Temporal Databases: state-of-the-art and applications

Alejandro Vaisman

Universidad de Buenos Aires

Departamento Computación

avaisman@dc.uba.ar

Outline

- Motivation
- Temporal Relational Databases
  - Temporal data models
  - Abstract temporal databases
  - Concrete temporal databases
  - TSQL2
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- Summary

Outline

- **Motivation**
- Temporal Relational Databases
  - Temporal data models
  - Abstract temporal databases
  - Concrete temporal databases
  - TSQL2
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- Summary

Temporal Databases

- Temporal database: a repository of temporal information
- Intuitively, a database that maintains past, present and future data, *i.e.* a database with a *time* dimension
- Many definitions proposed in the literature

Temporal Databases: Applications

- Applications
  - GIS : land use over time
  - Data Warehousing : historical trends for decision support
  - Inventory over time for time series analysis
  - Legal records : validity periods for laws

Temporal Databases: Applications

- Difficult to find applications that do not involve the management of temporal data.
- These applications could benefit from built-in temporal support in a DBMS, instead of ad-hoc temporal management.
  - More efficient application development
  - *Potential* increase of performance
  - Systems closer to the real world situation
- A TDBMS provides built-in support for recording and querying time-varying information.

Temporal Databases : a History

- Pioneers: Jacob Ben-Zvi (1982) and James Clifford (1983), independently from each other.

- Ben Zvi's doctoral disertation proposed the *Time Relational Model* (including a non-1NF data model, query language, storage, indexing, concurrency). Defined *effective* and *registration* times (now *valid* and *transaction* times). Many ideas borrowed from his work (maybe without knowing this fact).

- Clifford and Warren gave a first formal treatment of time using logic, and devised a query language.

- Snodgrass (1984) proposed TQUEL, extending QUEL.

Temporal Databases : a History (cont.)

- Other 1NF proposals (tuple-timestamping):
  - Navathe & Ahmed (1987-89, *Temporal Relational Model*, and the concept of Temporal Normalization)
  - Lorentzos (1988, *Interval-Extended Relational Model*)
  - Sarda (1990, HSQL)
- Attribute timestamping (non-1NF models)
  - Gadia (1985), views attributes as functions of time.
  - Clifford & Crooker (1985-87), defined the lifespan for attributes and a lifespan for tuples.
  - Tansel (1986) also timestamped attributes

Temporal Databases : a History (cont.)

- A standard: TSQL2
- Proposed by a committed of 18 researchers(Snodgrass *et al*, 1995)
- Consolidates approaches of previous proposals
- Extends SQL-92 (the standard query language at that time)
- Incorporated in SQL:1999 (and SQL 2003, the current standard)

Temporal Databases : a History (cont.)

- Time added to many data models
  - Entity-relationship, and semantic data models in general
  - Object – Oriented data models
  - Knowledge-based data models
  - Multidimensional data models
  - XML, RDF, databases for the WWW

Outline

- Motivation
- **Temporal Relational Databases**
  - **Temporal Data Models**
  - Abstract temporal databases
  - Concrete temporal databases
  - TSQL2
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- Summary

The Structure of Time

First thing to do: settle on an specific temporal domain, in particular, to define a Temporal **Ontology:** points vs intervals.

In databases: point-based view is predominant

In AI: interval- based view

A temporal domain is defined as a structure: **N = (**N, <) – naturals, **Z = (**Z, <) -integers, **Q = (**Q, <) - rationals, **R = (**R, <) –reals.

There is also a domain *D* of uninterpreted constants.

Other properties also important: density, granularities

The Structure of Time (cont.)

- Time Density:
  - *Discrete*: time isomorphic to the integers – the simplest option for databases
  - *Dense* : time isomorphic to rational numbers
  - *Continuous* : time isomorphic to real numbers

Multiple Temporal Dimensions

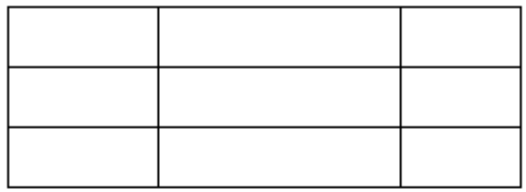
- *Valid time* of a fact: when the fact is true in the modeled reality
- *Transaction Time* of a fact: when the fact is current in the database and can be retrieved
- *Four kind of tables:*
  - Snapshot
  - Valid-time
  - Transaction-time
  - Bitemporal

Multiple Temporal Dimensions (cont.)

Example: Tom's Employment History

- On January 1st, 1996, Tom Joined the faculty as an Instructor.
- On December 1st, 1996, he was promoted to Assistant Professor, retroactively on July 1st, 1996.
- On March 1st, 2001 he was promoted to Associate Professor, proactively on July 1st, 2001.

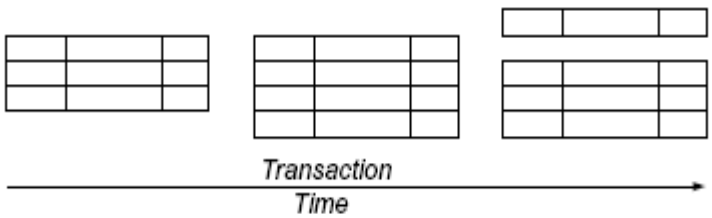
A Snapshot Table



- Can be modified
- Used for static queries
- What is Tom's rank?

```
SELECT Rank
FROM Faculty
WHERE Name = 'Tom'
```

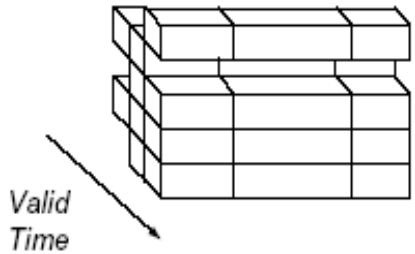
A Transaction-Time Table



- Append-only: correction to previous snapshot states is not permitted
- Supports transaction time
- Supports rollback queries
- What did we believe Tom's rank was on October 1, **1996 ?**

```
SELECT Rank
FROM Faculty
WHERE Name = 'Tom' AND
      TRANSACTION(Faculty) OVERLAPS DATE '1996-10-01'
```

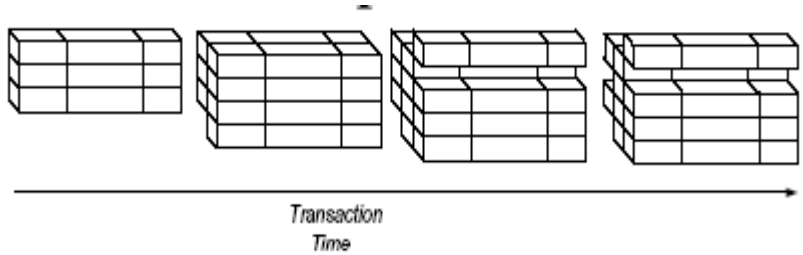
A Valid-Time Table



- May be modified
- Supports valid time
- Supports historical queries
- What was Tom's rank on October 1, **1996** (as best known)?

```
SELECT Rank
FROM Faculty
WHERE Name = 'Tom'
      AND VALID(Faculty) OVERLAPS  DATE '1996-10-01'
```

A Bitemporal Table



- Append-only
- Supports valid and transaction time
- Supports rollback coupled with historical queries

Outline

- Motivation
- Temporal Relational Databases
  - Temporal data models
  - **Abstract temporal databases**
  - Concrete temporal databases
  - TSQL2
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- Summary

Abstract Temporal Databases

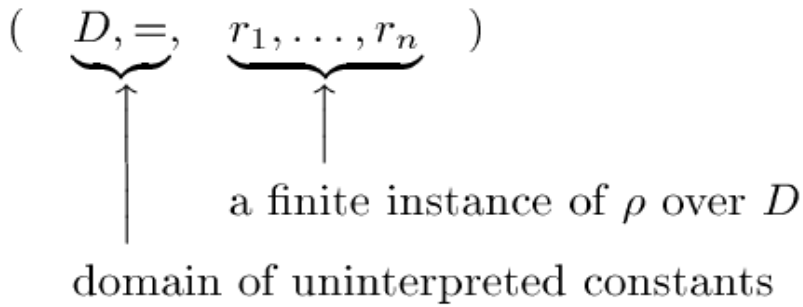
Definitions differ from each other.
Need to give a formal definition of a temporal database, providing a representation-independent semantics for temporal databases.

Advantages:

- allows high-level, declarative query languages
- provides a formal framework to solve outstanding problems in temporal databases:
  - interoperability of different data models
  - functional dependencies and normal forms

Abstract Temporal Databases (cont.)

- Temporal domain $(T, <)$ (linearly ordered, unbounded)
- Relational databases $DB(D, \rho)$ over $(D, =)$ and schema ρ:



A Temporal database is a structure

$D^t = (D, =, r_1^t, \ldots, r_n^t)$ for the two-sorted first-order language containing a temporal relation symbol r_i^t for every r_i, and constant symbols for all the elements of D. The arity of r_i^t is 1 plus the arity of r_i. The intuition is: $D^t \models r_1^t(x_1,..,x_n, t)$ iff $r_1(x_1,..,x_n)$ holds at instant t.

Abstract Temporal Databases (cont.)

- This definition provides a representation-independent meaning for any database defined within any of the temporal data models.

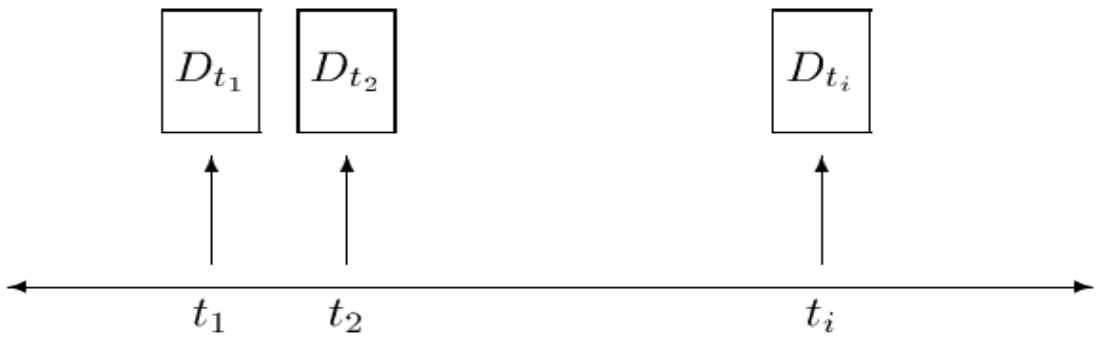
- Any model-specific database (*physical database)* is a *representation* of a temporal database.

- Two physical databases are equivalent if they represent the same temporal database.

- Two major approaches for representing a temporal database:
  - Snapshot model
  - Timestamp model
  - Also logic programs can be used (not covered here)

The Snapshot Model

Temporal database is a function $T \rightarrow DB(D, \rho)$



This approach is the closest to Propositional TL: every database describes the state of the world at a particular time instant; the order relation $<$ on T then defines the flow of time (the access relation).

The Snapshot Model (cont.)

Year	Snapshot
...	
1990	{Work(John, IBM), Work(Mary,IBM), Work(Steve,HP)}
1991	...
1992	...
1993	{ Work(John,Microsoft),Work(Mary,IBM), Work(Steve,HP)}
...	

The Timestamp Model

For every $r_i \in \rho$ we define a relation R_i such that

$$R_i = \{(t, a_1, \ldots, a_k) : (a_1, \ldots, a_k) \in r_i \text{ in } D_t\}$$

A *Timestamp Temporal Database* is a structure

$$(\underbrace{D, =}, \underbrace{T, <}, \underbrace{R_1, \ldots, R_n})$$

$D, =$: temporal domain

$T, <$: temporal domain

$R_1, \ldots, R_n$: an instance of ρ over T and D

The snapshot model is not very suitable for queries of the form

$$\{t : DB \models \varphi(t)\}$$

(all time instants such that φ holds in DB).

The alternative *timestamp model* makes time to be just another data type

The two models are equivalent (type-isomorphic).

The Timestamp Model (cont.)

Work		
Name	Company	Year
John	IBM	1990
John	IBM	1991
John	MicroSoft	1993
John	MicroSoft	...
Mary	DEC	1984
Mary	DEC	1985
Mary	IBM	1990
Mary	IBM	...
Steve	HP	1990
Steve	HP	...

Query Languages

- Main properties of interest:
  - Closure: the query result should be represented in the language of the underlying database. Trivially satisfied in RDB, but not for other kinds of DB (*v.g.* constraint databases).
  - Representation-independence (answer should be the same for different physical representations of a temporal database) [Gadia, 1993]
  - Query expressiveness [Chandra & Harel, 1980]
  - Computational complexity [Vardi, 1982]
  - Efficient implementation

Query Languages (cont.)

Start from *Relational Calculus* over $\rho \cup \{=\}$

$$L ::= r_i(x_1, \ldots, x_k) | x_i = x_j | L \wedge L | \neg L | \exists x.L$$

Example: *list everyone who works for IBM.*

$$\{x : \exists y.\text{Works}(x, y) \wedge y = \text{IBM}\}$$

Two principal temporal extensions:

- implicit references to time (temporal connectives)
- explicit references to time (variables and quantifiers)

Query Languages (cont.)

Temporal Relational Calculus (FO logic)

$$
\begin{array}{lll}
M ::= & \boxed{R_i(t_j, x_{i_1}, \ldots, x_{i_k})} & - \text{ extended database schema} \\
& M \wedge M & \} \text{ propositional connectives} \\
& \neg M & \\
& x_i = x_j & \} \text{ data variables} \\
& \exists x_i . M & \\
& \boxed{\begin{array}{l} t_i < t_j \\ \exists t_i . M \end{array}} & \} \text{ temporal variables}
\end{array}
$$

$\Rightarrow$ essentially a two-sorted first-order logic (2-FOL)

Query Languages (cont.)

- *list John's work history*

$$\exists x.\text{Works}(t_0, x, y) \wedge x = \text{John}$$

- *list all people who were rehired by the same company*

$$\exists y.\text{Works}(t_0, x, y) \wedge \exists t_1 (t_1 < t_0 \wedge \neg \text{Works}(t_1, x, y) \wedge \exists t_2 . t_2 < t_1 \wedge \text{Works}(t_2, x, y))$$

- *list all people who have been unemployed between two jobs*

$$\exists y.\text{Works}(t_0, x, y) \wedge \exists t_1 (t_1 < t_0 \wedge \neg \exists y.\text{Works}(t_1, x, y) \wedge \exists t_2 . t_2 < t_1 \wedge \exists y.\text{Works}(t_2, x, y))$$

Query Languages (cont.)

Temporal Connectives (Temporal logic languages – Tuzhilin-Clifford, 1990)

The standard connectives **since** and **until**:

$$X_1 \textbf{ until } X_2 \stackrel{\triangle}{=} \exists t_2.t_0 < t_2 \wedge X_2 \wedge \forall t_1(t_0 < t_1 < t_2 \rightarrow X_1)$$
$$X_1 \textbf{ since } X_2 \stackrel{\triangle}{=} \exists t_2.t_0 > t_2 \wedge X_2 \wedge \forall t_1(t_0 > t_1 > t_2 \rightarrow X_1)$$

The derived connectives:

$$\Diamond X_1 \stackrel{\triangle}{=} \text{true } \textbf{until } X_1 \qquad \Box X_1 \stackrel{\triangle}{=} \neg\Diamond\neg X_1$$
$$\blacklozenge X_1 \stackrel{\triangle}{=} \text{true } \textbf{since } X_1 \qquad \blacksquare X_1 \stackrel{\triangle}{=} \neg\Diamond\neg X_1$$

For discrete linear order we also define the $\bigcirc$ (next) and $\bullet$ (previous) operators as

$$\bigcirc X_1 \stackrel{\triangle}{=} \exists t_1.t_1 = t_0 + 1 \wedge X_1 \qquad \bullet X_1 \stackrel{\triangle}{=} \exists t_1.t_1 + 1 = t_0 \wedge X_1$$

Query Languages (cont.)

- *list John's work history*

$$\exists x.\mathrm{Works}(x, y) \wedge x = \mathrm{John}$$

- *list all people who were rehired by the same company*

$$\exists y.\mathrm{Works}(x, y) \wedge \blacklozenge(\neg\mathrm{Works}(x, y) \wedge \blacklozenge\mathrm{Works}(x, y))$$

- *list all people who have been unemployed between two jobs*

$$\exists y.\mathrm{Works}(x, y) \wedge \blacklozenge(\neg\exists y.\mathrm{Works}(x, y) \wedge \blacklozenge\exists y.\mathrm{Works}(x, y))$$

Temporal Relational Algebra

- Semantics of Relational Algebra: defined set-theoretically for arbitrary, not necessary finite relations, *i.e.*fits the model-theoretic point of view of abstract temporal databases.

- Under the timestamp view, Relational Algebra operations must be generalized, *v.g.*

$$R(A,T) \bowtie S(B,T) = \{(a,b,\phi_1 \wedge \phi_2) | (a,\phi_1) \in R, (b,\phi_2) \in S\}$$

where ϕ_1 and ϕ_2 are timestamp formulas.

Outline

- Motivation
- Temporal Relational Databases
  - Temporal data models
  - Abstract temporal databases
  - **Concrete temporal databases**
  - TSQL2
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- Summary

Concrete Temporal Databases

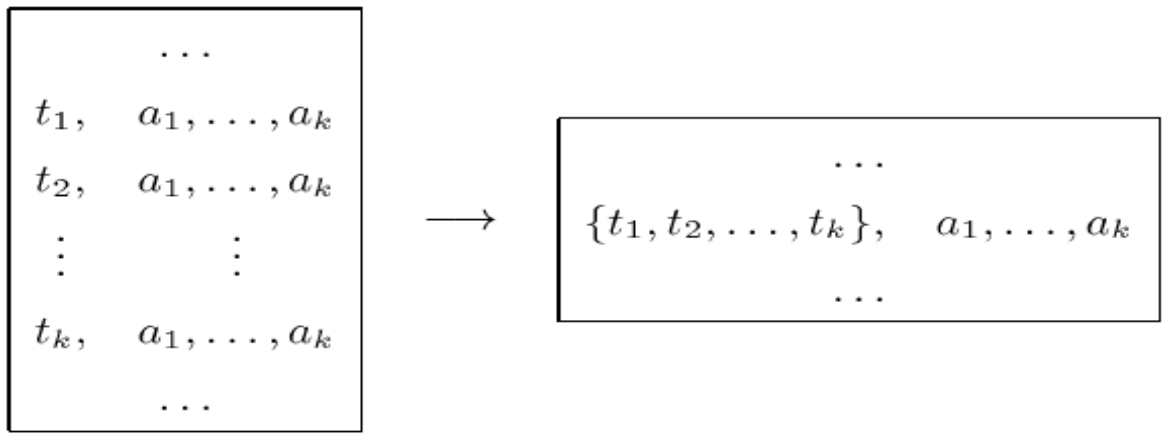
- The class of Abstract Temporal Databases (point-based) gives representation-independent meaning to a TDB
- The class of Concrete Temporal Databases (interval-based) contains space-efficient encodings of the abstract temporal databases.

Concrete Temporal Databases

Grouping by non-temporal attributes and

+

finite encodings of the resulting sets of time instants



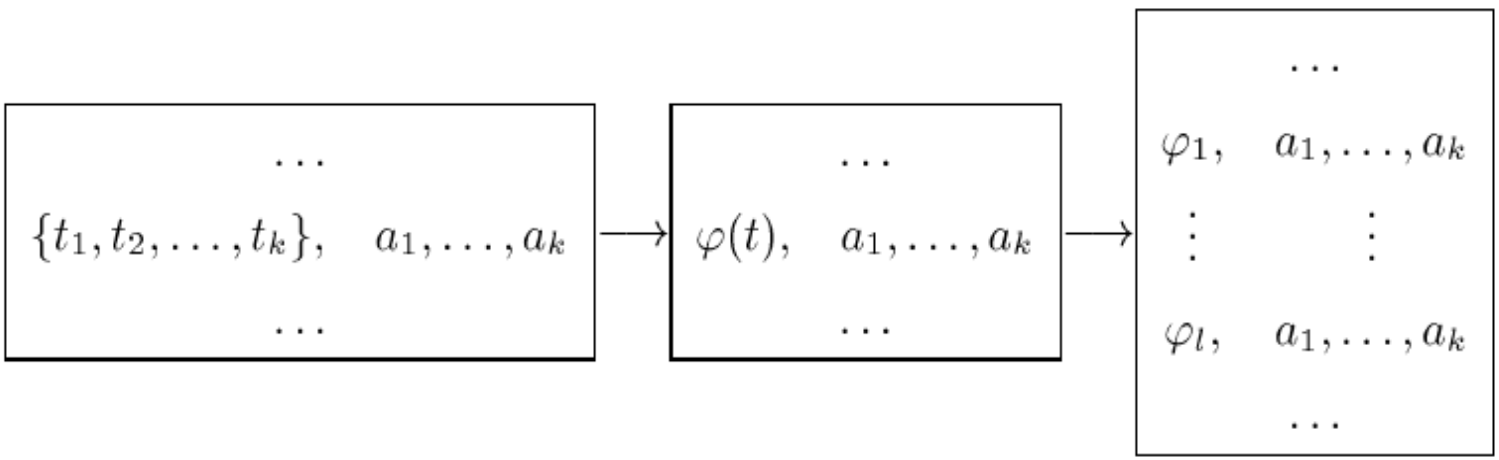
$T \times D^k$ $\qquad\qquad$ $2^T \times D^k$

Implicit functional dependency $A_1 \ldots A_k \to T$ in $2^T \times D^k$.

Concrete Temporal Databases (cont.)

Finite Encoding Using Constraints

- sets of time instants described by their characteristic formulas
- the language of constraints admits quantifier elimination
- fixed-size tuples = quantifier-free formulas in CNF



Concrete Temporal Databases (cont.)

Interval Encoding

Let T_P be the temporal domain. We define the set

$$I(T) = \{(a, b) : a \leq b, a \in T \cup \{-\infty\}, b \in T \cup \{\infty\}\}$$

Relations on the elements of $I(T)$:

$$([a, b] <_{--} [a', b']) \Longleftrightarrow a < a' \quad ([a, b] <_{+-} [a', b']) \Longleftrightarrow b < a'$$
$$([a, b] <_{-+} [a', b']) \Longleftrightarrow a < b' \quad ([a, b] <_{++} [a', b']) \Longleftrightarrow b < b'$$

The structure $T_I = (I(T), <_{--}, <_{+-}, <_{-+}, <_{++})$ is the *Interval-based Temporal Domain (corresponding to T_P).*

Concrete Temporal Databases (cont.)

Example

Work		
Name	Company	Year
John	IBM	[1990, 1991]
John	MicroSoft	[1993, ∞]
Mary	DEC	[1984, 1985]
Mary	IBM	[1990, ∞]
Steve	HP	[1990, ∞]

Interval Queries

M	$::=$	$R_i(I_j, x_{i_1}, \ldots, x_{i_k})$	–	extended database schema
		$M \wedge M$	}	propositional connectives
		$\neg M$		
		$x_i = x_j$	}	data variables
		$\exists x_i.M$		
		$I_i^* < I_j^*$	}	temporal variables
		$\exists I_i.M$		

L^I is the language of M-formulas.

Note that I's range over $I(T)$ – INTERVALS.

Genericity

Definition. $\varphi \in L_I$ is $\|.\|$-*generic* if

$$\|D_1\| = \|D_2\| \supset \|\varphi D_1\| = \|\varphi D_2\|$$

for all D_1, D_2 concrete temporal databases.

Let D_1, D_2 be two concrete temporal databases such that:

$$R^{D_1} = \{([0,3], a)\}$$
$$R^{D_2} = \{([0,2], a), ([1,3], a)\}$$

Then $\exists I, J. \exists x (R(I,x) \wedge R(J,x) \wedge I \neq J)$ is true in D_2 but false in D_1.

$\Rightarrow$ not generic.

Coalescing

A single-dimensional temporal relation is *coalesced* if every fact is associated only with maximal non-overlapping intervals.

- coalescing has to be enforced using non-logical operation on relations
- relational operations do not preserve coalescing in general e.g., projection, union, or set difference.

Outline

- Motivation
- Temporal Relational Databases
  - Temporal data models
  - Abstract temporal databases
  - Concrete temporal databases
  - **TSQL2**
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- Summary

TSQL2 [Snodgrass, 1995]

- similar extension to SQL
- valid and transaction time:
  $\Rightarrow$ temporal elements as timestamps
- no formal semantics
  $\Rightarrow$ unclear expressive power

TSQL2 (cont.)

- TSQL2 supports a bounded discrete representation of the real time line.
- The TSQL2 time line is composed of chronons, which is the smallest granularity.
- Consecutive chronons may be grouped together into granules, yielding multiple granularities.
- A variety of granularities are available, and it is possible to convert from one granularity to another (via *scaling*).

TSQL2 (cont.)

Snapshot queries

- Who has been prescribed drugs?

```
SELECT SNAPSHOT Name
FROM Prescription
```

- Result is a list of names of those with current or past prescriptions.
- Who is or was taking the drug Proventil?

```
SELECT SNAPSHOT Name
FROM Prescription
WHERE Drug = 'Proventil'
```

- Result is a list of names.

TSQL2 (cont.)

- Who has been on a drug for more than a total of six months?

```
SELECT Name, Drug
FROM Prescription(Name, Drug) AS P
WHERE CAST(VALID(P) AS INTERVAL MONTH)
                                > INTERVAL '6' MONTH
```

- Result will contain the maximal interval(s) when the patient has been on the drug.

TSQL2 (cont.)

Valid-time projection

- A new clause, the **VALID** clause, specifies the timestamp of the resulting tuple.
- What drugs was Melanie prescribed during 1994?

```
SELECT Drug
VALID INTERSECT(VALID(Prescription),
                PERIOD '[1994]' DAY)
FROM Prescription
WHERE Name = 'Melanie'
```

- The result is a list of drugs, each associated with a set of the periods during 1994 that they were prescribed to Melanie.

Outline

- Motivation
- Temporal Relational Databases
  - The structure of time
  - Abstract temporal databases
  - Concrete temporal databases
  - TSQL2
- **Beyond the relational model**
  - **XML**
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- Summary

How the Web is Today

- HTML documents
  - often generated by applications
  - consumed by humans only
  - easy access: across platforms, across organizations
- No application interoperability:
  - HTML not understood by applications

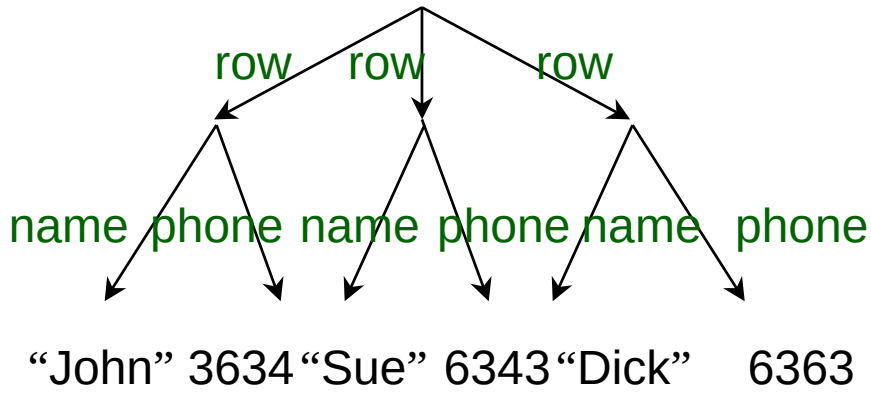
XML: A Universal Data Exchange Format

A recommendation from the W3C

- XML = data
- XML generated by applications
- XML consumed by applications
- Easy access: across platforms, organizations

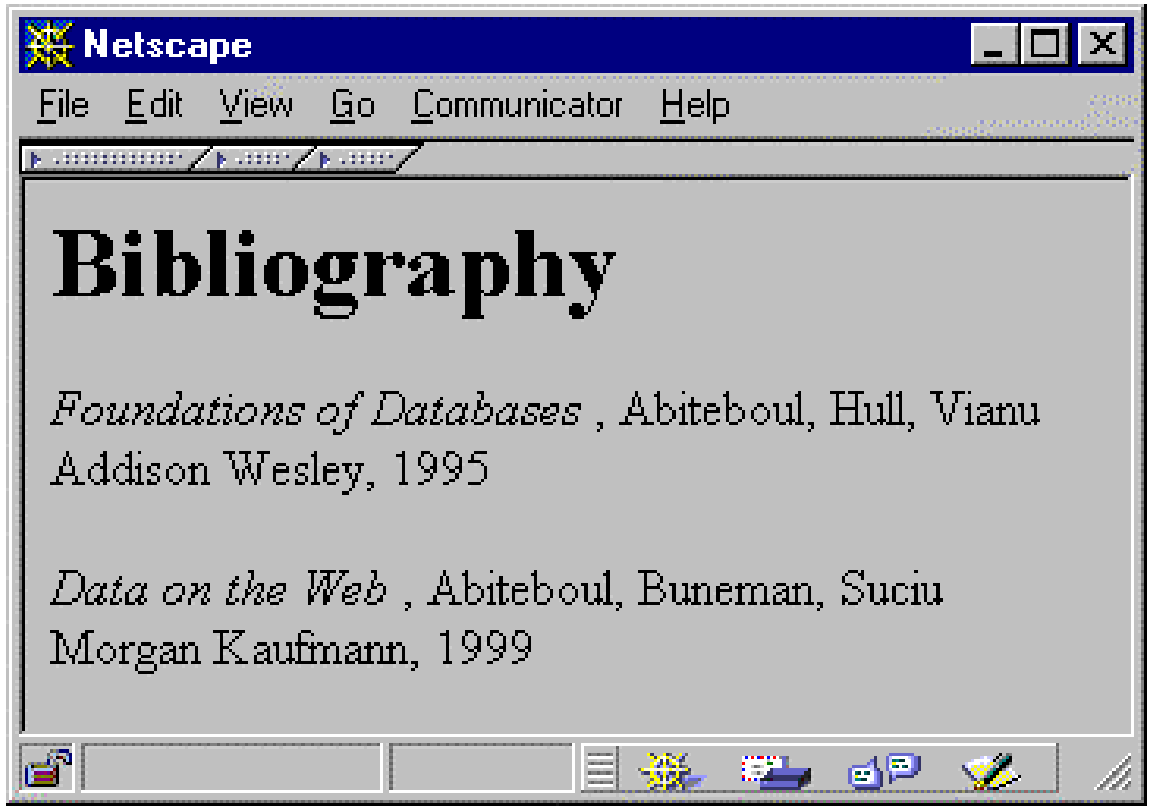
Comparison with Relational Data

name	phone
John	3634
Sue	6343
Dick	6363



{ row: { name: “John”, phone: 3634 },
row: { name: “Sue”, phone: 6343 },
row: { name: “Dick”, phone: 6363 }
}

From HTML to XML



HTML describes the presentation

HTML Characteristics

<h1> Bibliography </h1>
<p> Foundations of Databases
Abiteboul, Hull, Vianu

 Addison Wesley, 1995
<p> Data on the Web
Abiteboul, Buneman, Suciu

 Morgan Kaufmann, 1999

An XML Document

```
<bibliography>
    <book>   <title> Foundations… </title>
             <author> Abiteboul </author>
             <author> Hull </author>
             <author> Vianu </author>
             <publisher> Addison Wesley </publisher>
             <year> 1995 </year>
    </book>
    …
</bibliography>
```

XML describes the content

Why are we DB'ers interested?

- It's data => that's us.
- Proof by Google:
  - database+XML – 1,940,000 pages.
- Database issues:
  - How are we going to model XML? (graphs).
  - How are we going to query XML? (Xquery/Xpath)
  - How are we going to store XML (in a relational database? object-oriented? native?)
  - How are we going to process XML efficiently? (many interesting research questions!)

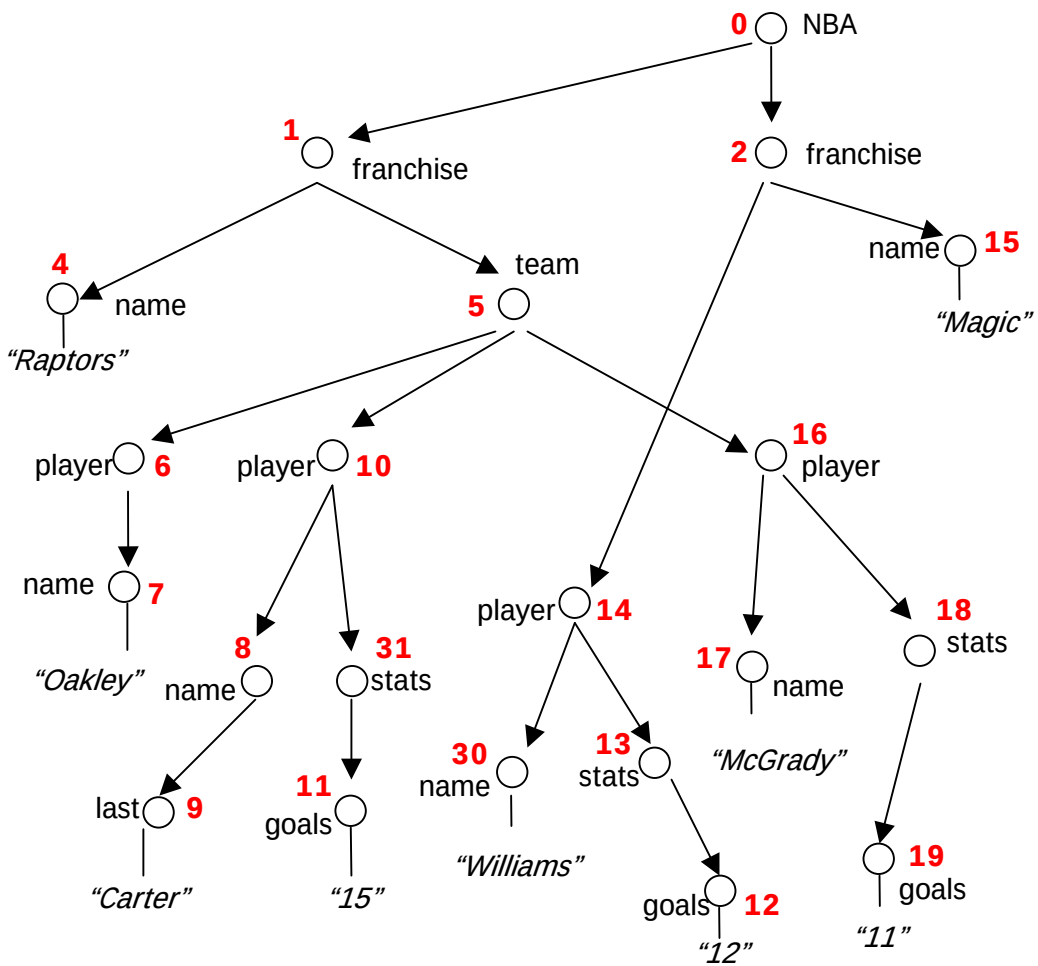
Outline

- Motivation
- Temporal Relational Databases
  - The structure of time
  - Abstract temporal databases
  - Concrete temporal databases
  - TSQL2
- Beyond the relational model
  - XML
  - **Temporal extensions to XML**
  - RDF
  - Temporal extensions to RDF
- Summary

Work on Temporal XML Databases

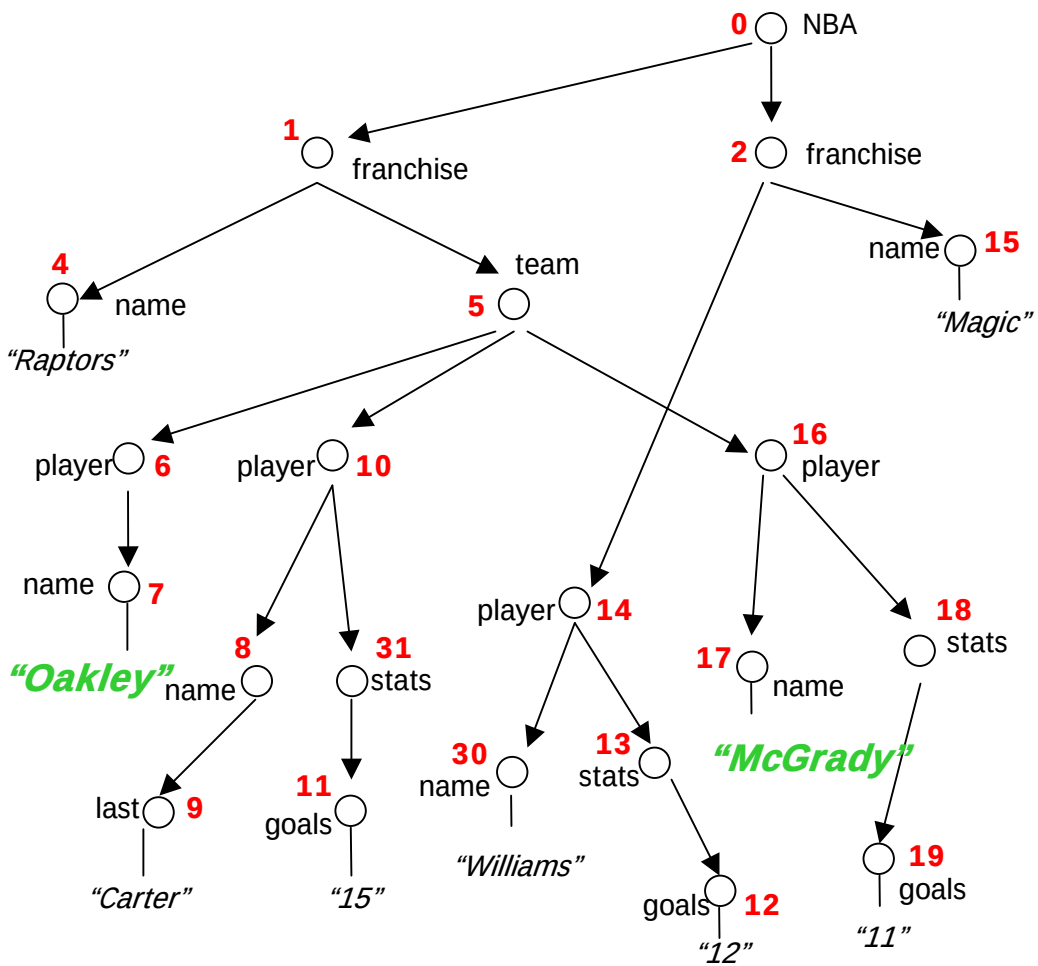
- First model:
  - **Amagasa *et al* (DEXA 2000) :** first simple model for temporal XML. No updates, nor query language.
- Models based on Versioning
  - **Dyreson (WISE'01) TTXPath.** Extends XPath with temporal axes to represent transaction time. *Snapshot* model – considers each version of an XML document as a snapshot
  - **Chien, Tsotras, Zaniolo (WebDB 2000, VLDB 2001)** : update and versioning for XML, based on edit scripts.
- Models based on Timestamping
  - **Gao and Snodgrass (VLDB 2003)** : τXQuery. Minimal changes to XQuery syntax and semantics, same XQuery data model. *Translates to long XQuery programs*, and pass them to an XQuery engine.
  - **Mendelzon, Rizzolo, Vaisman (VLDB 2004):** TXPath data model and query language – *no translation* to Xpath required.

NBA database (at instant 0)



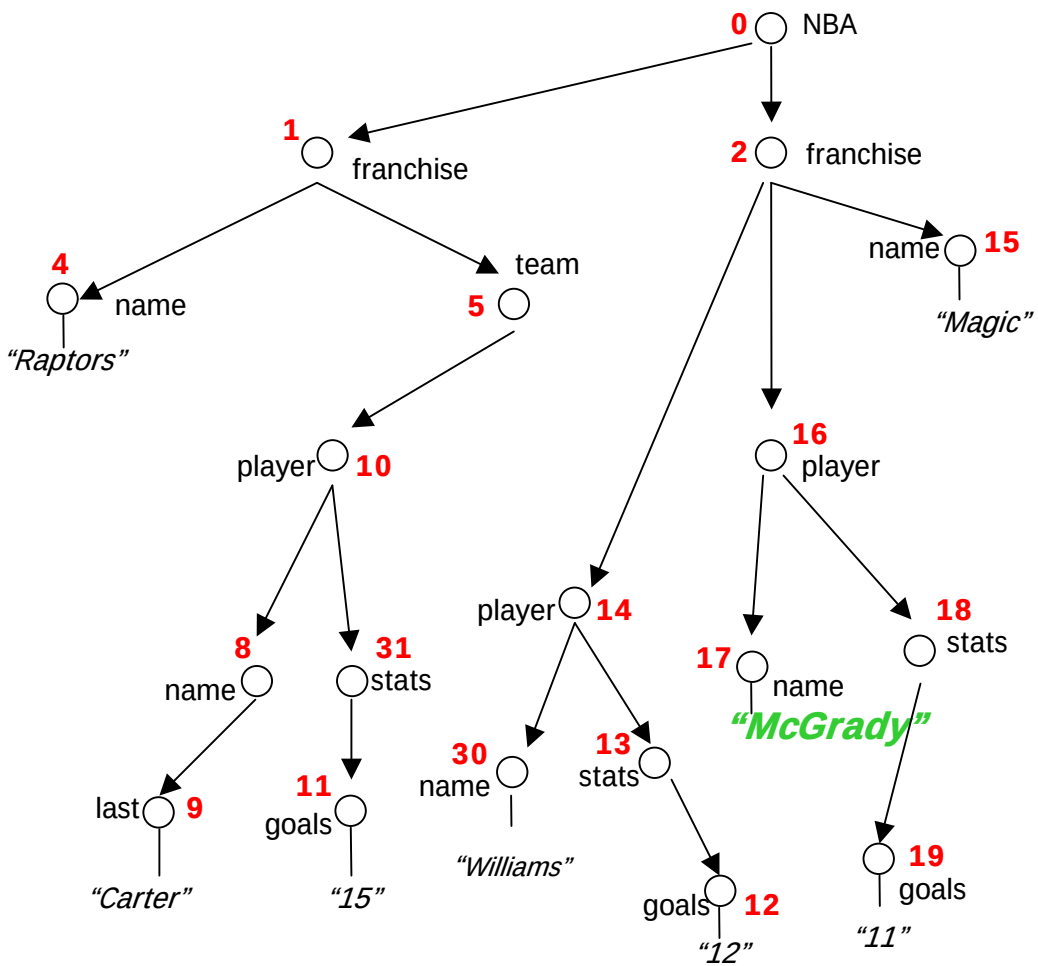

```
<NBA>
  <franchise ID=`1'>
    <name>Raptors</name>
    <team>
      <player>
        <name>Oakley</name>
      </player>
      ...
    </team>
...
  <franchise ID=`2'>
    <name>Magic</name>
    <player ID=`14' >
      <name>Williams</name>
      <stats>
        <goals>12</goals>
      </stats>
    </player>
```

NBA database (at instant 0)



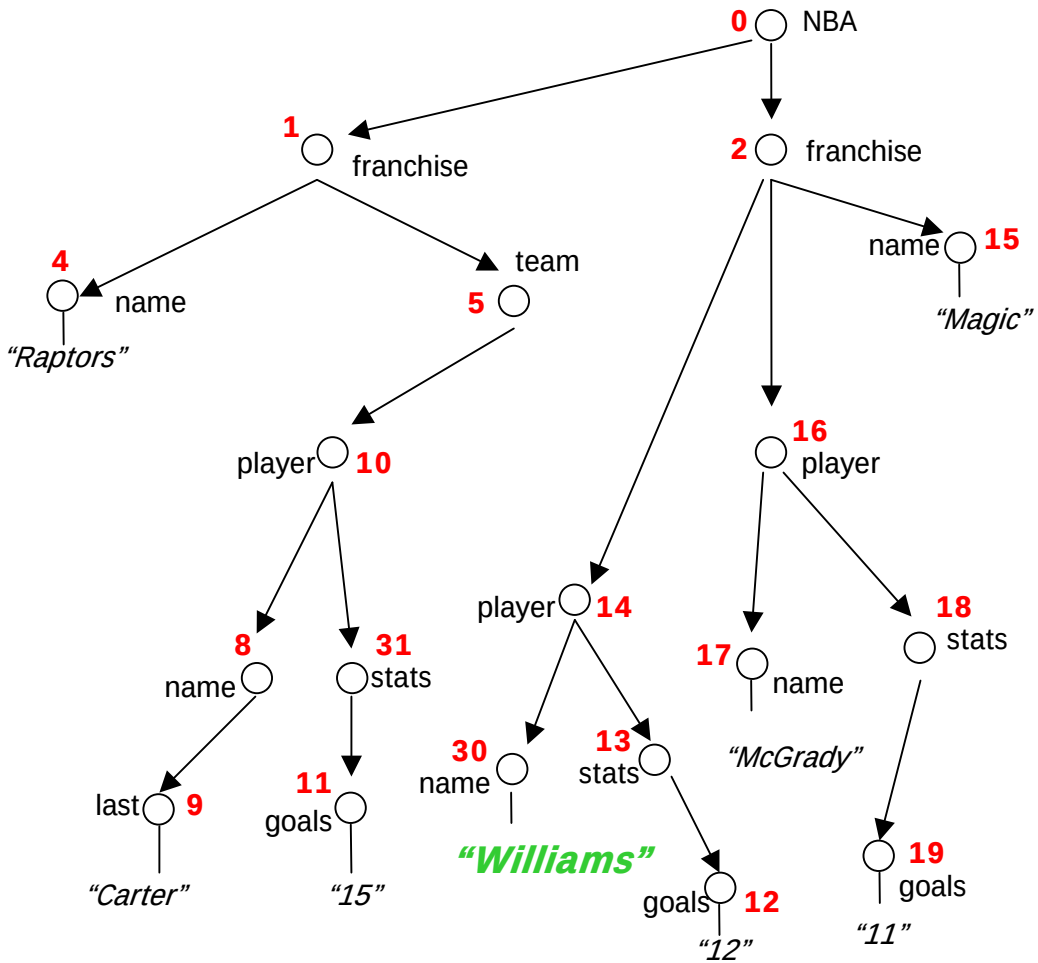

```
<NBA>
  <franchise ID=`1'>
    <name>Raptors</name>
    <team>
      <player ID= '6'>
        <name>Oakley</name>
      </player> ……
      <player ID='16'>
        <name>McGrady</name>
      </player>
    </team>
  <franchise ID='2'>
    <name>Magic</name>
    <player ID='14' >
      <name>Williams</name>
      <stats>
        <goals>12</goals>
      </stats>
    </player>…..
```

NBA database (at instant 21)



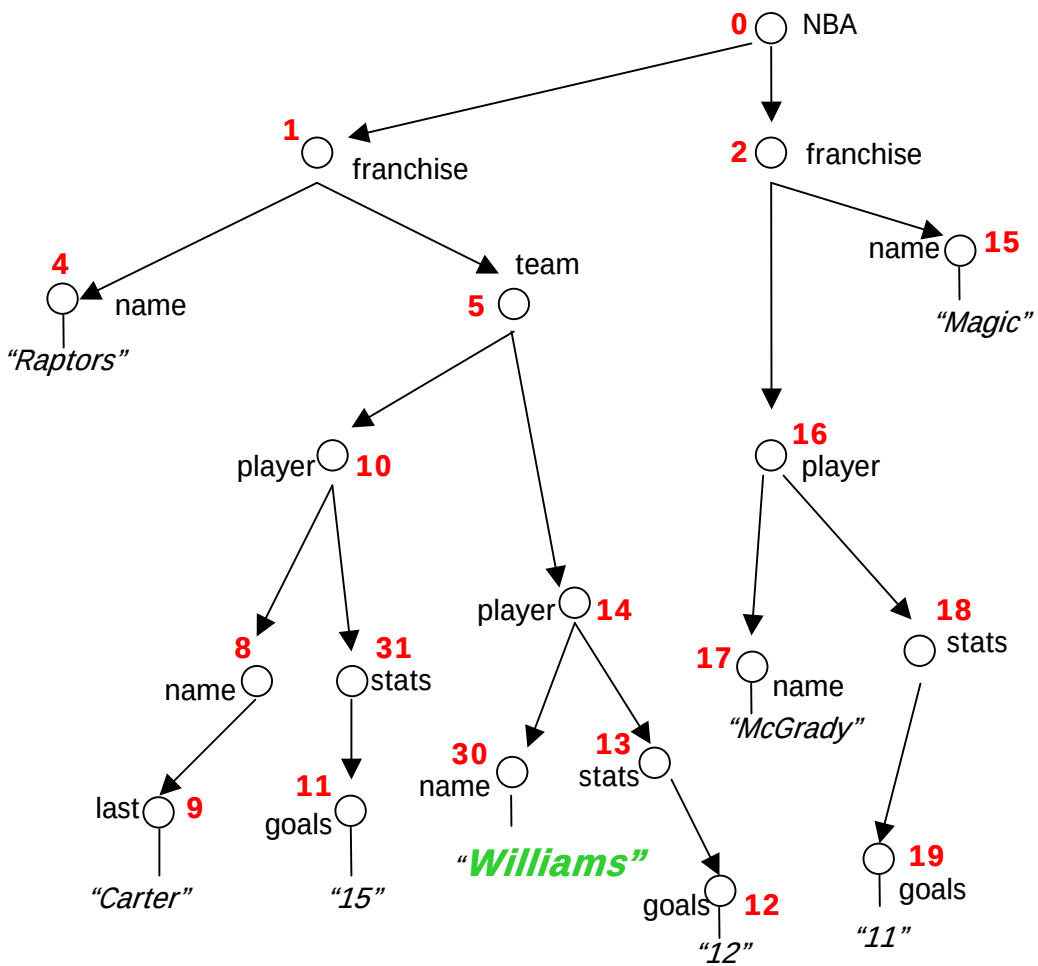

```
<NBA>
  <franchise ID=`1'>
    <name>Raptors</name>
    <team>
    ...
    </team>
  ...
  <franchise ID=`2'>
    <name>Magic</name>
    <player ID=`14' >
      <name>Williams</name>
      <stats>
        <goals>12</goals>
      </stats>
    </player>
    <player ID=`16' >
      <name>McGrady</name>
      ...
```

NBA database (at instant 21)




```
<NBA>
   <franchise ID=`1'>
      <name>Raptors</name>
      <team>
      …
      </team>
…
   <franchise ID=`2'>
      <name>Magic</name>
      <player ID=`14' >
         <name>Williams</name>
         <stats>
            <goals>12</goals>
         </stats>
      </player>
```

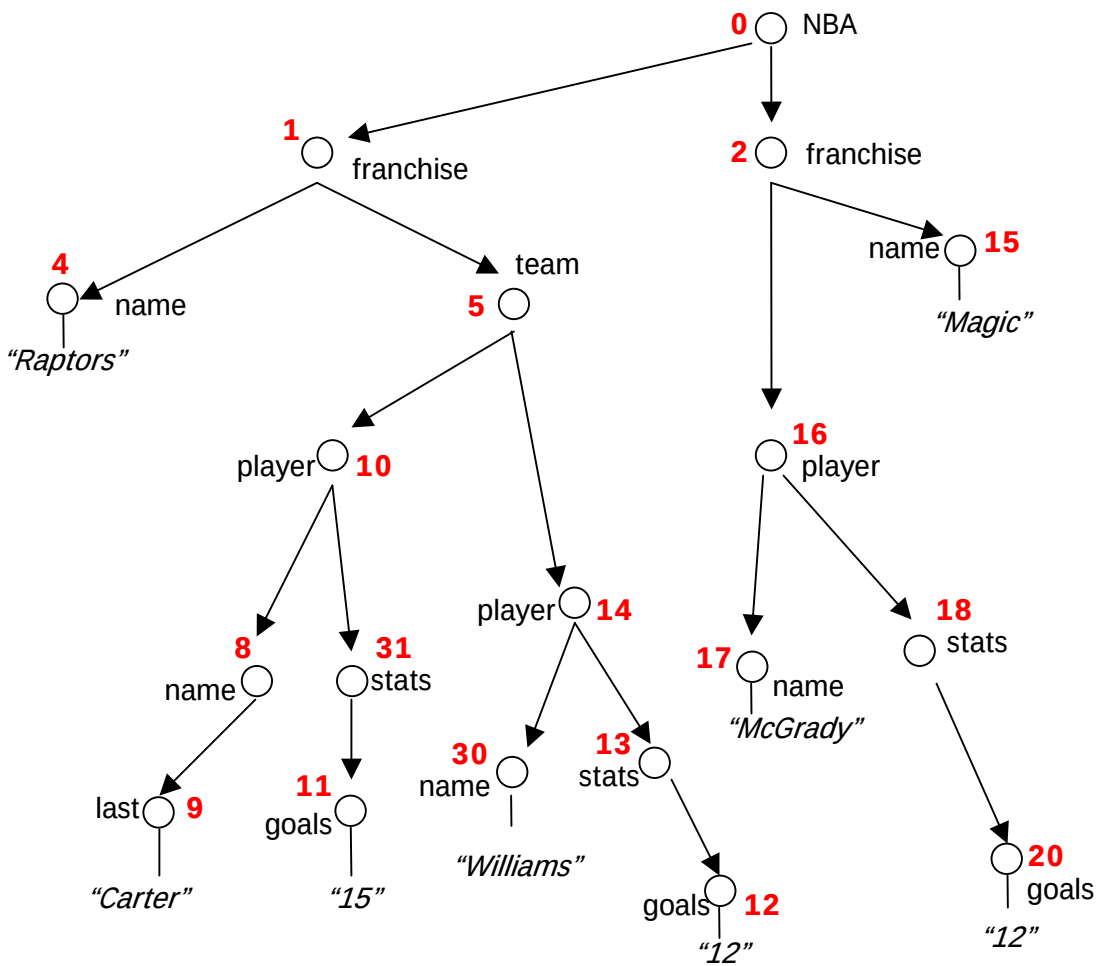
NBA database (at instant 23)




```
<NBA>
   <franchise ID=`1'>
      <name>Raptors</name>
      <team>
         …
         <player ID=`14' >
            <name>Williams</name>
            <stats>
               <goals>12</goals>
            </stats>
         </player>

      </team>
   …
   <franchise ID=`2'>
      <name>Magic</name>
      …
```

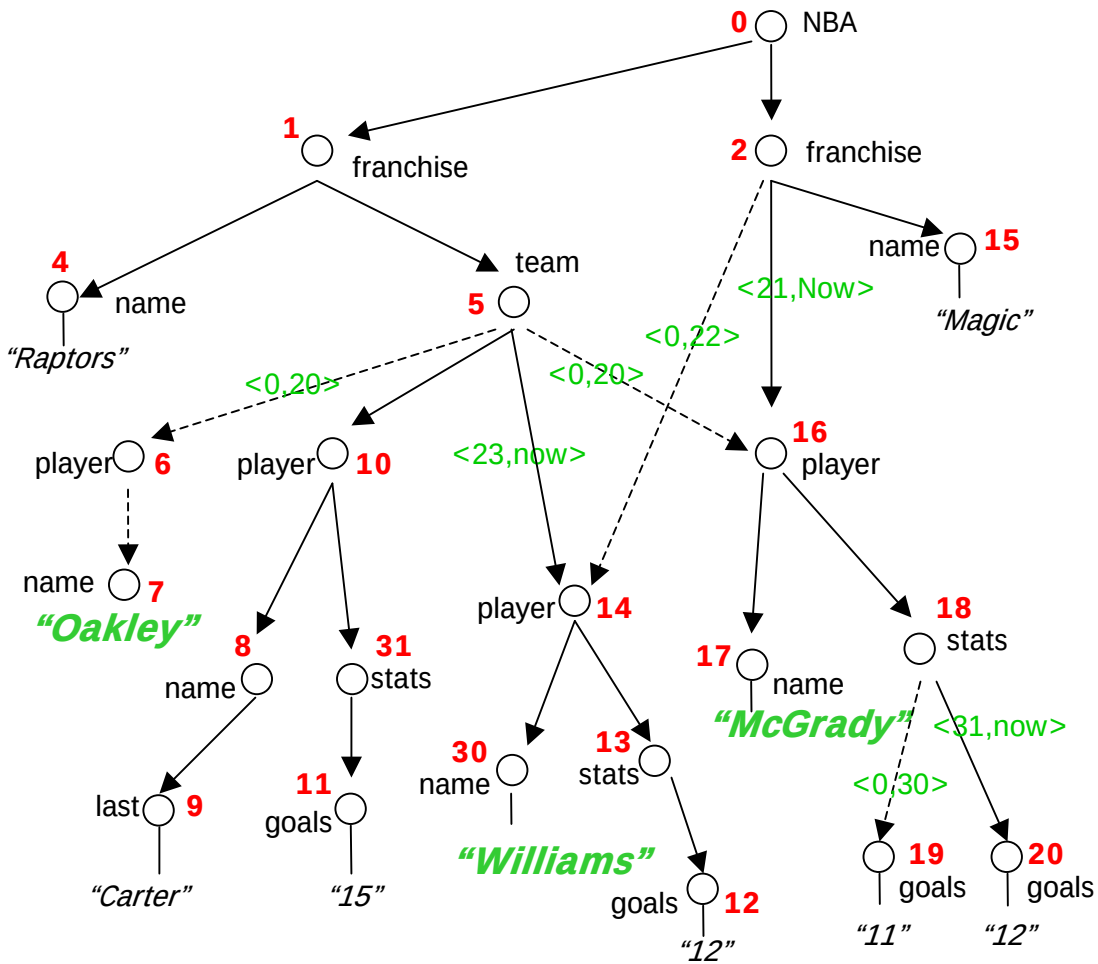
NBA database (at instant 31)




```
<NBA>
  <franchise ID=`1'>
    <name>Raptors</name>
    <team>
      ...
      <player ID=`14' >
        <name>Williams</name>
        <stats>
          <goals>12</goals>
        </stats>
      </player>

    </team>
...
  <franchise ID=`2'>
    <name>Magic</name>
    ...
```

Temporal NBA database

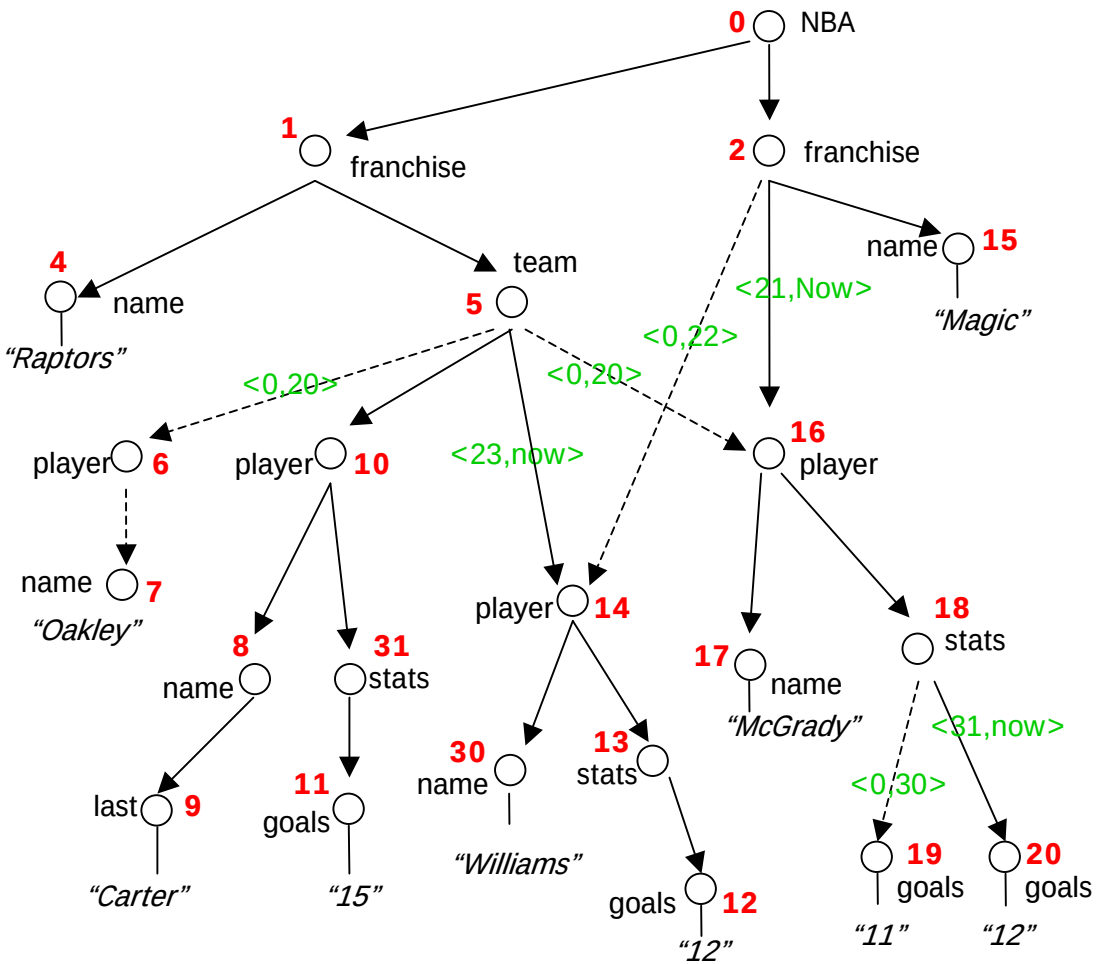



```
<NBA>
  <franchise ID=`1' [0,Now]>
    <name [0,Now]>Raptors</name>
    <team [0,Now] >
      …
      <player IN [23,Now]="14" />
    </team>
    …
  <franchise ID=`2' [0,Now]>
    <name [0,Now]>Magic</name>
    <player [0,22] ID=`14' >
      <name [0,Now]>Williams</name>
      <stats [0,Now]>
        <goals [0,Now]>12</goals>
      </stats>
    </player>
    …
```

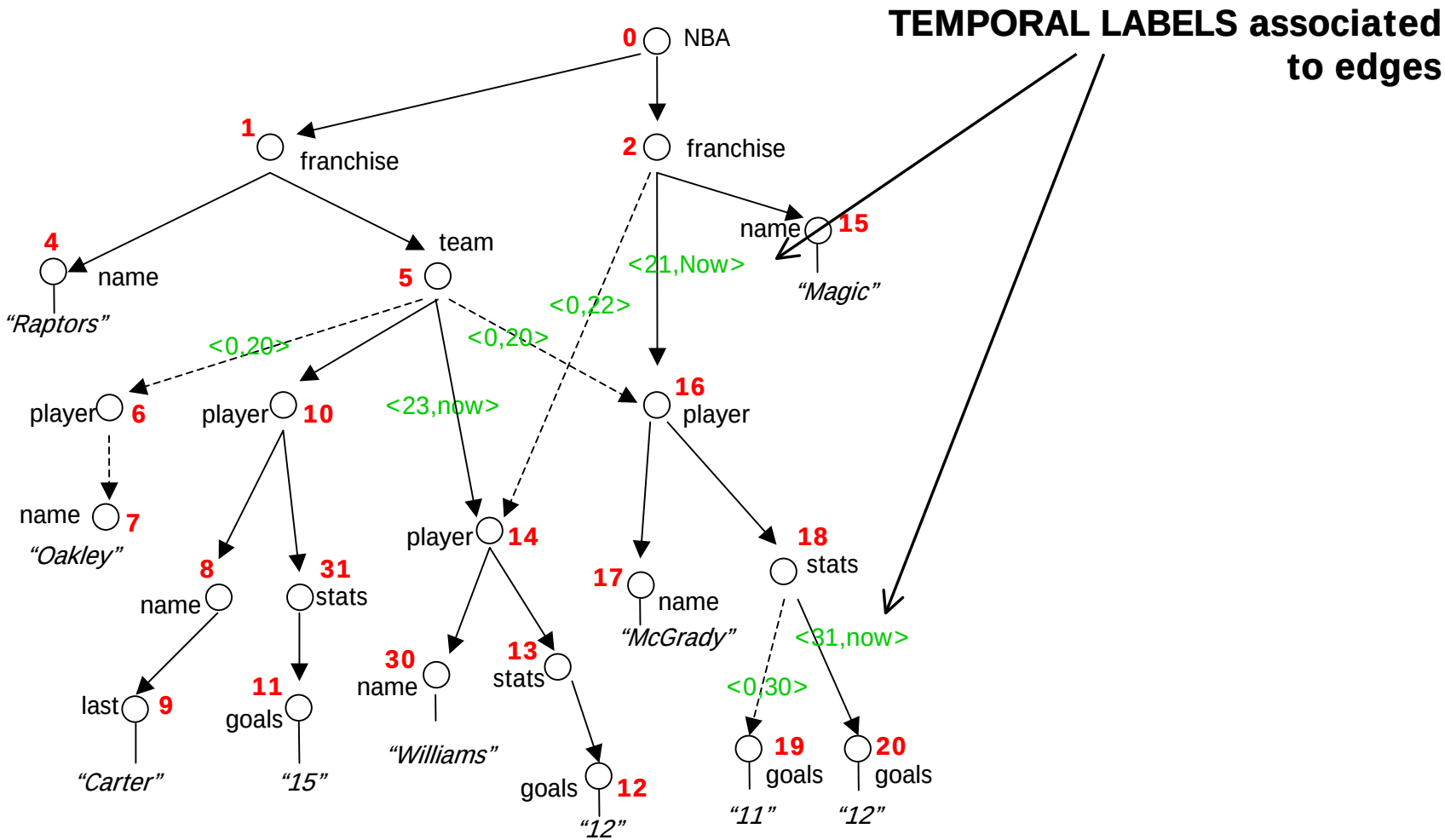
Motivation

- Keep historical information in a unique XML document
- Try to stay as close as possible to XPath data model
- Allow efficient implementation and indexing

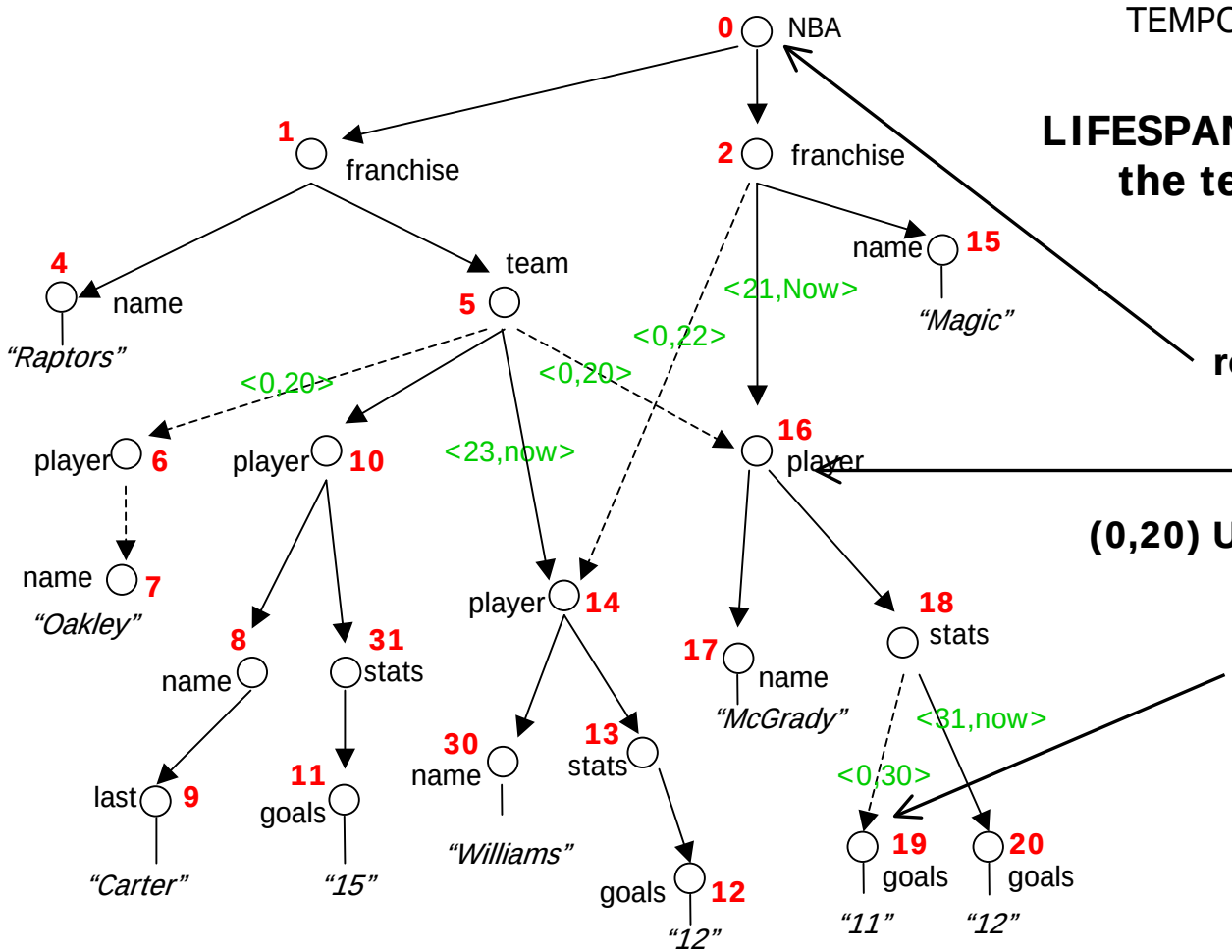
Data Model


0 NBA
1 franchise
2 franchise
4 name
"Raptors"
team
5
name 15
"Magic"
<21,Now>
<0,22>
<0,20>
<0,20>
player 6
player 10
<23,now>
16 player
name 7
"Oakley"
player 14
8 name
31 stats
17 name
"McGrady"
18 stats
<31,now>
<0,30>
30 name
13 stats
last 9
11 goals
"Carter"
"15"
"Williams"
goals 12
"12"
19 goals
20 goals
"11"
"12"

Data Model


TEMPORAL LABELS associated to edges
0 NBA
1 franchise
2 franchise
4 name
"Raptors"
team
5
name 15
"Magic"
<21,Now>
<0,22>
<0,20>
<0,20>
player 6
player 10
<23,now>
16 player
name 7
"Oakley"
player 14
8 name
31 stats
17 name
"McGrady"
18 stats
<31,now>
<0,30>
30 name
13 stats
last 9
"Carter"
11 goals
"15"
"Williams"
goals 12
"12"
19 goals
"11"
20 goals
"12"

Data Model



TEMPORAL LABELS associated to edges

LIFESPAN of a node: union of the temporal labels of the incoming edges

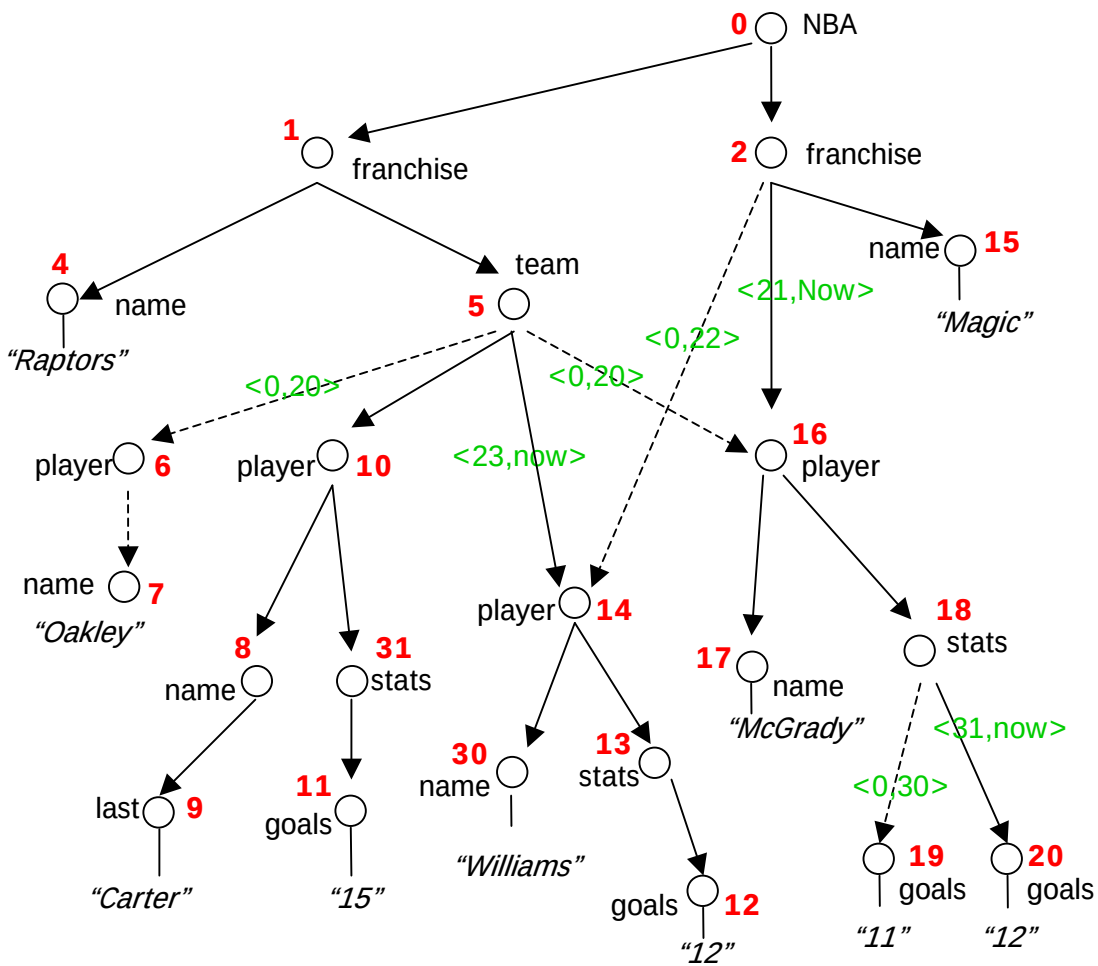
root's lifespan: **(0, Now)**

16 lifespan:

(0,20) U (21, Now) = (0, Now)

19 lifespan: **(0,30)**

Data Model



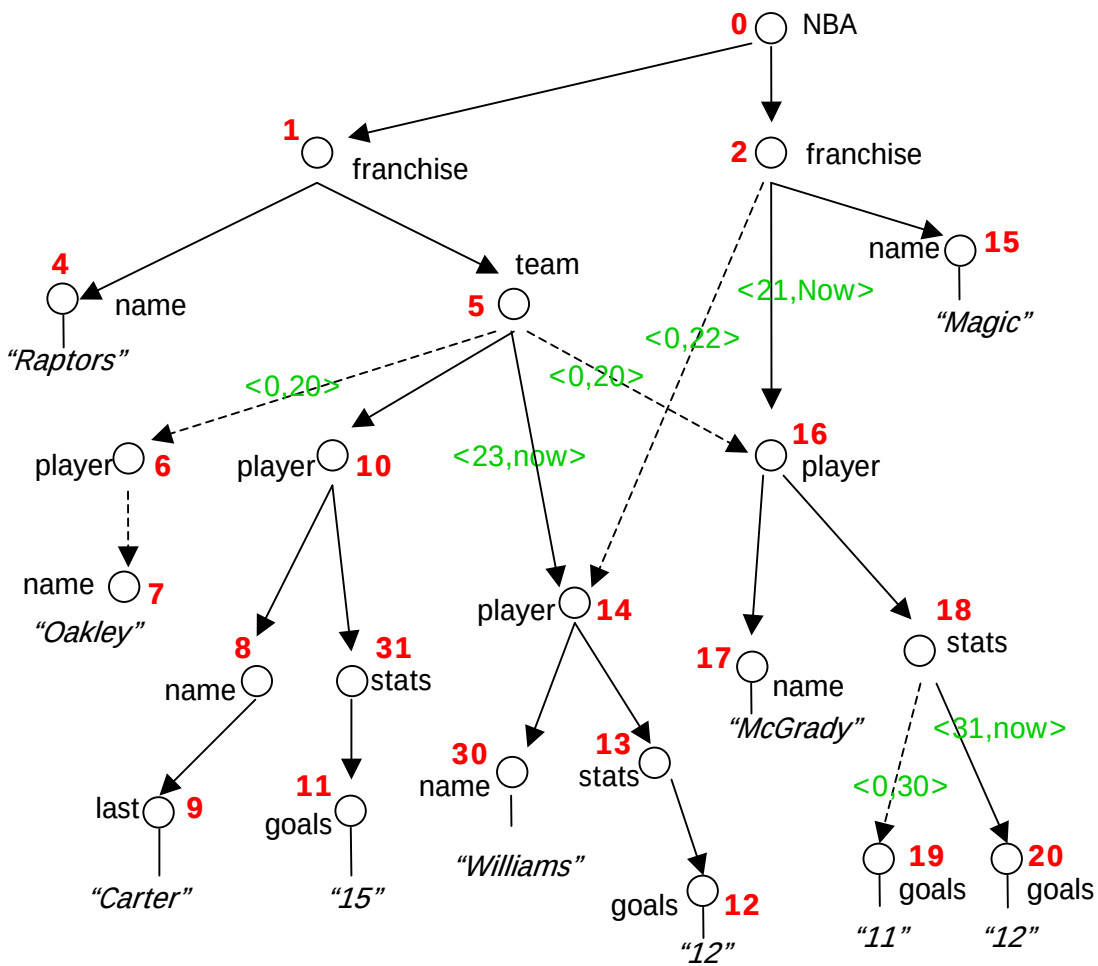
TEMPORAL LABELS associated to edges

LIFESPAN of a node: union of the temporal labels of the incoming edges

Union of the outgoing temporal labels of every node is contained in the lifespan of the node (*node consistency*)

Temporal labels of edges incoming to a node are consecutive

Data Model



TEMPORAL LABELS associated to edges

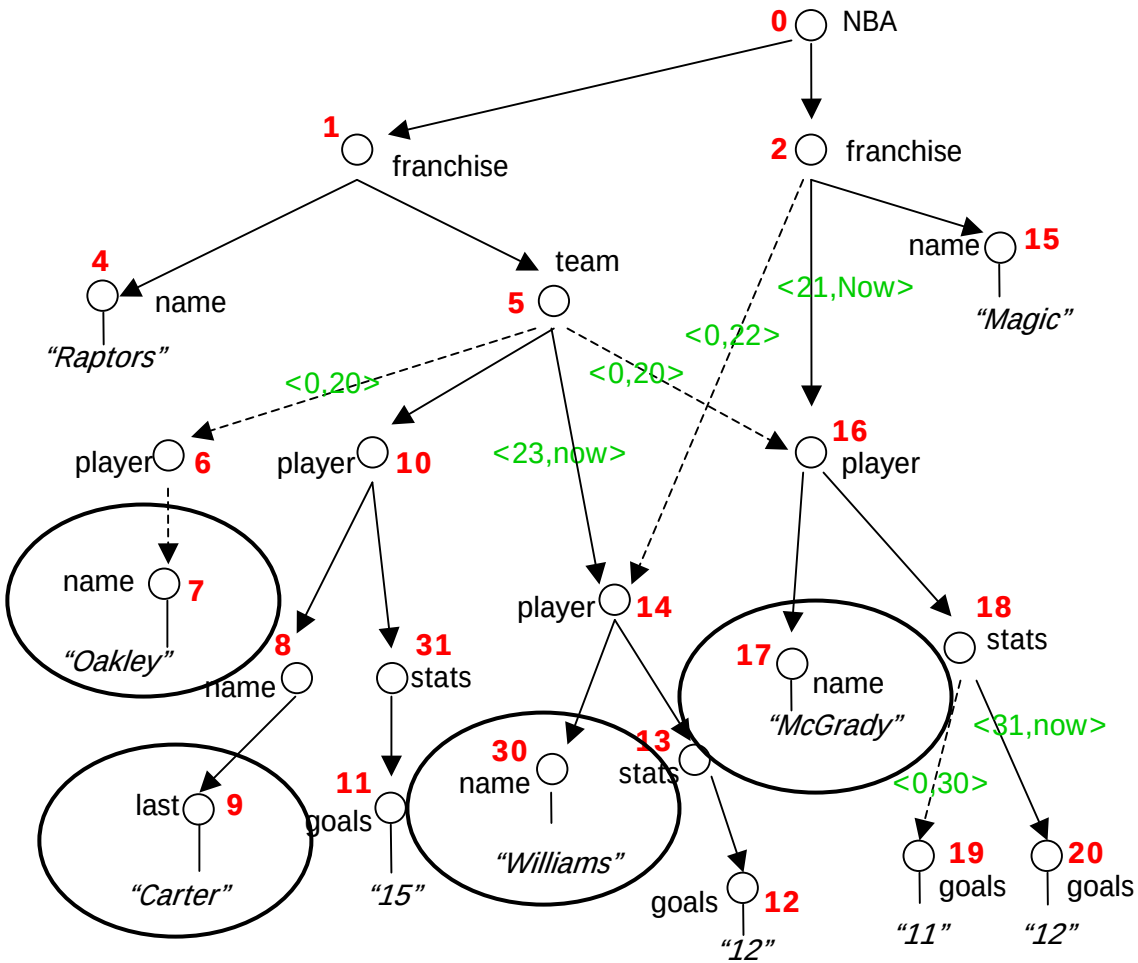
LIFESPAN of a node: union of the temporal labels of the incoming edges

Union of the outgoing temporal labels of every node is contained in the lifespan of the node (*node consistency*)

Temporal labels of edges incoming to a node are consecutive

At any t , the subgraph of all edges (excluding references) is a tree (*i.e.* the snapshot at *t* is an XML document)

TXPath Query Language



A TXPath expression returns a sequence of pairs **(node,interval)** s.t. the node has been continuously at the end of a matching path during the interval (a **Continuous Path (CP)**)

Query: "*Name of the players who ever played for the Toronto Raptors*"

Answer: ("Oakley", <0, 20>)
("Carter", <0, Now>)
("Williams", <23, Now>)
("McGrady", <0, 20>)

TXPath Query Language (cont.)

- *"Name of the players who ever played for the Toronto Raptors"*

//franchise[name="Raptors"]//player/name/text()

- *"Name of the players who where with the Toronto Raptors when Williams joined the franchise"*

let
$m=min(//franchise[name="Raptors"]//player[name="Williams"]/@from)
return
//franchise[name="Raptors"]//player[$m>@from and $m<@to]/name/text()

- *"Average goals scored by Raptors' players in the current season"*

aggregate-seq(//franchise[name="Raptors"]//goals[@to="Now"],Avg())

Outline

- Motivation
- Temporal Relational Databases
  - The structure of time
  - Abstract temporal databases
  - Concrete temporal databases
  - TSQL2
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - **RDF**
  - Temporal extensions to RDF
- Summary

RDF Basics

- Web needs semantic information => the Semantic Web proposal.
- Machine readable semantics for data on the web.
- RDF proposed by W3C in 1998 as a metadata language for the web.
- Simple data model, extensible vocabulary based on URIs.

RDF Basics (cont.)

• In RDF, the universe to be modeled is a set of resources.

•RDF Graph: A set of RDF triples of the form (v1,v2,v3) in (U ∪ B) x (U) x (U ∪ B ∪ L) (U,B,L: URI references, blanks, literals)

• A graph is *ground* if has no blank nodes

• A map is a function μ: UBL -> UBL preserving URIs and literals

RDF Basics (cont.)

- RDF is extended with a vocabulary, defined in RDFS (RDF Schema).
- Typical words: class, subClassof, property, subPropertyOf, type, range, domain ,
- reification : allows making statements about statements.

RDF Basics (cont.)

• First formal semantics given for RDF : Gutierrez, Hurtado & Mendelzon (PODS 2004).

• Gives a set of rules defining the semantics of RDF graphs.

• Defines the closure and the core of RDF graphs

= > a normal form.

• First to treat RDF graphs as databases.

RDF Basics (cont.)

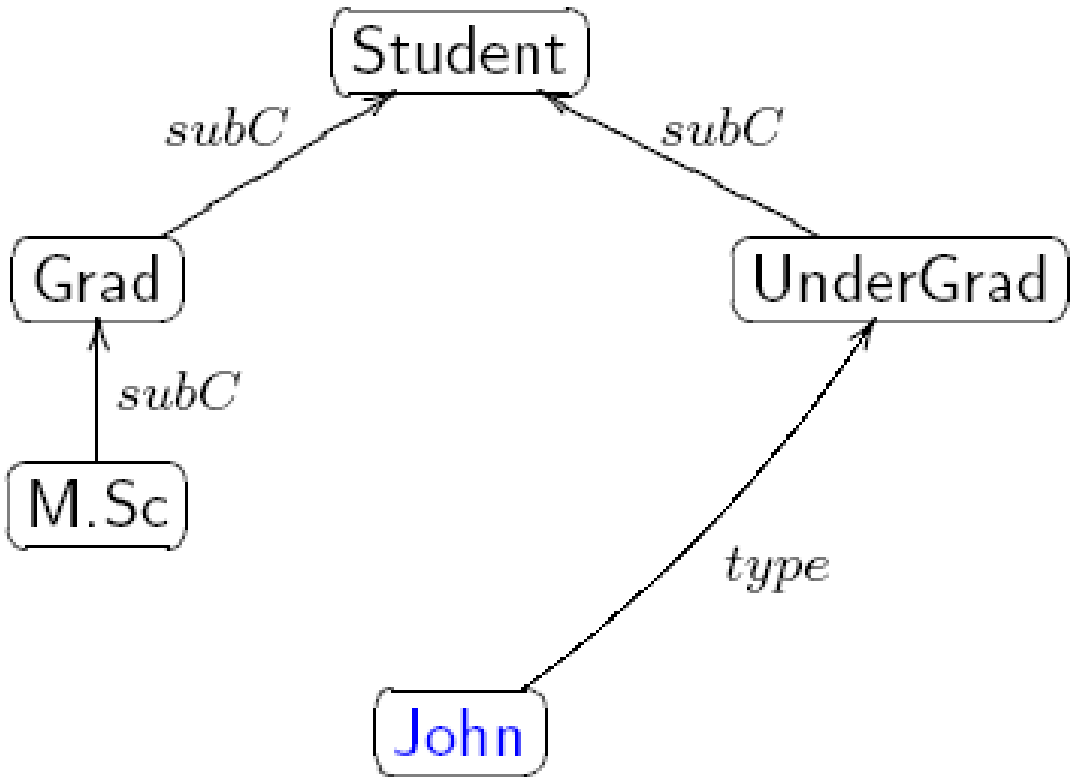
- Core : the unique minimal graph equivalent to an RDF graph G.
- Closure : a maximal set of triples G' over G, that contains G and is equivalent to G.
- Normal form (G) : the core of the closure.

(details in : "Foundations of Semantic Web Databases", , GHM, PODS 2004)

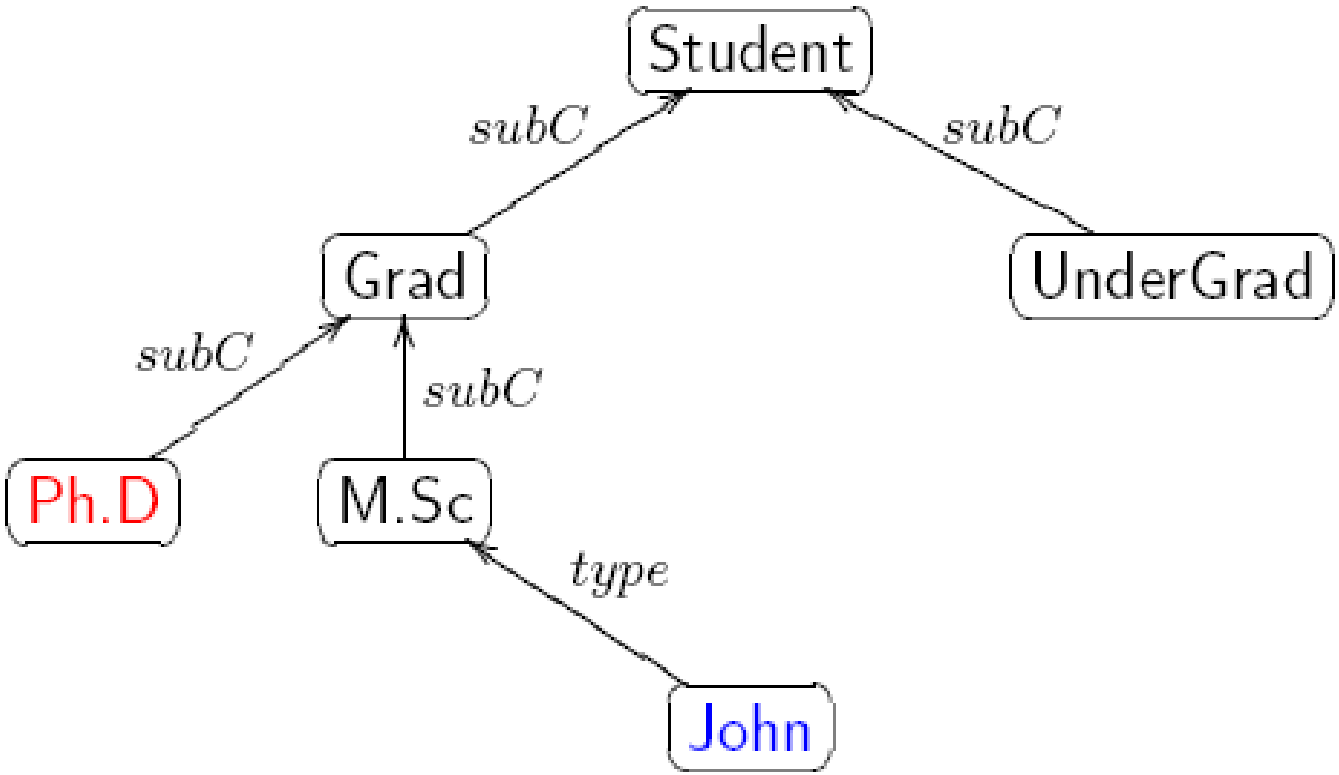
Outline

- Motivation
- Temporal Relational Databases
  - The structure of time
  - Abstract temporal databases
  - Concrete temporal databases
  - TSQL2
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - **Temporal extensions to RDF**
- Summary

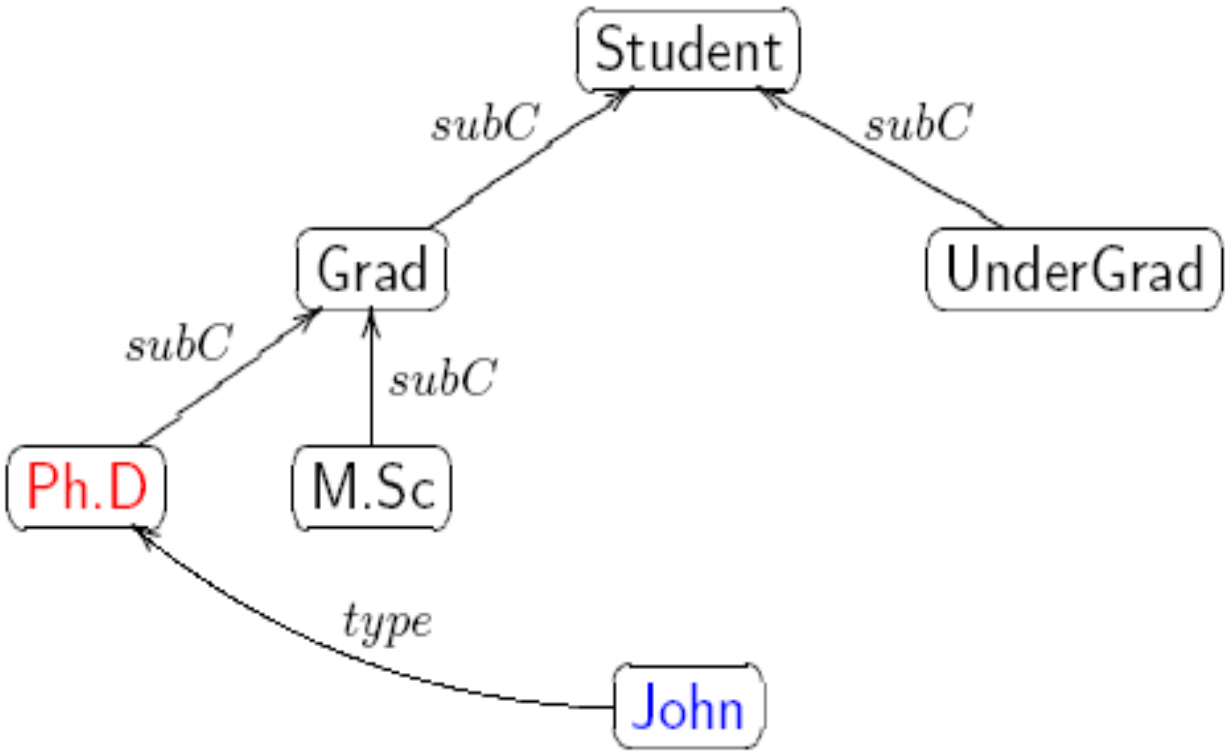
Updates in RDF Graphs



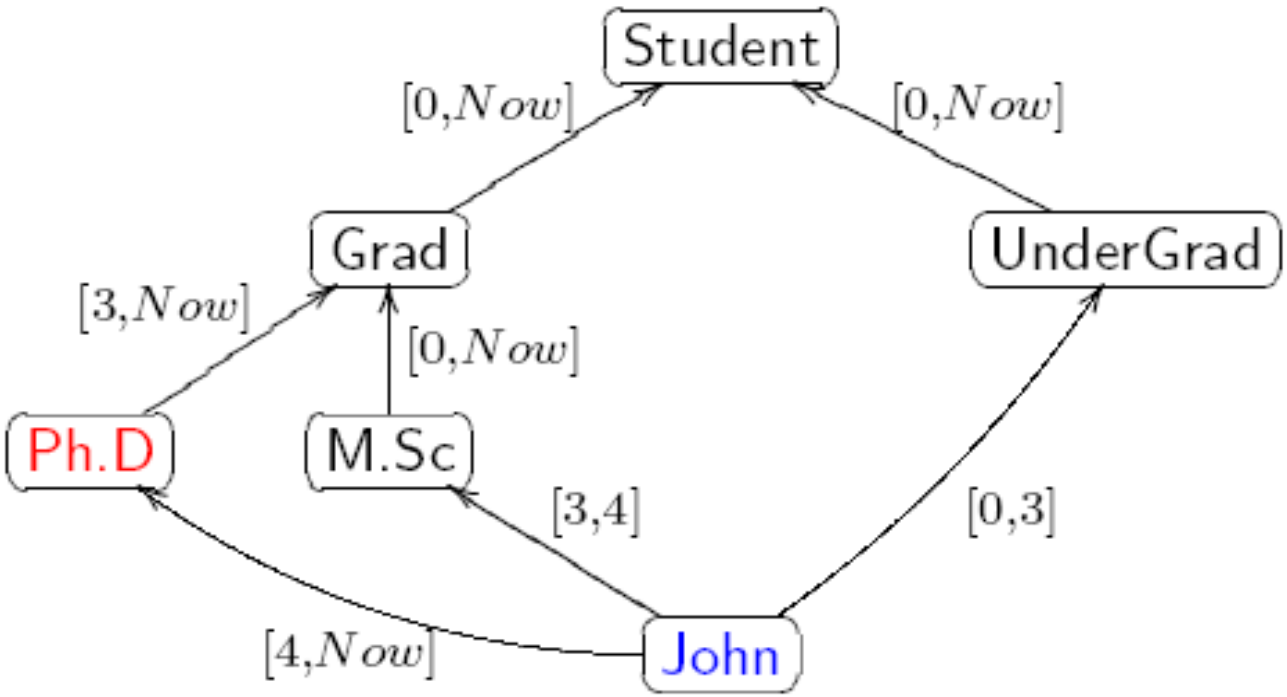
Updates in RDF Graphs (cont.)



Updates in RDF Graphs (cont.)



A Temporal RDF Graph



General Issues

- Versioning versus Labeling
  - Label elements subject to change
  - Maintain a snapshot of each state of the graph
- Time Points versus Time Intervals.

$$[4, 31] = [4] \cup [5] \cup \cdots \cup [30] \cup [31]$$

- Temporal Query Language
  - Point based (variables refer to point times)
  - Interval based (variables refer to intervals)

Definitions

Temporal Triple: an RDF triple with a temporal label, e.g. $(a, b, c)[t]$

Temporal Graph: set of temporal triples

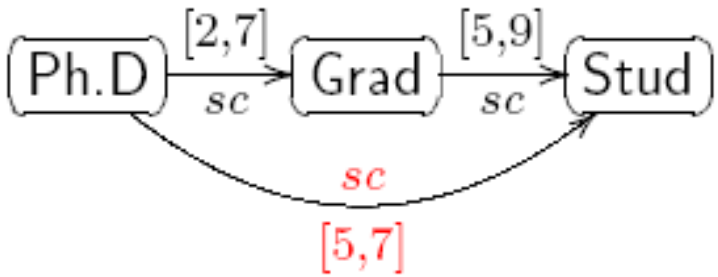
Snapshot of graph G at time t:

$$G(t) = \{(a, b, c) \ : \ (a, b, c)[t] \in G\}$$

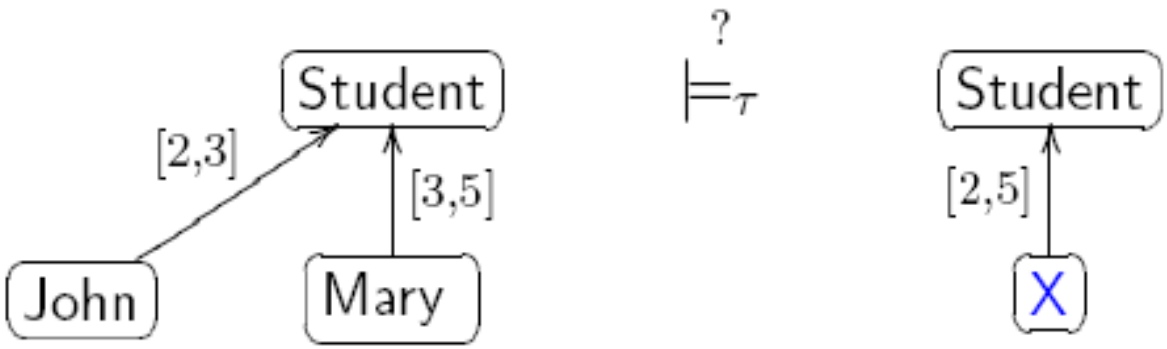
Notion of temporal entailment $G_1 \models_\tau G_2$

Temporal RDF Intrinsic Issues

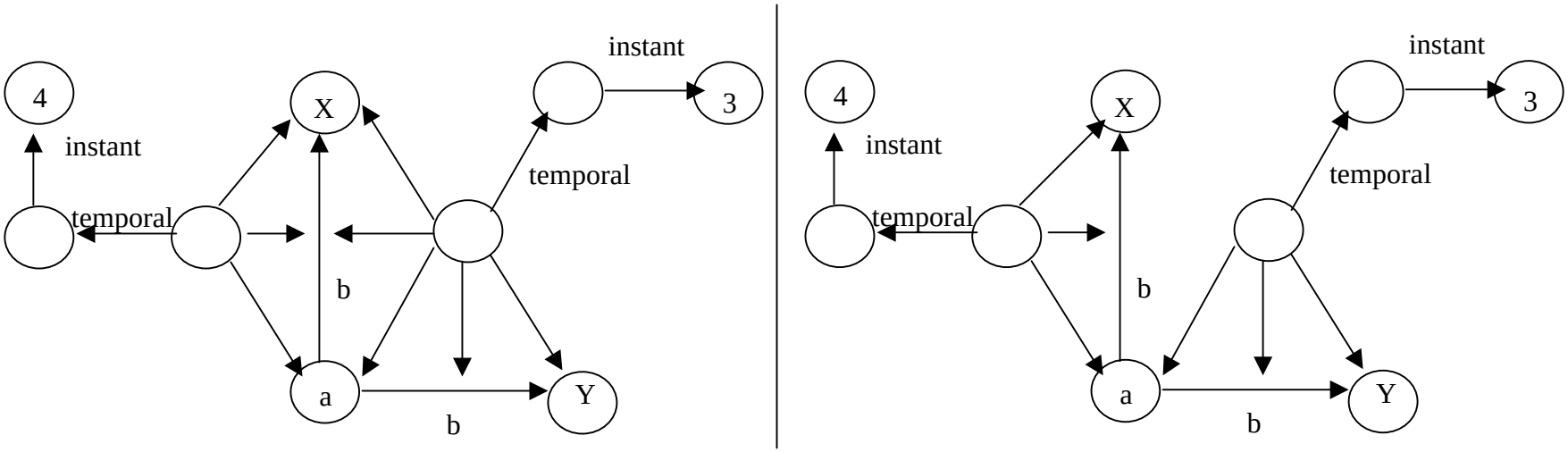
- Notion of temporal Entailment $\models_\tau$



- Treatment of temporal Blank Nodes:



Temporal RDF Intrinsic Issues



Both Graphs have equivalent snapshots at any time t; however, the graph on the LHS is not entailed by the Graph of the RHS. Note that for t=3, making X = Y both Graphs are the same.

Outline

- Motivation
- Temporal Relational Databases
  - The structure of time
  - Abstract temporal databases
  - Concrete temporal databases
  - Temporal query languages
- Beyond the relational model
  - XML
  - Temporal extensions to XML
  - RDF
  - Temporal extensions to RDF
- **Summary**

Summary

- Time is present in most real-life applications
- Usually ignored in Commercial RDBMS
- Many proposals in the last 25 years led to TSQL2
- Extensions to the relational model also suffer the same problem
- XML, RDF, OLAP data models must be extended

In memorian Alberto O. Mendelzon

(June 28th, 1951 – June 16, 2005)