USE sql\_cx\_live;

SELECT \* FROM laptops;

-- head, tail and sample

SELECT \* FROM laptops

ORDER BY `index` LIMIT 5;

SELECT \* FROM laptops

ORDER BY `index` DESC LIMIT 5;

SELECT \* FROM laptops

ORDER BY rand() LIMIT 5;

SELECT COUNT(Price) OVER(),

MIN(Price) OVER(),

MAX(Price) OVER(),

AVG(Price) OVER(),

STD(Price) OVER(),

PERCENTILE\_CONT(0.25) WITHIN GROUP(ORDER BY Price) OVER() AS 'Q1',

PERCENTILE\_CONT(0.5) WITHIN GROUP(ORDER BY Price) OVER() AS 'Median',

PERCENTILE\_CONT(0.75) WITHIN GROUP(ORDER BY Price) OVER() AS 'Q3'

FROM laptops

ORDER BY `index` LIMIT 1;

-- missing value

SELECT COUNT(Price)

FROM laptops

WHERE Price IS NULL;

-- outliers

SELECT \* FROM (SELECT \*,

PERCENTILE\_CONT(0.25) WITHIN GROUP(ORDER BY Price) OVER() AS 'Q1',

PERCENTILE\_CONT(0.75) WITHIN GROUP(ORDER BY Price) OVER() AS 'Q3'

FROM laptops) t

WHERE t.Price < t.Q1 - (1.5\*(t.Q3 - t.Q1)) OR

t.Price > t.Q3 + (1.5\*(t.Q3 - t.Q1));

SELECT t.buckets,REPEAT('\*',COUNT(\*)/5) 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 laptops) t

GROUP BY t.buckets;

SELECT Company,COUNT(Company) FROM laptops

GROUP BY Company;

SELECT cpu\_speed,Price FROM laptops;

SELECT \* FROM laptops;

SELECT Company,

SUM(CASE WHEN Touchscreen = 1 THEN 1 ELSE 0 END) AS 'Touchscreen\_yes',

SUM(CASE WHEN Touchscreen = 0 THEN 1 ELSE 0 END) AS 'Touchscreen\_no'

FROM laptops

GROUP BY Company;

SELECT DISTINCT cpu\_brand FROM laptops;

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 laptops

GROUP BY Company;

-- Categorical Numerical Bivariate analysis

SELECT Company,MIN(price),

MAX(price),AVG(price),STD(price)

FROM laptops

GROUP BY Company;

-- Dealing with missing values

SELECT \* FROM laptops

WHERE price IS NULL;

-- UPDATE laptops

-- SET price = NULL

-- WHERE `index` IN (7,869,1148,827,865,821,1056,1043,692,1114)

-- replace missing values with mean of price

UPDATE laptops

SET price = (SELECT AVG(price) FROM laptops)

WHERE price IS NULL;

-- replace missing values with mean price of corresponding company

UPDATE laptops l1

SET price = (SELECT AVG(price) FROM laptops l2 WHERE

l2.Company = l1.Company)

WHERE price IS NULL;

SELECT \* FROM laptops

WHERE price IS NULL;

-- corresponsing company + processor

SELECT \* FROM laptops;

-- Feature Engineering

ALTER TABLE laptops ADD COLUMN ppi INTEGER;

UPDATE laptops

SET ppi = ROUND(SQRT(resolution\_width\*resolution\_width + resolution\_height\*resolution\_height)/Inches);

SELECT \* FROM laptops

ORDER BY ppi DESC;

ALTER TABLE laptops ADD COLUMN screen\_size VARCHAR(255) AFTER Inches;

UPDATE laptops

SET screen\_size =

CASE

WHEN Inches < 14.0 THEN 'small'

WHEN Inches >= 14.0 AND Inches < 17.0 THEN 'medium'

ELSE 'large'

END;

SELECT screen\_size,AVG(price) FROM laptops

GROUP BY screen\_size;

-- One Hot Encoding

SELECT gpu\_brand,

CASE WHEN gpu\_brand = 'Intel' THEN 1 ELSE 0 END AS 'intel',

CASE WHEN gpu\_brand = 'AMD' THEN 1 ELSE 0 END AS 'amd',

CASE WHEN gpu\_brand = 'nvidia' THEN 1 ELSE 0 END AS 'nvidia',

CASE WHEN gpu\_brand = 'arm' THEN 1 ELSE 0 END AS 'arm'

FROM laptops