

**CRIAÇÃO DE UMA DATAWAREHOUSE EM ORACLE COM UTILIZAÇÃO DO
ORACLE DISCOVERY COMO TOOL DE EXPLORAÇÃO**

André P. Muga
apmuga@apmuga.com

Índice

1.	Introdução	4
2.	Criação do Modelo	5
2.1.	<i>Análise da informação disponível</i>	5
2.2.	<i>Análise das necessidade requeridas da datamart</i>	5
2.3.	<i>Modelo em estrela proposto</i>	6
2.4.	<i>Definição da granularidade.....</i>	6
2.5.	<i>Tamanho previsto</i>	6
2.6.	<i>Informação extra</i>	7
3.	Instalação e configuração do software Oracle	8
3.1.	<i>Software Utilizado</i>	8
3.2.	<i>Configuração da base de dados para o Data Mart.....</i>	8
3.2.1.	<i>Criação dos utilizadores e definição dos seus priviégios</i>	8
3.2.2.	<i>Configuração do espaço utilizado</i>	8
3.2.3.	<i>Criação das tabelas de dados.....</i>	9
4.	Extract Transform and Load.....	11
4.1.	<i>Definição do processo de importação.....</i>	11
4.2.	<i>Importação dos dados Operacionais</i>	11
4.3.	<i>Importação dos dados externos.....</i>	11
4.4.	<i>Geração da dimensão Tempo</i>	12
5.	Optimização	13
5.1.	<i>Criação de índices</i>	13
5.2.	<i>Optimização do Oracle 9i</i>	13
6.	Discoverer Administrator	14
6.1.	<i>Criação do EUL.....</i>	14
6.2.	<i>Definição de priviégios</i>	14
6.3.	<i>Preparação da Bussiness Area</i>	14
7.	Discoverer Desktop.....	15
7.1.	<i>CAD_MELHORES_CLIENTES.....</i>	15
7.2.	<i>CAD_LOCAL_MELHORES_CLI.....</i>	15
7.3.	<i>CAT_PROD MAIS VENDIDOS.....</i>	15
7.4.	<i>CAD_EVOLUCAO_VENDAS</i>	15
7.5.	<i>CAD_CAT_PROD MAIS VENDIDOS.....</i>	16
8.	Resumo e experiencia adquirida	17

Anexos

ANEXO 1 – MODELO OPERACIONAL	18
ANEXO 2 – MODELO ESTRELA.....	19
ANEXO 3 – SCRIPTS DE CRIAÇÃO DAS TABELAS DO DATAMART	20
ANEXO 4 – EXTRAÇÃO DOS DADOS EXTERNOS	22
ANEXO 5 – GERAÇÃO DA INFORMAÇÃO DE TEMPO	23
ANEXO 6 – TRANSFORMAÇÃO DOS DADOS DE PRODUÇÃO	24
ANEXO 7 – CRIAÇÃO DOS ÍNDICES BITMAP JOIN	27
ANEXO 8 – ALGUNS COMANDOS SQL UTILIZADOS	28
ANEXO 9 – SCREENSHOTS DO SOFTWARE CRIADOS PARA O APOIO.	29

Tabelas

TABELA 1	8
TABELA 2.....	8

1. Introdução

A necessidade de ferramentas adequadas para atender aos gerentes proporcionou o surgimento no mercado de novas tecnologias como respostas às suas solicitações.

Inicialmente, essas tecnologias não estavam preparadas para gerar e armazenar as informações estratégicas necessárias a uma gestão eficiente dos negócios ou operação da empresa, particularmente no momento da tomada de decisão. Essa inaptidão deve-se ao facto das ferramentas eram inicialmente desenvolvidas para o processamento de informação de produção e não para consultas de grande volume de informação. A verdade é que as metodologias utilizadas em OLTP não são compatíveis com as necessidades do OLAP.

Com o advento do Data WareHousing as ferramentas de processamento analítico on-line (OLAP - On-line Analytical Processing) passaram a ter um grande destaque. Segundo alguns autores, o sucesso de um Data Warehouse pode depender da ferramenta certa para atender as necessidades do utilizador, de onde podemos facilmente compreender o grau de importância em seleccionar a ferramenta adequada.

A ferramenta OLAP é constituída de um conjunto de tecnologias especialmente projectadas para dar suporte ao processo de decisão através de consultas, análises e cálculos mais sofisticados nos dados corporativos, estejam armazenados numa Data Warehouse ou não, por parte dos seus utilizadores, geralmente analistas, gestores e executivos.

Para permitir uma melhor classificação, as ferramentas OLAP estão divididas em ferramentas que utilizam um banco de dados multidimensional (MOLAP) ou em ferramentas que armazenam os dados em banco de dados relacionais (ROLAP).

Este tutorial propõe como objectivo criar e alimentar uma datamart e utilizar o Oracle Discoverer como ferramenta de análise inteligente sobre a datamart criada.

Todo o material utilizado na fase de ETL, scripts e programas utilizados encontra-se disponível dentro de um ficheiro zip alocado no seguinte endereço:

http://www.apmuga.com/Docs/DataWarehouse_Oracle.zip

2. Criação do Modelo

2.1. *Análise da informação disponível*

Foi utilizado um dump de uma base de dados operacional. O seu modelo esquemático ANEXO 1 – Modelo Operacional encontra-se no ficheiro ZIP já referido. Juntamente com isso foi disponibilizado também ficheiros em Excel contendo a informação sobre canais de vendas e promoções efectuadas.

Dessa documentação conseguimos extrair as seguintes entidades:

- Clientes (CUSTOMERS)
- Países (COUNTRIES)
- Vendas (SALES)
- Produtos (PRODUCTS)
- Compras (BUY)
- Subcategorias (SUB_CATEGORIES)
- Categorias (CATEGORIES)
- Fornecedores (SUPPLIERS)
- Promoções (PROMOTIONS)

2.2. *Análise das necessidade requeridas da datamart*

Como datamart, o sistema será utilizado para pesquisas de dados focadas no negócio. Como tal serão executadas pesquisas no sentido de responder as perguntas do negócio. Pensamos que as perguntas mais correntes serão as seguintes:

- Que produtos são os mais vendidos?
- Que produtos são menos vendidos?
- Quais são os melhores clientes?
- Quais são as melhores categorias de produtos?
- Como evolui as vendas no ano, na semana?
- Quantos clientes perdemos, por ano?
- Qual é o perfil dos nossos melhores clientes? E dos piores?

Devido as limitações do modelo operacional disponível não será possível responder a perguntas do género:

- Quais são os produtos mais lucrativos?

Neste caso falta a informação de preço de custo do item vendido pois não se consegue obter essa informação devido a relação entre produtos e fornecedores.

- Que produtos são mais dispendiosos de manter em stock?
Não existe informação sobre aprovisionamento.

2.3. Modelo em estrela proposto

No ANEXO 2 – Modelo Estrela está apresentado o nosso modelo em estrela que iremos utilizar neste projecto. Trata-se de uma simplificação clássica do modelo operacional.

As tabelas BD_PROMOCOES e BD_CANAISVENDAS provêm integralmente da conversão das folhas de excel respectivas e carregadas para a datamart. A tabela de facto deste modelo em estrela provêm dos dados da tabela de sales do sistema operacional e os dados dos clientes provêm da junção da tabela Customers e Countries.

2.4. Definição da granularidade

Como em qualquer projecto de Data Warehousing, é necessário definir a granularidade das dimensões do modelo. No nosso caso, essa tarefa é relativamente simples.

- Dimensão BD_TEMPO
Até ao dia já que não dispomos de informação de horas nos factos
- Dimensão BD_CLIENTE
Será até ao cliente, tendo os mesmos clientes do sistema operacional.
- Dimensão BD_PRODUTOS
Será até ao produto, tendo os mesmos produtos do sistema operacional.
- Dimensão BD_PROMOCOES
Será até a promoção, tendo as mesma promoções do sistema operacional.
- Dimensão BD_CANAISVENDA
Será até ao canal de venda, tendo os mesmos canais de venda do sistema operacional.

2.5. Tamanho previsto

Para o calculo do tamanho da nossa datamart é necessário ter certas assumções. Será válido pensar que o numero de clientes não irá variar e que o numero de produtos não aumentará muito durante o tempo de vida da datamart.

- Dimensão BD_TEMPO, 365 dias X 12 meses X 10 anos = 43800 dias.
- Dimensão BD_CLIENTE, 50000 clientes.
- Dimensão BD_PRODUTOS, 10000 produtos.

- Dimensão BD_PROMOCOES, entre 500 e 9999 promoções, assumimos 1000 promoções para este projecto.
- Dimensão BD_CANAISVENDA, 5 canais de vendas.
- Tamanho Médio de um Registo de Facto (TMRF), 8 campos * 4 Bytes = 32 Bytes.
- Percentagem de Espaço Ocupado pelas Dimensões em Relação aos Factos (PEODRF), 5%.

Tendo em conta que temos 8 campos por registo na tabela de factos obtemos a seguinte expressão:

$$(BD_TEMPO * BD_CLIENTE * BD_PRODUTOS * BD_PROMOCOES * BD_CANAISVENDA * TMRF) * PEODRF = 3.508.758.544.921,88 \text{ MB}$$

Este valor aplica-ria-se numa MOLAP em que teríamos todos os dados, nulos ou não, presente no sistema. No nosso caso, uma ROLAP com apenas a informação existente, temos que fazer a previsão de outro modo.

Olhando para os dados presentes na tabela SALES, conseguimos concluir que o numero de vendas por ano ronda os 350000 registos. Alterando a expressão anterior para considerar este facto temos:

$$\text{Num. Médio Vendas Ano} * \text{Tempo Vida Datamart} * \text{PEODRF} = 3.675.000 \text{ MB}$$

2.6. Informação extra

No lugar de ID numérico na chave do BD_TEMPO, optamos por definir uma chave com o formato ano + mês + dia para possíveis partições.

3. Instalação e configuração do software Oracle

3.1. Software Utilizado

Para realizar este projecto foram utilizados produtos:

- Oracle9i Enterprise Edition
- Oracle9i Development Suite
- SYBASE PowerDesigner 6
- Indus-Soft WinSQL 3.8
- Diversos

3.2. Configuração da base de dados para o Data Mart

3.2.1. Criação dos utilizadores e definição dos seus privilégios

Para a realização deste projecto foram criadas três contas no Oracle, duas para utilização dos elementos participantes e uma para alojamento dos objectos da datamart.

Conta	Grupo
adm1	Bda
adm2	Bda
BD	

Tabela 1

3.2.2. Configuração do espaço utilizado

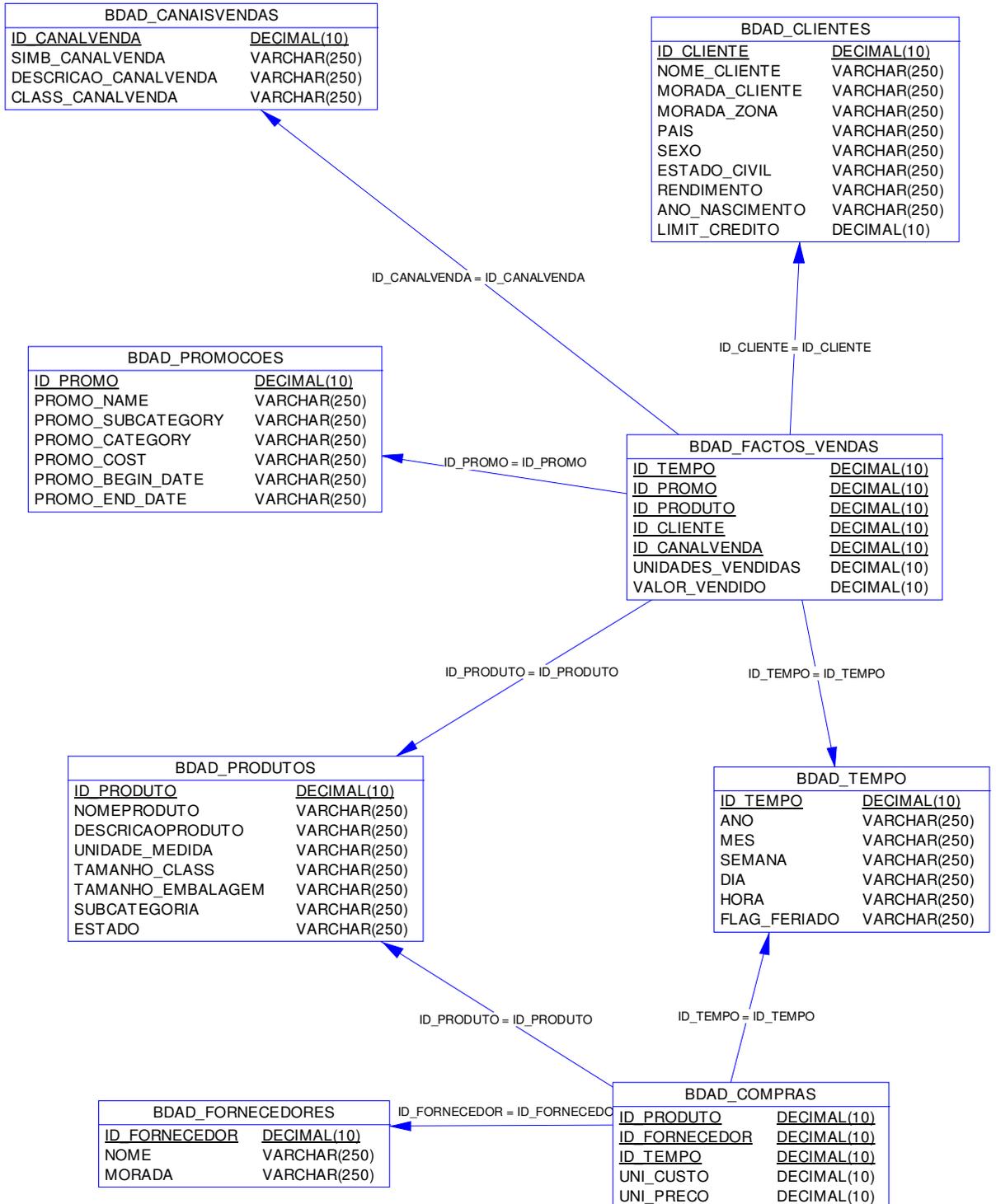
Tendo em conta o tamanho previsto para a datamart, foi logo criado no Oracle as dimensões necessárias.

Conta	Tam. Inicial	AutoExtend
adm1	150MB	Sim, com log
adm2	150MB	Sim, com log
BD	4.000.000MB	Sim, com log

Tabela 2

3.2.3.Criação das tabelas de dados

Foram criadas seis tabelas. A estrutura das mesmas estão situadas no



ANEXO 3 – Scripts de criação das tabelas do Datamart.

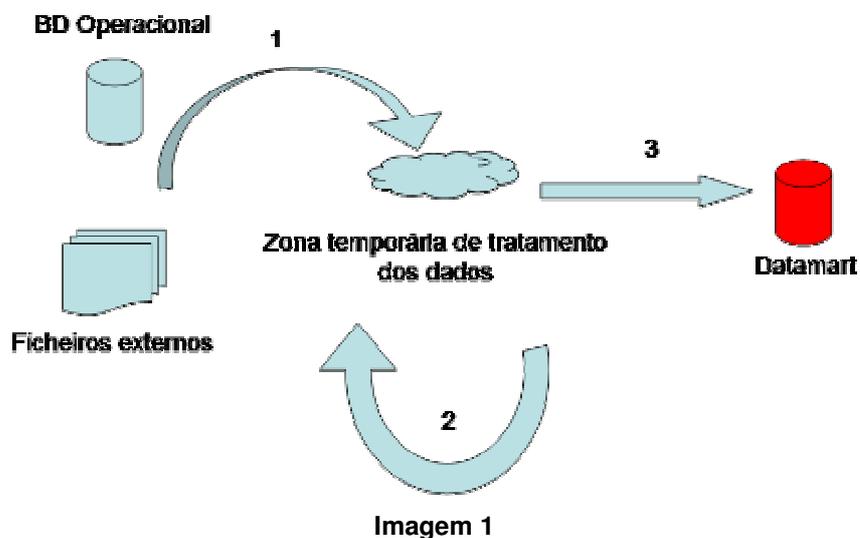
- BD_TEMPO
- BD_CLIENTE
- BD_PRODUTOS
- BD_PROMOCOES
- BD_CANAISVENDA
- BD_FACTOS_VENDAS

4. Extract Transform and Load

4.1. Definição do processo de importação

Como qualquer processo de alimentação de uma base de dados, temos três passos:

1. Selecção dos dados origem e import para zona temporária.
2. Limpeza e tratamento destes.
3. Importação da zona temporária para tabelas destino.



4.2. Importação dos dados Operacionais

Na tarefa de importação dos dados originais temos várias opções:

- Criar uma ligação entre a bd de produção e a datamart
- Fazer um dump da bd original e o devido load para a zona temporária.
- Usar uma ferramenta de transformação como o Warehouse Builder.

Optamos pelo método de dump e load dos dados que pretendemos inserir na datamart.

4.3. Importação dos dados externos

Existem ficheiros Excel que, apesar de não estar no sistema OLTP, fazem parte do negócio. Logo é necessário importar essa informação. A informação disponibilizada é promoções e canais de vendas.

O processo para transformação o conteúdo desses ficheiros é trabalhoso. Uma hipótese seria de criar uma ligação ODBC pelo driver fornecido pela Microsoft. Outra seria gravar o ficheiro como CSV (Comma Sep. Values) e transformar o mesmo em comandos SQL. Optamos pela segunda abordagem.

Temos primeiro alterar os ficheiros para facilitar a importação dos dados. De seguida temos que converter os ficheiros xls para csv. Por último, quando temos um ficheiro de texto com os dados convertidos para SQL introduzimos estes para uma tabela temporária. No final carregamos os dados para a Datamart.

No ANEXO 4 – Extração dos dados externos, estão exemplos desse processo.

4.4. Geração da dimensão Tempo

Existem várias maneiras de gerar a informação necessária para a dimensão tempo. Optamos por criar um pequeno programa em visual basic que gera essa informação. Assim tenhamos controlo sobre os dados necessários como por exemplo, dias de semana em português, definição dos feriados entre outros.

Essa abordagem, apesar de trabalhosa, dá-nos flexibilidade na gestão dos dados.

5. Optimizaç o

5.1. Criaç o de  ndices

Um dos m todos de optimizaç o das datamart e data Warehouse   a utilizaç o de  ndices bitmap join no lugar de  ndices btree que   o convencional numa base de dados relacional. No ANEXO 7 – Criaç o dos  ndices Bitmap Join est o descritos os  ndices utilizados.

5.2. Optimizaç o do Oracle 9i

Foi alterado tamb m a configuraç o da base de dados para optimizaç o nas queries. Os valores alterados foram os seguintes:

```
BITMAP_MERGE_AREA_SIZE=15728640  
CREATE_BITMAP_AREA_SIZE=15728640  
SORT_AREA_SIZE=15728640  
HASH_AREA_SIZE=15728640  
HASH_JOIN_ENABLED=TRUE  
DB_FILE_MULTIBLOCK_READ_COUNT=64
```

6. Discoverer Administrator

6.1. Criação do EUL

Neste passo criamos um EUL com o esquema de dados do utilizador BD.

De seguida foi criada uma Business Area com base nas tabelas do utilizador BD. Foram importadas as seguintes tabelas:

- BD_CANAISVENDAS
- BD_CLIENTES
- BD_COMPRAS
- BD_FACTOS_VENDAS
- BD_FORNECEDORES
- BD_PRODUTOS
- BD_PROMOCOES
- BD_TEMPO

As relações criadas foram com base nas chaves primárias e forasteiras.

6.2. Definição de privilégios

A definição de privilégios foi executada em dois passos. Primeiro foi dado acesso aos utilizadores amuga, pamaro e xpto.

De seguida foram atribuídos privilégios de administração aos utilizadores amuga e pamaro. O utilizador xpto ficou com apenas os privilégios de edição do utilizador.

6.3. Preparação da Business Area

Nesta fase escondemos todas as chaves primárias e corrigimos todos os nomes para terem um aspecto legível.

7. Discoverer Desktop

Junto com este relatório segue em anexo os cadernos criados.

7.1. CAD_MELHORES_CLIENTES

Este cadernos foi criado com base numa matriz com os valores NomeCliente e um campo calculado `SUM(VALOR_VENDIDO)`.

```
SELECT BD_CLIENTES.NOME_CLIENTE, SUM(BD_FACTOS_VENDAS.VALOR_VENDIDA)
FROM BD.BD_CLIENTES BD_CLIENTES, BD.BD_FACTOS_VENDAS BD_FACTOS_VENDAS
WHERE ( ( BD_CLIENTES.ID_CLIENTE = BD_FACTOS_VENDAS.ID_CLIENTE ) )
GROUP BY BD_CLIENTES.NOME_CLIENTE;
```

7.2. CAD_LOCAL_MELHORES_CLI

Este cadernos foi criado com base numa matriz com os valores Pais, ZonaCliente e um campo calculado `SUM(VALOR_VENDIDO)`.

```
SELECT BD_CLIENTES.MORADA_ZONA, BD_CLIENTES.PAIS, BD_TEMPO.ANO,
SUM(BD_FACTOS_VENDAS.VALOR_VENDIDO)
FROM BD.BD_CLIENTES BD_CLIENTES, BD.BD_FACTOS_VENDAS BD_FACTOS_VENDAS,
BD.BD_TEMPO BD_TEMPO
WHERE ( ( BD_CLIENTES.ID_CLIENTE = BD_FACTOS_VENDAS.ID_CLIENTE )
AND ( BD_TEMPO.ID_TEMPO = BD_FACTOS_VENDAS.ID_TEMPO ) )
GROUP BY BD_CLIENTES.MORADA_ZONA, BD_CLIENTES.PAIS, BD_TEMPO.ANO;
```

7.3. CAT_PROD MAIS VENDIDOS

Este cadernos foi criado com base numa matriz com os valores nomeprodutos e ano com um campo calculado `SUM(VALOR_VENDIDO)`.

```
SELECT BD_PRODUTOS.NOMEPRODUTO, BD_TEMPO.ANO, SUM(BD_FACTOS_VENDAS.UNIDADES_VENDIDAS)
FROM BD.BD_FACTOS_VENDAS BD_FACTOS_VENDAS, BD.BD_PRODUTOS BD_PRODUTOS, BD.BD_TEMPO
BD_TEMPO
WHERE ( ( BD_PRODUTOS.ID_PRODUTO = BD_FACTOS_VENDAS.ID_PRODUTO ) AND ( BD_TEMPO.ID_TEMPO
= BD_FACTOS_VENDAS.ID_TEMPO ) )
GROUP BY BD_PRODUTOS.NOMEPRODUTO, BD_TEMPO.ANO;
```

7.4. CAD_EVOLUCAO_VENDAS

Este cadernos foi criado com base numa matriz com os valores nomeprodutos, ano e mês com um campo calculado `SUM(VALOR_VENDIDO)`.

```
SELECT BD_PRODUTOS.NOMEPRODUTO, BD_TEMPO.ANO, BD_TEMPO.MES,
SUM(BD_FACTOS_VENDAS.VALOR_VENDIDO)
FROM BD.BD_FACTOS_VENDAS BD_FACTOS_VENDAS, BD.BD_PRODUTOS BD_PRODUTOS, BD.BD_TEMPO
BD_TEMPO
```

```
WHERE ( ( BD_PRODUTOS.ID_PRODUTO = BD_FACTOS_VENDAS.ID_PRODUTO ) AND ( BD_TEMPO.ID_TEMPO
= BD_FACTOS_VENDAS.ID_TEMPO ) )
GROUP BY BD_PRODUTOS.NOMEPRODUTO, BD_TEMPO.ANO, BD_TEMPO.MES;
```

7.5. CAD_CAT_PROD MAIS VENDIDOS

Caderno com matriz baseado nas categorias dos produtos por ano. Foi criado um campo `SUM(Unidades Vendidas)`.

```
SELECT BD_PRODUTOS.SUBCATEGORIA, BD_TEMPO.ANO,
SUM(BD_FACTOS_VENDAS.UNIDADES_VENDIDAS)
FROM BD.BD_FACTOS_VENDAS BD_FACTOS_VENDAS,
BD.BD_PRODUTOS BD_PRODUTOS,
BD.BD_TEMPO BD_TEMPO
WHERE ( ( BD_PRODUTOS.ID_PRODUTO = BD_FACTOS_VENDAS.ID_PRODUTO )
AND ( BD_TEMPO.ID_TEMPO = BD_FACTOS_VENDAS.ID_TEMPO ) )
GROUP BY BD_PRODUTOS.SUBCATEGORIA, BD_TEMPO.ANO;
```

8. Resumo e experiencia adquirida

Este projecto foi muito interessante e permitiu a aquisição de muitos conhecimentos na área do Data Warehousing.

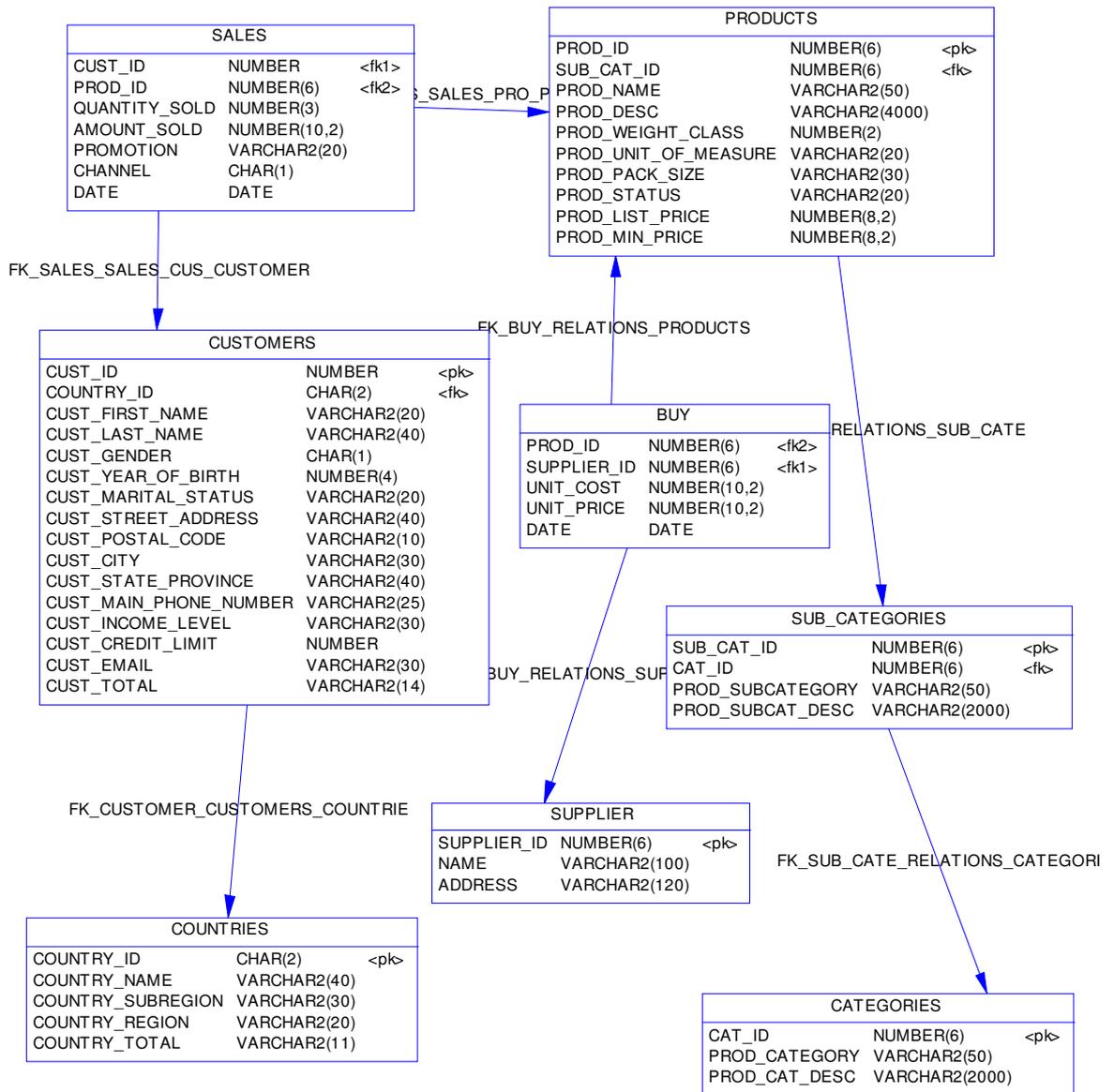
Conseguimos apreender que o desenho e implementação de uma datamart não se rege pelas mesmas regras que uma base de dados operacional, sendo que, em alguns casos, são incompatíveis. Basicamente todos os “truques” apreendidos anteriormente são para esquecer. Isto é algo de difícil de entender. Tudo que se aplica numa OLTP é para ser esquecido numa OLAP.

Tivemos muitos problemas com o software da Oracle. Os mais relevantes foram os wizards e os programas de instalação que acompanham o Oracle 9i. Por exemplo, o setup de instalação do Oracle 9i não detecta nem avisa que o meio de suporte aonde estão os ficheiros de instalação está corrompido, congelando apenas. O wizard de criação de uma base de dados não funcionava na criação de uma base de dados exemplo para Data Warehouse. Poderíamos enumerar aqui uma lista enorme de problemas com o software utilizado.

Ficamos agradavelmente impressionados com as capacidades do Oracle Discoverer e as suas potencialidades.

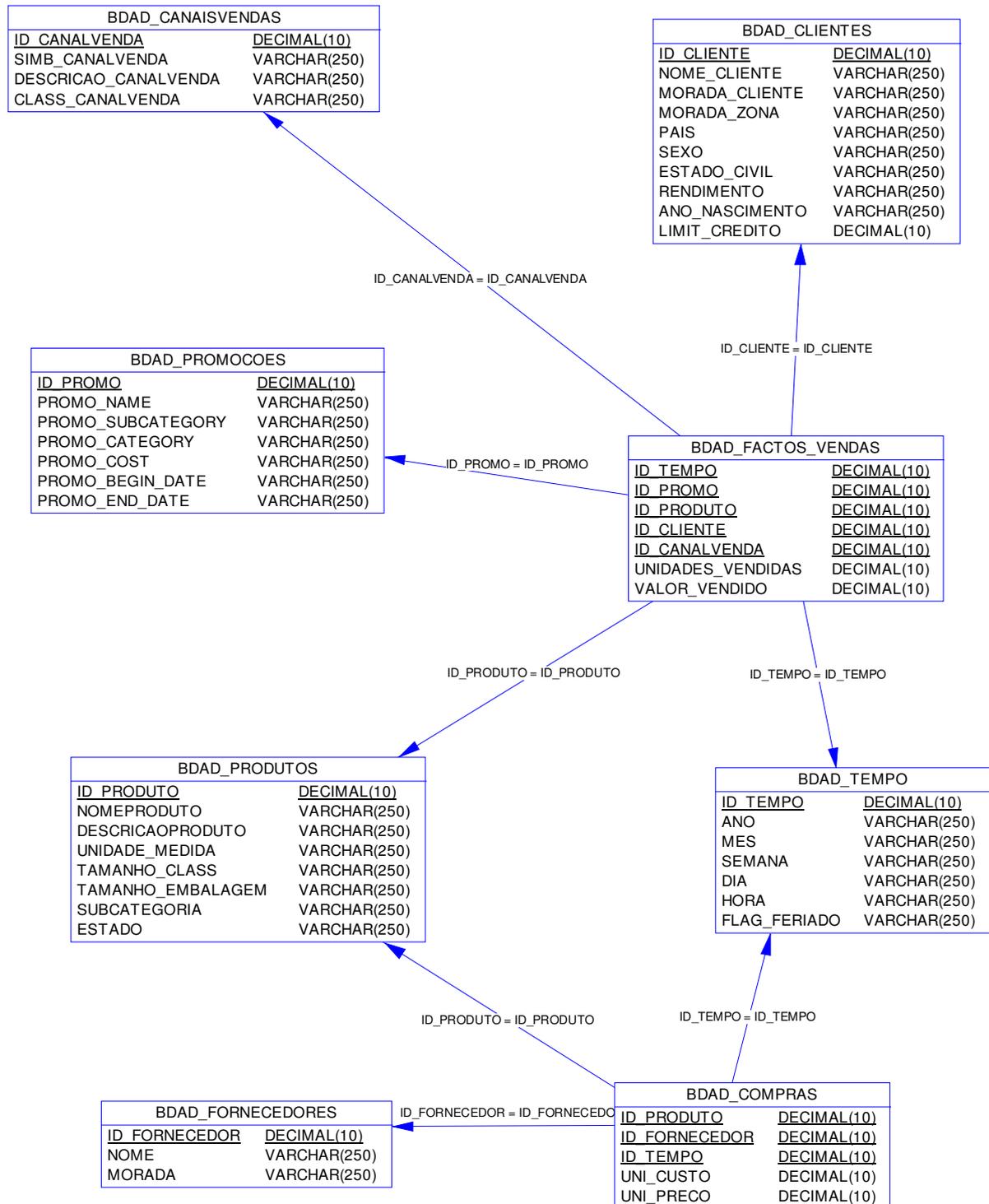
ANEXO 1 – Modelo Operacional

Esquema conceptual da base de dados operacional



ANEXO 2 – Modelo Estrela

Esquema conceitual da Data Mart



ANEXO 3 – Scripts de criação das tabelas do Datamart

```
create table BD_CANAISVENDAS
(
  ID_CANALVENDA          DECIMAL(10)          not null,
  SIMB_CANALVENDA        VARCHAR2(250)        not null,
  DESCRICAO_CANALVENDA   VARCHAR2(250)        not null,
  CLASS_CANALVENDA        VARCHAR2(250)        not null,
  primary key (ID_CANALVENDA)
);

create unique index BD_CANAISVENDAS_PK on BD_CANAISVENDAS (ID_CANALVENDA asc);

create table BD_CLIENTES
(
  ID_CLIENTE             DECIMAL(10)          not null,
  NOME_CLIENTE           VARCHAR2(250)        ,
  MORADA_CLIENTE         VARCHAR2(250)        ,
  MORADA_ZONA            VARCHAR2(250)        ,
  PAIS                   VARCHAR2(250)        ,
  SEXO                   VARCHAR2(250)        ,
  ESTADO_CIVIL           VARCHAR2(250)        ,
  RENDIMENTO             VARCHAR2(250)        ,
  DATA_NASCIMENTO       VARCHAR2(250)        ,
  TELEFONE               VARCHAR2(250)        ,
  LIMIT_CREDITO          DECIMAL(10)          ,
  primary key (ID_CLIENTE)
);

create unique index BD_CLIENTES_PK on BD_CLIENTES (ID_CLIENTE asc);

create table BD_PRODUTOS
(
  ID_PRODUTO             DECIMAL(10)          not null,
  NOMEPRODUTO            VARCHAR2(250)        ,
  DESCRICAOPRODUTO       VARCHAR2(250)        ,
  UNIDADE_MEDIDA         VARCHAR2(250)        ,
  TAMANHO_CLASS          VARCHAR2(250)        ,
  TAMANHO_EMBALAGEM     VARCHAR2(250)        ,
  SUBCATEGORIA           VARCHAR2(250)        ,
  FORNECEDOR_NOME        VARCHAR2(250)        ,
  FORNECEDOR_MORADA     VARCHAR2(250)        ,
  ULT_PRECOCUSTO         VARCHAR2(250)        ,
  ULT_PRECOUNIT          VARCHAR2(250)        ,
  ULT_PRECOACTUALIZACAO VARCHAR2(250)        ,
  primary key (ID_PRODUTO)
);

create unique index BD_PRODUTO_PK on BD_PRODUTOS (ID_PRODUTO asc);

create table BD_PROMOCOES
(
  ID_PROMO               DECIMAL(10)          not null,
  PROMO_NAME             VARCHAR2(250)        ,
  PROMO_SUBCATEGORY      VARCHAR2(250)        ,
  PROMO_CATEGORY         VARCHAR2(250)        ,
  PROMO_COST             VARCHAR2(250)        ,
  PROMO_BEGIN_DATE       VARCHAR2(250)        ,
  PROMO_END_DATE         VARCHAR2(250)        ,
  primary key (ID_PROMO)
);

create unique index BD_PROMOCOES_PK on BD_PROMOCOES (ID_PROMO asc);

create table BD_TEMPO
(
  ID_TEMPO               DECIMAL(10)          not null,
  ANO                    VARCHAR2(250)        ,
  MES                    VARCHAR2(250)        ,
```

```

        SEMANA                VARCHAR2 (250)                ,
        DIA                   VARCHAR2 (250)                ,
        HORA                  VARCHAR2 (250)                ,
        FLAG_FERIADO          VARCHAR2 (250)                ,
        primary key (ID_TEMPO)
    );

create unique index BD_TEMPO_PK on BD_TEMPO (ID_TEMPO asc);

create table BD_FACTOS_VENDAS
(
    ID_TEMPO                DECIMAL (10)                not null,
    ID_PROMO                DECIMAL (10)                not null,
    ID_PRODUTO              DECIMAL (10)                not null,
    ID_CLIENTE              DECIMAL (10)                not null,
    ID_CANALVENDA           DECIMAL (10)                not null,
    UNIDADES_VENDIDAS       DECIMAL (10)                ,
    VALOR_VENDIDA           DECIMAL (10)                ,
    primary key (ID_TEMPO, ID_PROMO, ID_PRODUTO, ID_CLIENTE, ID_CANALVENDA),
    foreign key (ID_CANALVENDA)
        references BD_CANAISVENDAS (ID_CANALVENDA),
    foreign key (ID_CLIENTE)
        references BD_CLIENTES (ID_CLIENTE),
    foreign key (ID_PRODUTO)
        references BD_PRODUTOS (ID_PRODUTO),
    foreign key (ID_PROMO)
        references BD_PROMOCOES (ID_PROMO),
    foreign key (ID_TEMPO)
        references BD_TEMPO (ID_TEMPO)
);

create unique index BD_FACTOS_VENDAS_PK on BD_FACTOS_VENDAS (ID_TEMPO asc, ID_PROMO asc,
ID_PRODUTO asc, ID_CLIENTE asc, ID_CANALVENDA asc);

```

ANEXO 4 – Extração dos dados externos

Exemplo de resultado da transformação do ficheiro Excel de promoções para csv pronto para carregamento.

```
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (1, 'promotion name# 1', 'downtown
billboard', 'post', 77200, '15-09-1998 0:00', '15-11-1998 0:00');
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (2, 'promotion name# 2', 'hospital
flyer', 'flyer', 47100, '13-04-1999 0:00', '13-07-1999 0:00');
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (3, 'promotion name# 3', 'coupon
news', 'newspaper', 23900, '25-08-2000 0:00', '25-09-2000 0:00');
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (4, 'promotion name#
4', 'manufacture rebate news', 'newspaper', 8700, '18-11-1998 0:00', '18-01-1999 0:00');
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (5, 'promotion name# 5', 'TV
program sponsorship', 'TV', 5700, '17-03-1999 0:00', '17-06-1999 0:00');
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (6, 'promotion name# 6', 'ad
news', 'newspaper', 76800, '08-05-2000 0:00', '08-06-2000 0:00');
```

Registo para definição de factos sem promoção.

```
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (0, 'NO PROMOTION', 'NO
PROMOTION', 'NO PROMOTION', 0, '01-01-9999 0:00', '01-01-9999 0:00');
```

Registo em falta.

```
INSERT INTO BD_PROMOCOES (ID_PROMO, PROMO_NAME, PROMO_SUBCATEGORY, PROMO_CATEGORY,
PROMO_COST, PROMO_BEGIN_DATE, PROMO_END_DATE) VALUES (420, 'promotion name#
420', 'UNKNOW', 'UNKNOW', 0, 'UNKNOW', 'UNKNOW');
```

Exemplo de resultado da transformação do ficheiro Excel de canais de vendas para csv pronto para carregamento.

```
INSERT INTO BD_CANAISVENDAS (ID_CANALVENDA, SIMB_CANALVENDA, DESCRICAO_CANALVENDA,
CLASS_CANALVENDA) VALUES (1, 'S', 'Direct Sales', 'Direct');
INSERT INTO BD_CANAISVENDAS (ID_CANALVENDA, SIMB_CANALVENDA, DESCRICAO_CANALVENDA,
CLASS_CANALVENDA) VALUES (2, 'T', 'Tele Sales', 'Direct');
INSERT INTO BD_CANAISVENDAS (ID_CANALVENDA, SIMB_CANALVENDA, DESCRICAO_CANALVENDA,
CLASS_CANALVENDA) VALUES (3, 'C', 'Catalog', 'Indirect');
INSERT INTO BD_CANAISVENDAS (ID_CANALVENDA, SIMB_CANALVENDA, DESCRICAO_CANALVENDA,
CLASS_CANALVENDA) VALUES (4, 'I', 'Internet', 'Indirect');
INSERT INTO BD_CANAISVENDAS (ID_CANALVENDA, SIMB_CANALVENDA, DESCRICAO_CANALVENDA,
CLASS_CANALVENDA) VALUES (5, 'P', 'Partners', 'Others');
```

Registo para definição de factos sem canal de venda.

```
INSERT INTO BD_CANAISVENDAS (ID_CANALVENDA, SIMB_CANALVENDA, DESCRICAO_CANALVENDA,
CLASS_CANALVENDA) VALUES (0, 'O', 'Others', 'Others');
```

ANEXO 5 – Geração da informação de TEMPO

Exemplo de dados de tempo pronto para carregamento.

```
insert into BD_TEMPO values( 19990101, '1999', '01', 'sexta-feira', '01', '00', ' ');
insert into BD_TEMPO values( 19990102, '1999', '01', 'sábado', '02', '00', ' ');
insert into BD_TEMPO values( 19990103, '1999', '01', 'domingo', '03', '00', ' ');
insert into BD_TEMPO values( 19990104, '1999', '01', 'segunda-feira', '04', '00', ' ');
insert into BD_TEMPO values( 19990105, '1999', '01', 'terça-feira', '05', '00', ' ');
insert into BD_TEMPO values( 19990106, '1999', '01', 'quarta-feira', '06', '00', ' ');
insert into BD_TEMPO values( 19990107, '1999', '01', 'quinta-feira', '07', '00', ' ');
insert into BD_TEMPO values( 19990108, '1999', '01', 'sexta-feira', '08', '00', ' ');
insert into BD_TEMPO values( 19990109, '1999', '01', 'sábado', '09', '00', ' ');
```

ANEXO 6 – Transformação dos dados de produção

Obtenção dos dados dos clientes.

```
--Criação do temp_clientes
drop table TEMP_CLIENTES

CREATE TABLE TEMP_CLIENTES AS
select
CUST_ID as ID_CLIENTE,
trim(CUST_FIRST_NAME || ' ' || CUST_LAST_NAME) as NOME_CLIENTE,
trim(CUST_STREET_ADDRESS || ' ' || CUST_POSTAL_CODE || ' ' || CUST_CITY ) as
MORADA_CLIENTE,
trim(CUST_STATE_PROVINCE ) as MORADA_ZONA,
trim(t2.COUNTRY_NAME) as PAIS,
trim(CUST_GENDER) as SEXO,
CUST_MARITAL_STATUS as ESTADO_CIVIL,
CUST_INCOME_LEVEL as RENDIMENTOS,
CUST_YEAR_OF_BIRTH as ANO_NASCIMENTO,
CUST_CREDIT_LIMIT as LIMIT_CREDITO
from amuga.customers t1, amuga.countries t2
where t1.COUNTRY_ID = t2.COUNTRY_ID
and 1 = 0;

--Alimentação
delete from TEMP_CLIENTES

insert into TEMP_CLIENTES
select
CUST_ID as ID_CLIENTE,
trim(CUST_FIRST_NAME || ' ' || CUST_LAST_NAME) as NOME_CLIENTE,
trim(CUST_STREET_ADDRESS || ' ' || CUST_POSTAL_CODE || ' ' || CUST_CITY ) as
MORADA_CLIENTE,
trim(CUST_STATE_PROVINCE ) as MORADA_ZONA,
trim(t2.COUNTRY_NAME) as PAIS,
trim(CUST_GENDER) as SEXO,
CUST_MARITAL_STATUS as ESTADO_CIVIL,
CUST_INCOME_LEVEL as RENDIMENTOS,
CUST_YEAR_OF_BIRTH as ANO_NASCIMENTO,
CUST_CREDIT_LIMIT as LIMIT_CREDITO
from amuga.customers t1, amuga.countries t2
where t1.COUNTRY_ID = t2.COUNTRY_ID;
--Tramamento
update temp_clientes
set ESTADO_CIVIL = 'Desconhecido'
where ESTADO_CIVIL is null;

--Alimentação da dm
insert into BD_CLIENTES
select * from temp_clientes;
```

Obtenção dos dados dos produtos.

```
drop table TEMP_produtos;  
  
CREATE TABLE TEMP_produtos AS  
select  
t1.PROD_ID as ID_PRODUTO,  
t1.PROD_NAME as NOME_PRODUTO,  
t1.PROD_DESC as DESCRICAO_PRODUTO,  
t1.PROD_UNIT_OF_MEASURE as UNIDADE_MEDIDA,  
t1.PROD_WEIGHT_CLASS as TAMANHO_CLASS,  
t1.PROD_PACK_SIZE as TAMANHO_EMBALAGEM,  
t2.prod_subcat_desc as SUBCATEGORIA,  
t1.PROD_STATUS as ESTADO  
  
from amuga.products t1, amuga.sub_categories t2  
where t1.sub_cat_id = t2.sub_cat_id  
  
and 1=0;  
  
delete from TEMP_produtos  
  
insert into TEMP_produtos  
select  
t1.PROD_ID as ID_PRODUTO,  
t1.PROD_NAME as NOME_PRODUTO,  
t1.PROD_DESC as DESCRICAO_PRODUTO,  
t1.PROD_UNIT_OF_MEASURE as UNIDADE_MEDIDA,  
t1.PROD_WEIGHT_CLASS as TAMANHO_CLASS,  
t1.PROD_PACK_SIZE as TAMANHO_EMBALAGEM,  
t2.prod_subcat_desc as SUBCATEGORIA,  
t1.PROD_STATUS as ESTADO  
  
from amuga.products t1, amuga.sub_categories t2  
where t1.sub_cat_id = t2.sub_cat_id;  
  
insert into BD_produtos  
select * from temp_produtos;
```

Obtenção dos dados dos fornecedores.

```
drop table TEMP_FORNECEDORES  
  
CREATE TABLE TEMP_FORNECEDORES AS  
select * from amuga.supplier  
where 1=0;  
  
insert into TEMP_FORNECEDORES  
select * from amuga.supplier  
  
insert into BD_fornecedores  
select * from TEMP_FORNECEDORES;  
select * from TEMP_FORNECEDORES;
```

Obtenção dos dados das compras.

```
drop table TEMP_COMPRAS;  
create table TEMP_COMPRAS as
```

```

SELECT prod_id as ID_PRODUTO, supplier_id as ID_FORNECEDOR, TO_CHAR(BUY_DATE,
'YYYYMMDD') AS ID_TEMPO,
t1.UNIT_COST as UNI_CUSTO,
t1.UNIT_PRICE as UNI_PRECO
from buy t1
where 1=0

```

```

insert into TEMP_COMPRAS
SELECT prod_id as ID_PRODUTO, supplier_id as ID_FORNECEDOR, TO_CHAR(BUY_DATE,
'YYYYMMDD') AS ID_TEMPO,
t1.UNIT_COST as UNI_CUSTO,
t1.UNIT_PRICE as UNI_PRECO
from buy t1;

```

Obtenção dos dados das vendas.

```
drop table TEMP_VENDAS;
```

```

create table TEMP_VENDAS as
SELECT
TO_CHAR(SALE_DATE, 'YYYYMMDD') AS ID_TEMPO,
999999999 as id_promo,
prod_id as ID_PRODUTO,
cust_id as ID_CLIENTE,
999999999 as ID_CANALVENDA,
t1.quantity_sold as "UNIDADES_VENDIDAS",
t1.amount_sold as "VALOR_VENDIDO"
from amuga.sales t1
where 1=0;

```

```

delete from TEMP_VENDAS;
insert into TEMP_VENDAS
SELECT
TO_CHAR(SALE_DATE, 'YYYYMMDD') AS ID_TEMPO,
case
  when PROMOTION = 'NO PROMOTION' then 0
  else To_NUMBER(SUBSTR(promotion, 16 ))
end
as ID_PROMO,
prod_id as ID_PRODUTO,
cust_id as ID_CLIENTE,
case
  when channel = 'S' then 1
  when channel = 'T' then 2
  when channel = 'C' then 3
  when channel = 'I' then 4
  when channel = 'P' then 5
  else 0
end as ID_CANALVENDA,
sum(t1.quantity_sold) as "UNIDADES_VENDIDAS",
sum(t1.amount_sold) as "VALOR_VENDIDO"
from amuga.sales t1
group by SALE_DATE,PROMOTION,prod_id,cust_id,channel;

```

```

insert into BD_factos_vendas
select * from TEMP_VENDAS
where id_tempo > 20000101

```

ANEXO 7 – Criação dos índices Bitmap Join

```
CREATE BITMAP INDEX BD_FACTOS_VENDAS_IDX_1 ON BD_FACTOS_VENDAS (BD_TEMPO.ID_TEMPO)
FROM BD_FACTOS_VENDAS , BD_TEMPO
WHERE BD_FACTOS_VENDAS.ID_TEMPO = BD_TEMPO.ID_TEMPO
```

```
CREATE BITMAP INDEX BD_FACTOS_VENDAS_IDX_2 ON BD_FACTOS_VENDAS (BD_CLIENTES.ID_CLIENTE)
FROM BD_FACTOS_VENDAS , BD_CLIENTES
WHERE BD_FACTOS_VENDAS.ID_CLIENTE = BD_CLIENTES.ID_CLIENTE
```

```
CREATE BITMAP INDEX BD_FACTOS_VENDAS_IDX_3 ON BD_FACTOS_VENDAS (BD_PRODUTOS.ID_PRODUTO)
FROM BD_FACTOS_VENDAS , BD_PRODUTOS
WHERE BD_FACTOS_VENDAS.ID_PRODUTO = BD_PRODUTOS.ID_PRODUTO
```

```
CREATE BITMAP INDEX BD_FACTOS_VENDAS_IDX_4 ON BD_FACTOS_VENDAS (BD_PROMOCOES.ID_PROMO)
FROM BD_FACTOS_VENDAS , BD_PROMOCOES
WHERE BD_FACTOS_VENDAS.ID_PROMO = BD_PROMOCOES.ID_PROMO
```

```
CREATE BITMAP INDEX BD_FACTOS_VENDAS_IDX_5 ON BD_FACTOS_VENDAS
(BD_CANALSVENDAS.ID_CANALVENDA)
FROM BD_FACTOS_VENDAS , BD_CANALSVENDAS
WHERE BD_FACTOS_VENDAS.ID_CANALVENDA = BD_CANALSVENDAS.ID_CANALVENDA
```

ANEXO 8 – Alguns comandos SQL utilizados

Criação de tabelas temporárias.

```
CREATE TABLE TEMP_CLIENTES AS  
select * from Tabela where 1=0;
```

Criação de vistas materializadas.

```
CREATE MATERIALIZED VIEW BD_CLIENTES_VIEW REFRESH WITH ROWID  
AS SELECT * FROM BD_CLIENTES;
```

Criação de vistas materializadas com auto refresh.

```
CREATE MATERIALIZED VIEW BD_CLIENTES_VIEW  
PCTFREE 5 PCTUSED 60  
TABLESPACE users  
STORAGE (INITIAL 50K NEXT 50K)  
REFRESH FAST NEXT sysdate + 7  
AS SELECT * FROM BD_CLIENTES;
```

ANEXO 9 – ScreenShots do software criados para o apoio.

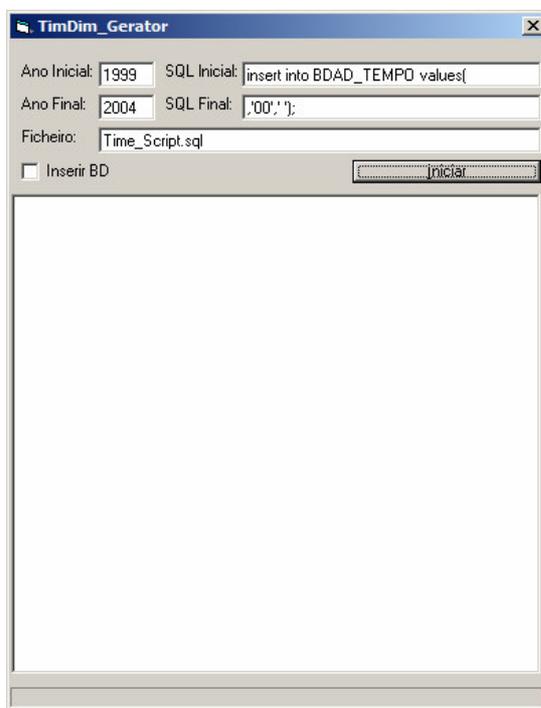


Imagem 2

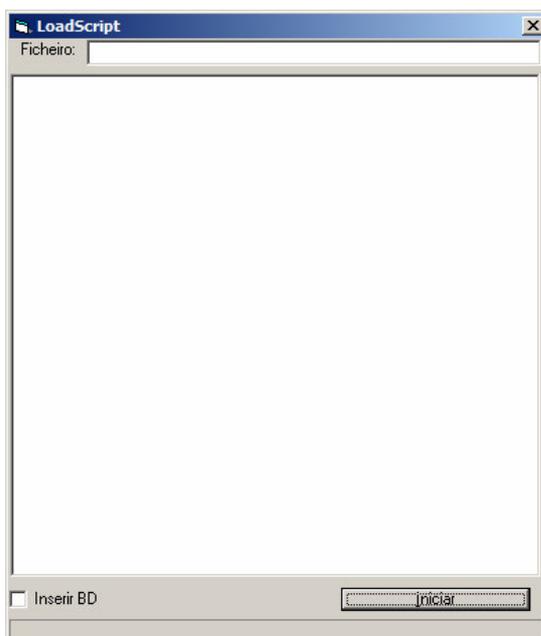


Imagem 3