SQL Code

```sql
USE DS63;
#DML ---> SORTING,GROUP BY,LIMIT,OFFSET,DELETE,UPDATE
#TCL-----> ROLLBACK , COMMIT
# SORTING QUESTIONS ----------------------------------------------------------------
SELECT * FROM SMARTPHONES;
-- 1. FIND TOP 5 SAMSUNG PHONES WITH BIGGEST SCREEN SIZE
SELECT MODEL,SCREEN_SIZE FROM SMARTPHONES WHERE BRAND_NAME='SAMSUNG' ORDER BY SCREEN_SIZE DESC LIMIT 5;
-- 2. SORT ALL THE PHONE WITH IN DECENDING ORDER OF NUMBER OF TOTAL CAMERAS
SELECT MODEL,(NUM_REAR_CAMERAS+NUM_FRONT_CAMERAS) AS TOTAL_CAMERAS FROM SMARTPHONES ORDER BY (NUM_REAR_CAMERAS+NUM_FRONT_CAMERAS) DESC;
-- 3. SORT DATA ON THE BASIS OF PPI IN DECEASING ORDER
SELECT MODEL,ROUND(SQRT(POW(resolution_width,2)+POW(resolution_height,2)/SCREEN_SIZE),2) AS PPI FROM SMARTPHONES ORDER BY PPI DESC;
-- 4. FIND THE PHONE WITH 2ND LARGEST BATTERY (INTERVIEW QUESTION)
SELECT MODEL , BATTERY_CAPACITY FROM SMARTPHONES ORDER BY BATTERY_CAPACITY DESC LIMIT 1 OFFSET 1;
-- 5. FIND THE NAME AND RATING OF THE WORST RATED APPLE PHONE
SELECT MODEL, RATING FROM SMARTPHONES WHERE BRAND_NAME = 'APPLE' ORDER BY RATING ASC LIMIT 1;
-- 6. SORT PHONE ALPHABETICALLY
SELECT BRAND_NAME,MODEL FROM SMARTPHONES ORDER BY BRAND_NAME ASC;
-- 7. SORT PHONES ALPHABETICALLY AND THEN ON THE BASIC OF PRICE IN ASC ORDER
SELECT BRAND_NAME, MODEL,PRICE FROM SMARTPHONES ORDER BY BRAND_NAME ASC, PRICE ASC;

-- ------------------------------------------------------------------------------------------------------------

-- GROUPING DATA --> AGGREGATION
# GROUP 5 PEOPLE --> PAISE
# SUM --> 500
# AVG --> 100

-- 1. GROUP SMARTPHONES BY BRAND AND GET THE COUNT , AVERAGE PRICE,
-- MAX RATING , AVG SCREEN SIZE AND AVG BATTERY CAPACITY
-- ---> group by animation
SELECT BRAND_NAME,COUNT(MODEL) AS TOTAL_PHONES,AVG(PRICE) AS AVG_PRICE,
MAX(RATING) AS MAX_RATING , AVG(SCREEN_SIZE) AS AVG_SCREEN_SIZE,AVG(BATTERY_CAPACITY)
AS AVG_BATTERY_CAPACITY FROM SMARTPHONES GROUP BY BRAND_NAME;
-- 2. GROUP SMARTPHONES BY WHEATER THEY HAVE AN NFC AND GET THE
-- average price
SELECT HAS_NFC,AVG(PRICE),AVG(RATING) FROM SMARTPHONES GROUP BY HAS_NFC;
-- 3. GROUP SMARTPHONES BY THE EXTENDED MEMORY AVAILABLE AND GET
 -- --     THE AVERAGE PRICE

SELECT extended_memory_available,AVG(PRICE)
FROM SMARTPHONES GROUP BY extended_memory_available;
-- 4.GROUP SMARTPHONES BY THE BRAND AND PROCESSOR  BRAND AND GET
-- THE COUNT OF MODELS AND THE AVERAGE PRIMARY CAMERA RESOLUTION
-- (REAL*)
```

```
45
46
47    SELECT BRAND_NAME,PROCESSOR_BRAND,
48    COUNT(MODEL),AVG(primary_camera_REAR) FROM SMARTPHONES
49    GROUP BY BRAND_NAME,PROCESSOR_BRAND;
50
51    # HAVING --> WHERE
52    SELECT BRAND_NAME,PROCESSOR_BRAND,
53    COUNT(MODEL),AVG(primary_camera_REAR) FROM SMARTPHONES
54    GROUP BY BRAND_NAME,PROCESSOR_BRAND HAVING COUNT(MODEL)>20;
55
56    -- 5. FIND TOP 5 MOST COSTLY PHONE BRANDS
57    SELECT BRAND_NAME, MAX(PRICE) AS MAX_PRICE FROM SMARTPHONES GROUP BY BRAND_NAME ORDE
R BY MAX_PRICE DESC LIMIT 5;
58    -- ------------------------------------------------------------------------------
------------------------------------------------------------------------
59    -- UPDATE -----------------------------------------
60    SET SQL_SAFE_UPDATES = 0;
61    SET AUTOCOMMIT = 0;
62    #ROLLBACK(CTRL+Z) AND COMMIT(ACCEPT) --> TCL
63    -- UPDATE QUERY------
64    START TRANSACTION;
65    -- UPDATE THE PRICE OF ALL SAMSUNG PHONES BY INCREASING IT BY 10%
66    UPDATE SMARTPHONES
67    SET PRICE = PRICE*1.10
68    WHERE BRAND_NAME='SAMSUNG';
69
70    SELECT MODEL,PRICE FROM SMARTPHONES WHERE BRAND_NAME = 'SAMSUNG';
71    ROLLBACK;
72    -- ------------------------------------------------------------------------
73    -- UPDATE THE RATING TO 45 FOR ALL PHONES THAT HAVE 5G SUPPORTY AND COST MORE THAN 5
0000
74    UPDATE SMARTPHONES
75    SET RATING = 45
76    WHERE HAS_5G = 'TRUE' AND PRICE>50000;
77
78    SELECT * FROM SMARTPHONES WHERE HAS_5G = 'TRUE' AND PRICE> 50000;
79    -- UPDATE THE HAS_NFC  VALUE TO FALSE FOR ALL PHONES WHOSE PRICE IS LESS THAN 15000.
80    UPDATE SMARTPHONES
81    SET HAS_NFC='FALSE'
82    WHERE PRICE<1500;
83    -- UPDATE THE BATTERY_ CAPACITY TO 5000 FOR ALL PHONES RUNNING ON
84    -- ANDROID OS AND HAVING BATTERY CAPACITY LESS THAN 5000.
85    SELECT OS FROM SMARTPHONES;
86    UPDATE SMARTPHONES
87    SET BATTERY_CAPACITY = 5000
88    WHERE OS = 'ANDROID' AND BATTERY_CAPACITY<5000;
89
90
91    -- ------------------------------------------------------------------------------
------------------------------------------------------------------
92    -- == ================================================== DELETE QUESTIONS ======
==========================================================================
93
```