

# Datawarehouse

Filipe Fidalgo

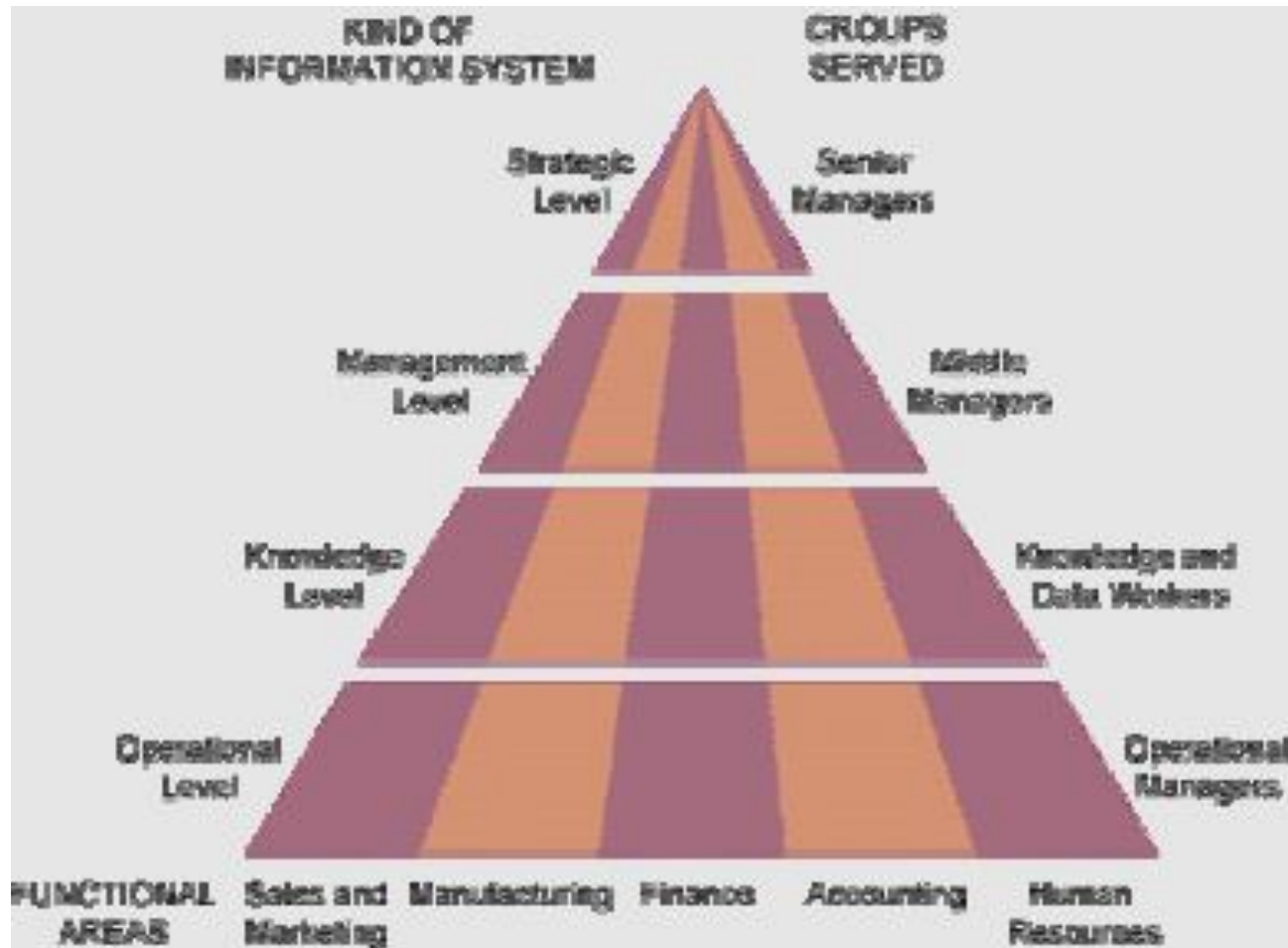
[ffidalgo@ipcb.pt](mailto:ffidalgo@ipcb.pt)

# Sumário

---

- Datawarehouse
- Modelação Multidimensional
- Ferramentas de análise dos dados em DW

# Sistemas de Informação



**Níveis dos Sistemas de Informação.**  
(Fonte: Laudon e Laudon, 2004)

# Sistemas de Informação

## Níveis e tipos de Sistemas de Informação (Adaptado de Laudon e Laudon, 2004)

Nível de SI	Tipo de SI	Definição	Utilizadores
Estratégico	Sistema de Suporte a Executivos (ESS)	SI a nível estratégico da organização desenhado para endereçar a tomada de decisão não estruturada	Gestor Sénior
Gestão	Sistemas de Informação de Gestão (MIS)	SI a nível de gestão da organização que suporta funções de planeamento, controlo e tomada de decisão, fornecendo relatórios e sumários rotineiros	Profissionais, gestores
	Sistemas de Suporte à Decisão (DSS)	SI a nível de gestão da organização que combina dados com capacidades analíticas poderosas, suportando a tomada de decisões semi-estruturadas	Gestores Intermediários
Conhecimento	Sistemas de Gestão do Conhecimento (KWS)	SI que auxiliam os trabalhadores do conhecimento na criação e integração de conhecimento	Pessoal técnico
	Sistemas de Escritório (Office Systems)	Sistema desenhado para aumentar a produtividade dos funcionários que manipulam os dados	Pessoal escritório
Operacional	Sistemas Transaccionais (TPS)	Sistemas que suportam e registam as actividades diárias necessárias ao desenvolvimento do negócio da organização	Operacionais

# Evolução SSD

Desde a década de 70, que os sistemas informáticos deixaram de ser usados exclusivamente para fins meramente científicos.

Progressivamente foram sendo introduzidos nas organizações, no início nas áreas contabilísticas e de processamento de salários e rapidamente se estendeu a outras áreas funcionais aprovisionamento, gestão da produção, etc.

Estas sistemas são normalmente denominados de sistemas operacionais, já que dão suporte à operacionalidade de cada organização.

# Evolução SSD

Na segunda metade da década de 70, foram atingidos os limites de utilização dos dados, e começaram a surgir novas formas de armazenamento e de acesso directo aos dados.

- O que conduziu às primeiras tentativas de esboço de um sistema de gestão de bases de dados.

Motivou uma melhoria nas aplicações informáticas, mas a inclusão da vertente analítica não era trivial. Era comum encontrara problemas tais como:

- Custo elevado das aplicações informáticas;
- Pouca flexibilidade na geração de relatórios;
- Limitações de recursos, agravados pela inexistência de mecanismos concorrentes aos dados e pela coexistência como os sistemas operacionais.

# Evolução SSD

Na década de 80, com o aparecimento da 4GL e dos PC's, tornou-se mais fácil a extracção e manipulação dos dados dos sistemas operacionais.

Passou a ser comum a difusão da informação dentro das organizações, embora sempre acompanhada de problemas:

- Gestores pouco interessados em abdicar *know how*;
- Informação gerada possa ser usada como base para outros estudos (sem o devido reconhecimento!)
- ...
- Mesmo quando era possível difundir a informação através de relatórios, estes não estavam pensados na partilha e eram gerados segundo os interesses do departamento ou área funcional que o gerava e não dos outros departamentos existentes na empresa.

# Evolução SSD

Desajustes de outros gestores e analistas acederem ao mesmo documento:

- Relatórios com informação reduzida da actividade operacional;
- Falta de visão temporal;
- Dúvidas na responsabilidade da informação;
- Surge então a necessidade de construção de um repositório comum para fins analíticos, onde todos pudessem aceder à informação, partilhando o mesmo nível de detalhe e com a garantia de que qualquer alteração seria repercutida ao nível do repositório de análise.



# Operacional *vs* Analítico

Quando se pensa uma base de dados para suporte operacional, temos de ter presente algumas preocupações:

- Modelo relacional normalizado;
  - Optimização quanto ao desempenho;
  - Construção de índices e vistas; etc.
- Quando se pensa numa aplicação analítica (tipicamente OLTP – Online Transaction Processing), podemos ter alguns problemas:
    - O acesso aos dados é feito de forma desligada da camada operacional (pondo em causa as regras de negócio que podem estar definidas a esse nível);
    - Configuração do sistemas desajustada para esta nova realidade (volume de dados envolvidos, predominância de interrogações em vez das inserções ou actualizações, alta necessidade de recursos físicos);

# Operacional *vs Analítico*

Possíveis resultados, da utilização de uma aplicação analítica sobre uma base de dados operacional:

- Incapacidade de integração, devido ao nível de detalhe do sistema operacional;
- Comprometimento do desempenho operacional;
- Incapacidade de lidar com informação proveniente de outras fontes de dados que não existam no modelo operacional.

# Operacional *vs Analítico*

Mesmo que uma determinada estrutura de dados contenha toda a informação relevante para o contexto, podem ser observadas as seguintes razões que complicam a aplicação de um Sistema Analítico sobre um Operacional:

- Diferentes níveis de detalhe entre a informação operacional e informação analítica;
- Actualização de dados num sistema operacional;
- Diferentes comunidades de utilizadores;
- Instruções de SQL típicas realizadas sobre sistemas operacionais e sistemas analíticos;
- Necessidades de recursos;
- Formas de acesso a dados e normalização relacional;
- Disponibilidade dos sistemas;
- Desempenho e recursos.

# Data Warehouse

Quando são identificadas tantas limitações no uso da informação operacional para fins analíticos, justifica-se pensar em criar uma estrutura independente para os dados operacionais e analíticos.

Um DW define-se como sendo um repositório analítico, onde a partir dos sistemas operacionais e por intermédio de um processo de transferência, os dados são armazenados de forma mensurável (tabelas de factos e medidas), sendo contemplada a sua caracterização, simultaneamente, segundo diversos eixos de análise (dimensões).

# Data Warehouse

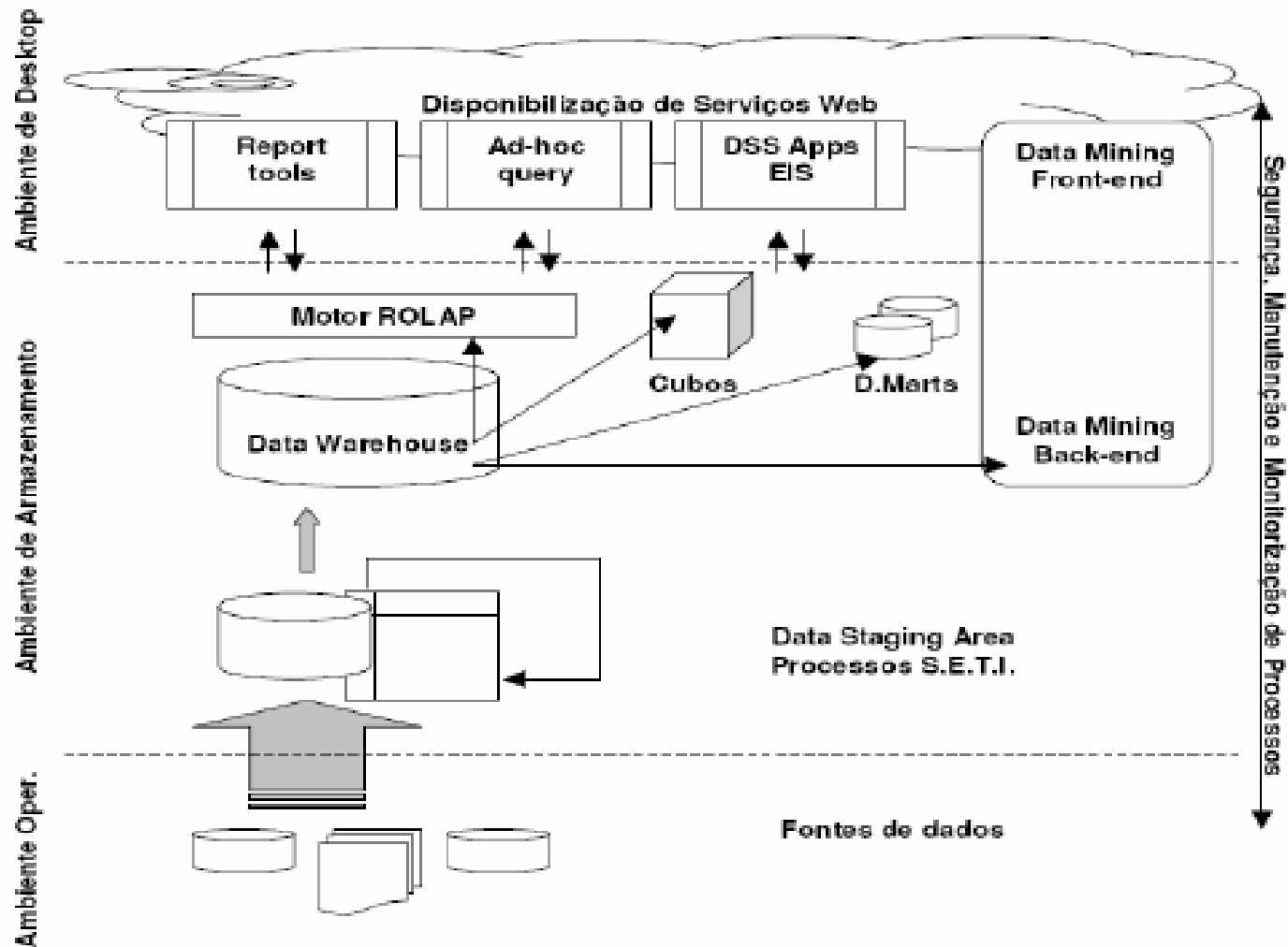
Um DW deverá garantir:

- Uma forma de acesso centralizada, mais facilitada e de forma legível a toda a informação da organização.
- Consistência em toda a informação que circula pela organização;
- Capacidade para lidar com requisitos flexíveis e como a própria dinâmica inerente à actividade da organização;
- Segurança no acesso e na circulação da informação dentro da organização;

# DataMarts vs DataWarehouse

- Um DM define-se como sendo uma divisão interna da organização criando sub-repositórios analíticos orientados para cada um dos departamentos ou áreas de negócio.
- O que significa que podemos olhar para um DW, como sendo uma conjugação de vários DM por áreas de negócio.
  - O que se tem verificado é quer por razões económicas quer por razões temporais a implementação de um DW pode ser feita incrementalmente.
  - Num primeiro instante pode ser contemplada a área de negócio mais de maior necessidade analítica – por exemplo a comercial, e com uma maior disponibilidade de recurso humanos e físicos

# Arquitectura



*Figura: Arquitectura Global de um Sistema de Suporte à Decisão*

(Fonte: Bruno Cortes, 2005)

# Arquitecturas SSD

- **1 nível:**

- numa única máquina é suportado tanto o sistema OLTP como a área de transformação de dados e o próprio DataWarehouse. Solução mais económica, mas com muitas limitações de utilização ao nível do processamento (não aplicável a SSD com múltiplas fontes de dados).

- **2 níveis:**

- OLTPs num lado e o ambiente de armazenamento suportado em hardware próprio. Uma arquitectura funcional mas onde o acesso ao DataWarehouse e os processos de transformação e carregamento concorrem por tempo de CPU. A sua aplicação é viável quando o SSD é flexível em termos de disponibilidade e a frequência da execução dos processos S.E.T.I. não é demasiado elevada.



# Arquitecturas SSD

## **3 níveis:**

- Para além do tratamento independente dos OLTPs, existem recursos alocados exclusivamente ao tratamento dos dados provenientes dos sistemas operacionais. Aplicáveis em soluções de alta disponibilidade com actualizações constantes. Convêm ser suportado por uma infra-estrutura de comunicação condizente e os testes de carga deverão reflectir os pontos críticos da solução.

## **• 4 níveis:**

- Um nível adicional, disponibilizando soluções aos utilizadores finais através de plataformas complementares de informação (Web, EDI, Palm, ...).

# Ciclo de vida de um SSD



*Figura: visão global do ciclo de vida de um SSD*

**(Fonte: Bruno Cortes, 2005)**

# Sumário

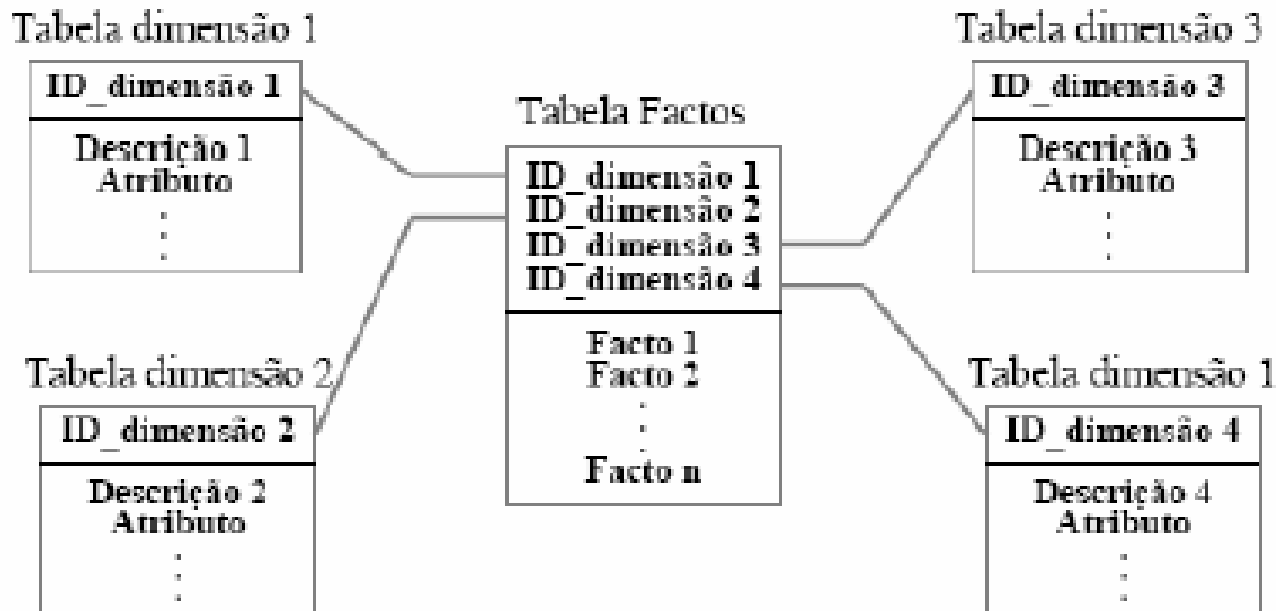
---

- Evolução SSD
- Operacional vs Analítico
- Data Mart vs Data Warehouse
- Ciclo de Vida de um SSD

# Modelação Dimensional

- A modelação dimensional num repositório analítico, traduz-se num conjunto de operações de desnormalização e de agregação dos dados que têm origem nos sistemas transaccionais.
- A nova estrutura de dados assenta num modelo com uma tabela central (onde se encontram os registos de actividade e eventos da organização) – designada por tabela de factos, e em tabelas complementares de caracterização dessa mesma actividade – designadas por dimensões.

# Estrutura lógica – Modelo em estrela



- Este modelo tem por finalidade garantir que o acesso à informação seja conseguido com uma única junção em álgebra relacional.

# Modelação Dimensional

- Não existe uma receita para a concepção de repositórios analíticos, mas podemos considerar um conjunto de passo mais ou menos comuns para o desenho de um Datawarehouse.
- Esta dificuldade prende-se com o facto de algumas condições ultrapassarem o consultor, como orçamentos, formas de armazenamento nas fontes operacionais, entre outras.

# Modelação Dimensional

- 1 – Especificação dos processo funcionais e de negócio para a definição da(s) tabela(s) de factos;
- 2 – Definição da granularidade de cada tabela de factos (nível de detalhe);
- 3 – Definição das dimensões necessárias;
- 4 – Definição das medidas dentro das tabelas de factos (ex: margem de lucro, rotatividade de stock);
- 5 – Definição dos atributos das dimensões;
- 6 – Detecção e tratamento de dimensões que vão variando no tempo (ex: morada de um cliente, ...)
- 7 – Definição dos níveis de agregação, dimensões heterogéneas, hierarquias e outras definições referentes à agregação dos dados;
- 8 – Análise do tempo de vida e do histórico de dados a carregar no repositório analítico, tendo em atenção questões como o desempenho;
- 9 – Necessidade de disponibilidade de dados.

# Tabela de Factos

É uma tabela de registos de observações da actividade de uma organização, observações essas caracterizada e mensuradas segundo as variáveis consideradas relevantes nos processos de suporte à decisão.

No desenho do esquema de dados, as tabelas de factos deverão conter todos os atributos relevantes e pelos quais será possível guiar a análise.

Cada um destes campos representará uma chave estrangeira para uma das tabelas de dimensão, que ajudarão a caracterizar a actividade da organização.

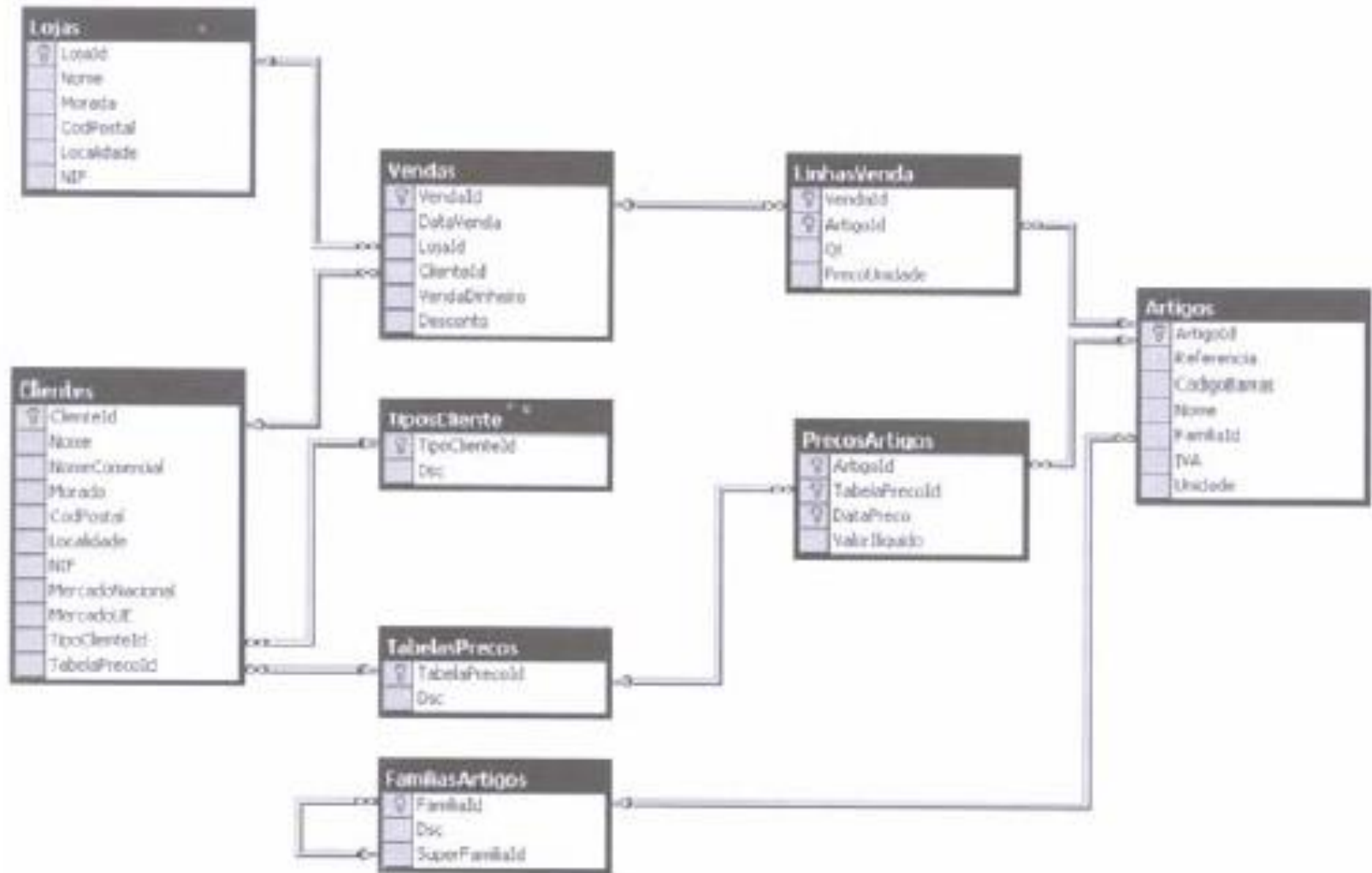


# Tabela de Factos

A realidade que se pretende modelar num repositório analítico, tem que ser mensurável, isto é, tem que ser possível definir quantitativamente o que representa num processo funcional.

Estes atributos são representados nas tabelas de factos como valores numéricos designados por “medidas”.

# Esquema Relacional



# Granularidade das Tabelas de Factos

Identificadas as tabelas de factos, segue-se a preocupação com a definição da granularidade.

Numa primeira fase necessitamos de analisar o tipo de perguntas que poderão ser colocadas pelos gestores ou analistas:

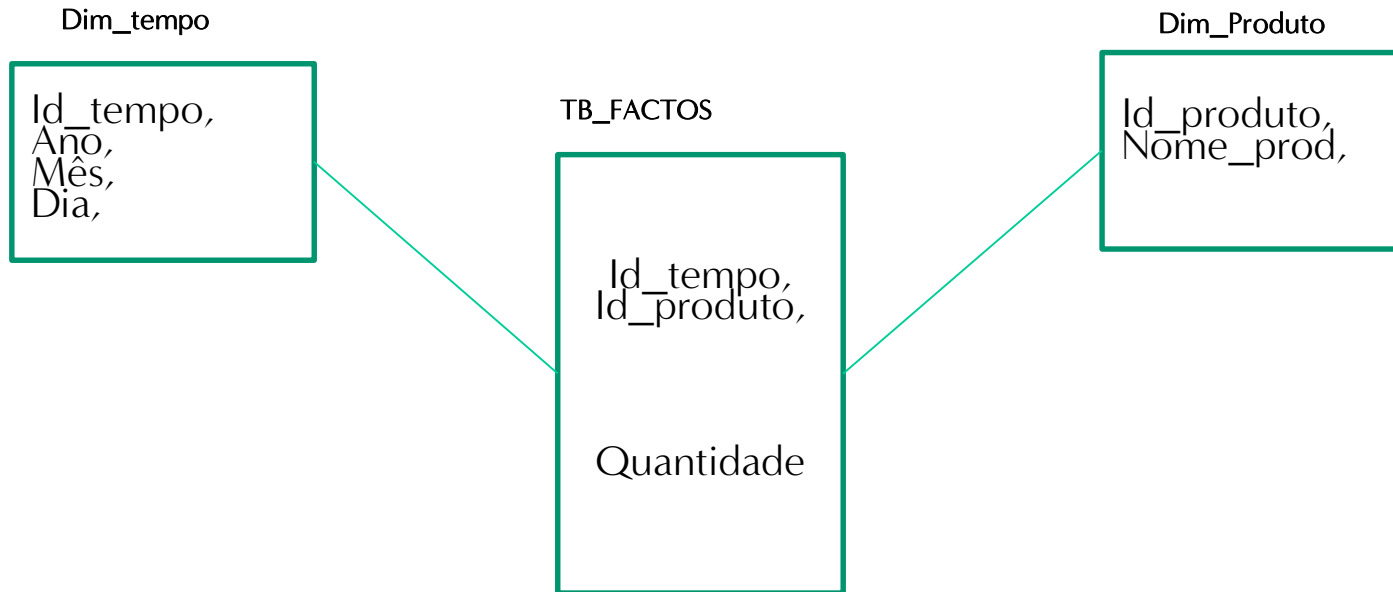
- Quanto é que se vendeu em caixas de CD hoje?
- Qual a loja com maior volume de vendas no mês de Maio?
- Quais os clientes que compraram um maior número de artigos na loja da Covilhã?

Para ilustrar a evolução da construção vai usar-se como exemplo um DataMart comercial.

# Modelo Multidimensional

Das questões para o modelo:

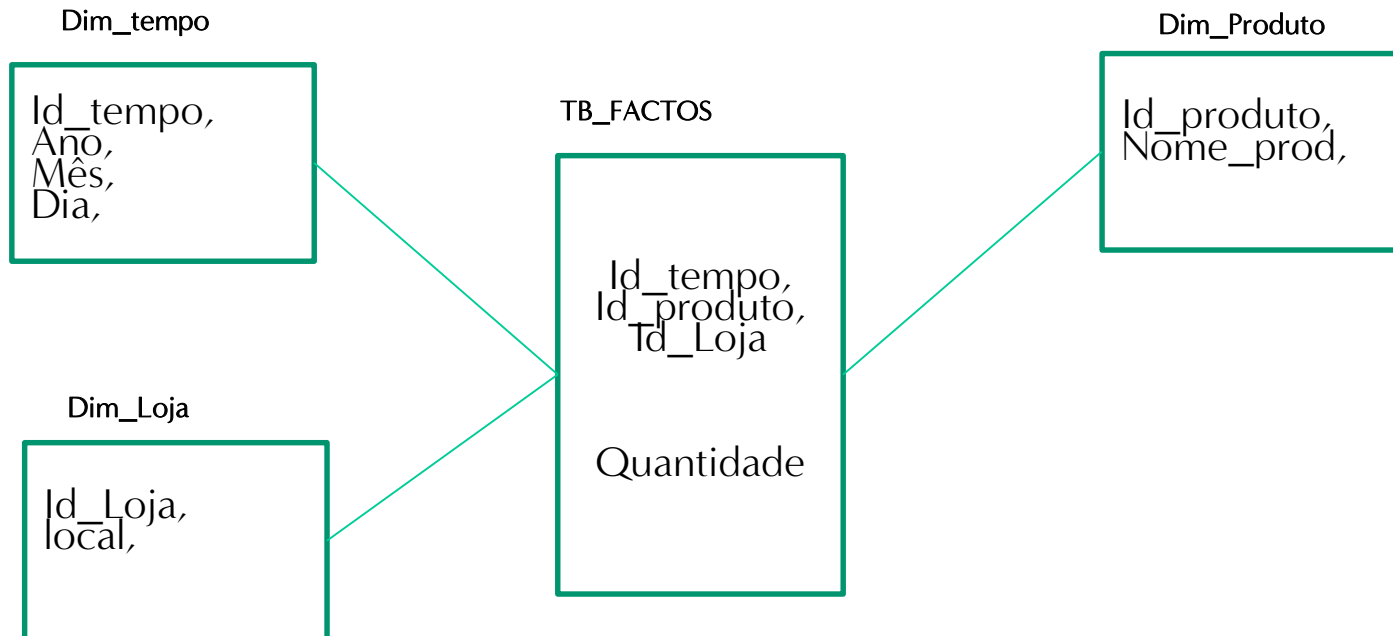
Quanto é que se vendeu em caixas de CD hoje?



# Modelo Multidimensional

## Das questões para o modelo:

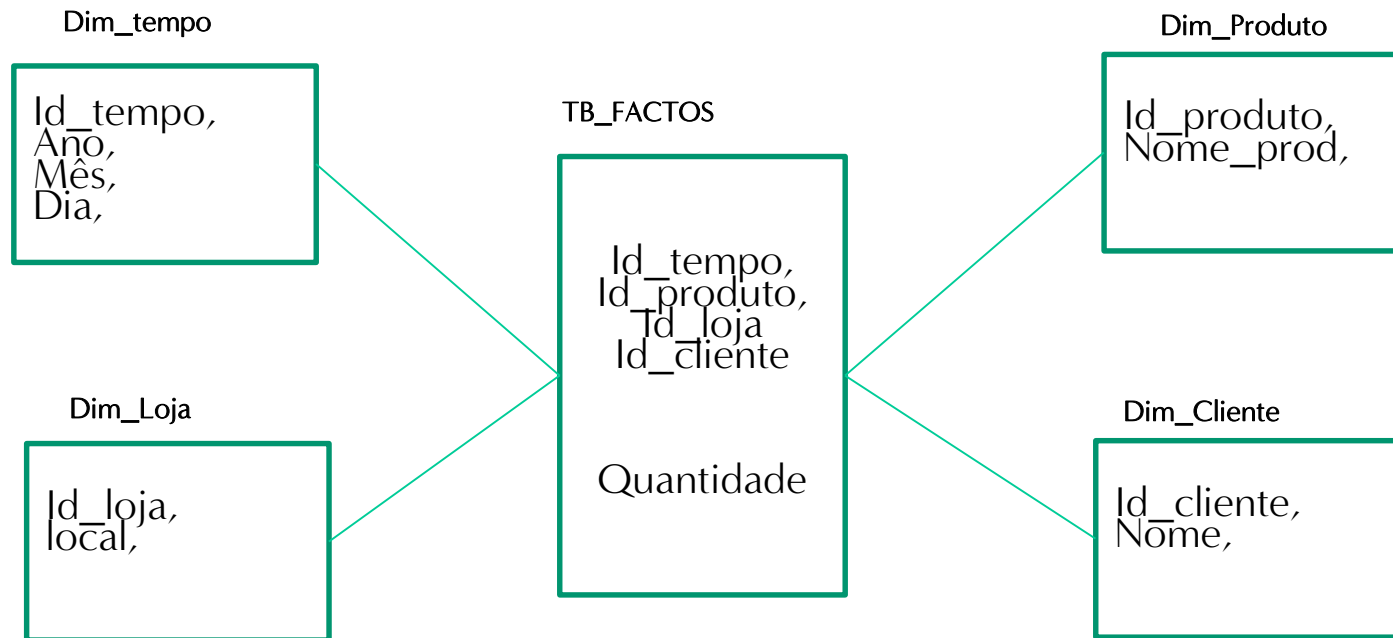
Qual a loja com maior volume de vendas no mês de Maio?



# Modelo Multidimensional

## Das questões para o modelo:

Quais os clientes que compraram um maior número de artigos na loja da Covilhã?



# Informação no modelo

Id_tempo	Id_produto	Id_loja	Id_cliente	.....	Quantidade
2013/01/01	1	1	1		50
2013/01/01	10	2	2		20
...					

- **1ª Linha:** Foi realizada uma compra pelo cliente 1 (Cliente Filipe – informação da relação com a dimensão “cliente”), do produto 1 (Produto XPTO – informação da relação com a dimensão “produto”), na loja 1 (Loja Castelo Branco - informação da relação com a dimensão “Loja”), na quantidade 50 (MEDIDA).
- ...

# Granularidade

De análise deste tipo de questões faz salientar do ponto de vista mais técnico as seguintes considerações:

- Haverá necessidade de conhecer informação detalhada ao nível dos acumulados das transacções diárias;
- Existirão perguntas que irão ao detalhe da consulta de vendas dos próprios artigos, enquanto que outras procurarão a agregação por famílias de artigos;
- A informação poderá ser analisada sobre a vertente dos clientes;
- A informação poderá se analisada do ponto de vistas das lojas que efectuam as vendas, sendo diferenciadas com base em critérios geográficos.

Deste cenário conclui-se que a granularidade será referente aos artigos vendidos, por cliente, por loja ao longo de cada dia.



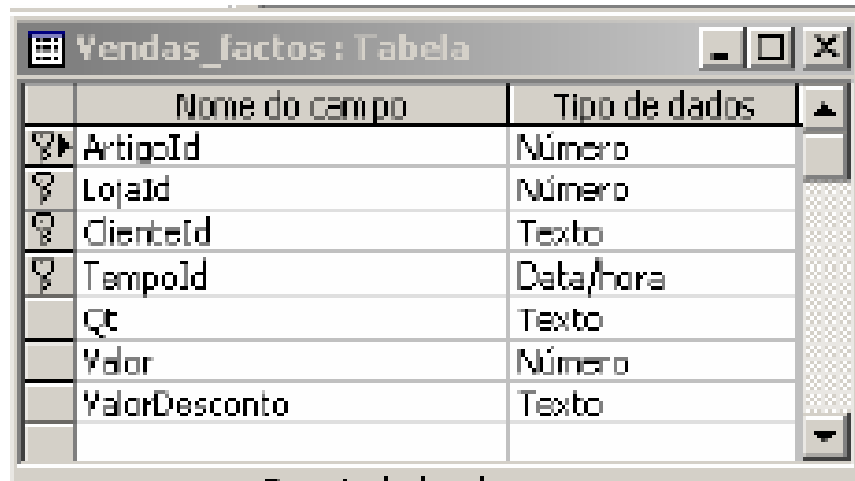
# Granularidade





- Cabe ao analista, definir esta granularidade ou tentar antecipar necessidades analíticas no futuro (redefinindo assim o grão).

Nas tabelas seguintes, apresentam-se duas propostas que definem o grão para resposta às questões apresentadas (tabela 1) e uma outra proposta que define o grão para necessidades futuras (tabela 2).


Note-se que no segundo caso a informação contida na tabela de factos é consideravelmente maior.

# Grão em tabelas de factos



	Nome do campo	Tipo de dados
	ArtigoId	Número
	LojaId	Número
	CienteId	Texto
	TempoId	Data/hora
	Qt	Texto
	Valor	Número
	ValorDesconto	Texto



	Nome do campo	Tipo de dados
	ArtigoId	Número
	VendaId	Número
	LojaId	Número
	CienteId	Número
	TempoId	Data/hora
	Qt	Número
	Valor	Número
	ValorDesconto	Número

# Dimensões

As dimensões são as tabelas que relacionadas com as tabelas de factos ajudam a caracterizar as observações segundo os diversos eixos de análise.

Segundo a proposta para a tabela de factos as possíveis dimensões de análise são:

- Temporal;
- Cliente;
- Artigos;
- Lojas.

(a inclusão de outras dimensões teriam de ser avaliadas segundo a informação disponível no sistema operacional e/ou outros).

# Dimensões

A grande maioria dos sistemas implementados costuma contemplar entre 4 a 12 dimensões.

A título de exemplo analisemos a dimensão temporal, uma das mais importantes na navegação de sistemas analíticos, nomeadamente:

- através das definição de níveis de agregação para operações de consulta em detalhe e de generalização (*Drill Down e Roll Up*) – *consultas por dia, por semana, por mês, por trimestre, por ano*
- ou pela aplicação de filtragem e análise seleccionada (Filtering e Slicing) – poder comparar, por exemplo vendas por períodos homólogos.

# Dimensões

Por exemplo se for

definido o grão de por  
dia durante 4 anos:

- Se for definido o grão do segundo durante os mesmos 4 anos:

Campos	Registos
Anos	4
Trimestres	16
Meses	48
Dias	1.461
Total	1.461

Campos	Registos
Anos	4
Trimestres	16
Meses	48
Dias	1.461
Horas	35.064
Minutos	2.103.840
Segundos	126.230.400
Total	126.230.400

# Exemplo de uma estrutura para a dimensão do tempo

## Tempo

ID\_data  
Dia\_do\_mês  
Dia\_da\_semana  
Dia\_do\_ano  
Semana\_do\_ano  
Mês  
Número\_do\_mês  
Trimestre  
Período\_fiscal  
Flag\_feriado  
Flag\_dia\_semana  
Flag\_último\_dia\_mês  
Estação\_ano  
Acontecimento\_espec  
.....

Existe sempre, pois representa a dependência temporal inerente à DW;

Deve descrever o tempo tal como ele é visto para fins de gestão da actividade (negócio) em causa;

Deve conter a caracterização do tempo nos atributos pelos quais se pretende posteriormente fazer pesquisas;

É gerada, normalmente, de um forma sintética (i.e., sem ser a partir de uma BD operacional) para todo o período de tempo considerado na DW.

# Desempenho - Agregação

- A concepção de um DW (ou DM), assenta no mais fino grão de informação, a partir do qual será possível efectuar consultas. As consultas poderão ir desde este máximo nível de detalhe até à informação agregada na forma de somatórios de valores para dar resposta a questões mais generalistas.

Contudo por muito preparadas que possam estas tabelas de factos, para eficientemente lidar com a questão do desempenho (correcta estrutura de índices) as operações serão sempre grandes consumidoras de recursos.

# Desempenho - Agregação

- Apesar da possível diversidade possível de questões, existem algumas que são colocadas com maior frequência e com uma certa periodicidade, como por exemplo:
- Quanto é que se vendeu do artigo X no dia de hoje?
  - Quais os equipamentos imobilizados que estiveram afectados, no ultimo mês, menos de 5% do tempo a um projecto?
  - Quais os clientes com maior volume de encomendas no ultimo mês?
- No primeiro exemplo, não são analisadas dimensões como clientes ou lojas, enquanto no segundo e no terceiro procedeu-se a uma agregação com base no mês quando, possivelmente o grão da dimensão tempo podia ser o segundo.



# Desempenho - Agregação

Para responder as estas questões temos de percorrer um vasto conjunto de dados armazenados na tabela de factos e fazer agregações com as mediadas necessárias.

- As agregações pré calculadas, são componentes transparentes do DW onde se pretende gerir vistas sumariadas sobre as tabelas de factos, para evitar o grande esforço de processamento.
- Em tabelas resumidas armazenam-se os mesmos resultados que se encontram disponíveis em tabelas de factos, só que envolvendo um outro grão e apenas um sub conjunto da dimensões originais.

# Desempenho - Agregação

Vantagens de valores agregados pré calculados:

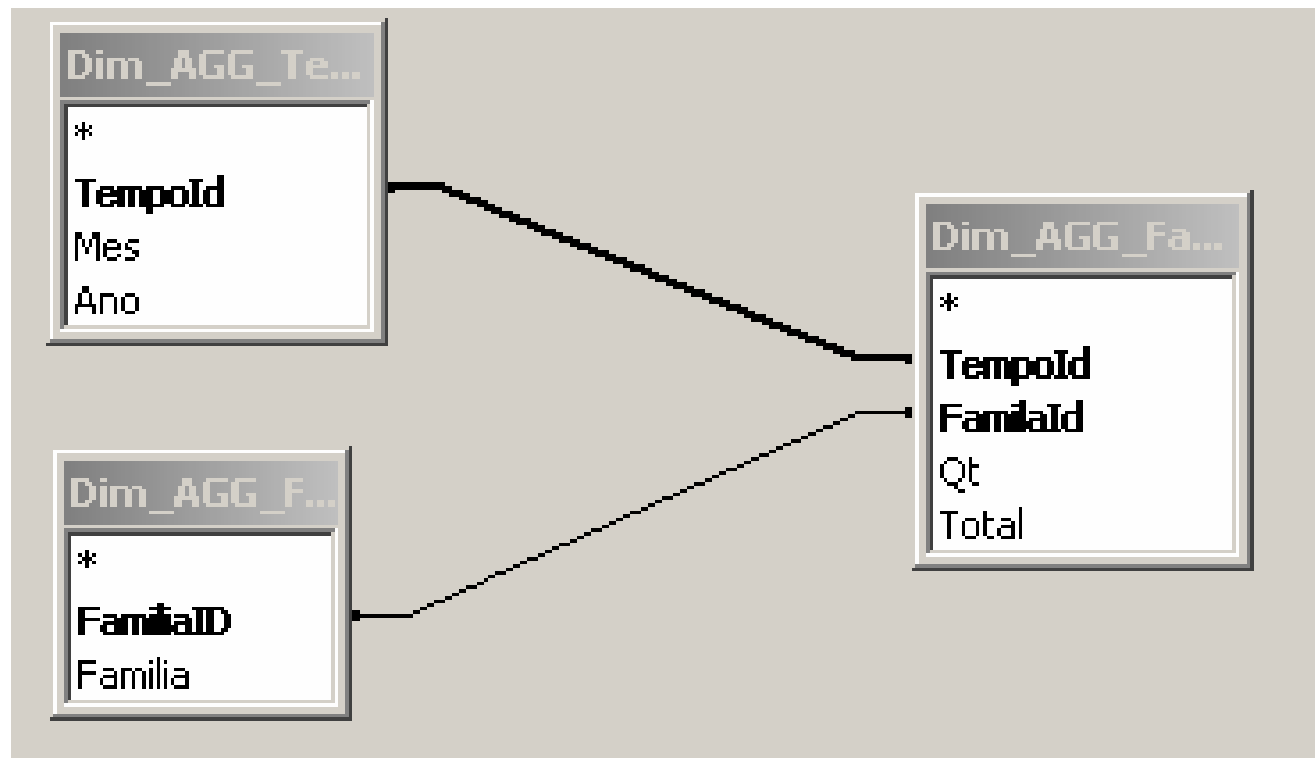
- Ganhos de tempo de respostas em questões que necessitam de trabalhar os registos ao nível do detalhe para realização de *Roll Up*.
- Economia de recursos no processamento das consultas de dados agregados;
- Utilização transparente para os utilizadores finais;
- Reduzido acréscimo do volume de dados do DW.

# Desempenho - Agregação

- Conjunto de linhas orientadoras para a implementação de valores agregados pré calculados:
  - Separação dos valores agregados pré calculados em tabelas próprias;
  - As tabelas de dimensões usadas como referência das tabelas de agregação deverão ser versões reduzidas – e provavelmente de graus menos refinado.
  - As tabelas de factos, de dimensões e de agregações que constituem uma base de análise deverão ser mantidas num mesmo esquema relacional e com um mesmo dono

# Desempenho - Agregação

## Estrutura de Agregação Pré Calculada



# Desempenho - Índices

- O planeamento de uma politica de índices deve ser levado em conta logo na fase de concepção do DW.
- Com a convicção de que essa politica irá sofrer diversas alterações ao longo do tempo (à medida que se vai consolidando a percepção dos dados que se têm armazenados, os mais relevantes e frequentemente solicitados, ... ).
- Devem ser realizados os mais variados testes, tendo em conta o(s) campo(s) a indexar e os diversos tipos de índices.

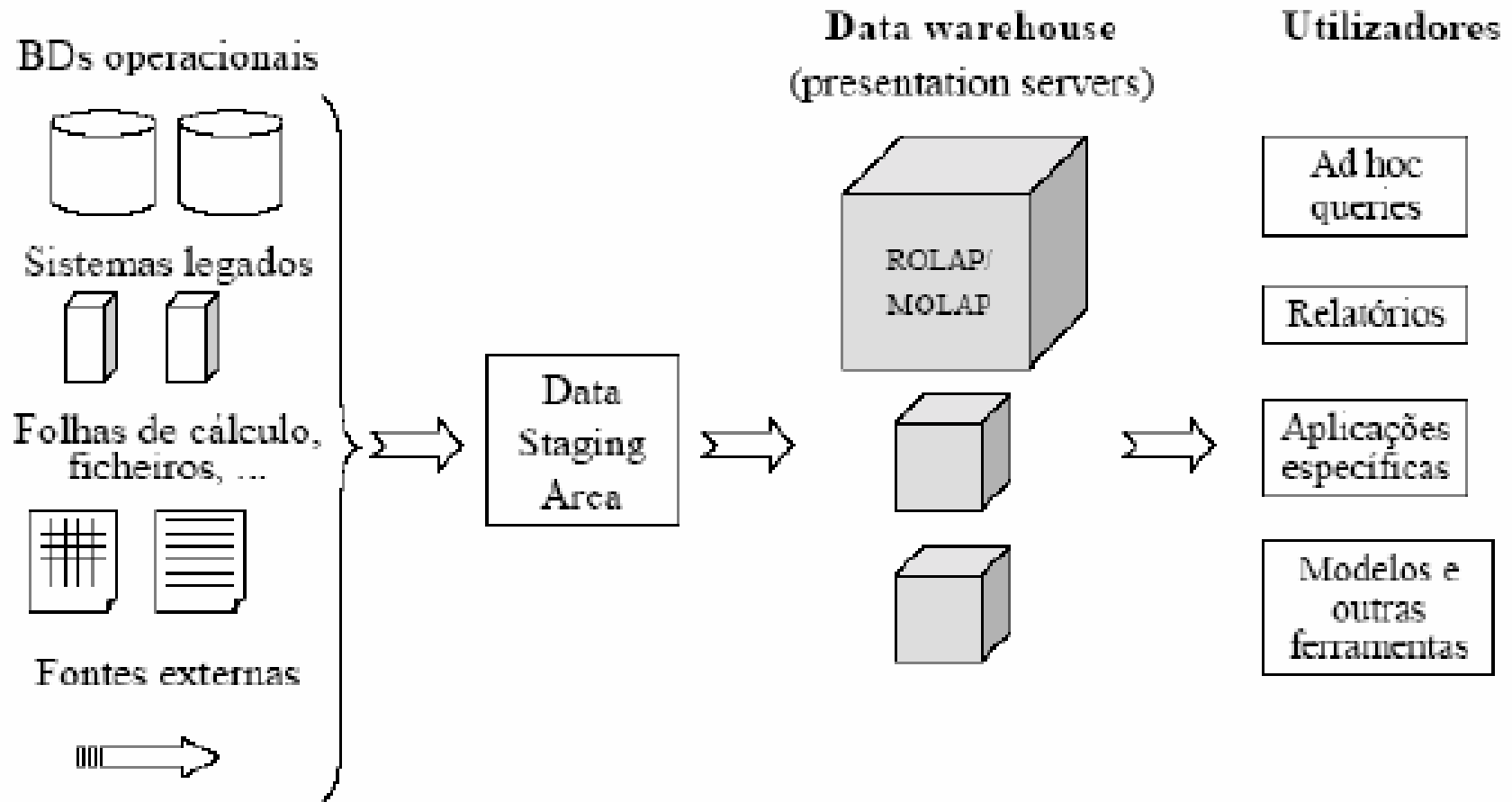
# SETI

- SETI – Selecção, Extracção, Transformação e Integração de dados.

Até a informação estar disponível num DW, muitos passos são necessários dar no sentido de transformar e integrar dados puramente operacionais com origem nos sistemas transaccionas da organização, dando-lhes um cariz mais analítico e permitindo a sua exploração numa vertente completamente distinta.

Este trabalho de preparação dos dados (SETI ou ETL – *Extract, Transform and Load* ) é realizado numa área chamada *Data Staging Area*.

# Data Staging Area



# Fontes Heterogéneas de Dados

- Muitas vezes podemos encontrar informação relevante para o DW em distintas fontes dentro de uma organização (CRM, ERP,...).

E ainda externamente à organização – muitas vezes podem ser relevantes dados de organizações externas como por exemplo dados demográficos, dados do INE, etc. Dados estes que podem ter o seu próprio formato de armazenamento.

Sendo então a DSA a responsável pelo trabalho de integração desses dados.



# Área de Processamento de Dados

Segundo Kimball:

- “como um restaurante, os cliente devem ter acesso ao prato final e não à sua confecção”

Idealmente a DSA deveria estar afastada em termos físicos quer do modelo operacional, quer do modelo analítico, por forma a não concorrer com os recursos de ambos.

# Estrutura física para um DW

- Devemos planejar qual a melhor forma de armazenar informação do DW.
- Muita da eficiência e da organização deve-se a este trabalho inicial.
- A nossa primeira preocupação vai para a criação de uma estrutura física que dê suporte ao nosso DW.

# Exemplo - Para o caso do SqlServer

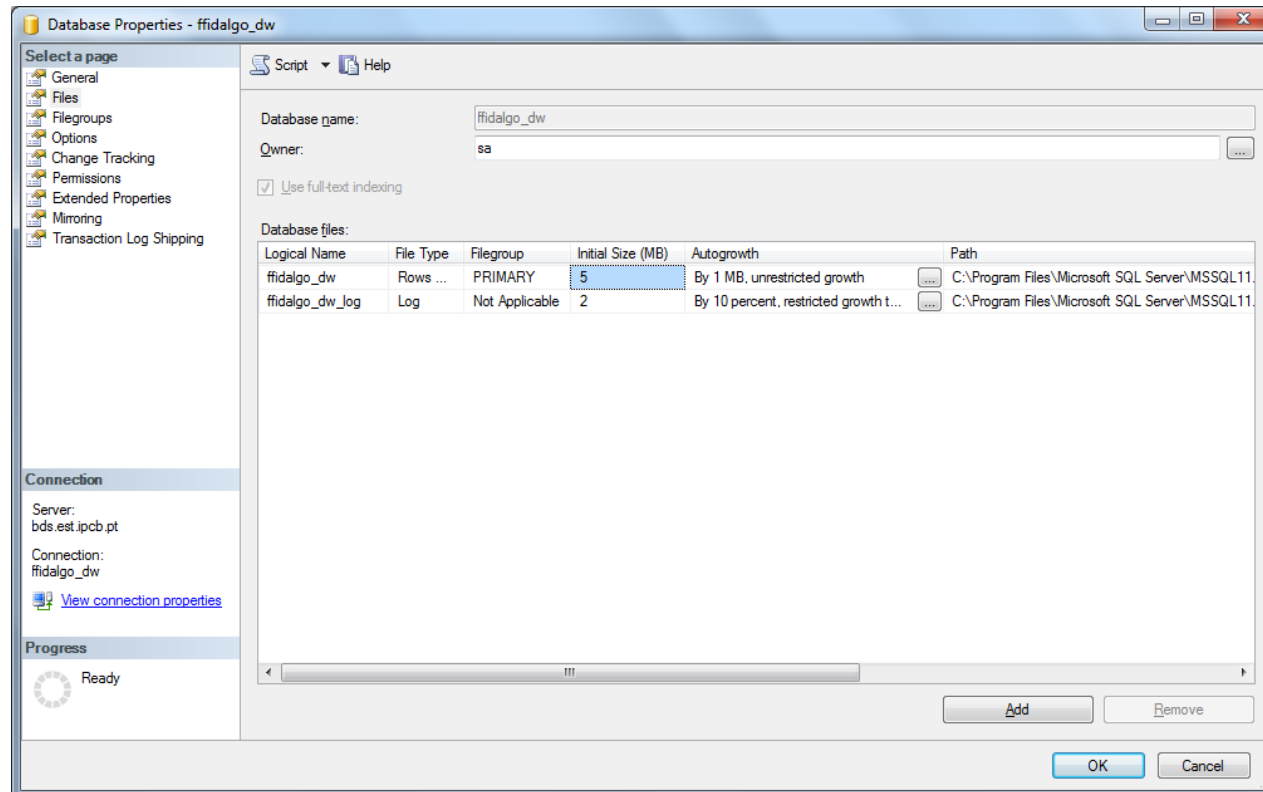
- Embora as abordagens que vamos ver se possam aplicar em qualquer circunstância, vamos usá-las para o modelo multidimensional.

Vamos ver, que podemos gerir a forma de como a nossa estrutura (tabelas, índices,...) pode ser criada.

Em particular, o local onde vamos colocar a informação, propriamente dita, constante nas tabelas vai ser guardada.

# Exemplo - Para o caso do SqlServer

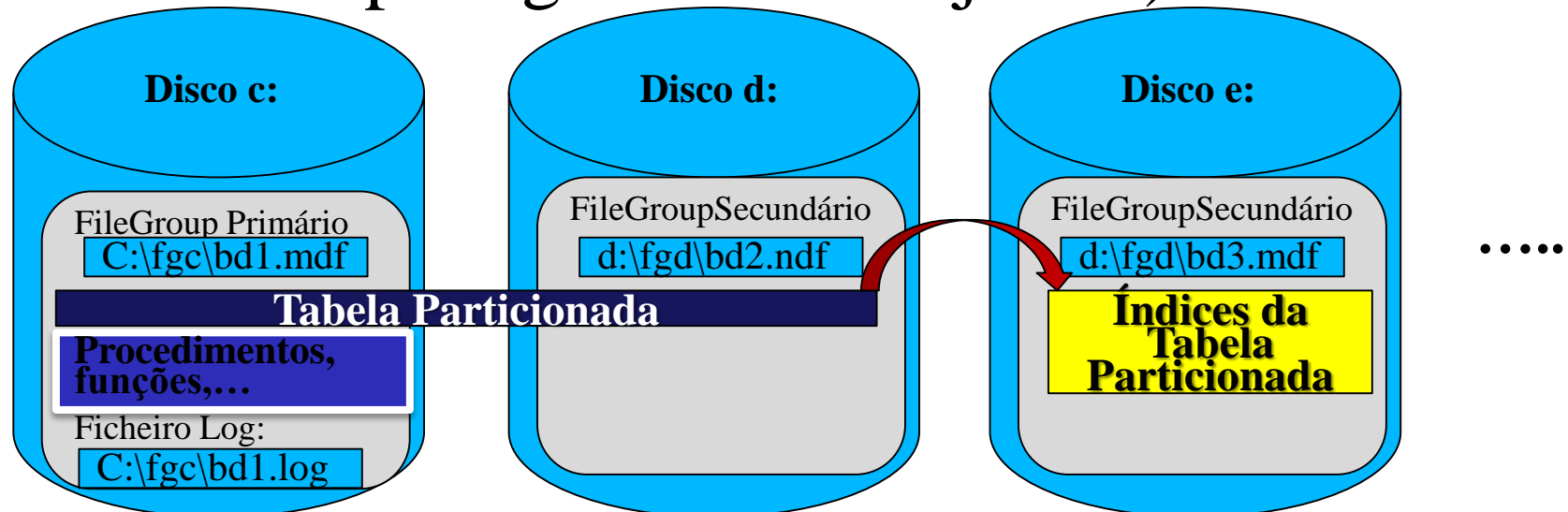
- Quando se cria uma base de dados (por defeito), fica associado um grupo de ficheiros (FileGroup) com dois ficheiros:
  - Primary (com extensão -> .mdf);
  - Log (com extensão -> .log).



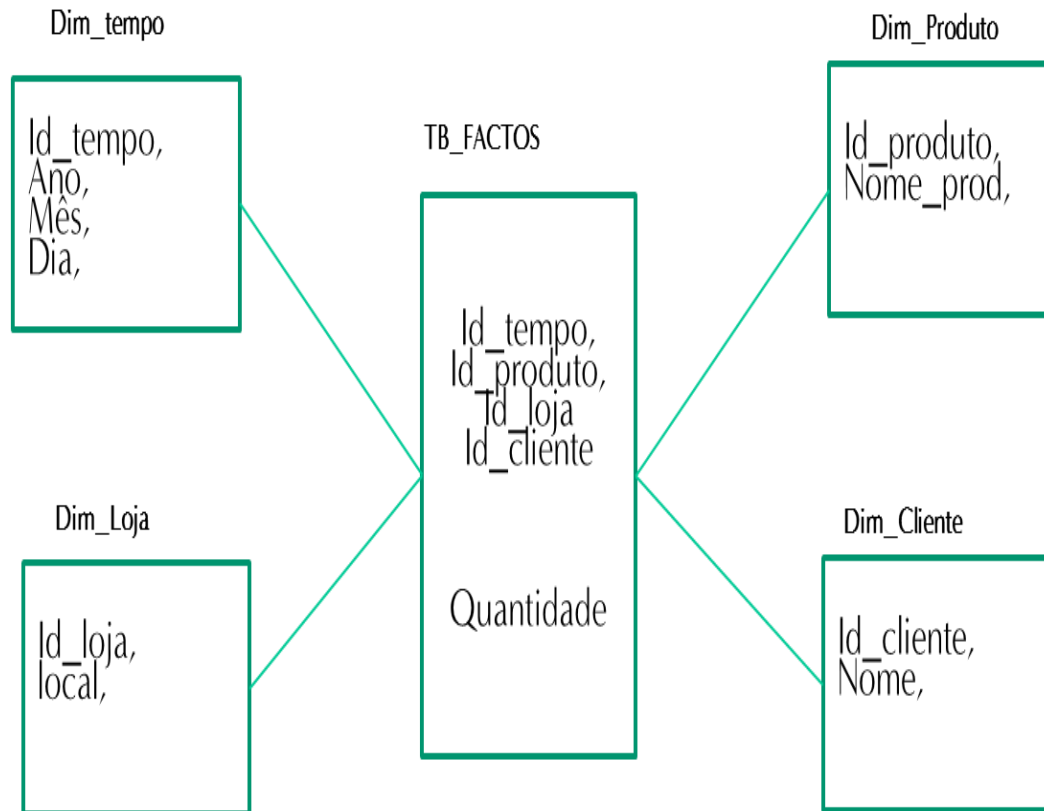
# Exemplo - Para o caso do SqlServer

- Adicionalmente podemos criar mais grupos de ficheiros (ficando estes com extensão -> .ndf)

Ao usarmos estes mecanismos, podemos controlar a criação dos objectos (usar diferentes discos/locais no servidor para guardar os objectos).



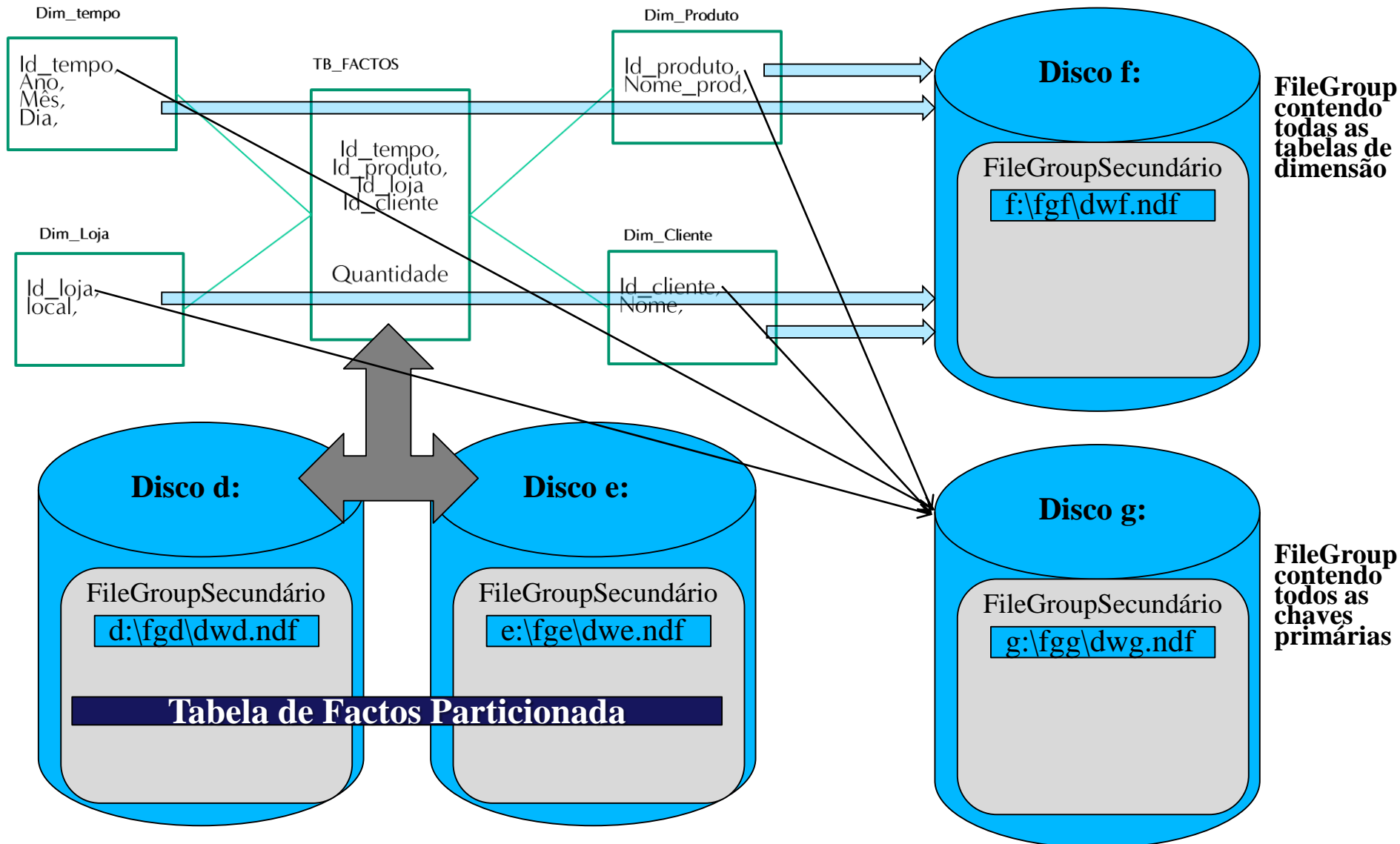
# Exemplo - Para o caso do SqlServer



•Aplicando estes conceitos ao nosso exemplo, vamos:

- 1) Criar 4 filegroups;
- 2) Num vamos colocar todas as tabelas de dimensões;
- 3) Noutro vamos colocar os índices;
- 4) Nos outros dois vamos particionar a tabela de factos

# Exemplo - Para o caso do SqlServer



# Exemplo - Para o caso do SqlServer

- Fase 1 – A criação dos Filegroups

Sendo que a nossa base de dados se chama BD2\_DW, vamos criar os 4 filegroups, com o ficheiro de dados, em discos distintos:

```
ALTER DATABASE bd2_dw  
ADD FILEGROUP FG_D;
```

```
ALTER DATABASE bd2_dw  
ADD FILEGROUP FG_E;
```

```
ALTER DATABASE bd2_dw  
ADD FILEGROUP FG_F;
```

```
ALTER DATABASE bd2_dw  
ADD FILEGROUP FG_G;
```



# Exemplo - Para o caso do SqlServer

- Fase 2 – A criação dos ficheiros para os filegroups

Teremos de realizar esta acção, para os 4 filegroups:

```
ALTER DATABASE bd2_dw ADD FILE  
( NAME = dw_d,  
  FILENAME = 'd:\fgd\dw_d.ndf',  
  SIZE = 1MB,  
  MAXSIZE = 50MB,  
  FILEGROWTH = 1MB)  
TO FILEGROUP FG_D
```

# Exemplo - Para o caso do SqlServer

- Fase 3 – A criação de uma tabela em determinado FileGroup

(A criação das chaves vamos fazer mais à frente, para que também estas sejam criadas num filegroup específico. Atenção apenas ao facto de que o atributo candidato a chave, terá de ser criado com a flag “not null”)

```
create table dim_produto(  
  id_produto int not null,  
  nome_produto varchar(30)  
)  
on fg_f;
```

# Exemplo - Para o caso do SqlServer

- Fase 3 – A criação de um índice num determinado FileGroup

(Não é obrigatório que se crie desta forma, mas esta é o processo que permite criar o índice num filegroup diferente)

```
alter table dim_produto  
add constraint tb_dim_produto_pk  
primary key(id_produto) on FG_G;
```

OU (no próprio ato da criação da tabela)

```
create table dim_produto(  
id_produto int constraint tb_dim_produto_pk primary key(id_produto) on FG_G,  
nome_produto varchar(30)  
)  
on fg_f;
```

# Exemplo - Para o caso do SqlServer

- Fase 3 – A criação de um índice num determinado FileGroup

No caso de não pretendermos criar os índices em filegroups diferentes podíamos criar da forma “tradicional”, sendo que assim o índice fica no mesmo filegroup da tabela:

```
create table dim_produto(  
  id_produto int constraint tb_dim_produto_pk primary key,  
  nome_produto varchar(30)  
)  
on fg_f;
```

# Exemplo - Para o caso do SqlServer

- Fase 3 – A criação de um índice num determinado FileGroup

Assim para os restantes casos:

```
create table dim_produto(  
id_produto int not null ,  
nome_produto varchar(30)  
) on fg_f;
```

• • • •

```
alter table dim_produto  
add constraint  
tb_dim_produto_pk  
primary key(id_produto)  
on FG_G;
```

# Exemplo - Para o caso do SqlServer

- Fase 4 – A criação da tabela de Factos Particionada

A criação de uma tabela particionada, envolve 3 passos:

- 1) A criação de uma Função de Partição que serve para definir "limites";
- 2) A criação de esquema de Partição, que serve para associar os limites, criado pela função de partição, aos diferentes filegroups;
- 3) A criação da tabela, indicando o atributo pelo qual a partição vai ser realizada (e que vai definir, por cada vez que se realize uma inserção, qual o “local”(ficheiro físico no disco) em que esta vai acontecer)

# Exemplo - Para o caso do SqlServer

- Fase 4 – A criação da tabela de Factos Particionada

- 1) A criação de uma Função de Partição que serve para definir "limites";

Podemos “partir” uma tabela da diversas formas, criando para o efeito determinadas classes. Neste caso vamos usar uma partição, pelo valor do “id\_produto”. Vamos criar duas classes:

	Min	Max
1º Intervalo	$-\infty$	100
2º Intervalo	101	$+\infty$

**CREATE PARTITION FUNCTION**

**partFuncPorIdProduto (int)**

**AS RANGE LEFT**

**FOR VALUES (100)**

# Exemplo - Para o caso do SqlServer

- Fase 4 – A criação da tabela de Factos Particionada

2) A criação de esquema de Partição, que serve para associar os limites, criado pela função de partição, aos diferentes filegroups;

É através desta estrutura que definimos em que Filegroup os registros serão armazenados, de acordo com o “id\_produto” e com base na PARTITION FUNCTION, criada no ponto anterior.

```
CREATE PARTITION SCHEME partSchemePorIdProduto  
AS PARTITION partFuncPorIdProduto  
TO (FG_D, FG_E)
```

Assim, cada tuplo inserido com o “id\_produto” menor do que 101, vai para o filegroup “FG\_D”, os maiores ou iguais vão para o filegroup “FG\_E”



# Exemplo - Para o caso do SqlServer

- Fase 4 – A criação da tabela de Factos Particionada

3) A criação da tabela, indicando o atributo pelo qual a partição vai ser realizada (e que vai definir, por cada vez que se realize uma inserção, qual o “local”(ficheiro físico no disco) em que esta vai acontecer)

```
CREATE TABLE tb_factos
```

```
( Id_tempo    date,
```

```
  id_produto  int,
```

```
  id_cliente  int,
```

```
  id_loja     int,
```

```
  quantidade  int )
```

```
ON partSchemePorIdProduto(id_produto);
```

# Como criar uma conta em SqlServer

- Terão de criar um utilizador, para tal poderão logar-se com o utilizador:

User: ffidalgo\_dw

Pass: fidalgo

E executar o seguinte código (alterando a parte sublinhada, para cada caso particular)

```
USE [master]
```

```
GO
```

```
CREATE LOGIN [MDSSI_N_ALUNO_1415] WITH PASSWORD=N'bd2',  
DEFAULT_DATABASE=[master], CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
```

```
GO
```

```
EXEC master..sp_addsrvrolemember @loginame = N'MDSSI_N_ALUNO_1415', @rolename =  
N'dbcreator'
```

```
GO
```

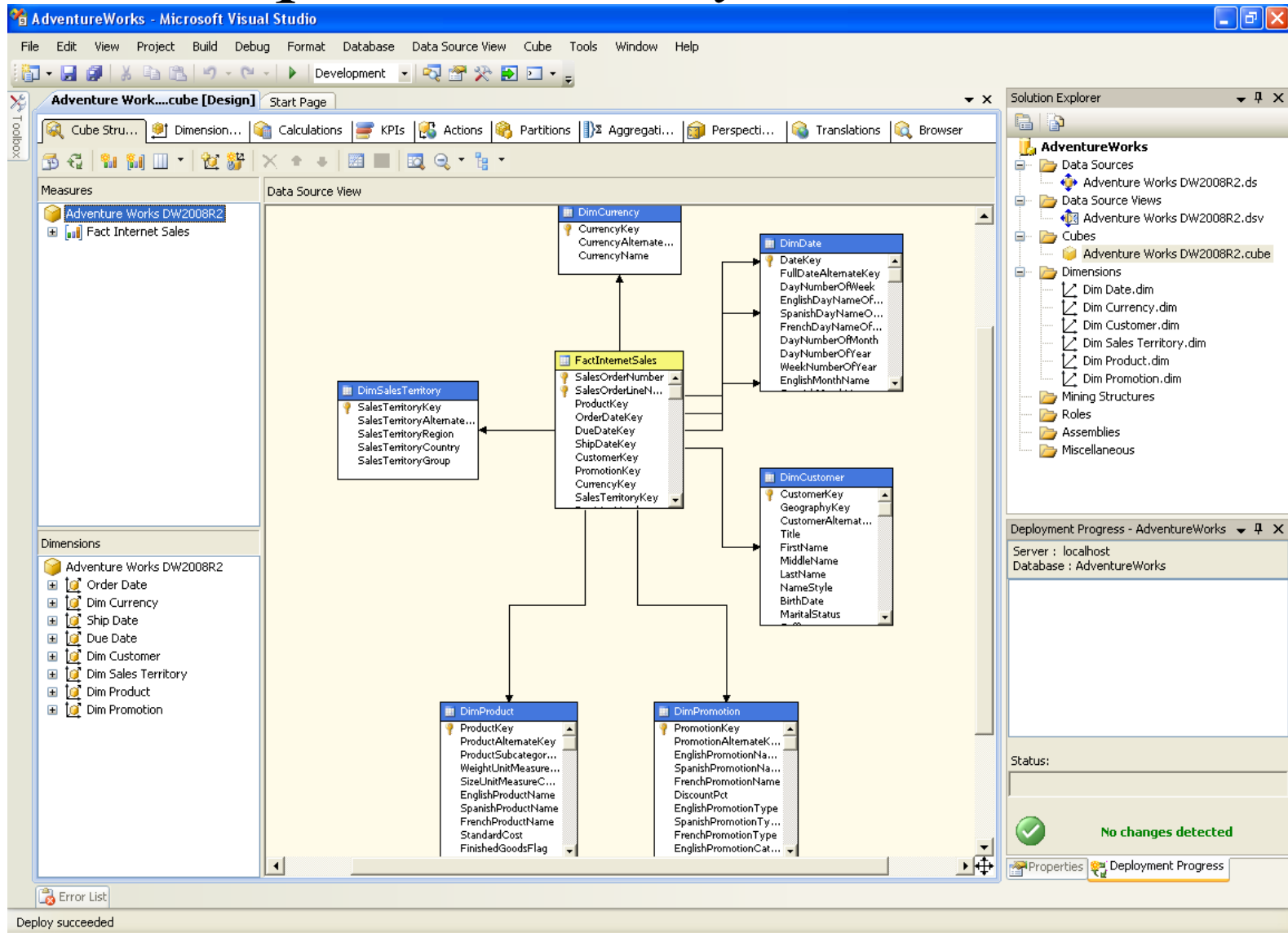
Depois terão de criar uma base de dados:

```
create database MDSSI_N_ALUNO_1415
```

# Business Intelligence Projects



# Ferramentas de análise: *Sql Server Analysis Services*



# Revisão

- Vimos:
  - **Datawarehouse;**
  - **Modelação Multidimensional;**
  - **Ferramentas de análise de dados em DW;**