



ORACLE

Academy



Database Programming with SQL

13-3

Modificando uma Tabela

ORACLE
Academy



Objetivos

- Nesta lição, você aprenderá a:
 - Explicar por que é importante ser capaz de modificar uma tabela
 - Explicar e dar um exemplo de cada uma das instruções DDL (ALTER, DROP, RENAME e TRUNCATE) e o efeito que têm em tabelas e colunas
 - Construir uma consulta e executar os comandos ADD, MODIFY e DROP de ALTER TABLE
 - Explicar e executar FLASHBACK QUERY em uma tabela
 - Explicar e executar operações FLASHBACK TABLE

Objetivos

- Nesta lição, você aprenderá a:
 - Controlar as alterações dos dados feitas por um certo período
 - Explicar a lógica de usar TRUNCATE, em vez de DELETE, para tabelas
 - Adicionar um comentário a uma tabela usando o comando COMMENT ON TABLE
 - Nomear as alterações que podem ou não ser feitas em uma coluna
 - Explicar quando e por que a instrução SET UNUSED é vantajosa

Finalidade

- Lembra da frase "Nada é permanente, exceto a mudança"?
- Não seria bom se nunca errássemos ou tivéssemos que mudar as coisas?
- Como você já sabe agora, os bancos de dados são entidades dinâmicas
- Eles provavelmente não seriam muito úteis se não pudessem ser modificados

Finalidade

- Até agora, você criou tabelas e fez alterações nos dados das linhas dentro das tabelas, mas como se faz mudanças nas próprias tabelas?
- Esta lição apresenta os comandos DDL que são usados para alterar, renomear, esvaziar ou simplesmente eliminar uma tabela

ALTER TABLE

- As instruções ALTER TABLE são usadas para:
 - Adicionar uma nova coluna
 - Modificar uma coluna existente
 - Definir um valor DEFAULT para uma coluna
 - Eliminar uma coluna
- Você pode adicionar ou modificar uma coluna em uma tabela, mas não especificar onde a coluna aparece

ALTER TABLE

- Uma coluna recém-adicionada sempre passa a ser a última da tabela
- Além disso, se uma tabela já tiver linhas de dados e você adicionar uma nova coluna, a nova coluna será inicialmente nula para todas as linhas preexistentes

ALTER TABLE: Adicionando uma Coluna

- Para adicionar uma nova coluna, use a sintaxe SQL mostrada:

```
ALTER TABLE tablename  
ADD (column name data type [DEFAULT expression],  
column name data type [DEFAULT expression], ...
```

```
ALTER TABLE nomedatabela  
ADD (nome da coluna tipo de dados [DEFAULT expressão],  
nome da coluna tipo de dados [DEFAULT expressão], ...
```

- Por exemplo:

```
ALTER TABLE my_cd_collection  
ADD (release_date DATE DEFAULT SYSDATE);
```

```
ALTER TABLE my_friends  
ADD (favorite_game VARCHAR2(30));
```

ALTER TABLE: Modificando uma Coluna

- A modificação de uma coluna pode incluir alterações em seu tipo de dados, tamanho e valor DEFAULT
- Eis as regras e restrições para modificar uma coluna:
 - Você pode aumentar a largura ou precisão de uma coluna numérica
 - Você pode aumentar a largura uma coluna de caracteres
 - Você pode diminuir a largura de uma coluna NUMBER se ela contiver apenas valores nulos, ou se a tabela não tiver linhas
 - Para os tipos VARCHAR, você pode diminuir a largura até o maior valor contido na coluna

ALTER TABLE: Modificando uma Coluna

- Você poderá alterar o tipo de dados somente se a coluna contiver valores nulos
- Você poderá converter uma coluna CHAR para VARCHAR2 ou VARCHAR2 para CHAR somente se ela contiver valores nulos ou se você não mudar o tamanho para algo menor do que qualquer valor na coluna
- Uma mudança no valor DEFAULT de uma coluna afeta somente as inserções posteriores à tabela

ALTER TABLE: Exemplo de Modificação de uma Coluna

- Exemplo: uma tabela foi criada com duas colunas:

```
CREATE TABLE mod_emp  
  (last_name VARCHAR2(20) ,  
   salary NUMBER(8,2));
```

- Quais das modificações a seguir seriam permitidas ou não? (Imagine a tabela com e sem linha de dados para dar suas respostas)

```
ALTER TABLE mod_emp  
  MODIFY (last_name VARCHAR2(30));
```

```
ALTER TABLE mod_emp  
  MODIFY (last_name VARCHAR2(10));
```

```
ALTER TABLE mod_emp  
  MODIFY (salary NUMBER(10,2));
```

```
ALTER TABLE mod_emp  
  MODIFY (salary NUMBER(8,2) DEFAULT 50);
```

ORACLE

Academy

ALTER TABLE: Exemplo de Modificação de uma Coluna

- Seria permitida somente se as colunas estivessem vazias ou se o maior nome tivesse dez caracteres ou menos

```
ALTER TABLE mod_emp  
    MODIFY (last_name VARCHAR2(10));
```

- Seria permitida com ou sem dados à medida que a largura da coluna aumentasse.

```
ALTER TABLE mod_emp  
    MODIFY (last_name VARCHAR2(30));
```

ALTER TABLE: Exemplo de Modificação de uma Coluna

- Seria permitida com ou sem dados à medida que a precisão da coluna aumentasse.

```
ALTER TABLE mod_emp  
    MODIFY (salary NUMBER(10,2));
```

- Seria permitida com ou sem dados se fosse adicionado somente um valor DEFAULT.

```
ALTER TABLE mod_emp  
    MODIFY (salary NUMBER(8,2) DEFAULT 50);
```

ALTER TABLE: Eliminando uma Coluna

- Ao eliminar uma coluna, siga estas regras:
 - É possível eliminar uma coluna que contém dados
 - É possível eliminar apenas uma coluna por vez
 - Você não pode eliminar todas as colunas de uma tabela; pelo menos uma deve permanecer
 - Após a eliminação da coluna, os valores de dados contidos nela não poderão ser recuperados

ALTER TABLE: Eliminando uma Coluna

- Sintaxe SQL:

```
ALTER TABLE tablename  
DROP COLUMN column name;
```

- Por exemplo:

```
ALTER TABLE my_cd_collection  
DROP COLUMN release_date;
```

```
ALTER TABLE my_friends  
DROP COLUMN favorite_game;
```


Colunas SET UNUSED

- A eliminação de uma coluna de uma tabela grande pode levar muito tempo
- Uma alternativa mais rápida é marcar a coluna como não utilizável
- Os valores da coluna permanecem no banco de dados, mas não podem ser acessados de forma alguma. Portanto, o efeito é o mesmo da eliminação da coluna
- Na verdade, você pode adicionar uma nova coluna ao banco de dados com o mesmo nome que a coluna não utilizada
- As colunas não usadas encontram-se lá, mas estão invisíveis!
- Sintaxe:

```
ALTER TABLE tablename SET UNUSED (column name);
```

Exemplo de Colunas com SET UNUSED

- Exemplo:

```
ALTER TABLE copy_employees  
    SET UNUSED (email);
```

- DROP UNUSED COLUMNS remove todas as colunas marcadas como não usadas
- Você deve utilizar essa instrução quando quiser recuperar o espaço em disco das colunas não usadas de uma tabela

- Exemplo:

```
ALTER TABLE copy_employees  
    DROP UNUSED COLUMNS;
```

Resumo de ALTER TABLE

- Esta tabela resume os usos do comando ALTER TABLE:

Sintaxe	Resultados	Preocupações
ALTER TABLE nomedatabela ADD (nome da coluna tipo de dados [DEFAULT expressão], nome da coluna tipo de dados [DEFAULT expressão], ...	Adiciona uma nova coluna a uma tabela	Você não pode especificar em que parte da tabela a coluna aparecerá. Ela será a última.
ALTER TABLE nomedatabela MODIFY (nome da coluna tipo de dados [DEFAULT expressão], nome da coluna tipo de dados, ...	Usada para alterar o tipo de dados, o tamanho e o valor padrão de uma coluna	Uma mudança no valor padrão de uma coluna afeta somente as inserções posteriores à tabela.
ALTER TABLE nomedatabela DROP COLUMN nome da coluna;	Usada para eliminar uma coluna de uma tabela	A tabela deverá ter pelo menos uma coluna restante após a alteração. Após eliminada, a coluna não poderá ser recuperada.
ALTER TABLE nomedatabela SET UNUSED (nome da coluna);	Usada para marcar uma ou mais colunas para que sejam eliminadas posteriormente	Não restaura espaço em disco. As colunas são tratadas como se tivessem sido eliminadas.
ALTER TABLE nomedatabela DROP UNUSED COLUMNS	Remove da tabela todas as colunas marcadas como não usadas	Após definidas como não usadas, as colunas não podem mais ser acessadas; nenhum dado é exibido com o uso de DESCRIBE. Remoção permanente; sem rollback.

DROP TABLE

- A instrução DROP TABLE remove a definição de uma tabela Oracle
- O banco de dados perde todos os dados na tabela e todos os índices associados a ela
- Quando uma instrução DROP TABLE é executada:
 - Todos os dados são excluídos da tabela
 - A descrição da tabela é removida do Dicionário de Dados
- O Servidor Oracle não questiona sua decisão e elimina a tabela imediatamente

DROP TABLE

- No próximo slide, você verá que talvez seja possível restaurar uma tabela eliminada
- Somente o criador da tabela ou um usuário com o privilégio DROP ANY TABLE (normalmente, apenas o DBA) pode remover uma tabela
- Sintaxe:

```
DROP TABLE tablename;
```

- Exemplo:

```
DROP TABLE copy_employees;
```

FLASHBACK TABLE

- Se você eliminar uma tabela por engano, talvez seja possível trazê-la de volta, junto com os seus dados
- Cada usuário do banco de dados tem sua própria lixeira, para a qual são movidos os objetos eliminados, e eles podem ser recuperados com o comando FLASHBACK TABLE
- Esse comando pode ser usado para restaurar uma tabela, uma view ou um índice que foi eliminado por engano
- A Sintaxe é:

```
FLASHBACK TABLE tablename TO BEFORE DROP;
```

FLASHBACK TABLE

- Por exemplo, se você eliminar a tabela EMPLOYEES por engano, pode restaurá-la simplesmente emitindo o comando:

```
FLASHBACK TABLE copy_employees TO BEFORE DROP;
```

- Como o proprietário de uma tabela, você pode executar o comando de flashback e, se a tabela que estiver sendo restaurada tiver índices, eles também serão recuperados
- É possível ver quais objetos podem ser restaurados consultando a view USER_RECYCLEBIN do dicionário de dados



FLASHBACK TABLE

- USER_RECYCLEBIN pode ser consultada como qualquer outra view do dicionário de dados:

```
SELECT original_name, operation, droptime  
FROM user_recyclebin
```

ORIGINAL_NAME	OPERATION	DROPTIME
EMPLOYEES	DROP	2007-12-05:12.34.24
EMP_PK	DROP	2007-12-05:12.34.24



FLASHBACK TABLE

- Após ser restaurada pelo comando FLASHBACK TABLE, a tabela não será mais visível na view USER_RECYCLEBIN
- Os índices que foram eliminados junto com a tabela original também serão restaurados
- Pode ser necessário (por questões de segurança) eliminar completamente uma tabela, ignorando a lixeira.
- Faça isso adicionando a palavra-chave PURGE

```
DROP TABLE copy_employees PURGE;
```

RENAME

- Para mudar o nome de uma tabela, use a instrução RENAME
- Isso pode ser feito apenas pelo proprietário do objeto ou pelo DBA
- Sintaxe:

```
RENAME nome_antigo TO nome_novo;
```

- Exemplo:

```
RENAME my_cd_collection TO my_music;
```

- Mais tarde, veremos que podemos renomear outros tipos de objetos, como views, sequências e sinônimos

TRUNCATE

- O truncamento remove todas as linhas de uma tabela e libera o espaço de armazenamento utilizado por ela
- Ao usar a instrução TRUNCATE TABLE:
 - Você não pode fazer rollback da remoção de linhas
 - Você deve ser o proprietário da tabela ou ter privilégios DROP ANY TABLE no sistema

TRUNCATE

- Sintaxe:

```
TRUNCATE TABLE nomedatabela;
```

- A instrução DELETE também remove linhas de uma tabela, mas não libera espaço de armazenamento
- TRUNCATE é mais rápido do que DELETE porque não gera informações de rollback

COMMENT ON TABLE

- Você pode adicionar um comentário de até 2.000 caracteres sobre uma coluna, tabela ou view usando a instrução COMMENT
- Sintaxe:

```
COMMENT ON TABLE nomedatabela | COLUMN tabela.coluna  
IS 'coloque seu comentário aqui';
```

- Exemplo:

```
COMMENT ON TABLE employees  
IS 'Western Region only';
```

COMMENT ON TABLE

- Para ver os comentários sobre a tabela no dicionário de dados:

```
SELECT table_name, comments  
FROM user_tab_comments;
```

TABLE_NAME	COMMENTS
EMPLOYEES	Western Region Only

- Se você quiser eliminar um comentário feito em uma tabela ou coluna use a string vazia (''):

```
COMMENT ON TABLE employees IS ' ' ;
```

FLASHBACK QUERY

- Pode ser que você descubra que os dados de uma tabela foram modificados de maneira inapropriada
- Felizmente, o Oracle tem um recurso que permite ver os dados das linhas em pontos específicos no tempo, para que você possa comparar as diferentes versões de uma linha com o passar do tempo
- Esse recurso é muito útil
- Imagine, por exemplo, que alguém executou acidentalmente uma instrução DML em uma tabela e, em seguida, fez um COMMIT dessas alterações
- O Oracle Application Express faz o commit automaticamente, portanto, é fácil cometer erros

FLASHBACK QUERY

- Você pode usar o recurso FLASHBACK QUERY para examinar como eram as linhas ANTES de as alterações serem aplicadas
- Quando modifica os dados, o Oracle sempre mantém uma cópia de como eles eram antes de as alterações terem sido feitas
- Portanto, ele mantém uma cópia do valor de coluna antigo para uma atualização de coluna, a linha inteira para uma exclusão e nada para uma instrução de inserção

FLASHBACK QUERY

- Essas cópias antigas são armazenadas em um local especial chamado tablespace de UNDO
- Os usuários podem acessar essa área especial do banco de dados usando Flashback Query
- Você pode usar a cláusula VERSIONS em uma instrução SELECT para ver versões mais antigas dos dados

FLASHBACK QUERY

- Por exemplo:

```
SELECT employee_id,first_name ||' '|| last_name AS "NAME",  
       versions_operation AS "OPERATION",  
       versions_starttime AS "START_DATE",  
       versions_endtime  AS "END_DATE", salary  
FROM employees  
   VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE  
WHERE  employee_id = 1;
```

- O número SCN referenciado na consulta acima significa o Número de Alteração do Sistema e é uma identificação precisa do tempo no banco de dados
- Ele é um número sequencial incrementado e mantido pelo próprio banco de dados

FLASHBACK QUERY

- A melhor maneira de demonstrar FLASHBACK QUERY é usar um exemplo
- O conteúdo é o seguinte para employee_id 1 na tabela de funcionários

```
SELECT employee_id,first_name ||' '|| last_name AS "NAME",  
       versions_operation AS "OPERATION",  
       versions_starttime AS "START_DATE",  
       versions_endtime  AS "END_DATE", salary  
FROM employees  
     VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE  
WHERE  employee_id = 1;
```

nenhum dado encontrado

FLASHBACK QUERY

- Em seguida, criamos o funcionário:

```
INSERT INTO copy_employees
VALUES (1, 'Natacha', 'Hansen', 'NHANSEN', '4412312341234',
       '07-SEP-1998', 'AD_VP', 12000, null, 100, 90, NULL);

SELECT employee_id, first_name || ' ' || last_name AS "NAME",
       versions_operation AS "OPERATION",
       versions_starttime AS "START_DATE",
       versions_endtime AS "END_DATE", salary
FROM employees
     VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
WHERE employee_id = 1;
```

EMPLOYEE_ID	NAME	OPERATION	START_DATE	END_DATE	SALARY
1	Natacha Hansen	I	07-SEP-1998 06.51.58 AM	-	12000

FLASHBACK QUERY

- Depois, você pode atualizar a linha:

```
UPDATE copy_employees  
SET salary = 1  
WHERE employee_id = 1;
```

```
SELECT employee_id, first_name || ' ' || last_name AS "NAME",  
       versions_operation AS "OPERATION",  
       versions_starttime AS "START_DATE",  
       versions_endtime   AS "END_DATE", salary  
FROM employees  
   VENSIONS BETWEEN SCN MINVALUE AND MAXVALUE  
WHERE employee_id = 1;
```

EMPLOYEE_ID	NAME	OPERATION	START_DATE	END_DATE	SALARY
1	Natacha Hansen	U	07-SEP-1998 06.57.01 AM	-	1
1	Natacha Hansen	I	07-SEP-1998 06.51.58 AM	07-SEP-1998 06.57.01 AM	12000

FLASHBACK QUERY

- Em seguida, você pode excluir a linha:

```
DELETE from copy_employees  
WHERE employee_id = 1;
```

```
SELECT employee_id,first_name ||' '|| last_name AS "NAME",  
       versions_operation AS "OPERATION",  
       versions_starttime AS "START_DATE",  
       versions_endtime   AS "END_DATE", salary  
FROM employees  
   VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE  
WHERE  employee_id = 1;
```

EMPLOYEE_ID	NAME	OPERATION	START_DATE	END_DATE	SALARY
1	Natacha Hansen	D	07-SEP-1998 07.00.10 AM	-	1
1	Natacha Hansen	U	07-SEP-1998 06.57.01 AM	07-SEP-1998 07.00.10 AM	1
1	Natacha Hansen	I	07-SEP-1998 06.51.58 AM	07-SEP-1998 06.57.01 AM	12000

FLASHBACK QUERY

- O resultado da última consulta no slide anterior só está disponível quando se usa Flashback Query, ou seja, a cláusula VERSIONS
- Se tentar fazer uma pesquisa normal a partir de employee_id = 1 após a instrução DELETE, você receberá o erro comum: No Data Found (Nenhum Dado Encontrado)

```
SELECT employee_id, salary  
FROM copy_employees  
WHERE employee_id = 1;
```

Terminologia

- Estes são os principais termos usados nesta lição:
 - ALTER TABLE
 - ADD
 - MODIFY
 - DROP
 - DROP TABLE
 - RENAME
 - TRUNCATE

Terminologia

- Estes são os principais termos usados nesta lição:
 - COMMENT ON TABLE
 - FLASHBACK TABLE
 - FLASHBACK QUERY
 - SET UNUSED

Resumo

- Nesta lição, você aprendeu a:
 - Explicar por que é importante ser capaz de modificar uma tabela
 - Explicar e dar um exemplo de cada uma das instruções DDL (ALTER, DROP, RENAME e TRUNCATE) e o efeito que têm em tabelas e colunas
 - Construir uma consulta e executar os comandos ADD, MODIFY e DROP de ALTER TABLE
 - Explicar e executar FLASHBACK QUERY em uma tabela
 - Explicar e executar operações FLASHBACK TABLE

Resumo

- Nesta lição, você aprendeu a:
 - Controlar as alterações dos dados feitas por um certo período
 - Explicar a lógica de usar TRUNCATE, em vez de DELETE, para tabelas
 - Adicionar um comentário a uma tabela usando o comando COMMENT ON TABLE
 - Nomear as alterações que podem ou não ser feitas em uma coluna
 - Explicar quando e por que a instrução SET UNUSED é vantajosa



ORACLE

Academy

