Checklist 2 - OR - GDI EC 2018.1  
Orientações:  
1. Os alunos devem indicar que itens são abordados em cada trecho de código. É possível  
apresentar um só trecho de código que contenha mais de um item.  
2. É necessário que os alunos saibam explicar exatamente o que cada trecho de código faz (não  
podem ser coisas absurdas!); caso contrário, não será considerada a pontuação do item.  
3. As consultas SQL devem sempre retornar pelo menos uma linha.  
4. Código PL​ (trigger, função ou procedimento) deve estar acompanhado de um trecho de código  
que o execute/teste.  
Atividades  
1. Criação de tipo e subtipo  
2. Criação de um tipo que contenha um atributo que seja de um outro tipo  
3. Criação de um tipo que contenha um atributo que seja de um tipo VARRAY  
4. Criação de um tipo que contenha um atributo que seja de um tipo NESTED TABLE  
5. Criação e chamada de um método construtor (diferente do padrão)  
6. Criação e chamada de um função membro em um comando SELECT e em um bloco PL  
7. Criação e chamada de um método MAP em um comando SELECT e em um bloco PL  
8. Criação e chamada de um método ORDER em um comando SELECT e em um bloco PL  
9. Criação e chamada de método abstrato  
10. Redefinição de método do supertipo dentro do subtipo  
11. Alteração de tipo: adição de atributo  
12. Alteração de tipo: modificação de atributo  
13. Alteração de tipo: remoção de atributo  
14. Alteração de supertipo com propagação de mudança  
15. Alteração de supertipo com invalidação de subtipos afetados  
16. Uso de referência e controle de integridade referencial  
17. Restrição de escopo de referência  
18. Criação de todas as tabela a partir de um tipo

Todas as tabelas são criadas a partir de tipos predefinidios  
19. Criação de uma consulta com expressão de caminho para percorrer três tabelas  
20. Criação de uma consulta com DEREF  
21. Criação de uma consulta com VALUE  
22. Criação de uma consulta com TABLE  
23. Criação de consultas com LIKE, BETWEEN, ORDER BY, GROUP BY, HAVING

--ordena toda a tabela de acordo com o nome do curso

SELECT \* FROM TB\_CURSO ORDER BY NOME;

--exibe 2 ou mais investimentos que sao acima de 1000.Nada eh exibido se houver apenas 1 investimento

--com o valor acima de 1000.

SELECT investimento, COUNT(\*) FROM tb\_projeto WHERE investimento > 1000

GROUP BY investimento HAVING COUNT(\*) > 1;

INSERT INTO TB\_Projeto VALUES (44,(SELECT REF(P) FROM TB\_Professor P WHERE P.cpf\_professor = 44), 'ac', 'alguma coisa', 15000.00);

--exibe nome do aluno que tem o seu cpf entre os valores determinados e tenha seu nome iniciado com S

SELECT nome

FROM tb\_aluno

WHERE (cpf\_aluno BETWEEN 2000 and 6000) AND nome LIKE 'S%';

24. Criação de subconsultas com IN , ALL, ANY,

SELECT A.CPF\_PROFESSOR

FROM TB\_PROJETO A

WHERE (A.INVESTIMENTO > ALL(1000)) AND (A.TITULO) IN ('GIT');

SELECT A.CPF\_aluno

FROM tb\_aluno A

WHERE A.REF\_Curso.codigo = any(select codigo from tb\_curso where nota\_mec > 5);

25. Criação de uma consulta que exiba os dados de um VARRAY  
26. Criação de uma consulta que exiba os dados de um NESTED TABLE

27. SELECT para acessar os dados de uma tabela A​ utilizando uma tabela B​ dentro da cláusula EXISTS, onde a tabela A​ tem uma referência para a tabela B​.  
 --Exibe os CPF dos professores que tem mais de 10000 investido em seus projetos.

select a.cpf\_professor from tb\_professor a

where exists (select \* from tb\_projeto b

where b.cpf\_professor = a.cpf\_professor and investimento > 10000);

28. Criação de TRIGGER de linha ao ocorrer um INSERT, DELETE ou UPDATE

--Se adicionar um professor com um cpf que ja existe na tabela, ira mostrar a mensagem

CREATE OR REPLACE TRIGGER verificar

BEFORE INSERT ON tb\_professor

FOR EACH ROW

DECLARE

cont NUMBER;

BEGIN

SELECT COUNT(\*) INTO cont FROM tb\_professor P WHERE P.CPF\_PROFESSOR = :NEW.CPF\_PROFESSOR;

IF(cont>0) THEN

RAISE\_APPLICATION\_ERROR(-20020,'CPF ja cadastrado');

END IF;

END verificar; /

29. Criação de TRIGGER de linha para impedir INSERT, DELETE ou UPDATE

--Não será permitido atualizar o valor do investimento se este for menor que o valor atual.

CREATE OR REPLACE TRIGGER verificar\_invest

BEFORE UPDATE ON tb\_projeto

FOR EACH ROW

WHEN(NEW.investimento < OLD.investimento)

DECLARE

BEGIN

RAISE\_APPLICATION\_ERROR(-20020,'Investimento nao pode ser reduzido');

END verificar\_invest;

/

CREATE OR REPLACE TRIGGER impedir\_del

BEFORE DELETE ON tb\_projeto

FOR EACH ROW

WHEN(CPF\_PROFESSOR = 22)

DECLARE

BEGIN

RAISE\_APPLICATION\_ERROR(-20020,’Este projeto nao pode ser deletado pois esta responsavel pelo professor cujo cpf eh 22');

END impedir\_del;;

/

30. Criação de TRIGGER de comando para impedir INSERT, DELETE ou UPDATE

-- Impede qualquer tipo de alteração na tabela Licenciatura

*CREATE OR REPLACE TRIGGER Impedir\_Lic BEFORE*

*UPDATE OR DELETE OR INSERT ON tb\_Licenciatura*

*DECLARE*

*BEGIN*

*IF INSERTING THEN*

*RAISE\_APPLICATION\_ERROR(-2010, 'Proibido inserir cursos de Licenciatura');*

*ELSIF UPDATING THEN*

*RAISE\_APPLICATION\_ERROR(-2011, 'Proibido atualizar cursos de Licenciatura');*

*ELSIF DELETING THEN*

*RAISE\_APPLICATION\_ERROR(-2012, 'Proibido deletar cursos de Licenciatura');*

*END IF;*

*END Impedir\_Lic;*

*/*