/**
 * Created by Xueyin Wang
 * 17 Feb 2016
 */

--Exercise one
step 3: handleQueryStatement()
In this function, it gets logical plan by parseQueryLogicalPlan(), then set the physical plan and logical plan.

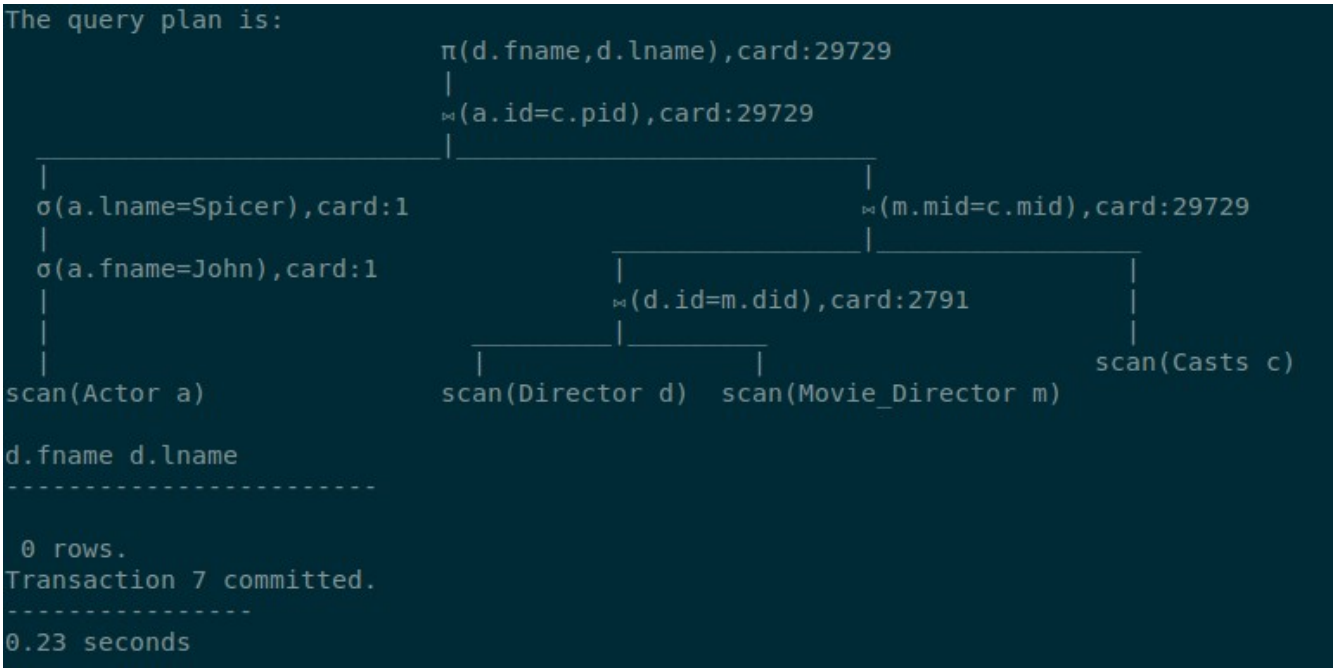
step 4: parseQueryLogicalPlan()
In this step, it parses where clause first to get filter and join nodes by processExpression();
Then, it checks validity of other parts of the query, like group by field and so on.

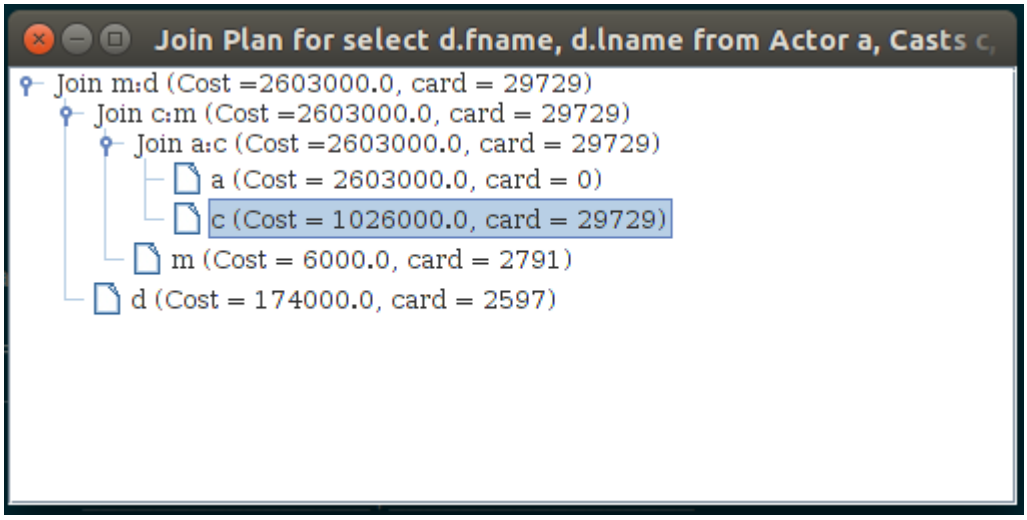
step 5: processExpression()
It recursively implements processExpression() function to get join nodes with where clause having "AND" or "OR"
Then add join node to LogicalPlan by addJoin(tab1field, tab2field, op)

parsing done

--Exercise six
6.1 1% version of IMDB dataset is chosen

```
select d.fname, d.lname from Actor a, Casts c, Movie_Director m, Director d where
a.id=c.pid and c.mid=m.mid and m.did=d.id and a.fname='John' and a.lname='Spicer';
```





There are several query plans with m, d, c, a join. However, the total cost of query plan d join m first, then join c, finally join a is the least. So, the optimizer select this plan.

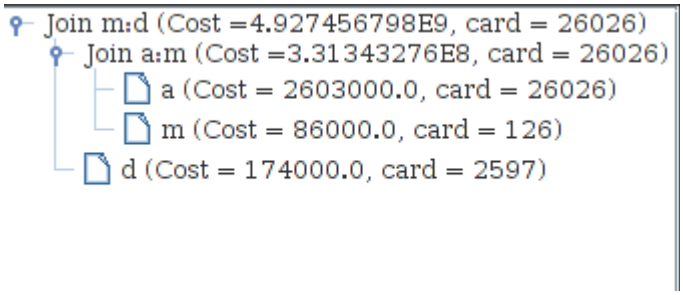
6.2

select a.fname, a.lname from Actor a, Movie m, Director d where a.id=m.id and m.id=d.id and m.year=2006;

```
The query plan is:
                              π(a.fname,a.lname),card:26026
                              |
                              ⋈(m.id=d.id),card:26026
                  ____________|_____________
                  |                        |
                  ⋈(m.id=a.id),card:26026  |
  ________________|_____________           |
  |                             |          |
  σ(m.year=2006),card:127       |          |
  |                             |     scan(Director d)
scan(Movie m)                 scan(Actor a)

a.fname a.lname
-----------------------

 0 rows.
Transaction 7 committed.
----------------
14.15 seconds
```



The optimizer chooses to select m first, then join it with a, finally to join with d. This plan cost least, which is about 4.92E9. So the optimizer chooses this plan.

--Changes to API
 # no big changes to return type, parameters, params type or function name

 --Incomplete Elements
 # functions required in project 3 are completed

 --Collaboration
 # It's a 2-person collaborated project, with my partner: Xiaoyang Xu
 # My work:
    Exercise 3
    Exercise 5
 # Partener's work:
   Exercise 2
   Exercise 4

 --Time Spend
 # Two days(6 hours / day) coding
 # Two hours debugging for tests and system tests

 --Confusing Points
 # For test cases which are randomly generated, the distribution is not uniform. There are possibilities that the percentage of cardinality is out of range.