Revision Sheet-3

1. Write a Python program read a text file line by line and display line started by T.

Solution-
```
def process_file(file_path):
    with open(file_path, 'r') as file:
        for line in file:
            if line.startswith('T'): # Check if the line starts with 'T'
                print(line.strip())     # Strip to remove extra whitespace/newlines

file_path = 'your_text_file.txt' # Specify the path to your file (Here you will write the exact file name of your text file/file path)

process_file(file_path) #Call the function
```

2. Define a function printpattern(line), which can print the given pattern.
IF the value of line is 3, then function can print a pattern like
#
##
###

Solution-
```
def printpattern(line):
    for i in range(1, line + 1):
        print('#' * i)
# Example usage: printpattern(3)
```

Consider the table Customer as given below

Id	Name	product	quantity	price
2001	Marry	Laptop	2	15000
2002	Bob	Smartphone	4	10000
2003	Charley	Headphone	8	1000

Write the following SQL queries:

(a)To display the total Quantity for each Product, excluding Products with total Quantity less than 5.

```
SELECT product, SUM(quantity) AS total_quantity
FROM Customer
GROUP BY product HAVING SUM(quantity) >= 5;
```

Explanation:

- We use GROUP BY to group the records by product.
- The SUM(quantity) function calculates the total quantity for each product.
- The HAVING clause filters the results to include only products where the total quantity is greater than or equal to 5.

(b) To display the customer table sorted by total price in descending order.

```
SELECT *, (quantity * price) AS total_price
FROM Customer
ORDER BY total_price DESC;
```

Explanation:

- We calculate the total price for each product by multiplying quantity by price and alias it as total_price.
- The ORDER BY clause sorts the results by total_price in descending order (DESC).

(c) To display the distinct customer names from the Orders table.

```
SELECT DISTINCT Name FROM Orders;
```

Explanation:

- The DISTINCT keyword ensures that only unique customer names are displayed from the Orders table.

(d) Display the sum of Price of all the customer for which the quantity is null.

```
SELECT SUM(price) AS total_price_with_null_quantity
FROM Customer
WHERE quantity IS NULL;
```

Explanation:

- The WHERE clause filters records where the quantity is NULL.
- The SUM(price) function calculates the total price for these records.

Note: The provided table Customer does not have any records with NULL values for quantity, so the result would be NULL unless there are such records in the actual database.