Database Technology Evolution II: Graph Database Language
MALCOLM CROWE, FRITZ LAUX
IARIA CONGRESS 2023
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 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).
Property graphs are useful

- Graph database products help in conceptual modeling
  - Mostly (e.g., Neo4j) node properties are not strongly typed
  - Model does not result in a corresponding relational database
  - A standard Graph Query Language is forthcoming
- We construct the relational database at the same time
  - Helps bridge the conceptual gap for app development
  - If models are in the database, all developers can agree
- Creating a Typed Graph Model by instances
  - Design by example: create the concepts incrementally
  - Repeating Patterns
- We believe it is useful to add the graphical support directly to the relational database management system

The Typed Graph Model
Property Graphs vs SQL

- The SQL programming model is well known
- Most organisations have an RDBMS so it avoids having a separate product and support team
- Enable SQL queries to process graph data
- Graph matching is often a useful shortcut
- The forthcoming GQL standard will be compatible with SQL/PGQ
- With a lot of work, we can do graphical things in SQL, but let’s make it easier..
An example: graph creation

```
[CREATE
 (Person {name:'Fred Smith'})
 <-[Child]-
 (Person {name:'Pete Smith'}),
 (a)-[Child]->
 (Person {name:'Mary Smith'})
 -[Child]->
 (Person {name:'Lee Smith'})]
```
Using MATCH on existing data

MATCH (P {name:'Pete Smith'})
  P.name='Peter Smith'
MATCH (P {name:'Mary Smith'})

MATCH and CREATE
MATCH and CREATE

This combination is used a lot!

```
MATCH
(a:Person {name:'Mary Smith'})
CREATE (a)-[:Child]->
(:Person {name:'Bill Smith'})
```

Repeating patterns
Repeating patterns

- We can get all descendants of Peter Smith by allowing iteration on the Child edge

MATCH ({name:'Peter Smith'}) [()-[:Child]->()]+( {name:D})]
Altering the Graph Schema

Improving the Child relation

SET ROLE PS

ALTER TABLE Child TO ChildOf
ALTER TABLE ChildOf ALTER LEAVING TO Parent
ALTER TABLE ChildOf ALTER ARRIVING TO Child
ALTER TABLE Person ADD PRIMARY KEY (name)
Visualizing the graph

A more real-life design
An Example ERP database

- A commercial enterprise buys parts and products
- Resells and/or creates products from parts (not working from raw materials)
Graph visualization
Conclusions

- This merging of TGM with relational technology allows graph-oriented data manipulation and queries.
- The forthcoming Graph Query Language standard GQL (ISO 39075) should be compatible with the ideas presented here.
References


