Team 17

107060008 張承勛
107062123 鄭宇軒
107062315 黃偉哲

Implementation

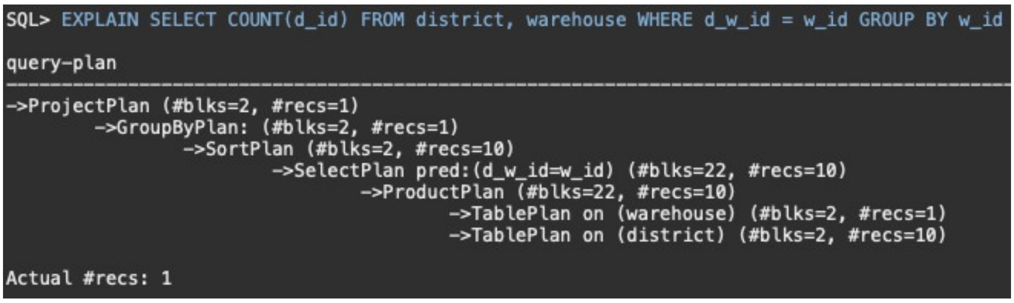
First and foremost, add the keyword “explain” into the keyword-list in `query/parse/Lexer` so that the lexer can recognize this new keyword. Then, check if the keyword “explain” is contained in the query inside `query/parse/Parser`; if so, eat it and pass the information to `QueryData`.

Next, create a new class as ExplainPlan.java and call it in the planner if we need to explain the query. By the specification of this homework, JDBC client will get the result through `RemoteResultSet.getString("query-plan")`, therefore we need to add a field called “query-plan” in the schema. Besides, add a method called `explainStr` in every plan so that each plan can explain its own estimations, and then we can call the method in ExplainPlan so that it can be operated recursively till the TablePlan. Collect all the information and pass it to ExplainScan. Thus, we also need to create a new class as ExplainScan.java for ExplainPlan.java to scan the output for ExplainPlan to employ. Again, by the specification of this homework, we should pass the information about those estimations from each plan to the method `getVal(String fldName)` inside ExplainScan so that the JDBC client can get the output successfully.

As for the number of actually accessed records, our implementation is to check how many s.next() can be called and return TRUE in ExplainPlan.

Experiment

- The example query

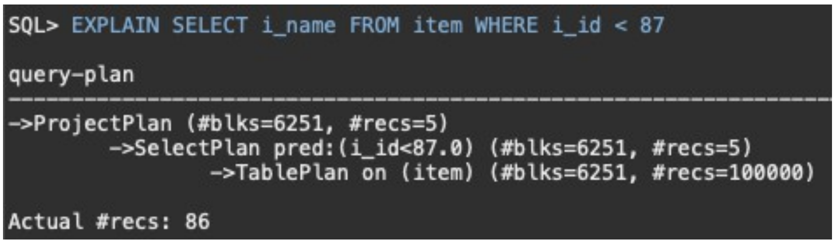


```
SQL> EXPLAIN SELECT COUNT(d_id) FROM district, warehouse WHERE d_w_id = w_id GROUP BY w_id

query-plan
------------------------------------------------------------------------------------------
->ProjectPlan (#blks=2, #recs=1)
        ->GroupByPlan: (#blks=2, #recs=1)
                ->SortPlan (#blks=2, #recs=10)
                        ->SelectPlan pred:(d_w_id=w_id) (#blks=22, #recs=10)
                                ->ProductPlan (#blks=22, #recs=10)
                                        ->TablePlan on (warehouse) (#blks=2, #recs=1)
                                        ->TablePlan on (district) (#blks=2, #recs=10)

Actual #recs: 1
```

- A query accessing single table with WHERE



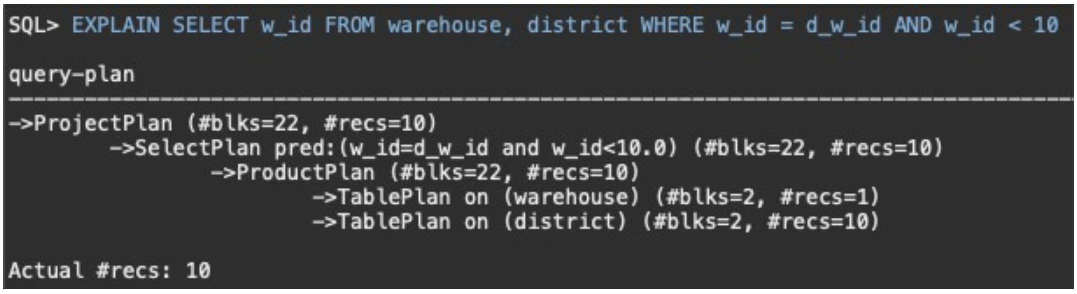
```
SQL> EXPLAIN SELECT i_name FROM item WHERE i_id < 87

query-plan
------------------------------------------------------------------
->ProjectPlan (#blks=6251, #recs=5)
        ->SelectPlan pred:(i_id<87.0) (#blks=6251, #recs=5)
                ->TablePlan on (item) (#blks=6251, #recs=100000)

Actual #recs: 86
```

In this experiment, it is clearly that the actual #recs is correct.

- A query accessing multiple tables with WHERE

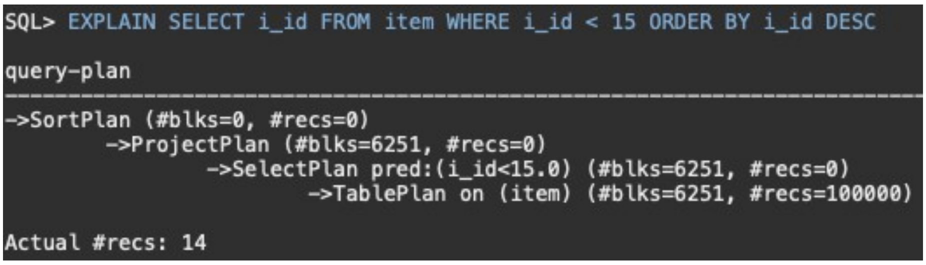


```
SQL> EXPLAIN SELECT w_id FROM warehouse, district WHERE w_id = d_w_id AND w_id < 10

query-plan
------------------------------------------------------------------------------------
->ProjectPlan (#blks=22, #recs=10)
        ->SelectPlan pred:(w_id=d_w_id and w_id<10.0) (#blks=22, #recs=10)
                ->ProductPlan (#blks=22, #recs=10)
                        ->TablePlan on (warehouse) (#blks=2, #recs=1)
                        ->TablePlan on (district) (#blks=2, #recs=10)

Actual #recs: 10
```

- A query with ORDER BY

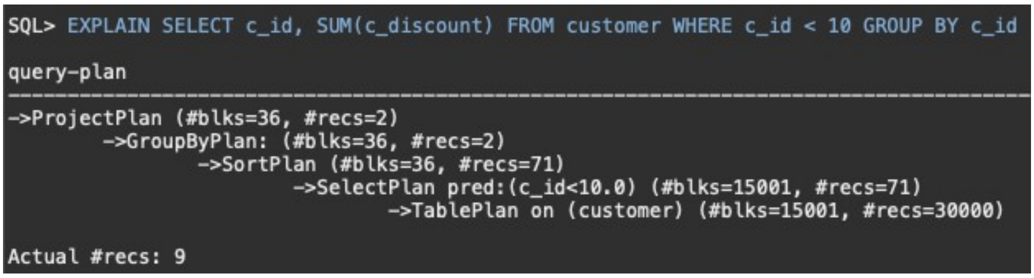


```
SQL> EXPLAIN SELECT i_id FROM item WHERE i_id < 15 ORDER BY i_id DESC

query-plan
-------------------------------------------------------------------------
->SortPlan (#blks=0, #recs=0)
        ->ProjectPlan (#blks=6251, #recs=0)
                ->SelectPlan pred:(i_id<15.0) (#blks=6251, #recs=0)
                        ->TablePlan on (item) (#blks=6251, #recs=100000)

Actual #recs: 14
```

- A query with GROUP BY and at least one aggregation function



```
SQL> EXPLAIN SELECT c_id, SUM(c_discount) FROM customer WHERE c_id < 10 GROUP BY c_id

query-plan
-------------------------------------------------------------------------------------
->ProjectPlan (#blks=36, #recs=2)
        ->GroupByPlan: (#blks=36, #recs=2)
                ->SortPlan (#blks=36, #recs=71)
                        ->SelectPlan pred:(c_id<10.0) (#blks=15001, #recs=71)
                                ->TablePlan on (customer) (#blks=15001, #recs=30000)

Actual #recs: 9
```