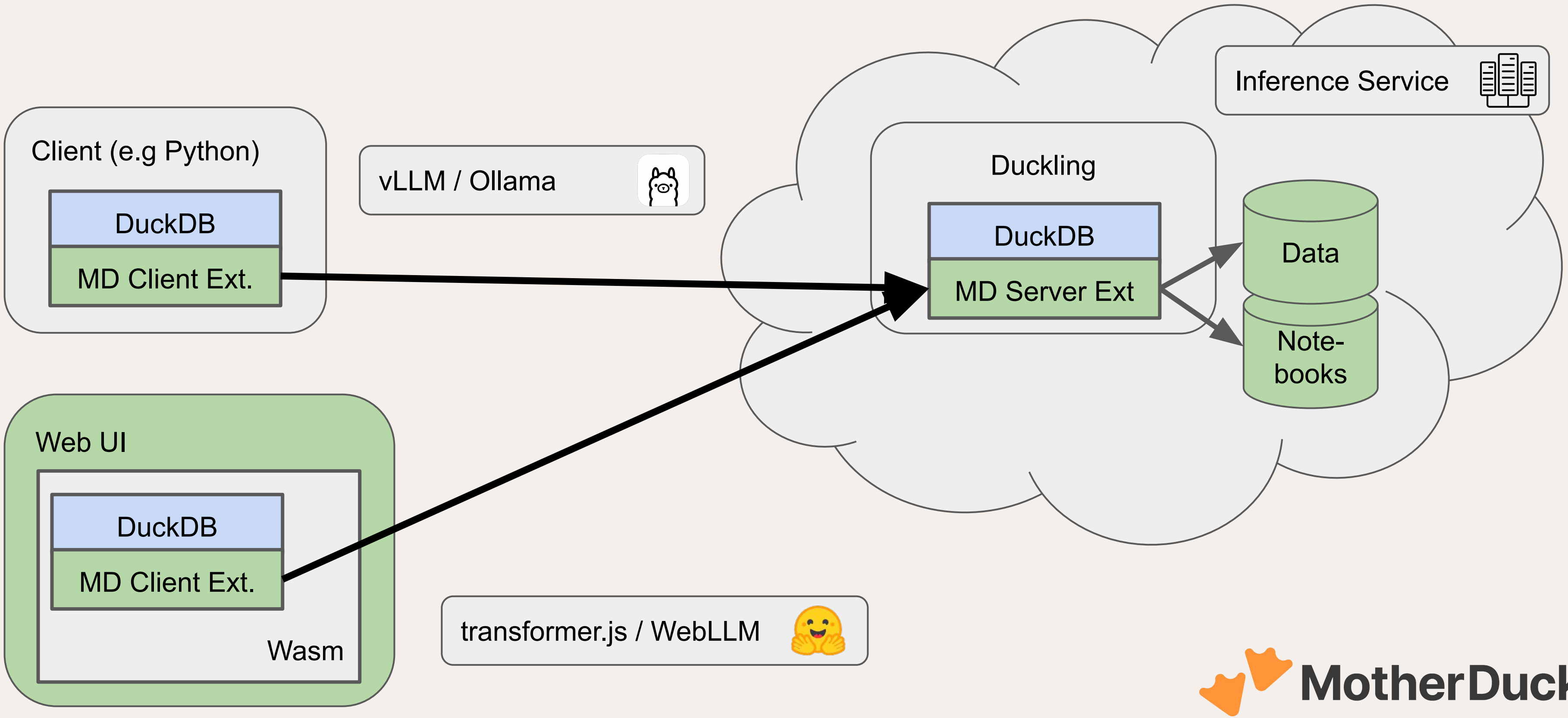


Local

Remote

This example is taken from the Morsel-Driven Parallelism , which DuckDB is based on.

Dual Execution for Prompt & Embedding Inference



MotherDuck

MotherDuck

NL to SQL

Text Processing with LLMs in SQL

Embeddings & Vector Search

Classical Machine Learning

Compute Embeddings in SQL.

```sql
SELECT embedding(my_text) FROM my_table;
```

Similarity Search

```sql
SELECT title, overview,
       array_cosine_similarity(embedding('artificial intelligence'), title_embeddings) as similarity
FROM kaggle.movies
ORDER BY similarity DESC
LIMIT 3
```

title	overview	similarity
A.I. Artificial Intelligence	A robotic boy, the first programmed to love, David is ad...	0.80
I, Robot	In 2035, where robots are common-place and abide by ...	0.46
Almost Human	Mark Fisher disappeared from his home in a brilliant fla...	0.45

Prompt + Embedding = Retrieval Augmented Generation (RAG)


```
CREATE OR REPLACE TEMP MACRO ask_question(question_text) AS TABLE (
  SELECT LIST('Title: ' || title || '; Description: ' || overview) as documents,
    prompt(
      'User asks the following question:\n' || question_text || '\n\n' ||
      'Here are the most revelant movies:\n' ||
      STRING_AGG('Title: ' || title || '; Description: ' || overview, '\n') || '\n' ||
      'Please write the answer based on them.',
      model := 'gpt-4o'
    ) AS response,
  FROM (
    SELECT title, overview
    FROM kaggle.movies
    ORDER BY array_cosine_similarity(overview_embeddings, embedding(question_text))
DESC
    LIMIT 3
  )
);

FROM ask_question('Can you recommend any movies involving fast ducks?')
```

response

If you're looking for movies involving fast ducks, you might enjoy "Bugs Bunny's 3rd Movie: 1001 Rabbit Tales" and "The Bugs Bunny/Road Runner Movie." Both films feature Daffy Duck, a character known for his quick and energetic antics. While these movies are compilations of classic Warner Bros. cartoons, they include several shorts where Daffy's fast-paced and humorous personality shines through.

Check out the docs at:
https://motherduck.com/docs/

Or try it out on:
app.motherduck.com

Free Trial (30 days):

- ~ 40k prompts / day
- ~ 1M embeddings / day

Embedding Function

Text Embeddings can be generated using the `embedding` scalar function.

The embedding fun
`large` with 1024 o

Consumption is me
`small` or 4k embe

Syntax

```
SELECT embedd
```

Model Paran

By default, the func
parameter.

Supported models:

family	
OpenAI	`text-`
OpenAI	`text-`

Prompt Function

Large Language Models (LLMs) can be prompted, using the `prompt` function. Outputs can be either text or structured data.

The prompt function uses OpenAI's `gpt-4o-mini` or `gpt-4o`. Both models support constant and single-row inputs. Multi-row (batch) processing is only permitted with `gpt-4o-mini`.

Consumption is measured in compute units (CU). One CU hour equates to approx. 1k prompt responses with `gpt-4o-mini` or 50 prompt responses with `gpt-4o`, assuming an input size of 1000 characters and response size of 250 characters.

Syntax

```sql
SELECT prompt('Write a poem about ducks'); -- returns a single cell table with the response
```

Optional parameters

Parameter	Description
`model`	Model type, either `'gpt-4o-mini'` (default), or `'gpt-4o-2024-08-06'` (alias: `'gpt-4o'`)
`temperature`	Model temperature value between `0` and `1`, default: `0.1`
`struct`	Output schema as struct, e.g. `{summary: 'VARCHAR', persons: 'VARCHAR[]'}`. Will result in `STRUCT` output.
`struct_descr`	Descriptions for struct fields that will be added to the model's context, e.g. `{summary: 'a 1 sentence summary of the text', persons: 'an array of all persons mentioned in the text'}`
`json_schema`	A json schema that adheres to this guide. Provides more flexibility than the struct/struct_descr parameters. Will result in `JSON` output.

Vector Search: Naiive Search, HNSW, IVFFlat

Naiive Search (https://duckdb.org/docs/sql/functions/array.html#array_cosine_similarityarray1-array2)

- ➕ No Index Maintenance
- ➕ 100% Retrieval Accuracy
- ➕ < 1s lookup times with up to 2M rows in DuckDB (Mac M2 Pro)
- ➖ Lookup times scale linearly with dataset size

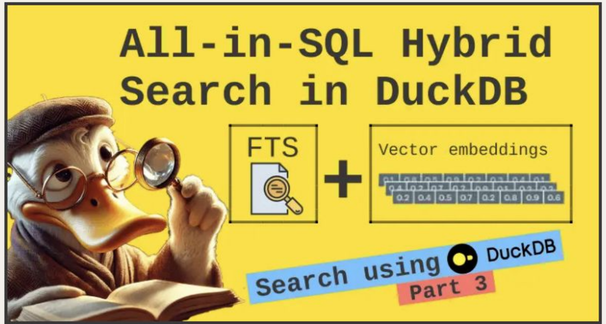
HNSW: (https://duckdb.org/docs/extensions/vss.html)

- ➕ High recall & QPS even on large datasets (>10M entries)
- ➕ Index is relatively robust to updates
- ➖ Index building takes time (~300s for a 2M row index)
- ➖ Large memory footprint (roughly 0.75x of the embedding size)

IVFFlat (https://community-extensions.duckdb.org/extensions/faiss.html)

- ➕ Low memory footprint (only save one cluster-id per row)
- ➕ Index creation is fast
- ➖ incremental updates require re-computation of centroids to maintain recall → not ideal for frequent updates
- ➖ lower QPS than HNSW (~ factor 10x)
- ➖ lower recall than HNSW for large datasets
- ➖ faiss extension still in early stages

Hybrid Search



2024/06/20 - Adithya Krishnan

ALL-IN-SQL HYBRID SEARCH IN DUCKDB: INTEGRATING FULL TEXT AND EMBEDDING METHODS

Explore search methods with DuckDB using Full-Text-Search and embeddings

https://motherduck.com/blog/search-using-duckdb-part-3/

MotherDuck

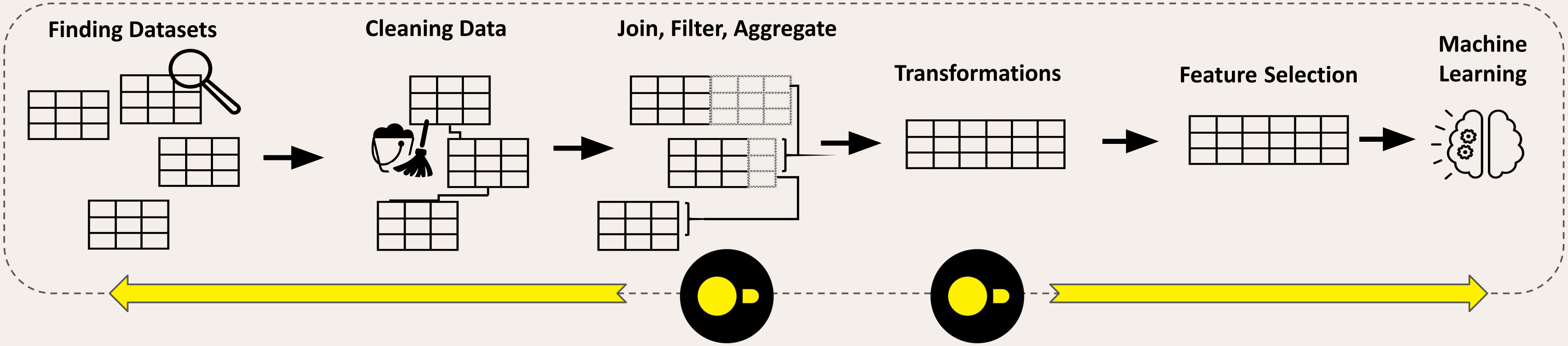
NL to SQL

Text Processing with LLMs in SQL

Embeddings & Vector Search

Classical Machine Learning

Machine Learning in A Data Warehouse



Linear Regression

`regr_intercept(y, x)`

Description	The intercept of the univariate linear regression line, where x is the independent variable and y is the dependent variable.
Formula	`regr_avgy(y, x) - regr_slope(y, x) * regr_avgx(y, x)`
Alias(es)	-

`regr_slope(y, x)`

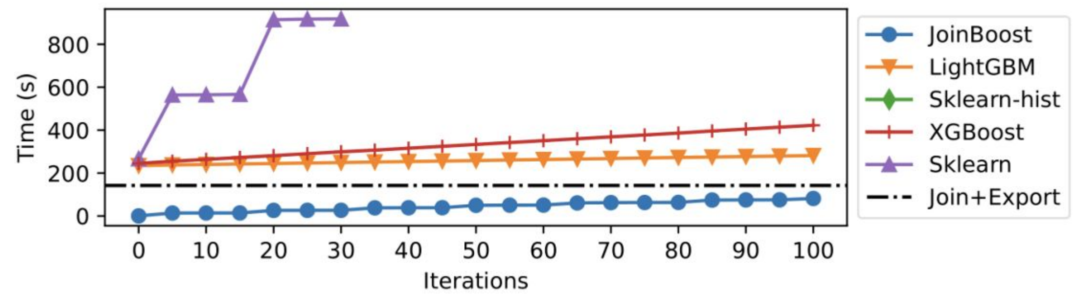
Description	Returns the slope of the linear regression line, where x is the independent variable and y is the dependent variable.
Formula	`regr_sxy(y, x) / regr_sxx(y, x)`
Alias(es)	-

https://duckdb.org/docs/sql/functions/aggregates.html#regr_slopey-x

K-Means Clustering in DuckDB

```
-- Create table with points
CREATE TABLE tbl1 AS SELECT [a,b]::FLOAT[2] as val1 FROM
read_csv('https://github.com/gagolews/clustering-data-v0/raw/master/s1.data.gz',  delim=' ',
column_names=['a', 'b']) ORDER BY random();

-- Run in-database clustering
SELECT kmeans(val1, 15) FROM tbl1;

--> Returns
MAP(INTEGER, FLOAT[2])
{0=[823831.25, 729752.75], 1=[617830.4, 398966.125], 2=[852081.44, 157823.08], 3=[423174.72,
167577.94], 4=[244692.33, 847647.75], 5=[337903.5, 562290.75], 6=[671153.8, 862441.9], 7=[417797.84,
786973.94], 8=[419718.5, 424007.7], 9=[801502.3, 320845.4], 10=[177010.39, 332601.53], 11=[140104.45,
556949.25], 12=[606544.94, 574494.75], 13=[859121.2, 545573.8], 14=[385950.84, 392737.8]}
```

Decision Trees in DuckDB



(a) Random forest training time. JoinBoost is ∼3× faster than LightGBM.

https://github.com/JoinBoost/JoinBoost

Data Cleaning and Wrangling

Can Foundation Models Wrangle Your Data?

Avanika Narayan, Ines Chami†, Laurel Orr, Simran Arora, Christopher Ré
Stanford University and †Numbers Station
{avanika,lorr1,chrismre,simarora}@cs.stanford.edu,ines.chami@numbersstation.ai

ABSTRACT

Foundation Models (FMs) are models trained on large corpora of data that, at very large scale, can generalize to new tasks without any task-specific finetuning. As these models continue to grow in size, innovations continue to push the boundaries of what these models can do on language and image tasks. This paper aims to understand an underexplored area of FMs: classical data tasks like cleaning and integration. As a proof-of-concept, we cast five data cleaning and integration tasks as prompting tasks and evaluate the performance of FMs on these tasks. We find that large FMs generalize and achieve SoTA performance on data cleaning and integration tasks, even though they are not trained for these data tasks. We identify specific research challenges and opportunities that these models present, including challenges with private and domain specific data, and opportunities to make data management systems more accessible to non-experts. We make our code and experiments publicly available at: https://github.com/HazyResearch/fm_data_tasks.

1 INTRODUCTION

Foundation Models (FMs) [19] are models trained on broad data that can be adapted to a wide range of downstream tasks. These models have achieved substantial gains across many semantically challenging tasks such as question answering [20], knowledge base construction [80], and information retrieval [39]. As they have scaled to hundreds of billions of parameters (e.g. GPT-3 [20], PaLM [22]), large FMs have demonstrated surprising emergent behaviors and good zero-shot generalization to new tasks (i.e. no task-specific finetuning) on domains vastly different from the data they were pretrained on [22]. These large FMs are often autoregressive language models (e.g. GPT-3 and PaLM) that are trained to predict the next word in large text corpora and can be adapted to new tasks given a simple natural language description of the task (see Figure 1). These breakthrough capabilities have led to a race for building bigger and better models, and innovations continue to push the boundaries of what large FMs can do on a variety of hard *language tasks*.

A natural question that arises is whether these advances can benefit hard classical *data tasks* (e.g. data cleaning and integration). While it is clear that FMs benefit text-intensive tasks, it is not clear whether these models can be applied to data tasks over structured data. The symbols commonly found in structured data (e.g. dates,

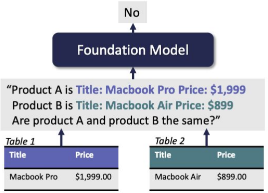


Figure 1: A large FM can address an entity matching task using prompting. Rows are serialized into text and passed to the FM with the question "Are products A and B the same?". The FM then generates a string "Yes" or "No" as the answer.

pretrained language models (PLMs) like BERT [32]—to semantically-complex data tasks. However, these approaches still require a significant amount of engineering effort as they rely on:

- **Task-specific architectures**: Data cleaning and integration encapsulate *many different tasks* such as entity matching [78], schema matching [93], and error detection [41]. Existing approaches, whether they are rule-, ML- or DL-based, vary greatly from one task to the other, often with complex, task-specific architectures. For instance, adapting BERT to data tasks requires architectural changes and finetuning the entire model for each task. This leads to siloed and hard-to-maintain systems.
- **Hard-coded knowledge**: Data tasks often rely on *domain knowledge* (e.g. understanding the relationship between a city and its zip code for data cleaning constraints) and commonsense reasoning. These are usually hard-coded with human-engineered rules or external knowledge bases [24, 85]. Consequently, systems can be brittle and fail to generalize to a diverse set of domains.
- **Labeled data**: ML- and DL-based solutions require copious amounts of hand-labeled data [9]. For instance, PLMs that have achieved state-of-the-art (SoTA) results on data tasks (e.g. Ditto [38]) require a significant amount of task-specific labeled data and fine-

Table 1: Entity matching results measured by F1 score where *k* is the number of task demonstrations.

Dataset	Magellan	Ditto	GPT3-175B (k=0)	GPT3-175B (k=10)
Fodors-Zagats	100	100	87.2	**100**
Beer	78.8	94.37	78.6	**100**
iTunes-Amazon	91.2	97.06	65.9	**98.2**
Walmart-Amazon	71.9	86.76	60.6	**87.0**
DBLP-ACM	98.4	**98.99**	93.5	96.6
DBLP-Google	92.3	**95.60**	64.6	83.8
Amazon-Google	49.1	**75.58**	54.3	63.5

Task	Imputation		Error Detection	
Dataset	Restaurant	Buy	Hospital	Adult
HoloClean	33.1	16.2	51.4	54.5
IMP	77.2	96.5	-	-
HoloDetect	-	-	94.4	99.1
GPT3-175B (k=0)	70.9	84.6	6.9	0.0
GPT3-6.7B (k=10)	80.2	86.2	2.1	99.1
GPT3-175B (k=10)	**88.4**	**98.5**	**97.8**	**99.1**

Task	Data Transformation		Schema Matching
Dataset	StackOverflow	Bing-QueryLogs	Synthea
Previous SoTA	63.0	32.0	38.5
GPT3-175B (k=0)	32.7	24.0	0.5
GPT3-175B (k=3)	**65.3**	**54.0**	**45.2**

Data Cleaning and Wrangling

Towards Parameter-Efficient Automation of Data Wrangling Tasks with Prefix-Tuning

David Vos
University of Amsterdam
d.j.a.vos@uva.nl

Till Döhmen
University of Amsterdam
t.r.dohmen@uva.nl

Sebastian Schelter
University of Amsterdam
s.schelter@uva.nl

Abstract

Data wrangling tasks for data integration and cleaning arise in virtually every data-driven application scenario nowadays. Recent research indicated the astounding potential of Large Language Models (LLMs) for such tasks. However, the automation of data wrangling with LLMs poses additional challenges, as hand-tuning task- and data-specific prompts for LLMs requires high expertise and manual effort. On the other hand, finetuning a whole LLM is more amenable to automation, but incurs high storage costs, as a copy of the LLM has to be maintained. In this work, we explore the potential of a lightweight alternative to finetuning an LLM, which automatically learns a continuous prompt. This approach called prefix-tuning does not require updating the original LLM parameters, and can therefore re-use a single LLM instance across tasks. At the same time, it is amenable to automation, as continuous prompts can be automatically learned with standard techniques. We evaluate prefix-tuning on common data wrangling tasks for tabular data such as entity matching, error detection, and data imputation, with promising results. We find that in five out of ten cases, prefix-tuning is within 2.3% of the performance of finetuning, even though it leverages only 0.39% of the parameter updates required for finetuning the full model. These results highlight the potential of prefix-tuning as a parameter-efficient alternative to finetuning for data integration and data cleaning with LLMs.

1 Introduction

Data wrangling tasks such as finding duplicates during data integration, detecting errors in tables or

Table 3: Prefix-tuning drastically outperforms (trainingless) zero-shot prompting across all tasks.

Task	Dataset	Metric	Prefix-tuning T5 (220M params)	Zero-shot prompting GPT-3 (175B params)
Entity matching	DBLP-Google	F1-score	0.9517	0.646
Entity matching	DBLP-ACM	F1-score	0.981	0.935
Entity matching	iTunes-Amazon	F1-score	0.9286	0.659
Entity matching	Fodors-Zagats	F1-score	0.9767	0.872
Entity matching	Beer	F1-score	0.8571	0.786
Entity matching	Walmart-Amazon	F1-score	0.7961	0.606
Entity matching	Amazon-Google	F1-score	0.6642	0.543
Imputation	Buy	Accuracy	0.9231	0.846
Imputation	Restaurant	Accuracy	0.8488	0.709
Error detection	Hospital	F1-score	0.9766	0.069

Data Cleaning and Wrangling

Towards Parameter Efficient Automation

Towards Efficient Data Wrangling with LLMs using Code Generation

Xue Li
MotherDuck & University of Amsterdam
Amsterdam, Netherlands
x.li3@uva.nl

Till Döhmen
MotherDuck
Amsterdam, Netherlands
till@motherduck.com

ABSTRACT

While LLM-based data wrangling approaches that process each row of data have shown promising benchmark results, computational costs still limit their suitability for real-world use cases on large datasets. We revisit code generation using LLMs for various data wrangling tasks, which show promising results particularly for data transformation tasks (up to 37.2 points improvement on F1 score) at much lower computational costs. We furthermore identify shortcomings of code generation methods especially for semantically challenging tasks, and consequently propose an approach that combines program generation with a routing mechanism using LLMs.

ACM Reference Format:
Xue Li and Till Döhmen. 2024. Towards Efficient Data Wrangling with LLMs using Code Generation. In *Workshop on Data Management for End-to-End Machine Learning (DEEM 24), June 9, 2024, Santiago, AA, Chile.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3650203.3663334

1 INTRODUCTION

Data wrangling tasks such as data cleaning, data integration, and data transformations are part of almost every data analytics workflow and ETL pipeline when working with real-world data. As wrangling tasks can be tedious and time-consuming [7], methods that automate or assist users with such tasks are a valuable addition to BI and data warehousing systems like MotherDuck. When talking to MotherDuck users, we observed that aside from ease-of-use, quality and computational efficiency, interpretable and deterministic solutions are crucial for the adoption and trust in automated wrangling solutions. For ad-hoc analytics, users want to write simple prompts that describe the desired wrangling task, and iterate over ideas quickly, without the need to extensively label data. At

makes them not well suitable for ad-hoc analytics scenarios. While large language models (LLMs) [9] perform decently in zero-shot or few-shot settings, they incur high latency and are expensive to apply to each row, which makes them unsuitable for million-row scale datasets. LLMPR methods are furthermore not well suited for structured-to-structured data transformation, such as unit conversion, as they struggle with calculations. Another direction is to use program synthesis or programming-by-example (PBE) methods [1, 4, 7] that derive a program (e.g. Pandas, or Excel-Macro) from a given set of input-and-output examples. Those methods have the desired property of being well interpretable and deterministic, and produce code that can be executed efficiently on millions of rows. However, traditionally, program synthesis and PBE methods were challenging to adapt to new tasks. Now, with LLMs, using PBE methods for data wrangling is becoming more feasible. However, even then, PBE methods struggle with semantically challenging tasks (see BingQL-semantics eval in Section 4.2) if they were not specifically implemented to handle them. Furthermore, giving natural language instruction rather than input-output pairs feels more natural for certain tasks, e.g. "detect faulty entries in this column or "convert Roman numerals to Arabic numbers".

We revisited LLMs as prompt-based code generators for data wrangling tasks, and evaluated our method on existing data wrangling benchmarks. Our experiments show that LLM-generated data wrangling code outperforms existing LLMPR and traditional PBE methods, in particular on data transformation tasks, while the performance on other tasks such as entity matching and error detection varies depending on the task and dataset. We attribute this to some tasks requiring a semantic understanding of the input, where code-based approaches underperform compared to LLMPR methods.

We conclude that neither of both directions alone have the potential to lead to high-quality and cost-efficient automated wrangling

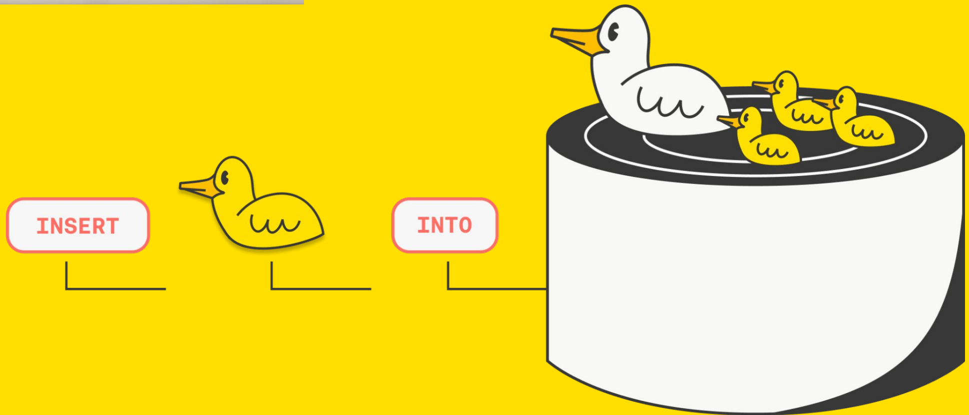
Dataset	PBE [4]	LLMPR [9]	Code Generation (Ours)
BingQL-semantics	32.0	54.0	**91.6**
BingQL-Unit	**96.0**	N/A	95.0
Stack-overflow	63.0	65.3	**87.4**
FF-GR-Trifacta	**91.0**	N/A	83.7
Head cases	**82.0**	N/A	74.6
Average	72.8	N/A	**86.46**

Table 1: F1 score on Data Transformation task, $k = 3$.

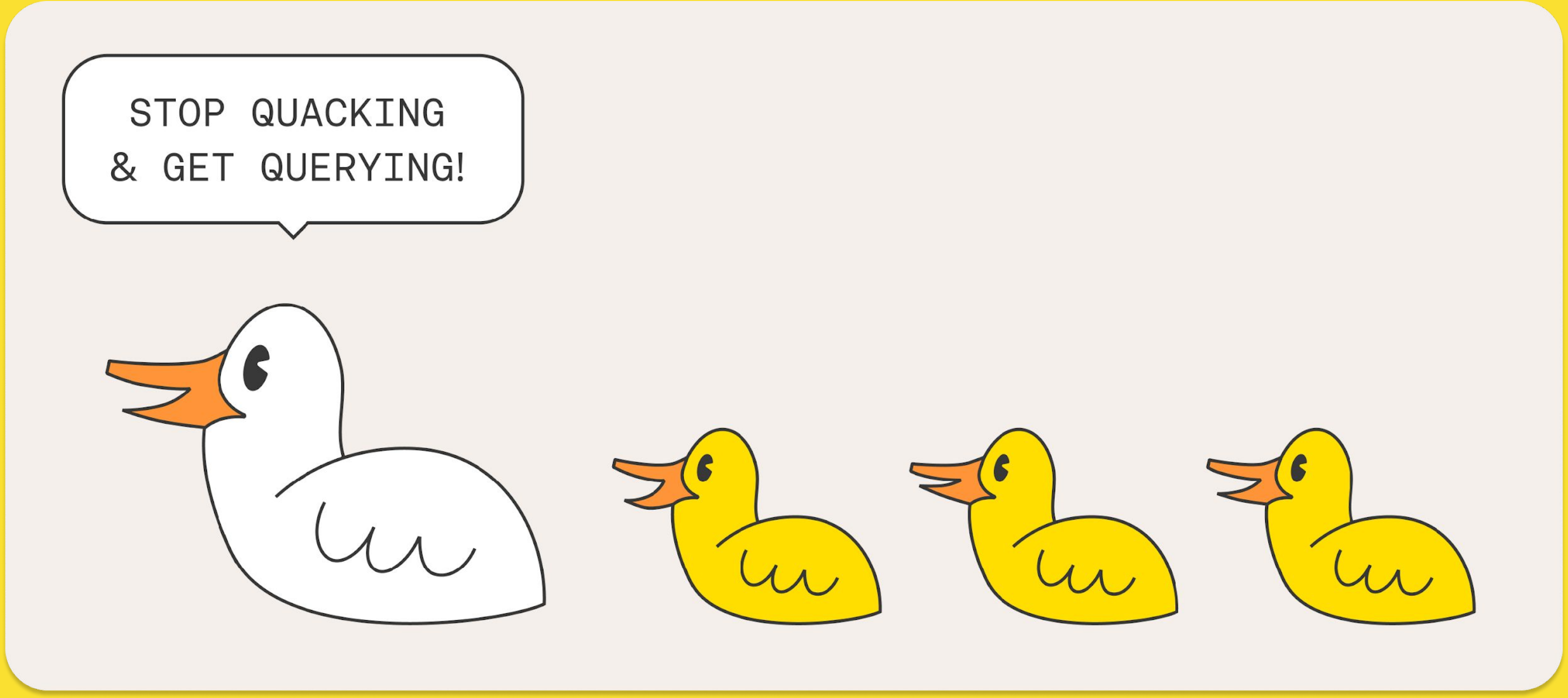
Task	Dataset	LLMPR[9]	Code Generation (Ours)
EM	Fodors-Zagats	100	95.5
EM	Beer	100	75.0
EM	DBLP-ACM	96.6	19.7
EM	DBLP-GoogleScholar	83.8	69.7
EM	Amazon-Google	63.5	42.1
EM	iTunes-Amazon	98.2	70.0
EM	Walmart-Amazon	87.0	25.5
DI	Buy	98.5	84.6
DI	Restaurant	88.4	50
ED	Hospital	97.8	23.5
ED	Adult	99.1	100*

Table 2: Accuracy on Data Imputation task and F1 on Entity Matching and Error Detection, $k = 10$.
(* score is only evaluated on the "income" column.)

MSc. Thesis @ MotherDuck



Thank you. Questions?



app.motherduck.com