Table of Contents

Displaying data from a database table is a basic requirement for database applications. Restricting the data volume is another requirement. This can be done on the client side as well as at the database.

At the client, data is filtered in the main memory.However, this requires that all data is transferred from the server/database to the client. This limits communication and should not be an issue for small data volume, but it should be noted that data has to be updated manually!

Filtering at the database is the default setting and is usually the better choice. Here, the sorting conditions are included in the SQL command and the data volume is therefore limited by the database. Only the limited data is provided to the client and, due to load-on-demand, only the required amount is transferred. In addition, the transferred data is always current.

Example

Our application manages contact information (people, addresses, etc.). One of our forms has a field in which a search can be entered. A button is available to start a wildcard search by first name, last name, street address, or city.

The following client action achieves the desired filtering:

```
/**
 * Searches the contacts with the search text.
 *
 * @throws ModelException if the search fails
 */
public void doFilter() throws ModelException
{
   String sText = (String)drSearch.getValue("SEARCH");

   if (sText == null)
   {
      //reset the filter: show all rows
      rdbContacts.setFilter(null);
   }
   else
   {
      //set the filter: show only found rows
      ICondition filter = new LikeIgnoreCase("FIRSTNAME", "*" + sText +
"*").or(
                     new LikeIgnoreCase("LASTNAME", "*" + sText + "*").or(
                     new LikeIgnoreCase("STREET", "*" + sText + "*").or(
                     new LikeIgnoreCase("TOWN", "*" + sText + "*"))));
      rdbContacts.setFilter(filter);
   }
}
```

The filter or condition is applied to a RemoteDataBook and is constructed just as it would in SQL. The search is based on our condition for the appearance of the entered text, either first name, last name, the street, or the city.

For sorting at the client, the RemoteDataBook would have to be configured as follows:

```
rdbContacts.setMemFilter(true);
```

The following filter classes are available:

- ContainsIgnoreCase
- EndsWithIgnoreCase
- Equals
- Greater
- GreaterEquals
- Less
- LessEquals
- Like
- LikeIgnoreCase
- LikeReverse
- LikeReverseIgnoreCase
- StartsWithIgnoreCase
- Not
- Or
- And

Please note that a filter must always be reassigned after changing it, e.g.:

```
filter = filter.and (new Equals ("CODE", "A-123"). and (new Equals ("REF",
"re133")));
```

Without the assignment of `filter = filter …` the filter is not changed.

For further details, see javadoc.