Managing a Fragmented XML Data Cube with Oracle and Timesten

Doulkifli BOUKRAA, ESI, Algiers, Algeria
Omar BOUSSAID, Fadila BENTAYEB, ERIC Lab, Univ. Lyon 2 France
{omar.boussaid, fadila.bentayeb}@univ-lyon2.fr
Djamel Eddine ZEGOUR, ESI, Algiers, Algeria
Outline

- Context and Motivation
- Timesten in-Memory Database
- XML Data Cube
- XML Cube Management Configurations
- Implementation and Testing
- Related Work
- Conclusion
Context and motivation

- Performance optimization of data warehouses (DW)
- Focus on a special type of DWs: XML warehouses
- Warehousing and analyzing complex data
  - Multidimensional model: conceptual, logical, physical level
  - Performance issues
  - Vertical fragmentation approach proposed
- Crossing two techniques
  - Vertical Fragmentation
  - Caching
Objectives and Contribution

Objective:
- Analyze the impact of vertical fragmentation on caching and vice versa
- A better cache management (data organization-aware)
- Leverage the vertical fragmentation

Contributions
- A set of configurations to manage a fragmented XML cube
- A comparison between the configurations
TimesTen In-Memory database

- Oracle’s In-memory database solution
- Different uses of TimesTen
  - As a database cache for a disk resident database
    - Read-only transactions
    - Read-Write transactions
  - As a full-featured relational database
    - Persistence
    - Recovery
    - ...
XML cubeModel

- General Cube Schema

![Diagram of XML cube schema with nodes labeled Dim1_2.xml, Dim 1_1.xml, Dim 2_1.xml, Dim 4_1.xml, and Dim 3_1.xml connected to Fact.xml through edges and additional nodes labeled ...]
XML Cube Model

- Instantiation: Auction cube
XML Cube Model

- Unfragmented XML cube
  - Basic configuration
  - Fact and each dimension member = one XML document
  - Formally:
    - $\text{UXCube} = \{D_i, \ i=1, \ldots\}$ set of XML documents
    - $D_i = \{P^i_j, \ i=1, \ldots, \ j=1,\ldots\}$ set of XML properties accessed by XPath
XML Cube Model

- Fragmented XML cube
  - Vertical fragmentation approach for XML Cubes (Dawak’11)
  - Each document of the Cube split into fragments
    - Homogeneous fragments: properties ∈ same original document
    - Heterogeneous fragments: properties

Before fragmentation

After fragmentation
XML Cube Model

- Fragmented XML cube
  - Frequent Fragment: derived from a frequent property set (Association rules)
  - InFrequent Fragment: properties $\notin$ frequent property set

Formally

$$\text{FXCube} = \{\text{FF}_m, m=1, \ldots\} \cup \{\text{IF}_n, n=1, \ldots\}$$
XML Cube Model

- Fragmented XML cube: example

```
<Auctions>
  <Auction ID="..">
    <Date>...</Date>
    <CurrentPrice>...</CurrentPrice>
    ...
  </Auction>
  ...
</Auctions>

<Person>
  <Person ID="..">
    <PersonName>...</PersonName>
    <Address>...</Address>
    ...
  </Person>
  ...
</Person>

<FGs>
  <FG>
    <Auction ID="..">
      <CurrentPrice>...</CurrentPrice>
      ...
      </Auction>
      ...
    </FG>
  ...
</FGs>
```
XML Cube Management Configurations

1. Disk Database
   - Frequent Properties
   - Infrequent Properties

2. Disk Database
   - Frequent Properties
   - Infrequent Properties
   - Database Cache
   - Frequent Properties

3. In-Memory Database
   - Frequent Properties
   - Infrequent Properties

4. Disk Database
   - Frequent Fragment
   - Infrequent Fragment

5. Disk Database
   - Frequent Fragment
   - Infrequent Fragment
   - Database Cache
   - Frequent Fragment

6. In-Memory Database
   - Frequent Fragment
   - Infrequent Fragment
XML Cube Management Configurations

Instantiation: Auction cube configurations
Implementation and Testing

- Disk Resident Database: Oracle 11g Rel. 2
- Database cache and in-Memory database: Oracle TimesTen 11.2.1
- Data Set:
  - XML Cube of auctions: 6 XML document types
  - Fragmented Cube: 28 XML fragment types
- Query load: 100 analytical queries targeting different aggregation levels of UXCube
- Queries rewritten against FXCube
Implementation and Testing

- First measure: average query response times
  - Unfragmented Vs Fragmented XML Cube
Implementation and Testing

- First measure: average query response times
  - Unfragmented Vs Fragmented XML Cube
Implementation and Testing

- First measure: average query response times
  - Unfragmented Vs Fragmented XML Cube
Implementation and Testing

- First measure: average query response times
  - Disk-resident Vs Cached Vs in-Memory XML Cube
Implementation and Testing

- First measure: average query response times
  - Disk-resident Vs Cached Vs in-Memory XML Cube
Implementation and Testing

- Second measure: percentage of efficient queries
  - Unfragmented Vs Fragmented XML Cube

Disk resident configurations
Implementation and Testing

- Second measure: percentage of efficient queries
  - Unfragmented Vs Fragmented XML Cube

![Diagram](image)

- M-FXCube; 100.00%
  - In-Memory configurations
Implementation and Testing

- Second measure: percentage of efficient queries
  - Unfragmented Vs Fragmented XML Cube

![Pie chart showing caching configurations with percentages: C-FXCube 62.50%, C-UxCube 30.00%, CU=CF 7.50%]
Implementation and Testing

- Second measure: percentage of efficient queries
  - Disk-resident Vs Cached Vs in-Memory XML Cube

![Diagram showing measures of efficient queries]

- DU=MU=CU: 7.50%
- C-Uxcube: 20.00%
- M-UXCube: 70.00%

Unfragmented Cube
Implementation and Testing

- Second measure: percentage of efficient queries
  - Disk-resident Vs Cached Vs in-Memory XML Cube

![Pie chart showing proportions: DF=MF=CF: 40.00%, M-FXCube: 60.00%, fragmented Cube: 0.00%]
# Related Work (1/2)

<table>
<thead>
<tr>
<th>Category of work</th>
<th>Examples</th>
</tr>
</thead>
</table>
| **Database & Web** | • Altinel et al (2003): Static and dynamic caching  
• Manegold et al (2000): Optimizing main memory access  
• Dar et al. (1996): Semantic caching  
• Huang and Hsu (2008): Web document caching |
| **Data Warehouses** | • Andrade et al. (2007): Optimizing multiple data analysis queries  
• Deshpande et al. (1998) Cache small regions of a multidimensional space  
• Lehner et al. (2000): Dynamic caching for multidimensional data  
• Scheuermann et al. (1996): Caching small sets of query results  
• Muto & Kitsuregawa (1998): Main memory for compressed cube management  
• Ross & Zaman (2000): Cache data cube subset materialization |
| **XML** | • Yang et al. (2003): Cache frequent XML patterns  
• Mandhani & Suciu (2005): Semantic cache of materialized Xpath queries  
• Obermeier and Bottcher (2008): XML splitting over mobile devices |
Discussion

- Our work meets the same motivation of Obermeier and Bottcher (2008) but applied to XML cubes

- Combination of Vertical fragmentation, main memory data management and caching not tackled before
Conclusion and Perspectives

Conclusion

Crossed two optimization techniques of data warehouses: caching and vertical fragmentation

Benefits of fragmentation when the cube is managed in main memory

In-memory enhances both fragmented and unfragmented cube

Main memory increases the % of efficient queries
Conclusion and Perspectives

- Perspectives
  - Implement our proposals on an ad hoc network
  - Combine horizontal and vertical fragmentation with cache and in-memory management