**Topic: Query Execution Plan, Full Table Scan, Indexing**

**Learning objective:**

1. Explore the semantic analysis and query optimization components of DBMS using MySQL as a use case.

**Instruction**

Answer all the questions.

1. Given sematic analysis performed by a DBMS (not just MySQL), is any of the queries below rejected by the semantic analysis process? If it does, provide a reason for rejection.
2. SELECT enamep

FROM Emp; **The DBMS should reject this due to the unknown attribute “enamep”**

1. SELECT ename

FROM Emp where ename=2; **The DBMS should reject this due to incompatible types (string vs integer) however MySQL does not throw any errors while running this query.**

1. Is any index (a data structure to speed up a query) used in any of the queries below? To answer this question, use MySQL Workbench to visualize the cheapest query execution plan of each of the queries on the threetables database. The documentation “CheckQueryExecutionPlan.pdf” on Canvas shows how to do that. If you see the word “PRIMARY”, the index on the primary key is used. If you see the word “Full Table Scan”, a full table scan which reads the entire relation one page at a time is used. The relation name is shown under the word “Full Table Scan”. If you see the word “Full Index Scan”, the entire index is read. If you see the word “nested loop” or “hash join”, a join operation is done using a nested loop join algorithm or a hash join algorithm, respectively.
2. Which of the following queries, a full table scan is used?
3. Which of the following queries, indexes are used?
4. Which of the following queries, a join operation is done?

|  |  |
| --- | --- |
| (a) | Select \*  FROM Emp; **Uses a Full Table Scan** |
| (b) | SELECT \*  FROM Emp WHERE eid = 101; **Uses the Index created** |
| (c) | SELECT \* FROM Emp  WHERE ename like '%John%'; **Uses a Full Table Scan** |
| (d) | SELECT \* FROM Emp  WHERE ename = 'John'; **Uses a Full Table Scan** |
|  | Create index enameIdx on Emp(ename);  See what happen when you run queries (c) and (d) again. Then run the following statement.  drop index enameIdx on emp; |
| (d) | SELECT count(\*) FROM Emp; **Uses a Full Index Scan** |
| (e) | SELECT did, dname  FROM Dept where did < 10; **Uses a Index Range Scan** |
| (f) | SELECT e.eid, e.ename, w.did  FROM emp e inner join works w on e.eid = w.eid; **Uses Full index scan + Unique Key look up (Join Operation)** |