

CONTEÚDO OCULTO

MIT em Engenharia de Banco de Dados
2021

Projeto Físico com SQL, PL/SQL & TSQL

CONTEÚDO OCULTO

Aluno
Felipe Wolff

Para a realização do Projeto, foi escolhido o [Microsoft SQL Server Management Studio 18](#).

Este documento está dividido em 3 (três) partes / módulos:

Ordem	Módulos	Conteúdo
1	MODELO FÍSICO	Database e Tabelas Trigger View PK / FK, Not Null, Unique e Default
2	CARGA DE DADOS	Volumetria Concorrência, Transação e Bloqueio Insert, Update e Delete
3	CONSULTA DE DADOS	5 Perguntas Count e Sum Min e Max Avg Group by e Having

1ª PARTE - MODELO FÍSICO

Base SQL Server

Foi criado o DataBase dbGun

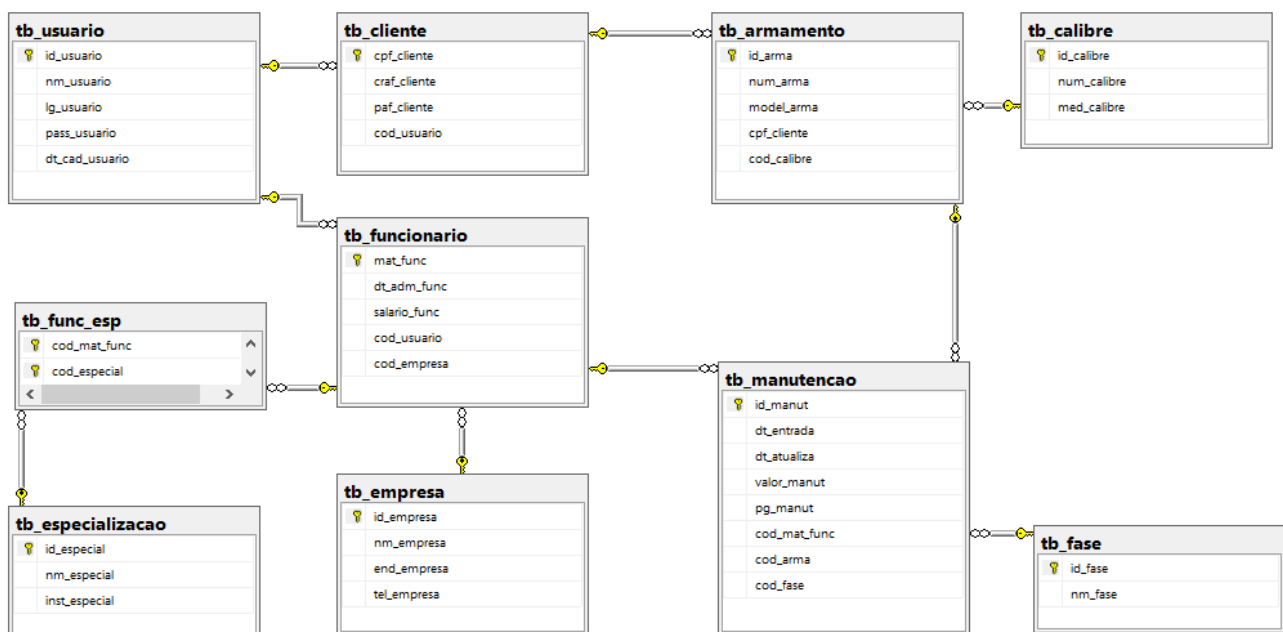
CREATE DATABASE [dbGun]

Banco de Dados	
Nome	dbGun
Status	Normal
Proprietário	DESKTOP-E4NQ1CD\Felipe e Priscila
Data de Criação	08/12/2021 15:37:06
Tamanho	464,00 MB
Espaço Disponível	19,09 MB
Número de Usuários	4
Memória Alocada a Objetos com Otimização	0,00 MB
Memória Usada por Objetos com Otimização	0,00 MB

Tabelas de acordo com o MER

Print da ferramenta de engenharia reversa do SSMS 18 (as queries serão apresentadas durante o trabalho)

dbo.tb_armamento
 dbo.tb_calibre
 dbo.tb_cliente
 dbo.tb_empresa
 dbo.tb_especializacao
 dbo.tb_fase
 dbo.tb_func_esp
 dbo.tb_funcionario
 dbo.tb_manutencao
 dbo.tb_usuario



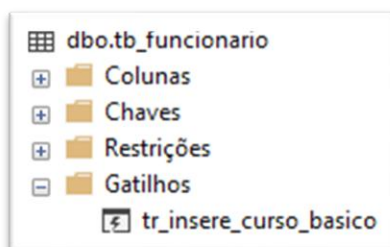
Trigger

Toda vez que for INSERIDO um registro na tabela de funcionário, APÓS a inserção o gatilho acionado irá cadastrar uma especialização para ele na tabela tb_func_esp

```
CREATE TRIGGER tr_insere_curso_basico ON dbo.tb_funcionario AFTER INSERT
AS
BEGIN
    DECLARE @matricula int

    --BUSCAR A MATRÍCULA QUE ACABOU DE SER ADICIONADA NA TB_FUNCIONARIO
    SELECT @matricula = INSERTED.mat_func FROM INSERTED

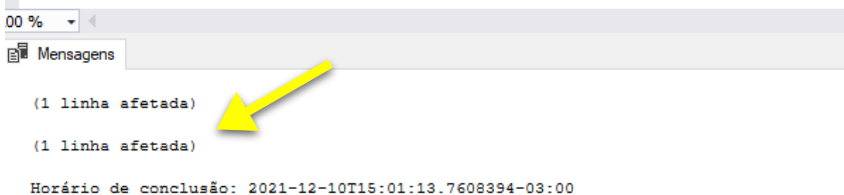
    --INSERE A NOVA MATRÍCULA + O CÓDIGO CORRESPONDENTE AO CURSO OBRIGATÓRIO (8)
    --NA TABELA TB_FUNC_ESP
    INSERT INTO dbo.tb_func_esp VALUES (@matricula, 8)
END
```



Trigger (exemplo)

1ª linha afetada: Insert
2ª linha afetada: Trigger

```
INSERT INTO tb_funcionario(dt_adm_func, salario_func, cod_usuario, cod_empresa)
VALUES ('2020-05-10', 3900, 9, 1)
```



View

Realiza joins para agrupar informações de um funcionário

```
CREATE VIEW vw_info_funcionarios
AS
SELECT
    f.mat_func AS [Matrícula]
    ,u.nm_usuario AS [Nome completo]
    ,u.lg_usuario AS [Login]
    ,f.salario_func AS [Salário]
    ,e.nm_empresa AS [Empresa]
FROM
    dbo.tb_usuario u
JOIN
    dbo.tb_funcionario f
ON
    u.id_usuario = f.cod_usuario
JOIN
    dbo.tb_empresa e
ON
    e.id_empresa = f.cod_empresa
```

View (exemplo)

Resultado de execução da View

```
SELECT * FROM vw_info_funcionarios
```

	Matrícula	Nome completo	Login	Salário	Empresa
1	1	Usuário Um	usuario_um	5000	Manutenção de Material Bélico LTDA
2	3	Novo Usuário	novo_usuario	4800	Da Selva Manutenções Limited
3	4	Salvador de Lá	salvador_de_la	6200	Manutenção de Material Bélico LTDA
4	5	Teste Trigger	teste_trigger	3900	Manutenção de Material Bélico LTDA

PK / FK

NOT NULL

UNIQUE

DEFAULT

Exemplos de duas tabelas do BD contendo os atributos citados



dbo.tb_manutencao	
Colunas	
id_manut	(PK, int, não nulo)
dt_entrada	(date, não nulo)
dt_atualiza	(date, nulo)
valor_manut	(float, nulo)
pg_manut	(bit, nulo)
cod_mat_func	(FK, int, não nulo)
cod_arma	(FK, int, não nulo)
cod_fase	(FK, int, não nulo)
Chaves	
PK_tb_manut	
FK_tb_mnt_arma	
FK_tb_mnt_fase	
FK_tb_mnt_func	
UO tb manut 7343D97222AE30B6	

```

IF OBJECT_ID('tb_manutencao') IS NOT NULL
    DROP TABLE dbo.tb_manutencao
BEGIN TRANSACTION
CREATE TABLE dbo.tb_manutencao
(
    id_manut          int          NOT NULL
    dt_entrada       date         NOT NULL,
    dt_atualiza      date         NULL,
    valor_manut      float        NULL,
    pg_manut         bit          NULL,
    cod_mat_func     int          NOT NULL,
    cod_arma         int          NOT NULL,
    cod_fase         int          NOT NULL,

    CONSTRAINT PK_tb_manut PRIMARY KEY CLUSTERED (id_manut),
    CONSTRAINT FK_tb_mnt_func FOREIGN KEY REFERENCES tb_funcionario(mat_func),
    CONSTRAINT FK_tb_mnt_arma FOREIGN KEY REFERENCES tb_armamento(id_arma),
    CONSTRAINT FK_tb_mnt_fase FOREIGN KEY REFERENCES tb_fase(id_fase)
)
COMMIT
    
```

dbo.tb_empresa	
Colunas	
id_empresa	(PK, int, não nulo)
nm_empresa	(varchar(50), não nulo)
end_empresa	(varchar(50), nulo)
tel_empresa	(varchar(15), nulo)
Chaves	
PK_tb_empresa	
UQ tb empre 4A0B7E2DC26BDA5F	

```

IF OBJECT_ID('tb_empresa') IS NOT NULL
    DROP TABLE dbo.tb_empresa
BEGIN TRANSACTION
CREATE TABLE dbo.tb_empresa
(
    id_empresa      int          NOT NULL
    nm_empresa     varchar(50)  NOT NULL,
    end_empresa     varchar(50)  NULL,
    tel_empresa     varchar(15) NULL,

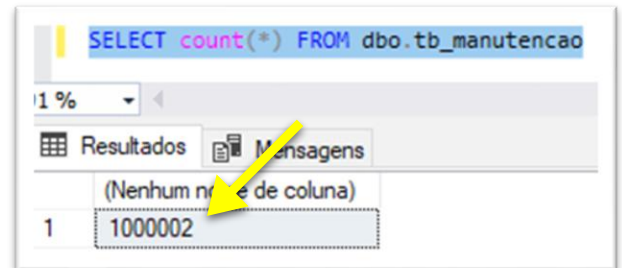
    CONSTRAINT PK_tb_empresa PRIMARY KEY CLUSTERED (id_empresa),
    CONSTRAINT UQ_tb_empresa UNIQUE (nm_empresa, end_empresa, tel_empresa)
)
COMMIT
    
```

2ª PARTE - CARGA DE DADOS

VOLUMETRIA

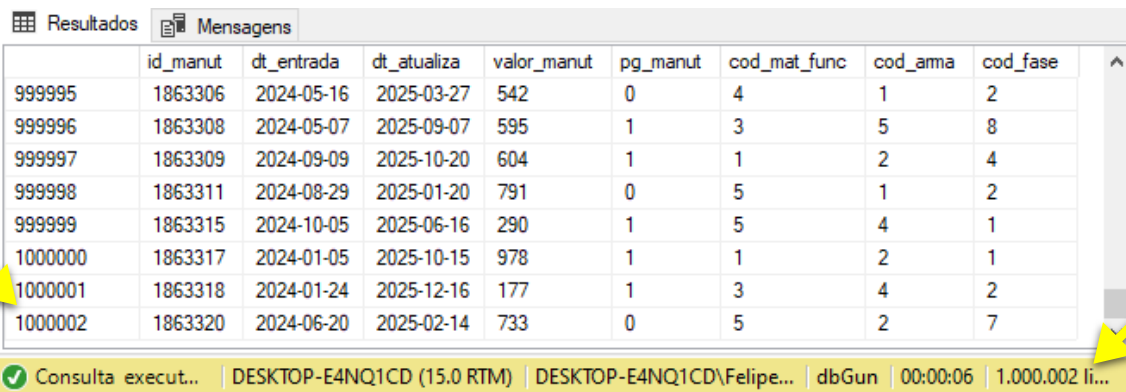
(+ 1 milhão de registros)

A tabela Manutenção (tb_manutencao) foi populada com 1.000.002 (um milhão e dois) registros por uma Stored Procedure que gerou números aleatórios e possíveis para cada coluna



```
SELECT count(*) FROM dbo.tb_manutencao
```

	1
	1000002



	id_manut	dt_entrada	dt_atualiza	valor_manut	pg_manut	cod_mat_func	cod_ama	cod_fase
999995	1863306	2024-05-16	2025-03-27	542	0	4	1	2
999996	1863308	2024-05-07	2025-09-07	595	1	3	5	8
999997	1863309	2024-09-09	2025-10-20	604	1	1	2	4
999998	1863311	2024-08-29	2025-01-20	791	0	5	1	2
999999	1863315	2024-10-05	2025-06-16	290	1	5	4	1
1000000	1863317	2024-01-05	2025-10-15	978	1	1	2	1
1000001	1863318	2024-01-24	2025-12-16	177	1	3	4	2
1000002	1863320	2024-06-20	2025-02-14	733	0	5	2	7

Consulta executada em: 00:00:06 | 1.000.002 li...

VOLUMETRIA (3 tabelas de cadastro)

Como exemplos de tabelas de cadastro, podemos verificar:

Usuário (tb_usuario)

Calibre (tb_calibre)

Empresa (tb_empresa)

	id_usuario	nm_usuario	lg_usuario	pass_usuario	dt_cad_usuario
1	1	Usuário Um	usuario_um	12341234	2021-12-08
2	2	Usuário Dois	usuario_dois	99998888	2021-12-08
3	3	Usuário Três	usuario_três	12345678	2021-12-08
4	6	Novo Usuário	novo_usuario	99887766	2021-12-20
5	7	Fulano de Tal	fulano_de_tal	22122021	2021-12-22
6	8	Salvador de Lá	salvador_de_la	33330011	2020-06-27
7	9	Teste Trigger	teste_trigger	78155355	2020-05-06

	id_calibre	num_calibre	med_calibre
1	1	9	mm
2	2	380	pol
3	3	357	pol
4	4	40	pol
5	5	50	pol
6	6	5,56	mm
7	7	7,62	mm
8	8	38	pol
9	9	22	pol

	id_empresa	nm_empresa	end_empresa	tel_empresa
1	1	Manutenção de Material Bélico LTDA	Av. Lorena Itajubá	(24)2356-8956
2	2	Da Selva Manutenções Limited	Rua Getúlio Maria	(55)9876-9876

CONCORRÊNCIA, TRANSAÇÃO E BLOQUEIO

Para evitar o travamento do sistema enquanto realiza-se uma transação em alguma tabela, vamos iniciar o código com o **BEGIN TRANSACTION** (que define o início de uma transação) e iremos terminar com **COMMIT**, caso quisermos validar os códigos executados, ou o **ROLLBACK** para voltar ao último estado.

Caso a transação não seja encerrada com um desses dois finalizadores, outro usuário ficará bloqueado na sessão, mostrando assim na prática a famosa **CONCORRÊNCIA** entre usuário e o **BLOQUEIO** ocorrendo.

Tendo isso exposto, seguimos com o exemplo para ver isso acontecendo.

INCLUSÃO

+ BEGIN TRANSACTION
+ COMMIT
= SUCESSO

```
BEGIN TRANSACTION
INSERT INTO [dbo].[tb_armamento]
([num_arma], [model_arma], [cpf_cliente], [cod_calibre])
VALUES
('92MA2022', 'Pistola G2C', '1234567891', 1)
COMMIT
```

	id_arma	num_arma	model_arma	cpf_cliente	cod_calibre
1	1	21RJ2010	Pistola G2C Tan	1234567891	1
2	2	55RS2017	Revólver Taurus	1234567891	8
3	3	51RS2020	Espingarda Mosquetão	1325648979	9
4	4	62GO2019	Imbel IA2	1568971621	6
5	5	62GO2019	Pistola Glock	1568971621	4
6	6	92MA2022	Pistola G2C	1234567891	1

ALTERAÇÃO

+ BEGIN TRANSACTION
+ COMMIT
= SUCESSO

```
BEGIN TRANSACTION
UPDATE [dbo].[tb_armamento]
SET [model_arma] = 'TESTE DE ATUALIZAÇÃO'
WHERE id_arma = 6
COMMIT
```

5	5	62GO2019	Pistola Glock	1568971621	4
6	6	92MA2022	TESTE DE ATUALIZAÇÃO	1234567891	1
7	7	92MA2023	Pistola G2C	1234567891	1

EXCLUSÃO

+ BEGIN TRANSACTION
+ COMMIT
= SUCESSO

Excluindo os dois
registros de teste

```
BEGIN TRANSACTION
DELETE tb_armamento
WHERE id_arma = 6 OR id_arma = 7
COMMIT
```

(2 linhas afetadas)

Caso a Transação fique em aberto (SEM Commit ou Rollback), teríamos o seguinte caminho:

1. TRANSAÇÃO

+ BEGIN TRANSACTION
- COMMIT ou ROLLBACK

```
BEGIN TRANSACTION
INSERT INTO [dbo].[tb_armamento]
([num_arma], [model_arma], [cpf_cliente], [cod_calibre])
VALUES
('92MA2023', 'Pistola G2C', '1234567891', 1)
```

2. CONSULTA

Outra sessão realiza
consulta

```
SELECT * FROM tb_armamento
```

3. LONGA ESPERA

Consulta não é executada,
para testar rodamos um
script para averiguar

```
SELECT *
FROM sys.dm_exec_requests
WHERE DB_NAME(database_id) = 'dbGun'
AND blocking_session_id <> 0
```

4. TRAVA

Confirmação do bloqueio

	session_id	request_id	start_time	status	command
1	57	0	2021-12-12 00:45:09.380	suspended	SELECT

5. VISUALIZAR

Então, para visualizar sem
o bloqueio em outra sessão
podemos usar a cláusula
WITH NOLOCK

```
SELECT * FROM tb_armamento WITH (NOLOCK)
```

5.1. ROLLBACK

Neste caso, executaremos
um Rollback para não
alterar nada no Banco

```
ROLLBACK
```

Obviamente, o correto para que isso não ocorra é o encerramento da transação utilizando as cláusulas: COMMIT ou ROLLBACK.

Por isso, WITH NOLOCK acaba não sendo uma boa prática, tendo em vista que poderemos ter resultados ainda não "comitados" (validados) pelo sistema.

3ª PARTE - CONSULTA DE DADOS

Perguntas* da primeira fase (Mini Mundo):

1. Um Funcionário tem alguma Especialização obrigatória?
2. Qual a estrutura de numeração de um Armamento?
3. Existem Usuários com senhas iguais?
4. Um Armamento pode ter mais de um Calibre?
5. Um Cliente pode ter mais de um Armamento?

PERG.	RESPOSTA	CONSULTA																																				
1	Sim. De acordo com a Trigger já vista, um funcionário deve ter o curso "8" obrigatoriamente cadastrado. Repare que todas as matrículas estão vinculadas a especialização "8".	<table><tr><th></th><th>cod_mat_func</th><th>cod_especial</th></tr><tr><td>1</td><td>1</td><td>5</td></tr><tr><td>2</td><td>1</td><td>7</td></tr><tr><td>3</td><td>1</td><td>8</td></tr><tr><td>4</td><td>3</td><td>3</td></tr><tr><td>5</td><td>3</td><td>8</td></tr><tr><td>6</td><td>4</td><td>8</td></tr><tr><td>7</td><td>5</td><td>8</td></tr></table>		cod_mat_func	cod_especial	1	1	5	2	1	7	3	1	8	4	3	3	5	3	8	6	4	8	7	5	8												
	cod_mat_func	cod_especial																																				
1	1	5																																				
2	1	7																																				
3	1	8																																				
4	3	3																																				
5	3	8																																				
6	4	8																																				
7	5	8																																				
2	De acordo com a consulta na tabela de armamento, verificamos a seguinte estrutura: 2 números + 2 letras + 4 números	<table><tr><th></th><th>id_ama</th><th>num_ama</th><th>model_ama</th><th>cpf_cliente</th><th>cod_calibre</th></tr><tr><td>1</td><td>1</td><td>21RJ2010</td><td>Pistola G2C Tan</td><td>1234567891</td><td>1</td></tr><tr><td>2</td><td>2</td><td>55RS2017</td><td>Revólver Taurus</td><td>1234567891</td><td>8</td></tr><tr><td>3</td><td>3</td><td>51RS2020</td><td>Espingarda Mosquetão</td><td>1325648979</td><td>9</td></tr><tr><td>4</td><td>4</td><td>62GO2019</td><td>Imbel IA2</td><td>1568971621</td><td>6</td></tr><tr><td>5</td><td>5</td><td>61DF2022</td><td>Pistola Glock</td><td>1568971621</td><td>4</td></tr></table>		id_ama	num_ama	model_ama	cpf_cliente	cod_calibre	1	1	21RJ2010	Pistola G2C Tan	1234567891	1	2	2	55RS2017	Revólver Taurus	1234567891	8	3	3	51RS2020	Espingarda Mosquetão	1325648979	9	4	4	62GO2019	Imbel IA2	1568971621	6	5	5	61DF2022	Pistola Glock	1568971621	4
	id_ama	num_ama	model_ama	cpf_cliente	cod_calibre																																	
1	1	21RJ2010	Pistola G2C Tan	1234567891	1																																	
2	2	55RS2017	Revólver Taurus	1234567891	8																																	
3	3	51RS2020	Espingarda Mosquetão	1325648979	9																																	
4	4	62GO2019	Imbel IA2	1568971621	6																																	
5	5	61DF2022	Pistola Glock	1568971621	4																																	
3	Não. Utilizando Count() e Group By, verificamos apenas uma senha para cada uma cadastrada. Obs.: Claro que o correto é o campo ser criptografado e não aberto.	<table><tr><th></th><th>Quantidade de senha</th><th>Senhas</th></tr><tr><td>1</td><td>1</td><td>12341234</td></tr><tr><td>2</td><td>1</td><td>12345678</td></tr><tr><td>3</td><td>1</td><td>22122021</td></tr><tr><td>4</td><td>1</td><td>33330011</td></tr><tr><td>5</td><td>1</td><td>78155355</td></tr><tr><td>6</td><td>1</td><td>99887766</td></tr><tr><td>7</td><td>1</td><td>99998888</td></tr></table>		Quantidade de senha	Senhas	1	1	12341234	2	1	12345678	3	1	22122021	4	1	33330011	5	1	78155355	6	1	99887766	7	1	99998888												
	Quantidade de senha	Senhas																																				
1	1	12341234																																				
2	1	12345678																																				
3	1	22122021																																				
4	1	33330011																																				
5	1	78155355																																				
6	1	99887766																																				
7	1	99998888																																				
4	Sim. Neste caso poderia ser um armamento com adaptador para calibres ou mesmo calibres de pontaria.	<table><tr><th></th><th>id_ama</th><th>num_ama</th><th>model_ama</th><th>num_calibre</th><th>med_calibre</th></tr><tr><td>1</td><td>8</td><td>99GE1915</td><td>Canhão Alemão</td><td>55</td><td>mm</td></tr><tr><td>2</td><td>9</td><td>99GE1915</td><td>Canhão Alemão</td><td>50</td><td>pol</td></tr></table>		id_ama	num_ama	model_ama	num_calibre	med_calibre	1	8	99GE1915	Canhão Alemão	55	mm	2	9	99GE1915	Canhão Alemão	50	pol																		
	id_ama	num_ama	model_ama	num_calibre	med_calibre																																	
1	8	99GE1915	Canhão Alemão	55	mm																																	
2	9	99GE1915	Canhão Alemão	50	pol																																	
5	Agrupando o resultado por CPF de cliente, verifica-se que sim. Neste caso, o cliente com o maior número de armamento é o de CPF 1325648979.	<table><tr><th></th><th>cpf_cliente</th><th>Quantidade de armamento</th></tr><tr><td>1</td><td>1234567891</td><td>2</td></tr><tr><td>2</td><td>1325648979</td><td>3</td></tr><tr><td>3</td><td>1568971621</td><td>2</td></tr></table>		cpf_cliente	Quantidade de armamento	1	1234567891	2	2	1325648979	3	3	1568971621	2																								
	cpf_cliente	Quantidade de armamento																																				
1	1234567891	2																																				
2	1325648979	3																																				
3	1568971621	2																																				

*As perguntas foram alteradas o mínimo possível para conseguirem ser respondidas por consultas / queries

CONCEITOS DE AGREGAÇÃO DE DADOS

Consultar o faturamento anual de todas as empresas

SUM()

Soma os orçamentos

GROUP BY

Agrupa por ano

HAVING

Seleciona serviços que já foram pagos

```
SELECT
    YEAR([dt_entrada]) AS [Ano de Entrada]
    ,FORMAT(SUM([valor_manut]), 'C', 'pt-br') AS [Faturamento]
FROM
    [dbo].[tb_manutencao]
GROUP BY
    YEAR([dt_entrada]), pg_manut
HAVING
    pg_manut = 1
ORDER BY
    YEAR([dt_entrada])
```

RESPOSTA

	Ano de Entrada	Faturamento
1	2020	R\$ 898.961,00
2	2021	R\$ 11.341.961,00
3	2022	R\$ 125.013.306,00
4	2023	R\$ 98.957.568,00
5	2024	R\$ 54.807.502,00

Vemos que 2022 foi o melhor ano em relação ao Faturamento e 2020 o pior

Verificar a média salarial dos funcionários

AVG()

Média dos salários

```
SELECT FORMAT(AVG(salario_func), 'C', 'pt-br') AS [Média salarial]
FROM tb_funcionario
```

RESPOSTA

	Média salarial
1	R\$ 4.975,00

Consulta para quantificar as manutenções que cada fase tem

COUNT()

Conta quantas manutenções foram cadastradas no sistema

GROUP BY

Agrupamento por fase de manutenção

```
SELECT
    count(m.cod_fase) AS [Quantidade de manutenções]
    , m.cod_fase AS [Código da fase]
    , f.nm_fase AS [Descrição da fase]
FROM tb_manutencao m
JOIN tb_fase f
ON m.cod_fase = f.id_fase
GROUP BY
    m.cod_fase, f.nm_fase
ORDER BY
    [Quantidade de manutenções] DESC
```

RESPOSTA

	Quantidade de manutenções	Código da fase	Descrição da fase
1	118116	2	Analisando
2	117878	4	Manutenção
3	117779	1	Aguardando análise
4	117696	7	Manutenção não realizada - Falta de peças
5	117660	3	Aprovação de orçamento
6	117430	8	Manutenção não realizada - Outros motivos
7	117398	5	Manutenção realizada - Aguardando retirada
8	117320	6	Manutenção não realizada - Orçamento
9	58725	9	Finalizada

Vemos que existe mais que o dobro de manutenções em 'Análise' do que 'Finalizadas'

Consulta para verificar o último cliente que se cadastrou no sistema (maior data)

MAX()

Estamos pegando a maior data de cadastro, ou seja, a última a ser cadastrada no sistema

```
SELECT
    c.cpf_cliente AS [CPF]
    , u.nm_usuario AS [Nome do cliente]
    , u.dt_cad_usuario AS [Data de cadastro]
FROM tb_usuario u
JOIN tb_cliente c
ON u.id_usuario = c.cod_usuario
WHERE u.dt_cad_usuario IN
    (SELECT MAX(dt_cad_usuario) FROM tb_usuario)
```

RESPOSTA

	CPF	Nome do cliente	Data de cadastro
1	1568971621	Fulano de Tal	2021-12-22

Consulta para ver qual foi o orçamento mais baixo de 2022

MIN()

Filtramos o ano de 2022 e selecionamos o valor mínimo da coluna de orçamento

```
SELECT MIN(valor_manut) AS [Orçamento mais baixo]
FROM [dbo].[tb_manutencao]
WHERE YEAR(dt_entrada) = 2022
```

RESPOSTA

	Orçamento mais baixo
1	200