ASSIGNMENT 4 : CS432

REPORT

Sachin Yadav 18110148

Directory Structure

```
./
├── filled_db_ques1b.db # database 'x' with dummy records ques 1b
├── filled_db_ques2.db # database 'RandomX' with dummy records ques 2
├── images #images of the output
├── initial_db_ques1b.db # database 'x' with empty tables ques 1b
├── initial_db_ques2.db # database 'RandomX' with empty tables ques 2
├── movie_info.xlsx
├── movie_reviews.xls
├── negative_words.txt # text file with negative words
├── positive_words.txt # text file with positive words
├── ques1b          # contains individual python function files for ques1 b par
│   ├── 01.py
│   ├── 02.py
│   ├── 03.py
│   ├── 05.py
│   ├── 06.py
│   ├── 13.py
│   ├── 16.py
│   ├── 19.py
│   ├── 20.py
│   ├── 21.py
│   └── create_schema.py
├── ques1b.ipynb  # Jupyter Notebook Ques 1b
├── ques2       # contains individual functions for ques 2, try jupyter noteboo
│   ├── a.py
│   ├── b.py
│   ├── c.py
│   └── d.py
├── ques2.ipynb # Jupyter Notebook Ques 2
├── report.html
├── report.md
├── report.pdf # report PDF
└── userdata.xls
```

Question 1b.

Question 5 (Assignment 2):

7 Ques 5 Assignment 2

In [12]:

```
def find_users_reside_city_purchase_greater(city_name, purchase_greater_equal):
    cursor.execute("""
        SELECT DISTINCT phone, email
        FROM users
        where city = %s AND user_id IN (
            SELECT user_id
            FROM orders
            GROUP BY user_id
            HAVING SUM(price) >= %s
        )""", (city_name, purchase_greater_equal));
    user_list = cursor.fetchall()

    for info in user_list:
        print(*info)
```

executed in 11ms, finished 22:25:12 2021-02-23

In [13]:

```
find_users_reside_city_purchase_greater('Madrid', 1000)
```

executed in 23ms, finished 22:25:12 2021-02-23

```
9972934723 sachin@gmail.com
2035753956 kavya@rediffmail.com
9383729512 anmol@yahoo.com
4832840284 andy043@gmai.com
```

Question 6 (Assignment 2):

8 Ques 6 Assignment 2

In [28]:

```
def find_products_like_and_users_bought(like_string):
    cursor.execute("""
        SELECT name, pdt_id, user_id
            FROM (
                SELECT name, pdt_id
                FROM products
                WHERE name LIKE %s
            )P LEFT OUTER JOIN (
                SELECT DISTINCT user_id, pdt_id
                FROM order_items
            )O USING (pdt_id)""", ('%' + like_string + '%', ))
    output_list = cursor.fetchall()
    for row in output_list:
        print(*row)
```

executed in 5ms, finished 22:31:18 2021-02-23

In [29]:

```
find_products_like_and_users_bought('mi')
```

executed in 10ms, finished 22:31:20 2021-02-23

```
Xiaomi S9 Pro 12 None
Xiomi Note 4 13 5
Neomi Watch R2 14 None
Neomi Watch D5 16 20
```

Question 13 (Assignment 2):

9 Ques 13 Assignment 2

In [30]:
```
def electronics_category_increasing_price(elec_category):
    cursor.execute("""
    SELECT pdt_id, electronic_id, name, electronics.category, electronics.brand, electronics.model, pri
    FROM electronics JOIN products USING (pdt_id)
    WHERE electronics.pdt_id = products.pdt_id AND electronics.category = %s
    ORDER BY price ASC""", (elec_category, ))
    output_list = cursor.fetchall()
    for row in output_list:
        print(*row)
```

executed in 5ms, finished 22:31:53 2021-02-23

In [31]:
```
electronics_category_increasing_price('Laptop')
```

executed in 18ms, finished 22:31:54 2021-02-23

```
19 8 Dell Inspiron D3 Laptop Dell D3 45000.00
20 9 Dell Inspiron G4 Laptop Dell G4 55000.00
17 6 HP Paviliion G6 Laptop HP G6 56000.00
18 7 Acer Predator Helio Laptop Acer Helio93 74500.00
21 10 Apple macbook Air Laptop Apple air 85000.00
25 13 Apple macbook Pro Laptop Apple Pro 90000.00
```

Question 16 (Assignment 2):

In [32]:
```
def users_with_cart_prdt_quantity(quantity_less_than):
    cursor.execute("""
        SELECT DISTINCT user_id, username, email, phone
        FROM users
        WHERE user_id in (
            SELECT user_id
            FROM cart_items
            WHERE quantity < %s
        )""", (quantity_less_than, ))
    output_list = cursor.fetchall()
    for row in output_list:
        print(*row)
```

executed in 5ms, finished 22:32:23 2021-02-23

In [33]:
```
users_with_cart_prdt_quantity(5)
```

executed in 16ms, finished 22:32:25 2021-02-23

```
1 Sachin Yadav sachin@gmail.com 9972934723
3 Kavya Agarwal kavya@rediffmail.com 2035753956
2 Anuj Jain anuj@yahoo.com 7502740385
6 Dhiraj Sinha dhiraj@outlook.com 3829893782
7 Kunal Kumar kunal@iitgn.ac.in 6483026593
9 Sushant Kumar sushant.kumar@gmail.com 3792052942
10 Kalyan G. Reddy kalyan.reddy@iitgn.ac.in 3920395343
11 Sushant Goyal sushant.goyal@gmail.com 7583925935
12 Andy Murray andy043@gmai.com 4832840284
13 Nicole Ray nicole58@yahoo.com 8302643252
14 Jon Duse jon380@rediffmail.com 7296937501
```

Question 19 (Assignment 2):

11 Ques 19 Assignment 2

In [34]:
```
def find_retailer_for_user_id(user_id):
    cursor.execute("""
        SELECT DISTINCT retailer_id
        FROM products
        WHERE pdt_id in (
            SELECT pdt_id
            FROM order_items
            WHERE user_id = %s
        )""", (user_id, ))
    output_list = cursor.fetchall()
    for row in output_list:
        print(*row)
```

executed in 6ms, finished 22:32:54 2021-02-23

In [35]:
```
find_retailer_for_user_id(1)
```

executed in 6ms, finished 22:32:56 2021-02-23

```
2
9
3
```

Question 20 (Assignment 2):

```
    mydb.commit()
    cursor.execute("""
        CREATE TABLE holi_Deals
        SELECT *
        FROM products
        WHERE created_at >= DATE_SUB(CURDATE(),INTERVAL 100 DAY)""")
    cursor.execute("""
        UPDATE holi_Deals
        SET price = price * %s""", (1 - discount_percentage/100, ))
    mydb.commit()
```

executed in 6ms, finished 22:33:27 2021-02-23

In [37]:
```
holi_discount_table(15)
```

executed in 105ms, finished 22:33:29 2021-02-23

In [38]:
```
cursor.execute("SELECT * FROM holi_Deals")
output_list = cursor.fetchall()
for row in output_list:
    print(*row)
```

executed in 11ms, finished 22:33:31 2021-02-23

```
4 The Lord of the Rings novels 2 573.75 2020-12-01 01:47:46
25 Apple macbook Pro electronics 9 76500.00 2020-12-30 01:47:46
26 Apple iPad Pro electronics 9 51000.00 2020-11-30 01:47:46
27 Apple iPad Air electronics 9 45900.00 2021-01-30 01:47:46
28 Apple iPad electronics 9 29750.00 2021-02-02 01:47:46
```

Question 21 (Assignment 2):

```
        ) T INNER JOIN products USING(pdt_id)
        WHERE T.user_id = %s AND pdt_id NOT IN (
            SELECT pdt_id
            FROM order_items
            WHERE user_id = %s
        )
        GROUP BY pdt_id
        ORDER BY MAX(T.created_at) DESC
        LIMIT 10""", (user_id, user_id))
    output_list = cursor.fetchall()
    for row in output_list:
        print(*row)
```

executed in 4ms, finished 22:34:10 2021-02-23

In [40]:

```
top_recommendations_user_id(1)
```

executed in 49ms, finished 22:34:12 2021-02-23

```
26 Apple iPad Pro 60000.00
27 Apple iPad Air 54000.00
28 Apple iPad 35000.00
21 Apple macbook Air 85000.00
25 Apple macbook Pro 90000.00
14 Neomi Watch R2 3960.00
16 Neomi Watch D5 11000.00
12 Xiaomi S9 Pro 15000.00
20 Dell Inspiron G4 55000.00
1 Da Vinci Code 450.00
```

Question 2c.

In [24]:

```
cursor.execute("SELECT * FROM movie_reviews")
output_list = cursor.fetchall()
for row in output_list:
    print(*row)
```

executed in 270ms, finished 19:51:43 2021-02-23

```
m/the_two_popes gamma Jonathan Pryce and Anthony Hopkins engage in a theological joust, equal parts levity, humor, and spiritual explorati
on. There is no resolution of millennia-old debates, just amusing exposition and proselytizing positive
m/the_two_popes hockeren Somehow the filmmakers found lightheartedness and - gasp - laughs in a story of political intrigue at the top of
the notoriously buttoned-up Catholic Church. neutral
m/the_two_popes carchery There's an unquestionable appeal to the way the movie transforms a weighty and divisive topic into more approacha
ble terms. positive
m/the_two_popes sentact Despite its over reliance on unnecessary flashbacks, "The Two Popes" is a masterclass in acting delivered by two l
egendary stalwarts of the craft: Pryce and Hopkins. neutral
m/the_two_popes oryoustald Hopkins and Pryce's finely tuned performances illuminate Benedict's shrewd intelligence and Francis's deep humi
lity. positive
m/the_lord_of_the_rings_the_fellowship_of_the_ring ivartherf You think Harry Potter had expectations? neutral
m/the_lord_of_the_rings_the_fellowship_of_the_ring carchery Thrilling -- a great picture, a triumphant picture, a joyfully conceived work
of cinema. positive
m/the_lord_of_the_rings_the_fellowship_of_the_ring gamma For those of you who, like me, were disappointed by Harry Potter, New Line has th
e perfect answer: The Lord of the Rings. positive
m/the_lord_of_the_rings_the_fellowship_of_the_ring oryoustald There are moments in The Fellowship of the Ring that I have only seen in my
most winsome and wildest fantasies. neutral
m/the_lord_of_the_rings_the_fellowship_of_the_ring reoloque Jackson's films are sure to take the "Star Wars" mantle as beloved mythic epic
for a new generation. positive
m/the_signal_2014 hockeren The Signal is a welcome "agitation" that makes some of Hollywood's latest sci-fi blunders nothing but a distant
memory. neutral
m/the_signal_2014 carchery The titular signal refers to the Nomad hacker's taunts, though it may as well point to the film's nature as a s
elf-styled calling card. positive
m/the_signal_2014 gamma William Eubank and co. dream up several stirring images but without the foundation that would allow us to genuinel
y care, it's a hell of a calling card for a gifted stylist, but little more. Still, there's no mistaking ambition. neutral
m/the_signal_2014 reoloque This is a great endorsement of Eubank to take on larger films in the future. Should he get his hands on a more
fleshed-out (and thought-through) screenplay, he may transform and adapt into a world class pop filmmaker. positive
```

Question 2d.

6 Fetching Top N movies above audience rating and positive sentiments

In [27]:

```
def fetch_top_n_movies_audience_rating(n, audience_rating, positive_sentiments):
    cursor.execute("""
        SELECT movie_id, movie_title, audience_rating, COUNT(*) AS 'positive_reviews_count'
        FROM movies NATURAL JOIN movie_reviews
        WHERE sentiment = 'positive'
        GROUP BY movie_id, movie_title, audience_rating
        HAVING COUNT(*) >= %s AND audience_rating > %s
        ORDER BY audience_rating DESC
        LIMIT %s
        """, (positive_sentiments, audience_rating, n) )
    output_list = cursor.fetchall()
    for row in output_list:
        print(*row)
```

executed in 4ms, finished 22:39:18 2021-02-23

In [28]:

```
fetch_top_n_movies_audience_rating(5, 3.50, 2)
```

executed in 9ms, finished 22:39:20 2021-02-23

```
m/the_lord_of_the_rings_the_fellowship_of_the_ring The Lord of the Rings: The Fellowship of the Ring 4.75 3
m/room_2015 Room 4.65 5
m/man_who_shot_liberty_valance The Man Who Shot Liberty Valance 4.6 3
m/run_lola_run Run Lola Run 4.5 2
m/star_trek_into_darkness Star Trek Into Darkness 4.45 5
```