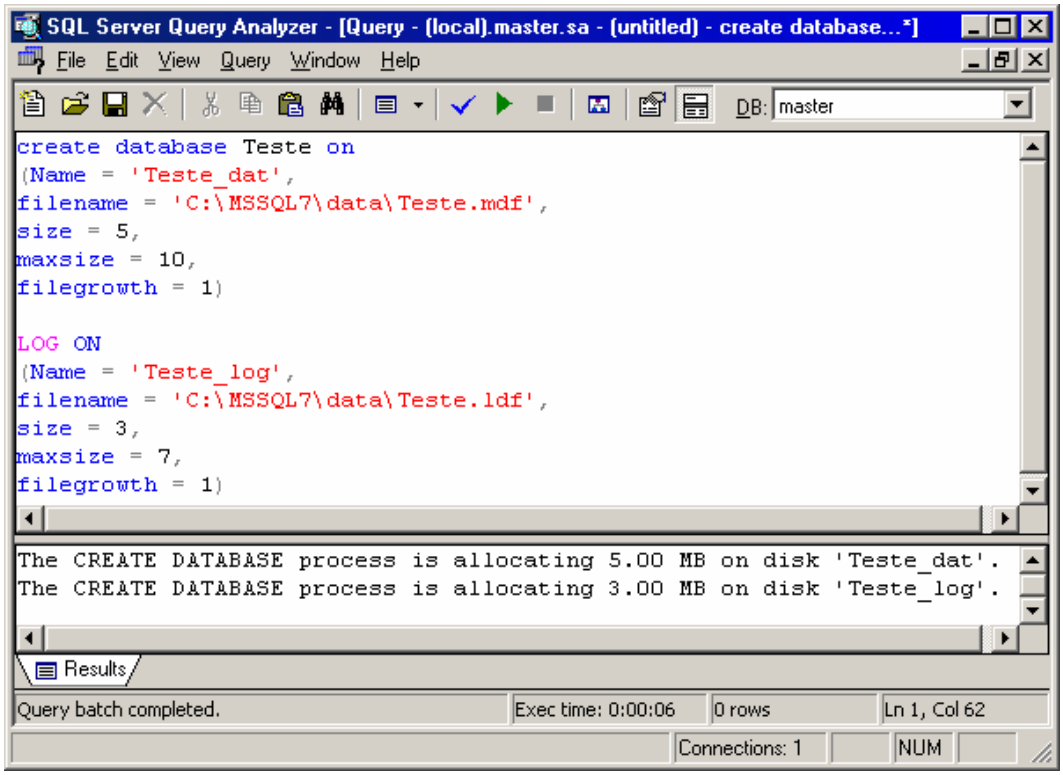
CRIANDO TABELAS E INSERINDO REGISTROS VIA SQL NO SQL Server 7.0

Obs: Estes comandos deverão ser realizados no **Query Analyzer**.

- Para criar um Banco de Dados no SQL Server via código SQL:



Depois de criado este banco de dados, selecione-o em **DB** do Query Analyzer para que os próximos comandos possam ser executados com sucesso.

1.0 CRIANDO UMA TABELA VIA SQL:

- ***Sintaxe para se criar uma tabela via SQL:***

Create Table NOME_TABELA **(**NOME_CAMPO1 [*Tipo de Dado*] [*Tamanho* ( )] [*Se é requerido ou não*] [*Índice*] **,** NOME_CAMPO2 [*Tipo de Dado*] [*Tamanho*] [*Se é requerido ou não*] [*Índice*].....**)**

- ***Exemplo de criação de uma tabela simples:***

Nome da Tabela: **CLIENTES**

Nome do Campo	Tipo de Dado
CODIGO	Inteiro
NOME	Texto

Comando a ser utilizado:

```
Create Table CLIENTES
(
 CODIGO int,
 NOME varchar(40)
)
```

Obs: Com este comando cria-se uma tabela com o nome Clientes e seus respectivos campos: Codigo e Nome. Note que para o campo é determinado um certo tipo de dado, como o **varchar** permite que o usuário defina a quantidade de caracteres que esse campo irá suportar, esta quantidade é definida em um valor numérico especificado entre parênteses. Para melhor compreender, visualize os Tipos de Dados mostrados a seguir:

➢ 1.1 TIPOS DE DADOS MAIS COMUNS UTILIZADOS EM SQL:

binary --> armazena informações binárias em pares de dois bytes.
tinyint --> corresponde ao tipo **byte** do MS Access e suporta inteiros entre 0 e 255.
smallint --> valor inteiro compreendido entre -32,768 e 32,767.
float --> armazena informações numéricas aproximadas com precisão de 1 a 38 casas decimais.
real --> armazena informações numéricas aproximadas com precisão de 7 casas decimais.
Obs. Os tipos float e real devem ser evitados quando a precisão é importante.
int --> valor inteiro compreendido entre -2.147.483.648 e 2.147.483.647.
decimal e numeric → armazenam informações numéricas exatas. Ex. decimal(7,2) - sete casas no total, sendo 2 decimais.
smalldatetime --> data compreendida entre 01/01/1900 e 06/06/2079 com precisão de horário de 1 minuto.
datetime --> data compreendida entre 01/01/1753 e 31/12/9999 com precisão de horário de 3,33 milissegundos.
Char(n) --> tamanho fixo**,** suporta até 8000 caracteres
nchar --> suporta até 4000 caracteres
varchar(n) --> tamanho variável**,** suporta até 8000 caracteres
nvarchar --> suporta até 4000 caracteres
text --> suporta até 2.147.483.647 caracteres (2GB)
money , smallmoney --> para armazenamento de valores monetários
timestamp --> valores de autoincrementação
image --> armazenamento de imagens (até 2 GB)
bit --> Dados para o tipo Sim/Não. Permite dois valores: 0 ou 1.

O trecho seguinte foi retirado do Help do SQL e fornece maiores detalhes sobre os tipos de dados.

Integers

bit
Integer data with either a 1 or 0 value.

int
Integer (whole number) data from -2^31 (-2,147,483,648) through 2^31 - 1 (2,147,483,647).

smallint
Integer data from 2^15 (-32,768) through 2^15 - 1 (32,767).

tinyint
Integer data from 0 through 255.

decimal and numeric

decimal
Fixed precision and scale numeric data from -10^38 -1 through 10^38 -1.

numeric
A synonym for **decimal**.

money and smallmoney

money
Monetary data values from -2^63 (-922,337,203,685,477.5808) through 2^63 - 1 (+922,337,203,685,477.5807), with accuracy to a ten-thousandth of a monetary unit.

smallmoney
Monetary data values from -214,748.3648 through +214,748.3647, with accuracy to a ten-thousandth of a monetary unit.

Approximate Numerics

float
Floating precision number data from -1.79E + 308 through 1.79E + 308.

real
Floating precision number data from -3.40E + 38 through 3.40E + 38.

datetime and smalldatetime

datetime
Date and time data from January 1, 1753, to December 31, 9999, with an accuracy of three-hundredths of a second, or 3.33 milliseconds.

smalldatetime
Date and time data from January 1, 1900, through June 6, 2079, with an accuracy of one minute.

Numerics

cursor
A reference to a cursor.

timestamp
A database-wide unique number.

uniqueidentifier
A globally unique identifier (GUID).

Character Strings

char
Fixed-length non-Unicode character data with a maximum length of 8,000 characters.

varchar
Variable-length non-Unicode data with a maximum of 8,000 characters.

text
Variable-length non-Unicode data with a maximum length of 2^31 - 1 (2,147,483,647) characters.

Unicode Character Strings

nchar
Fixed-length Unicode data with a maximum length of 4,000 characters.

nvarchar
Variable-length Unicode data with a maximum length of 4,000 characters. **sysname** is a system-supplied user-defined data type that is a synonym for **nvarchar(128)** and is used to reference database object names.

ntext
Variable-length Unicode data with a maximum length of 2^30 - 1 (1,073,741,823) characters.

Binary Strings

binary
Fixed-length binary data with a maximum length of 8,000 bytes.

varbinary
Variable-length binary data with a maximum length of 8,000 bytes.

image
Variable-length binary data with a maximum length of 2^31 - 1 (2,147,483,647) bytes.

EXEMPLOS COM OS COMANDOS MAIS UTILIZADOS:

Create Table CLIENTES (CODIGO int , NOME nvarchar(40)) ➔ Cria uma tabela com o nome CLIENTES com os seguintes campos: CODIGO do tipo **int** (Inteirio) e NOME do tipo **nvarchar** (Texto) limitado a 40 caracteres.

➢ 1.2 DEFINIÇÕES DE ÍNDICE:

- Criando uma chave primária:

Create Table CLIENTES **(** CODIGO **int CONSTRAINT** IdxCodigo **PRIMARY KEY,** NOME **varchar (**40**))** ➔ Para se criar uma chave primária utiliza-se o comando CONSTRAINT seguido de um nome qualquer (Nome do Índice), tipo de índice (No caso Chave Primária).

Obs: Com este comando deixaremos este campo indexado, ou seja, significa que o índice representa uma chave primária não podendo então haver duplicação de nenhum registro do campo CODIGO.
Para criarmos um índice para o campo **NOME** de tal forma que seja possível à duplicação de um ou mais registros devemos primeiramente criar a tabela com o comando abaixo sem colocarmos uma chave primária:

Create Table CLIENTES **(** CODIGO **int ,** NOME **varchar (**40**))**

E em seguida definiremos o índice com o seguinte comando:

Create Index IdxNome **On** CLIENTES **(**NOME**)**

Agora este campo estará indexado, mas não apresenta mais o símbolo de chave primária.
Podemos utilizar este comando também para criarmos índices para os demais campos da tabela como, por exemplo, o campo NOME.

➢ 1.3 DETERMINANDO SE UM CAMPO PERMITE OU NÃO VALORES NULOS:

Se na hora de criarmos a tabela não informarmos se um determinado campo permitirá valores nulos, este terá sua propriedade **Allow Nulls** checada permitindo valores nulos, caso contrário, devemos utilizar o comando **Not Null** logo após a definição do campo:

Create Table CLIENTES **(** CODIGO **int ,** NOME **varchar (**40**) NOT NULL)**

➢ 1.4 EXEMPLO DE CRIAÇÃO DE TABELA:

Veja um exemplo de criação de Tabela: **FUNCIONARIOS**

Nome do Campo	Tipo de Dado	Tamanho	Permite Nulo	Indexado
FUCODIGO	Inteiro (int)	-	Não	Sim (Duplicação não Autorizada)
FUNOME	Texto (varchar)	40	Não	Sim (Duplicação Autorizada)
FUDATANASC	Data/Hora (datetime)	-	Sim	Não
FUIDADE	Inteiro (int)	-	Não	Não
FUENDERECO	Texto (varchar)	50	Sim	Não
FUBAIRRO	Texto (varchar)	25	Sim	Não
FUSALARIO	Moeda (money)	-	Não	Não
FUCASAPROPRIA	Sim/Não (bit)	-	Não	Não
FUOBS	Texto Memorando (text)	-	Sim	Não

1º - ***Criando a tabela e seus respectivos campos e chaves primárias:***

Create Table FUNCIONARIOS **(** FUCODIGO **int CONSTRAINT** IdxFuCodigo **PRIMARY KEY,** FUNOME **varchar (**40**) Not Null,** FUDATANASC **DateTime,** FUIDADE **int Not Null,**

```
FUENDERECO varchar (50), FUBAIRRO varchar (25), FUSALARIO money Not Null,
FUCASAPROPRIA bit Not Null, FUOBS text)
```

2º - *Criando índice para o campo FUNOME:*

```
Create Index IdxFuNome On FUNCIONARIOS (FUNOME)
```

➢ 1.5 ALTERANDO A ESTRUTURA DE UMA TABELA:

Acrescentando um novo campo à tabela:
Vamos acrescentar o campo FUCEP do tipo Texto (nvarchar) e com Tamanho = 9 na tabela FUNCIONARIOS anteriormente criada. Para isso utilizaremos o comando **Alter Table**:

Alter Table FUNCIONARIOS **Add** FUCEP **char (9)** → Adiciona um novo campo.
Alter Table FUNCIONARIOS **Alter Column** FUBAIRRO **varchar(50)** → Altera um campo.
Alter Table FUNCIONARIOS **Drop Column** FUIDADE → Exclui o campo especificado da tabela.

➢ 1.6 EXCLUINDO UMA TABELA:

Para se excluir uma determinada tabela, utilizamos o comando **Drop Table** seguido do nome da tabela que desejamos excluir, como o seguinte exemplo:

Drop Table FUNCIONARIOS → Exclui a tabela inteira

➢ 1.7 CRIAÇÃO DE RELACIONAMENTOS ENTRE TABELAS:

Crie as seguintes tabelas:

Nome da Tabela: CLIENTES

Nome do Campo	Tipo de Dado	Tamanho	Nulo	Indexado
CL_CODIGO	Inteiro (int)	-	Não	Sim (Duplicação não Autorizada)
CL_NOME	Texto (varchar)	40	Não	Não

Nome da Tabela: COMPRAS

Nome do Campo	Tipo de Dado	Tamanho	Nulo	Indexado
CO_CODIGO	Auto Numeração(timestamp)	-	Não	Sim (Duplicação não Autorizada)
CL_CODIGO	Inteiro(int)	-	Não	Não
CO_DESCRICAO	Texto (varchar)	35	Sim	Não
CO_VALOR	Moeda (money)	-	Sim	Não

Código SQL para criar a Tabela CLIENTES:

```
Create Table CLIENTES ( CL_CODIGO int CONSTRAINT IdxCLCodigo PRIMARY KEY,
CL_NOME varchar (40) Not Null)
```

Depois de criado a tabela CLIENTES, já devemos inserir o código de relacionamento juntamente com o código de criação da tabela COMPRAS. O código utilizado para criar um relacionamento entre os campos é:
.....NOME_CAMPO1 [*Tipo de dado*] **CONSTRAINT** Nome **REFERENCES** NOME_TABELA **(**NOME_CAMPO2**)** → Este código só funciona juntamente com o código de criação da tabela seguindo sempre esta sintaxe, onde:

NOME_CAMPO1 --> Nome do campo que está sendo criado na nova tabela e que irá se relacionar com o de outra tabela. Lembrando que as definições de suas propriedades devem ser iguais para que o relacionamento funcione corretamente.
NOME --> Um nome qualquer que deve ser dado ao relacionamento a ser criado
NOME_TABELA --> Nome da tabela onde se encontra o campo a ser relacionado.
NOME_CAMPO2 --> Nome do campo a ser relacionado com o que está sendo criado nesta nova tabela.

Código SQL para criar a Tabela COMPRAS:

```
Create Table COMPRAS ( CO_CODIGO timestamp CONSTRAINT IdxCOCodigo PRIMARY KEY, CL_CODIGO int Not Null CONSTRAINT Relac_01 REFERENCES CLIENTES (CL_CODIGO), CO_DESCRICAO varchar(35), CO_VALOR money)
```

O relacionamento entre essas tabelas pode ser verificado tentando excluir a tabela clientes com o comando: **Drop Table CLIENTES**. Essa exclusão não será possível e o SQL Server retornará uma mensagem dizendo que a tabela CLIENTES está relacionada a uma outra tabela pela chave primária. Para que seja possível a exclusão dessa tabela é necessário que se exclua primeiramente a tabela COMPRAS e depois a tabela CLIENTES.

2.0 INSERINDO REGISTROS EM UMA TABELA VIA SQL:

➢ 2.1 INSERINDO REGISTROS NA TABELA:

Para inserir registros na tabela utilizamos o seguinte comando (**Insert Into**), onde primeiramente informamos os campos em que queremos adicionar um novo registro e logo em seguida os dados, que devem seguir a mesma ordem dos campos informados:

```
Insert Into FUNCIONARIOS ( FUCODIGO, FUNOME, FUDATANASC, FUENDERECO, FUBAIRRO, FUSALARIO, FUCASAPROPRIA, FUOBS )
Select '0001','José Manoel', '10/10/1950', 'Av. Beija Flor nº10', 'Centro', Convert(money,'180'), '0', 'Sem obs'
```

Nesse caso o conteúdo do CEP não foi fornecido, ou seja, ao inserir o registro, o valor do CEP assumirá Nulo.

Obs: Note que foi utilizada a função **Convert** para converter o dado do tipo texto (180) para um valor monetário. A função convert tem a seguintes sintaxe: **Convert (**<tipo_de_dado>**,** <expressão>**).**

Outra forma de Inserir registros:

Para não precisarmos definir todos os campos da tabela onde serão inseridos os dados, serão adicionados dados em todos os campos. Para isso, usamos o comando **Value** seguido das informações que queremos adicionar, **e que devem estar em ordem** de como os campos são apresentados na tabela. Veja o exemplo abaixo:

Ordem dos Campos: FUCODIGO, FUNOME, FUDATANASC, FUENDERECO, FUBAIRRO, FUSALARIO, FUCASAPROPRIA, FUOBS e FUCEP.

Insert Into FUNCIONARIOS **Values(** *0002* **,'***José Mauel***', '***10/10/1950***', '***Av. Beija Flor nº10***', '***Centro***', Convert(money,'***180***'), '***0***',** Null, '13330-000'**)**

Obs: Note que o primeiro dado a ser inserido na tabela é referente ao campo FUCODIGO correspondente ao código do funcionário, e que este não está entre apóstrofe, pois se refere a um valor do tipo numérico e não do tipo texto. Note também que para o campo FUOBS onde seriam digitadas as observações referentes ao funcionário foi utilizado o comando **Null** que serviu para deixar este campo em branco. Este comando só pode ser utilizado caso a Propriedade **Allow Nulls** (Permitido nulos) do campo seja igual à "Sim" (checada).

➢ 2.2 INSERINDO REGISTROS DE UMA OUTRA TABELA:

Podemos inserir registros de uma outra tabela desde que as propriedades dos campos de cada uma delas sejam iguais. Crie uma nova tabela de Funcionários igual a anterior modificando apenas o seu nome para **FUNCIONARIOS_2** e o índice primário para **IdxFuCodigo_2**.

Create Table FUNCIONARIOS_2 (FUCODIGO **int CONSTRAINT IdxFuCodigo_2 PRIMARY KEY**, FUNOME **varchar** (40) **Not Null**, FUDATANASC **DateTime**, FUENDERECO **varchar** (50), FUBAIRRO **varchar** (25), FUSALARIO **money Not** Null, FUCASAPROPRIA **bit Not Null**, FUOBS **text**, FUCEP **char**(9))

Insira alguns registros na tabela **FUNCIONARIOS** para que estes sejam copiados e inseridos na tabela **FUNCIONARIOS_2** com o seguinte comando:

Insert Into FUNCIONARIOS_2 **Select * From** FUNCIONARIOS ➔Este comando fará com que todos os registros da tabela FUNCIONARIOS sejam transferidos para a tabela FUNCIONARIOS_2. Verifique os registros inseridos na tabela FUNCIONARIOS_2 e note que são os mesmos registros da tabela FUNCIONARIOS.

Obs: Caso fosse necessário adicionar apenas os registros do campo FUNOME, por exemplo, seria utilizado o seguinte comando:
Insert Into FUNCIONARIOS_2 **(**FUNOME**) Select** FUNOME **From** FUNCIONARIOS
Este comando não funcionaria para estas tabelas, pois há outros campos além de FUNOME **que são requeridos** na hora de inserir um novo registro. Este comando só deve ser utilizado quando os outros campos não selecionados para adição de dados não sejam requeridos.

➢ 2.3 ALTERAÇÃO DE REGISTROS:

Para se alterar um determinado registro deve-se utilizar o comando **UPDATE** seguido do nome do campo a ser alterado e um critério qualquer para se definir qual registro deverá ou deverão ser alterados. Veja o exemplo:

Update FUNCIONARIOS **Set** FUNOME **=** **'***Zé Mané***' Where** FUCODIGO **= 0007** ➔Este comando faz com que o nome do funcionário, cujo código seja igual a 0007, passe a se chamar Zé Mané.

Obs: Se fosse necessário alterar todas as linhas do campo FUCEP para '13330-000', bastaria apenas utilizar o mesmo comando sem informar nenhum critério de alteração:

Update FUNCIONARIOS **Set** FUCEP = **'13330-000'** ➔ Com este comando, todas as linhas do campo FUCEP passariam a ter o mesmo conteúdo, as outras colunas continuariam com os mesmos dados.

➢ 2.4 EXCLUINDO REGISTROS DA TABELA:

Para excluirmos determinado registro de uma tabela utilizamos o comando **DELETE** seguido do nome da tabela e um critério qualquer para a exclusão, como o exemplo abaixo:

Delete From FUNCIONARIOS **Where** FUCODIGO = 4 ➔Este comando irá excluir o registro cujo código do funcionário (FUCODIGO) seja igual a 4.

Obs: Para excluirmos todos os registros da tabela FUNCIONARIOS, por exemplo, utilizamos o mesmo comando, porém sem nenhum critério de exclusão:
Delete From FUNCIONARIOS.

```
CREATE TABLE AUTONUMERA
(
 COD INT IDENTITY(1,1) ,
 NOME VARCHAR(50)
)

INSERT INTO AUTONUMERA VALUES('TESTE')
INSERT INTO AUTONUMERA VALUES('MAIS UM TESTE')

UPDATE TABELA1 SET TABELA1.CAMPO1 = TABELA2.CAMPO1
FROM TABELA2
WHERE TABELA1.ID = TABELA2.ID
```

Um Exemplo completo:

Serão criadas as seguintes tabelas com seus respectivos relacionamentos:

Nome da Tabela: ATORES

Nome do Campo	Tipo de Dado	Tamanho	Nulo	Indexado
AT_CODIGO	Inteiro (int)	-	Não	Sim (Duplicação não Autorizada)
AT_NOME	Texto (varchar)	50	Não	Não
AT_DATANASC	Data (datetime)		Sim	Não

Nome da Tabela: FILMES

Nome do Campo	Tipo de Dado	Tamanho	Nulo	Indexado
FI_NUMERO	Inteiro(int)	-	Não	Sim (Duplicação não Autorizada)
FI_TITULO	Texto(varchar)	50	Não	Não
FI_GENERO	Texto (varchar)	20	Não	Não
FI_DATAPROD	Data (datetime)		Sim	Não
FI_CUSTOPROD	Moeda (money)	-	Sim	Não

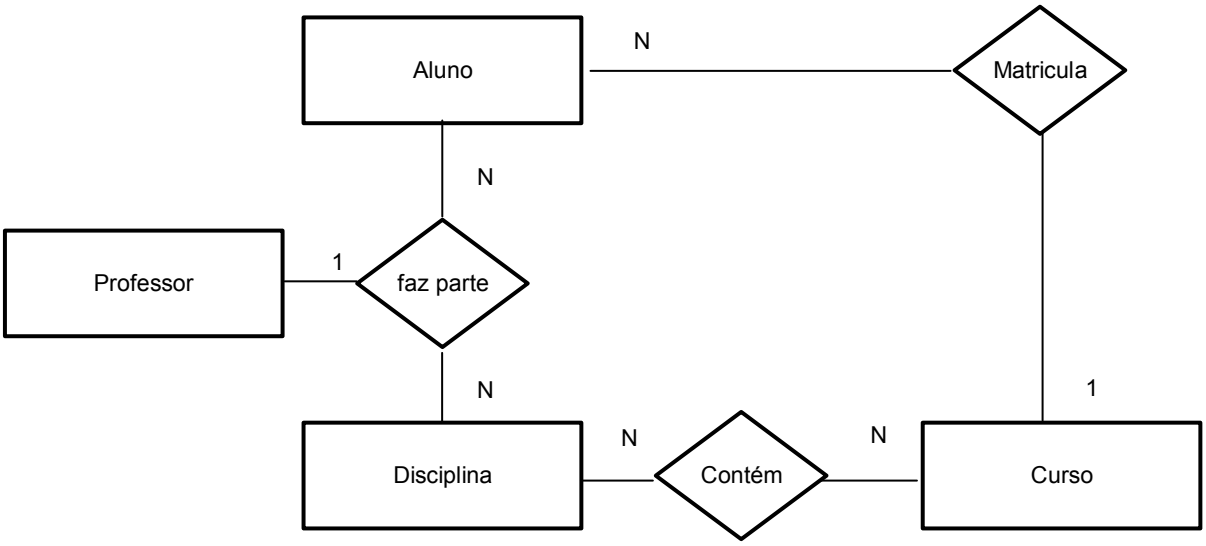
Nome da Tabela: ATORESFILMES

Nome do Campo	Tipo de Dado	Tamanho	Nulo	Indexado
AT_CODIGO	Inteiro (int)	-	Não	Sim (Duplicação não Autorizada)
FI_NUMERO	Inteiro (int)	-	Não	Sim (Duplicação não Autorizada)

Exercícios:

Uma universidade oferece vários cursos superiores para alunos que completaram o 2º grau. Para um aluno concluir um curso, deve cumprir todas as disciplinas contidas no curso. Para cumprir uma disciplina, o aluno precisa obter nota igual ou superior a 7.0. Toda disciplina é ministrada por único professor.

Foi criado o seguinte ME-R da especificação:



A partir do ME-R foi criado seguinte Modelo Relacional:
(os dados entre parênteses representam o relacionamento com a outra tabela no formato: nome da tabela.nome do campo)

Aluno = { NumAluno, Nome, Endereco, Cidade, Telefone, NumCurso(Curso.NumCurso) }
Disciplina = { NumDisp, Nome, QuantCreditos }
Professor = { NumFunc, Nome, Admissao, AreaPesquisa }
Curso = { NumCurso, Nome, TotalCréditos }
Aula = { NumAluno(Aluno.NumAluno), NumDisp(Disciplina.NumDisp), NumFunc(Professor.NumFunc), Semestre,Nota }
DisciplinaCurso = { NumDisp(Disciplina.NumDisp), NumCurso(Curso.NumCurso) }

Na base tem cadastrado os seguintes registros:

Professor			
NumFunc	Nome	Admissao	AreaPesquisa
45675	Abgair Simon Ferreira	10/04/1992	Banco de Dados
45690	Ramon Travanti	20/05/1993	Direito Romano
45691	Gustavo Golveia Netto	05/04/1993	Sociologia
45692	**Marcos Salvador**	**31/03/1993**	**Matemática Financeira**
45693	**Cintia Falcão**	**15/02/1993**	**Engenharia Software**

Curso		
NumCurso	Nome	TotalCrédito
2142	**Engenharia Civil**	**1500**
2143	Ciência da Computação	2000
2144	Direito	1750
2145	Pedagogia	1500
2146	Odontologia	1600

Disciplina		
NumDisp	Nome	QuantCreditos
1	Banco de Dados	30
2	Estrutura de Dados	30
3	Direito Penal	25
4	**Cálculo Numérico**	**30**
5	Psicologia Infantil	25
6	Direito Tributário	33
7	**Engenharia Software**	**27**

DisciplinaCurso	
NumDisp	NumCurso
1	2143
2	2143
3	2144
4	2143
4	**2142**
5	2145
6	2144
7	2143
7	**2142**

Aluno					
NumAluno	Nome	Endereco	Cidade	Telefone	NumCurso
111	Edvaldo Carlos Silva	Av.São Carlos, 186	São Carlos - SP	(017) 276-9999	2143
112	**João Benedito Scapin**	**R.José Bonifácio, 70**	**São Carlos - SP**	**(017) 273-8974**	**2142**
113	Carol Antonia Silveira	R.Luiz Camões, 120	Ibaté - SP	(017) 278-8568	2145
114	Marcos João Casanova	Av. São Carlos, 176	São Carlos - SP	(017) 274-9874	2143
115	Simone Cristina Lima	R.Raul Junior, 180	São Carlos - SP	(017) 273-9865	2144
116	Aílton Castro	R.Antonio Carlos, 120	Ibaté - SP	(017) 278-8568	2142
117	José Paulo Figueira	R. XV Novembro, 871	São Carlos - SP	(017) 274-9874	2145

Aula				
NumAluno	NumDisp	NumFunc	Semestre	Nota
111	1	45675	01/1998	8.5
111	2	45675	01/1998	6.0
111	2	45675	02/1998	7.0
115	3	45690	01/1998	4.5
115	3	45690	02/1998	7.5
111	4	45692	01/1998	8.0
112	**4**	**45692**	**01/1998**	**7.0**
113	5	45691	01/1998	7.5
115	6	45690	01/1998	9.0
111	7	45693	01/1998	10.0
112	**7**	**45693**	**01/1998**	**5.5**
112	**7**	**45693**	**02/1998**	**10.0**
114	1	45675	01/1998	7.0
114	2	45675	01/1998	8.0
114	4	45692	01/1998	6.5
114	4	45692	02/1998	8.5
116	4	45692	01/1998	3.5
116	4	45692	02/1998	9.5
114	7	45693	01/1998	9.5
116	7	45693	01/1998	8.5

1) Apresentar os Scripts necessários para criar a base de dados no SQL_Server, as tabelas com respectivas chaves, chaves estrangeiras e cadastrar os dados das tabelas.
2) Apresentar os comandos para as seguintes pesquisas:
a)Todos os nomes dos cursos da universidade.
b) Quais os nomes e telefones de alunos da cidade de São Carlos - SP em ordem DESC de nome.
c) Quais os nomes de professores que foram contratados antes que 01/jan/1993.
d) Quais os nomes de alunos que iniciam com a letra 'J'.
e) Quais os nomes das disciplinas do curso de Ciência da Computação.
f) Quais os nomes doS cursos que possuem no curriculum a disciplina Cálculo Numérico.
g) Quais os nomes das disciplinaS que o aluno Marcos João Casanova cursou no 1º semestre de 1998.
h) Quais os nomes de disciplinas que o aluno Ailton Castro foi reprovado.
i) Quais os nomes de alunos reprovados na disciplina de Cálculo Numérico no 1º semestre de 1998.
j) Quais os nomes das disciplinas ministradas pelo prof. Ramon Travanti.
k) Quais os nomes professores que já ministraram aula de Banco de Dados.
l) Qual a maior e a menor nota na disciplina de Cálculo Numérico no 1º semestre de 1998.
m) Qual o nome do aluno e nota que obteve maior nota na disciplina de Engenharia de Software no 1º semestre de 1998.
n) Quais nomes de alunos, nome de disciplina e nome de professores que cursaram o 1º semestre de 1998 em ordem de aluno.
o) Quais nomes de alunos, nome de disciplina e notas do 1º semestre de 1998 no curso de Ciência da Computação.
p) Qual a média de notas do professor Marcos Salvador.
q) Quais nomes de alunos, nomes de disciplinas e notas que tiveram nota entre 5.0 e 7.0 em ordem de disciplina.
r) Qual a média de notas da disciplina Cálculo Numérico no 1º semestre de 1998.
s) Quantos alunos o professor Abgair teve no 1º semestre de 1998.
t) Qual a média de notas do aluno Edvaldo Carlos Silva.
u) Quais as médias por nome de disciplina de todos os cursos do 1º semestre de 1998 em ordem de disciplina.
v) Quais as médias das notas por nome de professor no 1º semestre de 1998.

w) Qual a média por disciplina no 1º semestre de 1998 do curso do Ciência da Computação
x) Qual foi quantidade de créditos concluídos (somente as disciplinas aprovadas) do aluno Edvaldo Carlos Silva.
y) Quais nomes de alunos e quantidade de créditos que já completaram 70 créditos (somente os aprovados na disciplina).
z) Quais nomes de alunos, nome de disciplina e nome de professores que cursaram o 1º semestre de 1998 e pertencem ao curso de ciência da Computação que possuem nota superior à 8.0.

3) Comandos para as seguintes atualizações:
a) Mudar o nome da disciplina Banco de Dados para Banco de Dados Aplicado.
b) Zerar as notas de todos os alunos do 2º semestre de 1998.
c) Deixar todas as disciplinas com quantidade de créditos igual a 10.
d) Excluir todos os registros da tabela Aula.
e) Inserir um campo de observação na tabela Aula.