Lab 1-part 2

Update, Delete and Referential Integrity

This lab demonstrates SQL Update and Delete commands and constraints related to deleting the primary key values used in referential integrity constraints (Cascade, Set Null).

Table of contents:

1. Updating table records
2. COMMIT and ROLLBACK
3. Deleting table records
4. Referential integrity constraints -- ON DELETE CASCADE/ SET NULL

We will be using tables lab1_emp and lab1_hours, created and populated in part 1 of the lab.

As before, store the SQL code that you write in a file, so that you can access it later.

1. Updating a record: UPDATE command

   a. Syntax:

      UPDATE <table>
      SET <column> = <expression>[,<column> = <expression>]
      [WHERE condition];

      if the WHERE clause is omitted, the update is performed on all records, otherwise it is performed on records that satisfy the WHERE condition.

   b. Examples: Add the following to your script and run:

      The following will update the hourly wage of the employee with emp_no A02 to 12.

      ```sql
      UPDATE lab1_emp
      SET hourly_wage = 12
      WHERE emp_no=‘A02’;
      ```

      Now
      - add an update command to modify the last name of Alice Walter to Sidner.
      - give everyone a $1 raise
      - set the value of the state to ‘RI’ in all records with a missing (NULL) value of state (WHERE state is NULL).
      - try to modify the emp_no of Joseph Miller to ‘A06’ notice what happens (the update is rejected - why?)
        o you should see an error message
ORA-02292: integrity constraint (TBABAIAN.EMP_FK) violated -- child record found

2. COMMIT and ROLLBACK

Syntax: COMMIT

INSERT, UPDATE, and DELETE commands are not committed to the database until the COMMIT statement is executed. COMMIT is a transaction managing command that confirms operations to the database on the server (closing Oracle also acts as a confirmation of the commands entered).

Note that exiting Oracle via typing EXIT causes a COMMIT. If you exit from it in any other way - the COMMIT is not guaranteed, and you may lose the changes.

Syntax ROLLBACK

ROLLBACK (ROLL) can be issued to cancel any database operations since the most recent COMMIT. Like COMMIT, ROLLBACK is also a transaction managing command; however, it cancels operations instead of confirming them.

3. Deleting records: DELETE command

a. Syntax:

DELETE <table>
[WHERE condition];

if the WHERE clause is omitted, all rows are deleted, otherwise it is performed on records that satisfy the WHERE condition.

b. Examples: Add the following to your script and run:

The following will delete the record with emp_no ‘A02’ (unless, of course, there is a record in lab1_hours referring to employee ‘A02’)

DELETE lab1_emp
WHERE emp_no='A02';

Now try to delete a record for employee Joan Garfield (emp_no ‘A01’) from lab1_emp. The deletion command should be rejected!
4. **Constraints**

   a. **Referential integrity and deleting**

   When a value of a primary key of a table is removed or changed - that may result in a violation of a referential integrity constraint.

   The following options are available in **Oracle** for dealing with the deleted primary key values (these constraint definitions must appear inside CREATE TABLE). (Note that the standard SQL specifies other options as well).

   CONSTRAINT emp_fk FOREIGN KEY (emp_no) REFERENCES lab1_emp(emp_no) ON DELETE CASCADE
   • when a record is deleted from the parent table (lab1_emp) all records of the child table (lab1_hours) that refer to the deleted value of the emp_no in their foreign key column are also deleted.

   CONSTRAINT emp_fk FOREIGN KEY (emp_no) REFERENCES lab1_emp(emp_no) ON DELETE SET NULL
   • when a record is deleted from the parent table (lab1_emp) the value of the foreign key column in all records of the child table (lab1_hours) that refer to the deleted value of emp_no in their foreign key column are set to NULL.

   **Exercise:** to see the impact of CASCADE and SET NULL on deleting from the parent table do the following:
   1. drop table lab1_hours
   2. redefine it (CREATE TABLE) using a foreign key constraint with **ON DELETE CASCADE**
   3. insert a record lab1_hours for employee with emp_no ‘A01’
   4. delete a record with emp_no ‘A01’ from lab1_emp
   5. see how the lab1_hours table has changed (select * from lab1_hours)
   6. repeat steps 1-6 except in step 2 - specify **ON DELETE SET NULL**