



DuckDB - Meetup #3 - Amsterdam



Query.Farm

The place where DuckDB
Community Extensions Grow

Written by: L.Mangani, R. Conover

Presented by: L. Mangani on 2025-09-16

Sponsored by: Query.Farm LLC, QXIP BV

Contact: hello@query.farm

DuckDB Community Extensions



Where did we hatch from 🐣



query.farm
Rusty Conover

quackscience
Lorenzo Mangani



What will you learn today? 🐣

- ★ Query Farm has created **20+ extensions** with **over 2,000,000+** downloads and growing
- ★ You'll see a few very quick examples from a few of our most popular/useful extensions
- ★ Query Farm is interested in helping you adopt understand and extend DuckDB 🎓👁️
- ★ You are invited to join in and **grow** with us 🌾





Query.Farm

What we do:

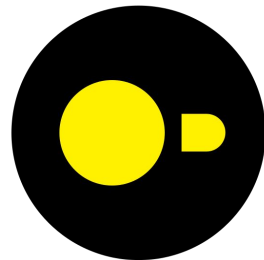
- Boost DuckDB with custom extensions
- Build developer and user's tools
- Embed DuckDB into modern data systems

Who we are:

- Database engineers & systems hackers
- DuckDB contributors
- Makers of elegant & lightweight solutions

Joined the Flock:

























- Quackscience joined us in mid-2025 🎉
- Fully Independent from DuckDB Labs/Foundation



What have we made possible?






If **DuckDB** doesn't do it yet, *we can try make it happen.*

So far we've created **20+** extensions that have been downloaded over **2,000,000+** times.

 airport	 fuzzycomplete	 pyroscope
 bitfilters	 hashfuncs	 rapidfuzz
 crypto	 httpserver	 redis
 chsql	 httpclient	 shellfs
 cronjob	 lindel	 stochastic
 datasketches	 marisa	 textplot
 evalexpr_rhai	 quickjs	 tributary
 tsid	 radio	 webmacro

We Can Build With/For You

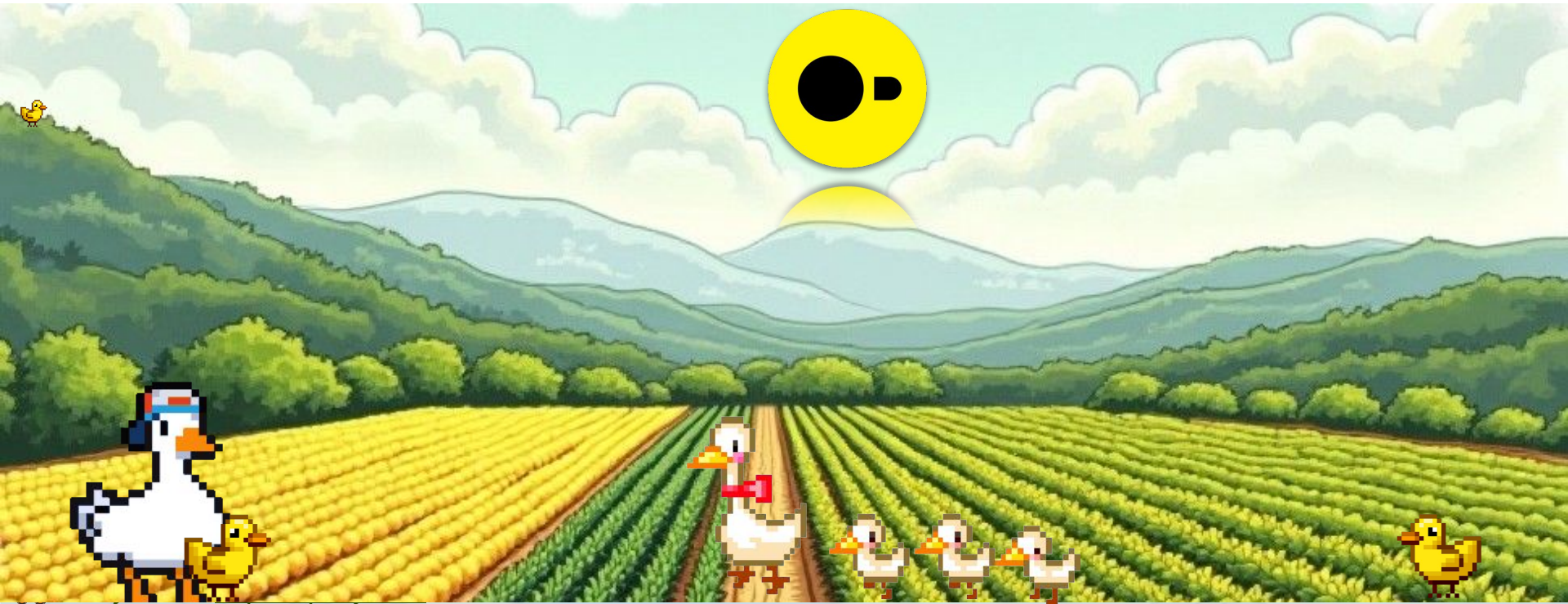
Who Do We Help:

-  Product teams planting analytics into apps
-  Data engineers modernizing pipelines
-  Platform teams updating legacy data infra
-  Software engineers designing integrations
-  Anyone building a DuckDB-powered architecture






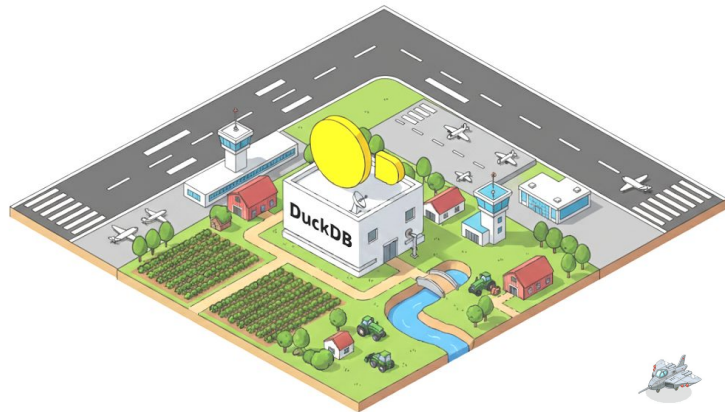
Query.Farm Extension Showcase



Airport Extension

Query, Modify and Serve Data with Apache Arrow Flight

1. Access data anywhere
APIs, services, non-tabular or unsupported formats
2. Enable **Data-as-a-Service**
3. Run remote scalar (UDF) and table functions
4. Serve data with fine-grained access control
Row & column level filters
5. Let  DuckDB fly to remote servers



```
pip install query-farm-airport-test-server
```

```
LOAD airport;  
  
-- Attach a database named 'hello'  
ATTACH 'hello' (TYPE AIRPORT, location 'grpc+tls://hello-airport.query.farm');  
  
-- Query the remote data  
SELECT * FROM hello.static.greetings WHERE language = 'Dutch';
```


Radio Extension

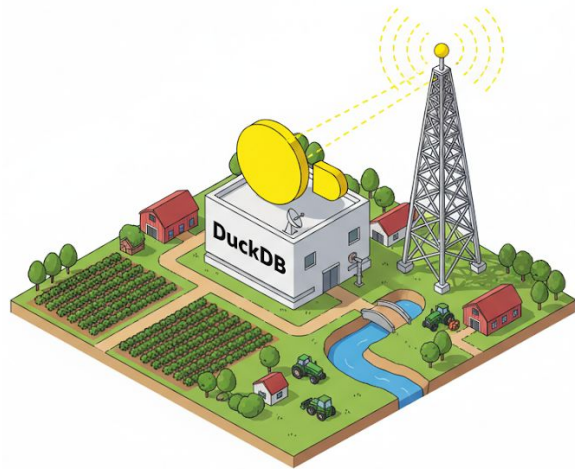
Interact with real-time event systems, Websockets, Redis Pub/Sub

1. Supports two-way communication

Receive events: buffered and queryable with SQL

Send events: buffered with delivery tracking

2. Bridges SQL and event driven architectures
3. Blocking and non-blocking modes available

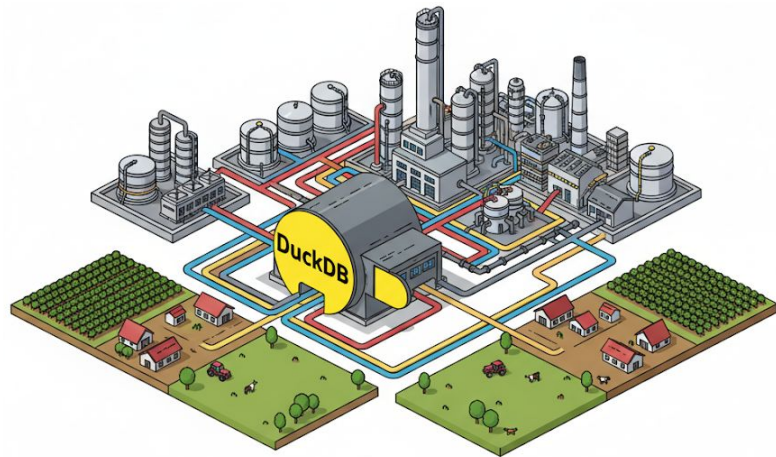


```
LOAD radio;  
  
-- Open a connection to a Bluesky websocket server  
CALL radio_subscribe('wss://jetstream2.us-east.bsky.network/subscribe');  
  
-- Block until a message is received for 10 seconds.  
CALL radio_listen(true, interval '10 seconds');  
  
-- Show messages  
SELECT * FROM radio_received_messages();
```

ShellFS Extension

Use pipes and treat shell commands like files in DuckDB

1. **Read** from stdout of any shell command
2. **Write** to the stdin of shell commands
(compress, upload, custom code)
3. **Pipeline** data through DuckDB for ad hoc ETL
4. **No** intermediate files, reducing disk I/O



```
LOAD shellfs;

-- Read CSV data from a external program
SELECT * FROM read_csv('python data-generator.py |');

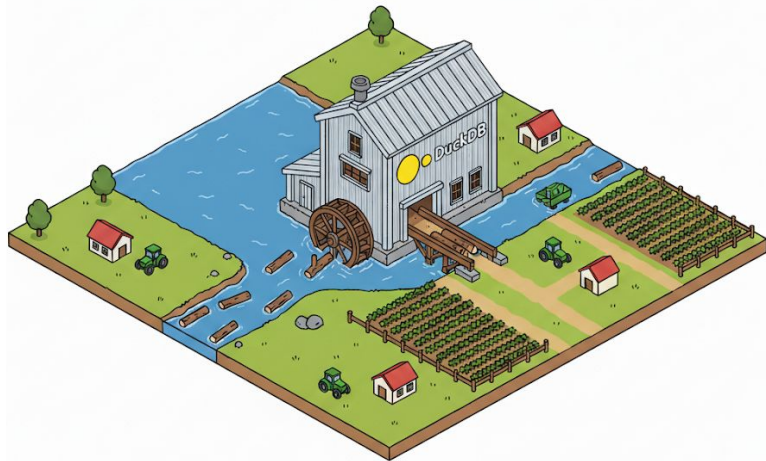
-- Read all data from a pipe
SELECT * FROM read_text('ps axuww |');

-- Write data to a Python program via a pipe
COPY (SELECT * FROM generate_series(1000)) TO '| python consumer.py' (FORMAT 'CSV');
```

Tributary Extension

Integration with Apache Kafka

1. **Read Support:** Directly stream records from topics into DuckDB tables and queries.
2. **Future Development:**
 - a. **Write** - write data back to Kafka from DuckDB
 - b. **Registry Service** Integration
 - c. **Protobuf, Avro, JSON** Record Formats
 - d. **CDC stream integration**

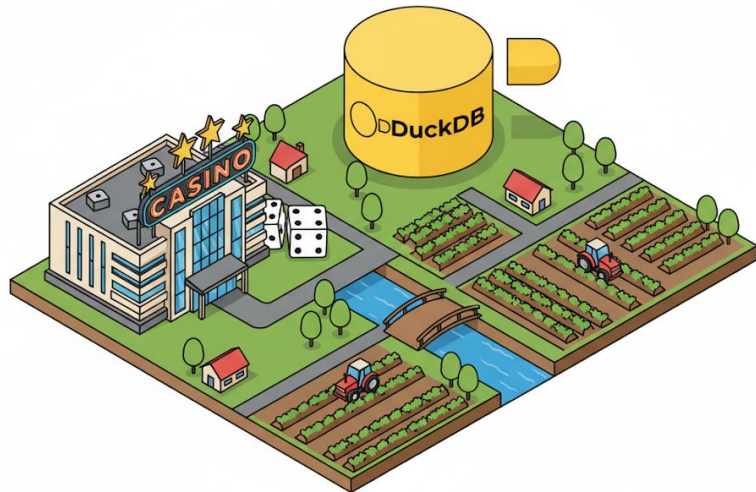


```
LOAD tributary;  
-- Read all messages on a topic  
SELECT * FROM tributary_scan_topic('test-topic', "bootstrap.servers" := 'kafka-cluster.query.farm:9092')
```

Stochastic Extension

Statistical distributions for DuckDB

1. Adds probability, statistical and random sampling functions.
2. Enables **PDF/PMF, CDF, quantiles and random sampling**
3. **Provides distribution properties** like mean and variance.
4. **Distributions:** Normal, Uniform, Binomial, Poisson, Exponential and more...
5. Uses: **A/B tests, anomaly detection, statistical significance, confidence intervals, Monte Carlo Simulations, hypothesis testing**



```
LOAD stochastic;
-- Probability density function
SELECT * FROM dist_normal_pdf(1.5, 0, 1);
-- Sample from the normal distribution
SELECT dist_normal_sample(0, 1.0) * FROM range(10);
```


Redis Extension

Native Redis Client for DuckDB

1. Supports Strings, Hashes, Lists and Key operations.
2. Operations include **SCAN**, **MGET**, **HGET**, **HSET**, **DEL**, **EXISTS**, **HSCAN**.
3. Allows data in Redis to interact with DuckDB without custom code



```
LOAD redis;

-- Set a value
SELECT redis_set('user:1', 'John Doe', 'redis') as result;

-- Get a value
SELECT redis_get('user:1', 'redis') as user_name;

-- Set multiple values in a query
INSERT INTO users (id, name)
SELECT id, redis_set('user:' || id::VARCHAR, name, 'my_redis') FROM new_users;

-- Get hash field
SELECT redis_hget('user:1', 'email', 'redis') as email;
```

HTTP Server Extension

Embedded API Server w/ Query User Interface

The **HTTP Server** extension transforms **DuckDB** instances into fast, tiny multi-player **HTTP OLAP API** designed to execute queries and stream results.

The service supports several formats (*CSV,TSV,JSON*) and it loosely compatible with the ClickHouse HTTP API baseline. The **HTTP Server** extension supports Authentication (*Basic Auth or X-Token*) and embeds a User-Interface to interact with the database(s), executing queries and returning data with zero overhead



```
INSTALL httpserver FROM community;  
LOAD httpserver;
```

```
-- Start the embedded API server
```

```
SELECT httpserve_start('localhost', 80, 'user:pass');
```

```
HTTP server started on 0.0.0.0:80
```

```
# Query with any HTTP client
```

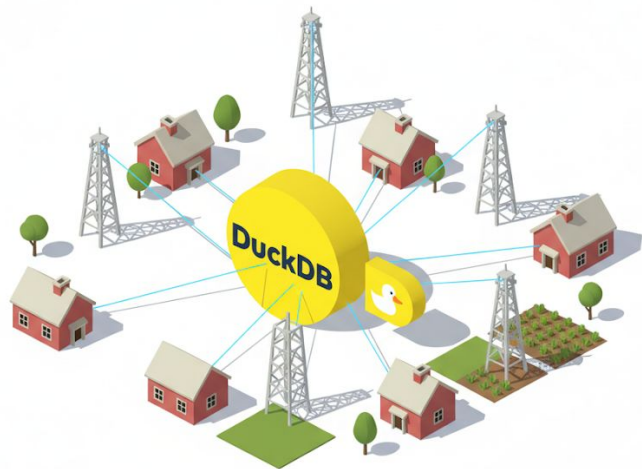
```
curl -X POST -d "SELECT version()" "http://user:pass@localhost/"
```

HTTP Client Extension

Simple HTTP GET/POST Client helpers for DuckDB

The **HTTP Client** extension enables direct querying of HTTP endpoints from within **DuckDB** SQL allowing users to fetch remote data over HTTP/HTTPS, integrate APIs, and join external web resources with local analytics.

This extension provides GET/POST helpers and was designed by and for data engineers who need to enrich DuckDB queries with any dataset from web services, REST APIs, or cloud-hosted archive, making it easier to build data pipelines and joins with remote systems without leaving native DuckDB SQL



```
INSTALL http_client FROM community;  
LOAD http_client;
```

Functions

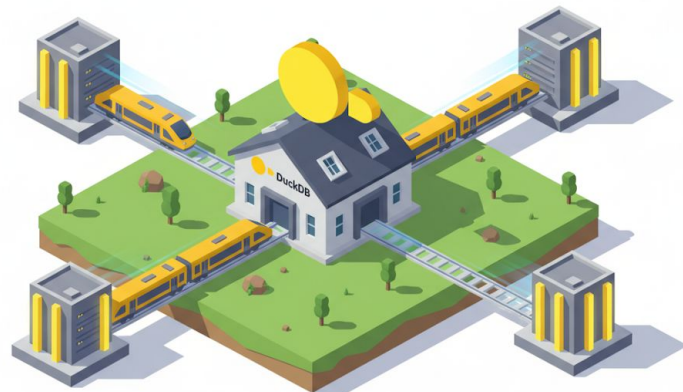
- `http_get(url)`
- `http_post(url, headers, params)`
 - Sends POST request with params encoded as a JSON object
- `http_post_form(url, headers, params)`
 - Sends POST request with params being application/x-www-form-urlencoded encoded

CHSQL Client Extension

100+ Clickhouse SQL Macros + Native Binary scanner

The **CHSQL** extension implements a 100+ command and function macros for SQL dialect compatibility between **ClickHouse** and **DuckDB** in queries and functions.

The **CHSQL Native** extension implements a native binary **ClickHouse** scanner and allows users to leverage DuckDB's analytical capabilities while accessing data and functions in ClickHouse servers, seamlessly bridging the two systems for efficient data analysis and interoperability. Supports encryption and authentication



```
INSTALL chsql FROM community;
LOAD chsql;

--- Use any of the 100+ ClickHouse Macros
SELECT IPv4StringToNum('127.0.0.1'), IPv4NumToString(2130706433);

INSTALL chsql_native FROM community;
LOAD chsql_native;

--- Use native binary scanner to access remote clickhouse services
SELECT * FROM clickhouse_scan("SELECT version(), 'quack'");
--- Read and Write native clickhouse binary files
SELECT * FROM clickhouse_native('/tmp/numbers.clickhouse');
SELECT number FROM numbers(1) INTO OUTFILE '/tmp/numbers.clickhouse' FORMAT Native;
```


Cronjob Extension

Execute and Schedule DuckDB queries using CRON

The **Cronjob** extension implements a cron scheduler and runner within DuckDB able to run and execute SQL queries, functions and macros. No surprises there. Ready to run trivial (downsampling, compact) or dangerous tasks (truncate, drop) and ready to interact with other event based extensions such as Radio

Scheduled jobs are accessible through a dedicated table function where they can be inspected, managed and deleted within a running DuckDB session.



```
INSTALL cronjob FROM community;
LOAD cronjob;

-- Every 15 seconds during hours 1-4
SELECT cron('SELECT now()', '* /15 * 1-4 * * *');

-- Every Monday through Friday at 7:00:00 AM
SELECT cron('SELECT data_cleanup()', '0 0 7 ? * MON-FRI');

-- Every 5 minute (wipe old data)
SELECT cron('DELETE FROM somewhere WHERE ts < NOW() - INTERVAL ''1 hour'', '* /5 * * * *');
-- Inspect running Jobs
SELECT * FROM cron_jobs();
```

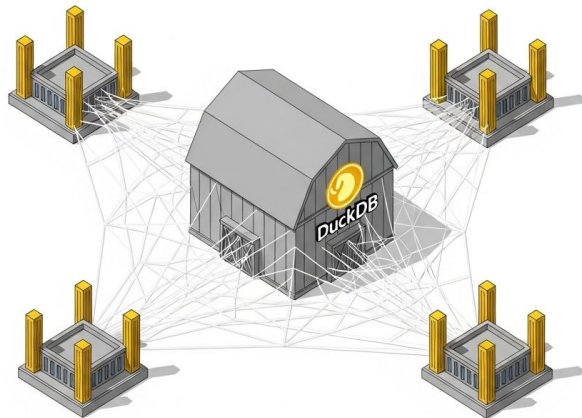
WebMacro Extension

Load and Share DuckDB Macros w/o Writing Extensions

The **WebMacro** extension was born from a discord idea - what if we could just hot-load scalar and table function macros as if they were DuckDB extensions?

Just write a simple or complex macro, upload it anywhere and you're ready to go!

All you need is DuckDB Webmacro and your SQL knowledge to become a maker!



```
INSTALL webmacro FROM community;
LOAD webmacro;

-- Share & Load a webmacro from github/gist/pasties/etc
SELECT load_macro_from_url('https://quacks.cc/r/search_posts') as res;

-- Use your webmacro loaded functions instantly
SELECT * FROM search_posts('qxp.bsky.social', text := 'quack');
```

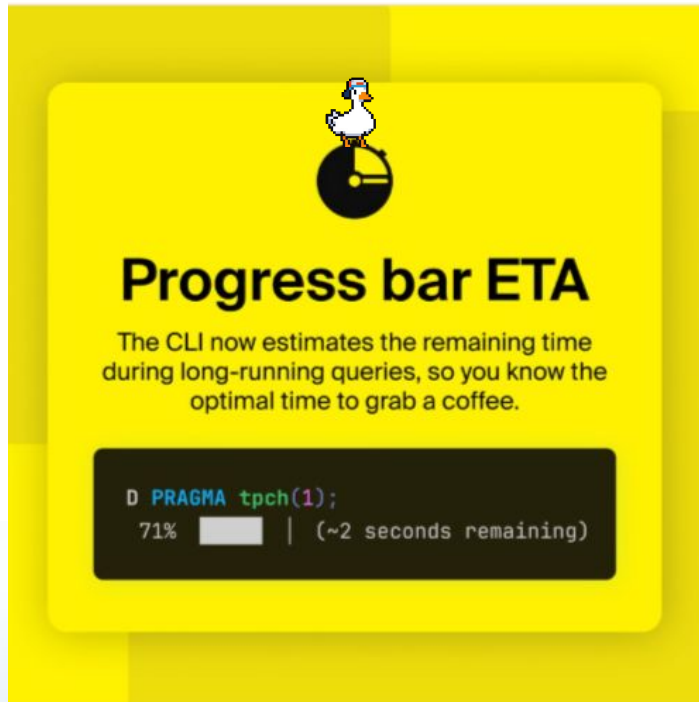
qxp.bsky.social	qxp	This is super cool...	...	1	0	1	0		
qxp.bsky.social	qxp	github.com/quacksc...	...	0	1	2	0		
qxp.bsky.social	qxp	#DuckDB works grea...	...	2	3	24	0		

DuckDB 1.4 Support 🙌

🙌 DuckDB 1.4

The latest LTS release is out and we're getting ready to support it across all of our extensions!

We're also proud to see the **Progress Bar ETA** being featured under the spotlight for this release and we hope you'll enjoy using it!



COMING SOON: Query.Farm Data Platform

DuckDB + Airport powered serverless data platform for On-Prem and Cloud

Query.Farm is working on its own platform based on *DuckDB, Apache Arrow and FlightSQL* - designed for heavyweight real-time data & time series, delivering ETL, serverless queries and scalability

Query.Farm Engine Features

- Flight Airport powered
- DuckLake/Iceberg/Delta storage
- Arrow + Parquet Storage
- Real-Time Ingestion
- Extensible Schemas
- Encryption / Row/Column ACLs
- Progressive Compaction
- Extensible Storage Tiers
- Serverless, Infinite Scalability
- Native Agentic support



Join Us / Build Upon Query.Farm Extensions 🙌🦆

🙌 Open Source

- Extensions are MIT/Apache2 licensed
- Free to use, inspect and build upon

💡 Share Your Ideas

- Have a new feature or extension idea?
We'd love to hear it!
- We are on the DuckDB Discord
- Join our mailing list and ask away

🤝 Contribute & Collaborate

- Submit pull requests, suggest improvements, or co-develop extensions
- Help us make DuckDB even more powerful for everyone

🌱 Support for Extension Developers

- We actively support other developers building DuckDB extensions
- Guidance, best practices and collaboration to bring your ideas to life



Query.Farm

The place where DuckDB
Community Extensions Grow

<https://query.farm>

```
ID SELECT tp_qr('https://query.farm');  
tp_qr('https://query.farm')
```

