Introduction to Distributed Databases

I. Basic Concepts and Notions

II. ***Distributed DB Design***

III. Distributed Query Processing

PSU - Master - Hameurlain@irit.fr

II. Distributed Database Design

II. 1. Design Approach

Top-Down Design Process

- Design a Global Schema
- Distribute the Global Schema objects to obtain Local Schemas
- ➔ Definition of **Homogeneous Distributed Database** : *"DDB in which each local DB is managed by the same DBMS";* Gardarin et al. 1989.
- ➔ **Advantages**
- The growing ease of incremental business : By the facility of adding a new site
- Increases performance: Exploit the capacities of parallel treatments/processing

Bottom-Up Design Process

- Consists of integrating existing LS in one or more Global Schemas
- Allows the integration of existing local Database into a **Federated Database**
- Needs a semantic reconciliation of schemas (data type,
- Ensures the continuity of services
  - Fragmentation & allocation are imposed

➔ Definition of Heterogeneous Distributed DB: *"Distr. DB in which the local databases are managed by different DBMS" ;* Gardarin et al. 1989

II.2. Fragmentation Strategies

- **Fragmentation (Definition):**

  Sub-relation obtained by selection of tuples and attributs from a global relation. Each fragment has a subset of the tuples of the relation.

  ➔ Two possible ways to divide a relation : *Horizontally* (selection of tuples*), Vertically* (selection of attributs). In addition, "there is possibility of nesting a fragments in hybrid fashion"

- **Fragmentation Rules:** the main objective is to preserve the semantic consistency of the DDB

  1.: Decomposition without loss of information (R1, R2, … Rn)
  2.: No duplication: restriction predicates define disjoint sets of tuples
  3.: Reconstruction rule (global relation) : R = Function (R1, R2, …Rn)

II.2. Fragmentation Strategies

Primary Horizontal Fragmentation PHF

- PHF partitions a relation along its tuples. It is defined by a selection operation on the the relation : Ri= $\sigma_{Fi}(R)$, 1<= i <= k where Fi is the selection formula used to obtain fragment Ri (also called fragmentation predicate)

- Wines (<u>w_Id</u>, vineyard, vintage, Area) ∈ GS
  - Wines1= $\sigma_{Area=Bourgogne}(Wines)$
  - Wines2= $\sigma_{Area\neq Bourgogne}(Wines)$

3: Reconstruction rule (global relation) : Wines = Wines1 ∪ Wines2

Benefits

- Favors the selection Queries by limiting the number of accessed fragments

II.2. Fragmentation Strategies: *Derived Horizontal Fragment*

- Objective: The PHF can also be useful for processing of join queries. We need to modify the HF definition By applying Semi-join predicates
- DHF Def.: Partitioning function of a relation into sub-sets of tuples, each being defined by a semi-join operation of the relation with a fragment of another relation.
- Example: Wines (w_Id, vineyard, vintage, area) , Drink (w_Id, drinker_Id, date, quantity)
  - Drink1= Drink $\propto$ Wines1
  - Drink2= Drink $\propto$ Wines2
  - Reconstruction Rule : Drink = Drink1 $\cup$ Drink2
- Exercice: Select * from W, D where W.w_Id= D.w_Id and area= «Bourgogne »
- DHF Benefits?
  - Improves joins between pairs of fragments Wines1 ∞ Drink1 and Wines2 ∞ Drink2

II.2. Derived Horizontal Fragmentation (2)

Wines1	w_Id	vineyard	vintage	aera
	1	Chablis	2010	Bourgogne
	2	Pommard	2009	Bourgogne

Wines2	w_Id	wineyard	vintage	aera
	3	Juliennas	2010	Beaujolais
	4	Chinon	2011	Loire

Drink1	drinker_Id	wineId	date	quantity
	1	1	14/02/2012	1
	1	2	1/1/2013	2

Drink2	drinker_Id	wineId	date	quantity
	1	3	14/02/2013	3
	3	4	1/1/2013	2

II.2. Fragmentation Strategies: *Vertical fragmentation*

- VF Defintion: «The VF function distributes a relation on projection attributes: fragments are obtained by projections applyed to a logical relation ».
- The reconstruction operator for VF : « is the join of the fragments on the common attributes »
- Example: Wines (w_Id, vineyard, vintage, area, degree, price) ∈ GS
  - Wines1= $\pi_{w\_Id,\ vineyard,\ area,\ vintage}$(Wines)
  - Wines2= $\pi_{w\_Id,\ degree,\ price}$ (Wines)
  - Reconstruction rule: Wines = Wines1 ∞ Wines2
- VF Benefits:
  - Favors project queries by limiting the number of accessed fragments.

II.2. Fragmentation Strategies: *Hybrid fragmentation*

- Wines (<u>w_Id</u>, vineyard, vintage, aera, degree, price) $\in$ GS
  - Wines1= $\pi_{w\_Id,\ degree,\ price}$ (Wines)
  - Wines2 = $\sigma_{area=bourgogne}(\pi_{w\_Id,\ wineyard,\ vintage,\ aera}$ (Wines))
  - Wines3 = $\sigma_{aera\neq bourgogne}(\pi_{w\_Id,\ wineyard,\ vintage,\ aera}$ (Wines))

- Reconstruction rule: Wines = Wines1 ∞ (Wines2 $\cup$ Wines3)
- Benefits?
  - Benefits of horizontal & vertical fragmentations!

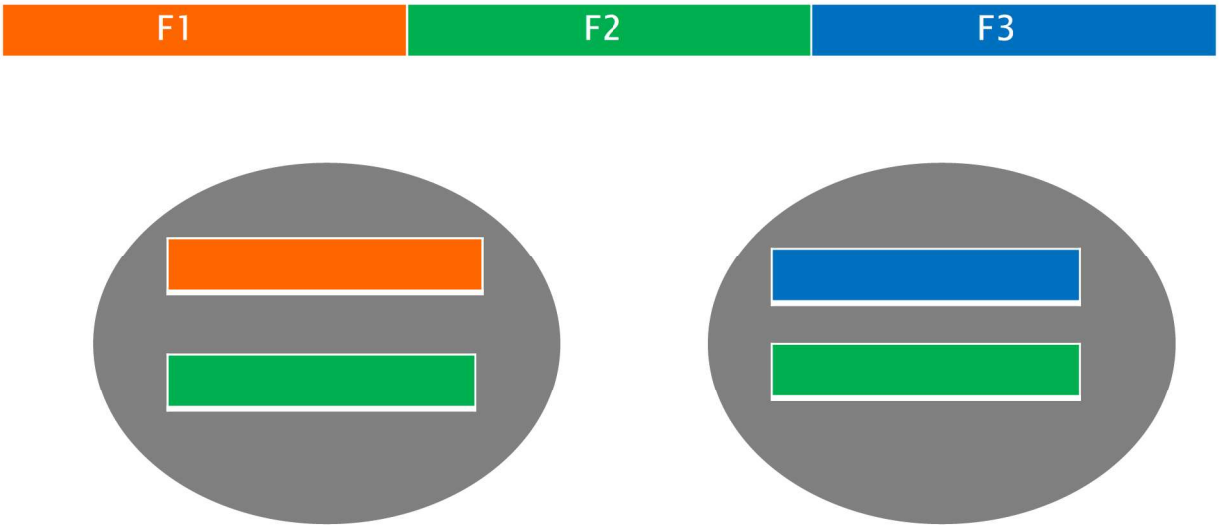
II.3. Choice of a fragmentation Strategy

- Complex issue because it depends on access requirements of queries
  - Classification of Queries:
    - Repetitive Queries
    - Ad-hoc Queries
  - Query Types: < Simple, Medium, Complex>
  - Query Natures: <Selection, Projection, Join>

II.4. Allocation of fragments (1/2)

- Logically group data in accordance with the access objectives of applications ⇒ performance
  - Ex: Wines1 & Drink1 will be grouped on the same site
- Data Placement
  - to allocate fragments on sites

II.4. Fragment allocation (2/2)



Site1 Site2

Exercice (will be done in the next tutorial)

- We consider the following Global Relational Schema GRS:

Cars (**regNum**, Brand(=marque), type, power, renting_agency, price)
Customers (**customerId**, name, firstname, address, phone)
Emp (**empId**, name, firstname, address, renting_agency, salary, commi.)
Rent (regNum, customerId, date, discount).

- Hypothesis: We consider 3 agencies : <Pau, Agen, Toulouse>
- Constraints:
  - Each car is managed by a single agency.
  - Each agency manages its sales performance and issues invoices to its customers.
  - Each employee is assigned to an agency, his salary and his commissions managed only by Toulouse.
- Questions :
  - Fragments ?
  - Fragments' allocation?
  - Reconstruction rules?