User Query Graph

```
Reader<Supervisor> ("db1", "allSup")

SelectComp<Supervisor, Supervisor>

Lambda <bool> getSelection (
```

```
return makeLambdaFromMethod (arg,
getDept) !=
makeLambdaFromLiteral("EE"); }
```

Handle<Supervisor> arg) {

SelectComp<Employee, Supervisor>

```
Lambda <bool> getSelection (
    Handle<Supervisor> arg) {
    return makeLambdaFromMethod (arg, getDept) ==
    makeLambdaFromLiteral("CS"); }
Lambda <Employee> getProjection (
    Handle<Supervisor> arg) {
    return makeLambdaFromMethod (arg,
```

Writer<Employee> ("db1", "csEmp"

PC API/Lambda Calculus

getEmp); }

Optimized TCAP Program

```
//read in data
In(sup) <= ScanSet ("db1", "allSup");</pre>
//push down projection
AppliedMethodCall1Select1(sup, emp)
<= Apply (In(sup), In(sup, emp),
'Select1', 'getEmp');
//fusion of the two filtering conditions
AppliedMethodCall0Select1(sup, emp,
dept) <= Apply
(AppliedMethodCall1Select1(emp),
AppliedMethodCall1Select1(sup, emp),
'Select1', 'getDept');
Filtered0(sup, emp) <= Filter
(AppliedMethodCall0Select1(dept),
"CS", 'Select1');
//write the results
Out(emp) <= WriteSet("db1", "csEmp");
```

Compiled Pipeline Stages

```
Each thread:
for (...) {
 VecListPtr curVecList = getNextVectorList(...);
 for (PipelineStage q : pipeline) {
  try
       curVecList = q->process(curVecList);
   catch (NotEnoughSpaceInCurrentBlock e) {
       postprocess(...);
                                   PC Object has
Distributed Execution:
                                   identical
                shuffle
                                   representation
                                   across disk and
                                   network, so no
                                   (de-)
           persistence
                                   serialization
                                   overhead.
 disk
                     disk
```

PC Execution Engine