Entrepôts de données

Deuxième partie

Thierry Hamon

Bureau H202
Institut Galilée - Université Paris 13
&
LIMSI-CNRS
hamon@limsi.fr
https://perso.limsi.fr/hamon/Teaching/P13/DWH-AIR3-2017-2018/

AIR3 – DWH

Introduction

- Entrepôt de données : stockage de données volumineuses, homogènes, exploitables, multidimensionnelles, consolidées
- Comment exploiter et analyser ces données ?
- Processus standard
  - Exécution des requêtes OLTP sur les données sources
  - Mise à jour régulière de l'entrepôt
  - Exécution des requêtes OLAP pour analyser les données de l'entrepôt
    - Résumer
    - Consolider
    - Observer
    - Appliquer des formules statistiques
    - Synthétiser des données selon plusieurs dimensions

OLTP vs. OLAP

Rappel

- OLTP (*On Line Transaction Processing*)
  - Applications: Applications opérationnelles
    Traitements factuels sur les produits, les ressources ou les clients
  - Exécution sur les données sources
- OLAP (*On Line Analytical Processing*)
  - Applications : Applications d'aide à la décision
    Traitements ensemblistes, réduction d'une population à une valeur ou un comportement
  - Exécution sur les données consolidées, agrégées

OLAP

Remarques

- OLAP : ensemble des moyens et techniques utilisés dans les systèmes d'aide à la décision efficaces
- Réalisation de traitements semi-automatiques définis et mis en œuvre par les décideurs
  - Interrogation
  - Synthèse des données
  - Visualisation
- *On-Line* : la réponse doit être obtenue quasi-instantanément

OLTP vs. OLAP

Conception	Caractéristiques Orientation	OLTP Transaction	OLAP Analyse
	Conception	Entité-Relation	Etoile/flocon
Données	Granularité Nature Actualisation Taille	Détail Relationnelle Actualisées, mises à jour 100 Mo/Go	Résumées, agrégées Multidimentionnelle Historisées, recalculées 100 Go/To
Traitements	Unité de travail Accés Nb de tuples accédés Métrique	Transaction simple Lecture/écriture Dizaines Débit de transactions	Requête complexe Lecture Millions Temps de réponse
Utilisateurs	Utilisateur opérationnel Nombre d'utilisateurs	Agent Milliers	Analyste/décideur Centaines

Analyse des données

Exemple

Structure de l'Entrepôt :

- table de faits

```
ventes(codeProduit, date, vendeur, montant)
```

- tables de dimension

```
produits(codeProduit, modèle, couleur)
vendeurs(nom, ville, département, état, pays)
temps(jour, semaine, mois, trimestre, année)
```

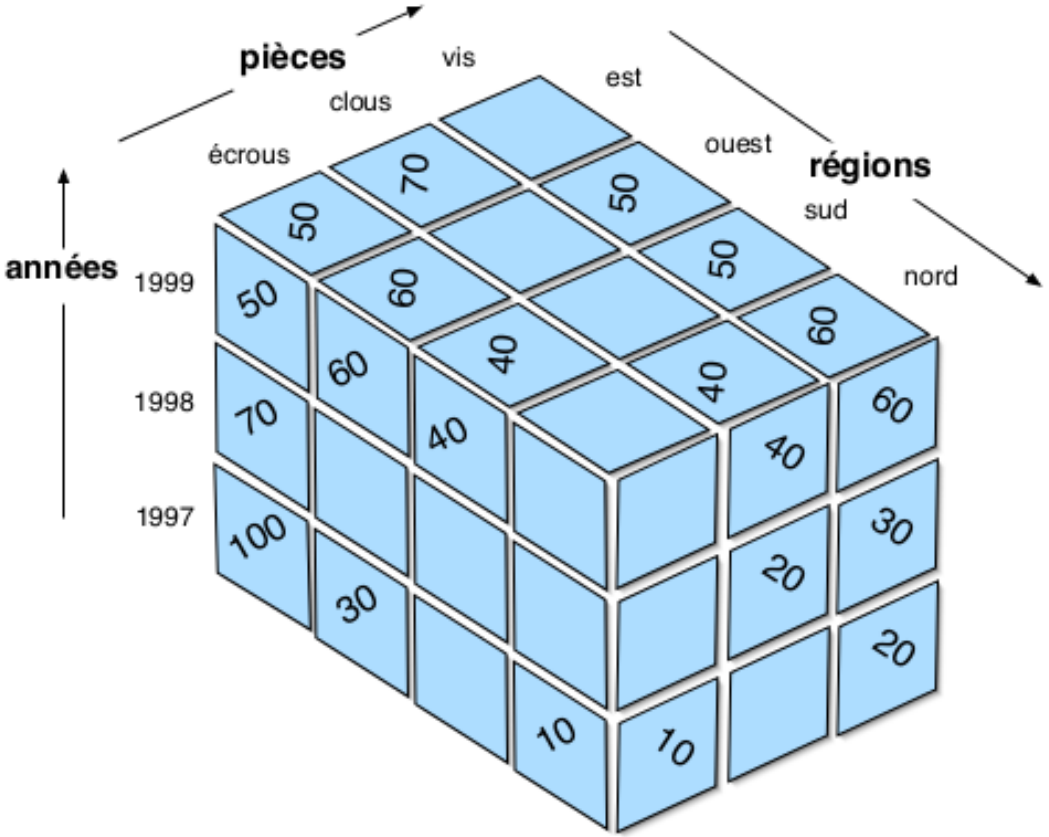
Analyse des ventes de divers produits

- Quels sont les produits dont les ventes ont chuté l'an dernier ?
- Quelles sont les quinze meilleures ventes par magasin et par semaine durant le premier trimestre de l'année 2001 ?
- Quelle est la tendance des chiffres d'affaire (CA) par magasin depuis 3 ans ?
- Quelles prévisions peut-on faire sur les ventes d'une catégorie de produits dans les 6 mois à venir ?

Problématique de l'OLAP

- Exécution de requêtes sur des BD de plusieurs Go
- Besoins spécifiques
  - Langages de manipulation
  - Organisation des données
  - Fonctions d'agrégation
  - ...
- Organiser les données de manière similaire aux abstractions de l'analyste
  - Plusieurs dimensions
  - Différents niveaux de détail
  - Vue d'ensemble
  - La donnée : point dans l'espace associé à des valeurs (cube OLAP)

Exemple de cube des ventes



Donnée : *quantité de vente d'une pièces, pendant une année, dans une région donnée*

Opérations élémentaires OLAP

Différents catégories d'opérations OLAP :

- Opérations de restructuration : `rotate`, `switch`, `split`, `nest`, `push`, `pull`
- Opérations de granularité : `roll-up`, `drill-down`
- Opérations ensemblistes : `slide`, `dice`, `jointure(drill-across)`, `data cube`

Manipulations OLAP :

- Modèles et langages pour l'OLAP
- Les règles de Codd pour les produits OLAP
- Problématique de la modélisation logique d'un ED

Catégories d'opérations OLAP

Opérations de restructuration

- Manipulation de la représentation, changement de points de vue (dimension)
- Opérations sur le cube liées :
  - Structure : `Rotate/pivot`
  - Manipulation : `Switch`
  - Visualisation : `Split`, `nest`, `push`,`pull`

Catégories d'opérations OLAP

Opérations de granularité

- Changement de niveau de détail
- Opérations liées aux hiérarchies des données :
  - `roll-up`,
  - `drill-down`

Catégories d'opérations OLAP

Opérations ensemblistes :

- Extraction des données et OLTP classique :
  - `slice, dice`
  - `selection`
  - `projection`
  - `jointure(drill-across)`

Opérations de restructuration

- Objectifs :
  - Changement de points de vue
  - Réorientation selon différentes dimensions de la vue multidimensionnelle
- Opérations :
  - Réorientation :
    - sélection graphique
    - flexibilité du schéma
    - membres complexes
    - symétrie membres/mesures
  - Manipulations :
    - bijectives
    - relatives
    - à niveau d'information constant

(`Rotate/pivot`, `Switch`, `Split`, `nest`, `push`,`pull`)

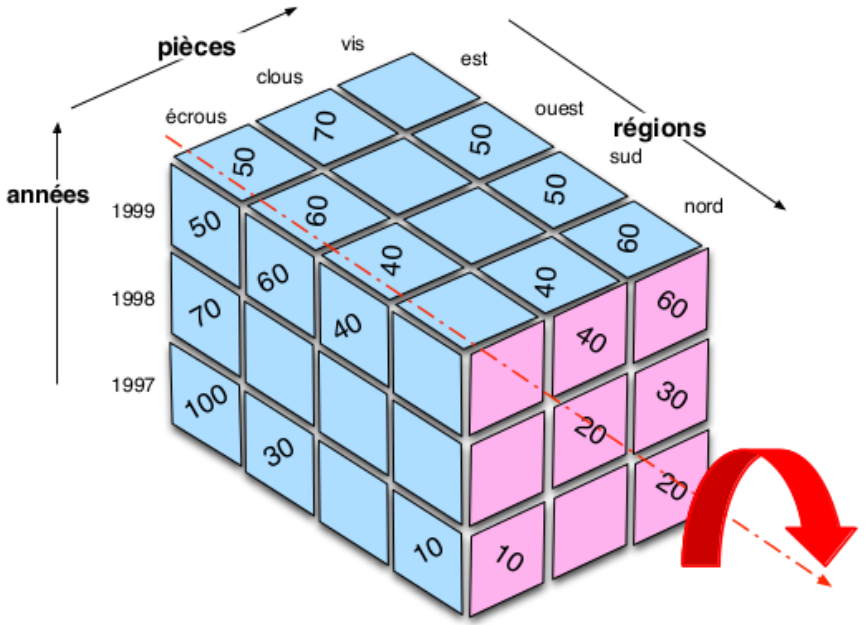
Opérations de restructuration

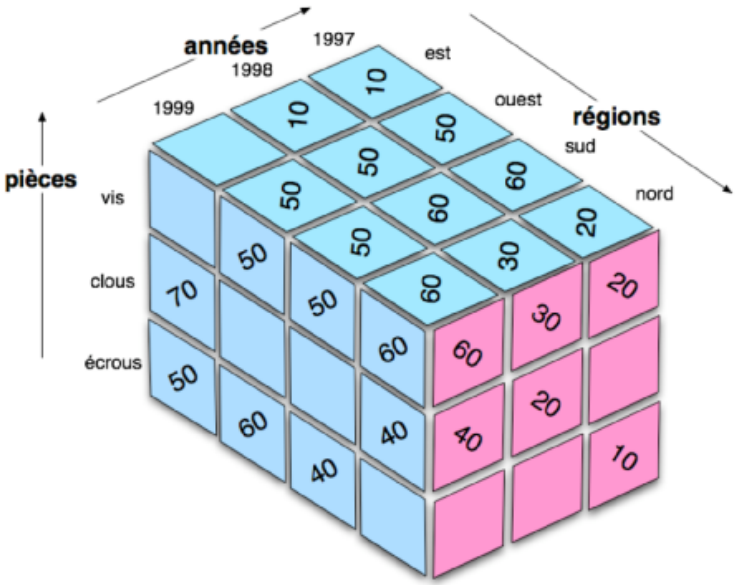
- `rotate`/pivot : Sélection de faces (et non des membres)
- `switch`/permutation : inter-changer la position des membres d'une dimension
- `split`/division : présentation de chaque tranche sous forme d'une table
- `nest`/emboîtement : imbrication des membres à partir du cube
- `push`/enfoncement : combiner les membres d'une dimension aux mesures du cube

rotate/pivot

rotate/pivot : sélection de faces

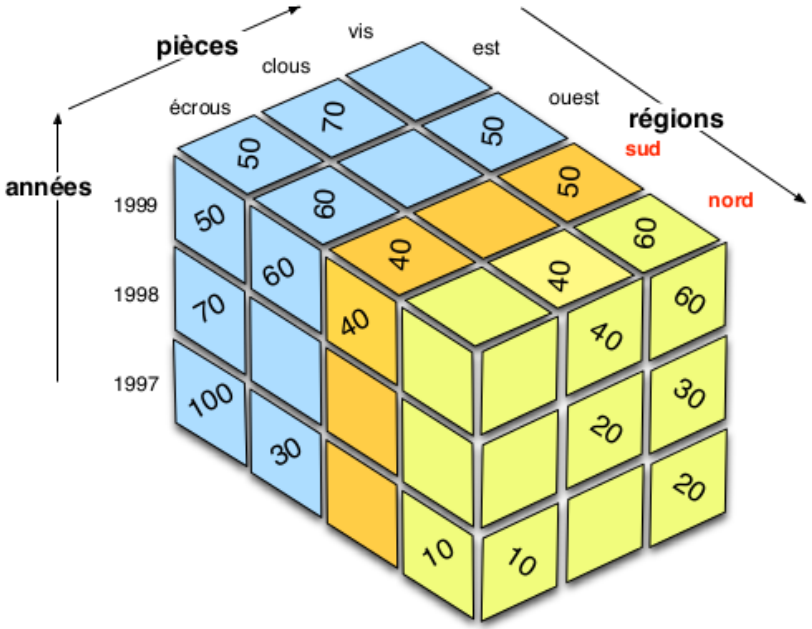
- Rotation du cube autour d'un de ses 3 axes passant par le centre de 2 faces opposées
- Présentation d'un ensemble de faces différent

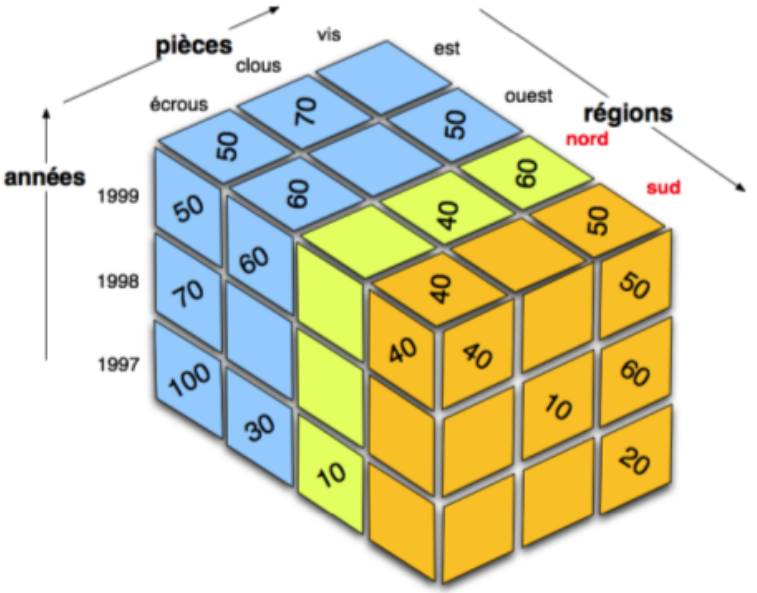




Opération de restructuration switch

`switch` ou permutation : interchanger la position des membres d'une dimension

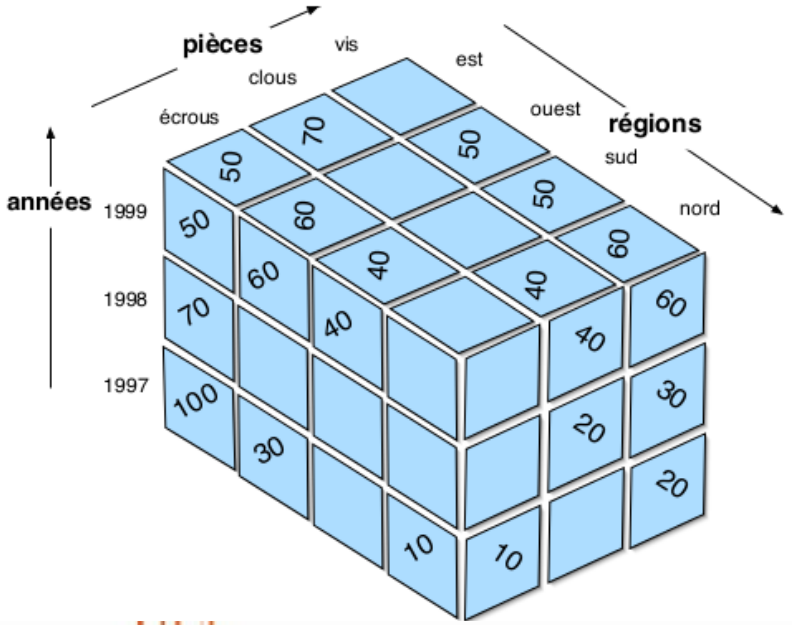




Opération de restructuration split

Split ou division :

- Présentation de chaque tranche du cube
- Passer d'une présentation tridimensionnelle à une présentation d'un ensemble de tables
- Généralisation : découpage d'un hypercube de dimension 4 en cubes (3D)



ventes est	1999	1998	1997
écrous	50	70	100
vis		10	10
clous	70	70	100

ventes sud	1999	1998	1997
écrous	40	20	
vis	50	60	60
clous		10	

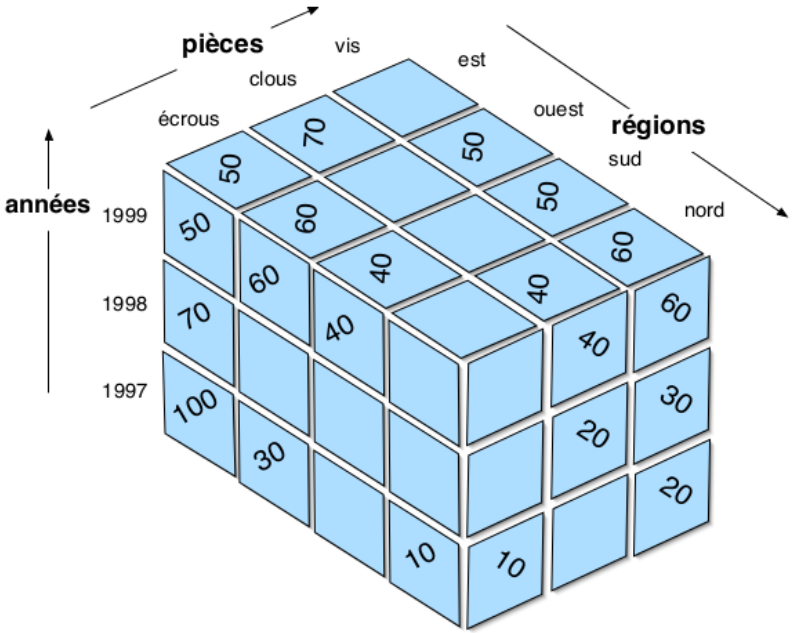
ventes ouest	1999	1998	1997
écrous		10	30
vis	50	50	50
clous		10	40

ventes nord	1999	1998	1997
écrous			10
vis	60	30	20
clous	40	20	

Opération de restructuration nest

`nest` ou l'emboîtement:

- Imbrication des membres à partir du cube
- Regrouper sur une même représentation bi-dimensionnelle de toutes les informations (mesures et membres) d'un cube indépendamment du nombre de ses dimensions.

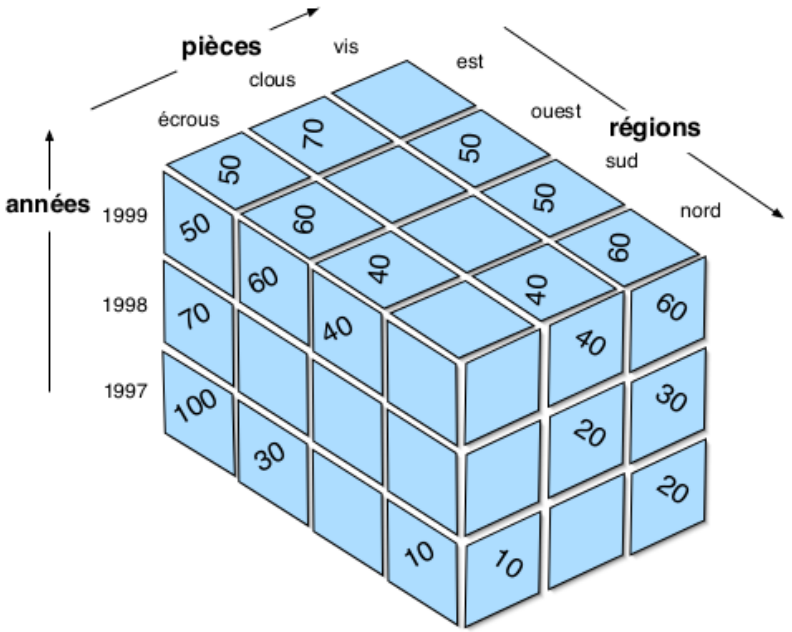


ventes nest		1999	1998	1997
écrous	est	50	70	100
	ouest		10	30
	nord			10
	sud	40	20	
vis	est		10	10
	ouest	50	50	50
	nord	60	30	20
	sud	50	60	60
clous	est	70	70	100
	ouest		10	40
	nord	40	20	
	sud		10	

Opération de restructuration push

`push` ou l'enfoncement :

- Combiner les membres d'une dimension aux mesures du cube
- Passage de membres comme contenu de cellules



ventes push	est	ouest	nord	sud
écrous	1999 50 1998 70 1997 100	1998 10 1997 30	1997 10	1999 40 1998 20
vis	1998 10 1997 10	1999 50 1998 50 1997 50	1999 60 1998 30 1997 20	1999 50 1998 60 1997 60
clous	1999 70 1998 70 1997 100	1998 10 1997 40	1999 40 1998 20	1998 10

Opérations de granularité

Granularité : Hiérarchisation de l'information en différents niveaux de détail (niveaux de granularité)

- Un niveau : un ensemble nommé de membres
- Niveau le plus bas : celui de l'entrepôt
- Définition de nouveaux points de vue
  - de moins en moins d'informations détaillés (niveaux supérieurs)
  - par des opérations d'agrégation
- Navigation entre les niveaux : `roll-up` et `drill-down`
  - groupements
  - agrégation
- Manipulations :
  - relatives
  - besoin d'informations non présentes dans le cube initial

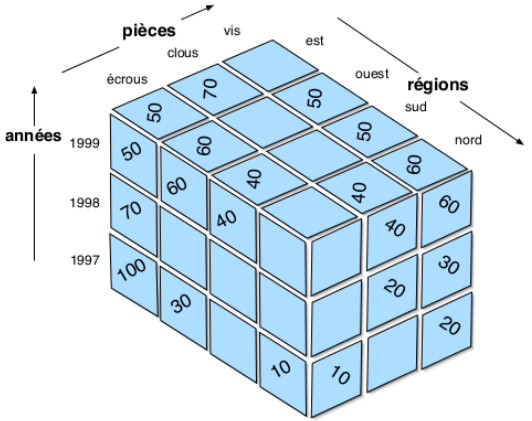
Opérations de granularité

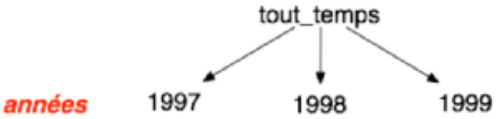
- Action sur la granularité d'observation des données
- Navigation entre les différents niveaux

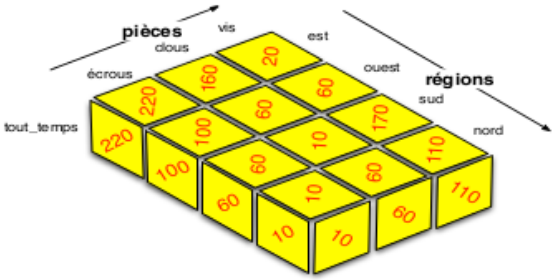
- `roll-up`/forage vers le haut :
  - Représentation des données du cube à un niveau de granularité supérieur dans la hiérarchie de la dimension concernée
  - Il faut spécifier une fonction d'agrégation pour indiquer comment calculer les valeurs du niveau supérieur à partir de celles du niveau inférieur
- `drill-down`/forage vers le bas :
  - Représentation les données du cube à un niveau de granularité de niveau inférieur dans la hiérarchie de la dimension concernée
  - Données présentées sous une forme plus détaillée (en fonction de la hiérarchie de la dimension)

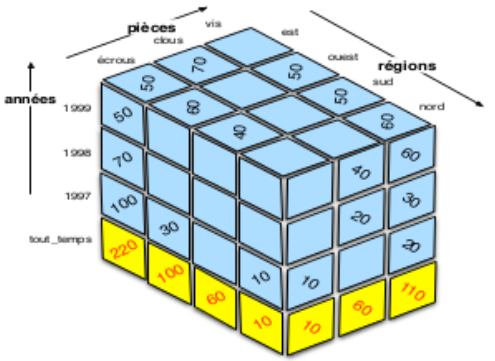
Opération de granularité Roll-Up

`roll-up(annees)`









Opération de granularité roll-Up/Cube

Opération CUBE :

- Représentation cubique généralisée du roll-up
- Calcul de tous les agrégats suivant tous les niveaux de toutes les dimensions
- Union de plusieurs group-by : nouveau cube

```
select ALL, ALL, ALL, Sum(quantite) from VENTES
    union
select pieces, ALL, ALL, Sum(quantite)
From VENTES group-by pieces ;
    union
select pieces, annees, ALL, Sum(quantite)
from VENTES group-by pieces, annees;
    union
select pieces, annees, regions, Sum(quantite)
from VENTES group-by pieces, annees, regions;
```

Opération de granularité roll-Up/Cube

Opérateur cube

- Généralisation N-dimensionnelle de fonctions d'agrégations simples
- Opérateur relationnel :

```
select pieces, annees, regions, Sum(quantite Ventes)
from VENTES
group-by cube pieces, annees, regions;
```

Opération de granularité Drill down

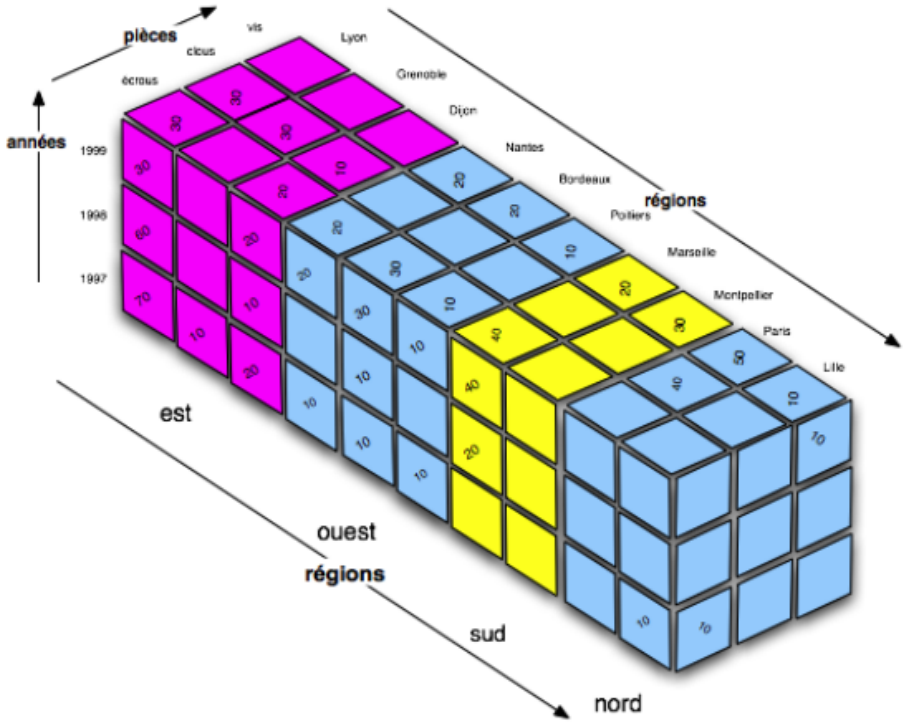
Drill-down ou forage vers le bas :

- Représentation des données du cube à un niveau de granularité de niveau inférieur
- Données sous une forme plus détaillée
- Réciproque de `roll-up`
- Récupération des détails sur un résultat en affinant ou en ajoutant une dimension
- Opération coûteuse si pas intégrée dans le système
- Exemple : un chiffre d'affaire suspect pour un produit donné :
  - Ajout de la dimension temps : impact du week-end
  - Ajout de la dimension magasin : prise en compte du lieu géographique

Opération de granularité Drill down

`drill-down(regions)`

Drill-down du niveau des régions au niveau villes :



Opérations ensemblistes

Objectifs :

- Extraction d'information
- Manipulations classiques
- Extension à plusieurs dimensions

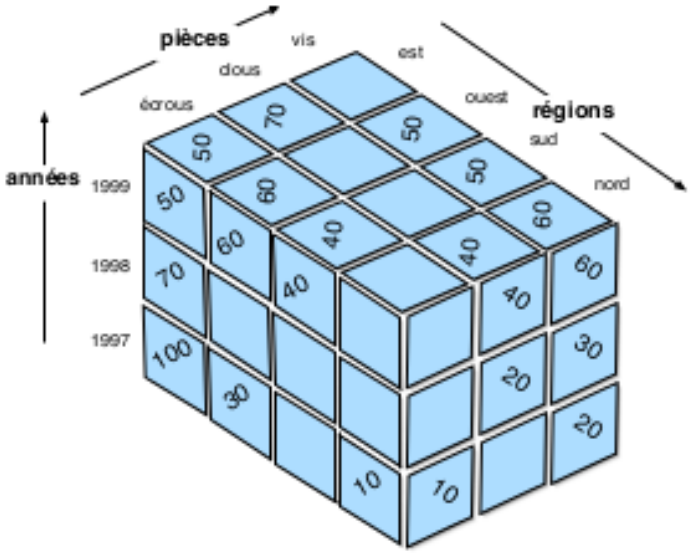
Opérations OLAP ensemblistes :

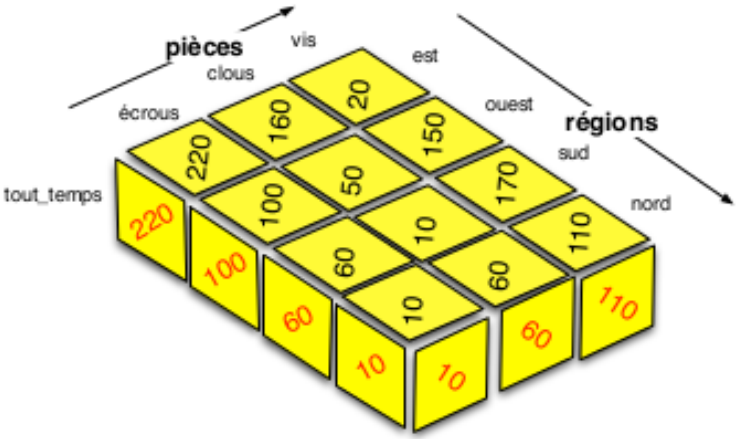
- `slice` : projection selon une dimension du cube
- `dice` : sélection du cube
- `drill-across` (jointure)

Opérations ensemblistes

Exemple

`slide` : projection selon une dimension du cube



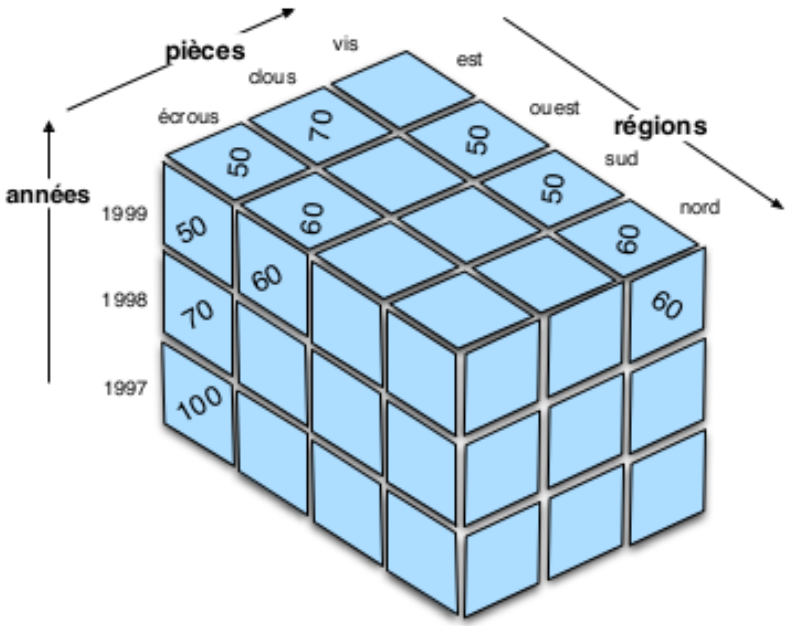


Opérations ensemblistes

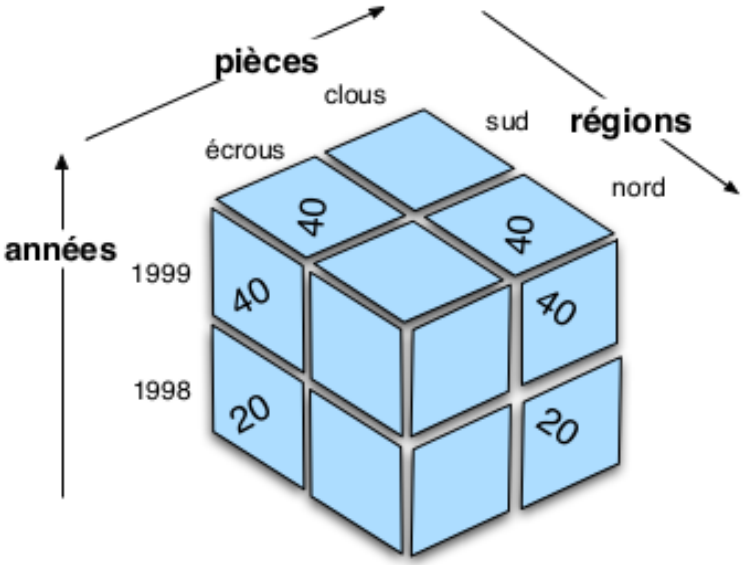
Exemples

`dice` : sélection du cube

Ventes ≥ 50



Ventes des écrous ou des clous, en 1998 ou 1999, dans les régions nord ou sud



Problème des opérateurs ensemblistes

Exemple : $\cup$

prix	1997-1999
écrous	1
clous	0,7
vis	0,8

prix	1997-1999
écrous	0,8
clous	1,1
vis	0,7

Quelle(s) valeur(s) utiliser ?

Exemples de traitement typique

- Quels sont les 10 produits les plus performants ?
- Calculer la moyenne glissante des ventes par région et par pièces, pour une fenêtre de 2 années
- Calculer les prévisions de ventes pour les années 2000 à 2002 avec comme hypothèse un accroissement annuel des ventes de 10%

Règles de Codd pour les produits OLAP

Edgar F. Codd (1993) : définition des bases du modèle OLAP

12 règles de Codd définissant l'évaluation des produits OLAP :

1. **Vue multidimensionnelle** des données dans une base OLAP
2. **Transparence** : éléments techniques invisibles pour l'utilisateur
3. **Accessibilité** : complexité et l'hétérogénéité des données masquées par les outils OLAP
4. **Stabilité** : performances stables indépendamment du contexte d'analyse
5. **Architecture Client/Serveur** : le serveur homogénéise les données, les clients se connectent simplement au serveur
6. **Traitement générique des dimensions** : une seule structure logique pour toutes les dimensions. Tout calcul effectué sur une dimension peut l'être sur les autres
7. **Gestion dynamique des matrices creuses** : gestion dynamique de la mémoire physique nécessaire pour stocker les données non nulles
8. **Support multi-utilisateurs** : gestion des accès concurrents aux données
9. **Croisement des dimensions**
10. **Manipulation intuitive des données**
11. **Flexibilité des restitutions**
12. **Nombre illimité de niveaux d'agrégations et de dimensions**

Caractéristiques des produits OLAP

FASMI – *Fast Analysis of Shared Multidimensional Information*

(Pendse 2005) : 18 règles (extension de Codd 1993)

- **F**ast : temps de réponse aux demandes des utilisateurs entre 1 et 20 secondes

  utilisation de pré-calculs pour réduire les durées des requêtes
- **A**nalysis :
  - Prendre en compte à toutes les logiques d'affaire et de statistiques
  - fournir la possibilité aux utilisateurs de construire leurs calculs et leurs analyses sans avoir à programmer
- **S**hared :
  - Possibilité de préservation de la confidentialité
  - Gestion des cas où plusieurs utilisateurs ont des droits en écritures
- **M**ultidimensional :
  - Production de vues conceptuelles multidimensionnelles des données
  - Possibilité de hiérarchies de dimensions
- **I**nformations : ensemble des données et les informations nécessaires

Type de systèmes OLAP

3 stratégies :

1. Utilisation d'un SGBD Relationnel (systèmes ROLAP)
   - SGBDR : Nécessité des adaptations pour répondre aux besoins des ED
   - Stockage des données dans un SGBDR
   - Utilisation d'un middle-ware pour implémenter les opérations spécifiques de l'OLAP
   - Lents et peu performants mais sans limites de taille
2. Utilisation d'un SGBD Multidimensionnel (systèmes MOLAP)
   - SGBD capable de stocker et traiter des données multidimensionnelles
   - Basé sur un stockage par tableaux (techniques des matrices creuses)
   - Indexation rapide des données calculées
   - Très rapides et performants mais limité au gigaoctet
3. Utilisation d'un SGBD Hybride (systèmes HOLAP)

   Tirer profit des avantages des technologies ROLAP et MOLAP :
   - un ROLAP pour stocker, gérer les données détaillées
   - un MOLAP pour stocker, gérer les données agrégées

Autres : DOLAP, OOLAP, WOLAP, SOLAP, Mobile OLAP

Quelques solution commerciales

DB2 UDB Server : ROLAP
Oracle : ROLAP
SQL Server 2000 : ROLAP
SAS OLAP Server : MOLAP
SQL Server : HOLAP
Oracle Express-server : MOLAP/ROLAP
DB2 OLAP Server : MOLAP/ROLAP

ROLAP

Stratégie ROLAP d'implantation d'un ED

- SGBD relationnels : plus de 80% des SGBD
- Stratégie la plus couramment pour implanter un ED
- Nécessité d'adaptation des SGBDR pour répondre aux besoins des ED :
  - Extensions du langage SQL à de nouveaux opérateurs
  - Usage de vues matérialisées
  - Indexation binaire pour améliorer les performances
- Réalisation de calculs dérivés et agrégations à différents niveaux
- Génération de requêtes adaptées au schéma relationnel de l'ED
- Exploitation des vues matérialisées existantes (facteur principal de performance)

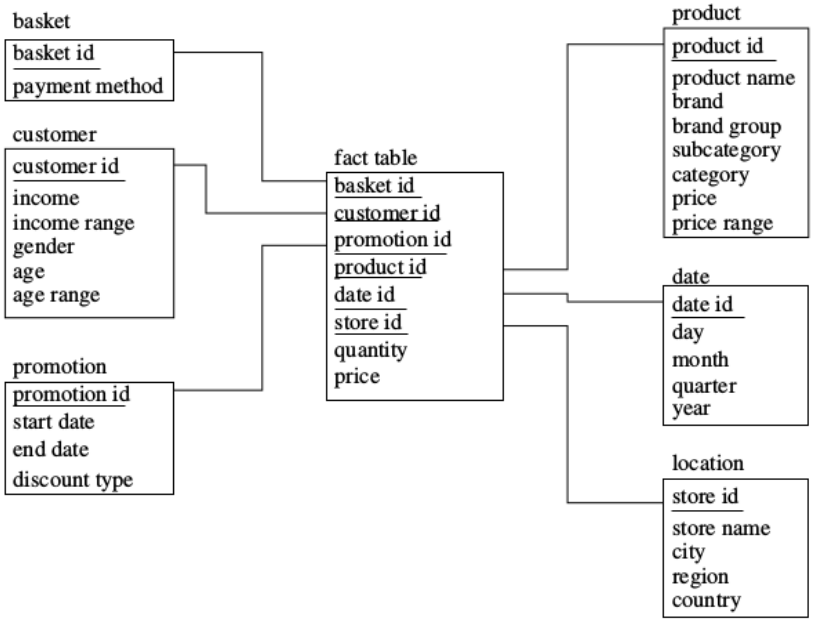
Modèles logiques d'un ED « ROLAP »

Modèle multidimensionnel :

- Chaque fait correspond à une table, appelée table de fait constituée
  - d'attributs représentant les mesures d'activité
  - des attributs clés étrangères de chacune des tables de dimension
- Chaque dimension correspond à une table, appelée table de dimension constituée :
  - les paramètres
  - une clé primaire permettant de réaliser des jointures avec la table de fait

Modèles logiques d'un ED ROLAP

Soit le schéma en étoile :

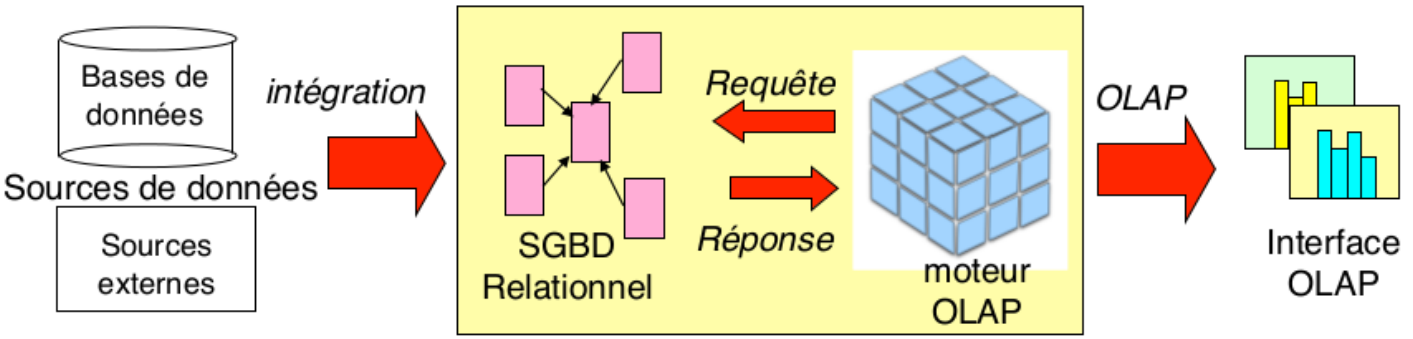


Modélisation logique ROLAP :

VENTE(CleTps\#, CleGeo\#, CleCat\#, Quantite , Montant) – **table** des faits
TEMPS(CleTps , Annee , Trimestre , Saison , Mois , Jour)) – **table** de dimension
GEOGRAPHIE(CleGeo , Region , Departement , Ville) – **table** de dimension
CATEGORIE(CleCat , TypeProd , Gamme, NomProd , Couleur) – **table** de dimension

Introduction à la technologie ROLAP

- Utilisation d'un SGBR relationnel pour stocker l'ED (structure en étoile ou flocon)
- Moteur OLAP : élément complémentaire pour
- Fournir une vision multidimensionnelle de l'ED
- Réaliser des calculs de données dérivés et des agrégations à différents niveaux
- Génèrer des requêtes SQL adaptées au schéma relationnelle de l'ED (grâce aux vues matérialisées existantes)



Introduction à la technologie ROLAP

- Systèmes ROLAP : technologie de stockage relationnelle
- Extension du modèle relationnel pour supporter les requêtes d'analyses multidimensionnelles du niveau d'application
  - Nouveaux opérateurs comme `cube`
  - Nouvelles fonctions : `rank`, `percentile` pour complèter les fonctions classiques de SQL (`count`, `sum` et `avg`

Introduction à la technologie ROLAP

- Rôle du moteur OLAP :
  - Traduction dynamique du modèle logique de données multidimensionnel M en modèle de stockage relationnel R (en étoile ou en flocon)
  - En fait, transformation d'une requête multidimensionnelle m sur M en une requête relationnelle r sur R
- Efficacité de la requête : performance et passage à l'échelle global du système
  $\rightarrow$ Importance des techniques et des stratégies d'optimisation dans les produits ROLAP :
  1. Techniques d'indexation spécifiques
  2. Sélection et matérialisation de vues
  3. Fragmentation des tables de l'entrepôt

Techniques d'indexation ROLAP

- Listes inversées (*inverted lists*)
- Indexation binaire (index de vecteurs de bits - *bitmap*) :
  (oracle, DB2, microsoft SQL server, sybase IQ)
- Index de jointure (*join indexing*)
  (oracle 9i)

Techniques d'indexation ROLAP

Listes inversées

Exemple d'utilisation :

- Requête :

  GET `people` WITH `age` = 20 **AND** `name` = `''Fred''`

- Soit `L1` : liste pour attribut `age = 20` :
  `{r4, r18, r34, r35}`
- Soit `L2` : liste pour attribut `name = Fred` : `{r18, r52}`

- Réponse à la requête : intersection de L1 avec L2
  L1 ∩ L2 = `{r4, r18, r34, r35}` ∩ `{r18, r52}` = `{r18}` = `r18`

Techniques d'indexation ROLAP

indexation binaire/bitmap index

Index de vecteurs de bits
Principe général :

- Dans une table, pour un attribut donné : association de la liste des tuples dont l'attribut contient la valeur
  idem pour un groupe d'attribut
- Index binaire : Représentation de la liste à l'aide d'un vecteur de bits
  - 1 si le tuple associé fait partie de la liste
  - 0 dans le cas contraire
- Stockage en mémoire centrale car structure de taille réduite
- Indexation adaptée si peu de valeurs pour un attribut

Techniques d'indexation ROLAP

Avantages de l'indexation binaire :

- Opérations sur les bits très rapides (AND, OR, XOR, NOT)
- Optimisation de requêtes de types sélection, comparaison, jointure, agrégation ...
- Un vecteur de bits pour chaque valeur d'attribut
- Longueur du vecteur de bits = nombre de tuples de la table
- Plus compact que les B-arbres

Inconvénients de l'indexation binaire :

- Coût de maintenance important : tous les index binaires d'une table doivent être actualisés lors de l'insertion d'un nouveau tuple dans la table
- Espace important pour leur stockage.
  Utilisation de technique de compression pour réduire ce coût

Techniques d'indexation ROLAP

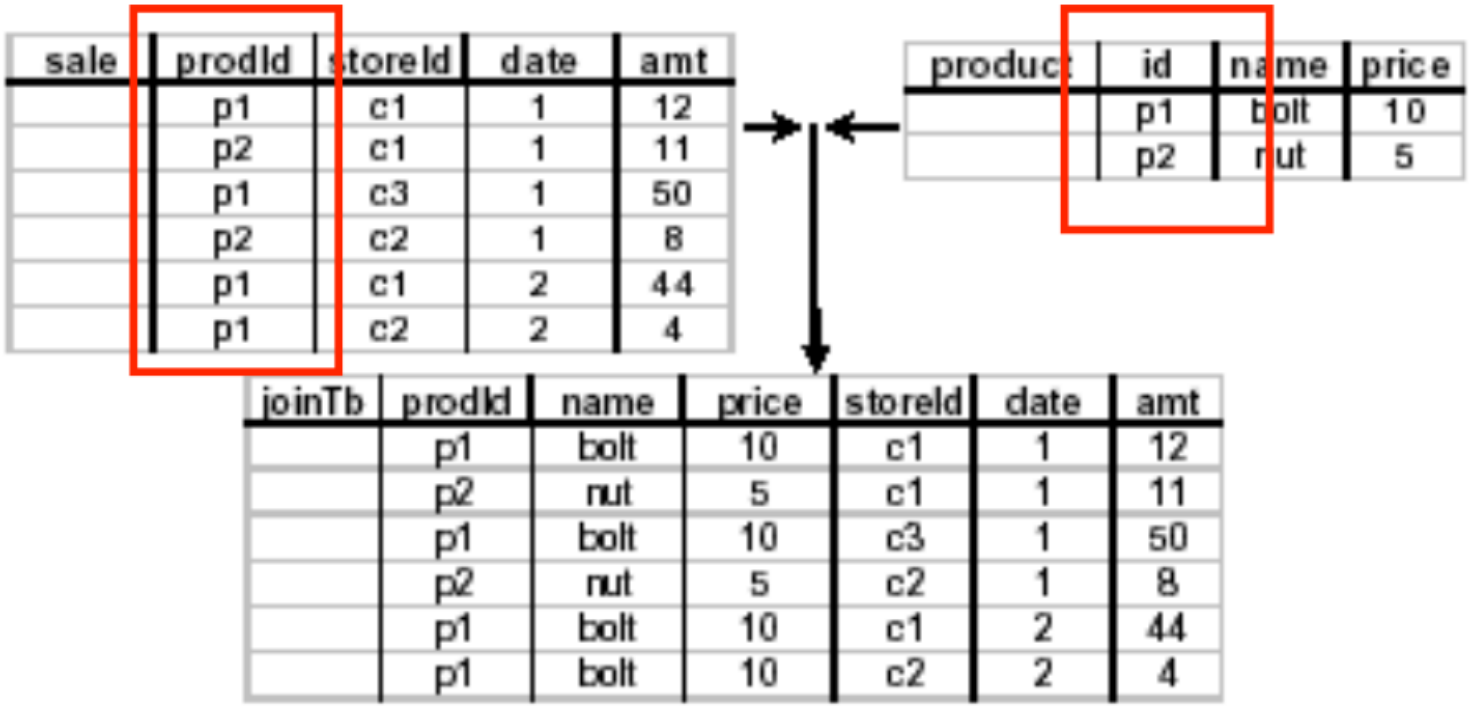
Indexation de jointure

- Pré-calcul de jointure binaire
- Utilisation dans les schéma en étoile
- Objectif : éviter de calculer la jointure
- Maintenance des relations entre :
  - une clé étrangère
  - les clés primaires qui la contienne

Remarque : les techniques bitmap index et join index peuvent être combinées

Techniques d'indexation ROLAP

Indexation de jointure



sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

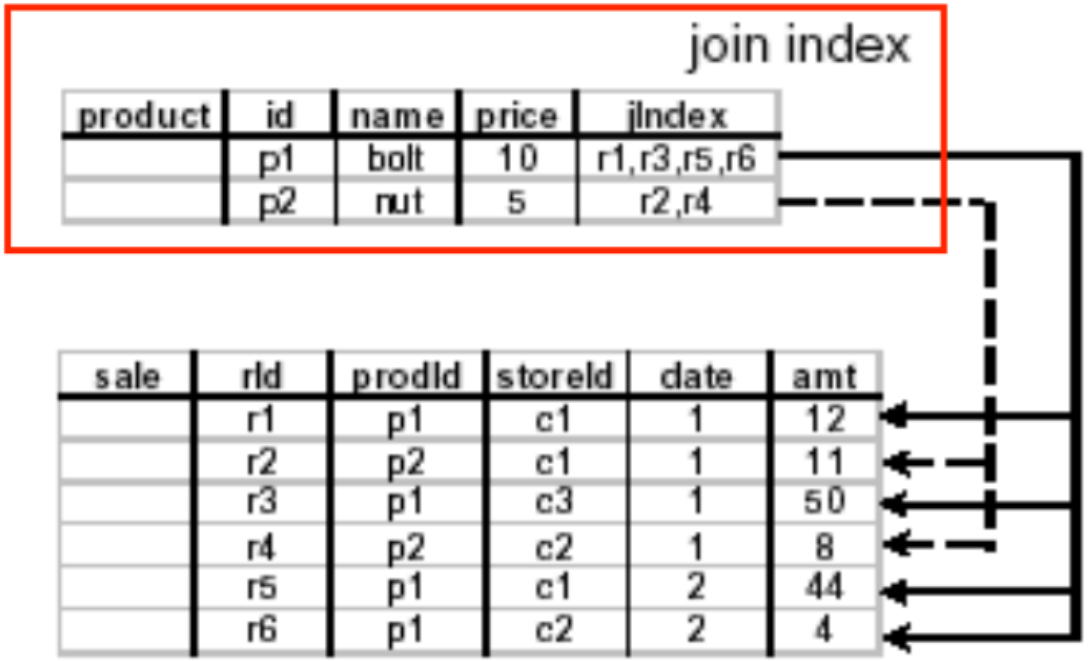
product	id	name	price
	p1	bolt	10
	p2	nut	5

joinTb	prodId	name	price	storeId	date	amt
	p1	bolt	10	c1	1	12
	p2	nut	5	c1	1	11
	p1	bolt	10	c3	1	50
	p2	nut	5	c2	1	8
	p1	bolt	10	c1	2	44
	p1	bolt	10	c2	2	4

(d'après H. Garcia-Molina)

Techniques d'indexation ROLAP

Indexation de jointure



(d'après H. Garcia-Molina)

Sélection et matérialisation des vues en ROLAP

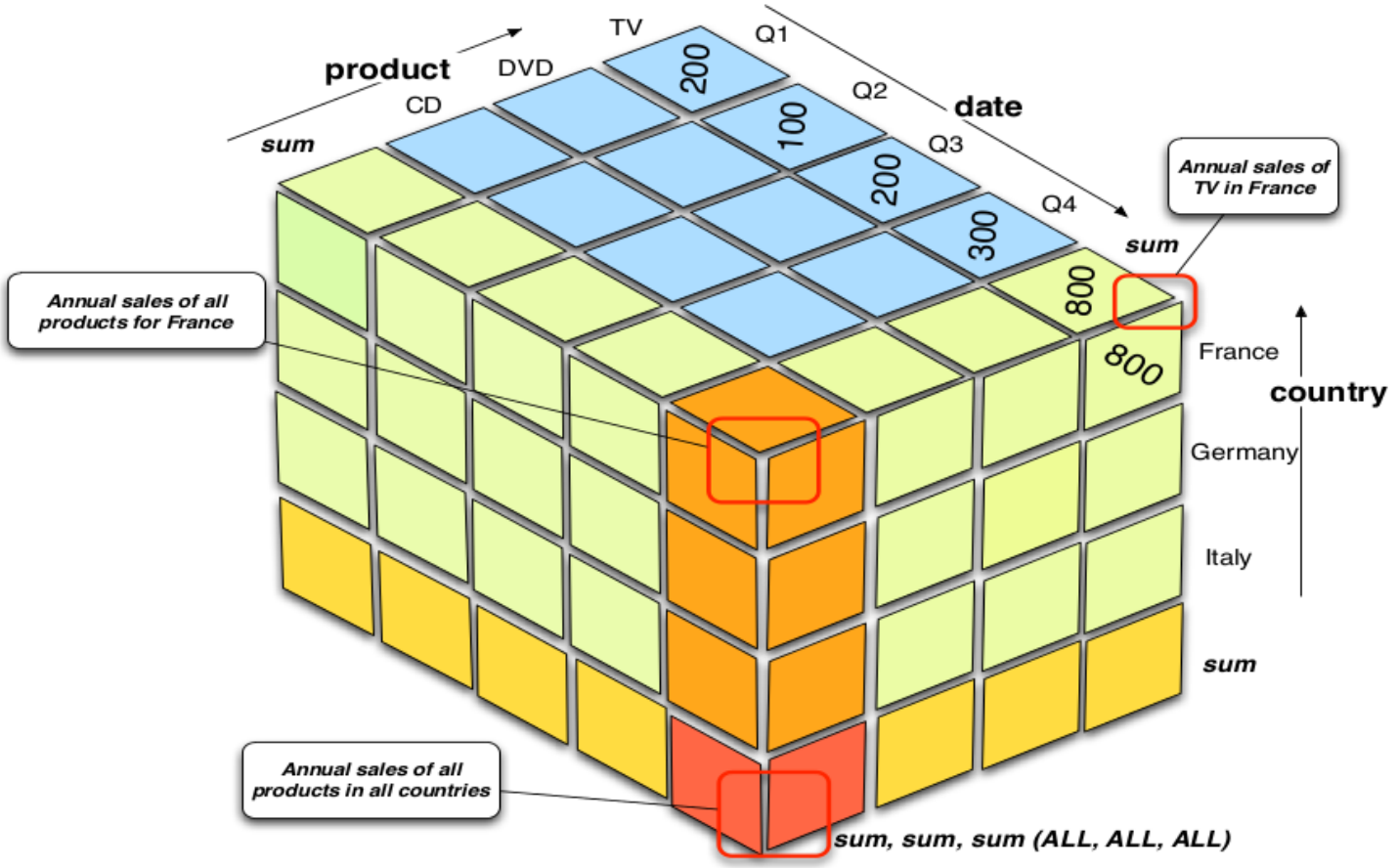
rappels

- Entrepôt : modèle multidimensionnel où les données sont vues comme des cubes de données (*data cubes*)
  Exemple : cube `Ventes`, vision des données selon plusieurs dimensions
  - Tables de dimension

    `Produit(nom_produit, marque, type)`
    `Date``(jour, semaine, mois, trimestre, annee)`
  - Table de faits
    - des mesures (`unites_vendues`)
    - clés externes : référence à chaque table de dimension
  - Cube de dimension n : cuboïde
  - Treillis des cuboïdes d'un entrepôt : `data cube`

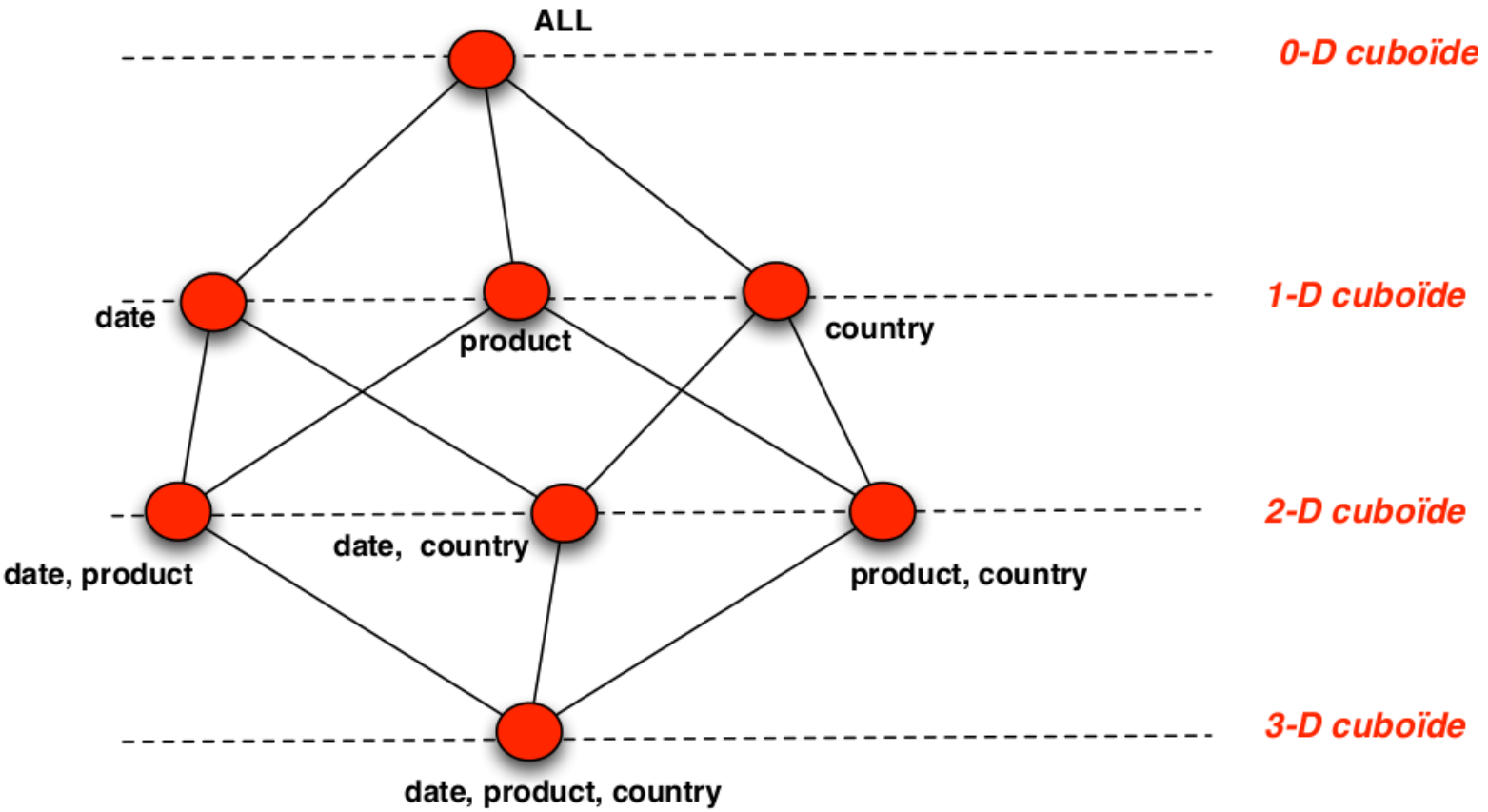
Sélection et matérialisation des vues en ROLAP

Cube `Ventes` avec tous les agrégats possibles :



Sélection et matérialisation des vues en ROLAP

Cube: treillis de cuboïdes



Sélection et matérialisation des vues en ROLAP

Pré-calcul d'agrégats

- 3 possibilités :
  1. pas stockage d'agrégat (coûteux en temps)
  2. stockage tous les agrégats (coûteux en espace)
  3. stockage d'une partie des agrégats : nécessite de déterminer lesquels stocker
- Datacube : faits + tous les cuboïdes possibles
  - cube dense : taille du datacube = taille de la table de faits
  - cube creux : chaque cuboïde = taille de la table de faits

  Choix de la matérialisation des cuboïdes en fonction :
  - du grain (niveau d'agrégation) : le grain doit être suffisamment fin pour pouvoir répondre aux requêtes
  - des requêtes utilisateurs

  $\rightarrow$ Matérialisation des cuboïdes grâce à des vues matérialisées

Sélection et matérialisation des vues en ROLAP

Vues matérialisées

- Résultat du calcul d'une vue stockée sur disque
- Représentation des agrégations des tables d'un schéma en étoile
- Utilisation de ces vues par les requêtes (données pré-agrégées) : amélioration les performances
- Possibilité de construction d'une vue matérialisée à partir d'une autre
- Matérialisation de toutes les vues souvent impossible
  → Sélection des vues à matérialiser en fonction :
  - du coût d'exécution des requêtes
  - du coût de maintenance (rafraîchissement)
  - du coût de calcul
  - de l'espace disque requis
- Réécriture des requêtes pour utiliser les vues matérialisées
  → Réécriture parfois difficile

Sélection et matérialisation des vues en ROLAP

Exemple

- Exemple de cube

  Ventes (modele, couleur, **date**, vendeur, prix, quantite)

- Vue matérialisée des agrégats par modele, couleur, mois, ville :

```
INSERT INTO ventesVue1
SELECT modele, couleur, mois, ville,
SUM(prix) as prix, SUM(quantite) as quantite
FROM ventes, vendeur, temps
WHERE ventes.vendeur = vendeur.nom
AND ventes.date = temps.jour
GROUP BY modele, couleur, mois, ville
```

- Vue matérialisée des agrégats par modele, semaine, departement :

```
INSERT INTO ventesVue2
SELECT modele, semaine, departement
SUM(prix) as prix, SUM(quantite) as quantite
FROM ventes, vendeur, temps
WHERE ventes.vendeur = vendeur.nom
AND ventes.date = temps.jour
GROUP BY modele, semaine, departement
```

Sélection et matérialisation des vues en ROLAP

Exemple

- Vente par modèle :

  ```
  SELECT modele, SUM(prix) FROM ventes GROUP BY modele
  ```

- peut être traitée par :

  ```
  SELECT modele, SUM(prix) FROM ventesVue1 GROUP BY modele
  ```

  ou

  ```
  SELECT modele, SUM(prix) FROM ventesVue2 GROUP BY modele
  ```

Sélection et matérialisation des vues en ROLAP

Exemple

- Vente par modèle, par année, par département :

```
SELECT modele, annee, departement, SUM(prix)
FROM ventes, vendeur, temps
WHERE ventes.vendeur = vendeur.nom
AND ventes.date = temps.jour
GROUP BY modele, annee, departement
```

peut être traitée par :

```
SELECT modele, annee, departement, SUM(prix)
FROM ventesVue1, vendeur, temps
WHERE ventesVue1.ville = vendeur.ville
AND ventesVue1.mois = temps.mois
GROUP BY modele, annee, departement
```

Sélection et matérialisation des vues en ROLAP

Exemple

- Vente par modèle, par couleur, par date

```
SELECT modele, couleur, date, SUM(prix)
FROM ventes
GROUP BY modele, couleur, date
```

ne peut être traitée grâce à `ventesVue1` ou `ventesVue2` :

- grain des vues trop gros
- `ventesVue2` ne regroupe pas par couleur

Fragmentation des tables

tables schéma en étoile

- Objectif : optimisation des requêtes
- Découpage des tables du schéma en étoile :
  - fragmentation horizontale : sélection
  - fragmentation verticale : projection
- Evaluation des requêtes sur chaque fragment
- Réponse complète : besoin d'une requête de reconstruction
  - fragmentation horizontale : union
  - fragmentation verticale : jointure

Forces et faiblesses de la technologie ROLAP

- Avantages :
  - Technologie relationnelle mure
  - Stokage de très grands volumes de données
  - Possibilité de définition de données complexes et multidimensionnelles en utilisant un modèle relativement simple
  - Réduction du nombre de jointures à réaliser dans l'exécution d'une requête
- Faiblesses :
  - Temps de réponses potentiellement élevés : manque d'efficité de la génération de SQL
  - Pas de possibilité de réaliser des requêtes OLAP avec des calculs complexes
  - Structure de l'entrepôt en étoile ou flocon seulement

Quelques produits de technologie ROLAP

- IBM DB2 UDB:
  - SGBD disponible sur de nombreuses plate-formes
  - possibilité de *fédération* de bases de données relationnelles (*shared nothing database*)
  - Ensemble de produits :
    - DB2 Performance Expert : création de rapports, d'analyses et recommande des changements pour améliorer la performance
    - DB2 Data Joiner: optimisation des requêtes SQL.
    - DB2 Integrated Cluster Environnement : passage à l'échelle
- Oracle :
  - SGBD disponible sur de nombreuses plate-formes
  - Possibilité de partitions de *hash*, *range* et *list*, et consolidation sur une base de données centralisée (*shared disk data base*)
  - Ensemble de produits :
    - *Real Application Clusters* : affectation de certains processeurs comme processeurs OLAP et d'autres comme processeurs de requêtes
    - Optimiseur : basé sur les coûts ou sur les règles

Quelques produits de technologie ROLAP

- SQL Server 2000 :
  - Possibilité de *fédération* de bases de données relationnelles (*shared nothing database*) e la liaison entre bases de données distribuées et hétérogènes
  - Ensemble d'outils :
    - Optimiseur de SQL Server : basé sur les coûts avec création automatique de statistiques et leur rafraîchissement
    - Query Processor : possibilité des requêtes multidimensionnelles, et des index composites et semi-jointures
    - SQL Query Analyzer : suggestions par rapport à l'implantation des index additionnels et des statistiques complémentaires
    - Microsoft DTS (Data Transformation Services) : outil ETL intégré dans Microsoft SQL Server.
  - Autres produits :
    - Sybase IQ, DSS Agents de MicroStrategy, MetaCube de Informix, ...

MOLAP

Stratégie MOLAP d'implémentation d'un entrepôt

- Utilisation un SGBD Multidimensionnel (SGBDM)
  avec capacité de stockage et de traitement des données multidimensionnelles
- Cadre technologique commun pour ces systèmes :
  Version du modèle multidimensionnel et stratégies de stockage différentes en fonction du produit
- Bonnes performances
  car pré-agrégation et pré-calcul des données sur tous les niveaux des hiérarchies du modèle de l'entrepôt
- Génération de très grands volumes d'information
- Techniques incrémentales de rafraîchissement encore limitées
  nécessite de reconstruire périodiquement l'entrepôt
- Adapter pour de petits ED (quelques Go) et lorsque le modèle multidimensionnel ne change pas beaucoup

Introduction à la technologie MOLAP

- Stockage les données de manière multidimensionnelle
- (+) Calcul des agrégats dans des tables : en colonne ou en ligne
  donc très rapide car pas de jointure à faire
- Taille limitée
- (-) Pas de langage d'interrogation des données :
  nécessite une redéfinition des opérations de manipulation des structures multidimensionnelles

Introduction à la technologie MOLAP

Technologie des bases de données multidimensionnelles :

- Structure de stockage : tableaux
- Correspondance directe avec la vue multidimensionnelle
- Les membres sont implicites :
  - correspond à l'adresse de la cellule
  - sont normalisés (vis = 0, clous = 1, ...)
- Gestion de la faible densité (*sparsity*) :
  - Techniques de compression spécifiques
  - Structure d'index spécifiques
  - Si le tableau est dense, la mémoire ne contiendra que les mesures

Introduction à la technologie MOLAP

Exemple de tableau MOLAP

Données :

- 1460 jours
- 200 000 produits (2×10^5)
- 300 magasins (3×10^2)
- promotion : 1 valeur booléenne (2)

Nombres de cellules du cube :

$$1460 \times 2x10^5 \times 3 \times 10^2 \times 2 = 1,75 \times 10^{11} \ \textit{cellules}$$

Densité du cube :

- seulement 10% de produits vendu par jour :
- densité $= 1,75 \times 10^{10}/1,75^{11} \times 10 = 0,1$

Densité/compression et indexation

- Compression
  - Typiquement, jusqu'à 90 % de cellules vides (D'après P. Marcel)
  - Stockage des données en blocs denses utilisation de techniques de compression (similaires pour certaines à celles utilisées en relationnel)
  - Mais ces techniques fonctionnant bien pour 2 ou 3 dimensions échouent en 20 dimensions
- Techniques d'indexation spécifiques

Agrégation et calcul des agrégats

Coût MOLAP de opération

- Agréger : parcourt et application de la fonction d'agrégat sur des lignes du tableau
  - Calcul des agrégats à la demande
  - Précalcul et stockage des agrégats comme des lignes du tableau

    Exemple naïf : cube c de dimension `A,B,C group by A,C`

```
for(a=0;a<a max;a++)
   for(b=0;b<b max;b++)
      for(c=0;c<c max;c++)
         res[a][c] += c[a][b][c]
```

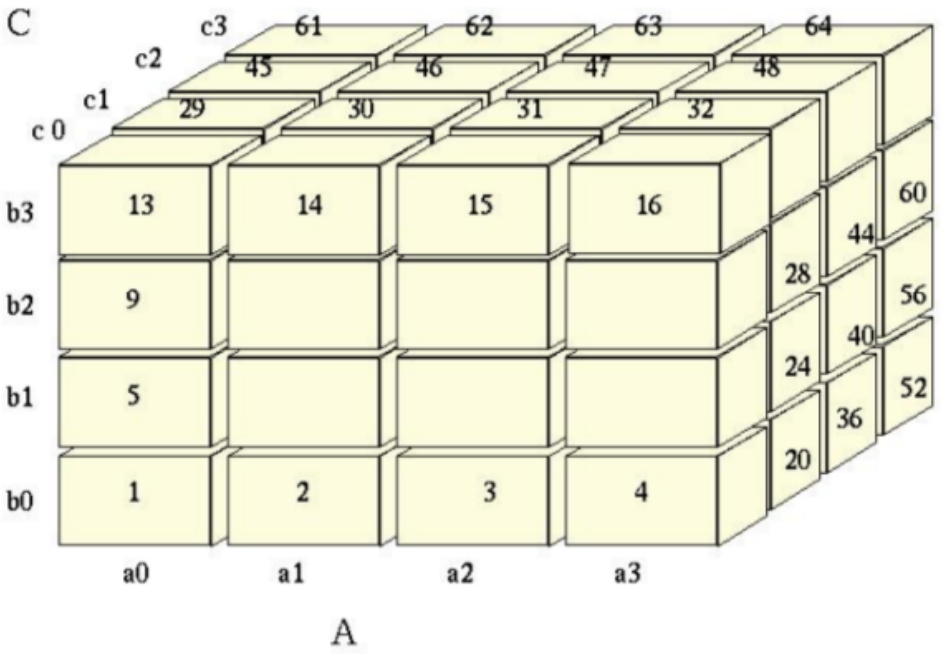
Agrégation et calcul des agrégats

Coût MOLAP de opération

- Calcul des agrégats en MOLAP :
  1. Partitionnement du tableau physique de n dimensions en sous-cubes (*chunks*)
     - de n dimensions
     - stockés en mémoire principale
     - compressés pour gérer la faible densité
  2. Calcul des agrégats :
     - visiter chaque cellule de chaque sous-cube
     - calculer l'agrégat partiel impliquant cette cellule

Agrégation et calcul des agrégats

Coût MOLAP de opération



(D'après P. Marcel)

Agrégation et calcul des agrégats

Coût MOLAP de opération

- Objectif : minimiser le nombre de visite par cellule dans le calcul d'agrégats
- Exploitation de l'ordre de visite pour calculer simultanément des agrégats partiels (réduction des accès mémoire et des coûts de stockage)

Agrégation et calcul des agrégats

Coût MOLAP de opération

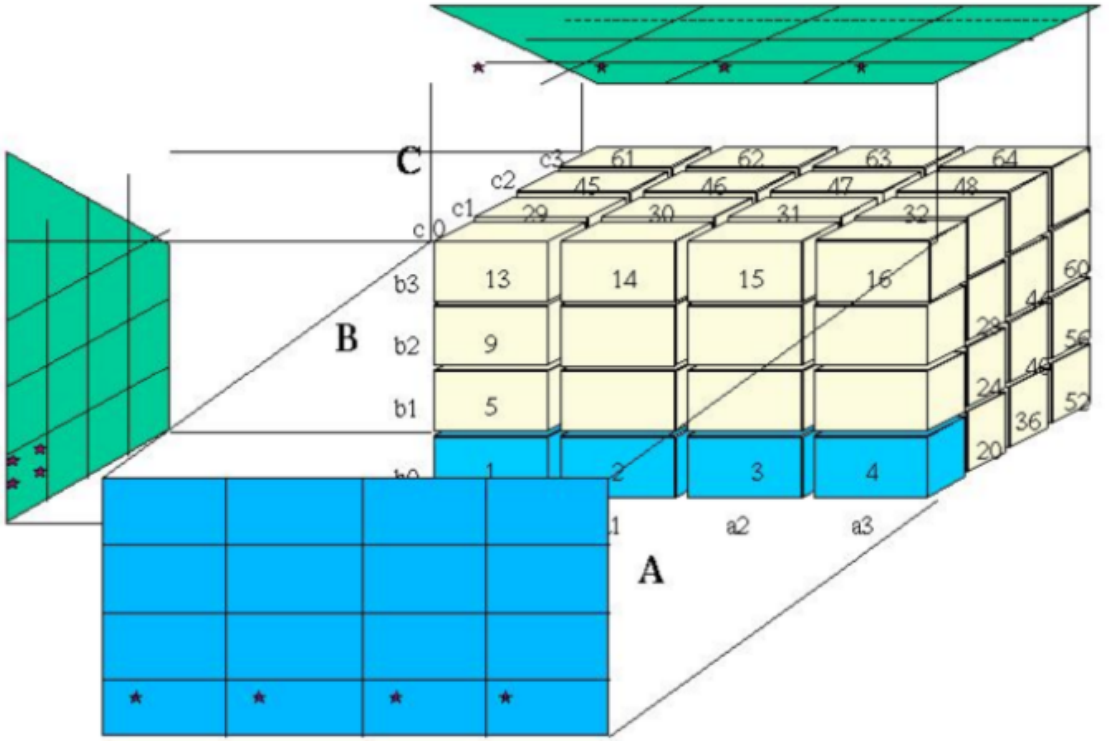
- Exemple :
  - cube à 3 dimension A, B, C
  - tailles :

	taille
A	40
B	400
B	4000
BC	1600000
AC	160000
AB	16000

- Dimensions partitionnées en 4 sous-cubes de même taille scan dans l'ordre : $1, 2, 3, ..., 64(BC, AC, AB)$
- le calcul de b0c0 demande 4 scans $(1, 2, 3, 4)$
  - le calcul de a0c0 demande 13 scans $(1, 5, 9, 13)$
  - le calcul de a0b0 demande 49 scans $(1, 17, 33, 49)$

Agrégation et calcul des agrégats

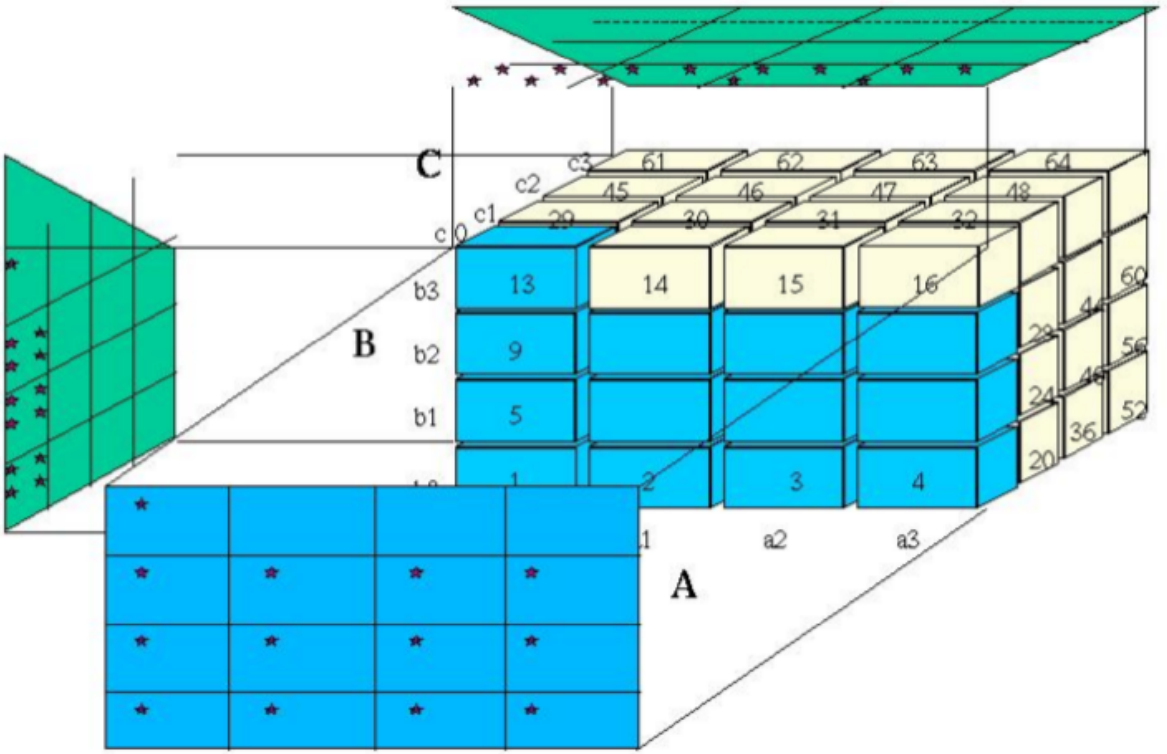
Coût MOLAP de opération



(D'après P. Marcel)

Agrégation et calcul des agrégats

Coût MOLAP de opération



(D'après P. Marcel)

Agrégation et calcul des agrégats

Coût MOLAP de opération

- Occupation mémoire minimum :
  AB + Une colonne de AC + un sous-cube de BC

$$16000 + 10 \times 4000 + 100 \times 1000 = 156000$$

- Scan dans l'ordre $1, 17, 33, 49, 5, 21, ...(AB, AC, BC)$
  le calcul de b0c0 demande 49 scans
  le calcul de a0c0 demande 13 scans
  le calcul de a0b0 demande 4 scans
- Occupation mémoire minimum :
  BC + une colonne de AC + un sous-cube de AB

$$1600000 + 10 \times 4000 + 10 \times 100 = 1641000$$

Agrégation et calcul des agrégats

Coût MOLAP de opération

- Méthode (valable pour un petit nombre de dimensions) :
  - Les cuboïdes doivent être calculés par taille croissante
  - On garde le plus petit cuboïdes en mémoire principale
  - On recherche et on calcule seulement un sous-cube à la fois pour le plus grand cuboïde

Agrégation et calcul des agrégats

Coût MOLAP des opérations typiques :

Opération	coût
roll-up	dépend de l'utilisation d'un cache
drill-down	
rotate	élévé : accès au cube
slice/dice	

(D'après P. Marcel)

Forces et faiblesses de la technologie MOLAP

- Avantages :
  - Très bonnes performances grâce à
    - La réalisation de nombreuses pré-agrégations et de pré-calculs de données sur tous les niveaux de hiérarchies des dimensions de l'ED
    - Un accès rapide à une position d'un tableau (par son indice)
- Inconvénients :
  - Pré-agrégations et les pré-calculs de données : génération de très importants volumes de données
  - Dégradation rapide des performances quand la taille de l'ED dépasse quelques Go et que le modèle multidimensionnel évolue
  - Techniques incrémentales de rafraîchissement limitées : nécessité de reconstruire périodiquement l'entrepôt

Quelques produits de technologie MOLAP

- Essbase (Arbor Software) :
  - SGBD multidimensionnel et multi-utilisateurs
  - Ensemble d'outils :
    - Hyperion Essbase Application Manager : outils graphiques, des modules pour la construction et le chargement des structures OLAP, pour le chargement des données, pour la définition des processus de calcul, pour la gestion des partitions de la base de données,...
  - Hyperion Essbase Query Designer : navigation des utilisateurs finaux facilitée
  - Hyperion Essbase Partition Option : création logique et physique de sous-ensembles des bases de données
    2 types de partitions sont supportés : transparente et liée.

Quelques produits de technologie MOLAP

- SAS OLAP Server (SAS) :
  - SGBD multidimensionnel
  - ensemble d'outils :
    - SAS OLAP Cube Studio : construction des cubes et utilisable facilement pour la définition des mesures, des dimensions et des agrégations
    - SAS Metadata Server : conteneur des métadonnées

Quelques produits de technologie MOLAP

Informix MetaCube (Informix) :

- SGBD multidimensionnel
- ensemble d'outils :
  - MetaCube Analysis Engine : interface multidimensionnel et extension de la fonctionnalité d'analyse de la base de données avec l'incorporation des opérations OLAP (opération `rotate`).
  - MetaCube Explorer : interface graphique permettant aux utilisateurs d'exécuter des requêtes au travers du MetaCube Analysis Engine
  - MetaCube Warehouse Manager : interface graphique pour créer une description multidimensionnelle des tables et des colonnes dans l'entrepôt de données
  - MetaCube Analysis Engine : stockage de la description multidimensionnelle comme un ensemble de tables
- Autres systèmes MOLAP :
  - Pilot (Pilot Software)
  - TMI (Applix)

Systèmes HOLAP

Stratégie HOLAP d'implémentation d'un entrepôt

Constat :

- dans l'approche relationnelle (ROLAP)
  30% du temps est consacré aux entrées/sorties
- Dans l'approche multidimensionnelle (MOLAP) :
  20% du temps est consacré aux entrées/sorties
  (70% calculs et 10% décompression)

Approche HOLAP (Hybride OLAP)

- Utilisation des tables comme structure permanente de stockage des données
- Des tableaux comme structure pour les requêtes

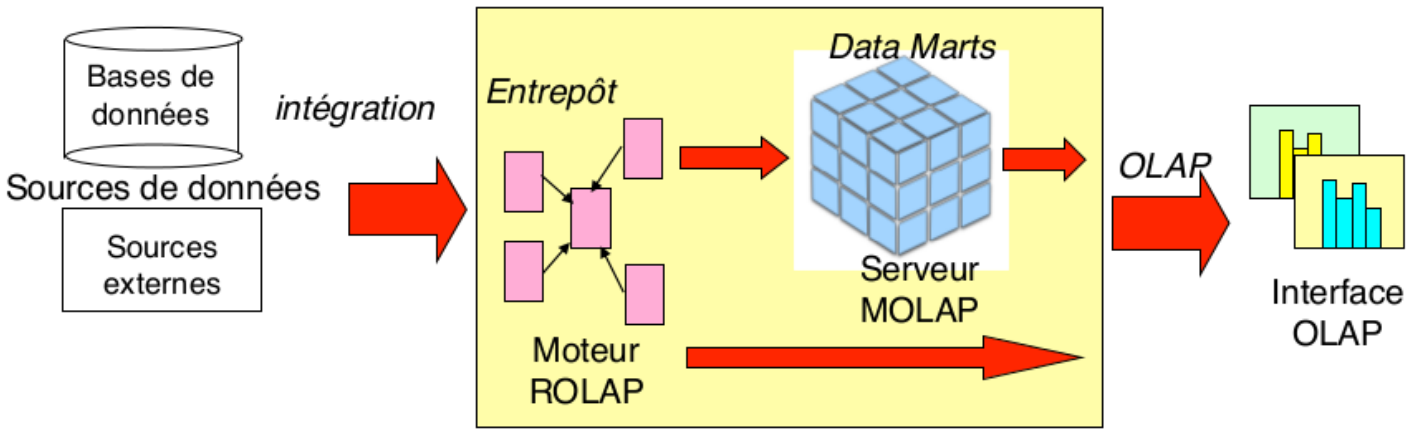
Introduction à la technologie HOLAP

- Tentative de combiner les bons cotés des systèmes ROLAP et MOLAP :
  - Stockage des données détaillées (granularité fine) de l'ED dans un SGBD Relationnel - ROLAP
  - Stockage les données agrégées, souvent des magasins de données (data marts) de l'ED dans un SGBD Multidimensionnel - MOLAP (granularité moins fine, index en mémoire centrale, ...)
- $\rightarrow$ Possibilité d'avoir des ED de taille importante tout en ayant des temps de réponse satisfaisants
- Produits : Express d'Oracle, Media/MR de Speedware, Holos de Seagate Technology, ...

Introduction à la technologie HOLAP

- De nombreux systèmes commerciaux utilisent l'approche HOLAP :
  - manipulation des informations de l'entrepôt de données avec un moteur ROLAP
  - exploitation des *data marts* avec une approche multidimensionnelle MOLAP

Architecture générale de ces systèmes :



Quelques produits de technologie HOLAP

- DB2 OLAP Server :
  - Calcul, de consolidation et acces à l'information à partir des bases de données multidimensionnelles, relationnelles ou les deux
  - Composants :
    - DB2 OLAP Integration Server : Utilisation d'outils graphiques et des services pour l'intégration des données
    - DB2 OLAP Server Administration Services : outils pour améliorer et faciliter des tâches d'administration

Quelques produits de technologie HOLAP

- Oracle Express Server :
  - SGBD exploitant un modèle de données multidimensionnel
  - Gestion d'un ensemble d'indicateurs à n dimensions (valeurs sont stockées ou calculées dynamiquement)
  - Stockage des données dans BD multidimensionnelle ou relationnelle
    - base Oracle Express Server :
      Stockage les agrégats multidimensionnels
      Stockage des données de détail dans la base relationnelle
    - 4GL Express Server : fonctions avancées pour la présentation et l'analyse des résultats
- Autres produits HOLAP : Media/MR (Speedware), Hollos (Seagate), ...

Evolution de la norme SQL

- SQL/86
- SQL/89 (FIPS 127-1)
- SQL/89 with Integrity Enhancement
- SQL/92 (July 92)
  - Entry Level (FIPS 127-2)
  - Intermediate Level
  - Full Level
- SQL CLI (Sept 95)
- SQL PSM (Nov 96)
- SQL/3 (SQL 99)
  - SQL Framework : July 99
  - SQL Foundation : July 99
  - SQL Call Level Interface (CLI) : Sept 99
  - SQL Persistent Stored Modules(PSM) : July 99
  - SQL Language Bindings : withdrawal
  - SQL Management of External Data Dec 00
  - SQL Object Language Bindings Aug 00

SQL99 Overview

- Superset of SQL/92 : Completely upward compatible ("object-oriented SQL")
- Significantly larger than SQL/92
- Object-Relational extensions :
  - User-defined data types
  - Reference types
  - Collection types (e.g., arrays)
  - Large object support (LOBs)
  - Table hierarchies
- Triggers
- Stored procedures and user-defined functions
- Recursive queries
- OLAP extensions (CUBE, ROLLUP, expressions in ORDER BY)
- SQL procedural constructs
- Expressions in ORDER BY
- Savepoints
- Update through unions and joins

Nouvelles fonction SQL d'agrégation

- Nouvelles fonctions SQL d'agrégation:
  - `N_tile`
  - `rank`, `dense_rank`
  - `every`
  - `any`
  - `some`
- Nouvelles fonctions de la clause GROUP BY :
  - `ROLLUP`
  - `CUBE`
  - `GROUPING`
  - `GROUPING SETS`
- Fenêtre glissante :
  - `WINDOWS/OVER/PARTITION`, ...

Nouvelles fonctions de la clause `GROUP BY`

- Fonctions classiques d'agrégation de SQL : `count`, `max`, `min`, `sum`, `avg`
- Ajout de nouvelles fonctions pour faire des calculs comme c'est possible dans les tableurs : `N_tile(expression, n)`
  - Calcul du domaine de l'expression en fonction des valeurs qui apparaissent dans la colonne
  - Division du domaine en n intervalles de tailles approximativement égales
  - La fonction retourne alors le numéro de l'intervalle qui contient la valeur de l'expression

Nouvelles fonctions de la clause GROUP BY

- Exemples d'utilisation
  - *si le solde du compte est parmi les 10% les plus élevés*

    N_tile(compte.solde, 10) retourne 10

  - *Trouver le min, le max et la moyenne des températures parmi les 10% les plus élevées*

```
SELECT Percentile, MIN(Temp), AVG(Temp), MAX(Temp)
FROM Weather
GROUP BY N_tile(Temp,10) as Percentile
HAVING Percentile = 10;
```

Nouvelles fonctions SQL de classement

- `rank(expression)` : rang de l'expression dans l'ensemble de toutes les valeurs du domaine (colonne)
- Exemples : la table `population(country, number)`
  - *Trouver le classement de chaque pays*

```
SELECT country, rank()
OVER (ORDER BY number DESC) AS n_rank
FROM population
```

Utilisation de la clause `ORDER BY` pour retourner les résultats triés :

```
SELECT country, rank()
OVER (ORDER BY number DESC) AS n_rank
FROM population ORDER BY n_rank
```

Nouvelles fonctions SQL de classement

- Exemples (suite) : la table `population(country, number)`
  - *trouver les 5 premiers pays les plus peuplés* (syntaxe DB2)

```
SELECT country, rank()
OVER (ORDER BY number DESC) AS n_rank
FROM population
ORDER BY n_rank
FETCH FIRST 5 ROWS ONLY
```

Autres nouvelles fonctions SQL

`dense_rank(expression)`: pas de prise en compte des doublons dans le calcul du rang

- avec la fonction `rank`, s'il y a 2 tuples qui ont les mêmes valeurs et sont classés de rang n, le tuple suivant est de rang $n + 2$
- avec `dense_rank`, le tuple suivant a le rand $n + 1$

Exemple : *Trouver le classement de chaque pays*

```
SELECT country, dense_rank()
OVER (ORDER BY number DESC) AS n_rank
FROM population
```

Autres nouvelles fonctions SQL

- `every(expression)` : vrai ssi la valeur de son argument (expression) est vraie pour tous les tuples, sinon faux ($\forall$)
- Exemple : table :
  `Film_v(titre, type_film, année, magasin, qté_stock, qté`
  Trouver la quantité maxi et mini de films vendus à plus de 10 exemplaires

```
SELECT COUNT(*), MAX(qté-vendus), MIN(qté-vendus),
SUM(qté-vendus), AVG(qté-vendus)
FROM
 Film_v
GROUP BY magasin Having EVERY(qté-vendus>=10)
```

- `any(expression)` / `some(expression)` : vrai ssi la valeur de son argument (expression) est vraie pour au moins un tuple, sinon faux ($\exists$)

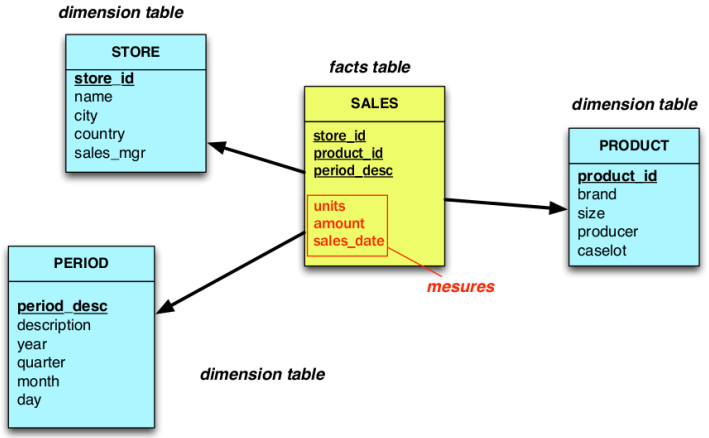
Nouvelles fonctions de la clause GROUP BY

`CUBE`, `ROLLUP` et `GROUPING SETS` : production d'un ensemble de tuples, équivalent à un UNION ALL de tuples groupés différemment

- `ROLLUP` : calcul des agrégats (`SUM`, `COUNT`, `MAX`, `MIN`, `AVG`) à différents niveaux d'agrégation
- `CUBE` : similaire à `ROLLUP` mais calcul toutes les combinaisons d'agrégations
- `GROUPING SETS` : pas de calcul du cube, quand il n'est pas globalement nécessaire
- Fonctions `GROUPING` : désignation du groupe d'appartenance de chaque tuple pour calculer les sous-totaux et les filtres

Nouvelles fonctions de la clause GROUP BY

Exemple



Création du cube Sales :

```
CREATE VIEW Sales AS
(SELECT ds.*, YEAR(sales_date) AS year, MONTH(sales_date) AS
month, DAY(sales_date) AS day
FROM (Sales NATURAL JOIN Store NATURAL JOIN Product
NATURAL JOIN Period) ds
```

GROUP BY ROLLUP

`GROUP BY ROLLUP` :

- Calcul des agrégats (`SUM`, `COUNT`, `MAX`, `MIN`, `AVG`) à tous les niveaux de totalisation sur une hiérarchie de dimensions
- calcul le total général :
  - Selon l'ordre de gauche à droite dans la clause `GROUP BY`
  - S'il y a n colonnes de regroupements, `GROUP BY ROLLUP` génère $n + 1$ niveaux de totalisation
- Exemples :

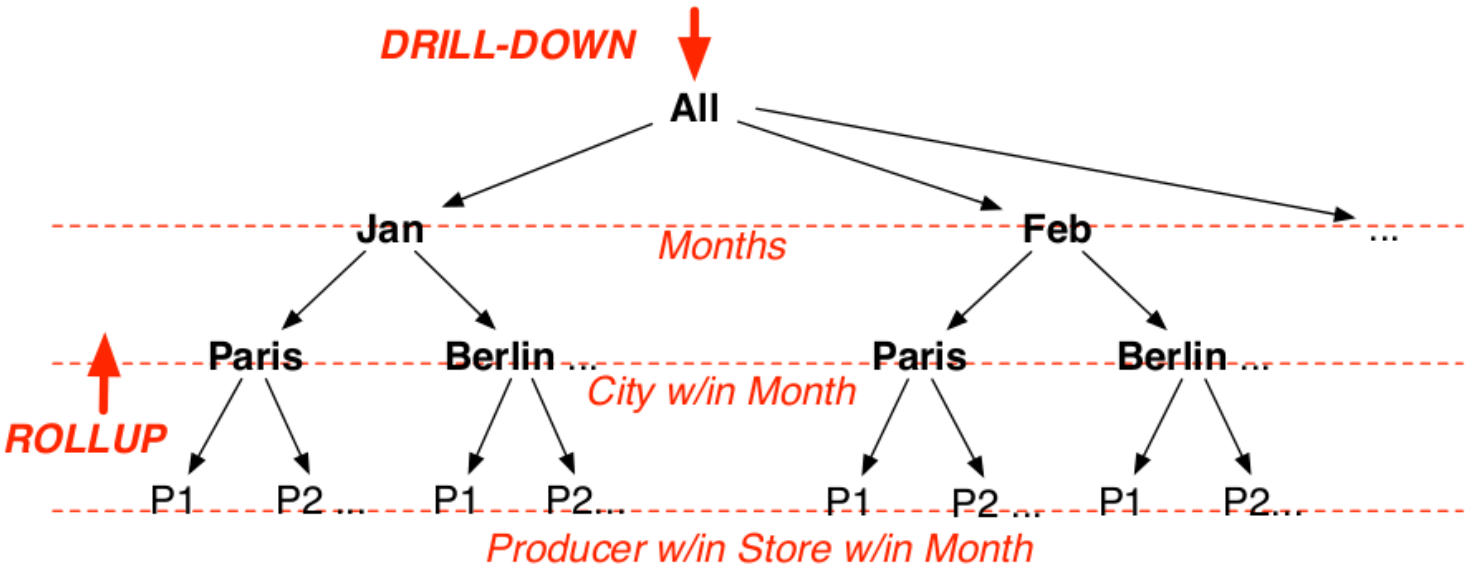
```
GROUP BY ROLLUP (year, month, day)
GROUP BY ROLLUP (country, city)
GROUP BY ROLLUP (month, city, producer)
```

- Simplification et accélération de la maintenance des tables de synthèse

GROUP BY ROLLUP

Exemple

```
SELECT month, city, producer, SUM(units) AS sum_units
FROM Sales
WHERE year = 2014
GROUP BY ROLLUP (month, city, producer) ;
```



GROUP BY ROLLUP

Exemple

Total des ventes par pays et responsable des ventes pour chaque mois de 2014, avec sous-totaux pour chaque mois, et grand total

```
SELECT month, country, sales_mgr, SUM(amount)
FROM Sales WHERE year = 2014
GROUP BY ROLLUP(month, country, sales_mgr)
```

Month	Country	Sales-mgr	SUM(amount)	
April	France	Martin	25000	
April	France	Smith	15000	
April	France	-	40000	Total France/April
April	Germany	Smith	15000	
April	Germany	-	15000	Total Germany/April
April	-	-	55000	Tot. April
May	France	Martin	25000	
May	France	-	25000	Total France/May
May	Germany	Smith	15000	
May	Germany	-	15000	Total France/May
May	-	-	40000	Total May
-	-	-	95000	Grand Total

GROUP BY CUBE

`GROUP BY CUBE` :

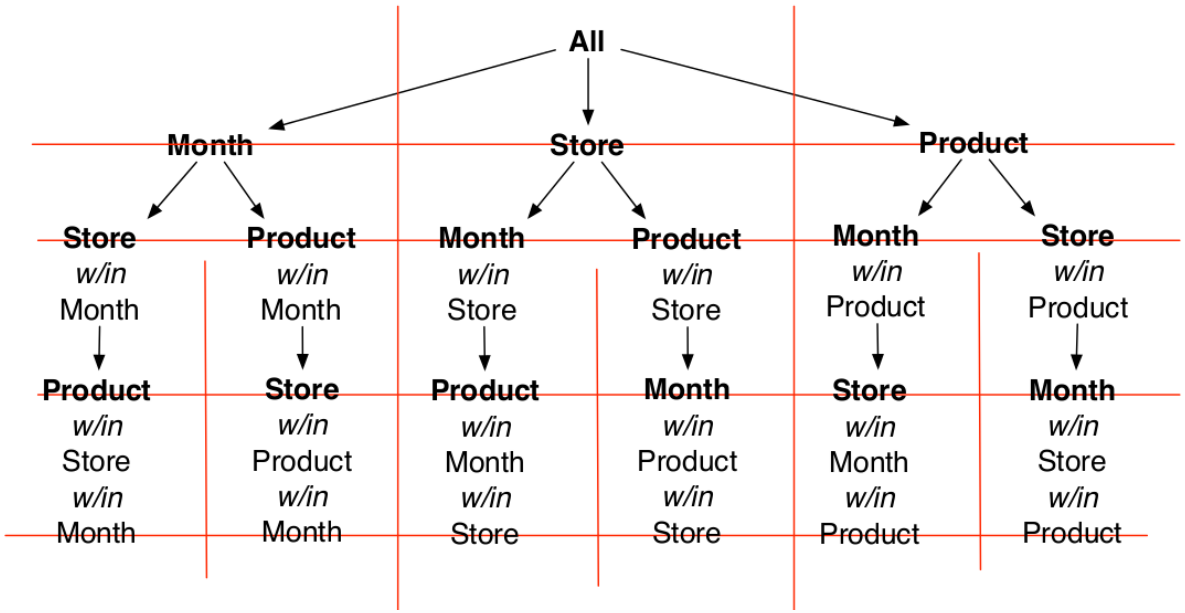
- Calcul des agrégats (`SUM`, `COUNT`, `MAX`, `MIN`, `AVG`) à différents niveaux d'agrégation comme ROLLUP
- Mais aussi calcul toutes les combinaisons d'agrégations :
  - Création des sous-totaux pour toutes les combinaisons possibles d'un ensemble de colonnes de regroupement
  - Si la clause `CUBE` contient n colonnes, `CUBE` calcule $2n$ combinaisons de totaux
  - Intérêt : lorsque les colonnes représentent des dimensions appartenant à des hiérarchies différentes
- Alternative plus performante à `UNION ALL`

GROUP BY CUBE

Exemple

Tous les totaux et sous totaux des quantités de produits vendus par ville et produit pour chaque mois de 2014

```
SELECT month, city, product_id, SUM(units)
FROM Sales
WHERE year = 2014
GROUP BY CUBE (month, city, product.id)
```



GROUP BY CUBE

Tous les totaux et sous totaux des ventes par pays et responsable des ventes pour chaque mois de 2014

```
SELECT month, country, sales_mgr, SUM(amount)
FROM Sales
WHERE year = 2014
GROUP BY CUBE(month, country, sales_mgr)
```

GROUP BY CUBE

Month	Country	Sales-mgr	SUM(amount)	
April	France	Martin	25000	
April	France	Smith	15000	
April	France	-	40000	Total France/April
April	Germany	Smith	15000	
April	Germany	-	15000	Total Germany/April
April	-	Martin	25000	Total Martin/April
April	-	Smith	30000	Total Smith/April
April	-	-	55000	Total April
May	France	Martin	25000	
May	France	-	25000	Total France/May
May	Germany	Smith	15000	
May	Germany	-	15000	Total Germany/May
May	-	Martin	25000	Total Martin/May
May	-	Smith	15000	Total Smith/May
May	-	-	40000	Total May
-	France	Martin	50000	Total Martin/France
-	France	Smith	15000	Total Smith/France
-	France	-	65000	Total France
-	Germany	Smith	30000	Total Smith/Germany
-	Germany	-	30000	Total Germany
-	-	Martin	50000	Total Martin
-	-	Smith	45000	Total Smith
-	-	-	95000	Grand Total

GROUP BY GROUPING SETS

- Utilisation avec les agrégations usuelles (`SUM`, `COUNT`, `MAX`, `MIN`, `AVG`)
- Groupements multiples (`month`, `country`) et (`month, sales_mgr`)
- Possibilité de restreindre les résultats par une clause `HAVING`

GROUP BY GROUPING SETS

Exemple

Total des ventes pour chaque mois de l'année 2014, par pays et par responsable des ventes

```
SELECT month, country, sales_mgr, SUM(amount)
FROM Sales
WHERE year = 2014
GROUP BY GROUPING SETS((month, country),(month, sales_mgr))
```

Month	Country	Sales-mgr	SUM(amount)	
April	France	-	40000	Total France/April
April	Germany	-	15000	Total Germany/April
April	-	Martin	25000	Total Martin/April
April	-	Smith	30000	Total Smith/April
May	France	-	25000	Total France/May
May	Germany	-	15000	Total Germany/May
May	-	Martin	25000	Total Martin/May
May	-	Smith	15000	Total Smith/May

GROUP BY GROUPING SETS

Inclusion du tuple *grand total* dans les résultats :

- Génération implicite des grands totaux avec ROLLUP et CUBE
- Ici, pas de génération avec un agrégat supplémentaire

Total de ventes par mois, pays, et responsable de vente, et aussi grand total de vente

```
SELECT month, country, sales_mgr, SUM(amount)
FROM Sales
WHERE year = 2014
GROUP BY GROUPING SETS((month, country), ())
```

Month	Country	Sales-mgr	SUM(amount)
April	France	Martin	25000
April	France	Smith	15000
April	Germany	Smith	15000
May	France	Martin	25000
May	Germany	Smith	15000
-	-	-	95000

Fonction GROUPING

- Création d'une nouvelle colonne avec des 0 et des 1
- Détection de tuples de total qui sont générés lors de l'exécution de CUBE ou ROLLUP (1 pour les tuples de total et 0 pour les autres)

Tous les totaux et sous totaux des ventes par pays et responsable des ventes pour chaque mois de 2014

```
SELECT month, country, sales_mgr, SUM(amount), GROUPING(sales_mgr)
FROM Sales
WHERE year = 2014
GROUP BY ROLLUP (month, country, sales_mgr)
```

Month	Country	Sales-mgr	SUM(amount)	GROUPED
April	France	Martin	25000	0
April	France	Smith	15000	0
April	France	-	40000	1
April	Germany	Smith	15000	0
April	Germany	-	15000	1
April	-	-	55000	1
May	France	Martin	25000	0
May	France	-	25000	1
May	Germany	Smith	15000	0

Fenêtre glissante

Pour répondre à des questions comme :

- Trouver pour chaque mois, les ventes de chaque magasin en faisant la moyenne entre le mois courant et les deux mois précédents : faire la moyenne sur les 3 derniers mois
- Trouver pour chaque magasin, les ventes cumulées pour chaque mois de l'an dernier, ordonnées par mois
- Comparer les ventes de chaque mois avec la moyenne des ventes de chaque magasin pendant les 3 mois précédents

$\rightarrow$ Nécessite de définir une fenêtre glissante du temps et y effectuer des cumuls, moyennes, dérivations, ...

Fenêtre glissante

Notion de fenêtre :

- Dans une requête, une fenêtre est une sélection de lignes (tuples) utilisée pour un certain calcul par rapport à la ligne courante
- La taille de la fenêtre est un nombre de lignes mais peut être exprimée de différentes façons (nombre entier ou par une condition)
- Une fenêtre se déclare soit explicitement par une clause `WINDOW` ou directement dans le `SELECT` où elle est utilisée

Fenêtre glissante

3 paramètres :

1. Partitionnement `PARTITION` : Chaque ligne de la partition a des valeurs égales sur un ensemble de colonnes (analogue au `GROUP BY`)
2. Ordre `ORDER` : Ordre des lignes dans la partition comme un tri partiel sur 1 ou plusieurs colonnes (attention aux valeurs nulles)
3. Cadre `FRAMING` : Définition de la taille de la fenêtre de manière physique (nombre de lignes – `ROWS`) ou logique (condition, plage de valeurs – `RANGE`) -

   ROWS 2 PRECEDING

Fenêtre glissante

Exemple

Trouver les populations de chaque pays et année, calculer la moyenne du taux de croissance de la population pour chaque pays et année, sur la base des années courante, précédente et suivante

Table : `population(country, year, number)`

```
SELECT country, year, avg(population)
OVER (ORDER BY country, year
ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING) AS p_avg
FROM population
ORDER BY country, year, p_avg;
```

(source : Adiba & Fauvet, 2005)

Fenêtre glissante

Exemple

Calculer pour chaque magasin et pour chaque mois du dernier tiers de 2001 le total des ventes, ainsi que la moyenne glissante des quantités vendues avec les 2 mois précédents

Table `Fvm(magasin, anmois, qté_totale)`

```
SELECT h.magasin, h.anmois, h.qté_totale,
AVG(h.qté_totale) OVER w AS moygliss
FROM
Fvm h
WHERE h.anmois BETWEEN 200109 AND 200112
WINDOW w AS
(PARTITION BY h.magasin ORDER BY h.anmois ROWS 2 PRECEDING)
```

(source : Adiba & Fauvet, 2005)

Fenêtre glissante

Exemple (groupe incomplet)

Calculer pour chaque magasin et pour chacun des 4 derniers mois 2001, le total des ventes ainsi que la moyenne glissante des quantités vendues sur les 2 mois précédents à condition qu'il y ait bien 2 mois, sinon NULL

Table `Fvm(magasin, anmois, qté_totale)`

```
SELECT h.magasin, h.anmois, h.qté_totale,
CASE WHEN Count(*) OVER w < 3 THEN NULL
ELSE AVG(h.qté_totale) OVER w End AS moygliss
FROM Fvm h Where h.anmois BETWEEN 200109 AND 200112
WINDOW w AS
(PARTITION BY h.magasin ORDER BY h.anmois ROWS 2 PRECEDING)
```

(source : Adiba & Fauvet, 2005)

Introduction à MDX

Origine de MDX

- MDX :
  - *Multi Dimensional eXpression*,
  - Langage de requêtes OLAP pour les bases de données multidimensionnelles
- Historiqye
  - Inventé en 1997 par Mosha Pasumansky au sein de Microsoft,
  - Version commerciale Microsoft OLAP Services 7.0 & Analysis Services en1998
  - Dernière spécification OLE DB for OLAP (ODBO) en 1999
  - adopté les autres systèmes OLAP $\rightarrow$ devenu un standard

Origine de MDX

- Objectifs :
  - naviguation dans les bases multidimensionnelles
  - définition des requêtes sur tous leurs objets (dimensions, hiérarchies, niveaux, membres et cellules)
- Requête MDX : rapport à plusieurs dimensions consistant en un ou plusieurs tableaux 2D
- Utilisé par de nombreux outils de BI commerciaux ou non
- Langage très complexe et puissant générant des requêtes plus compacte que les requêtes SQL équivalentes

MDX versus SQL

- Syntaxe de MDX similaire à celle de SQL
- Mots clé `SELECT`, `FROM`, `WHERE` mais leurs sémantiques sont différentes :
  - SQL : construction de vues relationnelles
  - MDX construction de vues multidimensionnelles des données

MDX versus SQL

- Analogies entre termes multidimensionnels (MDX) et relationnels (SQL)

Multidimensionnel (MDX)	Relationnel (SQL)
Cube	Table
Niveau (`Level`)	Colonne (chaine de caractère ou valeur numérique)
Dimension	plusieurs colonnes liées ou une table de dimension
Mesure (Measure)	Colonne (discrète ou numérique)
Membre de dimension (*Dimension member*)	Valeur dans une colonne et une ligne particulière de la table

MDX versus SQL

- Structure générale d'une requête :
  - SQL :
    `SELECT column1, column2, ..., columnn FROM table`
  - MDX :
    `SELECT axis1 ON COLUMNS, axis2 ON ROWS FROM cube`
- Clause `FROM` : Spécification de la source de données
  - SQL : une ou plusieurs tables
  - MDX : un cube

MDX versus SQL

Clause `SELECT` : indication des résultats que l'on souhaite récupérer par la requête

- en SQL :
  - Une vue des données en 2 dimensions : lignes et colonnes
  - Les lignes ont la même structure définie par les colonnes
- en MDX :
  - Nombre quelconque de dimensions pour former les résultats de la requête
  - On parle axe pour éviter confusion avec les dimensions du cube
  - Pas de signification particulière pour les lignes et les colonne
  - Mais il faut définir chaque axe : `axe1` définit l'axe horizontal et `axe2` définit l'axe vertical

Exemple de requête MDX

```
SELECT {Paris, Berlin} ON ROWS
       {[Q1], [Q2].CHILDREN} ON COLUMNS
FROM CubeSales
WHERE (MEASURES.SalesAmount,
       Time.[2014],
       Product.Product)
```

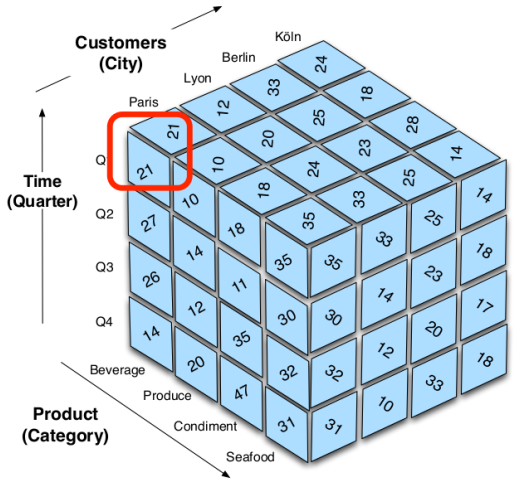
Entrepôt de données simplifiés

- Mesures : Unit Price, Quantity, Discount, SalesAmount, Freight
- Dimension : Temps
  - hiérarchie : *Annee* > *trimestre* > *Mois* >, membres :
    - Annee : 2010, 2011, 2012, 2013, 2014
    - Trimestre : Q1, Q2, Q3, Q4
    - Mois : January, February, March, ...
- Dimension : Client
  - hiérarchie : *Continent* > *Pays* > *Region* > *City*, membres :
    - Ville: Paris, Lyon, Berlin, Köln, Marseille, Nantes ...
    - Region: Loire atlantique, Bouches du Rhône, Bas Rhin, Torino...
    - Pays: Austria, Belgium, Danmark, France, ...
    - Continent level: Europe, North America, Sud America, Asia
- Dimension : Produit
  - hiérarchie : *Categorie* > *Sous* − *categorie* > *produit*, membres :
    - Categorie : Food, Drink ...
    - Sous-categorie `Food`: `Baked_food` ...
    - ...

Exemple

Cube *Sales* :

- 3 dimensions : Time, Customer, Product
- 1 mesure : SalesAmount



Cellule en haut à gauche : valeur 21, vente de boissons à Paris durant le premier trimestre (Q1)

(Product.Category.Beverage ; **Time**.Quarter.Q1 ; Customer.City.Paris)

(inspired from Vaisman & Zimányi)

Syntaxe de base de MDX

Structure générale d'une requête MDX

- Syntaxe générale d'une requête MDX :

```
SELECT [<specification d'un axe>
        [, <spécification d'un axe>...]]
FROM [<spécification d'un cube>]
[WHERE [<spécification d'un filtre (slicer)>]]
```

- Parenthèses en MDX :
  - { } : Ensemble des éléments servant à la création d'une dimension du résultat de la requête
  - ( ) : Sélection de tuples dans la clause WHERE
  - [ ] : Représentation d'espaces, de caractères spéciaux et d'interprétation non numérique de chiffres.

NB: [ ] : optionnels, sauf si le nom contient des caractères espace, des chiffres, ou est un mot-clé MDX

Structure générale d'une requête MDX

`SELECT` : description des axes du cube résultat
Chaque dimension du résultat :

- est associée à un rôle correspondant à sa représentation dans le tableau retourné par la requête MDX :

  ON COLUMNS, **ON ROWS**, **ON** PAGES, **ON** SECTIONS, **ON** CHAPTERS

- sur un ou plusieurs niveaux de la hiérarchie :
  `{Paris, Berlin}` (dimension Lieu, niveau Ville)
  `{[1er trimestre], [2nd trimestre].CHILDREN}`
  (dimension Temps, niveaux trimestre et mois)

Structure générale d'une requête MDX

`FROM` : Spécification du/des cube/s de départ

- Ensemble de cubes nécessaires à la création du cube résultat
- Si plusieurs cubes nécessaires : jointure multidimensionnelle chaque paire de cubes doit alors posséder au moins une dimension concordante

`WHERE` :Restriction sur le/s cube/s de départ

- Restrictions sur le/s cube/s de départ de la clause `FROM`
- Spécification des restrictions par une liste de noeuds de la hiérarchie d'une dimension nommée *slicer-dimension*

NB : les mesures sont des éléments d'une dimension spéciale nommée *Measures* (utiliséables aussi dans les clauses `WHERE` et `FROM`).

Membres en MDX

- membre : une instance d'un niveau d'une dimension
- généralement spécifié entre crochets `[...]`
- Exemple : membres de la dimension "Products" de niveau 1
  `[Food]`, `[Drink]`
- Membres : éléments accessibles dans les hiérarchies pouvant être référencés de différentes façons :

```
[2012]
[Time].[2012]
[Product].[Food]
[Product].[Food].[Baked Goods]
[Product].[All Products].[Food].[Baked Goods]
```

Membres en MDX

- enfants d'un membre : membres du niveau immédiatement en dessous de celui-ci

  Exemple dans des requêtes simples :

```
SELECT [Time].[2012] ON COLUMNS FROM [Sales]
SELECT [Product].[Food] ON COLUMNS FROM [Sales]
SELECT [Product].[Food].[Baked Goods] ON COLUMNS
    FROM [Sales]
```

Conclusion et perspectives

- Deux tendances actuelles
  - datamarts
  - dataweb
- construction du Data Warehouse : processus long et difficile
- Construction progressive par datamarts
  - Avantage : rapide
  - Inconvénient : risque de cohabitation de datamarts incohérents
- Dataweb
  - Ouverture du data warehouse au web

Solutions commerciales

Solutions intégrées

- Business Object
- SAS
- Oracle
- IBM DB2
- SQL Server

Solutions Open-sources (1)

Composants :

- ETL : Octopus, Talend, Kettle, CloverETL
- DWH : MySQL, Postgresql, GreenPlum/Bizgres
- OLAP : Mondrian, Palo
- Reporting : Birt, Open Report, Jasper Report, JFreeReport
- Data Mining : Weka, R, Xelopes

Solutions Open-sources (2)

Solutions intégrées :

- Pentaho (Kettle, Mondrian, JFreeReport)
- SpagoBI

Solutions spécifiques :

- I2B2 Data Warehouse
- Clinical Research / Hospital data warehouse

Plus loin...

Big data warehouses

- Volume
  - Optimisation/parallélisation des agrégations
  - OLAP dans un cloud
- Variété
  - Nouveaux modèles multidimensionnels et opérateurs d'agrégation
  - Entrepôts NoSQL
- Vélocité
  - Travailler en mémoire : problème de l'explosion dimensionnelle
  - Fonctions d'oubli
- Véracité
  - Qualité des données sources
  - Sécurité des données entreposées

Plus loin ...

- Tableur collaboratif
- Décisionnel sur mobile
  Utilisateur non-expert
- Confidentialité : données décisionnelles partagées dans le nuage