```java
package fi.paatti.paattidatabaseutils.dbservice.querydelegates;


import com.vaadin.data.util.sqlcontainer.RowItem;
import com.vaadin.data.util.sqlcontainer.TemporaryRowId;
import com.vaadin.data.util.sqlcontainer.query.generator.StatementHelper;
import fi.paatti.paattidatabaseutils.names.PaattiColumnNames;
import fi.paatti.paattidatabaseutils.names.PaattiTableNames;
import java.sql.*;


/**
 * A delegate for CHOICE-table.
 *
 * @author Lauri Satokangas, lauri.n.satokangas@student.jyu.fi
 * @since 0.0.3
 */
public class ChoiceQueryDelegate extends PaattiQueryDelegate {
  private static final long serialVersionUID = 1L;
  private final Object eventId;

  /**
   * Constructor.
   *
   * @param eventId The ID of the event.
   */
  public ChoiceQueryDelegate(Object eventId) {
    super(PaattiTableNames.CHOICE, PaattiColumnNames.CHOICE_choiceID,
        PaattiColumnNames.getColumnNamesInArray(PaattiTableNames.CHOICE));
    this.eventId = eventId;
  }

  /**
   * {@inheritDoc }
   */
  @Override
  public int storeRow(Connection conn, RowItem row) throws SQLException {
    PreparedStatement storeStatement;
    StringBuilder querySb = new StringBuilder();
    int retval;

    if (row.getId() instanceof TemporaryRowId) {
      querySb.append(getInsertQueryString());
      storeStatement = conn.prepareStatement(querySb.toString(), Statement.RETURN_GENERATED_KEYS);
      setRowValues(storeStatement, row);
      retval = storeStatement.executeUpdate();
```

```
      ResultSet generatedKeys = storeStatement.getGeneratedKeys();

      if (generatedKeys.first()) {
        Object object = generatedKeys.getObject(1);

        fireNewRowIdEvent(object);
      }
    } else {
      querySb.append(getUpdateQueryString());
      storeStatement = conn.prepareStatement(querySb.toString());
      storeStatement.setInt(7, (Integer) row.getItemProperty(PaattiColumnNames.CHOICE_choiceID).getValue());
      setRowValues(storeStatement, row);
      retval = storeStatement.executeUpdate();
    }

    storeStatement.close();
    return retval;
  }

  /**
   * {@inheritDoc }
   */
  @Override
  protected void setRowValues(PreparedStatement statement, RowItem row) throws SQLException {
    statement.setString(1, row.getItemProperty(PaattiColumnNames.CHOICE_rowStatus).toString());
    statement.setInt(2, (Integer) row.getItemProperty(PaattiColumnNames.CHOICE_TASK_leadsto_taskID).getValue());
    statement.setInt(3, (Integer) row.getItemProperty(PaattiColumnNames.CHOICE_TASK_contains_taskID).getValue());
    statement.setInt(4, (Integer) row.getItemProperty(PaattiColumnNames.CHOICE_value).getValue());
    statement.setInt(5, (Integer) row.getItemProperty(PaattiColumnNames.CHOICE_sequence).getValue());
    statement.setString(6, (String) row.getItemProperty(PaattiColumnNames.CHOICE_description).getValue());
  }

  /**
   * Returns a new StatementHelper containing the SELECT query statement.
   *
   * @param offset the starting row.
   * @param limit the number of rows to get.
   * @return The StatementHelper containing the SELECT query.
   * @throws UnsupportedOperationException.
   */
  public StatementHelper getQueryStatement(int offset, int limit) throws UnsupportedOperationException {
    StatementHelper queryHelper = new StatementHelper();

    queryHelper.setQueryString(createQueryString(queryHelper, offset, limit));
    return queryHelper;
  }
```

```
  /**
   * Creates the SELECT query statement.
   *
   * @param queryHelper
   * @param offset Starting row.
   * @param limit Number of rows to get.
   * @return The SELECT query string.
   */
  private String createQueryString(StatementHelper queryHelper, int offset, int limit) {
    StringBuilder querySb = new StringBuilder();

    querySb.append(" SELECT ").append(getArrayAsStringWithCommas(getColumnNames())).append(" FROM ").append(
        PaattiTableNames.CHOICE);
    querySb.append(" LEFT OUTER JOIN ").append(PaattiTableNames.TASK).append(" ON ");
    querySb.append(PaattiColumnNames.TASK_taskID).append(" = ").append(PaattiColumnNames.CHOICE_TASK_contains_taskID);
    querySb.append(" WHERE ").append(PaattiColumnNames.TASK_EVENT_eventID).append(" = ").append(eventId);
    // Add possible filters.
    if (getFilters() != null) {
      querySb.append(getFilterStringWithoutWhere(queryHelper, true));
    }
    querySb.append(" AND ").append(PaattiColumnNames.TASK_rowStatus).append(" = '").append(ROWSTATUS_ACTIVE).append("'");
    querySb.append(" AND ").append(PaattiColumnNames.CHOICE_rowStatus).append(" = '").append(ROWSTATUS_ACTIVE).append(
        "'");
    if (offset != -1 && limit != -1) {
      querySb.append(" LIMIT ").append(offset).append(", ").append(limit);
    }
    return querySb.toString();
  }

  /**
   * {@inheritDoc }
   */
  @Override
  public StatementHelper getCountStatement() throws UnsupportedOperationException {
    StatementHelper countHelper = new StatementHelper();
    StringBuilder querySb = new StringBuilder("SELECT COUNT(*) FROM (");

    querySb.append(createQueryString(countHelper, -1, -1)).append(") as COUNT");
    countHelper.setQueryString(querySb.toString());
    return countHelper;
  }
}
```