Nested Query Optimization

Group 6: Xinzhu Cai, Guancheng Li, Ian Romines
Goals

75% (Done) -> Support nested queries in optimizer
  ● Transform 4 basic types of nesting into joins with transformation rules

100% (Partially) -> Support nested predicates in execution engine
  ● COMPARE_IN operation - LogicalSemiJoin operator
  ● COMPARE_EQUAL operation
  ● (✗) COMPARE_NOT_IN operation, requires AntiJoin

125% (Todo) -> Rewrite views into nested queries
  ● Store view in system catalog
  ● Replace view nodes with operator trees

Summary: between the 75% - 100% goal
Project Goal Reiteration

Unnesting algorithms in optimizer

```
SELECT pno from shipment where sno = (select max(sno) from supplier where subdept = qty);
```
Project Goal Reiteration

Support nested predicates in execution engine

SELECT pno from shipment where sno in (select sno from supplier where subdget = qty);
Test & Benchmarks

- Correctness is tested with a mix of C++ code and java code
  - C++ code is for a specific functionality. e.g. Check if the transformation rules are applied successfully
  - Junit test is for an end-to-end test. 10 test cases for each type
Code Quality

- **Strong**: Well-defined and flexible transformation rules
- **Weak**: Avoid materialization, should we use CTE nodes in the future?
Future Work

- Support views by rewriting them into nested queries
- Materialization techniques & cost model
- Deal with Type D by supporting set operations in terrier
Thank you!