

# – Oracle – Comandos para Processamento de Transações

Laboratório de Bases de Dados  
Profa. Dra. Cristina Dutra de Aguiar Ciferri

# Transação

- Unidade lógica de trabalho
  - contém um ou mais comandos SQL para **manipulação** dos dados executados por um único usuário
- Início da transação
  - primeiro comando SQL executável
- Fim da transação
  - execução de **commit** ou **rollback** pelo usuário

# Esquema

Início de conexão com o Oracle (CONNECT)

início da transação 1 (SET TRANSACTION)

operações de escrita/leitura da transação 1

...

fim da transação 1 (COMMIT/ROLLBACK)

início da transação 2 (SET TRANSACTION)

operações de escrita/leitura da transação 2

...

fim da transação 2 (COMMIT/ROLLBACK)

Fim de conexão com o Oracle (DISCONNECT)

# CONNECT / DISCONNECT

```
CONN[ECT] usuário/senha;
```

- Abre uma conexão com o Oracle

```
DISC[ONNECT];
```

- Finaliza a conexão com o Oracle
- Torna persistente (i.e., *commit*) as alterações pendentes no banco de dados

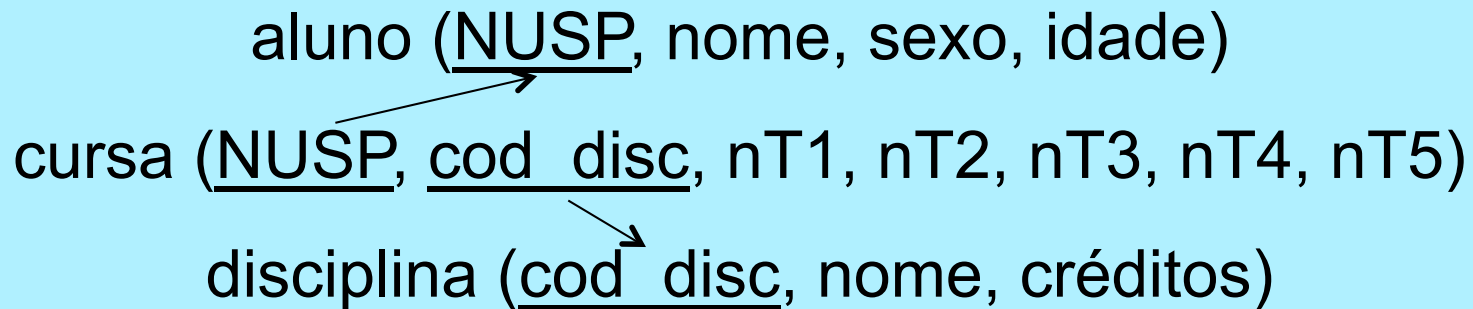
# COMMIT

```
COMMIT;
```

- Finaliza uma transação **com** sucesso
- Atualiza os dados do banco de dados
  - torna persistente as alterações da transação
- Apaga os *savepoints* da transação
- Libera todas as travas utilizadas pela transação

# Exemplo

aluno (NUSP, nome, sexo, idade)  
curso (NUSP, cod disc, nT1, nT2, nT3, nT4, nT5)  
disciplina (cod disc, nome, créditos)



DECLARE

...

BEGIN

...

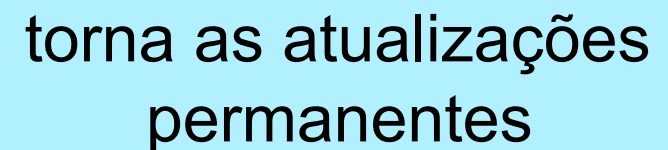
UPDATE curso SET nT2 = 0.0 WHERE NUSP = 2;

UPDATE curso SET nT2 = 0.0 WHERE NUSP = 5;

COMMIT;

END;

torna as atualizações  
permanentes



# ROLLBACK

ROLLBACK;

- Finaliza uma transação **sem** sucesso
- Desfaz os efeitos das operações da transação
- Apaga os *savepoints* da transação
- Libera todas as travas utilizadas pela transação

# Exemplo

DECLARE

...

BEGIN

...

INSERT INTO aluno VALUES (1, 'Alexandra', 'f');

INSERT INTO cursa VALUES (1, 1, 7.0, 8.0, 7.5, 9.0, NULL);

...

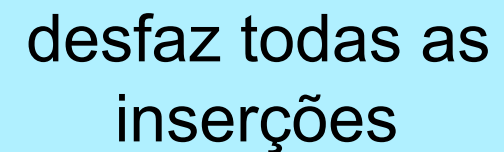
EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

ROLLBACK;

...

END;



desfaz todas as  
inserções



# SAVEPOINT

```
SAVEPOINT nome_savepoint ;
```

- Identifica um ponto em uma transação para o qual é possível desfazer as operações da transação
  - permite desfazer partes de uma transação, ao invés da transação como um todo
- Geralmente usado com o comando **ROLLBACK TO**

# Exemplo

DECLARE

aluno\_id aluno.NUSP%Type;

BEGIN

UPDATE aluno SET ... WHERE NUSP = aluno\_id;

DELETE FROM aluno WHERE ...

...

SAVEPOINT do\_insert;

INSERT INTO aluno VALUES (...);

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

ROLLBACK TO do\_insert;

END;

desfaz os comandos  
depois do savepoint

# Considerações

- Podem existir vários savepoints em uma transação
  - **commit** ou **rollback**: remove todos savepoints
  - **rollback to**: remove savepoints marcados depois do savepoint de retorno
- Nomes de savepoints
  - não precisam ser declarados
  - podem ser usados novamente em uma transação

savepoint ativo:  
último

# SET TRANSACTION

```
SET TRANSACTION
{ { READ ONLY | READ WRITE }
  | ISOLATION LEVEL { SERIALIZABLE |
                    READ COMMITTED }
  | USE ROLLBACK SEGMENT rollback_segment
}
| NAME 'nome_transação'
};
```

*itálico: opções default*

- Inicia uma transação
- Define características da transação

# SET TRANSACTION

- Exemplos:
  - SET TRANSACTION READ ONLY;
  - SET TRANSACTION READ WRITE;
  - SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
  - SET TRANSACTION ISOLATION LEVEL READ COMMITTED;
  - SET TRANSACTION READ ONLY NAME 'teste';

# Comportamento

- READ ONLY
  - identifica uma transação de **leitura**
  - permite somente consultas (i.e., SELECT)
  - especifica que as consultas da transação somente enxergam alterações realizadas com sucesso (i.e., *committed*) antes do início da transação
  - indicada para transações que executem consultas em várias tabelas do BD ao mesmo tempo que estas tabelas estão sendo alteradas por outros usuários

# Exemplo

COMMIT;

← finaliza transações anteriores

SET TRANSACTION READ ONLY;

```
SELECT empresa.emp_codigo, emp_nome,  
       count(filial_nro) as quantidade  
FROM empresa LEFT JOIN filial  
       ON filial.emp_codigo = empresa.emp_codigo  
GROUP BY empresa.emp_codigo, emp_nome  
ORDER BY empresa.emp_codigo;
```

SELECT ....

COMMIT;

← finaliza a transação READ ONLY

# Comportamento

- READ WRITE
  - identifica uma transação de **leitura e escrita**
  - permite a especificação de
    - consultas (i.e., SELECT)
    - operações (i.e., INSERT, UPDATE, DELETE)



# Exemplo

COMMIT;

SET TRANSACTION READ WRITE;

SELECT empresa.emp\_codigo, emp\_nome,  
count(filial\_nro) as quantidade

FROM empresa LEFT JOIN filial

ON filial.emp\_codigo = empresa.emp\_codigo

GROUP BY empresa.emp\_codigo, emp\_nome

ORDER BY empresa.emp\_codigo;

UPDATE empresa SET emp\_codigo = 5 WHERE emp\_codigo = 4;

COMMIT;

← finaliza a transação READ WRITE

Pode apresentar o problema de leitura não repetida !

# Grau de Isolação

- ISOLATION LEVEL
  - especifica como transações que contêm modificações no BD são manipuladas
  - SERIALIZABLE
    - transação somente vê as alterações que foram finalizadas com sucesso no momento que a transação começou, além das alterações feitas pela própria transação
  - READ COMMITTED
    - cada consulta executada por uma transação vê apenas dados que foram finalizados com sucesso antes da consulta (e não da transação)