

FIAP GRADUAÇÃO

Tecnologia em Análise e Desenvolvimento de Sistemas

Application Development For Databases

PROF. MILTON

Usando Cursores Explícitos

Objetivos

Ao concluir esta lição, você será capaz de:

- Fazer distinção entre cursores implícitos e explícitos
- Discutir os motivos para usar cursores explícitos
- Declarar e controlar cursores explícitos
- Usar os loop simples e os loops `FOR` de cursores para extrair dados com o comando `fetch`
- Declarar e usar cursores com parâmetros
- Bloquear as linhas com a cláusula `FOR UPDATE`
- Fazer referência à linha atual com a cláusula `WHERE CURRENT OF`

Objetivos

Você aprendeu sobre os cursores implícitos que são criados automaticamente pelo código PL/SQL quando você executa uma instrução SQL `SELECT` ou DML. Nesta lição, você conhecerá os cursores explícitos. Você aprenderá a diferenciar os cursores implícitos dos explícitos. Também será ensinado a declarar e controlar cursores simples, bem como cursores com parâmetros.

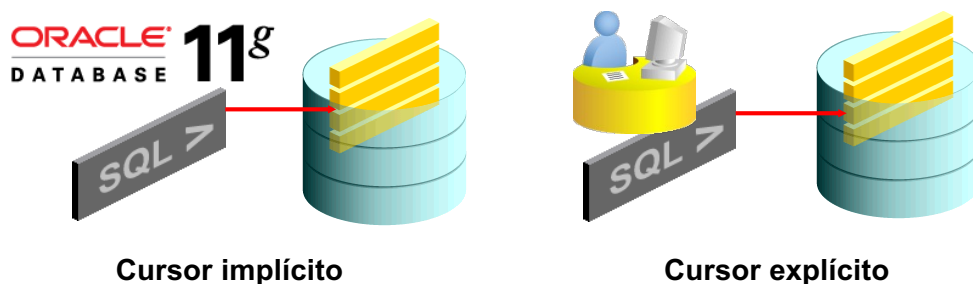
Agenda

- O que são cursores explícitos?
- Usando cursores explícitos
- Usando cursores com parâmetros
- Bloqueando linhas e fazendo referência à linha atual

Cursors

Cada instrução SQL que é executada pelo Oracle Server tem um cursor individual associado:

- Cursores implícitos: declarados e gerenciados pelo código PL/SQL em todas as instruções `SELECT` de DML e PL/SQL
- Cursores explícitos: declarados e gerenciados pelo programador



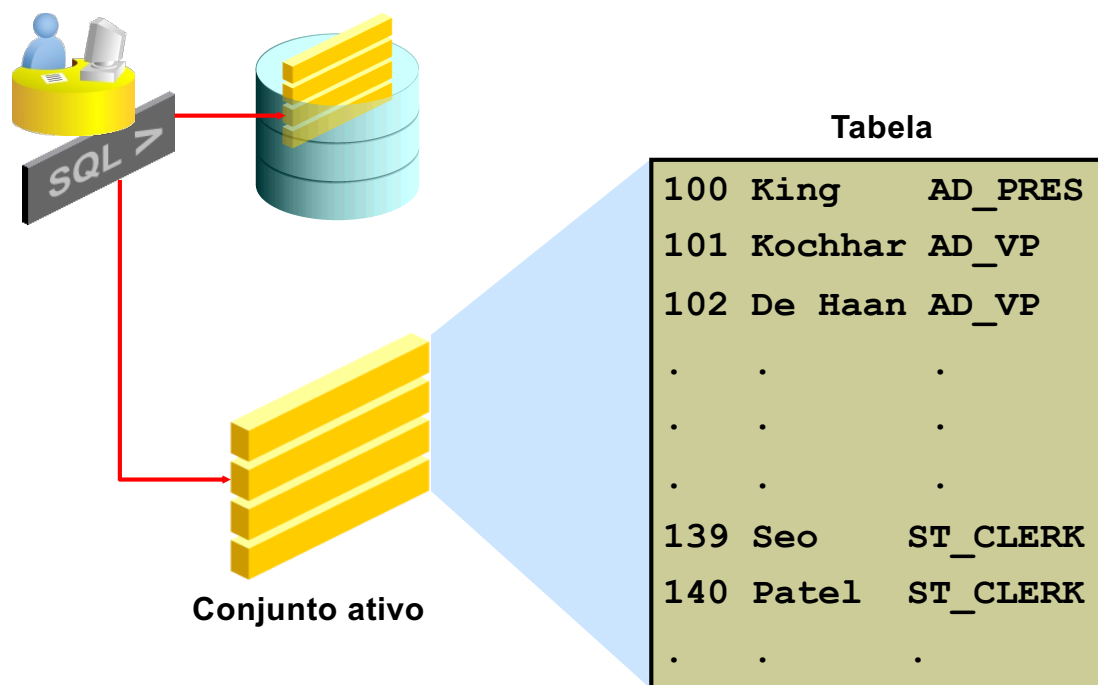
Cursors

O Oracle Server usa áreas de trabalho (chamadas de *áreas SQL privadas*) para executar instruções SQL e armazenar informações sobre o processamento. Você pode usar cursores explícitos para nomear uma área SQL privada e para acessar as informações armazenadas nela.

Tipo de Cursor	Descrição
Implícito	Os cursores implícitos são declarados por PL/SQL implicitamente para todas as instruções DML e PL/SQL <code>SELECT</code> .
Explícito	Para consultas que retornam mais de uma linha, os cursores explícitos são declarados e gerenciados pelo programador e manipulados por meio de instruções específicas nas ações executáveis do bloco.

O Oracle Server abre implicitamente um cursor para processar cada instrução SQL não associada a um cursor declarado explicitamente. Com o código PL/SQL, é possível fazer referência ao cursor implícito mais recente como o cursor `SQL`.

Operações com Cursores Explícitos



Operações com Cursores Explícitos

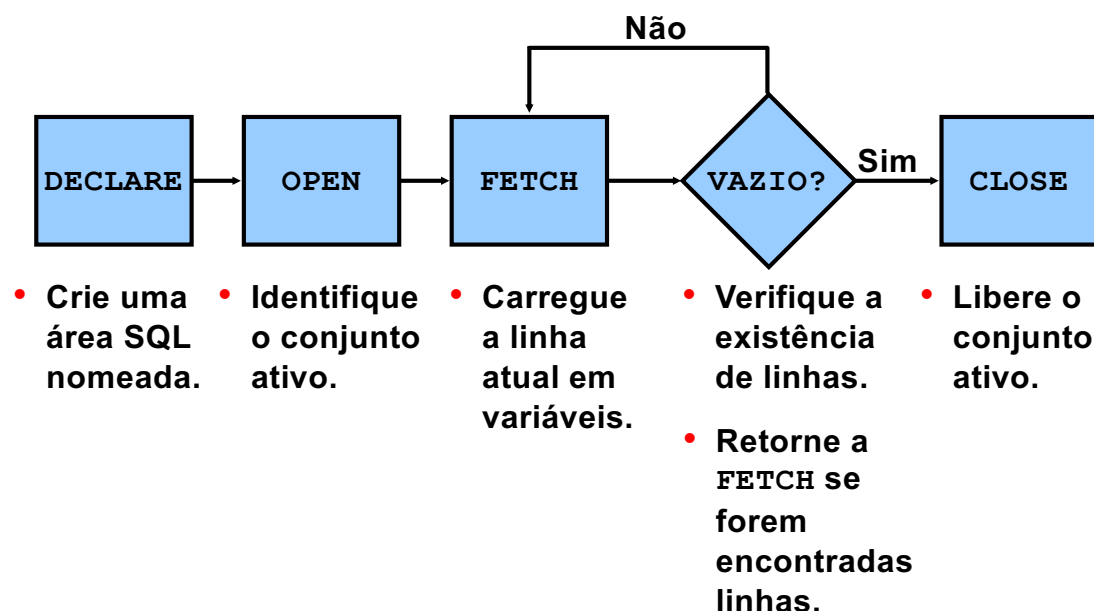
Declare cursores explícitos no código PL/SQL quando você quiser que uma instrução `SELECT` retorne várias linhas. Você pode processar cada linha retornada pela instrução `SELECT`.

O conjunto de linhas retornadas por uma consulta de várias linhas é denominado *conjunto ativo*. Seu tamanho é o número de linhas que satisfaz seus critérios de pesquisa. O diagrama no slide mostra como um cursor explícito “aponta” para a linha atual do conjunto ativo. Isso permite que seu programa processe as linhas uma de cada vez.

Funções de cursores explícitos:

- Podem executar o processamento, linha por linha, além da primeira linha retornada por uma consulta
- Controlam a linha que está sendo processada no momento
- Permitem que o programador controle manualmente os cursores explícitos no bloco PL/SQL

Controlando Cursores Explícitos

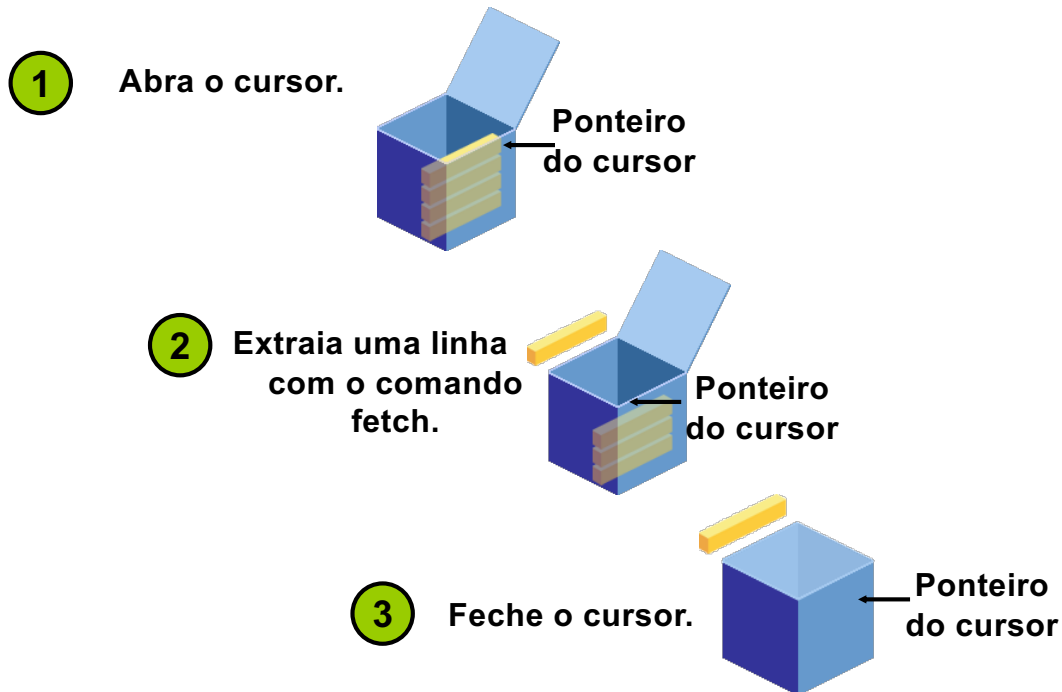


Controlando Cursores Explícitos

Agora que você conhece o conceito de cursor, verifique as etapas para utilizá-los.

1. Declare o cursor na seção declarativa de um bloco PL/SQL, nomeando e definindo a estrutura da consulta a ser associada a ele.
2. Abra o cursor.
A instrução OPEN executa a consulta e associa todas as variáveis referenciadas. As linhas identificadas pela consulta são chamadas de *conjunto ativo* e estão disponíveis para extração com a instrução FETCH.
3. Extraia dados do cursor.
No diagrama de fluxo mostrado no slide, após cada fetch, teste o cursor em qualquer linha existente. Se não houver mais linhas para processar, você deverá fechar o cursor.
4. Feche o cursor.
A instrução CLOSE libera o conjunto de linhas ativo. Agora é possível reabrir o cursor para estabelecer um novo conjunto ativo.

Controlando Cursores Explícitos



Controlando Cursores Explícitos (continuação)

Um programa PL/SQL abre um cursor, processa as linhas retornadas por uma consulta e fecha o cursor. O cursor marca a posição atual no conjunto ativo.

1. A instrução `OPEN` executa a consulta associada ao cursor, identifica o conjunto ativo e posiciona o cursor na primeira linha.
2. A instrução `FETCH` recupera a linha atual e avança o cursor para a próxima linha até que não existam mais linhas ou que uma condição especificada seja atendida.
3. A instrução `CLOSE` libera o cursor.

Agenda

- O que são cursores explícitos?
- **Usando cursores explícitos**
- Usando cursores com parâmetros
- Bloqueando linhas e fazendo referência à linha atual

Declarando o Cursor

Sintaxe:

```
CURSOR cursor_name IS  
    select_statement;
```

Exemplos:

```
DECLARE  
    CURSOR c_emp_cursor IS  
    SELECT employee_id, last_name FROM employees  
    WHERE department_id =30;
```

```
DECLARE  
    v_locid NUMBER:= 1700;  
    CURSOR c_dept_cursor IS  
    SELECT * FROM departments  
    WHERE location_id = v_locid;  
    ...
```

11

Declarando o Cursor

A sintaxe para declarar um cursor é mostrada no slide. Na sintaxe:

cursor_name É um identificador PL/SQL

select_statement É uma instrução SELECT sem uma cláusula INTO

O conjunto ativo de um cursor é determinado pela instrução SELECT na declaração do cursor. É obrigatória a existência de uma cláusula INTO para uma instrução SELECT no código PL/SQL. Entretanto, observe que a instrução SELECT na declaração do cursor não pode ter uma cláusula INTO. Isso acontece porque você está apenas definindo um cursor na seção declarativa e não recuperando linhas para dentro do cursor.

Observação

- Não inclua a cláusula INTO na declaração do cursor porque ela aparecerá mais tarde na instrução FETCH.
- Se você deseja que as linhas sejam processadas em uma sequência específica, use a cláusula ORDER BY na consulta.
- O cursor pode ser qualquer instrução SELECT válida, incluindo joins, subconsultas etc.

Abrindo o Cursor

```
DECLARE
  CURSOR c_emp_cursor IS
    SELECT employee_id, last_name FROM employees
    WHERE department_id =30;
  ...
BEGIN
  OPEN c_emp_cursor;
```

13

Abrindo o Cursor

A instrução `OPEN` executa a consulta associada ao cursor, identifica o conjunto ativo e posiciona o cursor na primeira linha. A instrução `OPEN` é incluída na seção executável do bloco PL/SQL.

`OPEN` é uma instrução executável que realiza as seguintes operações:

1. Aloca memória dinamicamente para uma área de contexto
2. Efetua parse da instrução `SELECT`
3. Associa as variáveis de entrada (define os valores para as variáveis de entrada obtendo seus endereços de memória)
4. Identifica o conjunto ativo (o conjunto de linhas que satisfaz os critérios de pesquisa). As linhas do conjunto ativo não são recuperadas em variáveis quando a instrução `OPEN` é executada. Mais especificamente, é a instrução `FETCH` que recupera as linhas do cursor nas variáveis.
5. Posiciona o ponteiro na primeira linha do conjunto ativo.

Observação: Se nenhuma linha for retornada pela consulta quando o cursor for aberto, o código PL/SQL não gerará uma exceção. Você pode saber o número de linhas retornadas com um cursor explícito, usando o atributo `<cursor_name>%ROWCOUNT`.

Extraindo Dados do Cursor

```
DECLARE
  CURSOR c_emp_cursor IS
    SELECT employee_id, last_name FROM employees
    WHERE department_id = 30;
  v_empno employees.employee_id%TYPE;
  v_lname employees.last_name%TYPE;
BEGIN
  OPEN c_emp_cursor;
  FETCH c_emp_cursor INTO v_empno, v_lname;
  DBMS_OUTPUT.PUT_LINE( v_empno || ' ' || v_lname);
END;
/
```

```
anonymous block completed
114 Raphaely
```

14

Extraindo Dados do Cursor

A instrução **FETCH** recupera as linhas do cursor uma de cada vez. Após cada extração, o cursor avança para a linha seguinte no conjunto ativo. É possível usar o atributo **%NOTFOUND** para verificar se o conjunto ativo inteiro foi recuperado.

Examine o exemplo mostrado no slide. Duas variáveis, **empno** e **lname**, são declaradas para armazenar os valores obtidos do cursor com o comando **fetch**. Examine a instrução **FETCH**.

Os valores foram extraídos com sucesso do cursor para as variáveis. Entretanto, existem seis funcionários no departamento 30, mas apenas uma linha foi extraída. Para extrair todas as linhas, é necessário usar loops. No próximo slide, você verá como um loop é usado para extrair todas as linhas.

A instrução **FETCH** executa as seguintes operações:

1. Lê os dados da linha atual nas variáveis PL/SQL de saída.
2. Avança o ponteiro para a linha seguinte do conjunto ativo.

Extraindo Dados do Cursor

```
DECLARE
  CURSOR c_emp_cursor IS
    SELECT employee_id, last_name FROM employees
    WHERE department_id =30;
  v_empno employees.employee_id%TYPE;
  v_lname employees.last_name%TYPE;
BEGIN
  OPEN c_emp_cursor;
  LOOP
    FETCH c_emp_cursor INTO v_empno, v_lname;
    EXIT WHEN c_emp_cursor%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE( v_empno ||' ' ||v_lname);
  END LOOP;
END;
/
```

16

Extraindo Dados do Cursor (continuação)

Observe que um LOOP simples é usado para extrair todas as linhas. Além disso, o atributo %NOTFOUND do cursor é usado para testar a condição de saída. A saída do bloco PL/SQL é mostrada abaixo:

```
anonymous block completed
114 Raphaely
115 Khoo
116 Baida
117 Tobias
118 Himuro
119 Colmenares
```

Fechando o Cursor

```
...  
  LOOP  
    FETCH c_emp_cursor INTO empno, lname;  
    EXIT WHEN c_emp_cursor%NOTFOUND;  
    DBMS_OUTPUT.PUT_LINE( v_empno || ' ' || v_lname);  
  END LOOP;  
  CLOSE c_emp_cursor;  
END;  
/
```

17

Fechando o Cursor

A instrução `CLOSE` desativa o cursor, libera a área de contexto e remove a definição do conjunto ativo. Feche o cursor após concluir o processamento da instrução `FETCH`. Você poderá reabrir o cursor se necessário. Um cursor só poderá ser reaberto se ele tiver sido fechado. Se você tentar extrair dados de um cursor com o comando `fetch` após ele ter sido fechado, uma exceção `INVALID_CURSOR` será gerada.

Observação: Embora seja possível encerrar o bloco PL/SQL sem fechar os cursores, convém que seja um hábito fechar todos os cursores declarados explicitamente para liberar recursos. Existe um limite máximo no número de cursores abertos por sessão, que é determinado pelo parâmetro `OPEN_CURSORS` no arquivo de parâmetros do banco de dados. (`OPEN_CURSORS` = 50 por default.)

Cursores e Registros

Processe as linhas do conjunto ativo extraindo valores para um registro PL/SQL.

```
DECLARE
  CURSOR c_emp_cursor IS
    SELECT employee_id, last_name FROM employees
    WHERE department_id = 30;
  v_emp_record c_emp_cursor%ROWTYPE;
BEGIN
  OPEN c_emp_cursor;
  LOOP
    FETCH c_emp_cursor INTO v_emp_record;
    EXIT WHEN c_emp_cursor%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE( v_emp_record.employee_id
                          || ' ' || v_emp_record.last_name);
  END LOOP;
  CLOSE c_emp_cursor;
END;
```

18

Cursores e Registros

Você já viu que é possível definir registros que tenham a estrutura de colunas em uma tabela. Também é possível definir um registro baseado na lista selecionada de colunas de um cursor explícito. Isso é conveniente para o processamento das linhas do conjunto ativo porque é possível simplesmente extrair (fetch) para dentro do registro. Portanto, os valores das linhas são carregados diretamente nos campos correspondentes do registro.

```
anonymous block completed
114 Raphaely
115 Khoo
116 Baida
117 Tobias
118 Himuro
119 Colmenares
```


Loops de Cursor FOR

Sintaxe:

```
FOR record_name IN cursor_name LOOP
    statement1;
    statement2;
    . . .
END LOOP;
```

- O loop de cursor FOR é um atalho para processar cursores explícitos.
- Os comandos implícitos open, fetch, exit e close são executados.
- O registro é declarado implicitamente.

19

Loops de Cursor FOR

Você aprendeu a extrair dados de cursores com o comando fetch usando loops simples. Agora você vai aprender a usar um loop FOR de cursor, que processa linhas em um cursor explícito. Trata-se de um atalho porque o cursor é aberto, uma linha é extraída uma vez para cada iteração do loop, o loop termina quando a última linha é processada e o cursor é fechado automaticamente. O loop propriamente dito é encerrado no fim da iteração que extrai a última linha.

Na sintaxe:

record_name

É o nome do registro declarado implicitamente

cursor_name

É um identificador PL/SQL para o cursor declarado anteriormente

Diretrizes

- Não declare o registro que controla o loop; ele é declarado implicitamente.
- Se necessário, teste os atributos do cursor durante o loop.
- Se preciso, informe os parâmetros para um cursor entre parênteses após o nome do cursor na instrução FOR.

Loops de Cursor FOR

```
DECLARE
  CURSOR c_emp_cursor IS
    SELECT employee_id, last_name FROM employees
    WHERE department_id =30;
BEGIN
  FOR emp_record IN c_emp_cursor
  LOOP
    DBMS_OUTPUT.PUT_LINE( emp_record.employee_id
      || ' ' || emp_record.last_name );
  END LOOP;
END;
/
```

```
anonymous block completed
114 Raphaely
115 Khoo
116 Baida
117 Tobias
118 Himuro
119 Colmenares
```

20

Loops For de Cursor (continuação)

O exemplo para demonstrar o uso de um loop simples na extração de dados dos cursores foi reescrito para usar o loop FOR de cursor.

`emp_record` é o registro declarado implicitamente. Você pode acessar os dados extraídos com esse registro implícito, como mostra o slide. Observe que nenhuma variável é declarada para armazenar os dados extraídos usando a cláusula INTO. O código não tem instruções OPEN e CLOSE para abrir e fechar o cursor, respectivamente.

Atributos de Cursores Explícitos

Use os atributos de cursores explícitos para obter as informações de status sobre um cursor.

Atributo	Tipo	Descrição
%ISOPEN	Booleano	Será avaliado como <code>TRUE</code> se o cursor estiver aberto
%NOTFOUND	Booleano	Será avaliado como <code>TRUE</code> se a extração mais recente não retornar uma linha
%FOUND	Booleano	Será avaliado como <code>TRUE</code> se a extração mais recente retornar uma linha; complemento de %NOTFOUND
%ROWCOUNT	Número	Será avaliado como o número total de linhas retornadas até agora

Atributos de Cursores Explícitos

A exemplo dos cursores implícitos, existem quatro atributos para a obtenção de informações de status sobre um cursor. Quando anexados ao nome da variável do cursor, esses atributos retornam informações úteis sobre a execução de uma instrução de manipulação de cursor.

Observação: Não é possível fazer referência a atributos de cursores diretamente em uma instrução SQL.

Atributo %ISOPEN

- Só é possível extrair linhas quando o cursor está aberto.
- Use o atributo de cursor %ISOPEN antes de executar um comando fetch para saber se o cursor está aberto.

Exemplo:

```
IF NOT c_emp_cursor%ISOPEN THEN
    OPEN c_emp_cursor;
END IF;
LOOP
    FETCH c_emp_cursor...
```


Atributo %ISOPEN

- Só é possível extrair linhas quando o cursor está aberto. Use o atributo de cursor %ISOPEN para determinar se o cursor está aberto.
- Extraia linhas em um loop. Use os atributos de cursores para determinar quando sair do loop.
- Use o atributo de cursor %ROWCOUNT para fazer o seguinte:
 - Processar um número exato de linhas.
 - Extrair as linhas de um loop com o comando fetch e determinar quando sair do loop.

Observação: %ISOPEN retorna o status do cursor: TRUE se estiver aberto e FALSE se não estiver aberto.

%ROWCOUNT e %NOTFOUND: Exemplo

```
DECLARE
  CURSOR c_emp_cursor IS SELECT employee_id,
    last_name FROM employees;
  v_emp_record c_emp_cursor%ROWTYPE;
BEGIN
  OPEN c_emp_cursor;
  LOOP
    FETCH c_emp_cursor INTO v_emp_record;
    EXIT WHEN c_emp_cursor%ROWCOUNT > 10 OR
      c_emp_cursor%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE( v_emp_record.employee_id
      || ' ' || v_emp_record.last_name);
  END LOOP;
  CLOSE c_emp_cursor;
END; /
```



```
anonymous block completed
174 Abel
166 Ande
130 Atkinson
105 Austin
204 Baer
116 Baida
167 Banda
172 Bates
192 Bell
151 Bernstein
```

%ROWCOUNT e %NOTFOUND: Exemplo

O exemplo no slide recupera os dez primeiros funcionários, um a um. Esse exemplo mostra como os atributos %ROWCOUNT e %NOTFOUND podem ser usados para condições de saída de um loop.

Loops For de Cursor Usando Subconsultas

Não é necessário declarar o cursor.

```
BEGIN
  FOR emp_record IN (SELECT employee_id, last_name
    FROM employees WHERE department_id =30)
  LOOP
    DBMS_OUTPUT.PUT_LINE( emp_record.employee_id
      ||' '||emp_record.last_name);
  END LOOP;
END;
/
```

```
anonymous block completed
114 Raphaely
115 Khoo
116 Baida
117 Tobias
118 Himuro
119 Colmenares
```

24

Loops For de Cursor Usando Subconsultas

Observe que não há seção declarativa nesse bloco PL/SQL. A diferença entre os loops FOR de cursor usando subconsultas e o loop FOR de cursor está na declaração do cursor. Se estiver criando loops FOR de cursor com subconsultas, você não precisará declarar o cursor na seção declarativa. Você precisará informar a instrução SELECT que determina o conjunto ativo no loop propriamente dito.

O exemplo usado para ilustrar um loop FOR de cursor foi reescrito para ilustrar um loop FOR de cursor que utiliza subconsultas.

Observação: Não será possível fazer referência aos atributos de cursores explícitos se você usar uma subconsulta em um loop FOR de cursor porque você não poderá dar um nome explícito ao cursor.

Agenda

- O que são cursores explícitos?
- Usando cursores explícitos
- **Usando cursores com parâmetros**
- Bloqueando linhas e fazendo referência à linha atual

Cursores com Parâmetros

Sintaxe:

```
CURSOR cursor_name
  [(parameter_name datatype, ...)]
IS
  select_statement;
```

- Especifique valores de parâmetros para um cursor quando o cursor for aberto e a consulta for executada.
- Abra um cursor explícito várias vezes com um conjunto ativo distinto a cada vez.

```
OPEN cursor_name(parameter_value,.....) ;
```

26

Cursores com Parâmetros

Você pode especificar parâmetros para um cursor. Isso significa que você pode abrir e fechar um cursor explícito várias vezes em um bloco, retornando um conjunto ativo diferente em cada ocasião. Para cada execução, o cursor anterior é fechado e reaberto com um novo conjunto de parâmetros.

Cada parâmetro formal na declaração do cursor deve ter um parâmetro real correspondente ^[1] na instrução OPEN. Os tipos de dados dos parâmetros são iguais aos das variáveis escalares, mas você não define seus tamanhos. Os nomes dos parâmetros são para referências na expressão de consulta do cursor.

Na sintaxe:

<i>cursor_name</i>	É um identificador PL/SQL para o cursor declarado
<i>parameter_name</i>	É o nome de um parâmetro
<i>datatype</i>	É o tipo de dados escalar do parâmetro
<i>select_statement</i>	É uma instrução SELECT sem a cláusula INTO

A notação do parâmetro não oferece melhor funcionalidade; ela simplesmente permite que você especifique valores de entrada com facilidade e clareza. Isso será especialmente útil quando o mesmo cursor for referido de forma repetitiva.

Cursores com Parâmetros

```
DECLARE
  CURSOR c_emp_cursor (deptno NUMBER) IS
    SELECT employee_id, last_name
    FROM employees
    WHERE department_id = deptno;
  ...
BEGIN
  OPEN c_emp_cursor (10);
  ...
  CLOSE c_emp_cursor;
  OPEN c_emp_cursor (20);
  ...
```

```
anonymous block completed
200 Whalen
201 Hartstein
202 Fay
```

27

Cursores com Parâmetros (continuação)

Os tipos de dados dos parâmetros são iguais aos das variáveis escalares, mas você não define seus tamanhos. Os nomes dos parâmetros servem para fazer referência na consulta do cursor. Neste exemplo, um cursor é declarado e definido com um parâmetro:

```
DECLARE
  CURSOR c_emp_cursor(deptno NUMBER) IS SELECT ...
```

As seguintes instruções abrem o cursor e retornam diferentes conjuntos ativos:

```
OPEN c_emp_cursor(10);
OPEN c_emp_cursor(20);
```

Você pode especificar parâmetros para o cursor usado em um loop FOR de cursor:

```
DECLARE
  CURSOR c_emp_cursor(p_deptno NUMBER, p_job VARCHAR2) IS
    SELECT ...
BEGIN
  FOR emp_record IN c_emp_cursor(10, 'Sales') LOOP ...
```

Agenda

- O que são cursores explícitos?
- Usando cursores explícitos
- Usando cursores com parâmetros
- Bloqueando linhas e fazendo referência à linha atual

Cláusula FOR UPDATE

Sintaxe:

```
SELECT ...  
FROM      ...  
FOR UPDATE [OF column_reference] [NOWAIT | WAIT n];
```

- Use o bloqueio explícito para negar acesso a outras sessões no período da transação.
- Bloqueie as linhas *antes* do comando update ou delete.

Cláusula FOR UPDATE

Se houver várias sessões para um único banco de dados, existirá a possibilidade de que as linhas de uma tabela sejam atualizadas depois de você ter aberto o cursor. Você verá os dados atualizados apenas quando reabrir o cursor. Portanto, convém colocar bloqueios nas linhas antes de atualizá-las ou deletá-las. Você pode bloquear as linhas com a cláusula FOR UPDATE na consulta do cursor.

Na sintaxe:

<i>column_reference</i>	É uma coluna em uma tabela com a qual a consulta é executada (Também é possível usar uma lista de colunas.)
NOWAIT	Retornará um erro do Oracle Server se as linhas forem bloqueadas por outra sessão.

A cláusula FOR UPDATE é a última cláusula em uma instrução SELECT, mesmo após ORDER BY (se ela existir). Quando você quiser consultar várias tabelas, poderá usar a cláusula FOR UPDATE para restringir o bloqueio de linha a determinadas tabelas. FOR UPDATE OF *col_name(s)* bloqueia linhas apenas em tabelas que contêm *col_name(s)*.

Cláusula WHERE CURRENT OF

Sintaxe:

```
WHERE CURRENT OF cursor ;
```

- Use cursores para atualizar ou deletar a linha atual.
- Inclua a cláusula `FOR UPDATE` na consulta do cursor para bloquear as linhas primeiro.
- Use a cláusula `WHERE CURRENT OF` para fazer referência à linha atual de um cursor explícito.

```
UPDATE employees  
SET    salary = ...  
WHERE CURRENT OF c_emp_cursor;
```

Cláusula WHERE CURRENT OF

A cláusula `WHERE CURRENT OF` é usada junto com a cláusula `FOR UPDATE` para se referir à linha atual em um cursor explícito. A cláusula `WHERE CURRENT OF` é usada na instrução `UPDATE` ou `DELETE`, enquanto a cláusula `FOR UPDATE` é especificada na declaração do cursor. Você pode usar a combinação para atualizar e deletar a linha atual da tabela correspondente do banco de dados. Isso permite aplicar atualizações e deleções à linha tratada no momento sem a necessidade de fazer referência explícita ao ID da linha. Inclua a cláusula `FOR UPDATE` na consulta do cursor para que as linhas sejam bloqueadas em `OPEN`.

Na sintaxe:

cursor É o nome do cursor declarado (o cursor deverá ter sido declarado com a cláusula `FOR UPDATE`).

Questionário

FIAP

Os cursores implícitos são declarados implicitamente por PL/SQL em todas as instruções `SELECT` de DML e PL/SQL. O Oracle Server abre implicitamente um cursor para processar cada instrução SQL não associada a um cursor declarado explicitamente.

- a. Verdadeiro
- b. Falso

32

Resposta: a

Sumário

Nesta lição, você aprendeu a:

- Distinguir os tipos de cursores:
 - Os cursores implícitos são usados em todas as instruções DML e consultas de uma linha.
 - Os cursores explícitos são usados em consultas com nenhuma, uma ou mais linhas
- Criar e gerenciar cursores explícitos
- Usar loops simples e loops `FOR` de cursores para gerenciar várias linhas nos cursores
- Avaliar o status do cursor usando os atributos de cursor
- Usar as cláusulas `FOR UPDATE` e `WHERE CURRENT OF` para atualizar ou deletar a linha atual extraída

Sumário

O Oracle Server usa áreas de trabalho para executar instruções SQL e armazenar as informações de processamento. Você pode usar uma estrutura PL/SQL chamada *cursor* para nomear uma área de trabalho e acessar as informações armazenadas. Há dois tipos de cursores: implícitos e explícitos. O bloco PL/SQL declara implicitamente um cursor para todas as instruções de manipulação de dados SQL, incluindo consultas que retornam apenas uma linha. No caso das consultas que retornam várias linhas, você deve declarar explicitamente um cursor para processar as linhas individualmente.

Os cursores explícitos e as variáveis de cursor têm quatro atributos: `%FOUND`, `%ISOPEN`, `%NOTFOUND` e `%ROWCOUNT`. Quando anexados ao nome da variável do cursor, esses atributos retornam informações úteis sobre a execução de uma instrução SQL. É possível usar atributos de cursor em instruções procedurais, mas não em instruções SQL.

Use loops simples ou loops `FOR` de cursor para operar várias linhas extraídas pelo cursor. Se usar loops simples, você terá de usar os comandos `open`, `fetch` e `close` com o cursor; no entanto, os loops `FOR` de cursor fazem isso implicitamente. Se estiver atualizando ou deletando linhas, bloqueie as linhas com a cláusula `FOR UPDATE`. Isso assegura que os dados que você está usando não sejam atualizados por outra sessão depois que você abriu o cursor. Use uma cláusula `WHERE CURRENT OF` junto com a cláusula `FOR UPDATE` para fazer referência à linha atual extraída pelo cursor.

Exercício 7: Visão Geral

Este exercício aborda os seguintes tópicos:

- Declarando e usando cursores explícitos para consultar linhas de uma tabela
- Usando um loop `FOR` de cursor
- Aplicando atributos de cursores para testar o status do cursor
- Declarando e usando cursores com parâmetros
- Usando as cláusulas `FOR UPDATE` e `WHERE CURRENT OF`

Exercício 7: Visão Geral

Neste exercício, você aplica seu conhecimento de cursores para processar uma série de linhas de uma tabela e preencher outra tabela com os resultados usando um loop `FOR` de cursor. Você também cria um cursor com parâmetros.