

**INSTITUTO POLITÉCNICO DE VIANA DO CASTELO**

**ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO**

**ENGENHARIA INFORMÁTICA**

**ADMINISTRAÇÃO DE BASES DE DADOS**

**2020/2021**

---

## **Trabalho Prático**

---

*Aluno:*

Tatiana Faria- 23478

*Docente:*

Marco Lima



**Instituto Politécnico  
de Viana do Castelo**

10 de junho de 2021



# Conteúdo

<b>1</b>	<b>INTRODUÇÃO</b>	<b>1</b>
<b>2</b>	<b>Modelos</b>	<b>3</b>
2.1	Modelo ER . . . . .	3
2.2	Modelo de dados- Tabelas e ligações . . . . .	3
<b>3</b>	<b>Scripts</b>	<b>5</b>
3.1	Script da base de dados . . . . .	5
3.2	Scripts do mockaroo . . . . .	6
3.3	Script criados manualmente . . . . .	7
<b>4</b>	<b>Resolução das alíneas</b>	<b>9</b>
4.1	Function que calcula o preco com iva . . . . .	9
4.2	View-WebFatura . . . . .	10
4.3	SP's com parâmetros . . . . .	12
4.4	Cursor . . . . .	14
4.5	Trigger . . . . .	15
4.6	SP com validação e retorno . . . . .	15
4.7	Instruções em sql . . . . .	17
4.8	Aplicar row-number, rank e denserank . . . . .	19
4.9	Pivot . . . . .	21
4.10	Filestream . . . . .	22
4.11	Execution Plan . . . . .	25
4.11.1	SQL Server Profiler . . . . .	25
4.11.2	Database Engine tuning advisor . . . . .	30
4.12	Importar base de dados para o azure . . . . .	33
4.13	Database Maintenance . . . . .	33
4.14	SSRS . . . . .	40
<b>5</b>	<b>CONCLUSÕES</b>	<b>43</b>
5.1	Referências . . . . .	43



## Lista de Figuras

2.1	Modelo ER . . . . .	3
2.2	Modelo de dados . . . . .	4
3.1	Script SQL para a criação da base de dados . . . . .	5
3.2	Script SQL para a criação da base de dados . . . . .	5
3.3	Script SQL que carrega dados para a tabela cliente gerado no mockaroo . . . . .	6
3.4	Script SQL que carrega dados para a tabela fatura gerado no mockaroo . . . . .	6
3.5	Script SQL que carrega dados para a tabela LinhaFatura gerado no mockaroo . . . . .	6
3.6	Script SQL que carrega dados para a tabela TipoPagamento criado manualmente . . . . .	7
3.7	Script SQL para carregar dados na tabela servico criado manualmente . . . . .	7
4.1	Function que calcula PrecoI (Preço com iva) . . . . .	9
4.2	View WebFatura . . . . .	10
4.3	View selecionada . . . . .	10
4.4	Login WebFatura . . . . .	11
4.5	Utilizador WebFatura criado . . . . .	11
4.6	Utilizador WebFatura criado . . . . .	11
4.7	Stored Procedure que apaga cliente . . . . .	12
4.8	Stored Procedure que apaga cliente- demonstração . . . . .	12
4.9	Stored Procedure que insere novo serviço . . . . .	12
4.10	Stored Procedure que insere novo serviço- demonstração . . . . .	13
4.11	Stored Procedure que faz update do cliente . . . . .	13
4.12	Cursor PrecoTotal . . . . .	14
4.13	Trigger . . . . .	15
4.14	SP que retorna informação . . . . .	15
4.15	SP que retorna informação . . . . .	16
4.16	Having . . . . .	17
4.17	Avg . . . . .	17
4.18	Sum . . . . .	18
4.19	Inner join . . . . .	18
4.20	Row Number . . . . .	19
4.21	Rank e Denserank . . . . .	20
4.22	Pivot . . . . .	21
4.23	Filestream . . . . .	22
4.24	Filestream . . . . .	22
4.25	Filestream . . . . .	23
4.26	Filestream . . . . .	23
4.27	Filestream . . . . .	23
4.28	Filestream . . . . .	24
4.29	Criar um trace . . . . .	25

4.30	Escolher caminho . . . . .	25
4.31	Opções do trace . . . . .	26
4.32	Escolha da duration . . . . .	26
4.33	Meter o nome da Base de dados . . . . .	26
4.34	Opções . . . . .	26
4.35	Run . . . . .	27
4.36	Execute script tuning . . . . .	27
4.37	Analisar o SQL Server Profiler após ter executado script tuning . . . .	27
4.38	Parar o SQL Server Profiler . . . . .	28
4.39	Analisar o SQL Server Profiler após o ter parado . . . . .	28
4.40	Mudar o caminho . . . . .	28
4.41	Voltar a executar o script tuning . . . . .	29
4.42	Executar script StopTrace . . . . .	29
4.43	General . . . . .	30
4.44	Options . . . . .	30
4.45	Recomendações . . . . .	30
4.46	Report . . . . .	31
4.47	Comparar . . . . .	31
4.48	Comparar . . . . .	31
4.49	Guardado . . . . .	31
4.50	Display do tuning . . . . .	31
4.51	Estimated Subtree Cost antes da execução do DTA Recommendations	32
4.52	Execução do DTA Recommendations . . . . .	32
4.53	Depois da execução do DTA Recommendations . . . . .	32
4.54	Estimated Subtree Cost depois da execução do DTA Recommendati- ons . . . . .	32
4.55	Criar um novo Job . . . . .	33
4.56	Escolher plano . . . . .	33
4.57	Escolher as tasks . . . . .	34
4.58	Tasks selecionadas . . . . .	34
4.59	Escolher base de dados . . . . .	34
4.60	Integrity . . . . .	35
4.61	Escolher base de dados . . . . .	35
4.62	Index . . . . .	36
4.63	Escolher base de dados . . . . .	36
4.64	Backup Full . . . . .	37
4.65	Backup Full-Destino . . . . .	37
4.66	Report Options . . . . .	37
4.67	Completo . . . . .	38
4.68	Executado com sucesso . . . . .	38
4.69	Diary executado com sucesso . . . . .	38
4.70	Ver histórico . . . . .	39
4.71	Refresh . . . . .	39
4.72	Diary Maintenance dentro da pasta Reports . . . . .	39
4.73	Base de dados na pasta Backup . . . . .	39
4.74	Criar report . . . . .	40
4.75	Conexão . . . . .	40
4.76	Propriedades das origens . . . . .	41
4.77	Ligação de origem de dados . . . . .	41
4.78	Escolher uma tabela . . . . .	41
4.79	Dispor os campos . . . . .	42

4.80 Pre-visualizar . . . . .	42
4.81 Report . . . . .	42
4.82 Report . . . . .	42





# Capítulo 1

## INTRODUÇÃO

No âmbito da cadeira de Administração de Base de Dados, o docente Marco Lima propos, mais precisamente, o estudo, de toda a matéria dada desde o início do 2º semestre através de um trabalho cujo tema era "Um sistema de gestão de faturação para serviços". Para tal, escolhi falar da gestão de faturação de um cabeleireiro, cuja base de dados suporta uma estrutura para clientes, serviços, faturas, linhas de faturas e tipos de pagamento.



## Capítulo 2

## Modelos

### 2.1 Modelo ER

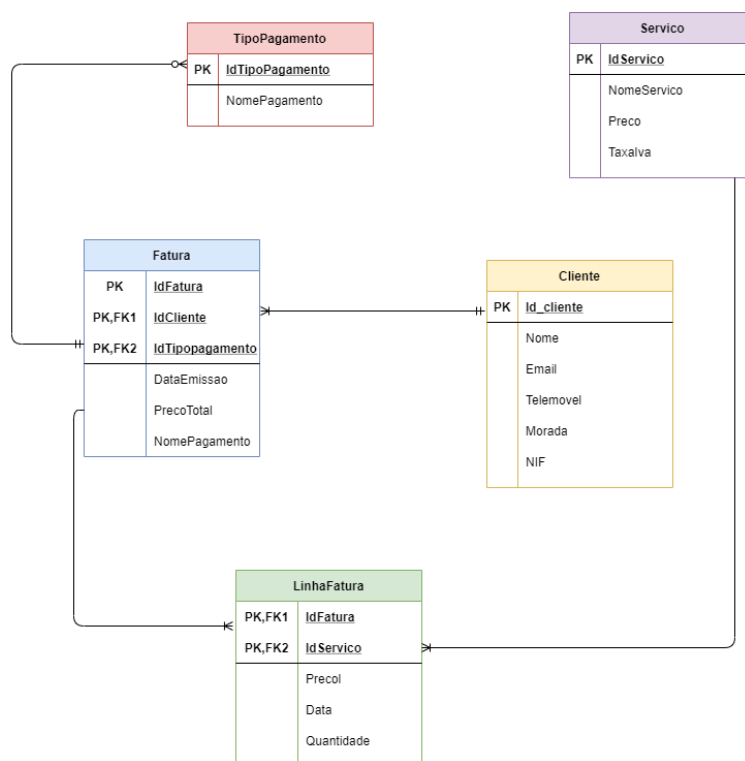


FIGURA 2.1: Modelo ER

### 2.2 Modelo de dados- Tabelas e ligações

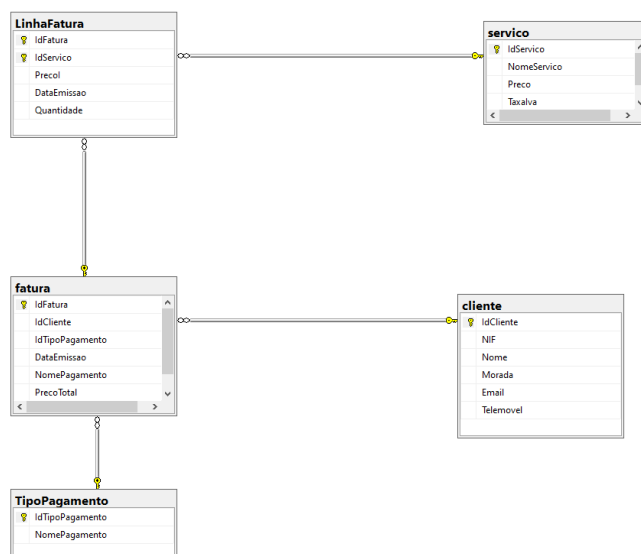


FIGURA 2.2: Modelo de dados

## Capítulo 3

## Scripts

### 3.1 Script da base de dados

```

TP_Cabeteiro.sql (54)
CREATE DATABASE TP_Cabeteiro;
GO
USE TP_Cabeteiro;
GO
CREATE TABLE cliente
(
    IdCliente INT NOT NULL IDENTITY, /*SQL Server*/
    NIF VARCHAR(10),
    Nome VARCHAR(250),
    Morada VARCHAR(250),
    Email VARCHAR(250),
    Telefone1 VARCHAR(20),
    Telefone2 VARCHAR(20),
    PRIMARY KEY (IdCliente)
);

CREATE TABLE TipoPagamento
(
    IdTipoPagamento INT NOT NULL,
    PRIMARY KEY (IdTipoPagamento),
    NomePagamento VARCHAR(250) NOT NULL
);

CREATE TABLE fatura
(
    IdFatura INT NOT NULL IDENTITY, /*SQL Server*/
    IdCliente INT,
    IdTipoPagamento INT,
    PRIMARY KEY (IdFatura),
    FOREIGN KEY (IdCliente) References cliente(IdCliente),
    FOREIGN KEY (IdTipoPagamento) References TipoPagamento(IdTipoPagamento),
    DataEmissao Date,
    Preco FLOAT,
    NomePagamento VARCHAR(250) NOT NULL,
    CONSTRAINT chk_Nome CHECK (NomePagamento IN('Numeraria', 'Transferencia', 'Paypal', 'Cheque', 'BWay')),
);

CREATE TABLE servico
(
    IdServico INT NOT NULL, /*SQL Server*/
    PRIMARY KEY (IdServico),
    NomeServico VARCHAR(250) NOT NULL,
    Preco FLOAT,
    TaxaIva FLOAT
);

```

FIGURA 3.1: Script SQL para a criação da base de dados

```

);

CREATE TABLE servico
(
    IdServico INT NOT NULL, /*SQL Server*/
    PRIMARY KEY (IdServico),
    NomeServico VARCHAR(250) NOT NULL,
    Preco FLOAT,
    TaxaIva FLOAT
);

CREATE TABLE LinhaFatura
(
    IdFatura INT NOT NULL,
    IdServico INT NOT NULL,
    PRIMARY KEY (IdFatura, IdServico),
    FOREIGN KEY (IdFatura) References fatura(IdFatura),
    FOREIGN KEY (IdServico) References servico(IdServico),
    PrecoI FLOAT,
    DataEmissao Date,
    Quantidade INT
);

```

FIGURA 3.2: Script SQL para a criação da base de dados

## 3.2 Scripts do mockaroo

Com mockaroo, gerei dados para as tabelas cliente, fatura e LinhaFatura.



FIGURA 3.3: Script SQL que carrega dados para a tabela cliente gerado no mockaroo

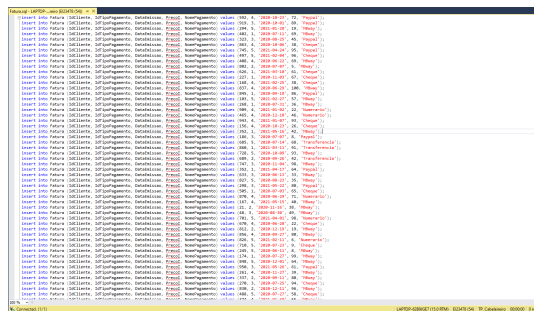


FIGURA 3.4: Script SQL que carrega dados para a tabela fatura gerado no mockaroo

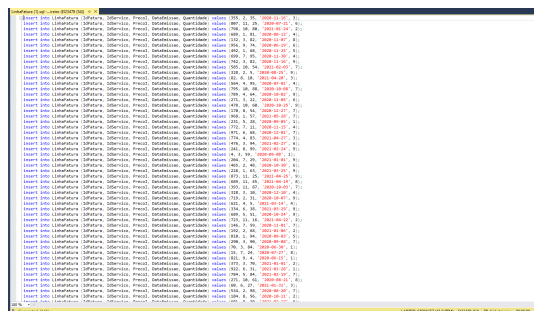
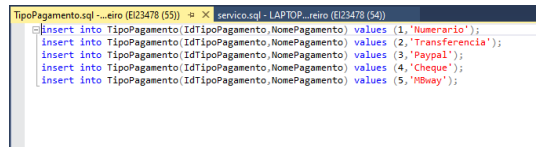


FIGURA 3.5: Script SQL que carrega dados para a tabela LinhaFatura gerado no mockaroo

### 3.3 Script criados manualmente



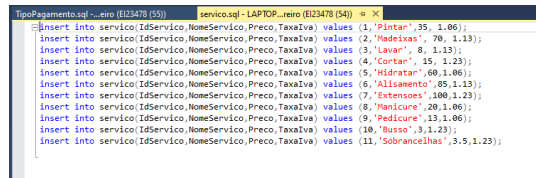
```

TipoPagamento.sql --leiro (E123478 (55))
servico.sql - LAPTOP...leiro (E123478 (54))

insert into TipoPagamento (IdTipoPagamento, NomePagamento) values (1, 'Numerario');
insert into TipoPagamento (IdTipoPagamento, NomePagamento) values (2, 'Transferencia');
insert into TipoPagamento (IdTipoPagamento, NomePagamento) values (3, 'Paypal');
insert into TipoPagamento (IdTipoPagamento, NomePagamento) values (4, 'Cheque');
insert into TipoPagamento (IdTipoPagamento, NomePagamento) values (5, 'MWay');

```

FIGURA 3.6: Script SQL que carrega dados para a tabela TipoPagamento criado manualmente



```

TipoPagamento.sql --leiro (E123478 (55))
servico.sql - LAPTOP...leiro (E123478 (54))

insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (1, 'Pintar', 35, 1.06);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (2, 'Madeiras', 70, 1.13);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (3, 'Lavar', 6, 1.13);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (4, 'Cortar', 15, 1.23);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (5, 'Hidratar', 60, 1.06);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (6, 'Alisamento', 85, 1.13);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (7, 'Extensões', 100, 1.23);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (8, 'Manicure', 20, 1.06);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (9, 'Pedicure', 13, 1.06);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (10, 'Buxxo', 3, 1.23);
insert into servico (IdServico, NomeServico, Preco, TaxaIva) values (11, 'Sobrancelhas', 3.5, 1.23);

```

FIGURA 3.7: Script SQL para carregar dados na tabela servico criado manualmente





## Capítulo 4

### Resolução das alíneas

#### 4.1 Function que calcula o preco com iva

Esta função da-nos o PrecoI(preço com iva) de cada serviço. Assim para obter o PrecoI multiplica-se o Preco (preço sem iva) com a TaxaIva, a qual lhe atribui valores com 1.06,1.13 e 1.23 para ser mais fácil fazer a multiplicação.

```
GO
CREATE FUNCTION Calcula_PrecosIva(@Preco float, @TaxaIva float)
RETURNS float
AS BEGIN
    DECLARE @PrecoI float;
    SET @PrecoI = @Preco * @TaxaIva;
    RETURN @PrecoI;
END
GO

SELECT s.IdServico, s.NomeServico, s.Preco, s.TaxaIva, (SELECT [dbo].[Calcula_PrecosIva] (s.Preco, s.TaxaIva)) AS 'PrecoI' FROM servico s
```

FIGURA 4.1: Function que calcula PrecoI (Preço com iva)

## 4.2 View-WebFatura

```
CREATE VIEW Clientes_Fatura
AS
SELECT c.Nome, c.NIF, f.DataEmissao, f.PrecoI, f.NomePagamento from fatura f, cliente c
WHERE
c.IdCliente = f.IdCliente;
```

FIGURA 4.2: View WebFatura

The screenshot shows the SQL Server Enterprise Manager interface. The top pane displays the SQL script for creating the 'Clientetes\_Fatura' view. The bottom pane shows the results of the query, which is a table with 5 columns: Nome, NIF, DataEmissao, PrecoI, and NomePagamento. The results list 19 rows of data, including client names, their NIFs, issue dates, prices, and payment methods.

	Nome	NIF	DataEmissao	PrecoI	NomePagamento
1	Vigilio Toman	492883225	2021-03-20	99	MBWay
2	Baltesonne Coubeck	162801017	2020-06-20	29	MBWay
3	Vaclav Playhill	909770168	2021-03-02	64	Transferencia
4	Bobine Emisson	346385501	2020-08-03	27	Paypal
5	Julie Abson	521769962	2021-05-04	36	Cheque
6	Julie Abson	521769962	2020-09-08	85	Cheque
7	Julie Abson	521769962	2020-12-11	75	Cheque
8	Violetta Lettley	437073680	2020-09-09	27	Cheque
9	Joana Backkaler	276549841	2020-08-08	49	MBWay
10	Joana Backkaler	276549841	2021-05-01	35	Numerato
11	Joana Backkaler	276549841	2021-03-11	75	Cheque
12	Joana Backkaler	276549841	2021-02-26	79	Cheque
13	Kelbee Sowards	186798415	2021-01-11	74	Cheque
14	Kelbee Sowards	186798415	2021-02-01	93	Cheque
15	Kelbee Sowards	186798415	2020-08-21	32	Paypal
16	Oria Evers	470404016	2021-03-26	84	MBWay
17	Belancia Margentson	598274329	2021-04-03	67	MBWay
18	Belancia Margentson	598274329	2020-08-05	60	Cheque
19	Dane Lancken	918057238	2020-11-16	38	MBWay

FIGURA 4.3: View selecionada

Depois de criar a view, criei o utilizador "WebFatura" e de seguida dei a permissão de leitura.

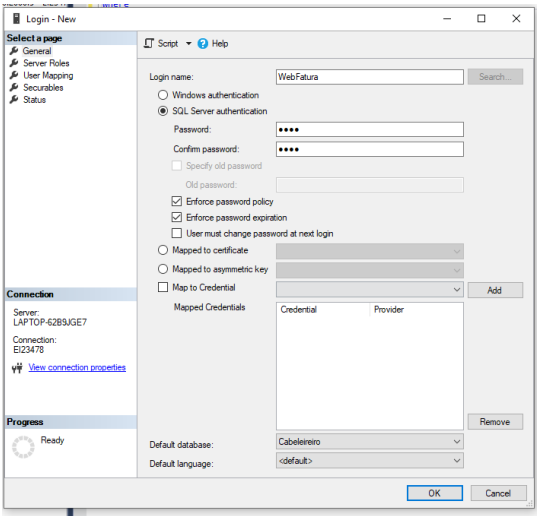


FIGURA 4.4: Login WebFatura

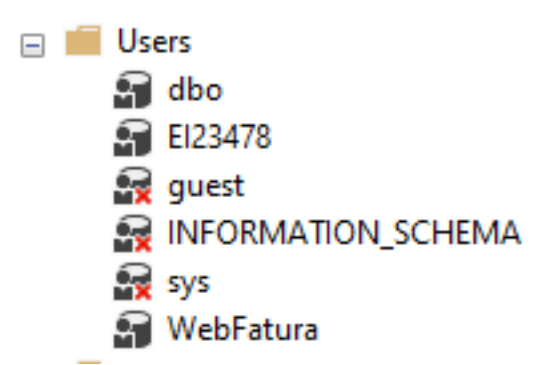


FIGURA 4.5: Utilizador WebFatura criado

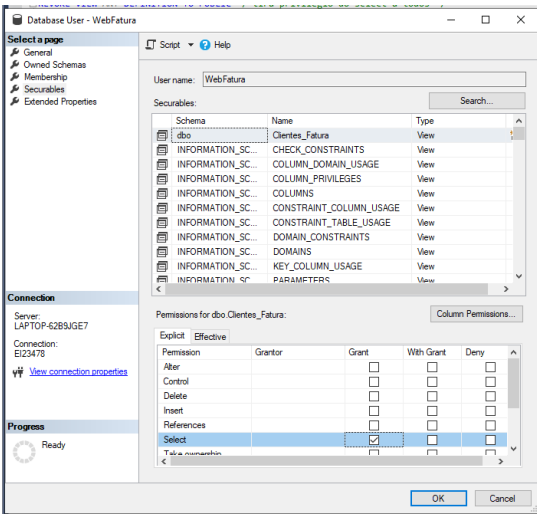


FIGURA 4.6: Utilizador WebFatura criado

### 4.3 SP's com parâmetros

```

GO
CREATE PROCEDURE Delete_Cliente @idCliente INT, @Nome VARCHAR(250), @Email VARCHAR(250), @Telefone INT, @CPF INT
AS
BEGIN
    DELETE FROM Cliente WHERE idCliente = @idCliente;
    COMMIT TRAN
END

EXEC Delete_Cliente @idCliente = 77, @Nome = 'Stacey Bailey', @Email = 'stacey20@com.ec.uk', @Telefone = 997059910, @CPF = '983 Vermet Hill', @CPF = 740554006

```

FIGURA 4.7: Stored Procedure que apaga cliente

SQLQuery13.sql - LA...reio (E13478 (76))

Select \* from cliente;

idCliente	NIF	Nome	Morada	Email	Telefone
64	590947052	Corine Robble	047 Holy Cross Plaza	cmblee1@buccz.com	971450985
65	190020806	Lauretta De Ruggiero	312 Mcguire Junction	kle1s@q.co	960562049
66	997802402	Mareesa Patricsson	86 Boyd Crossing	mpatricsson1@wordpress.com	906212717
67	899289257	Fred Burhkie	3966 Lawn Plaza	frubhie1u@godaddy.com	954059782
68	233791970	Darl McPartling	5 Hensck Park	dmcpartling1u@echoanchor.com	947637372
69	920626211	Gabouella Rosewaine	40 Towne Junction	grrosewaine1u@icloud.com	901856326
70	745993811	Paula Westmoneland	61424 Cambridge Trail	pwestmoneland1u@seesaa.net	905164430
71	835945592	Bethany Brach	4312 Rusk Avenue	bbrach1u@amazon.co.jp	995016259
72	956132290	Grata Laughton	1873 Stone Corner Court	glaughton1u@mtv.com	992762655
73	670731532	Karl Mecke	7 Karstene Pass	kmecke120@buzzfeed.com	918762247
74	487757271	Trescha Sargeant	2 Laurel Circle	tsargeant21@iscnoodledy.com	900532379
75	588584105	Peggie Margett	37940 South Trail	pmargett22@quantcast.com	903744001
76	858292173	Juliana Beare	4 Norway Maple Terrace	jbeare23@medafire.com	900818136
77	217888947	Teddy Argabrite	2 Melvin Alley	targabrite25@163.com	999377448
78	434613251	Deana Irvine	901 Westerfeld Parkway	d Irvine26@nbean.gov.cn	905642559

FIGURA 4.8: Stored Procedure que apaga cliente- demonstração

SQLQuery12.sql - LA...reio (E13478 (52))

SQLQuery10.sql - LA...reio (E13478 (60))

sql\_server\_6.sql - 1...master (E13478 (80))

```

CREATE PROCEDURE Inserir_Servico @idServico INT, @NomeServico VARCHAR(250), @Preco float, @Taxaiva float
AS
BEGIN
    INSERT INTO servico(idservico, NomeServico, Preco, Taxaiva)
    VALUES (@idServico, @NomeServico, @Preco, @Taxaiva)
END
PRINT ('O servico foi adicionado com sucesso!')

EXEC Inserir_Servico @idServico = 12, @NomeServico = 'Maquiagem', @Preco = 25, @Taxaiva = 1.86;

```

1 row affected!  
O servico foi adicionado com sucesso!  
Completion time: 2021-04-06T16:56:02.2045588+01:00

FIGURA 4.9: Stored Procedure que insere novo serviço

SQLQuery13.sql - LA...reiro (EI23478 (53))\* SQLQuery12.sql - LA...reir

```
Select * from servico;
```

100 %

Results Messages

	IdServico	NomeServico	Preco	Taxalva
1	1	Pintar	35	1,06
2	2	Madeixas	70	1,13
3	3	Lavar	8	1,13
4	4	Cortar	15	1,23
5	5	Hidratar	60	1,06
6	6	Alisamento	85	1,13
7	7	Extensoes	100	1,23
8	8	Manicure	20	1,06
9	9	Pedicure	13	1,06
10	10	Busso	3	1,23
11	11	Sobrancelhas	3,5	1,23
12	12	Maquiagem	25	1,06

FIGURA 4.10: Stored Procedure que insere novo serviço- demonstração

```

EXECUTE PROCEDURE update_cliente (@cliente INT, @nome VARCHAR(255), @email VARCHAR(255), @telefone BIGINT, @senha VARCHAR(255), @cpf BIGINT)
AS
BEGIN
    UPDATE cliente
    SET nome = @nome, email = @email,
        senha = @senha, telefone = @telefone, cpf = @cpf
    WHERE id_cliente = @cliente;
END

```

Results Messages

	id_cliente	nome	email	telefone
1	1	João Batista	joao@baptista.com	94331111
2	2	João Batista	joao@baptista.com	94331111
3	3	João Batista	joao@baptista.com	94331111
4	4	João Batista	joao@baptista.com	94331111
5	5	João Batista	joao@baptista.com	94331111
6	6	João Batista	joao@baptista.com	94331111
7	7	João Batista	joao@baptista.com	94331111
8	8	João Batista	joao@baptista.com	94331111
9	9	João Batista	joao@baptista.com	94331111
10	10	João Batista	joao@baptista.com	94331111
11	11	João Batista	joao@baptista.com	94331111
12	12	João Batista	joao@baptista.com	94331111
13	13	João Batista	joao@baptista.com	94331111
14	14	João Batista	joao@baptista.com	94331111
15	15	João Batista	joao@baptista.com	94331111
16	16	João Batista	joao@baptista.com	94331111
17	17	João Batista	joao@baptista.com	94331111

FIGURA 4.11: Stored Procedure que faz update do cliente

## 4.4 Cursor

```

1 DECLARE PrecoTotal_cursor CURSOR FOR
2   SELECT PrecoTotal, IdFatura FROM fatura
3   FOR UPDATE;
4
5 OPEN PrecoTotal_cursor;
6 DECLARE @Preco1 float;
7 DECLARE @IdFatura int;
8
9 FETCH NEXT FROM PrecoTotal_cursor INTO @Preco1, @IdFatura;
10 WHILE @@FETCH_STATUS = 0
11 BEGIN
12   UPDATE fatura
13   SET PrecoTotal = (@Preco1 + (@Preco1 * Quantidade)) FROM LinhasFatura WHERE IdFatura = @IdFatura WHERE CURRENT OF PrecoTotal_cursor
14   ROLLBACK;
15
16   FETCH NEXT FROM PrecoTotal_cursor INTO @Preco1, @IdFatura;
17 END
18
19 CLOSE PrecoTotal_cursor;
20
21 DEALLOCATE PrecoTotal_cursor;
22
23 SELECT * FROM fatura;

```

25 % -

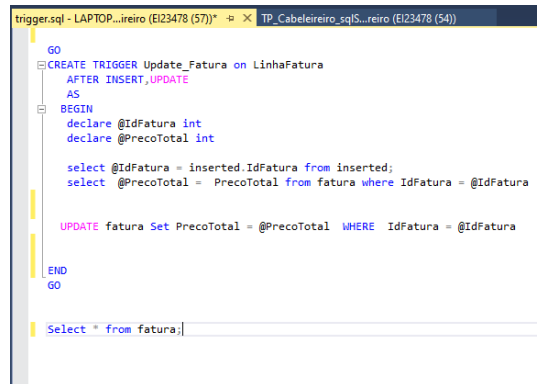
Results Messages

	IdFatura	IdCliente	IdTipoPagamento	DataEmissao	NomePagamento	PrecoTotal
1	1	592	4	2020-10-23	Paypal	818
2	2	919	3	2020-10-01	Paypal	NULL
3	3	294	5	2021-01-28	MBway	819
4	4	402	1	2020-07-11	MBway	59
5	5	323	3	2020-09-25	Paypal	NULL
6	6	863	4	2020-10-06	Cheque	560
7	7	745	5	2021-04-04	Paypal	475
8	8	497	5	2021-02-04	Cheque	104
9	9	408	4	2020-08-22	MBway	21
10	10	602	2	2020-07-07	MBway	96
11	11	626	1	2021-03-10	Cheque	180
12	12	227	1	2020-11-09	Cheque	NULL
13	13	186	4	2021-02-25	Cheque	784
14	14	837	4	2020-06-29	MBway	NULL
15	15	845	1	2020-09-18	Paypal	280
16	16	103	5	2021-02-27	MBway	802

Query executed successfully.

FIGURA 4.12: Cursor PrecoTotal

## 4.5 Trigger



```
trigger.sql - LAPTOP...reiro (E123478 (57)) * X TP_Cabeleireiro.sqlS...reiro (E123478 (54))

GO
CREATE TRIGGER Update_Fatura on LinhaFatura
AFTER INSERT, UPDATE
AS
BEGIN
    declare @IdFatura int
    declare @PrecoTotal int

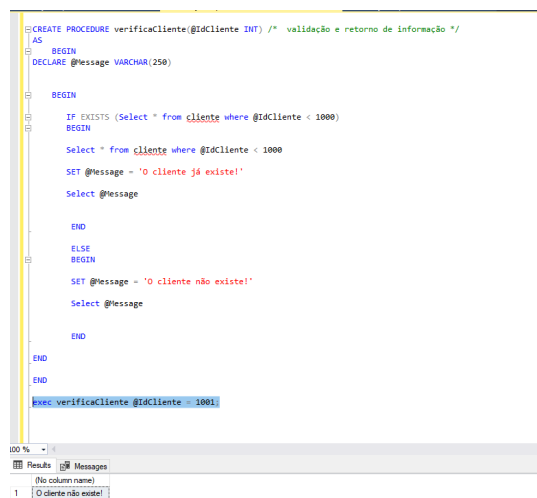
    select @IdFatura = inserted.IdFatura from inserted;
    select @PrecoTotal = PrecoTotal from fatura where IdFatura = @IdFatura

    UPDATE fatura Set PrecoTotal = @PrecoTotal WHERE IdFatura = @IdFatura
END
GO

Select * from fatura;
```

FIGURA 4.13: Trigger

## 4.6 SP com validação e retorno



```
--CREATE PROCEDURE verificaCliente(@IdCliente INT) /* validação e retorno de informação */
AS
BEGIN
    DECLARE @Message VARCHAR(250)

    BEGIN
        IF EXISTS (Select * from cliente where @IdCliente < 1000)
        BEGIN
            Select * from cliente where @IdCliente < 1000

            SET @Message = 'O cliente já existe!'
            Select @Message

        END
        ELSE
        BEGIN
            SET @Message = 'O cliente não existe!'
            Select @Message

        END
    END
END

EXEC verificaCliente @IdCliente = 1001
```

100 %

Results	Messages
(No column name)	
1	O cliente não existe!

FIGURA 4.14: SP que retorna informação

```
CREATE PROCEDURE verificaCliente(@IdCliente INT) /* validação e retorno de informação */
AS
BEGIN
    DECLARE @message VARCHAR(250)

    BEGIN
        IF EXISTS (Select * from cliente where @IdCliente < 1000)
        BEGIN
            Select * from cliente where @IdCliente < 1000

            SET @message = 'O cliente já existe!'

            Select @message

        END
        ELSE
        BEGIN
            SET @message = 'O cliente não existe!'

            Select @message

        END
    END
END

EXEC verificaCliente @IdCliente = 897
```

Results	Messages
5	5
6	6
7	7
8	8
9	9
10	10
11	11
(No column name)	
1	O cliente já existe!

FIGURA 4.15: SP que retorna informação



4.7 Instruções em sql

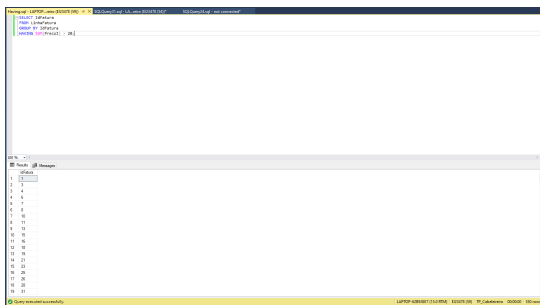


FIGURA 4.16: Having

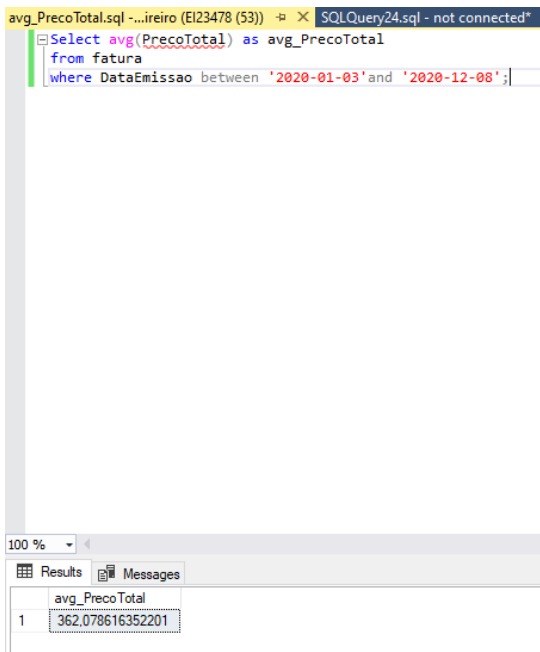
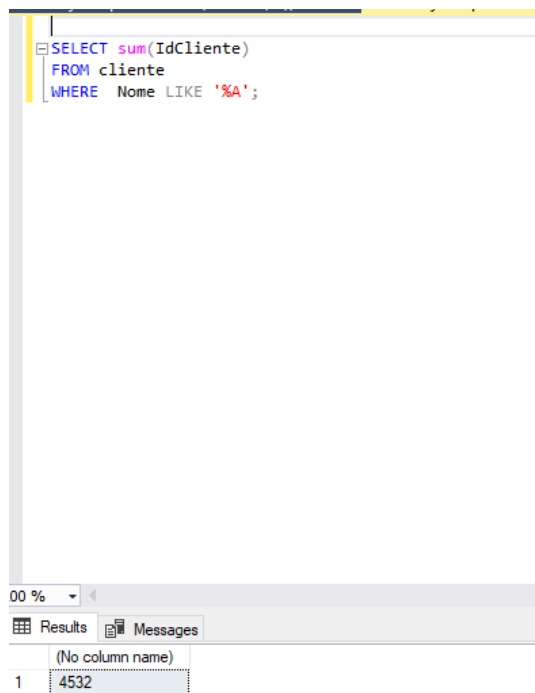


FIGURA 4.17: Avg



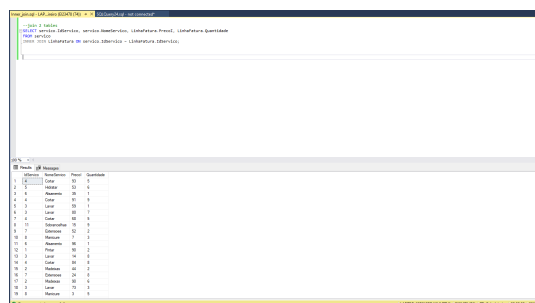
The screenshot shows a SQL query window with the following text:

```
SELECT sum(IdCliente)  
FROM cliente  
WHERE Nome LIKE '%A';
```

Below the query window, the 'Results' tab is active, displaying a single row of data:

	(No column name)
1	4532

FIGURA 4.18: Sum



The screenshot shows a SQL query window with the following text:

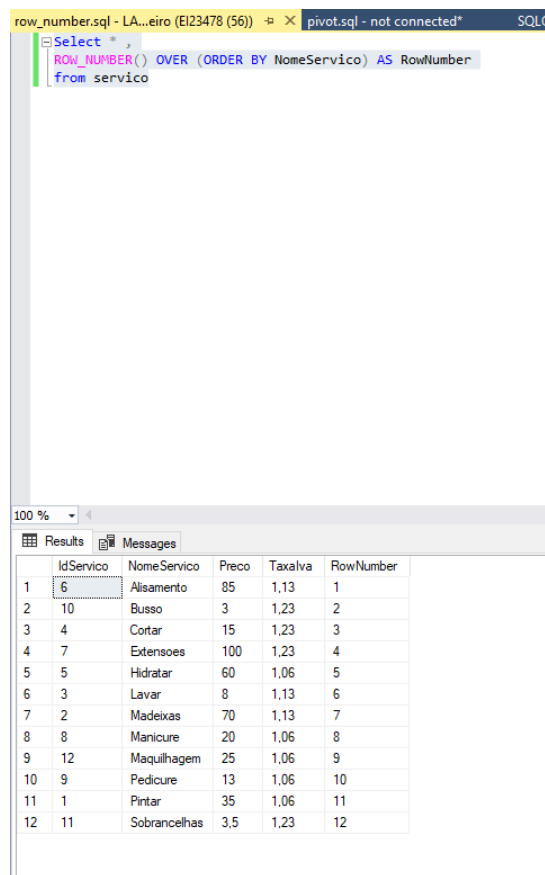
```
SELECT * FROM cliente  
INNER JOIN cliente ON cliente.IdCliente = cliente.IdCliente
```

Below the query window, the 'Results' tab is active, displaying a list of results:

IdCliente	Nome	Quantidade
1	Adriano	1
2	Adriano	1
3	Adriano	1
4	Adriano	1
5	Adriano	1
6	Adriano	1
7	Adriano	1
8	Adriano	1
9	Adriano	1
10	Adriano	1
11	Adriano	1
12	Adriano	1
13	Adriano	1
14	Adriano	1
15	Adriano	1
16	Adriano	1
17	Adriano	1
18	Adriano	1
19	Adriano	1
20	Adriano	1
21	Adriano	1
22	Adriano	1
23	Adriano	1
24	Adriano	1
25	Adriano	1
26	Adriano	1
27	Adriano	1
28	Adriano	1
29	Adriano	1
30	Adriano	1
31	Adriano	1
32	Adriano	1
33	Adriano	1
34	Adriano	1
35	Adriano	1
36	Adriano	1
37	Adriano	1
38	Adriano	1
39	Adriano	1
40	Adriano	1
41	Adriano	1
42	Adriano	1
43	Adriano	1
44	Adriano	1
45	Adriano	1
46	Adriano	1
47	Adriano	1
48	Adriano	1
49	Adriano	1
50	Adriano	1
51	Adriano	1
52	Adriano	1
53	Adriano	1
54	Adriano	1
55	Adriano	1
56	Adriano	1
57	Adriano	1
58	Adriano	1
59	Adriano	1
60	Adriano	1
61	Adriano	1
62	Adriano	1
63	Adriano	1
64	Adriano	1
65	Adriano	1
66	Adriano	1
67	Adriano	1
68	Adriano	1
69	Adriano	1
70	Adriano	1
71	Adriano	1
72	Adriano	1
73	Adriano	1
74	Adriano	1
75	Adriano	1
76	Adriano	1
77	Adriano	1
78	Adriano	1
79	Adriano	1
80	Adriano	1
81	Adriano	1
82	Adriano	1
83	Adriano	1
84	Adriano	1
85	Adriano	1
86	Adriano	1
87	Adriano	1
88	Adriano	1
89	Adriano	1
90	Adriano	1
91	Adriano	1
92	Adriano	1
93	Adriano	1
94	Adriano	1
95	Adriano	1
96	Adriano	1
97	Adriano	1
98	Adriano	1
99	Adriano	1
100	Adriano	1

FIGURA 4.19: Inner join

## 4.8 Aplicar row-number, rank e denserank

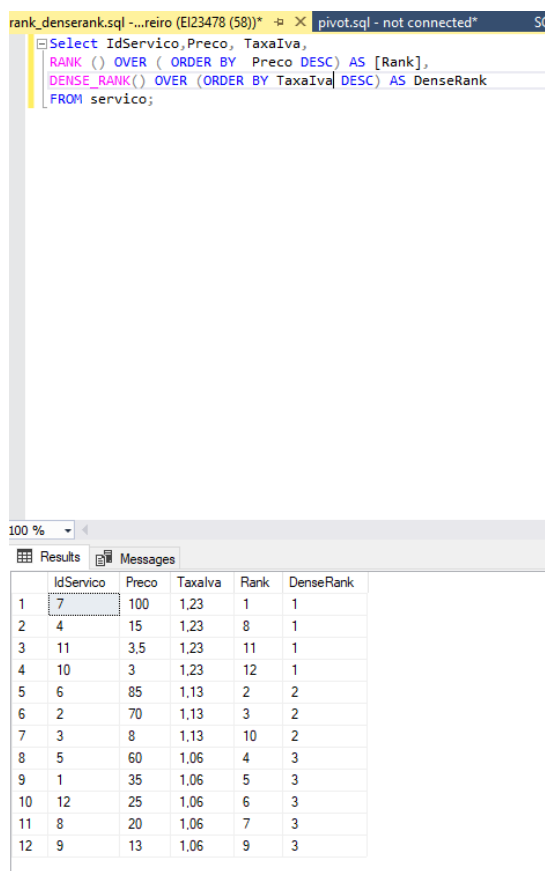


```
row_number.sql - LA...eiro (EI23478 (56)) - X pivot.sql - not connected* SQLC
Select *
, ROW_NUMBER() OVER (ORDER BY NomeServico) AS RowNumber
from servico
```

100 %

	IdServico	NomeServico	Preco	Taxalva	RowNumber
1	6	Alisamento	85	1,13	1
2	10	Busso	3	1,23	2
3	4	Cortar	15	1,23	3
4	7	Extensoes	100	1,23	4
5	5	Hidratar	60	1,06	5
6	3	Lavar	8	1,13	6
7	2	Madeixas	70	1,13	7
8	8	Manicure	20	1,06	8
9	12	Maquilhagem	25	1,06	9
10	9	Pedicure	13	1,06	10
11	1	Pintar	35	1,06	11
12	11	Sobrancelhas	3,5	1,23	12

FIGURA 4.20: Row Number



The screenshot shows a SQL query editor with a query window titled 'rank\_denserank.sql - ...reio (EI23478 (58))' and a results window titled 'pivot.sql - not connected\*'. The query is as follows:

```
Select IdServico, Preco, TaxaIva,  
RANK () OVER ( ORDER BY Preco DESC) AS [Rank],  
DENSE_RANK() OVER (ORDER BY TaxaIva DESC) AS DenseRank  
FROM servico;
```

The results window shows a table with 12 rows and 5 columns: IdServico, Preco, TaxaIva, Rank, and DenseRank. The data is as follows:

	IdServico	Preco	TaxaIva	Rank	DenseRank
1	7	100	1,23	1	1
2	4	15	1,23	8	1
3	11	3,5	1,23	11	1
4	10	3	1,23	12	1
5	6	85	1,13	2	2
6	2	70	1,13	3	2
7	3	8	1,13	10	2
8	5	60	1,06	4	3
9	1	35	1,06	5	3
10	12	25	1,06	6	3
11	8	20	1,06	7	3
12	9	13	1,06	9	3

FIGURA 4.21: Rank e Denserank



## 4.10 Filestream

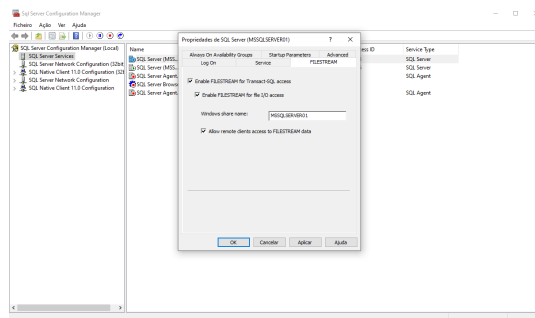


FIGURA 4.23: Filestream

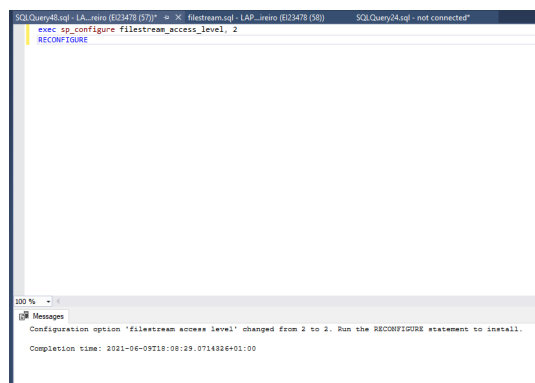


FIGURA 4.24: Filestream

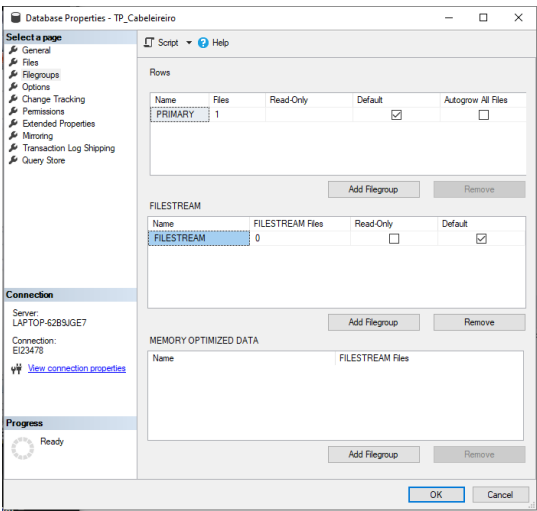


FIGURA 4.25: Filestream

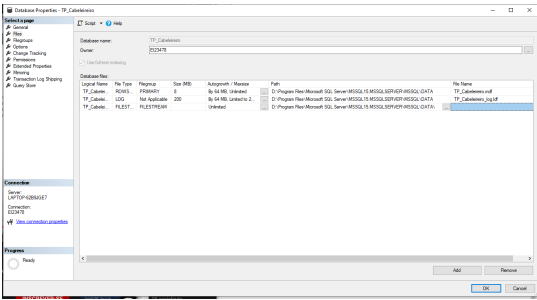


FIGURA 4.26: Filestream

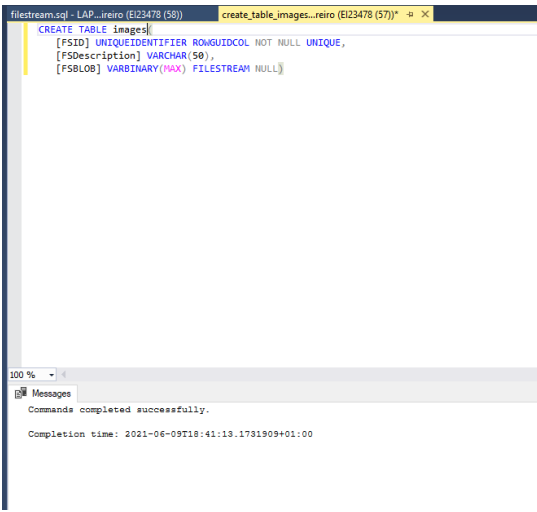


FIGURA 4.27: Filestream

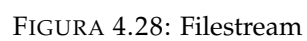


FIGURA 4.28: Filestream



Aparece duas imagens porque eu estava a ter problemas com o caminho da imagem e para tentar se o problema era do caminho da minha imagem ou se era algo que tinha feito mal, coloquei uma imagem que o professor tinha na tarefa5. Com isso conclui que era o caminho da minha imagem que estava errado. Então decidi criar uma pasta "TrabalhoPratico" no disco C, onde coloquei a minha imagem e com isso o caminho da imagem ficou correto, tendo conseguido carregar a imagem para a tabela images que criei.

## 4.11 Execution Plan

### 4.11.1 SQL Server Profiler

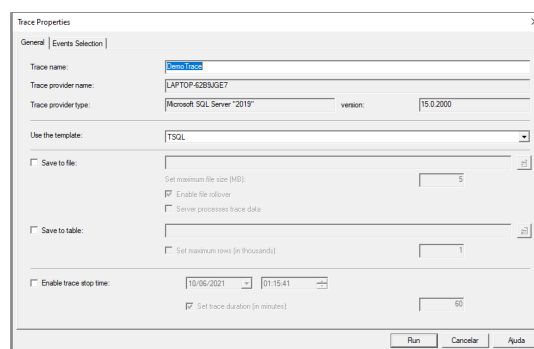


FIGURA 4.29: Criar um trace

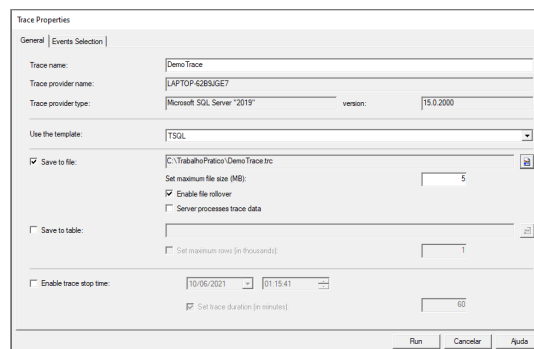


FIGURA 4.30: Escolher caminho

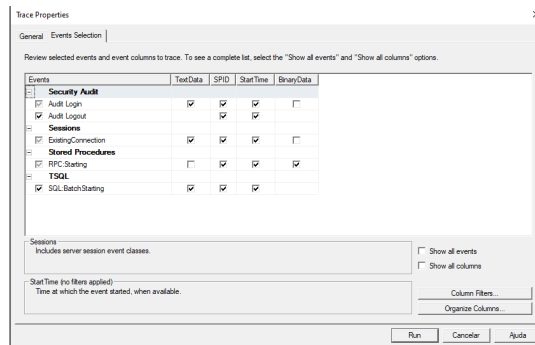


FIGURA 4.31: Opções do trace

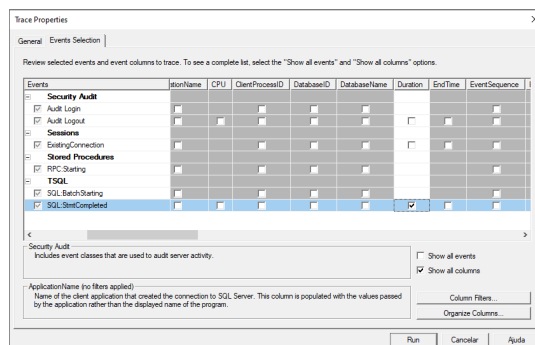


FIGURA 4.32: Escolha da duration

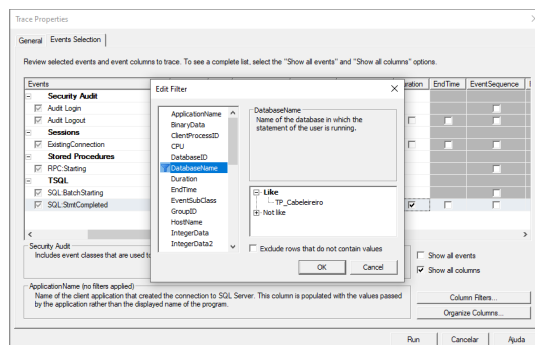


FIGURA 4.33: Meter o nome da Base de dados

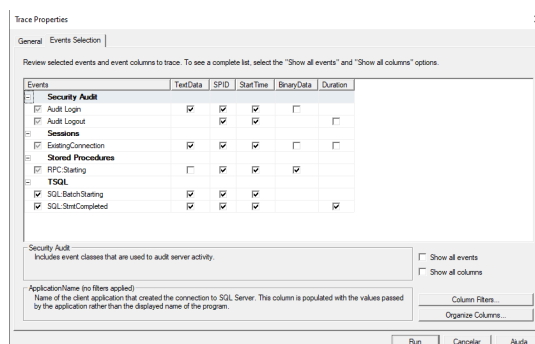


FIGURA 4.34: Opções

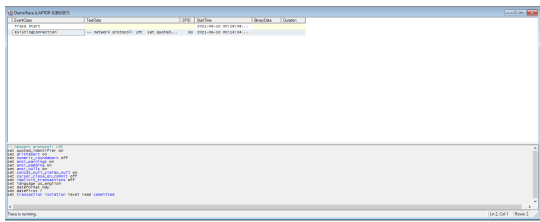


FIGURA 4.35: Run

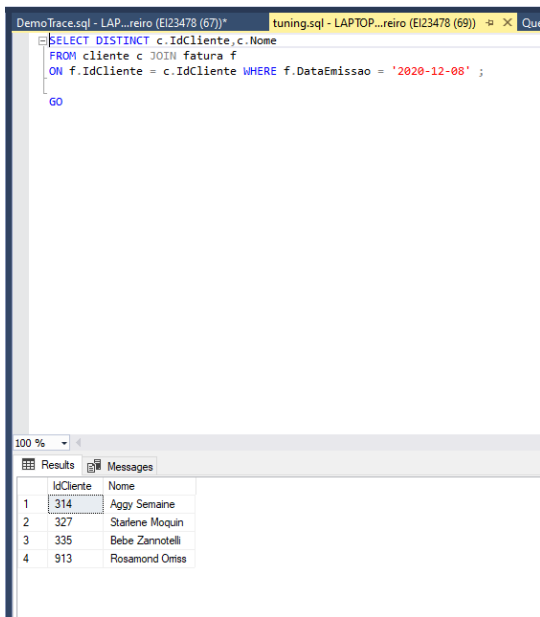


FIGURA 4.36: Execute script tuning

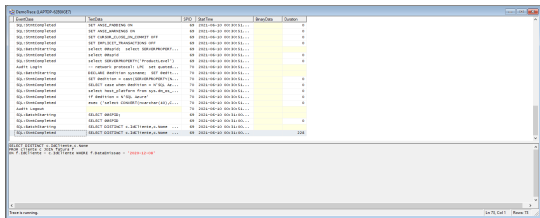


FIGURA 4.37: Analisar o SQL Server Profiler após ter executado script tuning



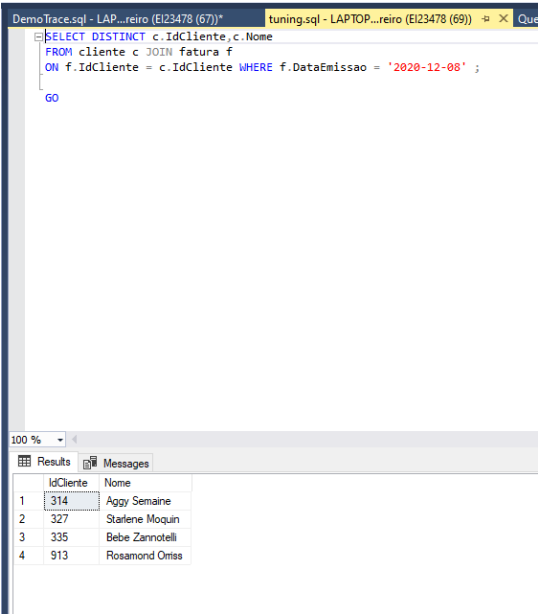


FIGURA 4.41: Voltar a executar o script tuning

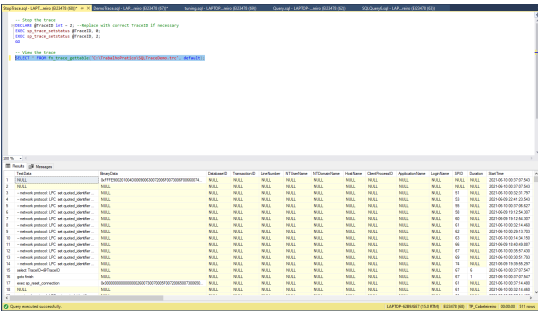


FIGURA 4.42: Executar script StopTrace

### 4.11.2 Database Engine tuning advisor

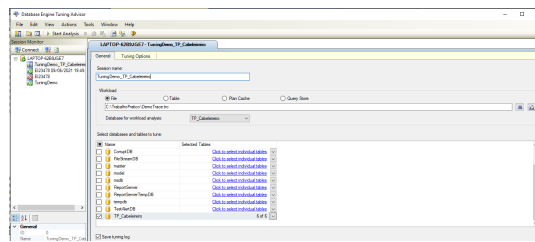


FIGURA 4.43: General

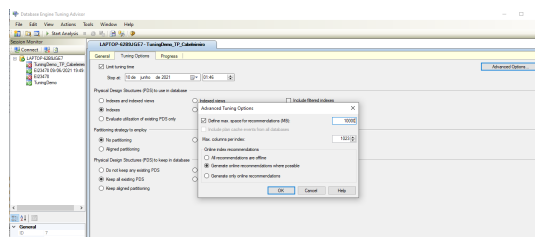


FIGURA 4.44: Options

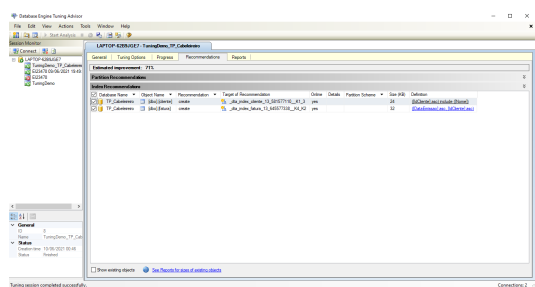


FIGURA 4.45: Recomendações

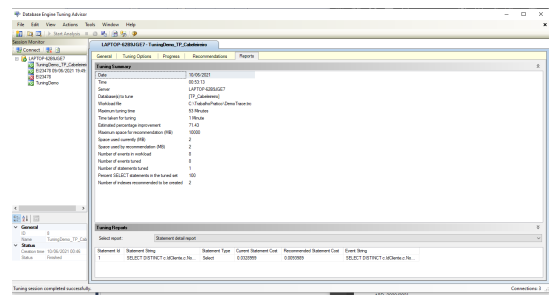


FIGURA 4.46: Report

Current Statement Cost	Recommended Statement Cost
0.0328959	0.0093989

FIGURA 4.47: Comparar

Current Statement Cost	Recommended Statement Cost
0.0328959	0.0093989

FIGURA 4.48: Comparar

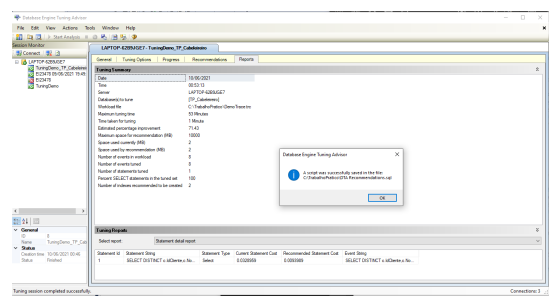


FIGURA 4.49: Guardado

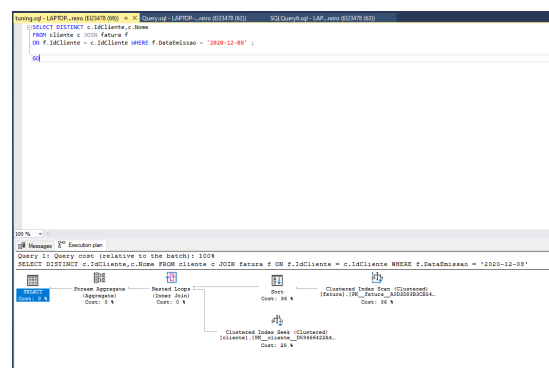


FIGURA 4.50: Display do tuning

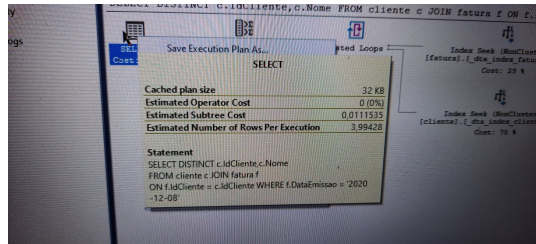


FIGURA 4.51: Estimated Subtree Cost antes da execução do DTA Recommendations

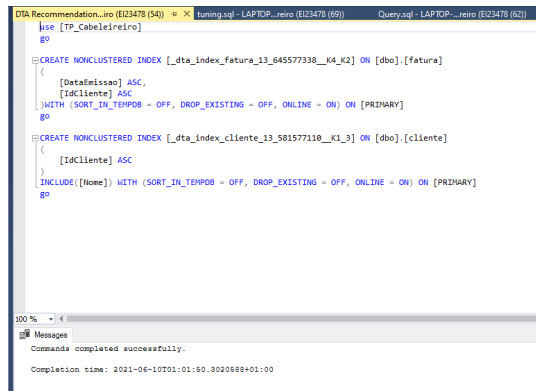


FIGURA 4.52: Execução do DTA Recommendations

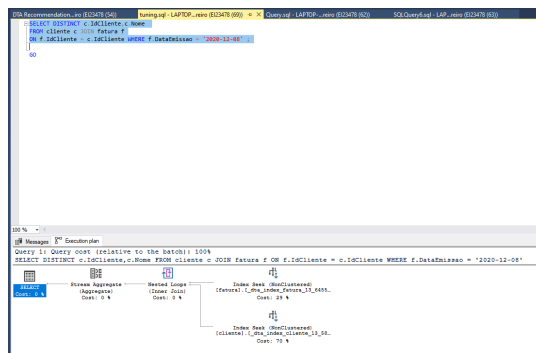


FIGURA 4.53: Depois da execução do DTA Recommendations

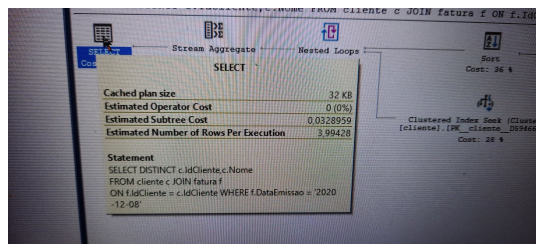


FIGURA 4.54: Estimated Subtree Cost depois da execução do DTA Recommendations



## 4.12 Importar base de dados para o azure

Infelizmente, não me foi possível importar a base de dados para o cloud azure, visto que enquanto resolvia o exercício apercebi-me que a collation da base de dados do azure e do sql não correspondiam. Tentei alterar a collation da base de dados do sql server para "SQL Latin1 General CP1 CI AS ", mas não foi possível, uma vez que quando criei a base de dados utilizei uma constraint na tabela fatura e por causa dessa constraint não era possível alterar a collation.

## 4.13 Database Maintenance

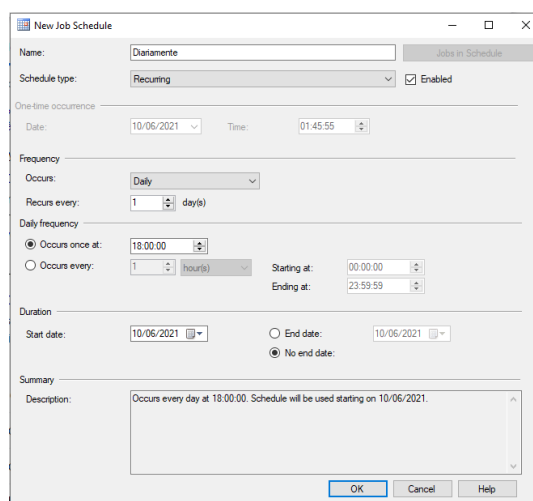


FIGURA 4.55: Criar um novo Job

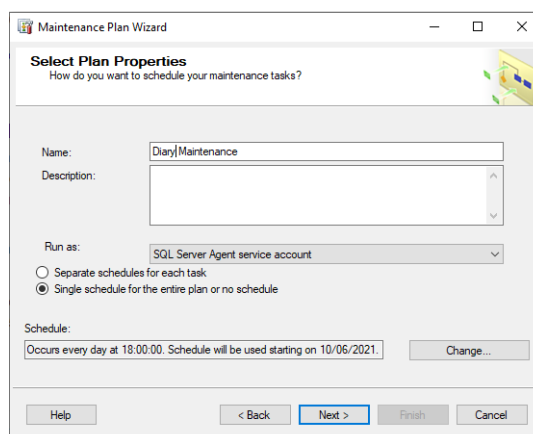


FIGURA 4.56: Escolher plano

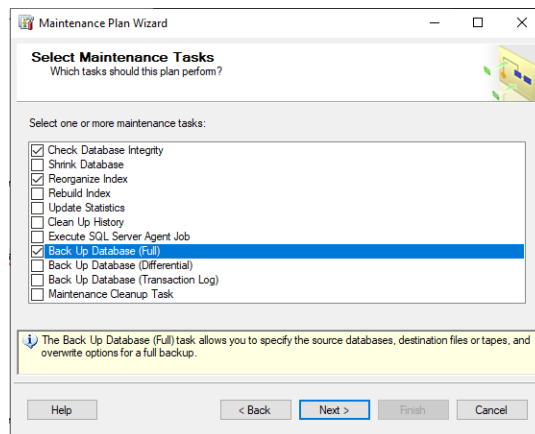


FIGURA 4.57: Escolher as tasks

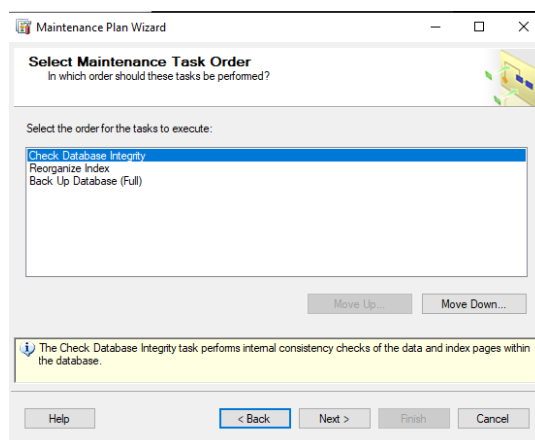


FIGURA 4.58: Tasks seleccionadas

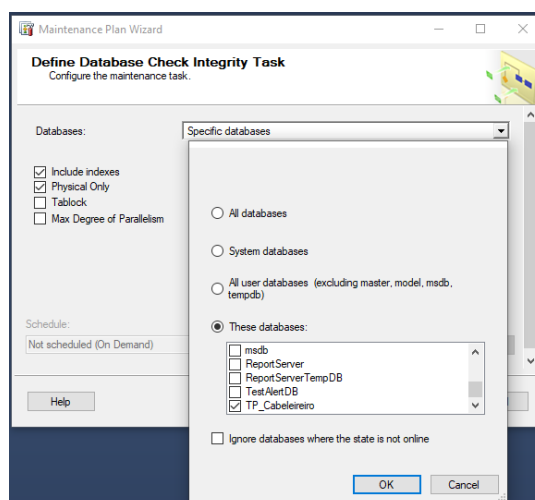


FIGURA 4.59: Escolher base de dados

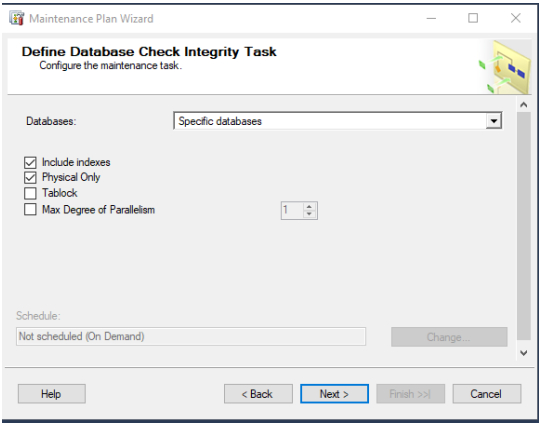


FIGURA 4.60: Integrity

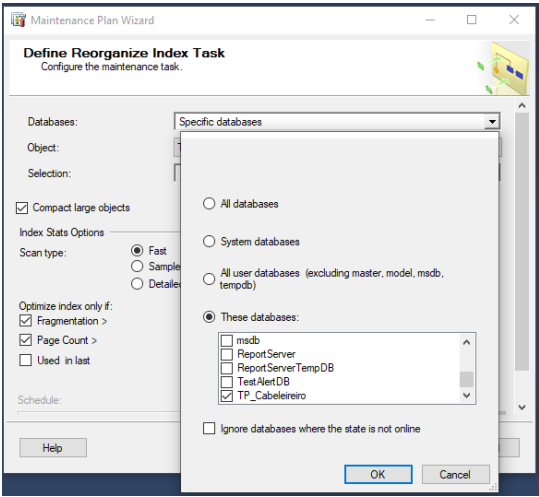


FIGURA 4.61: Escolher base de dados

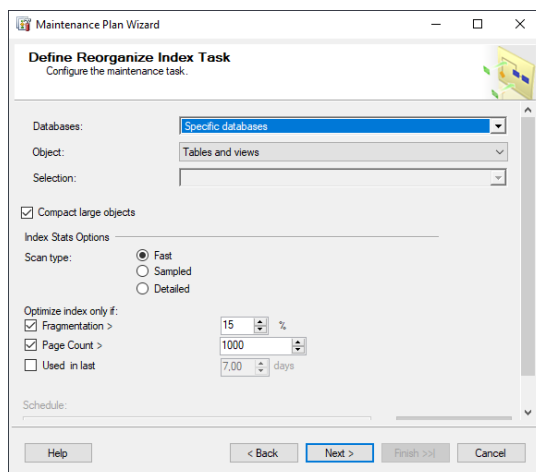


FIGURA 4.62: Index

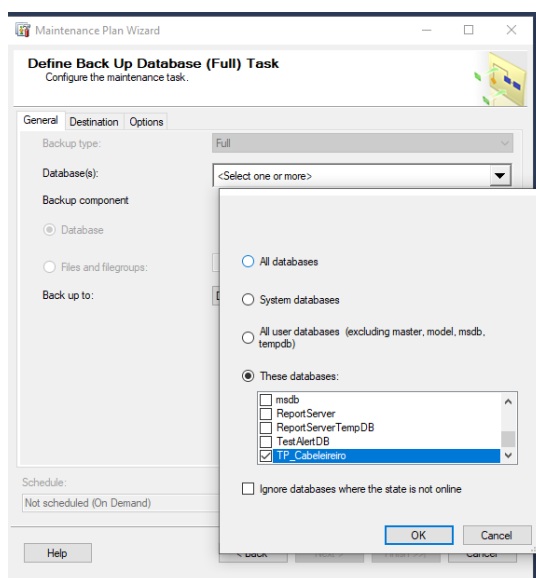


FIGURA 4.63: Escolher base de dados

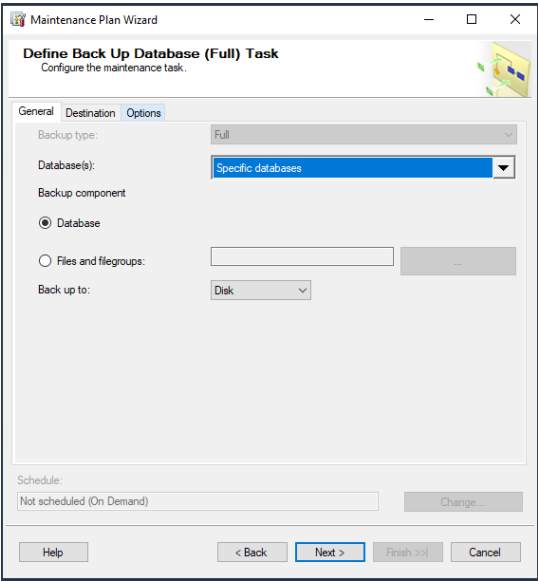


FIGURA 4.64: Backup Full

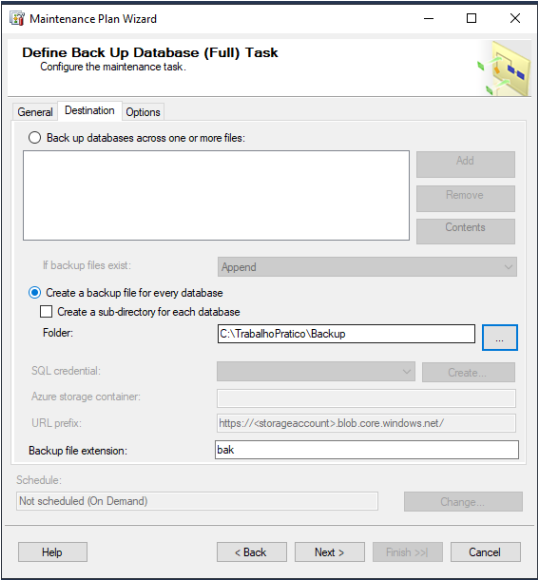


FIGURA 4.65: Backup Full-Destino

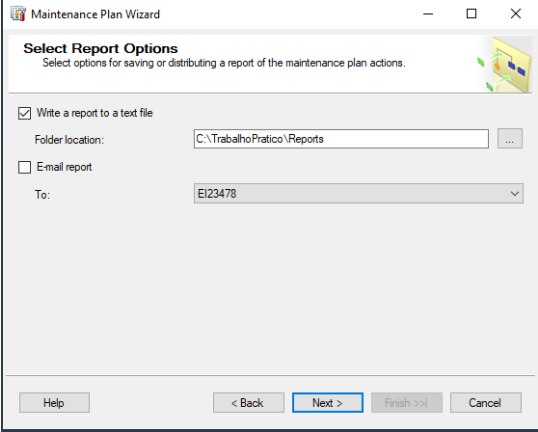


FIGURA 4.66: Report Options

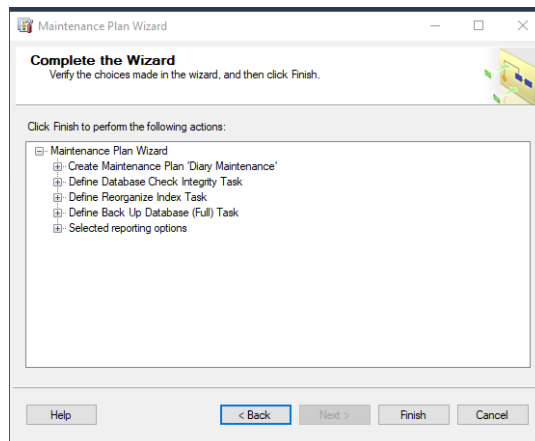


FIGURA 4.67: Completo

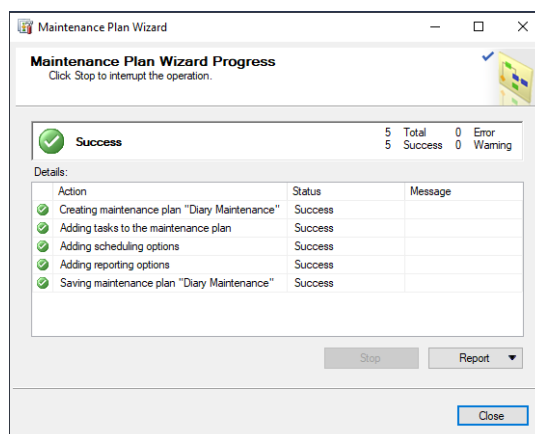


FIGURA 4.68: Executado com sucesso

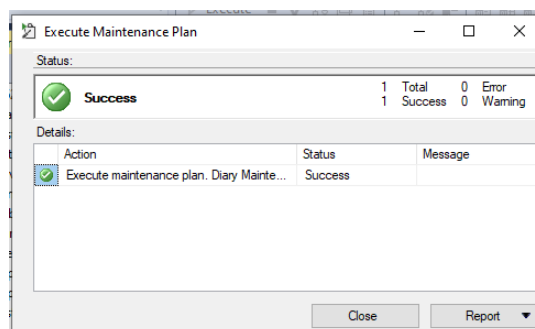


FIGURA 4.69: Diary executado com sucesso

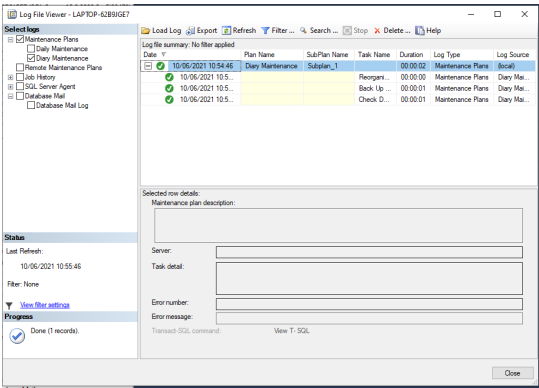


FIGURA 4.70: Ver histórico

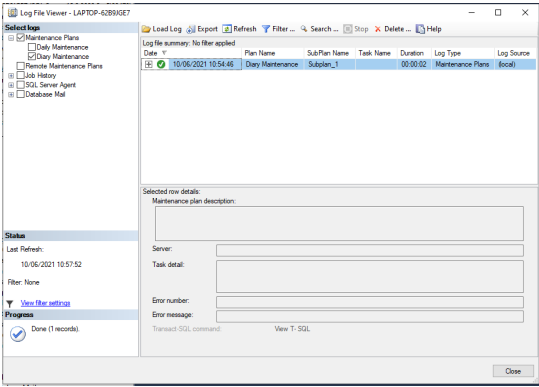


FIGURA 4.71: Refresh

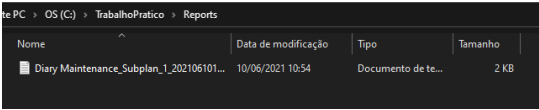


FIGURA 4.72: Diary Maintenance dentro da pasta Reports

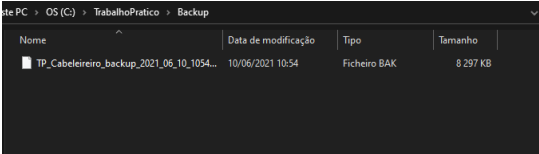


FIGURA 4.73: Base de dados na pasta Backup

## 4.14 SSRS

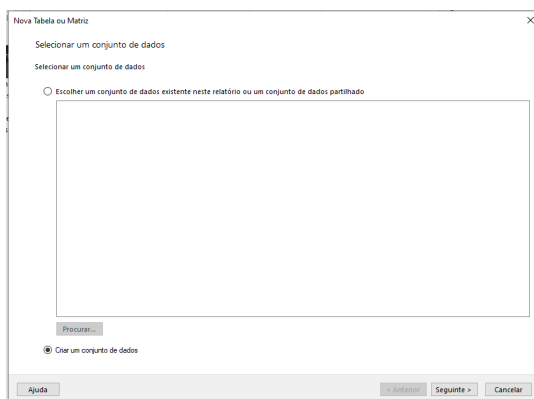


FIGURA 4.74: Criar report

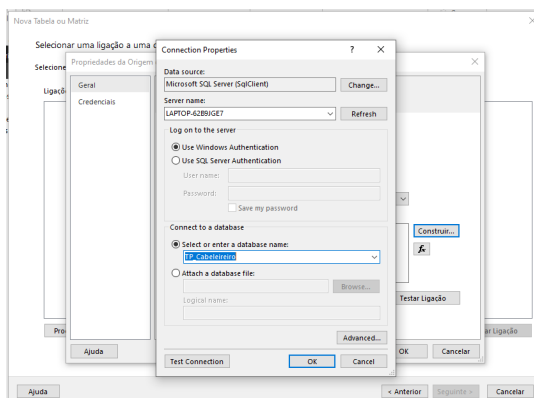


FIGURA 4.75: Conexão



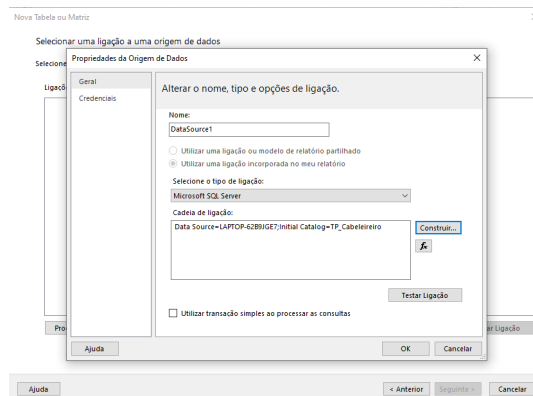


FIGURA 4.76: Propriedades das origens

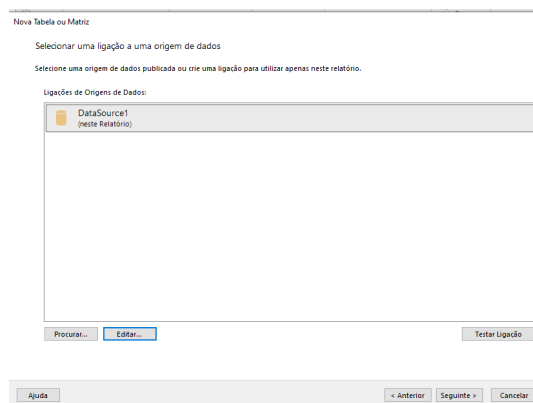


FIGURA 4.77: Ligação de origem de dados

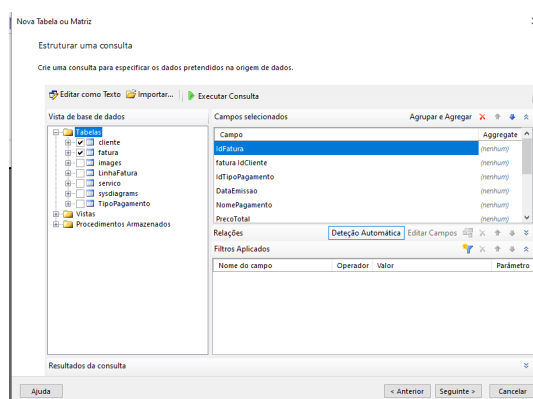


FIGURA 4.78: Escolher uma tabela

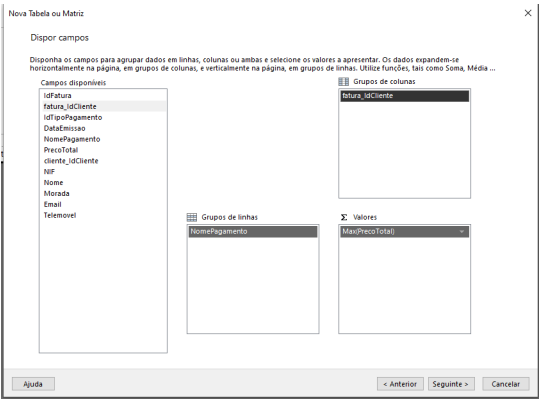


FIGURA 4.79: Dispor os campos

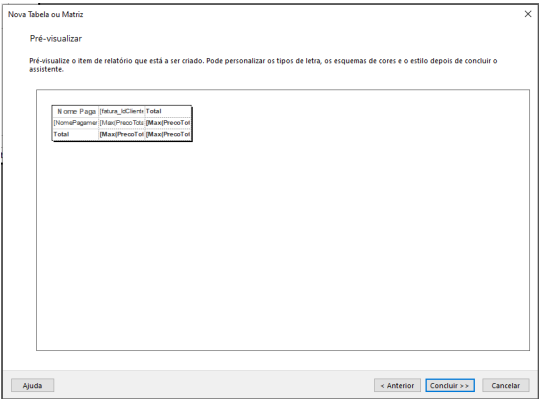


FIGURA 4.80: Pre-visualizar

TP_Report																													
Nome	2	3	5	6	7	10	15	17	18	20	21	27	29	30	31	32	34	37	38										
Pagamento																													
Ordenar																													
Mostrar																													
Numeração																													
Mostrar																													
Transferência																													
Total																													

FIGURA 4.81: Report

972	975	976	977	979	980	981	983	984	985	986	987	988	989	990	991	992	993	994	995	Total	
158		222	225	740		444	45	696	18					282		166	1227	170	183	730	1341
										1366							23				2442
																					1224
																					1366
																					1327
158		222	225	740		444	45	696	18	1366				282	166	1227		170	183	730	2442

FIGURA 4.82: Report

## Capítulo 5

### CONCLUSÕES

Com este trabalho consegui realizar maior parte dos objetivos, excetuando a alínea "n" e algumas partes das alíneas o e p.

Ao longo deste trabalho senti algumas dificuldades, que passarei a destacar de seguida. Logo no início tive algumas dificuldades ao fazer o cursor, uma vez que era a primeira vez que trabalhava com cursores e não estava a perceber muito bem como estes funcionavam. Além disso, também senti dificuldades ao fazer o trigger. Essas foram as que se mais destacaram, mas ao longo do trabalho fui tendo outras dificuldades que consegui resolver ao procurar na internet, ou nos powerpoints da cadeira ou até mesmo perguntando aos meus colegas.

Como disse, não consegui fazer a alínea n que era de importar a base de dados para o cloud azure, devido a incompatibilidade das collations das bases de dados do azure e do sql server. Tentei resolver o problema tentando alterar a collation da base de dados do sql server, mas não consegui já que na minha base de dados tinha uma constraint que dependia daquela collation. Já na alínea "o" não consegui fazer a parte de mandar o email e como o prazo do desconto mínimo estava a acabar, não consegui resolver o problema. Por fim na alínea p não consegui me conectar no report builder, coisa que já me aconteceu na tarefa 5, e por isso não consegui publicar o report.

O facto de eu estar a trabalhar sozinha e ter outros trabalhos para fazer fez com que eu não conseguisse entregar a tempo o trabalho.

Contudo, achei interessante a realização deste trabalho visto que permitiu que eu adquirisse novas competências e habilidades que certamente me irão ajudar futuramente.

#### 5.1 Referências

Mockaroo - Random Data Generator and API Mocking Tool | JSON / CSV / SQL / Excel. (2021). Retrieved June 10, 2021, from Mockaroo.com website: <https://www.mockaroo.com/>