



## **Universidade de Aveiro**

Departamento de Eletrónica, Telecomunicações e Informática

## **Bases de Dados – Ano Letivo 2018-2019**

Relatório informativo sobre a realização do Trabalho Prático Final

### **Nome do Projeto**

NutriConvert

### **Projeto Realizado por:**

- 84734 – Fábio Daniel Ribeiro Alves
- 84831 – Sérgio Gabriel Pacheco de Aguiar

# Introdução ao Projeto

Com o passar do tempo, o mundo foi evoluindo e cada vez mais dependendo de novas e mais sofisticadas tecnologias. Contudo, algo que sempre se manteve importante foi a necessidade de armazenar informação.

Desde escrita em rochas, pergaminhos e folhas de papel até ao formato digital, o armazenamento de informação foi desde sempre crucial à progressão do mundo e, nos dias de hoje, não existem exceções. Vários métodos de armazenamento foram desenvolvidos ao longo do tempo e, entre eles, bases de dados são dos mais cruciais.

Existem vários tipos de Bases de Dados dos quais Relacional é um exemplo e também o foco do projeto em questão.

Uma empresa aveirense de tratamento e armazenamento de produtos alimentares tem, ao longo do último ano e meio, convertido os seus ficheiros Microsoft Excel em Tabelas de uma Base de Dados Relacional. Contudo, uma porção destes ficheiros detinha uma estrutura incomum, o que resultou na falta de capacidade de sua transformação em tabelas da base de dados previamente mencionada.

Dado o conhecimento deste problema, o grupo realizador deste projeto decidiu ajudar a empresa a conceber uma estrutura de base de dados que resolvesse o problema de forma eficiente e que fosse compatível com a atualmente existente base de dados da empresa.

Adicionalmente, o grupo também iniciou o desenvolvimento de uma aplicação que permitisse realizar as várias operações vitais de acesso à base de dados, para demonstrar a eficácia da solução proposta.

## Tecnologias utilizadas no decorrer do projeto

- Microsoft SQL Server 2017
- Microsoft SQL Server Management Studio V17.9.1
- Microsoft Visio Professional 2019
- Microsoft Visual Studio Community 2017
- Visual Paradigm V15.1
- Linguagens: C#, SQL

# Análise de Requisitos

Dada a natureza do trabalho realizado, foi necessário reunir com representantes tanto dos laboratórios como dos escritórios da empresa em questão para se poderem extrair os requisitos necessários ao correto desenvolvimento de uma solução ao problema obtido.

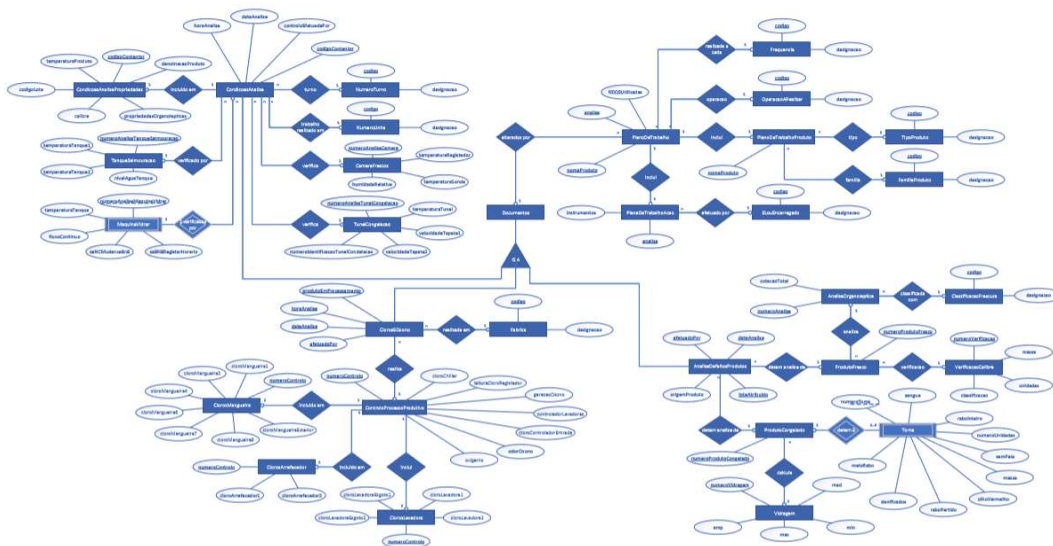
## **Os requisitos obtidos inicialmente foram os seguintes:**

- No Plano de Trabalho são registados os produtos a serem tratados, identificados por um nome, tipo de produto, a sua família, a operação a realizar sobre eles, a análise a realizar, quem realiza a análise e a frequência a que essa análise será executada.
- Os tipos de produto existentes são: Matéria-prima, identificado por “MP”, Produto Final, identificado por “PF”, ou até ambos, identificado por “MP/PF”.
- Os tipos de família de produtos existentes são: Hortícola (H), Pescado (P), Crustáceo (C), Farinha (F), Óleo (O), Reagentes Químicos (R) e Sal (S).
- Os tipos de operação existentes são: Embalamento, Mistura, Processamento e Receção.
- Um produto pode ser efetuado pelo Serviço do Laboratório (SL) ou pelo Encarregado.
- As frequências de realização existentes são: 1x, 2x ou Nx.
- As análises existentes são: Análise a defeitos do produto, Verificação da concentração do Cloro, Geração de Ozonos e Concentração (%), Verificação de Odor a Ozono, Temperaturas das Maquinas de Vidrar, Temperatura dos tanques de Salmoura, Densidade na água de salmoura, Temperatura do túnel e Velocidade dos tapetes, e Temperatura do produto final.
- No Plano de Trabalho são também especificados os instrumentos utilizados na realização das operações e as RDQs utilizadas.
- Cada registo no Plano de Trabalho tem ainda uma referência para o documento a ser alterado com a informação desejada.
- Não existem instruções no plano de trabalho com o mesmo nome de produto e análise.
- Os três tipos de documentos existentes são: Análise a defeitos do produto, Cloros e Ozono, e Condições de Análise.
- Os documentos são todos identificados por um nome único.
- O documento de Análise a defeitos do produto guarda informação sobre a data em que a análise foi efetuada, quem a efetuou, a origem do produto, o lote que lhe foi atribuído e dados da análise em si, realizadas quando o produto se encontra fresco e quando se encontra congelado.
- Ao produto fresco é efetuada uma análise organoléptica onde é guardada informação da cotação total, bem como a sua classificação de frescura, e uma verificação do calibre onde é guardada informação da massa do produto, o número de unidades e a classificação em unidades por massa. É também guardado um número único que identifica o produto fresco.
- Os valores de classificação de frescura existentes são: A, B, C, D, E, F e extra.
- Ao produto congelado são realizadas até três amostras, denominadas de tomas, cada uma contendo informação do seu número de Toma, massa, número de unidades, e

quantidade em percentagem de sangue, olho vermelho, rabo inteiro, meio rabo, rabo partido, danificados e sem pele. É-lhes também calculada as quantidades, em percentagem, de vidragem mínima, máxima, média e amplitude. É ainda guardado um número único que identifica no produto fresco.

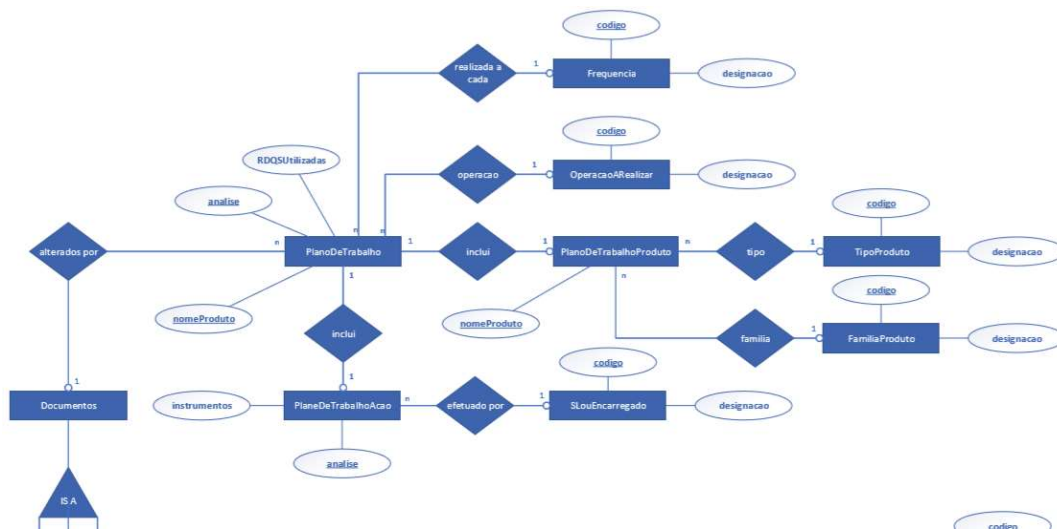
- Cada número de Toma é único dentro do mesmo produto congelado.
- Não existem registos em documentos de Análise a defeitos do produto com a mesma data de análise, a serem realizados pela mesma pessoa, com o mesmo lote atribuído.
- O documento de Cloros e Ozono guarda informação sobre a data e hora em que análise foi efetuada, quem a efetuou, a fábrica onde foi efetuada, o produto a ser processado e vários controlos do processo produtivo.
- A fábrica onde a análise foi efetuada pode ser a Norte, identificada por FN, ou a Sul, identificada por FS.
- Os controlos efetuados para Cloros e Ozono são a leitura do cloro de registados, cloro nos arrefecedores 1 e 2, cloro no chiller, cloro nas mangueiras 1, 2, 4, 6, 7, 9, e exterior, cloro nas lavadoras 1, 2, 1 do esgoto e 2 do esgoto, percentagens de oxigénio e geração de ozono, cloro controlador de entrada, controlador das lavadoras e odor a ozono.
- O único valor de controlo necessário é o odor a ozono.
- Não existem registos em documentos de Cloros e Ozono com a mesma data e hora de análise, a serem efetuados sobre o mesmo produto pela mesma pessoa.
- O documento de Condições de Análise guarda informação sobre a data e hora em que a análise foi efetuada, o produto a ser analisado, o turno onde se realizou a análise, quem efetuou o controlo, o calibre (que não é necessário), os códigos do lote e do contentor, as propriedades organolépticas, a temperatura do produto, a linha em que o produto se encontra, e informação do estado da câmara de frescos, túnel de congelação, tanque de salmouração e das duas máquinas de vidrar usadas no processo.
- Os turnos disponíveis são: 1, 2 e 3.
- As linhas disponíveis são: 1, 2, 3, 4, 5 e 6.
- Da câmara de frescos é retirada a informação relativa ao número da análise na câmara, à temperatura do registador, à temperatura da sonda e à humidade relativa.
- Do túnel de congelação é retirada informação relativa ao número da análise no túnel, à temperatura no túnel, às velocidades dos tapetes 1 e 2 e ao número de identificação do túnel.
- Do tanque de salmouração é retirada informação relativa ao número de análise no tanque, às temperaturas dos tanques 1 e 2, e ao nível da água do tanque.
- Das máquinas de vidrar é retirada a informação relativa ao número da análise nas máquinas, à temperatura do tanque, ao fluxo contínuo, à necessidade de mudança de oito em oito horas no caso de não ser contínuo e ao horário, no caso de haver a mudança previamente enunciada. Estes dois últimos campos não são necessários.
- Não existem registos em documentos de Condições de Análise com o mesmo código do contentor.
- Lotes são compostos por nove Algarismos, e números de contentor por onze.

# Diagrama Entidade-Relação



A solução desenvolvida para a organização da estrutura de dados é a demonstrada acima. Consiste de quatro secções principais: Plano de Trabalho, Análise de Defeitos do Produto, Cloros e Ozono, e Condições de Análise.

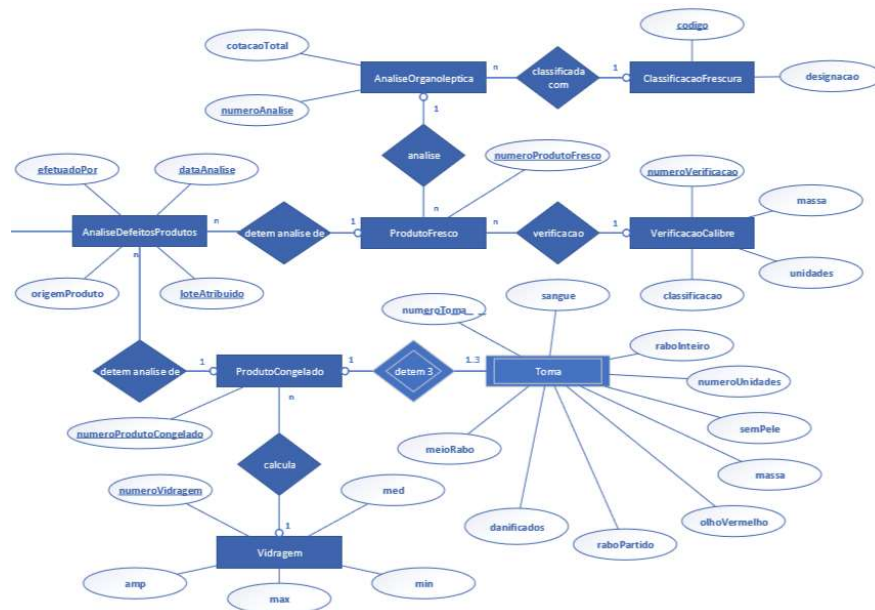
## Plano de Trabalho



O Plano de Trabalho é a secção mais importante do projeto, sendo a que necessitava de modelação.

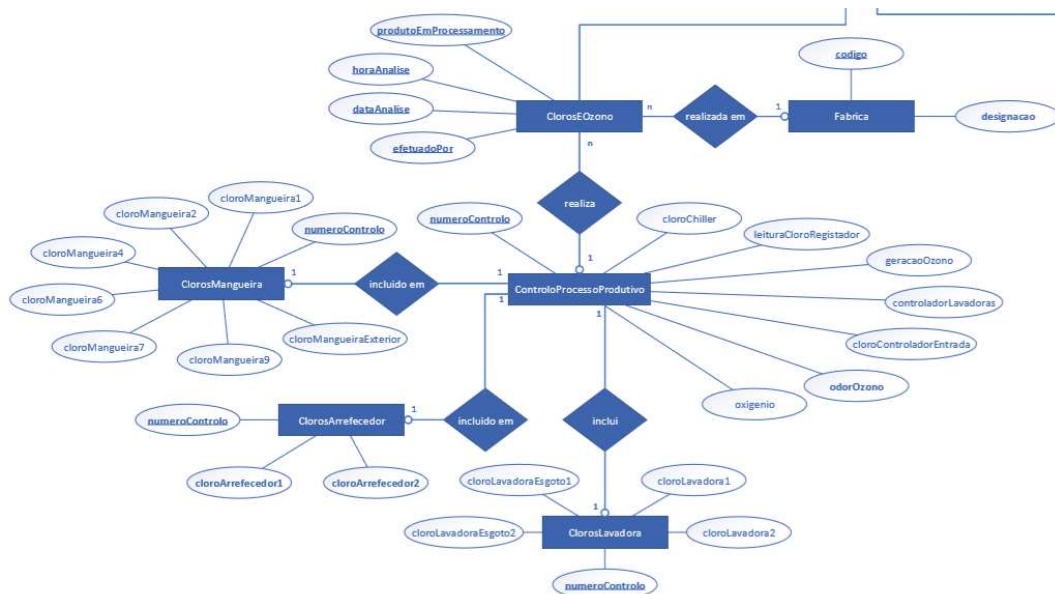
Consiste na informação base das operações que necessitam de ser realizadas num dia normal de trabalho e onde os dados obtidos deverão ser guardados. Os dados são guardados em documentos que podem ser de três tipos, as três secções seguintes.

## Análise a Defeitos do Produto



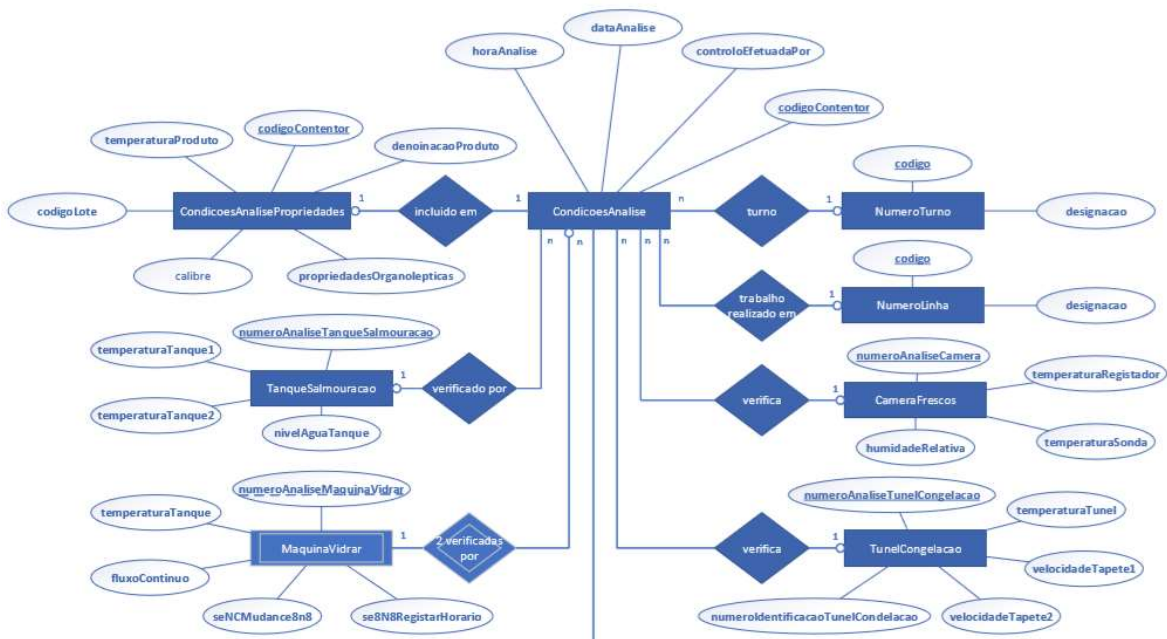
O primeiro dos tipos de documentos é a Análise a Defeitos do Produto. A informação contida nesta secção é relativa a algumas análises realizadas a certos produtos tanto no seu estado fresco como congelado.

## Cloros e Ozono



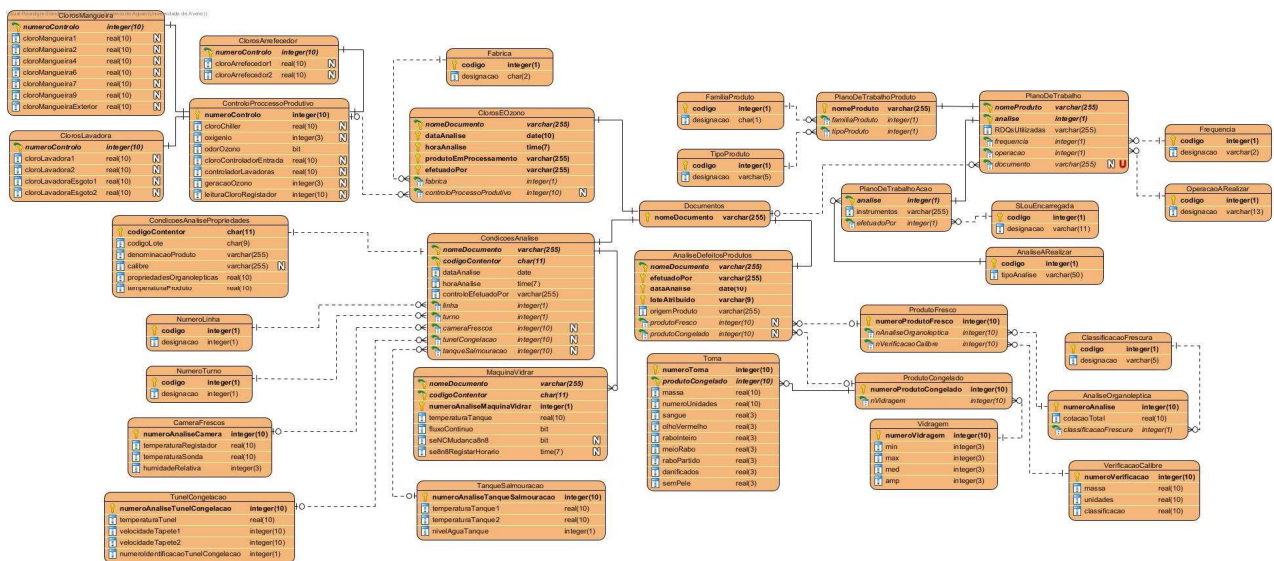
O segundo dos tipos de documento é Cloros e Ozono, que contém a informação da quantidade de cloro nas águas de vários equipamentos, bem como alguns parâmetros extra.

## Cloros e Ozono



O último dos tipos de documentos é Condições de Análise. Este contém informação relativa ao estado dos vários equipamentos utilizados nas diversas análises.

# Esquema Relacional



## SQL DDL associa do à definição da estrutura da BD

```
CREATE TABLE Proj.Documentos (  
    nomeDocumento VARCHAR(255) NOT NULL,  
  
    CONSTRAINT PKDOC  
        PRIMARY KEY(nomeDocumento)  
);
```

```
CREATE TABLE Proj.AnaliseARealizar (  
    codigo TINYINT NOT NULL,  
    tipoAnalise VARCHAR(50) NOT NULL,  
  
    CONSTRAINT PKAAR  
        PRIMARY KEY(codigo)  
);
```

```
CREATE TABLE Proj.Frequencia (  
    codigo TINYINT NOT NULL,  
    designacao CHAR(2) NOT NULL,  
  
    CONSTRAINT PKFREQ  
        PRIMARY KEY(codigo)  
);
```

```
CREATE TABLE Proj.SLouEncarregada (  
    codigo TINYINT NOT NULL,  
    designacao VARCHAR(11) NOT NULL,  
  
    CONSTRAINT PKSLE  
        PRIMARY KEY(codigo)  
);
```

```
CREATE TABLE Proj.TipoProduto (  
    codigo TINYINT NOT NULL,  
    designacao VARCHAR(5) NOT NULL,  
  
    CONSTRAINT PKTPD  
        PRIMARY KEY(codigo)  
);
```

```
CREATE TABLE Proj.FamiliaProduto (  
    codigo TINYINT NOT NULL,  
    designacao CHAR(1) NOT NULL,  
  
    CONSTRAINT PKFPD  
        PRIMARY KEY(codigo)  
);
```

```
CREATE TABLE Proj.OperacaoARealizar (  
    codigo TINYINT NOT NULL,  
    designacao VARCHAR(13) NOT NULL,  
  
    CONSTRAINT PKOAR  
        PRIMARY KEY(codigo)  
);
```



```

CREATE TABLE Proj.PlanoDeTrabalhoAcao (
    analise          TINYINT          NOT NULL,
    instrumentos     VARCHAR(255)      NOT NULL,
    efetuadoPor      TINYINT          NOT NULL,

    CONSTRAINT PKPDTA
        PRIMARY KEY(analise),
    CONSTRAINT FKPDTA1
        FOREIGN KEY(analise) REFERENCES Proj.AnaliseARealizar(codigo)
            ON UPDATE CASCADE
            ON DELETE NO ACTION,
    CONSTRAINT FKPDTA2
        FOREIGN KEY(efetuatedPor) REFERENCES Proj.SLouEncarregada(codigo)
            ON UPDATE CASCADE
            ON DELETE NO ACTION
);

CREATE TABLE Proj.PlanoDeTrabalhoProduto (
    nomeProduto      VARCHAR(255)      NOT NULL,
    tipoProduto      TINYINT          NOT NULL,
    familiaProduto   TINYINT          NOT NULL,

    CONSTRAINT PKPDTS2
        PRIMARY KEY(nomeProduto),
    CONSTRAINT FKPDTS21
        FOREIGN KEY(tipoProduto) REFERENCES Proj.TipoProduto(codigo)
            ON UPDATE CASCADE
            ON DELETE NO ACTION,
    CONSTRAINT FKPDTS22
        FOREIGN KEY(familiaProduto) REFERENCES Proj.FamiliaProduto(codigo)
            ON UPDATE CASCADE
            ON DELETE NO ACTION
);

CREATE TABLE Proj.ClassificacaoFrescura (
    codigo           TINYINT          NOT NULL,
    designacao       VARCHAR(5)       NOT NULL,

    CONSTRAINT PKCFC
        PRIMARY KEY(codigo)
);

CREATE TABLE Proj.AnaliseOrganoleptica (
    numeroAnalise    INT              NOT NULL,
    cotacaoTotal     DECIMAL(10,1)    NOT NULL,
    classificacaoFrescura TINYINT      NOT NULL,

    CONSTRAINT PKAOL
        PRIMARY KEY(numeroAnalise),
    CONSTRAINT FKAOL
        FOREIGN KEY(classificacaoFrescura) REFERENCES
Proj.ClassificacaoFrescura(codigo)
            ON UPDATE CASCADE
            ON DELETE NO ACTION,
    CONSTRAINT CHECKAOL
        CHECK(cotacaoTotal > 0)
);

```

```

CREATE TABLE Proj.PlanoDeTrabalho (
    nomeProduto          VARCHAR(255)          NOT NULL,
    analise              TINYINT                NOT NULL,
    RDQsUtilizadas      VARCHAR(255)          NOT NULL,
    frequencia          TINYINT                NOT NULL,
    operacao            TINYINT                NOT NULL,
    documento           VARCHAR(255),

    CONSTRAINT PKPDT
        PRIMARY KEY(nomeProduto, analise),
    CONSTRAINT FKPD1
        FOREIGN KEY(nomeProduto) REFERENCES
Proj.PlanoDeTrabalhoProduto(nomeProduto)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKPD2
        FOREIGN KEY(analise) REFERENCES Proj.PlanoDeTrabalhoAcao(analise)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKPD3
        FOREIGN KEY(frequencia) REFERENCES Proj.Frequencia(codigo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKPD4
        FOREIGN KEY(operacao) REFERENCES Proj.OperacaoARealizar(codigo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKPD5
        FOREIGN KEY(documento) REFERENCES Proj.Documentos(nomeDocumento)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT UNIQPDT
        UNIQUE(documento)
);

```

```

CREATE TABLE Proj.VerificacaoCalibre (
    numeroVerificacao    INT                  NOT NULL,
    massa                DECIMAL(10,1)        NOT NULL,
    unidades             DECIMAL(10,1)        NOT NULL,
    classificacao        DECIMAL(10,1)        NOT NULL,

    CONSTRAINT PKVCB
        PRIMARY KEY(numeroVerificacao),
    CONSTRAINT CHECKVCB1
        CHECK(massa > 0 AND unidades > 0 AND classificacao > 0)
);

```

```

CREATE TABLE Proj.Vidragem (
    numeroVidragem          INT                NOT NULL,
    [min]                    TINYINT             NOT NULL,
    [max]                    TINYINT             NOT NULL,
    med                      TINYINT             NOT NULL,
    amp                      TINYINT             NOT NULL,

    CONSTRAINT PKVDR
        PRIMARY KEY(numeroVidragem),
    CONSTRAINT CHECKVDR1
        CHECK([min] >= 0 AND [min] <= 100),
    CONSTRAINT CHECKVDR2
        CHECK([max] >= 0 AND [max] <= 100),
    CONSTRAINT CHECKVDR3
        CHECK(med >= 0 AND med <= 100),
    CONSTRAINT CHECKVDR4
        CHECK([min] <= [max] AND [min] <= med AND med <= [max]),
    CONSTRAINT CHECKVDR5
        CHECK(amp >= 0 AND amp <= 100),
    CONSTRAINT CHECKVDR6
        CHECK(amp = [max] - [min])
);

CREATE TABLE Proj.ProdutoCongelado (
    numeroProdutoCongelado  INT                NOT NULL,
    nVidragem               INT                NOT NULL,

    CONSTRAINT PKPCG
        PRIMARY KEY(numeroProdutoCongelado),
    CONSTRAINT FKPCG
        FOREIGN KEY(nVidragem) REFERENCES Proj.Vidragem(numeroVidragem)
            ON UPDATE CASCADE
            ON DELETE NO ACTION
);

CREATE TABLE Proj.NumeroLinha (
    codigo                  TINYINT             NOT NULL,
    designacao              TINYINT             NOT NULL,

    CONSTRAINT PKNLN
        PRIMARY KEY(codigo)
);

CREATE TABLE Proj.NumeroTurno (
    codigo                  TINYINT             NOT NULL,
    designacao              TINYINT             NOT NULL,

    CONSTRAINT PKNTN
        PRIMARY KEY(codigo)
);

```

```

CREATE TABLE Proj.Toma (
    numeroToma          TINYINT          NOT NULL,
    produtoCongelado    INT              NOT NULL,
    massa               DECIMAL(10,1)    NOT NULL,
    numeroUnidades       DECIMAL(10,1)    NOT NULL,
    sangue              DECIMAL(4,1)     NOT NULL,
    olhoVermelho        DECIMAL(4,1)     NOT NULL,
    raboInteiro         DECIMAL(4,1)     NOT NULL,
    meioRabo            DECIMAL(4,1)     NOT NULL,
    raboPartido         DECIMAL(4,1)     NOT NULL,
    danificados         DECIMAL(4,1)     NOT NULL,
    semPele             DECIMAL(4,1)     NOT NULL,

    CONSTRAINT PKTOM
        PRIMARY KEY(numeroToma,produtoCongelado),
    CONSTRAINT FKTM
        FOREIGN KEY(produtoCongelado) REFERENCES
Proj.ProdutoCongelado(numeroProdutoCongelado)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT CHECKTOM1
        CHECK(numeroToma = 1 OR numeroToma = 2 OR numeroToma = 3),
    CONSTRAINT CHECKTOM2
        CHECK(massa > 0 AND numeroUnidades > 0),
    CONSTRAINT CHECKTOM3
        CHECK(sangue >= 0 AND sangue <= 100),
    CONSTRAINT CHECKTOM4
        CHECK(olhoVermelho >= 0 AND olhoVermelho <= 100),
    CONSTRAINT CHECKTOM5
        CHECK(raboInteiro >= 0 AND raboInteiro <= 100),
    CONSTRAINT CHECKTOM6
        CHECK(meioRabo >= 0 AND meioRabo <= 100),
    CONSTRAINT CHECKTOM7
        CHECK(raboPartido >= 0 AND raboPartido <= 100),
    CONSTRAINT CHECKTOM8
        CHECK(danificados >= 0 AND danificados <= 100),
    CONSTRAINT CHECKTOM9
        CHECK(semPele >= 0 AND semPele <= 100)
);

CREATE TABLE Proj.CameraFrescos (
    numeroAnaliseCamera SMALLINT          NOT NULL,
    temperaturaRegistador DECIMAL(10,1)    NOT NULL,
    temperaturaSonda     DECIMAL(10,1)    NOT NULL,
    humidadeRelativa     TINYINT          NOT NULL,

    CONSTRAINT PKCMF
        PRIMARY KEY(numeroAnaliseCamera),
    CONSTRAINT CHECKCMF1
        CHECK(temperaturaRegistador >= 0 AND temperaturaSonda >= 0 AND
humidadeRelativa >= 0),
    CONSTRAINT CHECKCMF2
        CHECK(humidadeRelativa >= 0 AND humidadeRelativa <= 100)
);

```

```

CREATE TABLE Proj.AnaliseDefeitosProdutos (
    nomeDocumento          VARCHAR(255)          NOT NULL,
    efetuadoPor            VARCHAR(255)          NOT NULL,
    dataAnalise            DATE                  NOT NULL,
    loteAtribuido          CHAR(9)               NOT NULL,
    origemProduto          VARCHAR(255)          NOT NULL,
    produtoFresco          INT,
    produtoCongelado       INT,

    CONSTRAINT PKADP
        PRIMARY KEY(nomeDocumento,efetuadoPor,dataAnalise,loteAtribuido),
    CONSTRAINT FKADP1
        FOREIGN KEY(nomeDocumento) REFERENCES
Proj.Documentos(nomeDocumento)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKADP2
        FOREIGN KEY(produtoFresco) REFERENCES
Proj.ProdutoFresco(numeroProdutoFresco)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKADP3
        FOREIGN KEY(produtoCongelado) REFERENCES
Proj.ProdutoCongelado(numeroProdutoCongelado)
        ON UPDATE CASCADE
        ON DELETE NO ACTION
);

```

```

CREATE TABLE Proj.TunelCongelacao (
    numeroAnaliseTunelCongelacao  SMALLINT          NOT NULL,
    temperaturaTunel              DECIMAL(10,1)       NOT NULL,
    velocidadeTapete1              TINYINT            NOT NULL,
    velocidadeTapete2              TINYINT            NOT NULL,
    numeroIdentificacaoTunelCongelacao  TINYINT       NOT NULL,

    CONSTRAINT PKTCL
        PRIMARY KEY(numeroAnaliseTunelCongelacao),
    CONSTRAINT CHECKTCL1
        CHECK(temperaturaTunel <= 0),
    CONSTRAINT CHECKTCL2
        CHECK(velocidadeTapete1 > 0 AND velocidadeTapete2 > 0),
    CONSTRAINT CHECKTCL3
        CHECK(numeroIdentificacaoTunelCongelacao >= 0)
);

```

```

CREATE TABLE Proj.TanqueSalmouracao (
    numeroAnaliseTanqueSalmouracao SMALLINT NOT NULL,
    temperaturaTanque1 DECIMAL(10,1) NOT NULL,
    temperaturaTanque2 DECIMAL(10,1) NOT NULL,
    nivelAguaTanque TINYINT NOT NULL,

    CONSTRAINT PKTSM
        PRIMARY KEY(numeroAnaliseTanqueSalmouracao),
    CONSTRAINT CHECKTSM1
        CHECK(temperaturaTanque1 > 0 AND temperaturaTanque2 > 0),
    CONSTRAINT CHECKTSM2
        CHECK(nivelAguaTanque = 1 OR nivelAguaTanque = 2)
);

```

```

CREATE TABLE Proj.CondicoesAnalisePropriedades (
    codigoContentor CHAR(11) NOT NULL,
    denominacaoProduto VARCHAR(255) NOT NULL,
    calibre VARCHAR(255),
    propriedadesOrganolepticas DECIMAL(3,1) NOT NULL,
    temperaturaProduto DECIMAL(10,1) NOT NULL,
    codigoLote CHAR(9) NOT NULL,

    CONSTRAINT PKCDAP
        PRIMARY KEY(codigoContentor),
    CONSTRAINT CHECKCDAP1
        CHECK(propriedadesOrganolepticas >= 0 AND
propriedadesOrganolepticas <= 10),
    CONSTRAINT CHECKCDAP2
        CHECK(temperaturaProduto < 0)
);

```

```

CREATE TABLE Proj.Fabrica (
    codigo TINYINT NOT NULL,
    designacao CHAR(2) NOT NULL,

    CONSTRAINT PKFBR
        PRIMARY KEY(codigo)
);

```

```

CREATE TABLE Proj.CondicoesAnalise (
    nomeDocumento          VARCHAR(255)          NOT NULL,
    codigoContentor        CHAR(11)              NOT NULL,
    dataAnalise            DATE                  NOT NULL,
    horaAnalise            TIME                  NOT NULL,
    controloEfetuadoPor    VARCHAR(255)          NOT NULL,
    linha                  TINYINT               NOT NULL,
    turno                  TINYINT               NOT NULL,
    cameraFrescos          SMALLINT,
    tunelCongelacao        SMALLINT,
    tanqueSalmouracao      SMALLINT,

    CONSTRAINT PKCDA
        PRIMARY KEY(nomeDocumento,codigoContentor),
    CONSTRAINT FKCA1
        FOREIGN KEY(codigoContentor) REFERENCES
Proj.CondicoesAnalisePropriedades(codigoContentor)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKCA2
        FOREIGN KEY(linha) REFERENCES Proj.NumeroLinha(codigo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKCA3
        FOREIGN KEY(turno) REFERENCES Proj.NumeroTurno(codigo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKCA4
        FOREIGN KEY(cameraFrescos) REFERENCES
Proj.CameraFrescos(numeroAnaliseCamera)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKCA5
        FOREIGN KEY(tunelCongelacao) REFERENCES
Proj.TunelCongelacao(numeroAnaliseTunelCongelacao)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKCA6
        FOREIGN KEY(tanqueSalmouracao) REFERENCES
Proj.TanqueSalmouracao(numeroAnaliseTanqueSalmouracao)
        ON UPDATE CASCADE
        ON DELETE NO ACTION
);

CREATE TABLE Proj.ClorosArrefecedor (
    numeroControlo          SMALLINT              NOT NULL,
    cloroArrefecedor1       DECIMAL(4,1),
    cloroArrefecedor2       DECIMAL(4,1),

    CONSTRAINT PKCAF
        PRIMARY KEY(numeroControlo),
    CONSTRAINT FKCAF
        FOREIGN KEY(numeroControlo) REFERENCES
Proj.ControloProcessoProdutivo(numeroControlo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT CHECKCAF
        CHECK(cloroArrefecedor1 >= 0 AND cloroArrefecedor2 >= 0 AND
cloroArrefecedor1 <= 100 AND cloroArrefecedor2 <= 100)
);

```

```

CREATE TABLE Proj.MaquinaVidrar (
    nomeDocumento          VARCHAR(255)          NOT NULL,
    codigoContentor        CHAR(11)              NOT NULL,
    numeroAnaliseMaquinaVidrar SMALLINT          NOT NULL,
    temperaturaTanque      DECIMAL(10,1)         NOT NULL,
    fluxoContínuo          BIT                   NOT NULL,
    seNCMudanca8n8         BIT,
    se8n8RegistrarHorario  TIME,

    CONSTRAINT PKMQV
        PRIMARY
KEY(nomeDocumento,codigoContentor,numeroAnaliseMaquinaVidrar),
    CONSTRAINT FKMQV
        FOREIGN KEY(nomeDocumento,codigoContentor) REFERENCES
Proj.CondicoesAnalise(nomeDocumento,codigoContentor)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT CHECKMQV1
        CHECK(temperaturaTanque > 0),
    CONSTRAINT CHECKMQV2
        CHECK(numeroAnaliseMaquinaVidrar >= 1 AND
numeroAnaliseMaquinaVidrar <= 3)
);

CREATE TABLE Proj.ControloProcessoProdutivo (
    numeroControlo          SMALLINT              NOT NULL,
    cloroChiller            DECIMAL(4,1),
    cloroControladorEntrada DECIMAL(4,1),
    oxigenio               TINYINT,
    odorOzono              BIT                   NOT NULL,
    controladorLavadoras    DECIMAL(4,1),
    geracaoOzono            TINYINT,
    leituraCloroRegistador  TINYINT,

    CONSTRAINT PKCPP
        PRIMARY KEY(numeroControlo),
    CONSTRAINT CHECKCPP1
        CHECK(cloroChiller >= 0 AND cloroControladorEntrada >= 0),
    CONSTRAINT CHECKCPP2
        CHECK(cloroChiller <= 100 AND cloroControladorEntrada <= 100),
    CONSTRAINT CHECKCPP3
        CHECK(oxigenio >= 0 AND oxigenio <= 100),
    CONSTRAINT CHECKCPP4
        CHECK(controladorLavadoras > 0),
    CONSTRAINT CHECKCPP5
        CHECK(geracaoOzono >= 0 AND geracaoOzono <= 100),
    CONSTRAINT CHECKCPP6
        CHECK(leituraCloroRegistador > 0)
);

```



```

CREATE TABLE Proj.ProdutoFresco (
    numeroProdutoFresco      INT                NOT NULL,
    nAnaliseOrganoleptica     INT                NOT NULL,
    nVerificacaoCalibre       INT                NOT NULL,

    CONSTRAINT PKPFC
        PRIMARY KEY(numeroProdutoFresco),
    CONSTRAINT FKPFC1
        FOREIGN KEY(nAnaliseOrganoleptica) REFERENCES
Proj.AnaliseOrganoleptica(numeroAnalise)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKPFC2
        FOREIGN KEY(nVerificacaoCalibre) REFERENCES
Proj.VerificacaoCalibre(numeroVerificacao)
        ON UPDATE CASCADE
        ON DELETE NO ACTION
);

```

```

CREATE TABLE Proj.ClorosMangueira (
    numeroControlo           SMALLINT            NOT NULL,
    cloroMangueira1          DECIMAL(4,1),
    cloroMangueira2          DECIMAL(4,1),
    cloroMangueira4          DECIMAL(4,1),
    cloroMangueira6          DECIMAL(4,1),
    cloroMangueira7          DECIMAL(4,1),
    cloroMangueira9          DECIMAL(4,1),
    cloroMangueiraExterior   DECIMAL(4,1),

    CONSTRAINT PKCMG
        PRIMARY KEY(numeroControlo),
    CONSTRAINT FKCMG
        FOREIGN KEY(numeroControlo) REFERENCES
Proj.ControloProcessoProdutivo(numeroControlo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT CHECKCMG1
        CHECK(cloroMangueira1 >= 0 AND cloroMangueira2 >= 0 AND
cloroMangueira4 >= 0 AND cloroMangueira6 >= 0 AND cloroMangueira7 >= 0 AND
cloroMangueira9 >= 0 AND cloroMangueiraExterior >= 0),
    CONSTRAINT CHECKCMG2
        CHECK(cloroMangueira1 <= 100 AND cloroMangueira2 <= 100 AND
cloroMangueira4 <= 100 AND cloroMangueira6 <= 100 AND cloroMangueira7 <= 100 AND
cloroMangueira9 <= 100 AND cloroMangueiraExterior <= 100)
);

```

```

CREATE TABLE Proj.ClorosLavadora (
    numeroControlo          SMALLINT          NOT NULL,
    cloroLavadora1          DECIMAL(4,1),
    cloroLavadora2          DECIMAL(4,1),
    cloroLavadoraEsgoto1    DECIMAL(4,1),
    cloroLavadoraEsgoto2    DECIMAL(4,1),

    CONSTRAINT PKCLD
        PRIMARY KEY(numeroControlo),
    CONSTRAINT FKCLD
        FOREIGN KEY(numeroControlo) REFERENCES
Proj.ControloProcessoProdutivo(numeroControlo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT CHECKCLD1
        CHECK(cloroLavadora1 >= 0 AND cloroLavadora2 >= 0 AND
cloroLavadoraEsgoto1 >= 0 AND cloroLavadoraEsgoto2 >= 0),
    CONSTRAINT CHECKCDL2
        CHECK(cloroLavadora1 <= 100 AND cloroLavadora2 <= 100 AND
cloroLavadoraEsgoto1 <= 100 AND cloroLavadoraEsgoto2 <= 100)
);

CREATE TABLE Proj.ClorosEOzono (
    nomeDocumento          VARCHAR(255)        NOT NULL,
    dataAnalise            DATE                NOT NULL,
    horaAnalise            TIME                NOT NULL,
    produtoEmProcessamento VARCHAR(255)        NOT NULL,
    efetuadoPor            VARCHAR(255)        NOT NULL,
    fabrica                TINYINT            NOT NULL,
    controloProcessoProdutivo SMALLINT,

    CONSTRAINT PKCOE
        PRIMARY
KEY(nomeDocumento,dataAnalise,horaAnalise,produtoEmProcessamento,efetuadoPor),
    CONSTRAINT FKCOE1
        FOREIGN KEY(nomeDocumento) REFERENCES
Proj.Documentos(nomeDocumento)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKCOE2
        FOREIGN KEY(fabrica) REFERENCES Proj.Fabrica(codigo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION,
    CONSTRAINT FKCOE3
        FOREIGN KEY(controloProcessoProdutivo) REFERENCES
Proj.ControloProcessoProdutivo(numeroControlo)
        ON UPDATE CASCADE
        ON DELETE NO ACTION
);

```

Estas instruções SQL DDL são as necessárias para implementar as relações/tabelas da solução concebida. Contudo, para a aplicação, ainda foi utilizada uma relação/tabela adicional.

```

CREATE TABLE [Log].UserCredentials (
    nMec INT NOT NULL,
    [password] VARCHAR(64) NOT NULL,
    permissionLevel BIT NOT NULL,

    CONSTRAINT PKUC
        PRIMARY KEY(nMec),
    CONSTRAINT CHECKUC
        CHECK(permissionLevel = 0 OR permissionLevel = 1)
);

```

## SQL DML associados a cada formulário gráfico

### Login

```

SELECT UC.permissionLevel, CNTT.counter
FROM
    (
        SELECT COUNT(*) as counter
        FROM [Log].UserCredentials AS UC
        WHERE UC.nMec = @nMec
        AND UC.password = @password
    ) AS CNTT,
    [Log].UserCredentials AS UC
WHERE UC.nMec = @nMec AND UC.password = @password

```

### Insert - Plano de Trabalho

Stored Procedure: Proj.InsertPlanoDeTrabalhoProduto  
(ver secção relativa às Stored Procedures)

```

INSERT
Proj.PlanoDeTrabalhoProduto(nomeProduto,familiaProduto,
tipoProduto)
VALUES (@nomeProduto,@familiaProduto,@tipoProduto);

```

Stored Procedure: Proj.InsertPlanoDeTrabalhoAcao  
(ver secção de Stored Procedures)

```

INSERT Proj.PlanoDeTrabalhoAcao(analise,instrumentos,efetuadoPor)
VALUES (@analise,@instrumentos,@efetuadoPor);

```

Stored Procedure: Proj.InsertPlanoDeTrabalho  
(ver secção de Stored Procedures)

```

INSERT
Proj.PlanoDeTrabalho(nomeProduto,analise,RDQsUtilizadas,frequencia,operacao,documento)
VALUES (@nomeProduto,@analise,@RDQsUtilizadas,@frequencia,@operacao,@documento);

```

## Insert - Análise a Defeitos do Produto

Stored Procedure: Proj.InsertAnaliseDefeitosProdutos  
(ver secção relativa às Stored Procedures)

### INSERT

Proj.AnaliseDefeitosProdutos(nomeDocumento,efetuadoPor,  
dataAnalise,loteAtribuido,origemProduto,produtoFresco,p  
rodutoCongelado)

### VALUES

(@nomeDocumento,@efetuadoPor,@dataAnalise,@loteAtribuid  
o,@origemProduto,@produtoFresco,@produtoCongelado);

The screenshot shows the 'Análise a Defeitos do Produto' form. It has a sidebar with categories: Plano de Trabalho, Análise a Defeitos, Cloros e Ozono, Condições de Análise, and Documentos. The main form area contains the following fields: 'Nome do Documento' (text), 'Efetuado Por' (text), 'Data de Análise' (date, 6/1/2019), 'Lote Atribuído' (text), 'Origem do Produto' (text), 'Produto Fresco' (text), 'Produto Congelado' (text), and buttons for 'Inserir Dados' and 'Limpar Informação'.

## Insert - Cloros e Ozono

Stored Procedure: Proj.ControloProcessoProdutivoBatch  
(ver secção relativa às Stored Procedures)

### INSERT

ControloProcessoProdutivo(numeroControlo,cloroChiller,o  
xigenio,odorOzono,cloroControladorEntrada,controladorLa  
vadoras,geracaoOzono,leituraCloroRegistador)

VALUES(@numeroControlo,@cloroChiller,@oxigenio,@odorOzo  
no,@cloroControladorEntrada,@controladorLavadoras,@gera  
caoOzono,@leituraCloroRegistador);

### INSERT

ClorosArrefecedor(numeroControlo,cloroArrefecedor1,clor  
oArrefecedor2)

VALUES(@numeroControlo,@cloroArrefecedor1,@cloroArrefec  
edor2);

### INSERT

ClorosLavadora(numeroControlo,cloroLavadora1,cloroLavad  
ora2,cloroLavadoraEsgoto1,cloroLavadoraEsgoto2)

VALUES(@numeroControlo,@cloroLavadora1,@cloroLavadora2,  
@cloroLavadoraEsgoto1,@cloroLavadoraEsgoto2);

### INSERT

ClorosMangueira(numeroControlo,cloroMangueira1,cloroMan  
gueira2,cloroMangueira4,cloroMangueira6,cloroMangueira7  
,cloroMangueira9,cloroMangueiraExterior)

VALUES(@numeroControlo,@cloroMangueira1,@cloroMangueira  
2,@cloroMangueira4,@cloroMangueira6,@cloroMangueira7,@c  
loroMangueira9,@cloroMangueiraExterior);

The screenshot shows the 'Controlo do Processo Produtivo' form. It has a sidebar with categories: Plano de Trabalho, Análise a Defeitos, Cloros e Ozono, Condições de Análise, and Documentos. The main form area contains the following fields: 'Número do Controlo Produtivo' (text), 'Cloro Chiller' (text), 'Oxigenio' (text), 'Odor Ozono' (text), 'Cloro Controlador Entrada' (text), 'Controlador Lavadora' (text), 'Geração de Ozono' (text), 'Leitura Cloro Registador' (text), and buttons for 'Inserir Dados' and 'Limpar Informação'.

Stored Procedure: Proj.InsertClorosEOzono  
(ver secção relativa às Stored Procedures)

### INSERT

Proj.ClorosEOzono(nomeDocumento,dataAnalise,horaAnalise,produtoEmProcessamento,efetuadoPo  
r,fabrica,controloProcessoProdutivo) VALUES  
(@nomeDocumento,@dataAnalise,@horaAnalise,@produtoEmProcessamento,@efetuadoPor,@fabrica,@  
controloProcessoProdutivo);

## Insert - Condições de Análise

Stored Procedure: Proj.InsertCondicoesAnalise  
(ver secção relativa às Stored Procedures)

### INSERT

```
Proj.CondicoesAnalise(nomeDocumento,codigoContentor,dataAnalise,horaAnalise,controloEfetuadoPor,linha,turno,cameraFrescos,tunelCongelacao,tanqueSalmouracao)
```

### VALUES

```
(@nomeDocumento,@codigoContentor,@dataAnalise,@horaAnalise,@controloEfetuadoPor,@linha,@turno,@cameraFrescos,@tunelCongelacao,@tanqueSalmouracao);
```

## Insert - Documentos

Stored Procedure: Proj.InsertDocumento  
(ver secção relativa às Stored Procedures)

```
INSERT Proj.Documentos(nomeDocumento)
```

```
VALUES (@nomeDocumento)
```

## Insert - Produto Fresco


Stored Procedure: Proj.InsertProdutFreco  
(ver secção relativa às Stored Procedures)

### INSERT

```
Proj.ProdutoFresco(numeroProdutoFresco,nAnaliseOrganoleptica,nVerificacaoCalibre)
```

```
VALUES (@numeroProdutoFresco,@nAnaliseOrganoleptica,@nVerificacaoCalibre);
```

## Insert - Análise Organoléptica



Stored Procedure: Proj.InsertAnaliseOrganoleptica  
(ver secção relativa às Stored Procedures)

```
INSERT Proj.AnaliseOrganoleptica(numeroAnalise,cotacaoTotal,classificacaoFrescura)
VALUES (@numeroAnalise,@cotacaoTotal,@classificacaoFrescura);
```

## Insert – Verificação de Calibre



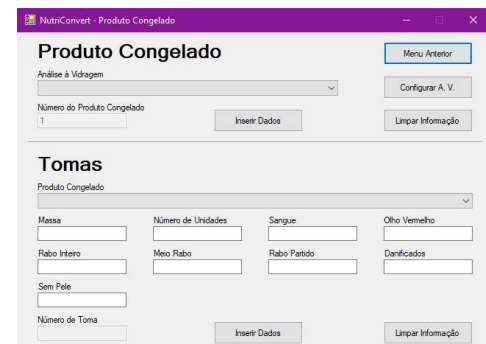
Stored Procedure: Proj.InsertVerificacaoCalibre  
(ver secção relativa às Stored Procedures)

```
INSERT Proj.VerificacaoCalibre(numeroVerificacao,masa,unidades,classificacao) VALUES
(@numeroVerificacao,@masa,@unidades,@classificacao);
```

## Insert – Produto Congelado

Stored Procedure: Proj.InsertProdutoCongelado  
(ver secção relativa às Stored Procedures)

```
INSERT
Proj.ProdutoCongelado(numeroProdutoCongelado,nVidragem)
VALUES (@numeroProdutoCongelado,@nVidragem);
```



Stored Procedure: Proj.InsertToma  
(ver secção relativa às Stored Procedures)

#### INSERT

Proj.Toma(numeroToma,produtoCongelado,masa,numeroUnidades,sangue,olhoVermelho,raboInteiro,meioRabo,raboPartido,danificados,semPele)

#### VALUES

(@numeroToma,@produtoCongelado,@masa,@numeroUnidades,@sangue,@olhoVermelho,@raboInteiro,@meioRabo,@raboPartido,@danificados,@semPele);

### Insert – Análise à Vidragem

NutriConvert - Análise à Vidragem

## Análise à Vidragem

Valor Mínimo  Valor Máximo  Valor Médio  Amplitude

Número da Vidragem

Inserir Dados Limpar Informação

Menu Anterior

Stored Procedure: Proj.InsertVidragem  
(ver secção relativa às Stored Procedures)

INSERT Proj.Vidragem(numeroVidragem,[min],[max],med,amp)

VALUES (@numeroVidragem,@min,@max,@med,@amp);

### Insert – Contentores

NutriConvert - Contentores

## Contentores

Código do Contendor  Código Lote  Propriedades Organolépticas  Temperatura do Produto

Denominação do Produto

Calibre

Inserir Dados Limpar Informação

Menu Anterior

Stored Procedure: Proj.InsertCondiçoesAnalisePropriedades  
(ver secção relativa às Stored Procedures)

#### INSERT

Proj.CondiçoesAnalisePropriedades(codigoContendor,denominacaoProduto,calibre,propriedadesOrganolepticas,temperaturaProduto,codigoLote)

#### VALUES

(@codigoContendor,@denominacaoProduto,@calibre,@propriedadesOrganolepticas,@temperaturaProduto,@codigoLote);

## Insert – Câmara de Frescos



Stored Procedure: Proj.InsertCameraFrescos  
(ver secção relativa às Stored Procedures)

### INSERT

```
Proj.CameraFrescos(numeroAnaliseCamera, temperaturaRegistrador, temperaturaSonda, humidade  
Relativa) VALUES  
(@numeroAnaliseCamera, @temperaturaRegistrador, @temperaturaSonda, @humidadeRelativa);
```

## Insert – Túnel de Congelação



Stored Procedure: Proj.InsertTunelCongelacao  
(ver secção relativa às Stored Procedures)

### INSERT

```
Proj.TunelCongelacao(numeroAnaliseTunelCongelacao, temperaturaTunel, velocidadeTapete1, v  
elocidadeTapete2, numeroIdentificacaoTunelCongelacao) VALUES  
(@numeroAnaliseTunelCongelacao, @temperaturaTunel, @velocidadeTapete1, @velocidadeTapete2  
, @numeroIdentificacaoTunelCongelacao);
```

## Insert – Tanque de Salmouraço





Stored Procedure: Proj.InsertTanqueSalmouracao  
(ver secção relativa às Stored Procedures)

#### INSERT

Proj.TanqueSalmouracao(numeroAnaliseTanqueSalmouracao,temperaturaTanque1,temperaturaTanque2,nivelAguaTanque) VALUES  
(@numeroAnaliseTanqueSalmouracao,@temperaturaTanque1,@temperaturaTanque2,@nivelAguaTanque);

#### Insert – Máquina de Vidrar

The screenshot shows a web application window titled 'NutriConvert - Máquina de Vidrar'. The main heading is 'Máquina de Vidrar' with a 'Menu Anterior' button. Below the heading is a 'Nome do Documento' dropdown. There are three input fields: 'Número de Análise à Máquina', 'Código do Contentor' (with a dropdown arrow), and 'Temperatura do Tanque'. Below these are three questions with dropdown menus: 'Nesta Máquina de Vidrar, há fluxo contínuo?', 'Não havendo fluxo contínuo, há mudança a cada 8 horas?', and 'Havendo mudança a cada 8 horas, a que hora se realizou a última?'. The last question has a time input field showing '18:22:17'. At the bottom are two buttons: 'Inserir Dados' and 'Limpar Informação'.

Stored Procedure: Proj.InsertMaquinaVidrar  
(ver secção relativa às Stored Procedures)

#### INSERT

Proj.MaquinaVidrar(nomeDocumento,codigoContentor,numeroAnaliseMaquinaVidrar,temperaturaTanque,fluxoContínuo,seNCMudanca8n8,se8n8RegistarHorario) VALUES  
(@nomeDocumento,@codigoContentor,@numeroAnaliseMaquinaVidrar,@temperaturaTanque,@fluxoContínuo,@seNCMudanca8n8,@se8n8RegistarHorario);

#### Update – Plano de Trabalho

The screenshot shows a web application window titled 'NutriConvert - Atualizar Dados'. On the left is a sidebar with navigation links: 'Plano de Trabalho', 'Análise a Defeitos', 'Cloros e Ozono', 'Condições de Análise', and 'Documentos'. The main area has a table with columns 'Nome do Produto', 'Família do Produto', and 'Tipo de Produto'. The table contains four rows: 'Ingusmao', 'Envelha', 'Igusdigi', and 'Pate Parado Pescada'. Below the table are three buttons: 'Informação do Produto', 'Informação da Análise', and 'Informação de Plano'. To the right of the table are three sections: 'Informação do Produto' with a 'Nome do Produto' field and 'Atualizar Dados'/'Limpar Informação' buttons; 'Informação da Análise' with 'Análise a Realizar', 'Realizada Por', and 'Instrumentos Necessários' fields and 'Atualizar Dados'/'Limpar Informação' buttons; and 'Informação de Plano' with 'Nome do Produto', 'Operação a Realizar', 'Análise a Realizar', 'Frequência', 'RDGs Utilizadas', and 'Documento a Editar' fields and 'Atualizar Dados'/'Limpar Informação' buttons.

Stored Procedure: Proj.UpdatePlanoDeTrabalhoProduto  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.PlanoDeTrabalhoProduto SET familiaProduto = @familiaProduto, tipoProduto = @tipoProduto WHERE nomeProduto = @nomeProduto;
```

Stored Procedure: Proj.UpdatePlanoDeTrabalhoAcao  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.PlanoDeTrabalhoAcao SET instrumentos = @instrumentos, efetuadoPor = @efetuadoPor WHERE analise = @analise;
```

Stored Procedure: Proj.UpdatePlanoDeTrabalho  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.PlanoDeTrabalho SET RDQsUtilizadas = @RDQsUtilizadas, frequencia = @frequencia, operacao = @operacao, documento = @documento WHERE nomeProduto = @nomeProduto AND analise = @analise;
```

## Update – Análise a Defeitos do Produto

Nome do Documento	Efetuado por	Data de Análise	Lote Atribuído	Origem do Produto
ADP566	Rui Miguel	4/14/2019	485739495	Espanha
tester	sdfighnm	6/7/2019	666666666	wedgh

Stored Procedure: Proj.UpdateAnaliseDefeitosProdutos  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.AnaliseDefeitosProdutos SET origemProduto = @origemProduto, produtoFresco = @produtoFresco, produtoCongelado = @produtoCongelado WHERE nomeDocumento = @nomeDocumento AND efetuadoPor = @efetuadoPor AND dataAnalise = @dataAnalise AND loteAtribuido = @loteAtribuido;
```

## Update – Cloros e Ozono

Número do Controlo do Processo Produtivo	Cloro Chiller	Oxigénio	Odor a Ozono?	Cloro Controlo de Entrada
0	3.3	60	<input checked="" type="checkbox"/>	

Stored Procedure: Proj. UpdateControloProcessoProdutivoBatch  
(ver secção relativa às Stored Procedures)

```
UPDATE ControloProcessoProdutivo SET cloroChiller = @cloroChiller, oxigenio =  
@oxigenio, odorOzono = @odorOzono, cloroControladorEntrada =  
@cloroControladorEntrada, controladorLavadoras = @controladorLavadoras, geracaoOzono  
= @geracaoOzono, leituraCloroRegistador = @leituraCloroRegistador WHERE  
numeroControlo = @numeroControlo;
```

```
UPDATE ClorosArrefecedor SET cloroArrefecedor1 = @cloroArrefecedor1,  
cloroArrefecedor2 = @cloroArrefecedor2 WHERE numeroControlo = @numeroControlo;
```

```
UPDATE ClorosLavadora SET cloroLavadora1 = @cloroLavadora1, cloroLavadora2 =  
@cloroLavadora2, cloroLavadoraEsgoto1 = @cloroLavadoraEsgoto1, cloroLavadoraEsgoto2 =  
@cloroLavadoraEsgoto2 WHERE numeroControlo = @numeroControlo;
```

```
UPDATE ClorosMangueira SET cloroMangueira1 = @cloroMangueira1, cloroMangueira2 =  
@cloroMangueira2, cloroMangueira4 = @cloroMangueira4, cloroMangueira6 =  
@cloroMangueira6, cloroMangueira7 = @cloroMangueira7, cloroMangueira9 =  
@cloroMangueira9, cloroMangueiraExterior = @cloroMangueiraExterior WHERE  
numeroControlo = @numeroControlo;
```

Stored Procedure: Proj.UpdateClorosEOzono  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.ClorosEOzono SET fabrica = @fabrica, controloProcessoProdutivo =  
@controloProcessoProdutivo WHERE nomeDocumento = @nomeDocumento AND dataAnalise =  
@dataAnalise AND horaAnalise = @horaAnalise AND produtoEmProcessamento =  
@produtoEmProcessamento AND efetuadoPor = @efetuadoPor;
```

## Update – Condições de Análise

Nome do Documento	Código do Contentor	Data da Análise	Hora da Análise	Controlo efetuado
CDA124	37583384593	1/1/2019	12:13:14	Marta

Stored Procedure: Proj.UpdateCondicoesAnalise  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.CondicoesAnalise SET dataAnalise = @dataAnalise, controloEfetuadoPor =  
@controloEfetuadoPor, linha = @linha, turno = @turno, cameraFrescos = @cameraFrescos,  
tunelCongelacao = @tunelCongelacao, tanqueSalmouracao = @tanqueSalmouracao,  
horaAnalise = @horaAnalise WHERE nomeDocumento = @nomeDocumento AND codigoContentor =  
@codigoContentor;
```

## Update – Documentos

Nome do Documento
ADP566

Stored Procedure: Proj.UpdateDocumento  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.Documentos SET nomeDocumento = @nomeDocumento2 WHERE nomeDocumento = @nomeDocumento1;
```

### Update – Produto Fresco

Número do Produto Fresco	Número da Análise Organoléptica	Cotação Total
0	0	5.3

Stored Procedure: Proj.UpdateProdutoFresco  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.ProdutoFresco SET nAnaliseOrganoleptica = @nAnaliseOrganoleptica, nVerificacaoCalibre = @nVerificacaoCalibre WHERE numeroProdutoFresco = @numeroProdutoFresco;
```

### Update – Análise Organoléptica

Número da Análise Organoléptica	Cotação Total	Classificação de Frescura
0	5.3	D

Stored Procedure: Proj.UpdateAnaliseOrganoleptica  
(ver secção relativa às Stored Procedures)

```
UPDATE Proj.AnaliseOrganoleptica SET cotacaoTotal = @cotacaoTotal, classificacaoFrescura = @classificacaoFrescura WHERE numeroAnalise = @numeroAnalise;
```

### Update – Verificação de Calibre

Número da Verificação de Calibre	Massa	Unidades	Classificação de Calibre
0	3.6	8.8	17.5

Stored Procedure: Proj.UpdateVerificacaoCalibre  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.VerificacaoCalibre **SET** massa = @massa, unidades = @unidades,  
classificacao = @classificacao **WHERE** numeroVerificacao = @numeroVerificacao;

## Update – Produto Congelado

Número do Produto Congelado	Número da Vidragem	Mínimo
0	0	10

**Produto Congelado**

Análise à Vidragem:

Número do Produto Congelado:

Atualizar Dados

Limpar Informação

**Tomas**

Produto Congelado:

Massa:  Número de Unidades:  Sangue:  Olho Vermelho:

Rabo Inteiro:  Meio Rabo:  Rabo Partido:  Danificados:

Sem Pele:  Número de Toma:

Atualizar Dados

Limpar Informação

Stored Procedure: Proj.UpdateProdutoCongelado  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.ProdutoCongelado **SET** nVidragem = @nVidragem **WHERE** numeroProdutoCongelado = @numeroProdutoCongelado;

Stored Procedure: Proj.UpdateToma  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.Toma **SET** massa = @massa, numeroUnidades = @numeroUnidades, sangue = @sangue, olhoVermelho = @olhoVermelho, raboInteiro = @raboInteiro, meioRabo = @meioRabo, raboPartido = @raboPartido, danificados = @danificados, semPele = @semPele **WHERE** numeroToma = @numeroToma **AND** produtoCongelado = @produtoCongelado;

## Update – Análise à Vidragem

Número da Análise à Vidragem	Mínimo	Máximo	Média	Amplitude
0	10	30	20	20

**Análise à Vidragem**

Valor Mínimo:  Valor Máximo:  Valor Médio:  Amplitude:

Número da Vidragem:

Atualizar Dados

Limpar Informação

Stored Procedure: Proj.UpdateVidragem  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.Vidragem **SET** [min] = @min, [max] = @max, med = @med, amp = @amp **WHERE**  
numeroVidragem = @numeroVidragem;

## Update – Contentores

Código do Contentor	Código Lote	Denominação do Produto	Calibre
37583384593	123456789	Ervilha	34

**Contentores**

Código do Contentor:  Código Lote:  Propriedades Organolépticas:  Temperatura do Produto:

Denominação do Produto:

Calibre:

Stored Procedure: Proj.UpdateCondiçoesAnalisePropriedades  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.CondiçoesAnalisePropriedades **SET** denominacaoProduto =  
@denominacaoProduto, calibre = @calibre, propriedadesOrganolepticas =  
@propriedadesOrganolepticas, temperaturaProduto = @temperaturaProduto, codigoLote =  
@codigoLote **WHERE** codigoContentor = @codigoContentor;

## Update – Câmara de Frescos

Número da Análise à Câmara de Frescos	Temperatura do Registrador	Temperatura da Sonda
0	23.3	56.7

**Câmara de Frescos**

Temperatura Registrador:  Temperatura Sonda:  Humidade Relativa:

Número de Análise da Câmara:

Stored Procedure: Proj.UpdateCameraFrescos  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.CameraFrescos **SET** temperaturaRegistrador = @temperaturaRegistrador,  
temperaturaSonda = @temperaturaSonda, humidadeRelativa = @humidadeRelativa **WHERE**  
numeroAnaliseCamera = @numeroAnaliseCamera;

## Update – Túnel de Congelação

Número da Análise ao Túnel de Congelação	Temperatura do Túnel	Velocidade do Tapete #1
0	-4.4	10

**Túnel de Congelação**

Número de Identificação:  Velocidade Tapete #1:  Velocidade Tapete #2:  Temperatura do Túnel:

Número da Análise ao Túnel:

Stored Procedure: Proj.UpdateTunelCongelacao  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.TunelCongelacao **SET** temperaturaTunel = @temperaturaTunel, velocidadeTapete1 = @velocidadeTapete1, velocidadeTapete2 = @velocidadeTapete2, numeroIdentificacaoTunelCongelacao = @numeroIdentificacaoTunelCongelacao **WHERE** numeroAnaliseTunelCongelacao = @numeroAnaliseTunelCongelacao;

## Update – Tanque de Salmouração

Número da Análise ao Tanque de Salmouração	Temperatura Tanque #1	Temperatura Tanque #2
0	10.0	20.0

Stored Procedure: Proj.UpdateTanqueSalmouracao  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.TanqueSalmouracao **SET** temperaturaTanque1 = @temperaturaTanque1, temperaturaTanque2 = @temperaturaTanque2, nivelAguaTanque = @nivelAguaTanque **WHERE** numeroAnaliseTanqueSalmouracao = @numeroAnaliseTanqueSalmouracao;

## Update – Máquina de Vidrar

Nome do Documento	codigoContendor	Número de Análise à Máquina de Vidrar
CDA124	37583384593	1

Stored Procedure: Proj.UpdateMaquinaVidrar  
(ver secção relativa às Stored Procedures)

**UPDATE** Proj.MaquinaVidrar **SET** temperaturaTanque = @temperaturaTanque, fluxoContinuo = @fluxoContinuo, seNCMudanca8n8 = @seNCMudanca8n8, se8n8RegistarHorario = @se8n8RegistarHorario **WHERE** nomeDocumento = @nomeDocumento **AND** codigoContendor = @codigoContendor **AND** numeroAnaliseMaquinaVidrar = @numeroAnaliseMaquinaVidrar;

Estas instruções SQL DDL são apenas as mais importantes de cada Form visual. Existem ainda instruções mínimas de SELECT para preenchimento de estruturas de dados.



# Normalização

Relação: Proj.Documentos{nomeDocumento}

Dependências Funcionais: {}

Chave Primária: {nomeDocumento}

2FN: Proj.Documentos{<<PK>> nomeDocumento}

3FN: Proj.Documentos{<<PK>> nomeDocumento}

BCFN: Proj.Documentos{<<PK>> nomeDocumento}

Relação: Proj.AnaliseARealizar{codigo, tipoAnalise}

Dependências Funcionais: {codigo} -> {tipoAnalise}

Chave Primária: {codigo}

2FN: Proj.AnaliseARealizar{<<PK>> codigo, tipoAnalise}

3FN: Proj.AnaliseARealizar{<<PK>> codigo, tipoAnalise}

BCFN: Proj.AnaliseARealizar{<<PK>> codigo, tipoAnalise}

Relação: Proj.Frequencia{codigo, designacao}

Dependências Funcionais: {codigo} -> {designacao}

Chave Primária: {codigo}

2FN: Proj.Frequencia{<<PK>> codigo, designacao}

3FN: Proj.Frequencia{<<PK>> codigo, designacao}

BCFN: Proj.Frequencia{<<PK>> codigo, designacao}

Relação: Proj.SLouEncarregada{codigo, designacao}

Dependências Funcionais: {codigo} -> {designacao}

Chave Primária: {codigo}

2FN: Proj.SLouEncarregada{<<PK>> codigo, designacao}

3FN: Proj.SLouEncarregada{<<PK>> codigo, designacao}

BCFN: Proj.SLouEncarregada{<<PK>> codigo, designacao}

Relação: Proj.TipoProduto{codigo, designacao}

Dependências Funcionais: {codigo} -> {designacao}

Chave Primária: {codigo}

2FN: Proj.TipoProduto{<<PK>> codigo, designacao}

3FN: Proj.TipoProduto{<<PK>> codigo, designacao}

BCFN: Proj.TipoProduto{<<PK>> codigo, designacao}



Relação: Proj.ClassificacaoFrescura{codigo,designacao}

Dependências Funcionais: {codigo}->{designacao}

Chave Primária: {codigo}

2FN: Proj.ClassificacaoFrescura{<<PK>> codigo,designacao}

3FN: Proj.ClassificacaoFrescura{<<PK>> codigo,designacao}

BCFN: Proj.ClassificacaoFrescura{<<PK>> codigo,designacao}

Relação: Proj.AnaliseOrganoleptica{numeroAnalise,cotacaoTotal,classificacaoFrescura}

Dependências Funcionais: {numeroAnalise}->{cotacaoTotal,classificacaoFrescura}

Chave Primária: {numeroAnalise}

2FN: Proj.AnaliseOrganoleptica{<<PK>> numeroAnalise,cotacaoTotal,classificacaoFrescura}

3FN: Proj.AnaliseOrganoleptica{<<PK>> numeroAnalise,cotacaoTotal,classificacaoFrescura}

BCFN: Proj.AnaliseOrganoleptica{<<PK>> numeroAnalise,cotacaoTotal,classificacaoFrescura}

Relação: Proj.VerificacaoCalibre{numeroVerificacao,massa,unidades,classificacao}

Dependências Funcionais: {numeroVerificacao}->{massa,unidades,classificacao}

Chave Primária: {numeroVerificacao}

2FN: Proj.VerificacaoCalibre{<<PK>> numeroVerificacao,massa,unidades,classificacao}

3FN: Proj.VerificacaoCalibre{<<PK>> numeroVerificacao,massa,unidades,classificacao}

BCFN: Proj.VerificacaoCalibre{<<PK>> numeroVerificacao,massa,unidades,classificacao}

Relação:

Proj.ProdutoFresco{numeroProdutoFresco,nAnaliseOrganoleptica,nVerificacaoCalibre}

Dependências Funcionais: {numeroProdutoFresco}-

>{nAnaliseOrganoleptica,nVerificacaoCalibre}

Chave Primária: {numeroProdutoFresco,nAnaliseOrganoleptica,nVerificacaoCalibre}

2FN: Proj.ProdutoFresco{<<PK>>  
numeroProdutoFresco,nAnaliseOrganoleptica,nVerificacaoCalibre}

3FN: Proj.ProdutoFresco{<<PK>>  
numeroProdutoFresco,nAnaliseOrganoleptica,nVerificacaoCalibre}

BCFN: Proj.ProdutoFresco{<<PK>>  
numeroProdutoFresco,nAnaliseOrganoleptica,nVerificacaoCalibre}

Relação: Proj.Vidragem{numeroVidragem,min,max,amp,med}  
Dependências Funcionais: {numeroVidragem}->{min,max,med,amp}  
Chave Primária: {numeroVidragem}

2FN: Proj.Vidragem{<<PK>> numeroVidragem,min,max,amp,med}  
3FN: Proj.Vidragem{<<PK>> numeroVidragem,min,max,amp,med}  
BCFN: Proj.Vidragem{<<PK>> numeroVidragem,min,max,amp,med}

Relação: Proj.ProdutoCongelado{numeroProdutoCongelado,nVidragem}  
Dependências Funcionais: {numeroProdutoCongelado}<-{nVidragem}  
Chave Primária: {numeroProdutoCongelado,nVidragem}

2FN: Proj.ProdutoCongelado{<<PK>> numeroProdutoCongelado,nVidragem}  
3FN: Proj.ProdutoCongelado{<<PK>> numeroProdutoCongelado,nVidragem}  
BCFN: Proj.ProdutoCongelado{<<PK>> numeroProdutoCongelado,nVidragem}

Relacao:  
Proj.Toma{numeroToma,produtoCongelado,produtoVidragem,massa,numeroUnidades,sangue  
,olhoVermelho,raboInteiro,raboInteiro,raboPartido,danificados,semPele}  
Dependências Funcionais: {produtoCongelado,produtoVidragem}-  
>{numeroToma,massa,numeroUnidades,sangue,olhoVermelho,raboInteiro,raboInteiro,raboPa  
rtido,danificados,semPele}  
Chave Primária: {numeroToma,produtoCongelado,produtoVidragem}

2FN: Proj.Toma{<<PK>> numeroToma, <<PK>> produtoCongelado, <<PK>>  
produtoVidragem,massa,numeroUnidades,sangue,olhoVermelho,raboInteiro,raboInteiro,rabo  
Partido,danificados,semPele}

3FN: Proj.Toma{<<PK>> numeroToma, <<PK>> produtoCongelado, <<PK>>  
produtoVidragem,massa,numeroUnidades,sangue,olhoVermelho,raboInteiro,raboInteiro,rabo  
Partido,danificados,semPele}

BCFN: Proj.Toma{<<PK>> numeroToma, <<PK>> produtoCongelado, <<PK>>  
produtoVidragem,massa,numeroUnidades,sangue,olhoVermelho,raboInteiro,raboInteiro,rabo  
Partido,danificados,semPele}

Relacao:

Proj.AnaliseDefeitosProdutos{nomeDocumento,efetuadoPor,dataAnalise,loteAtribuido,origem  
Produto,origemProduto,analiseOrganoleptica,verificacaoCalibre,produtoCongelado,vidragem}

Dependências Funcionais: {nomeDocumento,efetuadoPor,dataAnalise,loteAtribuido}-  
>{origemProduto,origemProduto,analiseOrganoleptica,verificacaoCalibre,produtoCongelado,vi  
dragem}

Chave Primária: {nomeDocumento,efetuadoPor,dataAnalise,loteAtribuido}

2FN: Proj.AnaliseDefeitosProdutos{<<PK>> nomeDocumento,<<PK>> efetuadoPor,<<PK>>  
dataAnalise,<<PK>>

loteAtribuido,origemProduto,origemProduto,analiseOrganoleptica,verificacaoCalibre,produto  
Congelado,vidragem}

3FN: Proj.AnaliseDefeitosProdutos{<<PK>> nomeDocumento,<<PK>> efetuadoPor,<<PK>>  
dataAnalise,<<PK>>

loteAtribuido,origemProduto,origemProduto,analiseOrganoleptica,verificacaoCalibre,produto  
Congelado,vidragem}

BCVN: Proj.AnaliseDefeitosProdutos{<<PK>> nomeDocumento,<<PK>> efetuadoPor,<<PK>>  
dataAnalise,<<PK>>

loteAtribuido,origemProduto,origemProduto,analiseOrganoleptica,verificacaoCalibre,produto  
Congelado,vidragem}

Relação: Proj.NumeroLinha{codigo,designacao}

Dependências Funcionais: {codigo}->{designacao}

Chave Primária: {codigo}

2FN: Proj.NumeroLinha{<<PK>> codigo,designacao}

3FN: Proj.NumeroLinha{<<PK>> codigo,designacao}

BCFN: Proj.NumeroLinha{<<PK>> codigo,designacao}

Relação: Proj.NumeroTurno{codigo,designacao}

Dependências Funcionais: {codigo}->{designacao}

Chave Primária: {codigo}

2FN: Proj.NumeroTurno{<<PK>> codigo,designacao}

3FN: Proj.NumeroTurno{<<PK>> codigo,designacao}

BCFN: Proj.NumeroTurno{<<PK>> codigo,designacao}

Relação:

Proj.CameraFrescos{numeroAnaliseCamera,temperaturaRegistador,temperaturaSonda,humidadeRelativa}

Dependências Funcionais: {numeroAnaliseCamera}-

>{temperaturaRegistador,temperaturaSonda,humidadeRelativa}

Chave Primária: {numeroAnaliseCamera}

2FN: Proj.CameraFrescos{<<PK>>

numeroAnaliseCamera,temperaturaRegistador,temperaturaSonda,humidadeRelativa}

3FN: Proj.CameraFrescos{<<PK>>

numeroAnaliseCamera,temperaturaRegistador,temperaturaSonda,humidadeRelativa}

BCFN: Proj.CameraFrescos{<<PK>>

numeroAnaliseCamera,temperaturaRegistador,temperaturaSonda,humidadeRelativa}

Relação:

Proj.TunelCongelacao{numeroAnaliseTunelCongelacao,temperaturaTunel,velocidadeTunel1,velocidadeTunel2,numeroidentificacaoTunelCongelacao}

Dependências Funcionais: {numeroAnaliseTunelCongelacao}-

>{temperaturaTunel,velocidadeTunel1,velocidadeTunel2,numeroidentificacaoTunelCongelacao}

Chave Primária: {numeroAnaliseTunelCongelacao}

2FN: Proj.TunelCongelacao{<<PK>>

numeroAnaliseTunelCongelacao,temperaturaTunel,velocidadeTunel1,velocidadeTunel2,numeroidentificacaoTunelCongelacao}

3FN: Proj.TunelCongelacao{<<PK>>

numeroAnaliseTunelCongelacao,temperaturaTunel,velocidadeTunel1,velocidadeTunel2,numeroidentificacaoTunelCongelacao}

BCFN: Proj.TunelCongelacao{<<PK>>

numeroAnaliseTunelCongelacao,temperaturaTunel,velocidadeTunel1,velocidadeTunel2,numeroidentificacaoTunelCongelacao}

Relação:

Proj.TanqueSalmouracao{numeroAnaliseTanqueSalmouracao,temperaturaTanque1,temperaturaTanque2,nivelAguaTanque}

Dependências Funcionais: {numeroAnaliseTanqueSalmouracao}->{temperaturaTanque1,temperaturaTanque2,nivelAguaTanque}

Chave Primária: {numeroAnaliseTanqueSalmouracao}

2FN: Proj.TanqueSalmouracao{<<PK>>

numeroAnaliseTanqueSalmouracao,temperaturaTanque1,temperaturaTanque2,nivelAguaTanque}

3FN: Proj.TanqueSalmouracao{<<PK>>

numeroAnaliseTanqueSalmouracao,temperaturaTanque1,temperaturaTanque2,nivelAguaTanque}

BCFN: Proj.TanqueSalmouracao{<<PK>>

numeroAnaliseTanqueSalmouracao,temperaturaTanque1,temperaturaTanque2,nivelAguaTanque}

Relação:

Proj.MaquinaVidrar{nomeDocumento,codigoContentor,numeroAnaliseMaquinaVidrar,temperaturaTanque,fluxoContinuo,seNCMudanca8n8,se8n8RegistarHorario}

Dependências Funcionais: {nomeDocumento,codigoContentor,numeroAnaliseMaquinaVidrar}->{temperaturaTanque,fluxoContinuo,,seNCMudanca8n8,se8n8RegistarHorario}

Chave Primária: {nomeDocumento,codigoContentor,numeroAnaliseMaquinaVidrar}

2FN: Proj.MaquinaVidrar{<<PK>> nomeDocumento,<<PK>> codigoContentor,<<PK>>

numeroAnaliseMaquinaVidrar,temperaturaTanque,fluxoContinuo,seNCMudanca8n8,se8n8RegistarHorario}

3FN: Proj.MaquinaVidrar{<<PK>> nomeDocumento,<<PK>> codigoContentor,<<PK>>

numeroAnaliseMaquinaVidrar,temperaturaTanque,fluxoContinuo,seNCMudanca8n8,se8n8RegistarHorario}

BCFN: Proj.MaquinaVidrar{<<PK>> nomeDocumento,<<PK>> codigoContentor,<<PK>>

numeroAnaliseMaquinaVidrar,temperaturaTanque,fluxoContinuo,seNCMudanca8n8,se8n8RegistarHorario}

Relação: Proj.Fabrica{codigo,designacao}

Dependências Funcionais: {codigo}->{designacao}

Chave Primária: {codigo}

2FN: Proj.Fabrica{<<PK>> codigo,designacao}

3FN: Proj.Fabrica{<<PK>> codigo,designacao}

BCFN: Proj.Fabrica{<<PK>> codigo,designacao}

Relação:

Proj.CondicoesAnalise{nomeDocumento,codigoContentor,codigoLote,denominacaoProduto,dataAnalise,horaAnalise,controloEfetuadoPor,calibre,propriedadesOrganolepticas,temperaturaProduto,linha,turno,cameraFrescos,tunelCongelacao,tanqueSalmouracao}

Dependências Funcionais: {nomeDocumento,codigoContentor}-

>{dataAnalise,horaAnalise,controloEfetuadoPor,linha,turno,cameraFrescos,tunelCongelacao,tanqueSalmouracao}

{codigoContentor}-

>{codigoLote,denominacaoProduto,calibre,propriedadesOrganolepticas,temperaturaProduto}

Chave Primária: {nomeDocumento,codigoContentor}

2FN: Proj.CondicoesAnalise{<<PK>> nomeDocumento, <<PK>> codigoContentor, codigoLote,denominacaoProduto,dataAnalise,horaAnalise,controloEfetuadoPor,calibre,propriedadesOrganolepticas,temperaturaProduto,linha,turno,cameraFrescos,tunelCongelacao,tanqueSalmouracao}

3FN: Proj.CondicoesAnalise{<<PK>> nomeDocumento, <<PK>> codigoContentor,dataAnalise, horaAnalise, controloEfetuadoPor, linha, turno, cameraFrescos, tunelCongelacao, tanqueSalmouracao}

Proj.CondicoesAnalisePropriedades{<<PK>>

codigoContentor,codigoLote,denominacaoProduto,calibre,propriedadesOrganolepticas,temperaturaProduto}

BCFN: Proj.CondicoesAnalise{<<PK>> nomeDocumento, <<PK>>

codigoContentor,dataAnalise, horaAnalise, controloEfetuadoPor, linha, turno, cameraFrescos, tunelCongelacao, tanqueSalmouracao}

Proj.CondicoesAnalisePropriedades{<<PK>>

codigoContentor,codigoLote,denominacaoProduto,calibre,propriedadesOrganolepticas,temperaturaProduto}

Relação: Proj.ClorosArrefecedor{numeroControlo,cloroArrefecedor1,cloroArrefecedor2}

Dependências Funcionais: {numeroControlo}->{cloroArrefecedor1,cloroArrefecedor2}

Chave Primária: {numeroControlo}

2FN: Proj.ClorosArrefecedor{<<PK>> numeroControlo,cloroArrefecedor1,cloroArrefecedor2}

3FN: Proj.ClorosArrefecedor{<<PK>> numeroControlo,cloroArrefecedor1,cloroArrefecedor2}

BCFN: Proj.ClorosArrefecedor{<<PK>> numeroControlo,cloroArrefecedor1,cloroArrefecedor2}



Relação:

Proj.ControloProcessoProdutivo{numeroControlo,cloroChiller,cloroControladorEntrada,oxigenio,odorOzono,controladorLavadoras,geracaoOzono,leituraCloroRegistador}

Dependências Funcionais: {numeroControlo}-

>{cloroChiller,cloroControladorEntrada,oxigenio,odorOzono,controladorLavadoras,geracaoOzono,leituraCloroRegistador}

Chave Primária: {numeroControlo}

2FN: Proj.ControloProcessoProdutivo{<<PK>>

numeroControlo,cloroChiller,cloroControladorEntrada,oxigenio,odorOzono,controladorLavadoras,geracaoOzono,leituraCloroRegistador}

3FN: Proj.ControloProcessoProdutivo{<<PK>>

numeroControlo,cloroChiller,cloroControladorEntrada,oxigenio,odorOzono,controladorLavadoras,geracaoOzono,leituraCloroRegistador}

BCFN: Proj.ControloProcessoProdutivo{<<PK>>

numeroControlo,cloroChiller,cloroControladorEntrada,oxigenio,odorOzono,controladorLavadoras,geracaoOzono,leituraCloroRegistador}

Relação:

Proj.ClorosMangueira{numeroControlo,cloroMangueira1,cloroMangueira2,cloroMangueira4,cloroMangueira6,cloroMangueira7,cloroMangueira9,cloroMangueiraExterior}

Dependências Funcionais: {numeroControlo}-

>{cloroMangueira1,cloroMangueira2,cloroMangueira4,cloroMangueira6,cloroMangueira7,cloroMangueira9,cloroMangueiraExterior}

Chave Primária: {numeroControlo}

2FN: Proj.ClorosMangueira{<<PK>>

numeroControlo,cloroMangueira1,cloroMangueira2,cloroMangueira4,cloroMangueira6,cloroMangueira7,cloroMangueira9,cloroMangueiraExterior}

3FN: Proj.ClorosMangueira{<<PK>>

numeroControlo,cloroMangueira1,cloroMangueira2,cloroMangueira4,cloroMangueira6,cloroMangueira7,cloroMangueira9,cloroMangueiraExterior}

BCFN: Proj.ClorosMangueira{<<PK>>

numeroControlo,cloroMangueira1,cloroMangueira2,cloroMangueira4,cloroMangueira6,cloroMangueira7,cloroMangueira9,cloroMangueiraExterior}

Relação:

Proj.ClorosEOzono{nomeDocumento,dataAnalise,horaAnalise,produtoEmProcessamento,efetuadoPor,fabrica,controloProcessoProdutivo}

Dependências Funcionais:

{nomeDocumento,dataAnalise,horaAnalise,produtoEmProcessamento,efetuadoPor}->{fabrica,controloProcessoProdutivo}

Chave Primária:

{nomeDocumento,dataAnalise,horaAnalise,produtoEmProcessamento,efetuadoPor}

2FN: Proj.ClorosEOzono{<<PK>> nomeDocumento,<<PK>> dataAnalise,<<PK>> horaAnalise,<<PK>> produtoEmProcessamento,<<PK>> efetuadoPor, fabrica,controloProcessoProdutivo}

3FN: Proj.ClorosEOzono{<<PK>> nomeDocumento,<<PK>> dataAnalise,<<PK>> horaAnalise,<<PK>> produtoEmProcessamento,<<PK>> efetuadoPor, fabrica,controloProcessoProdutivo}

BCFN: Proj.ClorosEOzono{<<PK>> nomeDocumento,<<PK>> dataAnalise,<<PK>> horaAnalise,<<PK>> produtoEmProcessamento,<<PK>> efetuadoPor, fabrica,controloProcessoProdutivo}

## Índices

Devido ao facto de na empresa se realizarem, em média, para cada produto cerca de 12 inserções diárias, e cerca de 30 verificações por hora, podemos afirmar que verificações de dados têm um impacto enorme no uso da base de dados, quando comparando com o número de inserções. De tal forma, elas foram o nosso maior foco na escolha de índices.

**Enumerados:** As tabelas de enumerados (TipoProduto, FamíliaProduto, AnaliseARealizar, SLouEncarregada, Frequencia, OperacaoARealizar, ClassificacaoFrescura, NumeroLinha, NumeroTurno e Fabrica), devido à sua natureza de pesquisa e facto da chave primária ser pequena, mantiveram os seus índices default (Unique Clustered Index).

**PlanoDeTrabalho, PlanoDeTrabalhoProduto e PlanoDeTrabalhoAcao:** Devido à sua pesquisa ser realizada principalmente através de primary keys, estas relações/tabelas mantiveram os seus índices default (Unique Clustered Index).

**Documentos:** Devido a ser uma relação/tabela contendo apenas um atributo, o índice default é suficiente (Unique Clustered Index).

**AnaliseDefeitosProdutos, Toma, Vidragem e VerificacaoCalibre:** Devido a pesquisas principalmente baseadas em chaves primárias, as relações/tabelas mantiveram os seus índices default (Unique Clustered Index).

**ProdutoFresco, ProdutoCongelado e AnaliseOrganoleptica:** Devido ao tipo de pesquisas efetuadas, foi verificado um aumento na eficiência de pesquisas sobre estas relações/tabelas ao se adicionar um índice novo em ambas.

```
CREATE INDEX Idx_NC_PF ON
Proj.ProdutoFresco(nAnaliseOrganoleptica,nVerificacaoCalibre);
CREATE INDEX Idx_NC_PC ON Proj.ProdutoCongelado(nVidragem);
CREATE INDEX Idx_NC_AO ON Proj.AnaliseOrganoleptica(classificacaoFrescura);
```

**CondicoesAnalise, MaquinaVidrar, TanqueSalmouracao, TunelCongelacao, CameraFrescos e CondicoesAnalisePropriedades:** Devido ao tipo de pesquisas efetuadas serem principalmente efetuadas sobre chaves primárias, as relações/tabelas mantiveram os seus índices default (Unique Clustered Index).

**ClorosEOzono, ControloProcessoProdutivo, ClorosArrefecedor, ClorosLavadora e ClorosMangueira:** Devido ao tipo de pesquisas efetuadas serem principalmente efetuadas sobre chaves primárias, as relações/tabelas mantiveram os seus índices default (Unique Clustered Index).

Devido à maior parte das nossas instruções ou não terem a chave reservada “WHERE”, ou serem parametrizadas, não existem casos em que seja de benefício usar índices Filtered.

## Triggers

Devido a documentos apenas poderem ser de um tipo (Análise a Defeitos de Produto, Cloros e Ozono ou Condições de Análise), triggers foram feitos para não permitir que documentos possam ser inseridos em mais que uma relação/tabela que derive de documentos.

```
CREATE TRIGGER CEO_DOC_trigger ON Proj.ClorosEOzono
AFTER INSERT, UPDATE
AS

    DECLARE @doc1 tinyint
    DECLARE @doc2 tinyint

    SELECT @doc1=count(*) FROM Proj.AnaliseDefeitosProdutos WHERE nomeDocumento in
(SELECT nomeDocumento FROM inserted )
    SELECT @doc2=count(*) FROM Proj.CondicoesAnalise WHERE nomeDocumento in
(SELECT nomeDocumento FROM inserted )

    IF @doc1=1 or @doc2=1
        BEGIN
            RAISERROR ('Document Duplicated.', 1,1);
            ROLLBACK TRAN;
        END
    ELSE
        PRINT 'Log: Operation completed with success!'

GO
```

```

CREATE TRIGGER ADF_DOC_trigger ON Proj.AnaliseDefeitosProdutos
AFTER INSERT, UPDATE
AS

    DECLARE @doc1 tinyint
    DECLARE @doc2 tinyint

    SELECT @doc1=count(*) FROM Proj.ClorosEOzono WHERE nomeDocumento in (SELECT
nomeDocumento FROM inserted )
    SELECT @doc2=count(*) FROM Proj.CondicoesAnalise WHERE nomeDocumento in
(SELECT nomeDocumento FROM inserted )

    IF @doc1=1 or @doc2=1
        begin
            RAISERROR ('Document Duplicated.', 1,1);
            ROLLBACK TRAN;
        END
    ELSE
        PRINT 'Log: Operation completed with success!'

GO

CREATE TRIGGER CA_DOC_trigger ON Proj.CondicoesAnalise
AFTER INSERT, UPDATE
AS

    DECLARE @doc1 tinyint
    DECLARE @doc2 tinyint

    SELECT @doc1=count(*) FROM Proj.ClorosEOzono WHERE nomeDocumento in (SELECT
nomeDocumento FROM inserted )
    SELECT @doc2=count(*) FROM Proj.AnaliseDefeitosProdutos WHERE nomeDocumento in
(SELECT nomeDocumento FROM inserted )

    IF @doc1=1 or @doc2=1
        begin
            RAISERROR ('Document Duplicated.', 1,1);
            ROLLBACK TRAN;
        END
    ELSE
        PRINT 'Log: Operation completed with success!'

GO

```

Adicionalmente, devido à base de dados não dever permitir deleções, criaram-se “instead of” triggers para não as permitir.

```

CREATE TRIGGER Proj.AR_delete_trigger ON Proj.AnaliseARealizar
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

REATE TRIGGER Proj.ADP_delete_trigger ON Proj.AnaliseDefeitosProdutos
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

```

```

CREATE TRIGGER Proj.AO_delete_trigger ON Proj.AnaliseOrganoleptica
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.CF_delete_trigger ON Proj.CameraFrescos
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.CLF_delete_trigger ON Proj.ClassificacaoFrescura
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.CA_delete_trigger ON Proj.ClorosArrefecedor
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.CEO_delete_trigger ON Proj.ClorosEOzono
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.CL_delete_trigger ON Proj.ClorosLavadora
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.CM_delete_trigger ON Proj.ClorosMangueira
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

```

```
CREATE TRIGGER Proj.COA_delete_trigger ON Proj.CondicoesAnalise  
INSTEAD OF DELETE  
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.CAP_delete_trigger ON Proj.CondicoesAnalisePropriedades  
INSTEAD OF DELETE  
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.CPP_delete_trigger ON Proj.ControloProcessoProdutivo  
INSTEAD OF DELETE  
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.D_delete_trigger ON Proj.Documentos  
INSTEAD OF DELETE  
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.FP_delete_trigger ON Proj.FamiliaProduto  
INSTEAD OF DELETE  
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.F_delete_trigger ON Proj.Frequencia  
INSTEAD OF DELETE  
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.MV_delete_trigger ON Proj.MaquinaVidrar  
INSTEAD OF DELETE  
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.NL_delete_trigger ON Proj.NumeroLinha  
INSTEAD OF DELETE  
AS
```

```
RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.NT_delete_trigger ON Proj.NumeroTurno  
INSTEAD OF DELETE  
AS
```

```
RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.OAR_delete_trigger ON Proj.OperacaoARealizar  
INSTEAD OF DELETE  
AS
```

```
RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.PDT_delete_trigger ON Proj.PlanoDeTrabalho  
INSTEAD OF DELETE  
AS
```

```
RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.PDTA_delete_trigger ON Proj.PlanoDeTrabalhoAcao  
INSTEAD OF DELETE  
AS
```

```
RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.PTP_delete_trigger ON Proj.PlanoDeTrabalhoProduto  
INSTEAD OF DELETE  
AS
```

```
RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```
CREATE TRIGGER Proj.PC_delete_trigger ON Proj.ProdutoCongelado  
INSTEAD OF DELETE  
AS
```

```
RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

```

CREATE TRIGGER Proj.PF_delete_trigger ON Proj.ProdutoFresco
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.SLE_delete_trigger ON Proj.SLouEncarregada
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.TS_delete_trigger ON Proj.TanqueSalmouracao
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.TP_delete_trigger ON Proj.TipoProduto
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.T_delete_trigger ON Proj.Toma
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.TC_delete_trigger ON Proj.TunelCongelacao
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

CREATE TRIGGER Proj.VC_delete_trigger ON Proj.VerificacaoCalibre
INSTEAD OF DELETE
AS

    RAISERROR ('Operation Delete Invalid.', 2,1);

GO

```



```
CREATE TRIGGER Proj.V_delete_trigger ON Proj.Vidragem
INSTEAD OF DELETE
AS
```

```
    RAISERROR ('Operation Delete Invalid.', 2,1);
```

```
GO
```

## Stored Procedures

Na realização deste projeto, stored procedures foram principalmente utilizadas para inserção e atualização de dados.

```
CREATE PROC Proj.InsertControloProcessoProdutivoBatch @numeroControlo INTEGER,
@cloroChiller DECIMAL(4,1), @oxigenio TINYINT, @odorOzono BIT,
@cloroControladorEntrada DECIMAL(4,1), @controladorLavadoras DECIMAL(4,1),
@geracaoOzono TINYINT, @leituraCloroRegistador TINYINT, @cloroArrefecedor1
DECIMAL(4,1), @cloroArrefecedor2 DECIMAL(4,1), @cloroMangueira1 DECIMAL(4,1),
@cloroMangueira2 DECIMAL(4,1), @cloroMangueira4 DECIMAL(4,1), @cloroMangueira6
DECIMAL(4,1), @cloroMangueira7 DECIMAL(4,1), @cloroMangueira9 DECIMAL(4,1),
@cloroMangueiraExterior DECIMAL(4,1), @cloroLavadora1 DECIMAL(4,1), @cloroLavadora2
DECIMAL(4,1), @cloroLavadoraEsgoto1 DECIMAL(4,1), @cloroLavadoraEsgoto2 DECIMAL(4,1)
AS
    INSERT
ControloProcessoProdutivo(numeroControlo,cloroChiller,oxigenio,odorOzono,cloroControla
dorEntrada,controladorLavadoras,geracaoOzono,leituraCloroRegistador)
VALUES(@numeroControlo,@cloroChiller,@oxigenio,@odorOzono,@cloroControladorEntrada,@co
ntroladorLavadoras,@geracaoOzono,@leituraCloroRegistador);
    INSERT
ClorosArrefecedor(numeroControlo,cloroArrefecedor1,cloroArrefecedor2)
VALUES(@numeroControlo,@cloroArrefecedor1,@cloroArrefecedor2);
    INSERT
ClorosLavadora(numeroControlo,cloroLavadora1,cloroLavadora2,cloroLavadoraEsgoto1,cloro
LavadoraEsgoto2)
VALUES(@numeroControlo,@cloroLavadora1,@cloroLavadora2,@cloroLavadoraEsgoto1,@cloroLav
adoraEsgoto2);
    INSERT
ClorosMangueira(numeroControlo,cloroMangueira1,cloroMangueira2,cloroMangueira4,cloroMa
ngueira6,cloroMangueira7,cloroMangueira9,cloroMangueiraExterior)
VALUES(@numeroControlo,@cloroMangueira1,@cloroMangueira2,@cloroMangueira4,@cloroMangue
ira6,@cloroMangueira7,@cloroMangueira9,@cloroMangueiraExterior);
GO

CREATE PROC Proj.InsertPlanoDeTrabalhoProduto @nomeProduto VARCHAR(255),
@familiaProduto TINYINT, @tipoProduto TINYINT
AS
    INSERT Proj.PlanoDeTrabalhoProduto(nomeProduto,familiaProduto,tipoProduto)
VALUES (@nomeProduto,@familiaProduto,@tipoProduto);
GO

CREATE PROC Proj.UpdatePlanoDeTrabalhoProduto @nomeProduto VARCHAR(255),
@familiaProduto TINYINT, @tipoProduto TINYINT
AS
    UPDATE Proj.PlanoDeTrabalhoProduto SET familiaProduto = @familiaProduto,
tipoProduto = @tipoProduto WHERE nomeProduto = @nomeProduto;
GO
```

```

CREATE PROC Proj.UpdateControloProcessoProdutivoBatch @numeroControlo INTEGER,
@cloroChiller DECIMAL(4,1), @oxigenio TINYINT, @odorOzono BIT,
@cloroControladorEntrada DECIMAL(4,1), @controladorLavadoras DECIMAL(4,1),
@geracaoOzono TINYINT, @leituraCloroRegistador TINYINT, @cloroArrefecedor1
DECIMAL(4,1), @cloroArrefecedor2 DECIMAL(4,1), @cloroMangueira1 DECIMAL(4,1),
@cloroMangueira2 DECIMAL(4,1), @cloroMangueira4 DECIMAL(4,1), @cloroMangueira6
DECIMAL(4,1), @cloroMangueira7 DECIMAL(4,1), @cloroMangueira9 DECIMAL(4,1),
@cloroMangueiraExterior DECIMAL(4,1), @cloroLavadora1 DECIMAL(4,1), @cloroLavadora2
DECIMAL(4,1), @cloroLavadoraEsgoto1 DECIMAL(4,1), @cloroLavadoraEsgoto2 DECIMAL(4,1)
AS
    UPDATE ControloProcessoProdutivo SET cloroChiller = @cloroChiller, oxigenio =
@oxigenio, odorOzono = @odorOzono, cloroControladorEntrada =
@cloroControladorEntrada, controladorLavadoras = @controladorLavadoras, geracaoOzono
= @geracaoOzono, leituraCloroRegistador = @leituraCloroRegistador WHERE
numeroControlo = @numeroControlo;
    UPDATE ClorosArrefecedor SET cloroArrefecedor1 = @cloroArrefecedor1,
cloroArrefecedor2 = @cloroArrefecedor2 WHERE numeroControlo = @numeroControlo;
    UPDATE ClorosLavadora SET cloroLavadora1 = @cloroLavadora1, cloroLavadora2 =
@cloroLavadora2, cloroLavadoraEsgoto1 = @cloroLavadoraEsgoto1, cloroLavadoraEsgoto2 =
@cloroLavadoraEsgoto2 WHERE numeroControlo = @numeroControlo;
    UPDATE ClorosMangueira SET cloroMangueira1 = @cloroMangueira1, cloroMangueira2
= @cloroMangueira2, cloroMangueira4 = @cloroMangueira4, cloroMangueira6 =
@cloroMangueira6, cloroMangueira7 = @cloroMangueira7, cloroMangueira9 =
@cloroMangueira9, cloroMangueiraExterior = @cloroMangueiraExterior WHERE
numeroControlo = @numeroControlo;
GO

CREATE PROC Proj.InsertPlanoDeTrabalhoAcao @analise TINYINT, @instrumentos
VARCHAR(255), @efetuadoPor TINYINT
AS
    INSERT Proj.PlanoDeTrabalhoAcao(analise,instrumentos,efetuadoPor) VALUES
(@analise,@instrumentos,@efetuadoPor);
GO

CREATE PROC Proj.UpdatePlanoDeTrabalhoAcao @analise TINYINT, @instrumentos
VARCHAR(255), @efetuadoPor TINYINT
AS
    UPDATE Proj.PlanoDeTrabalhoAcao SET instrumentos = @instrumentos, efetuadoPor
= @efetuadoPor WHERE analise = @analise;
GO

CREATE PROC Proj.InsertPlanoDeTrabalho @nomeProduto VARCHAR(255), @analise TINYINT,
@RDQsUtilizadas VARCHAR(255), @frequencia TINYINT, @operacao TINYINT, @documento
VARCHAR(255)
AS
    INSERT
Proj.PlanoDeTrabalho(nomeProduto,analise,RDQsUtilizadas,frequencia,operacao,documento)
VALUES (@nomeProduto,@analise,@RDQsUtilizadas,@frequencia,@operacao,@documento);
GO

CREATE PROC Proj.UpdatePlanoDeTrabalho @nomeProduto VARCHAR(255), @analise TINYINT,
@RDQsUtilizadas VARCHAR(255), @frequencia TINYINT, @operacao TINYINT, @documento
VARCHAR(255)
AS
    UPDATE Proj.PlanoDeTrabalho SET RDQsUtilizadas = @RDQsUtilizadas, frequencia =
@frequencia, operacao = @operacao, documento = @documento WHERE nomeProduto =
@nomeProduto AND analise = @analise;
GO

```

```

CREATE PROC Proj.InsertAnaliseOrganoleptica @numeroAnalise INT, @cotacaoTotal
DECIMAL(10,1), @classificacaoFrescura TINYINT
AS
    INSERT
Proj.AnaliseOrganoleptica(numeroAnalise,cotacaoTotal,classificacaoFrescura) VALUES
(@numeroAnalise,@cotacaoTotal,@classificacaoFrescura);
GO

CREATE PROC Proj.UpdateAnaliseOrganoleptica @numeroAnalise INT, @cotacaoTotal
DECIMAL(10,1), @classificacaoFrescura TINYINT
AS
    UPDATE Proj.AnaliseOrganoleptica SET cotacaoTotal = @cotacaoTotal,
classificacaoFrescura = @classificacaoFrescura WHERE numeroAnalise = @numeroAnalise;
GO

CREATE PROC Proj.InsertVerificacaoCalibre @numeroVerificacao INT, @massa
DECIMAL(10,1), @unidades DECIMAL(10,1), @classificacao DECIMAL(10,1)
AS
    INSERT Proj.VerificacaoCalibre(numeroVerificacao,massa,unidades,classificacao)
VALUES (@numeroVerificacao,@massa,@unidades,@classificacao);
GO

CREATE PROC Proj.UpdateVerificacaoCalibre @numeroVerificacao INT, @massa
DECIMAL(10,1), @unidades DECIMAL(10,1), @classificacao DECIMAL(10,1)
AS
    UPDATE Proj.VerificacaoCalibre SET massa = @massa, unidades = @unidades,
classificacao = @classificacao WHERE numeroVerificacao = @numeroVerificacao;
GO

CREATE PROC Proj.InsertProdutoFresco @numeroProdutoFresco INT, @nAnaliseOrganoleptica
INT, @nVerificacaoCalibre INT
AS
    INSERT
Proj.ProdutoFresco(numeroProdutoFresco,nAnaliseOrganoleptica,nVerificacaoCalibre)
VALUES (@numeroProdutoFresco,@nAnaliseOrganoleptica,@nVerificacaoCalibre);
GO

CREATE PROC Proj.UpdateProdutoFresco @numeroProdutoFresco INT, @nAnaliseOrganoleptica
INT, @nVerificacaoCalibre INT
AS
    UPDATE Proj.ProdutoFresco SET nAnaliseOrganoleptica = @nAnaliseOrganoleptica,
nVerificacaoCalibre = @nVerificacaoCalibre WHERE numeroProdutoFresco =
@numeroProdutoFresco;
GO

CREATE PROC Proj.InsertVidragem @numeroVidragem INT, @min TINYINT, @max TINYINT, @med
TINYINT, @amp TINYINT
AS
    INSERT Proj.Vidragem(numeroVidragem,[min],[max],med,amp) VALUES
(@numeroVidragem,@min,@max,@med,@amp);
GO

CREATE PROC Proj.UpdateVidragem @numeroVidragem INT, @min TINYINT, @max TINYINT, @med
TINYINT, @amp TINYINT
AS
    UPDATE Proj.Vidragem SET [min] = @min, [max] = @max, med = @med, amp = @amp
WHERE numeroVidragem = @numeroVidragem;
GO

```

```

CREATE PROC Proj.InsertProdutoCongelado @numeroProdutoCongelado INT, @nVidragem INT
AS
    INSERT Proj.ProdutoCongelado(numeroProdutoCongelado,nVidragem) VALUES
    (@numeroProdutoCongelado,@nVidragem);
GO

CREATE PROC Proj.UpdateProdutoCongelado @numeroProdutoCongelado INT, @nVidragem INT
AS
    UPDATE Proj.ProdutoCongelado SET nVidragem = @nVidragem WHERE
    numeroProdutoCongelado = @numeroProdutoCongelado;
GO

CREATE PROC Proj.InsertToma @numeroToma TINYINT, @produtoCongelado INT, @massa
DECIMAL(10,1), @numeroUnidades DECIMAL(10,1), @sangue DECIMAL(4,1), @olhoVermelho
DECIMAL(4,1), @raboInteiro DECIMAL(4,1), @meioRabo DECIMAL(4,1), @raboPartido
DECIMAL(4,1), @danificados DECIMAL(4,1), @semPele DECIMAL(4,1)
AS
    INSERT
    Proj.Toma(numeroToma,produtoCongelado,masa,numeroUnidades,sangue,olhoVermelho,raboInte
    iro,meioRabo,raboPartido,danificados,semPele) VALUES
    (@numeroToma,@produtoCongelado,@massa,@numeroUnidades,@sangue,@olhoVermelho,@raboInte
    iro,@meioRabo,@raboPartido,@danificados,@semPele);
GO

CREATE PROC Proj.UpdateToma @numeroToma TINYINT, @produtoCongelado INT, @massa
DECIMAL(10,1), @numeroUnidades DECIMAL(10,1), @sangue DECIMAL(4,1), @olhoVermelho
DECIMAL(4,1), @raboInteiro DECIMAL(4,1), @meioRabo DECIMAL(4,1), @raboPartido
DECIMAL(4,1), @danificados DECIMAL(4,1), @semPele DECIMAL(4,1)
AS
    UPDATE Proj.Toma SET massa = @massa, numeroUnidades = @numeroUnidades, sangue
    = @sangue, olhoVermelho = @olhoVermelho, raboInteiro = @raboInteiro, meioRabo =
    @meioRabo, raboPartido = @raboPartido, danificados = @danificados, semPele = @semPele
    WHERE numeroToma = @numeroToma AND produtoCongelado = @produtoCongelado;
GO

CREATE PROC Proj.InsertAnaliseDefeitosProdutos @nomeDocumento VARCHAR(255),
@efetuadoPor VARCHAR(255), @dataAnalise DATE, @loteAtribuido CHAR(9), @origemProduto
VARCHAR(255), @produtoFresco INT, @produtoCongelado INT
AS
    INSERT
    Proj.AnaliseDefeitosProdutos(nomeDocumento,efetuadoPor,dataAnalise,loteAtribuido,orig
    emProduto,produtoFresco,produtoCongelado) VALUES
    (@nomeDocumento,@efetuadoPor,@dataAnalise,@loteAtribuido,@origemProduto,@produtoFresc
    o,@produtoCongelado);
GO

CREATE PROC Proj.UpdateAnaliseDefeitosProdutos @nomeDocumento VARCHAR(255),
@efetuadoPor VARCHAR(255), @dataAnalise DATE, @loteAtribuido CHAR(9), @origemProduto
VARCHAR(255), @produtoFresco INT, @produtoCongelado INT
AS
    UPDATE Proj.AnaliseDefeitosProdutos SET origemProduto = @origemProduto,
    produtoFresco = @produtoFresco, produtoCongelado = @produtoCongelado WHERE
    nomeDocumento = @nomeDocumento AND efetuadoPor = @efetuadoPor AND dataAnalise =
    @dataAnalise AND loteAtribuido = @loteAtribuido;
GO

CREATE PROC Proj.InsertCameraFrescos @numeroAnaliseCamera SMALLINT,
@temperaturaRegistador DECIMAL(10,1), @temperaturaSonda DECIMAL(10,1),
@humidadeRelativa TINYINT
AS
    INSERT
    Proj.CameraFrescos(numeroAnaliseCamera,temperaturaRegistador,temperaturaSonda,humidad
    eRelativa) VALUES
    (@numeroAnaliseCamera,@temperaturaRegistador,@temperaturaSonda,@humidadeRelativa);
GO

```

```

CREATE PROC Proj.UpdateCameraFrescos @numeroAnaliseCamera SMALLINT,
@temperaturaRegistador DECIMAL(10,1), @temperaturaSonda DECIMAL(10,1),
@humidadeRelativa TINYINT
AS
    UPDATE Proj.CameraFrescos SET temperaturaRegistador = @temperaturaRegistador,
temperaturaSonda = @temperaturaSonda, humidadeRelativa = @humidadeRelativa WHERE
numeroAnaliseCamera = @numeroAnaliseCamera;
GO

CREATE PROC Proj.InsertTunelCongelacao @numeroAnaliseTunelCongelacao SMALLINT,
@temperaturaTunel DECIMAL(10,1), @velocidadeTapete1 TINYINT, @velocidadeTapete2
TINYINT, @numeroIdentificacaoTunelCongelacao TINYINT
AS
    INSERT
Proj.TunelCongelacao(numeroAnaliseTunelCongelacao, temperaturaTunel, velocidadeTapete1,
velocidadeTapete2, numeroIdentificacaoTunelCongelacao) VALUES
(@numeroAnaliseTunelCongelacao, @temperaturaTunel, @velocidadeTapete1, @velocidadeTapete
2, @numeroIdentificacaoTunelCongelacao);
GO

CREATE PROC Proj.UpdateTunelCongelacao @numeroAnaliseTunelCongelacao SMALLINT,
@temperaturaTunel DECIMAL(10,1), @velocidadeTapete1 TINYINT, @velocidadeTapete2
TINYINT, @numeroIdentificacaoTunelCongelacao TINYINT
AS
    UPDATE Proj.TunelCongelacao SET temperaturaTunel = @temperaturaTunel,
velocidadeTapete1 = @velocidadeTapete1, velocidadeTapete2 = @velocidadeTapete2,
numeroIdentificacaoTunelCongelacao = @numeroIdentificacaoTunelCongelacao WHERE
numeroAnaliseTunelCongelacao = @numeroAnaliseTunelCongelacao;
GO

CREATE PROC Proj.InsertTanqueSalmouracao @numeroAnaliseTanqueSalmouracao SMALLINT,
@temperaturaTanque1 DECIMAL(10,1), @temperaturaTanque2 DECIMAL(10,1),
@nivelAguaTanque TINYINT
AS
    INSERT
Proj.TanqueSalmouracao(numeroAnaliseTanqueSalmouracao, temperaturaTanque1, temperaturaT
anque2, nivelAguaTanque) VALUES
(@numeroAnaliseTanqueSalmouracao, @temperaturaTanque1, @temperaturaTanque2, @nivelAguaTa
nque);
GO

CREATE PROC Proj.UpdateTanqueSalmouracao @numeroAnaliseTanqueSalmouracao SMALLINT,
@temperaturaTanque1 DECIMAL(10,1), @temperaturaTanque2 DECIMAL(10,1),
@nivelAguaTanque TINYINT
AS
    UPDATE Proj.TanqueSalmouracao SET temperaturaTanque1 = @temperaturaTanque1,
temperaturaTanque2 = @temperaturaTanque2, nivelAguaTanque = @nivelAguaTanque WHERE
numeroAnaliseTanqueSalmouracao = @numeroAnaliseTanqueSalmouracao;
GO

CREATE PROC Proj.InsertCondicoesAnalisePropriedades @codigoContendor CHAR(11),
@denominacaoProduto VARCHAR(255), @calibre VARCHAR(255), @propriedadesOrganolepticas
DECIMAL(3,1), @temperaturaProduto DECIMAL(10,1), @codigoLote CHAR(9)
AS
    INSERT
Proj.CondicoesAnalisePropriedades(codigoContendor, denominacaoProduto, calibre, propried
adesOrganolepticas, temperaturaProduto, codigoLote) VALUES
(@codigoContendor, @denominacaoProduto, @calibre, @propriedadesOrganolepticas, @temperatu
raProduto, @codigoLote);
GO

```

```
CREATE PROC Proj.UpdateCondicoesAnalisePropriedades @codigoContendor CHAR(11),
@denominacaoProduto VARCHAR(255), @calibre VARCHAR(255), @propriedadesOrganolepticas
DECIMAL(3,1), @temperaturaProduto DECIMAL(10,1), @codigoLote CHAR(9)
AS
```

```
UPDATE Proj.CondicoesAnalisePropriedades SET denominacaoProduto =
@denominacaoProduto, calibre = @calibre, propriedadesOrganolepticas =
@propriedadesOrganolepticas, temperaturaProduto = @temperaturaProduto, codigoLote =
@codigoLote WHERE codigoContendor = @codigoContendor;
GO
```

```
CREATE PROC Proj.InsertCondicoesAnalise @nomeDocumento VARCHAR(255), @codigoContendor
CHAR(11), @dataAnalise DATE, @horaAnalise TIME, @controleEfetuadoPor VARCHAR(255),
@linha TINYINT, @turno TINYINT, @cameraFrescos SMALLINT, @tunelCongelacao SMALLINT,
@tanqueSalmouracao SMALLINT
AS
```

```
INSERT
Proj.CondicoesAnalise(nomeDocumento, codigoContendor, dataAnalise, horaAnalise, controleE
fetuadoPor, linha, turno, cameraFrescos, tunelCongelacao, tanqueSalmouracao) VALUES
(@nomeDocumento, @codigoContendor, @dataAnalise, @horaAnalise, @controleEfetuadoPor, @linh
a, @turno, @cameraFrescos, @tunelCongelacao, @tanqueSalmouracao);
GO
```

```
CREATE PROC Proj.UpdateCondicoesAnalise @nomeDocumento VARCHAR(255), @codigoContendor
CHAR(11), @dataAnalise DATE, @horaAnalise TIME, @controleEfetuadoPor VARCHAR(255),
@linha TINYINT, @turno TINYINT, @cameraFrescos SMALLINT, @tunelCongelacao SMALLINT,
@tanqueSalmouracao SMALLINT
AS
```

```
UPDATE Proj.CondicoesAnalise SET dataAnalise = @dataAnalise,
controleEfetuadoPor = @controleEfetuadoPor, linha = @linha, turno = @turno,
cameraFrescos = @cameraFrescos, tunelCongelacao = @tunelCongelacao, tanqueSalmouracao
= @tanqueSalmouracao, horaAnalise = @horaAnalise WHERE nomeDocumento = @nomeDocumento
AND codigoContendor = @codigoContendor;
GO
```

```
CREATE PROC Proj.InsertMaquinaVidrar @nomeDocumento VARCHAR(255), @codigoContendor
CHAR(11), @numeroAnaliseMaquinaVidrar SMALLINT, @temperaturaTanque DECIMAL(10,1),
@fluxoContinuo BIT, @seNCMudanca8n8 BIT, @se8n8RegistrarHorario TIME
AS
```

```
INSERT
Proj.MaquinaVidrar(nomeDocumento, codigoContendor, numeroAnaliseMaquinaVidrar, temperatu
raTanque, fluxoContinuo, seNCMudanca8n8, se8n8RegistrarHorario) VALUES
(@nomeDocumento, @codigoContendor, @numeroAnaliseMaquinaVidrar, @temperaturaTanque, @flux
oContinuo, @seNCMudanca8n8, @se8n8RegistrarHorario);
GO
```

```
CREATE PROC Proj.UpdateMaquinaVidrar @nomeDocumento VARCHAR(255), @codigoContendor
CHAR(11), @numeroAnaliseMaquinaVidrar SMALLINT, @temperaturaTanque DECIMAL(10,1),
@fluxoContinuo BIT, @seNCMudanca8n8 BIT, @se8n8RegistrarHorario TIME
AS
```

```
UPDATE Proj.MaquinaVidrar SET temperaturaTanque = @temperaturaTanque,
fluxoContinuo = @fluxoContinuo, seNCMudanca8n8 = @seNCMudanca8n8,
se8n8RegistrarHorario = @se8n8RegistrarHorario WHERE nomeDocumento = @nomeDocumento AND
codigoContendor = @codigoContendor AND numeroAnaliseMaquinaVidrar =
@numeroAnaliseMaquinaVidrar;
GO
```

```
CREATE PROC Proj.InsertClorosEOzono @nomeDocumento VARCHAR(255), @dataAnalise DATE,
@horaAnalise TIME, @produtoEmProcessamento VARCHAR(255), @efetuadoPor VARCHAR(255),
@fabrica TINYINT, @controleProcessoProdutivo SMALLINT
AS
```

```
INSERT
Proj.ClorosEOzono(nomeDocumento, dataAnalise, horaAnalise, produtoEmProcessamento, efetua
doPor, fabrica, controleProcessoProdutivo) VALUES
(@nomeDocumento, @dataAnalise, @horaAnalise, @produtoEmProcessamento, @efetuadoPor, @fabri
ca, @controleProcessoProdutivo);
GO
```

```

CREATE PROC Proj.UpdateClorosEOzono @nomeDocumento VARCHAR(255), @dataAnalise DATE,
@horaAnalise TIME, @produtoEmProcessamento VARCHAR(255), @efetuadoPor VARCHAR(255),
@fabrica TINYINT, @controleProcessoProdutivo SMALLINT
AS
    UPDATE Proj.ClorosEOzono SET fabrica = @fabrica, controleProcessoProdutivo =
@controleProcessoProdutivo WHERE nomeDocumento = @nomeDocumento AND dataAnalise =
@dataAnalise AND horaAnalise = @horaAnalise AND produtoEmProcessamento =
@produtoEmProcessamento AND efetuadoPor = @efetuadoPor;
GO

CREATE PROC Proj.InsertDocumento @nomeDocumento VARCHAR(255)
AS
    INSERT Proj.Documentos(nomeDocumento) VALUES (@nomeDocumento)
GO

CREATE PROC Proj.UpdateDocumento @nomeDocumento1 VARCHAR(255), @nomeDocumento2
VARCHAR(255)
AS
    UPDATE Proj.Documentos SET nomeDocumento = @nomeDocumento2 WHERE nomeDocumento =
@nomeDocumento1;
GO

CREATE PROCEDURE Proj.percentage_work_factory
AS
    DECLARE @total INT;

    SELECT @total=count(fabrica) FROM Proj.ClorosEOzono

    SELECT fabrica AS [Fábrica], cast(cast((count(fabrica)) AS decimal(4,2))/@total AS
decimal(4,2)) AS [Porcentagem de Trabalho] FROM Proj.ClorosEOzono GROUP BY fabrica
GO

```

## User-Defined Functions

User-Defined Functions foram utilizadas no projeto em operações de visualização de dados, incluindo em casos não parametrizados por questões de facilidade.

```

CREATE FUNCTION Proj.freshness_classification_avg(@max varchar(5))RETURNS TABLE
AS
    RETURN SELECT origemProduto AS [Origem do Produto], CF.designacao AS
[Classificação de Frescura]
FROM(
    SELECT origemProduto, cast (avg(classificacaoFrescura) AS
tinyint) AS classfresq
FROM(
    Proj.AnaliseOrganoleptica
JOIN Proj.ProdutoFresco
ON numeroAnalise=numeroAnaliseOrganoleptica
JOIN Proj.AnaliseDefeitosProdutos
ON numeroProdutoFresco = produtoFresco
)
GROUP BY origemProduto
HAVING cast (avg(classificacaoFrescura) AS tinyint) <
(SELECT codigo FROM Proj.ClassificacaoFrescura WHERE designacao=@max)
) AS S
join Proj.ClassificacaoFrescura AS CF ON S.classfresq =
CF.codigo
GO

```

```

CREATE FUNCTION Proj.production_block_list (@minVal decimal(4,1),@maxVal decimal(4,1))RETURNS TABLE
AS
    return SELECT PDTPNF.nomeProduto AS [Nome do Produto], TP.designacao AS [Tipo de Produto],
PDTPNF.designacao AS [Familia do Produto]
FROM(
    SELECT PDTPN.nomeProduto, PDTPN.tipoProduto, FP.designacao
FROM
(
    SELECT PDTP.nomeProduto, PDTP.tipoProduto,
FROM
(
    SELECT nomeProduto
FROM
(
    SELECT nomeDocumento
FROM
(
    SELECT
FROM
WHERE
cloroArrefecedor1 < @minVal or cloroArrefecedor1 > @maxVal or cloroArrefecedor2 < @minVal or cloroArrefecedor2
> @maxVal
)
UNION
(
    SELECT
FROM
WHERE
cloroMangueira1 < @minVal or cloroMangueira1 > @maxVal or cloroMangueira2 < @minVal or cloroMangueira2 >
@maxVal or cloroMangueira4 < @minVal or cloroMangueira4 > @maxVal or cloroMangueira6 < @minVal or
cloroMangueira6 > @maxVal or cloroMangueira7 < @minVal or cloroMangueira7 > @maxVal or cloroMangueira9 <
@minVal or cloroMangueira9 > @maxVal or cloroMangueiraExterior < @minVal or cloroMangueiraExterior > @maxVal
)
UNION
(
    SELECT
FROM
WHERE
cloroLavadora1 < @minVal or cloroLavadora1 > @maxVal or cloroLavadora2 < @minVal or cloroLavadora2 > @maxVal
or cloroLavadoraEsgoto1 < @minVal or cloroLavadoraEsgoto1 > @maxVal or cloroLavadoraEsgoto2 < @minVal or
cloroLavadoraEsgoto2 > @maxVal
)
UNION
(
    SELECT
FROM
WHERE
cloroChiller < @minVal or cloroChiller > @maxVal or cloroControladorEntrada < @minVal or
cloroControladorEntrada > @maxVal
)
) AS CPP
JOIN Proj.ClorosEOzono AS CEO
ON CPP.numeroControlo =

CEO.controloProcessoProdutivo

) AS CEO
JOIN Proj.PlanoDeTrabalho AS PDT
ON CEO.nomeDocumento = PDT.documento
) AS PDT
JOIN Proj.PlanoDeTrabalhoProduto AS PDTP
ON PDT.nomeProduto = PDTP.nomeProduto
) AS PDTPN
JOIN Proj.FamiliaProduto AS FP
ON PDTPN.familiaProduto=FP.codigo
)AS PDTPNF
JOIN Proj.TipoProduto AS TP
ON PDTPNF.tipoProduto=TP.codigo
GO

```



```

CREATE FUNCTION Proj.store_product_problem(@frescTemp decimal(4,1),@congTemp
decimal(4,1))RETURNS TABLE
AS
    RETURN SELECT CAP.codigoContentor AS [Código do Contentor],
CAP.denominacaoProduto AS [Denominação do Produto], CAP.calibre AS [Calibre],
CAP.propriedadesOrganolepticas AS [Propriedades Organolepticas],
CAP.temperaturaProduto AS [Temperatura do Produto], CAP.codigoLote AS [Código do
Lote]
        FROM(
            (
                SELECT codigoContentor
                FROM
                (
                    SELECT numeroAnaliseCamera
                    FROM Proj.CameraFrescos
                    WHERE temperaturaRegistador >
@frescTemp or temperaturaSonda > @frescTemp
                ) AS AC
                JOIN Proj.CondicoesAnalise
                ON numeroAnaliseCamera = cameraFrescos
            )
            UNION
            (
                SELECT codigoContentor
                FROM
                (
                    SELECT numeroAnaliseTunelCongelacao
                    FROM Proj.TunelCongelacao
                    WHERE temperaturaTunel > @congTemp
                ) AS ATC
                JOIN Proj.CondicoesAnalise
                ON numeroAnaliseTunelCongelacao =
tunelCongelacao
            )
        )AS CA
        JOIN Proj.CondicoesAnalisePropriedades AS CAP
        ON CA.codigoContentor = CAP.codigoContentor

GO

CREATE FUNCTION Proj.vidrag (@prodconj int) RETURNS TABLE
AS
    RETURN SELECT numeroVidragem AS [Número de Vidragem], [min] AS [Mínimo], [max]
AS [Máximo], [med] AS [Média], [amp] AS [Amplitude]
        FROM Proj.ProdutoCongelado join Proj.Vidragem ON nVidragem =
numeroVidragem
        WHERE numeroProdutoCongelado = @prodconj

GO

CREATE FUNCTION Proj.calib (@prodfresc int) RETURNS TABLE
AS
    RETURN SELECT numeroVerificacao AS [Número da Verificação], massa AS [Massa],
unidades AS [Unidades], classificacao AS [Classificação]
        FROM Proj.ProdutoFresco JOIN Proj.VerificacaoCalibre ON
nVerificacaoCalibre=numeroVerificacao
        WHERE numeroProdutoFresco=@prodfresc

GO

```

```

CREATE FUNCTION Proj.freshness_classification() RETURNS TABLE
AS

    RETURN SELECT designacao AS [Classificação de Frescura], count(numeroAnalise)
AS [Total de Análises]
    FROM Proj.AnaliseOrganoleptica
    JOIN Proj.ClassificacaoFrescura ON classificacaoFrescura =
codigo
    GROUP BY designacao

GO

CREATE FUNCTION Proj.Select_AnaliseOrganoleptica() RETURNS TABLE
AS

    RETURN SELECT AO.numeroAnalise AS [Número da Análise Organoléptica],
AO.cotacaoTotal AS [Cotação Total], CF.designacao AS [Classificação de Frescura]
    FROM Proj.AnaliseOrganoleptica AS AO
    JOIN Proj.ClassificacaoFrescura AS CF
    ON AO.classificacaoFrescura = CF.codigo

GO

CREATE FUNCTION Proj.Select_Vidragem() RETURNS TABLE
AS

    RETURN SELECT VD.numeroVidragem AS [Número da Análise à Vidragem], VD.[min] AS
[Mínimo], VD.[max] AS [Máximo], VD.med AS [Média], VD.amp AS [Amplitude]
    FROM Proj.Vidragem AS VD

GO

CREATE FUNCTION Proj.Select_CameraFrescos() RETURNS TABLE
AS

    RETURN SELECT CF.numeroAnaliseCamera AS [Número da Análise à Câmara de
Frescos], CF.temperaturaRegistador AS [Temperatura do Registador],
CF.temperaturaSonda AS [Temperatura da Sonda], CF.humidadeRelativa AS [Humidade
Relativa]
    FROM Proj.CameraFrescos AS CF

GO

CREATE FUNCTION Proj.Select_CondicoesAnalisePropriedades() RETURNS TABLE
AS

    RETURN SELECT CAP.codigoContentor AS [Código do Contentor], CAP.codigoLote AS
[Código Lote], CAP.denominacaoProduto AS [Denominação do Produto], CAP.calibre AS
[Calibre], CAP.propriedadesOrganolepticas AS [Propriedades Organolépticas],
CAP.temperaturaProduto AS [Temperatura do Produto]
    FROM Proj.CondicoesAnalisePropriedades AS CAP

GO

```

```

CREATE FUNCTION Proj.Select_PlanoDeTrabalho() RETURNS TABLE
AS

    RETURN SELECT PDT.nomeProduto AS [Nome do Produto], AAR.tipoAnalise AS [Análise a
Realizar], PDT.RDQsUtilizadas AS [RDQs Utilizadas], F.designacao AS Frequencia,
OAR.designacao AS [Operação a Realizar], PDT.documento AS [Documento a Editar]
    FROM Proj.PlanoDeTrabalho AS PDT
    JOIN Proj.Frequencia AS F
    ON PDT.frequencia = F.codigo
    JOIN Proj.OperacaoARealizar AS OAR
    ON PDT.operacao = OAR.codigo
    JOIN Proj.PlanoDeTrabalhoAcao AS PDTA
    ON PDT.analise = PDTA.analise
    JOIN Proj.AnaliseARealizar AS AAR
    ON PDTA.analise = AAR.codigo

```

GO

```

CREATE FUNCTION Proj.Select_PDT_Produto() RETURNS TABLE
AS

    RETURN SELECT PDTP.nomeProduto AS [Nome do Produto], FP.designacao AS [Família do
Produto], TP.designacao AS [Tipo de Produto]
    FROM Proj.PlanoDeTrabalhoProduto AS PDTP
    JOIN Proj.FamiliaProduto AS FP
    ON PDTP.familiaProduto = FP.codigo
    JOIN Proj.TipoProduto AS TP
    ON PDTP.tipoProduto = TP.codigo

```

GO

```

CREATE FUNCTION Proj.Select_PDT_Acao() RETURNS TABLE
AS

    RETURN SELECT AAR.tipoAnalise AS [Tipo de Análise], PDTA.instrumentos AS
[Instrumentos], SLE.designacao AS [Efetuado Por]
    FROM Proj.PlanoDeTrabalhoAcao AS PDTA
    JOIN Proj.AnaliseARealizar AS AAR
    ON PDTA.analise = AAR.codigo
    JOIN Proj.SLouEncarregada AS SLE
    ON PDTA.efetuadoPor = SLE.codigo

```

GO

```

CREATE FUNCTION Proj.Select_ADP() RETURNS TABLE
AS

    RETURN SELECT ADP.nomeDocumento AS [Nome do Documento], ADP.efetuadoPor AS
[Efetuated por], ADP.dataAnalise AS [Data de Análise], ADP.loteAtribuido AS [Lote
Atribuído], ADP.origemProduto AS [Origem do Produto], ADP.produtoFresco AS [Número do
Produto Fresco], PF.nAnaliseOrganoleptica AS [Número da Análise Organoléptica],
PF.nVerificacaoCalibre AS [Número da Verificação de Calibre], ADP.produtoCongelado AS
[Número do Produto Congelado], PC.nVidragem AS [Número da Análise à Vidragem], SUBQ.count
AS [Número de Tomas]
    FROM Proj.AnaliseDefeitosProdutos AS ADP
    JOIN Proj.ProdutoCongelado AS PC
    ON ADP.produtoCongelado = PC.numeroProdutoCongelado
    JOIN Proj.ProdutoFresco AS PF
    ON ADP.produtoFresco = PF.numeroProdutoFresco
    LEFT JOIN (
        SELECT produtoCongelado, COUNT(numeroToma) AS count
        FROM Proj.Toma
        GROUP BY produtoCongelado) AS SUBQ
    ON PC.numeroProdutoCongelado = SUBQ.produtoCongelado

```

GO

```
CREATE FUNCTION Proj.Select_CPP()RETURNS TABLE
AS
```

```
    RETURN SELECT CPP.numeroControlo AS [Número do Controlo do Processo
Produtivo], CPP.cloroChiller AS [Cloro Chiller], CPP.oxigenio AS [Oxigénio],
CPP.odorOzono AS [Odor a Ozono?], CPP.cloroControladorEntrada AS [Cloro Controlador
de Entrada], CPP.controladorLavadoras AS [Controlador de Lavadoras], CPP.geracaoOzono
AS [Geração de Ozono], CPP.leituraCloroRegistador AS [Leitura do Cloro do
Registador], CA.cloroArrefecedor1 AS [Cloro Arrefecedor #1], CA.cloroArrefecedor2 AS
[Cloro Arrefecedor #2], CL.cloroLavadora1 AS [Cloro Lavadora #1], CL.cloroLavadora2
AS [Cloro Lavadora #2], CL.cloroLavadoraEsgoto1 AS [Cloro Lavadora Esgoto #1],
CL.cloroLavadoraEsgoto2 AS [Cloro Lavadora Esgoto #2], CM.cloroMangueira1 AS [Cloro
Mangueira #1], CM.cloroMangueira2 AS [Cloro Mangueira #2], CM.cloroMangueira4 AS
[Cloro Mangueira #4], CM.cloroMangueira6 AS [Cloro Mangueira #6], CM.cloroMangueira7
AS [Cloro Mangueira #7], CM.cloroMangueira9 AS [Cloro Mangueira #9],
CM.cloroMangueiraExterior AS [Cloro Mangueira Exterior]
    FROM Proj.ControloProcessoProdutivo AS CPP
    JOIN Proj.ClorosArrefecedor AS CA
    ON CPP.numeroControlo = CA.numeroControlo
    JOIN Proj.ClorosLavadora AS CL
    ON CPP.numeroControlo = CL.numeroControlo
    JOIN Proj.ClorosMangueira AS CM
    ON CPP.numeroControlo = CM.numeroControlo
```

```
GO
```

```
CREATE FUNCTION Proj.Select_CEO()RETURNS TABLE
AS
```

```
    RETURN SELECT CEO.nomeDocumento AS [Nome do Documento], CEO.dataAnalise AS
[Data da Análise], CEO.horaAnalise AS [Hora da Análise], CEO.produtoEmProcessamento
AS [Produto em processamento], CEO.efetuadoPor AS [Efetuado Por], F.designacao AS
[Fábrica], CEO.controloProcessoProdutivo AS [Número do Controlo do Processo
Produtivo]
    FROM Proj.ClorosEOzono AS CEO
    JOIN Proj.Fabrica AS F
    ON CEO.fabrica = F.codigo
```

```
GO
```

```
CREATE FUNCTION Proj.Select_Documentos()RETURNS TABLE
AS
```

```
    RETURN SELECT nomeDocumento AS [Nome do Documento]
    FROM Proj.Documentos
```

```
GO CREATE FUNCTION Proj.Select_MaquinaVidrar()RETURNS TABLE
AS
```

```
    RETURN SELECT MV.nomeDocumento AS [Nome do Documento], MV.codigoContentor AS
[codigoContentor], MV.numeroAnaliseMaquinaVidrar AS [Número de Análise à Máquina de
Vidrar], MV.temperaturaTanque AS [Temperatura do Tanque], MV.fluxoContínuo AS [Fluxo
Contínuo?], MV.seNCMudanca8n8 AS [Se não fluxo Contínuo, mudança a cada 8 horas?],
MV.se8n8RegistarHorario AS [Se mudança a cada 8 horas, registar Horário]
    FROM Proj.MaquinaVidrar AS MV
```

```
GO
```

```
CREATE FUNCTION Proj.Select_CDA() RETURNS TABLE
AS
```

```
    RETURN SELECT CDA.nomeDocumento AS [Nome do Documento], CDA.codigoContentor AS
[Código do Contentor], CDA.dataAnalise AS [Data da Análise], CDA.horaAnalise AS [Hora
da Análise], CDA.controloEfetuadoPor AS [Controlo efetuado por], NL.designacao AS
[Linha de Produção], NT.designacao AS [Turno de Trabalho], CDA.cameraFrescos AS
[Número da Análise a Câmara de Frescos], CDA.tunelCongelacao AS [Número da Análise a
Túnel de Congelação], CDA.tanqueSalmouracao AS [Número da Análise a Tanque de
Salmouração], SUBQ.count AS [Número de Máquinas de Vidrar]
        FROM Proj.CondicoesAnalise AS CDA
        JOIN Proj.NumeroLinha AS NL
        ON CDA.linha = NL.codigo
        JOIN Proj.NumeroTurno AS NT
        ON CDA.turno = NT.codigo
        JOIN Proj.CameraFrescos AS CF
        ON CDA.cameraFrescos = CF.numeroAnaliseCamera
        JOIN Proj.TunelCongelacao AS TC
        ON CDA.tunelCongelacao = TC.numeroAnaliseTunelCongelacao
        JOIN Proj.TanqueSalmouracao AS TS
        ON CDA.tanqueSalmouracao = TS.numeroAnaliseTanqueSalmouracao
        LEFT JOIN ( SELECT
nomeDocumento, codigoContentor, COUNT(numeroAnaliseMaquinaVidrar) AS count
                        FROM Proj.MaquinaVidrar
                        GROUP BY nomeDocumento, codigoContentor
                    ) AS SUBQ
        ON CDA.nomeDocumento = SUBQ.nomeDocumento
AND CDA.codigoContentor = SUBQ.codigoContentor
```

```
GO
```

```
CREATE FUNCTION Proj.Select_ProdutoCongelado() RETURNS TABLE
AS
```

```
    RETURN SELECT PC.numeroProdutoCongelado AS [Número do Produto Congelado],
VD.numeroVidragem AS [Número da Vidragem], VD.[min] AS [Mínimo], VD.[max] AS
[Máximo], VD.med AS [Média], VD.amp AS [Amplitude], SUBQ.numTom AS [Número de Tomas]
        FROM Proj.ProdutoCongelado AS PC
        JOIN Proj.Vidragem AS VD
        ON PC.nVidragem = VD.numeroVidragem
        JOIN ( SELECT PC.numeroProdutoCongelado, COUNT(T.numeroToma) AS numTom
                FROM Proj.ProdutoCongelado AS PC
                JOIN Proj.Toma AS T
                ON PC.numeroProdutoCongelado = T.produtoCongelado
                GROUP BY PC.numeroProdutoCongelado
            ) AS SUBQ
        ON PC.numeroProdutoCongelado = SUBQ.numeroProdutoCongelado
```

```
GO
```

```
CREATE FUNCTION Proj.Select_Toma() RETURNS TABLE
AS
```

```
    RETURN SELECT T.numeroToma AS [Número da Toma], T.produtoCongelado AS [Número
do Produto Congelado], T.massa AS [Massa], T.numeroUnidades AS [Número de Unidades],
T.sangue AS [Sangue], T.olhoVermelho AS [Olho Vermelho], T.raboInteiro AS [Rabo
Inteiro], T.meioRabo AS [Meio Rabo], T.raboPartido AS [Rabo Partido], T.danificados
AS [Danificados], T.semPele AS [Sem Pele]
        FROM Proj.Toma AS T
```

```
GO
```

```

CREATE FUNCTION Proj.Select_ProdutoFresco() RETURNS TABLE
AS

    RETURN SELECT PF.numeroProdutoFresco AS [Número do Produto Fresco],
PF.nAnaliseOrganoleptica AS [Número da Análise Organoléptica], AO.cotacaoTotal AS
[Cotação Total], CF.designacao AS [Classificação de Frescura], VC.numeroVerificacao
AS [Número da Verificação do Calibre], VC.massa AS [Massa], VC.unidades AS
[Unidades], VC.classificacao AS [Classificação do Calibre]
    FROM Proj.ProdutoFresco AS PF
    JOIN Proj.AnaliseOrganoleptica AS AO
    ON PF.nAnaliseOrganoleptica = AO.numeroAnalise
    JOIN Proj.ClassificacaoFrescura AS CF
    ON AO.classificacaoFrescura = CF.codigo
    JOIN Proj.VerificacaoCalibre AS VC
    ON PF.nVerificacaoCalibre = VC.numeroVerificacao

GO

CREATE FUNCTION Proj.Select_TanqueSalmouracao() RETURNS TABLE
AS

    RETURN SELECT TS.numeroAnaliseTanqueSalmouracao AS [Número da Análise ao
Tanque de Salmouração], TS.temperaturaTanque1 AS [Temperatura Tanque #1],
TS.temperaturaTanque2 AS [Temperatura Tanque #2], TS.nivelAguaTanque AS [Nível da
Água do Tanque]
    FROM Proj.TanqueSalmouracao AS TS

GO

CREATE FUNCTION Proj.Select_TunelCongelacao() RETURNS TABLE
AS

    RETURN SELECT TC.numeroAnaliseTunelCongelacao AS [Número da Análise ao Túnel
de Congelação], TC.temperaturaTunel AS [Temperatura do Túnel], TC.velocidadeTapete1
AS [Velocidade do Tapete #1], TC.velocidadeTapete2 AS [Velocidade do Tapete #2],
TC.numeroIdentificacaoTunelCongelacao AS [Número de Identificação do Túnel de
Congelação]
    FROM Proj.TunelCongelacao AS TC

GO

CREATE FUNCTION Proj.Select_VerificacaoCalibre() RETURNS TABLE
AS

    RETURN SELECT VC.numeroVerificacao AS [Número da Verificação de Calibre],
VC.massa AS [Massa], VC.unidades AS [Unidades], VC.classificacao AS [Classificação de
Calibre]
    FROM Proj.VerificacaoCalibre AS VC

GO

```

## Utilização da Aplicação

A aplicação foi desenvolvida de forma a minimamente replicar o sistema de autenticação existente na empresa. De tal forma, credenciais são necessárias para utilizar a aplicação. Utilizadores com permissões completas são Encarregados(as). Como exemplo, use a autenticação [ Nmec: 84831, Password: Sergio ] para testar.

Utilizadores com permissões parciais (podem inserir dados mas não editar) são do Serviço de Laboratório. Use as credenciais [ Nmec: 84734, Password: Fabio ] para testar.

Adicionalmente, para alterar a SQL Connection String, abra o ficheiro C# relativo ao form LogInForm (LogInForm.cs) e altere a linha 28.