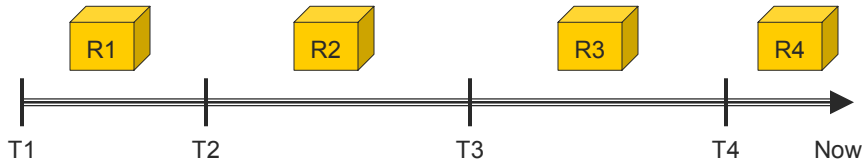
Temporal Relational Databases: The Final Chapter

Words Delivered by James Terwilliger

Today's Menu

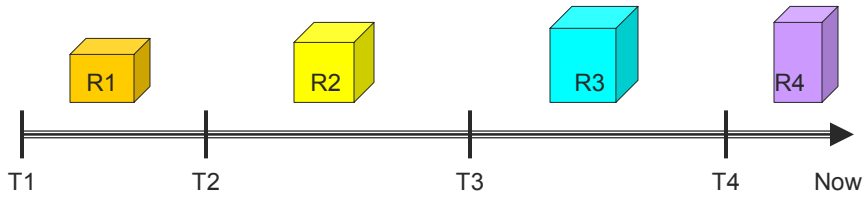
- Schema Evolution
- Temporal Operations in Modern DBMS' and Research

Fixed Schema, Evolving Data



Models a time-enabled relation as a sequence of relations separated by transaction commitments **over a constant relation schema**

Schema Evolution (Multi-temporal)



Models a time-enabled dynamic relation as a sequence of relations **and relation schemas** separated by transaction commitments

Schema Evolution, Formalism

<(R1, d1), (R2, d2), (R3, d3), … (Rk, dk)>

Each pair (Ri, di) represents a "version", a relation schema and a snapshot of the extent of the relation

Consider two time points, i and i+1:

If Ri = R(i+1) and di = d(i+1), there is no change

If Ri = R(i+1) and di ≠ d(i+1), DML occurred

If Ri ≠ R(i+1), DDL occurred

DDL Possibilities

- Change domain of an existing attribute
- Rename an attribute
- Add a new attribute
- Remove an existing attribute
- Add a new empty relation
- Remove an existing relation
- *Add/remove temporal column*

Several Issues with Evolution

- Schema Update
- Storage
- Query

Schema Evolution: Schema Updates

- Schema updates can only add information
  - Effectively, the schema of an evolving relation is the union of all columns ever seen by the relation in its lifetime
  - Called a “completed schema”
- May want to delete columns/add columns with certain timespans
  - Delete a column, effective last week
  - Add a column for only a limited time

Schema Evolution: DDL

- Add a new attribute
  - Create new attribute with interval [addTime, now]
- Remove an existing attribute
  - Give the attribute's interval an end time
- Add a new empty relation
  - Create new relation with interval [addTime, now]
- Remove an existing relation
  - Give the relation's interval an end time

Schema Evolution: DDL, part deux

- Change domain of an existing attribute
  - Create new attribute with old values translated into new domain
  - Give new attribute interval [addTime, now]
  - Give old attribute an end time
- Rename an attribute
  - Same as above
- These are interesting because it is the same process as a DML update
  - Propagate values to new structure
  - Mark old and new structures with appropriate intervals

Schema Evolution: DDL Example

- Start with Employee [1994, now]:
  - Name:String [1994, now]
  - Salary:Float [1994, now]
  - Dept:String [1998, now]
  - T: DateTime Interval [1994, now]
- ADD COLUMN Degree:String
- DROP COLUMN Dept
- ALTER COLUMN Salary ALTER DOMAIN Integer
- End with Employee [1994, now]
  - Name:String [1994, now]
  - Salary:Float [1994, 2006]
  - Salary:Int [2006, now]
  - Dept:String [1998, 2006]
  - Degree:String [2006, now]
  - T: DateTime Interval [1994, now]

Schema Evolution: Storage

- Standard relational storage counts on having a static schema for efficiency
- Think “union-compatibility” for tuples
- One possible option: store each schema version in its own storage
  - As if each is its own table
  - Called “multi-pool”
  - Query processor must pull from all tables

Schema Evolution: Single-Pool Storage

- Keep all tuples in the same storage pool
- Assign to each column a timestamp interval, just like each tuple
- Assign to each relation a timestamp interval as well

Schema Evolution: Queries and Query Processing

SELECT * FROM Employee
SELECT Dept FROM Employee

- Which columns are returned?
- If Dept was deleted, would this query cause a compile/syntax error?
- If the domain of Dept was ever changed, which domain would the query processor consider?

Query Processing, Continued

- In a multi-pool architecture, query processor must know from which relations to draw data
- One possibility: a “meta-relation”
  - Table of tables – all table versions that correspond with one table stored in one place

More Query Processing

- Which schema should we use?
  - Could use completed schema
  - Another option: allow user to choose schema in query
- TSQL2 gives the user an option to set the schema time

SET SCHEMA VALID 2006
SELECT * FROM Employee

Your Time Has Not Been Wasted

- Temporal extensions have been made to some major DBMS'
- Some databases are built from the ground up to accommodate time
- Active research areas also still exist
- But…

SQL:2003

- Apparently, Part 7 (SQL/Temporal) was withdrawn from the standard prior to publication
  - Part 7 was recommended unanimously, but the ISO board could not come to a consensus about it
- Implications?
  - Do implementations follow standards?

Oracle 9g, 10g

- Technology called “flashback” queries and databases
  - Uses system log to simulate rollback relations
- New product in version 10g called “Workspace Manager”
  - Adds valid and transaction times to relations
  - Some operations, such as enabling/disabling versions, are done through stored procedures
  - Not TSQL2

```
SELECT * FROM employees e WHERE WM_CONTAINS
    (e.wm_valid, WMSYS.WM_PERIOD(TO_DATE('01-01-
    1991', 'MM-DD-YYYY'), TO_DATE('01-02-1991', 'MM-DD-
    YYYY')) = 1;
```

Microsoft SQL Server: ImmortalDB

- Currently research-only
- Instead of giving tuples an end time stamp…
  - Creates chains of tuples
  - An updated tuple is added to the end of a chain, effectively “deleting” the previous version
  - A delete operation adds a “delete stub”
- Focus is on “as of” queries

```
Begin Tran AS OF "8/12/2004 14:15:20“
SELECT * FROM MovingObjects
WHERE Oid < 11 Commit Tran
```

Leverage Existing DBMS Using Outside Product

- Leverage transaction logs
- LogExplorer from Lumigent over SQL Server
- IBM DataPropagator over DB2
- These products replicate the logs so that, even when the DBMS cleans out the log, that information is still around

TimeDB

- DBMS built from the ground up in Java to support temporal constructs
- http://www.timeconsult.com/Software/Software.html
- Uses a flavor of SQL that borrows ideas from TSQL2, but with different syntax

MADS - Modeling of Application Data with Spatio-temporal features

- Start with a normal database
- Add spatial and temporal awareness
- Add multiple-representation capability

- Tool (MADSTRA) translates a MADS conceptual model into a logical model on an existing DBMS or GIS system

The Punchline

- All major DBMS vendors have some product or extension that provides temporal capabilities
- None of them use TSQL2 directly
- All of them employ the concepts that we've seen, and the constructs from TSQL2, if not the exact syntax
- None of them are as clean as TSQL2

- TSQL2 Reference:
  - http://portal.acm.org/citation.cfm?id=187436.187449