On the Graph Query Language

MALCOLM CROWE, FRITZ LAUX

INFOSYS 2025 CONGRESS

Malcolm Crowe

University of the West of Scotland
Email: malcolm.crowe@uws.ac.uk

IARIA

- Malcolm Crowe is an Emeritus Professor at the University of the West of Scotland, where he worked from 1972 (when it was Paisley College of Technology) until 2018.
- He gained a D.Phil. in Mathematics at the University of Oxford in 1979.
- He was appointed head of the Department of Computing in 1985. His funded research projects before 2001 were on Programming Languages and Cooperative Work.
- Since 2001 he has worked steadily on PyrrhoDBMS to explore optimistic technologies for relational databases and this work led to involvement in DBTech, and a series of papers and other contributions at IARIA conferences with Fritz Laux, Martti Laiho, and others.
- Prof. Crowe has recently been appointed an IARIA Fellow.

Prof. Dr. Fritz Laux

(Retired), Reutlingen University
Email: fritz.laux@reutlingen-university.de

- Prof. Dr. Fritz Laux was professor (now emeritus) for Database and Information Systems at Reutlingen University from 1986 - 2015. He holds an MSc (Diplom) and PhD (Dr. rer. nat.) in Mathematics.
- His current research interests include
  - Information modeling and data integration
  - Transaction management and optimistic concurrency control
  - Business intelligence and knowledge discovery
- He contributed papers to DBKDA and PATTERNS conferences that received DBKDA 2009 and DBKDA 2010 Best Paper Awards. He is a panellist, keynote speaker, and member of the DBKDA advisory board.
- Prof. Laux is a founding member of DBTech.net (http://www.dbtechnet.org/), an initiative of European universities and IT-companies to set up a transnational collaboration scheme for Database teaching. Together with colleagues from 5 European countries he has conducted projects supported by the European Union on state-of-the-art database teaching.
- He is a member of the ACM and the German Computer Society (Gesellschaft für Informatik).

Plan of this tutorial

- Presenting Database Language GQL
- Explaining the LDBC FinBench data model
- Pointing out some shortcomings and proposing an improved data model
- Demonstration of GQL using the FinBench data model and some of the benchmark queries.

- Next: Graph data

IARIA

Graph Data?

- Databases mostly hold data in tables
- Internet is all about linked information
  - Data linked to more data
- In SQL based systems this uses keys
  - Foreign key is a reference to another table
  - Exploring linked data means joining tables
    - By foreign keys given by values of key columns
- Lots of links to follow means many joins
- So instead of tables, use idea of nodes and edges
  - Edges link nodes by reference to node identity (pointers)
  - Nodes and edges can have properties
  - Labels to indicate different types of object
- Labeled Property Graphs (LPG)
  - Many database management systems for LPG already

GQL Background

- Standardization and Database Technology
  - ISO/IEC 9075 (1987-) Information Technology - Database Languages – SQL
  - ISO/IEC 39075 (2024-) Information Technology - Database Languages – GQL [1]
- Follows Fritz Laux's Typed Graph Model [2]
- Malcolm Crowe's PyrrhoDBMS [3] is a partial implementation of GQL on top of SQL
- LDBC has a Financial Benchmark for GQL [4]

- Next: a little about Pyrrho

A little about PyrrhoDBMS

- Pyrrho [1] is a relational DBMS developed by Malcolm Crowe
  - Implements optimistic Concurrency Control supporting true transactional Serialization [5]
- Pyrrho supports a Typed Graph Model (TGM) on top of its relational DBMS
  - Node and Edge types are mapped into tables [6]
  - As consequence it supports a schema, in contrast to other graph data models
  - Lately, the new Database Language GQL [7,8] was implemented by Malcolm Crowe
  - GQL has a graph like pattern syntax, defined by ISO/IEC 39075

IARIA

LDBC FinBench Data Schema[4]

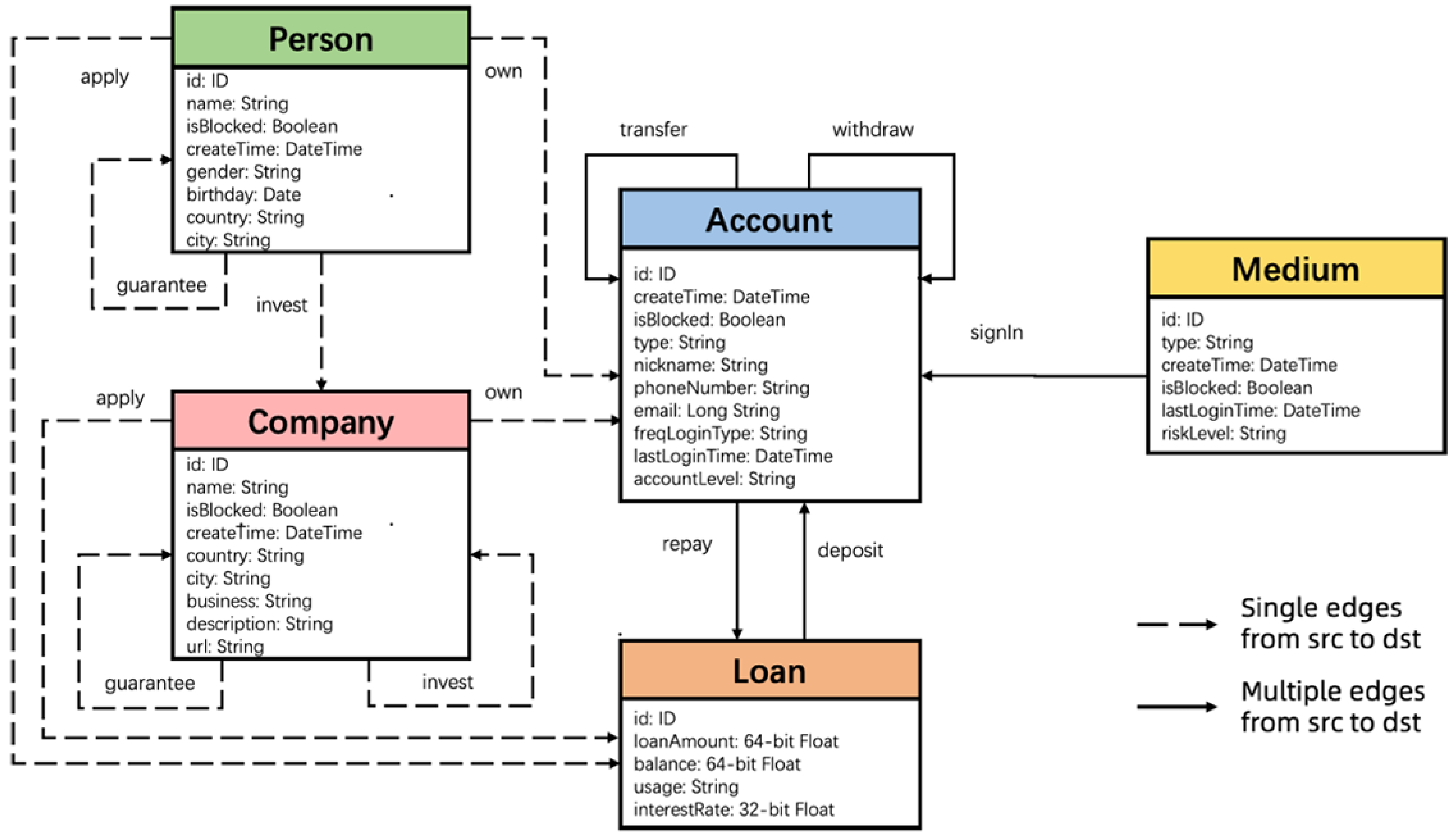


Figure 1. The LDBC FinBench data schema (from [6])

- Comments and weaknesses

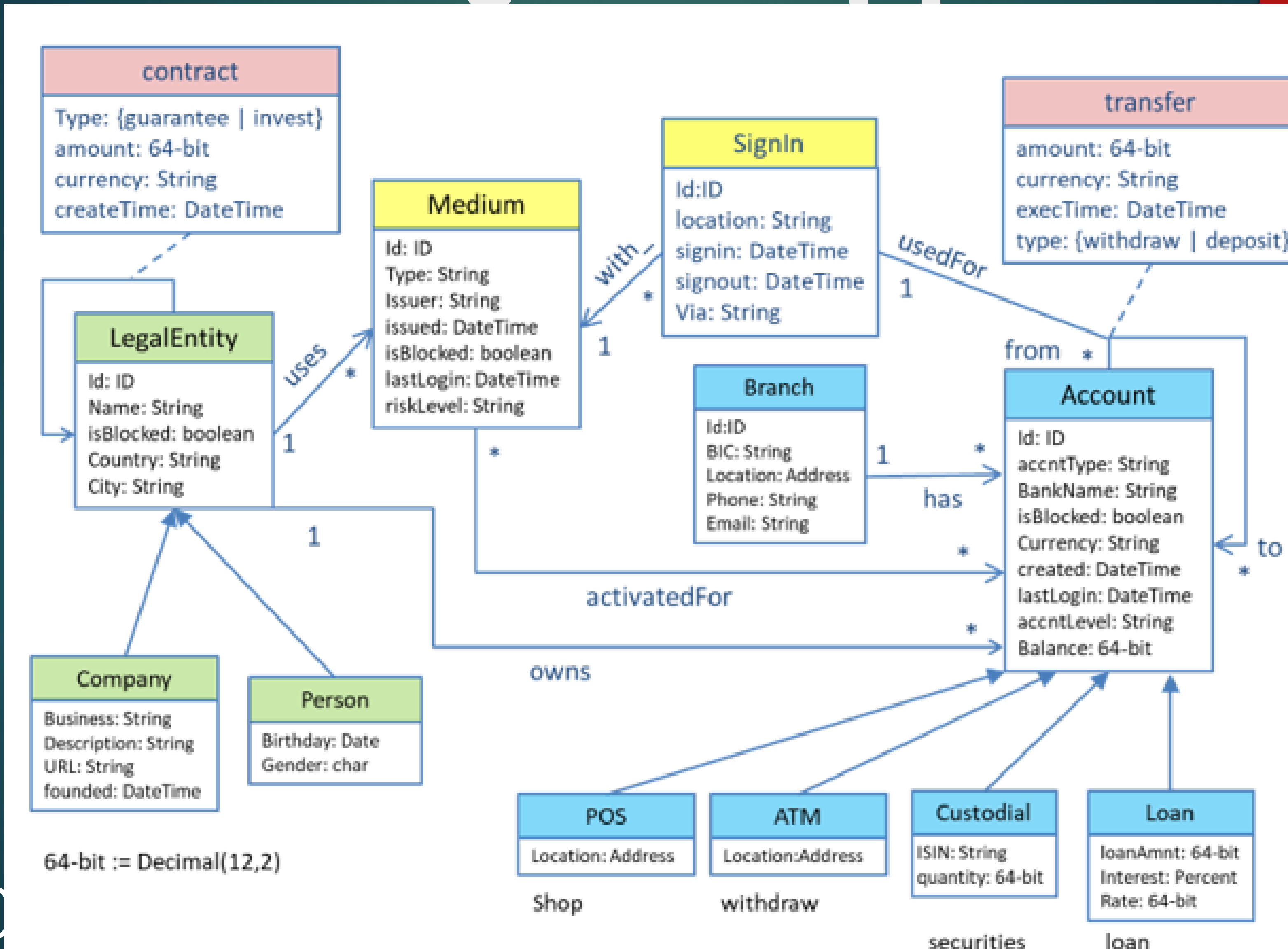
Comments on this model

- From TGM perspective it has some weaknesses and formal errors:
  - Only 1 signIn edge is allowed between account and medium (in UML & PGM)
    - signIn should be an entity
  - Multiple associations (own, guarantee, invest, apply) suggest similarity
    - Use inheritance and generalization
  - Repay and deposit are transactions as well
    - Every transaction needs a source and destination account to comply with legal accounting rules.
  - Not every Loan application will be granted

- A TGM confomant model

IARIA

Conforming to TGM[4]



Notes on conforming model

- 1) uses ternary edge type transfer to identify each transfer
- 2) uses generalization and inheritance to make the model simpler, more realistic and avoid repetition
- 3) Every transaction including withdraw and shop payment has a source and destination account. This complies with accounting rules.
- 4) Records only granted Loans as special type of an account.

- Implementation in tables

Implementation

- All TGM elements are mapped onto tables, for example Table Medium, Table activatedFor, Table transfer
- Each element has a unique position in the log-file. This position never changes
- The subgraph (:Medium)-[:activatedFor]->(:Account) is mapped to 3 tables
- `Select id, type, issuer, position from Medium where id = 1`

ID	TYPE	ISSUER	POSITION
1	creditCard	Amex	11862

- Select * from activatedFor where leaving = 11862

LEAVING	ARRIVING
11862	8525
11862	10683
11862	11104

- `Select id, accntType, bankName, position from Account where position = 8525 or position = 10683 or position = 11104`

ID	ACCNTTYPE	BANKNAME	POSITION
1	custodial account	Union Invest	8525
31	checking account	KSK Tübingen	10683
34	credit card	Amex	11104

- The demo shows that Medium #1 is activated for Accounts #1, #31, and #34

- GQL demo for this model

IARIA

GQL Demo for this model

- Queries
  - match (p:Person) return p.id, p.name, p.birthday, p.gender
  - match (le1:LegalEntity)-[c:contract {type:'guarantee'}]->(le2:LegalEntity)
  - match (le:legalentity)-[:owns]->(a:account)
  - [match (le:LegalEntity)-[:uses]->(m:Medium)<-[:with_]-(s:signIn)-[u:usedFor]->(t:transfer) return le.name, m.issuer, s.location, s.signIn, s.signout, t.Id as tId, t.amount, t.execTime]
  - [match (le:LegalEntity)-[:uses]->(m:Medium)<-[:with_]-(s:signIn)-[u:usedFor]->(t:transfer) match (b1:Branch)-[:has]->(fa:Account)-[t]->(ta:Account)<-[:has]-(b2:Branch) return le.name as accntOwner, m.issuer as MediumIssuer, s.location as SignInLocation, s.via, t.Id as taId, fa.id as fromId, fa.bankName as fromBank, b1.location as fromBranch, t.amount, t.execTime, ta.id as toId, ta.bankName as toBank, b2.location as toBranch]
- Inserting Nodes and Edges
  - If p, m, a & b exist, then the node s and edges [:uses], [:with_], [:transfer] and [:usedFor] will be inserted.
  - [match (p:person {id:2}), (m:medium {id:2}), (a:account {id:33}), (b:account {id:5})
    insert (p)-[:uses]->(m)<-[:with_]-(s:signin {id:15, location:'Stgt, Home PC', signin:timestamp'2024-01-06 20:10:00', signout:timestamp'2024-01-06 20:18:05', Via:'Home PC' }),
    (a)-[t:transfer {id:13, type:'transfer', amount:234.0, currency:'€', exectime: timestamp'2024-01-06 20:14:00'}]->(b),
    insert (s)-[:usedFor]->(t)]

- More about GQL

More details about GQL

- GQL statements include: Call, Match, Let, For, Filter, Return, Group, Order By and Page, Select
- A weakness: binding only for nodes, edges and paths; graphs are disjoint.
- A strength: the construction of working tables by rows and combining queries.
- Match pursues links through a given set of patterns: when we reach the end, we have a row for the working table
- Matches can be optional, e.g.

```
Match(p:Person) optional {Match(p)-[:worksFor]->(q)}
```

More complex queries

- To check consistency in the new model we can try queries such as

```
match (:legalEntity{id:lid})
-[:owns]->(:account{id:aid,accntType:atyp,bankname:bnm})
except match (:legalentity{id:lid})-[:uses]->(:medium)
-[:activatedFor]->(:account{id:aid,accntType:atyp,bankname:bnm})
order by aid
```

```
{match (l:legalEntity)-[:owns]->(a:account)}
except {match (l:legalentity)-[:uses]->(:medium)
-[:activatedFor]->(a:account)} return l.id, a.id as aid,
a.accntType, a.bankname order by aid
```

- The output

```
QL> [match (:legalEntity{id:lid})
> -[:owns]->(:account{id:aid,accntType:atyp,bankname:bnm})
> except match (:legalentity{id:lid})-[:uses]->(:medium)
> -[:activatedFor]->(:account{id:aid,accntType:atyp,bankname:bnm}) order by aid]
|---|---|-------------------|-------------------|
|LID|AID|ATYP               |BNM                |
|---|---|-------------------|-------------------|
|9  |2  |ATM / SB account   |Deutsche Bank      |
|9  |17 |ATM / SB account   |Deutsche Bank      |
|12 |12 |loan account       |Ing DiBa           |
|13 |13 |custodial account  |IBM stock, UBS     |
|14 |14 |loan account       |BNP                |
|14 |27 |checking account   |Deutsche Bank      |
|15 |22 |retirement account |Deutsche Bank      |
|16 |34 |credit card        |Amex               |
|21 |28 |checking account   |KSK Ulm            |
|23 |32 |POS account        |Deutsche Bank Stgt.|
|---|---|-------------------|-------------------|
QL> [{match (l:legalEntity)-[:owns]->(a:account)}
> except {match (l:legalentity)-[:uses]->(:medium)
> -[:activatedFor]->(a:account)} return l.id, a.id as aid, a.accntType, a.bankname order by aid]
|--|---|-------------------|-------------------|
|ID|AID|ACCNTTYPE          |BANKNAME           |
|--|---|-------------------|-------------------|
|9 |2  |ATM / SB account   |Deutsche Bank      |
|12|12 |loan account       |Ing DiBa           |
|13|13 |custodial account  |IBM stock, UBS     |
|14|14 |loan account       |BNP                |
|9 |17 |ATM / SB account   |Deutsche Bank      |
|15|22 |retirement account |Deutsche Bank      |
|14|27 |checking account   |Deutsche Bank      |
|21|28 |checking account   |KSK Ulm            |
|23|32 |POS account        |Deutsche Bank Stgt.|
|16|34 |credit card        |Amex               |
|--|---|-------------------|-------------------|
QL>
```

Complex queries in LDBC

- ComplexRead8 "Given a Loan and a specified time window between startTime and endTime, trace the fund transfer or withdraw by at most 3 steps from the account the Loan deposits. Note that the transfer paths of edge1, edge2, edge3 and edge4 are in a specific time range between startTime and endTime. Amount of each transfers or withdrawals between the account and the upstream account should exceed a specified threshold of the upstream transfer. Return all the accounts' id in the downstream of loan with the final ratio and distanceFromLoan."
- This query contains a path pattern, in synopsis it requires

```
[MATCH (:Loan{id:4612532092624966603})-[:deposit{amount:damt}]->()
  [()-[:transfer|withdraw {amount:amt,createTime:x}]->()]{1,3}
(:Account{id:dstId})
    return min(cardinality(amt)+1) as distancefromLoan, damt,
dstId, sum(amt[cardinality(amt)-1]/damt) as ratio
    group by (damt, dstId)]
```

- We will add clauses to set a threshold and time window and force the transfer times x to be in a temporal sequence:
- `Where (cardinality(x)=1 or x[cardinality(x)-2]<createtime`

- The work continues

The work continues ..

- Questions?

- References

References

- [1] ISO/IEC 39075:2024 Information technology — Database languages — GQL, https://www.iso.org/standard/76120.html [retrieved Feb. 2025].
- [2] F. Laux, The Typed Graph Model, DBKDA 2020, ISBN: 978-1-61208-790-0, https://www.thinkmind.org/articles/dbkda_2020_1_30_50016.pdf
- [3] M. Crowe, The Pyrrho Book, ISBN: 978-1-903978-50-4 University of the West of Scotland 2015 https://myresearchspace.uws.ac.uk/ws/portalfiles/portal/56771820/2015_Crowe_Pyrrho_Book.pdf (retrieved February 2025)
- [4] Linked Data Benchmark Council, The LDBC Financial Benchmark v.0.1.0 https://arxiv.org/pdf/2306.15975 (retrieved Feb 2025)
- [5] M. Laiho, F. Laux, On SQL Concurrency Technologies, https://www.researchgate.net/publication/389127227_On_SQL_Concurrency_Technologies_-for_Application_Developers
- [6] F. Laux, The Typed Graph Model – a Supermodel for Model Management and Data Integration, International journal on advances in software, 2021, Vol. 14, Num. 1&2, ISSN: 1942-2628, https://www.thinkmind.org/articles/soft_v14_n12_2021_2.pdf
- [7] M. Crowe, F.Laux, Implementing the typed graph data model using relational database technology, International journal on advances in software, 2023, Vol. 16, Num. 3&4, ISSN:1942-2628, https://www.thinkmind.org/articles/soft_v16_n34_2023_6.pdf
- [8] M. Crowe, F. Laux, Implementing the draft Graph Query Language Standard : the Financial Benchmark, DBKDA 2024, ISBN: 978-1-68558-138-1, https://www.thinkmind.org/articles/dbkda_2024_1_60_50042.pdf
- [9] M. Crowe, F. Laux, Database technology evolution III: knowledge graphs and linked data, IARIA Congress 2024, ISBN: 978-1-68558-180-0, https://www.thinkmind.org/articles/iaria_congress_2024_2_130_50061.pdf