HW: Week 12

36-350 – Statistical Computing

Week 12 – Spring 2022

Name: Sophia Li

Andrew ID: sophiali

You must submit **your own** lab as a knitted PDF file on Gradescope.

This week's homework is a little different. Here you will be working with `SQL`, specifically with the variant `postgres`. You will do your work "remotely" in a `postgres` terminal and cut-and-paste your answers into plain code blocks below:

```
This is a plain code block. Note the lack of a {r} above.
Try to avoid issues of text going off the page by utilizing
line breaks.
```

Cut-and-paste both your input command and the output. *If the output has many rows, it is sufficient to cut-and-paste the first five, unless otherwise instructed.*

Question 1

(10 points)

Notes 12A (7-11) + Notes 12B (3)

Create a table dubbed `rdata` that has five columns: `id` (type `serial primary key`), `a` and `b` (consisting of strings that should be no more than five characters), `moment` (which is a date), and `x` (which is a number that should have no more than five digits total and two to the right of the decimal point).

```
create table rdata(
  id serial primary key,
  a varchar(5),
  b varchar(5),
  moment date,
  x numeric(5,2)
);
\d
```

Question 2

(10 points)

Notes 12B (4,8)

Delete the table and create it again, with certain constraints: `a` and `b` cannot be null and must be unique; `moment` should have a default value of `2020-01-01` (); and `x` should be larger than zero.

```
drop table rdata;
# output: DROP TABLE

\d
# output: Did not find any relations.

create table rdata(
  id serial primary key,
  a varchar(5) unique not null,
  b varchar(5) unique not null,
  moment date default ('2020-01-01'),
  x numeric(5,2) check (x > 0)
);
```

Question 3

(10 points)

Notes 12A (4)

Use `\d` in combination with the name of your table to display its properties. Copy and paste all output here.

```
postgres=# \d rdata
                                Table "public.rdata"
 Column |         Type          | Collation | Nullable |              Default
--------+-----------------------+-----------+----------+-----------------------------------
 id     | integer               |           | not null | nextval('rdata_id_seq'::regclass)
 a      | character varying(5)  |           | not null |
 b      | character varying(5)  |           | not null |
 moment | date                  |           |          | '2020-01-01'::date
 x      | numeric(5,2)          |           |          |
Indexes:
    "rdata_pkey" PRIMARY KEY, btree (id)
    "rdata_a_key" UNIQUE CONSTRAINT, btree (a)
    "rdata_b_key" UNIQUE CONSTRAINT, btree (b)
Check constraints:
    "rdata_x_check" CHECK (x > 0::numeric)
```

Question 4

(10 points)

Notes 12B (5)

Insert three rows of data into your table. Do this without explicity referring to the column `id` or `moment`. Display your table using the command `select * from rdata`. You should see a default date in your `moment` column!

```
insert into rdata (a, b, x) values
  ('hi', 'bye', 3.14),
  ('no', '333', 555.55),
  ('aspen', 'birch', 5.67);

postgres=# select * from rdata;
```

```
 id |   a   |   b   |   moment   |   x
----+-------+-------+------------+--------
  1 | hi    | bye   | 2020-01-01 |   3.14
  2 | no    | 333   | 2020-01-01 | 555.55
  3 | aspen | birch | 2020-01-01 |   5.67
(3 rows)
```

Question 5

(10 points)

Notes 12B (5)

Attempt to add another row of data with a negative value for `x`. Show what happens. (If the row is added... that's bad. It means you don't have the constraint `x` > 0 defined.) Afterwords, add a valid row of data, and show the table. Is there anything weird?

```
insert into rdata (a, b, x) values
  ('aaa', 'bbb', -3.14);

postgres=# insert into rdata (a, b, x) values
postgres-#   ('aaa', 'bbb', -3.14);
ERROR:  new row for relation "rdata" violates check constraint "rdata_x_check"
DETAIL:  Failing row contains (4, aaa, bbb, 2020-01-01, -3.14).


insert into rdata (a, b, x) values
  ('aaa', 'bbb', 1.001);

postgres=# insert into rdata (a, b, x) values
postgres-#   ('aaa', 'bbb', 1.001);
INSERT 0 1
postgres=# select * from rdata;
 id |   a   |   b   |   moment   |   x
----+-------+-------+------------+--------
  1 | hi    | bye   | 2020-01-01 |   3.14
  2 | no    | 333   | 2020-01-01 | 555.55
  3 | aspen | birch | 2020-01-01 |   5.67
  5 | aaa   | bbb   | 2020-01-01 |   1.00
(4 rows)

The id goes from 3 to 5. The rejected row still made the id increase.
```

Question 6

(10 points)

Notes 12B (6)

Change the table `rdata` so as to change the data in `moment` so that two of the dates are in March 2020, while the other two are not in March 2020. Use `where` as shown in the notes, and utilize a logical "or" to update two of the dates to the same date in March at the same time. Show your updated table. (Note that the rows may be rearranged. This is OK.)

```
update rdata
  set moment = ('2020-03-01')
  where id >= 3;

postgres=# select * from rdata;
 id |   a   |   b   |   moment   |   x
----+-------+-------+------------+--------
  1 | hi    | bye   | 2020-01-01 |   3.14
  2 | no    | 333   | 2020-01-01 | 555.55
  3 | aspen | birch | 2020-03-01 |   5.67
  5 | aaa   | bbb   | 2020-03-01 |   1.00
(4 rows)
```

Question 7

(10 points)

Notes 12B (7)

Add a new column to `rdata` dubbed `y`, and let it be of `boolean` type with default value `false`. Display your updated table.

```
alter table rdata
  add column y boolean default false;

postgres=# alter table rdata
postgres-#   add column y boolean default false;
ALTER TABLE
postgres=# select * from rdata;
 id |   a   |   b   |   moment   |   x    | y
----+-------+-------+------------+--------+---
  1 | hi    | bye   | 2020-01-01 |   3.14 | f
  2 | no    | 333   | 2020-01-01 | 555.55 | f
  3 | aspen | birch | 2020-03-01 |   5.67 | f
  5 | aaa   | bbb   | 2020-03-01 |   1.00 | f
(4 rows)
```

Question 8

(10 points)

Notes 12B (8)

Remove the row of your table with `id` value 2. Display your updated table.

```
delete from rdata
  where id = 2;

postgres=# delete from rdata
postgres-#   where id = 2;
DELETE 1
postgres=# select * from rdata;
 id |   a   |   b   |   moment   |  x   | y
----+-------+-------+------------+------+---
```

```
  1 | hi    | bye   | 2020-01-01 | 3.14 | f
  3 | aspen | birch | 2020-03-01 | 5.67 | f
  5 | aaa   | bbb   | 2020-03-01 | 1.00 | f
(3 rows)
```

Question 9

(10 points)

Notes 12B (7)

Rename the column `moment` to have the name `date`. Display your updated table.

```
alter table rdata
  rename column moment to date;
```

```
postgres=# alter table rdata
postgres-#   rename column moment to date;
ALTER TABLE
postgres=# select * from rdata;
 id |   a   |   b   |    date    |  x   | y
----+-------+-------+------------+------+---
  1 | hi    | bye   | 2020-01-01 | 3.14 | f
  3 | aspen | birch | 2020-03-01 | 5.67 | f
  5 | aaa   | bbb   | 2020-03-01 | 1.00 | f
(3 rows)
```

Question 10

(10 points)

Notes 12C (2-4)

Download the file `GalaxyStatistics.txt` from the `DATA` directory on `Canvas`. This file contains three columns: the sky field name, the Gini coefficient value, and the concentration statistic value for each of 8,358 observed galaxies. (Feel free to call the concentration statistic column `conc` for short.) Copy it into a `postgres` session to populate a table named `galaxies`. You should add explicit checks that ensure that `gini` and `conc` have values greater than zero. Hint: you'll have to explicitly provide a `delimiter` value here.

```
create table galaxies (
  field text,
  gini numeric check (gini > 0),
  conc numeric check (conc > 0)
);

\copy galaxies from 'C:\Users\sop12\Downloads\GalaxyStatistics.txt' with (format csv, header, delimiter
```

```
postgres=# create table galaxies (
postgres(#   field text,
postgres(#   gini numeric check (gini > 0),
postgres(#   conc numeric check (conc > 0)
postgres(# );
CREATE TABLE
postgres=# \copy galaxies from 'C:\Users\sop12\Downloads\GalaxyStatistics.txt' with (format csv, header
COPY 8358
```

Question 11

(10 points)

Notes 12B (8)

Delete all rows of the table for which the value of `gini` is less than 0.5 or the value of `conc` is greater than 3. You should find that 7,883 rows are deleted. (Show this by showing the output of your `SQL` command.)

```
delete from galaxies
  where gini < 0.5 or conc > 3;
```

```
postgres=# delete from galaxies
postgres-#   where gini < 0.5 or conc > 3;
DELETE 7883
```

Question 12

(10 points)

Notes 12B (6-7)

Alter the table to create a new variable `cg_rat` that is the ratio of `conc` to `gini`. Then display all the rows where the value of `cg_rat` is greater than 5.9. (This last part is accomplished by combining `select` with `where` in a way that should be hopefully becoming somewhat intuitive.) You should have nine rows of output overall.

```
alter table galaxies
  add column cg_rat numeric;

update galaxies
  set cg_rat = conc/gini;

select * from galaxies where cg_rat > 5.9;
```

```
postgres=# alter table galaxies
postgres-#   add column cg_rat numeric;
ALTER TABLE
postgres=# update galaxies
postgres-#   set cg_rat = conc/gini;
UPDATE 475
postgres=# select * from galaxies where cg_rat > 5.9;
 field  |       gini        |       conc       |       cg_rat
--------+-------------------+------------------+--------------------
 COSMOS | 0.505315816622831 |  2.9826252928467 | 5.9024973981230812
 COSMOS | 0.500590685140458 | 2.97597832910617 | 5.9449334904646828
 COSMOS |  0.50361598800351 |   2.987334888289 | 5.9317713485064725
 EGS    |  0.50042450368941 | 2.98645170853111 | 5.9678366796855743
 COSMOS | 0.500671122036005 | 2.97465381069856 | 5.9413329025288627
 COSMOS | 0.503719930512369 | 2.97337141688731 | 5.9028266240386491
 GOODSS | 0.500338113641828 | 2.98312251852409 | 5.9622132257939235
 UDS    | 0.503803338671752 | 2.98048530133007 | 5.9159697297519801
 UDS    | 0.500305684629919 | 2.97898274914929 | 5.9543252069040004
(9 rows)
```