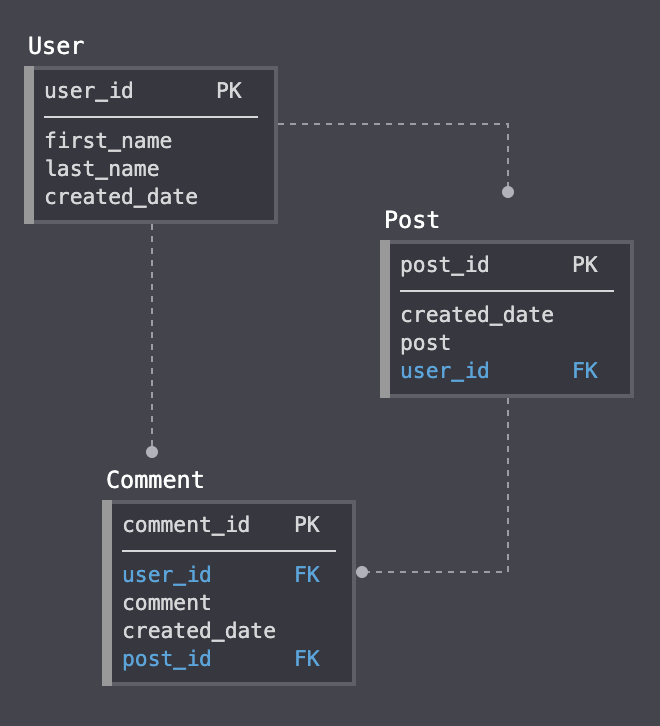
**Firstly:-**

Firstly ,in very simplistic terms a user “comments” on a post as the simplest form of comments are level first comments.

* A ***user*** creates a ***post.***
* A ***user*** ***comments*** on a ***post.***

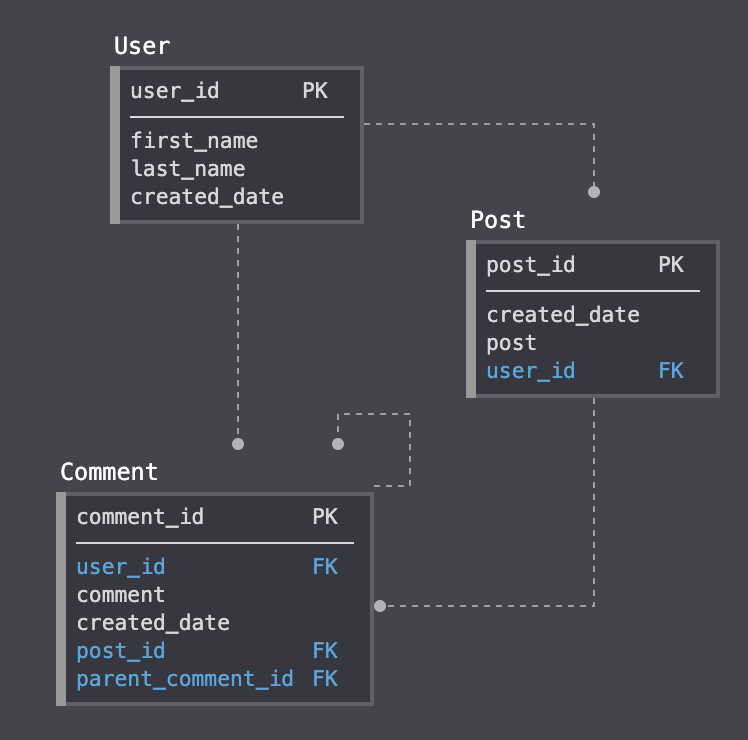


Now the user wants to respond to a comment on his post by giving it as upvote or downvote,.

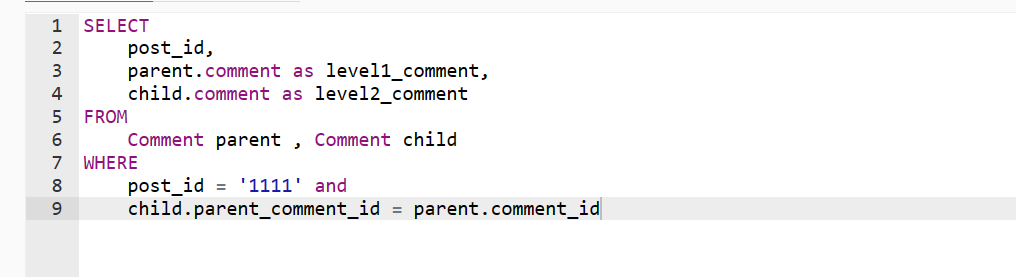
*A***user***creates a***post.**

*A***user****downvote or upvote***on a***post.**

* A ***user*** ***comments***(sub comment)on a ***comment.***

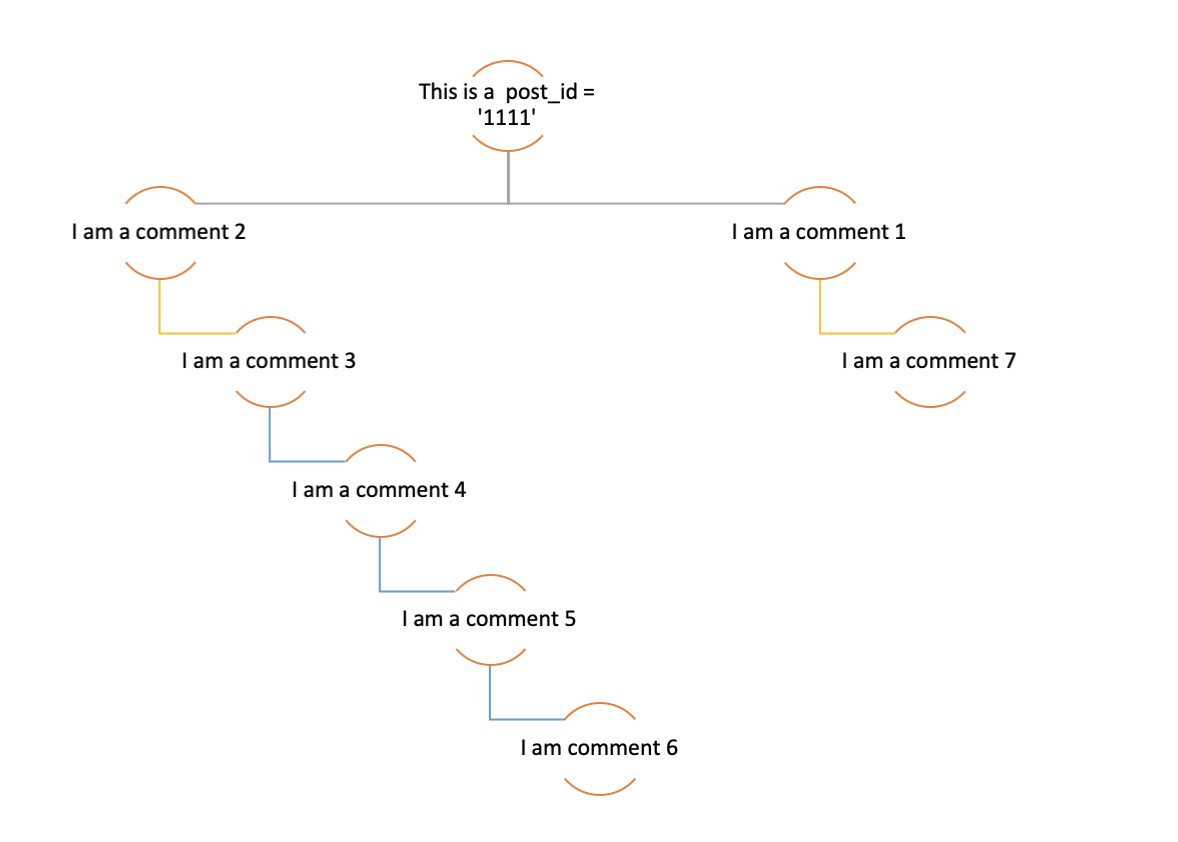


Now let see how we can query the comments on a post “1111” by each level



Lets imagine what if we had “n” levels nested comments like in “reddit” , How would the query look like ? now we will have to self join the data 6 times.

Here it looks like:

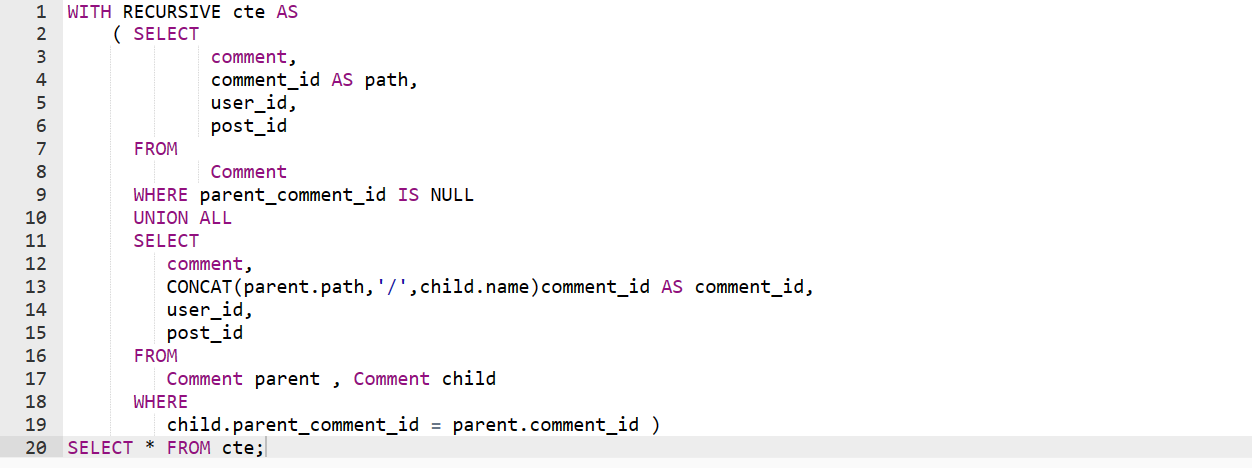


## Approach 1 — Repeated Self-joins :

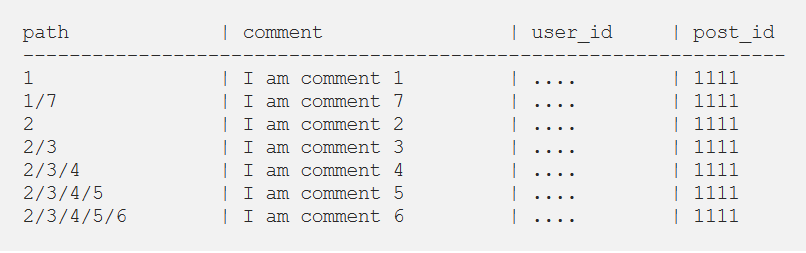
Repeated self joins can get quite complicated, these are also one of the most inefficient queries to execute, slowing down the performance



MySQL support recursive **cte**(Common Table Expression) , the recursive CTE can be used to create a **Path-style Identifiers ,**to query the database. However this just simplifies the query syntax and data representation , it by no means reduces the complexity of the underlying query engine.



The data can now looks like , denormalized and flattened.



You can now query parent comments by comparing the current row’s path to a pattern formed from the path of another row.

