Universidade do Minho
Escola de Engenharia

Universidade do Minho
Escola de Engenharia

**Mestrado em Engenharia Informática**
Análise de Dados

# Trabalho prático

**Grupo:**
Paulo Barros – PG38421
Sofia Silva – PG38426
Telmo Cardoso – PG38427

*Braga, 20 de janeiro de 2018*

# Índice

1. Introdução ..... 3
1. Enquadramento Teórico ..... 4
2. Modelo Dimensional e Físico ..... 8
3. Registo de alterações ..... 17
4. Passagem dos dados entre bases de dados ..... 24
5. Carregamento dos dados para o Data Warehouse ..... 40
6. Indicadores de Business Intelligence ..... 41

# 1. Introdução

Estamos num mundo em que cada vez há mais dados a circular pela internet tornando-se assim necessário possuirmos bases de dados que consigam suportar grandes quantidade de dados. As empresas por exemplo necessitam de armazenar informações das suas atividades de forma consolidada para que se possa realizar análises de grandes volumes de dados, obter bons relatórios e informações estratégicas para uma fácil tomada de decisões. Os *data warehouses* surgem com esse mesmo intuito, gerenciam o fluxo de informação das bases de dados corporativas e fontes de dados externas à empresa. Ele permite assegurar a qualidade dos dados, um acesso rápido, facilidade de uso, entre outras.

Este trabalho, proposto na unidade curricular de Análise de Dados, vem pôr em prática as competências adquiridas nas aulas práticas e teóricas. Iremos realizar uma análise, planeamento e implementação de um sistema de bases de dados multidimensionais(SBDMD). Para isso utilizaremos o MySQL e uma base de dados, cedida pelo docente, denominada por "*NorthWind*" que suporta a operacionalidade de uma empresa fictícia que se dedica ao comércio internacional de produtos alimentares. Será projetado e implementado um *data warehouse* para a *Northwind*, com base na informação contida na sua base de dados operacional. Procederemos ao seu povoamento inicial, contemplando as estruturas analíticas necessárias as seu refrescamento. A passagem das alterações feitas na base de dados de produção serão passadas para a área de retenção através de código C# implementado no Visual Studio. Após a implementação do *data warehouse* serão definidos e criados indicadores de *Business Intelligence*, a partir da base de dados implementada, através do Power BI.

# 1. Enquadramento Teórico

**O que é um datawarehouse?**

Um datawarehouse pode ser definido como uma coleção de dados orientada por assuntos, integrada, variável no tempo, que tem por objetivo dar suporte aos processos de tomada de decisão.

- Orientado por assunto

Um datawarehouse é orientado por assunto pois permite que os dados sejam organizados em torno de um processo de negócio. Estes dados permitem fazer a análise sobre determinados temas particulares tornando assim possível aos gestores tomarem decisões com uma base sólida no histórico de dados da empresa (longo prazo), em oposição ao que acontece com os sistemas operacionais, que apenas permitem a tomada de decisões operacionais (curto prazo).

- Integrado

O *data warehouse* é um sistema integrado pois é constituído pela integração de dados de vários sistemas. É fundamental garantir que exista uma consistência nos dados no nosso sistema.

- Variavel Tempo

O *data warehouse* refere-se a algum momento específico, ou seja não é atualizável, enquanto que os dados de produção são atualizados de acordo com a mudança de estado do objeto em estudo, refletindo, em geral, o estado do objeto no momento do acesso. Num datawarehouse é criada uma nova entrada sempre que exista a ocorrência de uma mudança.

- Não Volatil

Significa que no *data warehouse*, após a inserção inicial dos dados só são permitidas operações de consulta sobre esses dados. Depois dos dados serem integrados e transformados, estes são carregados num bloco para o Datawarehouse para que fiquem disponíveis para utilizadores

O grande objetivo do *data warehouse* é manter um histórico dos dados de negócio de uma empresa, recolhendo, organizando e criando padrões a partir desses mesmos dados.

No final do processamento dos dados são criados relatórios que servem para análise das diversas áreas da empresa.

## Vantagens e desvantagens de um datawarehouse

Vantagens

- Rapidez no acesso aos dados;
- Simplicidade na organização e representação dos dados;
- Qualidade dos dados;
- Acesso rápido a informação cuidada e tratada;
- Análise específica de uma área, facilitando a tomada de decisões;
- Segurança uma vez que é criada uma espécie de backup dos dados.

Desvantagens

- Alto custo operacional;
- Tempo de desenvolvimento do datawarehouse;
- Complexidade de desenvolvimento do datawarehouse.

## Datawarehouse e data mart

O Data Mart pode ser definido como um subconjunto de dados de um *data warehouse*. Normalmente são dados referentes a um assunto em especifico ou que focalizam numa ou mais áreas específicas.

| *Data Mart* | *Data warehouse* |
|---|---|
| **Orientado ao assunto.** | Contém múltiplos assuntos. |
| **Informação por departamento.** | Informação da empresa. |
| **Dados agregados (granularidade alta).** | Dados no nível mais elementar possível (baixa granularidade). |
| **Histórico parcial dos dados.** | Histórico total dos dados. |
| **Um processo de negócio** | Múltiplos processos de negócio |

Tabela 1 - Diferenças entre Data Mart e Data Warehouse

## Staging area

A *Staging Area* é uma localização temporária onde serão guardados os dados de uma tabela que sofreu alterações nos sistemas de origem. Desta forma, ao invés de acessar os dados diretamente da fonte, o processo de "transformação" do ETL pega nos dados da *Staging Area* para tratar e entregar os dados.

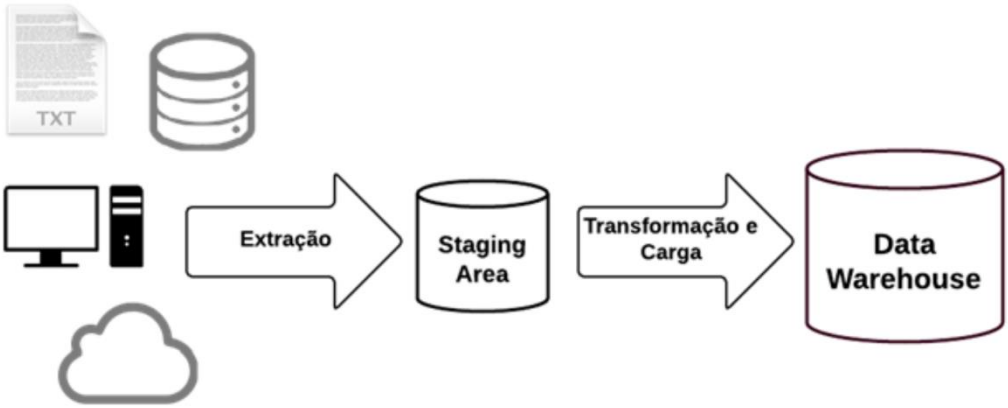



Figura 1 - Representação do percurso dos dados deste a sua leitura até ao armazenamento no DW

## Modelo Dimensional

A modelagem assenta sobretudo em dois pilares imprescindíveis: tabelas de factos e as tabelas de dimensão. Ambas são obrigatórias e possuem características complementares dentro de um *Data Warehouse*.

As tabelas de dimensões são os descritores dos dados oriundos das tabelas de factos. Possuiem o caráter qualitativo da informação e relacionamento de "um para muitos" com as tabelas de factos. São as tabelas de dimensão que permitem a visualização das informações e detalhes através de diversos aspetos e perspetivas.

A tabela de factos contém apenas as descrições métricas (valores numéricos). Possui o caráter quantitativo das informações descritivas armazenadas nas tabelas de dimensões. É onde estão armazenadas as ocorrências do negócio.

A modelagem dimensional possui três modelos base: o modelo em estrela (star schema), o modelo floco de neve (snow flake) e o modelo em constelação. Cada um com aplicabilidade diferente a depender da especificidade do problema.

O *esquema em estrela* caracteriza-se por uma tabela de factos ligada a um conjunto de tabelas de dimensão.

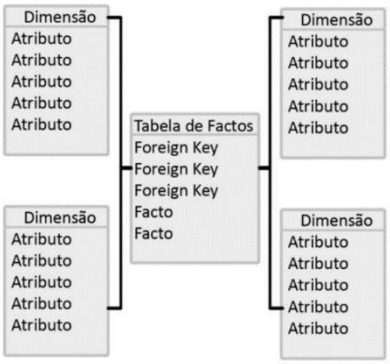



**Figura 2 - Representação do esquema em estrela**

O *esquema em floco de neve* é um esquema onde temos uma tabela de factos rodeada por várias tabelas de dimensões e onde estas se encontram normalizadas. O que se traduz no desaparecimento do esquema de estrela e toma lugar um esquema onde as várias dimensões se encontram ligadas entre si e a tabela de factos.

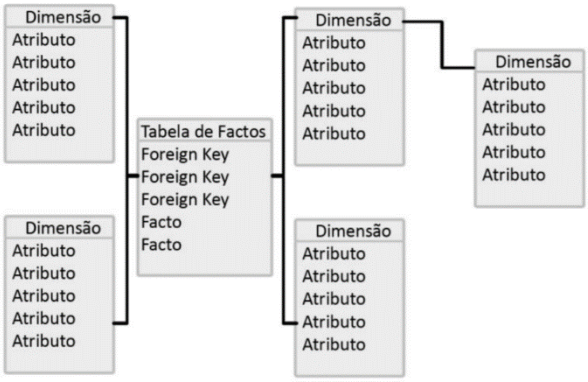



**Figura 3 - Representação do esquema em floco de neve**

Já o caso do *esquema em constelação* consiste num esquema onde existem pelo menos duas tabelas de factos e estas partilham pelo menos uma tabela de dimensão

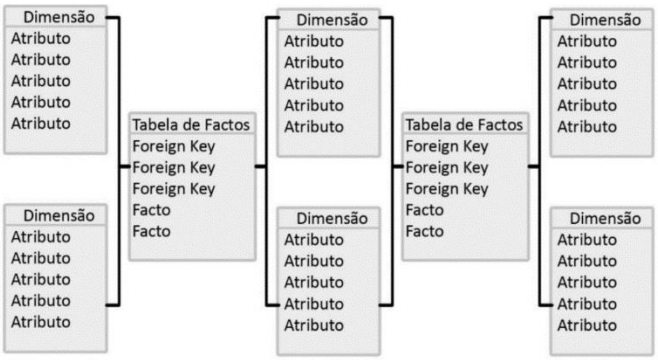



Figura 4 - Representação do esquema constelação

# 2. Modelo Dimensional e Físico

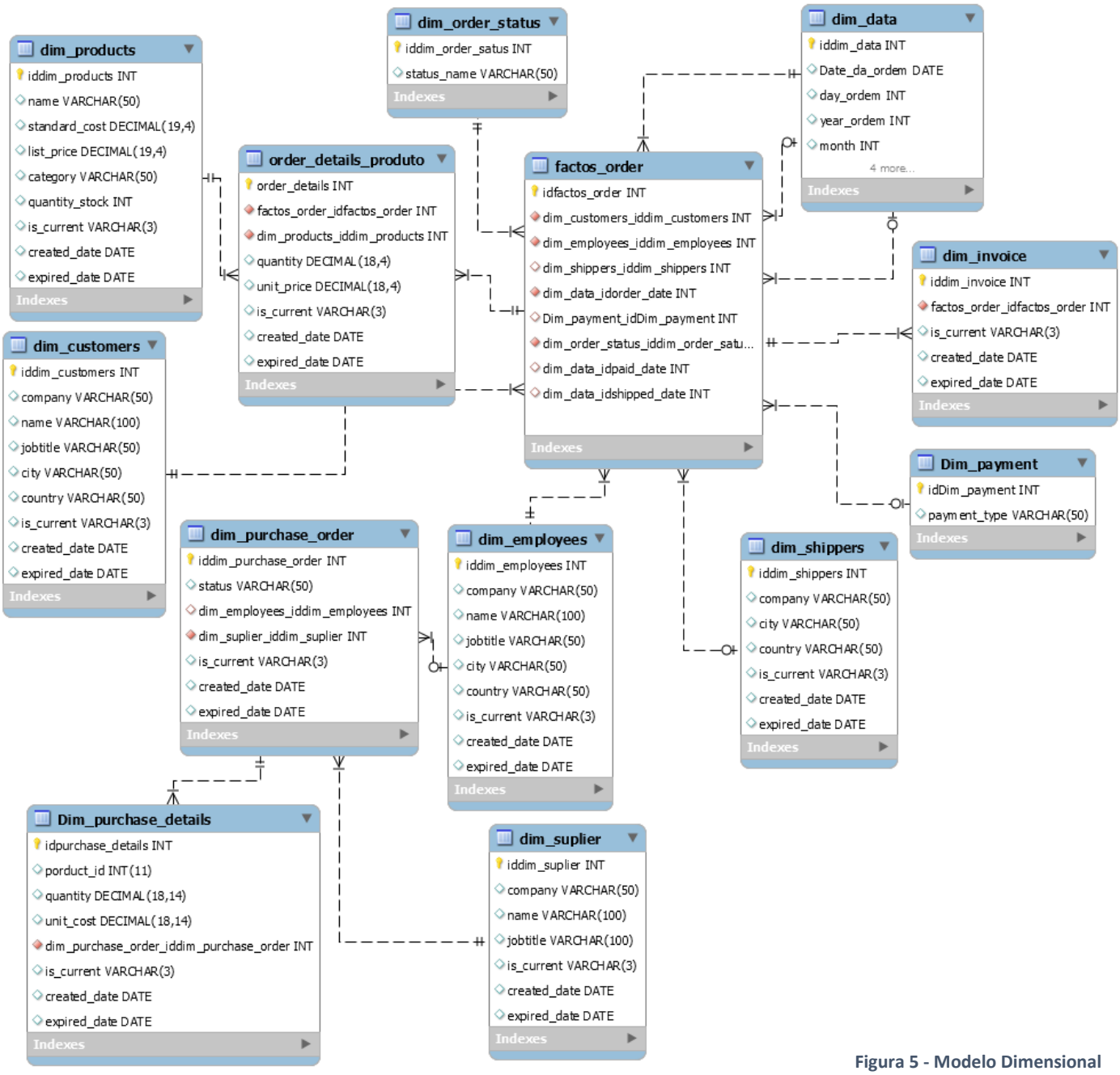



Figura 5 - Modelo Dimensional

Procedemos a criação de um modelo dimensional utilizando um esquema em floco de neve. Inicialmente tínhamos na ideia de avançar com um esquema em estrela, mas ao realizarmos uma análise dos dados verificamos que haviam algumas relações que tinham vários registos associados a uma ordem, e assim por diante, por isso optamos por esse esquema utilizado.

```sql
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `dw_northwind` DEFAULT CHARACTER SET utf8 ;
USE `dw_northwind` ;

-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_customers`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_customers` (
  `iddim_customers` INT NOT NULL AUTO_INCREMENT,
  `company` VARCHAR(50) NULL,
  `name` VARCHAR(100) NULL,
  `jobtitle` VARCHAR(50) NULL,
  `city` VARCHAR(50) NULL,
  `country` VARCHAR(50) NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`iddim_customers`),
  UNIQUE INDEX `iddim_customers_UNIQUE` (`iddim_customers` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_employees`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_employees` (
  `iddim_employees` INT NOT NULL AUTO_INCREMENT,
  `company` VARCHAR(50) NULL,
  `name` VARCHAR(100) NULL,
  `jobtitle` VARCHAR(50) NULL,
  `city` VARCHAR(50) NULL,
  `country` VARCHAR(50) NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`iddim_employees`),
  UNIQUE INDEX `iddim_employees_UNIQUE` (`iddim_employees` ASC) VISIBLE)
ENGINE = InnoDB;
```

```sql
-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_shippers`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_shippers` (
  `iddim_shippers` INT NOT NULL AUTO_INCREMENT,
  `company` VARCHAR(50) NULL,
  `city` VARCHAR(50) NULL,
  `country` VARCHAR(50) NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`iddim_shippers`),
  UNIQUE INDEX `iddim_shippers_UNIQUE` (`iddim_shippers` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_data`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_data` (
  `iddim_data` INT NOT NULL AUTO_INCREMENT,
  `Date_da_ordem` DATE NULL,
  `day_ordem` INT NULL,
  `year_ordem` INT NULL,
  `month` INT NULL,
  `day_of_week` VARCHAR(45) NULL,
  `week_of_year` INT NULL,
  `dia_util_ordem` VARCHAR(4) NULL,
  `feriado_ordem` VARCHAR(4) NULL,
  PRIMARY KEY (`iddim_data`),
  UNIQUE INDEX `iddim_data_UNIQUE` (`iddim_data` ASC) VISIBLE)
ENGINE = InnoDB;
```

```sql
-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_products`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_products` (
  `iddim_products` INT NOT NULL,
  `name` VARCHAR(50) NULL,
  `standard_cost` DECIMAL(19,4) NULL,
  `list_price` DECIMAL(19,4) NULL,
  `category` VARCHAR(50) NULL,
  `quantity_stock` INT NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`iddim_products`),
  UNIQUE INDEX `iddim_products_UNIQUE` (`iddim_products` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_purchase_order`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_purchase_order` (
  `iddim_purchase_order` INT NOT NULL,
  `status` VARCHAR(50) NULL,
  `dim_employees_iddim_employees` INT NULL,
  `dim_suplier_iddim_suplier` INT NOT NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`iddim_purchase_order`),
  UNIQUE INDEX `iddim_purchase_order_UNIQUE` (`iddim_purchase_order` ASC) VISIBLE,
  INDEX `fk_dim_purchase_order_dim_employees1_idx` (`dim_employees_iddim_employees` ASC) VISIBLE,
  INDEX `fk_dim_purchase_order_dim_suplier1_idx` (`dim_suplier_iddim_suplier` ASC) VISIBLE)
ENGINE = InnoDB;
```

```sql
-- -----------------------------------------------------
-- Table `dw_northwind`.`Dim_payment`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`Dim_payment` (
  `idDim_payment` INT NOT NULL AUTO_INCREMENT,
  `payment_type` VARCHAR(50) NULL,
  PRIMARY KEY (`idDim_payment`),
  UNIQUE INDEX `idDim_payment_UNIQUE` (`idDim_payment` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_order_status`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_order_status` (
  `iddim_order_satus` INT NOT NULL,
  `status_name` VARCHAR(50) NULL,
  PRIMARY KEY (`iddim_order_satus`),
  UNIQUE INDEX `idOrder_satus_UNIQUE` (`iddim_order_satus` ASC) VISIBLE)
ENGINE = InnoDB;
```

```sql
-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_invoice`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_invoice` (
  `iddim_invoice` INT NOT NULL,
  `factos_order_idfactos_order` INT NOT NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`iddim_invoice`),
  UNIQUE INDEX `iddim_invoice_UNIQUE` (`iddim_invoice` ASC) VISIBLE,
  INDEX `fk_dim_invoice_factos_order1_idx` (`factos_order_idfactos_order` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `dw_northwind`.`order_details_produto`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`order_details_produto` (
  `order_details` INT NOT NULL,
  `factos_order_idfactos_order` INT NOT NULL,
  `dim_products_iddim_products` INT NOT NULL,
  `quantity` DECIMAL(18,4) NULL,
  `unit_price` DECIMAL(18,4) NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  INDEX `fk_table1_factos_order1_idx` (`factos_order_idfactos_order` ASC) VISIBLE,
  INDEX `fk_table1_dim_products1_idx` (`dim_products_iddim_products` ASC) VISIBLE,
  PRIMARY KEY (`order_details`),
  UNIQUE INDEX `order_details1_UNIQUE` (`order_details` ASC) VISIBLE)
ENGINE = InnoDB;
```

```sql
-- -----------------------------------------------------
-- Table `dw_northwind`.`Dim_purchase_details`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`Dim_purchase_details` (
  `idpurchase_details` INT NOT NULL,
  `porduct_id` INT(11) NULL,
  `quantity` DECIMAL(18,14) NULL,
  `unit_cost` DECIMAL(18,14) NULL,
  `dim_purchase_order_iddim_purchase_order` INT NOT NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`idpurchase_details`),
  UNIQUE INDEX `idpurchase_details_UNIQUE` (`idpurchase_details` ASC) VISIBLE,
  INDEX `fk_purchase_details_dim_purchase_order1_idx` (`dim_purchase_order_iddim_purchase_order` ASC) VISIBLE)
ENGINE = InnoDB;
```

```sql
-- Table `dw_northwind`.`factos_order`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`factos_order` (
  `idfactos_order` INT NOT NULL,
  `dim_customers_iddim_customers` INT NOT NULL,
  `dim_employees_iddim_employees` INT NOT NULL,
  `dim_shippers_iddim_shippers` INT NULL,
  `dim_data_idorder_date` INT NOT NULL,
  `Dim_payment_idDim_payment` INT NULL,
  `dim_order_status_iddim_order_satus` INT NOT NULL,
  `dim_data_idpaid_date` INT NULL,
  `dim_data_idshipped_date` INT NULL,
  PRIMARY KEY (`idfactos_order`),
  UNIQUE INDEX `idfactos_order_UNIQUE` (`idfactos_order` ASC) VISIBLE,
  INDEX `fk_factos_order_dim_customers1_idx` (`dim_customers_iddim_customers` ASC) VISIBLE,
  INDEX `fk_factos_order_dim_employees1_idx` (`dim_employees_iddim_employees` ASC) VISIBLE,
  INDEX `fk_factos_order_dim_shippers1_idx` (`dim_shippers_iddim_shippers` ASC) VISIBLE,
  INDEX `fk_factos_order_dim_data1_idx` (`dim_data_idorder_date` ASC) VISIBLE,
  INDEX `fk_factos_order_Dim_payment1_idx` (`Dim_payment_idDim_payment` ASC) VISIBLE,
  INDEX `fk_factos_order_dim_order_status1_idx` (`dim_order_status_iddim_order_satus` ASC) VISIBLE,
  INDEX `fk_factos_order_dim_data2_idx` (`dim_data_idpaid_date` ASC) VISIBLE,
  INDEX `fk_factos_order_dim_data3_idx` (`dim_data_idshipped_date` ASC) VISIBLE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `dw_northwind`.`dim_suplier`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `dw_northwind`.`dim_suplier` (
  `iddim_suplier` INT NOT NULL AUTO_INCREMENT,
  `company` VARCHAR(50) NULL,
  `name` VARCHAR(100) NULL,
  `jobtitle` VARCHAR(100) NULL,
  `is_current` VARCHAR(3) NULL,
  `created_date` DATE NULL,
  `expired_date` DATE NULL,
  PRIMARY KEY (`iddim_suplier`),
  UNIQUE INDEX `iddim_suplier_UNIQUE` (`iddim_suplier` ASC) VISIBLE)
ENGINE = InnoDB;
```

*Figura 6 - Modelo Físico*

De seguida realizamos a conversão do modelo dimensional para o modelo físico, obtendo o modelo ilustrado a cima. É de salientar que este modelo não tem as chaves estrangeiras criadas pois quando procedíamos a conexão com o *Power Bi*, não nos permitia a conexão por causa delas.

Para fazermos o povoamento destas tabelas com os dados da *northwind* criamos procedures que continham cursores, serão apresentados em baixo.

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoarcustomer`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare empresa, posicao,cidade,pais varchar(50);
Declare nome varchar(100);

Declare cur1 cursor for Select company,job_title,city,country_region, concat(first_name,_utf8' ',last_name)
from northwind.customers;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into empresa,posicao,cidade,pais,nome;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_customers (company,name,jobtitle,city,country,is_current,created_date,expired_date)
values(empresa,nome,posicao,cidade,pais,'yes',current_date(),null);
end loop;
insert into dw_northwind.dim_customers (company,name,jobtitle,city,country,is_current,created_date,expired_date)
values('Undefined','Undefined','Undefined','Undefined','Undefined',null,null,null);
close cur1;

END
```

Figura 7 – Cursor para a tabela Dim_Customer

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoar_employees`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare empresa, posicao,cidade,pais varchar(50);
Declare nome varchar(100);

Declare cur1 cursor for Select company,job_title,city,country_region, concat(first_name,_utf8' ',last_name)
from northwind.employees;


DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into empresa,posicao,cidade,pais,nome;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_employees (company,name,jobtitle,city,country,is_current,created_date,expired_date)
values(empresa,nome,posicao,cidade,pais,'yes',current_date(),null);
end loop;
insert into dw_northwind.dim_employees (company,name,jobtitle,city,country,is_current,created_date,expired_date)
values('Undefined','Undefined','Undefined','Undefined','Undefined',null,null,null);
close cur1;
```

Figura 8 - Cursor para a tabela Dim_Employees

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoarinvoice`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare ide, ido int;

Declare cur1 cursor for Select id,order_id from northwind.invoices;


DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into ide,ido;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_invoice (iddim_invoice,factos_order_idfactos_order,is_current,created_date,expired_date)
values(ide,ido,'yes',current_date(),null);
end loop;
close cur1;


END
```

Figura 9 - Cursor para a tabela Dim_Invoice

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoar_supplier`()
BEGIN

DECLARE done INT DEFAULT FALSE;
Declare empresa, posicao varchar(50);
Declare nome varchar(100);

Declare cur1 cursor for Select company,job_title, concat(first_name,_utf8' ',last_name)
from northwind.suppliers;


DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into empresa,posicao,nome;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_suplier (company,name,jobtitle,is_current,created_date,expired_date)
values(empresa,nome,posicao,'yes',current_date(),null);
end loop;
insert into dw_northwind.dim_suplier (company,name,jobtitle,is_current,created_date,expired_date)
values('Undefined','Undefined','Undefined',null,null,null);
close cur1;

END
```

Figura 10 - Cursor para a tabela Dim_Supplier

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoarorderstatus`()
BEGIN

DECLARE done INT DEFAULT FALSE;
Declare estado varchar(50);
Declare ido int(11);


Declare cur1 cursor for Select id, status_name from northwind.orders_status;


DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into ido,estado;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_order_status (iddim_order_satus,status_name)
 values(ido,estado);
end loop;
insert into  dw_northwind.dim_order_status (iddim_order_satus,status_name)
values(4,'Undefined');
close cur1;
```

Figura 11 - Cursor para a tabela Dim_Order_status

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoarshippers`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare empresa, cidade,pais varchar(50);

Declare cur1 cursor for Select company,city,country_region from northwind.shippers;


DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into empresa,cidade,pais;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_shippers (company,city,country, is_current, created_date, expired_date)
values(empresa,cidade,pais,'yes',current_date(),null);
end loop;
insert into dw_northwind.dim_shippers (company,city,country, is_current, created_date, expired_date)
values('Undefined','Undefined','Undefined',null,null,null);
close cur1;

END
```

*Figura 12 - Cursor para a tabela Dim_Shippers*

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoarintermedia`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare ido,idp, id1, id2 int(11);
Declare quantidade, preco_unidade decimal(18,4);


Declare cur1 cursor for Select id,order_id,product_id from northwind.order_details;
Declare cur2 cursor for Select d.id, quantity, unit_price from northwind.order_details d,
northwind.order_details_status s, northwind.orders o
where d.order_id=o.id and d.status_id=s.id order by d.id;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;
open cur2;

read_loop:loop
fetch cur1 into id1,ido,idp;
fetch cur2 into id2,quantidade, preco_unidade;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.order_details_produto (order_details,factos_order_idfactos_order,
dim_products_iddim_products,quantity,unit_price,is_current,created_date,expired_date)
    values(id1,ido,idp,quantidade, preco_unidade,'yes',current_date(),null);
end loop;
close cur1;
close cur2;

END
```

Figura 13 - Cursor para a tabela order_details_produto

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoarpayment`()
BEGIN

DECLARE done INT DEFAULT FALSE;
Declare tipo varchar(50);


Declare cur1 cursor for Select distinct payment_type from northwind.orders
where payment_type is not null;


DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;
insert into  dw_northwind.dim_payment (payment_type) values('Undefined');
read_loop:loop
fetch cur1 into tipo;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_payment (payment_type) values(tipo);
end loop;
```

Figura 14 - Cursor para a tabela Dim_Payment

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoar_purchase_details`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare quantidade,custo decimal(18,4);
Declare id_pd, id1,pid int(11);

Declare cur1 cursor for Select d.id,o.id,d.quantity, d.unit_cost,d.product_id
 from northwind.purchase_order_details d, northwind.purchase_orders o
where o.id=d.purchase_order_id;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into id1,id_pd,quantidade,custo,pid;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_purchase_details (idpurchase_details,quantity,unit_cost,
dim_purchase_order_iddim_purchase_order, is_current,created_date,expired_date,porduct_id)
        values(id1,quantidade,custo,id_pd,'yes',current_date(),null,pid);
end loop;
close cur1;
```

Figura 15 - Cursor para a tabela Dim_Purchase_details

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoarproducts`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare preco_standard, preco_lista decimal(19,4);
Declare nome,categoria varchar(50);
Declare id1,quantidade,id_produto int(11);

Declare cur1 cursor for Select id,standard_cost,list_price,product_name,category from northwind.products;
Declare cur2 cursor for select id, quantity from(
Select p.id as id,sum(t.quantity) as quantity from northwind.inventory_transactions t, northwind.products p
where t.product_id=p.id group by p.id
union
SELECT distinct p.id as id, 0 as quantaty FROM northwind.inventory_transactions t, northwind.products p
WHERE p.id NOT IN (Select distinct p.id from northwind.inventory_transactions t, northwind.products p
                    where t.product_id=p.id)) as f order by id;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;
open cur2;

read_loop:loop
fetch cur1 into id_produto,preco_standard,preco_lista,nome,categoria;
fetch cur2 into id1,quantidade;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_products (iddim_products,standard_cost,list_price,name,category,quantity_stock,is_current,
created_date,expired_date)
    values(id_produto,preco_standard,preco_lista,nome,categoria, quantidade,'yes',current_date(),null);
end loop;
insert into dw_northwind.dim_products (iddim_products,standard_cost,list_price,name,category,quantity_stock,
is_current,created_date,expired_date)
insert into dw_northwind.dim_products (iddim_products,standard_cost,list_price,name,category,quantity_stock,
is_current,created_date,expired_date)
    values(100,0,0,'Undefined','Undefined', 0,null,null,null);
close cur1;
close cur2;
END
```

Figura 16 - Cursor para a tabela Dim_Products

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoardata`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare datas date;
declare dia, mes,ano,semana int(11);
declare diasemana varchar(45);
declare U,F varchar(2);

Declare cur1 cursor for SELECT STR_TO_DATE(datar, '%d-%m-%Y') as d from calendario.calendario;
DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;

read_loop:loop
fetch cur1 into datas;

    IF done THEN
      LEAVE read_loop;
    END IF;

    select C.UTIL into U from calendario.calendario C where str_to_date(C.datar, '%d-%m-%Y') = datas;
    select C.FERIADO into F from calendario.calendario C where str_to_date(C.datar, '%d-%m-%Y') = datas;
    select date_format(datas, '%d') into dia;
    select date_format(datas, '%m') into mes;
    select date_format(datas, '%Y') into ano;
    select date_format(datas, '%v') into semana;
    select DAYNAME(datas) into diasemana;

insert into dw_northwind.dim_data (Date_da_ordem,day_ordem,year_ordem,month,day_of_week,week_of_year,dia_util_ordem,feriado_ordem)
    values(datas,dia,ano,mes,diasemana,semana,U,F);

end loop;
close cur1;
END
```

Figura 17 - Cursor para a tabela Dim_Data

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoar_factos_order`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare quantidade decimal(18,4);
Declare preco_unidade decimal(18,4);
Declare id_customer,id_pagamento int(11);
Declare id_employees,id_shippers int(11);
Declare id1,id2,id_data,id_orders, id_estado, id_paid, id_ship int(11);


Declare cur1 cursor for Select id, if(shipper_id is null,4,shipper_id) as shipper_id, employee_id, customer_id from northwind.orders;
declare cur2 cursor for select distinct i,if(d is null,7141,d) as d from(
select o.id as i,o.paid_date as d from northwind.orders o where  o.paid_date is null
union
select o.id as i,d.iddim_data as d from northwind.orders o, dw_northwind.dim_data d where  o.paid_date=d.Date_da_ordem) as f order by i;
Declare cur3 cursor for Select idDim_data from dw_northwind.dim_data d, northwind.orders o where
cast(o.order_date as date)=cast(d.Date_da_ordem as date);
Declare cur6 cursor for select distinct i,if(d is null,7141,d) as d from(
select o.id as i,o.shipped_date as d from northwind.orders o where  o.shipped_date is null
union
select o.id as i,d.iddim_data as d from northwind.orders o, dw_northwind.dim_data d where  o.shipped_date=d.Date_da_ordem) as f order by i;
declare cur8 cursor for select o.id, r.id from northwind.orders_status r, northwind.orders o where o.status_id=r.id order by o.id;
Declare cur5 cursor for select distinct o.id, if(o.payment_type is null,1,pa.idDim_payment) from  dw_northwind.dim_payment pa,northwind.orders o

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;
open cur2;
open cur3;
open cur5;
open cur6;
open cur8;
read_loop:loop
fetch cur1 into id_orders,id_shippers,id_employees,id_customer;
fetch cur2 into id1,id_paid;
fetch cur3 into id_data;
fetch cur5 into id1,id_pagamento;
fetch cur6 into id1,id_ship;
fetch cur8 into id2,id_estado;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.factos_order(idfactos_order,dim_customers_iddim_customers,dim_employees_iddim_employees,
dim_shippers_iddim_shippers,dim_data_idorder_date,Dim_payment_idDim_payment,dim_order_status_iddim_order_satus,
dim_data_idpaid_date, dim_data_idshipped_date)
        values (id_orders,id_customer,id_employees,id_shippers,id_data,id_pagamento,id_estado,id_paid, id_ship);
end loop;
close cur1;
close cur2;
close cur3;
close cur5;
close cur6;
close cur8;

END
```

Figura 18 - Cursor para a tabela de Factos

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `povoar_purchase_order`()
BEGIN
DECLARE done INT DEFAULT FALSE;
Declare estado varchar(50);
Declare id1,id_compra, id_s,id_emplo int(11);

Declare cur1 cursor for Select p.id,p.supplier_id, s.status from northwind.purchase_orders p,
northwind.purchase_order_status s where p.status_id=s.id order by p.id;
Declare cur2 cursor for select id,if(created_by is null,10,created_by) from northwind.purchase_orders order by id;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
open cur1;
open cur2;



read_loop:loop
fetch cur1 into id_compra,id_s,estado;
fetch cur2 into id1,id_emplo;

    IF done THEN
      LEAVE read_loop;
    END IF;
insert into dw_northwind.dim_purchase_order (iddim_purchase_order,status,dim_employees_iddim_employees,
dim_suplier_iddim_suplier, is_current, created_date,expired_date)
        values(id_compra,estado,id_emplo,id_s,'yes',current_date(),null);
end loop;
close cur1;
close cur2;

END
```

Figura 19 - Cursor para a tabela Dim_purchase_order

# 3. Registo de alterações

## Tabelas auxiliares criadas:

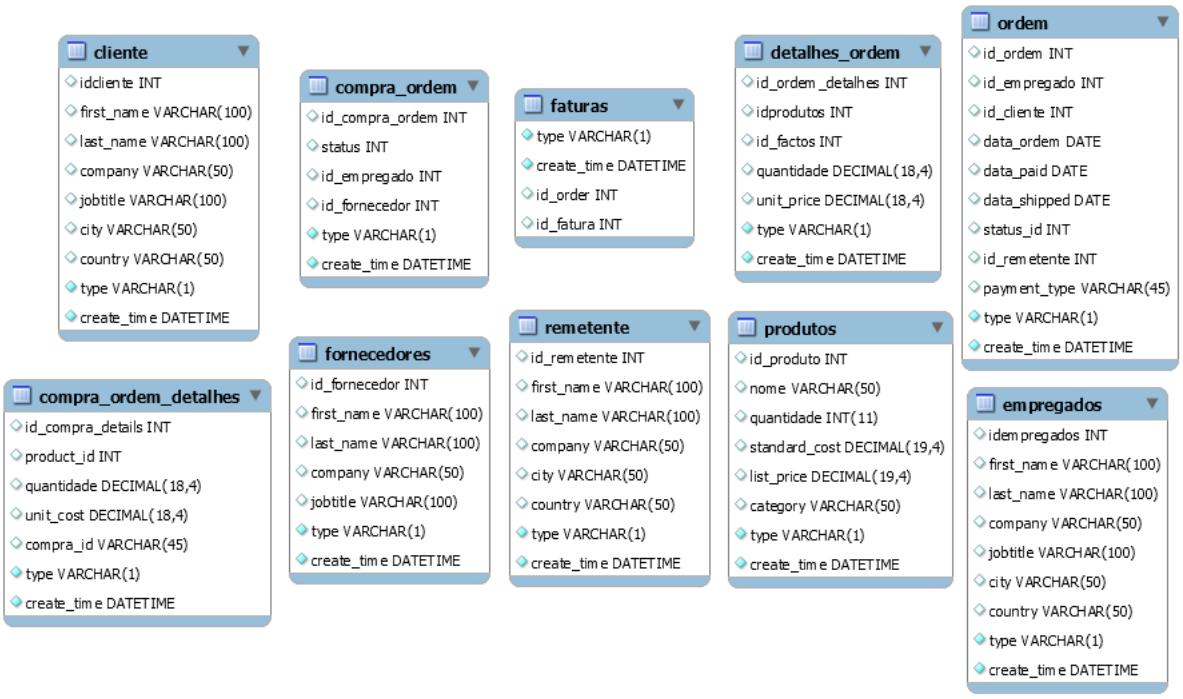



Figura 20 - Tabelas auxiliares

Procedemos a criação destas tabelas na base de dados de produção e na área de retenção. Sempre que surgir uma alteração, seja ela de *insert*, *update* ou *delete*, na base de dados de produção, através de triggers, essas são passada para uma destas tabelas correspondentes a alteração que se encontram nesta base de dados. De seguida para acontecer a passagem dessa informação para a área de retenção produzimos código em C# que permitirá conectar a base de dados de produção a área de retenção procedendo-se a escrita das alterações da primeira para a segunda, respetivamente.

Os atributos selecionados para estarem presentes nestas tabelas foram os que nos pareceram mais importantes e que sofriam mais alterações, também achamos de extrema importância adicionar uma coluna do *type* e *create_time*, de forma a sabermos que operação foi efetuada na base de dados e a hora e a data do acontecimento para que depois na área de retenção consigamos trabalhar com essa informação.

## Triggers criados na base de dados de produção:

```
CREATE TRIGGER insercecliente
AFTER INSERT
ON customers
FOR EACH ROW
BEGIN

    INSERT INTO cliente(idcliente,first_name,last_name,company,jobtitle,city,country,type,create_time)
        values (new.id,new.first_name, new.last_name,new.company,new.job_title,new.country_region,'I',Now());
END#


CREATE TRIGGER updatecliente
AFTER Update
ON customers
FOR EACH ROW
BEGIN
        INSERT INTO cliente(idcliente,first_name,last_name,company,jobtitle,city,country,type,create_time)
        values (new.id,new.first_name, new.last_name,new.company,new.job_title,new.country_region,'U',Now());
END#


CREATE TRIGGER deletecliente
after delete
ON customers
FOR EACH ROW
BEGIN
        INSERT INTO cliente(idcliente,first_name,last_name,company,jobtitle,city,country,type,create_time )
        values (old.id,old.first_name, old.last_name,old.company,old.job_title,old.country_region,'D',NOW());
END#
```

Figura 22 - Trigger para a tabela cliente

```
CREATE TRIGGER insereempregados
AFTER INSERT
ON employees
FOR EACH ROW
BEGIN

    INSERT INTO empregados(idempregados,first_name,last_name,company,jobtitle,city,country,type,create_time)
        values (new.id,new.first_name, new.last_name,new.company,new.job_title,new.city,new.country_region,'I',Now());
END#


CREATE TRIGGER updateempregados
AFTER Update
ON employees
FOR EACH ROW
BEGIN
        INSERT INTO empregados(idempregados,first_name,last_name,company,jobtitle,city,country,type,create_time)
        values (new.id,new.first_name, new.last_name,new.company,new.job_title,new.city,new.country_region,'U',Now());
END#


CREATE TRIGGER deleteempregados
after delete
ON employees
FOR EACH ROW
BEGIN
        INSERT INTO empregados(idempregados,first_name,last_name,company,jobtitle,city,country,type,create_time )
        values (old.id,old.first_name, old.last_name,old.company,old.job_title,old.city,old.country_region,'D',NOW());
END#
```

Figura 22 - Trigger para a tabela empregados

```
CREATE TRIGGER inserecompra_ordem
AFTER INSERT
ON purchase_orders
FOR EACH ROW
BEGIN

    INSERT INTO compra_ordem(id_compra_ordem,status,id_empregado,id_fornecedor,type,create_time)
        values (new.id,new.status_id,new.created_by,new.supplier_id,'I',Now());
END#


CREATE TRIGGER updatecompra_ordem
AFTER Update
ON purchase_orders
FOR EACH ROW
BEGIN

   INSERT INTO compra_ordem(id_compra_ordem,status,id_empregado,id_fornecedor,type,create_time)
        values (new.id,new.status_id,new.created_by,new.supplier_id,'U',Now());

END#


CREATE TRIGGER deletecompra_ordem
after delete
ON purchase_orders
FOR EACH ROW
BEGIN

    INSERT INTO compra_ordem(id_compra_ordem,status,id_empregado,id_fornecedor,type,create_time)
        values (old.id,old.status_id,old.created_by,old.supplier_id,'D',Now());

END#
```

Figura 24 - Trigger para a tabela compra_ordem

```
CREATE TRIGGER inserecompra_details
AFTER INSERT
ON purchase_order_details
FOR EACH ROW
BEGIN

    INSERT INTO compra_details_detalhes(id_compra_details,product_id,quantidade,unit_cost,compra_id,type,create_time)
        values (new.id,new.product_id,new.quantity,new.unit_cost,new.purchase_order_id,'I',Now());
END#


CREATE TRIGGER updatecompra_details
AFTER Update
ON purchase_order_details
FOR EACH ROW
BEGIN

    INSERT INTO compra_details_detalhes(id_compra_details,product_id,quantidade,unit_cost,compra_id,type,create_time)
        values (new.id,new.product_id,new.quantity,new.unit_cost,new.purchase_order_id,'U',Now());

END#


CREATE TRIGGER deletecompra_details
after delete
ON purchase_order_details
FOR EACH ROW
BEGIN

    INSERT INTO compra_details_detalhes(id_compra_details,product_id,quantidade,unit_cost,compra_id,type,create_time)
        values (old.id,old.product_id,old.quantity,old.unit_cost,old.purchase_order_id,'D',Now());

END#
```

Figura 24 - Trigger para a tabela compra_details

```
CREATE TRIGGER insereprodutos
AFTER INSERT
ON products
FOR EACH ROW
BEGIN
    INSERT INTO produtos(id_produto,quantidade,nome,standard_cost,list_price,category,type,create_time)
    values (new.id,0,new.product_name,new.standard_cost,new.list_price,new.category,'I',Now());
END#


CREATE TRIGGER updateprodutos
AFTER Update
ON products
FOR EACH ROW
BEGIN
    Declare quant int(11);

    Select sum(quantity) into quant from northwind.inventory_transactions where id=new.id;

    INSERT INTO produtos(id_produto,quantidade,nome,standard_cost,list_price,category,type,create_time)
    values (new.id,quant,new.product_name,new.standard_cost,new.list_price,new.category,'U',Now());

END#


CREATE TRIGGER deleteprodutos
after delete
ON products
FOR EACH ROW
BEGIN

    INSERT INTO produtos(id_produto,nome,standard_cost,list_price,category,type,create_time)
    values (old.id,old.product_name,old.standard_cost,old.list_price,old.category,'D',Now());

END#
```

Figura 26 - Trigger para a tabela produtos

```
CREATE TRIGGER insereorder
AFTER INSERT
ON orders
FOR EACH ROW
BEGIN

    INSERT INTO ordem(id_ordem,id_empregado,id_cliente,data_ordem,data_paid,data_shipped,status_id,id_remetente,payment_type,type,create_time)
        values (new.id,new.employee_id,new.customer_id,cast(new.order_date as date),cast(new.paid_date as date),cast(new.shipped_date),
        new.status_id,new.shipper_id,new.payment_type,'I',Now());
END#


CREATE TRIGGER updateorder
AFTER Update
ON orders
FOR EACH ROW
BEGIN

    INSERT INTO ordem(id_ordem,id_empregado,id_cliente,data_ordem,data_paid,data_shipped,status_id,id_remetente,payment_type,type,create_time)
        values (new.id,new.employee_id,new.customer_id,cast(new.order_date as date),cast(new.paid_date as date),cast(new.shipped_date),
        new.status_id,new.shipper_id,new.payment_type,'U',Now());
END#


CREATE TRIGGER deleteorder
after delete
ON orders
FOR EACH ROW
BEGIN

    INSERT INTO ordem(id_ordem,id_empregado,id_cliente,data_ordem,data_paid,data_shipped,status_id,id_remetente,payment_type,type,create_time)
        values (old.id,old.employee_id,old.customer_id,cast(old.order_date as date),cast(old.paid_date as date),cast(old.shipped_date),
        old.status_id,old.shipper_id,old.payment_type,'D',Now());
END#
```

Figura 25 - Trigger para a tabela ordem

```
CREATE TRIGGER inseredetalhes_ordem
AFTER INSERT
ON order_details
FOR EACH ROW
BEGIN

    INSERT INTO detalhes_ordem(id_ordem_detalhes,idprodutos,id_factos,quantidade,unit_price,type,create_time)
        values (new.id,new.product_id,new.order_id,new.quantity,new.unit_price,'I',Now());
END#


CREATE TRIGGER updatedetalhes_ordem
AFTER Update
ON order_details
FOR EACH ROW
BEGIN

   INSERT INTO detalhes_ordem(id_ordem_detalhes,idprodutos,id_factos,quantidade,unit_price,type,create_time)
        values (new.id,new.product_id,new.order_id,new.quantity,new.unit_price,'U',Now());

END#
```

Figura 27 - Trigger para a tabela detalhes_ordem

```
CREATE TRIGGER insereforneceedores
AFTER INSERT
ON suppliers
FOR EACH ROW
BEGIN

    INSERT INTO fornecedores(id_fornecedor,company,first_name, last_name,jobtitle,type,create_time)
        values (new.id,new.company,new.first_name, new.last_name,new.job_title,'I',Now());
END#


CREATE TRIGGER updatefornecedores
AFTER Update
ON shippers
FOR EACH ROW
BEGIN

INSERT INTO fornecedores(id_fornecedor,company,first_name, last_name,jobtitle,type,create_time)
        values (new.id,new.company,new.first_name, new.last_name,new.job_title,'U',Now());
END#


CREATE TRIGGER deletefornecedores
after delete
ON shippers
FOR EACH ROW
BEGIN

INSERT INTO fornecedores(id_fornecedor,company,first_name, last_name,jobtitle,type,create_time)
        values (old.id,old.company,old.first_name,old.last_name,old.job_title,'D',Now());
END#
```

Figura 28 - Trigger para a tabela fornecedores

```
CREATE TRIGGER insereremetente
AFTER INSERT
ON shippers
FOR EACH ROW
BEGIN

    INSERT INTO remetente(id_remetente,first_name, last_name,company,city,country,type,create_time)
        values (new.id,new.first_name, new.last_name,new.company,new.city,new.country_region,'I',Now());
END#


CREATE TRIGGER updateremetente
AFTER Update
ON shippers
FOR EACH ROW
BEGIN

INSERT INTO remetente(id_remetente,first_name, last_name,company,city,country,type,create_time)
        values (new.id,new.first_name, new.last_name,new.company,new.city,new.country_region,'U',Now());
END#


CREATE TRIGGER deleteremetente
after delete
ON shippers
FOR EACH ROW
BEGIN

  INSERT INTO remetente(id_remetente,first_name, last_name,company,city,country,type,create_time)
        values (old.id,old.first_name,old.last_name,old.company,old.city,old.country_region,'D',Now());
END#
```

Figura 30 - Trigger para a tabela remetente

```
CREATE TRIGGER inserefaturas
AFTER INSERT
ON invoices
FOR EACH ROW
BEGIN

    INSERT INTO faturas(id_order,id_fatura,type,create_time)
        values (new.order_id,new.id,'I',Now());
END#


CREATE TRIGGER updatefaturas
AFTER Update
ON invoices
FOR EACH ROW
BEGIN

   INSERT INTO faturas(id_order,id_fatura,type,create_time)
        values (new.order_id,new.id,'U',Now());
END#


CREATE TRIGGER deletefaturas
after delete
ON invoices
FOR EACH ROW
BEGIN

 INSERT INTO faturas(id_order,id_fatura,type,create_time)
        values (old.order_id,old.id,'D',Now());
END#
```

Figura 29 - Trigger para a tabela faturas

```
CREATE TRIGGER insereinventorio
AFTER INSERT
ON inventory_transactions
FOR EACH ROW
BEGIN
    Declare nome, cat varchar(50);
    Declare std_cost,price decimal(19,4);
    Declare quant int(11);

    Select sum(quantity) into quant from northwind.inventory_transactions where product_id=new.product_id;
    Select product_name,category,standard_cost,list_price into nome,cat,std_cost,price from northwind.products where id=new.product_id;

    INSERT INTO produtos(id_produto,quantidade,nome,standard_cost,list_price,category,type,create_time)
    values (new.product_id,quant,nome,std_cost,price,cat,'U',Now());
END#


CREATE TRIGGER updateinventorio
AFTER Update
ON inventory_transactions
FOR EACH ROW
BEGIN

    Declare nome, cat varchar(50);
    Declare std_cost,price decimal(19,4);
    Declare quant int(11);

    Select sum(quantity) into quant from northwind.inventory_transactions where product_id=new.product_id;
    Select product_name,category,standard_cost,list_price into nome,cat,std_cost,price from northwind.products where id=new.product_id;

    INSERT INTO produtos(id_produto,quantidade,nome,standard_cost,list_price,category,type,create_time)
    values (new.product_id,quant,nome,std_cost,price,cat,'U',Now());

END#
 CREATE TRIGGER deleteinventorio
 after delete
 ON inventory_transactions
 FOR EACH ROW
 BEGIN

     Declare nome, cat varchar(50);
     Declare std_cost,price decimal(19,4);
     Declare quant int(11);

     Select sum(quantity) into quant from northwind.inventory_transactions where product_id=old.product_id;
     Select product_name,category,standard_cost,list_price into nome,cat,std_cost,price from northwind.products where id=old.product_id;

     INSERT INTO produtos(id_produto,quantidade,nome,standard_cost,list_price,category,type,create_time)
     values (old.product_id,quant,nome,std_cost,price,cat,'U',Now());

 END#
```

**Figura 31 - Trigger para a tabela inventorio**

Estes triggers irão permitir que quando seja então detectada alguma alteração sejam disparados e façam um *insert* dessas alterações para as tabelas que foram referidas anteriormente.

# 4. Passagem dos dados entre bases de dados

Como ponte entre a nossa base de dados de produção e o *data warehouse* foi necessário criarmos um programa em C#, cuja funcionalidade rege-se em periodicamente verificar se houveram alterações na base de dados de produção para depois repercutir e atualizar, se necessário, os dados do *data warehouse* para que este armazene sempre um histórico atualizado.

Outro aspeto fundamental neste programa, foi a implementação de uma funcionalidade que, após ser feita a atualização do *data warehouse*, e a determinada hora do dia, seja feita uma limpeza da *staging area*.

Para que tal acontecesse, foi necessário estabelecer uma conexão entre as bases de dados *NorthWind*, *dw_NorthWind* e o nosso programa através da utilização das connection *string*s definidas abaixo

```
public static string connectionstring = "server =localhost;user id = root; Password=1234; persistsecurityinfo=True;database=northwind";
public static string connectionstring2 = "server =localhost;user id = root; Password=1234; persistsecurityinfo=True;database=dw_northwind";
```

Figura 32 - Estabelecimento das connection Strings

A imagem seguinte representa a porção de código correspondente a parte inicial do nosso programa que é responsável por fazer a leitura e escrita dos dados das bases de dados de 5 em 5 minutos para fazer a atualização dos dados. Por outro lado, se a hora atual for igual as 23h, é feita uma limpeza das tabelas da *staging area*.

```
static void ChamarMetodos()
{
    while (true)
    {
        if (DateTime.Now.Hour == 23 && DateTime.Now.Minute < 5)
        {
            DeleteAllData();

        }
        else
        {
            ReadCustomers();
            ReadPurchaseOrders();
            ReadPurchaseOrdersDetails();
            ReadOrderDetails();
            ReadEmployees();
            ReadFaturas();
            ReadFornecedores();
            ReadOrders();
            ReadProducts();
            ReadRemetende();
            WriteCustomers();
            WritePurchaseOrders();
            WritePurchaseOrdersDetails();
            WriteOrdersDetails();
            WriteEmployees();
            WriteFaturas();
            WriteFornecedores();
            WriteOrders();
            WriteProducts();
            WriteRemetente();

        }
        Thread.Sleep(1000 * 60 * 5); //executa o codigo de 5 em 5 minutos
    }
```

Figura 33 - Código inicial do programa

Em baixo é apresentado o código responsável por fazer a limpeza das tabelas da *staging area*.

```
public static void DeleteAllData()
{
    string sql1 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM Cliente  ";
    string sql2 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM compra_ordem  ";
    string sql3 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM compra_ordem_detalhes  ";
    string sql4 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM detalhes_ordem  ";
    string sql5 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM empregados  ";
    string sql6 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM faturas  ";
    string sql7 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM fornecedores  ";
    string sql8 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM inventorios  ";
    string sql9 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM ordem  ";
    string sql10 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM produtos  ";
    string sql11 = " SET SQL_SAFE_UPDATES = 0;" +
                  "   Delete FROM Remetente  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlConnection con2 = new MySqlConnection(connectionstring2);
    MySqlCommand cmd1 = new MySqlCommand(sql1, con);
    MySqlCommand cmd2 = new MySqlCommand(sql2, con);
    MySqlCommand cmd3 = new MySqlCommand(sql3, con);
    MySqlCommand cmd4= new MySqlCommand(sql4, con);
    MySqlCommand cmd5 = new MySqlCommand(sql5, con);
    MySqlCommand cmd6 = new MySqlCommand(sql6, con);
    MySqlCommand cmd7 = new MySqlCommand(sql7, con);
    MySqlCommand cmd8 = new MySqlCommand(sql8, con);
    MySqlCommand cmd9 = new MySqlCommand(sql9, con);
    MySqlCommand cmd10 = new MySqlCommand(sql10, con);
    MySqlCommand cmd11 = new MySqlCommand(sql11, con);
    MySqlCommand cmd12 = new MySqlCommand(sql1, con2);
    MySqlCommand cmd13 = new MySqlCommand(sql2, con2);
    MySqlCommand cmd14 = new MySqlCommand(sql3, con2);
    MySqlCommand cmd15 = new MySqlCommand(sql4, con2);
    MySqlCommand cmd16 = new MySqlCommand(sql5, con2);
    MySqlCommand cmd17 = new MySqlCommand(sql6, con2);
    MySqlCommand cmd18 = new MySqlCommand(sql7, con2);
    MySqlCommand cmd19 = new MySqlCommand(sql8, con2);
    MySqlCommand cmd20 = new MySqlCommand(sql9, con2);
    MySqlCommand cmd21 = new MySqlCommand(sql10, con2);
    MySqlCommand cmd22 = new MySqlCommand(sql11, con2);
    con.Open();
    try
    {
        cmd1.ExecuteNonQuery();
        cmd2.ExecuteNonQuery();
        cmd3.ExecuteNonQuery();
        cmd4.ExecuteNonQuery();
        cmd5.ExecuteNonQuery();
        cmd6.ExecuteNonQuery();
        cmd7.ExecuteNonQuery();
        cmd8.ExecuteNonQuery();
        cmd9.ExecuteNonQuery();
        cmd10.ExecuteNonQuery();
        cmd11.ExecuteNonQuery();
        cmd12.ExecuteNonQuery();
        cmd13.ExecuteNonQuery();
        cmd14.ExecuteNonQuery();
        cmd15.ExecuteNonQuery();
        cmd16.ExecuteNonQuery();
        cmd17.ExecuteNonQuery();
        cmd18.ExecuteNonQuery();
        cmd19.ExecuteNonQuery();
        cmd20.ExecuteNonQuery();
        cmd21.ExecuteNonQuery();
        cmd22.ExecuteNonQuery();

    }
    catch (Exception ex)
    {
```

**Figura 34 - Código de limpeza para a staging area**

Como foi referido acima, esta porção de código apenas será executada às 23h de cada dia.

# Customers

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela Cliente do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Customers.cs.

```
76      public static void ReadCustomers()
77      {
78          string sql = " SELECT * FROM Cliente  ";
79          MySqlConnection con = new MySqlConnection(connectionstring);
80          MySqlCommand cmd = new MySqlCommand(sql, con);
81          Console.WriteLine("-----Leitura dos Clientes---------\n");
82          con.Open();
83          try
84          {
85              MySqlDataReader dr = cmd.ExecuteReader();
86
87              while (dr.Read())
88              {
89                  Customers customers = new Customers();
90
91                  customers.Id_Cliente = (int)dr["idcliente"];
92                  customers.First_name = (string)dr["first_name"];
93                  customers.Last_name= (string)dr["last_name"];
94                  customers.Company= (string)dr["company"];
95                  customers.Job_title = (string)dr["jobtitle"];
96                  customers.City = (string)dr["city"];
97                  customers.Country = (string)dr["country"];
98                  customers.Type = (string)dr["type"];
99                  customers.Create_Time = (DateTime)dr["create_time"];
100
101                 Listcustomers.Add(customers);
102             }
103
104             Console.WriteLine("Clientes lidos com sucesso!!\n");
105         }
106         catch (Exception ex)
107         {
108             Console.WriteLine("Erro:" + ex.Message);
109         }
110         finally
111         {
112             con.Close();
113         }
114     }
```

Figura 35 - Leitura dos customers

Após o armazenamento dos dados na lista,é feita a inserção dos dados na tabela Cliente da base de dados *Dw_Northwind*.

```
public static void WriteCustomers()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Customers customers in Listcustomers)
            {

                string date = customers.Create_Time.ToString("yyyy-MM-dd HH:mm:ss");

                cmd.CommandText = "INSERT INTO cliente (idcliente,first_name,last_name,company,jobtitle,city,country,type,create_time)" +
                    " VALUES("
                    + customers.Id_Cliente + ","
                    + "'" + customers.First_name + "'" + ","
                    + "'" + customers.Last_name + "'" + ","
                    + "'" + customers.Company + "'" + ","
                    + "'" + customers.Job_title + "'" + ","
                    + "'" + customers.City + "'" + ","
                     + "'" + customers.Country + "'" + ","
                     + "'" + customers.Type + "'" + ","
                     + "'" + date + "'"
                    + ");";

                cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
```

Figura 36 - Escrita dos dados na tabela cliente da base de dados dw_Northwind

## Purchase Orders

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela Compra_Ordem do schema *Northwind*.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo *Puchase_Orders.cs.*

```
public static void ReadPurchaseOrders()
{
    string sql = " SELECT * FROM compra_ordem  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos Puchase Orders---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Purchase_Orders po = new Purchase_Orders();

            po.Id = (int)dr["id_compra_ordem"];
            po.Id_Empregado = (int)dr["id_empregado"];
            po.Status = (int)dr["status"];
            po.Id_Fornecedor= (int)dr["id_fornecedor"];
            po.Type= (string)dr["type"];
            po.Create_Date = (DateTime)dr["create_time"];

            ListPurchaseOrders.Add(po);
        }

        Console.WriteLine("Purchase Orders lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
    finally
    {
        con.Close();
    }
```

Figura 37 - Leitura dos Puchase Orders

Após o armazenamento dos dados na lista, é feita a inserção dos dados tabela Compra_Ordem da base de dados Dw_Northwind.

```
public static void WritePurchaseOrders()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Purchase_Orders po in ListPurchaseOrders)
            {
                string date = po.Create_Date.ToString("yyyy-MM-dd HH:mm:ss");

                    cmd.CommandText = "INSERT INTO compra_ordem (id_compra_ordem,id_empregado,status,id_fornecedor,type,create_time)" +
                        " VALUES("
                        + po.Id + ","
                        + "'" + po.Id_Empregado + "'" + ","
                        + "'" + po.Status + "'" + ","
                        + "'" + po.Id_Fornecedor + "'" + ","
                        + "'" + po.Type + "'" + ","
                         + "'" + date + "'"
                        + ");";

                    cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
}
```

**Figura 38 - Escrita dos dados na tabela Compra Ordem da base de dados dw_Northwind**

# Purchase Orders Details

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela Compra_ordem_detalhes do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Puchase_Order_details.cs.

```
public static void ReadPurchaseOrdersDetails()
{
    string sql = " SELECT * FROM compra_ordem_detalhes  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos Puchase Orders Details---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Purchase_order_details po = new Purchase_order_details();

            po.Id = (int)dr["id_compra_details"];
            po.Product_id= (int)dr["product_id"];
            po.Quantity = (decimal)dr["quantidade"];
            po.Unit_cost = (decimal)dr["unit_cost"];
            po.Purchase_order_id = (string)dr["compra_id"];
            po.Type = (string)dr["type"];
            po.Create_Time= (DateTime)dr["create_time"];

            ListPurchaseOrderDetails.Add(po);
        }

        Console.WriteLine("Purchase Orders Details lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
    finally
    {
        con.Close();
    }
}
```

Figura 39 - Leitura dos Purchase Order Details

Após o armazenamento dos dados na lista, é feita a inserção dos dados na tabela Compra_Ordem_Detalhes da base de dados *Dw_Northwind*.

```
public static void WritePurchaseOrdersDetails()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Purchase_order_details po in ListPurchaseOrderDetails)
            {
                string date = po.Create_Time.ToString("yyyy-MM-dd HH:mm:ss");

                    cmd.CommandText = "INSERT INTO compra_ordem_detalhes (id_compra_details,product_id,quantidade,unit_cost,compra_id,type,creat
                        " VALUES("
                        + po.Id + ","
                        + po.Product_id + ","
                        + po.Quantity.ToString().Replace(",", ".") + ","
                        + po.Unit_cost.ToString().Replace(",", ".") + ","
                        + "'" + po.Purchase_order_id + "'" + ","
                        + "'" + po.Type + "'" + ","
                         + "'" + date + "'"
                        + ");";

                    cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
}
```

Figura 40 - Escrita dos dados na tabela Compra Ordem Detalhes da base de dados dw_Northwind

## Order Details

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela Ordem_detalhes do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Order_details.cs

```
public static void ReadOrderDetails()
{
    string sql = " SELECT * FROM detalhes_ordem ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos Order Details---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Order_Details od = new Order_Details();

            od.Id = (int)dr["id_ordem_detalhes"];
            od.Product_id = (int)dr["idprodutos"];
            od.Quantity = (decimal)dr["quantidade"];
            od.Unit_price = (decimal)dr["unit_price"];
            od.Type = (string)dr["type"];
            od.Create_Date = (DateTime)dr["create_time"];
            od.Id_Factos = (int)dr["id_factos"];

            ListOrderDetails.Add(od);
        }

        Console.WriteLine("Order details lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
    finally
    {
        con.Close();
    }
}
```

Figura 41 - Leitura dos Order Details

Após o armazenamento dos dados na lista, é feita a inserção dos dados na tabela Ordem_Detalhes da base de dados *Dw_Northwind*.

```
public static void WriteOrdersDetails()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Order_Details od in ListOrderDetails)
            {
                string date = od.Create_Date.ToString("yyyy-MM-dd HH:mm:ss");

                    cmd.CommandText = "INSERT INTO detalhes_ordem (id_ordem_detalhes,id_factos,idprodutos,quantidade,unit_price,type,create_time
                        " VALUES("
                        + od.Id + ","
                         + od.Id_Factos + ","
                        + od.Product_id + ","
                        + od.Quantity.ToString().Replace(",", ".") + ","
                        + od.Unit_price.ToString().Replace(",", ".") + ","
                        + "'" + od.Type + "'" + ","
                         + "'" + date + "'"
                        + ");";

                    cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
    }
```

*Figura 42 - Escrita dos dados na tabela Ordem Detalhes da base de dados dw_Northwind*

# Employees

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela empregados do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Employees.cs.

```
public static void ReadEmployees()
{
    string sql = " SELECT * FROM empregados  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos Empregados---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Employees e = new Employees();

            e.Id = (int)dr["idempregados"];
            e.First_name = (string)dr["first_name"];
            e.Last_name = (string)dr["last_name"];
            e.Company = (string)dr["company"];
            e.Job_title = (string)dr["jobtitle"];
            e.City = (string)dr["city"];
            e.Country = (string)dr["country"];
            e.Type = (string)dr["type"];
            e.Create_Time = (DateTime)dr["create_time"];

            ListEmployees.Add(e);
        }

        Console.WriteLine("Clientes lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
    finally
    {
        con.Close();
    }
```

Figura 41 - Leitura dos Employees

Após o armazenamento dos dados na lista, é feita a inserção dos dados na tabela Ordem_Detalhes da base de dados *Dw_Northwind*.

```
public static void WriteEmployees()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Employees e in ListEmployees)
            {


                string date = e.Create_Time.ToString("yyyy-MM-dd HH:mm:ss");


                    cmd.CommandText = "INSERT INTO empregados (idempregados,first_name,last_name,company,jobtitle,city,country,type,create_time)"
                        " VALUES("
                        + e.Id + ","
                        + "'" + e.First_name + "'" + ","
                        + "'" + e.Last_name + "'" + ","
                        + "'" + e.Company + "'" + ","
                        + "'" + e.Job_title + "'" + ","
                        + "'" + e.City + "'" + ","
                         + "'" + e.Country + "'" + ","
                         + "'" + e.Type + "'" + ","
                         + "'" + date + "'"
                        + ");";

                    cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
        {
```

Figura 42 - Escrita dos dados na tabela Empregados da base de dados dw_Northwind

## Supliers

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela fornecedores do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Supliers.cs.

```
public static void ReadFornecedores()
{
    string sql = " SELECT * FROM fornecedores  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos fornecedores---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Suppliers s = new Suppliers();
             s.Id = (int)dr["id_fornecedor"];
            s.First_name = (string)dr["first_name"];
            s.Last_name = (string)dr["last_name"];
            s.Company = (string)dr["company"];
            s.Job_title = (string)dr["jobtitle"];
            //s.City = (string)dr["city"];
            //s.Country = (string)dr["country"];
            s.Type = (string)dr["type"];
            s.Create_Time = (DateTime)dr["create_time"];

            ListSuppliers.Add(s);
}

        Console.WriteLine("fornecedores lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
    finally
    {
        con.Close();
    }
```

Figura 45 - Leitura dos Supliers

Após o armazenamento dos dados na lista, é feita a inserção dos dados na tabela Fornecedores da base de dados *Dw_Northwind*.

```
public static void WriteFornecedores()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Suppliers s in ListSuppliers)
            {

                string date = s.Create_Time.ToString("yyyy-MM-dd HH:mm:ss");


                    cmd.CommandText = "INSERT INTO fornecedores (id_fornecedor,first_name,last_name,company,jobtitle,type,create_time)" +
                        " VALUES("
                        + s.Id + ","
                        + "'" + s.First_name + "'" + ","
                        + "'" + s.Last_name + "'" + ","
                        + "'" + s.Company + "'" + ","
                        + "'" + s.Job_title + "'" + ","
                       /* + "'" + s.City + "'" + ","
                         + "'" + s.Country + "'" + ","*/
                         + "'" + s.Type + "'" + ","
                         + "'" + date + "'"
                        + ");";

                    cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
```

*Figura 46 - Escrita dos dados na tabela Fornecedores da base de dados dw_Northwind*

# Orders

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela ordem do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Orders.cs.

```
public static void ReadOrders()
{
    string sql = " SELECT * FROM ordem  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos Orders---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Orders o = new Orders();

            o.Id = (int)dr["id_ordem"];
            o.Employee_id = (int)dr["id_empregado"];
            o.Customer_id = (int)dr["id_cliente"];
            o.Status_id = (int)dr["status_id"];
            o.Type = (string)dr["type"];
            o.Created_Time= (DateTime)dr["create_time"];
            o.Order_date = (DateTime)dr["data_ordem"];
            o.Paid_date= (DateTime)dr["data_paid"];
            o.Shipped_date = (DateTime)dr["data_shipped"];
            o.Payment_type = (string)dr["payment_type"];
            o.Id_remetente = (int)dr["id_remetente"];

            ListOrders.Add(o);
        }

        Console.WriteLine("Orders lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
```

Figura 43 - Leitura das Orders

Após o armazenamento dos dados na lista, é feita a inserção dos dados na tabela Ordem da base de dados Dw_Northwind.

```
public static void WriteOrders()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Orders o in ListOrders)
            {

                string created_time = o.Created_Time.ToString("yyyy-MM-dd HH:mm:ss");
                string order_date=o.Order_date.ToString("yyyy-MM-dd HH:mm:ss");
                string paid_date = o.Paid_date.ToString("yyyy-MM-dd HH:mm:ss");
                string shipped_date = o.Shipped_date.ToString("yyyy-MM-dd HH:mm:ss");



                    cmd.CommandText = "INSERT INTO ordem (id_ordem,id_empregado,id_cliente,status_id,type," +
                    "create_time,data_ordem,data_paid,data_shipped,id_remetente,payment_type)" +
                        " VALUES("
                        + o.Id + ","
                        + o.Employee_id + ","
                        + o.Customer_id + ","
                         + o.Status_id + ","
                         + "'" + o.Type + "'" + ","
                         + "'" + created_time + "',"
                         + "'" + order_date + "',"
                         + "'" + paid_date + "',"
                         + "'" + shipped_date + "' ,"
                          + o.Id_remetente + ","
                          + "'" + o.Payment_type + "'"
                        + ");";

                    cmd.ExecuteNonQuery();
```

Figura 44 - Escrita dos dados na tabela Ordem da base de dados dw_Northwind

# Products

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela produtos do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Products.cs

```
public static void ReadProducts()
{
    string sql = " SELECT * FROM produtos  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos Produtos---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Products p = new Products();

            p.Id= (int)dr["id_produto"];
            p.Category = (string)dr["category"];
            p.Product_name= (string)dr["nome"];
            p.Type = (string)dr["type"];
            p.Quantity = (decimal)dr["quantidade"];
            p.Standard_cost = (decimal)dr["standard_cost"];
            p.List_price = (decimal)dr["list_price"];
            p.Create_Date = (DateTime)dr["create_time"];


            ListProducts.Add(p);
        }

        Console.WriteLine("Produtos lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
```

Figura 49 - Leitura dos dados na tabela Ordem da base de dados Northwind

Após o armazenamento dos dados na lista, é feita a inserção dos dados na tabela Produtos da base de dados Dw_Northwind.

```
public static void WriteProducts()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Products p in ListProducts)
            {

                string created_time = p.Create_Date.ToString("yyyy-MM-dd HH:mm:ss");



                cmd.CommandText = "INSERT INTO produtos (id_produto,category,nome,standard_cost,quantidade,list_price,type,create_time)" +
                    " VALUES("
                    + p.Id + ","
                    + "'" + p.Category + "',"
                     + "'" + p.Product_name + "',"
                     + p.Standard_cost.ToString().Replace(",", ".") + ","
                     + p.Quantity.ToString().Replace(",", ".") + ","
                     + p.List_price.ToString().Replace(",", ".") + ","
                     + "'" + p.Type + "'" + ","
                     + "'" + created_time + "'"

                    + ");";

                cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
        {
```

Figura 50 - Escrita dos dados na tabela Produtos da base de dados dw_Northwind

# Shippers

A figura abaixo demonstra o código desenvolvido para procedermos a leitura dos dados da tabela Remetente do schema Northwind.

Estes valores após serem lidos são armazenados numa lista de elementos do tipo Shippers.cs.

```
public static void ReadRemetende()
{
    string sql = " SELECT * FROM Remetente  ";
    MySqlConnection con = new MySqlConnection(connectionstring);
    MySqlCommand cmd = new MySqlCommand(sql, con);
    Console.WriteLine("-----Leitura dos Remetentes---------\n");
    con.Open();
    try
    {
        MySqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            Shippers s = new Shippers();

            s.Id = (int)dr["id_remetente"];
            s.First_name = (string)dr["first_name"];
            s.Last_name = (string)dr["last_name"];
            s.Company = (string)dr["company"];
           // s.Job_title = (string)dr["jobtitle"];
            s.City = (string)dr["city"];
            s.Country = (string)dr["country"];
            s.Type = (string)dr["type"];
            s.Create_Time = (DateTime)dr["create_time"];



            ListShippers.Add(s);
        }

        Console.WriteLine("Remetentes lidos com sucesso!!\n");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Erro:" + ex.Message);
    }
```

Figura 50 - Leitura dos Shippers

Após o armazenamento dos dados na lista, é feita a inserção dos dados na tabela Remetente da base de dados Dw_Northwind.

```
public static void WriteRemetente()
{
    using (MySqlConnection connection = new MySqlConnection(connectionstring2))
    {
        MySqlCommand cmd = new MySqlCommand();
        cmd.Connection = connection;
        try
        {
            connection.Open();

            foreach (Shippers s in ListShippers)
            {

                string date = s.Create_Time.ToString("yyyy-MM-dd HH:mm:ss");

                    //jobtitle
                    cmd.CommandText = "INSERT INTO remetente (id_remetente,first_name,last_name,company,city,country,type,create_time)" +
                        " VALUES("
                        + s.Id + ","
                        + "'" + s.First_name + "'" + ","
                        + "'" + s.Last_name + "'" + ","
                        + "'" + s.Company + "'" + ","
                        + "'" + s.City + "'" + ","
                         + "'" + s.Country + "'" + ","
                         + "'" + s.Type + "'" + ","
                         + "'" + date + "'"
                        + ");";

                    cmd.ExecuteNonQuery();


            }
            connection.Close();

        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
```

**Figura 51 - Escrita dos dados na tabela Remetente da base de dados dw_Northwind**

# 5. Carregamento dos dados para o Data Warehouse

Após os dados já se encontrarem na área de retenção procedemos a sua limpeza e transformação para um posterior carregamento deles para o data warehouse. Para executar esta tarefa implementamos triggers que conseguissem efetuar esses processos, eliminar nulos, juntar dados, entre outros.

**Trigger criado na área de retenção:**

```
IF comp is null then
SET comp := 'Undefined';
END IF;

IF ci is null then
SET ci := 'Undefined';
END IF;

IF co is null then
SET co := 'Undefined';
END IF;

IF new.type = 'I'then

insert into dw_northwind.dim_customers(company,name,jobtitle,city,country,is_current,created_date,expired_date)
values(comp,concat(first,_utf8' ',last),job,ci,co,'yes',dat,null);

else

IF new.type ='U' then

insert into dw_northwind.dim_customers(company,name,jobtitle,city,country,is_current,created_date,expired_date)
values(comp,concat(first,_utf8' ',last),job,ci,co,'yes',dat,null);

Update dw_northwind.dim_customers set expired_date=Now(),is_current='no' where iddim_customers=new.idcliente;
else
Update dw_northwind.dim_customers set expired_date=Now(),is_current='no' where iddim_customers=new.idcliente;
END IF;
END IF;
UD#
```

Figura 45 - Trigger do cliente

Esta é a estrutura que seguimos para todos os triggers implementados, não os colocamos aqui todos pois são bastante extensos, mas seguem em anexo com o trabalho.

# 6. Indicadores de Business Intelligence

Para apresentar os dados a empresa de forma a que esta possa ter suporte às suas tomadas de decisões procedemos a criação de indicadores de Business Intelligence, utilizamos a ferramenta de Power BI. No total criamos 8 *dashboards* que serão apresentadas em seguida. É de salientar que estes indicadores tornam-se muito úteis para uma análise fácil, rápida e também muito interativa, o que irá permitir o tal suporte à toma de decisões.

## Dashboard1

A primeira *dashboard* é utilizada para apresentar informações relativas as encomendas.

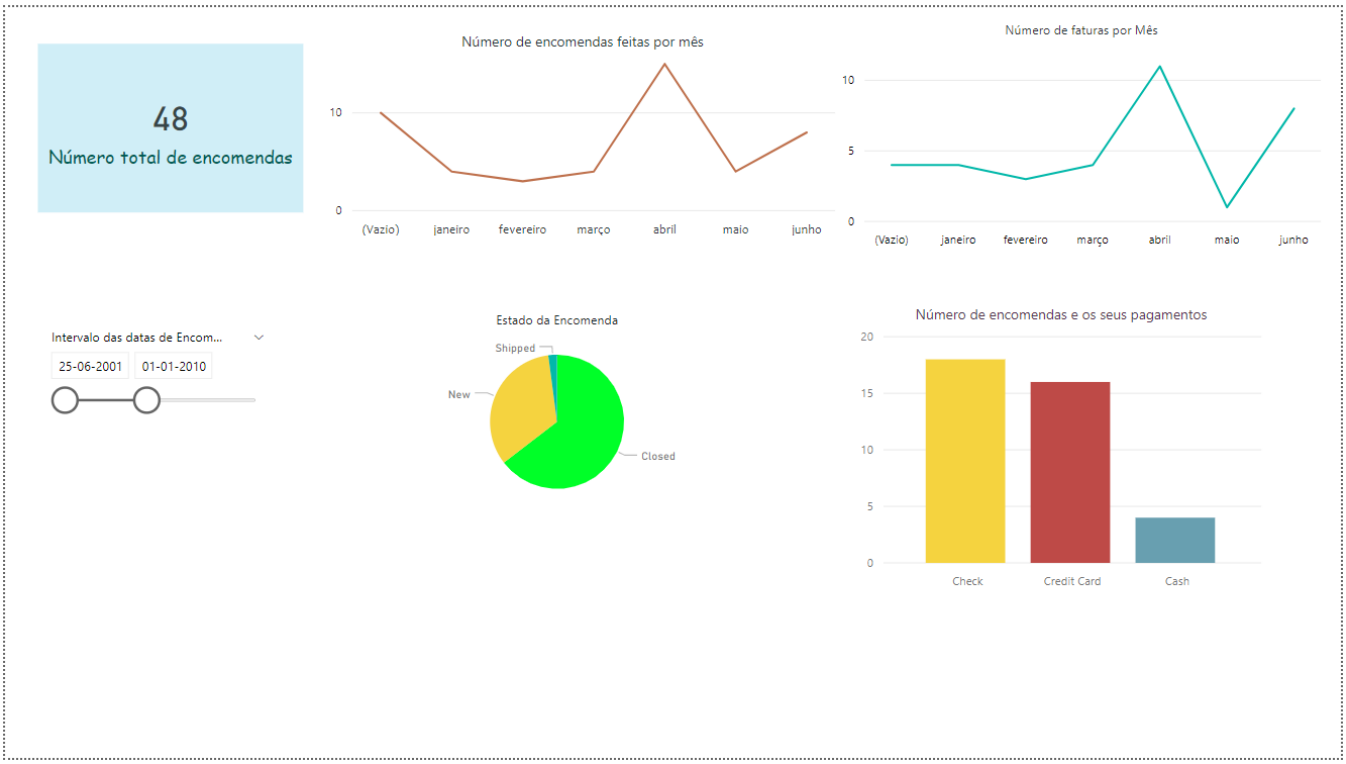



Figura 46 - Dashboard 1

Nesta página foram utilizadas diversas formas de visualizar os dados. Um cartão para apresentar o número total de encomendas, gráficos de linhas onde é exibido o número de encomendas feitas por cada mês e um outro a apresentar o número de faturas imitidas por mês. Um gráfico de colunas empilhadas para exibir o número de encomendas e os seus pagamentos e um gráfico circular para apresentar o número de encomendas em cada estado. Por fim utilizamos a segmentação dos dados para controlar o limite do intervalo entre as datas.

Figura 47 - Indicador 1

Este indicador ilustra o número total de encomendas. É bastante útil para, por exemplo, um administrador do sistema da empresa saber quantas encomendas já foram feitas na totalidade. Se a empresa está a ter um bom número de pedidos ou se pode ser um fator com que a empresa se tenha a preocupar.

*Figura 48 - Indicador 2*

O indicador 2 apresenta o número de encomendas feitas por mês. Este pode vir a ser bastante informacional na medida em que a empresa pode querer saber em que mês se verifica terem mais encomendas para, por exemplo, contratar mais funcionários e despachar as encomendas mais cedo.

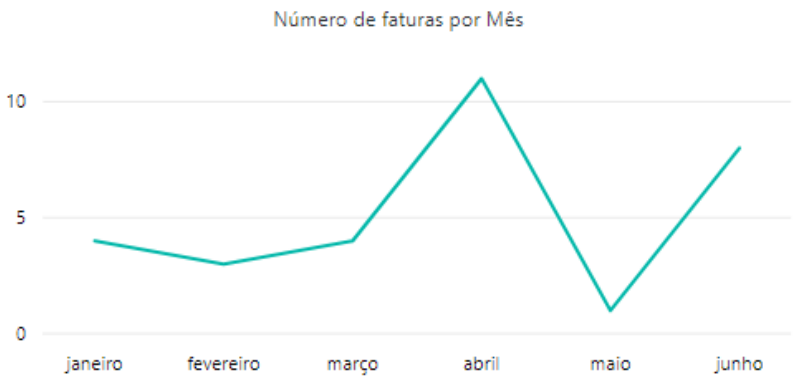



Figura 49 - Indicador3

Este indicador exibe o número de faturas emitidas por mês. Isto pode ser útil para os administradores da empresa terem noção de quantas faturas efetuadas e se estão de acordo com o número de encomendas entregues.

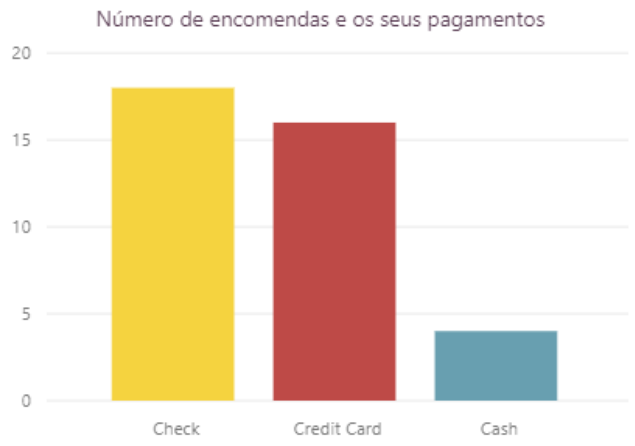



Figura 50 - Indicador4

O indicador 4 apresenta qual a forma de pagamento tendo em conta o número total de encomendas. Este indicador pode ser utilizado para a empresa ter a percepção de qual costuma ser o meio preferido dos clientes para efetuar o pagamento. A empresa pode utilizar está informação estrategicamente na medida em que poderá disponibilizar a aceitação de mais cartões de créditos e cheques de diferentes bancos.

Figura 51 - Indicador5

O indicador 5 demonstra o estado do número total das encomendas. Por ser tão intuitivo possibilita s4os administradores da empresa ver o estado das encomendas e ter uma percepção da quantidade de novas encomendas, das que já foram efetuadas e das que foram enviadas.

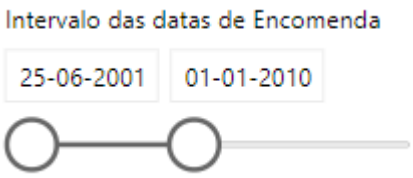



Figura 52 - Indicador6

Por último este indicador permite delimitar as datas que queremos que os dados apareçam. Este é ótimo para a empresa conseguir ver dados de um determinado ano ou de vários.

## Dashboard2

Esta dashboard exibe informações relacionada com os clientes.

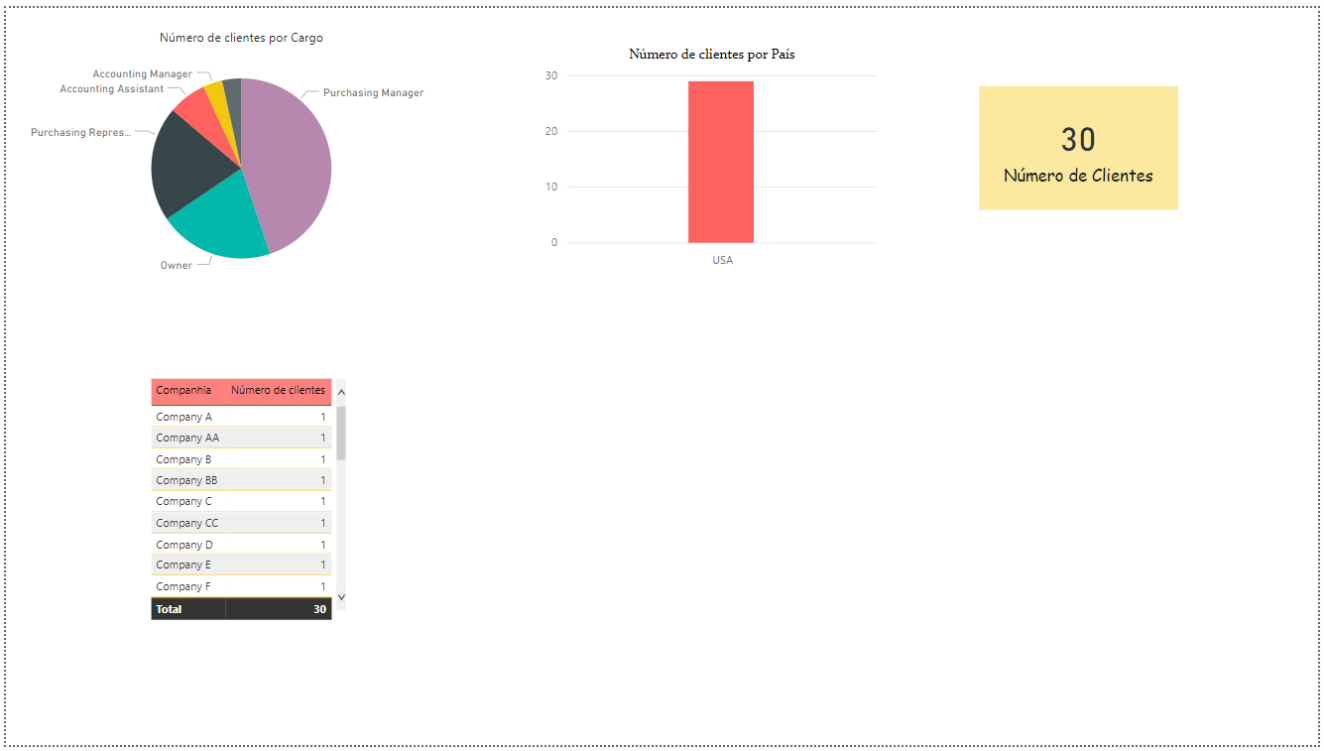



Figura 53 - Dashboard2

Nesta dashboard mais uma vez foram utilizadas diversas formas de visualização dos dados. Temos a utilização de um gráfico circular para ilustrar o número de clientes por cargo, um gráfico de colunas empilhadas para apresentar o número de clientes por país, um cartão para demostrar o número total de clientes da empresa e por último uma tabela para mostrar o número de clientes por companhia.

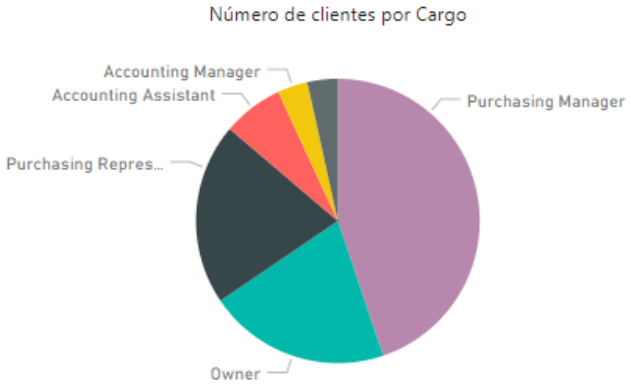



Figura 54 - Indicador7

Este indicador apresenta o número de clientes por cargo. Pode ser útil para a empresa saber qual é o cargo dos clientes que mais utilizam os seus serviços.

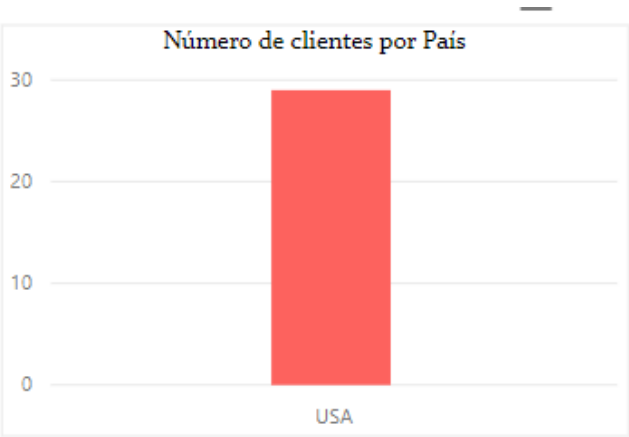



Figura 55 - Indicador8

O indicador 8 apresenta o número de clientes por país. É bastante útil para a empresa ter noção de qual é o país que mais utiliza os seus serviços, podendo tentar realizar campanhas para tentar conquistar clientes em por exemplo, países que ainda não tem conhecimento da empresa.

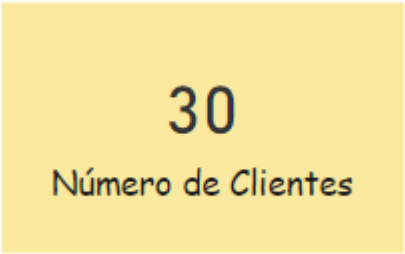



Figura 56 - Indicador9

Este indicador exibe o número total de clientes na empresa. É útil para a empresa ter noção do número total dos seus clientes de forma a verificar se a empresa tem um bom número de clientes ou se terão que fazerem mais marketing para conquistar mais clientes.

| Companhia | Número de clientes |
|---|---|
| Company AA | 1 |
| Company B | 1 |
| Company BB | 1 |
| Company C | 1 |
| Company CC | 1 |
| Company D | 1 |
| Company E | 1 |
| Company F | 1 |
| Company G | 1 |
| **Total** | **30** |

Figura 57 - Indicador10

Por último, este indicador ilustra o número de clientes por companhia. Pode ser utilizado para os administradores da empresa terem noção de qual é a companhia que dá mais uso aos serviços dela.

## Dashboard3

Esta dashboard diz respeito aos funcionários da empresa.

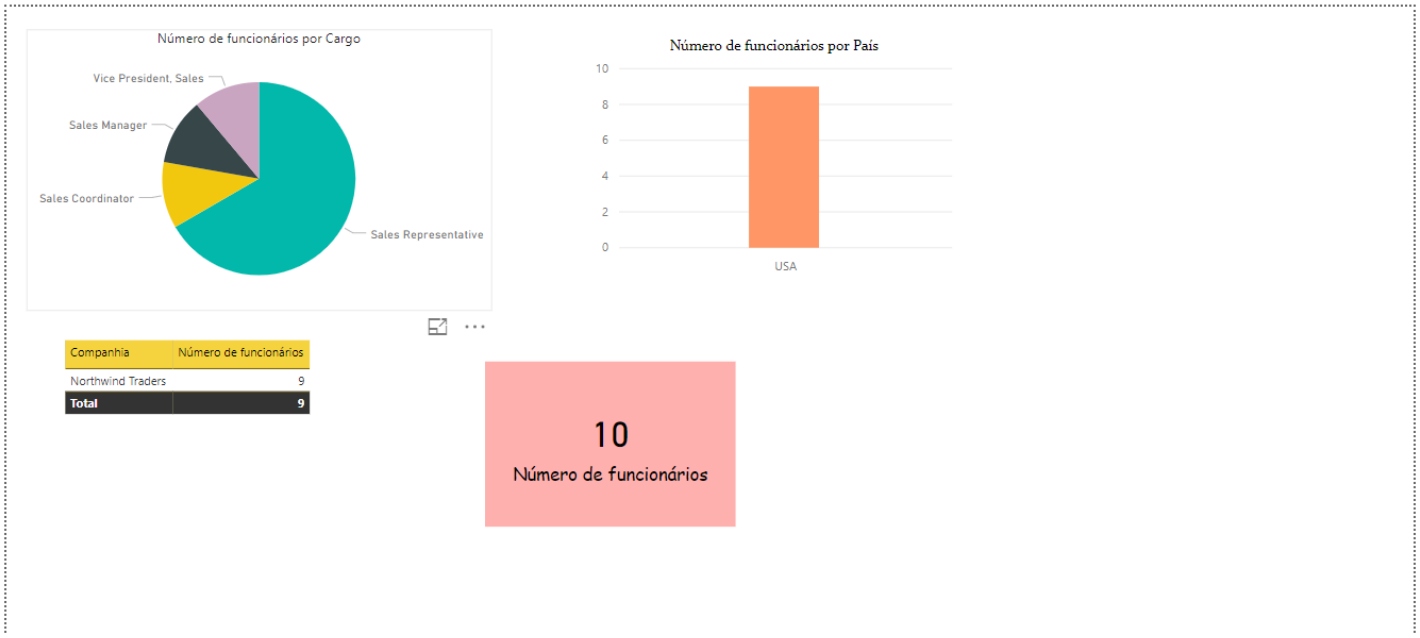



Figura 58 - Dashboard3

As formas de visualização utilizadas nesta dashboard são o gráfico circular, que indica o número de funcionários por cargo, um gráfico de colunas empilhadas, que indica o número de funcionários por país, uma tabela que mostra o número de funcionários por companhia e um cartão que exibe o número total de funcionários.

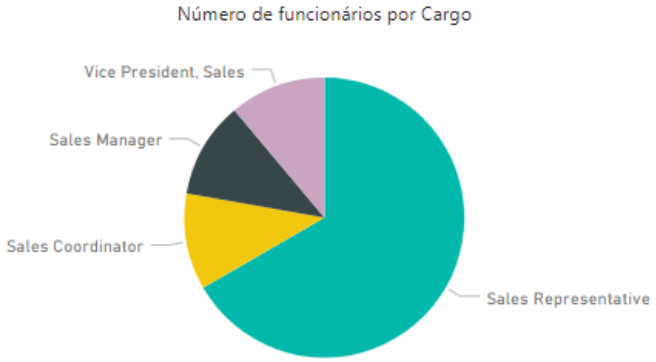



Figura 59 - Indicador11

Neste indicador é possível visualizar o número de funcionários por cargo. Isto é bastante útil para a empresa saber quais os cargos que tem mais funcionários e, por exemplo, caso apareçam pessoas a pedir emprego o administrador saiba onde o poderá colocar.

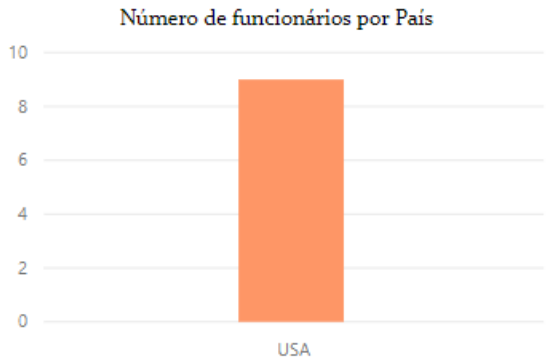



Figura 60 - Indicador12

O indicador 12 exibe o número de funcionários por país. Este pode ser útil por exemplo para uma pesquisa para um estudo em que seja necessário saber se a empresa contrata mais de países estrangeiros ou não.

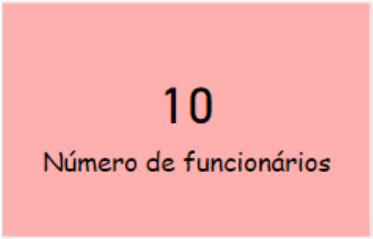



Figura 61 - Indicador13

O indicador 13 ilustra o número de funcionários total da empresa. Este pode ser utilizado para questões estatísticas por exemplo.

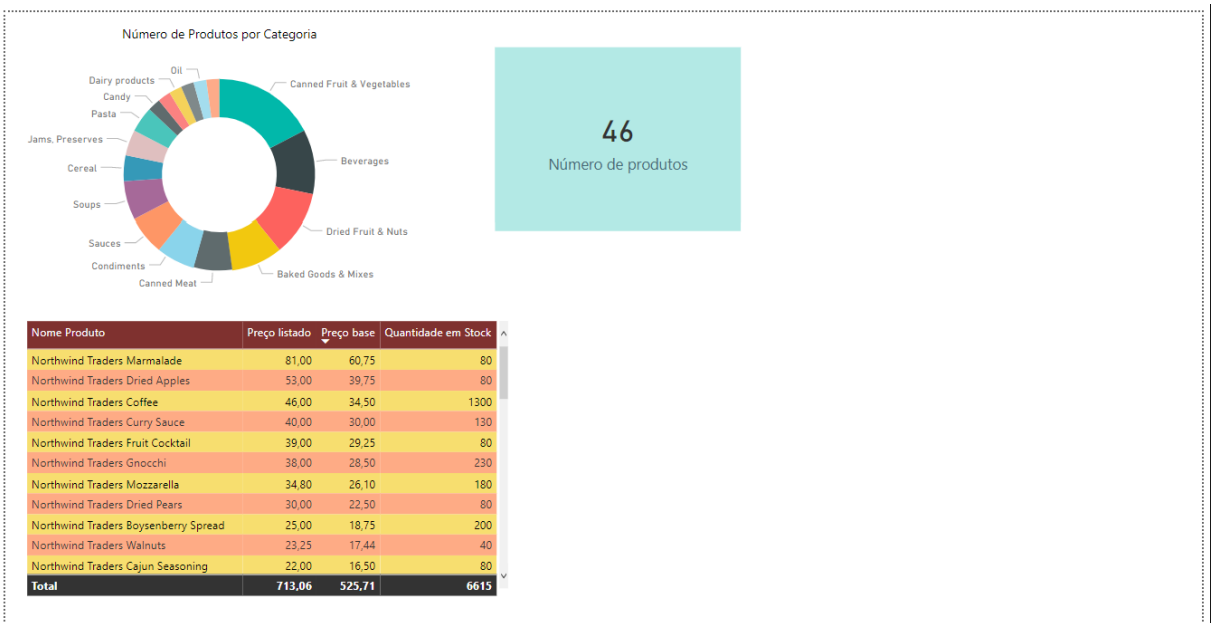
| Companhia | Número de funcionários |
|---|---|
| Northwind Traders | 9 |
| **Total** | **9** |

Figura 62 - Indicador14

Este indicador apresenta o número de funcionários por companhia. Isto pode permitir a empresa saber qual a companhia que dispõe de mais funcionários. Também pode ser utilizado por questões estatísticas.

## Dashboard4

Esta dashboard diz respeito ao produto.




| Nome Produto | Preço listado | Preço base | Quantidade em Stock |
|---|---|---|---|
| Northwind Traders Marmalade | 81,00 | 60,75 | 80 |
| Northwind Traders Dried Apples | 53,00 | 39,75 | 80 |
| Northwind Traders Coffee | 46,00 | 34,50 | 1300 |
| Northwind Traders Curry Sauce | 40,00 | 30,00 | 130 |
| Northwind Traders Fruit Cocktail | 39,00 | 29,25 | 80 |
| Northwind Traders Gnocchi | 38,00 | 28,50 | 230 |
| Northwind Traders Mozzarella | 34,80 | 26,10 | 180 |
| Northwind Traders Dried Pears | 30,00 | 22,50 | 80 |
| Northwind Traders Boysenberry Spread | 25,00 | 18,75 | 200 |
| Northwind Traders Walnuts | 23,25 | 17,44 | 40 |
| Northwind Traders Cajun Seasoning | 22,00 | 16,50 | 80 |
| **Total** | **713,06** | **525,71** | **6615** |

Figura 63 - Dasboard4

Nesta dashboard utilizamos a visualização do gráfico em anel, para ilustrar o número de produtos por categoria, um cartão para exibir o número total de produtos da empresa e por fim uma tabela onde é apresentado informações relativas ao produto.

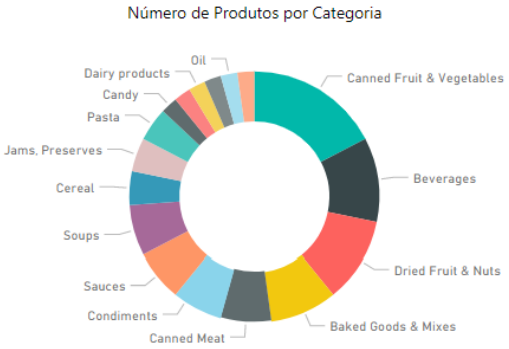



Figura 64 - Indicador15

Este indicador apresenta o número de produtos por categoria. Esta informação pode ser útil para informar os administradores do tipo de produtos existentes e do número deles. Para assim saber quais é que têm que encomendar e assim.

Figura 65 - Indicador16

Mais uma vez esta visualização surge para apresentar o número total de produtos da empresa. Pode ser utilizada por questões estatísticas ou simplesmente para ter noção de qunatos produtos diferentes existem na empresa.

| Nome Produto | Preço listado | Preço base | Quantidade em Stock |
|---|---|---|---|
| Northwind Traders Marmalade | 81,00 | 60,75 | 80 |
| Northwind Traders Dried Apples | 53,00 | 39,75 | 80 |
| Northwind Traders Coffee | 46,00 | 34,50 | 1300 |
| Northwind Traders Curry Sauce | 40,00 | 30,00 | 130 |
| Northwind Traders Fruit Cocktail | 39,00 | 29,25 | 80 |
| Northwind Traders Gnocchi | 38,00 | 28,50 | 230 |
| Northwind Traders Mozzarella | 34,80 | 26,10 | 180 |
| Northwind Traders Dried Pears | 30,00 | 22,50 | 80 |
| Northwind Traders Boysenberry Spread | 25,00 | 18,75 | 200 |
| Northwind Traders Walnuts | 23,25 | 17,44 | 40 |
| Northwind Traders Cajun Seasoning | 22,00 | 16,50 | 80 |
| **Total** | **713,06** | **525,71** | **6615** |

Figura 66 - Indicador17

Este indicador apresenta diversas informações do produto, o nome, o preço que está listado, o preço base e a quantidade em stock. É de extrema utilidade para a empresa para saber se lucra com a venda do produto, se sim o quanto e também permite ter sempre uma visão do stock permitindo assim que esta nunca se verifique ficar vazio pois a empresa pode efetuar mais encomendas para os repor.

## Dashboard5

Esta dashboard diz respeito a ordem de compras.

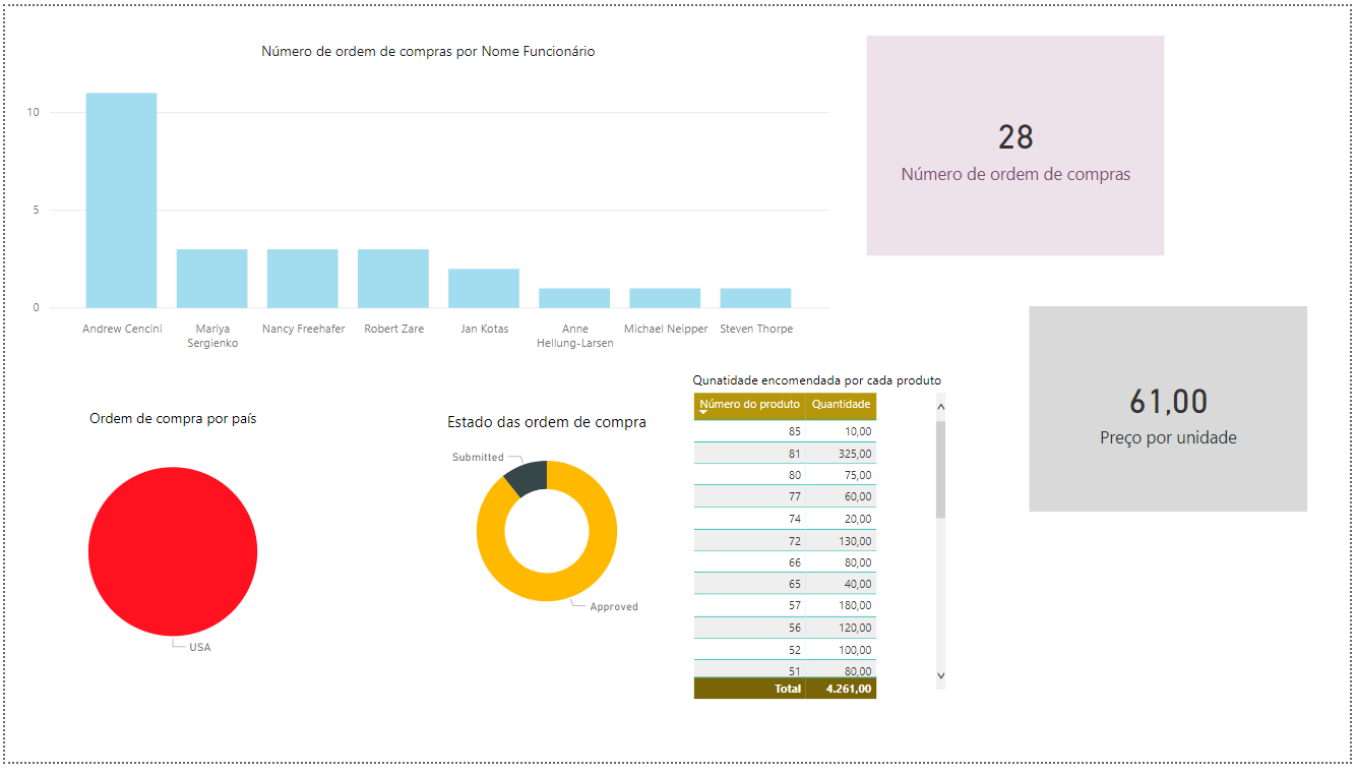



**Figura 67 - Dashboard5**

Nesta dashboard tal como as outros diversos modos de visualização são utilizados tais como o gráfico de colunas empilhadas, que apresenta o número de ordem de compras por nome de funcionário. É utilizado dois cartões um para apresentar o número de ordem de compras e outro que exibi o preço por unidade do produto. Utiliza uma tabela para apresentar o número de produto por quantidade numa compra. Um gráfico circular para indicar o número de ordem de compras por país. E um gráfico em anel para apresentar o estado da ordem das compras.

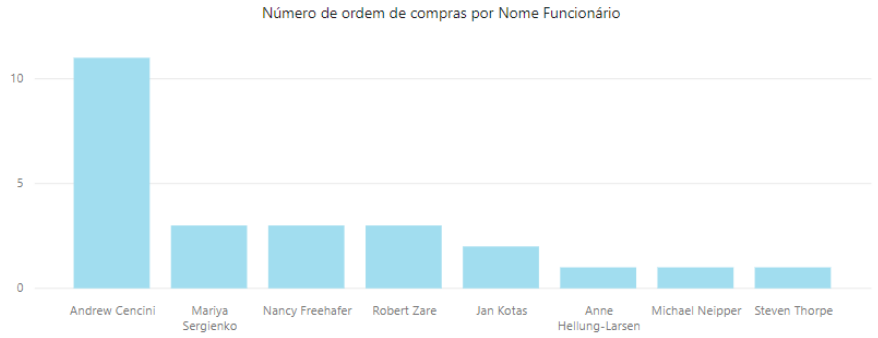



**Figura 68 - Indicador18**

Este indicador exibe o número de ordem de compras por nome de funcionário. É útil para a empresa saber qual é o funcionário que criou mais compras e por exemplo ao fim do mês pode dar um bônus a quem registra mais compras.

*Figura 69 - Indicador19*

Este indicador apresenta o número de compras por país. É útil para estatísticas por exemplo saber qual o paí s que efetua mais compras.

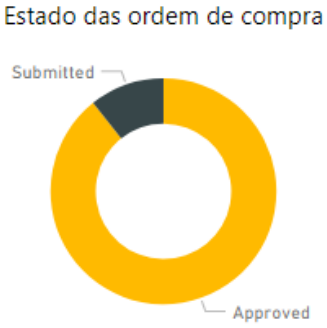



**Figura 70 - Indicador20**

Este indicador exibe o estado das ordem de compras. É bastante útil pois de forma rápida e simples a empresa sabe a quantidade do número de compras em cada estado.

Qunatidade encomendada por cada produto

| Número do produto | Quantidade |
|---|---|
| 85 | 10,00 |
| 81 | 325,00 |
| 80 | 75,00 |
| 77 | 60,00 |
| 74 | 20,00 |
| 72 | 130,00 |
| 66 | 80,00 |
| 65 | 40,00 |
| 57 | 180,00 |
| 56 | 120,00 |
| 52 | 100,00 |
| 51 | 80,00 |
| **Total** | **4.261,00** |

**Figura 71 - Indicador21**

O indicador 21 permite verificar a quantidade de cada produto encomendado. Pode ser usado para questões estatísticas para saber qual o produto que vende mais.

**Figura 72 - Indicador22 e 23**

Estes indicadores ilustram o número de ordem de compras e o preço por unidade de cada produto, respetivamente. O primeiro pode ser útil para questões estatísticas para saber se têm bastantes número de ordem de compras e o segundo é vantajoso pois permite visualizar de forma rápida o preço de cada produto por unidade.

# Dashboard6

Esta dashboard diz respeito aos fornecedores.

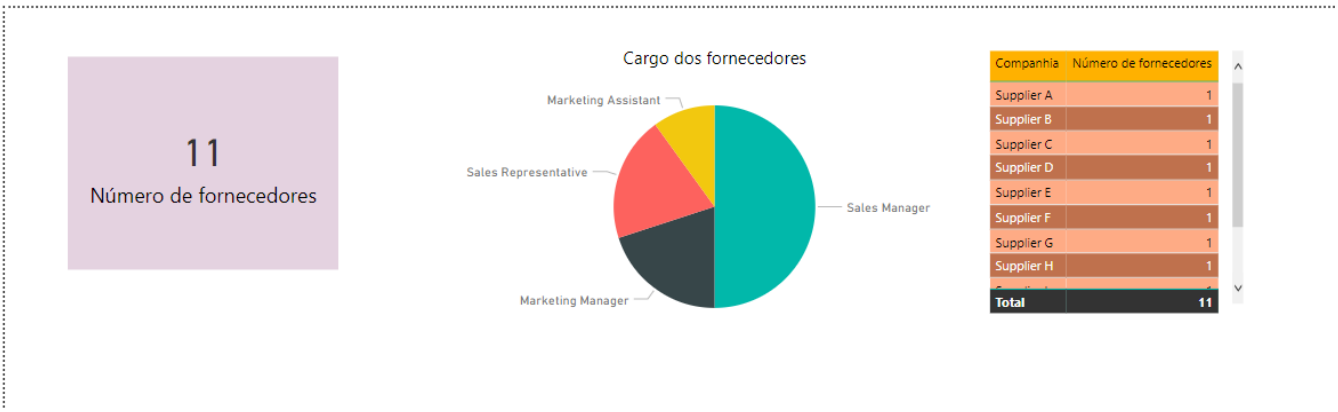



**Figura 73 - Dashboard6**

As visualizações presentes nesta dashboard são mais uma vez o cartão, apresenta o número de fornecedores, o gráfico circular que apresenta o cargo dos funcionários e por último uma tabela que ilustra o número de fornecedores por companhia.

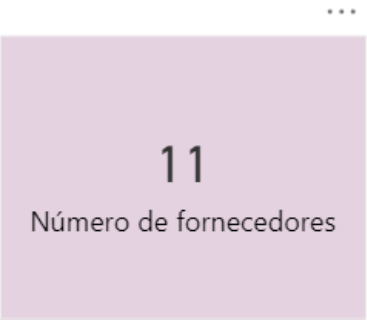



***Figura 74 - Indicador 24***

Este indicador dá a informação do número de fornecedores da empresa. Pode ser útil por questões estáticas.

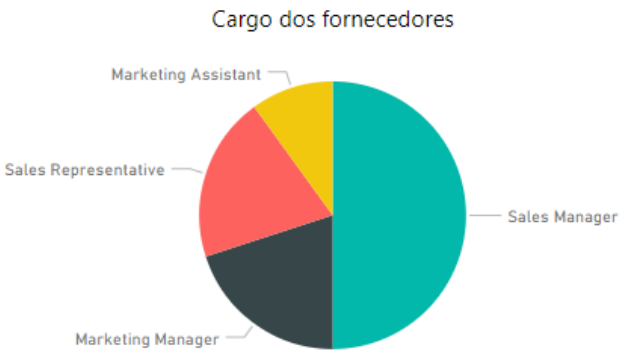



Figura 75 - Indicador25

O indicador 25 dá-nos a informação do cargo dos funcionários da empresa. Este pode ser útil pois o administrador da empresa pode querer saber quando fornecedores tem de cada cargo e realizar contratações para cargos que têm menos.

| Companhia | Número de fornecedores |
|---|---|
| Supplier A | 1 |
| Supplier B | 1 |
| Supplier C | 1 |
| Supplier D | 1 |
| Supplier E | 1 |
| Supplier F | 1 |
| Supplier G | 1 |
| Supplier H | 1 |
| Total | 11 |

Figura 76 - Indicador26

O indicador 26 informa do número de fornecedores por companhia. É útil para realizar estatísticas, saber qual é a companhia que mais fornecedores tem na empresa.

## Dashboard7

Esta dasboard diz respeito ao remetente.

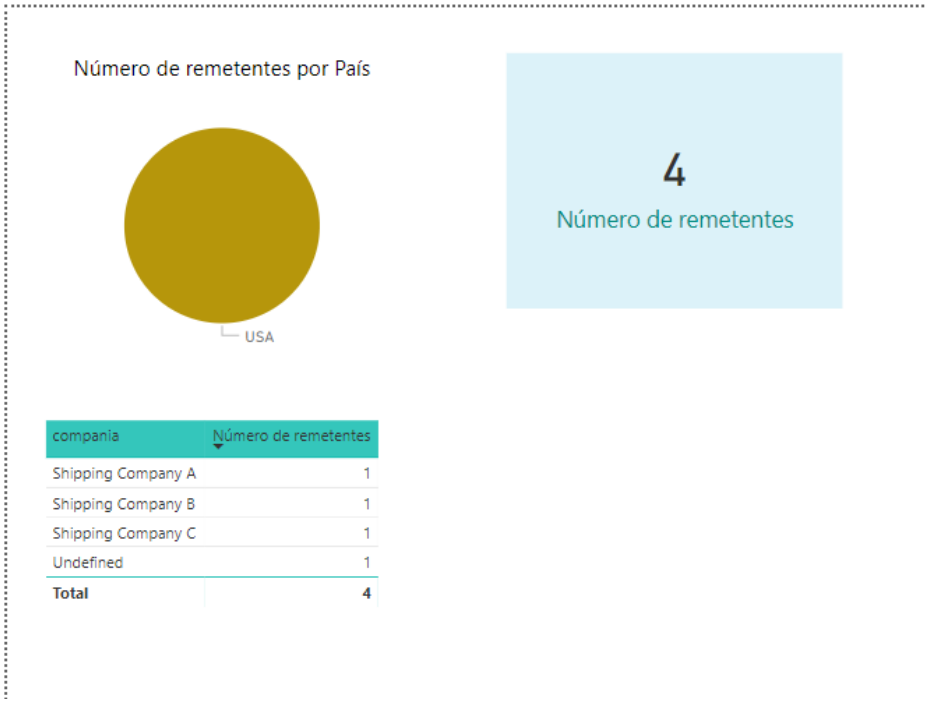



Figura 77 - Dashboard7

As visualizações utilizadas neste dashboard são o gráfico circular, apresenta o número de remetentes por país, o cartão, apresenta o número de remetentes e uma tabela que apresenta o número de remetentes por companhia.

Figura 78 - Indicador27

Este indicador exibe o número de remetentes por país. Pode ser bastante útil para questões estatísticas, ou seja, saber qual o país que apresenta mais remetentes.

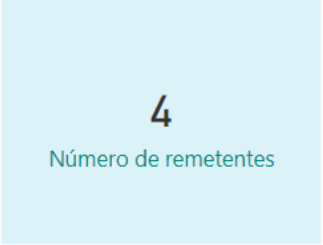



Figura 79 - Indicador28

O indicador 28 permite saber o número total de remetentes da empresa. Mais uma vez pode ser útil para questões estatísticas ou mesmo para a empresa ter noção se tem um número de remetentes necessários para a empresa ter futuro.

| compania | Número de remetentes |
|---|---|
| Shipping Company A | 1 |
| Shipping Company B | 1 |
| Shipping Company C | 1 |
| Undefined | 1 |
| **Total** | **4** |

Figura 80 - Indicador29

Por último este apresenta o número de remetentes por companhia. Tais como os já apresentados anteriormente este pode ser utilizado para fins estatísticos, saber qual a companhia que tem mais remetentes.

## Dashboard8

Figura 81 - Dashboard8

Por último esta da utiliza visualizações de gráfico circular para exibir o número da semana que se enviam mais encomendas, dois gráficos de linhas para apresentar o número de encomendas pagas por mês e o número de encomendas enviadas por mês. Mais uma vez apresenta a segmentação de dados para permitir limitar um intervalo de datas.

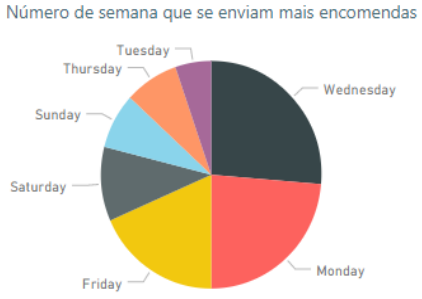



**Figura 82 - Indicador30**

Este indicador apresenta o nome da semana que se verificam mais encomendas. Este pode ser útil para o administrador da empresa contratar mais gente nos dias de semana que se verificam haver mais encomendas, e assim há um despacho do trabalho maior nesses dias.

**Figura 83 - Indicador31**

O indicador 31 exibe o número de encomendas pagas por mês. Este indicador é útil para a empresa conseguir fazer uma estatística sobre se o valor ganho em cada mês bate certo com o número de encomendas pagas.

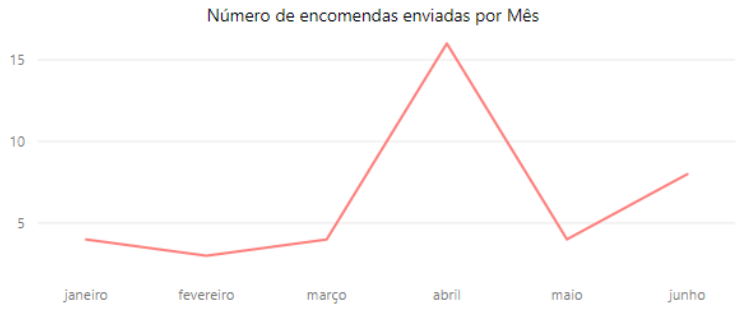



**Figura 84 - Indicador 32**

O indicador 32 ilustra o número de encomendas enviadas por mês. Mais uma vez este torna-se útil para saber quais os meses que os administradores da empresa têm que ter mais gente a trabalhar para despachar as encomendas mais cedo. Sendo que este indicador é utilizado para estatísticas.

# 7.Conclusão

A falta de informações para gerir qualquer atividade empresarial está a tornar-se um dos fatores mais críticos relacionados à sobrevivência das empresas.

Para combater este problema sugiram os *Data Warehouse*. A sua idealização e implementação é de extrema importância ao suporte de *Business Intelligence* para auxiliar na tomada de decisão dentro de uma organização, na medida em que estas ficam munidas de um sistema capaz de dar uma visão histórica a nível de cada processo de negócio ou se necessário da empresa no seu todo. A capacidade de se efetuarem análises mais rápidas e flexíveis.

Para além disso, destaca-se ainda a centralização dos dados num só sistema assim como a possibilidade de extrair informação a partir do que eram apenas dados soltos pelos diversos sistemas operacionais.