PRIMA - Transaction Time Database Systems

Seminar: Techniques for Non-Traditional Data Management

Jörg Schad

Universität des Saarlandes

02.12.2008

Outline

Transaction Time Database Systems

What are Transaction Time Database Systems?

Transaction Time Database Systems retain and provide access to the actual state but also to all prior states of a database.
This history should be created and maintained transparently for the user.

Motivation

Motivation

- Often we are not only interested in the actual information but also a trace of previous information
- Example: Version history in Wikipedia

Other Benefits

- Rollback to old database versions
- Auditing
- Tracebility of actions

Example: Snapshot Isolation

Snapshot Isolation

- Each transaction works on consistent snapshot of database
- Transaction can only commit if no prior conflicting updates
- Better performance than serializability
- Still some anomalies (not serializable)

Can be easily built on top of Transaction Time Databases as snapshots are natively provided.

Relational Approaches

How could this be represented in a relational Database System?

empno	name	title	depno	ts	te
1001	Bob	Engineer	d01	2004-01-01	2005-09-30
1001	Bob	Sr. Engineer	d02	2005-10-01	2007-02-28
1001	Bob	Tech. Lead	d02	2005-03-01	now

Table: Relational Snapshot History

Relational Approaches

How could this be represented in a relational Database System?

empno	name	title	depno	ts	te
1001	Bob	Engineer	d01	2004-01-01	2005-09-30
1001	Bob	Sr. Engineer	d02	2005-10-01	2007-02-28
1001	Bob	Tech. Lead	d02	2005-03-01	now

Table: Relational Snapshot History

Problems ([WZZ95])

- Redundant information
- Coalescing with temporal queries

Visual Representation

Visual Representation of Temporal Data

id	name	title	depno
2004-01-01	2004-01-01	2004-01-01 **Engineer** 2005-09-30	2004-01-01 **d01** 2005-09-30
1001	**Bob**	2005-10-01 **Sr. Engineer** 2007-02-28	2005-10-01 **d02**
now	now	2007-03-01 **Tech Lead** now	now

Figure: Temporally grouped history

XML Representation

VDocument representation of data versions

```
<employees tstart="2004-01-01" tend="now">
  <employee tstart="2004-01-" tend="now">
    <id tstart="2004-01-01" tend="now">1001</id>
    <name tstart="2004-01-01" tend="now"> Bob</name>
    <title tstart="2004-01-01" tend="2005-09-30"> Engineer </title>
    <title tstart="2005-10-01" tend="2007-02-28"> Sr. Engineer </title>
    <title tstart="2007-03-01" tend="now"> Tech Lead </title>
    <deptno tstart="2004-01-01" tend="2005-09-30">d01</deptno>
    <deptno tstart="2005-10-01" tend="now">d02</deptno>
  </employee>
</employees>
```

Hierachical structure can be represented naturally using XML!

Quering VDocuments

How to query VDocuments?

- XML queries can be naturally expressed using XQuery
- Powerful language on based on XPath

Simple XQuery returning the department history of Bob

```
for $t in doc("employees.xml")/employees/employee[name="Bob"]/deptno
return $t
```

Complex XQuery

XQuery returning all Sr. Engineers before 2005

```
for $e in doc("employee.xml")/employees/employee
    let $m := $e/title[.="Sr Engineer"and tend(.)= 2005-01-01]
where not empty ($e) and not empty ($m)
return <employee> {$e/id, $e/name} </employee>
```

Example of typical XQuery FL(O)WR expression

- **F**or
- **L**et
- **O**rder by [1]
- **W**here
- **R**eturn

[1] Not actually used here!

Transaction Time Database Systems: Immortal DB

Immortal DB [LBM+05]

- Transaction Time Database built into MS SQL Server
- Implemented in the database engine itself (not middleware)
- Indexing over time
- Microsoft Research Project

Transaction Time Database Systems 2: ArchIS

ArchIS ([WZZ95])

- **Arch**ivial **I**nformation **S**ystem
- General architecture on top of DBMS (Middleware)

Implementations

- DB2 (V7.2) $\Rightarrow$ object relational mapping
- Tamino XML Server $\Rightarrow$ native XML database
- ATLas $\Rightarrow$ object relational mapping

Best performance naturally on native XML database [WZZ95].

Problem

What if not only the data changes but also the database schema?

Problem

What if not only the data changes but also the database schema?

- Cannot be expressed by VDocument
- Difficult to store in relational tables as tables usually instantiates a schema

Need for Schema Evolution

Need for schema evolution

- Reorganization (performance, design purposes,...)
- Requirement changes (auditing ,...)
- Additional data (Create Table departmentHeads;)
- Removing data (privacy issues)

Example for Schema Evolution

Example for schema Evolution: Wikipedia

- Wikimedia is the system behind Wikipedia
- Wikimedia Database: over 170 schema changes in four years
- Schema Evolution Benchmark also provides more details ([CMTZ08])

Example: Blocking user in Wikimedia

Alter table users add column blocked (boolean);

Wikipedia's Schema Evolution

Example: Number of tables in Wikimedia Database

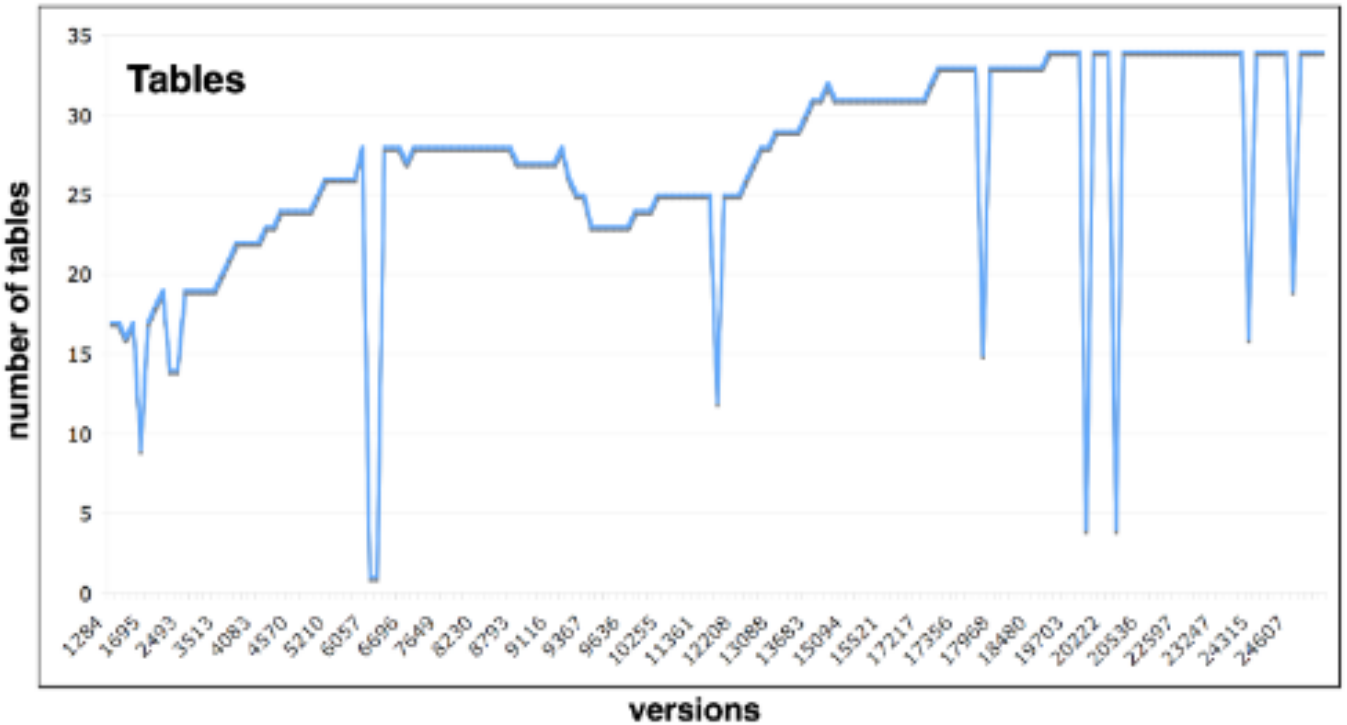


Figure: Number of tables in Wikimedia adapted from [CMTZ08]

Schema Modification Operators

How can one modify the schema?

Table altering SMOs

Create table t;
Drop table t;
Rename table t into t1;
Distribute table t into t1 with cond1, t2 with cond2;
Merge table t1,t1 into t;

Column altering SMOs

Add column c As f(c1,c1,...) into t;
Drop column c from t;
Rename column c in t to c*;
Copy column c from t1 into t2 where cond1;
Move column c from t1 into t2 where cond1;

Schema Evolution Example

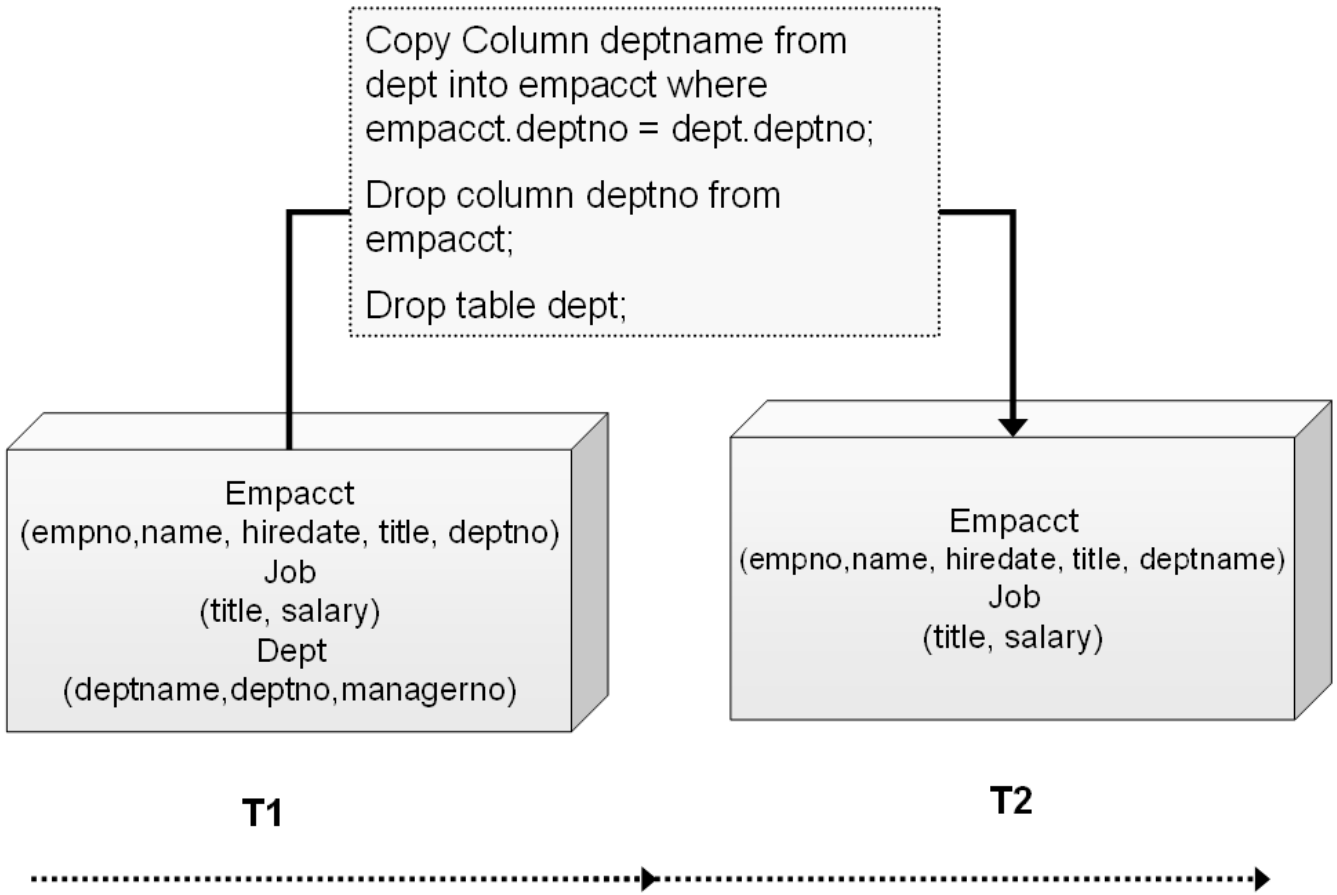


Figure: Schema changes in example database adapted from

XML Representation

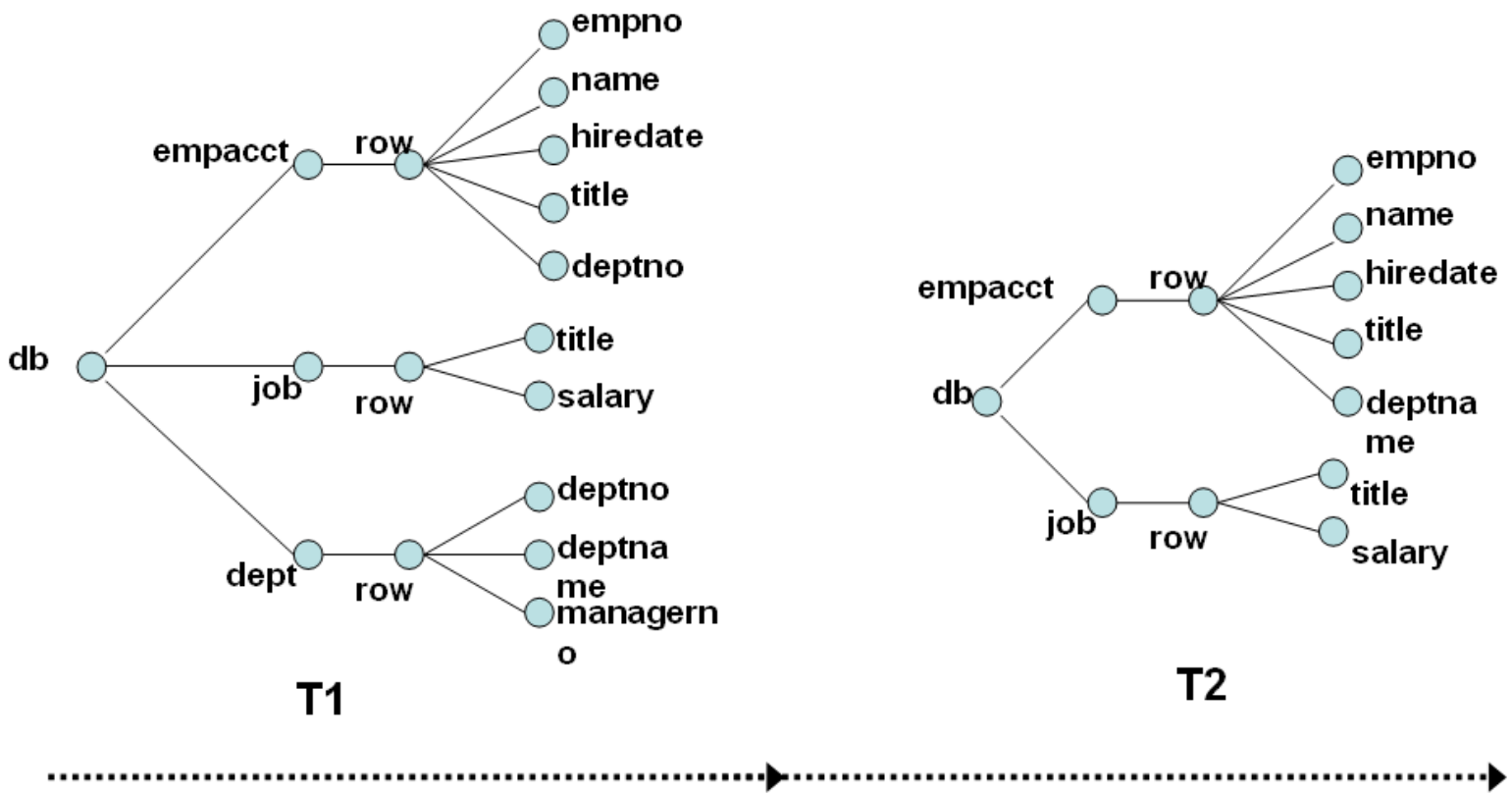


Figure: XML tree representation of schemes

XML Representation(2)

```
<db ts=T1 te=now>
    <dept ts =t1 te= t2>
        <row ts=t1 te=t2>
            <deptno ts=t1 te=t2>d01</deptno>
            <deptname ts=1 te=t2>Research</deptname>
        </row>
    </dept>
        <empacct ts=t1 te=now>
            <row ts=t2 te=now>
                <epno ts=t1 te=now>10925</epno>
                <title ts=t1 te=t2>Engineer</title>
                <title ts=t2 te=now>Senior Engineer</title>
                <deptname ts=2 te=now>Research</deptname>
                <deptno  ts=t1 te=t2> d01</deptno>
                ....
            </row>
        </empacct>
    </db>
```

MV Document

This representation is called Multi Version Document as an extension to normal VDocuments.

XQuery

First attempt to retrieve a list of department names:

```
for $t in doc("emp.xml")/db/empacct/row[name="Bob"]/deptname
return $t
```

XQuery

First attempt to retrieve a list of department names:

```
for $t in doc("emp.xml")/db/empacct/row[name="Bob"]/deptname
return $t
```

Problem?

This query only considers the current schema version!

Query Rewriting

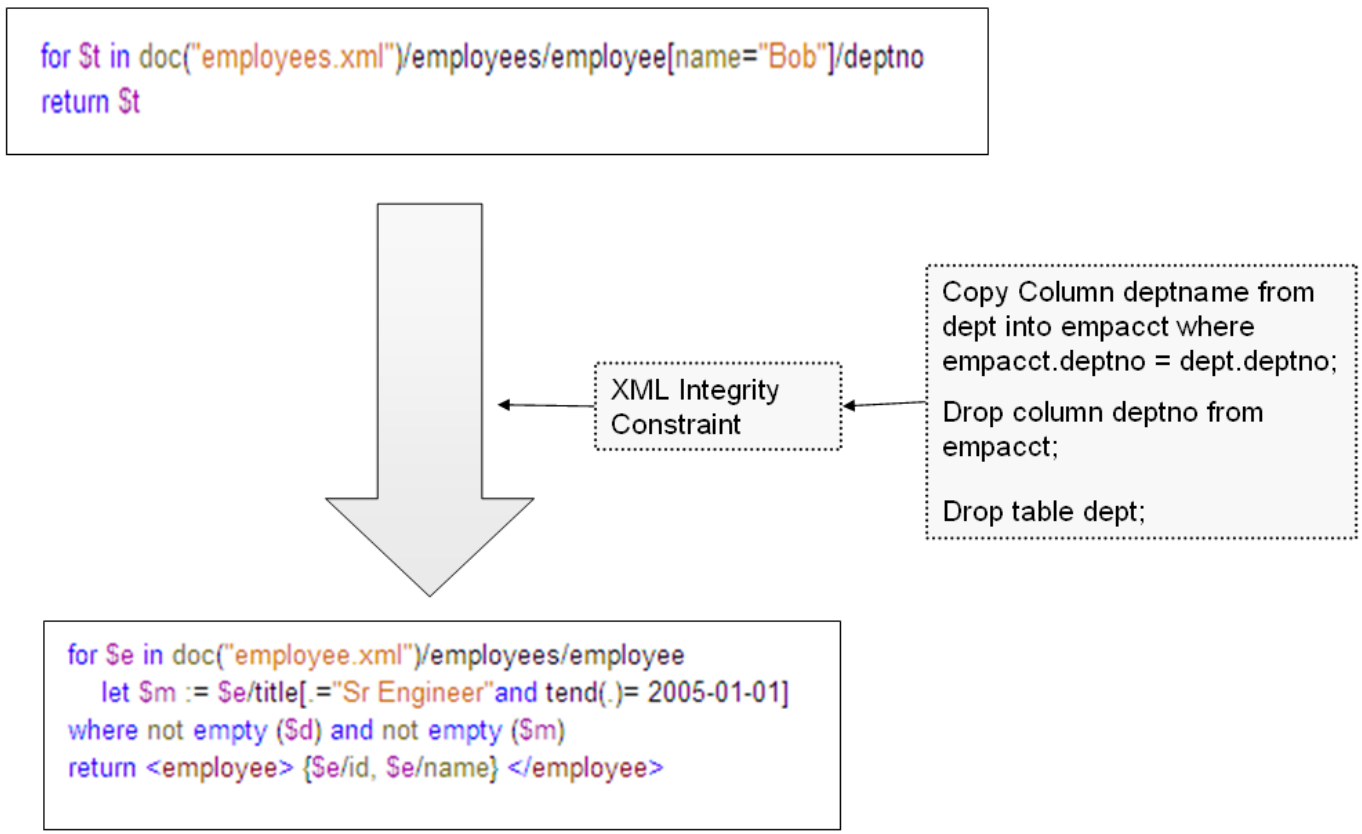
Correct XQuery considering all versions

```
for $t in doc("emp.xml")/db/empacct/row[name="Bob"]/deptname
return $t
union
for $s in doc("emp.xml")/db/empacct/row[name="Bob"]/deptno
    return { for $p in doc("emp.xml")/db/dept
          where $s/custno = $p/deptno
            return $p/deptname }
```

This translation should be performed transparently to the user!

Query Rewriting(2)

Query Rewriting Architecture



Query Rewriting(3)

Query Rewriting to different versions is not trivial!
Consider Query Taxonomy for different Query Types

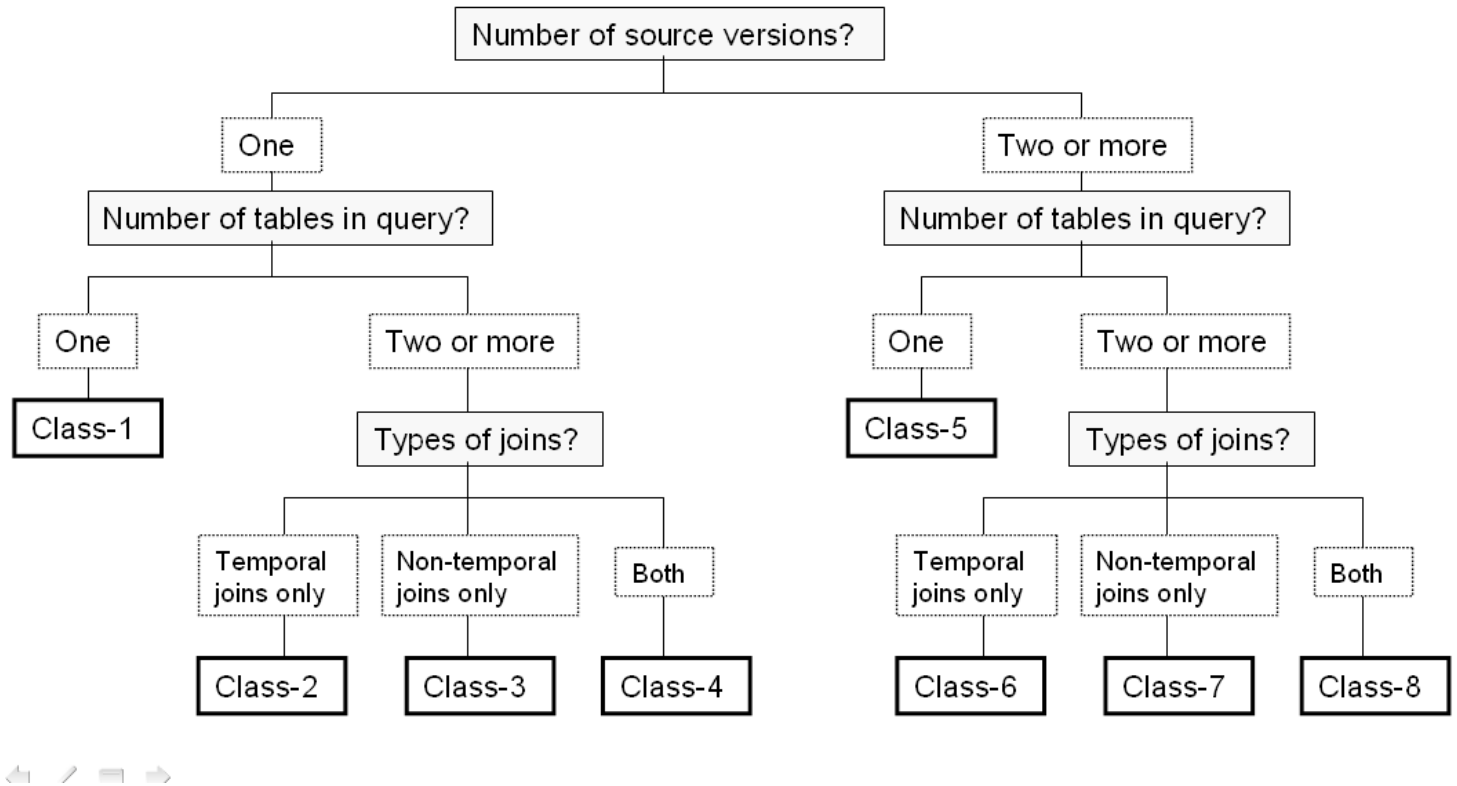


Figure: Query Taxonomy adapted from [MCD+08]

Query Rewriting(3)

Query Rewriting to different versions is not trivial!
Consider Query Taxonomy for different Query Types

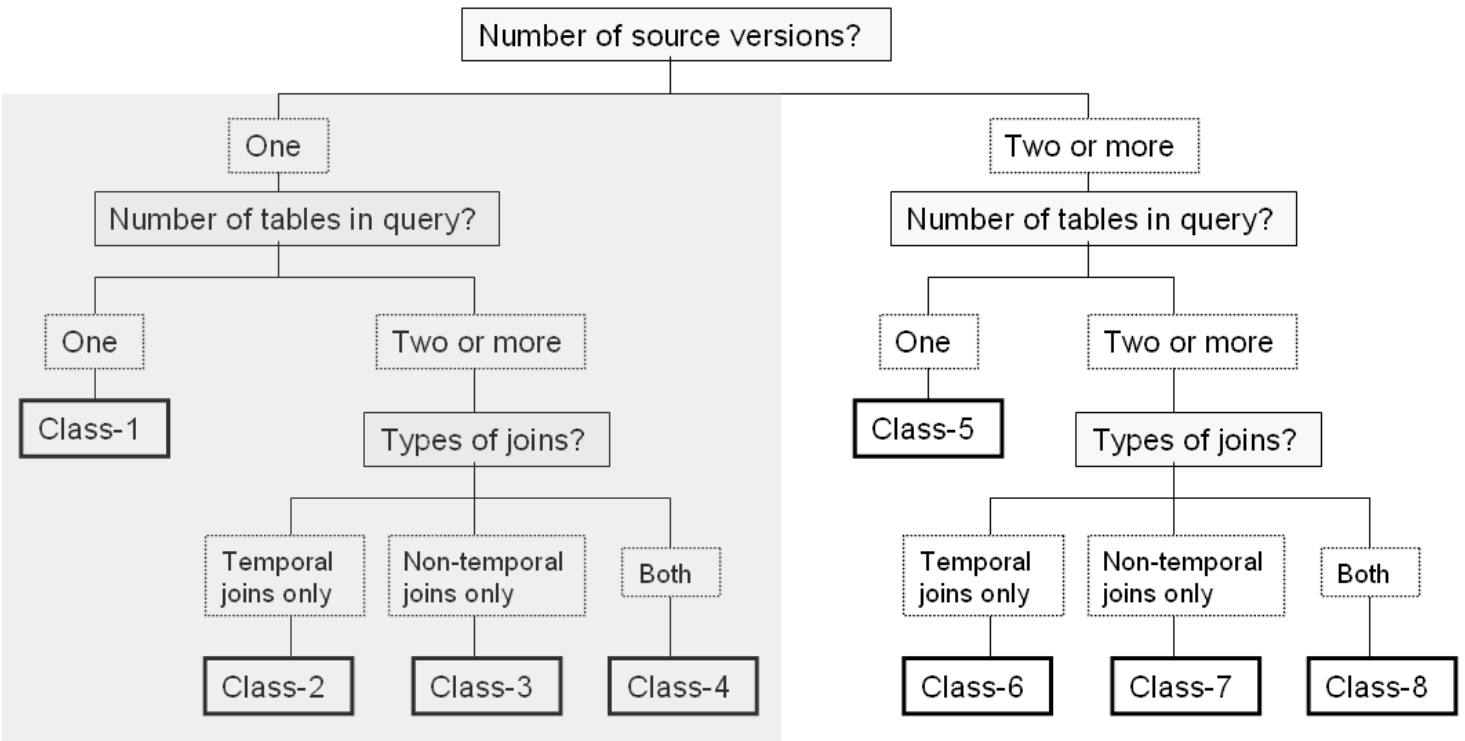


Figure: Query Taxonomy adapted from [MCD+08]

MinSourceFind

In order to identify the corresponding query class and make queries efficient

MinSourceFind

Identify minimal set of source versions involved in given query and prune storage partions accordingly.

Reduces rewriting effort and queried XML data

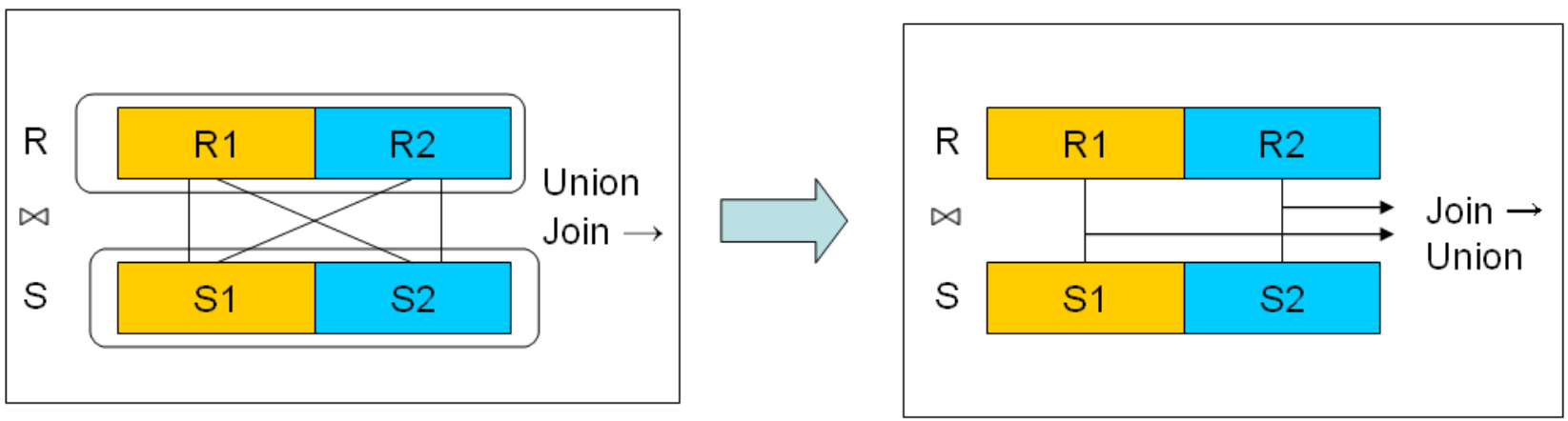
$\Rightarrow$ **enhances performance**

TJoinFind

In order to identify the corresponding query class and make queries efficient

TJoinFind

Identify temporal joins in given query.
A temporal join acts onto two temporally overlapped node set (tables).



Further Optimizations

Optimizations for improving efficiency of Queries

SMO pruning

Prune SMOs that only change columns not involved in query
$\Rightarrow$ reduces the number of schema versions.

SMO compression

Compress several SMOs into a single one (example renaming of three different tables)
$\Rightarrow$ again reduces the number of schema versions.

PRIMA

PRIMA

- Panta Rhei Information Management and Archival
- Transactional Time DBMS.
- First based on XML DB (native XML DB)
- Now based on H-Prima (Relational Database)

Architecture

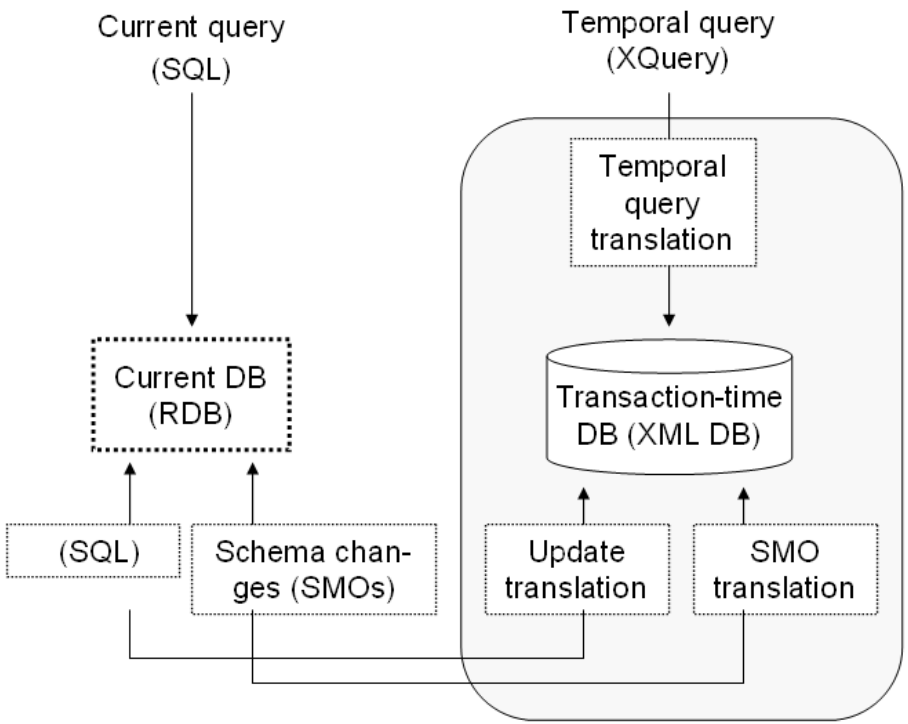


Figure: PRIMA architecture adapted from [MCD+08]

Performance Study

Does PRIMA scale to larger applications?

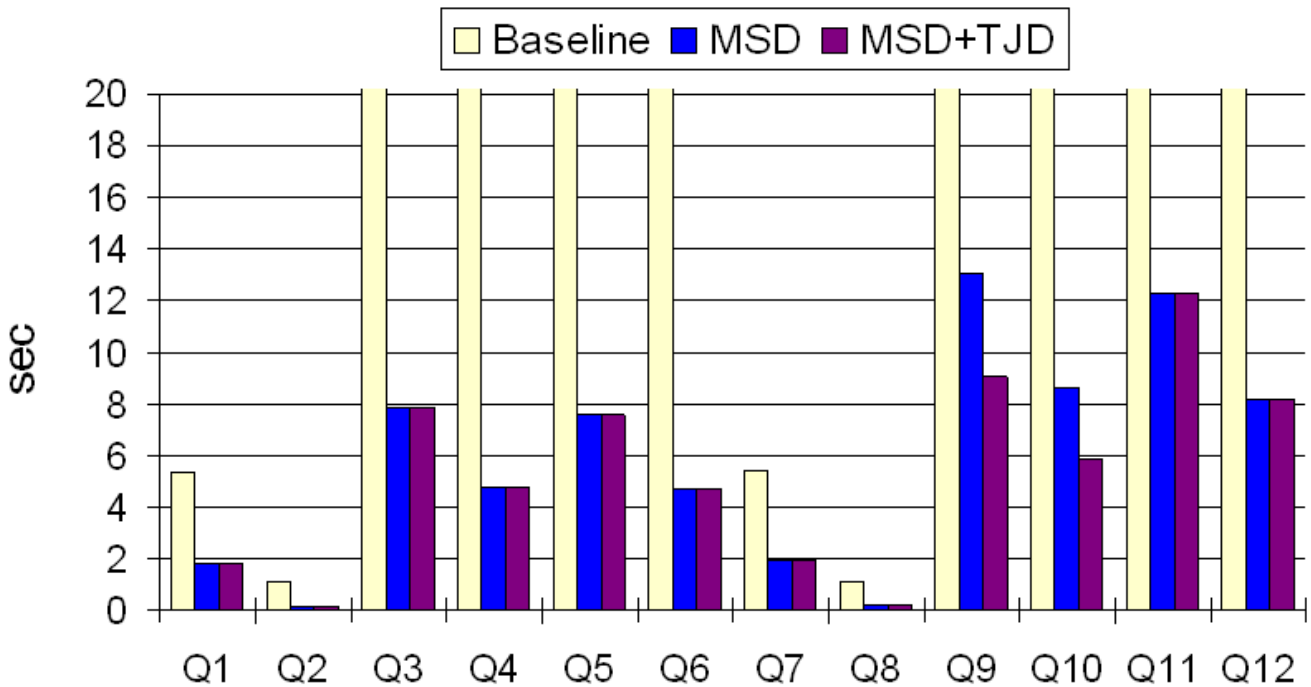


Figure: Performance study adapted from [MCD+08]

Usability Study

But increased usability:

```
for $t in doc("employees.xml")/employees/employee[name="Bob"]/deptno
return $t
```

VS

```
for $t in doc("emp.xml")/db/empacct/row[name="Bob"]/deptname
return $t
union
for $s in doc("emp.xml")/db/empacct/row[name="Bob"]/deptno
   return { for $p in doc("emp.xml")/db/dept
          where $s/custno = $p/deptno
            return $p/deptname }
union
.....
union
.....
union
```

Figure: Usability increase

Future of Prima

Future of PRIMA

- Exploit advances in XML Database development (e.g. Hybrid DBMS)
- Exploit advances in mapping tools

$\Rightarrow$ **make PRIMA better scalable!**

Issues

Issues with PRIMA

- Implementation for XML on top of relational DBMS
- Consolidate architecture into one system (Hybrid DBMS)
- Performance does not scale well (yet)

Conclusion

PRIMA is..

- A fully functional Transaction Time Database System
- Supporting schema changes to the underlying database system
- Makes use of XML to represent temporal information and schema evolution

PRIMA is not..

- Performance efficient due to rewriting of queries and XQuery engine
- Not yet usable for large applications

Conclusion

Thank you for your attention!
Questions?

Literatur I


CA Curino, HJ Moon, L. Tanca, and C. Zaniolo.
Schema Evolution in Wikipedia: toward a Web Information System Benchmark.
ICEIS, 2008.


D. Lomet, R. Barga, M.F. Mokbel, G. Shegalov, R. Wang, and Y. Zhu.
Immortal DB: transaction time support for SQL server.
In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 939–941. ACM New York, NY, USA, 2005.

Literatur II


H.J. Moon, C.A. Curino, A. Deutsch, C.Y. Hou, and C. Zaniolo.
Managing and querying transaction-time databases under schema evolution.
Proceedings of the VLDB Endowment archive, 1(1):882–895, 2008.


F. Wang, X. Zhou, and C. Zaniolo.
Using XML to Build Efficient Transaction-Time Temporal Database Systems on Relational Databases.
Engineer, 1995:05–01, 1995.

XML Integrety Constraints

How do we generate differnt mappings from SMOs?

Translate SMOs into XML Integrity Constraints

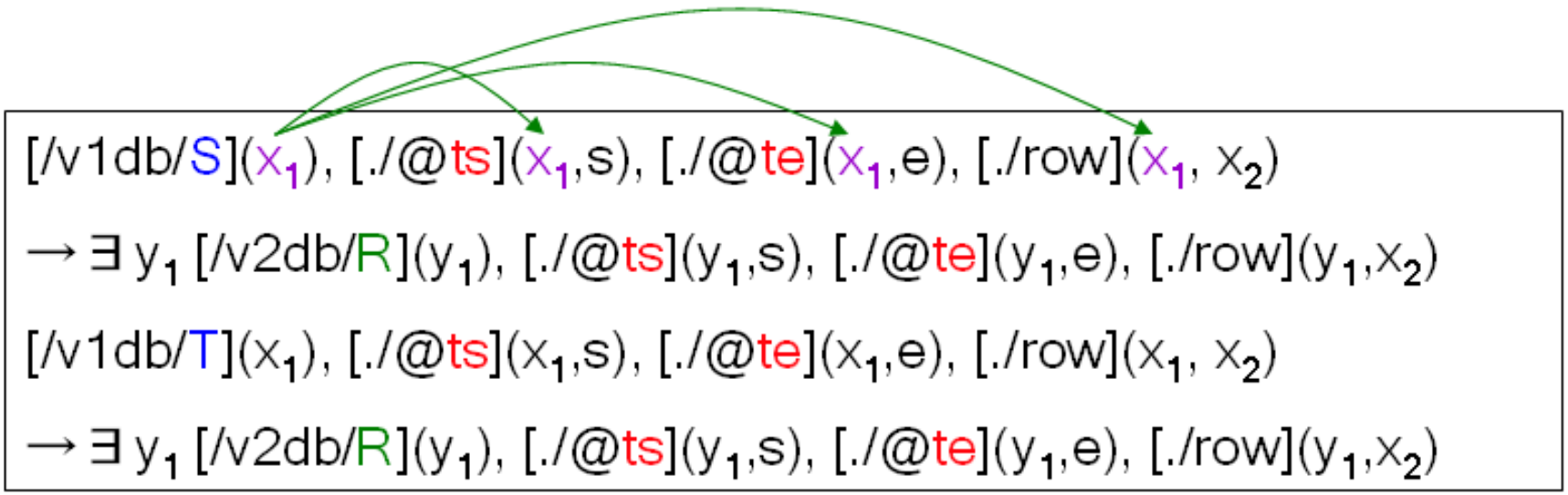


Figure: Xml Integrity Constraint adapted from [MCD+08]