

# A Taxonomy of Time in Databases

Richard Snodgrass, Ilsoo Ahn

Presentation By: Kristen LeFevre

9/23/09

EECS 584, Fall 2008

1

## Motivation (1)

- Classical relational *data model* comprised of flat tables
  - Data updated in place
  - Old versions “discarded” when transactions commit
- Challenge 1: Incorporate temporal information into the relational data model, query language
- Challenge 2: Understand all of the different kinds of time we may want to represent

9/23/09

EECS 584, Fall 2008

2

## Motivation (2)

- Paper is an instance of an alternative / extended data model
- Moved this paper up for 2 reasons:
  - Auditing paper (next week) uses transaction time
  - Multiversion concurrency control (informal coverage)

9/23/09

EECS 584, Fall 2008

3

## Classical Relational Model (“Static Database”)

Faculty[1]		Faculty[2]	
Name	Rank	Name	Rank
Merrie	Associate	Merrie	Full
Tom	Associate	Tom	Associate

Promote Merrie

A state of the database reflects data currently in the database, which may or may not reflect reality.

- Does not support queries/updates based on time:
1. What was Merrie’s rank 2 years ago? (*historical query*)
  2. How did the number of faculty change over the last 5 years? (*trend analysis*)
  3. Merrie was promoted to Full starting last month. (*retroactive change*)
  4. James is joining the faculty starting next month. (*postactive change*)

9/23/09

EECS 584, Fall 2008

4

## Static Rollback Databases

Store all past states of the database, indexed by time

name	rank	transaction time	
		(start)	(end)
Merrie	associate	08/25/77	12/15/82
Merrie	full	12/15/82	∞
Tom	associate	12/07/82	∞
Mike	assistant	01/10/83	09/25/84

**Example:**  
Find Merrie’s Rank as it was recorded in the database on 12/10/82

**Answer:**

rank	full
------	------

A static DB

- **Transaction time** indicates when the tuple was physically changed in the database.
- **Disadvantage:** Transaction time might not reflect when the value changed in the real world.

9/23/09

EECS 584, Fall 2008

5

## Historical Databases

Store the real-world history, as best as it is known.

name	rank	valid time	
		(from)	(to)
Merrie	associate	09/01/77	12/01/82
Merrie	full	12/01/82	∞
Tom	associate	12/05/82	∞
Mike	assistant	01/01/83	03/01/84

**Example:**  
Find Merrie’s rank when Tom arrived.

**Answer:**

rank	full
------	------

Also a historical DB

- **Valid time** indicates the time from the real world, which is being modeled by the database.

9/23/09

EECS 584, Fall 2008

6

## Observations

- Two kinds of time:
  - *Transaction time*
    - Reflects when the DB was physically changed
    - Can never be updated
  - *Valid time*
    - Reflects our understanding of the real world
    - Can be updated
- Both serve a unique purpose
- Can we use them simultaneously?
  - “Temporal” Database

9/23/09

EECS 584, Fall 2008

7

## Temporal Database

name	rank	valid time		transaction time	
		(from)	(to)	(start)	(end)
Merrie	associate	09/01/77	∞	08/25/77	12/15/82
Merrie	associate	09/01/77	12/01/82	12/15/82	∞
Merrie	full	12/01/82	∞	12/15/82	∞
Tom	full	12/05/82	∞	12/01/82	12/07/82
Tom	associate	12/05/82	∞	12/07/82	∞
Mike	assistant	01/01/83	∞	01/10/83	02/25/84
Mike	assistant	01/01/83	03/01/84	02/25/84	∞

- Merrie started working on 09/01/77
  - Information entered in the DB on 08/25/77
- Merrie promoted as of 12/01/82
  - Information not entered until 12/15/82
- Tom entered into DB as a full prof on 12/05/82
  - Later fixed to indicate he was only an associate (12/07/82)

9/23/09

EECS 584, Fall 2008

8

## Temporal Database

name	rank	valid time		transaction time	
		(from)	(to)	(start)	(end)
Merrie	associate	09/01/77	∞	08/25/77	12/15/82
Merrie	associate	09/01/77	12/01/82	12/15/82	∞
Merrie	full	12/01/82	∞	12/15/82	∞
Tom	full	12/05/82	∞	12/01/82	12/07/82
Tom	associate	12/05/82	∞	12/07/82	∞
Mike	assistant	01/01/83	∞	01/10/83	02/25/84
Mike	assistant	01/01/83	03/01/84	02/25/84	∞

**Example:** Find Merrie's rank when Tom arrived, according to the state of the DB on 12/10/82

**Answer:**

rank	valid time		transaction time	
	(from)	(to)	(start)	(end)
associate	09/01/77	∞	08/25/77	12/15/82

Also a temporal DB

9/23/09

EECS 584, Fall 2008

9

## User-Defined Time

- Necessary when additional temporal information required (not handled by transaction or valid time)
  - E.g., For promotions, effective date shown on the promotion letter
  - Valid time is the date when letter signed
  - Transaction time when the DB updated

9/23/09

EECS 584, Fall 2008

10

## User-Defined Time

name	rank	effective date	valid time (at)		transaction time (start) (end)	
			(start)	(end)	(start)	(end)
Merrie	associate	09/01/77	08/25/77	∞	08/25/77	∞
Merrie	full	12/01/82	12/11/82	∞	12/15/82	∞
Tom	full	12/05/82	12/05/82	12/01/82	12/07/82	∞
Tom	associate	12/05/82	12/07/82	∞	12/07/82	∞
Mike	assistant	01/01/83	01/01/83	01/10/83	∞	∞
Mike	left	03/01/84	02/25/84	∞	∞	∞

Merrie promoted to full professor as of 12/11/82

Promotion letter dated 12/01/82

DB actually updated (retroactively) 12/15/82

9/23/09

EECS 584, Fall 2008

11

## Key Points

- Time is an important issue not handled by traditional relational model
- Time is subtle
  - Should it reflect reality (*valid time*) or the database's representation (*transaction time*)?
- Paper argues that it is important to support both, plus user-defined time
  - Defines *temporal database*
  - Essentially, a new data model

9/23/09

EECS 584, Fall 2008

12

## Temporal Data Model Summary

	No Rollback	Rollback
Static Queries	Static	Static Rollback
Historical Queries	Historical	Temporal

Figure 10 Types of Databases

	Transaction	Valid	User-defined
Static	✓		
Static Rollback		✓	
Historical	✓	✓	✓
Temporal			✓

Figure 11 Attributes of the New Kinds of Databases

9/23/09

EECS 584, Fall 2008

13

## Implementing a Transaction-Time Database

(Brief overview to help in reading next paper)

Content borrowed from  
Jensen et al., "Incremental Implementation Model for  
Relational Databases with Transaction Time" TKDE 1991

9/23/09

EECS 584, Fall 2008

14

## High-Level Overview

- The time-oriented data models are great, but how do we implement them?
  - Focus on implementing the static rollback database (supporting transaction time only)
- Propose the *backlog database* implementation model

9/23/09

EECS 584, Fall 2008

15

## Backlog DB Representation

- *Backlog* contains the complete history of change requests to each base relation R
- Records transaction time

name	rank	transaction time	
		(start)	(end)
Merrie	associate	08/25/77	12/15/82
Merrie	full	12/15/82	∞
Tom	associate	12/07/82	∞
Mike	assistant	01/10/83	02/25/84

Op	Time	Name	Rank
Ins	08/25/77	Merrie	associate
Ins	12/07/82	Tom	associate
Mod	12/15/82	Merrie	full
Ins	1/10/83	Mike	assistant

Ins, Mod, Del

Can retrieve time sliced base relations from the backlog (e.g., the relation at some point in time)

9/23/09

EECS 584, Fall 2008

16

## Query Language, Implementation Considerations

- One additional operator (beyond relational algebra)
  - Slice(R, t); R is a base relation, t is a time interval
- Time-slices of base relations can be either stored (materialized) or re-computed when needed
  - Depends on query / update ratio, and available space
  - Various optimizations for materialized time-slices (e.g., eager or lazy updates)

9/23/09

EECS 584, Fall 2008

15

## Time-Oriented DBMSs

- Many systems support some time-oriented features
  - Oracle flashback (static rollback database)
  - Postgres
  - Others?
- Often combined with multiversion concurrency control (MVCC)
  - Will revisit informally

9/23/09

EECS 584, Fall 2008

15

9/23/09

EECS 584, Fall 2008

16



## Discussion Questions

- Are there interesting applications for (the various permutations of) time-oriented databases?
  - Historical
  - Static rollback
  - Temporal