Design and Implementation of Exists Subquery in Datafuse

Yizheng Jiao

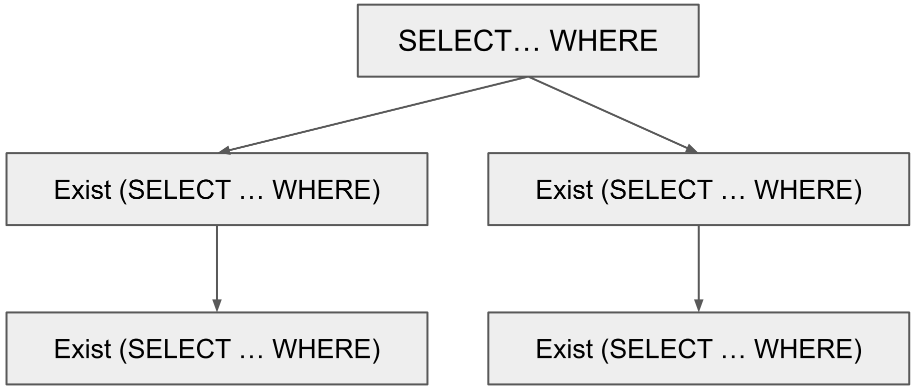
Content

Exists Introduction

- EXISTS is followed with a subquery
- EXISTS can be use with NOT
- If the subquery returns any rows at all
  - EXISTS subquery is TRUE
  - NOT EXISTS subquery is FALSE
- Examples
  - SELECT column1 FROM t1 WHERE **EXISTS** (SELECT * FROM t2);
  - SELECT column1 FROM t1 WHERE **EXISTS** (SELECT * FROM t2) and **EXISTS** (SELECT * FROM t3);
  - SELECT store_type FROM stores WHERE **NOT EXISTS** (SELECT * FROM cities WHERE **NOT EXISTS** (SELECT * FROM cities_stores WHERE cities_stores.city = cities.city));

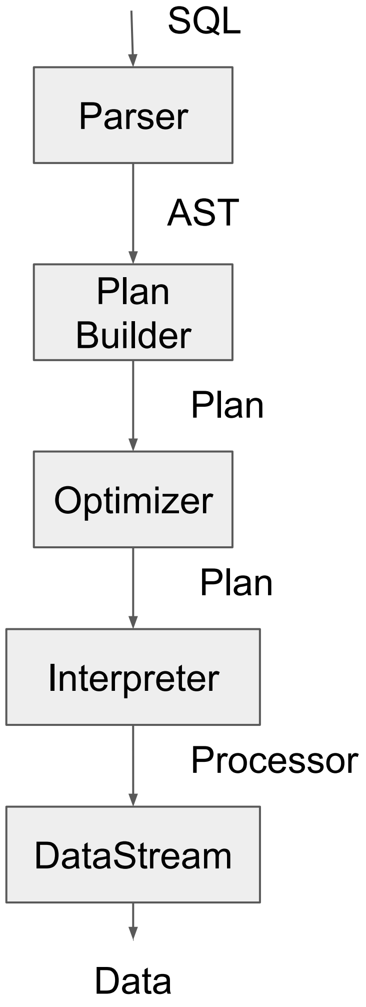
Design

- The main query depends on the result of Exists subquery
    - Evaluate the result of subquery first
    - Pass the result along the execution of the pipeline
- How to handle **nested** Exists?
    - Each Select can have multiple Exists
    - This forms tree structure
- Evaluate the leaf first
    - Given a root of a tree, traverse the nodes level-by-level
    - Use breadth-first-search to solve it



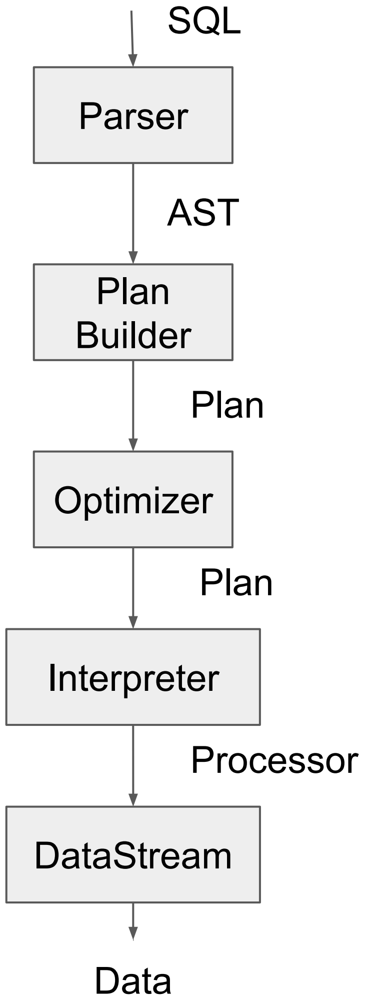
Implementation (1/2)

- Datafuse uses sqlparser as the parser
- Handle sqlparser::ast::Expr::Exists in PlanParser::sql_to_rex
- Build the plan for the subquery by calling PlanParser::query_to_plan
- Introduce a new expression --- Exists(Arc<PlanNode>)
- Exists subquery is an expression in FilterPlan
- This expression wraps a subquery plan



Implementation (2/2)

- After parser and plan builder, we get a serial of plans
    - ProjectionPlan ← FilterPlan ← ReadSourcePlan
- FilterPlan's Exists subquery with the same structure
    - Exists subquery is an expression in the predicate
- Schedule these plans carefully in **Interpreter::Executor**
    - The leaves plans are executed first
    - Save the result of Exists subqueries in a **HashMap**
    - Pass the map when execution the upper level query
- Use the unique name of Exists subquery to look the hashmap for subquery result to filter data stream
    - implemented in **ExpressionExecutor::execute**



How about the cluster mode?

- For cluster mode, the HashMap is transmitted to the flight server
    - Add a HashMap field to ExecutePlanWithShuffleAction
        - The HashMap is sent along with the RPC
- The flight server pass this HashMap around when evaluating the the query locally

Summary

Results

- Both Exists and Not Exists subquery can return collect answer
- Nested Exists subquery also works fine

Limitation

- Not incorporate with the implementation of the pipeline/processor
- Not applicable to In operator
    - Passing HashMap with the datablocks as values is expensive

Future work

- Use the current pipeline design to schedule plans automatically
- Implement In operator and go back to unify them

Thanks for your attention!

Q & A