**Introdução ao SQL com ORACLE**

**Entidades:**

**Tablespace** (espaço para tabelas) = entidade mais básica > entidades associadas, a mais básica (onde se armazenam informações) são as Tabelas

**Tabelas** (como uma planilha de excel) tem linhas infinitas, porém as colunas possuem propriedades únicas (todas as informações são do mesmo tipo) e números limitados.

**Colunas** = CAMPOS; **Linhas** = REGISTROS.

**Chave Primária** (pode ou não se repetir nas linhas da tabela) = uma ou mais colunas, onde o conjunto delas não se repetem nas linhas. Ex (1 campo): CPF. IMPORTANTE: o banco de dados não deixa incluir 2 iguais.

**Chave Estrangeira** = um campo com relação com outro campo de outra tabela.

PS: um campo pode ser chave primária e estrangeira ao mesmo tempo.

**Índice** = estrutura a parte que ajuda a fazer buscas na tabela.

**Esquemas** (schemas) = assunto de um grupo de tabelas.

**View** (Visão) = tabela lógica, "cria um atalho" para dados com determinados critérios de uma tabela.

**Consulta** = pode ser feita em várias tabelas ao mesmo tempo e pode conter filtros. A partir do resultado da consulta é construida uma view.

**Procedures** (**PLSQUL**) = extensão do sql para realizar procedimentos mais complexos.

**Trigger** = ações que são realizados caso tal coisa aconteça

**CHAR** = campos tipo texto, de tamanho fixo, com no máximo 2000 caracteres. Ex: CHAR(10) = "A " (ele preenche os espaços vazios).

**VARCHAR2** = tamanho variável de 1 a 4000 carac. Ex: VARCHAR2(10) = "A" (não preenche o resto).

**NCHAR** ou **NVARCHAR2** = tamanho fixo ou variável, com caracters unicode (de todas as línguas e acentos)

**CLOB** = VARCHAR2 com 8 Terabytes de dados por campo.

**NLOB** = NVARCHAR2 com 8 Terabytes.

**NUMBER** = tamanho fixo ou ponto flutuante (quando não especifica o tamanho). Ex: NUMBER(\*,2) = tamanho, casas decimais. \* = ponto flutuante.

**DATE** = datas Julianas, de 1/1/4712 a.c. até 31/12/9999 d.c., no formato DD-MM-YY. sempre tem uma hora relacionado à data, por padrão 00:00:00 AM.

**TIMESTAMP** = aceita segundos fracionários;

**TIMESTAMP WITH TIME ZONE** = reconhece fuso horário;

**TIMESTAMP WITH LOCAL TIME ZONE** = reconhece horário de verão;

**BLOB** = binários, armazena até 128 terabytes.

**BFILE** = cria um atalho para os dados (binários) armazenados no S.O. (tamanho limitado pelo S.O.).

**Criar tabela** = CREATE TABLE nome\_da\_tabela (nome NVARCHAR2(150), numero\_teste FLOAT);

**Apagar uma tabela** = DROP TABLE nome\_da\_tabela;

**Inserir linha** = INSERT INTO nome\_da\_tabela (nome, numero\_teste) VALUES ('Karen', 27.09); os nomes das colunas podem ser omitidos;

**Editar linha** = UPDATE nome\_da\_tabela SET nome = 'Karen Brissow' WHERE numero\_teste = '27.09'; pode alterar mais de uma coluna de uma vez, é só separar com vírgulas, também pode fazer operações matemáticas no valor EX: coluna = coluna \* 1.5 ( aumenta em 50% o valor)

**Excluir linha** = DELETE FROM nome\_da\_tabela WHERE nome = 'Karen Brissow';

**Chave primária** = ALTER TABLE nome\_da\_tabela ADD CONSTRAINT pk\_nome\_da\_tabela PRIMARY KEY (nome); nome não vai se repetir

**Visualizar conteúdo da tabela** = SELECT \* FROM nome\_da\_tabela;

**Add coluna** = ALTER TABLE nome\_da\_tabela ADD nova\_coluna DATE;

**Add data** = INSERT INTO nome\_da\_tabela (nome, numero\_teste, nova\_coluna) VALUES ('KAREN BRISSOW', 27.09, TO\_DATE ('27/09/1998', 'DD/MM/YYYY');

**Mostrar mês** = SELECT TO\_CHAR (nova\_coluna, 'MM') FROM nome\_da\_tabela;

**Mostrar campos específicos** = SELECT nome, nova\_coluna FROM nome\_da\_tabela; Se mudar a ordem dos campos, altera só na visualização

**Mostrar campo outro nome** = SELECT nome AS nome\_do\_cliente FROM nome\_da\_tabela; altera o nome do campo só na visualização

PS: where não precisa ser só na chave primária.

**Condições** = SELECT + WHERE aceita (=, <, >, <=, >=, <> diferente) também aceita (AND e OR);

**Filtrar por data específica** = SELECT \* FROM tb\_clientes WHERE TO\_CHAR(data\_nascimento, 'MM') = 9;

**Consultas SQL no ORACLE**

SELECT \* FROM TABELA WHERE SABOR IN ('Laranja', 'Manga'); -- é igual a OR

SELECT \* FROM TABELA WHERE SABOR = 'Laranja' OR SABOR = 'Manga';

LIKE -- Condição para filtrar textos

LIKE '%Soares%' -- busca Soares em qualquer lugar do texto;

LIKE '%Soares' -- busca Soares no final do texto;

LIKE 'Soares%' -- busca Soares no início do texto.

SELECT \* FROM TABELA WHERE NOME LIKE '%Cruz%';

DISTINCT -- não repete

SELECT DISTINCT SABOR FROM TABELA; -- mostra quantos sabores tem sem repetir;

SELECT DISTINCT SABOR, TAMANHO FROM TABELA; -- mostra quantas cominações tem.

ROWNUM -- Número da linha (só funciona com < ou <=)

SELECT \* FROM TABELA WHERE ROWNUM <= 4; -- Só vai mostrar as 4 primeiras linhas da tabela.

ORDER BY -- Ordenar o resultado da consulta

SELECT \* FROM TABELA WHERE SABOR = 'Manga' ORDER BY CODIGO; -- mostra todos que tem sabor manga e ordena pelo código do menor para o maior;

ORDER BY CODIGO DESC; -- do maior p/ o menor;

GROUP BY -- Agrupa e realiza uma operação com os números

SUM -- soma

MAX -- máximo

MIN -- mínimo

AVG -- média

COUNT -- conta ocorrência

-- dentro pode fazer operações exemplo SUM (X \* Y)

SELECT X, SUM(Y) FROM TABELA GROUP BY X; -- Agrupa os valores da coluna X e soma os da coluna Y quando X se repetir --

SELECT SUM(Y) FROM TABELA -- Soma todos os valores da coluna Y

CASE -- igual a switch case

SELECT X, (CASE

WHEN Y>=8 AND Y<=10 THEN 'OTIMO'

WHEN Y>=7 AND Y<8 THEN 'BOM'

WHEN Y>=5 AND Y<7 THEN 'MEDIO'

ELSE 'RUIM'

END) AS STATUS FROM TABELA

HAVING -- condição aplicada no resultado do group by

SELECT X, SUM(Y) FROM TABELA GROUP BY X HAVING SUM(Y) >= 6;

--Soma Y quando X repete, mas só mostra quando o resultado for maior ou igual a 6;

INNER JOIN -- mostra os dados de duas tabelas com uma chave em comum

SELECT A.NOME, B.HOBBY FROM TABELA\_ESQUERDA A INNER JOIN TABELA\_DIREITA B ON A.IDENTIFICADOR = B.IDENTIFICADOR;

LEFT JOIN -- mostra todos os dados da tabela antes do comando e só mostra quem se repete da tabela depois do comando (os que não se repetem aparece NULL)

RIGHT JOIN -- o inverso da left join

-- pode ser usado WHERE COLUNA IS NULL para mostrar só as que não se repetem.

FULL JOIN -- é o left e o right ao mesmo tempo, ou seja, mostra quando tem relação e mostra null quando não tem, mas sempre mostra as duas tabelas

CROSS JOIN -- mostra uma combinação entre as tabelas;

SELECT TABELA\_A.NOME, TABELA\_B.BAIRRO FROM TABELA\_A, TABELA\_B;

UNION -- junta tabelas com campos de tipos iguais, as linhas que repetirem ele junta como distinct

UNION ALL -- não junta as linhas repetidas

IN -- usar uma sub consulta dentro da consulta;

SELECT \* FROM TABELA\_1 WHERE COLUNA IN (SELECT DISTINCT COLUNA FROM TABELA\_2);

--só mostra as linhas da tabela 1 que tem os mesmos itens da coluna na tabela 2;

VIEW -- tabela lógica, resultado de uma consulta, que pode ser usada depois em qualquer outra consulta (colocar VW\_ na frente do nome da view);

LOWER -- coloca tudo que está dentro do varchar em letras minúsculas;

UPPER -- coloca tudo em letras maiúsculas;

INITCAP -- coloca as primeiras letras de cada palavra em maiúscula;

CONCAT -- concatena duas strings (pode ser substituído por ||);

LPAD -- preenche a esquerda do texto com caracteres;

RPAD -- preenche a direita do texto com caracteres;

SUBSTR -- pega uma parte específica do string;

INSTR -- mostra a posição de um caractere ou string dentro de outra e mostra zero se não tiver ocorrência;

LTRIM -- tira os espaços em branco do lado esquerdo ou um caractere específico;

RTRIM -- tira os espaços em branco do lado direito ou um caractere específico;

LENGTH -- mostra o comprimento da string;

TRANSLATE -- substitui um caractere específico por outro;

REPLACE -- substitui uma string por outra;

TO\_CHAR -- converte a data em string;

SYSDATE -- mostra a hora no computador no momento que o comando for executado;

(Data + número) = adiciona dias;

(Data - número) = subtrai dias;

(Data - data) = mostra o número de dias entre as datas;

MONTHS\_BETWEEN -- mostra o número de meses entre duas datas;

ADD\_MONTHS -- adiciona meses;

NEXT\_DAY -- mostra o próximo dia da semana (1 domingo, 7 sábado);

LAST\_DAY -- mostra o último dia do mês da data especificada;

TRUNC -- devolve a primeira data do mês, ano ou primeira hora do dia;

ROUND -- igual ao trunc, mas pula um ano se estiver no final;

ROUND e TRUNC -- funcionam da mesma forma para arredondar números;

CEIL -- determina o menor inteiro, maior ou igual que o valor especificado (como um trunc pra cima);

FLOOR -- igual ao trunc, mas esse tem mais compatibilidade com outras plataformas;

POWER -- potenciação;

EXP -- exponencial elevado ao número especificado;

SQRT -- raiz quadrada;

SIGN -- mostra o sinal do número;

ABS -- módulo;

MOD -- resto da divisão;

TO\_DATE -- converte string em data;

TO\_CHAR -- converte data em string;

EXTRACT -- separa uma parte da data;

TO\_NUMBER -- converte string em número;

NVL -- se o campo for nulo a função usa um valor especificado;

GREATEST -- devolve o maior valor entre 3 valores especificados;

**Manipulação de dados no ORACLE**

**Adicionar chave estrangeira** = ALTER TABLE tabela\_filha ADD CONSTRAINT FK\_tabela\_mãe FOREIGN KEY (nome\_da\_coluna) REFERENCES tabela\_mãe (nome\_da\_coluna);

**Inserir mais de uma linha** = INSERT ALL INTO nome\_da\_tabela (colunas) VALUES (valores) INTO nome\_da\_tabela (colunas) VALUES (valores2) INTO nome\_da\_tabela (colunas) VALUES (valores3);

**Modificar tipo de uma coluna** = ALTER TABLE nome\_da\_tabela MODIFY (coluna VARCHAR2(50));

**Inserir todos os dados de uma tabela em outra** = INSERT INTO nome\_da\_tabela

SELECT coluna\_1 AS coluna FROM tabela\_2;

**Inserir com ressalva** = INSERT INTO nome\_da\_tabela

SELECT coluna\_1 AS coluna FROM tabela\_2 WHERE coluna NOT IN (SELECT coluna FROM nome\_da\_tabela); Insere todos exceto quem já existe na tabela “nome\_da\_tabela”

**Alterar linhas de forma que fique igual à outra tabela** = UPDATE nome\_da\_tabela A SET A.coluna = (SELECT B.coluna FROM tabela\_fonte B WHERE A.identificador = B.identificador) WHERE EXISTS (SELECT 1 FROM tabela\_fonte B WHERE A.identificador = B.identificador);

**Excluir linhas de forma que fique iguam a uma tabela base** = DELETE FROM nome\_da\_tabela WHERE coluna NOT IN (SELECT coluna FROM tabela\_base);

**Criar uma tabela igual à outra, com os mesmos dados** = CREATE TABLE tabela2 AS (SELECT \* FROM tabela1);

**Desfazer os comandos não comitados** = ROLLBACK;

**Criar uma coluna de autoincremento** (preenche os valores sozinha de forma numérica e crescente, sempre acrescenta +1 em cima do último valor adicionado) = CREATE TABLE nome\_da\_tabela (coluna NUMBER GENERATED BY DEFAULT ON NULL AS IDENTITY, DESCRITOR VARCHAR(20), PRIMARY KEY (coluna));

PS: Só pode haver uma por tabela (é possível, mas não é recomendável forçar um valor diferente)

**Forçar valor em casos predefinidos** = CREATE TABLE nome\_da\_tabela (coluna VARCHAR(10) DEFAULT ON NULL ‘Texto de exemplo’);

**Criar um procedimento automático quando algo acontecer** = CREATE TRIGGER TG\_NOME\_DA\_TRIGGER AFTER INSERT OR UPDATE OR DELETE ON nome\_da\_tabela BEGIN ....comando.... END; (vai realizar o comando depois de um insert ou update ou delete)

PS: pode ser AFTER (depois), BEFORE (antes) ou INSTEAD OF (em vez de)