Sincennes, Alexandre

Colotouros, Nicholas

Lu, Rita

COMP-421, group 3

Project M3

Q1

Before adding trigger: one tuple exists in table rating where pid = 1

```
CS421=> SELECT * FROM rating where pid = 1;
 pid | cid | rating_amt
-----+-----+------------
   1 |   9 |          5
(1 row)
```

Create trigger that deletes all the ratings for a song when the song is deleted

```
CS421=> CREATE OR REPLACE FUNCTION delete_rating() RETURNS TRIGGER AS $_$
CS421$> BEGIN
CS421$>         DELETE FROM rating WHERE rating.pid = OLD.pid;
CS421$>
ABORT         COMMIT        DROP          LOAD          RESET         TABLE
ALTER         COPY          END           LOCK          REVOKE        TRUNCATE
ANALYZE       CREATE        EXECUTE       MOVE          ROLLBACK      UNLISTEN
BEGIN         DEALLOCATE    EXPLAIN       NOTIFY        SAVEPOINT     UPDATE
CHECKPOINT    DECLARE       FETCH         PREPARE       SELECT        VACUUM
CLOSE         DELETE FROM   GRANT         REASSIGN      SET           VALUES
CLUSTER       DISCARD       INSERT        REINDEX       SHOW          WITH
COMMENT       DO            LISTEN        RELEASE       START
CS421$> RETURN OLD;
CS421$> END $_$ LANGUAGE 'plpgsql';
CREATE FUNCTION
CS421=>
CS421=> CREATE TRIGGER deleteRating BEFORE
CS421-> DELETE ON song
CS421-> FOR EACH ROW EXECUTE PROCEDURE delete_rating();
CREATE TRIGGER
```

Query affected by trigger: deleting song with pid = 1 also cause all ratings with pid = 1 to be deleted. The same SELECT query from before returned 0 rows instead of 1.

```
CS421=> DELETE FROM song WHERE song.pid = 1;
DELETE 1
CS421=> SELECT * FROM rating WHERE pid = 1;
 pid | cid | rating_amt
-----+-----+------------
(0 rows)
```

Before adding trigger: one tuple exists in table rating where pid = 1

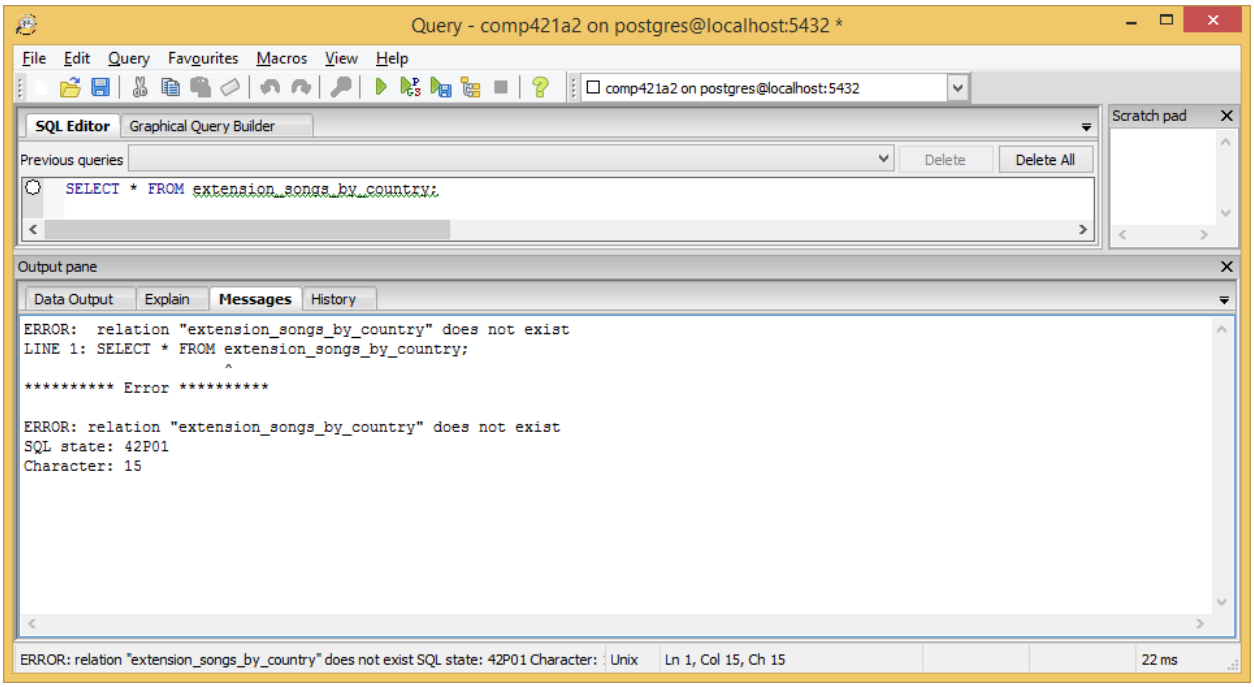
```
CS421=> SELECT * FROM rating where pid = 2;
 pid | cid | rating_amt
-----+-----+------------
   2 |   4 |          2
(1 row)
```

Query not affeccted by trigger: updating song with pid = 2 did not cause all the ratings for the song to be deleted. The same SELECT query from before still returns the same row.

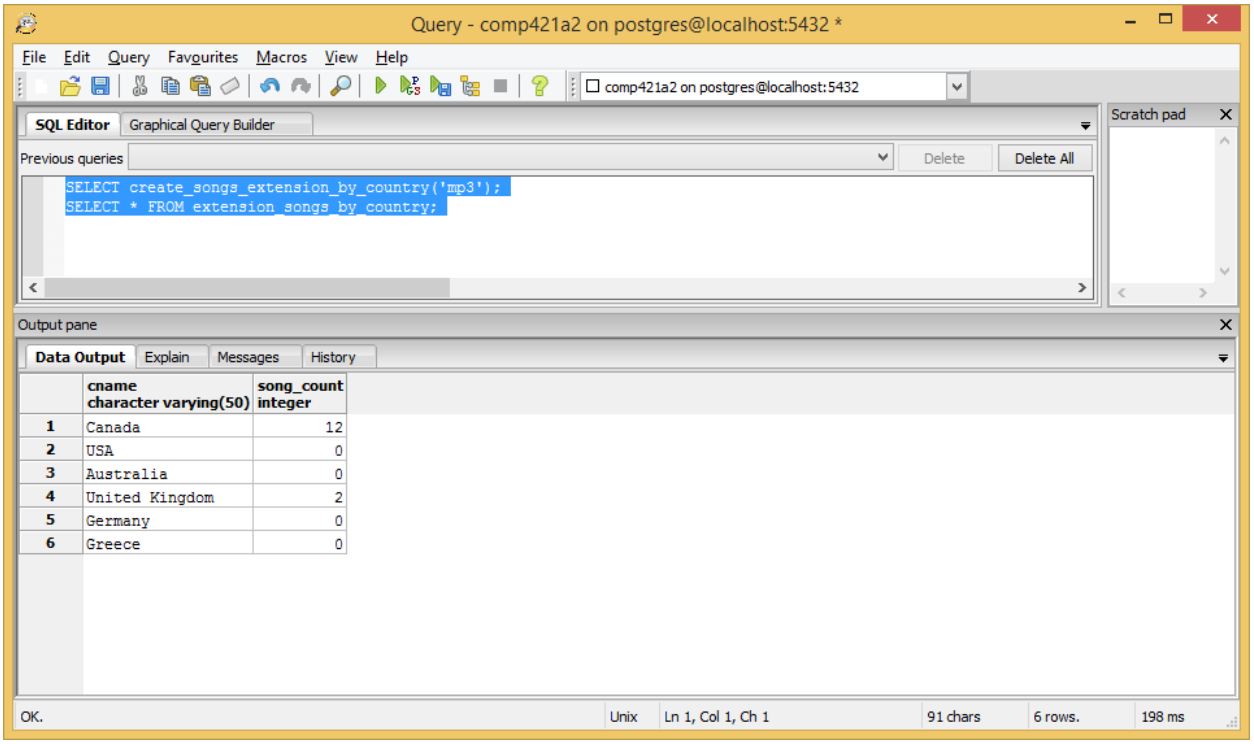
```
CS421=> UPDATE song SET title = 'New Song' WHERE pid = 2;
UPDATE 1
CS421=> SELECT * FROM rating WHERE pid = 2;
 pid | cid | rating_amt
-----+-----+------------
   2 |   4 |          2
(1 row)
```

Q2

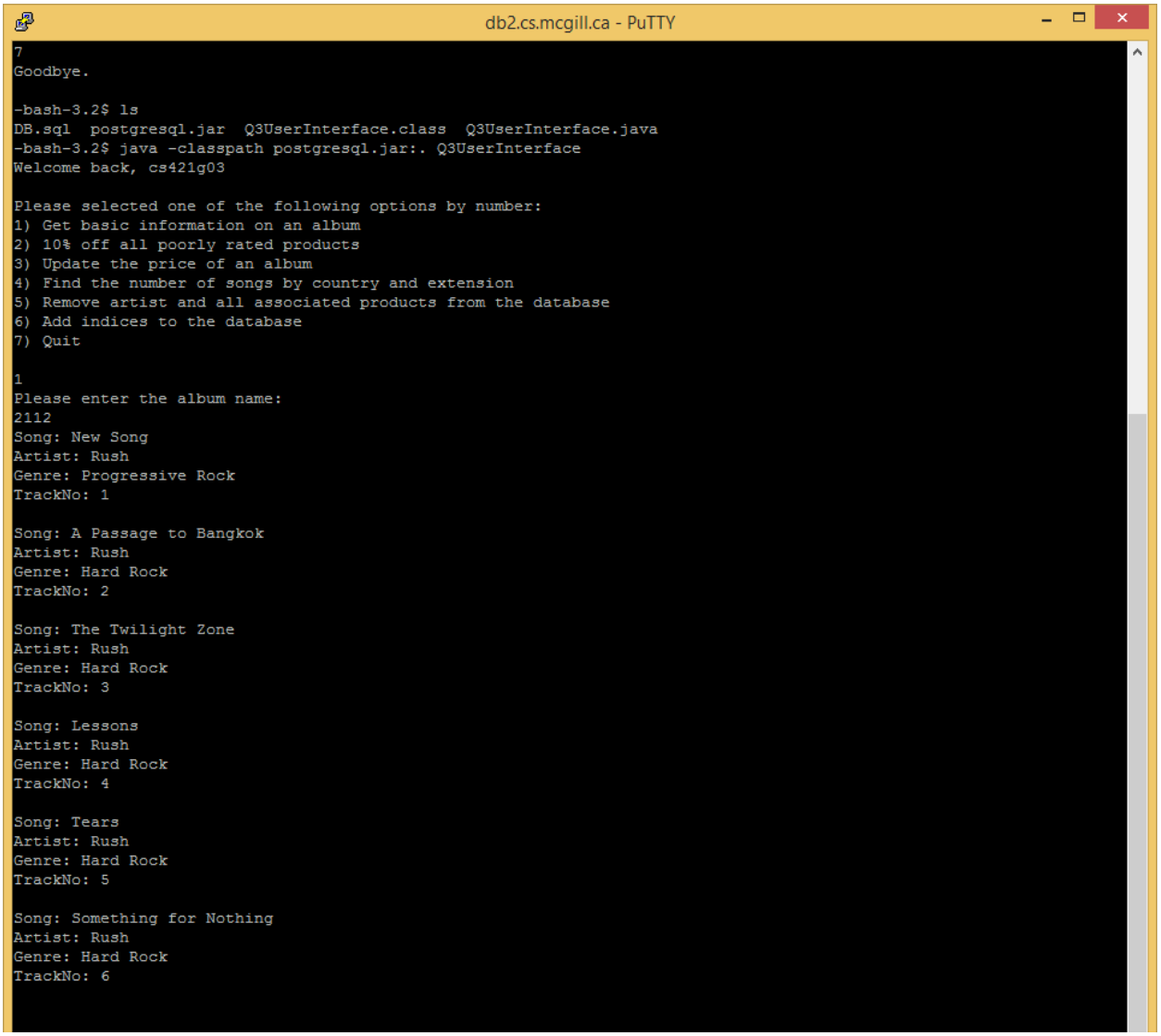
Before execution of the stored procedure:

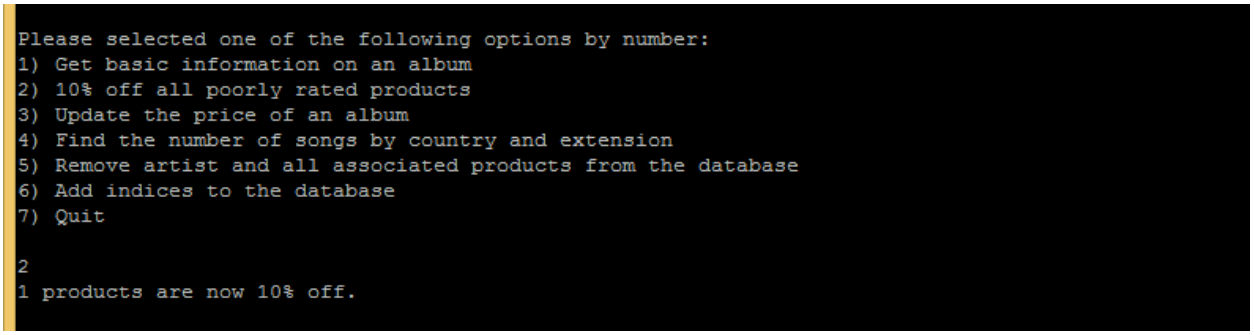


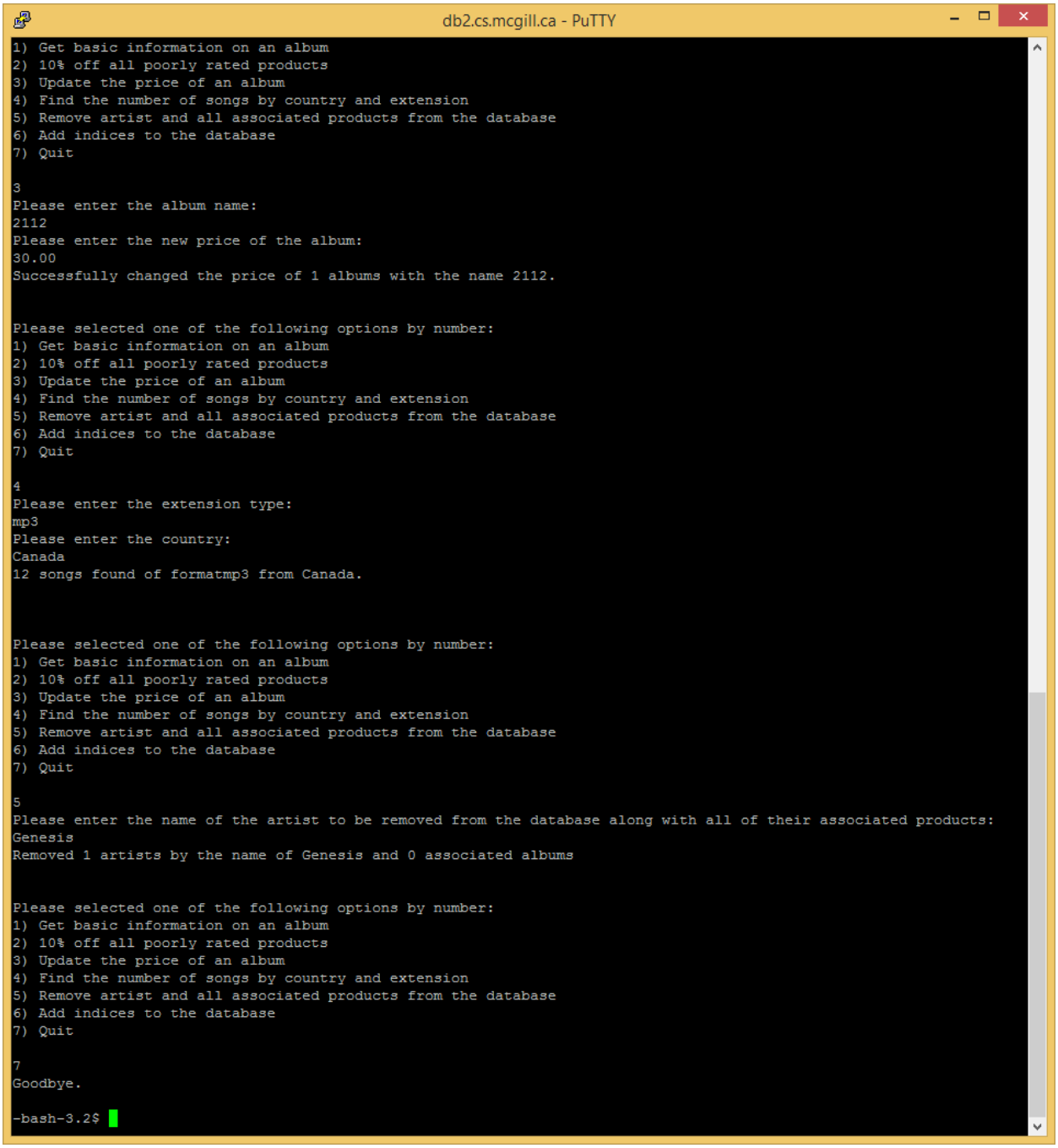
After execution:



Q3


db2.cs.mcgill.ca - PuTTY
7
Goodbye.
-bash-3.2$ ls
DB.sql postgresql.jar Q3UserInterface.class Q3UserInterface.java
-bash-3.2$ java -classpath postgresql.jar:. Q3UserInterface
Welcome back, cs421g03
Please selected one of the following options by number:
1) Get basic information on an album
2) 10% off all poorly rated products
3) Update the price of an album
4) Find the number of songs by country and extension
5) Remove artist and all associated products from the database
6) Add indices to the database
7) Quit
1
Please enter the album name:
2112
Song: New Song
Artist: Rush
Genre: Progressive Rock
TrackNo: 1
Song: A Passage to Bangkok
Artist: Rush
Genre: Hard Rock
TrackNo: 2
Song: The Twilight Zone
Artist: Rush
Genre: Hard Rock
TrackNo: 3
Song: Lessons
Artist: Rush
Genre: Hard Rock
TrackNo: 4
Song: Tears
Artist: Rush
Genre: Hard Rock
TrackNo: 5
Song: Something for Nothing
Artist: Rush
Genre: Hard Rock
TrackNo: 6


Please selected one of the following options by number:
1) Get basic information on an album
2) 10% off all poorly rated products
3) Update the price of an album
4) Find the number of songs by country and extension
5) Remove artist and all associated products from the database
6) Add indices to the database
7) Quit
2
1 products are now 10% off.


db2.cs.mcgill.ca - PuTTY
1) Get basic information on an album
2) 10% off all poorly rated products
3) Update the price of an album
4) Find the number of songs by country and extension
5) Remove artist and all associated products from the database
6) Add indices to the database
7) Quit
3
Please enter the album name:
2112
Please enter the new price of the album:
30.00
Successfully changed the price of 1 albums with the name 2112.
Please selected one of the following options by number:
1) Get basic information on an album
2) 10% off all poorly rated products
3) Update the price of an album
4) Find the number of songs by country and extension
5) Remove artist and all associated products from the database
6) Add indices to the database
7) Quit
4
Please enter the extension type:
mp3
Please enter the country:
Canada
12 songs found of formatmp3 from Canada.
Please selected one of the following options by number:
1) Get basic information on an album
2) 10% off all poorly rated products
3) Update the price of an album
4) Find the number of songs by country and extension
5) Remove artist and all associated products from the database
6) Add indices to the database
7) Quit
5
Please enter the name of the artist to be removed from the database along with all of their associated products:
Genesis
Removed 1 artists by the name of Genesis and 0 associated albums
Please selected one of the following options by number:
1) Get basic information on an album
2) 10% off all poorly rated products
3) Update the price of an album
4) Find the number of songs by country and extension
5) Remove artist and all associated products from the database
6) Add indices to the database
7) Quit
7
Goodbye.
-bash-3.2$

Last function of Q3, as well as Q4:

```
Welcome back, cs421g03

Please selected one of the following options by number:
1) Get basic information on an album
2) 10% off all poorly rated products
3) Update the price of an album
4) Find the number of songs by country and extension
5) Remove artist and all associated products from the database
6) Add indices to the database
7) Quit

6
Testing the effect of two indices on two queries:
Query 1:
SELECT genre.genid, genre.name, COUNT(Genre.genid) FROM genre JOIN song_genre ON
genre.genid = song_genre.genid JOIN song ON song_genre.sid = song.pid JOIN produc
t ON song.pid = product.pid JOIN purchase_product ON purchase_product.pid = produ
ct.pid GROUP BY genre.genid, genre.name;
Query 1 took 20miliseconds to complete before adding indices.
Index on genre.name added.
Query 1 took 2miliseconds to complete after adding indices.
Query 2:
SELECT artist.name, AVG(rating_amt) FROM artist JOIN song_artist ON artist.artid
= song_artist.artid JOIN song ON song_artist.sid = song.pid JOIN rating ON rating
.pid = song.pid GROUP BY artist.name;
Query 2 took 2miliseconds to complete before adding indices.
Index on artist.name added.
Query 2 took 1miliseconds to complete after adding indices.
```

We go from 20 ms to 2 ms, and 2 ms to 1 ms, as shown above.