EVALUATION-2

QUES-1: Write a query to find the root node.

SELECT

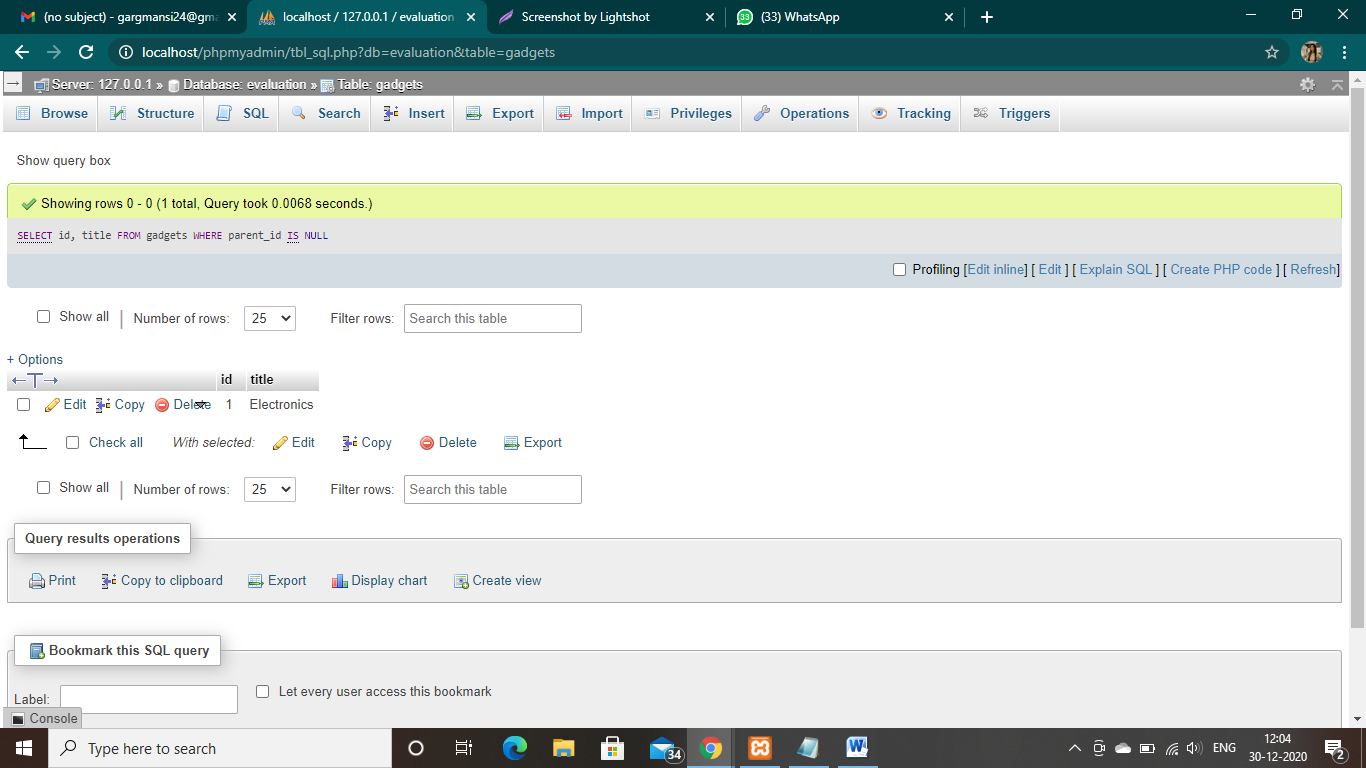
id, title

FROM

gadgets

WHERE

parent\_id IS NULL;



QUES-2: Write a query to find leaf node.

SELECT

c1.id, c1.title

FROM

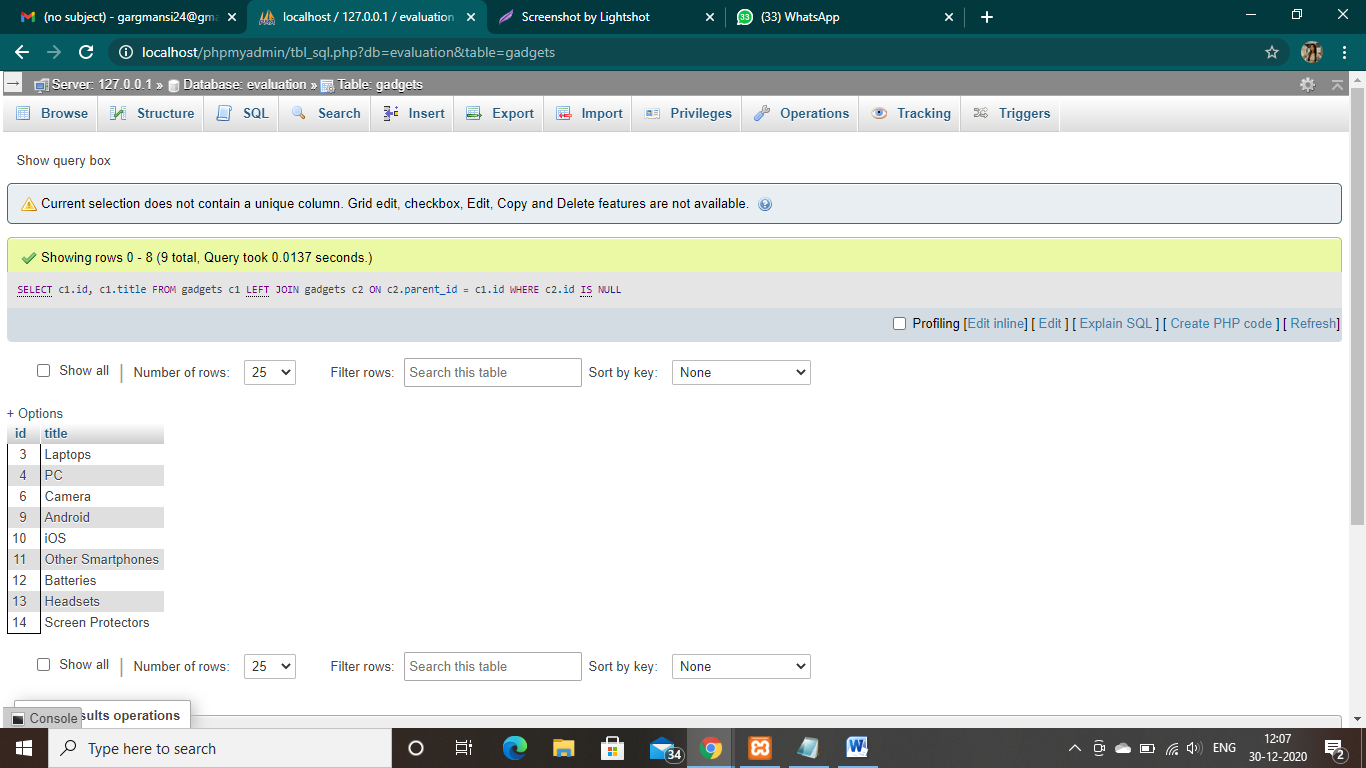
gadgets c1

LEFT JOIN

gadgets c2 ON c2.parent\_id = c1.id

WHERE

c2.id IS NULL;



Ques-3: Write a query to find non-leaf node.

SELECT

DISTINCT( c1.id), c1.title

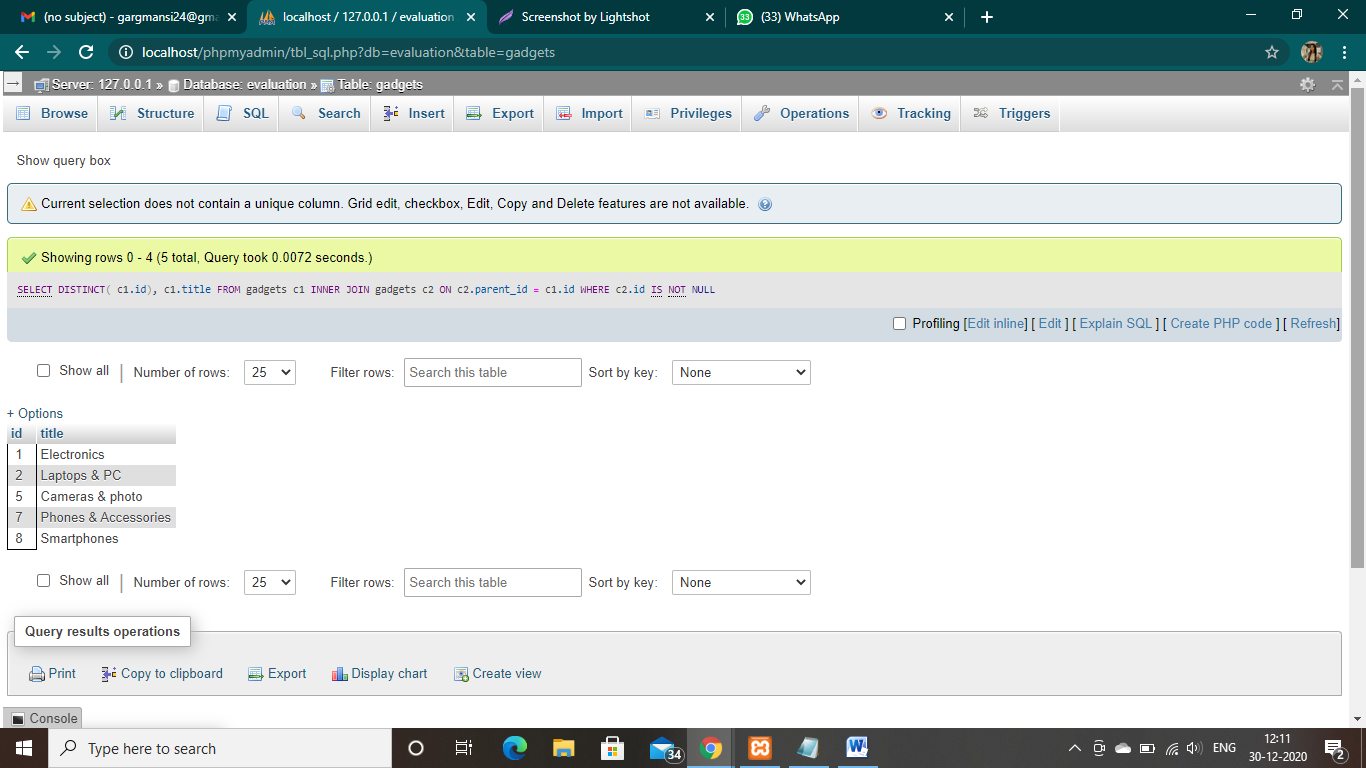
FROM

gadgets c1

INNER JOIN

gadgets c2 ON c2.parent\_id = c1.id

WHERE

c2.id IS NOT NULL

Ques-4: Write a query to find the path of each node.

WITH RECURSIVE gadgets\_path (id, title, path) AS

(

SELECT id, title, title as path

FROM gadgets

WHERE parent\_id IS NULL

UNION ALL

SELECT c.id, c.title, CONCAT(cp.path, ' > ', c.title)

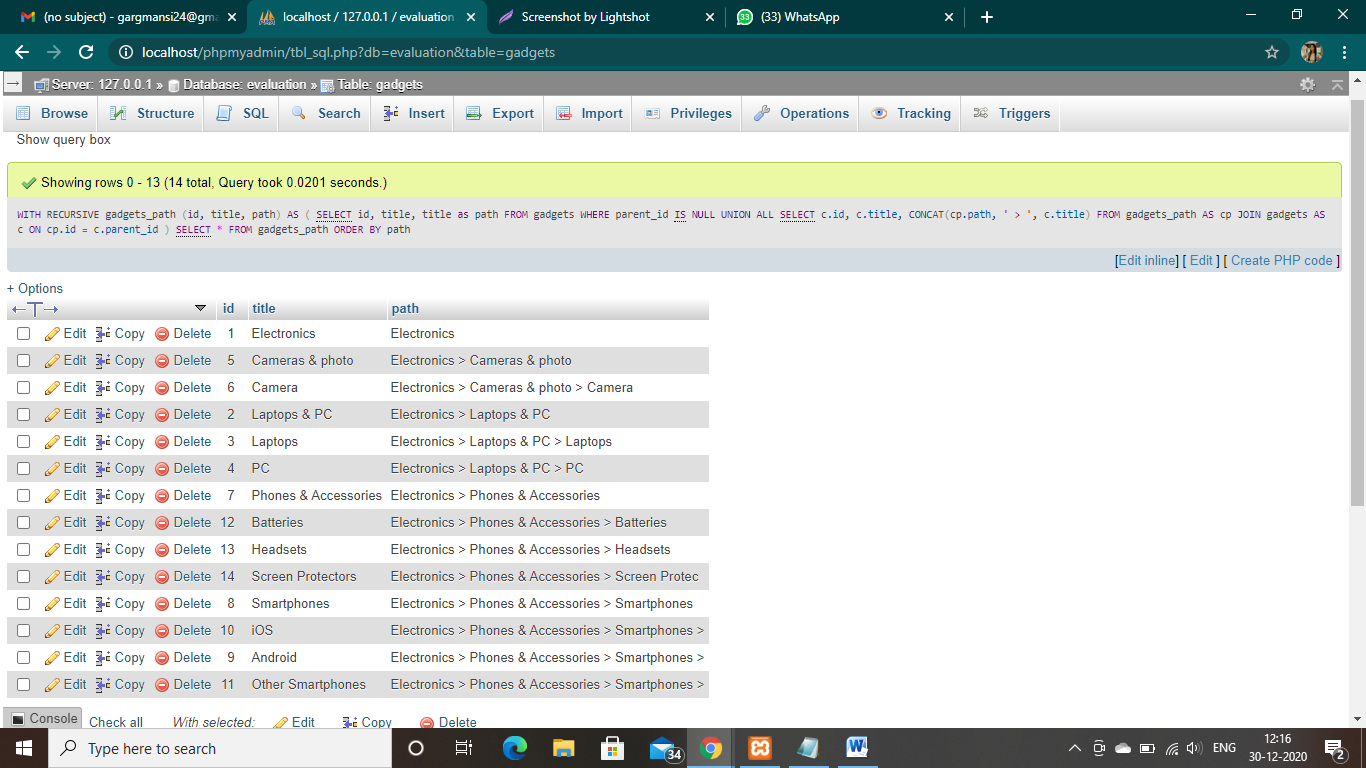
FROM gadgets\_path AS cp JOIN gadgets AS c

ON cp.id = c.parent\_id

)

SELECT \* FROM gadgets\_path

ORDER BY path;



Ques-5: Write a function to calculate node level. e.g. Electronics is at 0 level, Camera is on level 2 and iOs is on level 3.

WITH RECURSIVE gadgets\_path (id, title, lvl) AS

(

SELECT id, title, 0 lvl

FROM gadgets

WHERE parent\_id IS NULL

UNION ALL

SELECT c.id, c.title,cp.lvl + 1

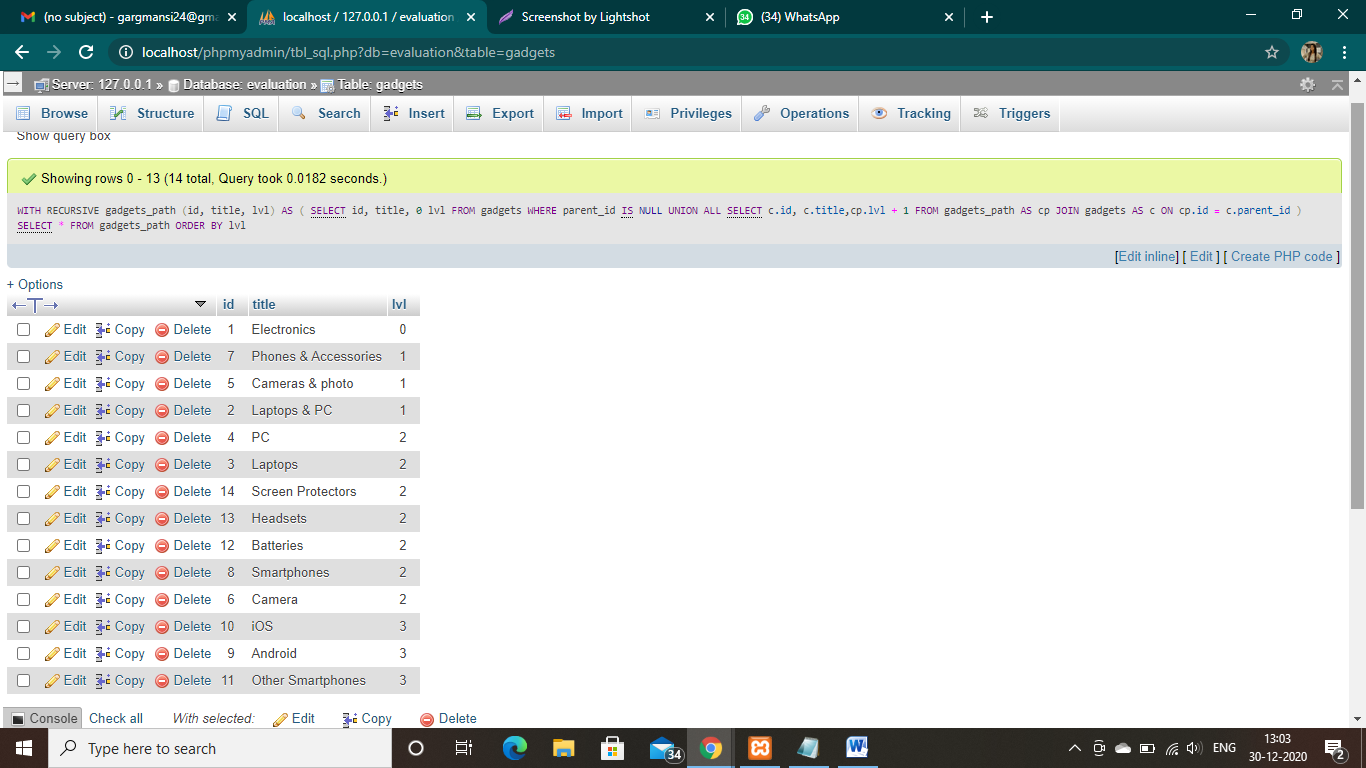
FROM gadgets\_path AS cp JOIN gadgets AS c

ON cp.id = c.parent\_id

)

SELECT \* FROM gadgets\_path

ORDER BY lvl;



Ques-6: Write a procedure to get the immediate children.

SELECT

id, title

FROM

gadgets

WHERE

parent\_id = 1

