Melissa Galonsky
CS133-HM
March 23, 2016

3. simpledb.Parser.handleQueryStatement()
   This method starts by generating logical and physical plans for the query by calling

   ```
   LogicalPlan lp = parseQueryLogicalPlan(tId, s);
   ```

   and then invoking

   ```
   DbIterator physicalPlan = lp.physicalPlan(tId,
                   TableStats.getStatsMap(), explain);
   ```

   It then packages these in a query object and returns it.

4. parseQueryLogicalPlan(TransactionId tid, ZQuery q)
   This method creates a LogicalPlan object. It then creates objects for holding intermediate representations of the FROM, WHERE, GROUP BY, and SELECT clauses, and parses the clauses. When parsing the select clause, it figures out what fields it needs to project onto and if it has any aggregates, and adds this information to the LogicalPlan.

5. `physicalPlan(TransactionId t, Map<String,TableStats> baseTableStats, boolean explain)`
   This method is responsible for creating the physical plan encapsulated in a DBIterator by running methods that use the appropriate statistics to estimate an optimal join. It starts by gathering statistics on every table. It makes the call

   ```
   joins = jo.orderJoins(statsMap,filterSelectivities,explain);
   ```

   which estimates a series of optimal joins, returned as a Vector of LogicalJoinNodes, wich contain information about joins. The method then creates the iterators needed for the joins and packages them together into one to return.