CSCI 3357: Database System Implementation
Homework Assignment 9
Due Monday, November 25

1. In HW 6 you added the methods `afterLast` and `previous` to `TableScan`.

a) Add these methods to the `Scan` interface and the classes `SelectScan`, `ProjectScan`, and `ProductScan`. You will discover that there will be scan classes in other packages that will no longer compile. You are not obligated to fix them, although you can if you wish.

b) Go to the package simpledb.jdbc.embedded, and add appropriate methods `afterLast` and `previous` to the class `EmbeddedResultSet`.

2. In HW 8 you implemented NULL constants and modified the parser to understand them. You also added the method `isNull(fldname)` to `TableScan`. Unfortunately, the JDBC `ResultSet` interface does not have the method `isNull`. Instead, it has the method `wasNull()`. This method takes no arguments, and returns `true` if the most recently retrieved value was a null. For example, the following JDBC code prints the name and major id of all students, printing "null" when a major id is null.

```
ResultSet rs = stmt.executeQuery("select sname, majorid
                                  from student");
while (rs.next()) {
   String s = rs.getString("sname");
   int major = rs.getInt("majorid");
   if (rs.wasNull())
       System.out.println(s + " null");
   else
       System.out.println(s + " " + major);
}
```

a) In HW6 you added the method `isNull` to `TableScan`. Now add the method to the `Scan` interface and the classes `SelectScan`, `ProjectScan`, and `ProductScan`, just like you did in problem 1a.

b) Go to the package simpledb.jdbc.embedded, and add the method `wasNull` to the class `EmbeddedResultSet`. You will also need to modify `getInt` and `getString` to check to see if the retrieved value is null. If so, they should set a flag so that `wasNull` will know to return true.

3. In SimpleDB, all clients use JDBC's *autocommit* mode, in which the current transaction is committed automatically whenever an update statement executes or a result set closes. In standard JDBC a client can turn off autocommit mode, so it can commit and rollback its transactions explicitly. The JDBC `Connection` interface has a method `setAutoCommit(boolean ac)`, which turns auto-commit mode on or off based on the argument, a method `getAutoCommit`, which returns the current auto-commit status, and the methods `commit` and `rollback`.

Your task is to implement `setAutoCommit` and `getAutoCommit` in the class `EmbeddedConnection`. Note that the methods `commit` and `rollback` are already public, so you don't need to do anything further with them. However, you will need to modify the `executeUpdate` method of `EmbeddedStatement` and the `close` method of `EmbeddedResultSet` appropriately.

4. Rewrite the demo client program `SimpleIJ` in the following ways:
   - It runs with autocommit mode off.
   - It has two new commands, "commit" and "rollback", which allow the user to commit or roll back the current transaction, at will.
   - It always prints the output of a query in reverse order.
   - Whenever it prints a value, it checks to see if the value is null, and if so prints "null".

Call the new program `HW9IJ`.

For this assignment, your modifications to JDBC only need to work for embedded connections. Optionally, you can extend your modifications to network connections by modifying classes in the package simpledb.jdbc.network. Read sections 11.3-11.5 of the text for details.