



Erl

Wood

Cheminformatics

 **KNIME** – software's social network

*Lilly*

Answers That Matter.



Computational Drug Discovery group at the ErlWood Manor, Surrey site of Eli Lilly

Early adopters of KNIME(since 2006)

**General KNIME approach:**

**Enable end-users (research chemist) to conduct computational chemistry workflows on their own**

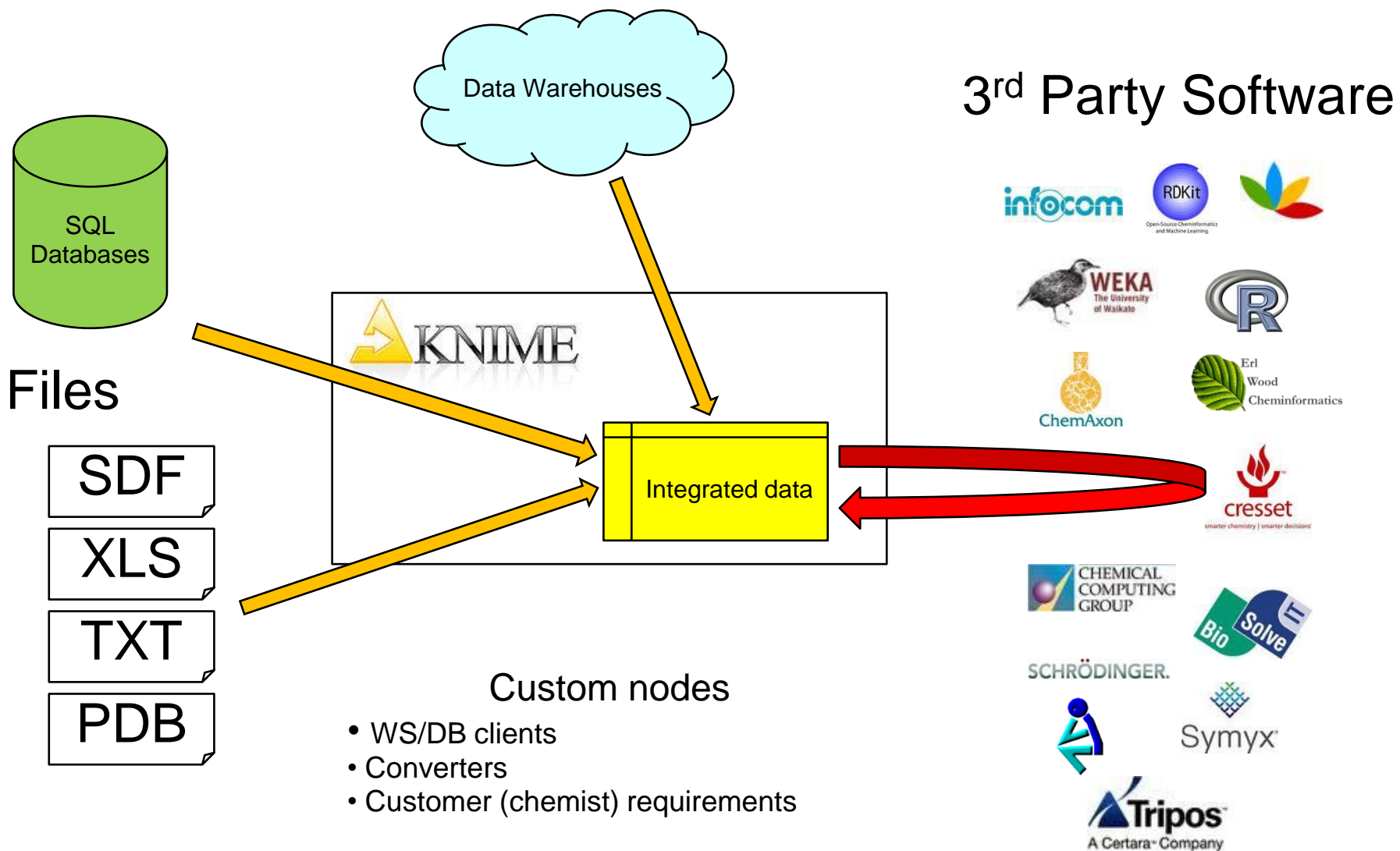


- Train users
- Wrap complex common tasks into single nodes
- Provide interfaces to internal data sources and services
- Provides interfaces to/between 3<sup>rd</sup> party software



Over 80 nodes developed internally  
26 released as community contributions

# Motivation



# Erl Wood Community Node Collection

**File input**

- Chemical Reactions File Reader (Node 1) [£]

**Local database**

- Reaction Vectors Database Reader (Node 20)
- Reaction Generator (Node 19)
- Reaction Vectors Database Writer (Node 21)

**Webservice**

- Docking Job Lister (Node 8)
- Docking Job Submitter (Node 10)
- Docking Job Retriever (Node 9)

**Converters and preprocessors**

- Fingerprints Expander (Node 6)
- Old Bit Vector To New Bit Vector (Node 7)
- Column Merger (Node 3)
- Matched Pairs Finder (Node 17)

**Result presentation and visualizing**

- 2D/3D Scatterplot (Node 27)
- Jmol Docking Pose Viewer (Node 22)
- Similarity Viewer (Node 24) [\*]
- Jmol Viewer (Node 23)
- Virtual Screening Metrics (Node 5)
- Vida Viewer (Node 25)

**Special functionalities**

- MCS Distance (Node 14)
- Desirability (Node 12)
- RGroup Efficiency (Node 18) [£]
- Matched Pairs Detector (Node 16) [\*]
- MCS Matrix (Node 15) [£]
- Activity Cliffs Viewer (Node 2) [£]
- Pareto Ranking (Node 13)
- Fingerprint Similarity (Node 4) [\*]
- Text Input (Node 11)

\* uses RDKit      £ Chemaxon Marvin license required

and growing ...

# Data and Software Infrastructure

---

## Make different data sources available in KNIME

File readers

SQL Wrappers

Hide SQL from user

Problem for OS release: General DB interface

Webservice clients

Server-side execution of computational expensive jobs (e.g. Docking)

Platform integration; some external tools might only run on Unix

Machine-locked licenses sharing

Data warehouse interface

Problem for OS release: Server-side is user specific, only WSDL interface can be specified

Special column types

Visualization

Serialization of complex data

Filtering of compatible columns

# SQL Backbone - De Novo Design

The image displays the KNIME software interface. On the left is the 'Preferences' dialog, with the 'Database Driver' section selected. The main workspace shows a workflow with four nodes: 'MarvinSketch' (Node 11), 'Chemical Reactions File Reader' (Node 10), 'Reaction Vectors Database Writer' (Node 12), and 'SDF Reader' (Node 14). A yellow dialog box is overlaid on the workflow, titled 'Select Database Driver'. It contains the following fields: 'Select Database Driver' (org.h2.Driver), 'Protocol' (jdbc:h2:), 'Reactions column' (RAW Reaction), 'Timeout (sec)' (30), 'Selected File' (C:\Documents and Settings\yex7847\My Documents\Kni...), 'User name', and 'Password'. A yellow arrow labeled 'Dialog' points from the 'Reaction Vectors Database Writer' node to the dialog box. Below the dialog box, the text 'Database information: Driver, URL, UID, PW' is visible. The 'Reaction Generator' node (Node 13) is also present in the workflow.

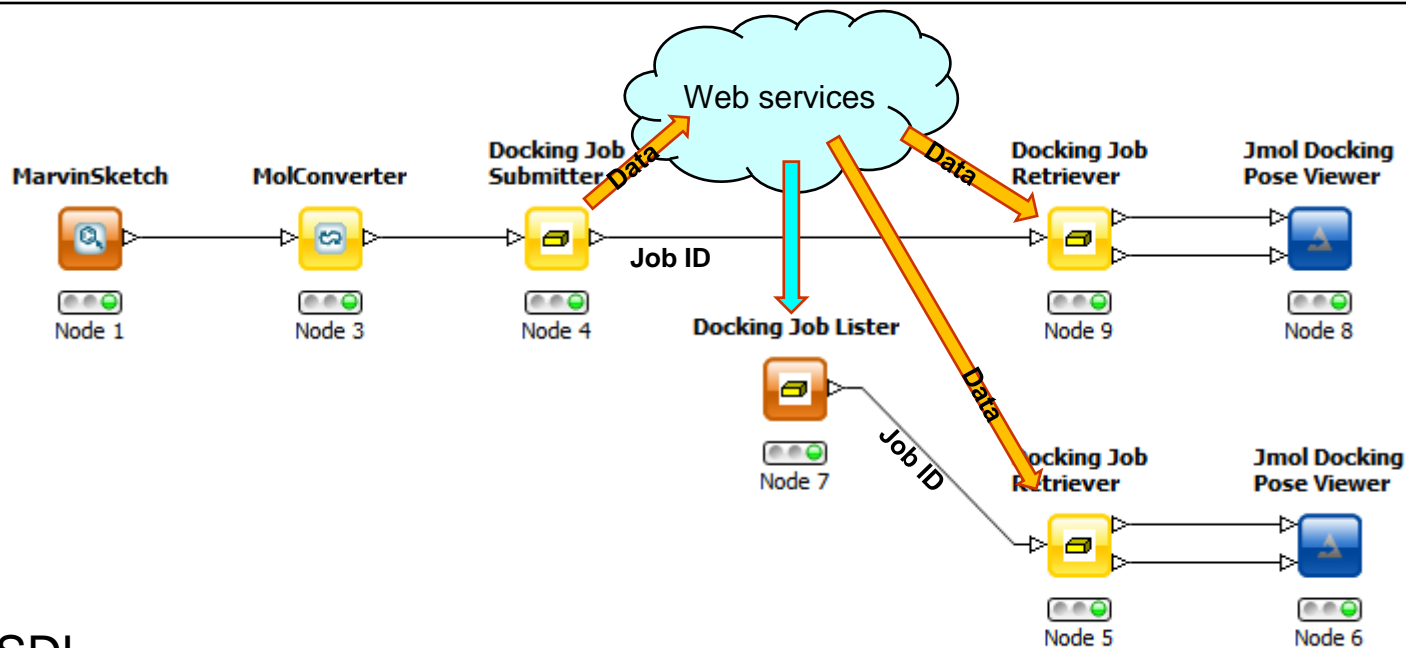
# SQL Backbone - De Novo Design

Generated reactions are stored in a SQL database

Drivers are obtained from the driver installation panel in KNIME → the same DB drivers can be used

```
String[] info = m_driver.getStringValue().split("£");//obtain driver and protocol
String url = new File(fname).toURI().getPath();
url = url.replaceAll(" ", "%20");
DatabaseConnectionSettings db = new DatabaseConnectionSettings(
    info[0], info[1]+url, m_db_user.getStringValue(), m_db_pass.getStringValue(), null);
try {
    connection = db.createConnection(getCredentialsProvider());
} catch (Exception e) {
    ...
}...
```

# Webservices – Molecular Docking



## WSDL

```
- <message name="GOLDdockSubmit">
  <part name="parameters" element="tns:GOLDdockSubmit" />
</message>
- <message name="GOLDdockRetrieve">
  <part name="parameters" element="tns:GOLDdockRetrieve" />
</message>
- <message name="listGOLDmodels">
  <part name="parameters" element="tns:listGOLDmodels" />
</message>
- <message name="GOLDdockListCompleted">
  <part name="parameters" element="tns:GOLDdockListCompleted" />
</message>
```

# Webservices – Molecular Docking

Depends on the Webservice client plugin from the KNIME labs → provides the required Apache libraries and useful tools

Relies on a specific webservice API for submitting, requiring and querying docking jobs

```
//Query models
import org.knime.ext.webservice.client.node.CxfClient;
import org.knime.ext.webservice.client.node.SingleParamNode;
.
.
.
CxfClient cxfClient = CxfClient.getCachedClient(new URI(m_endpoint.getStringValue()));
BindingInfo bindingInfo = cxfClient.getBindingInfo(
    new QName(m_namespace.getStringValue(), "EWCDDPortBinding"));
BindingOperationInfo bindingOperationInfo = cxfClient.getBindingOperationInfo(
    bindingInfo, new QName(
        m_namespace.getStringValue(), DockingJobSubmitterNodeModel.GOLDWS_OP_LISTRESULTS));
SingleParamNode inputParamRoot = cxfClient.createInputParameterTree(bindingOperationInfo);
SingleParamNode outputParamRoot = cxfClient.createOutputParameterTree(bindingOperationInfo);
BindingOperationInfo oi = cxfClient.getBindingOperationInfo(bindingInfo, new
    QName(m_namespace.getStringValue(), DockingJobSubmitterNodeModel.GOLDWS_OP_LISTRESULTS));
oi = oi.getUnwrappedOperation();
String user = System.getProperty("user.name");
final Object[] os = cxfClient.getClient().invoke(oi, user);
String ret = (String)os[0];
```

# General Interest Nodes

---

## Nodes we were missing in KNIME and OS extensions

(some of them were made also available by others in the meantime)

3D scatterplot

Column merger

Coloured / pivoted XLS writer (not yet available)

Multi-objective optimization (Pareto, Desirability)

Special (cheminformatics) nodes

- Fingerprint similarity ranking / virtual screening metrics

- Activity cliffs visualization

- Matched molecular pairs

- Maximum Common Substructures (matrix, similarity, *Marvin license required*)

# 2D/3D Scatterplot

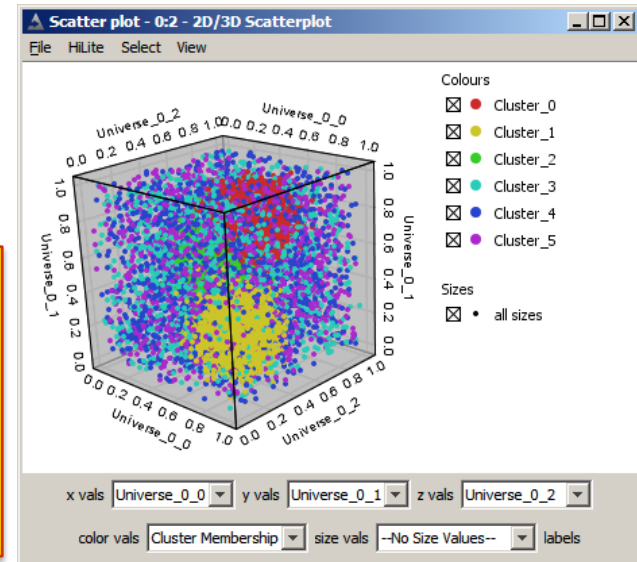
Renders many thousand data points

Visualizes additional property by colour and size

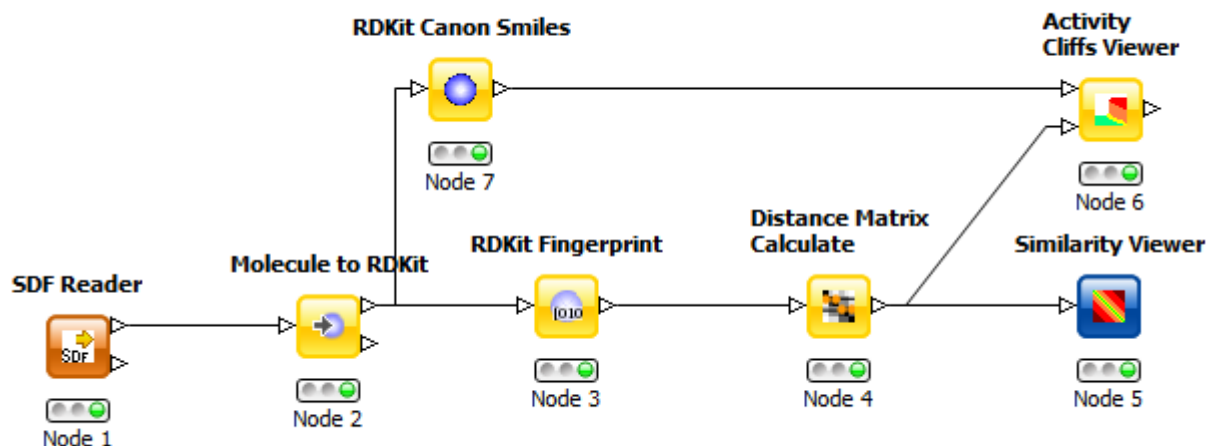
Can reuse column renderers to label points

```
DataCell dc = r.getCell(fInSpec.findColumnIndex(s));
DataType dt = fInSpec.getColumnSpec(
    fInSpec.findColumnIndex(s)).getType();
Component cc = dt.getRenderer(fInSpec.getColumnSpec(
    fInSpec.findColumnIndex(s))).getRendererComponent(dc);
pp.add(new JLabel(s),gridBagConstraints);
gridBagConstraints.gridx = 1;
pp.add(getLabel(cc),gridBagConstraints);
pp.add(cc,gridBagConstraints);
```

```
public Component getLabel(Component c) {
    c.setSize(c.getPreferredSize());
    BufferedImage bi = new BufferedImage(c.getSize().width,c.getSize().height,BufferedImage.TYPE_INT_RGB);
    Graphics g = bi.getGraphics();
    Color cc = g.getColor();
    g.setColor(Color.white);
    g.fillRect(0, 0, bi.getWidth(), bi.getHeight());
    g.setColor(cc);
    c.paint(g);
    JLabel ll = new JLabel();
    ll.setIcon(new ImageIcon(bi));
    ll.setHorizontalAlignment(SwingConstants.LEFT);
    return ll;
}
```



# Activity cliffs visualization



Could also be regarded as general purpose conditional clustering visualization

1. Rows are clustered according to similarity matrix
2. Colours indicate target properties

# Matched pairs

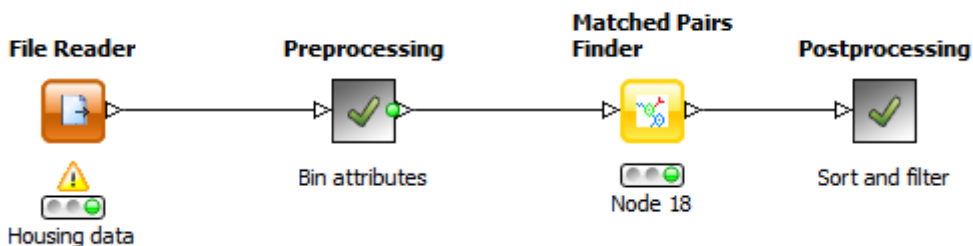
Detector is specialized on molecules and automatically finds all appropriate matched pairs

Finder requires a precomputed decomposition of molecules, because it just searches for row pairs that differ in *exactly* one of the selected string columns

→ can be used for arbitrary data

Both allow for computing the differences of additional property(s) in the identified pairs

Example: Property prices in Boston 1990s



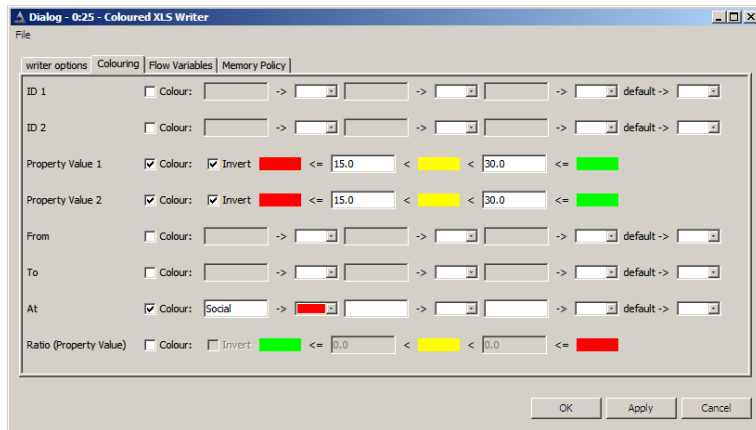
S At	D Ratio ...	SMI From	SMI To	S ID 1	S ID 2	D ...	D ..
. Social	1.729	Bin 3	Bin 2	Row138	Row132	13.3	23
. Social	1.654	Bin 4	Bin 2	Row30	Row29	12.7	21
. Social	1.643	Bin 4	Bin 2	Row140	Row132	14	23
. pupils/teacher	1.641	Bin 5	Bin 4	Row31	Row314	14.5	23.8

# Coloured XLS Writer

Conditional cell colours in generated xls file

nominal: cells containing a specific string

numeric: traffic light colouring according to user defined threshold



	A	B	C	D	E	F	G	H	I
1	ID 1	ID 2	Property V	Property V	From	To	At	Ratio (Prope	
2	Row138	Row132	13.3	23	Bin 3	Bin 2	Social	1.729323	
3	Row30	Row29	12.7	21	Bin 4	Bin 2	Social	1.653543	
4	Row140	Row132	14	23	Bin 4	Bin 2	Social	1.642857	
5	Row31	Row314	14.5	23.8	Bin 5	Bin 4	pupils/teac	1.641379	
6	Row33	Row29	13.1	21	Bin 3	Bin 2	Social	1.603053	
7	Row174	Row181	22.6	36.2	Bin 2	Bin 1	Property te	1.60177	
8	Row88	Row179	23.6	37.2	Bin 5	Bin 3	Age	1.576271	
9	Row30	Row21	12.7	19.6	Bin 4	Bin 2	Social	1.543307	
10	Row33	Row21	13.1	19.6	Bin 3	Bin 2	Social	1.496183	
11	Row227	Row228	31.6	46.7	Bin 4	Bin 1	Age	1.477848	
12	Row138	Row131	13.3	19.6	Bin 3	Bin 2	Social	1.473684	
13	Row30	Row28	12.7	18.4	Bin 4	Bin 2	Social	1.448819	
14	Row23	Row29	14.5	21	Bin 3	Bin 2	Social	1.448276	
15	Row138	Row130	13.3	19.2	Bin 3	Bin 2	Social	1.443609	
16	Row30	Row14	12.7	18.2	Bin 4	Bin 2	Social	1.433071	
17	Row127	Row132	16.2	23	Bin 3	Bin 2	Social	1.419753	
18	Row315	Row6	16.2	22.9	Bin 4	Bin 2	pupils/teac	1.41358	
19	Row33	Row28	13.1	18.4	Bin 3	Bin 2	Social	1.40458	
20	Row266	Row261	30.7	43.1	Bin 2	Bin 1	Social	1.403909	
21	Row140	Row131	14	19.6	Bin 4	Bin 2	Social	1.4	
22	Row57	Row256	31.6	44	Bin 4	Bin 3	city distan	1.392405	
23	Row263	Row261	31	43.1	Bin 2	Bin 1	Social	1.390323	
24	Row33	Row14	13.1	18.2	Bin 3	Bin 2	Social	1.389313	

Output of pivoted tables possible

# Some useful tricks

Often a plugin dependency substitutes an additional jar-file on the classpath

Refer to environment variables required for plugin activation (e.g. classpaths)

MANIFEST.MF

```
Bundle-ClassPath: bin/,
lib/lib_included_in_plugin.jar,
external:$VAR1$/external_lib1_at_location_VAR1.jar,
external:$VAR2$/external_lib2_at_location_VAR2.jar
```

## Install database driver upon plugin activation

Add to constructor of YourActivator

```
URL H2 = YourActivator.class.getResource("h2-1.3.151.jar"); //jar contain in SAME directory as class
URL MYSQL = YourActivator.class.getResource("mysql-connector-java-5.1.16-bin.jar");
try {
    URL H2URL = FileLocator.toFileURL(H2);
    File h2file = new java.io.File(H2URL.getPath());
    URL MYSQLURL = FileLocator.toFileURL(MYSQL);
    File mysqlfile = new java.io.File(MYSQLURL.getPath());
    IPreferenceStore store =
        org.knime.workbench.core.KNIMECorePlugin.getDefault().getPreferenceStore();
    store.setDefault(HeadlessPreferencesConstants.P_DATABASE_DRIVERS,
        h2file.getAbsolutePath()+";"+mysqlfile.getAbsolutePath()+";");
    DatabaseDriverLoader.loadDriver(h2file);
} catch (Exception e) {
    ...
}
```



# Acknowledgements

## Active contributors

Michael Bodkin, David Evans, David Thorner,  
Gary Sharman, George Papadatos, Hina Patel, Nikolas Fechner

## Former group members

Dimitar Hristozov, Fred Ludlow, Swanand Gore