Kùzu

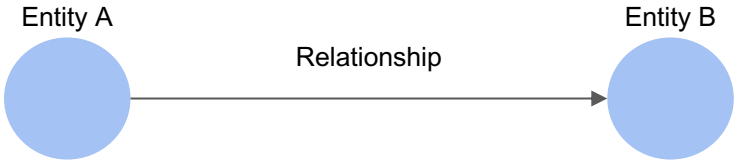
Graph Database Management System

https://kuzudb.com

TUC Aug 2024

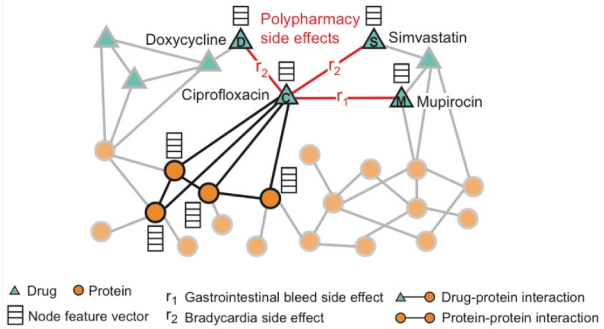
What are graphs?

Abstract representation of entities and relationships



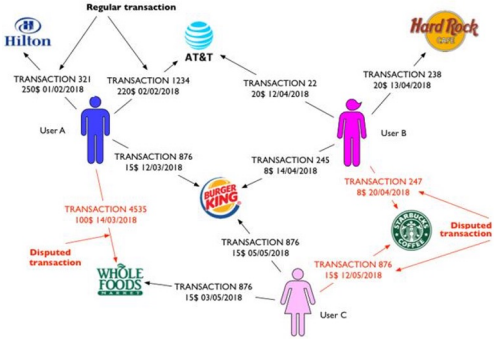
Graphs: Natural ways to represent data

Social network

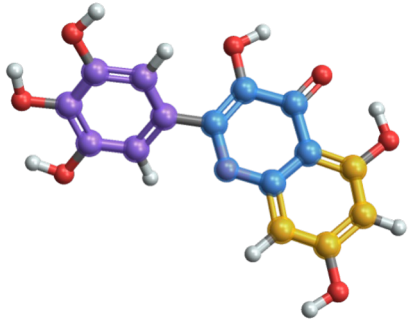


Drug interactions

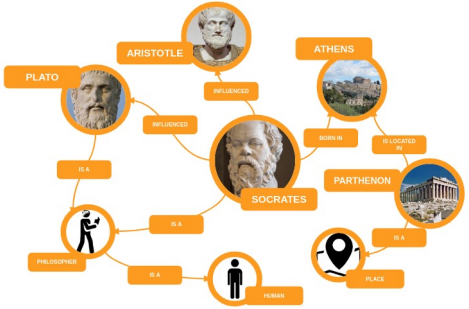
Traffic networks



Transaction graph



Molecular networks



Knowledge graphs

Competent feature set for graph database

Pattern Matching

- Pre-defined joins
- Complex pattern

Recursive Join

- Different algorithms
- Parallel computation
- Skewness handling

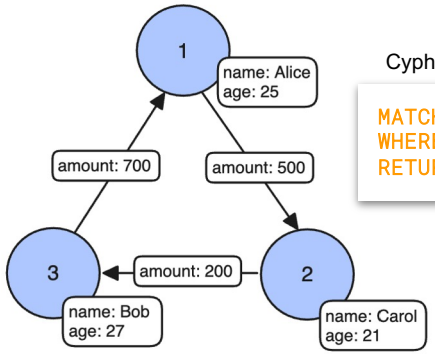
Relational Operation

- Filter
- Projection
- …

Blog post: https://kuzudb.com/blog/what-every-gdbms-should-do-and-vision.html

What is Kùzu

Structure property graph data model



Cypher query language

```
MATCH (a)-[:]->(b)
WHERE a.age > 25
RETURN b.name
```

Embeddable
(similar to DuckDB/SQLite)

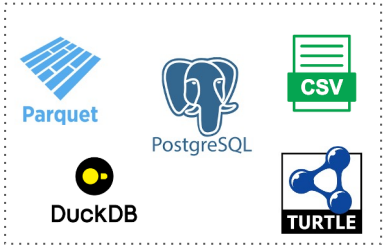
```
import kuzu

db = kuzu.Database("db")
conn = kuzu.Connection(db)
res = conn.execute("MATCH (a)-[:]->(b)")
print(res.get_as_df())
```

ACID transactions

Permissively licensed

Integrations with ML/AI frameworks

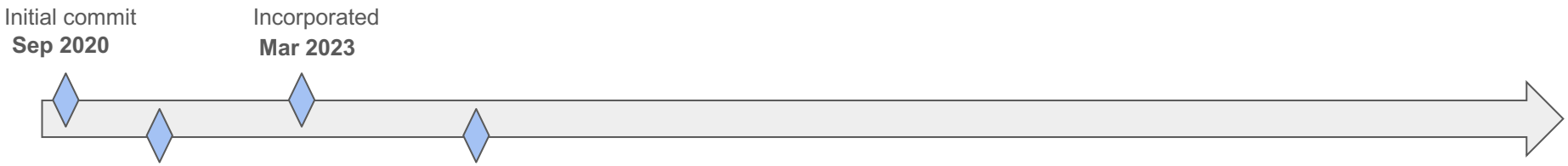


Interoperable with databases
and data formats

Rich bindings

Learn more at https://kuzudb.com

History of Kùzu



	Name	Age
Node Group 0	Jean-Christophe	20
	Karl	
	…	…
Node Group 1	Alice	29
	Dave	30
Node Group …	…	…

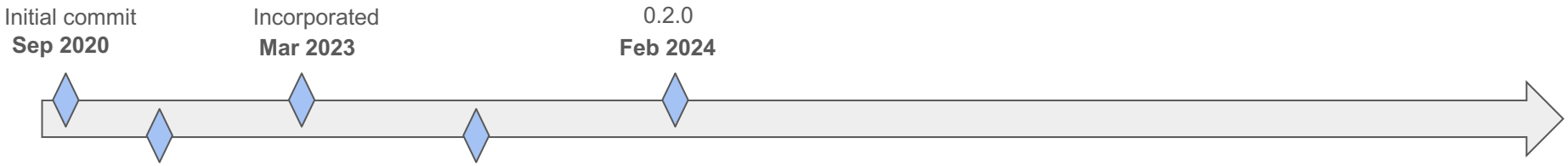
History of Kùzu

RDF support

```
CREATE RDFGRAPH KuzuKG;

COPY KuzuKG FROM 'kuzu_kg.ttl';

WITH "http://kuzu.io/rdf-ex#" as kz
MATCH (s {iri: kz+"Waterloo"})-[p]->(o)
RETURN s.iri, p.iri, o.iri;
```



	Name	Age
Node Group 0	Jean-Christophe	20
	Karl	
	…	…
Node Group 1	Alice	29
	Dave	30
Node Group …	…	…

History of Kùzu

RDF support

```
CREATE RDFGRAPH KuzuKG;

COPY KuzuKG FROM 'kuzu_kg.ttl';

WITH "http://kuzu.io/rdf-ex#" as kz
MATCH (s {iri: kz+"Waterloo"})-[p]->(o)
RETURN s.iri, p.iri, o.iri;
```

Initial commit
Sep 2020

Incorporated
Mar 2023

0.2.0
Feb 2024

Nov 2022

Open sourced

Nov 2023

0.1.0

NodeGroup-based storage

	Name	Age
Node Group 0	Jean-Christophe	20
	Karl	
	…	…
Node Group 1	Alice	29
	Dave	30
Node Group …	…	…

May 2024

0.4.0

Scan DataFrame, DuckDB, Postgres & SQLite

```
df = pd.DataFrame({
    "name": ["Adam", "Karissa"],
    "age": [30, 40]
})
result = conn.execute("LOAD FROM df RETURN *;")
```

History of Kùzu

RDF support

```
CREATE RDFGRAPH KuzuKG;

COPY KuzuKG FROM 'kuzu_kg.ttl';

WITH "http://kuzu.io/rdf-ex#" as kz
MATCH (s {iri: kz+"Waterloo"})-[p]->(o)
RETURN s.iri, p.iri, o.iri;
```

MVCC, Json & remote db

```
ATTACH 's3://kuzu-example/university' AS uw (dbtype kuzu);

CREATE NODE TABLE Example (
    id INT64,
    content STRING[],
    PRIMARY KEY (id));
COPY Example FROM 'data.json';
```

Initial commit
Sep 2020

Incorporated
Mar 2023

0.2.0
Feb 2024

0.5.0
July 2024

Nov 2022

Open sourced

Nov 2023

0.1.0

NodeGroup-based storage

	Name	Age
Node Group 0	Jean-Christophe	20
	Karl	
	…	…
Node Group 1	Alice	29
	Dave	30
Node Group …	…	…

May 2024

0.4.0

Scan DataFrame, DuckDB, Postgres & SQLite

```
df = pd.DataFrame({
    "name": ["Adam", "Karissa"],
    "age": [30, 40]
})
result = conn.execute("LOAD FROM df RETURN *;")
```

History of Kùzu

RDF support

```
CREATE RDFGRAPH KuzuKG;

COPY KuzuKG FROM 'kuzu_kg.ttl';

WITH "http://kuzu.io/rdf-ex#" as kz
MATCH (s {iri: kz+"Waterloo"})-[p]->(o)
RETURN s.iri, p.iri, o.iri;
```

MVCC, Json & remote db

```
ATTACH 's3://kuzu-example/university' AS uw (dbtype kuzu);

CREATE NODE TABLE Example (
    id INT64,
    content STRING[],
    PRIMARY KEY (id));
COPY Example FROM 'data.json';
```

Initial commit
Sep 2020

Incorporated
Mar 2023

0.2.0
Feb 2024

0.5.0
July 2024

Nov 2022

Open sourced

Nov 2023

0.1.0

NodeGroup-based storage

	Name	Age
Node Group 0	Jean-Christophe	20
	Karl	
	…	…
Node Group 1	Alice	29
	Dave	30
Node Group …	…	…

May 2024

0.4.0

Scan DataFrame, DuckDB, Postgres & SQLite

```
df = pd.DataFrame({
    "name": ["Adam", "Karissa"],
    "age": [30, 40]
})
result = conn.execute("LOAD FROM df RETURN *;")
```

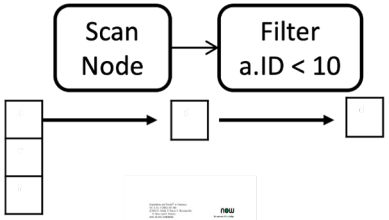
Aug 2024

0.6.0

In-memory mode

Kùzu Internals

Vectorization



Factorization

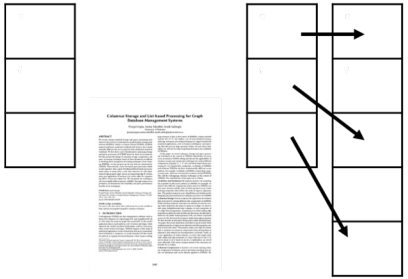
b, name		a		c
$\{L_1$, Liz$\}$	X	$\{U_1,\ldots,U_{100}\}$	X	$\{C_1, \ldots, C_{100}\}$
U				
$\{L_2$, Liz$\}$	X	$\{U_{100},\ldots,U_{199}\}$	X	$\{C_{100},\ldots,C_{199}\}$

Morsel-driven Parallelism

- Scans are "morselized" across threads

Novel Join Algorithms

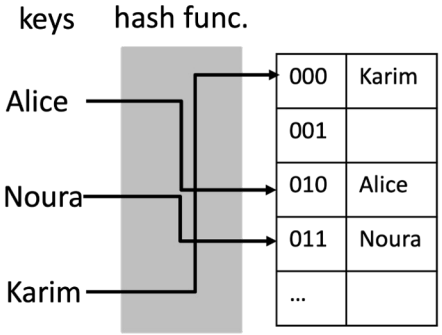
Disk-based Columnar Storage



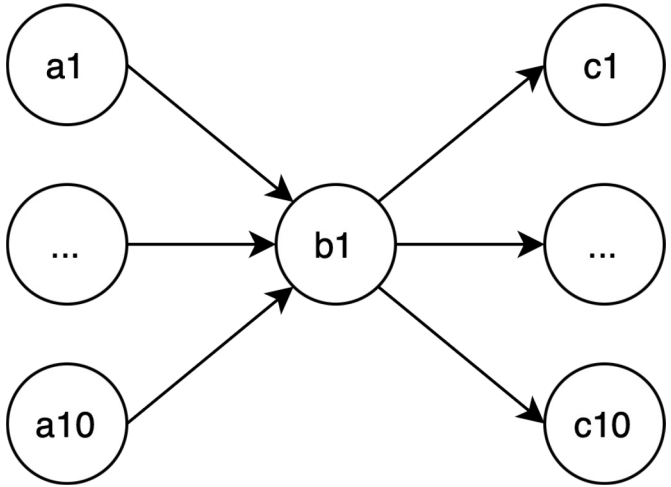
node properties

CSR-based join indices & edge props

Disk-based Hash Index



Factorized query processing



a1	b1	c1
...	...	...
a1	b1	c10
a10	b1	c1
...	...	...
a10	b1	c10

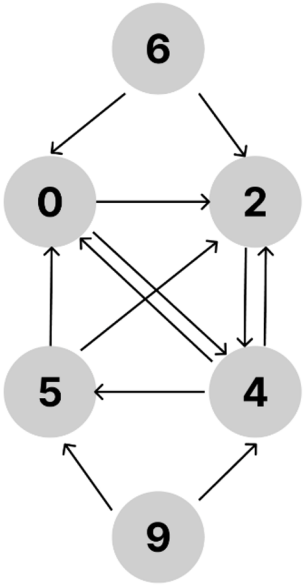
100 tuples

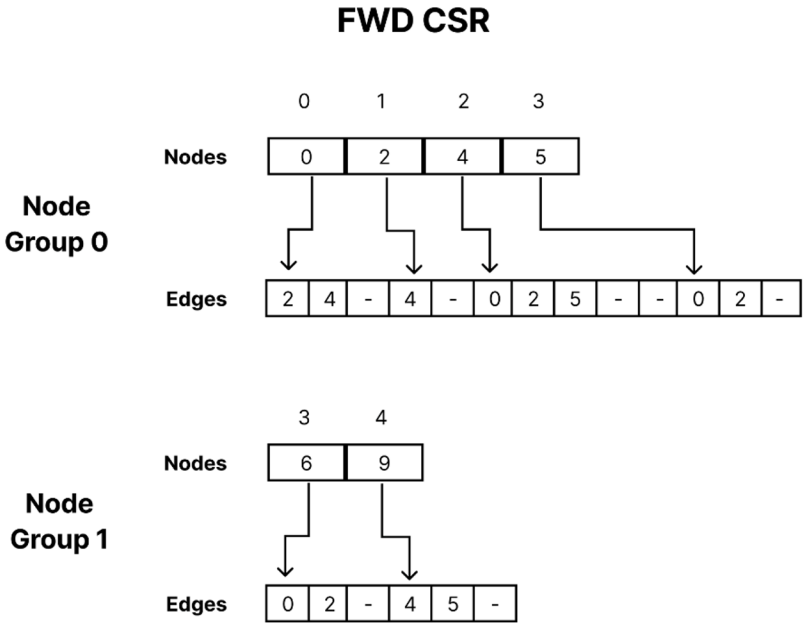
Flat representation

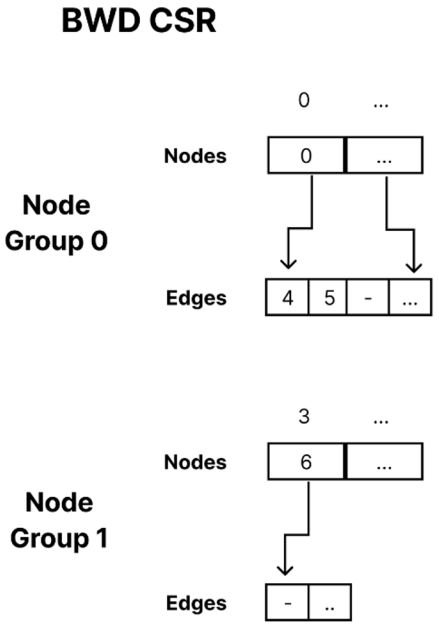
a1, ..., a10	b1	c1, ..., c10

Factorized representation

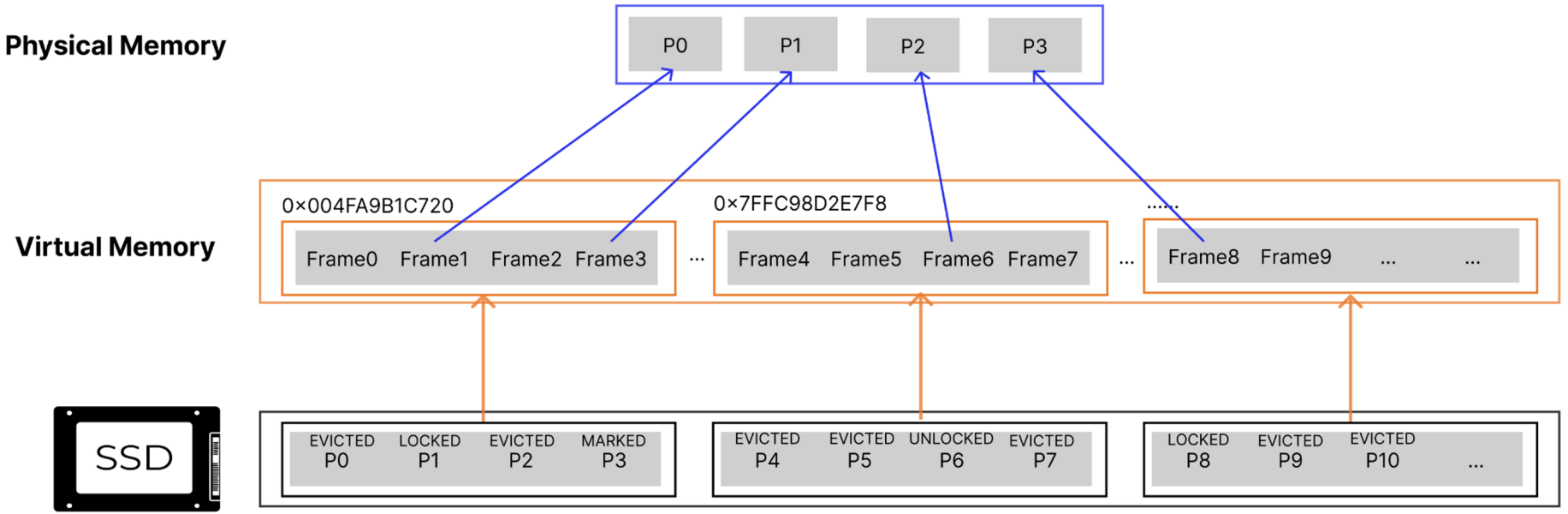
CSR index


6
0
2
5
4
9


FWD CSR
Node Group 0
Nodes
0 1 2 3
0 2 4 5
Edges
2 4 - 4 - 0 2 5 - - 0 2 -
Node Group 1
Nodes
3 4
6 9
Edges
0 2 - 4 5 -


BWD CSR
Node Group 0
Nodes
0 ...
0 ...
Edges
4 5 - ...
Node Group 1
Nodes
3 ...
6 ...
Edges
- ..

VM-backed buffer manager

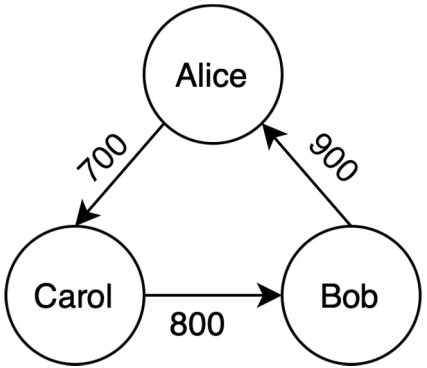


- Viktor Leis, et al. Virtual-Memory Assisted Buffer Management. SIGMOD'23.

Roadmap - Direct Query over RDBMS

Account		
ID	name	country
1	Alice	CA
2	Carol	USA
3	Bob	CA
...	...	...

eTransfer		
from	to	amount
1	2	700
2	3	800
3	1	900
...	...	...




```
CREATE NODE TABLE Account (
  ID INT64,
  name STRING,
  country STRING,
  PRIMARY KEY (id)
);

CREATE REL TABEL eTransfer (
  FROM Person TO Person,
  amount INT64
);

ATTACH duck.db AS duck (dbtype duckdb);
Copy Account FROM duck.Account;
Copy eTransfer FROM duck.eTransfer;
```

Require graph modeling & data moving

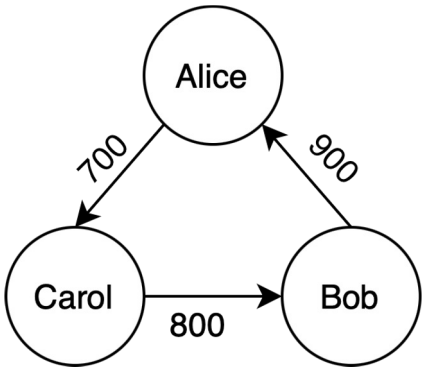
Roadmap - Direct Query over RDBMS

Account		
ID	name	country
1	Alice	CA
2	Carol	USA
3	Bob	CA
...	...	...

eTransfer		
from	to	amount
1	2	700
2	3	800
3	1	900
...	...	...

```
ATTACH duck.db AS duck (dbtype duckdb);
Create Node Table Account AS duck.Account;
Create REL Table eTransfer AS duck.eTransfer;

MATCH (p:Account)-[:eTransfer]->(p2:Account)
WHERE a.name = 'Alice'
RETURN p2.name
```



- Data stored in DuckDB
- Computation happens in Kùzu
- Extendable to Postgres, SQLite, CSV, …

Roadmap - Recursive Join

- Parallelize within single source node
  - Handle skew
- Extendable framework
  - Shortest path
  - Weight shortest path
  - Connected component

Join our Discord!

github.com/kuzudb/kuzu

@kuzudb

https://kuzudb.com/blog/