Tipps & Tricks: Erweiterung des RENAME- Befehls

Bereich:	SQL	Erstellung:	03/2003 MM
Versionsinfo:	9.2 10.2 11.1	Letzte Überarbeitung:	05/2009 BK

Erweiterung des RENAME Befehls

Bisher war es nicht möglich, Tabellenspalten, Indizes und Constraints innerhalb der Datenbank umzubenennen. Sollte dies für Indizes und Constraints geschehen, mussten diese zunächst gelöscht und anschließend unter dem neuen Namen wieder erstellt werden.

Bei Spaltenumbenennung hatte man entweder die Möglichkeit die gesamte Tabelle mit der "neuen" Spalte neu zu erstellen (CREATE TABLE ... AS SELECT ...), musste anschließend aber Constraints und Indizes neu anlegen. Oder man fügte der bestehenden Tabelle eine neue Spalte - mit entsprechendem neuem Namen - hinzu, füllte sie mit den Werten der alten Spalte und löschte diese. Aber auch hier mussten bestehende Constraints und Indizes der alten Spalte neu angelegt werden.

Ab Version 9.2 können Spalten, Indizes und Constraints durch die RENAME-Klausel innerhalb des ALTER TABLE- bzw. ALTER INDEX-Befehls nun einen neuen Namen bekommen.

Vorbereitung des Beispiels

Zu Beginn des Beispiels wird eine Tabelle namens TEST_NEU inklusive Primärschlüssel erstellt.

```
CREATE TABLE test_neu (
col1 number(4) CONSTRAINT pk_test_neu PRIMARY KEY,
col2 varchar2(20) NOT NULL);

Tabelle wurde angelegt.

desc test_neu

Name                 Null?    Typ
-----------------    -------- ------------
COL1                 NOT NULL NUMBER(4)
COL2                 NOT NULL VARCHAR2(20)

SELECT constraint_name, column_name
FROM user_cons_columns
WHERE table_name = 'TEST_NEU';

CONSTRAINT_NAME      COLUMN_NAME
--------------------- --------------------
SYS_C004336          COL2
PK_TEST_NEU          COL1

SELECT index_name, column_name
FROM user_ind_columns
WHERE table_name = 'TEST_NEU';

INDEX_NAME           COLUMN_NAME
--------------------- --------------------
```

```
PK_TEST_NEU          COL1
```

Tabelle umbenennen

Als erstes wird die Tabelle TEST_NEU in TEST umbenannt.
(Dies ist zwar kein neues Feature der Version 9.2, soll an dieser Stelle dennoch einmal vorgeführt werden.)

```
RENAME test_neu TO test;

Tabelle wurde umbenannt.
```

Alternativ ist auch die folgende Syntax möglich:

```
ALTER TABLE test_neu RENAME TO test;
```

Spalten umbenennen

Die Spalte COL1 soll in ID_NR und die Spalte COL2 in TEXT umbenannt werden.

```
ALTER TABLE test RENAME COLUMN col1 TO id_nr;

Tabelle wurde geändert.

ALTER TABLE test RENAME COLUMN col2 TO text;

Tabelle wurde geändert.
```

Constraints umbenennen

Die auf die Tabelle gelegten Constraints werden in pk_test und nn_text umbenannt.

```
ALTER TABLE test RENAME CONSTRAINT pk_test_neu TO pk_test;

Tabelle wurde geändert.

ALTER TABLE test RENAME CONSTRAINT SYS_C004336 TO nn_text;

Tabelle wurde geändert.
```

Indizes umbenennen

Der aufgrund eines Primärschlüssels angelegte Index wird beim Umbenennen des Constraints nicht automatisch mit umbenannt. Dies muss manuell erfolgen.

```
ALTER INDEX pk_test_neu RENAME TO pk_test;

Index wurde geändert.
```

Überprüfung der Änderungen

Zum Abschluss des Beispiels werden die vorgenommenen Änderungen überprüft.

```
desc test

Name                Null?    Typ
```

```
----------------- -------- ------------
ID_NR             NOT NULL NUMBER(4)
TEXT              NOT NULL VARCHAR2(20)

SELECT constraint_name, column_name
FROM user_cons_columns
WHERE table_name = 'TEST';

CONSTRAINT_NAME      COLUMN_NAME
-------------------- --------------------
NN_TEXT              TEXT
PK_TEST              ID_NR

SELECT index_name, column_name
FROM user_ind_columns
WHERE table_name = 'TEST';

INDEX_NAME           COLUMN_NAME
-------------------- --------------------
PK_TEST              ID_NR
```