



Sistemas de Informação

Prof. Me. Alexandre Barcelos
profalexandre.barcelos@fiap.com.br



Manipulação de Dados com o PL/SQL

Objetivos

- Avaliar quais instruções SQL podem ser incluídas diretamente em um bloco executável PL/SQL
- Manipular dados com instruções DML no PL/SQL
- Utilizar instruções de controle de transações em PL/SQL
- Utilizar a cláusula INTO para manter os valores retornados por uma instrução SQL
- Diferenciar cursores implícitos e cursores explícitos
- Utilizar atributos de cursor SQL

Nesta lição, você aprenderá a incorporar instruções INSERT, UPDATE, DELETE e SELECT SQL padrões nos blocos PL/SQL. Você também aprenderá como controlar transações e determinar o resultado das instruções DML SQL em PL/SQL.

Instruções SQL no PL/SQL

- Recuperar uma linha do banco de dados usando o comando **SELECT**.
- Alterar linhas no banco de dados utilizando comandos **DML**.
- Controlar uma transação com o comando **COMMIT**, **ROLLBACK**, ou **SAVEPOINT command**.

Visão Geral

Quando você precisar extrair informações ou aplicar alterações aos bancos de dados, você deverá usar o SQL. O PL/SQL suporta integralmente a linguagem de manipulação de dados e os comandos de controle de transação no SQL. Você poderá usar as instruções **SELECT** para preencher as variáveis com os valores consultados em uma linha na tabela. Seus comandos **DML** (manipulação de dados) podem processar várias linhas.

Comparando os Tipos de Instrução SQL e PL/SQL

- Um bloco PL/SQL não é uma unidade de transação. Os comandos **COMMIT**, **SAVEPOINT** e **ROLLBACK** são independentes dos blocos, mas você pode emitir esses comandos em um bloco.
- O PL/SQL não suporta instruções em **DDL** (Data Definition Language) como, por exemplo, **CREATE TABLE**, **ALTER TABLE** ou **DROP TABLE**. Utilize o comando **EXECUTE IMMEDIATE** para executar as instruções **DDL**.
- O PL/SQL não suporta instruções em **DCL** (Data Control Language) como, por exemplo, **GRANT** ou **REVOKE**. Utilize o comando **EXECUTE IMMEDIATE** para executar as instruções **DCL**.

Exemplo:

```
BEGIN
CREATE TABLE My_emp_table AS SELECT * FROM employees;
END;
/
create table My_table as select * from table_name; * ERROR
at line 5:
ORA-06550: line 5, column 1:
PLS-00103: Encountered the symbol "CREATE" when
expecting one of the following:
```

...

Utilize o comando EXECUTE IMMEDIATE:

```
BEGIN
EXECUTE IMMEDIATE 'CREATE TABLE My_emp_table AS
SELECT * FROM employees';
END;
/
```

Instruções SELECT no PL/SQL

Recuperar dados do banco de dados com uma instrução SELECT.

Sintaxe:

```
SELECT select_list
INTO {variable_name[, variable_name]...
    | record_name}
FROM table
[WHERE condition];
```

Recuperando Dados em PL/SQL

Use a instrução SELECT para recuperar dados no banco de dados.

Na sintaxe:

- `select_list`: é uma lista de pelo menos uma coluna e pode incluir funções de linhas, de grupo ou expressões SQL
- `variable_name`: é a variável escalar para armazenar o valor recuperado
- `record_name`: é o registro PL/SQL para armazenar os valores recuperados
- `tabela`: especifica o nome da tabela do banco de dados
- `condição`: é composta de nomes de coluna, expressões, constantes e operadores de comparação, incluindo as variáveis PL/SQL e operadores

Aproveite a faixa completa de sintaxe do Oracle Server para a instrução SELECT.

Lembre-se que as variáveis de host devem ter dois-pontos como prefixo.

Instruções SELECT no PL/SQL

- A cláusula INTO é obrigatória.
- As consultas devem retornar apenas uma linha.

Exemplo:

```
SET SERVEROUTPUT ON
DECLARE
    fname VARCHAR2(25);
BEGIN
    SELECT first_name
    INTO fname
    FROM employees WHERE employee_id=200;
    DBMS_OUTPUT.PUT_LINE(' First Name is : '||fname);
END;
/
```

Cláusula INTO

A cláusula INTO é mandatória e ocorre entre as cláusulas SELECT e FROM. Ela é usada para especificar os nomes das variáveis que armazenarão os valores que o SQL retorna a partir da cláusula SELECT. Você deve oferecer uma variável para cada item selecionado e a ordem delas deve corresponder aos itens selecionados

Você deve usar a cláusula INTO para preencher as variáveis PL/SQL ou as variáveis de host.

As Consultas Devem Retornar Apenas Uma Linha

As instruções SELECT em um bloco PL/SQL caem na classificação ANSI de SQL Embutido (Embedded SQL), para a qual se aplicam as regras a seguir: as consultas devem retornar apenas uma linha. Mais de uma ou nenhuma linha geram mensagens de erro. O PL/SQL lida com essas mensagens de erro destacando as exceções padrão, as quais você poderá capturar na seção de exceções do bloco com as exceções NO_DATA_FOUND e TOO_MANY_ROWS (o tratamento de exceções é abordado em uma

lição posterior). Você deverá codificar as instruções SELECT para retornar uma única linha.

Recuperando Dados no PL/SQL

Selecione o hire_date e o salary para o empregado específico.

Exemplo:

```
DECLARE
  emp_hiredate    employees.hire_date%TYPE;
  emp_salary      employees.salary%TYPE;
BEGIN
  SELECT  hire_date, salary
  INTO    emp_hiredate, emp_salary
  FROM    employees
  WHERE   employee_id = 100;
END;
/
```

Diretrizes

Siga essas diretrizes para recuperar os dados em PL/SQL:

- Finalize cada instrução SQL com um ponto-e-vírgula (;).
- A cláusula INTO é obrigatória para a instrução SELECT quando ela está embutida no PL/SQL.
- A cláusula WHERE é opcional e poderá ser usada para especificar variáveis de entrada, constantes, literais ou expressões PL/SQL.
- Especifique o mesmo número de variáveis de saída na cláusula INTO como colunas de banco de dados na cláusula SELECT. Certifique-se de que elas correspondam em posição e que os seus tipos de dados sejam compatíveis.

Recuperando Dados no PL/SQL

Selecione o somatório dos salários para todos os funcionários de um departamento específico.

Exemplo:

```
SET SERVEROUTPUT ON
DECLARE
    sum_sal  NUMBER(10,2);
    deptno   NUMBER NOT NULL := 60;
BEGIN
    SELECT SUM(salary) -- group function
    INTO sum_sal FROM employees
    WHERE department_id = deptno;
    DBMS_OUTPUT.PUT_LINE ('The sum of salary is '
    || sum_sal);
END;
/
```

1-10

Diretrizes (continuação)

- Para garantir que os tipos de dados dos identificadores correspondam aos tipos de dados das colunas, use o atributo %TYPE. O tipo de dados e o número de variáveis na cláusula INTO correspondem àqueles na lista SELECT.
- Use as funções de grupo como, por exemplo, SUM, em uma instrução SQL, porque as funções de grupo se aplicam a grupos de linhas em uma tabela.

Observação: As funções de grupo não poderão ser usadas na sintaxe do PL/SQL. Elas são usadas em instruções SQL em um bloco PL/SQL.

Convenções de Nomenclatura

```
DECLARE
  hire_date      employees.hire_date%TYPE;
  sysdate        hire_date%TYPE;
  employee_id     employees.employee_id%TYPE :=
176;
BEGIN
  SELECT      hire_date, sysdate
  INTO        hire_date, sysdate
  FROM        employees
  WHERE       employee_id = employee_id;
END;
/
```

DECLARE

*

ERROR at line 1:

ORA-01422: exact fetch returns more than requested number of rows

ORA-06512: at line 6

Convenções de Nomenclatura

- Use a convenção de nomenclatura para evitar ambiguidade na cláusula `WHERE`.
- Evite usar nomes de *coluna* de banco de dados como identificadores.
- Os nomes de variáveis locais e parâmetros têm precedência sobre os nomes de tabelas de banco de dados.
- Os nomes das colunas têm precedência sobre os nomes das variáveis locais.

Convenções para Nomeação

Evite a ambiguidade na cláusula `WHERE` aderindo a uma convenção de nomeação que distingue os

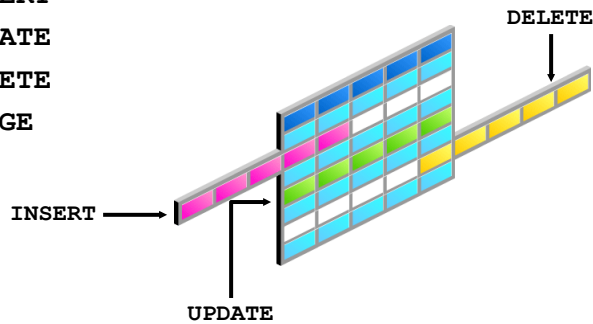
nomes de coluna do banco de dados dos nomes de variável PL/SQL.

- Os identificadores e as colunas de banco de dados devem ter nomes diferentes.
- Podem surgir erros de sintaxe, pois o PL/SQL verifica primeiro a coluna na tabela.

Manipulando dados usando PL/SQL

Faça alterações nas tabelas do banco de dados usando os comandos DML:

- INSERT
- UPDATE
- DELETE
- MERGE



Manipulando Dados Usando o PL/SQL

Você poderá manipular os dados no banco de dados usando os comandos DML (manipulação de dados). Você poderá emitir os comandos DML, por exemplo, INSERT, UPDATE e DELETE sem restrições no PL/SQL. A inclusão das instruções COMMIT ou ROLLBACK no código PL/SQL libera os bloqueios de linha (e bloqueios de tabela).

- A instrução INSERT adiciona novas linhas de dados à tabela.
- A instrução UPDATE modifica linhas existentes na tabela.
- A instrução DELETE remove linhas não desejadas da tabela.

Inserindo Dados

**Adicione um novo empregado à tabela EMPLOYEES.
Exemplo:**

```
BEGIN
INSERT INTO employees
(employee_id, first_name, last_name, email,
hire_date, job_id, salary)
VALUES(employees_seq.NEXTVAL, 'Ruth', 'Cores',
'RCORES',sysdate, 'AD_ASST', 4000);
END;
/
```

Inserindo Dados

- Use as funções SQL como, por exemplo, USER e SYSDATE.
- Gere valores de chaves primárias usando as seqüências de banco de dados.
- Crie valores no bloco PL/SQL.
- Adicione valores default de coluna.

Observação: Não há possibilidade de ambigüidade entre os identificadores e nomes de coluna na instrução INSERT. Qualquer identificador na cláusula INSERT deve ser um nome de coluna do banco de dados.

Atualizando Dados

Aumentar o salário de todos os funcionários que tem o cargo de stock clerks.

Exemplo:

```
DECLARE
    sal_increase    employees.salary%TYPE := 800;
BEGIN
    UPDATE          employees
    SET              salary = salary + sal_increase
    WHERE            job_id = 'ST_CLERK';
END;
```

Atualizando e Deletando Dados

Poderá haver ambigüidade na cláusula SET da instrução UPDATE porque, embora o identificador à esquerda do operador de atribuição seja sempre uma coluna de banco de dados, o identificador à direita podem ser uma coluna de banco de dados ou uma variável PL/SQL.

Lembre-se de que a cláusula WHERE é usada para determinar que linhas são afetadas. Se nenhuma linha for modificada, não ocorrerão erros, diferente da instrução SELECT no PL/SQL.

Observação: As atribuições de variável PL/SQL sempre usam :=, e as atribuições de coluna SQL sempre usam =. Lembre-se de que os nomes de coluna e de identificadores são idênticos na cláusula WHERE, o Oracle Server procura primeiro pelo nome no banco de dados.

Deletando Dados

Exclua as linhas que pertencem ao departamento 10 da tabela empregados.

Exemplo:

```
DECLARE
    deptno employees.department_id%TYPE := 10;
BEGIN
    DELETE FROM employees
    WHERE department_id = deptno;
END;
/
```

Excluindo dados

A instrução DELETE remove linhas de uma tabela. Se a cláusula WHERE não for usada, todas as linhas em uma tabela poderão ser removidas, desde que não haja restrições de integridade.

Cursor SQL

- Um cursor é um ponteiro para a área de memória privada alocada pelo servidor Oracle.
- Existem dois tipos de cursores:
 - Cursores implícitos: Criado e gerenciado internamente pelo servidor Oracle para processar instruções SQL
 - Cursores explícitos: declarado explicitamente pelo programador

Cursor SQL

Sempre que você emitir uma instrução SQL, o Oracle Server abrirá uma área de memória na qual o comando é analisado e executado. Essa área é chamada de cursor. Quando a parte executável de um bloco emite uma instrução SQL, o PL/SQL cria um cursor implícito, o qual tem o identificador SQL. O PL/SQL gerencia esse cursor automaticamente. O programador declara e nomeia um cursor explícito. Há quatro atributos disponíveis no PL/SQL que poderão aplicar-se aos cursores.

Observação: Mais informações sobre os cursores explícitos são abordadas em uma lição subsequente.

Atributos para Cursores Implícitos

Usando atributos de cursor SQL, você pode testar o resultado de suas instruções SQL.

SQL%FOUND	Atributo booleano que é avaliado como TRUE se a instrução SQL mais recente retornou pelo menos uma linha.
SQL%NOTFOUND	Atributo booleano que é avaliado como TRUE se a instrução SQL mais recente não retornar mesmo uma linha.
SQL%ROWCOUNT	Um valor inteiro que representa o número de linhas afetadas pela instrução SQL mais recente.

Atributos do Cursor SQL

Os atributos do cursor SQL permitem que você avalie o que aconteceu quando o cursor implícito foi usado pela última vez. Você deve usar esses atributos nas instruções PL/SQL como, por exemplo, as funções. Você não poderá usá-las em instruções SQL.

Você poderá usar os atributos SQL%ROWCOUNT, SQL%FOUND, SQL%NOTFOUND e SQL%ISOPEN na seção de exceção de um bloco para coletar informações sobre a execução de uma

instrução de manipulação de dados. Ao contrário da instrução SELECT, que retorna uma exceção, o PL/SQL não considera uma instrução DML que não afete linhas onde ocorreram falhas.

Atributos para Cursores Implícitos

Exclua as linhas que têm o ID de funcionário especificado da tabela de funcionários. Exiba o número de linhas excluídas.

Exemplo:

```
VARIABLE rows_deleted VARCHAR2(30)
DECLARE
    empno employees.employee_id%TYPE := 176;
BEGIN
    DELETE FROM employees
    WHERE employee_id = empno;
    :rows_deleted := (SQL%ROWCOUNT || ' row deleted. ');
END;
/
PRINT rows_deleted
```

Atributos do Cursor SQL

Remova o funcionário de número 176. Ao usar o atributo SQL%ROWCOUNT, você poderá imprimir o número de linhas deletadas.

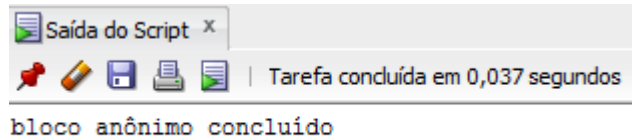
+ Exercícios: Visão Geral

Esta prática abrange os seguintes tópicos:

- Selecionando dados de uma tabela
- Inserindo dados em uma tabela
- Atualizando dados em uma tabela
- Excluindo uma linha de uma tabela

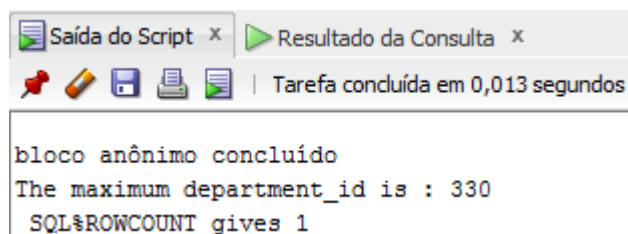
Exercícios

1. Crie um bloco PL/SQL que seleciona o número máximo de departamento na tabela DEPARTMENTS e o armazena em uma variável. Imprima os resultados na tela. Salve o bloco PL/SQL em um arquivo nomeado lab_03_01_soln.sql.



```
bloco anônimo concluído
```

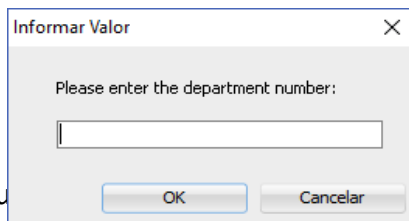
2. Modifique o bloco (The maximum department_id is : 330 03_01_soln.sql) para inserir um novo departamento na tabela DEPT. Salve o bloco PL/SQL em um arquivo nomeado (lab_03_02_soln.sql).
 - a. Em vez de imprimir o número do departamento recuperado do exercício 1, adicione 10 a ele (Se o número retornado foi 330 então adicione 10) e use-o como o número do departamento do novo departamento. O novo departamento deve ser definido como: EDUCATION
 - b. Use uma variável de substituição do tipo BIND VARIABLE para definir o número do departamento novo departamento.
 - c. Deixe um valor nulo nas colunas como MANAGER_ID e como o LOCATION_ID do novo departamento.
 - d. Execute o bloco PL/SQL.
 - e. Exiba as seguintes mensagens como a seguir demonstrado. (Utilize o atributo de cursor SQL%ROWCOUNT para exibir a quantidade de linhas manipuladas.)



```
bloco anônimo concluído
The maximum department_id is : 330
SQL%ROWCOUNT gives 1
```

Exercícios (continuação)

3. Crie um bloco PL/SQL que atualize a localização (location_id) para um departamento existente (department_id) (Tabela: DEPARTMENTS). Salve o bloco PL/SQL em um arquivo denominado lab_03_03_soln.sql.
- a. Use uma variável de substituição (VARIABLE ou ACCEPT) para informar o número de departamento



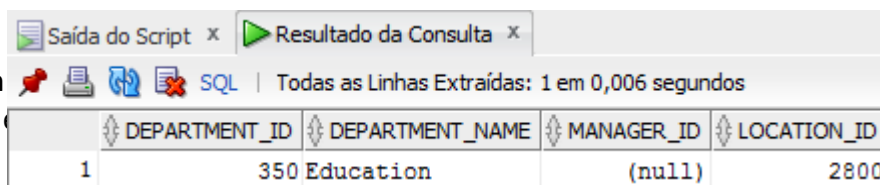
Informar Valor

Please enter the department number:

OK Cancelar

- b. Use uma variável de substituição (VARIABLE ou ACCEPT) para informar a localização de departamento. (Consulte a tabela LOCATIONS para ter um número de localização válido)
- c. Teste o bloco PL/SQL.
- d. Exiba o nome e o número do departamento, além da localização do departamento atualizado.

4. Crie um bloco PL/SQL que atualize a localização (location_id) para um departamento existente (department_id) (Tabela: DEPARTMENTS). Salve o bloco PL/SQL em um arquivo denominado lab_03_03_soln.sql.

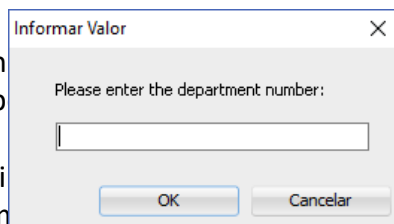


Saída do Script x Resultado da Consulta x

Todas as Linhas Extraídas: 1 em 0,006 segundos

	DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
1	350	Education	(null)	2800

- a. Imprima o número de linhas atualizadas e exiba essa mensagem após a execução do bloco PL/SQL.
- b. Teste o bloco PL/SQL.
- c. O que acontece se você informar um departamento inexistente?
- d. Confirme que o departamento atualizado.



Informar Valor

Please enter the department number:

OK Cancelar

OBRIGADO



profalexandre.barcelos@fiap.com.br



<https://www.linkedin.com/in/alexandrebarcelos>

FIAP

Copyright © 2023 | Professor Me. Alexandre Barcelos
Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento, é expressamente
proibido sem consentimento formal, por escrito, do professor/autor.

