SYSTÈMES D'INFORMATION DÉCISIONNELS

DATA WAREHOUSE: OLAP

Département d'informatique

Module: Systèmes d'information décisionnels

Chargé du module: Mokeddem, S

Email: sidahmed.mokadem@univ-mosta.dz

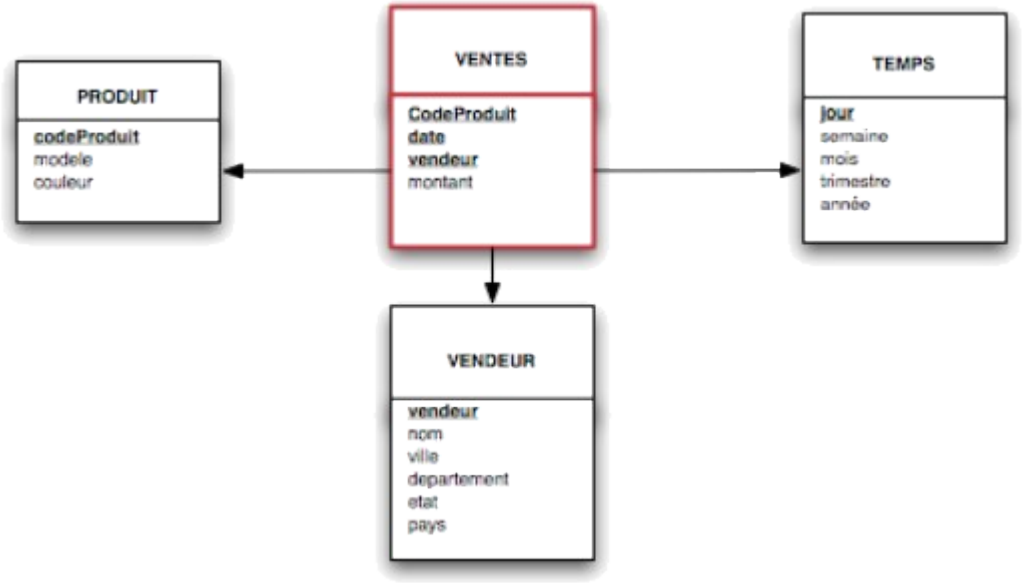
Année universitaire: 2018/2019

OLAP (ON LINE ANALYTICAL PROCESSING)

- Les applications OLAP sont des applications d'aide à la décision
- Elles sont constituées de traitements ensemblistes réduisant une population à une valeur ou un comportement
  - Des traitements semi-automatiques visant à interroger, visualiser et synthétiser les données, traitements définis et mis en œuvre par les décideurs
- Les requêtes OLAP sont exécutées sur le DW
  - On-Line :le processus se fait en ligne, l'utilisateur doit avoir la réponse de façon quasi-instantanée

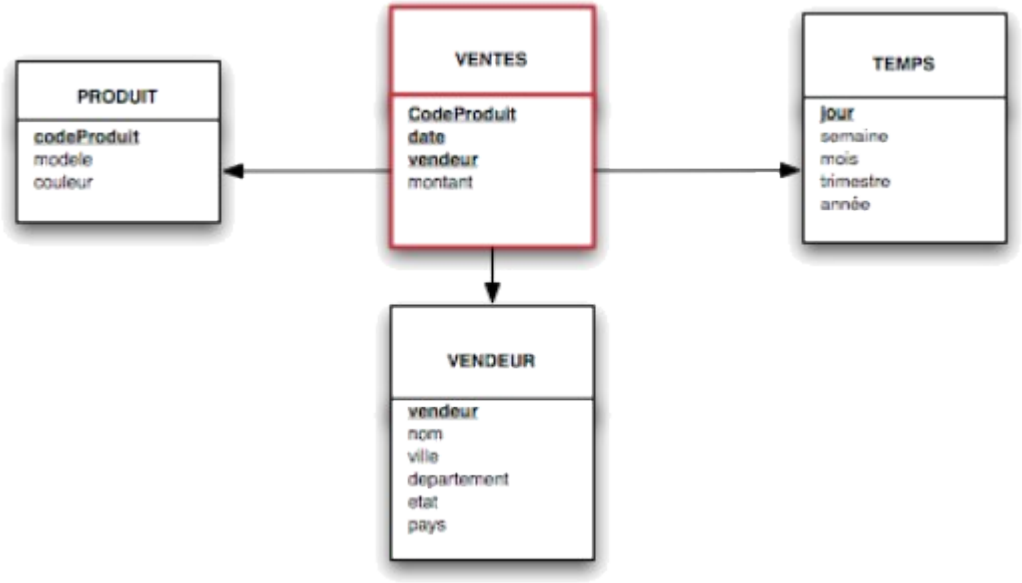
EXEMPLE DW

- Soit DW en schéma étoile suivant :
  - ventes(codeProduit, date, vendeur, montant) (table faits)
  - produits(codeProduit, modèle, couleur) (table dimension)
  - vendeurs(nom, ville, département, état, pays) (table dimension)
  - temps(jour, semaine, mois, trimestre, année) (table dimension)



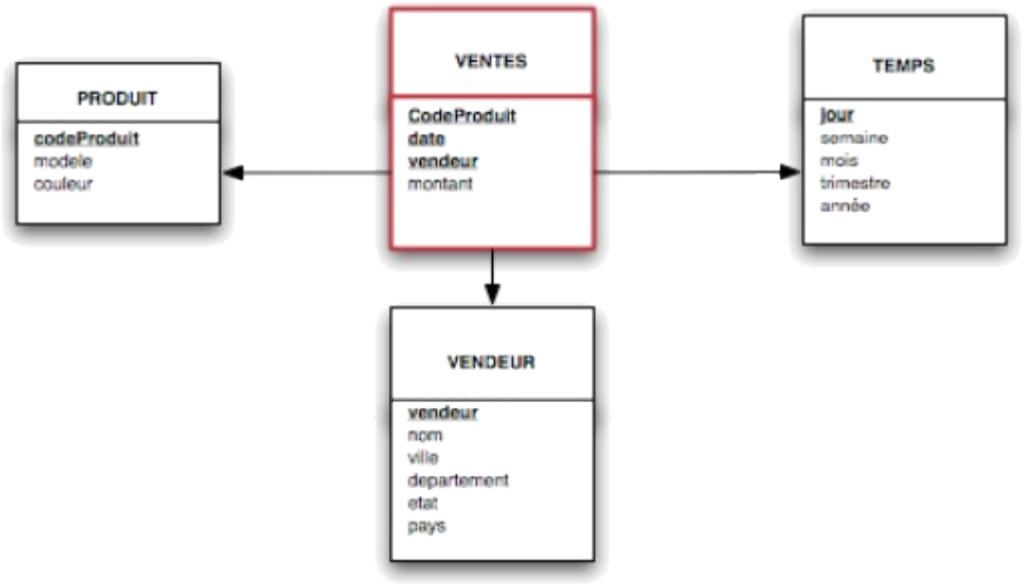
EXEMPLE DW

- Soit DW en schéma étoile suivant :
  - ventes(codeProduit, date, vendeur, montant) (table faits)
  - produits(codeProduit, modèle, couleur) (table dimension)
  - vendeurs(nom, ville, département, état, pays) (table dimension)
  - temps(jour, semaine, mois, trimestre, année) (table dimension)



EXEMPLE DW

- Selon la notation de Golfarelli (1998) :

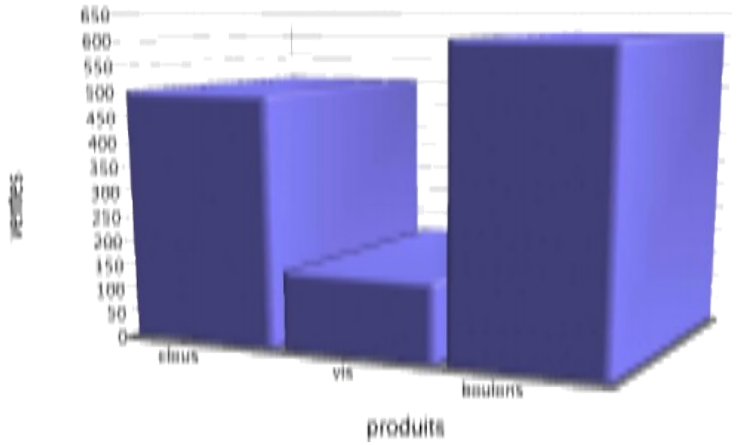


BESOIN D'ANALYSE

- Exemple de questions associées :
- 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 ?

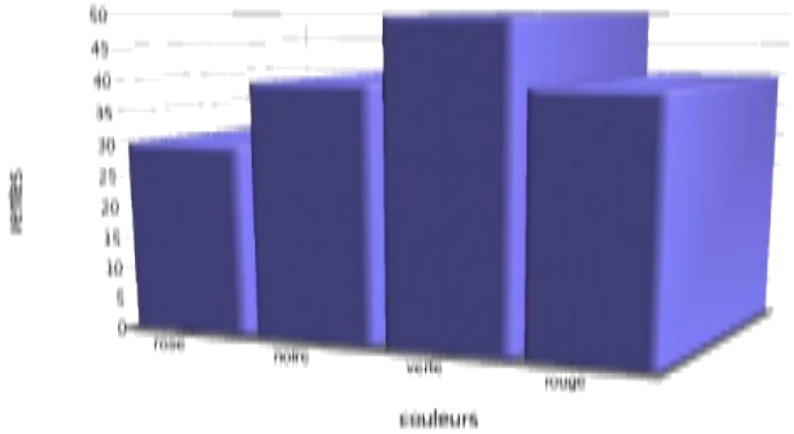
BESOIN D'ANALYSE

- Analyse des ventes de divers produits :
  - SELECT modele, SUM(montant) FROM ventes, produits WHERE ventes.codeProduit = produits.codeProduit GROUP BY modele ;



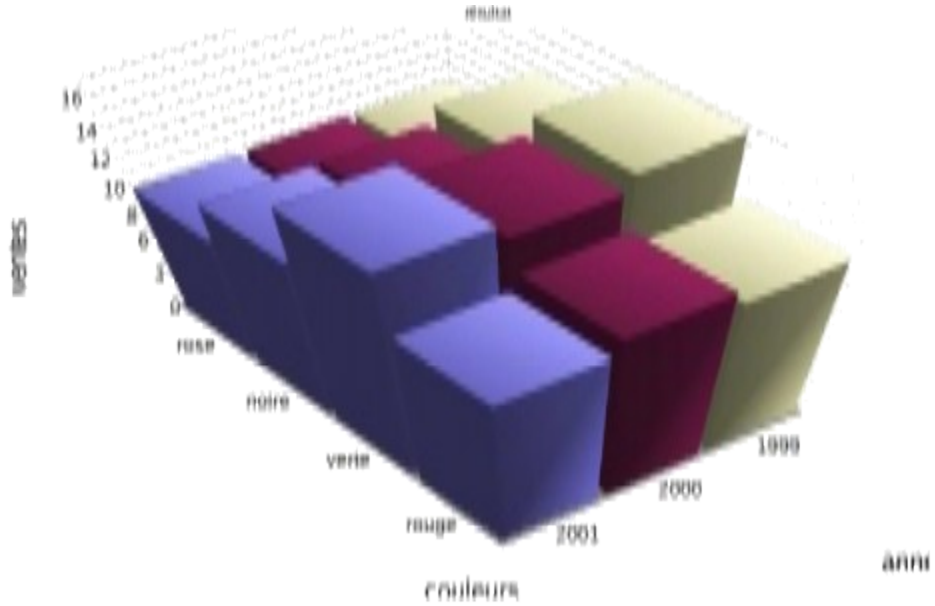
BESOIN D'ANALYSE

- Les ventes de vis sont plus faibles que prévu... quelles couleurs sont responsables ?
  - SELECT couleur, SUM(montant) FROM ventes, produits WHERE ventes.codeProduit = produits.codeProduit AND modele = "vis" GROUP BY couleur ;



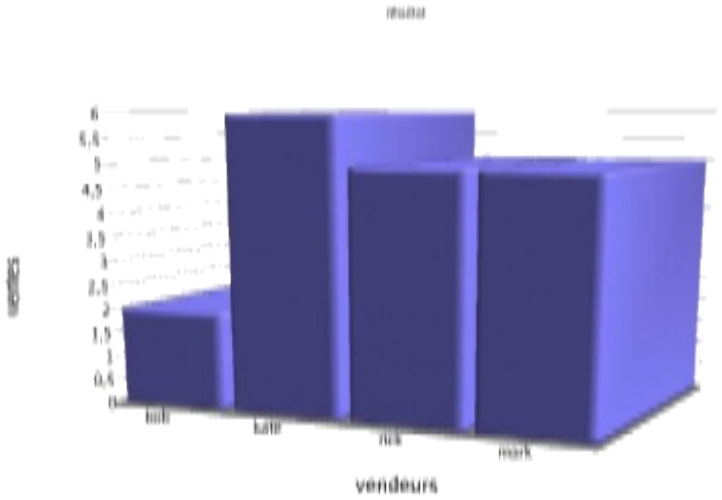
BESOIN D'ANALYSE

- Les ventes de vis sont plus faibles que prévu... quelles années sont responsables ?
  - SELECT couleur, annees, SUM(montant) FROM ventes, produits, temps WHERE ventes.codeProduit = produits.codeProduit AND ventes.date = temps.jour AND modele = "vis" GROUP BY couleur, annees ;



BESOIN D'ANALYSE

- Les ventes de vis sont plus faibles que prévu... Quels vendeurs sont responsables ?
  - SELECT vendeur, somme FROM(SELECT trimestre, vendeur, SUM(montant) as somme FROM ventes, produits, temps, vendeur WHERE ventes.codeProduit = produits.codeProduit AND ventes.date = temps.jour AND ventes.vendeur = vendeurs.nom AND modele = "vis" GROUP BY trimestre, vendeur) WHERE trimestre = "jui-sep";



PROBLEMATIQUE OLAP

- Besoins spécifiques :
  - langages de manipulation
  - organisation des données
  - fonctions d'agrégation
- Organisation des données proche des abstractions de l’analyste :
  - selon plusieurs dimensions
  - selon différents niveaux de détail
  - en ensemble
  - donnée = point dans l'espace associé à des valeurs

DU TABLE AU CUBE

De la table …

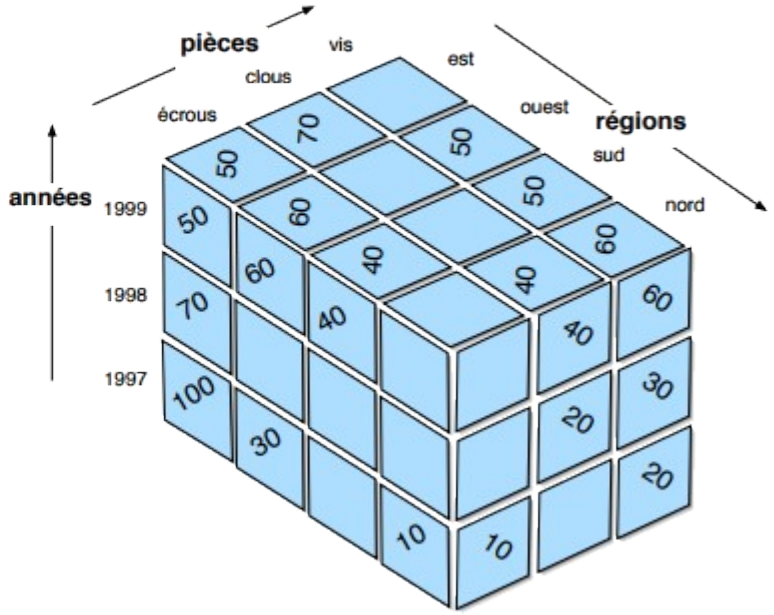
Table Ventes :

VENTES	pièces	Régions	Années	quantités
	écrous	est	1999	50
	clous	est	1997	100
	vis	ouest	1998	50
	…	…	…	…
	écrous	est	total	220
	…	…	…	…
	écrous	total	total	390
	…	…	…	…
	total	total	total	1200

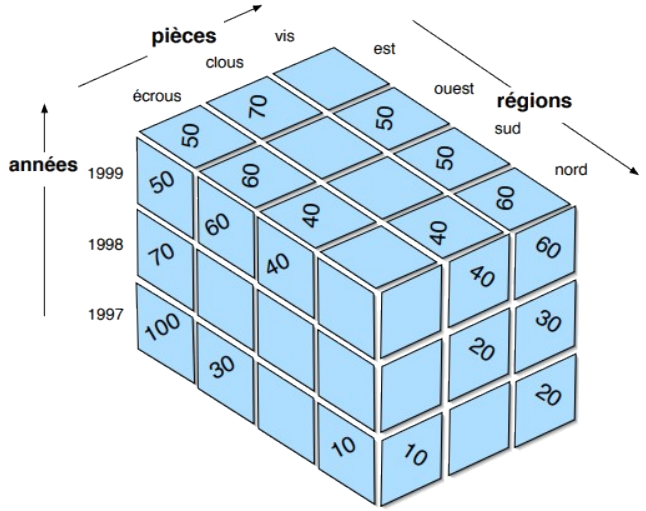
(pièce, région, année) → quantité

… au cube

Cube Ventes :



TERMINOLOGIE CUBE



Terme	Valeur
Cube	Ventes
Cellule	ecrous, est, 1997, 100
Référence	ecrous, est, 1997
mesure	100
Membre/paramètre	est
dimension	lieu
niveau	région

OPÉRATIONS ÉLÉMENTAIRES OLAP

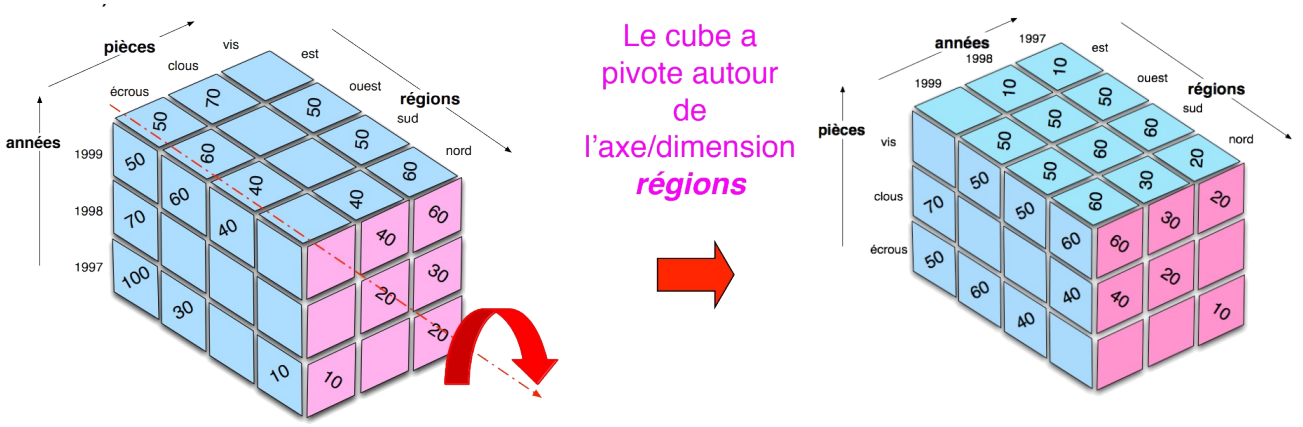
- Catégories d'opérations OLAP
  - Opérations de restructuration : rotate, switch, split, nest, push, pull
  - Opérations de granularité : roll-up, drill-down
  - Opérations ensemblistes : slide, dice

OPÉRATIONS ÉLÉMENTAIRES OLAP

- Restructuration : concerne la représentation, permet un changement de points de vue selon différentes dimensions : opérations liées à la structure, manipulation et visualisation du cube :
  - Rotate/pivot
  - Switch
  - Split, nest, push, pull
- Granularité : concerne un changement de niveau de détail : opérations liées au niveau de granularité des données : roll-up, drill-down
- Ensembliste : concerne l'extraction et l'OLTP classique :
  - slice, dice
  - selection , projection et jointure (drill-across)

OPÉRATIONS RESTRUCTURATION

- Rotate ou Pivot :
  - Effectuer à un cube une rotation autour d'un de ses trois axes passant par le centre de 2 faces opposées, de façon à présenter un ensemble de faces différent
  - Une sorte de sélection de faces et non des membres.



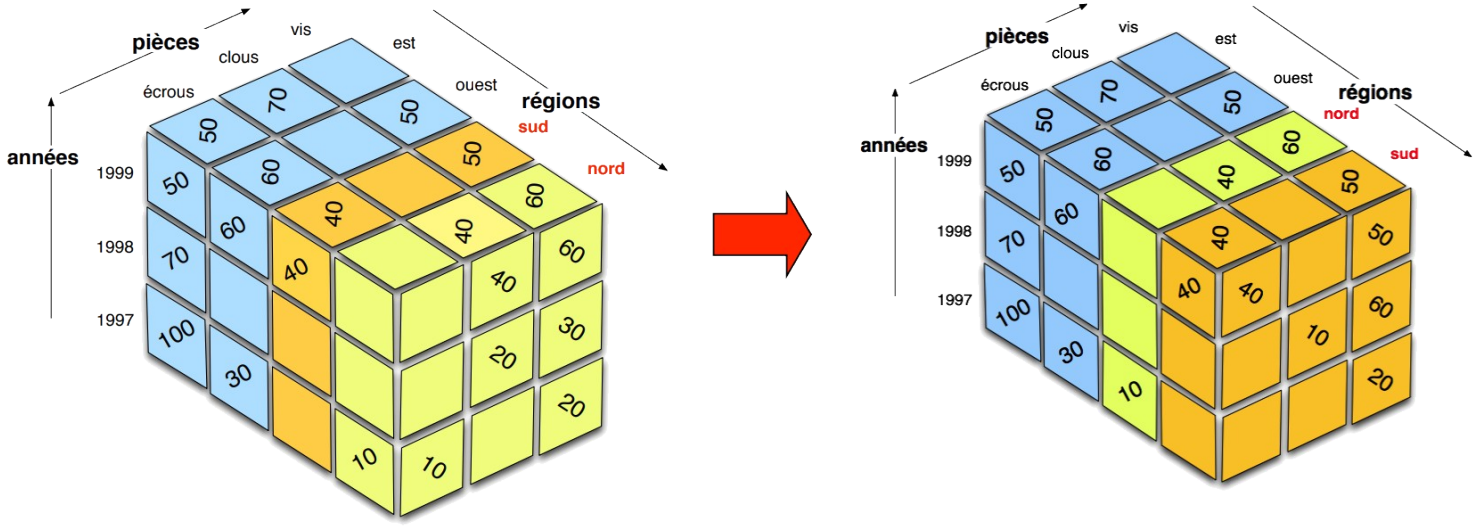
la visualisation résultante est souvent 2D :

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

vis	1999	1998	1997
est		10	10
ouest	50	50	50
sud	50	60	60
nord	60	30	20

OPÉRATIONS RESTRUCTURATION

- Switch ou permutation :
  - consiste à inter-changer la position des membres d'une dimension.



Ici sont interchangés les membres *nord* et *sud* de la dimension *régions*

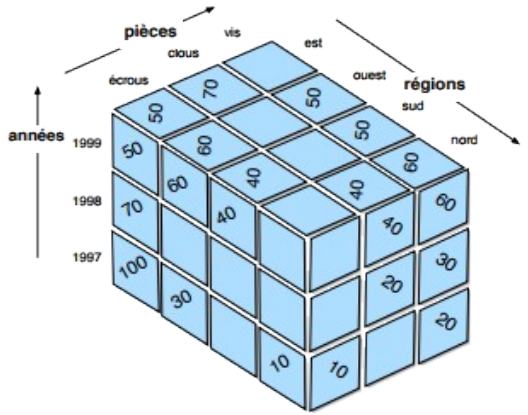
visualisation résultante est souvent 2D :

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

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

OPÉRATIONS RESTRUCTURATION

- Split ou division :
  - consiste à présenter chaque tranche du cube et de passer d'une présentation tridimensionnelle d'un cube à sa présentation sous la forme d'un ensemble de tables
  - sa généralisation permet de découper un hypercube de dimension 4 en cubes.



ici un **split(region)** du cube Ventes conduit aux 4 tables suivantes :

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

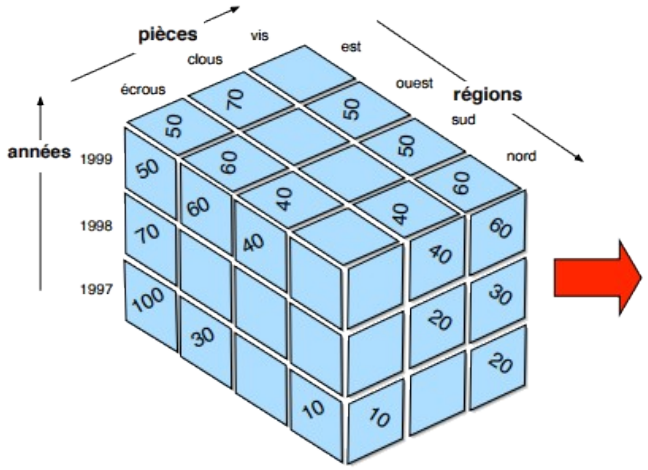
ventes ouest	1999	1998	1997
écrous		10	30
vis	50	50	50
clous		10	40

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

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

OPÉRATIONS RESTRUCTURATION

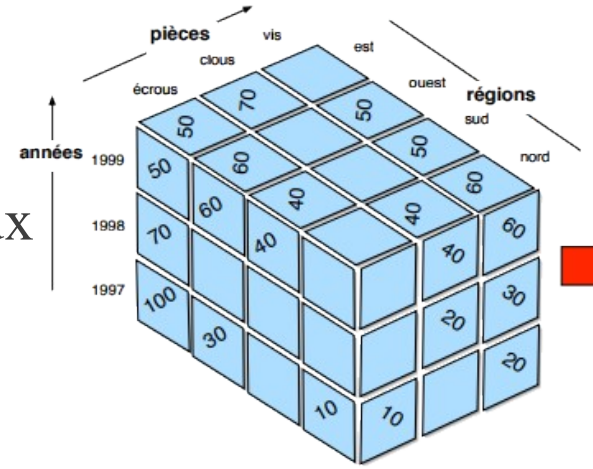
- Nest ou l'emboîtement :
  - imbrication des membres à partir du cube.
  - Permet de grouper sur une même représentation bi-dimensionnelle toutes les informations (mesures et membres) d'un cube quelque soit le nombre de ses dimensions.



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	

OPÉRATIONS RESTRUCTURATION

- Push ou l'enfoncement :
  - consiste à combiner les membres d'une dimension aux mesures du cube, i.e. de faire passer des membres comme contenu de cellules.

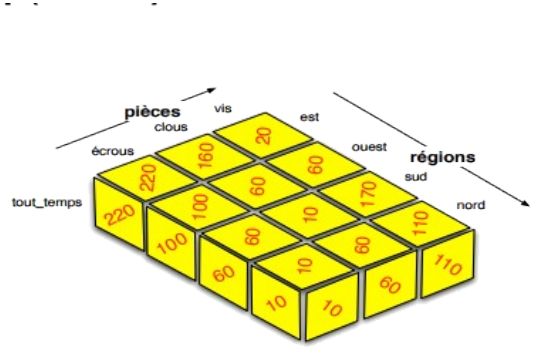


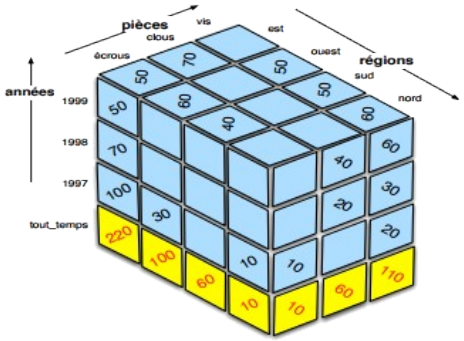
push(année) :

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

- Roll-up ou forage vers le haut: consiste à représenter les données du cube à un niveau de granularité supérieur conformément à la hiérarchie définie sur la dimension. Soit :



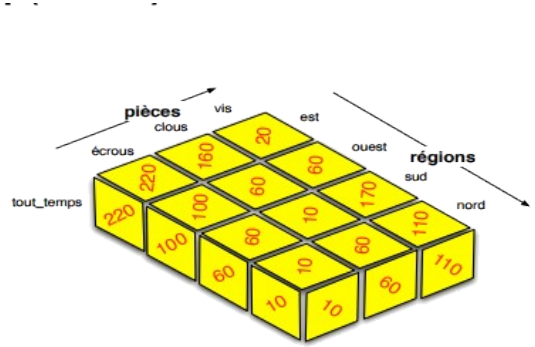


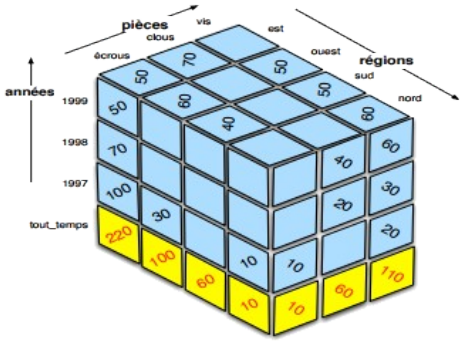
roll-up(annees, pieces) : la visualisation est souvent 2D :

nord	1999	1998	1997	tout_temps
vis	60	30	20	110
clous	40	20		60
écrous			10	10
tout_produit	100	50	30	180

OPÉRATIONS DE GRANULARITÉ

- Roll-up ou forage vers le haut: consiste à représenter les données du cube à un niveau de granularité supérieur conformément à la hiérarchie définie sur la dimension. Soit :





roll-up(annees, pieces) : la visualisation est souvent 2D :

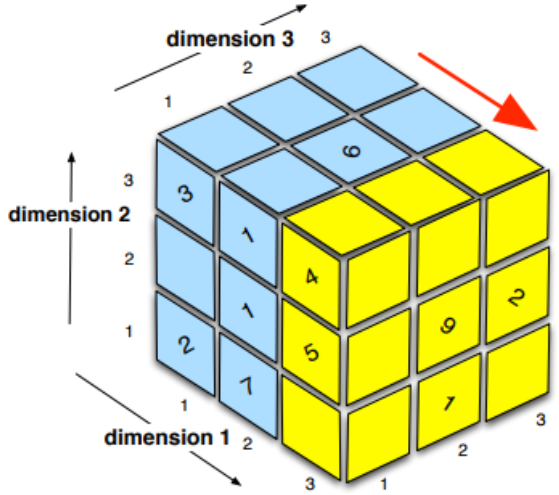
nord	1999	1998	1997	tout_temps
vis	60	30	20	110
clous	40	20		60
écrous			10	10
tout_produit	100	50	30	180

OPÉRATIONS DE GRANULARITÉ

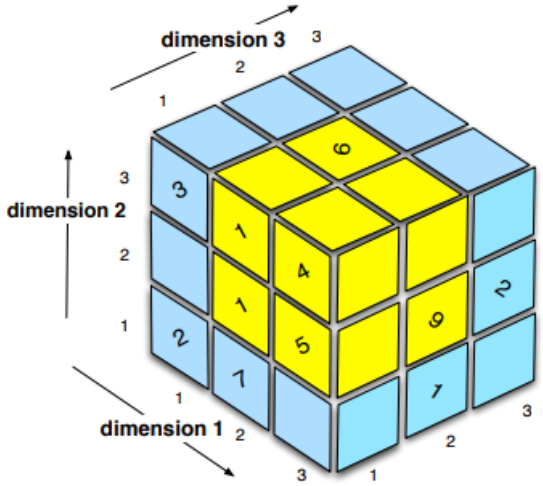
- Drill-down ou forage vers le bas : consiste à représenter les données du cube à un niveau de granularité de niveau inférieur, donc sous une forme plus détaillée.
  - opération réciproque de roll-up, drill-down permet d'obtenir des détails sur la signification d'un résultat en affinant une dimension ou en ajoutant une dimension

OPÉRATIONS ENSEMBLISTES

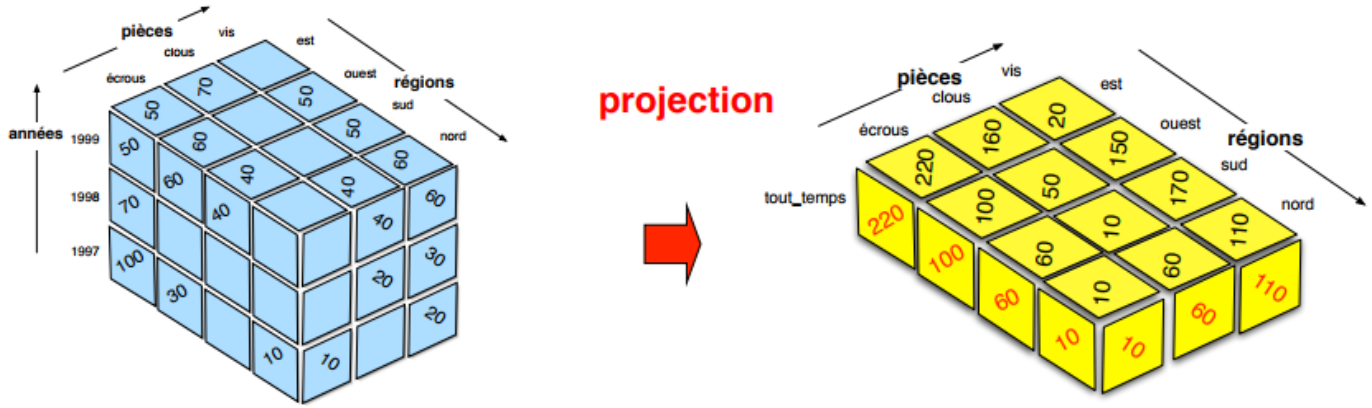
slide : correspond à une **projection** selon une dimension du cube :



dice : correspond à une **sélection** du cube :



OPÉRATIONS ENSEMBLISTES



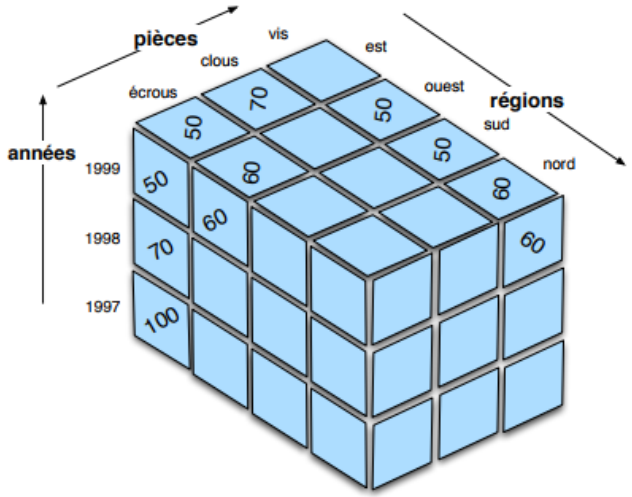
π piece, region :

ventes 97-99	est	ouest	sud	nord
écrous	220	100	60	10
clous	160	50	10	60
vis	20	150	170	110

OPÉRATIONS ENSEMBLISTES

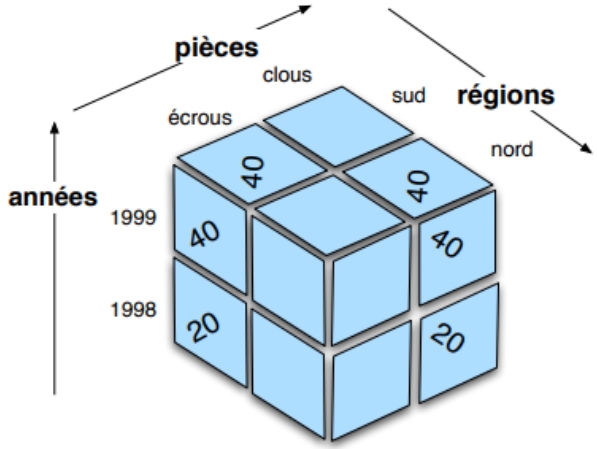
Selection 1

vente ≥ 50



Sélection 2

(regions = nord ou regions = sud) et
(pieces = clous ou pieces = ecrous) et
(annees = 1998 ou années = 1999)



INTRODUCTION: MDX

- MDX, acronyme de Multi Dimensional eXpression, est un langage de requêtes OLAP pour les bases de données multidimensionnelles
- inventé en 1997 par Mosha Pasumansky au sein de Microsoft, version commerciale Microsoft OLAP Services 7.0 & Analysis Services en 1998, dernière spécification OLE DB for OLAP (ODBO) en 1999
- MDX est fait pour naviguer dans les bases multidimensionnelles, et définir des requêtes sur tous leurs objets (dimensions, hiérarchies, niveaux, membres et cellules)
- Une requête MDX retourne un 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 VS SQL

- La syntaxe de MDX ressemble à celle de SQL par ses mots clé SELECT, FROM, WHERE, mais leurs sémantiques sont différentes :
  - SQL construit des vues relationnelles
  - MDX construits des vues multidimensionnelles des données
  - 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 VS 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 spécifie la source de données :
  - en SQL : une ou plusieurs tables
  - en MDX : un cube

MDX VS SQL

- La clause SELECT indique les résultats que l'on souhaite récupérer par la requête :
  - en SQL :
    - une vue des données en 2 dimensions : lignes (rows) et colonnes (columns)
    - les lignes ont la même structure définie par les colonnes
  - en MDX :
    - nombre quelconque de dimensions pour former les résultats de la requête
    - terme d'axe pour éviter confusion avec les dimensions du cube
    - pas de signification particulière pour les rows et les columns,
    - mais il faut définir chaque axe : axe1 définit l'axe horizontal et axe2 définit l'axe vertical

MDX EXEMPLE

Soit la requête MDX suivante : *(Q=Quarter)*

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

Agrégation de la mesure « SaleAmount » avec la fonction SUM function

Sélection de la dimension Time (2014 seulement)

Sélection de la dimension Product (all product)

Résultat :

	Q1 2014	April 2014	May 2014	June 2014
Paris	12,567	3,360	5,450	4,570
Berlin	12,567	3,360	5,450	4,570
…	…	…	SalesAmount.values	…

SYNTAXE DE BASE 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 :
    - Ex : ON COLUMNS, ON ROWS, ON PAGES, ON SECTIONS, ON CHAPTERS
  - sur un ou plusieurs niveaux de la hiérarchie :
    - Ex1 : {Paris, Berlin} de la dimension Lieu, niveau Ville
    - Ex2 : {[1er trimestre], [2nd trimestre].CHILDREN} de la dimension Temps, niveaux trimestre et mois.

SYNTAXE DE BASE MDX

- FROM - Spécification du/des cube/s de départ
  - Ensemble de cubes nécessaires à la création du cube résultat
  - Si plusieurs cubes nécessaires, cela implique une 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 »

REMARQUE : En MDX les mesures sont des éléments d'une dimension spéciale nommée « Measures » (ces mesures peuvent être utilisées aussi dans les clauses WHERE et FROM).

SYNTAXE DE BASE MDX

- Un membre = une instance d'un niveau d'une dimension,
- Est généralement spécifié entre crochets [...]
  - Ex : [Food], [Drink] = membres de la dimension "Products" de niveau 1
- Les membres = items 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]
- Les enfants d'un membre = membres du niveau immédiatement en dessous de celui-ci
- Ex. d'utilisations de membres 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]

SYNTAXE DE BASE MDX

- Un membre = une instance d'un niveau d'une dimension,
- Est généralement spécifié entre crochets [...]
  - Ex : [Food], [Drink] = membres de la dimension "Products" de niveau 1
- Les membres = items 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]
- Les enfants d'un membre = membres du niveau immédiatement en dessous de celui-ci
- Ex. d'utilisations de membres 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]

TUPLES, CELLULE ET MESURES

- Un tuple = suite de membres entre parenthèses séparés par une virgule:
  - Ex : ([Time].[2012] , [Product].[Food]) [on peut omettre les parenthèses si on a un tuple avec un seul membre.]
- Un tuple permet d'identifier une ou plusieurs cellules dans un cube situées à l'intersection de ses membres :
  - SELECT ([Time].[2012], [Product].[Food]) ON COLUMNS FROM [Sales]
  - SELECT ([Product].[All Products].[Food].[Baked Goods], [2012]) ON COLUMNS FROM [Sales]
- Dans un tuple, les mesures sont traitées comme une dimension particulière, nommée [Measures] :
  - SELECT ([Measures].[Unit Sales], [Product].[All Products].[Food].[Baked Goods]) ON COLUMNS FROM [Sales]

SETS (1)

- Un set = un ensemble ordonné de tuples définit sur une même dimension
- Un set commence par une accolade "{", dans laquelle sont énumérés les tuples séparés par des virgules, et se termine par une accolade appariée "}"

```
SELECT
{
 ([Measures].[Unit Sales], [Product].[All Products].[Food].[Baked Goods]),
 ([Measures].[Store Sales], [Product].[All Products].[Food].[Baked Goods])
}
ON COLUMNS
FROM [Sales]
```

ce set contient :

• 2 mesures différentes (Units sales et Store Sales) et

• le même membre (Baked Goods) :

SETS (2)

- Ex2 : un set qui comporte 2 mesures et 2 membres différents de la même dimension sur 2 niveaux différents ([Food] et [Baked Goods]) :
  - SELECT { ([Measures].[Unit Sales], [Product].[Food]), ([Measures].[Store Sales], [Product].[Food].[Baked Goods]) } ON COLUMNS FROM [Sales]
- Ex3 : un set qui a la même mesure et 2 membres contigus différents ([Food] et [Drink]) :
  - SELECT { ([Measures].[Unit Sales], [Product].[Food]), ([Measures].[Unit Sales], [Product].[Drink]) } ON COLUMNS FROM [Sales]
- Ex4 : un set qui ne contient qu'un seul membre ([2012]) :
  - SELECT { ([2012]) } ON COLUMNS ou { [2012] } ON COLUMNS FROM [Sales]

SPÉCIFICATION D'AXE EN MDX (1)

- Plusieurs spécifications possibles pour un même axe en MDX :
  - un set suivi du mot clef « ON » suivi d'un nom d'axe spécifique
  - fait référence à un numéro d'ordre s'il y a plus de 2 axes de restitution, ou simplement aux noms d'axes explicites « COLUMNS » et « ROWS »

Ex: *unités vendues "[Measures].[Unit Sales]" par an en 2012 et 2013 pour les produits "Drink" et "Food" :*

```
SELECT
{  ([Measures].[Unit Sales], [Product].[Food]),
     ([Measures].[Unit Sales], [Product].[Drink]) } ON AXIS(0),
{ ([Time].[2012]), ([Time].[2013]) } ON AXIS(1)
FROM [Sales]
```

ou

```
SELECT
   { ([Measures].[Unit Sales], [Product].[Food]),
      ([Measures].[Unit Sales], [Product].[Drink]) } ON COLUMNS,
   { ([Time].[2012]), ([Time].[2013]) } ON ROWS
FROM [Sales]
```

SPÉCIFICATION D'AXE EN MDX (2)

- Une façon simple est de définir un axe est de présenter sur l'axe tous les membres d'une dimension :
  - <dimension name>.MEMBERS
- Si l'on veut voir apparaître tous les membres de la dimension à un certain niveau de cette dimension :
  - <dimension name>.<level name>.MEMBERS
- par exemple la requête :
  - SELECT Years.MEMBERS ON COLUMNS, Régions.Continent.MEMBERS ON ROWS FROM Sales

SPÉCIFICATION D'AXE EN MDX (3)

- Cette requête MDX :
  - SELECT Years.MEMBERS ON COLUMNS, Régions.Continent.MEMBERS ON ROWS FROM Sales

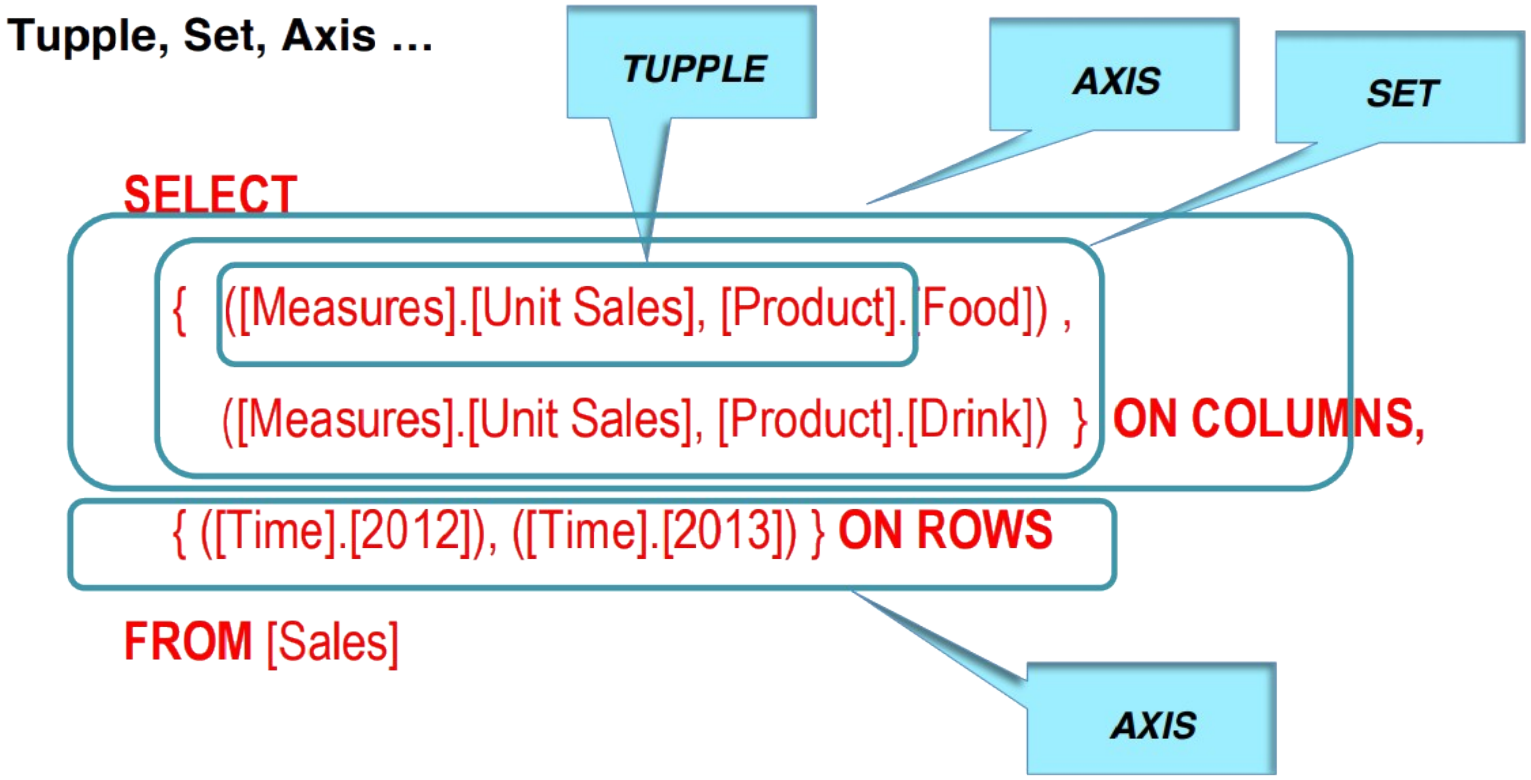
	2010	***2011***	***2013***	***2014***
N. America	120,000	200,000	400,000	600,000
S. America	-	10,000	30,000	70,000
Europe	55,000	95,000	160,000	310,000
Asia	30,000	80,000	220,000	200,000

  - l'axe horizontal est (i.e. COLUMNS) contient tous les membres de la dimension 'Years' (ici 2010, 2011, 2013, 2014)
  - l'axe vertical est (i.e. ROWS) contient tous les membres du niveau 'Continent' de la dimension 'Regions' (ici N & S America, Europe, Asia)

SPÉCIFICATION D'AXE EN ÉNUMÉRATION DANS MDX

- Certaines dimensions ou niveaux ont plus de 1000 membres !
- On peut souhaiter ne pas considérer tous les membres de la dimension ou du niveau, aussi dans MDX on peut spécifier une liste de membres à considérer : { dim.member1, dim.member2, … , dim.membern}
- Ex : On considère seulement les ventes sur les 2 années 2012 et 2013:
  - SELECT { Years.[2012], Years.[2013] } ON COLUMNS, Regions.Continent.MEMBERS ON ROWS FROM Sales

RÉCAPITULATIF



SPÉCIFICATION DE FILTRES (SLICERS) DANS MDX (1)

- Dans la requête précédente on considère que 2 dimensions : Regions et Years
- Supposons qu'on s'intéresse non plus aux ventes de tous les produits, mais seulement aux ventes d'ordinateurs, on définie alors le nouvel axe :
  - { Products.[Product Group].Computers }
- On pourrait ajouter cet axe à notre requête, mais elle contiendrait alors 3 axes, et tous les outils OLAP ne pourraient le visualiser, aussi on préfère utiliser une opération de Slice (filtre) :
- Dans MDX l'opération de Slice est traitée par une clause WHERE :
  - SELECT { Years.[2012], Years.[2013] } ON COLUMNS, Regions.Continent.MEMBERS ON ROWS FROM Sales WHERE (Products.[Product Group].[Computers])

SPÉCIFICATION DE FILTRES (SLICERS) DANS MDX (2)

- La clause WHERE a la syntaxe :
  - WHERE (member-of-dim1, member-of-dim2, …, member-of-dimn)
- Dans un Slice en MDX on peut avoir plusieurs membres, mais ils doivent appartenir à des dimensions différentes (pas de Slice sur 2 produits différents, par ex. computer et printers)
- Ex: Slice sur un produit (computer) et un client particulier (AT&T) : Ventes de computer à AT&T pour les années 2012 et 2013 pour tous les continents.
  - SELECT { Years.[2012], Years.[2013] } ON COLUMNS, Regions.Continent.MEMBERS ON ROWS FROM Sales WHERE (Products.[Product Group].[Computers], Customers.[AT&T])

SPÉCIFICATION DE FILTRES (SLICERS) DANS MDX (3)

- On peut souhaiter voir plutôt que les ventes (sales, mesure par défaut) de computer, le nombre d'unité expédiées
- On doit juste ajouter la mesure « units » dans le Slice :
- Ex : Unités de computer expédiées à AT&T pour les années 2012 et 2013 pour tous les continents.
  - SELECT { Years.[2012], Years.[2013] } ON COLUMNS, Regions.Continent.MEMBERS ON ROWS FROM Sales WHERE (Products.[Product Group].[Computers], Customers. [AT&T], Measures.[Units])

INSERTION DE COMMENTAIRE

- Les commandes MDX peuvent être commentées de trois façons différentes :
  - // Commentaire en fin de ligne
  - -- Commentaire en fin de ligne
  - /* Commentaire sur plusieurs lignes */
- Les commentaires peuvent être imbriqués comme le montre l'exemple cidessous :
  - /* Commentaire sur plusieurs lignes /* Commentaire imbriqué */ */

EMBOITEMENT (NEST) DE TUPLES DANS MDX

- Les axes peuvent contenir des membres ou des tuples
- Ex : On veut voir les ventes de computers et printers en Europe et en Asie sur les années 2012 et 2013.
- On peut le faire avec une requête MDX avec 3 axes :
  - SELECT { Continent.[Europe], Continent.[Asia] } ON AXIS(0), { Product.[Computers], Product.[Printers] } ON AXIS(1), { Years.[2012], Years.[2013] } ON AXIS(2) FROM Sales
- Le résultat de cette requête n'est pas une table à 2 dimensions, mais un cube à 3 dimensions, plus difficile à interpréter

EMBOITEMENT (NEST) DE TUPLES DANS MDX

- Les lignes de cette table contiennent 2 dimensions : Computer et Printers
- La dimension Product est emboité (nested) dans la dimension Regions conduisant aux 4 combinaisons des membres de dimensions {Europe, Asie} et {Computers, Printers}
- Chaque combinaison de membres de dimensions différentes est un Tuple
- Ex : Ventes de computers et printers pour l'Europe et l'Asie pour 2012 et 2013:
  - SELECT { Year.[2012], Year.[2013] } ON COLUMNS, { (Continent.Europe, Product.Computers), (Continent.Europe, Product.Printers), (Continent.Asia, Product.Computers) (Continent.Asia, Product.Printers) } ON ROWS FROM Sales

MEMBRES CALCULÉS DANS MDX (1)

- MDX permet d'étendre le cube en définissant d'autres membres de dimensions, membres qui sont calculés à partir de membres existants
- La syntaxe pour les membres calculés est de mettre la construction suivante WITH en face de l'instruction SELECT :
  - WITH MEMBER parent.name AS 'expression'
- Ex : Trouver les profits des produits au cours des années
  - WITH MEMBER Measures.Profit AS 'Measures.Sales – Measures.Cost' SELECT Products.MEMBERS ON COLUMNS, Year.MEMBERS ON ROWS FROM Sales WHERE (Measures.Profit)

MEMBRES CALCULÉS DANS MDX (2)

- Les membres calculés sont traités de la même manière que des membres ordinaires, ils peuvent donc être utilisés :
  - dans la définition d'axes
  - dans une clause WHERE
- On peut définir des membres calculés avec d'autres membres calculés
- Ex : membre calculé ProfitPercent (profit pourcentage du coût) : ProfitPercent = Profit / Cost
  - WITH MEMBER Measures.Profit AS 'Measures.Sales – Measures.Cost' MEMBER Measures.ProfitPercent AS 'Measures.Profit / Measures.Cost', FORMAT_STRING = '#.#%' SELECT { Measures.Profit, Measures.ProfitPercent } ON COLUMNS FROM Sales

MEMBRES CALCULÉS DANS MDX (2)

- Les membres calculés sont traités de la même manière que des membres ordinaires, ils peuvent donc être utilisés :
  - dans la définition d'axes
  - dans une clause WHERE
- On peut définir des membres calculés avec d'autres membres calculés
- Ex : membre calculé ProfitPercent (profit pourcentage du coût) : ProfitPercent = Profit / Cost
  - WITH MEMBER Measures.Profit AS 'Measures.Sales – Measures.Cost' MEMBER Measures.ProfitPercent AS 'Measures.Profit / Measures.Cost', FORMAT_STRING = '#.#%' SELECT { Measures.Profit, Measures.ProfitPercent } ON COLUMNS FROM Sales

MEMBRES CALCULÉS DANS MDX (3)

- Ex : Supposons que l'on veuille comparer la compagnie entre 2012 et 2013, on peut :
  - construire une requête qui a un axe {[2012], [2013]} et regarder les paires de nombres pour chaque mesure
  - définir un membre calculé dans le niveau Year, parallèle à 2012 et 2013, qui contiendra la différence entre eux :

```
WITH MEMBER Time.[12 to 13] AS 'Time.[2012] – Time.[2013]' SELECT
{ Time.[12 to 13] } ON COLUMNS, Measures.MEMBERS ON ROWS FROM
Sales
```

MEMBRES CALCULÉS DANS MDX (4)

- Supposons qu'on veuille voir comment les profits ont évolués entre 2012 et 2013:
  - WITH MEMBER Measures.Profit AS 'Measures.Sales – Measures.Cost' MEMBER Time.[12 to 13] AS 'Time.[2012] – Time.[2013]' SELECT { Measures.Sales, Measures.Cost, Measures.Profit } ON COLUMNS, { Time.[2012], Time.[2013], Time.[12 to 13] } ON ROWS FROM Sales

	Sales	***Cost***	***Profit***
2012	300	220	80
2013	350	210	140
12 to 13	50	-10	60

MEMBRES NULL (1)

- WITH MEMBER Measures.[Sales Growth] AS '(Sales) – (Sales, Time.PrevMember)' SELECT { [Sales], [Sales Growth] } ON COLUMNS, Month.MEMBERS ON ROWS FROM Sales
- Calcule pour chaque mois la croissante des ventes comparée au mois précédent.
- Pour le premier mois, il n'y a pas de mois précédent dans le cube : au lieu de retourner une erreur, MDX a la notion de membre NULL représentant des membres qui n'existent pas

MEMBRES NULL (2)

- La sémantique d'un membre NULL est :
  - Quand la cellule contient un de ses coordonnés un membre NULL, la valeur numérique de la cellule sera zero ainsi : (Sales, [January].PrevMember) = 0
  - Toute fonction de membre appliquée à un membre NUL retourne NULL
  - Quand un membre NULL member est inclus dans un set, il est juste ignoré, ainsi : {[September], [January].PrevMember} = {[September]}

FONCTIONS DE NAVIGATION SUR LES MEMEBRES DES DIMENSIONS (1)

- On a besoin de fonctions pour naviguer dans les hiérarchies de dimensions
- Par exemple écrire [WA].Parent (Etat du Washington) est équivalent à écrire [USA]
  - Si l'on veut connaître les coordonnées d'une cellule dans le sous-espace de la requête : <dimension name>.CurrentMember, <level name>.CurrentMember
- Ex : on veut calculer le pourcentage des ventes dans chaque région relative son état, on utilisera pour trouver l'état d'une ville donnée la fonction Parent :
  - WITH MEMBER Measures.PercentageSales AS '(Regions.CurrentMember, Sales) / (Regions.CurrentMember.Parent, Sales)',FORMAT_STRING = '#.00%' SELECT { Sales, PercentageSales } ON COLUMNS, Regions.Cities.MEMBERS ON ROWS FROM Sales

FONCTIONS DE NAVIGATION SUR LES MEMEBRES DES DIMENSIONS (2)

Fonction	Signification	Remarque
Parent	Donne le parent du membre de dimension considéré	Déplacement vertical sur une hiérarchie, et on change ainsi de niveau
FirstChild	Donne le premier fils du membre considéré	
LastChild	Donne le dernier fils du membre considéré	
FirstSibling	…	Déplacement horizontal à l'intérieur d'un même niveau
LastSibling	…	
NextMember	Donne le membre suivant du membre considéré	
PrevMember	Donne le membre précédent fils du membre considéré	

FONCTIONS DE NAVIGATION SUR LES MEMEBRES DES DIMENSIONS (3)

- Ex 1 : On veut définir un nouveau membre calculé Sales Growth montrant la croissance ou le déclin des ventes comparées avec le précédent mois, trimestre ou année
  - WITH MEMBER Measures.[Sales Growth] AS '(Sales) – (Sales, Time.PrevMember)'
- On peut alors utiliser ce nouveau membre dans la requête :
  - SELECT { [Sales], [Sales Growth] } ON COLUMNS, Month.MEMBERS ON ROWS FROM Sales
- Ex 2 : On veut observer une chute des ventes pour différents produits et comment ces ventes ont augmenté pour chaque produit dans le dernier mois :
  - SELECT { [Sales], [Sales Growth] } ON COLUMNS, Product.MEMBERS ON ROWS FROM Sales

OPÉRATIONS SUR LES SETS DANS MDX: INTERSECT & UNION

- Ces fonctions ont en entrée et en sortie des sets de membres ou de tuples
  - Intersection : INTERSECT(set1, set2) (Rarement utilisée)
  - Union : UNION(set1, set2)
- Ex: on veut voir sur les axes résultats : l'Europe, les USA, tous les états des USA, toutes les villes dans l'état WA, et pour l'Asie (scénario de Drilldown typique)
  - On défini alors alors un set : { [Europe], [USA], [USA].Children, [WA].Children, [Asia] }
  - L'équivalent avec l'opérateur UNION sera : UNION({ [Europe], [USA] }, UNION([USA.Children], UNION([WA].Children, { [Asia] })))
  - Certaines implémentations de MDX dispose de l'opérateur + (plus) pour faire l'union de 2 sets : set1 + set2 + … + setn

FONCTIONS SUR LES SETS

Fonction	Syntaxe	Description
Head	Head(<< Set >> [,<< Numeric Expression >>])	Eléments de tête d'un set
Tail	Tail(<< Set >> [,<< Numeric Expression >>])	Derniers éléments d'un set
Subset	Subset(<< Set >>, << Start >> [,<< Count >>])	Sous-ensemble d'éléments d'un set
TopCount	TopCount(<< Set >>, << Count >> [,<< Numeric Expression >>])	Les premiers éléments ayant la plus grande valeur de la mesure
Order	Order (<< Set <<, {<<String Expression>> | <<Numeric Expression >>} [, ASC | DESC | BASC | BDESC])	Tri des éléments d'un set
Filter	Filter(<< Set >>, << conditions >>)	Les éléments d'un set qui satisfont le filtre

FONCTIONS SUR LES SETS: TOPCOUNT, TAIL, HEAD, FILTER

- Exemple d'utilisation des fonctions Topcount, Tail et Head :
  - SELECT { ([Measures].[Unit Sales]) } ON COLUMNS, { (Head([Time].Children, 2)), (Tail([Time].Children, 3)), (Topcount([Time].Children,1,[Measures].[Unit Sales])) } ON ROWS FROM [Sales]
- La requête suivante, avec la fonction Filter n'affiche que les cellules dont la mesure "[Unit Sales]" est supérieur ou égale (≥) à une certaine valeur :
  - SELECT { ([Measures].[Unit Sales]) } ON COLUMNS, { Filter([Time].Children, ([Measures].[Unit Sales]) > 70000) } ON ROWS FROM [Sales]

FONCTIONS SUR LES SETS: CROSSJOIN (1)

- La fonction CrossJoin permet de croiser des « sets » et ainsi réaliser des tableaux croisés
- Etant donné un set A et un set B, CrossJoin construit un nouveau set contenant tous les tuples (a,b) où a est dans A et b dans B :
- Ex 1 : SELECT { CrossJoin ({([Time].[2012].[Q1]), ([Time].[2012].[Q2])}, {([Measures].[Unit Sales]), ([Measures].[Store Sales])}) } ON COLUMNS, { ([Product].[Drink].Children) } ON ROWS FROM [Sales]

FONCTIONS SUR LES SETS: CROSSJOIN (2)

- Ex 2 : SELECT { CrossJoin ({([Time].[2012].[Q1]), ([Time].[2012].[Q2])}, {([Measures].[Unit Sales]), ([Measures].[Store Sales])}) } ON COLUMNS, { ([Product].[Drink].Children), ([Product].[Food].[Baked Goods]) } ON ROWS FROM [Sales]
- Ex 3 : SELECT Crossjoin ({[Measures].[Unit Sales], [Measures].[Store Sales]}, {[Time].[2012].[Q1]}) ON COLUMNS, Crossjoin ({[Promotion Media].[All Media].[Cash Register Handout], [Promotion Media].[All Media].[Sunday Paper, Radio, TV], [Promotion Media].[All Media].[TV]}, {[Gender].Children}) ON ROWS FROM [Sales]

FONCTIONS SUR LES SETS: CROSSJOIN (3)

- Dans l'exemple précédent il y a des cellules vides, dû au fait qu'aucun croisement n'est possible sur le 2 axes
- Pour n'afficher que les cellules pleines, utilisez l'opérateur Non Empty CrossJoin :
  - SELECT Crossjoin ({[Measures].[Unit Sales], [Measures].[Store Sales]}, {[Time].[2012].[Q1]}) ON COLUMNS, Non Empty Crossjoin ({[Promotion Media].[All Media].[Cash Register Handout], [Promotion Media].[All Media].[Sunday Paper, Radio, TV], [Promotion Media].[All Media].[TV]}, {[Gender].Children}) ON ROWS FROM [Sales]
- Remarque : on ne peut pas mettre plus d'une fois la même hiérarchie dans plusieurs axes indépendants d'une requête.

EXPRESSION CONDITIONNELLES: IIF (1)

- Elles permettent de réaliser des tests conditionnels à l'aide du mot clé IIF.
  - [Measures].[Q-1] : donne [Unit Sales] au trimestre précédent
  - [Measures].[Evolution] : donne l'évolution de [Unit Sales] entre deux trimestres consécutifs
  - [Measures].[%] : donne l'évolution en pourcentage
  - [Measures].[Performance] : renvoie une chaîne de caractères qui nous permet d'appliquer des règles de gestion liées à des conditions

EXPRESSION CONDITIONNELLES: IIF (2)

```
WITH
   MEMBER [Measures].[Q-1] AS
  ( ParallelPeriod([Time].[2012].[Q1],1, [Time].CurrentMember), [Measures].[Unit Sales] )
   MEMBER [Measures].[Evolution]  AS
   ( ([Time].CurrentMember, [Measures].[Unit Sales]) - (ParallelPeriod([Time].[2012].[Q1],1,
    [Time].CurrentMember), [Measures].[Unit Sales]) ), solve_order = 1
   MEMBER [Measures].[%] AS
   ( [Measures].[Evolution] /  (ParallelPeriod([Time].[2012].[Q1],1,[Time].CurrentMember),
     [Measures].[Unit Sales]) ), format_string = "Percent", solve_order = 0
   MEMBER [Measures].[Performance]  AS
IIF([Measures].[Q-1]<>0,
   IIF([Measures].[%] < 0, "-",
      IIF([Measures].[%] > 0 and [Measures].[%] < 0.01, "+",
        IIF([Measures].[%] > 0.01 and  [Measures].[%] < 0.05, "++", "better performance") )),"null")
SELECT
  { ([Measures].[Unit Sales]), ([Measures].[Q-1]),
   ([Measures].[Evolution]),   ([Measures].[%]), ([Measures].[Performance]) } ON COLUMNS,
   Descendants([Time], 1) ON ROWS
FROM [Sales]
```