

VEM SER

Banco de Dados Oracle

Introdução e Conceitos Básicos

Conteúdo do módulo

- Introdução e conceitos básicos
 - SQL
 - Criação de tabelas
 - Inserção de informações
- Modelagem de Dados
 - Criação de tabelas relacionadas (FK)
 - Comandos de atualização e remoção de registros
- Junção de tabelas
 - Comandos avançados de seleção de informações
 - Junção de tabelas
- JDBC
 - Conectar o banco de dados com uma aplicação Java Real
- Projeto Final

Sumário

- Conceitos básicos
- SGBD
- Banco de dados Relacional
- Tabelas
- Tipos de Dados
- Chaves
- SQL
 - DDL
 - DML



O que é um Banco de Dados?

O que é um Banco de Dados?

- Um banco de dados é uma coleção organizada de informações - ou dados - estruturadas, normalmente armazenadas eletronicamente em um sistema de computador;
- Um banco de dados é geralmente controlado por um [sistema de gerenciamento de banco de dados \(SGBD / DBMS\)](#);
- Os dados nos tipos mais comuns de bancos de dados em operação atualmente são modelados em linhas e colunas em uma série de tabelas para tornar o processamento e a consulta de dados eficientes.

SGBD ou DBMS



- <https://dicasdeprogramacao.com.br/o-que-e-um-sgbd>

Banco de Dados Relacional

- Um banco de dados relacional é um tipo de banco de dados que armazena e fornece acesso a pontos de dados relacionados entre si.
- Bancos de dados relacionais são baseados no modelo relacional, uma maneira intuitiva e direta de representar dados em tabelas.
- Em um banco de dados relacional, cada linha na tabela é um registro com uma ID exclusiva chamada chave.
- As colunas da tabela contêm atributos dos dados e cada registro geralmente tem um valor para cada atributo, facilitando o estabelecimento das relações entre os pontos de dados.

Alguns Banco de Dados Relacionais



Tabelas

- Nos **modelos** de bases de dados relacionais, a tabela é um conjunto de dados dispostos em número infinito de colunas e número ilimitado de **linhas** (ou tuplas).
- As **colunas** são tipicamente consideradas os campos da tabela, e caracterizam os tipos de dados que deverão constar na **tabela** (numéricos, alfa-numéricos, datas, coordenadas, etc).
- O número de linhas pode ser interpretado como o número de combinações de valores dos campos da tabela, e pode conter linhas idênticas, dependendo do objetivo, ou também chamadas de **registros**.
- A forma de referenciar inequivocamente uma única linha é através da utilização de uma **chave primária**.

Tabelas

	Emp_Id	Last_Name	First_Name	Gender	Title
▶	1000	Torbati	Yolanda	F	Programmer
	1001	Kleinn	Joel	M	Programmer
	1002	Ginsburg	Laura	F	President
	1003	Cox	Jennifer	F	Programmer
	1005	Ziada	Mauri	M	Product Designer
	1006	Keyser	Cara	F	Account Executive
	1010	Smith	Roxie	M	Programmer
	1011	Nelson	Robert	M	Programmer
	1012	Sachsen	Lars	M	Support Technician
	1013	Shannon	Don	M	Product Designer

Gravar 1

Tipo de Dados – Caracteres / Textos

- **VARCHAR2** – Sequencia de caracteres alfanuméricos de tamanho variável com limite de 4000 Bytes.
- **VARCHAR** – Sinônimo para VARCHAR2, por recomendação da própria Oracle, este tipo de dados **não deve** ser usado, pois existe a possibilidade do tipo VARCHAR integrar versões futuras do banco de dados Oracle com características diferentes do VARCHAR2. O comprimento para este tipo de dados é variável, assim somente o espaço que realmente for preenchido será armazenado na memória.
- **CHAR** – Armazena caracteres alfanuméricos de tamanho 1 até 255. Esse tipo de dados é de comprimento fixo.
 - Sua melhor utilização é quando sabe-se que o conteúdo tem um tamanho fixo, exemplo uma Flag que irá gravar “Sim” ou Não, em todas as situação sempre serão preenchidos 3 caracteres, ou então a sigla de um Estado que sempre será composta por dois caracteres.

Tipo de Dados – Números

- **NUMBER(x,y)** – Para valores inteiros:
 - x = valor inteiro
 - y= valor de casas decimais
- **DECIMAL(x,y)** – Valores reais onde:
 - x = valor inteiro
 - y= valor de casas decimais
- **INTEGER** – Tipo de dados para números inteiros. Equivalente ao NUMBER.
- **SMALLINT** – Equivalente ao NUMBER, porém ocupa a metade do espaço em memória.
- <http://paulokaupa.blogspot.com/p/tipos-de-dados.html>

Tipo de Dados – Datas

- **DATE** – Permite armazenar datas que vão de 1 de Janeiro de 4712 AC à 31 de Dezembro de 9999 DC. Os valores armazenados incluem século, ano, mês, dia, hora, minuto e segundo.
- **TIMESTAMP** – Similar ao tipo DATE, mas com uma maior precisão para segundos.

Tipo de Dados – Diversos

- [BLOB, CLOB, NCLOB, BFILE, NVARCHAR2, MLSLABEL e NCHAR](#) são também tipos de dados possíveis mas menos usados. Consulte a documentação do Oracle.
- https://docs.oracle.com/cd/B28359_01/server.111/b28318/datatype.htm

Chaves (Keys)

- Chave Primária
- Chave Única
- Chave Estrangeira

Chave Primária (Primary Key (PK))

- É o identificador único de um registro na tabela.
- Pode ser constituída de um campo (chave simples) Exemplo: ID
- Dois ou mais campos (chave composta), de tal maneira que não existam dois registros com o mesmo valor de chave primária.
- Não permite valores nulos e impõe a exclusividade de linhas.

Chave Única (Unique Key(UK))

- Pode ser constituída de um campo.
- Dois ou mais campos (chave composta), de tal maneira que não existam dois registros com o mesmo valor de chave única.
- Permite valores nulos e impõe a exclusividade de linhas.

Chave Estrangeira (Foreign Key (FK))

- É a chave que permite a referência a registros oriundos de outras tabelas.
- Ou seja, é o campo ou conjunto de campos que compõem a chave primária de uma outra tabela.
- A utilização da chave estrangeira possibilita a implementação da integridade de dados diretamente no banco de dados, conhecida como integridade referencial.
- Uma chave estrangeira é a representação de um relacionamento entre tabelas.

Structured Query Language (SQL)

- Resumidamente, é uma linguagem de programação para lidar com banco de dados relacional.
- Foi criado para que vários desenvolvedores pudessem acessar e modificar dados de uma empresa simultaneamente, de maneira descomplicada e unificada.



Para que serve o SQL?

Para que serve o SQL?

- A programação SQL pode ser usada para analisar ou executar tarefas em tabelas.
- Tipos de Comandos:
 - DDL (Data Definition Language)
 - DML (Data Manipulation Language)



Vamos instalar!



Ambiente

- Docker Com Oracle 11G
- <https://hub.docker.com/r/epiclabs/docker-oracle-xe-11g>
- `docker pull epiclabs/docker-oracle-xe-11g`
- Rodar (comando todo na mesma linha)
- `docker run --name bd-oracle -d -p 1521:1521 -e ORACLE_ALLOW_REMOTE=true -e ORACLE_PASSWORD=oracle -e RELAX_SECURITY=1 epiclabs/docker-oracle-xe-11g`
- Baixar DBeaver Community Edition
- <https://dbeaver.io/download>

Ambiente

Connect to a database

Oracle Connection Settings

Oracle connection settings

ORACLE

Main Oracle properties Driver properties SSH Proxy

Connection Type:

Basic TNS Custom

Host: localhost Port: 1521

Database: xe Service Name

Authentication

Authentication: Database Native

Username: system Role: Normal

Password: Save password locally

Client: <not present>

i You can use variables in connection parameters. Connection details (name, type, ...)

Driver name: Oracle Edit Driver Settings

Test Connection ...

< Back Next > Finish Cancel



Ambiente

- Rodar Comandos (linha por linha)

```
CREATE USER VEM_SER IDENTIFIED BY oracle;  
GRANT CONNECT TO VEM_SER;  
GRANT CONNECT, RESOURCE, DBA TO VEM_SER;  
GRANT CREATE SESSION TO VEM_SER;  
GRANT DBA TO VEM_SER;  
GRANT CREATE VIEW, CREATE PROCEDURE, CREATE SEQUENCE to VEM_SER;  
GRANT UNLIMITED TABLESPACE TO VEM_SER;  
GRANT CREATE MATERIALIZED VIEW TO VEM_SER;  
GRANT CREATE TABLE TO VEM_SER;  
GRANT GLOBAL QUERY REWRITE TO VEM_SER;  
GRANT SELECT ANY TABLE TO VEM_SER;
```

Comandos Data Definition Language (DDL)

- Create Table
- Drop Table
- Create Sequence
- Drop Sequence

Create Table

```
CREATE TABLE SCHEMA_NAME.TABLE_NAME (  
    column_1 DATA_TYPE COLUMN_CONSTRAINT,  
    column_2 DATA_TYPE COLUMN_CONSTRAINT,  
    ...  
    TABLE_CONSTRAINT  
);
```

```
CREATE TABLE VEM_SER.PESSOA (  
    id_pessoa NUMBER NOT NULL,  
    nome VARCHAR2(255) NOT NULL,  
    data_nascimento DATE NOT NULL,  
    telefone VARCHAR2(14), -- +5551999999999  
    idade NUMBER(3) NOT NULL,  
    altura DECIMAL(4,2) NOT NULL,  
    cpf CHAR(11) UNIQUE NOT NULL,  
    PRIMARY KEY(id_pessoa)  
);
```



Drop Table

```
DROP TABLE SCHEMA_NAME.TABLE_NAME;
```

```
DROP TABLE VEM_SER.PESSOA;
```



Create Sequence

```
CREATE SEQUENCE SCHEMA_NAME.NAME_OF_SEQUENCE  
START WITH 1  
INCREMENT BY 1  
NOCACHE NOCYCLE;
```

```
CREATE SEQUENCE VEM_SER.SEQ_PESSOA  
START WITH 1  
INCREMENT BY 1  
NOCACHE NOCYCLE;
```



Drop Sequence

```
DROP SEQUENCE SCHEMA_NAME.NAME_OF_SEQUENCE;
```

```
DROP SEQUENCE VEM_SER.SEQ_PESSOA;
```



Exercício #1

- Vamos usar comandos DDL;
- Criar a tabela conforme especificação abaixo:
 - Nome: VEM_SER.PESSOA
 - Campos:

```
id_pessoa NUMBER NOT NULL,  
nome VARCHAR2(255) NOT NULL,  
data_nascimento DATE NOT NULL,  
telefone VARCHAR2(14), -- +5551999999999  
idade NUMBER(3) NOT NULL,  
altura DECIMAL(4,2) NOT NULL,  
cpf CHAR(11) UNIQUE NOT NULL,  
PRIMARY KEY(id_pessoa)
```

Comandos Data Manipulation Language (DML)

- Select
- Insert
- Update
- Delete



Select

```
SELECT <campos> FROM SCHEMA.TABELA;
```

```
SELECT * FROM VEM_SER.PESSOA;
```

```
SELECT id_pessoa, cpf, nome FROM vem_ser.pessoa;
```



Insert

```
INSERT INTO SCHEMA.TABELA (<campo1>, <campo2>, <campo3>, ...)  
VALUES(<valor1>, <valor2>, <valor3>, ...);
```

```
INSERT INTO VEM_SER.PESSOA (id_pessoa, nome, data_nascimento, telefone, idade, altura, cpf)  
VALUES(1, 'Miguel Machado', TO_DATE('01-01-1990', 'dd-mm-yyyy'), '5199999999', 30, 1.85, '12345678999');
```

Update

```
UPDATE SCHEMA.TABELA  
SET <campo1> = <valor1>, <campo2> = <valor2>, ...  
WHERE <condicao> ;
```

```
UPDATE VEM_SER.PESSOA  
SET nome = 'Pedro Machado', idade = 30  
WHERE id_pessoa = 1;
```



Delete

```
DELETE FROM SCHEMA.TABELA WHERE <campo> = 'X';
```

```
DELETE FROM VEM_SER.PESSOA WHERE id_pessoa = 1;
```



Exercício #2

- Vamos usar comandos DML;
- Inserir 4 linhas na tabela PESSOA com o comando INSERT;
- Selecionar todas as linhas com SELECT;
- Atualizar a idade das pessoas com UPDATE, para que tenham +1 ano;
- Apagar a última linha com o comando DELETE;
- Selecionar todas as linhas com SELECT e observar as mudanças;



Task

- Criar uma pasta “vsXX-back/modulo-02-bd/aula-01” na raiz do seu repositório do git;
- Criar um arquivo de script com o nome de “task-01.sql”;
- Criar scripts para:
 - Criar uma tabela **ESTUDANTE** com os campos:
 - **id_estudante**: numérico e chave primária
 - **nome**: texto até 200 caracteres não nulo
 - **data_nascimento**: Data não nulo
 - **nr_matricula**: numérico de 10 não nulo, único (UK)
 - **ativo**: caractere não nulo, usar dados (‘S’ = ativo, ‘N’ = não ativo)
 - Criar uma sequence para essa tabela (**SEQ_ESTUDANTE**);
 - Inserir 10 registros para essa tabela;
 - Selecionar os registros;



Task #2 Grupo

- Criar uma pasta “bd” na raiz do seu projeto;
- Criar um arquivo de script com o nome de “criar.sql”;
- Neste arquivo, ter a criação DDL de pelo menos 1 tabela e sequence principal;
- Criar um arquivo de script com o nome de “dados.sql”;
- Inserir 10 registros para essa tabela;



Let's *Tech Up Together*