[MAC0313] Introdução aos Sistemas de Bancos de Dados Aula 27 *Data Warehouses* e Bancos de Dados Multidimensionais

Kelly Rosa Braghetto

DCC–IME–USP

17 de novembro de 2016

Perspectiva histórica e motivação

- Anos 1970: criação do **modelo relacional e adoção da linguagem SQL** como padrão para os SGBDs
  - **Facilitou a manipulação, manutenção e recuperação dos dados**
  - Popularizou uso de BDs em organizações do mundo todo
- Os SGBDs Relacionais são amplamente usados para documentar operações diárias, ou seja, **transações rotineiras**
- Essas transações geralmente fazem pequenas alterações nos dados
- Os SGBDs Relacionais precisam lidar com um **grande número de transações** desse tipo de forma eficiente
- Os SGBDs Relacionais têm sido extensivamente otimizados para lidar com as aplicações de **processamento de transações online** (OLTP) $\Rightarrow$ **bancos de dados transacionais**

Perspectiva histórica e motivação

- Com o aumento da competitividade entre empresas e da busca por melhorias nos processos de produção, intensificou-se a necessidade de se obter **visões analíticas e gerenciais** sobre os dados armazenados nos BDs
- **Problema:** os BDs transacionais sozinhos não se mostravam adequados para prover visões de apoio à tomada de decisões
- **Consequência:** desenvolvimento de **aplicações de apoio à decisão** que analisam e exploram dados atuais e históricos, identificando tendências e criando resumos
- Fabricantes de SGBDs Relacionais já vêm adicionando a seus produtos funcionalidades especiais de apoio à tomada de decisão
- A SQL vem sendo estendida para suportar consultas complexas

Apoio à tomada de decisão

Problemas do uso de sistemas SQL tradicionais para a realização de consultas de apoio à tomada de decisão:

- Cláusula WHERE de consultas com muitos ANDs e ORs $\Rightarrow$ processamento ineficiente
- Aplicações exigem o uso de funções estatísticas não suportadas pelo padrão SQL
- Muitas consultas envolvem condições ao longo do tempo ou exigem agregação ao longo do tempo $\Rightarrow$ padrão SQL oferece suporte deficiente para isso

Apoio à tomada de decisão

- A tomada de decisão organizacional exige uma visão mais abrangente de todos os aspectos de uma empresa
- Essa visão geralmente é construída a partir de dados extraídos de vários BDs mantidos por diferentes unidades empresariais, com informações históricas e de resumo $\Rightarrow$ ***data warehouse***

Data Warehouses

- Um ***Data Warehouse*** **(DW)**, assim como um sistema de banco de dados, **pode ser compreendido como uma coleção de dados + um sistema de apoio**
- DWs se diferem bastante dos BDs tradicionais – em sua estrutura, funcionalidade, desempenho e propósito
- O uso inicial do termo *Data Warehouse* é creditado a **William (Bill) Inmon (1992)**

Definição de DW, segundo Inmon:

"Um *Data Warehouse* é uma coleção de dados orientada a assunto, integrada, não volátil, variável no tempo, que dá apoio às decisões de gerência"

Data Warehouses – outras definições

- A definição de Inmon restringe o DW a uma "coleção de dados"
- Entretanto, textos mais recentes empregam o termo com um significado mais abragente
- DW – "sistema projetado com o propósito de dar apoio à extração, processamento e apresentação eficiente (dos dados) para fins analíticos e de tomada de decisão"

Data Warehouses

Características importantes

- Geralmente integram quantidades muito grandes de dados, provenientes de múltiplas fontes [heterogêneas]
- São otimizados para a recuperação de dados (e não para o processamento rotineiro de transações, como os BDs "tradicionais")
- Levam em consideração o armazenamento, a manutenção e a recuperação eficiente de dados históricos
  - DWs tipicamente dão apoio a análises de séries temporais e tendências, ambas as quais requerem mais dados históricos do que costuma-se manter em BDs transacionais

Data Warehouses

Mais características importantes

- A informação em um DW muda com pouca frequência (= não-volátil), e pode ser considerada como não sendo de tempo real e com atualização periódica.
  - A política de atualização, que geralmente é incremental, define também a periodicidade das cargas de dados
- Apoiam diferentes tipos de aplicações de análise. Exemplos:
  - OLAP (*online analytical processing*)
  - Mineração de dados

OLAP (*online analytical processing*)

- Aplicações OLAP geralmente suportam uma classe de consultas estilizadas que normalmente envolvem:
  - Operadores de agrupamento e agregação
  - Excelente suporte para condições booleanas complexas
  - Funções estatísticas e recursos para a análise de séries temporais
- Alguns SGBDs Relacionais são também projetados para suportar eficientemente (além das consultas SQL tradicionais) consultas OLAP

Mineração de Dados

- Técnicas de mineração de dados são técnicas para análise exploratória e descoberta de conhecimento em grandes volumes de dados
- Os algoritmos de mineração buscam relações de similaridade ou discordância entre dados, com o objetivo de encontrar padrões, irregularidades e regras
- Envolvem estatística, probabilidade, inteligência artificial, etc.

Modelo multidimensional de dados

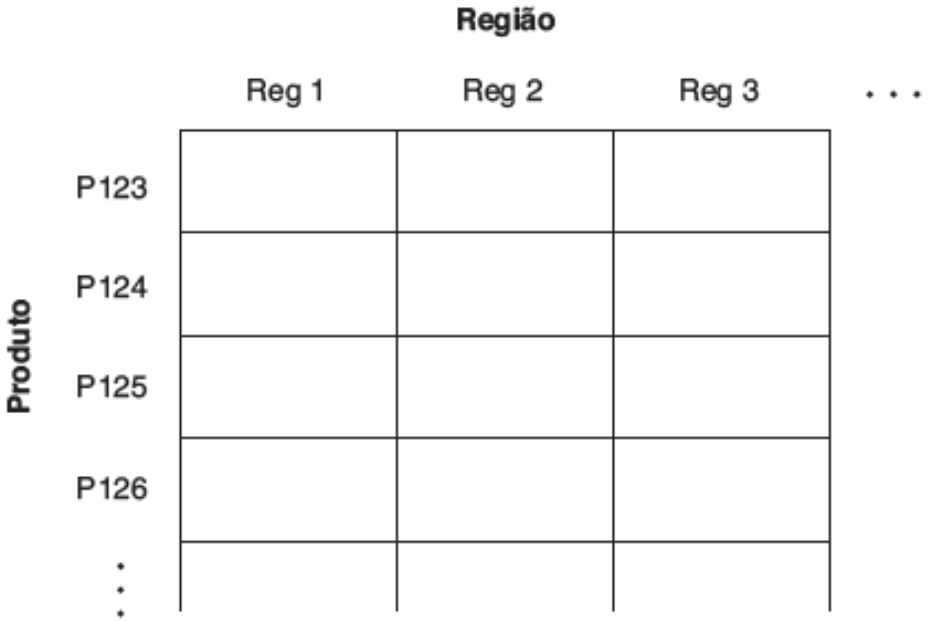
- Bancos dados de transacionais geralmente se baseiam no modelo de dados relacional
- *Data Warehouses* se baseiam no **modelo de dados (multi)dimensional**
- O modelo multidimensional de dados representa os indicadores importantes para uma área de negócio, que são chamados de **medidas ou fatos**, e os seu parâmetros, chamados de **dimensões**

Modelo multidimensional de dados

- **Fatos** $\Rightarrow$ variáveis observadas; medidas de interesse
- **Dimensões** $\Rightarrow$ diferentes perspectivas sob as quais os fatos podem ser vistos
- Os modelos multidimensionais tiram proveito das relações inerentes aos dados para gerar dados em **matrizes multidimensionais** chamadas de **cubos de dados** (ou **hipercubos**, quando há mais de 3 dimensões)
- O desempenho de consultas realizadas sobre dados representados em matrizes multidimensionais pode ser bem melhor que no modelo de dados relacional

Exemplo de modelo bidimensional – dados de vendas de produtos por regiões

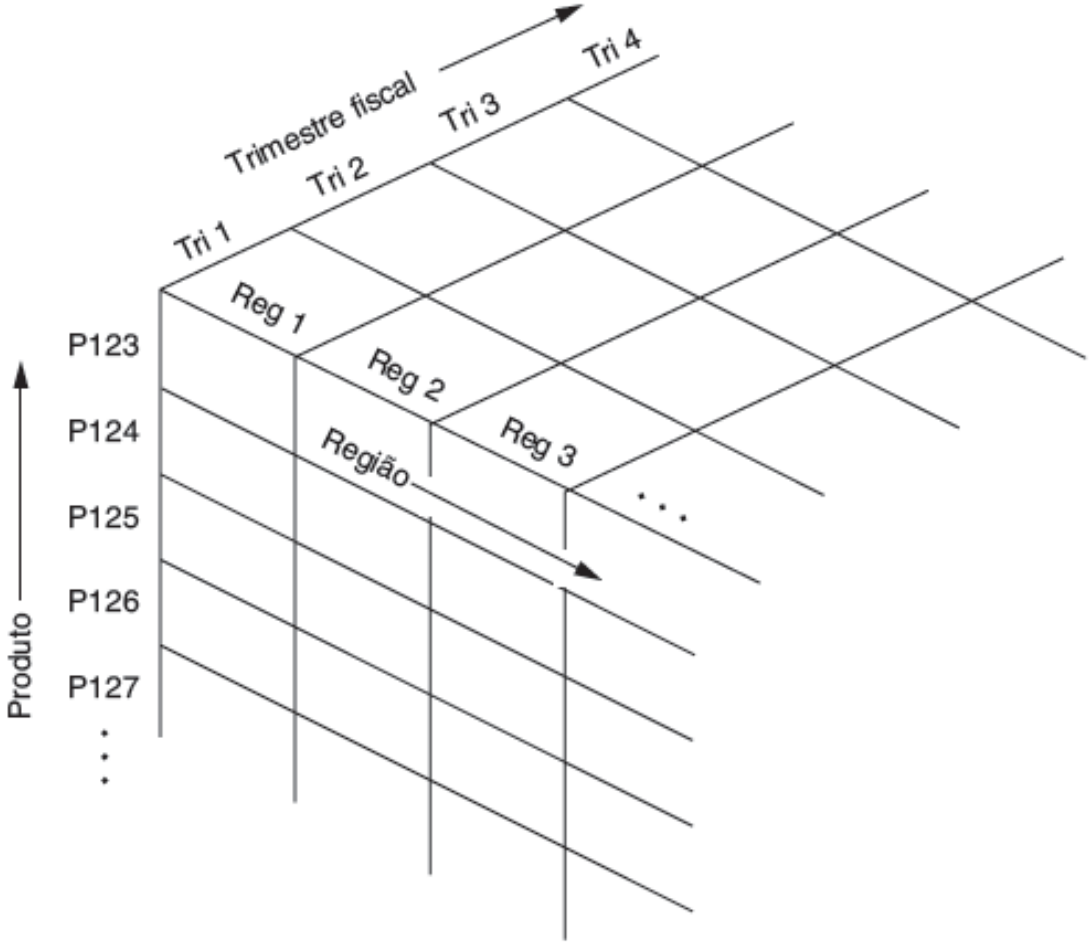
Dimensões: Produto e Região



Cada célula contém fatos (dados de interesse analítico) para um produto específico e uma região específica.

Exemplo de cubo – dados de vendas de produtos por trimestres fiscais e regiões de vendas

Dimensões: Produto, Região e Trimestre fiscal (tempo)



Cada célula contém fatos (dados de interesse analítico) para um produto específico, um trimestre fiscal específico e uma região específica.

Modelo multidimensional

Cubos

- Na estrutura dos cubos, os dados podem ser consultados diretamente em qualquer combinação das dimensões (evitando consultas complexas ao BD)
- Existem ferramentas que permitem a visualização dos dados de acordo com a escolha de dimensões do usuário.

MOLAP × ROLAP

- Alguns sistemas OLAP realmente armazenam os dados fisicamente em matrizes multidimensionais; esses sistemas são chamados de **OLAP multidimensionais (MOLAP – multidimensional OLAP)**
- Entretanto, os sistemas OLAP podem armazenar os dados multidimensionais também como tabelas (relações); nesse caso, eles são chamados **OLAP relacionais (ROLAP – relational OLAP)**.

Modelo de armazenamento multidimensional

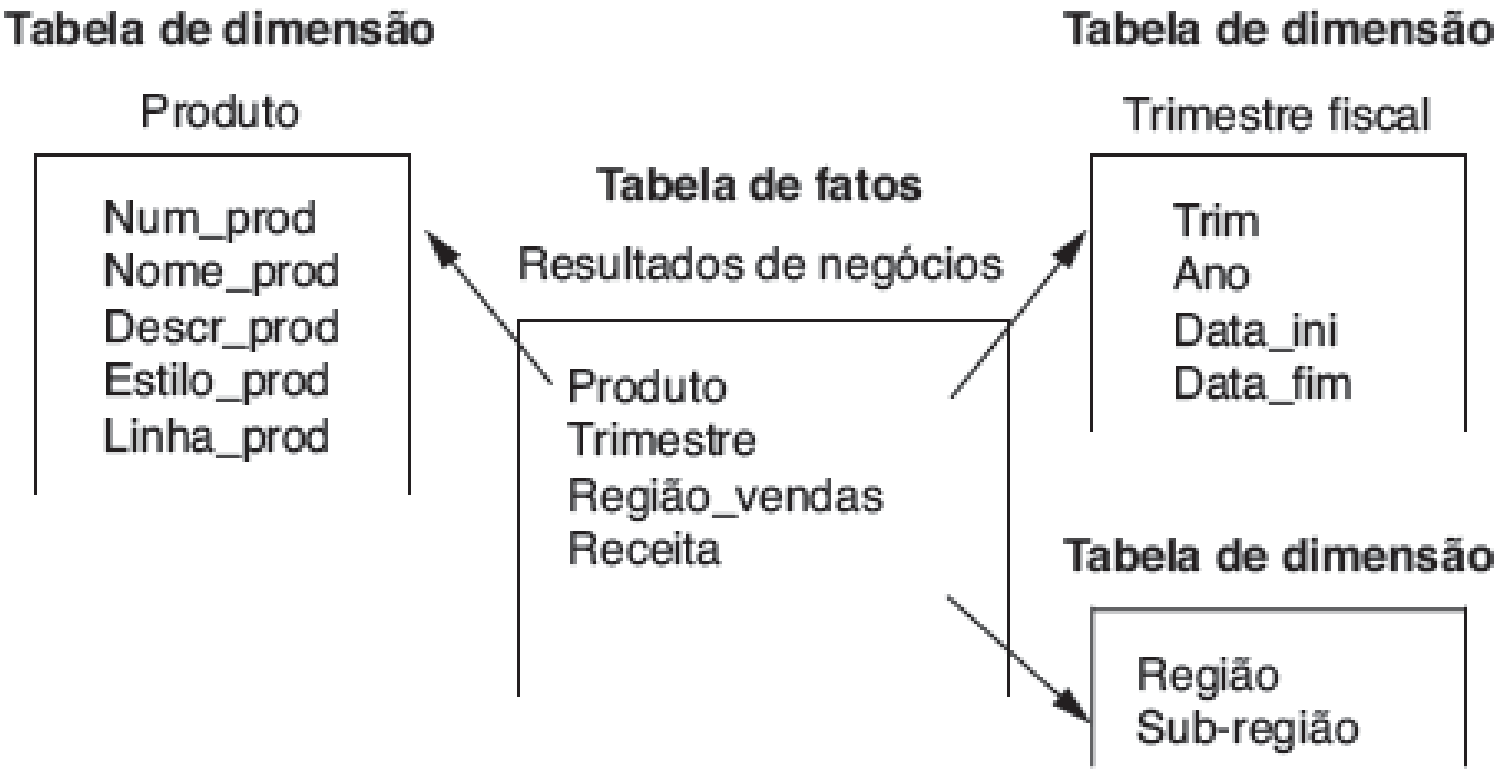
Envolve dois tipos de tabelas:

- **Tabela de dimensão** – consiste em tuplas com atributos da dimensão
- **Tabela de fatos** – possui uma tupla para cada fato registrado. Uma tabela fato contém alguma(s) variável(is) medida(s) ou observada(s) e a(s) identifica com ponteiros para a tabelas de dimensões.

A tabela de fatos contém os dados, e as tabelas de dimensões identificam cada tupla naqueles dados.

Tabelas de fato e de dimensões

Exemplo – um esquema estrela



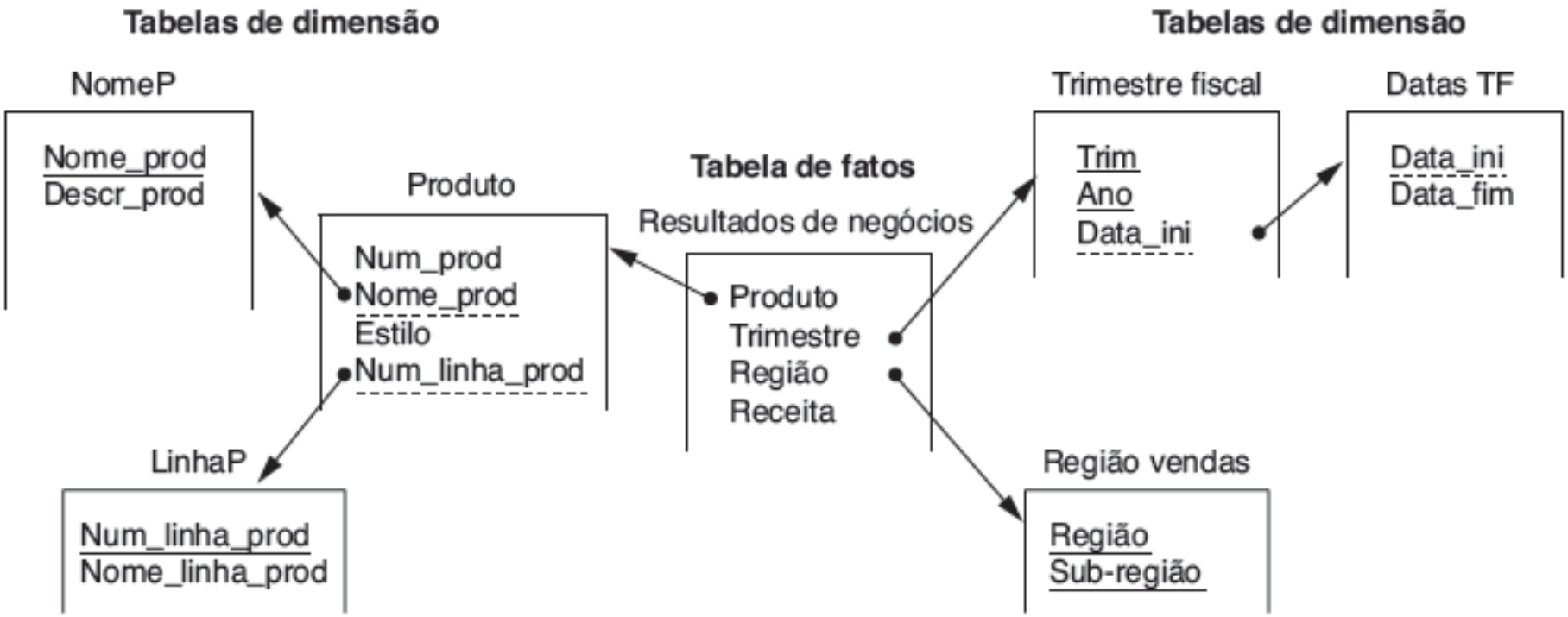
Esquemas de armazenamento

Dois esquemas bastante utilizados para representar dados no modelo multidimensional são:

- **Esquema estrela** – bastante fácil de ser entendido por uma pessoa da área de negócios, uma vez que "as coisas a serem avaliadas" estão na parte central do diagrama, enquanto que "as formas de se olhar para elas" estão nos quadros em volta. Obs.: O exemplo no slide anterior é um esquema estrela.
- **Esquema floco de neve** – é uma variação do esquema estrela, na qual as tabelas de dimensões de um esquema estrela são organizadas em hierarquias.

Esquemas de armazenamento

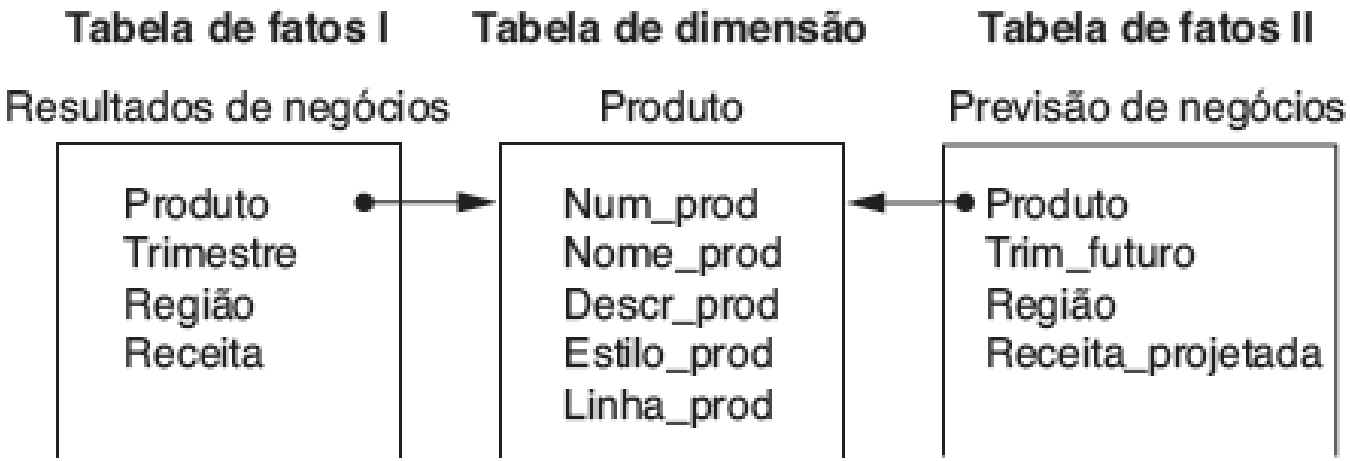
Exemplo de esquema floco de neve



Esquemas de armazenamento

⇒ Uma **constelação de fatos** é um conjunto de tabelas de fatos que compartilham algumas tabelas de dimensão.

Exemplo de constelação de fatos



Esquemas de armazenamento

Esquema estrela

- É muito comum em BDs projetados para OLAP
- Tipicamente, a maior parte dos dados está na tabela de fatos
- A tabela de fatos não tem redundância
- Para minimizar o tamanho da tabela de fatos, os identificadores de dimensão são gerados pelo sistema (= chaves artificiais)

Esquemas de armazenamento

Esquema estrela (cont.)

- Normalmente, tabelas de dimensão **não são normalizadas** (= contêm redundâncias). Motivos:
  - tabelas de dimensões em BD para OLAP são estáticas (portanto, as anomalias de alteração, inserção e remoção não são preocupantes)
  - comparado ao tamanho da tabela de fatos, o espaço economizado pela normalização das dimensões é desprezível
  - evitar a divisão da dimensão em tabelas menores diminui o tempo gasto em junções de tabelas na execução de consultas

Operações de consulta multidimensionais

- As operações de consulta e manipulação suportadas pelo modelo multidimensional são fortemente influenciadas pelas ferramentas de usuário final (como as planilhas eletrônicas)
- **Objetivo**: fornecer aos usuários finais (não especialistas) uma interface intuitiva e poderosa para tarefas de análise comuns orientadas a negócio
- As ferramentas OLAP oferecem um conjunto de operações para a (des)agregação, seleção e projeção de dados organizados em um modelo multidimensional

Consultas de agregação multidimensionais

- Operação bastante comum: **agregar uma medida sobre uma ou mais dimensões**
- Exemplo:
  - Encontrar o total de vendas por trimestre
  - Encontrar o total de vendas por produto e região
- Essas consultas podem ser expressas como consultas SQL sobre as tabelas de fatos e dimensões

```
SELECT trimestre, SUM(receita)
FROM VENDAS
GROUP BY trimestre;
```

```
SELECT produto, regiao, SUM(receita)
FROM VENDAS
GROUP BY produto, regiao;
```

Consultas de agregação multidimensionais

- Outro uso da agregação: **fazer um resumo em diferentes níveis de uma hierarquia de dimensões**
- Em OLAP, essa operação é chamada de ***roll-up***
- Exemplo de *roll-up*: se recebemos dados de vendas semanais, podemos agregar na dimensão Tempo para obtermos as vendas por mês, ano, etc.
- A operação inversa é chamada de ***drill-down***, que f**ornece uma visão de granularidade mais fina, desagregando elementos**
- Exemplo de *drill-down*: dado o total de vendas por categoria de produtos em cada trimestre, podemos solicitar uma apresentação mais detalhada, apresentando dados de vendas para cada produto em cada semana.

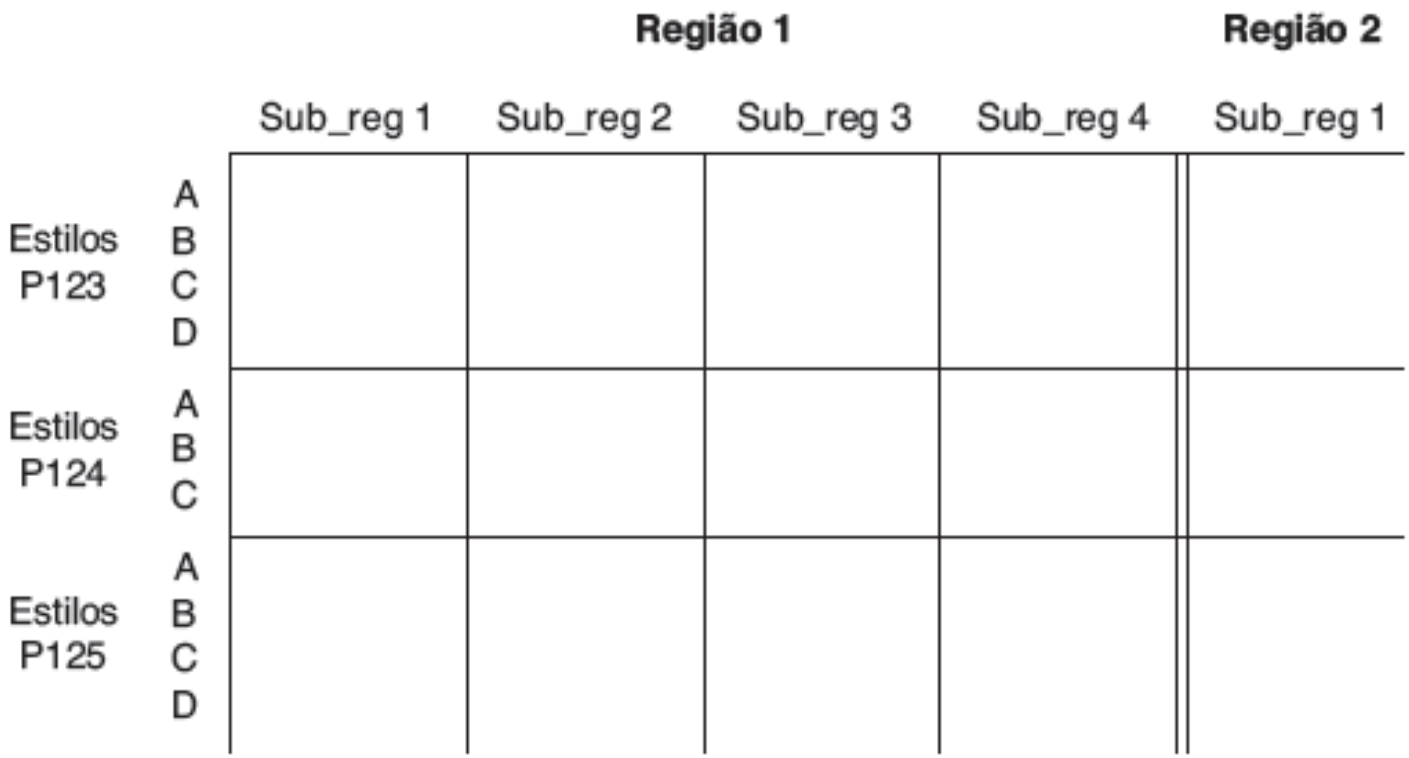
Consultas de agregação multidimensionais

Exemplo de *roll-up* na dimensão Produto

Operação *roll-up*, que passa de produtos individuais para uma granularidade mais espessa, baseada em categorias de produtos.

Consultas de agregação multidimensionais

Exemplo de *drill-down* nas dimensões Região e Produto

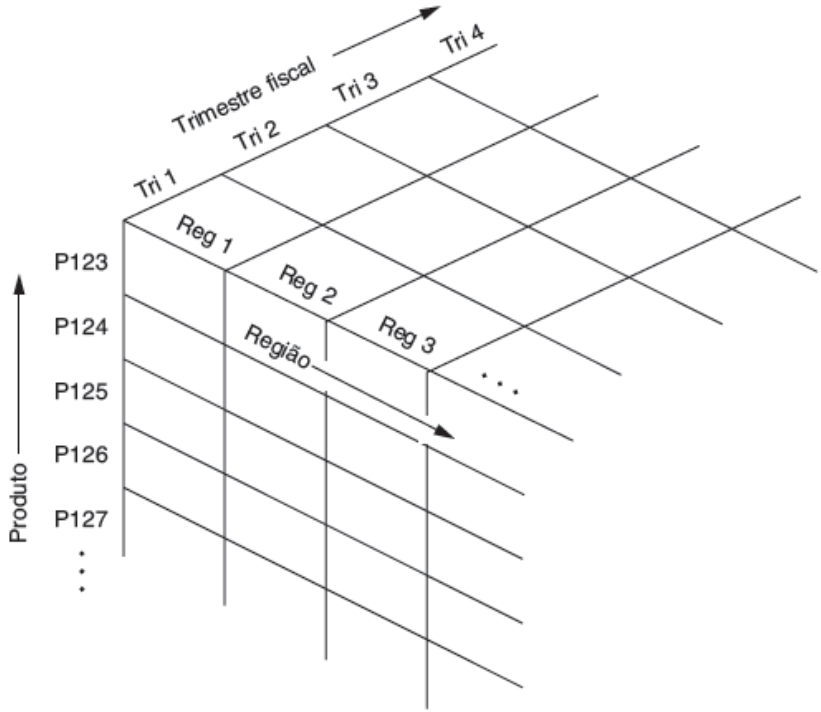


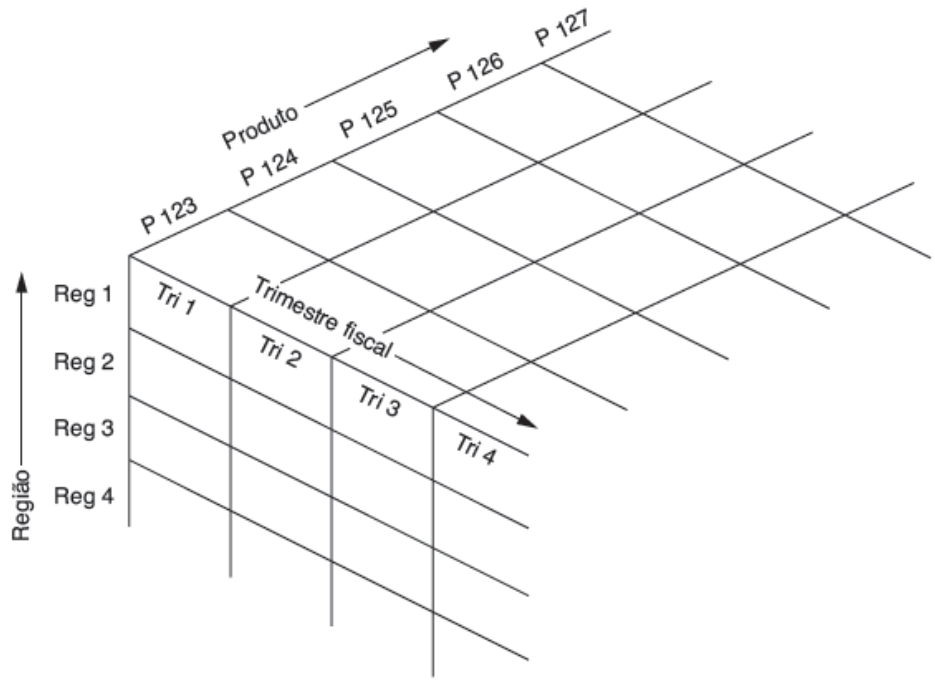
Operação *drill-down*, desagregando vendas nacionais em vendas por região e depois vendas regionais em sub-regiões, e separando as categorias de produtos em estilos.

Consultas de agregação multidimensionais

Operação de *rotação* (ou *giro*, ou *pivoting*)

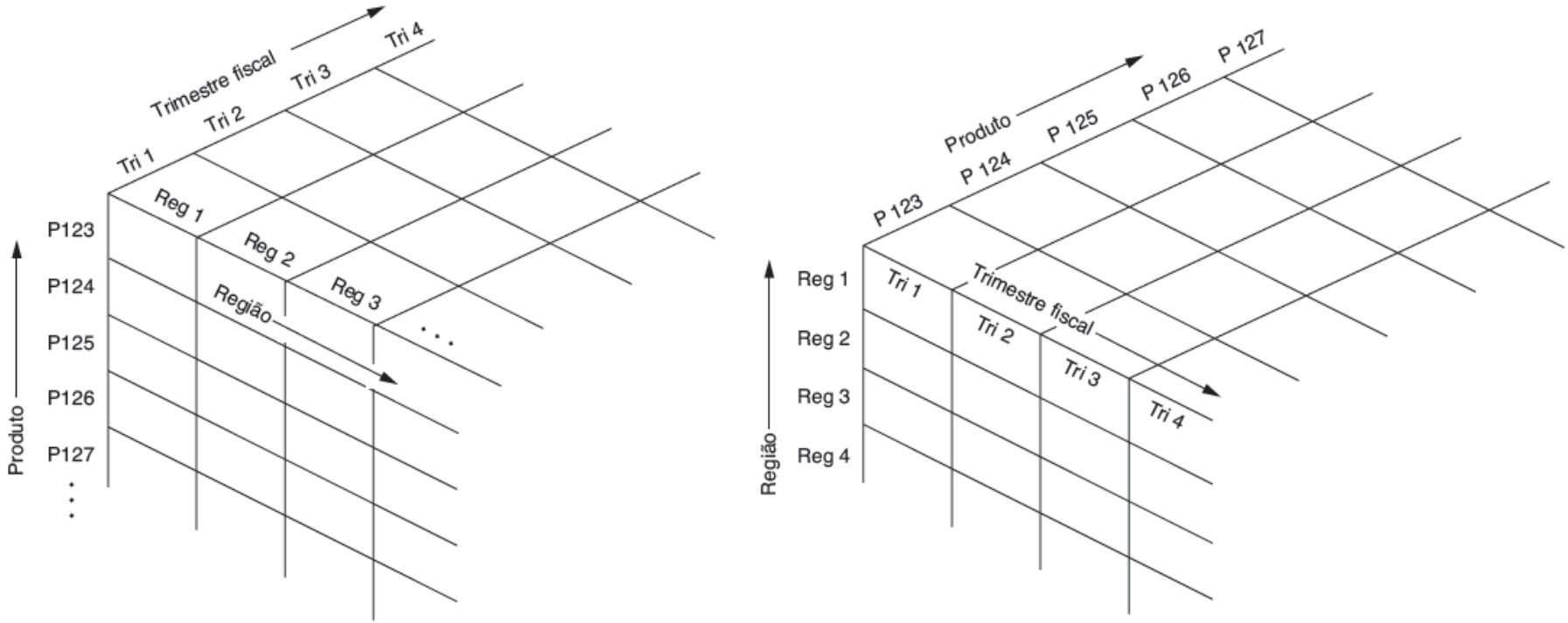
- Nessa operação, o cubo pode ser imaginado girando para mostrar uma orientação diferente dos eixos. Exemplo:





Consultas de agregação multidimensionais

Operação de rotação – Exemplo



O cubo à direita é resultado de uma rotação no à esquerda. Depois da rotação, vemos os dados como se tivéssemos uma tabela de vendas regionais separadamente para cada produto, com as vendas trimestrais para o produto, região por região.

Outras consultas comuns aplicáveis sobre cubos

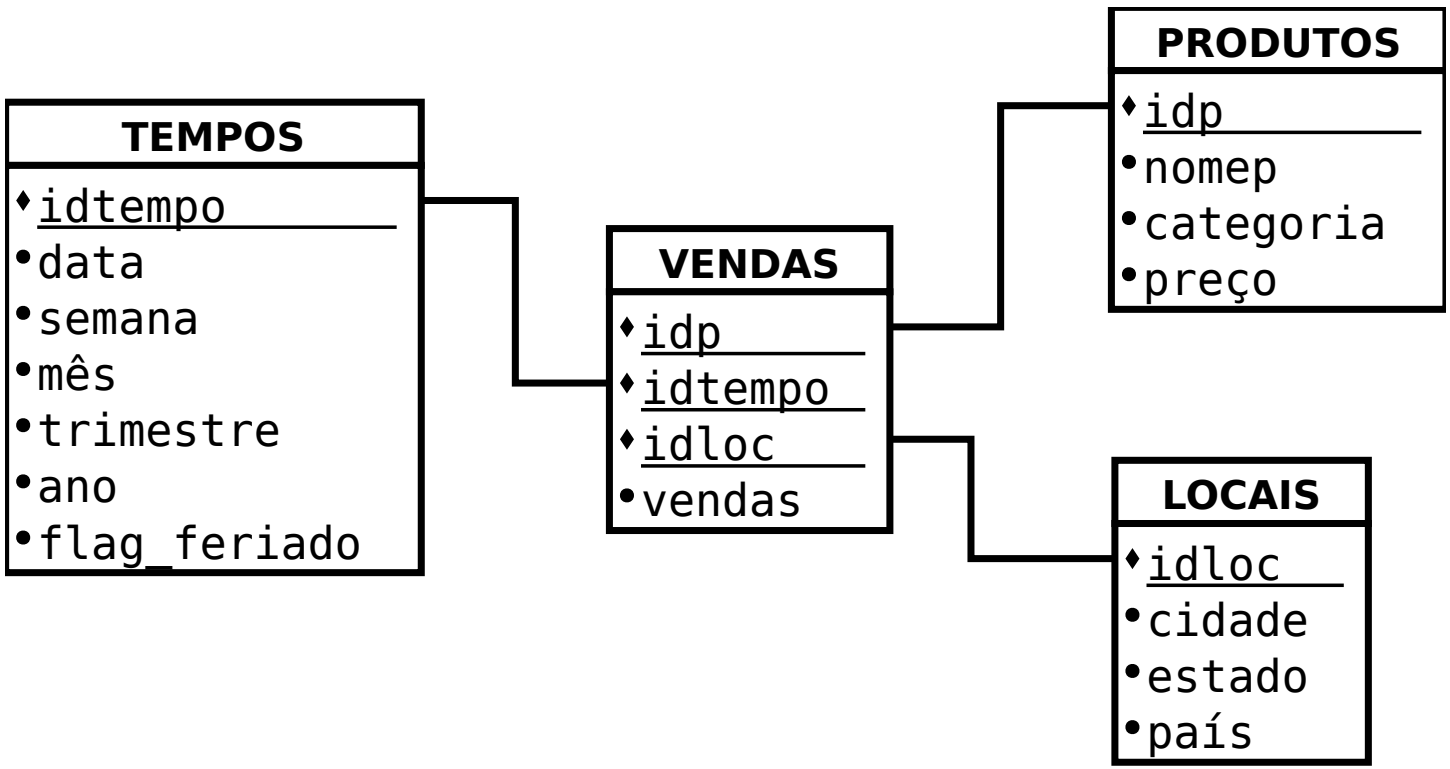
Operação de fatiar (slice)

- **Fatiar** um conjunto de dados significa fazer uma seleção por igualdade em uma ou mais dimensões, possivelmente com algumas dimensões removidas

Operação de cortar (dice)

- **Cortar** um conjunto de dados significa fazer uma seleção por intervalo

Exemplo: DW de Vendas (esquema estrela)



Exemplo: DW de Vendas (esquema estrela)

Tabulação cruzada

- Tabulação cruzada = resultado de uma operação de rotação no cubo
- **Exemplo**: uma rotação nas dimensões de Local e Tempo gera uma tabela de total de vendas de cada local para cada valor de tempo

	SP	**RJ**	**Total**
1995	63	81	**144**
1996	38	107	**145**
1997	75	35	**110**
Total	**176**	**223**	**399**

- Nessa apresentação tabular dos dados, além do total de vendas, também temos resumos adicionais das vendas por ano e das vendas por estado

Correspondente SQL de operações OLAP

Uma única operação OLAP leva a várias consultas SQL relacionadas, com agregações e agrupamentos.

- **Exemplo:** para produzir as informações da tabulação cruzada mostrada no slide anterior, precisamos de 4 consultas SQL

1) Para gerar os valores do "corpo" da tabela:

```
SELECT    T.ano, L.estado, SUM(V.vendas)
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
GROUP BY  T.ano, L.estado
```

Correspondente SQL de operações OLAP

Uma única operação OLAP leva a várias consultas SQL relacionadas, com agregações e agrupamentos.

- **Exemplo:** para produzir as informações da tabulação cruzada mostrada no slide anterior, precisamos de 4 consultas SQL

2) A coluna de resumo à direita pode ser gerada com:

```
SELECT    T.ano, SUM(V.vendas)
FROM      Vendas V, Tempos T
WHERE     V.idtempo = T.idtempo
GROUP BY  T.ano
```

Correspondente SQL de operações OLAP

Uma única operação OLAP leva a várias consultas SQL relacionadas, com agregações e agrupamentos.

- **Exemplo:** para produzir as informações da tabulação cruzada mostrada no slide anterior, precisamos de 4 consultas SQL

3) A linha de resumo na parte inferior da tabela pode ser gerada com:

```
SELECT    L.estado, SUM(V.vendas)
FROM      Vendas V, Locais L
WHERE     V.idloc = L.idloc
GROUP BY  L.estado
```

Correspondente SQL de operações OLAP

Uma única operação OLAP leva a várias consultas SQL relacionadas, com agregações e agrupamentos.

- **Exemplo:** para produzir as informações da tabulação cruzada mostrada no slide anterior, precisamos de 4 consultas SQL

4) A soma acumulada, mostrada no canto inferior direito da tabela pode ser gerado com:

```
SELECT    SUM(V.vendas)
FROM      Vendas V
```

Tabulação cruzada ou *roll-ups*?

- O exemplo da tabulação cruzada pode ser considerado um *roll-up*:
  - no conjunto de dados inteiro (= tratar tudo como um único grupo grande)
  - na dimensão Local
  - na dimensão Tempo
  - nas dimensões Local e Tempo juntas
- Cada roll-up corresponde a uma única consulta SQL com agrupamento

Consultas de agrupamento/agregação

- Dado um fato com k dimensões associadas, podemos fazer um *roll-up* em qualquer subconjunto dessas k dimensões
- temos 2^k consultas SQL de agrupamento possíveis
- Por meio de operações de alto-nível (como a de rotação), usuários podem gerar muitas dessas consultas
- Reconhecer características comuns entre essas consultas possibilita um cálculo mais eficiente

Padrão SQL:1999 – GROUP BY CUBE

- O padrão SQL:1999 oferece um suporte especial a consultas *roll-up* e de tabulação cruzada
- Cláusula `GROUP BY CUBE` – equivale a uma coleção de instruções GROUP BY, com um GROUP BY para cada subconjunto possível das k dimensões listadas na cláusula.
- **Exemplo**:

```
SELECT    T.ano, L.estado, SUM(V.vendas)
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
GROUP BY CUBE (T.ano, L.estado)
```

$\Rightarrow$ Dá o total de vendas geral, o total de vendas por ano, o total de vendas por estado e o total de vendas por ano e estado

Padrão SQL:1999 – GROUP BY CUBE

- **Exemplo**:

```
SELECT    T.ano, L.estado, SUM(V.vendas)
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
GROUP BY CUBE (T.ano, L.estado)
```

- **Resultado**:

T.ano	T.estado	SUM(V.vendas)
1995	WI	63
1995	CA	81
1995	*nulo*	114
1996	WI	38
1996	CA	107
1996	*nulo*	145
1997	WI	75
1997	CA	35
1997	*nulo*	110
nulo	WI	176
nulo	CA	223
nulo	*nulo*	399

Padrão SQL:1999 – GROUP BY ROLLUP

- Cláusula `GROUP BY ROLLUP` – calcula subconjuntos da tabulação cruzada calculada pelo GROUP BY CUBE
- **Exemplo**:

```
SELECT    T.ano, L.estado, SUM(V.vendas)
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
GROUP BY ROLLUP (T.ano, L.estado)
```

$\Rightarrow$ Dá o total de vendas geral, o total de vendas por ano e o total de vendas por ano e estado, mas não o total de vendas por estado.

Padrão SQL:1999 – GROUP BY ROLLUP

- **Exemplo**:

```
SELECT    T.ano, L.estado, SUM(V.vendas)
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
GROUP BY ROLLUP (T.ano, L.estado)
```

- **Resultado**:

T.ano	T.estado	SUM(V.vendas)
1995	WI	63
1995	CA	81
1995	*nulo*	114
1996	WI	38
1996	CA	107
1996	*nulo*	145
1997	WI	75
1997	CA	35
1997	*nulo*	110
~~*nulo*~~	~~WI~~	~~176~~
~~*nulo*~~	~~CA~~	~~223~~
nulo	*nulo*	399

Padrão SQL:1999 – Ainda sobre o GROUP BY CUBE

- **Exemplo**:

```
SELECT     SUM(V.vendas)
FROM       Vendas V
GROUP BY CUBE (idp,idloc,idtempo)
```

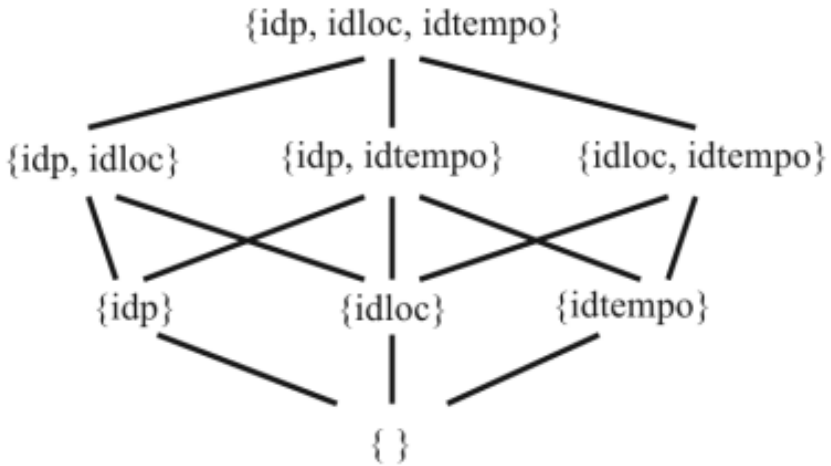
  Essa consulta aplica *roll-up* à tabela de Vendas em todos os oito subconjuntos do conjunto {idp, idloc, idtempo} (incluindo o conjunto vazio).

- Equivale a oito consultas da forma:

```
SELECT     SUM(V.vendas)
FROM       Vendas V
GROUP BY   <lista de agrupamento>
```

Padrão SQL:1999 – Ainda sobre o GROUP BY CUBE

- Pode-se considerar que as oito consultas estão organizadas em uma treliça:



- As tuplas resultantes em um nó podem ser agregadas ainda mais, para calcular o resultado de qualquer nó filho
- Os relacionamentos entre consultas em uma operação GROUP BY CUBE é explorado para se obter uma avaliação eficiente

Dimensão tempo

- A dimensão tempo é muito importante no apoio à decisão
- De forma geral, consultas que envolvem análise de tendências são difíceis de se expressar em SQL
- **Janela de consulta** – extensão introduzida no SQL:1999 para lidar com isso

Exemplos de consultas muito difíceis ou impossíveis na SQL sem janela de consulta

1. Encontrar a média móvel de vendas dos n dias anteriores. (Para cada dia, deve-se calcular as vendas médias diárias sobre os n dias anteriores.)
2. Encontrar os cinco produtos mais vendidos, classificados pelas vendas acumuladas, para cada mês no ano anterior.
3. Classificar todos os produtos pelo total de vendas no ano anterior e, para cada produto, imprimir a diferença no total de vendas relativa ao produto classificado atrás dele.

Padrão SQL:1999 – cláusula WINDOW

- Cláusula WINDOW – identifica uma "janela" ordenada de linhas "em torno" de cada tupla em uma tabela
- Permite aplicar funções de agregação na janela de uma linha e assim estender a linha com os resultados
- **Exemplo**: podemos associar as vendas médias dos últimos 3 dias a cada tupla de Vendas (cada tupla em Vendas registra o total de vendas de um dia)
  - Isso fornece a média móvel das vendas de 3 dias

Padrão SQL:1999 – cláusula WINDOW

Exemplo

```
SELECT    L.estado, T.mês, AVG(V.vendas) OVER W AS medmov
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
WINDOW    W AS (PARTITION BY L.estado
           ORDER BY T.mês
           RANGE BETWEEN INTERVAL '1' MONTH PRECEDING
           AND INTERVAL '1' MONTH FOLLOWING)
```

- A definição de uma janela inclui:
  1. definição de uma partição (ideia semelhante à criação de grupos com GROUP BY)
  2. definição da ordenação das linhas dentro da partição
  3. definição dos limites da janela associada à cada linha (em termos da ordem das linhas dentro da partição)

Padrão SQL:1999 – cláusula WINDOW

Exemplo

```
SELECT    L.estado, T.mês, AVG(V.vendas) OVER W AS medmov
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
WINDOW   W AS (PARTITION BY L.estado
          ORDER BY T.mês
          RANGE BETWEEN INTERVAL '1' MONTH PRECEDING
          AND INTERVAL '1' MONTH FOLLOWING)
```

- **Interpretação da consulta:**
  Considere uma linha em Vendas para uma loja em SP. A janela dessa linha inclui todas as linhas que descrevem as vendas em SP dentro do mês anterior ou posterior e *medmov* é a média das vendas (sobre todos os produtos) em SP dentro desse período.

Enquadramento de uma janela

Há 2 maneiras de enquadrar uma janela em SQL:1999

1. Uso da construção RANGE – define uma janela com base nos valores de uma coluna, que deve ter tipo numérico, ou data-hora ou intervalo (ou seja, tipos para os quais as operações de adição e subtração estão definidas)
2. Uso direto da ordenação e da especificação de quantas linhas estão na janela. Exemplo:

```
SELECT    L.estado, T.mês, AVG(V.vendas) OVER W AS medmov
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
WINDOW   W AS (PARTITION BY L.estado
          ORDER BY T.mês
          ROWS BETWEEN 1 PRECEDING 1 FOLLOWING)
```

Enquadramento de uma janela

```
SELECT    L.estado, T.mês, AVG(V.vendas) OVER W AS medmov
FROM      Vendas V, Tempos T, Locais L
WHERE     V.idtempo = T.idtempo AND V.idloc = L.idloc
WINDOW    W AS (PARTITION BY L.estado
           ORDER BY T.mês
           ROWS BETWEEN 1 PRECEDING 1 FOLLOWING)
```

- Se houver exatamente uma tupla para cada mês no resultado gerado pelas cláusulas FROM-WHERE, então essa consulta terá resultado igual à anterior
- Mas se um determinado mês não tiver alguma tupla ou tiver várias linhas, as duas consultas produzirão resultados diferentes

No PostgreSQL

Não há:

- GROUP BY CUBE
- GROUB BY ROLLUP

Há (mas não como está na SQL:1999)

- Suporte a consultas com janelas
  `https://www.postgresql.org/docs/9.5/static/tutorial-window.html`

Esquemas de armazenamento

Materialização de tabelas de resumo

- Artifício usado para acelerar o tempo de resposta em consultas interativas em aplicações OLAP
- Consultas ad-hoc podem ser respondidas usando tabelas originais junto com resumos calculados previamente
- **Problema de projeto**: decidir quais tabelas de resumo devem ser materializadas para se obter o melhor uso da memória disponível e consultas interativas mais eficientes
- Essa decisão pode ser a mais importante no projeto de um DW

Síntese da comparação entre OLTP e *Data Warehouse*

BDs Transacional (OLTP)	***Data Warehouse***
Orientado a transações	Orientado a processos de negócio
Milhares de usuários	Poucos usuários (geralmente, $<$ 100)
Geralmente pequeno (vários GB até poucos TB)	Grande (vários TB)
Dados atuais	Dados históricos
Dados normalizados (muitas tabelas, poucas colunas por tabela)	Dados não normalizados (poucas tabelas, muitas colunas por tabela)
Atualizações contínuas	Atualizações por lote
Consultas de simples a complexas	Normalmente, consultas muito complexas

Ferramentas para DWs

- Geradores de relatórios (Crystal Reports, Pentaho Report Designer, JFreeReport, ...)
- Ferramentas OLAP (Mondrian/Jpivot, PALO, ...)
- Planilhas de cálculos (Microsoft Excel, ...)
- Geradores de mapas (Google Earth, Mapinfo, ...)
- Ferramentas de mineração de dados (Pentaho Weka, SAS Enterprise Miner, ...)
- etc...
- *Business Intelligence products suite* – **SpagoBI**, **Pentaho**, Microsoft SQL Server Analysis Server, Oracle Business Intelligence Suite Enterprise Edition, ...

Referências Bibliográficas

- ***Sistemas de Gerenciamento de Banco de Dados*** (3ª edição), Ramakrishnan e Gehrke.
  Capítulo 25
- ***Sistemas de Bancos de Dados*** (6ª edição), Elmasri e Navathe.
  Capítulo 29
- *Projeto e Modelagem de Bancos de Dados*, Teorey, Lighstone, Nadeau.
  Capítulo 8
- *Building the DataWarehouse* (3ª edição), William H. Inmon.
- *The data warehouse toolkit* (1996), Ralph Kimball.