ESCOLA E FACULDADE DE TECNOLOGIA SENAI ROBERTO MANGE

DESENVOLVIMENTO DE SISTEMAS

JULHO DE 2022 — SENAI-SP

SENAI SÃO PAULO

SQL – STRUCTURED QUERY LANGUAGE

CREATE DATABASE

Criação de um novo banco de dados

Obs.: SQL não é case sensitive

```
create database senai;
```

```
create DATABASE SeNaI2;
```

DROP DATABASE

Deleta um banco de dados

```
drop database senai;
```

USE DATABASE

Inicia conexão com um banco de dados

```
use senai;
```

SENAI SÃO PAULO

SQL – STRUCTURED QUERY LANGUAGE

DATATYPES – STRING (TEXTOS)

CHAR(size)	A FIXED length string (can contain letters, numbers, and special characters). The *size* parameter specifies the column length in characters - can be from 0 to 255. Default is 1
VARCHAR(size)	A VARIABLE length string (can contain letters, numbers, and special characters). The *size* parameter specifies the maximum column length in characters - can be from 0 to 65535
BINARY(size)	Equal to CHAR(), but stores binary byte strings. The *size* parameter specifies the column length in bytes. Default is 1
VARBINARY(size)	Equal to VARCHAR(), but stores binary byte strings. The *size* parameter specifies the maximum column length in bytes.
TINYBLOB	For BLOBs (Binary Large Objects). Max length: 255 bytes
TINYTEXT	Holds a string with a maximum length of 255 characters
TEXT(size)	Holds a string with a maximum length of 65,535 bytes
BLOB(size)	For BLOBs (Binary Large Objects). Holds up to 65,535 bytes of data
MEDIUMTEXT	Holds a string with a maximum length of 16,777,215 characters
MEDIUMBLOB	For BLOBs (Binary Large Objects). Holds up to 16,777,215 bytes of data
LONGTEXT	Holds a string with a maximum length of 4,294,967,295 characters
LONGBLOB	For BLOBs (Binary Large Objects). Holds up to 4,294,967,295 bytes of data

SQL – STRUCTURED QUERY LANGUAGE

DATATYPES – STRING (TEXTOS)

ENUM(val1, val2, val3, ...)	A string object that can have only one value, chosen from a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted. The values are sorted in the order you enter them
SET(val1, val2, val3, ...)	A string object that can have 0 or more values, chosen from a list of possible values. You can list up to 64 values in a SET list

SQL – STRUCTURED QUERY LANGUAGE

DATATYPES – NUMERIC (NÚMEROS)

BIT(*size*)	A bit-value type. The number of bits per value is specified in *size*. The *size* parameter can hold a value from 1 to 64. The default value for *size* is 1.
TINYINT(*size*)	A very small integer. Signed range is from -128 to 127. Unsigned range is from 0 to 255. The *size* parameter specifies the maximum display width (which is 255)
BOOL	Zero is considered as false, nonzero values are considered as true.
BOOLEAN	Equal to BOOL
SMALLINT(*size*)	A small integer. Signed range is from -32768 to 32767. Unsigned range is from 0 to 65535. The *size* parameter specifies the maximum display width (which is 255)
MEDIUMINT(*size*)	A medium integer. Signed range is from -8388608 to 8388607. Unsigned range is from 0 to 16777215. The *size* parameter specifies the maximum display width (which is 255)
INT(*size*)	A medium integer. Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295. The *size* parameter specifies the maximum display width (which is 255)
INTEGER(*size*)	Equal to INT(size)
BIGINT(*size*)	A large integer. Signed range is from -9223372036854775808 to 9223372036854775807. Unsigned range is from 0 to 18446744073709551615. The *size* parameter specifies the maximum display width (which is 255)
FLOAT(*size*, *d*)	A floating point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter. This syntax is deprecated in MySQL 8.0.17, and it will be removed in future MySQL versions

SQL – STRUCTURED QUERY LANGUAGE

DATATYPES – NUMERIC (NÚMEROS)

FLOAT(*p*)	A floating point number. MySQL uses the *p* value to determine whether to use FLOAT or DOUBLE for the resulting data type. If *p* is from 0 to 24, the data type becomes FLOAT(). If *p* is from 25 to 53, the data type becomes DOUBLE()
DOUBLE(*size*, *d*)	A normal-size floating point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter
DOUBLE PRECISION(*size*, *d*)	
DECIMAL(*size*, *d*)	An exact fixed-point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter. The maximum number for *size* is 65. The maximum number for *d* is 30. The default value for *size* is 10. The default value for *d* is 0.
DEC(*size*, *d*)	Equal to DECIMAL(size,d)

SQL – STRUCTURED QUERY LANGUAGE

DATATYPES – DATE AND TIME

DATE	A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'
DATETIME(*fsp*)	A date and time combination. Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. Adding DEFAULT and ON UPDATE in the column definition to get automatic initialization and updating to the current date and time
TIMESTAMP(*fsp*)	A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC. Automatic initialization and updating to the current date and time can be specified using DEFAULT CURRENT_TIMESTAMP and ON UPDATE CURRENT_TIMESTAMP in the column definition
TIME(*fsp*)	A time. Format: hh:mm:ss. The supported range is from '-838:59:59' to '838:59:59'
YEAR	A year in four-digit format. Values allowed in four-digit format: 1901 to 2155, and 0000. MySQL 8.0 does not support year in two-digit format.

SQL – STRUCTURED QUERY LANGUAGE

DATATYPES – DATE AND TIME

DATE	A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'
DATETIME(*fsp*)	A date and time combination. Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. Adding DEFAULT and ON UPDATE in the column definition to get automatic initialization and updating to the current date and time
TIMESTAMP(*fsp*)	A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC. Automatic initialization and updating to the current date and time can be specified using DEFAULT CURRENT_TIMESTAMP and ON UPDATE CURRENT_TIMESTAMP in the column definition
TIME(*fsp*)	A time. Format: hh:mm:ss. The supported range is from '-838:59:59' to '838:59:59'
YEAR	A year in four-digit format. Values allowed in four-digit format: 1901 to 2155, and 0000. MySQL 8.0 does not support year in two-digit format.

SQL – STRUCTURED QUERY LANGUAGE

DATATYPES – EXEMPLOS:

```
`id` bigint(20) NOT NULL,
`is_active` tinyint(1) NOT NULL,
```

```
`id` int(11) NOT NULL,
`action_time` datetime(6) NOT NULL,
`object_id` longtext DEFAULT NULL,
`object_repr` varchar(200) NOT NULL,
`action_flag` smallint(5) UNSIGNED NOT NULL
`change_message` longtext NOT NULL,
`content_type_id` int(11) DEFAULT NULL,
`user_id` int(11) NOT NULL
```

SQL – STRUCTURED QUERY LANGUAGE

CRIAÇÃO DE TABELAS:

CREATE TABLE	Cria tabelas no banco de dados
Sintaxe:	**CREATE TABLE** nome_tabela (nome_atributo1 datatype [NULL/NOTNULL] [DEFAULT valor] [AUTO_INCREMENT], nome_atributo2 datatype ... , : : nome_atributoN datatype ... , **[PRIMARY KEY** (nome_atributoX)**]** , **[FOREIGN KEY** (nome_atributoY) **REFERENCES** nome_tabela_origem (nome_atributo_origem) **]**)

SQL – STRUCTURED QUERY LANGUAGE

CRIAÇÃO DE TABELAS:

```
CREATE TABLE   ALUNO   (
  Matricula       INT(15)      NOT NULL,
  Nome            VARCHAR(40)  NULL,
  CPF             INT(11)      NULL,
  DtNascimento    DATE         NULL,
  Idade           INT(3)       NULL,
  PRIMARY KEY (Matricula)
);
```

String de tamanho 40
Coluna tipo data (dia, mês, ano).

Aluno

Coluna	Tipo	Tamanho
Matrícula	INT	15
Nome	VARCHAR	40
CPF	INT	11
DtNascimento	DATE	
Idade	INT	3

SQL – STRUCTURED QUERY LANGUAGE

CRIAÇÃO DE TABELAS:

nome_tabela	Nome que você determina para a tabela a ser criada.
nome_atributoX	Nome de cada coluna que comporá a tabela.
datatype	É o tipo de dado a ser utilizado na coluna (Inteiro, String, ...).

NULL / NOT NULL	Indica se a coluna pode ser gravada com valor nulo (NULL) ou se seu preenchimento é obrigatório (NOT NULL).
DEFAULT valor	Indica um conteúdo que será inserido automaticamente naquela coluna se não for informado outro conteúdo.
AUTO_INCREMENT	Cria uma numeração automática na coluna; não sendo mais necessário indicar valor para a coluna. É usado com chaves primárias para numerar automaticamente os elementos de uma tabela.
PRIMARY KEY	Uma ou mais colunas que comporão a PK da tabela.
FOREIGN KEY	Uma ou mais colunas que relacionadas a PK de outra tabela (indicada após REFERENCES). Essa(s) coluna(s) forma(m) a FK da tabela.

SQL – STRUCTURED QUERY LANGUAGE

SHOW TABLES

Mostra as tabelas do banco em conexão

```
show tables;
```

DESCRIBE

Descreve a estrutura da tabela desejada

```
describe alunos;
```

DROP TABLE

Deleta a tabela mencionada

```
drop table teste;
```

SENAI SÃO PAULO

SQL – STRUCTURED QUERY LANGUAGE

CRIE UMA TABELA PARA ABRIGAR OS SEGUINTES DADOS, TREINE O CREATE, SHOW, DESCRIBE E DEPOIS O DROP:

id	name	department	birth
1	Maria Gloria	CS	1994-03-12
2	John Smith	IT	1993-02-07
3	Gal Rao	CS	1992-09-11
4	Jakey Smith	EC	1990-08-31
5	Rama Saho	IT	1994-12-09
6	Maria Gaga	EC	1993-10-09

SQL – STRUCTURED QUERY LANGUAGE

ALTERANDO UMA TABELA:

ALTER TABLE — **Altera a estrutura de uma tabela já criada.**

Sintaxe:

```
ALTER TABLE nome_tabela (

    [ADD COLUMN nome_atributox datatype [NULL / NOT NULL]
    [DEFAULT valor][AUTO_INCREMENT] [PRIMARY KEY] ,]

    [DROP COLUMN nome_atributoy]
    [ADD PRIMARY KEY (nome_atributoX)] ]
        [DROP PRIMARY KEY ],

    [ADD FOREIGN KEY (nome_atributoY) REFERENCES
        nome_tabela_origem (nome_atributo_origem) ] ],

    [DROP FOREIGN KEY (nome_atributoY)]

)
```

```
ALTER TABLE Persons
ADD DateOfBirth date;
```

```
ALTER TABLE Persons
MODIFY COLUMN DateOfBirth year;
```

```
ALTER TABLE Persons
DROP COLUMN DateOfBirth;
```

SENAI SÃO PAULO

SQL – STRUCTURED QUERY LANGUAGE

PRATIQUE A ALTERAÇÃO DA TABELA MUDANDO O TAMANHO PERMITIDO PARA A COLUNA NAME E ADICIONANDO MAIS UMA QUARTA COLUNA DE SUA PREFERÊNCIA

id	name	department	birth
1	Maria Gloria	CS	1994-03-12
2	John Smith	IT	1993-02-07
3	Gal Rao	CS	1992-09-11
4	Jakey Smith	EC	1990-08-31
5	Rama Saho	IT	1994-12-09
6	Maria Gaga	EC	1993-10-09

SQL – STRUCTURED QUERY LANGUAGE

INSERINDO DADOS NA TABELA:

INSERT	Inclui elementos novos em uma tabela
Sintaxe:	**INSERT [INTO]** nome_tabela **[** (coluna1, coluna2, ..., colunaN) **]** **VALUES (** conteúdo1, conteúdo2, ..., conteúdoN **)**

```
INSERT INTO tbl_autores
VALUES
(1, 'Daniel', 'Barret'),
(2, 'Gerald', 'Carter'),
(3, 'Mark', 'Sobell'),
(4, 'William', 'Stanek'),
(5, 'Richard', 'Blum'),
(6, 'Jostein', 'Gaarder'),
(7, 'Umberto', 'Eco'),
(8, 'Neil', 'De Grasse Tyson'),
(9, 'Stephen', 'Hawking'),
(10, 'Stephen', 'Jay Gould'),
(11, 'Charles', 'Darwin'),
(12, 'Alan', 'Turing'),
(13, 'Simon', 'Monk'),
(14, 'Paul', 'Scherz');
```

```
INSERT INTO `main_usuario` (`id`, `nome`, `identificador`, `senha`, `nivelAcesso`) VALUES
(1, 'Marcos Vinicius Cardoso Correa', '777', 'master', '1'),
(2, 'Andre', '123', 'master', '2');
```

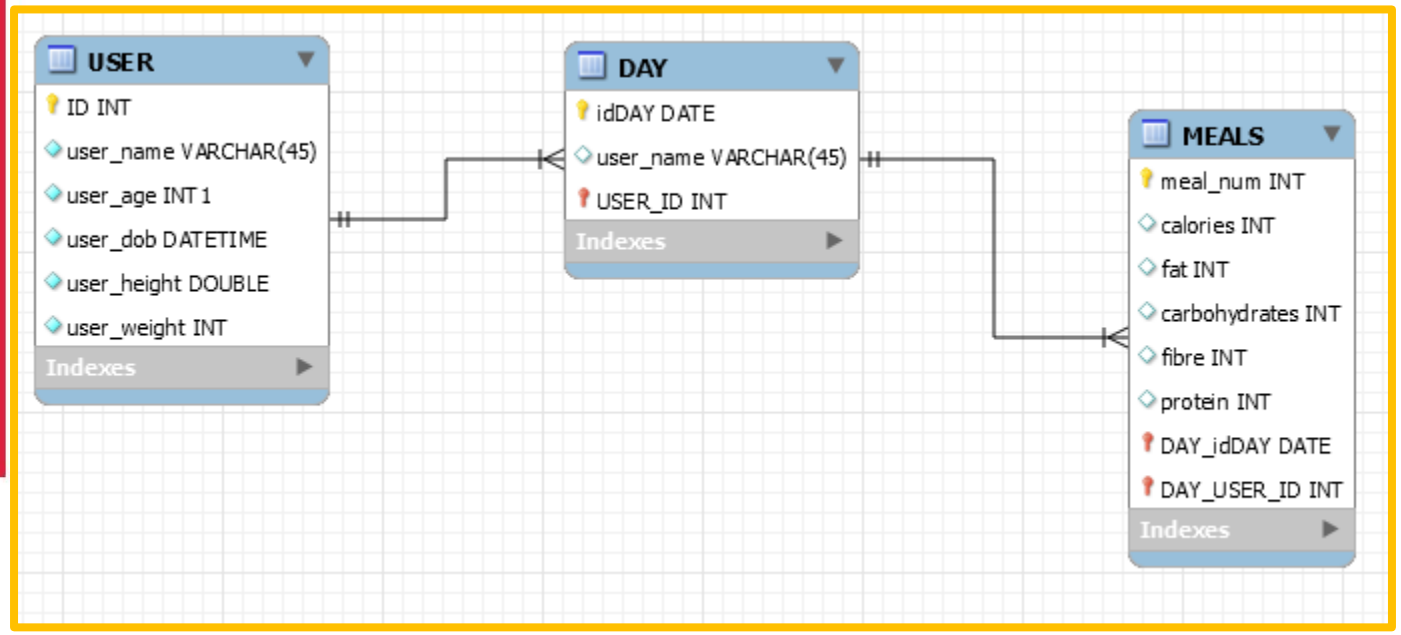
```
INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

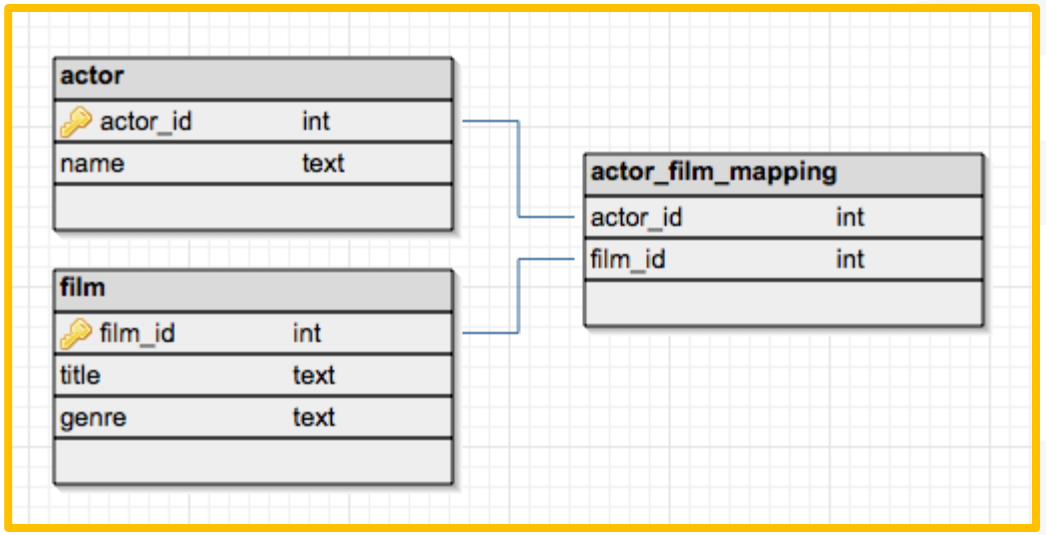
SENAI SÃO PAULO

SQL – STRUCTURED QUERY LANGUAGE

RELACIONANDO DADOS ENTRE TABELAS!

No paradigma relacional, é muito comum a necessidade de relacionarmos uma ou mais tabelas entre si, sendo motivado por basicamente dois motivos: Garantir a integridade de algumas informações e também evitar a criação de tabelas demasiadamente grandes e complexas.





SENAI SÃO PAULO

SQL – STRUCTURED QUERY LANGUAGE

RELACIONANDO DADOS ENTRE TABELAS!

Dessa maneira, tais relacionamentos ocorrem através das chamadas CHAVES ESTRANGEIRAS (FOREIGN KEYS), também denominadas FK!

Analisando isso, será que podemos melhorar a tabela abaixo?

Como podemos garantir que somente sejam atribuídos departamentos existentes?

Como melhorar a manutenibilidade desta tabela caso necessite mudar a sigla de um setor?

id	name	department	birth
1	Maria Gloria	CS	1994-03-12
2	John Smith	IT	1993-02-07
3	Gal Rao	CS	1992-09-11
4	Jakey Smith	EC	1990-08-31
5	Rama Saho	IT	1994-12-09
6	Maria Gaga	EC	1993-10-09

SQL – STRUCTURED QUERY LANGUAGE

RELACIONANDO DADOS ENTRE TABELAS!

Dessa maneira, tais relacionamentos ocorrem através das chamadas CHAVES ESTRANGEIRAS (FOREIGN KEYS), também denominadas FK!

Analisando isso, será que podemos melhorar a tabela abaixo?

Como podemos garantir que somente sejam atribuídos departamentos existentes?

Como melhorar a manutenibilidade desta tabela caso necessite mudar a sigla de um setor?

id	name	birth	department id
1	Maria Gloria	1994-03-12	1
2	John Smith	1993-02-07	2
3	Gal Rao	1992-09-11	1
4	Jakey Smith	1990-08-31	3
5	Rama Saho	1994-12-09	2
6	Maria Gaga	1993-10-09	3

id	department
1	CS
2	IT
3	EC

SQL – STRUCTURED QUERY LANGUAGE

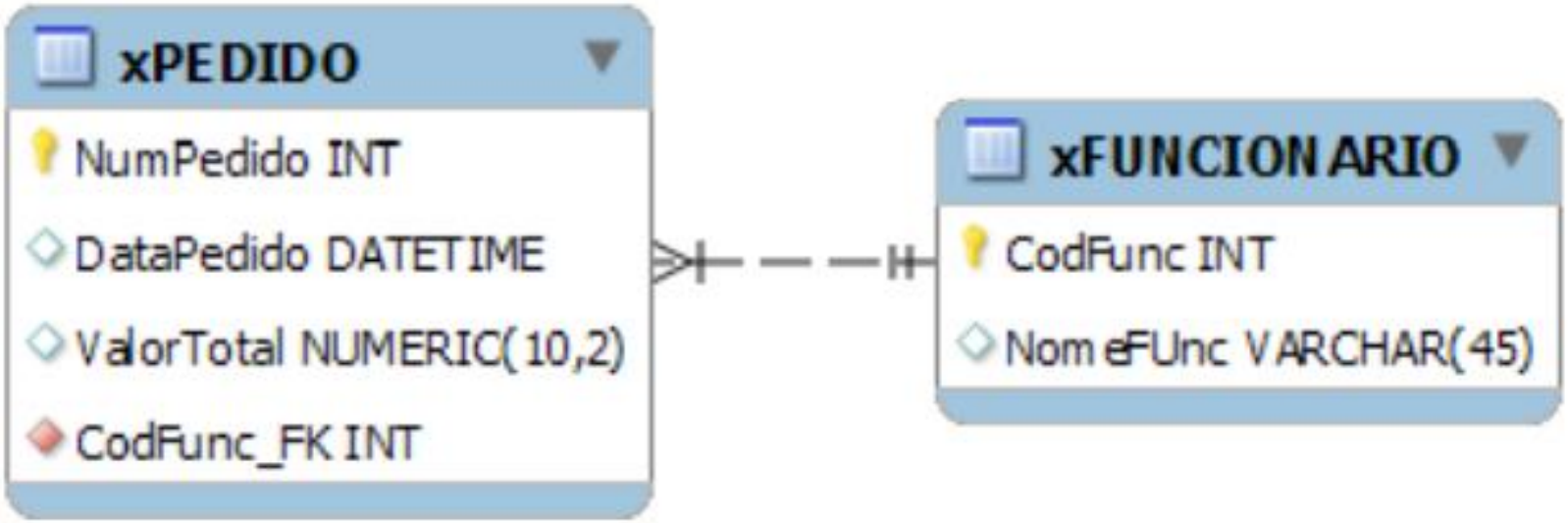
CRIE AS SEGUINTES TABELAS COM OS RESPECTIVOS RELACIONAMENTOS:

id	name	birth	department id
1	Maria Gloria	1994-03-12	1
2	John Smith	1993-02-07	2
3	Gal Rao	1992-09-11	1
4	Jakey Smith	1990-08-31	3
5	Rama Saho	1994-12-09	2
6	Maria Gaga	1993-10-09	3

id	department
1	CS
2	IT
3	EC

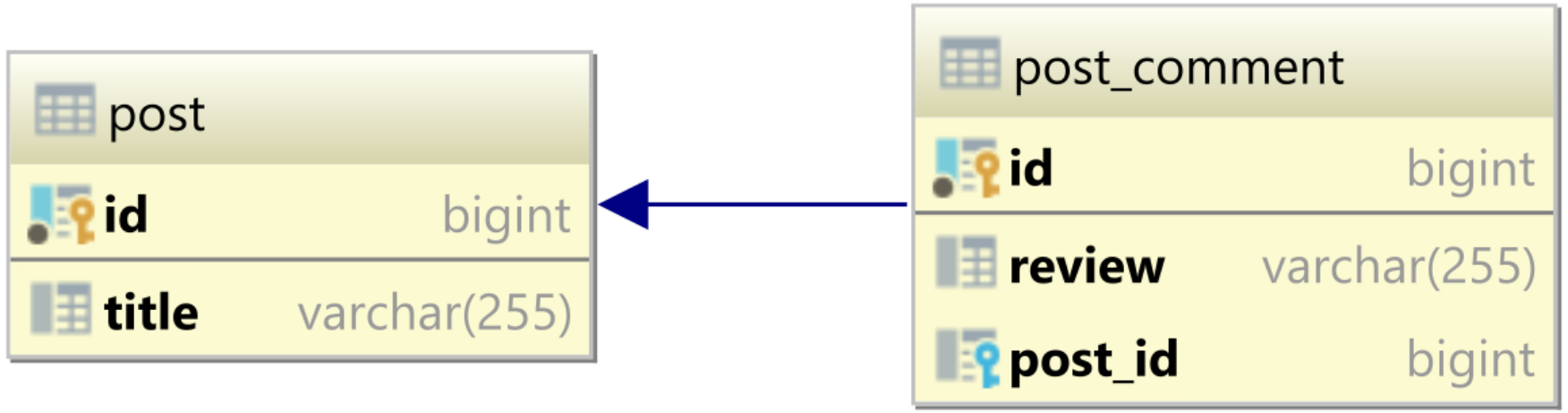
SQL – STRUCTURED QUERY LANGUAGE

CRIE AS SEGUINTES TABELAS COM OS RESPECTIVOS RELACIONAMENTOS:



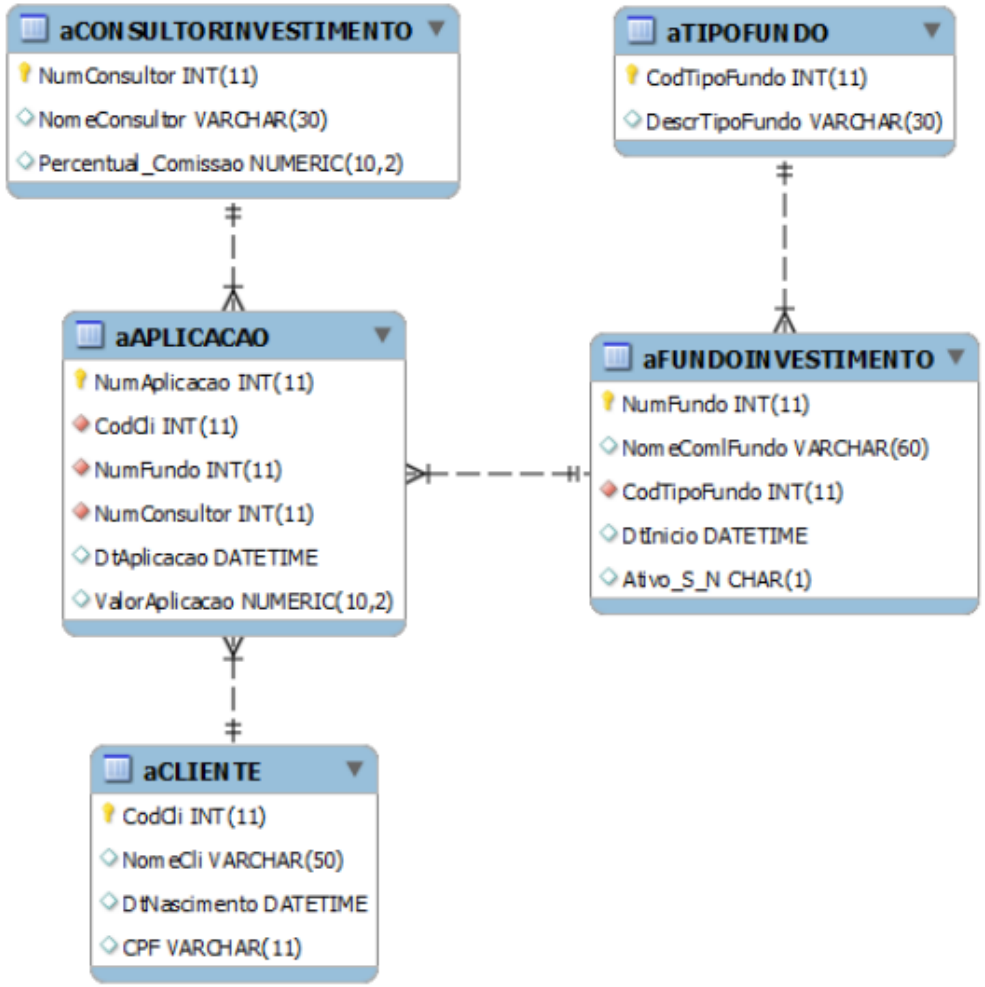
SQL – STRUCTURED QUERY LANGUAGE

CRIE AS SEGUINTES TABELAS COM OS RESPECTIVOS RELACIONAMENTOS:



SQL – STRUCTURED QUERY LANGUAGE

CRIE AS SEGUINTES TABELAS COM OS RESPECTIVOS RELACIONAMENTOS:



SENAI SÃO PAULO