```
USE cleaning_eda;

SELECT * FROM laptop;

-- head
SELECT * FROM laptop
ORDER BY `Index` ASC
LIMIT 5;

-- tail
SELECT * FROM laptop
ORDER BY `Index` DESC
LIMIT 5;

-- random
SELECT * FROM laptop
ORDER BY rand()
LIMIT 5;

-- COUNT, MAX, MIN, AVG, STDDEV, Q1, Median/Q2, Q3
-- Resolution_width
SELECT
   COUNT(resolution_width) AS count_resolution_width,
   MAX(resolution_width) AS max_resolution_width,
   MIN(resolution_width) AS min_resolution_width,
   AVG(resolution_width) AS avg_resolution_width,
   STDDEV(resolution_width) AS stddev_resolution_width,
   (SELECT resolution_width FROM (SELECT resolution_width, NTILE(4) OVER (ORDER BY
resolution_width) AS quartile FROM laptop) AS subquery WHERE quartile = 1 LIMIT 1) AS Q1,
   (SELECT resolution_width FROM (SELECT resolution_width, NTILE(2) OVER (ORDER BY
resolution_width) AS median FROM laptop) AS subquery WHERE median = 1 LIMIT 1) AS Q2,
   (SELECT resolution_width FROM (SELECT resolution_width, NTILE(4) OVER (ORDER BY
resolution_width) AS quartile FROM laptop) AS subquery WHERE quartile = 3 LIMIT 1) AS Q3
FROM laptop;

-- resolution_height
SELECT
   COUNT(resolution_height) AS count_resolution_height,
   MAX(resolution_height) AS max_resolution_height,
   MIN(resolution_height) AS min_resolution_height,
   AVG(resolution_height) AS avg_resolution_height,
   STDDEV(resolution_height) AS stddev_resolution_height,
   (SELECT resolution_height FROM (SELECT resolution_height, NTILE(4) OVER (ORDER BY
resolution_height) AS quartile FROM laptop) AS subquery WHERE quartile = 1 LIMIT 1) AS Q1,
    (SELECT resolution_height FROM (SELECT resolution_height, NTILE(2) OVER (ORDER BY
resolution_height) AS median FROM laptop) AS subquery WHERE median = 1 LIMIT 1) AS Q2,
    (SELECT resolution_height FROM (SELECT resolution_height, NTILE(4) OVER (ORDER BY
resolution_height) AS quartile FROM laptop) AS subquery WHERE quartile = 3 LIMIT 1) AS Q3
FROM laptop;

-- Ram
SELECT
    COUNT(Ram) AS count_Ram,
    MAX(Ram) AS max_Ram,
    MIN(Ram) AS min_Ram,
    AVG(Ram) AS avg_Ram,
    STDDEV(Ram) AS stddev_Ram,
    (SELECT Ram FROM (SELECT Ram, NTILE(4) OVER (ORDER BY Ram) AS quartile FROM
laptop) AS subquery WHERE quartile = 1 LIMIT 1) AS Q1,
    (SELECT Ram FROM (SELECT Ram, NTILE(2) OVER (ORDER BY Ram) AS median FROM
laptop) AS subquery WHERE median = 1 LIMIT 1) AS Q2,
    (SELECT Ram FROM (SELECT Ram, NTILE(4) OVER (ORDER BY Ram) AS quartile FROM
laptop) AS subquery WHERE quartile = 3 LIMIT 1) AS Q3
FROM laptop;

-- primary_storage
SELECT
    COUNT(primary_storage) AS count_primary_storage,
    MAX(primary_storage) AS max_primary_storage,
    MIN(primary_storage) AS min_primary_storage,
    AVG(primary_storage) AS avg_primary_storage,
    STDDEV(primary_storage) AS stddev_Ram,
    (SELECT primary_storage FROM (SELECT primary_storage, NTILE(4) OVER (ORDER BY
primary_storage) AS quartile FROM laptop) AS subquery WHERE quartile = 1 LIMIT 1) AS Q1,
    (SELECT primary_storage FROM (SELECT primary_storage, NTILE(2) OVER (ORDER BY
primary_storage) AS median FROM laptop) AS subquery WHERE median = 1 LIMIT 1) AS Q2,
    (SELECT primary_storage FROM (SELECT primary_storage, NTILE(4) OVER (ORDER BY
primary_storage) AS quartile FROM laptop) AS subquery WHERE quartile = 3 LIMIT 1) AS Q3
FROM laptop;

-- secondary_storage
SELECT
    COUNT(primary_storage) AS count_secondary_storage,
    MAX(secondary_storage) AS max_secondary_storage,
    MIN(secondary_storage) AS min_secondary_storage,
    AVG(secondary_storage) AS avg_secondary_storage,
    STDDEV(secondary_storage) AS stddev_secondary_storage,
    (SELECT secondary_storage FROM (SELECT secondary_storage, NTILE(4) OVER (ORDER
BY secondary_storage) AS quartile FROM laptop) AS subquery WHERE quartile = 1 LIMIT 1)
AS Q1,
    (SELECT secondary_storage FROM (SELECT secondary_storage, NTILE(2) OVER (ORDER
BY secondary_storage) AS median FROM laptop) AS subquery WHERE median = 1 LIMIT 1)
AS Q2,
    (SELECT secondary_storage FROM (SELECT secondary_storage, NTILE(4) OVER (ORDER
BY secondary_storage) AS quartile FROM laptop) AS subquery WHERE quartile = 3 LIMIT 1)
AS Q3
FROM laptop;

-- Weight
SELECT
    COUNT(Weight) AS count_Weight,
    MAX(Weight) AS max_Weight,
    MIN(Weight) AS min_Weight,
    AVG(Weight) AS avg_Weight,
    STDDEV(Weight) AS stddev_Weight,
    (SELECT Weight FROM (SELECT Weight, NTILE(4) OVER (ORDER BY Weight) AS quartile
FROM laptop) AS subquery WHERE quartile = 1 LIMIT 1) AS Q1,
    (SELECT Weight FROM (SELECT Weight, NTILE(2) OVER (ORDER BY Weight) AS median
FROM laptop) AS subquery WHERE median = 1 LIMIT 1) AS Q2,
    (SELECT Weight FROM (SELECT Weight, NTILE(4) OVER (ORDER BY Weight) AS quartile
FROM laptop) AS subquery WHERE quartile = 3 LIMIT 1) AS Q3
FROM laptop;

-- Price
SELECT
    COUNT(Price) AS count_Price,
    MAX(Price) AS max_Price,
    MIN(Price) AS min_Price,
    AVG(Price) AS avg_Price,
    STDDEV(Price) AS stddev_Price,
    (SELECT Price FROM (SELECT Price, NTILE(4) OVER (ORDER BY Price) AS quartile
FROM laptop) AS subquery WHERE quartile = 1 LIMIT 1) AS Q1,
    (SELECT Price FROM (SELECT Price, NTILE(2) OVER (ORDER BY Price) AS median
FROM laptop) AS subquery WHERE median = 1 LIMIT 1) AS Q2,
    (SELECT Price FROM (SELECT Price, NTILE(4) OVER (ORDER BY Price) AS quartile
FROM laptop) AS subquery WHERE quartile = 3 LIMIT 1) AS Q3
FROM laptop;

-- Missing Values
SELECT COUNT(Resolution_width)
FROM laptop
```

```sql
WHERE Price IS NULL;

SELECT COUNT(resolution_height)
FROM laptop
WHERE Price IS NULL;

SELECT COUNT(Ram)
FROM laptop
WHERE Price IS NULL;

SELECT COUNT(primary_storage)
FROM laptop
WHERE Price IS NULL;

SELECT COUNT(secondary_storage)
FROM laptop
WHERE Price IS NULL;

SELECT COUNT(Weight)
FROM laptop
WHERE Price IS NULL;

SELECT COUNT(Price)
FROM laptop
WHERE Price IS NULL;

-- Outliers Using Box Plot/IQR method
-- Price
WITH
q1_cte AS (
   SELECT Price FROM (
      SELECT Price, NTILE(4) OVER (ORDER BY Price) AS quartile
      FROM laptop
   ) AS subquery
   WHERE quartile = 1 ORDER BY Price LIMIT 1
),
q2_cte AS (
   SELECT Price FROM (
      SELECT Price, NTILE(2) OVER (ORDER BY Price) AS median
      FROM laptop
   ) AS subquery
   WHERE median = 1 ORDER BY Price LIMIT 1
),
q3_cte AS (
    SELECT Price FROM (
        SELECT Price, NTILE(4) OVER (ORDER BY Price) AS quartile
        FROM laptop
    ) AS subquery
    WHERE quartile = 3 ORDER BY Price LIMIT 1
)
SELECT *
FROM laptop
WHERE Price < (SELECT Price FROM q1_cte) - (1.5 * ((SELECT Price FROM q3_cte) -
(SELECT Price FROM q1_cte)))
   OR Price > (SELECT Price FROM q3_cte) + (1.5 * ((SELECT Price FROM q3_cte) - (SELECT
Price FROM q1_cte)));


-- Ram
WITH
q1_cte AS (
    SELECT Ram FROM (
        SELECT Ram, NTILE(4) OVER (ORDER BY Ram) AS quartile
        FROM laptop
    ) AS subquery
    WHERE quartile = 1 ORDER BY Ram LIMIT 1
),
q2_cte AS (
    SELECT Ram FROM (
        SELECT Ram, NTILE(2) OVER (ORDER BY Ram) AS median
        FROM laptop
    ) AS subquery
    WHERE median = 1 ORDER BY Ram LIMIT 1
),
q3_cte AS (
    SELECT Ram FROM (
        SELECT Ram, NTILE(4) OVER (ORDER BY Ram) AS quartile
        FROM laptop
    ) AS subquery
    WHERE quartile = 3 ORDER BY Ram LIMIT 1
)
SELECT *
FROM laptop
WHERE Ram < (SELECT Ram FROM q1_cte) - (1.5 * ((SELECT Ram FROM q3_cte) -
(SELECT Ram FROM q1_cte)))
   OR Ram > (SELECT Ram FROM q3_cte) + (1.5 * ((SELECT Ram FROM q3_cte) - (SELECT
Ram FROM q1_cte)));
```

```
-- Weight
WITH
q1_cte AS (
    SELECT Weight FROM (
        SELECT Weight, NTILE(4) OVER (ORDER BY Weight) AS quartile
        FROM laptop
    ) AS subquery
    WHERE quartile = 1 ORDER BY Weight LIMIT 1
),
q2_cte AS (
    SELECT Weight FROM (
        SELECT Weight, NTILE(2) OVER (ORDER BY Weight) AS median
        FROM laptop
    ) AS subquery
    WHERE median = 1 ORDER BY Weight LIMIT 1
),
q3_cte AS (
    SELECT Weight FROM (
        SELECT Weight, NTILE(4) OVER (ORDER BY Weight) AS quartile
        FROM laptop
    ) AS subquery
    WHERE quartile = 3 ORDER BY Weight LIMIT 1
)
SELECT *
FROM laptop
WHERE Weight < (SELECT Weight FROM q1_cte) - (1.5 * ((SELECT Weight FROM q3_cte) -
(SELECT Weight FROM q1_cte)))
   OR Weight > (SELECT Weight FROM q3_cte) + (1.5 * ((SELECT Weight FROM q3_cte) -
(SELECT Weight FROM q1_cte)));

-- Histogram
-- DRAWING CONCLUSIONS: which price segment has the most number of laptops and least
number of laptops
SELECT
    buckets, COUNT(*) AS count
FROM (
    SELECT price,
           CASE
               WHEN price BETWEEN 0 AND 25000 THEN '0-25K'
               WHEN price BETWEEN 25001 AND 50000 THEN '25K-50K'
               WHEN price BETWEEN 50001 AND 75000 THEN '50K-75K'
               WHEN price BETWEEN 75001 AND 100000 THEN '75K-100K'
               ELSE '>100K'
           END AS buckets
    FROM laptop
) t
GROUP BY buckets
ORDER BY
    CASE
        WHEN buckets = '0-25K' THEN 1
        WHEN buckets = '25K-50K' THEN 2
        WHEN buckets = '50K-75K' THEN 3
        WHEN buckets = '75K-100K' THEN 4
        ELSE 5
    END;


-- Categorical Columns
-- VALUE COUNTS
-- Number of laptops produced by each company
SELECT Company, COUNT(Company) FROM laptop
GROUP BY Company;

-- Different types of (Ultrabook/Notebook/Gaming/etc.) laptops
SELECT TypeName, COUNT(TypeName) FROM laptop
GROUP BY TypeName;

-- Number of laptops with Types of CPU brand
SELECT cpu_brand, COUNT(cpu_brand) FROM laptop
GROUP BY cpu_brand;

-- Number of laptops with Types of CPU name
SELECT cpu_name, COUNT(cpu_name) FROM laptop
GROUP BY cpu_name;

-- Number of laptops with Types of memeory
SELECT memory_type, COUNT(memory_type) FROM laptop
GROUP BY memory_type;

-- Number of laptops with GPU brand
SELECT gpu_brand, COUNT(gpu_brand) FROM laptop
GROUP BY gpu_brand;

-- Number of laptops with GPU name
SELECT gpu_name, COUNT(gpu_name) FROM laptop
GROUP BY gpu_name;

-- Number of laptops with Operating System
```

```sql
SELECT OpSys, COUNT(OpSys) FROM laptop
GROUP BY OpSys;

-- Missing Values
SELECT COUNT(Company)
FROM laptop
WHERE Company IS NULL;

SELECT COUNT(TypeName)
FROM laptop
WHERE TypeName IS NULL;

SELECT COUNT(cpu_brand)
FROM laptop
WHERE cpu_brand IS NULL;

SELECT COUNT(cpu_name)
FROM laptop
WHERE cpu_name IS NULL;

SELECT COUNT(gpu_brand)
FROM laptop
WHERE gpu_brand IS NULL;

SELECT COUNT(gpu_name)
FROM laptop
WHERE gpu_name IS NULL;

SELECT COUNT(OpSys)
FROM laptop
WHERE OpSys IS NULL;

-- Numerical - Numerical


SELECT cpu_speed,SUM(Price) AS total_sum FROM laptop
GROUP BY cpu_speed
ORDER BY cpu_speed;

-- Examine the relationship between Ram and Price.
SELECT Ram,SUM(Price) AS total_sum FROM laptop
GROUP BY Ram
ORDER BY total_sum;
```

```sql
SELECT Inches, resolution_width, resolution_height
FROM laptop
ORDER BY Inches;

SELECT Inches, Weight
FROM laptop
ORDER BY Weight;

-- Determining Average price of laptops according to its Primary Storage
SELECT primary_storage, ROUND(AVG(Price),2) AS total FROM laptop
GROUP BY primary_storage
ORDER BY primary_storage;

-- Compairing Average Price of laptops to Weight
SELECT Weight, ROUND(AVG(Price),2) AS total FROM laptop
GROUP BY Weight
ORDER BY Weight;

-- Catergorical - Numerical

-- Average price of by each cpu brand company
SELECT cpu_brand, ROUND(AVG(Price),2) AS total_sum FROM laptop
GROUP BY cpu_brand
ORDER BY total_sum;

-- Determine the average price of laptops produced by different companies.
SELECT Company, ROUND(AVG(Price),2) AS avg_price
FROM laptop
GROUP BY Company
ORDER BY avg_price DESC;

-- Compare the average prices of different types of laptops.
SELECT TypeName, AVG(Price) AS avg_price
FROM laptop
GROUP BY TypeName
ORDER BY avg_price DESC;

-- Analyze the price differences between laptops with different CPU brands.
SELECT cpu_brand, ROUND(AVG(Price),2) AS avg_price
FROM laptop
GROUP BY cpu_brand
ORDER BY avg_price DESC;
```

```
-- See how different types of memory affect the laptop price.
SELECT memory_type, ROUND(AVG(Price),2) AS avg_price
FROM laptop
GROUP BY memory_type
ORDER BY avg_price DESC;

-- Examine the price variations based on the operating system.
SELECT OpSys, ROUND(AVG(Price),2) AS avg_price
FROM laptop
GROUP BY OpSys
ORDER BY avg_price DESC;

-- Determine how the operating system relates to the weight of the laptop.
SELECT OpSys, ROUND(AVG(Weight),2) AS avg_weight
FROM laptop
GROUP BY OpSys
ORDER BY avg_weight;

-- Categorical - Categorical

-- Determine the variety of laptop types offered by different companies.
SELECT Company,
SUM(CASE WHEN TypeName = 'Ultrabook' THEN 1 ELSE 0 END) AS 'Ultrabook',
SUM(CASE WHEN TypeName = 'Notebook' THEN 1 ELSE 0 END) AS 'Notebook',
SUM(CASE WHEN TypeName = 'Gaming' THEN 1 ELSE 0 END) AS 'Gaming',
SUM(CASE WHEN TypeName = '2 in 1 Convertible' THEN 1 ELSE 0 END) AS '2 in 1
Convertible',
SUM(CASE WHEN TypeName = 'Workstation' THEN 1 ELSE 0 END) AS 'Workstation',
SUM(CASE WHEN TypeName = 'Netbook' THEN 1 ELSE 0 END) AS 'Netbook'
FROM laptop
GROUP BY Company;

-- Identify the distribution of CPU brands used by different companies.
SELECT Company,
SUM(CASE WHEN cpu_brand = 'Intel' THEN 1 ELSE 0 END) AS 'intel',
SUM(CASE WHEN cpu_brand = 'AMD' THEN 1 ELSE 0 END) AS 'amd',
SUM(CASE WHEN cpu_brand = 'Samsung' THEN 1 ELSE 0 END) AS 'samsung'
FROM laptop
GROUP BY Company;

-- See the prevalence of different memory types (e.g., SSD, HDD) across companies.
SELECT Company,
SUM(CASE WHEN memory_type = 'SSD' THEN 1 ELSE 0 END) AS 'SSD',
SUM(CASE WHEN memory_type = 'Flash Storage' THEN 1 ELSE 0 END) AS 'Flash Storage',
SUM(CASE WHEN memory_type = 'HDD' THEN 1 ELSE 0 END) AS 'HDD',
SUM(CASE WHEN memory_type = 'Hybrid' THEN 1 ELSE 0 END) AS 'Hybrid',
SUM(CASE WHEN memory_type IS NULL THEN 1 ELSE 0 END) AS 'NULL'
FROM laptop
GROUP BY Company;

-- Understand which GPU brands are more commonly used by each company.
SELECT Company,
SUM(CASE WHEN gpu_brand = 'Intel' THEN 1 ELSE 0 END) AS 'Intel',
SUM(CASE WHEN gpu_brand = 'AMD' THEN 1 ELSE 0 END) AS 'AMD',
SUM(CASE WHEN gpu_brand = 'Nvidia' THEN 1 ELSE 0 END) AS 'Nvidia',
SUM(CASE WHEN gpu_brand = 'ARM' THEN 1 ELSE 0 END) AS 'ARM'
FROM laptop
GROUP BY Company;

-- Analyze the operating systems used in different types of laptops.
SELECT TypeName,
SUM(CASE WHEN OpSys = 'MacOS' THEN 1 ELSE 0 END) AS 'MacOS',
SUM(CASE WHEN OpSys = 'N/A' THEN 1 ELSE 0 END) AS 'N/A',
SUM(CASE WHEN OpSys = 'Windows' THEN 1 ELSE 0 END) AS 'Windows',
SUM(CASE WHEN OpSys = 'Linux' THEN 1 ELSE 0 END) AS 'Linux',
SUM(CASE WHEN OpSys = 'Other' THEN 1 ELSE 0 END) AS 'Other'
FROM laptop
GROUP BY TypeName;

-- Commonly used CPU brands used in different types of laptop
SELECT TypeName,
SUM(CASE WHEN cpu_brand = 'Intel' THEN 1 ELSE 0 END) AS 'Intel',
SUM(CASE WHEN cpu_brand = 'AMD' THEN 1 ELSE 0 END) AS 'AMD',
SUM(CASE WHEN cpu_brand = 'Samsung' THEN 1 ELSE 0 END) AS 'Samsung'
FROM laptop
GROUP BY TypeName;

-- Analyze the different memory types used in different types of laptop
SELECT TypeName,
SUM(CASE WHEN memory_type = 'SSD' THEN 1 ELSE 0 END) AS 'SSD',
SUM(CASE WHEN memory_type = 'Flash Storage' THEN 1 ELSE 0 END) AS 'Flash Storage',
SUM(CASE WHEN memory_type = 'HDD' THEN 1 ELSE 0 END) AS 'HDD',
SUM(CASE WHEN memory_type = 'Hybrid' THEN 1 ELSE 0 END) AS 'Hybrid',
SUM(CASE WHEN memory_type IS NULL THEN 1 ELSE 0 END) AS 'NULL'
FROM laptop
GROUP BY TypeName;
```