

INSTITUTO POLITÉCNICO DE VIANA DO CASTELO

ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

ENGENHARIA INFORMÁTICA

ADMINISTRAÇÃO DE BASES DE DADOS

2020/2021

Trabalho Prático

Aluno:
Nelson Novais - 24844

Docente:
Marco Lima
Domingos Gomes



**Instituto Politécnico
de Viana do Castelo**

6 de junho de 2022

Conteúdo

Lista de Figuras	v
1 Objetivos do Trabalho Prático e Revisão Teórica	1
1.1 Objetivos do Trabalho	1
1.2 Revisão Teórica	1
1.2.1 Stored Procedures	1
1.2.2 Triggers	2
1.2.3 Funções	2
1.2.4 Cursores	2
1.2.5 FileStream	2
1.2.6 Back up	2
1.2.7 Database Engine Tuning Advisor	2
1.2.8 Plano de Manutenção	2
2 Modelo Relacional e Criação da Base de dados	3
2.1 Diagrama de Entidades e Relacionamentos	3
2.2 Script para a criação da base de dados	4
2.3 Scripts para a criação das tabelas da base de dados	4
2.4 Diagrama gerado pelo SSMS	6
3 Function e View	7
3.1 Function	7
3.2 Views	7
4 Stored Procedure	9
4.1 SPs de inserção, atualização e remoção	9
4.2 SP de Verificação	10
5 Cursor	11
5.1 Cursor	11
6 Triggers	13
6.1 Triggers	13
7 Selects	15
7.1 Select Com Pivot	15
7.2 Select Com Rank,DenseRank e RowNumber	15
7.3 Outros Selects	16
8 Filestream	19
8.1 Criação da Base de Dados	19
8.2 Inserção de dados na Base de Dados	19
9 Otimização de instrução SQL	21
9.1 otimização de uma instrução SQL	21

10 Construção do Plano de Manutenção	23
10.1 Criação do Operador	23
10.2 Criação e Execução do Plano	23
11 Elaboração de um relatório com o Report Builder	29
11.1 Construção do relatório	29
12 CONCLUSÕES	33
Bibliografia	35

Lista de Figuras

2.1	Diagrama criado para projetar a base de dados	3
2.2	Script para a criação da base de dados filestream	4
2.3	Script para a criação das Tabelas CodPostal, Cliente e TipoPagamento	4
2.4	Script para a criação das Tabelas Servico e Fatura	5
2.5	Script para a criação da Tabela LinhaFatura	5
2.6	Diagrama gerado pelo SQL Server após a criação das tabelas da base de dados	6
3.1	Função CalcularValorIva e resultado	7
3.2	View VerFatura	7
3.3	Resultado da View VerFatura	7
3.4	View VerFaturaPorServico e o seu resultado	8
3.5	Criação do utilizador "WebFatura" e permissões para ler a View	8
4.1	SP de Inserção	9
4.2	SP de Atualização	9
4.3	SP de Eliminação	10
4.4	SP que verifica a existência de um determinado serviço	10
5.1	Cursor "Venda1000Prod1503"	11
6.1	Trigger "InsercaoLinhaFaturaAtualiz"	13
6.2	Trigger "updateLinhaFatura"	13
6.3	Trigger "RemocaoDeLinha"	14
7.1	Select com pivot	15
7.2	Select com Dense rank	15
7.3	Select com Rank	16
7.4	Select com RowNumber	16
7.5	Select que retorna o serviço que mais vezes apareceu nas faturas.	16
7.6	Select que retorna o Codigo Postal que mais vezes apareceu nas faturas.	16
7.7	Select que retorna o tipoPagamento mais utilizado nas faturas.	17
8.1	Criação da Base de Dados Com suporte para Filestream.	19
8.2	Insert no Filestream.	19
8.3	Insert no Filestream.	19
8.4	Como podemos observar foi criado um ficheiro que contém lá dentro a imagem inserida.	20
9.1	Com esta imagem podemos observar o custo do Select antes de utilizar o Database Engine Tuning Advisor	21
9.2	Seleção das tabelas no Database Engine Tuning Advisor	21
9.3	Recomendações geradas pelo Database Engine Tuning Advisor	22
9.4	Neste print é possível verificar o custo atual da instrução e o custo recomendado	22

9.5	Custo do select depois de aplicar as recomendações	22
10.1	Criação do operador	23
10.2	Criação dos horários de execução do plano	24
10.3	Seleção das tarefas a fazer no plano de manutenção	24
10.4	Seleção da base de dados para as verificações de integridade no plano de manutenção	25
10.5	Seleção da base de dados para reorganizar os índices no plano de ma- nutenção	25
10.6	Seleção da base de dados para fazer o back up no plano de manutenção	26
10.7	O plano de manutenção foi criado com sucesso	26
10.8	Como é possível verificar pela imagem o plano foi executado com su- cesso às 18h	27
11.1	Para criar o grafico foi criado um novo conjunto de dados	29
11.2	Ligação ao servidor e à base de dados	30
11.3	Criação do select para gerar o gráfico	30
11.4	Disposição dos campos pelas valores e colunas	31
11.5	Estrutura do esquema que gerado	31
11.6	Disposição dos campos no gráfico	32
11.7	Esquema do relatório	32
11.8	Relatório final	32

Capítulo 1

Objetivos do Trabalho Prático e Revisão Teórica

1.1 Objetivos do Trabalho

Este trabalho tem como objetivo fazer um sistema de gestão de faturação de serviços.

Requisitos:

- Inserir 10 mil registos
- Criar uma Function
- Criar uma view e dar permissões de leitura para o utilizador WebFatura
- Criar SPs para inserir, atualizar e remoção
- Criar um Cursor
- Criar triggers para o normal funcionamento da base de dados
- Criar um SP para validar informação
- Criar Selects que retornem informações pertinentes
- Utilizar o FileStream
- Utilizar o Database Engine Tuning Advisor para uma instrução SQL
- Criar um plano de manutenção
- Criar um relatório com gráfico e registos em tabela

1.2 Revisão Teórica

1.2.1 Stored Procedures

Stored Procedure é uma conjunto de comandos em SQL que podem ser executados de uma vez. Os Sps têm vantagens como serem executados em processos do servidor, aproveitam mecanismos de optimização de desempenho disponibilizados pelo SGBD, podem devolver ou alterar valores e receber parâmetros de entrada, facilitam a identificação clara entre as tarefas que devem ser executadas pelo lado do cliente e do servidor, as aplicações deixam de necessitar de um conhecimento completo e minucioso da estrutura da base de dados

1.2.2 Triggers

Um Trigger é código que é ativado, despoletado e automaticamente executado quando ocorre um determinado evento, pode ser ativado sempre que há insert, update, delete. Os triggers permitem implementar as regras ao nível da base de dados que ficam disponíveis para todas as aplicações que manipulem a base de dados

1.2.3 Funções

As funções retornam valores ou tabelas, e podem ser criadas para depois serem utilizadas em outras instruções sql.

1.2.4 Cursores

Os cursores permitem executar operações como: atualizar, excluir ou mover dados. O conjunto de linhas para o qual um cursor aponta é definido pelo comando SELECT, os cursores também permitem aceder individualmente a cada linha de um conjunto de dados

1.2.5 FileStream

O FILESTREAM permite armazenar dados não estruturados, como documentos e imagens, O FILESTREAM integra o Mecanismo de Banco de Dados do SQL Server a um sistema de arquivos NTFS ou ReFS, armazenando os dados varbinary (max) de objeto binário (BLOB) como ficheiros.

1.2.6 Back up

É muito importante fazer backups para assim não perder dados se houver alguns problemas com a base de dados. Neste trabalho fizemos 3 tipos de diferentes backup o full database backup, que como o nome indica é uma cópia completa, o diferencial backup captura o estado das extensões alteradas no momento em que o backup foi criado. Como um diferencial backup não faz backup de tudo, o backup geralmente funciona mais rápido do que um full backup. É recomendado que faça um full database backup em intervalos de tempo definido e diferencial database backup no meio, já o transaction log backup regista todas as alterações à base de dados. Foi também restaurada a base de dados

1.2.7 Database Engine Tuning Advisor

Com Database Engine Tuning Advisor é possível ver recomendações para melhorar o desempenho e implementá-las. O Tuning Advisor é capaz de fazer recomendações para otimizar o desempenho das mesmas, através de modificações nas estruturas da base de dados, tais como a criação de índices, views indexadas e particionamentos

1.2.8 Plano de Manutenção

Um plano de manutenção é uma sequência de tarefas que ocorrem segundo um agendamento predefinido. Os planos de manutenção, permitem executar automaticamente tarefas de manutenção sobre uma ou mais bases de dados.

Capítulo 2

Modelo Relacional e Criação da Base de dados

Nota: Apresentar evidências da realização do tutorial

2.1 Diagrama de Entidades e Relacionamentos

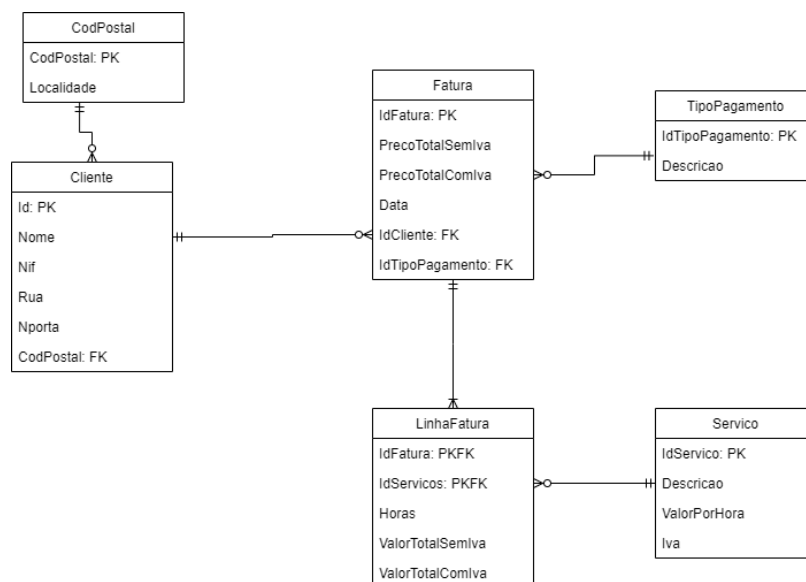


FIGURA 2.1: Diagrama criado para projetar a base de dados

2.2 Script para a criação da base de dados

```

CREATE DATABASE FileStreamTP
ON
PRIMARY ( NAME = FileStreamTP,
          FILENAME = 'C:\FileTRab\FileStreamTP.mdf'),
          FILEGROUP FileStreamTPFS CONTAINS FILESTREAM(
          NAME = FileStreamTPFS,
          FILENAME = 'C:\FileTRab\FileStreamTPFS')
LOG ON (
          NAME = FileStreamTPLOG,
          FILENAME = 'C:\FileTRab\FileStreamTPLOG.ldf')
GO

```

FIGURA 2.2: Script para a criação da base de dados filestream

2.3 Scripts para a criação das tabelas da base de dados

```

create table CodPostal
(
    CodPostal varchar(20) not null
        constraint CodPostal_pk
            primary key,
    Localidade varchar(100),
)
go
create table Cliente
(
    IdCliente int not null identity
        constraint Cliente_pk
            primary key,
    Nome varchar(100) not null,
    Nif int unique,
    Rua varchar(100),
    NPorta varchar(10),
    CodPostal varchar(20)
        constraint Cliente_CodPostal_fk
            references CodPostal,
    [FSID] UNIQUEIDENTIFIER ROWGUIDCOL NOT NULL UNIQUE,
    [FSDescription] VARCHAR(50),
    [FSBLOB] VARBINARY(MAX) FILESTREAM NULL
)
go
create table TipoPagamento
(
    descricao varchar(100),
    IdTipoPagamento int not null identity
        constraint TipoPagamento_pk
            primary key
)
go

```

FIGURA 2.3: Script para a criação das Tabelas CodPostal, Cliente e TipoPagamento

```
create table Servico
(
    IdServico int not null identity
        constraint Servico_pk
        primary key,

    Iva float not null,
    ValorPorHora float not null,
    Descricao varchar(100),
)
go

create table Fatura
(
    IdFatura int not null identity
        constraint Fatura_pk
        primary key,
    Id_cliente int not null
        constraint Fatura_IdCliente_fk
        references Cliente,
    IdTipoPagamento int not null
        constraint Fatura_Cliente_IdCliente_fk
        references TipoPagamento,
    data_venda datetime not null,
    precoTotalSemIva float not null,
    precoTotalComIva float not null,
    quantidade float,
)
go
```

FIGURA 2.4: Script para a criação das Tabelas Servico e Fatura

```
create table LinhaFatura
(
    IdFatura int not null
        constraint LinhaFatura_Fatura_Id_Fatura_fk
        references Fatura,
    IdServico int not null
        constraint LinhaFatura_Fatura_Id_Servico_fk
        references Servico,
    horas float,
    precoTotalSemIva float,
    precoTotalComIva float,
    constraint LinhaFatura_pk
        primary key (IdFatura, IdServico)
)
go
```

FIGURA 2.5: Script para a criação da Tabela LinhaFatura

2.4 Diagrama gerado pelo SSMS

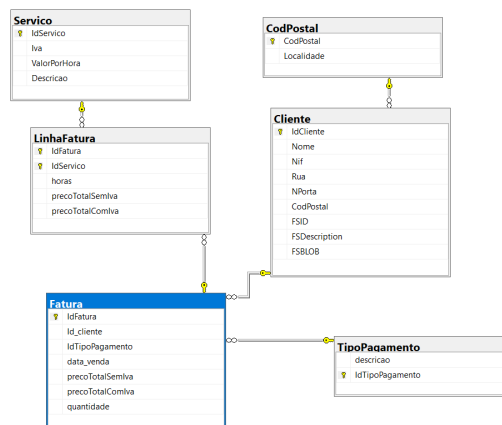


FIGURA 2.6: Diagrama gerado pelo SQL Server após a criação das tabelas da base de dados

Capítulo 3

Function e View

Nota: Apresentar evidências da realização do ponto 3 e 4

3.1 Function

Foi criada uma função que permite calcular o valor do serviço com o IVA para um determinado serviço que é recebido como parâmetro e o número de horas que deseja do serviço, a função retorna o valor que teria de pagar pelo serviço.

```
create function CalcularValorComIva(@IdS int, @horas float)
returns float
as
begin
    declare @resultado float;
    select @resultado = (s.ValorPorHora * @horas) + ((s.ValorPorHora * @horas) * s.Iva)
    from dbo.Servico s
    where @IdS = s.IdServico;
    return @resultado
end

select dbo.CalcularValorComIva(1501,2)
```

Results Messages

(No column name)

10.6

FIGURA 3.1: Função CalcularValorIva e resultado

3.2 Views

Para este trabalho criei duas views uma (Ver faturas) para visualizar as o valor total com IVA por ano das faturas, valor do IVA das faturas e o valor médio das faturas. E também criei outra views (VerFaturaPorServico) que vê os mesmos valores por ano mas para cada serviço.

```
create or alter view VerFaturas
as
select year(f.data_venda) as ano, sum(f.precoTotalComIva) as totalVendasComIva, sum(f.precoTotalSemIva) as totalVendasSemIva,
sum(f.precoTotalComIva) - sum(f.precoTotalSemIva) as valorIva, avg(f.precoTotalComIva) as MediaValorFaturas
from Fatura f group by year(f.data_venda);
```

FIGURA 3.2: View VerFatura

	ano	totalVendasComIva	totalVendasSemIva	valorIva	MediaValorFaturas	maiorServico
1	2022	38134.79	33695	4439.789999999996	65.5236941580755	582
2	2021	90317.33	79811	10506.33	67.6027919161676	1336
3	2020	79082.25999999999	69838	9244.259999999988	62.8135504368545	1259
4	2019	90951.34000000001	80140	10811.340000000001	67.7729806259315	1342
5	2018	44765.72	39566	5199.720000000002	65.1611644832606	687

FIGURA 3.3: Resultado da View VerFatura

```

create view VerFaturasPorServico
AS
select TOP 1000 year(f.data_venda) as ano, sum(f.precoTotalComIva) as totalVendasComIva, sum(f.precoTotalSemIva) as totalVendasSemIva,
sum(f.precoTotalComIva) - sum(f.precoTotalSemIva) as valorIva, avg(f.precoTotalComIva) as MediaValorFaturas, s.Descricao
from Fatura f inner join LinhaFatura lf on f.IdFatura = lf.IdFatura inner join Servico s on lf.IdServico = s.IdServico
group by year(f.data_venda), s.Descricao order by year(f.data_venda) desc;

select * from VerFaturasPorServico

```

ano	totalVendasComIva	totalVendasSemIva	valorIva	MediaValorFaturas	Descricao
2022	5609.69	4853	726.6900000000001	65.9963529411765	Jardinagem
2022	5607.43	4858	749.4300000000001	61.6201098901099	Limpeza
2022	6583.31	5975	608.3100000000001	81.2754326867854	Saude
2022	4122.53	3650	472.5299999999999	56.4730136886301	Transporte
2022	5197.13	4676	521.1299999999999	62.616024963855	Telecomunicacao
2022	4338.69	3756	582.69	58.6242307892208	Lavagem
2022	6676.01	5897	778.0599999999998	73.3627472527472	Seguranca
2021	14413.49	13085	1328.49	72.06745	Saude
2021	13359.29	11615	1744.2899999999999	63.9200478468899	Limpeza
2021	11620.6	10087	1533.6	61.8117021276986	Jardinagem
2021	11195.2	9866	1327.2	60.9434762608696	Transporte
2021	11901.84	10308	1593.8399999999999	66.1213333333333	Lavagem
2021	14111.27	12470	1641.2699999999999	77.1107650273224	Seguranca
2021	13715.64	12378	1337.6399999999999	71.435625	Telecomunicacao
2020	10536.81	9178	1458.8100000000001	56.881336863968	Limpeza
2020	13218.71	12023	1195.7100000000001	71.9408152173814	Saude
2020	13034.64	11502	1532.64	71.6188010869011	Seguranca
2020	11647.45	10534	1113.45	67.3263005780347	Telecomunicacao
2020	9527.04	8229	1298.04	56.3739177514793	Lavagem
2020	10591.35	9363	1228.35	57.2505405405405	Transporte
2020	10426.26	9009	1417.26	58.2478262568324	Jardinagem
2019	12518.56	11284	1234.56	66.9441711220946	Telecomunicacao
2019	12939.45	11175	1764.4499999999999	63.1192682826829	Limpeza
2019	14729.7	13290	1439.7	76.3196889119171	Saude
2019	12857.28	11160	1697.28	65.5983673469388	Jardinagem
2019	13506.86	11924	1582.86	75.0381111111111	Seguranca

Query executed successfully.

FIGURA 3.4: View VerFaturaPorServico e o seu resultado

Como foi pedido no enunciado foi criado um utilizador "WebFatura" que pode ler a view.

```

CREATE LOGIN WebFatura WITH PASSWORD = '123';
CREATE USER WebFatura FOR LOGIN WebFatura;

GRANT SELECT ON VerFaturas TO WebFatura;

```

FIGURA 3.5: Criação do utilizador "WebFatura" e permissões para ler a View

Capítulo 4

Stored Procedure

4.1 SPs de inserção, atualização e remoção

O SP "NovoServico" permite criar um novo Serviço para isso o sp recebe como parâmetros, o IVA, o nome do serviço e o valor de hora, o sp também controla se o valor do IVA.

```
create procedure NovoServico(@ValorPortora float,@Descricao varchar(100),@Iva float)
as
if @Iva = 0.06 OR @Iva = 0.13 OR @Iva = 0.23
insert into Servico (Iva, ValorPortora, Descricao) values (@Iva, @ValorPortora, @Descricao)
else
print 'ERRO: Valor de Iva incorreto';
return
exec NovoServico @ValorPortora = 6, @Descricao = 'Tecnologia', @Iva = 0.23
```

100 %
Messages
(1 row affected)
Completion time: 2022-06-09T14:34:15.4869786+01:00

FIGURA 4.1: SP de Inserção

O SP "Atualizar" permite atualizar o IVA de um Serviço para isso o sp recebe como parâmetros, o IVA e o IdServiço, o sp também controla se o valor do IVA.

```
create or alter procedure Atualizar(@IdServiço int,@Iva float)
as
if @Iva = 0.06 OR @Iva = 0.13 OR @Iva = 0.23
update Servico set Iva=@Iva where IdServiço=@IdServiço;
else
print 'Erro: Valores de iva incorretos';
return;
go
exec Atualizar @IdServiço = 1509, @Iva = 0.13
```

100 %
Messages
(1 row affected)
Completion time: 2022-06-09T14:39:22.4969181+01:00

FIGURA 4.2: SP de Atualização

O SP "EliminarServico" permite eliminar um serviço recebendo como parâmetro o id do serviço.

```
alter procedure EliminarServico(@idServico int)
as
begin
begin try
DELETE FROM Servico WHERE IdServico = @idServico;
end try
begin catch
print 'Erro ao Eliminar Servico'
end catch
end;

exec EliminarServico @idServico = 1508
```

%

Messages

(1 row affected)

Completion time: 2022-06-05T14:24:54.6516068+01:00

FIGURA 4.3: SP de Eliminação

4.2 SP de Verificação

O Stored Procedure "VerServico" recebe como parametro um nome de um serviço e verifica se ele existe na base de dados ou não.

```
create Procedure VerServico (@Nome varchar(100))
as
begin
declare @aux varchar(100)
select @aux = Descricao from Servico where Descricao = @Nome;
if @aux = null
select 'Servico inexistente'
else
print 'Servico existente'
end;

exec VerServico @Nome = 'Saude'
```

0 %

Messages

Servico existente

Completion time: 2022-06-05T15:38:46.9466007+01:00

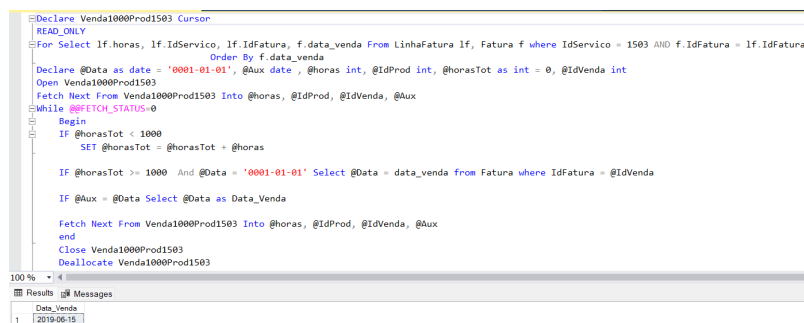
FIGURA 4.4: SP que verifica a existência de um determinado serviço

Capítulo 5

Cursor

5.1 Cursor

Foi criado um cursor "Venda1000Prod1503" que retorna a data em que o produto com id 1503 chegou as 1000 horas vendidas.



```

--Declare Venda1000Prod1503 Cursor
READ_ONLY
--For Select If.horas, If.IdServico, If.IdFatura, f.data_venda From LinhaFatura If, Fatura f where IdServico = 1503 AND f.IdFatura = If.IdFatura
Order By f.data_venda
Declare @Data as date = '0001-01-01', @Aux date, @horas int, @IdProd int, @horasTot as int = 0, @IdVenda int
Open Venda1000Prod1503
Fetch Next From Venda1000Prod1503 Into @horas, @IdProd, @IdVenda, @Aux
While @@FETCH_STATUS=0
Begin
IF @horasTot < 1000
SET @horasTot = @horasTot + @horas
IF @horasTot >= 1000 And @Data = '0001-01-01' Select @Data = data_venda from Fatura where IdFatura = @IdVenda
IF @Aux = @Data Select @Data as Data_Venda
Fetch Next From Venda1000Prod1503 Into @horas, @IdProd, @IdVenda, @Aux
end
Close Venda1000Prod1503
Deallocate Venda1000Prod1503
  
```

100 %

Results Messages

	Data_Venda
1	2019-06-18

FIGURA 5.1: Cursor "Venda1000Prod1503"

Capítulo 6

Triggers

6.1 Triggers

O trigger "InsercaoLinhaFaturaAtualiz" é disparado sempre que há uma inserção na LinhaFatura e calcula o valor total com e sem iva para a linha fatura e acrescenta esses valores aos campos da tabela Fatura "valorTotalComIva" e "valorTotalSemIva" respectivamente.

```

--create TRIGGER InsercaoLinhaFaturaAtualiz
ON LinhaFatura
AFTER INSERT
AS
BEGIN
    declare @valorPorHora float, @horas float, @valorTotalComIva float, @valorTotalSemIva float, @idFatura int, @idServico int, @iva float;
    select @idServico = i.idServico from Inserted i;
    select @idFatura = i.idFatura from Inserted i;
    select @horas = i.horas from Inserted i;
    select @iva = iva from Servico s, inserted i where s.IdServico = i.IdServico;
    select @valorPorHora = valorPorHora from Servico where IdServico=@idServico;
    update LinhaFatura set precoTotalSemIva = @horas * @valorPorHora,
    precoTotalComIva = (@horas * @valorPorHora)+((@horas * @valorPorHora) *@iva) where @idServico = IdServico
    update Fatura set Fatura.precoTotalSemIva = Fatura.precoTotalSemIva + (@horas * @valorPorHora),
    Fatura.precoTotalComIva = Fatura.precoTotalComIva + ((@horas * @valorPorHora)+((@horas * @valorPorHora) *@iva)) where IdFatura = @idFatura
END

```

FIGURA 6.1: Trigger "InsercaoLinhaFaturaAtualiz"

O trigger "updateLinhaFatura" é disparado sempre que há um update na Linha fatura e atualiza a Fatura Fazendo a diferença entre o valor que estava antes do update e o valor que foi colocado no update.

```

--create TRIGGER updateLinhaFatura
ON LinhaFatura
AFTER UPDATE
AS
BEGIN
    declare @precoTotalComIvaIn float, @precoTotalSemIvaIn float, @precoTotalComIvaDe float, @precoTotalSemIvaDe float, @idFIN int, @idFDe int;
    select @precoTotalComIvaIn = precoTotalComIva from inserted;
    select @precoTotalSemIvaIn = precoTotalSemIva from inserted;
    select @precoTotalComIvaDe = precoTotalComIva from deleted;
    select @precoTotalSemIvaDe = precoTotalSemIva from deleted;
    select @idFIN = idFatura from inserted;
    select @idFDe = idFatura from deleted;
    update Fatura set precoTotalComIva = Fatura.precoTotalComIva + (@precoTotalComIvaIn - @precoTotalComIvaDe),
    precoTotalSemIva = Fatura.precoTotalSemIva + (@precoTotalSemIvaIn - @precoTotalSemIvaDe)
    from Inserted, Deleted where Fatura.IdFatura = @idFIN and Fatura.IdFatura = @idFDe;
END

```

FIGURA 6.2: Trigger "updateLinhaFatura"

O trigger "RemocaoDeLinha" é disparado sempre que há um delete na LinhaFatura e atualiza a Fatura subtraindo o valor da LinhaFatura removida nos valores da fatura.

```
create TRIGGER RemocaoDeLinha
ON LinhaFatura
AFTER delete
As
Begin
    Declare @precoTotalSemIva float, @precoTotalComIva float, @id int;
    select @precoTotalSemIva = precoTotalSemIva from deleted;
    select @precoTotalComIva = precoTotalComIva from deleted;
    select @id = IdFatura from deleted;
Update Fatura set precoTotalSemIva = precoTotalSemIva - @precoTotalSemIva , precoTotalComIva = precoTotalComIva - @precoTotalComIva
where IdFatura = @id
end
```

FIGURA 6.3: Trigger "RemocaoDeLinha"

Capítulo 7

Selects

7.1 Select Com Pivot

O select que criei como o uso do pivot calcula a média do valor das faturas para os anos de de 2018 até 2022.

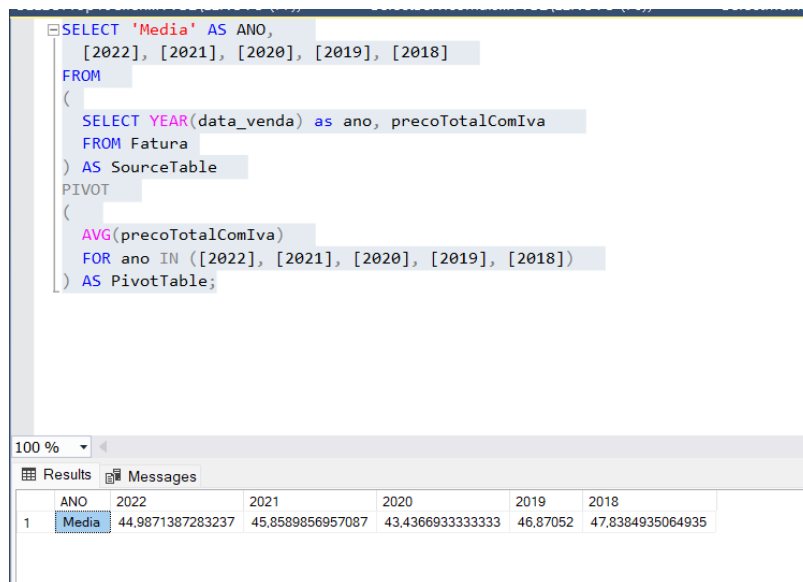


FIGURA 7.1: Select com pivot

7.2 Select Com Rank,DenseRank e RowNumber

A instrução sql foi feita com o DenseRank retrona os serviços que foram mais vendidos

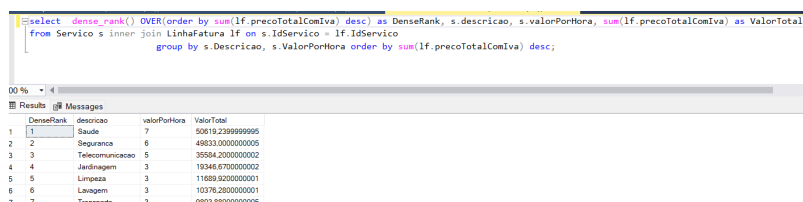


FIGURA 7.2: Select com Dense rank

O select criado utilizando o Rank retorna os 10 cliente que mais gastaram.

```

select TOP 10 RANK() OVER(order by sum(f.precoTotalComIva) desc) as Rank, c.Nome, c.Nif, sum(f.precoTotalComIva) as Valorgasto
from Cliente c inner join Fatura f on c.IdCliente = f.Id_cliente
group by c.Nome, c.Nif order by valorgasto desc;

```

Rank	Nome	Nif	Valorgasto
1	Rebekkah Brothwell	778005493	330.59
2	Greg Minster	549439000	320.72
3	Edin d' Elboux	393817019	312.95
4	Larissa Grills	191564692	293.36
5	Andreas Oullivent	147802994	286.23
6	Ansley Blackwell	565506694	275.02
7	Chris Stutterd	747608668	271.27
8	Flor Knappitt	303903458	270.01
9	Johnathan Natalie	288156815	269.87
10	Maximo Biagini	473377731	267.95

FIGURA 7.3: Select com Rank

O select com RowNumber retorna os clientes com mais de 3 compras

```

select ROW_NUMBER() Over(order by count(f.Id_cliente) desc) as Rownumber, c.Nome, count(f.Id_cliente) as NFaturas
from Cliente c inner join Fatura f on c.IdCliente = f.Id_cliente
group by c.Nome having COUNT(f.Id_cliente) > 3 order by COUNT(f.Id_cliente) desc;

```

Rownumber	Nome	NFaturas
1	Ansley Blackwell	5
2	Bertie Emmitt	4
3	Deane Pauli	4
4	Edin d' Elboux	4
5	Greg Minster	4
6	Nyssa Dewsbury	4
7	Rebekkah Brothwell	4
8	Sayers Danler	4
9	Thaddeus Whitman	4

FIGURA 7.4: Select com RowNumber

7.3 Outros Selects

O select da figura abaixo retorna o serviço que mais vezes apareceu nas faturas.

```

select num, s.idServico, s.Descricao from (
select TOP 1 (count(idServico)) as num, IdServico from LinhaFatura group by IdServico order by num desc) a
Left JOIN Servico s on a.IdServico = s.IdServico;

```

num	idServico	Descricao
792	1507	Limpeza

FIGURA 7.5: Select que retorna o serviço que mais vezes apareceu nas faturas.

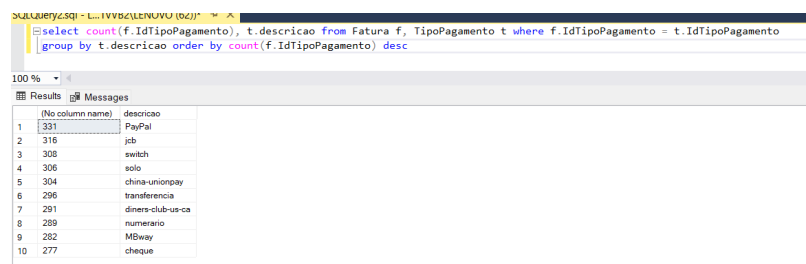
```

select TOP 1 c.localidade, count(f.Id_cliente) as QTD from Fatura f, Cliente cl, CodPostal c
where c.CodPostal = cl.CodPostal and cl.IdCliente = f.Id_cliente group by c.localidade order by count(f.Id_cliente) desc

```

localidade	QTD
Guadalupe	51

FIGURA 7.6: Select que retorna o Codigo Postal que mais vezes apareceu nas faturas.



```
SQLQueryZSQL - L... IVVSc(LTHVVU (6Z))  
select count(f.IdTipoPagamento), t.descricao from Fatura f, TipoPagamento t where f.IdTipoPagamento = t.IdTipoPagamento  
group by t.descricao order by count(f.IdTipoPagamento) desc
```

100 %

	(No column name)	descricao
1	331	PayPal
2	316	job
3	308	switch
4	306	solo
5	304	china-unionpay
6	296	transferencia
7	291	diners-club-us-ca
8	289	numerario
9	282	Midway
10	277	cheque

FIGURA 7.7: Select que retorna o tipoPagamento mais utilizado nas faturas.

Capítulo 8

Filestream

8.1 Criação da Base de Dados

Para utilizar o FileStream é necessário criar uma base de dados que suporte dados Filestream.

```
CREATE DATABASE FileStreamTP
ON
PRIMARY ( NAME = FileStreamTP,
          FILENAME = 'C:\FileTRab\FileStreamTP.mdf'),
          FILEGROUP FileStreamTPFS CONTAINS FILESTREAM(
          NAME = FileStreamTPFS,
          FILENAME = 'C:\FileTRab\FileStreamTPFS')
LOG ON (
          NAME = FileStreamTPLOG,
          FILENAME = 'C:\FileTRab\FileStreamTPLOG.ldf')
GO
```

FIGURA 8.1: Criação da Base de Dados Com suporte para Filestream.

8.2 Inserção de dados na Base de Dados

Inserção de um novo cliente na Base de dados fileStream, nos dados que são inserido está incluído uma foto

```
Insert into Cliente (Nome, Nif, Rua, NPorta, CodPostal, FSID, FSDescription, FSBlob) values
('Frank Facadas', 45565245, 'Rua Velha', 54, 4755, newid(), 'FACE', (select * from OPENROWSET
(BULK N'C:\Users\LENOVO\OneDrive - Instituto Politécnico de Viana do Castelo\Desktop\abd\trab\pessoa.jpg', SINGLE_BLOB) as FS));
```

96 -> |

Messages

(1 row affected)

Completion time: 2022-06-04T10:31:49.5749725+01:00

FIGURA 8.2: Insert no Filestream.

```
Insert into Cliente (Nome, Nif, Rua, NPorta, CodPostal, FSID, FSDescription, FSBlob) values
('Frank Facadas', 45565245, 'Rua Velha', 54, 4755, newid(), 'FACE', (select * from OPENROWSET
(BULK N'C:\Users\LENOVO\OneDrive - Instituto Politécnico de Viana do Castelo\Desktop\abd\trab\pessoa.jpg', SINGLE_BLOB) as FS));
```

96 -> |

Messages

(1 row affected)

Completion time: 2022-06-04T10:31:49.5749725+01:00

FIGURA 8.3: Insert no Filestream.


Nome	Data de modificação	tipo	tamanho
 00000029-00001048-0004	04/06/2022 15:05	Ficheiro	39 KB

FIGURA 8.4: Como podemos observar foi criado um ficheiro que contém lá dentro a imagem inserida.

Capítulo 9

Otimização de instrução SQL

9.1 otimização de uma instrução SQL

Neste capítulo vamos usar o Database Engine Tuning Advisor para analisar e aplicar recomendações para reduzir o custo de uma instrução SQL.

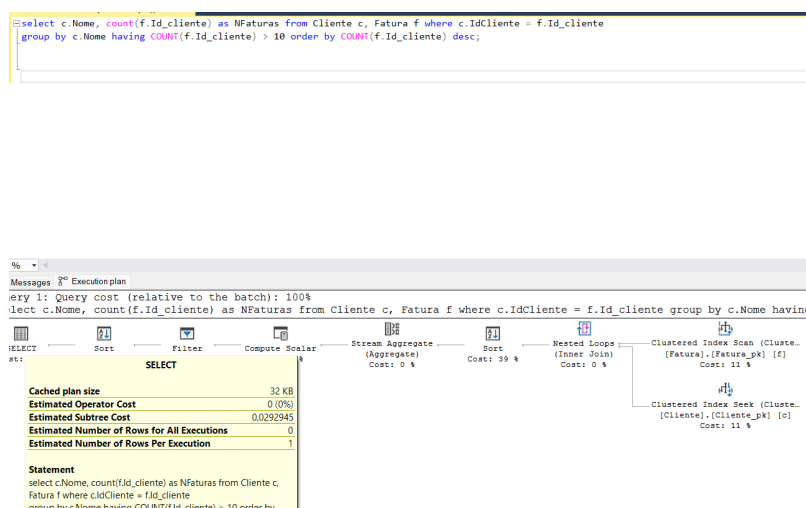


FIGURA 9.1: Com esta imagem podemos observar o custo do Select antes de utilizar o Database Engine Tuning Advisor

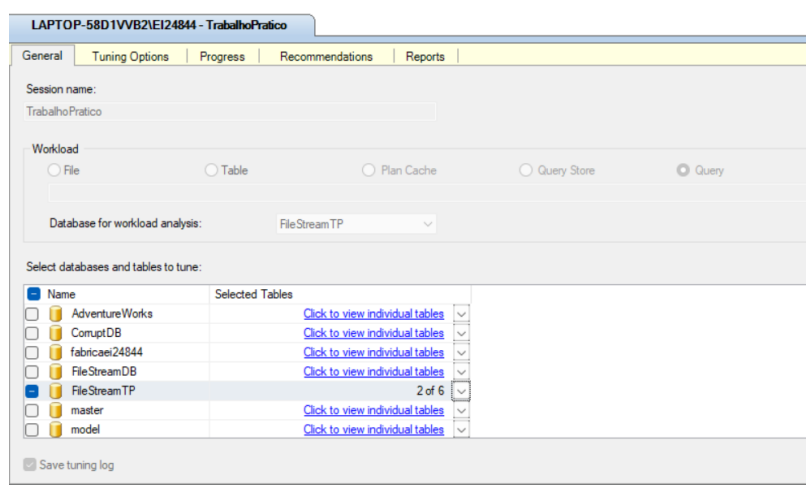


FIGURA 9.2: Seleção das tabelas no Database Engine Tuning Advisor

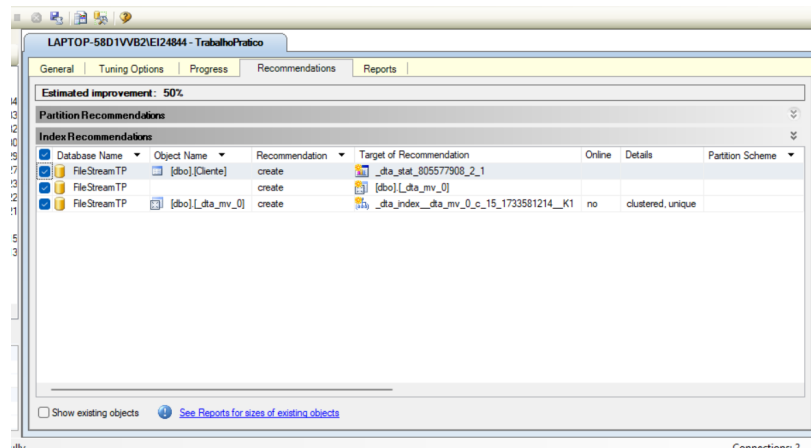


FIGURA 9.3: Recomendações geradas pelo Database Engine Tuning Advisor

Tuning Reports					
Select report: Statement detail report					
Statement Id	Statement String	Statement Type	Current Statement Cost	Recommended Statement Cost	Event String
1	select c.Nome, count(f.Id_cliente) as ...	Select	0.0292945	0.0146450	select c.Nome, count(f.Id_cliente) as ...

FIGURA 9.4: Neste print é possível verificar o custo atual da instrução e o custo recomendado

Query 1: Query cost (relative to the batch): 100%

select c.Nome, count(f.Id_cliente) as NFaturas from Cliente c, Fatura f where c.IdCliente = f.Id_cliente group by c.Nome having COUNT(f.Id_cliente) > 10 order by COUNT(f.Id_cliente) desc;

Execution plan details:

- SELECT operator: Cost: 0.014645
- Clustered Index Scan (ViewCl...): Cost: 22%
- Estimated Subtree Cost: 0.014645
- Estimated Number of Rows for All Executions: 0
- Estimated Number of Rows Per Execution: 1

FIGURA 9.5: Custo do select depois de aplicar as recomendações

Como é possível observar pelas figuras depois de aplicar as recomendações geradas pelo Database Engine Tuning Advisor, o custo de

Capítulo 10

Construção do Plano de Manutenção

10.1 Criação do Operador

Para notificar o administrador da base de dados é necessário criar um operador no SQL Server Agent.

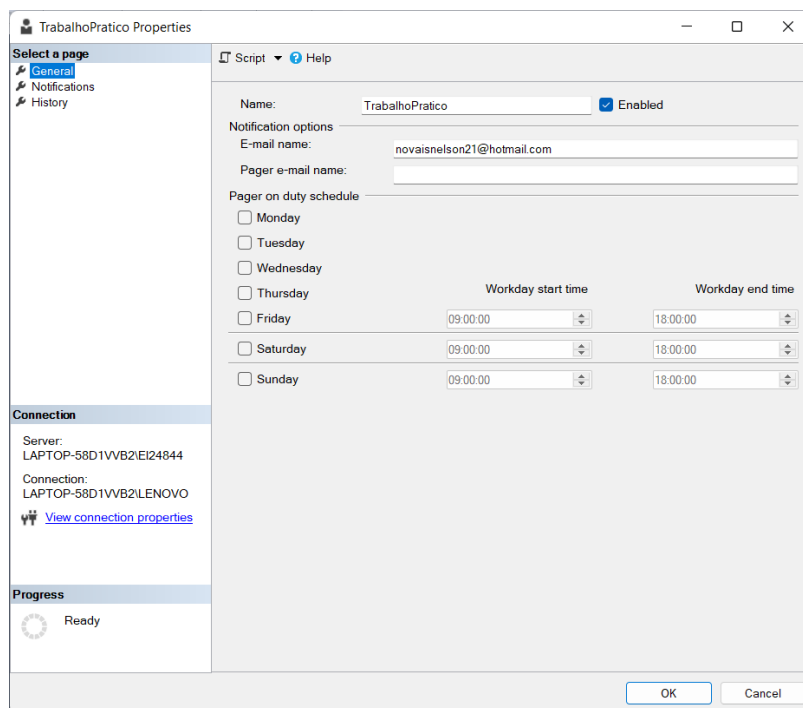


FIGURA 10.1: Criação do operador

10.2 Criação e Execução do Plano

Depois de criar o operador Foi Feito um plano diário que reorganiza Dados e Índices, valida a integridade de dados e faz um full back up

New Job Schedule

Name: MaintenancePlanTrabPratico Jobs in Schedule

Schedule type: Recurring ☒ Enabled

One-time occurrence

Date: 04/06/2022 Time: 15:49:13

Frequency

Occurs: Daily

Recurs every: 1 day(s)

Daily frequency

☒ Occurs once at: 18:00:00

☐ Occurs every: 1 hour(s) Starting at: 00:00:00 Ending at: 23:59:59

Duration

Start date: 04/06/2022 ☐ End date: 04/06/2022 ☒ No end date

Summary

Description: Occurs every day at 18:00:00. Schedule will be used starting on 04/06/2022.

OK Cancel Help

FIGURA 10.2: Criação dos horários de execução do plano

Maintenance Plan Wizard

Select Maintenance Tasks

Which tasks should this plan perform?

Select one or more maintenance tasks:

- ☒ Check Database Integrity
- ☐ Shrink Database
- ☒ Reorganize Index
- ☐ Rebuild Index
- ☐ Update Statistics
- ☐ Clean Up History
- ☐ Execute SQL Server Agent Job
- ☒ Back Up Database (Full)
- ☐ Back Up Database (Differential)
- ☐ Back Up Database (Transaction Log)
- ☐ Maintenance Cleanup Task

The Back Up Database (Full) task allows you to specify the source databases, destination files or tapes, and over full backup.

Help < Back Next > Finish Cancel

FIGURA 10.3: Seleção das tarefas a fazer no plano de manutenção

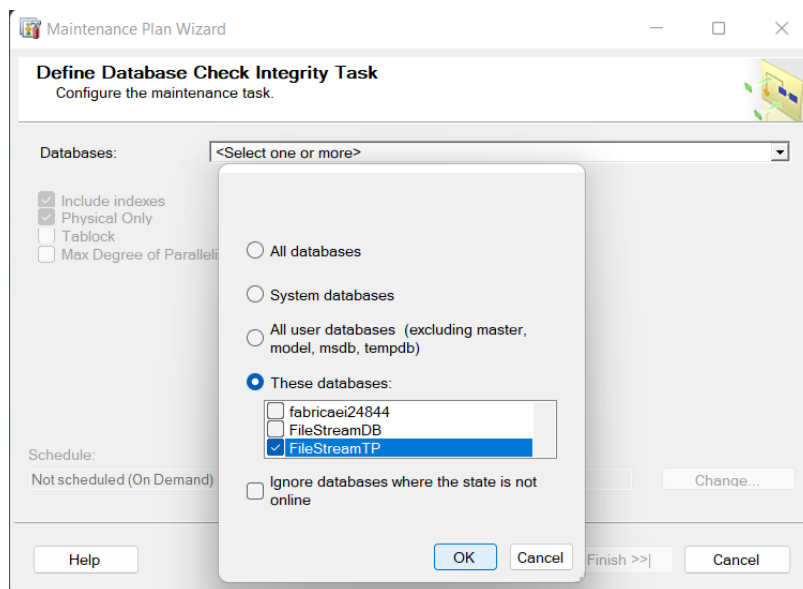


FIGURA 10.4: Seleção da base de dados para as verificações de integridade no plano de manutenção

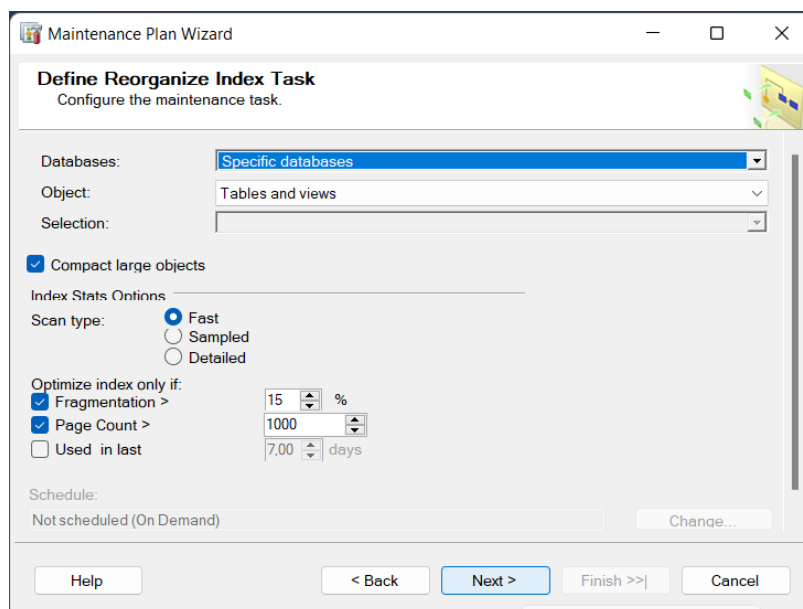


FIGURA 10.5: Seleção da base de dados para reorganizar os índices no plano de manutenção

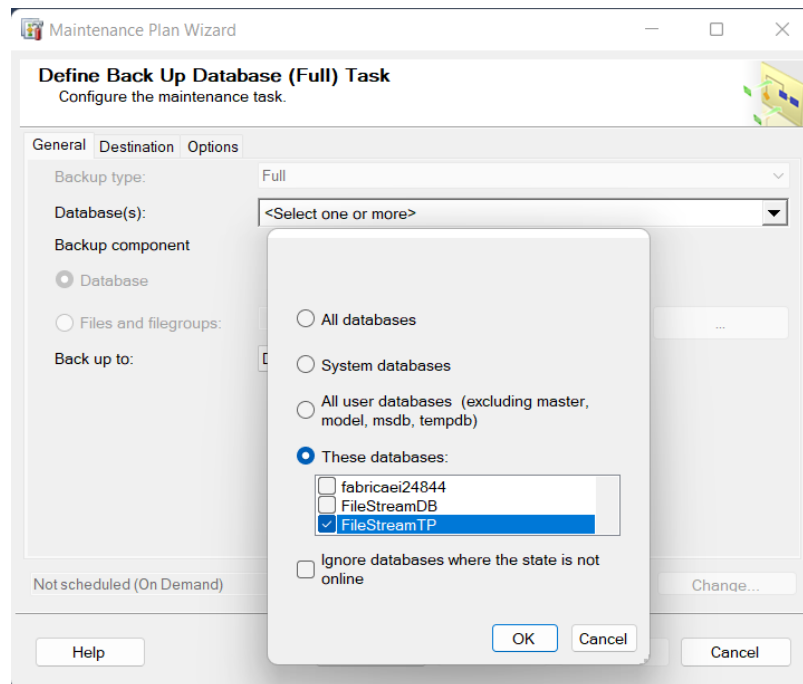


FIGURA 10.6: Seleção da base de dados para fazer o back up no plano de manutenção

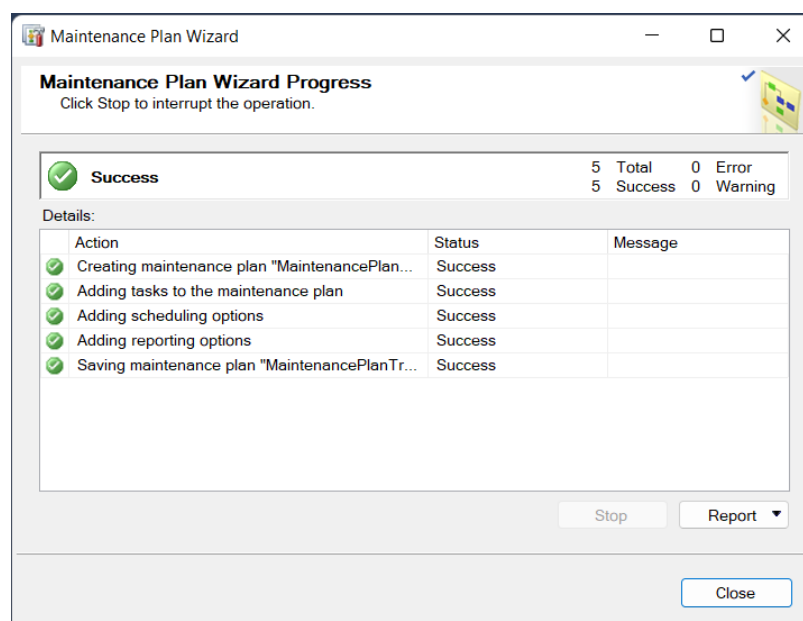


FIGURA 10.7: O plano de manutenção foi criado com sucesso

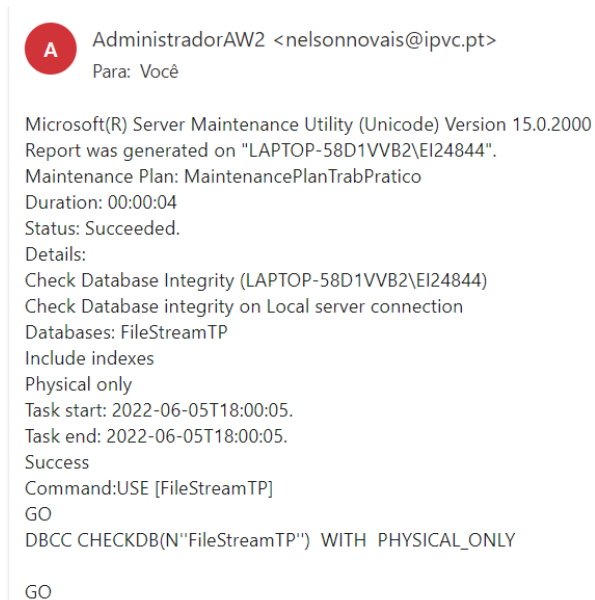


FIGURA 10.8: Como é possível verificar pela imagem o plano foi executado com sucesso às 18h

Capítulo 11

Elaboração de um relatório com o Report Builder

11.1 Construção do relatório

Nesta secção foi criado um gráfico a partir de uma instrução select

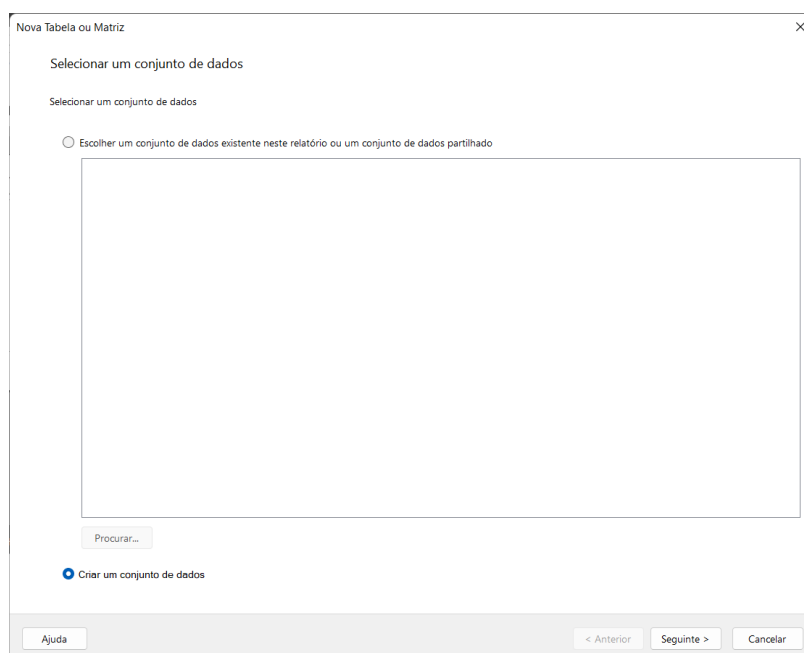


FIGURA 11.1: Para criar o grafico foi criado um novo conjunto de dados

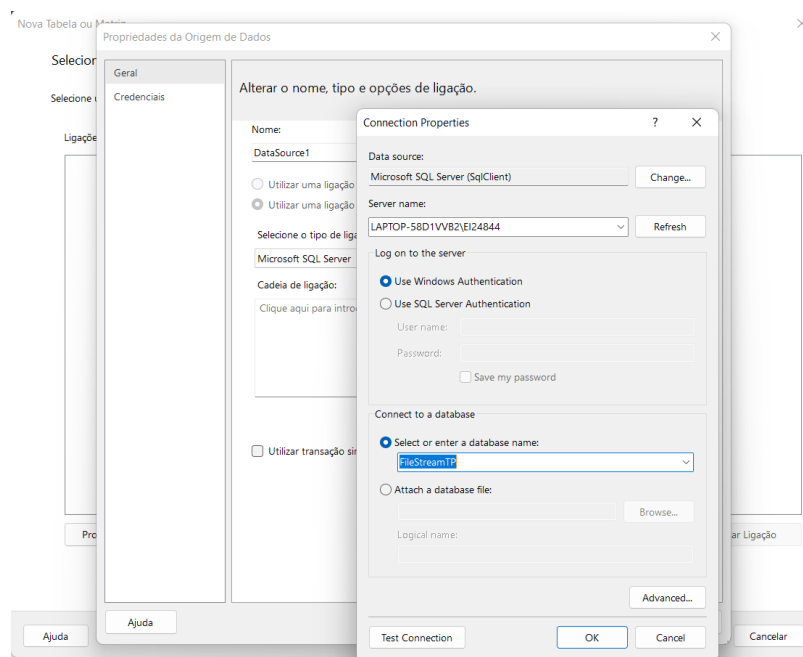


FIGURA 11.2: Ligação ao servidor e à base de dados

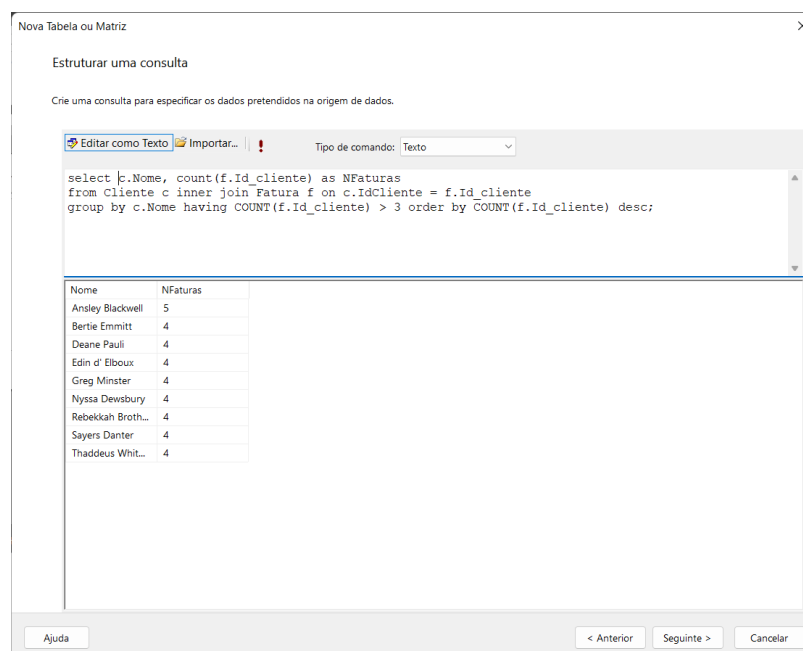


FIGURA 11.3: Criação do select para gerar o gráfico

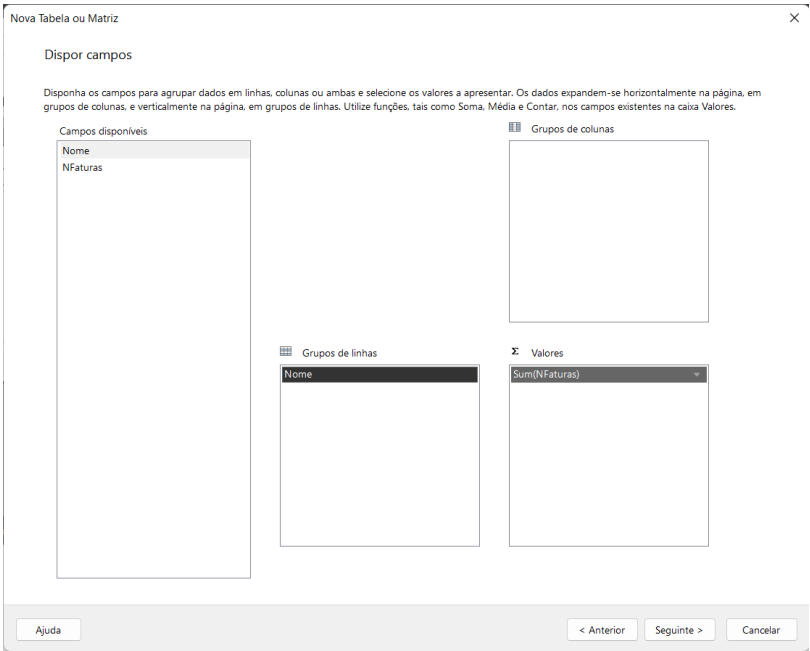


FIGURA 11.4: Disposição dos campos pelas valores e colunas

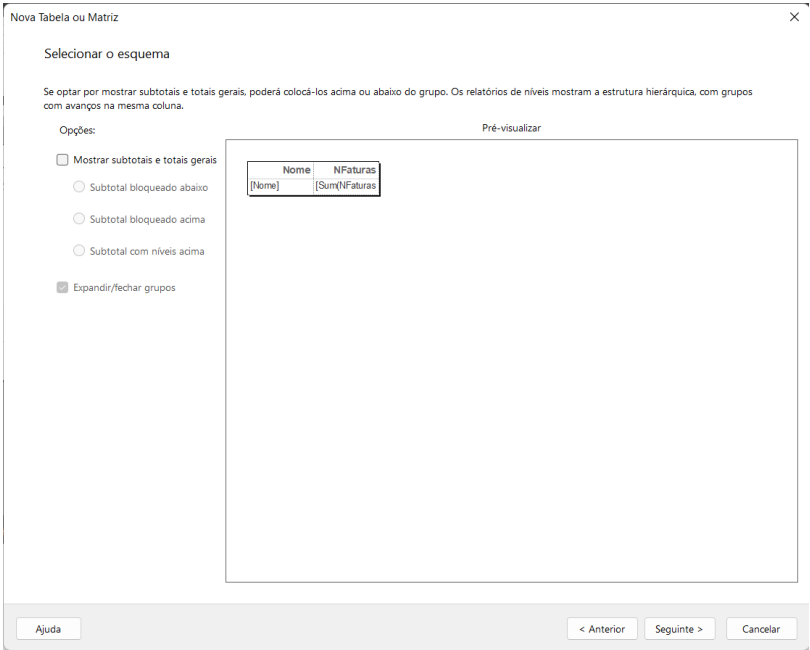


FIGURA 11.5: Estrutura do esquema que gerado

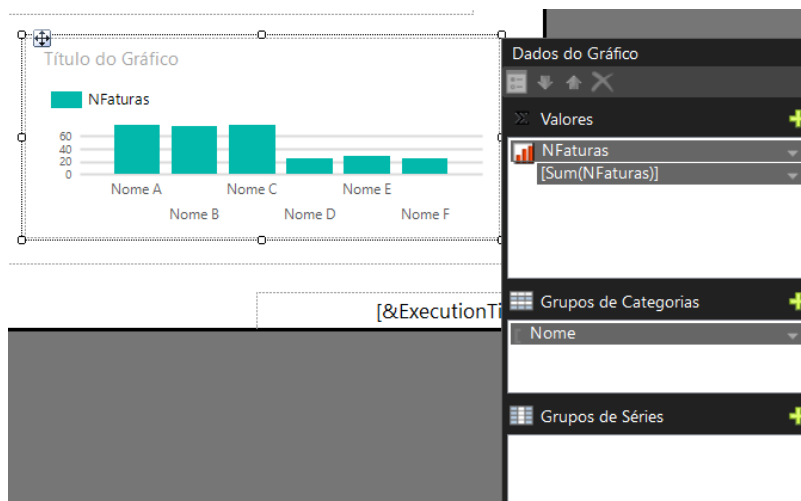


FIGURA 11.6: Disposição dos campos no gráfico

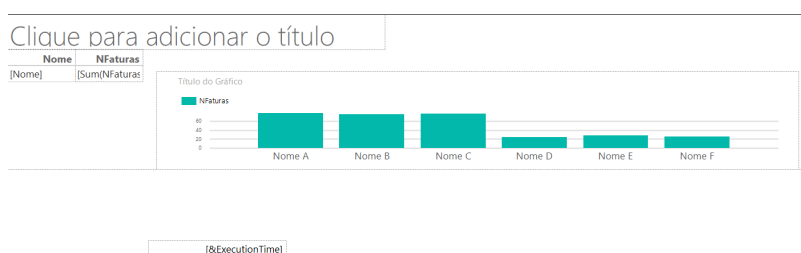


FIGURA 11.7: Esquema do relatório

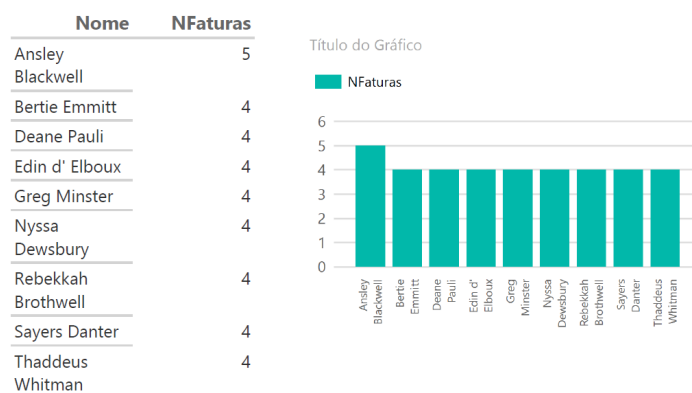


FIGURA 11.8: Relatório final

Capítulo 12

CONCLUSÕES

Este trabalho prático foi muito importante para melhorar as capacidades de administração de base de dados, e também para consolidar algumas das matérias lecionadas na disciplina de Base de Dados do semestre anterior como a criação de modelos relacionais, instruções usando join, group by, sum(), count(), entre outros.

Devido à realização do trabalho prático foi possível criar triggers, que foram a parte onde tive mais dificuldade, nomeadamente na ordem em que os triggers eram executados, mas foi resolvido juntando os dois triggers num só. Foram também criados stored procedures para validar informação, inserir, atualizar e remover dados. Uma função que retorna um valor, e cursores, entre outras matérias lecionadas nas aulas. O trabalho prático também permitiu simular as tarefas que um administrador de base de dados realiza, nomeadamente a realização de back ups, planos de manutenção, criação de utilizadores, gerar relatórios e otimizar a base de dados. O trabalho prático foi importante para fortalecer o conhecimento sobre triggers, Store Procedure, Functions e Cursores.

Este trabalho permitiu para além de trabalhar-mos com uma grande base de dados e com muitos registos conhecer muitas ferramentas do SQL SERVER que permitem fazer uma gestão correta da base de dados.

Bibliografia

Moodle de ABD(2022):

Link:<https://elearning.ipvc.pt/ipvc2021/course/view.php?id=12>

Tarefa 5(2022):

Link: <https://elearning.ipvc.pt/ipvc2021/mod/resource/view.php?id=41619>

Microsoft Docs

Link: <https://docs.microsoft.com/pt-br/sql>

Gestão de servidores BD : administração, segurança, permissões, backups e mirroring

Link:<https://elearning.ipvc.pt/ipvc2021/mod/resource/view.php?id=40891>

FILESTREAMS,PIVOT,RANK e ROW NUMBER

Link:<https://elearning.ipvc.pt/ipvc2021/mod/resource/view.php?id=42941>

Automação,monitorização e manutençãoFicheiro

Link:<https://elearning.ipvc.pt/ipvc2021/mod/resource/view.php?id=42942>