Monitoria GDI
Aula Prática

DML + PL/SQL
parte 1

DML

linguagem de manipulação de dados

SQL

- **Estrutura básica de uma consulta SQL**

```
SELECT Coluna1[,Coluna2[,Coluna3[,...]]]
FROM Tabela1[,Tabela2[,...]]
WHERE Condição
```

SQL

- **Estrutura genérica de uma consulta SQL**

```
SELECT [DISTINCT|ALL] {*|[Tabela.]Coluna1 [AS Alias1]
                [[Tabela.]Coluna2 [AS Alias2] [,...]]}
FROM Tabela1[,Tabela2[,...]]
[WHERE {Condição Simples|Condição de Sub-consulta}]
[ORDER BY Coluna1 [ASC|DESC] [,Coluna2 [ASC|DESC] [, ... ]]]
[GROUP BY Coluna1 [,Coluna2[,...]] [HAVING Condição]]
[{UNION|INTERSECT|EXCEPT} SELECT ...]
```

Exercício 1

- **Selecione a matrícula e o nome de todas as mulheres, ordenando-as por ordem alfabética.**

Exercício 2

- **Repita a operação anterior exibindo apenas aquelas que são professoras.**

Exercício 3

- **Quais são as disciplinas que o professor 'Sirenio Arruda' está ministrando atualmente?**

Exercício 4

- **Repita a consulta anterior utilizando JOIN.**

Exercício 5

- **Para as disciplinas de código 1, 2 e 3, mostre quais alunos já foram seus monitores. (Use IN)**

Exercício 6

- **Mostre os nomes de TODOS os professores e, caso existam, os nomes dos seus líderes.**

Exercício 7

- **Mostre os alunos que não têm nenhum projeto. Exiba também as informações de quando eles pagaram a cadeira. (Use IS NULL)**

Exercício 8

- **Selecione todos os professores, exceto aqueles que entraram entre 1995 e 2005. (Use BETWEEN)**

Exercício 9

- **Mostre quantas vezes que o professor 'Jose Alcantara' já esteve a lecionar**

Exercício 10

- **Mostre a média das notas dos alunos agrupadas por período.**

Exercício 11

- **Considere um relatório e mostre, numa mesma consulta, para o semeste '2009.1', os registros dos professores em todas as ministrações que realizaram mais os registros dos alunos nas vezes em que pagaram alguma cadeira. Exiba o código da disciplina, o código do curso e a matrícula do professor ou do aluno que realizou a atividade. (Realize SELECTS independentes e use UNION)**

PL/SQL

Procedural Language/ Structured Query Language

PROCEDURE

- **Por padrão não retornam valor (exceção: modo OUT ou IN OUT).**

- **Estrutura básica de um PROCEDURE**

```
PROCEDURE nome IS
BEGIN
   [EXCEPTION]
END;
```

FUNCTION

- **Por padrão, necessariamente, retornam um único valor.**

- **Estrutura básica de uma FUNCTION**

```
FUNCTION nome RETURN tipo IS
BEGIN
   RETURN valor
   [EXCEPTION]
END;
```

Exercício 12

- **Admita que cada uma das cadeiras que um aluno paga vale 5 créditos, que cada projeto vale 1 e que cada monitoria vale 2 créditos.**
- **Implemente uma função que, dado um número de matrícula, retorna os créditos totais da carreira estudantil do aluno.**

Exercício 13

- **Implemente um procedimento que recebe como parâmetro de entrada um título de um projeto e imprime os seus dados.**

Exercício 14

- **Implemente um novo procedimento, semelhante ao anterior, que seja mais genérico e pesquise todos os projetos que possuam o valor do parâmetro como substring do seu título. (Utilize LIKE '%' e CURSOR)**

Exercício 15

- **Crie um PROCEDURE que recebe um VARCHAR do tipo ano_semestre e produz dois parâmetros numéricos de saída: ano e semestre;**

Exercício 16

- **Implemente uma FUNCTION que receberá o código de uma disciplina e retornará uma STRING com todos os ANOS em que ela foi ofertada no 1º semestre e todos os anos para o 2º semestre (EX: '1º: 1992; 1990; 2000; 2º: 1990; 2001;').**
- **Crie uma tabela (IS TABLE OF) com registros do tipo (IS RECORD [cod_curso, ano, semestre]) que receberá as informações de todas as turmas que já existiram e utilize o PROCEDURE anterior para separar os campos ano_semestre.**
- **Em seguida, verifique um a um os registros da tabela já povoada e vá preenchendo a variável de retorno.**

TRIGGER

- **Executado implicitamente pelo SGBD na ocorrência de um determinado evento ou combinação deste.**

- **Estrutura básica de um TRIGGER**

```
CREATE [OR REPLACE] TRIGGER nome_trigger
      momento evento1 [OR evento2 OR evento3]
       [OF coluna] ON nome_objeto
[[REFERENCING OLD AS apelido1 | NEW AS apelido2]

FOR EACH ROW

[WHEN (condição)]]

corpo_trigger
```

Exercício 17

- **Criar um TRIGGER que faça um comparativo entre os ANTIGOS e NOVOS valores logo após inserção, atualização ou deleção de um projeto.**

Exercício 18

- **Implemente um TRIGGER que não permita que um professor coordene mais do que uma disciplina. Caso alguma irregularidade ocorra, imprima uma mensagem do tipo "RAISE APPLICATION ERROR".**

Monitoria GDI
Aula Prática

DML + PL/SQL
parte 1