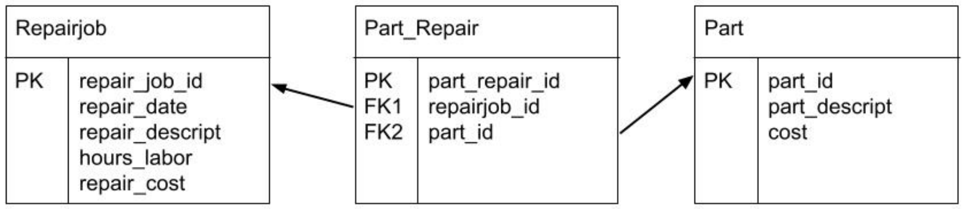
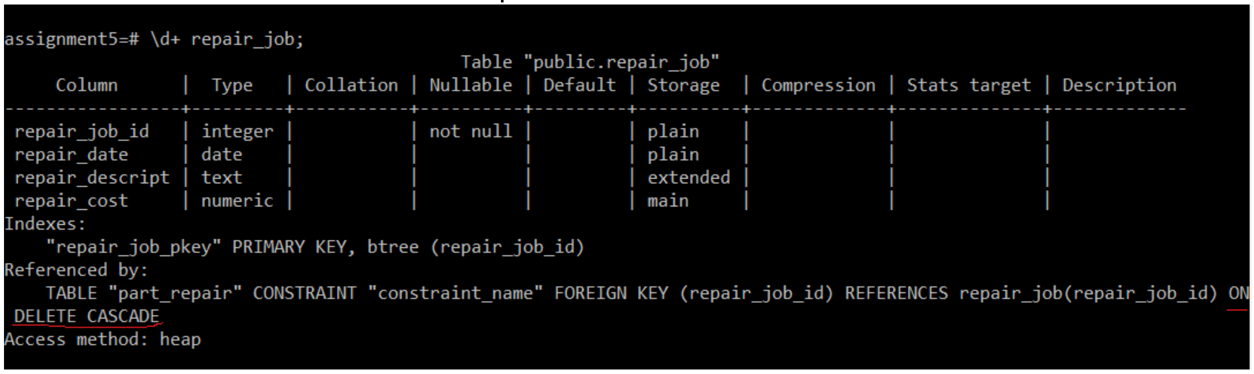
1. Consider for the following schema:

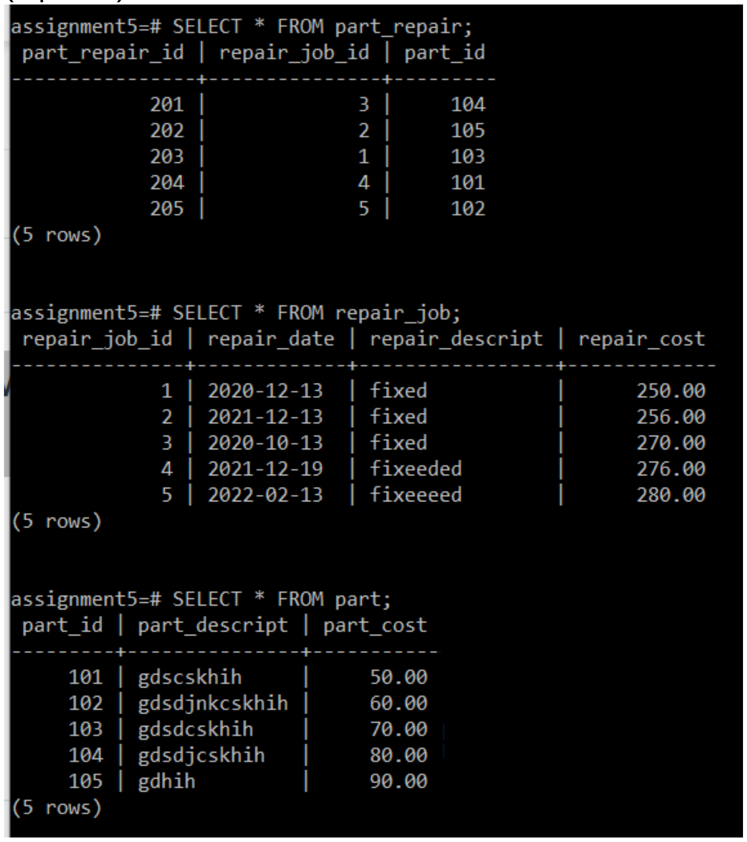


   a. (3 points) Create tables in PostgreSQL for the three tables. Note that the referentially triggered action on the foreign keys in the Repairjob table should be "ON DELETE CASCADE". Make sure to create a textfile with extension .sql first, and then read it into PostgreSQL. If you create the table in psql you will not later be able to extract that sql code.



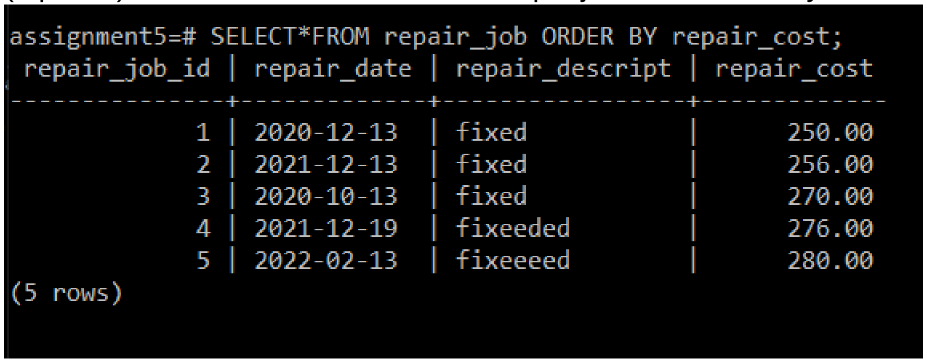
```
assignment5=# \d+ repair_job;
                                              Table "public.repair_job"
     Column      |  Type   | Collation | Nullable | Default | Storage  | Compression | Stats target | Description
-----------------+---------+-----------+----------+---------+----------+-------------+--------------+-------------
 repair_job_id   | integer |           | not null |         | plain    |             |              |
 repair_date     | date    |           |          |         | plain    |             |              |
 repair_descript | text    |           |          |         | extended |             |              |
 repair_cost     | numeric |           |          |         | main     |             |              |
Indexes:
    "repair_job_pkey" PRIMARY KEY, btree (repair_job_id)
Referenced by:
    TABLE "part_repair" CONSTRAINT "constraint_name" FOREIGN KEY (repair_job_id) REFERENCES repair_job(repair_job_id) ON
 DELETE CASCADE
Access method: heap
```

   b. (3 points) Insert at least 5 records into each of the tables



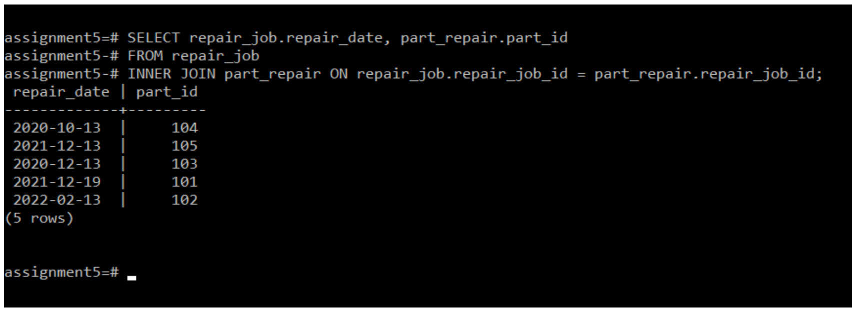
```
assignment5=# SELECT * FROM part_repair;
 part_repair_id | repair_job_id | part_id
----------------+---------------+---------
            201 |             3 |     104
            202 |             2 |     105
            203 |             1 |     103
            204 |             4 |     101
            205 |             5 |     102
(5 rows)


assignment5=# SELECT * FROM repair_job;
 repair_job_id | repair_date | repair_descript | repair_cost
---------------+-------------+-----------------+-------------
             1 | 2020-12-13  | fixed           |      250.00
             2 | 2021-12-13  | fixed           |      256.00
             3 | 2020-10-13  | fixed           |      270.00
             4 | 2021-12-19  | fixeeded        |      276.00
             5 | 2022-02-13  | fixeeeed        |      280.00
(5 rows)


assignment5=# SELECT * FROM part;
 part_id | part_descript | part_cost
---------+---------------+-----------
     101 | gdscskhih     |     50.00
     102 | gdsdjnkcskhih |     60.00
     103 | gdsdcskhih    |     70.00
     104 | gdsdjcskhih   |     80.00
     105 | gdhih         |     90.00
(5 rows)
```

2. Do the following queries
    a. (2 points) List all information from the Repairjob table sorted by cost



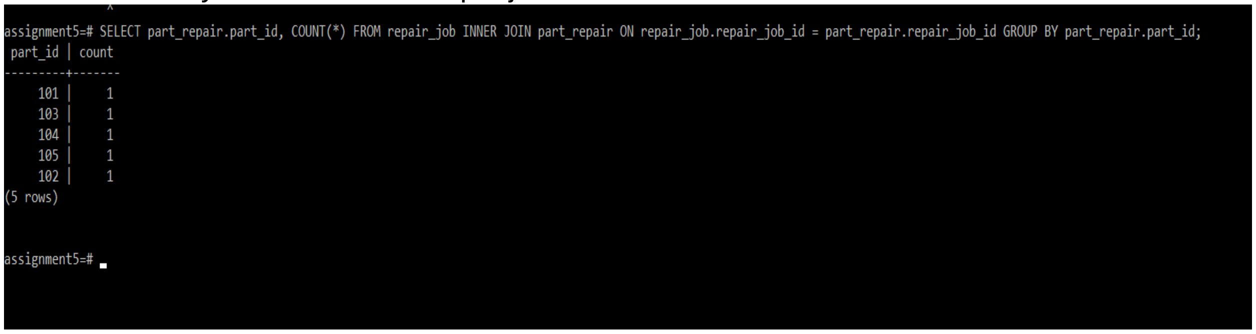
```
assignment5=# SELECT*FROM repair_job ORDER BY repair_cost;
 repair_job_id | repair_date | repair_descript | repair_cost
---------------+-------------+-----------------+-------------
             1 | 2020-12-13  | fixed           |      250.00
             2 | 2021-12-13  | fixed           |      256.00
             3 | 2020-10-13  | fixed           |      270.00
             4 | 2021-12-19  | fixeeded        |      276.00
             5 | 2022-02-13  | fixeeeed        |      280.00
(5 rows)
```

    b. (2 points) List the date (from the Repairjob table) of each use of a part in a Repairjob together with the part_number from the Part table



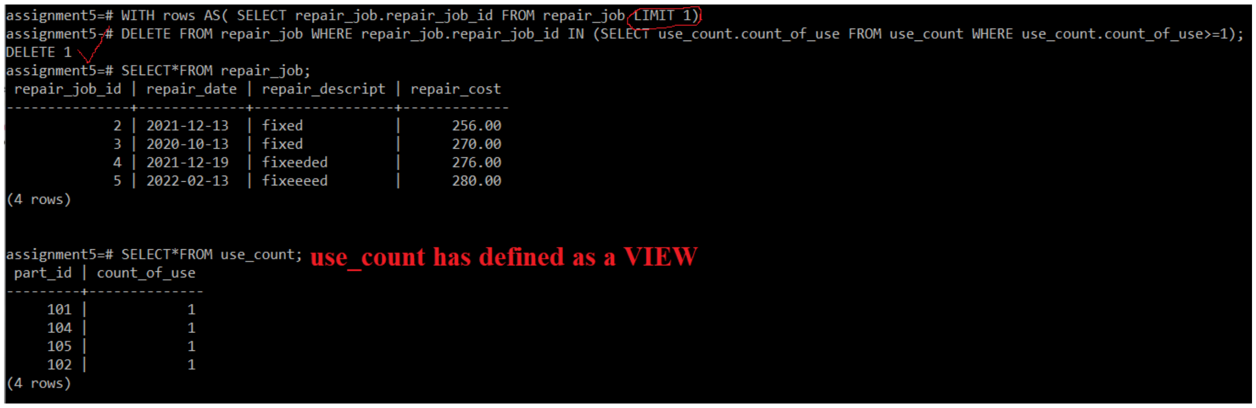
```
assignment5=# SELECT repair_job.repair_date, part_repair.part_id
assignment5-# FROM repair_job
assignment5-# INNER JOIN part_repair ON repair_job.repair_job_id = part_repair.repair_job_id;
 repair_date | part_id
-------------+---------
 2020-10-13  |     104
 2021-12-13  |     105
 2020-12-13  |     103
 2021-12-19  |     101
 2022-02-13  |     102
(5 rows)


assignment5=#
```

    c. (2 points) List all the Parts in the part table together with the number of times they were used in a Repairjob



```
assignment5=# SELECT part_repair.part_id, COUNT(*) FROM repair_job INNER JOIN part_repair ON repair_job.repair_job_id = part_repair.repair_job_id GROUP BY part_repair.part_id;
 part_id | count
---------+-------
     101 |     1
     103 |     1
     104 |     1
     105 |     1
     102 |     1
(5 rows)


assignment5=#
```

    d. (2 points) Delete one record from the Repairjob table that is used at least one part (let's assume the repair never happened for some reason)

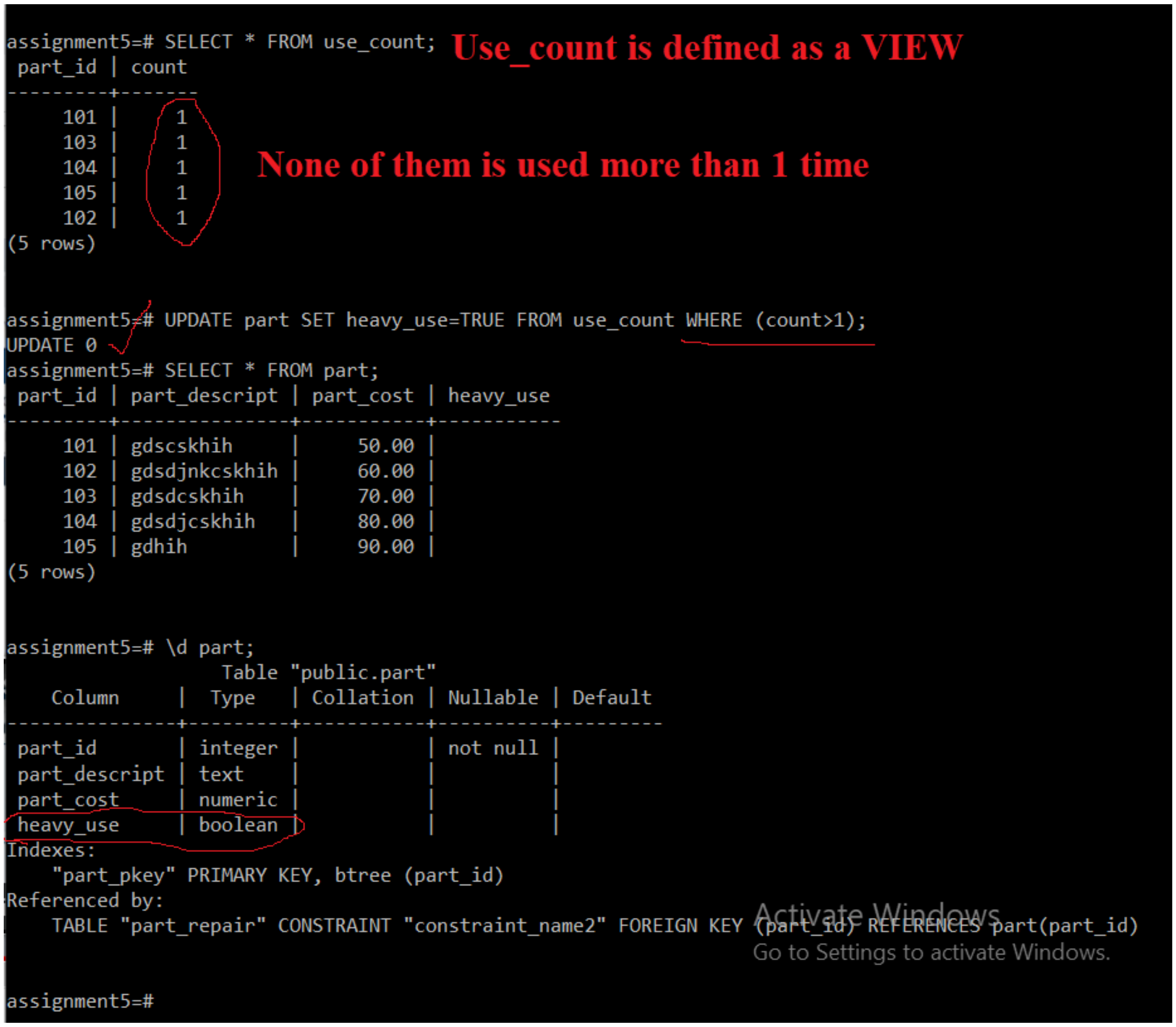


```
assignment5=# WITH rows AS( SELECT repair_job.repair_job_id FROM repair_job LIMIT 1)
assignment5-# DELETE FROM repair_job WHERE repair_job.repair_job_id IN (SELECT use_count.count_of_use FROM use_count WHERE use_count.count_of_use>=1);
DELETE 1
assignment5=# SELECT*FROM repair_job;
 repair_job_id | repair_date | repair_descript | repair_cost
---------------+-------------+-----------------+-------------
             2 | 2021-12-13  | fixed           |      256.00
             3 | 2020-10-13  | fixed           |      270.00
             4 | 2021-12-19  | fixeeded        |      276.00
             5 | 2022-02-13  | fixeeeed        |      280.00
(4 rows)


assignment5=# SELECT*FROM use_count;
 part_id | count_of_use
---------+--------------
     101 |            1
     104 |            1
     105 |            1
     102 |            1
(4 rows)
```

use_count has defined as a VIEW

e. (2 points) Redo the query from c) and show that the number of Repairjobs is decreased accordingly
The answer has shown in the d picture. Before delete, we had 5 rows.
In the picture, we have 4 rows

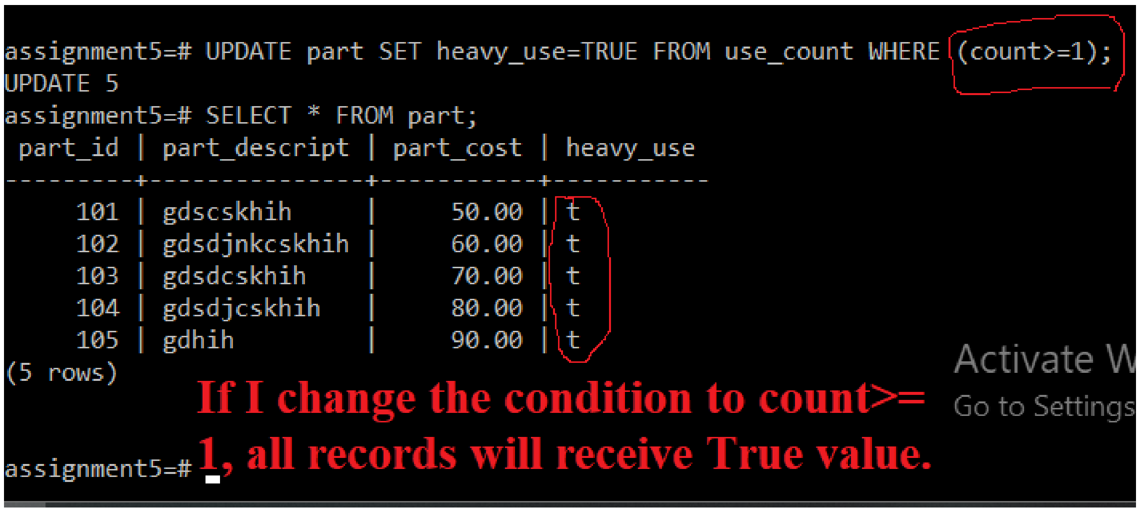
f. (2 points) Add a Boolean type column to the Part table that is called heavy_use. Set it to TRUE for those Parts that have been used more than once



```
assignment5=# SELECT * FROM use_count;
 part_id | count
---------+-------
     101 |     1
     103 |     1
     104 |     1
     105 |     1
     102 |     1
(5 rows)


assignment5=# UPDATE part SET heavy_use=TRUE FROM use_count WHERE (count>1);
UPDATE 0
assignment5=# SELECT * FROM part;
 part_id | part_descript | part_cost | heavy_use
---------+---------------+-----------+-----------
     101 | gdscskhih     |     50.00 |
     102 | gdsdjnkcskhih |     60.00 |
     103 | gdsdcskhih    |     70.00 |
     104 | gdsdjcskhih   |     80.00 |
     105 | gdhih         |     90.00 |
(5 rows)


assignment5=# \d part;
                 Table "public.part"
    Column     |  Type   | Collation | Nullable | Default
---------------+---------+-----------+----------+---------
 part_id       | integer |           | not null |
 part_descript | text    |           |          |
 part_cost     | numeric |           |          |
 heavy_use     | boolean |           |          |
Indexes:
    "part_pkey" PRIMARY KEY, btree (part_id)
Referenced by:
    TABLE "part_repair" CONSTRAINT "constraint_name2" FOREIGN KEY (part_id) REFERENCES part(part_id)

assignment5=#
```

Use_count is defined as a VIEW

None of them is used more than 1 time

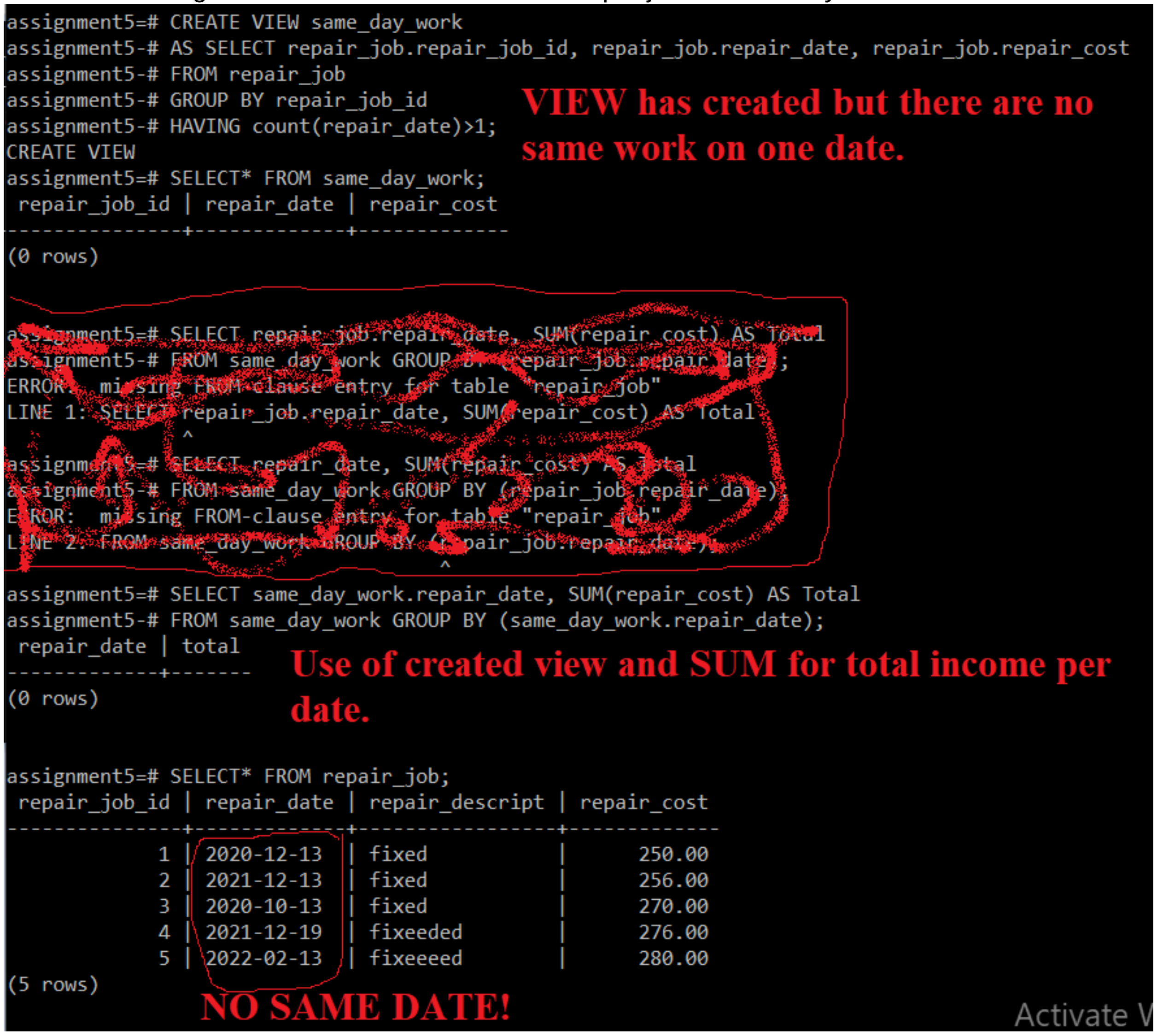


```
assignment5=# UPDATE part SET heavy_use=TRUE FROM use_count WHERE (count>=1);
UPDATE 5
assignment5=# SELECT * FROM part;
 part_id | part_descript | part_cost | heavy_use
---------+---------------+-----------+-----------
     101 | gdscskhih     |     50.00 | t
     102 | gdsdjnkcskhih |     60.00 | t
     103 | gdsdcskhih    |     70.00 | t
     104 | gdsdjcskhih   |     80.00 | t
     105 | gdhih         |     90.00 | t
(5 rows)

assignment5=#
```

If I change the condition to count>= 1, all records will receive True value.

Activate W
Go to Settings

g. (2 points) Create a view that lists the dates on which any Repairjobs happened together with the total cost of the Repairjobs on that day.



```
assignment5=# CREATE VIEW same_day_work
assignment5-# AS SELECT repair_job.repair_job_id, repair_job.repair_date, repair_job.repair_cost
assignment5-# FROM repair_job
assignment5-# GROUP BY repair_job_id
assignment5-# HAVING count(repair_date)>1;
CREATE VIEW
assignment5=# SELECT* FROM same_day_work;
 repair_job_id | repair_date | repair_cost
---------------+-------------+-------------
(0 rows)

[redacted]

assignment5=# SELECT same_day_work.repair_date, SUM(repair_cost) AS Total
assignment5-# FROM same_day_work GROUP BY (same_day_work.repair_date);
 repair_date | total
-------------+-------
(0 rows)


assignment5=# SELECT* FROM repair_job;
 repair_job_id | repair_date | repair_descript | repair_cost
---------------+-------------+-----------------+-------------
             1 | 2020-12-13  | fixed           |      250.00
             2 | 2021-12-13  | fixed           |      256.00
             3 | 2020-10-13  | fixed           |      270.00
             4 | 2021-12-19  | fixeeded        |      276.00
             5 | 2022-02-13  | fixeeeed        |      280.00
(5 rows)
```

VIEW has created but there are no same work on one date.

Use of created view and SUM for total income per date.

NO SAME DATE!

Activate V