```
USE cleaning_eda;

SELECT * FROM laptopdata;

-- Creating a copy of the original dataset
CREATE TABLE laptop LIKE laptopdata;

-- Inserting elements in the new table
INSERT INTO laptop
SELECT * FROM laptopdata;

-- Checking if every element inserted succesfully
SELECT * FROM laptop;

-- Checking the memory consumed by the data in KB
SELECT DATA_LENGTH/1024 FROM information_schema.TABLES
WHERE TABLE_SCHEMA = 'cleaning_eda'
AND TABLE_NAME = 'laptop';

-- Removing unnecessary columns
ALTER TABLE laptop DROP COLUMN `Unnamed: 0`;

-- Removing null values that exists in every column in that particular row
DELETE FROM laptop
WHERE `index` IN
                (SELECT `index` FROM laptop
                WHERE Company IS NULL AND TypeName IS NULL
AND Inches IS NULL AND
                ScreenResolution IS NULL AND `Cpu` IS NULL AND
Ram IS NULL AND
                `Memory` IS NULL AND Gpu IS NULL AND OpSys IS
NULL AND Weight IS NULL AND Price IS NULL);

SELECT * FROM laptop;
```

```
-- Changing the dtype of Inches so that it takes lesser space in the memory
ALTER TABLE laptop MODIFY COLUMN Inches DECIMAL(10,1);

-- Changing the dtype of price so that it takes lesser space in the memory
UPDATE laptop t1
JOIN (SELECT `Index`, ROUND(price) AS rounded_price FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.Price = t2.rounded_price;

ALTER TABLE laptop MODIFY COLUMN Price INTEGER;

-- Changing the dtype of Ram
UPDATE laptop t1
JOIN (SELECT `Index`, REPLACE(Ram,'GB','') AS Update_Ram FROM
laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.Ram = t2.Update_Ram;

ALTER TABLE laptop MODIFY COLUMN Ram INTEGER;

-- Changing the dtype of Weight
UPDATE laptop t1
JOIN (SELECT `Index`, REPLACE(Weight,'kg','') AS Update_Weight
FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.Weight = t2.Update_Weight;

ALTER TABLE laptop MODIFY COLUMN Weight DECIMAL(5,2);

SELECT * FROM laptop;

-- Checking the memory consumed by the data in FROM 272 KB - 256 KB
SELECT DATA_LENGTH/1024 FROM information_schema.TABLES
WHERE TABLE_SCHEMA = 'cleaning_eda'
AND TABLE_NAME = 'laptop';
```

```
SELECT * FROM laptop;

-- Cleaning the GPU column
-- Making 2 different columns GPU Brand, GPU Name
ALTER TABLE laptop
ADD COLUMN gpu_brand VARCHAR(255) AFTER Gpu,
ADD COLUMN gpu_name VARCHAR(255) AFTER gpu_brand;

-- Inserting Values in the GPU_BRAND column
UPDATE laptop t1
JOIN (SELECT `Index`, SUBSTRING_INDEX(Gpu,' ',1) AS
updated_gpu_brand FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.gpu_brand = t2.updated_gpu_brand;

-- Inserting Values in the GPU_NAME column
UPDATE laptop t1
JOIN (SELECT `Index`, REPLACE(Gpu,gpu_brand,'') AS
updated_gpu_name FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.gpu_name = t2.updated_gpu_name;

SELECT * FROM laptop;

-- Dropping uneccessary Columns
ALTER TABLE laptop DROP COLUMN Gpu;

-- Now cleaning the column CPU
-- Making 3 different columns CPU Brand, CPU Name, CPU Speed
ALTER TABLE laptop
ADD COLUMN cpu_brand VARCHAR(255) AFTER Cpu,
ADD COLUMN cpu_name VARCHAR(255) AFTER cpu_brand,
ADD COLUMN cpu_speed DECIMAL(10,1) AFTER cpu_name;
```

```
-- Extracting the CPU brand from column CPU
UPDATE laptop t1
JOIN (SELECT `Index`, SUBSTRING_INDEX(Cpu,' ',1) AS
update_cpu_brand FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.cpu_brand = t2.update_cpu_brand;


-- Fetching the elements from the 1st occurrence from behind.
-- Replacing the GHz with nothing
-- Casting the result or changing the data type to decimal
UPDATE laptop t1
JOIN (SELECT `Index`,CAST(REPLACE(SUBSTRING_INDEX(Cpu,'
',-1),'GHz','') AS DECIMAL(10,2)) AS updated_value FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.cpu_speed = t2.updated_value;

SELECT * FROM laptop;

-- Updating the column CPU Speed
UPDATE laptop t1
JOIN (SELECT
`Index`,REPLACE(REPLACE(Cpu,cpu_brand,''),SUBSTRING_INDEX(REP
LACE(Cpu,cpu_brand,''),' ',-1),'') AS updated_value FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.cpu_name = t2.updated_value;

-- Dropping the column CPU
ALTER TABLE laptop DROP COLUMN Cpu;

-- Removing unecessary details from CPU Name
UPDATE laptop t1
JOIN (SELECT `Index`,SUBSTRING_INDEX(TRIM(cpu_name),' ',2) AS
updated_value FROM laptop) t2
```

```
ON t1.`Index` = t2.`Index`
SET t1.cpu_name = t2.updated_value;

SELECT * FROM laptop;

-- Cleaning Screen Resolution
ALTER TABLE laptop
ADD COLUMN resolution_width INTEGER AFTER ScreenResolution,
ADD COLUMN resolution_height INTEGER AFTER resolution_width;

-- Inserting values in the column resolution_height
UPDATE laptop t1
JOIN (SELECT `Index`,
SUBSTRING_INDEX(SUBSTRING_INDEX(ScreenResolution,' ',-1),'x',1)
AS updated_value FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.resolution_height = t2.updated_value;

-- Inserting values in the column resolution_width
UPDATE laptop t1
JOIN (SELECT `Index`,
SUBSTRING_INDEX(SUBSTRING_INDEX(ScreenResolution,' ',-1),'x',-1)
AS updated_value FROM laptop) t2
ON t1.`Index` = t2.`Index`
SET t1.resolution_width = t2.updated_value;

SELECT * FROM laptop;

-- Adding another column to see if it's touchscreen or not
ALTER TABLE laptop ADD COLUMN touchscreen INTEGER AFTER
resolution_height;

-- Wherever we are seeing touchscreen we are marking it as true
UPDATE laptop SET touchscreen = ScreenResolution LIKE '%Touch%';
```

```
-- Dropping column ScreenResolution
ALTER TABLE laptop DROP COLUMN ScreenResolution;

SELECT * FROM laptop;

-- Cleaning Operating System
SELECT DISTINCT OpSys FROM laptop;
-- Primarily there are mainly 5 types of OS
-- mac
-- windows
-- Linux
-- no os
-- Android chrome(others)

UPDATE laptop t1
SET OpSys = CASE
                WHEN OpSys LIKE '%mac%' THEN 'MacOS'
                WHEN OpSys LIKE 'windows%' THEN 'Windows'
                WHEN OpSys LIKE '%linux%' THEN 'Linux'
                WHEN OpSys = 'No OS' THEN 'N/A'
                ELSE 'Other'
        END;

SELECT * FROM laptop;

-- Cleaning the last column that is Memory
SELECT Memory FROM laptop;

ALTER TABLE laptop
ADD COLUMN memory_type VARCHAR(255) AFTER Memory,
ADD COLUMN primary_storage INTEGER AFTER memory_type,
ADD COLUMN secondary_storage INTEGER AFTER primary_storage;

UPDATE laptop
SET memory_type = CASE
```

```
        WHEN Memory LIKE '%SSD%' AND `Memory` LIKE '%HDD%'
THEN 'Hybrid'
   WHEN Memory LIKE '%SSD%' THEN 'SSD'
   WHEN Memory LIKE '%HDD%' THEN 'HDD'
   WHEN Memory LIKE '%Flash Storage%' THEN 'Flash Storage'
   WHEN Memory LIKE '%Hybrid%' THEN 'Hybrid'
   WHEN Memory LIKE '%Flash Storage%' AND `Memory` LIKE '%HDD%'
THEN 'Hybrid'
   ELSE NULL
END;

SELECT Memory,
SUBSTRING_INDEX(Memory,'+',1),
REGEXP_SUBSTR(SUBSTRING_INDEX(Memory,'+',1),'[0-9]+'),
SUBSTRING_INDEX(Memory,'+',-1),
CASE WHEN Memory LIKE '%+%' THEN
REGEXP_SUBSTR(SUBSTRING_INDEX(Memory,'+',-1),'[0-9]+') ELSE 0
END
FROM laptop;

 -- Fetching the primary storage and also value beyond the + sign is
secondary storage
UPDATE laptop
SET primary_storage =
REGEXP_SUBSTR(SUBSTRING_INDEX(Memory,'+',1),'[0-9]+'),
secondary_storage = CASE WHEN Memory LIKE '%+%' THEN
REGEXP_SUBSTR(SUBSTRING_INDEX(Memory,'+',-1),'[0-9]+') ELSE 0
END;

SELECT Memory FROM laptop;
SELECT * FROM laptop;

-- As we've removed the Units. We also need to change the TB to GB and
1 TB > 256 GB but due to wrong representation we'll face issue
UPDATE laptops
```

```
SET primary_storage =
CASE WHEN primary_storage <= 2 THEN primary_storage*1024 ELSE
primary_storage END,
secondary_storage = CASE WHEN secondary_storage <= 2 THEN
secondary_storage*1024 ELSE secondary_storage END;

ALTER TABLE laptop DROP COLUMN Memory;
SELECT * FROM laptop;
```