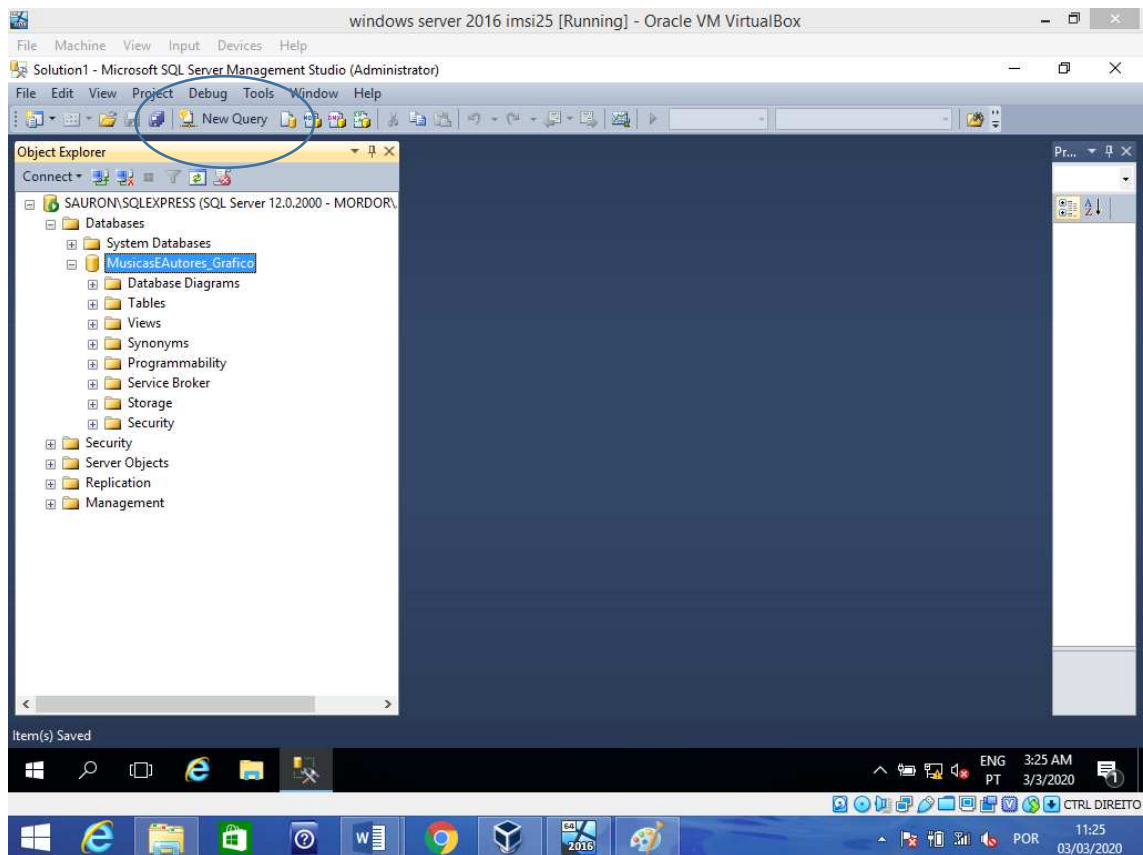




Queries SQL Server



Criar a base de dados:

CREATE DATABASE MusicasEAutores;

Para criar uma tabela, usa-se o comando CREATE TABLE referindo a base de dados e o schema,

Cada campo é identificado com tipo de dados, tamanho, se é nulo ou não, e se é chave primária:

CREATE TABLE MusicasEAutores.dbo.Musicas

(

musica varchar(100) PRIMARY KEY,

autor varchar(100) NOT NULL

);

Para inserir registos, têm que ser indicados base de dados, schema, tabela, campos (podem não ser todos):

INSERT INTO MusicasEAutores.dbo.Musicas (musica, autor)



```
VALUES ('Como o macaco gosta de banana', 'jose cid');  
INSERT INTO MusicasEAutores.dbo.Musicas(musica, autor)  
VALUES ('cabana', 'jose cid');  
INSERT INTO MusicasEAutores.dbo.Musicas(musica, autor)  
VALUES ('banana2', 'jose cid');
```

Podem ser inseridos vários registos de uma forma só, separando com uma vírgula:

```
INSERT INTO MusicasEAutores.dbo.Musicas(musica, autor)  
VALUES ('banana3', 'jose cid'),('a cabana2', 'jose cid'),('babalu', 'tonicha');
```

```
INSERT INTO MusicasEAutores.dbo.Musicas(musica, autor)  
VALUES('Marcha dos Marinheiros', 'tonicha');  
INSERT INTO MusicasEAutores.dbo.Musicas(musica, autor)  
VALUES('Boca de Amora', 'tonicha');  
INSERT INTO MusicasEAutores.dbo.Musicas(musica, autor)  
VALUES('deixa-me rir', 'jorge palma');
```

Seleccionar campos, filtrando registos:

```
SELECT autor,musica FROM MusicasEAutores.dbo.Musicas;  
SELECT * FROM MusicasEAutores.dbo.Musicas;
```

Onde autor for igual a 'jose cid':

```
SELECT autor FROM MusicasEAutores.dbo.Musicas  
WHERE autor='jose cid';
```

Onde autor diferente de 'jose cid':

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor!='jose cid';
```

Com expressões usa-se o LIKE em vez de =



Onde autor começa por 'j':

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE 'j%';
```

Onde autor acaba com 'a':

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE '%a';
```

Onde autor tem 6 caracteres e depois o 'a':

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE '_____a';
```

Onde autor tem 't', 1 caracter, 'n' e acaba com 'a':

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE 't_n%a';
```

Onde autor não tem a expressão: 't', 1 caracter, 'n' e acaba com 'a':

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor NOT LIKE 't_n%a';
```

Case sensitive (SQL Server):

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE 'T_n%a' COLLATE SQL_Latin1_General_CP1_CS_AS
```

(Em mysql select com case sensitive seria:

```
SELECT musica FROM MusicasEAutores.Musicas  
WHERE autor LIKE BINARY 'T_n%a')
```

Pode-se ordenar usando o comando ORDER BY:

```
SELECT autor,musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE '%'  
ORDER BY autor ASC,musica DESC;
```



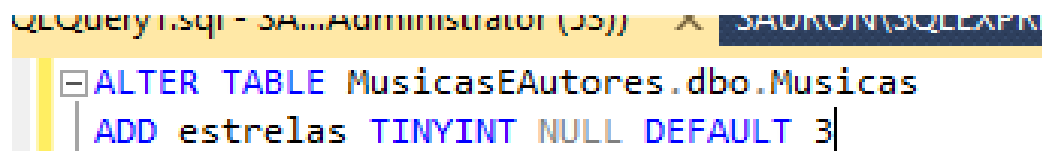
Pode-se fazer uma combinação lógica com AND e OR:

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE '%a' AND musica LIKE 'b%'  
ORDER BY autor ASC,musica DESC;
```

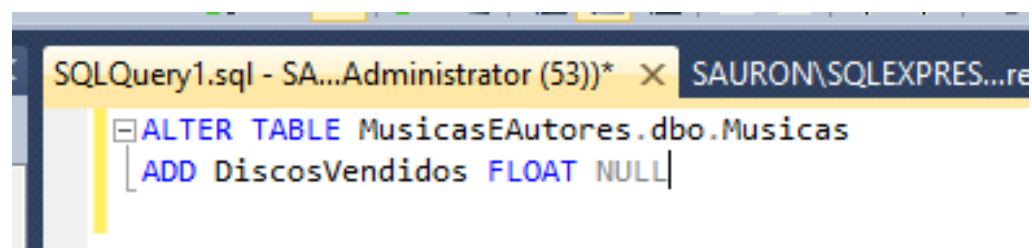
```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE '%a' OR musica LIKE 'b%'  
ORDER BY autor ASC,musica DESC;
```

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE '%a' OR musica LIKE '%2'  
ORDER BY autor ASC,musica DESC;
```

Para adicionar um campo à tabela, usa-se o ALTER TABLE:



```
SQLQuery1.sql - SA...Administrator (53)) X SAURON\SQLEXPRES...re  
ALTER TABLE MusicasEAutores.dbo.Musicas  
ADD estrelas TINYINT NULL DEFAULT 3
```



```
SQLQuery1.sql - SA...Administrator (53)) X SAURON\SQLEXPRES...re  
ALTER TABLE MusicasEAutores.dbo.Musicas  
ADD DiscosVendidos FLOAT NULL
```



SAURON\SQLEXPRES...res - dbo.Musicas SQLQuery1.sql - SA...Adminis

```
SELECT * FROM MusicasEAutores.dbo.Musicas
WHERE estrelas>2 AND estrelas<4
```

100 %

Results Messages

	musica	autor	DiscosVendidos	estrelas
1	a cabana2	jose cid	10000	5
2	banana2	jose cid	20000	5
3	banana3	jose cid	20000	5

SAURON\SQLEXPRES...res - dbo.Musicas SQLQuery1.sql - SA...Adm

```
SELECT * FROM MusicasEAutores.dbo.Musicas
WHERE estrelas>2 AND DiscosVendidos>20000
```

ows server 2016 imsi25 [Running] - Oracle VM VirtualBox

ORDOR\Administrator (53))* - Microsoft SQL Server Management Studio (Administrator)

Help

debug

SAURON\SQLEXPRES...res - dbo.Musicas SQLQuery1.sql - SA...Administrator (53))*

```
SELECT * FROM MusicasEAutores.dbo.Musicas
WHERE estrelas>2 AND DiscosVendidos>estrelas*2000
```

100 %

Results Messages

	musica	autor	DiscosVendidos	estrelas
1	banana2	jose cid	20000	5
2	banana3	jose cid	20000	5
3	Boca de Amora	tonicha	15000	4



Distinct permite não repetir valores:

`SELECT DISTINCT autor FROM MusicasEAutores.dbo.Musicas`

Existem várias funções como por exemplo o MAX,MIN, COUNT(*),LEN:

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor contains the following SQL query:

```
SELECT MAX(DiscosVendidos) FROM MusicasEAutores.dbo.Musicas
```

The query has been executed, and the results are displayed in the 'Results' pane. The results show a single row with the value 25000.

(No column name)
25000

The status bar at the bottom indicates that the query was executed successfully, returning 1 row.

The screenshot shows two SQL queries in the query editor:

```
SQLQuery4.sql - SA...Administrator (53))* X SQLQuery3.sql - SA...Administrator (52)
```

```
SELECT COUNT(*) FROM MusicasEAutores.dbo.Musicas;
```

```
SELECT autor,COUNT(*) FROM MusicasEAutores.dbo.Musicas  
GROUP BY autor
```

Também é possível o query usar subqueries mas os mesmos só podem retornar 1 valor:

The screenshot shows a SQL query in the query editor that uses a subquery:

```
SELECT * FROM MusicasEAutores.dbo.Musicas  
WHERE autor=  
(  
SELECT DISTINCT autor FROM MusicasEAutores.dbo.Musicas  
WHERE autor LIKE '%a'  
AND LEN(autor)=7  
)
```



Union permite juntar vários selects:

```
SELECT musica FROM MusicasEAutores.dbo.Musicas  
WHERE DiscosVendidos=20*(SELECT MIN(DiscosVendidos) FROM MusicasEAutores.dbo.Musicas)  
UNION  
SELECT musica FROM MusicasEAutores.dbo.Musicas
```

Para alterar registos, usa-se o UPDATE SET:

```
UPDATE MusicasEAutores.dbo.Musicas
```

```
SET autor = 'Grande jose'
```

```
WHERE autor='jose cid'
```

```
UPDATE MusicasEAutores.dbo.Musicas
```

```
SET autor = 'pequeno jose'
```

```
WHERE musica='banana2'
```

```
UPDATE MusicasEAutores.dbo.Musicas
```

```
SET autor = 'pequeno jose'
```

```
WHERE musica IS NULL
```

Vamos agora adicionar um campo ano inteiro:

```
ALTER TABLE MusicasEAutores.dbo.Musicas
```

```
ADD ano INT NULL
```

```
UPDATE MusicasEAutores.dbo.Musicas
```

```
SET ano = 1975
```

```
WHERE musica='babalu'
```

```
UPDATE MusicasEAutores.dbo.Musicas
```



SET ano = 1990

WHERE musica LIKE 'marcha%'

Para renomear uma tabela:

```
use MusicasEAutores;  
EXEC sp_rename 'Musicas', 'ObrasPrimas';
```

```
EXEC sp_rename 'ObrasPrimas', 'Musicas';
```

(em mysql, seria:

```
RENAME TABLE MusicasEAutores.dbo.Musicas TO  
MusicasEAutores.dbo.ObrasPrimas)
```

Vamos agora criar o campo anoeurovisao:

```
ALTER TABLE MusicasEAutores.dbo.Musicas
```

```
ADD anoeurovisao INT;
```

Para alterar o tipo de um campo:

```
ALTER TABLE MusicasEAutores.dbo.Musicas
```

```
ALTER COLUMN anoeurovisao DATETIME NULL
```

```
ALTER TABLE MusicasEAutores.dbo.Musicas
```

```
ALTER COLUMN anoeurovisao DATE NULL
```

Para renomear um campo, usa-se uma stored procedure:

```
EXEC sp_RENAME 'Musicas.anoeurovisao' , 'dataEurovisao' , 'COLUMN'
```

Em mysql seria:

```
ALTER TABLE MusicasEAutores.dbo.Musicas
```

```
CHANGE anoEurovisao
```

```
dataEurovisao DATE NULL DEFAULT NULL
```

JOIN permite fazer select de várias tabelas em simultâneo:

Vamos começar por criar uma segunda tabela:



```
CREATE TABLE MusicasEAutores.dbo.DadosAutores
```

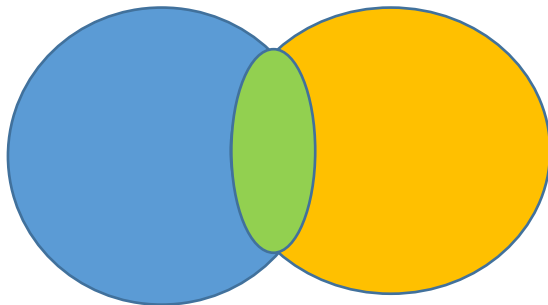
```
(  
  Autor varchar(100) PRIMARY KEY,  
  AnoNascimento varchar(5)  
)
```

E adicionar 2 registros:

```
INSERT INTO MusicasEAutores.dbo.DadosAutores(Autor, AnoNascimento)  
VALUES ('Grande Tonicha', '1946'), ('Jorge Palma', '1950'),('quim barreiros', '1947');  
  
INSERT INTO MusicasEAutores.dbo.DadosAutores(Autor, AnoNascimento)  
VALUES ('tonicha', '1946')
```

```
SELECT DB_NAME();
```

```
USE MusicasEAutores;  
SELECT Musicas.musica, Musicas.autor, Dadosautores.AnoNascimento  
FROM Musicas INNER JOIN Dadosautores  
ON Musicas.autor = Dadosautores.Autor
```



```
SELECT Musicas.musica, Musicas.autor, Dadosautores.AnoNascimento  
FROM Musicas LEFT JOIN Dadosautores  
ON Musicas.autor = Dadosautores.Autor
```

```
SELECT Musicas.musica, Musicas.autor, Dadosautores.AnoNascimento  
FROM Musicas RIGHT JOIN Dadosautores  
ON Musicas.autor = Dadosautores.Autor
```

FOREIGN KEY:

```
CREATE TABLE MusicasEAutores2.dbo.Concertos
```

```
(  
  ID int PRIMARY KEY,  
  Autor varchar(100) FOREIGN KEY REFERENCES Dadosautores(Autor),  
  Ano int
```



)

OU então dando um nome à CONSTRAINT (permite nomeadamente apagar):

```
CREATE TABLE MusicasEAutores2.dbo.Concertos
```

```
(
```

```
ID int PRIMARY KEY,
```

```
Autor varchar(100),
```

```
Ano int,
```

```
CONSTRAINT c1 FOREIGN KEY(Autor) REFERENCES Dadosautores(Autor)
```

```
)
```

Também seria possível adicionar a FOREIGN KEY posteriormente fazendo:

```
ALTER TABLE MusicasEAutores.dbo.Concertos
```

```
ADD FOREIGN KEY(Autor) REFERENCES Dadosautores(Autor)
```

Se executarmos:

```
INSERT INTO dbo.Concertos (ID,Autor,Ano)  
VALUES (1, 'Tonicha', 1997)
```

Dá erro porque 'Tonicha' já não existe na primeira tabela.

Mas se fizermos:

```
INSERT INTO dbo.Concertos (ID,Autor,Ano)  
VALUES (1, 'Grande Tonicha', 1997)
```

Já vai dar.

Se fizermos:

```
UPDATE MusicasEAutores.dbo. DadosAutores
```

```
SET autor = 'Tonicha'
```

```
WHERE autor = 'Grande Tonicha'
```

Não dá porque já existem registos na segunda tabela com o valor 'Grande Tonicha'

Com ON DELETE CASCADE e ON UPDATE CASCADE, alterações e exclusões são propagadas:

```
ALTER TABLE MusicasEAutores.dbo.Concertos
```

```
ADD FOREIGN KEY(Autor) REFERENCES Dadosautores(Autor)
```

```
ON DELETE CASCADE
```

```
ON UPDATE CASCADE
```



Volte a executar:

```
UPDATE MusicasEAutores.dbo. DadosAutores
```

```
SET autor = 'Tonicha'
```

```
WHERE autor = 'Grande Tonicha'
```

As opções são:

NO ACTION | CASCADE | SET NULL | SET DEFAULT

Por exemplo se fizermos:

Para apagar a constraint:

```
ALTER TABLE MusicasEAutores.dbo. Concertos
```

```
DROP CONSTRAINT c1;
```

E

```
ALTER TABLE MusicasEAutores.dbo. Concertos
```

```
ADD CONSTRAINT c2 FOREIGN KEY(Autor) REFERENCES Dadosautores(Autor)
```

```
ON DELETE NO ACTION
```

```
ON UPDATE NO ACTION
```