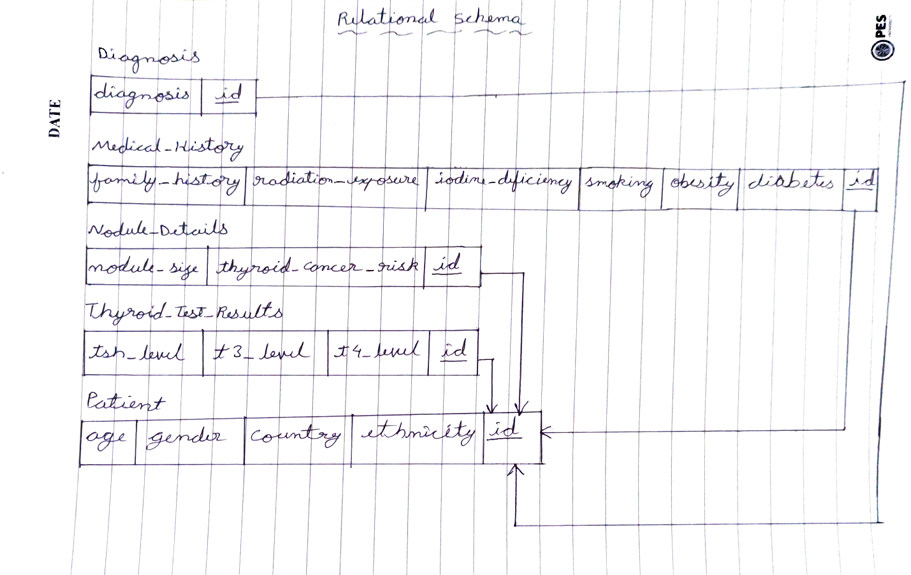
**DBT Assignment**

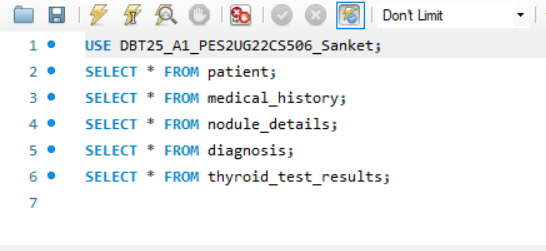
-Sanket Bhandarkar

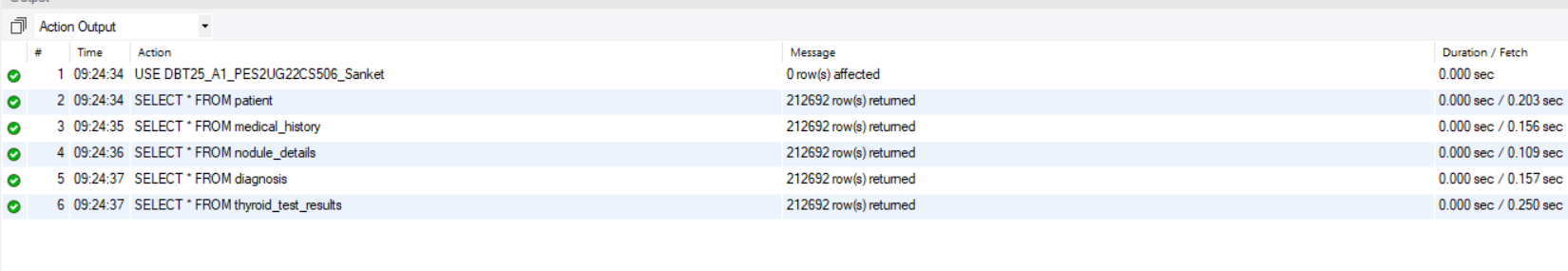
PES2UG22CS506

Relational Schema-

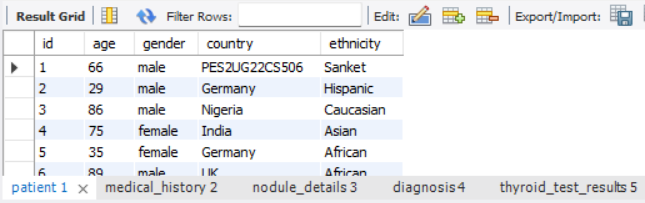


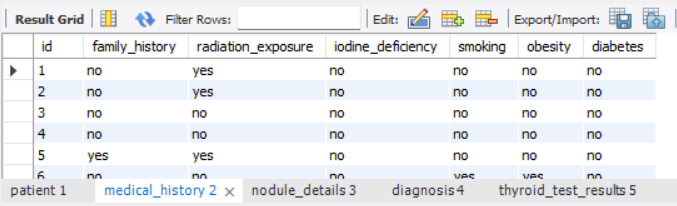
Select Queries-

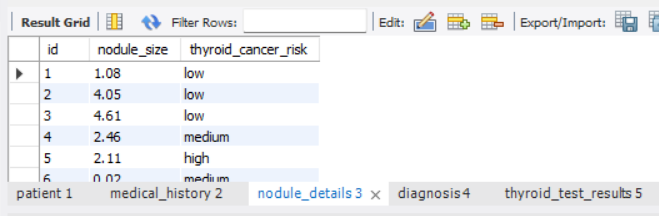


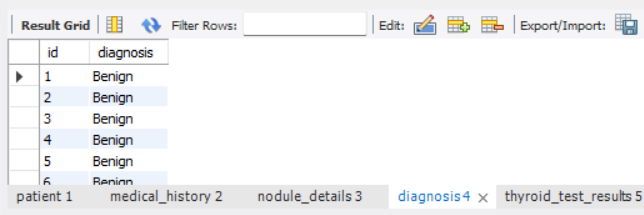


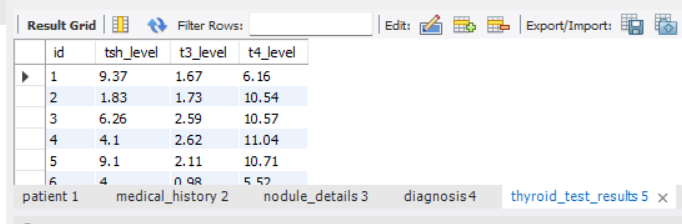
(SRN and name is inserted in the first record here )



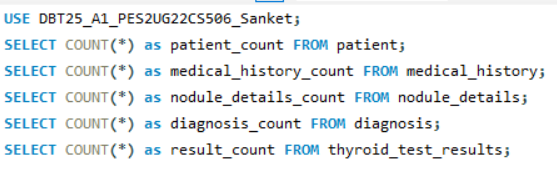


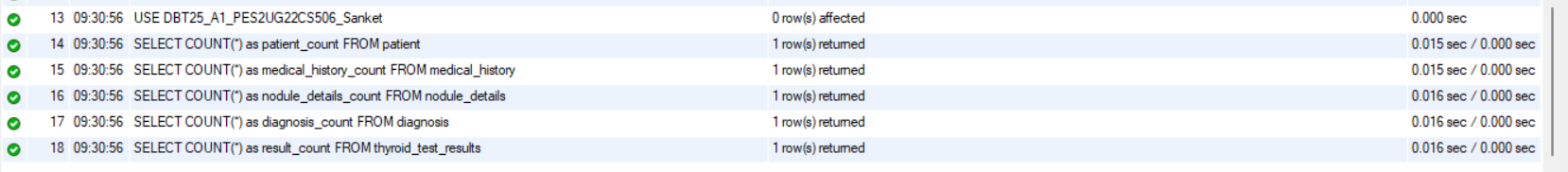






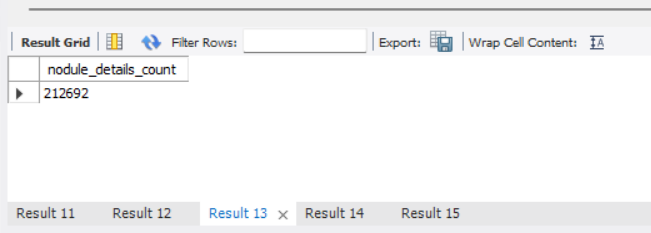
Row Count-

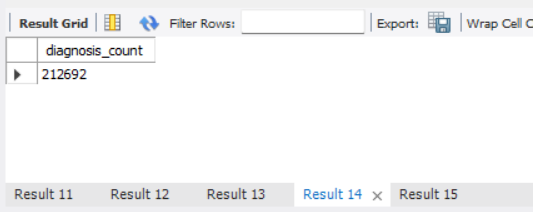


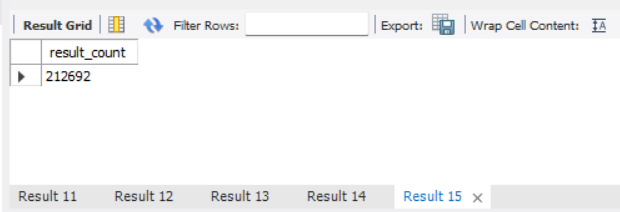




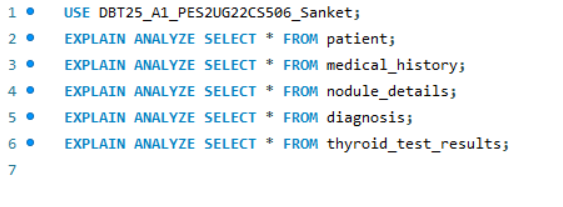


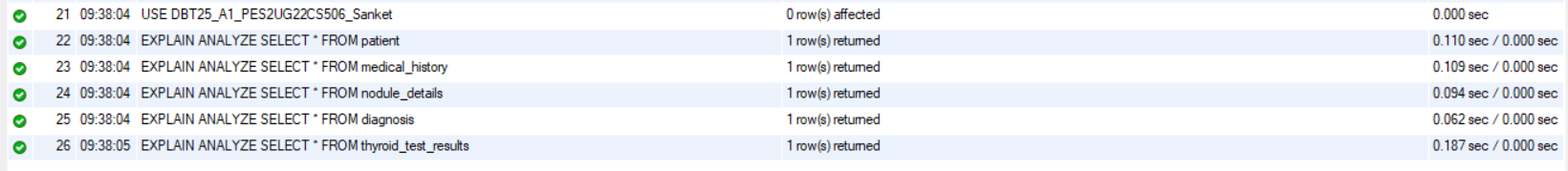


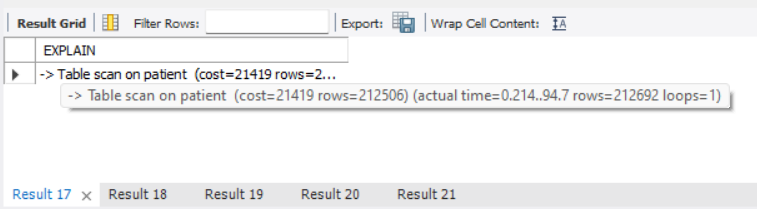


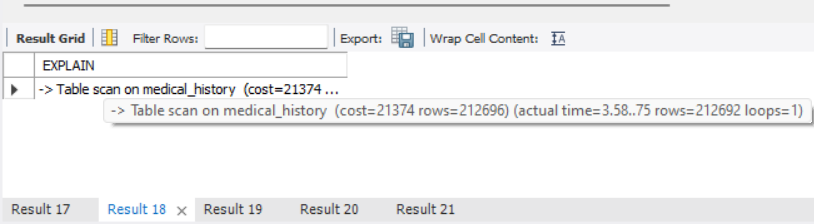


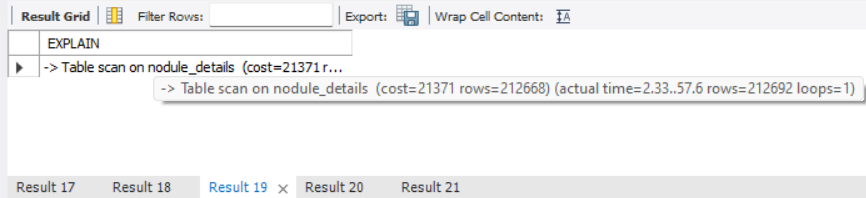
Select Queries Performance Analysis Before Index Creation-

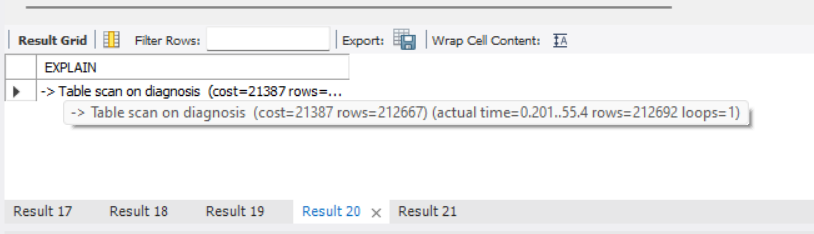


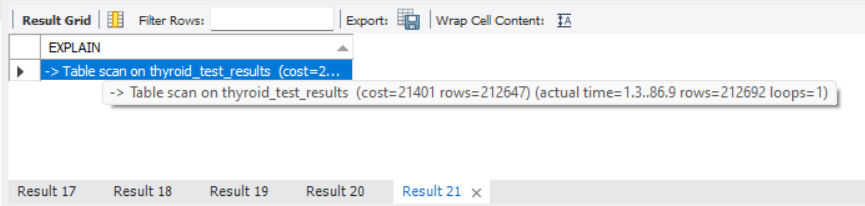




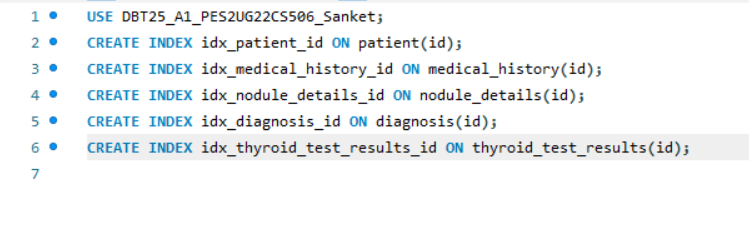


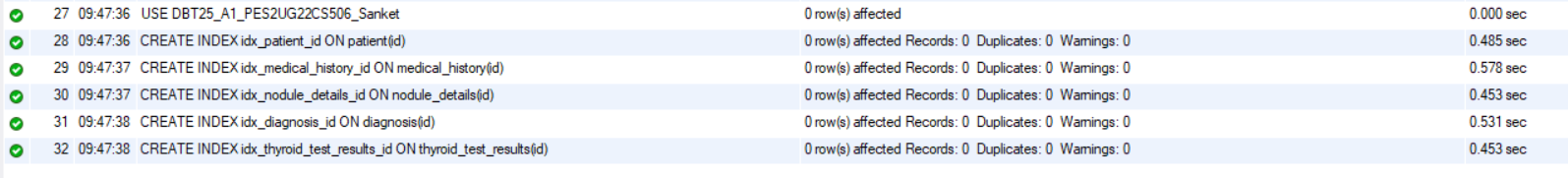


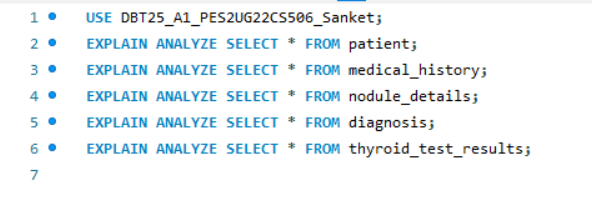


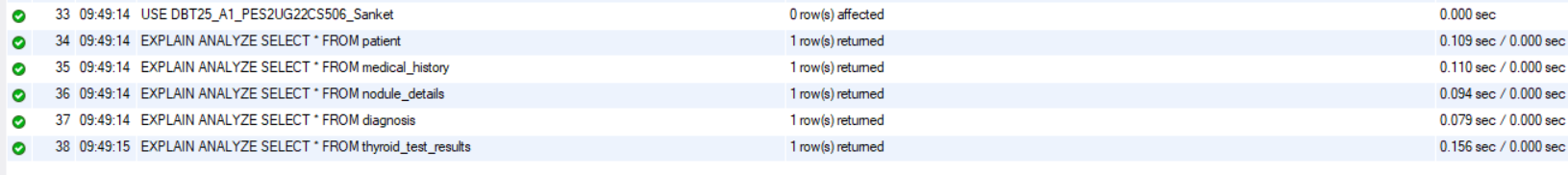


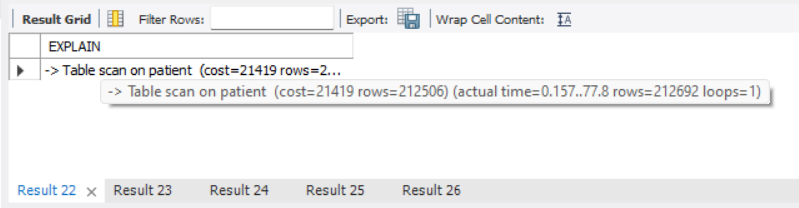
Select Queries Performance Analysis After Index Creation-

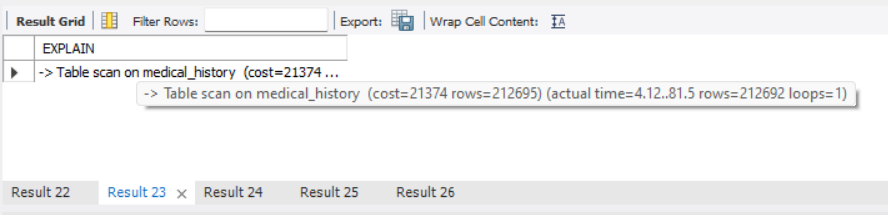


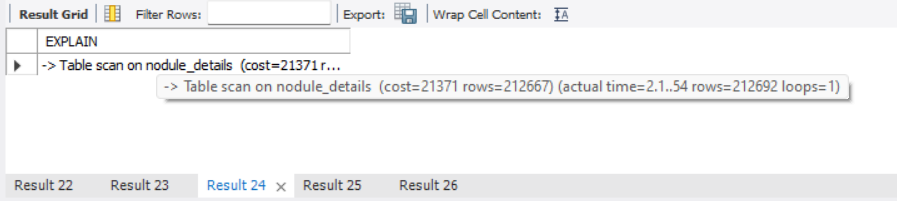


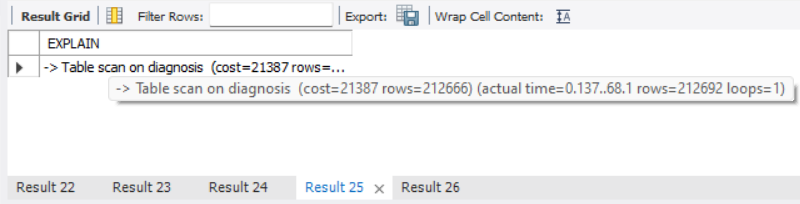


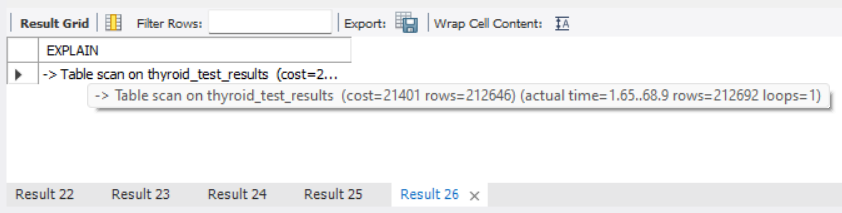






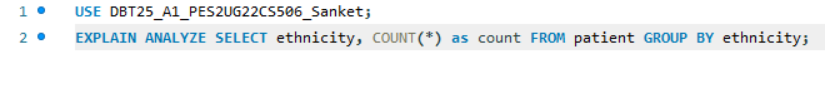


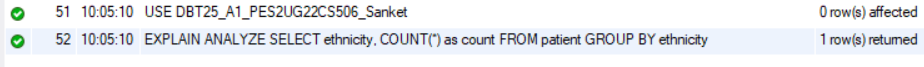


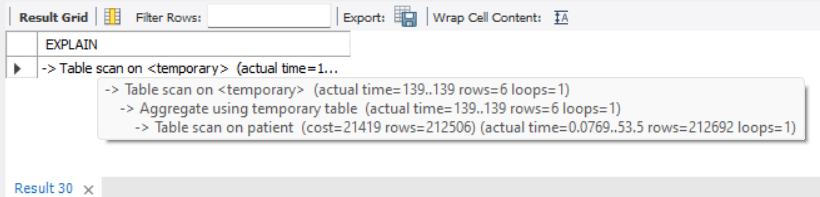


**Observation** -> No significant performance improvement due to lack of where clause , order by clause or join clause to leverage the indexed columns hence a regular table scan just like the case without indexing .

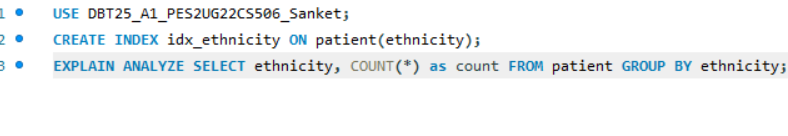
Count and Group by Query Before Index Creation-(table scan)

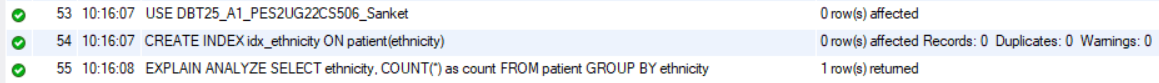


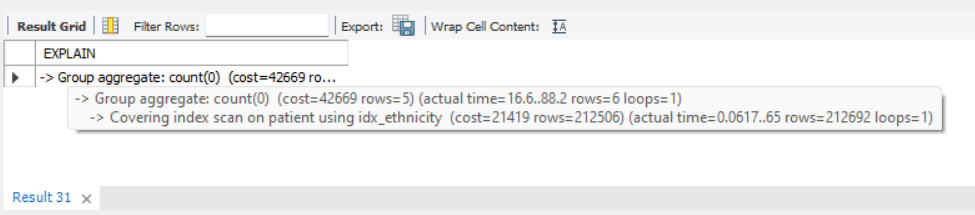




Count and Group by Query After Index Creation-(index scan)

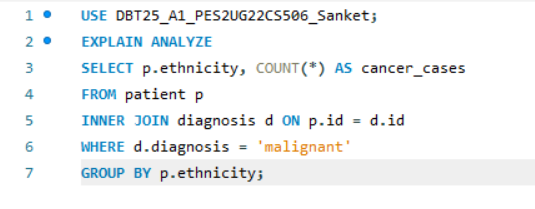




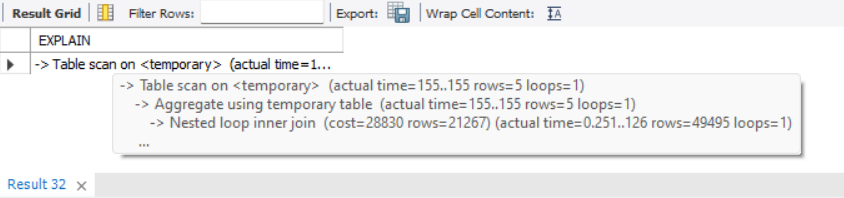


**Observation**-> Unlike the last comparison we see an actual significant cost reduction after the index creation which is due to the fact that we have used a group by statement to leverage the attribute/column on which index is created.

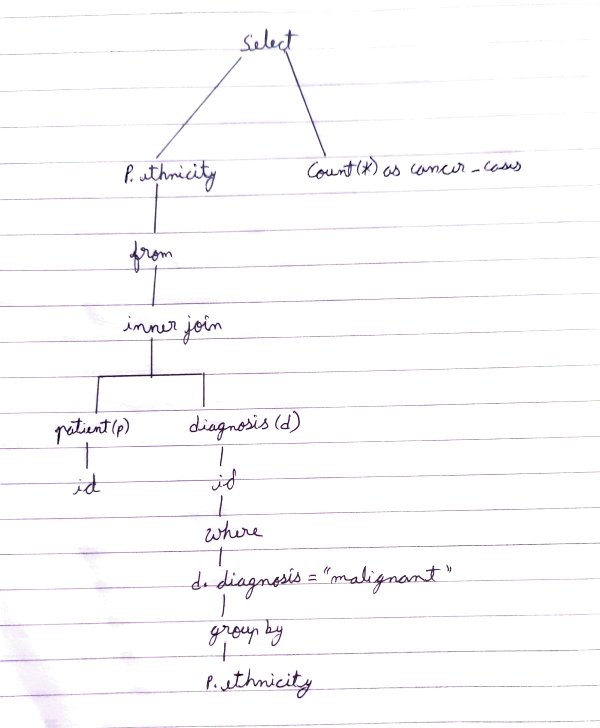
Join Query With Aggregation-



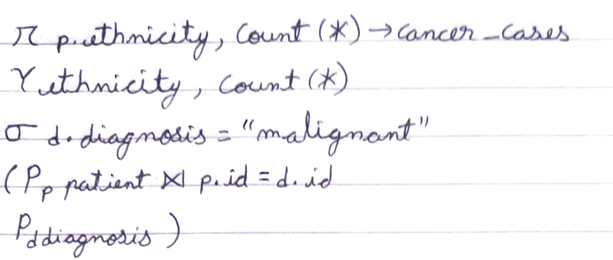




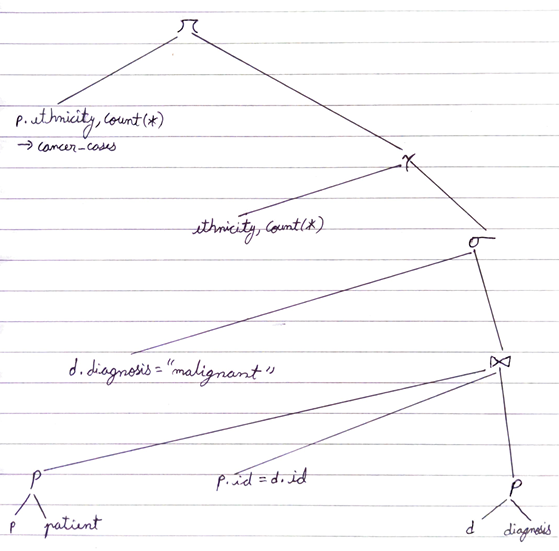
(Parse Tree)



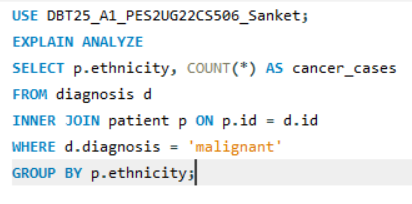
(Relational Algebra Query)



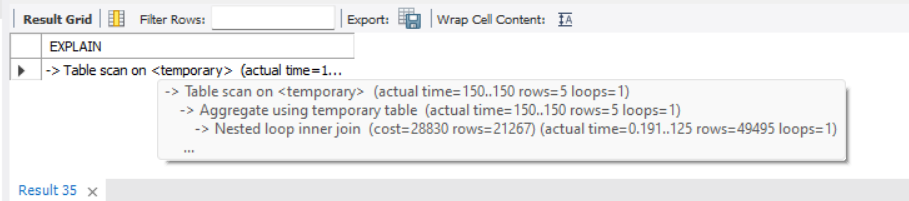
(Query Tree)



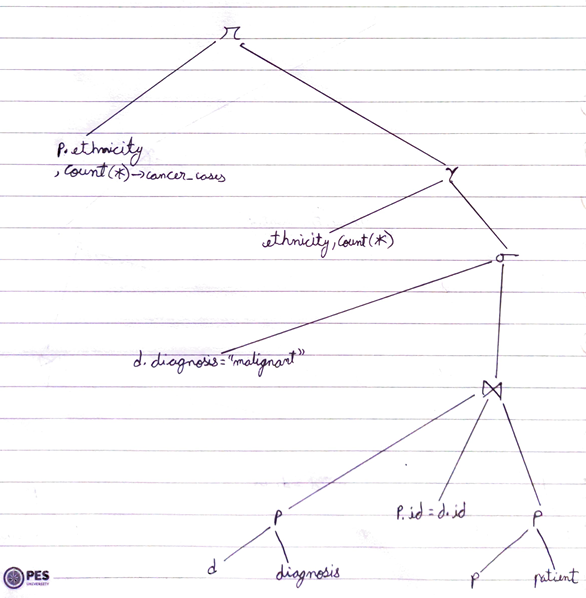
Join Query With Aggregation and Optimization (join order changed)-





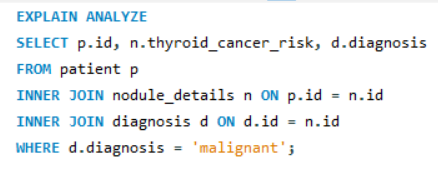


(Optimized Query Tree)

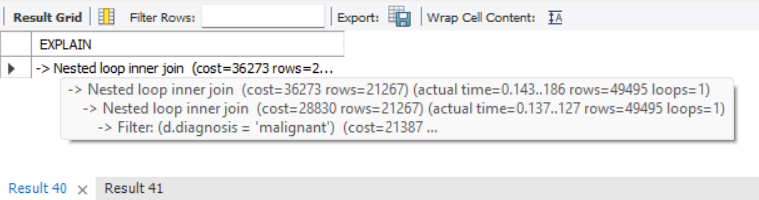


**Observation->** Very minimal performance improvement which is not too noticeable in this case as we are filtering with where clause before joining in this optimization.

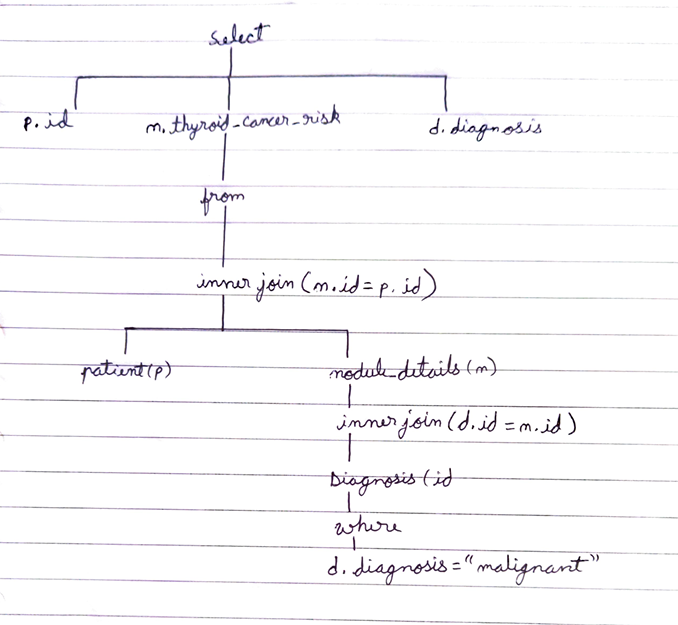
3 Table Join Query-



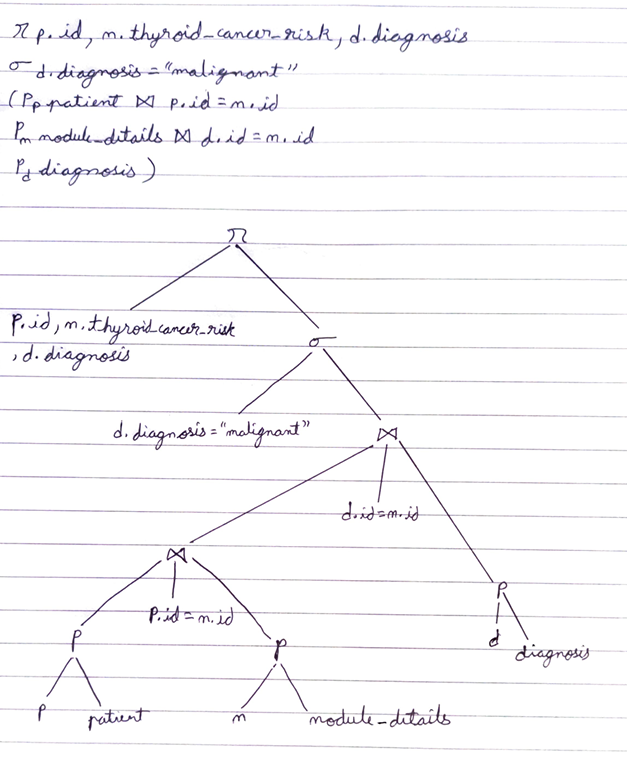




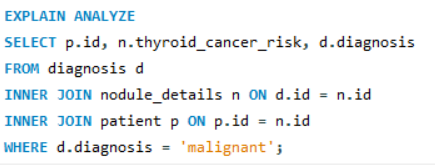
(Parse Tree)



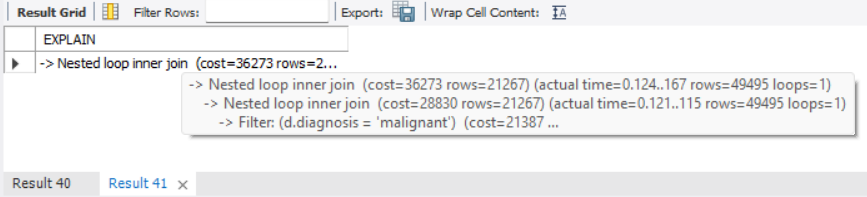
(Relational Algebra Query and Query Tree)



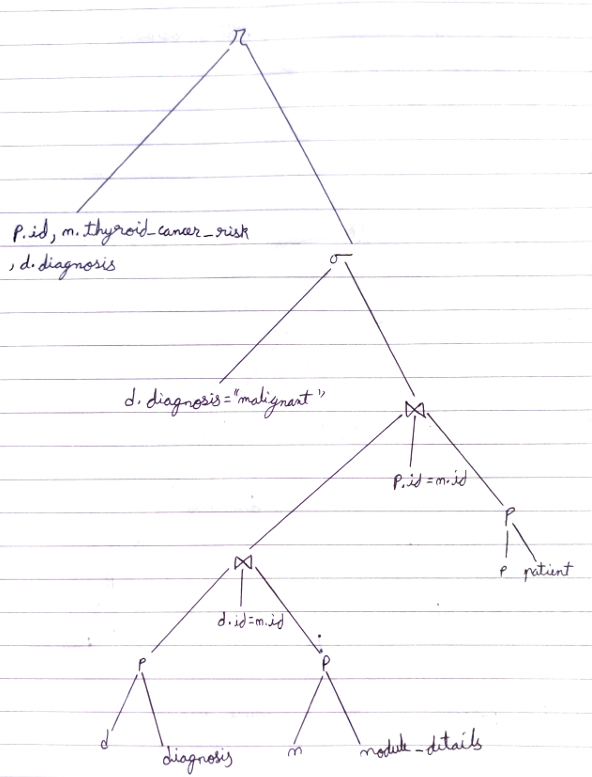
3 Table Join Query Optimization (Join order changed)-





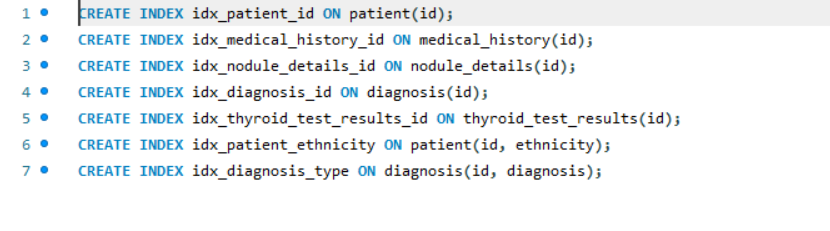


(Optimized Query Tree)



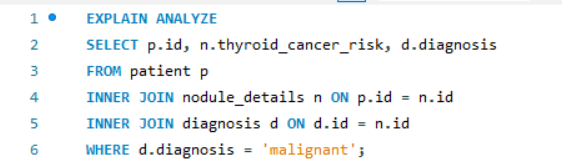
**Observation**-> The cost remains the same but there is minor performance improvements in terms of time taken which would be more

3 Table Join Query With Optimization (Index Creation)





(Unoptimized Join Order)





(Optimized Join Order)