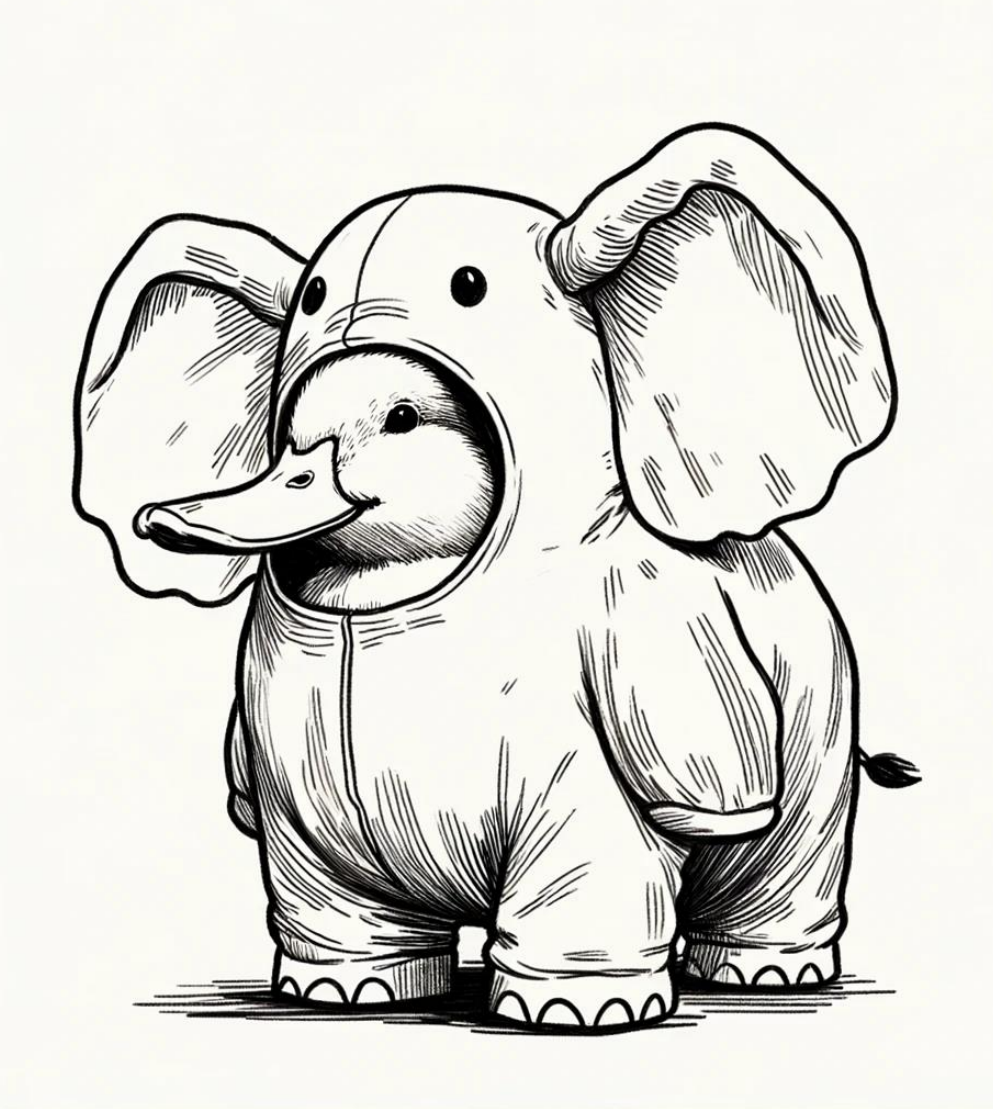
Building a Postgres Data Warehouse using DuckDB

Marco Slot – marco.slot@crunchydata.com



crunchy data

About me

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

crunchy data

MSc

PhD

Acquisition

2009

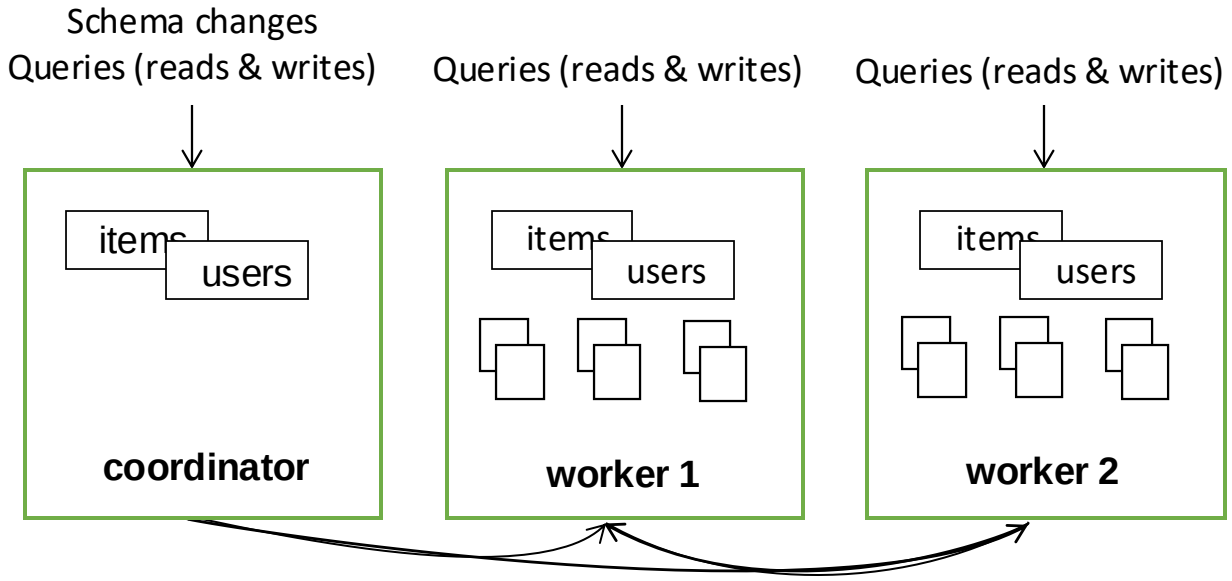
2014

2019

2024

Past Project: Citus

Citus is a PostgreSQL *extension* that can distribute tables across a cluster of PostgreSQL servers.



Multi-tenant apps
Real-time analytics

GitHub: https://github.com/citusdata/citus

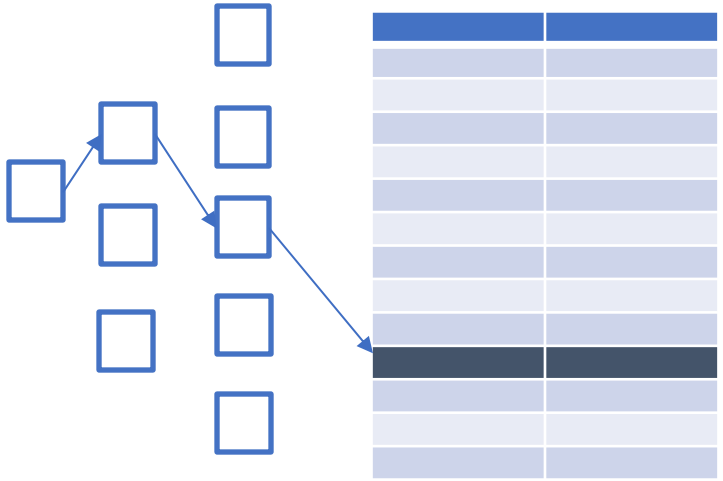
"Citus: Distributed PostgreSQL for Data-Intensive Applications" (SIGMOD '21)

crunchy data

OLTP	OLAP
Operational system of record	Analytics on collection of data sources
SQL	SQL
Transactions	Transactions
High query rate, small queries	Low query rate, big queries
Low response time	High scan throughput
User-facing applications	Business-facing dashboards
Mission-critical, always on	On demand, business hours

Row-oriented vs Column-oriented

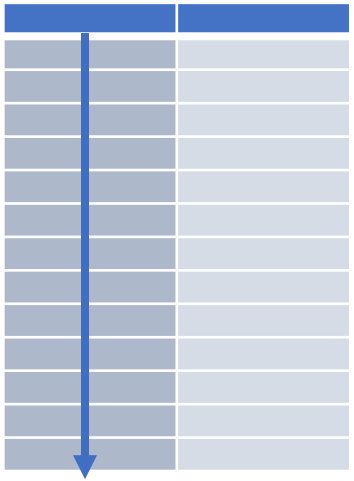
Row-oriented storage & execution



```
SELECT * FROM orders
WHERE orderid = $1
```

At scale: Fast on OLTP, Slow on OLAP

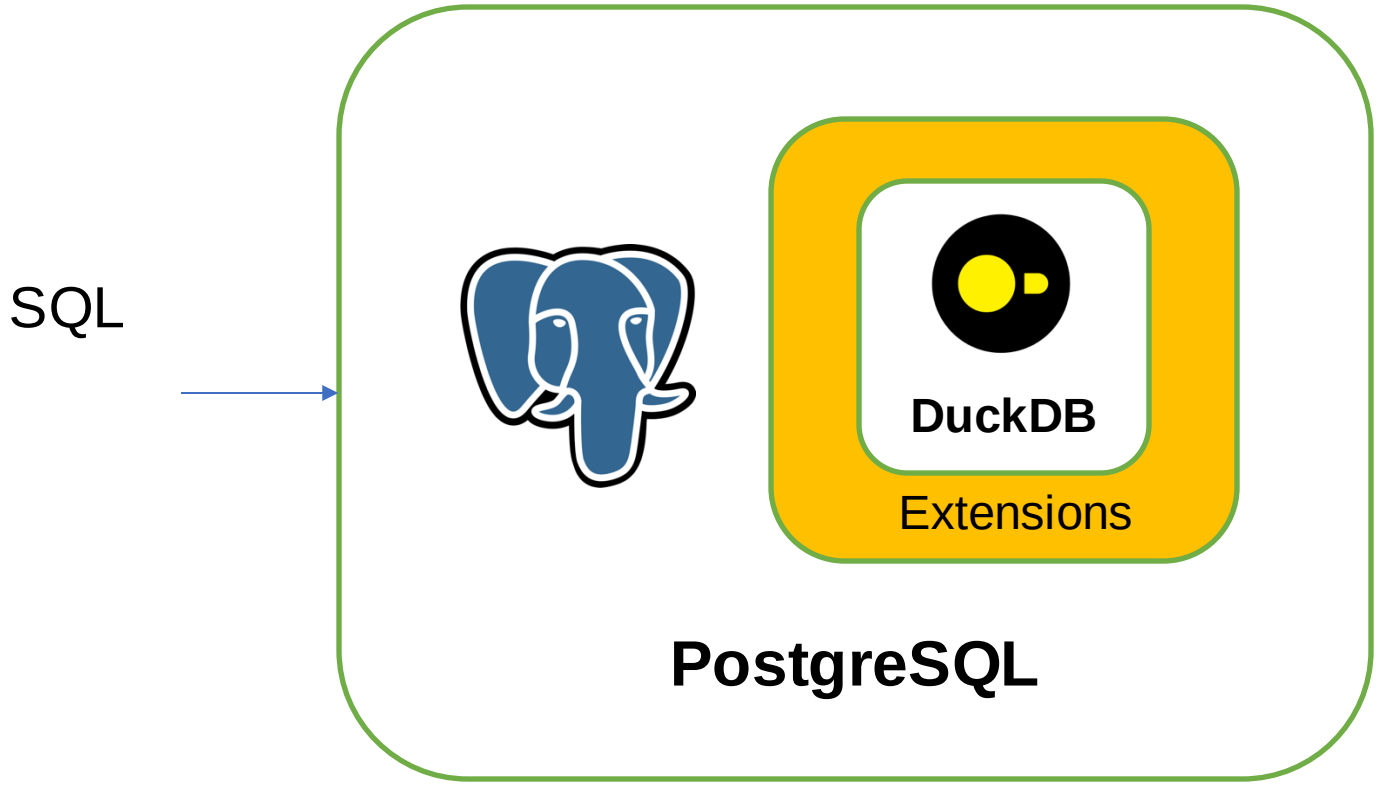
Column-oriented storage & execution



```
SELECT productid, count(*) FROM orders
GROUP BY 1 ORDER BY 2 DESC LIMIT 10
```

At scale: Slow on OLTP, Fast on OLAP

Hybrid OLTP/OLAP architecture



Why Hybrid OLTP/OLAP architecture?

Bad idea:

- Run analytical queries on operational system of record

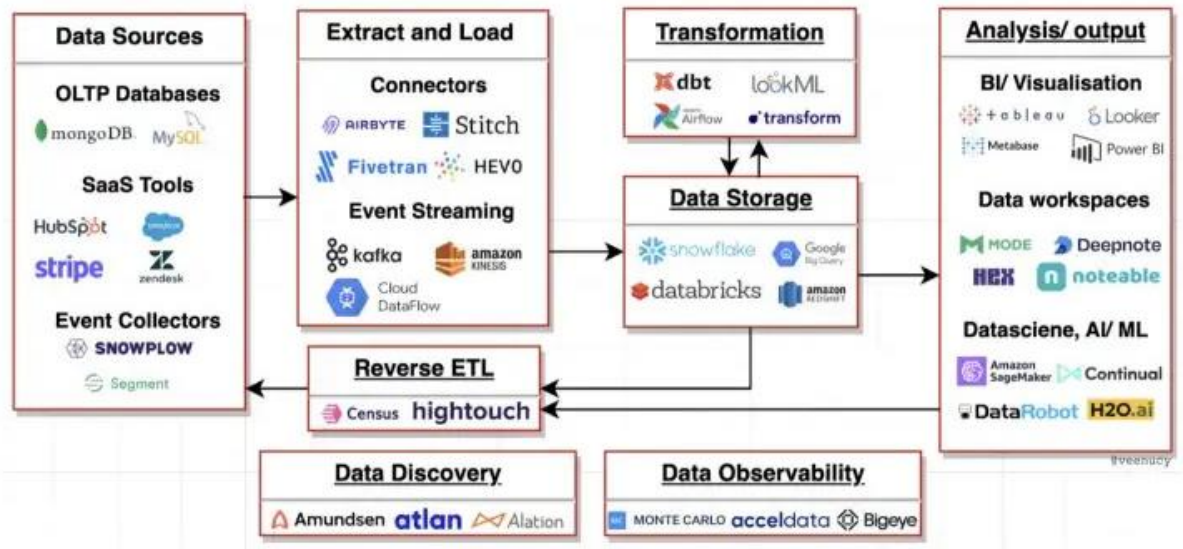
Good ideas:

- Speed up “accidental Postgres data warehouse”
- Fast insert queues for analytics tables
- Build and query materialized views in same system
- Bookkeeping for file import/export
- …

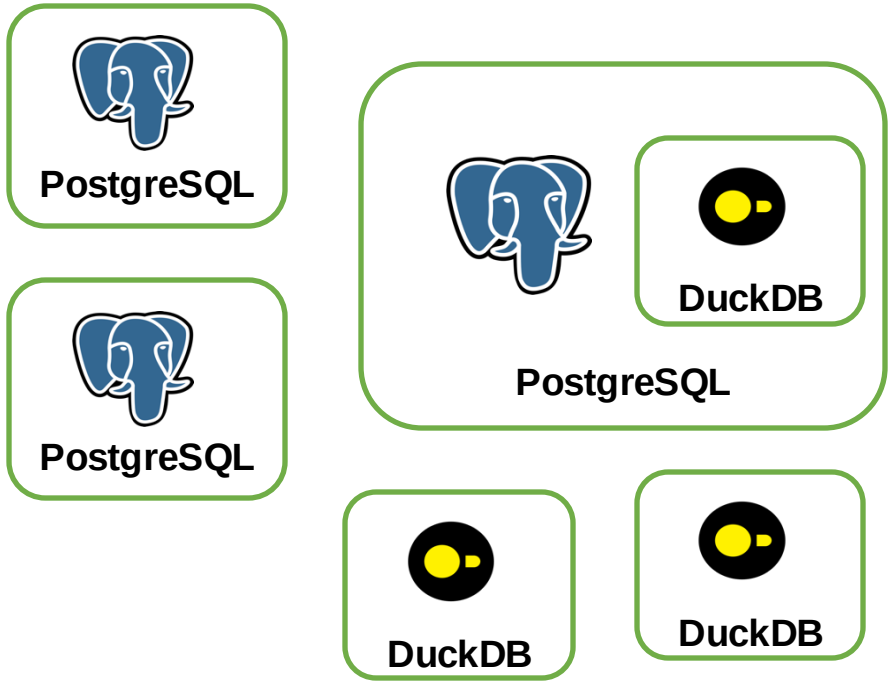
Stack simplification!

Modern Data Stack

Modern Data Stack



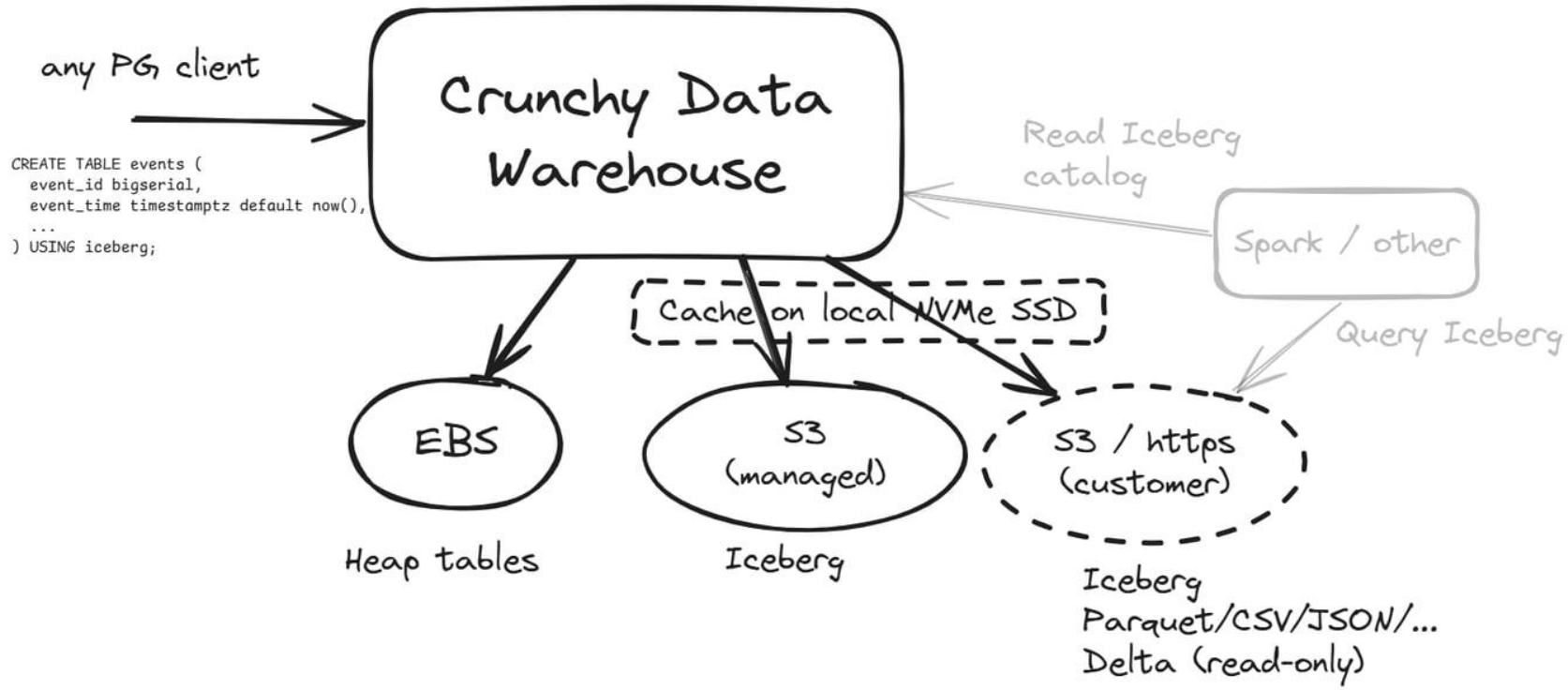
Post-Modern Data Stack



crunchy data

Crunchy Data Warehouse

Managed PostgreSQL with Iceberg and data lake tables, 10-100x faster for analytics by integrating DuckDB and write-through file caching.



crunchy data

Constellation of Postgres Extensions

Add new Postgres user experiences through many small, composable extensions.

List of installed extensions

Name	Version	Schema	Description
crunchy_base	1.4	pg_catalog	Crunchy Data base extension
crunchy_copy	2.0	pg_catalog	Data lake copy extension
crunchy_data_warehouse	2.0	pg_catalog	Data Warehouse for Crunchy Bridge
crunchy_extension_updater	1.0	pg_catalog	Crunchy Data Extension Updater
crunchy_iceberg	2.0	pg_catalog	Iceberg in PostreSQL
crunchy_lake_analytics	2.1	pg_catalog	Crunchy lake analytics and Iceberg tables
crunchy_map	1.1	pg_catalog	Associate array, dict / map type.
crunchy_query_engine	2.1	pg_catalog	Crunchy query engine common library
crunchy_spatial_analytics	2.1	pg_catalog	spatial analytics on data lakes
pg_cron	1.6	pg_catalog	Job scheduler for PostgreSQL
pg_incremental	1.2	pg_catalog	Incremental Processing by Crunchy Data
pg_parquet	0.2.0	public	copy data between Postgres and Parquet

Querying data lakes

Adjust DDL and SQL behavior to query files in data lakes.

```
create foreign table events ()
server crunchy_lake_analytics
options (
 path 's3://mybucket/events/*.csv');
```

→

```
DESCRIBE FROM 's3://…';
```
Generate composite types if needed

```
select date_bin(…), count(*)
from events
where event_time > now() – interval '7 days'
group by 1;
```

→

```
SELECT time_bucket(…), count(*)
FROM read_csv('s3://…')
WHERE event_time > '2025-02-13 …'
GROUP BY 1;
```

```
copy (…) to 's3://mybucket/res.parquet';
```

→

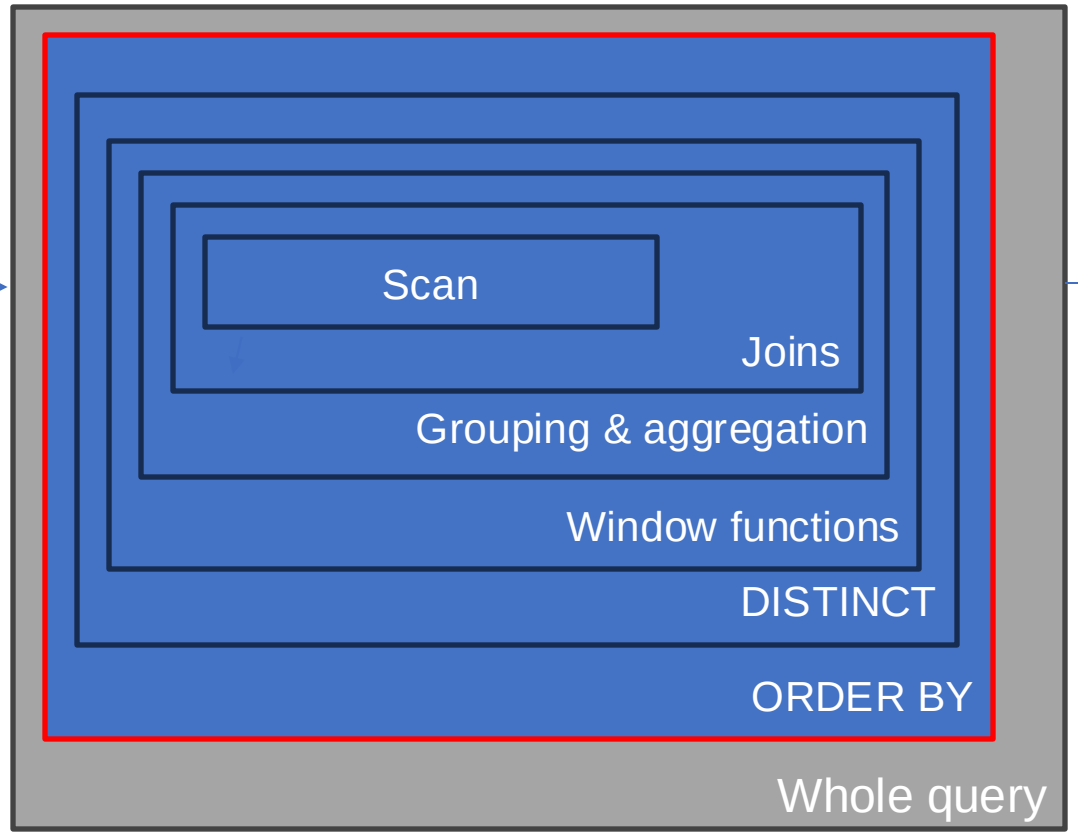
```
COPY …
```

crunchy data

Extending the Postgres query planner

Extensions can propose or enforce alternative plans for whole query or fragments.

```
with top10 as (
  select cust_id, sum(…)
  from sales
  where …
  order by 2 desc limit 10
)
select pg_func(cust_id)
from top10;
```



CteScan
pg_func(cust_id)

CustomScan (DuckDB)

```
select …
from read_parquet(…)
where …
order by …
```

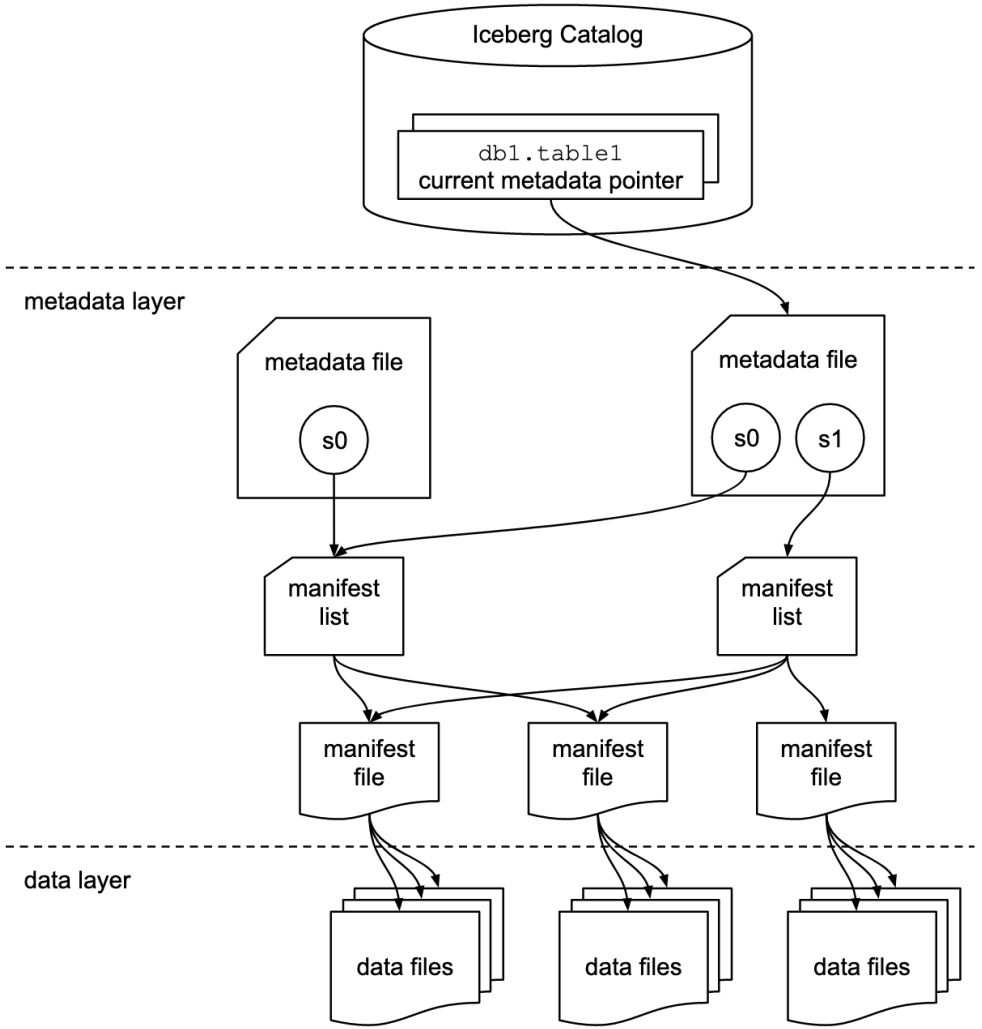
Storing data for analytics: Apache Iceberg

Iceberg defines a way to store a table in object storage

with support for schema changes, transactions, ….

Iceberg ingredients:

- Parquet (columnar) data files
- Parquet deletion files
- Tree of metadata files on which Parquet files are part of the table
- Catalog to find top-level metadata file



Storing data for analytics: Apache Iceberg

Capture queries, writes, & schema changes to provide a transactional table experience, backed by Iceberg in S3.

```
create table chats (
  message_id bigserial not null,
  thread_id bigint not null,
  …
) using iceberg;
```

Write metadata (avro, json) files to S3, insert to catalog

```
…
update chats set answer = '42'
where question is null;
```

```
SELECT * FROM read_parquet(…, filename=…, file_row_number=…)
WHERE question IS NULL;
```

Write updated rows into new Parquet file.
Write deleted rows into position delete Parquet file.
Write metadata (avro, json) files to S3, update catalog

```
select count(*) from chats;
```

```
SELECT count(*)
FROM read_parquet(…, schema=…, filename=…, file_row_number=…))
WHERE (filename, file_row_number)
NOT IN (SELECT (file_path, pos) FROM read_parquet(…));
```

crunchy data

```
postgres=# /*
postgres*#  Use Iceberg with an insert queue for fast inserts
postgres*#  in Crunchy Data Warehouse.
postgres*# */
postgres-#
postgres-#
```

DuckDB experiences

Great:

Performance

read_parquet/json/csv

Wildcards

Near Postgres compatibility

Extensibility

Concurrency

Bugs get fixed

Could be better:

Memory management for complex queries

Parsing and escaping of nested types

Parquet pruning

Azure/GCP/S3 feature support

Query interrupts

Summary

PostgreSQL with state-of-the-art analytics is possible by integrating DuckDB and Iceberg. Can significantly simplify data stacks.

Crunchy Data Warehouse is the first production-ready solution.

Let’s see how it goes ☺

Others under development: pg_analytics, pg_duckdb, pg_mooncake

crunchy data

Questions?

Or drinks…