|  |  |
| --- | --- |
|  | Curso: **Tecnologia em Banco de Dados**  Disciplina: **Database Modeling & SQL**  Prof. André Santos - profandre.santos@fiap.com.br |



**Comandos SQL-DDL: exercício 03**

Implementar o banco de dados conforme os modelos de dados descritos abaixo.

O objetivo é elaborar um "**script SQL**" contendo:

* os comandos SQL-DDL necessários para criação das tabelas, constraints e sequences, conforme as especificações.
* os comandos SQL-DDL para as alterações solicitadas.

Além disso, são solicitadas consultas solicitadas simples ao dicionário de dados do SGBD.

Dicionário de dados (lógico relacional):

**Tabela:** FUNCIONARIO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nome da coluna** | **Tipo** | **Tam.** | **Dec.** | **Validação** | **Observações** |
| CD\_FUNC | N | 10 |  | PK | Código do funcionário |
| NM\_FUNC | C | 50 |  | NN | Nome do funcionário |
| IN\_ATIVO | N | 1 |  | NN, CK (0, 1) | Indica funcionário ativo: 0=Inativo, 1=Ativo |
| IN\_SEXO | C | 1 |  | NN, CK ('F','M') | Sexo (F=feminino, M=masculino) |
| DT\_NASC | D | - |  | NN, CK (< DT\_ADMISSAO) | Data de nascimento |
| DT\_ADMISSAO | D | - |  | NN, Default = SYSDATE | Data de admissão na empresa |
| VL\_SALARIO | N | 9 | 2 | CK (> 0) | Salário do funcionário (R$) |
| CD\_DEPTO | N | 6 |  | NN, FK (DEPTO) | Código do departamento relacionado |
| CD\_CHEFE\_FUNC | N | 10 |  | FK (FUNC) | Código do chefe do funcionário |

**Tabela:** DEPARTAMENTO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nome da coluna** | **Tipo** | **Tam.** | **Dec.** | **Validação** | **Observações** |
| CD\_DEPTO | N | 6 |  | PK | Código do departamento |
| NM\_DEPTO | C | 50 |  | NN | Nome do departamento |
| SG\_DEPTO | C | 10 |  | NN, UK | Sigla do departamento (identificador alternativo) |

**Tabela:** PROJETO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nome da coluna** | **Tipo** | **Tam.** | **Dec.** | **Validação** | **Observações** |
| CD\_PROJ | N | 10 |  | PK | Código do projeto |
| NM\_PROJ | C | 10 |  | NN | Nome do projeto |
| TX\_OBS\_PROJ | C | 100 |  | NN | Observações |

Legenda

* Tipos: N = numérico, C = caractere (texto), D = data.
* Validações: PK = chave primária, UK = chave de unicidade, FK = chave estrangeira, NN = campo obrigatório, CK = regra de validação.

**Solicitações para elaboração do Script SQL:**

1. Comando(s) SQL para criação da tabela “FUNCIONARIO”, com todas as constraints.

|  |
| --- |
| CREATE TABLE FUNCIONARIO (  CD\_FUNC NUMBER(10) ,  NM\_FUNC VARCHAR2(50) NOT NULL ,  IN\_ATIVO NUMBER(1) NOT NULL ,  IN\_SEXO VARCHAR2(1) NOT NULL ,  DT\_NASC DATE NOT NULL ,  DT\_ADMISSAO DATE DEFAULT SYSDATE NOT NULL ,  VL\_SALARIO NUMBER(9,2) ,  CD\_DEPTO NUMBER(6) /\*NN, FK (DEPTO) \*/ ,  CD\_CHEFE\_FUNC NUMBER(10) /\*FK (FUNC)\*/ ,  CONSTRAINT sal\_maior  CHECK(VL\_SALARIO >0) ,  CONSTRAINT pk\_funcionario  PRIMARY KEY (CD\_FUNC) ,  CONSTRAINT sexo\_bin  CHECK(IN\_SEXO IN('M','F')) ,  CONSTRAINT dt\_nasci\_ch  CHECK(DT\_NASC < DT\_ADMISSAO) ,  CONSTRAINT in\_ativo\_ch  CHECK (IN\_ATIVO IN (0,1))  );  ALTER TABLE FUNCIONARIO  ADD CONSTRAINT fk\_cod\_dept  Foreign key (CD\_DEPTO)  REFERENCES DEPARTAMENTO;  ALTER TABLE FUNCIONARIO  ADD CONSTRAINT fk\_cod\_chefe  FOREIGN KEY (CD\_CHEFE\_FUNC)  REFERENCES FUNCIONARIO; |

2. Comando(s) SQL para criação da tabela “DEPARTAMENTO”, com todas as constraints.

|  |
| --- |
| Drop table DEPARTAMENTO CASCADE CONSTRAINTS PURGE;  CREATE TABLE DEPARTAMENTO (  CD\_DEPTO NUMBER(6) ,  NM\_DEPTO VARCHAR2(50) NOT NULL ,  SG\_DEPTO VARCHAR2(10) NOT NULL UNIQUE ,  CONSTRAINT pk\_cd\_depto  PRIMARY KEY(CD\_DEPTO)  ); |

3. Comando(s) SQL para criação da tabela “PROJETO”, com todas as constraints.

|  |
| --- |
| DROP TABLE PROJETO CASCADE CONSTRAINTS PURGE;  CREATE TABLE PROJETO (  CD\_PROJ NUMBER(10) ,  NM\_PROJ VARCHAR2(10) NOT NULL,  TX\_OBS\_PROJ VARCHAR2(100) NOT NULL,  CONSTRAINT pk\_cd\_proj  PRIMARY KEY(CD\_PROJ)  ); |

4. Crie sequences para cada uma das tabelas:SQ\_FUNC, SQ\_DEPT e SQ\_PROJ.

|  |
| --- |
| CREATE SEQUENCE SQ\_FUNCIONARIO;  CREATE SEQUENCE SQ\_DEPARTAMENTO;  CREATE SEQUENCE SQ\_PROJETO; |

5. Crie um relacionamento “**n:n**” entre as tabelas “FUNCIONARIO” e “PROJETO”, com as devidas constraints para garantir as integridades de dados.

|  |
| --- |
| CREATE TABLE NPROJFUC(  CD\_DEPTO NUMBER(6),  CD\_PROJ NUMBER(10),  Constraint fknn\_departamento  Foreign key (CD\_DEPTO)  REFERENCES DEPARTAMENTO,  Constraint fknn\_projeto  Foreign key(CD\_PROJ)  REFERENCES PROJETO,  PRIMARY KEY (CD\_DEPTO,CD\_PROJ)  ); |

6. Aumente o tamanho da coluna “TX\_OBS\_PROJ”, da tabela “PROJETO”, para 200 (duzentos) caracteres.

|  |
| --- |
| ALTER TABLE PROJETO  MODIFY TX\_OBS\_PROJ VARCHAR2(200); |

7. A coluna “VL\_SALARIO”, da tabela “FUNCIONARIO”, deve ser obrigatória. Realize a modificação.

|  |
| --- |
| ALTER TABLE FUNCIONARIO  MODIFY VL\_SALARIO NOT NULL;  ALTER TABLE FUNCIONARIO  DISABLE CONSTRAINT SYS\_C006995885;  ALTER TABLE FUNCIONARIO  ADD CONSTRAINT nn\_nullfuc  Check(VL\_SALARIO IS NOT NULL); |

8. Acrescente a coluna “TX\_OBS\_FUNC” na tabela “FUNCIONARIO”, com tipo “caractere” e tamanho = 500.

|  |
| --- |
| ALTER TABLE FUNCIONARIO  ADD TX\_OBS\_FUNC varchar2(500); |

9. Consulte o dicionário de dados para verificar todos os objetos criados (USER\_OBJECTS), ordenando por tipo do objeto e nome do objeto, apresentando: tipo, nome, data de criação e status.

|  |
| --- |
| DESC FUNCIONARIO;  SELECT \* FROM TAB;  SELECT \* FROM USER\_OBJECTS  ORDER BY OBJECT\_TYPE, OBJECT\_NAME; |

10. Consulte o dicionário de dados para verificar todas as colunas (USER\_TAB\_COLUMNS), ordenando por tabela e nome da coluna, apresentando: tabela, nome da coluna, tipo.

|  |
| --- |
| SELECT TABLE\_NAME, COLUMN\_NAME, DATA\_TYPE FROM USER\_TAB\_COLUMNS  ORDER BY TABLE\_NAME, COLUMN\_NAME, DATA\_TYPE; |

11. Consulte o dicionário de dados para verificar as constraints (USER\_CONSTRAINTS), apresentando, pelo menos, o nome da tabela, nome da constraint, tipo, status.

|  |
| --- |
| SELECT TABLE\_NAME, CONSTRAINT\_NAME,STATUS FROM USER\_CONSTRAINTS; |

12. Consulte o dicionário de dados para verificar as sequences (USER\_SEQUENCES).

Verifique as colunas com nome, valores mínimo e máximo, incremento, cache e próximo número.

|  |
| --- |
| SELECT MIN\_VALUE, MAX\_VALUE,INCREMENT\_BY, CACHE\_SIZE,(LAST\_NUMBER +1) AS NEXT\_NUMBER FROM USER\_SEQUENCES; |

13. Altere as configurações das sequences SQ\_FUNC e SQ\_DEPTO para deixá-las sem cache.

|  |
| --- |
| alter sequence SQ\_FUNCIONARIO  NOCACHE;  alter sequence SQ\_DEPARTAMENTO  NOCACHE; |

14. Aumente para 100 o cache da sequence SQ\_PROJ e verifique novamente a USER\_SEQUENCES.

|  |
| --- |
| ALTER SEQUENCE SQ\_PROJETO  CACHE 100; |

15. Execute o comando "**SELECT sq\_func.CURRVAL FROM DUAL;**".

O que ocorreu?

|  |
| --- |
| ERRO |

16. Abra outra janela do SQL Develper e teste as sequences com **NEXTVAL** e **CURRVAL** nas duas janelas. Verifique a USER\_SEQUENCES novamente.

|  |
| --- |
| SELECT SQ\_FUNCIONARIO.NEXTVAL FROM DUAL;  SELECT SQ\_FUNCIONARIO.CURRVAL FROM DUAL; |