13-3: Modifying a Table

Try It / Solve It

1.There is nothing permanent in this world except change and I do make mistakes, that is why databases are also dynamic in nature and so are the tables could be modified.

2.
a.

```
CREATE TABLE  artists
(artist_id NUMBER(5,0),
first_name VARCHAR2(25) CONSTRAINT ait_first_name_nn NOT NULL ENABLE,
last_name VARCHAR2(30) CONSTRAINT ait_last_name_nn NOT NULL ENABLE,
band_name VARCHAR2(30),
email VARCHAR2(75) CONSTRAINT ait_email_nn NOT NULL ENABLE,
hr_rate NUMBER(8,2) CONSTRAINT ait_hr_rate_nn NOT NULL ENABLE,
song_id NUMBER(5,0) CONSTRAINT ait_song_id_nn NOT NULL ENABLE,
CONSTRAINT ait_id_pk PRIMARY KEY (artist_id));
```

b.

```
ALTER TABLE artists
DROP CONSTRAINT ait_email_nn;
ALTER TABLE artists
DROP CONSTRAINT ait_hr_rate_nn;
INSERT INTO artists (artist_id, first_name, last_name, band_name, email, hr_rate,
song_id)
SELECT 1 AS artist_id,
CASE
WHEN artist IS NULL THEN 'first name unknown'
WHEN INSTR(artist,' ') = 0 THEN artist
ELSE SUBSTR(artist,1,INSTR(artist,' ') -1)
END
AS first_name,
CASE
WHEN artist IS NULL THEN 'last name unknown'
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 hr_rate,
id as song_id
FROM d_songs
WHERE  ROWNUM =1 ;
```

c.
```
ALTER TABLE artists
DROP CONSTRAINT ait_song_id_nn;
INSERT INTO artists (artist_id, first_name, last_name, band_name, email, hr_rate,
song_id)
VALUES(2,'Carla','Gray','Carla''s band','carla.email@yahoo.com','999999.99',NULL);
```
d.
1)
```
ALTER TABLE artists
ADD (specialty VARCHAR2(100), college VARCHAR2(100));
```
2)
```
ALTER TABLE artists
DROP COLUMN specialty;
```
3)
```
ALTER TABLE artists
  RENAME TO artists_new_name;
```
4)
```
TRUNCATE TABLE artists_new2;
SELECT * FROM artists_new2
```
5)
Comment on table artists
Is ‘ana’
```
 SELECT table_name, comments FROM user_tab_comments WHERE
LOWER(table_name) = 'artists';
```

3.
```
ALTER TABLE o_employees
ADD ("Termination" VARCHAR2(100)  DEFAULT  TO_CHAR(SYSDATE,'Month ddth,
YYYY') );
```

4.
```
ALTER TABLE o_employees
ADD (start_date TIMESTAMP WITH LOCAL TIME ZONE);\
```

5.
truncate table o_jobs
Columns are still there, data is gone.

6.
The DROP TABLE statement removes the definition of oracle table along with data and indexes. Recovery of a dropped table along with even indexes may be done but it's not guaranteed using FLASHBACK:
TRUNCATE TABLE removes all rows and release storage space without possibility of rollback. It will be faster than DELETE. It won't remove columns from table.
DELETE SQL statement will remove the rows but won't clean storage space.

7.
1. I can increase precision of a number column.
2. In can increase length of character column.
3. I can decrease precision of number column if: it contains only nulls till now or there is no row in table. Otherwise I will get ORA-00940: invalid ALTER command

4. varchar2 can be decreased to length down to the largest value present currently in all rows.
5. Datatype can be changed altogether if all values in this column are nulls.
6. char can become varchar2 if column contain nulls or the size given is not less than any existing field for that column.
6. varchar2 can become char if column contain nulls or the size given is not less than any existing field for that column.
7. Change in default value is effective to new inserts only not the already present rows.
8. A column containing values can be dropped if this is not referenced as foreign key in further tables. Also, data values in it not recovered after column drop.
9. I can drop only one column at a time. Also, at least one column must remain, I cannot drop last column.
10. Since dropping column may take time, it does modify each row before deleting, I can use SET UNUSED command as a replacement for practical purposes and DROP UNUSED later on.
11. I can rename a column if I want.

8.
COMMENT ON TABLE o_jobs IS 'New job description added';

SELECT table_name, comments
FROM user_tab_comments WHERE LOWER(table_name) = ‘o_jobs';

9.
ALTER TABLE o_jobs
RENAME TO o_job_description;

10.
A.CREATE TABLE copy_f_staffs1
AS (SELECT * FROM f_staffs);
B.describe copy_f_staffs1
C.DROP TABLE copy_f_staffs1;
D. Select * from copy_f_staffs1
E.DESCRIBE user_recyclebin ;
F--
G.FLASHBACK TABLE copy_f_staffs 1TO BEFORE DROP;
H.DESCRIBE copy_f_staffs;

11.
A.SELECT * FROM copy_f_staffs1;
B.UPDATE copy_f_staffs1
SET salary = 12
WHERE first_name = 'Sue' AND last_name = 'Doe';
C.SELECT * FROM copy_f_staffs1;
D.UPDATE copy_f_staffs1
SET salary = 2
WHERE first_name = 'Sue' AND last_name = 'Doe';
E.SELECT * FROM copy_f_staffs1;
F.SELECT versions_operation, versions_starttime, versions_endtime, id, first_name, last_name, birthdate, salary,overtime_rate,training,staff_type,manager_id, manager_budget,manager_target

```
FROM copy_f_staffs1
VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
WHERE id = 12;
G.UPDATE copy_f_staffs1
SET salary = (SELECT salary
FROM copy_f_staffs1
VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE
WHERE first_name = 'Sue' AND last_name = 'Doe' AND versions_operation IS NULL AND
versions_starttime IS NULL)
WHERE first_name = 'Sue' AND last_name = 'Doe';

SELECT versions_operation, versions_starttime,  versions_endtime, id, first_name,
last_name, birthdate, salary,overtime_rate,training,staff_type,manager_id,
manager_budget,manager_target
FROM copy_f_staffs1
VERSIONS BETWEEN SCN MINVALUE AND MAXVALUE


WHERE id = 12;
```