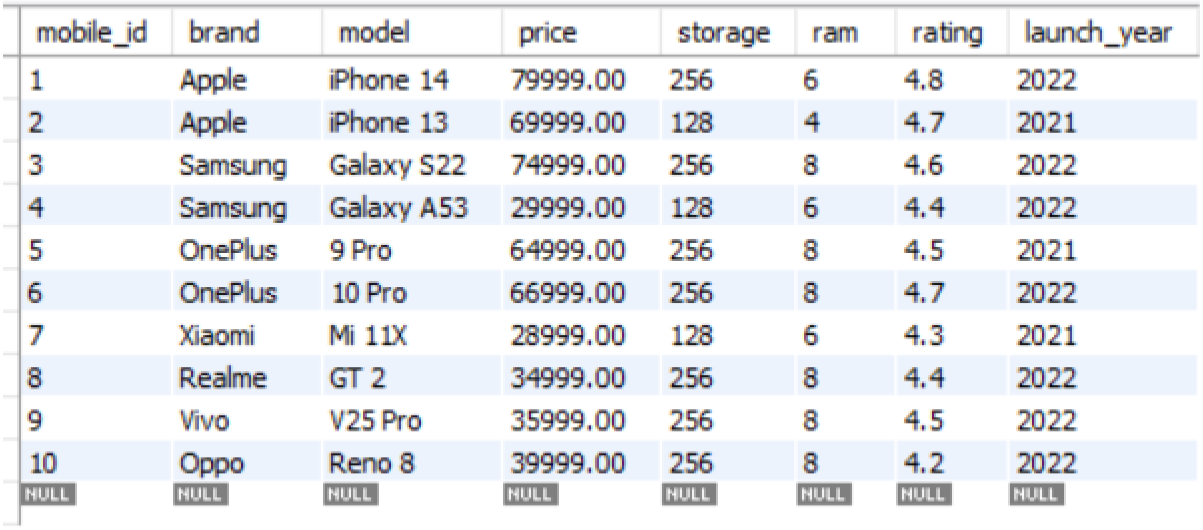
```
-- Create table
CREATE TABLE mobiles (
    mobile_id INT PRIMARY KEY,
    brand VARCHAR(30),
    model VARCHAR(50),
    price DECIMAL(10,2),
    storage INT,
    ram INT,
    rating DECIMAL(3,1),
    launch_year INT
);

-- Sample data
INSERT INTO mobiles VALUES
(1, 'Apple', 'iPhone 14', 79999, 256, 6, 4.8, 2022),
(2, 'Apple', 'iPhone 13', 69999, 128, 4, 4.7, 2021),
(3, 'Samsung', 'Galaxy S22', 74999, 256, 8, 4.6, 2022),
(4, 'Samsung', 'Galaxy A53', 29999, 128, 6, 4.4, 2022),
(5, 'OnePlus', '9 Pro', 64999, 256, 8, 4.5, 2021),
(6, 'OnePlus', '10 Pro', 66999, 256, 8, 4.7, 2022),
(7, 'Xiaomi', 'Mi 11X', 28999, 128, 6, 4.3, 2021),
(8, 'Realme', 'GT 2', 34999, 256, 8, 4.4, 2022),
(9, 'Vivo', 'V25 Pro', 35999, 256, 8, 4.5, 2022),
(10, 'Oppo', 'Reno 8', 39999, 256, 8, 4.2, 2022);
```

```
Select * from mobiles;
```



mobile_id	brand	model	price	storage	ram	rating	launch_year
1	Apple	iPhone 14	79999.00	256	6	4.8	2022
2	Apple	iPhone 13	69999.00	128	4	4.7	2021
3	Samsung	Galaxy S22	74999.00	256	8	4.6	2022
4	Samsung	Galaxy A53	29999.00	128	6	4.4	2022
5	OnePlus	9 Pro	64999.00	256	8	4.5	2021
6	OnePlus	10 Pro	66999.00	256	8	4.7	2022
7	Xiaomi	Mi 11X	28999.00	128	6	4.3	2021
8	Realme	GT 2	34999.00	256	8	4.4	2022
9	Vivo	V25 Pro	35999.00	256	8	4.5	2022
10	Oppo	Reno 8	39999.00	256	8	4.2	2022
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
-- □ 1) Find top 3 brands with the highest average price.

SELECT brand, AVG(price) AS avg_price

FROM mobiles

GROUP BY brand

ORDER BY avg_price DESC

LIMIT 3;
```

```
-- □ 2) Show brands that have more than 1 model launched after 2021.

SELECT brand, COUNT(*) AS model_count

FROM mobiles

WHERE launch_year > 2021

GROUP BY brand

HAVING COUNT(*) > 1;
```

```
-- □ 3) Display top 5 brands with the highest average rating above 4.5.

SELECT brand, AVG(rating) AS avg_rating

FROM mobiles

GROUP BY brand

HAVING AVG(rating) > 4.5

ORDER BY avg_rating DESC
```

```
LIMIT 5;

-- □ 4) List brands along with their total mobile count, ordered by total count descending.
SELECT brand, COUNT(*) AS total_mobiles
FROM mobiles
GROUP BY brand
ORDER BY total_mobiles DESC;

-- □ 5) Find brands whose total storage capacity exceeds 500 GB, ordered by total storage descending.
SELECT brand, SUM(storage) AS total_storage
FROM mobiles
GROUP BY brand
HAVING SUM(storage) > 500
ORDER BY total_storage DESC;

-- □ 6) Show the 5 cheapest average-priced brands, skipping the first 2 results.
SELECT brand, AVG(price) AS avg_price
FROM mobiles
GROUP BY brand
ORDER BY avg_price ASC
LIMIT 5 OFFSET 2;

-- □ 7) Display top 5 brands with the most models rated above 4.5.
SELECT brand, COUNT(*) AS high_rating_models
FROM mobiles
WHERE rating > 4.5
GROUP BY brand
ORDER BY high_rating_models DESC
LIMIT 5;
```

```sql
-- □ 8) Find top 3 brands whose average price is above ₹50,000 and order by their average rating.
SELECT brand, AVG(price) AS avg_price, AVG(rating) AS avg_rating
FROM mobiles
GROUP BY brand
HAVING AVG(price) > 50000
ORDER BY avg_rating DESC
LIMIT 3;

-- □ 9) Find average RAM per brand where models launched after 2021.
SELECT brand, AVG(ram) AS avg_ram
FROM mobiles
WHERE launch_year > 2021
GROUP BY brand
ORDER BY avg_ram DESC;

-- □ 10) Show total price spent on all Samsung phones.
SELECT brand, SUM(price) AS total_spent
FROM mobiles
WHERE brand = 'Samsung'
GROUP BY brand;

-- □ 11) Find brands with maximum storage capacity per brand.
SELECT brand, MAX(storage) AS max_storage
FROM mobiles
GROUP BY brand;

-- □ 12) List brands with an average rating between 4.4 and 4.7.
SELECT brand, AVG(rating) AS avg_rating
FROM mobiles
```

```
GROUP BY brand
HAVING AVG(rating) BETWEEN 4.4 AND 4.7;

-- □ 13) Count how many phones were launched each year.
SELECT launch_year, COUNT(*) AS total_phones
FROM mobiles
GROUP BY launch_year
ORDER BY launch_year;

-- □ 14) Find top 3 models with the highest price-to-rating ratio.
SELECT model, (price / rating) AS price_per_rating
FROM mobiles
ORDER BY price_per_rating DESC
LIMIT 3;

-- □ 15) Find all brands that have at least one model with rating below 4.5.
SELECT DISTINCT brand
FROM mobiles
WHERE brand IN (
  SELECT brand FROM mobiles WHERE rating < 4.5
);

-- □ 16) Show brands and total models having price above the average price.
SELECT brand, COUNT(*) AS premium_models
FROM mobiles
WHERE price > (SELECT AVG(price) FROM mobiles)
GROUP BY brand;

-- □ 17) Find the cheapest model per brand using subquery.
SELECT brand, model, price
```

```
FROM mobiles m1
WHERE price = (
  SELECT MIN(price)
  FROM mobiles m2
  WHERE m1.brand = m2.brand
);

-- ⬚ 18) Find the latest model launched per brand.
SELECT brand, MAX(launch_year) AS latest_launch
FROM mobiles
GROUP BY brand;

-- ⬚ 19) List models where price is above average for that brand.
SELECT m1.brand, m1.model, m1.price
FROM mobiles m1
WHERE price > (
  SELECT AVG(m2.price)
  FROM mobiles m2
  WHERE m2.brand = m1.brand
);

-- ⬚ 20) Find the overall average price per year, ordered by year descending.
SELECT launch_year, AVG(price) AS avg_price_per_year
FROM mobiles
GROUP BY launch_year
ORDER BY launch_year DESC;
```