 



Homework 13-3:

Modifying a Table

# Try It / Solve It

Before beginning the practice exercises, execute a DESCRIBE for each of the following tables: o\_employees, o\_departments and o\_jobs. These tables will be used in the exercises. If they do not exist in your account, create them as follows:

1. **Create the three o\_tables – jobs, employees, and departments – using the syntax: CREATE TABLE o\_jobs AS (SELECT \* FROM jobs);**

**CREATE TABLE o\_employees AS (SELECT \* FROM employees); CREATE TABLE o\_departments AS (SELECT \* FROM departments);**

1. **Add the Human Resources job to the jobs table:**

**INSERT INTO o\_jobs (job\_id, job\_title, min\_salary, max\_salary) VALUES('HR\_MAN', 'Human Resources Manager', 4500, 5500);**

1. **Add the three new employees to the employees table:**

**INSERT INTO o\_employees (employee\_id, first\_name, last\_name, email, hire\_date, job\_id)**

**VALUES(210, 'Ramon', 'Sanchez', 'RSANCHEZ', SYSDATE, 'HR\_MAN');**

1. **Add Human Resources to the departments table:**

**INSERT INTO o\_departments(department\_id, department\_name) VALUES (210,'Human Resources');**

You will need to know which columns do not allow null values.

1. Why is it important to be able to modify a table?

Pentru ca intr-o baza de date singurul lucru constant este modificarea, este necesar in permanenta sa le modificam pentru a le putea actualiza.

1. CREATE a table called Artists.
   1. Add the following to the table:
      * artist ID
      * first name
      * last name
      * band name
      * email
      * hourly rate
      * song ID from d\_songs table

CREATE TABLE Artists

(artist\_id NUMBER(5,0),

first\_name VARCHAR2(15),

last\_name VARCHAR2(15),

band\_name VARCHAR2(20),

email VARCHAR2(40),

hourly\_rate NUMBER(5,0),

song\_id NUMBER (5, 0));

INSERT INTO artists (song\_id)

SELECT type\_code

FROM d\_songs;

* 1. INSERT one artist from the d\_songs table.

INSERT INTO artists(artist\_id, first\_name, last\_name, band\_name, email, hourly\_rate, song\_id)

SELECT 1 AS artist\_id,

CASE

WHEN artist IS NULL THEN 'Necunoscut'

WHEN INSTR(artist,' ')=0 THEN artist

ELSE SUBSTR(artist, 1, INSTR(artist,' ') -1)

END

AS first\_name,

CASE

WHEN artist IS NULL THEN 'Necnoscut'

WHEN INSTR(artist,' ')=0 THEN artist

ELSE SUBSTR(artist, INSTR(artist,' '), LENGTH(artist))

END

AS last\_name,

artist AS band\_name,

NULL AS email,

NULL AS hourly\_rate,

NULL AS song\_id

FROM d\_songs

WHERE ROWNUM = 1;

* 1. INSERT one artist of your own choosing; leave song\_id blank.

INSERT INTO artists(artist\_id, first\_name, last\_name, band\_name, email, hourly\_rate)

VALUES

(100,'DA','CRISS', 'Rotarell','Crisro', 20)

* 1. Give an example how each of the following may be used on the table that you have created:

1. ALTER TABLE

ALTER TABLE artists

ADD(data\_curenta DATE DEFAULT SYSDATE);

1. DROP TABLE

DROP TABLE artists

FLASHBACK TABLE artists TO BEFORE DROP;

1. RENAME TABLE

RENAME artists TO my\_artists

1. TRUNCATE

TRUNCATE TABLE my\_artists

1. COMMENT ON TABLE

COMMENT ON TABLE my\_artists

IS 'Comentariul meu!';

1. In your o\_employees table, enter a new column called “Termination.” The datatype for the new column should be VARCHAR2. Set the DEFAULT for this column as SYSDATE to appear as character data in the format: February 20th, 2003.

ALTER TABLE o\_employees

ADD("Termination" VARCHAR(20) DEFAULT TO\_CHAR(SYSDATE,'fmMonth ddth, yyyy'));

1. Create a new column in the o\_employees table called start\_date. Use the TIMESTAMP WITH LOCAL TIME ZONE as the datatype.

ALTER TABLE o\_employees

ADD(start\_date TIMESTAMP WITH LOCAL TIME ZONE);

1. Truncate the o\_jobs table. Then do a SELECT \* statement. Are the columns still there? Is the data still there?

TRUNCATE TABLE o\_employees

SELECT\*

FROM o\_employees;

La rulare apare: no data found, adica nu mai avem coloane si date in table.

1. What is the distinction between TRUNCATE, DELETE, and DROP for tables?

TRUNCATE sterge toate liniile si elibereaza spatial, in timp ce DELETE sterge toate liniile, dar nu elibereaza spatiul. Aceste 2 metode nu ne mai ajut sa recuperam tabelul, in timp ce

Daca facem DROP cu ajutorul FLASHBACK putem recupera tabelul.

1. List the changes that can and cannot be made to a column.

->putem face DROP cu o singura coloana o data,

->putem schimba tipul de data doar daca coloana contine valori NULL,

->putem converti din CHAR in VARCHAR2 si din VARCHAR2 in CHAR doar cand coloana contine valori NULL sau daca nu schimbam size-ul in ceva mai mic.

->ultima coloana pe care o adaugam se va afla mereu la sfarsitul tabelului

1. Add the following comment to the o\_jobs table: "New job description added"

View the data dictionary to view your comments.

COMMENT ON TABLE o\_jobs

IS 'New job description added';

SELECT table\_name, comments

FROM user\_tab\_comments;

1. Rename the o\_jobs table to o\_job\_description.

RENAME o\_jobs TO o\_job\_description

1. F\_staffs table exercises:
   1. Create a copy of the f\_staffs table called copy\_f\_staffs and use this copy table for the remaining labs in this lesson.

CREATE TABLE copy\_f\_staffs

AS (SELECT \* FROM f\_staffs);

* 1. Describe the new table to make sure it exists.

DESCRIBE copy\_f\_staffs;

* 1. Drop the table.

DROP TABLE copy\_f\_staffs;

* 1. Try to select from the table.

SELECT \*

FROM copy\_f\_staffs; 🡪table or view does not exist

* 1. Investigate your recyclebin to see where the table went.

SELECT \*

FROM user\_recyclebin;

* 1. Try to select from the dropped table by using the value stored in the OBJECT\_NAME column. You will need to copy and paste the name as it is exactly, and enclose the new name in “ “ (double quotes). So if the dropped name returned to you is BIN$Q+x1nJdcUnngQESYELVIdQ==$0, you need to write a query that refers to “BIN$Q+x1nJdcUnngQESYELVIdQ==$0”.

Eroare: table or view does not exist in abele cazuri

* 1. Undrop the table.

FLASHBACK TABLE copy\_f\_staffs TO BEFORE DROP;

* 1. Describe the table.

DESCRIBE copy\_f\_staffs;

1. Still working with the copy\_f\_staffs table, perform an update on the table.
   1. Issue a select statement to see all rows and all columns from the copy\_f\_staffs table;

SELECT\*

FROM copy\_f\_staffs;

* 1. Change the salary for Sue Doe to 12 and commit the change.

UPDATE copy\_f\_staffs

SET salary = 12

WHERE first\_name = 'Sue' and last\_name = 'Doe'

* 1. Issue a select statement to see all rows and all columns from the copy\_f\_staffs table;

SELECT\*

FROM copy\_f\_staffs;

* 1. For Sue Doe, update the salary to 2 and commit the change.

UPDATE copy\_f\_staffs

SET salary = 2

WHERE first\_name = 'Sue' and last\_name = 'Doe'

* 1. Issue a select statement to see all rows and all columns from the copy\_f\_staffs table;

SELECT\*

FROM copy\_f\_staffs;

* 1. Now, issue a FLASHBACK QUERY statement against the copy\_f\_staffs table, so you can see all the changes made.

SELECT id,first\_name ||' '|| last\_name AS "NAME",

versions\_operation AS "OPERATION",

versions\_starttime AS "START\_DATE",

versions\_endtime AS "END\_DATE", salary

FROM copy\_f\_staffs

VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE

WHERE id = 12;

* 1. Investigate the result of f), and find the original salary and update the copy\_f\_staffs table salary column for Sue Doe back to her original salary.

UPDATE copy\_f\_staffs

SET salary = 10

WHERE first\_name = 'Sue' and last\_name = 'Doe'

🡪salariul a fost initial 10