CSC 370 Assignment 5

Amaan Makhani

[redacted]

Below the output of the table schemas, the creation of the stored procedure, and the implementation of the trigger is shown. Furthermore, the whole .sql file used to run the commands is shown below the trigger creation. All text in red is the output of the postgresql command.

```
db107=> \d parts
                    Table "public.parts"
 Column |         Type          | Collation | Nullable | Default
--------+-----------------------+-----------+----------+---------
 pid    | integer               |           | not null |
 pname  | character varying(40) |           | not null |
 color  | character varying(20) |           | not null |
Indexes:
    "parts_pkey" PRIMARY KEY, btree (pid)
Check constraints:
    "parts_check" CHECK ((length(pname::text) % 2) = (length(color::text) % 2))
    "parts_color_check" CHECK (color::text ~ '^[a-zA-Z]+[-][0-9]+'::text)
Triggers:
    keep_part_history AFTER INSERT OR DELETE OR UPDATE ON parts FOR EACH ROW EXECUTE PROCEDURE
track_parts_history()

db107=> \d partshistory
                      Table "public.partshistory"
  Column   |            Type             | Collation | Nullable | Default
-----------+-----------------------------+-----------+----------+---------
 pid       | integer                     |           | not null |
 pname     | character varying(40)       |           | not null |
 color     | character varying(20)       |           | not null |
 operation | character(1)                |           | not null |
 opwhen    | timestamp without time zone |           | not null |
 opuser    | character(20)               |           | not null |
Indexes:
    "partshistory_pkey" PRIMARY KEY, btree (pid)
Check constraints:
    "partshistory_check" CHECK ((length(pname::text) % 2) = (length(color::text) % 2))
    "partshistory_color_check" CHECK (color::text ~ '^[a-zA-Z]+[-][0-9]+'::text)

CREATE OR REPLACE FUNCTION track_parts_history() RETURNS TRIGGER AS $keep_part_history$
    BEGIN
        IF (TG_OP = 'DELETE') THEN
            INSERT INTO partshistory SELECT OLD.*, 'D', now(), user;
        ELSIF (TG_OP = 'UPDATE') THEN
            INSERT INTO partshistory SELECT OLD.*, 'U', now(), user;
        ELSIF (TG_OP = 'INSERT') THEN
            INSERT INTO partshistory SELECT NEW.*, 'I', now(), user;
        END IF;
        RETURN NULL;
    END;
$keep_part_history$ LANGUAGE plpgsql;

CREATE TRIGGER keep_part_history
AFTER INSERT OR UPDATE OR DELETE ON parts
    FOR EACH ROW EXECUTE PROCEDURE track_parts_history();
```

```
amaanmakhani@linux6c:~$ psql -h studentdb.csc.uvic.ca -d db107 -U amaanmakhani
Password for user amaanmakhani:
psql (10.14 (Ubuntu 10.14-0ubuntu0.18.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

db107=> \i A5.sql

db107=> CREATE TABLE parts(
    pid         integer         NOT NULL,
    pname       varchar(40)     NOT NULL,
    color       varchar(20)     NOT NULL        CHECK (color ~ '^[a-zA-Z]+[-][0-9]+'),
    PRIMARY KEY(pid),
    check ((LENGTH(pname) % 2) = (LENGTH(color) % 2))
);

CREATE TABLE

--Check if pid can be null
db107=> INSERT INTO parts(pname, color) VALUES ('test','abc-123');

psql:A5.sql:10: ERROR:  null value in column "pid" violates not-null constraint
DETAIL:  Failing row contains (null, test, abc-123).

--Check if pname can be null
db107=> INSERT INTO parts(pid, color) VALUES (2,'abc-123');

psql:A5.sql:12: ERROR:  null value in column "pname" violates not-null constraint
DETAIL:  Failing row contains (2, null, abc-123).

--Check if color can be null
db107=> INSERT INTO parts(pid, pname) VALUES (2, 'test');

psql:A5.sql:14: ERROR:  null value in column "color" violates not-null constraint
DETAIL:  Failing row contains (2, test, null).

--Check incorrect color format
db107=> INSERT INTO parts(pid, pname, color) VALUES (2, 'test', '123-ab');

psql:A5.sql:16: ERROR:  new row for relation "parts" violates check constraint
"parts_color_check"
DETAIL:  Failing row contains (2, test, 123-ab).

--Check incorrect color format
db107=> INSERT INTO parts(pid, pname, color) VALUES (2, 'test', 'ab');

psql:A5.sql:18: ERROR:  new row for relation "parts" violates check constraint
"parts_color_check"
DETAIL:  Failing row contains (2, test, ab).
```

```
--Check incorrect color format
db107=> INSERT INTO parts(pid, pname, color) VALUES (2, 'test', '1');

psql:A5.sql:20: ERROR:  new row for relation "parts" violates check constraint "parts_check"
DETAIL:  Failing row contains (2, test, 1).

--Check incorrect color format
db107=> INSERT INTO parts(pid, pname, color) VALUES (2, 'test', '');

psql:A5.sql:22: ERROR:  new row for relation "parts" violates check constraint
"parts_color_check"
DETAIL:  Failing row contains (2, test, ).

--Check incorrect color format
db107=> INSERT INTO parts(pid, pname, color) VALUES (2, 'test', 'a2bc-123');

psql:A5.sql:24: ERROR:  new row for relation "parts" violates check constraint
      "parts_color_check"
DETAIL:  Failing row contains (2, test, a2bc-123).

--Check if pname and color can be of different odd and even classes
db107=> INSERT INTO parts(pid, pname, color) VALUES (2, 'test', 'abc-123');

psql:A5.sql:26: ERROR:  new row for relation "parts" violates check constraint "parts_check"
DETAIL:  Failing row contains (2, test, abc-123).

--Insert even pname and color length
db107=> INSERT INTO parts(pid, pname, color) VALUES (1, 'Nail', 'N-11');

INSERT 0 1

--Insert odd pname and color length
db107=> INSERT INTO parts(pid, pname, color) VALUES (2, 'Hammer', 'H-11');

INSERT 0 1

--Insert correct rows
db107=> INSERT INTO parts(pid, pname, color) VALUES (3, 'Screw', 'S-1');

INSERT 0 1

db107=> INSERT INTO parts(pid, pname, color) VALUES (4, 'Drill', 'D-1');

INSERT 0 1

db107=> INSERT INTO parts(pid, pname, color) VALUES (5, 'Knife', 'K-1');

INSERT 0 1

db107=> INSERT INTO parts(pid, pname, color) VALUES (6, 'Bracket', 'B-1');
```

```
INSERT 0 1

db107=> INSERT INTO parts(pid, pname, color) VALUES (7, 'Glue', 'G-11');

INSERT 0 1

db107=> SELECT * FROM parts;

pid  |  pname  | color
-----+---------+-------
   1 | Nail    | N-11
   2 | Hammer  | H-11
   3 | Screw   | S-1
   4 | Drill   | D-1
   5 | Knife   | K-1
   6 | Bracket | B-1
   7 | Glue    | G-11
(7 rows)

db107=> CREATE TABLE partshistory(
    pid             integer         NOT NULL,
    pname           varchar(40)     NOT NULL,
    color           varchar(20)     NOT NULL        CHECK (color ~ '^[a-zA-Z]+[-][0-9]+'),
    operation       CHAR(1)         NOT NULL,
    opwhen          TIMESTAMP       NOT NULL,
    opuser          CHAR(20)        NOT NULL,
    PRIMARY KEY(pid),
    check ((LENGTH(pname) % 2) = (LENGTH(color) % 2))
);

CREATE TABLE

db107=> CREATE OR REPLACE FUNCTION track_parts_history() RETURNS TRIGGER AS $keep_part_history$
    BEGIN
        IF (TG_OP = 'DELETE') THEN
            INSERT INTO partshistory SELECT OLD.*, 'D', now(), user;
        ELSIF (TG_OP = 'UPDATE') THEN
            INSERT INTO partshistory SELECT OLD.*, 'U', now(), user;
        ELSIF (TG_OP = 'INSERT') THEN
            INSERT INTO partshistory SELECT NEW.*, 'I', now(), user;
        END IF;
        RETURN NULL;
    END;
$keep_part_history$ LANGUAGE plpgsql;

CREATE FUNCTION

db107=> CREATE TRIGGER keep_part_history
AFTER INSERT OR UPDATE OR DELETE ON parts
```

```
    FOR EACH ROW EXECUTE PROCEDURE track_parts_history();

CREATE TRIGGER

db107=> SELECT * FROM partshistory;

pid | pname | color | operation | opwhen | opuser
-----+-------+-------+-----------+--------+--------
(0 rows)

db107=> DELETE FROM parts WHERE pid = 1;

DELETE 1

db107=> SELECT * FROM parts;

pid  |  pname  | color
-----+---------+-------
   2 | Hammer  | H-11
   3 | Screw   | S-1
   4 | Drill   | D-1
   5 | Knife   | K-1
   6 | Bracket | B-1
   7 | Glue    | G-11
(6 rows)

db107=> SELECT * FROM partshistory;

pid  | pname | color | operation |           opwhen           |        opuser
-----+-------+-------+-----------+----------------------------+----------------------
   1 | Nail  | N-11  | D         | 2020-11-16 20:56:22.323635 | amaanmakhani
(1 row)

db107=> UPDATE parts SET pid = pid * 2 WHERE pid = 7;

UPDATE 1

db107=> SELECT * FROM parts;

pid  |  pname  | color
-----+---------+-------
   2 | Hammer  | H-11
   3 | Screw   | S-1
   4 | Drill   | D-1
   5 | Knife   | K-1
   6 | Bracket | B-1
  14 | Glue    | G-11
(6 rows)

db107=> SELECT * FROM partshistory;
```

```
pid  | pname | color | operation |           opwhen           |       opuser
-----+-------+-------+-----------+----------------------------+---------------------
   1 | Nail  | N-11  | D         | 2020-11-16 20:56:22.323635 | amaanmakhani
   7 | Glue  | G-11  | U         | 2020-11-16 20:56:22.3287   | amaanmakhani
(2 rows)

db107=> INSERT INTO parts(pid, pname, color) VALUES (8, 'Nail', 'N-11');

INSERT 0 1

db107=> SELECT * FROM parts;

pid  |  pname  | color
-----+---------+-------
   2 | Hammer  | H-11
   3 | Screw   | S-1
   4 | Drill   | D-1
   5 | Knife   | K-1
   6 | Bracket | B-1
  14 | Glue    | G-11
   8 | Nail    | N-11
(7 rows)

db107=> SELECT * FROM partshistory;

pid  | pname | color | operation |           opwhen           |       opuser
-----+-------+-------+-----------+----------------------------+---------------------
   1 | Nail  | N-11  | D         | 2020-11-16 20:56:22.323635 | amaanmakhani
   7 | Glue  | G-11  | U         | 2020-11-16 20:56:22.3287   | amaanmakhani
   8 | Nail  | N-11  | I         | 2020-11-16 20:56:22.33165  | amaanmakhani
(3 rows)
```