# Context Aware Data Management in Relational Databases

Ioannis (Yannis) Roussos, Timos Sellis
National Technical University of Athens, Greece

# Motivation


normal


dark place & user is moving

| Lighting | User | Schema |
|---|---|---|
| Normal | Not moving | Bus_Schedule (BusNo, Depart_time, Depart_Station, Route_ID)<br>Bus_Route(Route_ID, …) |
| Dark | Moving | Bus_Schedule(BusNo, Depart_Abr, Near_Stop, Time) |
| Sunny | … | |

# Context

- All those external parameters that affect the information that is returned to a user as a result of a query.
- [Dey2000]:
  “Any information that can be used to characterize the situation of an entity. An entity could be a person, a place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves”
- A system is context-aware if:
  “It uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task”

# Context Categories and Types

| Category | Type | Examples |
|---|---|---|
| User-Centric Context | User's profile | Age, nationality, educational background. |
| | User's dynamic behavior | Activity, task, situation or intention. |
| | User's physiology and emotional state | Body temperature, heart rate, happiness, stress. |
| Environmental Context | Spatial information and spatial relationships | Location, orientation, distance from points of interest, traffic conditions. |
| | Temporal information | Current date or time. |
| | Physical environment | Temperature, lighting, noise level. |
| System-related Context | Device characteristics | Screen resolution, memory, computational power, operating system. |
| | Network characteristics | Network bandwidth, neighboring devices. |

- Context of the *query* or the *user* in general
  *vs.*
- Context of the *stored information (data)*

# Examples of Context aware information

- Information on Products
  - *Country* → price, brand name, description, specs differ
  - *Expertise* of the user → more or less (different) attributes
  - *Device* used to browse the information → are high res pictures and/or animations part of the information?
  - *Language* of user, *preferences*, cultural discrepancies…
- Information on Persons
  - Different countries store different information for the same person:
    - Values: Different address, phone, etc
    - Schema: In Greece we also stored religious beliefs
  - Different roles: At the same time he is {an engineer, a researcher, a member of a movie club, …}
  - Time → evolving information / schema

# More Examples …

- Restaurants
  - *According to time: café → restaurant → restaurant bar*
  - *Season  a terrace is/is not available or address changes*
  - *Attribute "distance from the beach"  has **no** meaning for Munich*
- Books
  - Paperback/Audio book/e-book
- Academic Research
  - The same piece of work published in {conference(s), journal}
  - Related book chapters, presentations, experimental results
- Schema evolution/versioning
  - What if versions are tagged and queried (altogether) using not only version number and/or temporal information?

# Context in Information Management

- The main tasks of context aware applications are to:
  1. Characterize provided information using context.
  2. When a user with a specific context C requests for some kind of information, to retrieve the appropriate information for context C, i.e. match user's context with information's context.
- Context in Relational Database Management Systems
  - Main Tasks:
  1. Characterize the context of each record (tuple).
  2. Retrieve all records for a specific input context.
  - The information stored in a record is valid only within the context where it is defined.
  - Modeling context aware information ≡ Modeling relations with *different instances* or even *different schemas* in different contexts.

# Modeling Context Aware Data

- A lot of explicit decisions that must be made from the database and the application designers:
  1. How to model and store contextual information in a database.
  2. How to model context aware relations with multiple instances in different contexts and even different relation schemas in some contexts.
  3. How to uniformly manage even simple virtual queries of the form “find me a restaurant” and dynamically generate the specific queries that will be sent to the DBMS based on the requested context or set of contexts.

# Modeling Context Aware Data (II)

4. How to manage query plans on application level.
   - Complex queries like cross-context joins or set operations over context relations.
   - Some complex queries require handling of intermediate results using procedural methods and posing of new queries to the database according to the result of these intermediate computations.

5. How to merge results from the different schemas and instances of the same relation
   - A uniform result set must be returned.

# In practice (without an appropriate context aware data model)

- Management of context and context aware data is done through procedural methods at the application layer.
- Each application uses an implementation-specific solution:
  - how context is modeled?
  - how context aware relations are represented?
- Implementing a part of the context aware data modeling and management at the application layer is:

  (i) error prone (ii) implementation specific.

  (iii) a hindrance for maintenance.

  (iv) it obstructs query optimization techniques

  (v) in the long run, results in worse system performance.

# Why not use views?

- Using the view mechanism without *inherent* support for context has a number of drawbacks:
  - The semantics of context are lost:
    - Simple key attributes with no distinction among other attributes.
    - Worst case: it is only part of the view description or stored in the data dictionary as schema information.
  - Choosing the correct view can only be done in the application layer and not through a first order query language with data scope , as for example the SQL language.
  - Queries over multiple contexts need complex view joins in order to be evaluated.
- Using the view mechanism is more appropriate as a relational *implementation* than a data model per se.

# Requirements for a context aware data model

- A context aware relation $R_c$ should be defined with respect to a context definition C and may have a *different instance* and/or *different schema* defined for each different context instance of context C.
- Operators (as well as a context aware query language) should be defined on context aware relations and not on a specific instance or schema.
- In general, results of queries over context relations are also context relations with multiple schemas and/or instances in multiple context instances.
- Additional operators must allow the reference to any instance or schema of a context aware relation and the treatment of each instance or schema as if it were a completely different relation.

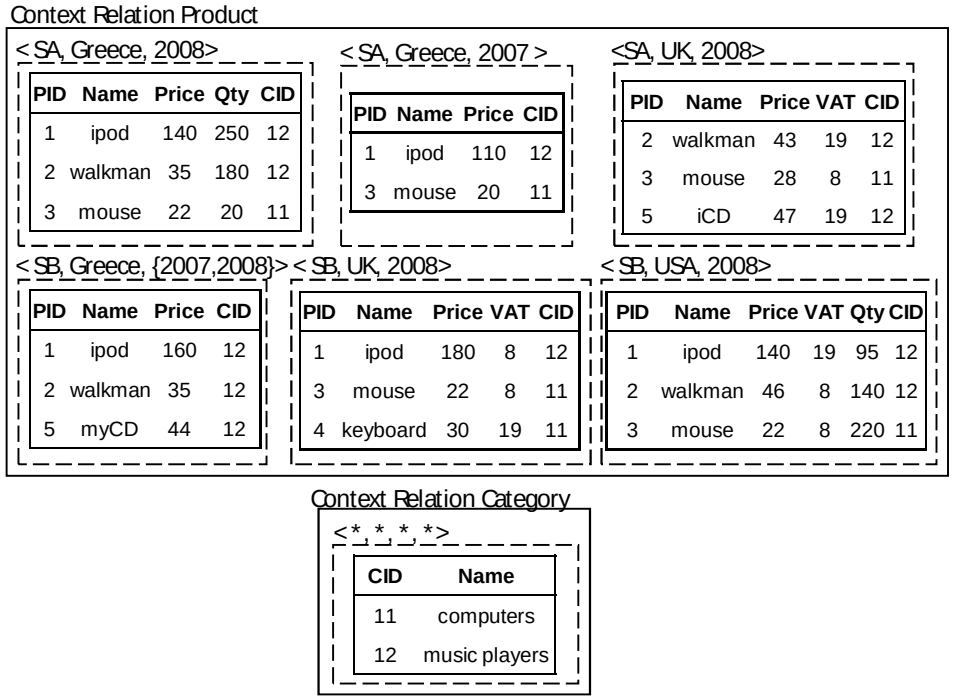
# Illustrative Example

- A worldwide electronic marketplace where retail companies and suppliers sell products.
- The system allows each individual seller to dynamically construct a structured schema for his products, catalogue, collaborating stores and other basic, predefined relations.
- Each schema can be built using some global predefined attributes with well defined semantics.
  - e.g. product name, price, delivery time, description, etc.
- A seller creates his schema by iteratively:

  (a) choosing from the global pool of relations which relation he wants to create,
  (b) adding attributes that he can find from a set of predefined attributes for this relation and
  (c) confirming predefined constraints, like “NOT NULL” or foreign keys.

# Illustrative Example (II)

- Users are characterized by a set of metadata defining their context.
- Suppliers can define many data instances for each relation, namely one for each dynamically specified context.



# Illustrative Example

Context Relation Product

< SA, Greece, 2008>

| PID | Name | Price | Qty | CID |
|---|---|---|---|---|
| 1 | ipod | 140 | 250 | 12 |
| 2 | walkman | 35 | 180 | 12 |
| 3 | mouse | 22 | 20 | 11 |

< SA, Greece, 2007 >

| PID | Name | Price | CID |
|---|---|---|---|
| 1 | ipod | 110 | 12 |
| 3 | mouse | 20 | 11 |

<SA, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 2 | walkman | 43 | 19 | 12 |
| 3 | mouse | 28 | 8 | 11 |
| 5 | iCD | 47 | 19 | 12 |

< SB, Greece, {2007,2008}>

| PID | Name | Price | CID |
|---|---|---|---|
| 1 | ipod | 160 | 12 |
| 2 | walkman | 35 | 12 |
| 5 | myCD | 44 | 12 |

< SB, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 1 | ipod | 180 | 8 | 12 |
| 3 | mouse | 22 | 8 | 11 |
| 4 | keyboard | 30 | 19 | 11 |

< SB, USA, 2008>

| PID | Name | Price | VAT | Qty | CID |
|---|---|---|---|---|---|
| 1 | ipod | 140 | 19 | 95 | 12 |
| 2 | walkman | 46 | 8 | 140 | 12 |
| 3 | mouse | 22 | 8 | 220 | 11 |

Context Relation Category

< *, *, *>

| CID | Name |
|---|---|
| 11 | computers |
| 12 | music players |

# A simple Context Definition

- A context is formally defined through the definition of a *context schema* $C_s$, i.e. the set of attributes that compose it.
- *Context instance* $C_i$ → An assignment of values to a given context schema $C_s$.
- A *context specifier* $C_p$ can store a specific context instance $C_i$ or a set of context instances $\{C_i, C_j, \ldots, C_m\}$.

| Context Schema Cs | Context Instance Ci | Context Specifier Cp |
|---|---|---|
| Cs1 = <Language> | Ci1 = <Italian> | Cp1 = {<French>, <Greek>} |
| Cs2 = <Location, Date> | Ci2 = <*, 2005> | Cp2 = {<*, 2003>, <Athens, 2004> } |
| Cs3 = <Users, Devices> | Ci3 = <david, PDA> | Cp3 = { <george, *>, <jack, PC>, <mary, PC>} |

# Context Aware Data Model

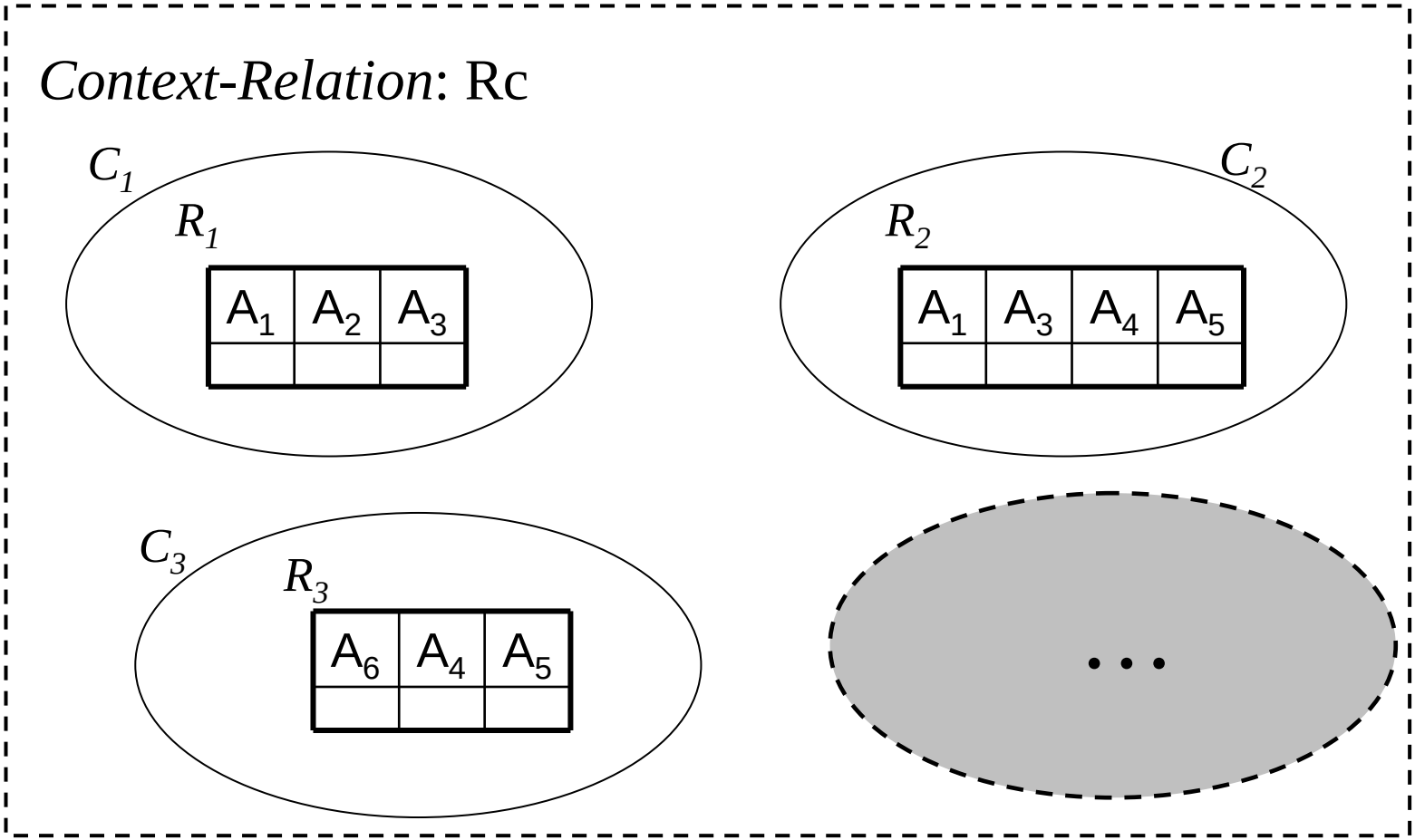
- *A **context relation schema $R_c$** is an (ordered) pair of (i) a reference to a context schema $C_s$ and (ii) a finite set of (ordered) pairs of relation schemas and context specifiers defined with respect to context schema $C_s$:*

$$R_c = < C_s, \{<R_1(A_{11}, A_{12}, \ldots, A_{1n}), C_{p1}>, <R_2(A_{21}, A_{22}, \ldots, A_{2k}), C_{p2}>, \ldots\} >$$

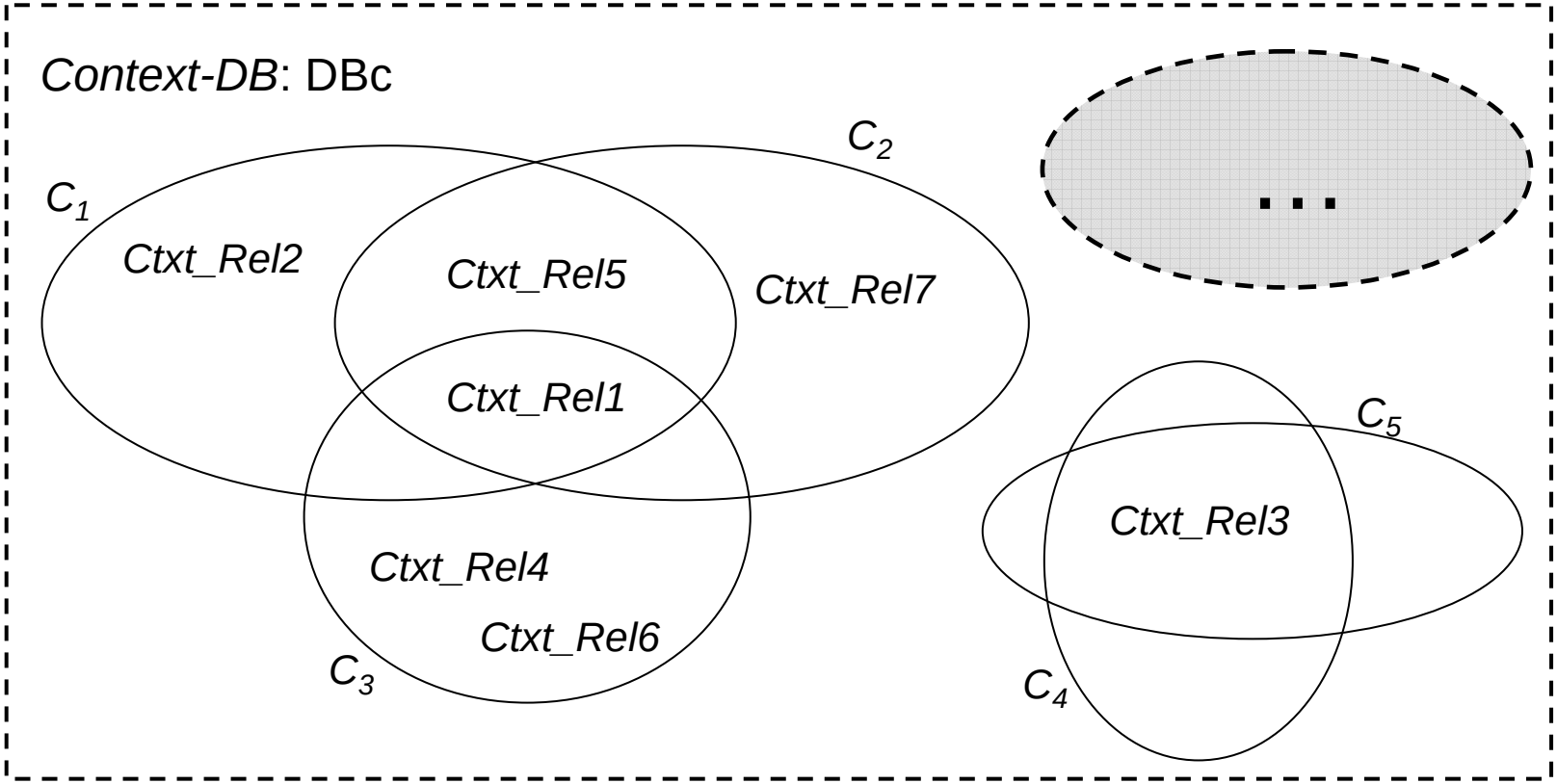
- *Context specifiers $\{C_{p1}, C_{p2}, \ldots\}$ are defined with respect to the same context schema $C_s = \{c\_att_1, c\_att_2, \ldots, c\_att_m\}$. All the defined context specifiers are disjoint, i.e. for every $C_{pi}, C_{pj} \in \{C_{p1}, C_{p2}, \ldots\}$, with $i \neq j$, $C_{pi} \cap C_{pj} = \varnothing$.*

*Second order?*

# Context Aware Data Model (II)



# Context Aware Data Model (III)



# Operations

A. Extended Relational Operations
   - Project, Select, Cartesian Product, Set Operations

B. Operations for the management of Relation schemas
   - Referencing and selecting relation schemas of a context relation.
   - Management of context schema and its instances.
   - Creating context relations from traditional non-context relations and vice versa.

C. Weak versions of {Select, Project} + non primitive operations

- Using (A, B) → our extended set of operations is *almost* downwards compatible to the relational model.

# Illustrative Example

Context Relation Product

< SA, Greece, 2008>

| PID | Name | Price | Qty | CID |
|---|---|---|---|---|
| 1 | ipod | 140 | 250 | 12 |
| 2 | walkman | 35 | 180 | 12 |
| 3 | mouse | 22 | 20 | 11 |

< SA, Greece, 2007 >

| PID | Name | Price | CID |
|---|---|---|---|
| 1 | ipod | 110 | 12 |
| 3 | mouse | 20 | 11 |

<SA, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 2 | walkman | 43 | 19 | 12 |
| 3 | mouse | 28 | 8 | 11 |
| 5 | iCD | 47 | 19 | 12 |

< SB, Greece, {2007,2008}>

| PID | Name | Price | CID |
|---|---|---|---|
| 1 | ipod | 160 | 12 |
| 2 | walkman | 35 | 12 |
| 5 | myCD | 44 | 12 |

< SB, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 1 | ipod | 180 | 8 | 12 |
| 3 | mouse | 22 | 8 | 11 |
| 4 | keyboard | 30 | 19 | 11 |

< SB, USA, 2008>

| PID | Name | Price | VAT | Qty | CID |
|---|---|---|---|---|---|
| 1 | ipod | 140 | 19 | 95 | 12 |
| 2 | walkman | 46 | 8 | 140 | 12 |
| 3 | mouse | 22 | 8 | 220 | 11 |

Context Relation Category

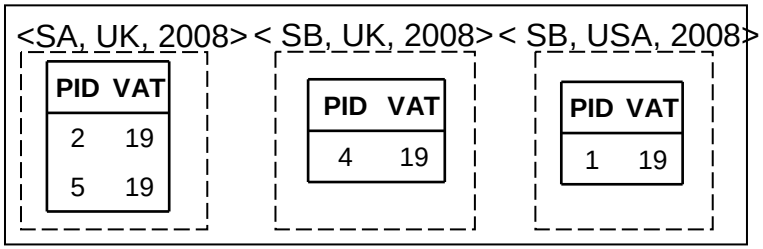
< *, *, *>

| CID | Name |
|---|---|
| 11 | computers |
| 12 | music players |

# Extended Relational Operations

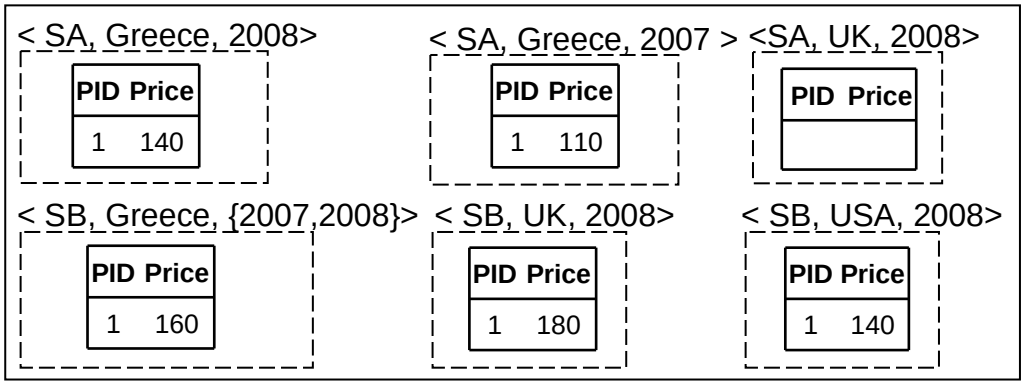
- *Project*: $\pi_{\{A1, \ldots, An\}} R_c \rightarrow R_c'$
  - $\pi_{\{VAT, Qty\}}$Product → < Result(VAT , Qty), <SB, USA, 2008 > >.
- *Select*: $\sigma_{CONDITIONS} R_c \rightarrow R_c'$
  - Result = $\pi_{\{PID,VAT\}} (\sigma_{(VAT > 10)}$Product)

Context Relation Result




- *Cartesian Product*: $R_c \times S_c \rightarrow T_c$
  - CartResult = $\pi_{PID, Price} (\sigma_{(Price > 50)}$ AND (Category.Name = "music players") AND AND (Product.CID = Category.CID) (Product × Category ) )

Context Relation CartResult



# Extended Relational Operations

- *Set Operations*: $R_c \{\cup, \cap, -\} S_c \rightarrow T_c$

| $C_s$ | $C_1$ | $C_2$ | $C_3$ |
|---|---|---|---|
| *Input Context Relations* | | | |
| $R_c$ | $R_1$ | $R_1$ | |
| $S_c$ | | $S_1$ | $S_2$ |
| ***Result*** | | | |
| *Strong Set Operations (Relational Definition)* | | | |
| $Op=\{\cup,\cap,-\}_{strong}$ | | $R_1$ Op $S_1$ | |
| *Context Aware Set Operations* | | | |
| $\cup$ | $R_1$ | $R_1 \cup S_1$ | $S_2$ |
| $\cap$ | | $R_1 \cap S_1$ | |
| $-$ | $R_1$ | $R_1 - S_1$ | |

# Management of Relation schemas

- *Schema_Select*: $\sigma^{c}_{\text{CONDITIONS}} R_c \rightarrow R_c'$
  - Conditions over:

    (i) context specifiers or
    (ii) relation schema definitions, e.g. 'VAT is defined'.
  - Return products available only in 2007:

    $\sigma^{c}_{\text{(Date = 2007)}}$ Product
  - $\sigma^{c}_{\text{(Location = Greece) AND (Qty is defined)}}$ Product → context relation with the single relation schema for < SA, Greece, 2008>

- *De-Contextualization*: deCxt ($R_c$) → R
- *Contextualization*: Cxt (R, $C_s$, $C_p$) → $R_c$ (=<$C_s$, {<R, $C_p$>}>)

# Management of Relation schemas

- *Map*: Map ($R_c$, c_att, ctxt_expression) → $T_c$
  - Alter the values of the context attributes of each context instance in $R_c$
- *Add Context Attribute*: Add_Cxt_attr ($R_c$, c_att = *c_val*)
- *Delete Context Attribute*: Del_Cxt_attr ($R_c$, c_att )

- Result = Add_Cxt_attr( Del_Cxt_attr( π{PID,Name} Product, Supplier), Device='PC')

Context Relation Result

<Greece, 2008, PC>

| PID | Name |
|---|---|
| 1 | ipod |
| 2 | walkman |
| 3 | mouse |
| 5 | myCD |

<Greece, 2007, PC >

| PID | Name |
|---|---|
| 1 | ipod |
| 2 | walkman |
| 3 | mouse |
| 5 | myCD |

<UK, 2008, PC >

| PID | Name |
|---|---|
| 1 | ipod |
| 2 | walkman |
| 3 | mouse |
| 4 | keyboard |
| 5 | iCD |

<USA, 2008, PC>

| PID | Name |
|---|---|
| 1 | ipod |
| 2 | walkman |
| 3 | mouse |

# Non Primitive Operations

- *Merge*: Merge ($R_c$, $R_i$, $R_j$) → $T_c$
- *Split*: Split ($R_c$, $R_i$, $C_{i1}$, $C_{i2}$) → $T_c$
- *Cross Context Cartesian Product*: $R_c.R_i \times^c S_c.S_j \rightarrow T_c.T_1$

Context Relation Result1

<{SA,SB}, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 1 | ipod | 180 | 8 | 12 |
| 2 | walkman | 43 | 19 | 12 |
| 3 | mouse | 28 | 8 | 11 |
| 3 | mouse | 22 | 8 | 11 |
| 4 | keyboard | 30 | 19 | 11 |
| 5 | iCD | 47 | 19 | 12 |

Context Relation Result2

<SA, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 1 | ipod | 180 | 8 | 12 |
| 2 | walkman | 43 | 19 | 12 |
| 3 | mouse | 28 | 8 | 11 |
| 3 | mouse | 22 | 8 | 11 |
| 4 | keyboard | 30 | 19 | 11 |
| 5 | iCD | 47 | 19 | 12 |

<SB, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 1 | ipod | 180 | 8 | 12 |
| 2 | walkman | 43 | 19 | 12 |
| 3 | mouse | 28 | 8 | 11 |
| 3 | mouse | 22 | 8 | 11 |
| 4 | keyboard | 30 | 19 | 11 |
| 5 | iCD | 47 | 19 | 12 |

# Illustrative Example

Context Relation Product

< SA, Greece, 2008>

| PID | Name | Price | Qty | CID |
|---|---|---|---|---|
| 1 | ipod | 140 | 250 | 12 |
| 2 | walkman | 35 | 180 | 12 |
| 3 | mouse | 22 | 20 | 11 |

< SA, Greece, 2007 >

| PID | Name | Price | CID |
|---|---|---|---|
| 1 | ipod | 110 | 12 |
| 3 | mouse | 20 | 11 |

<SA, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 2 | walkman | 43 | 19 | 12 |
| 3 | mouse | 28 | 8 | 11 |
| 5 | iCD | 47 | 19 | 12 |

< SB, Greece, {2007,2008}>

| PID | Name | Price | CID |
|---|---|---|---|
| 1 | ipod | 160 | 12 |
| 2 | walkman | 35 | 12 |
| 5 | myCD | 44 | 12 |

< SB, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 1 | ipod | 180 | 8 | 12 |
| 3 | mouse | 22 | 8 | 11 |
| 4 | keyboard | 30 | 19 | 11 |

< SB, USA, 2008>

| PID | Name | Price | VAT | Qty | CID |
|---|---|---|---|---|---|
| 1 | ipod | 140 | 19 | 95 | 12 |
| 2 | walkman | 46 | 8 | 140 | 12 |
| 3 | mouse | 22 | 8 | 220 | 11 |

Context Relation Category

< *, *, *>

| CID | Name |
|---|---|
| 11 | computers |
| 12 | music players |

# Evaluation Example

- Supplier SA has realized that during the last years his profits in products that cost less than 50 Euros have been significantly reduced.
- Main reason → many customers in UK buy the products from abroad, especially from countries where he doesn't sell those products.
- So, he wants to find for each country and year, those products that cost less than 50 Euros, are offered by at least one supplier but not by him and cost less than his products for UK in the same year.

# Evaluation Example

- The first step is to find all products for each country and year that are sold only by other suppliers and cost less than 50 Euros :
  - $R_{SA} = \text{Del\_Cxt\_attr}\ (\pi_{PID}\ (\ \sigma^{c}_{(Supplier = SA)}\ Product), Supplier)$
  - $R_{NotSA} = \text{Del\_Cxt\_attr}\ (\pi_{\{PID,\ Price\}}(\ \sigma^{c}_{(Supplier \neq SA)}\ Product), Supplier)$
  - $R_{others} = \sigma^{c}_{(Country \neq UK)}\ (\ \pi_{PID}\ (R_{NotSA})\ -\ R_{SA}\ )$
  - $R_{Tmp1} = \sigma_{(Price < 50)}\ (\ (R_{NotSA}\ \times\ R_{others})\ )$

Context Relation $R_{Tmp1}$

<Greece, 2007 >

| PID | Price |
|---|---|
| 2 | 35 |
| 5 | 44 |

<Greece, 2008>

| PID | Price |
|---|---|
| 5 | 44 |

<USA, 2008>

| PID | Price |
|---|---|
| 3 | 22 |

# Evaluation Example

- The next step is to isolate all relation schemas of supplier SA in UK in order to join them with $R_{Tmp1}$:

$$R_{Tmp2} = \text{Map}(\ \text{Del\_Cxt\_attr}(\ \pi_{\{PID,\ Price\}}(\sigma^{c}_{(Supplier = SA)\ AND\ (Location = UK)}\ Product)\ ,\ Supplier),\ Location,\ *)$$

- Context attribute Location was mapped to '*' so that information for UK will be joinable with that of other countries for the same year.

Context Relation $R_{Tmp2}$

<*, 2008>

| PID | Price |
|---|---|
| 2 | 43 |
| 3 | 28 |
| 5 | 47 |

# Evaluation Example

- Finally, using the temporary results $R_{Tmp1}$ and $R_{Tmp2}$, he is able to retrieve the products that are the answer to his analysis query:

$$\text{Result} = \pi_{\text{RTmp1.PID, RTmp1.Price}} \left( \sigma_{\text{(RTmp1.Price < RTmp2.Price) AND (RTmp1.PID = RTmp2.PID)}} (R_{Tmp1} \times R_{Tmp2}) \right)$$

Context Relation $R_{Tmp1}$

<Greece, 2007 >

| PID | Price |
|---|---|
| 2 | 35 |
| 5 | 44 |

<Greece, 2008>

| PID | Price |
|---|---|
| 5 | 44 |

<USA, 2008>

| PID | Price |
|---|---|
| 3 | 22 |

Context Relation $R_{Tmp2}$

<*, 2008>

| PID | Price |
|---|---|
| 2 | 43 |
| 3 | 28 |
| 5 | 47 |

Context Relation Result

<Greece, 2008>

| PID | Price |
|---|---|
| 5 | 44 |

<USA, 2008>

| PID | Price |
|---|---|
| 3 | 22 |

# Evaluation Example

- By merging all the operations presented → We are able to write the data analysis query of supplier SA in a *single expression* that uses our proposed operations.
- Moreover, this expression is *always* the same, regardless of the number of relation schemas defined in our context aware database.
  - Even if we had information about products in 100 countries and for the last 20 years, the query would not have to change.
- If we have used a pure relational implementation and relational algebra:
  - We would have to externally process the intermediate results using a procedural language and pose multiple queries to the database.
  - Any query created would be
  1. Valid only for the specific set of relation schemas defined
  2. Dependent on the number of countries or years defined in the database.

# Future Work

- Query languages over the context aware data model:
  - Extended SQL
  - Realize a non-intrusive implementation that would require minimal additions to the query engine of a traditional relational Database Management System.
- Implementation of the proposed model:
  - Implementation of context aware relations over the relational model.
  - Native storage scheme: Specialized index for context aware data.
- Query Optimization: identify the best query plans and optimize queries expressed in the extended SQL.
- Constraints/mappings over context aware relations.

*Questions ?*

# Weak Versions of Select and Project

- In order to fetch as much data as possible, we may also want a second interpretation of relational operations that will allow us to evaluate a given query even in those schemas that is not considered valid by strict relational algebra evaluation rules.
- Solution: a 4-valued logic over {TRUE, FALSE, NULL, NDF} by extending the 3-valued logic for nulls.

| A | op | B | Result |
|---|---|---|---|
| NDF | {AND, OR} | NDF | NDF |
| NDF | NOT | | NDF |
| NDF | AND | TRUE | NDF |
| NDF | AND | FALSE | FALSE |
| NDF | AND | NULL | NDF |
| NDF | OR | TRUE | TRUE |
| NDF | OR | FALSE | NDF |
| NDF | OR | NULL | NULL |

# Weak Versions of Select and Project

- *Force Evaluation Project*: $\pi^F_{\{A1, \ldots, An\}} R_c \rightarrow R_c'$
  - $\pi^F_{\{VAT, Qty\}}$ Product →

    { <Result(Qty), <SA, Greece, 2008 > >, <Result(VAT), <SA,UK,2008> >,
    <Result(VAT),<SB,UK,2008> >, <Result(VAT , Qty), <SB, USA, 2008> > }
- *Force Evaluation Select*: $\sigma^F_{CONDITIONS} R_c \rightarrow R_c'$
  - Return all products except those that have both a high VAT, i.e. more than 10%, and a small available quantity (Qty), i.e. less than 200 available for sale products: Result = $\sigma^F_{NOT\ (\ (VAT > 10)\ AND\ (Qty < 200))}$ Product

Context Relation Result

< SA, Greece, 2008>

| PID | Name | Price | Qty | CID |
|---|---|---|---|---|
| 1 | ipod | 140 | 250 | 12 |

<SA, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 3 | mouse | 28 | 8 | 11 |

< SB, UK, 2008>

| PID | Name | Price | VAT | CID |
|---|---|---|---|---|
| 1 | ipod | 180 | 8 | 12 |
| 3 | mouse | 22 | 8 | 11 |

< SB, USA, 2008>

| PID | Name | Price | VAT | Qty | CID |
|---|---|---|---|---|---|
| 2 | walkman | 46 | 8 | 140 | 12 |
| 3 | mouse | 22 | 8 | 220 | 11 |

# Extended SQL - DDL

```
CREATE CONTEXT SCHEMA Cs {
  Varchar(50) Supplier,
  DateTime Date,
  Varchar(20) Device
};
CREATE CONTEXT RELATION Product    // Or Polymorph
UNDER Cs
IDENTIFIED BY (Integer PID);

CREATE SCHEMA IN Product  {              // Or Morph
  Name          Varchar(20) NOT NULL,
  Price         FLOAT       NOT NULL,
  DeliveryTime  INTEGER,
   …   }
FOR  <'SupplierA', 2008, 'PC'>;

CREATE SCHEMA [name_of_schema] IN Product {…} FOR <….>;
…
```

# Extended SQL - Queries

- Return the ID and price of all the products that cost less than 50 Euros in 2008 from relation schemas where delivery time is not defined

```
SELECT   P.PID, P.Price
FROM     Product P
WITH     (P::DATE = 2008) AND (P.DeliveryTime NOT Defined)
WHERE  (P.Price < 50)
```

Context Relation Q1_Result

<SA, 2008, PC>

| PID | Price |
|---|---|
| 2 | 35 |
| 3 | 22 |

<SA, 2008, PDA>

| PID | Price |
|---|---|
| 2 | 35 |
| 3 | 25 |
| 5 | 47 |

- Return the product ID and price of Products in the category “music players” that were sold in France in 2007 and cost more than 40 Euros”

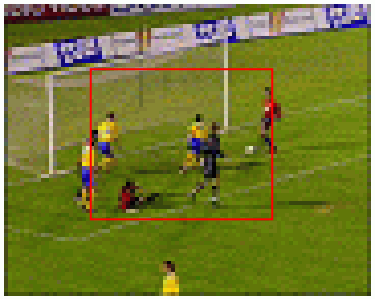
```
SELECT    P.PID, P.Price
FROM      Product P, Category C
WITH      (P::Location = ‘France’) AND (P::DATE = 2007)
WHERE    (C.Name = ‘music players’) AND (C.C_ID = P.C_ID)
              AND (P.Price > 40)
```

# More Examples …

- Efficient UI


normal


dark place & user is moving



big display

small display