



SQL Based Data Access

Bodo Bachmann

Empowering Business in Real Time
PI Infrastructure for the Enterprise

Agenda

❑ New Architecture

- Motivation
- PI SQL Data Access Server (PI SQL DAS)
- Roadmap

❑ PI JDBC

- PI SQL DAS 1.0
- Cross Platform Demo

❑ PI OLEDB 64bit (x64)

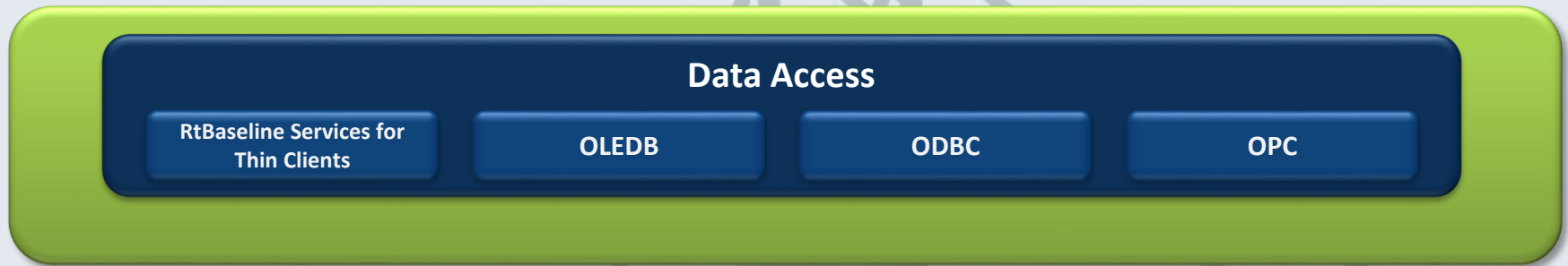
- Details

❑ PI System OLEDB Provider 1.0

- Functionality
- Demo
- Future (Tag based data, Event Frames,...)

Data Access - Overview

PI Data is available via common standards



- WebServices (i.e. RtBaseline Services) – Provide Data to Web Applications
- OLEDB – Data Access via SQL Queries
- ODBC – Data Access via SQL Queries
- OPC DA/HDA/UA – Data Access to the PI System via OPC Standard

Data Access

Via PI OLEDB

- PI OLEDB provider allows applications (OLE DB consumers) working with PI data through SQL queries:

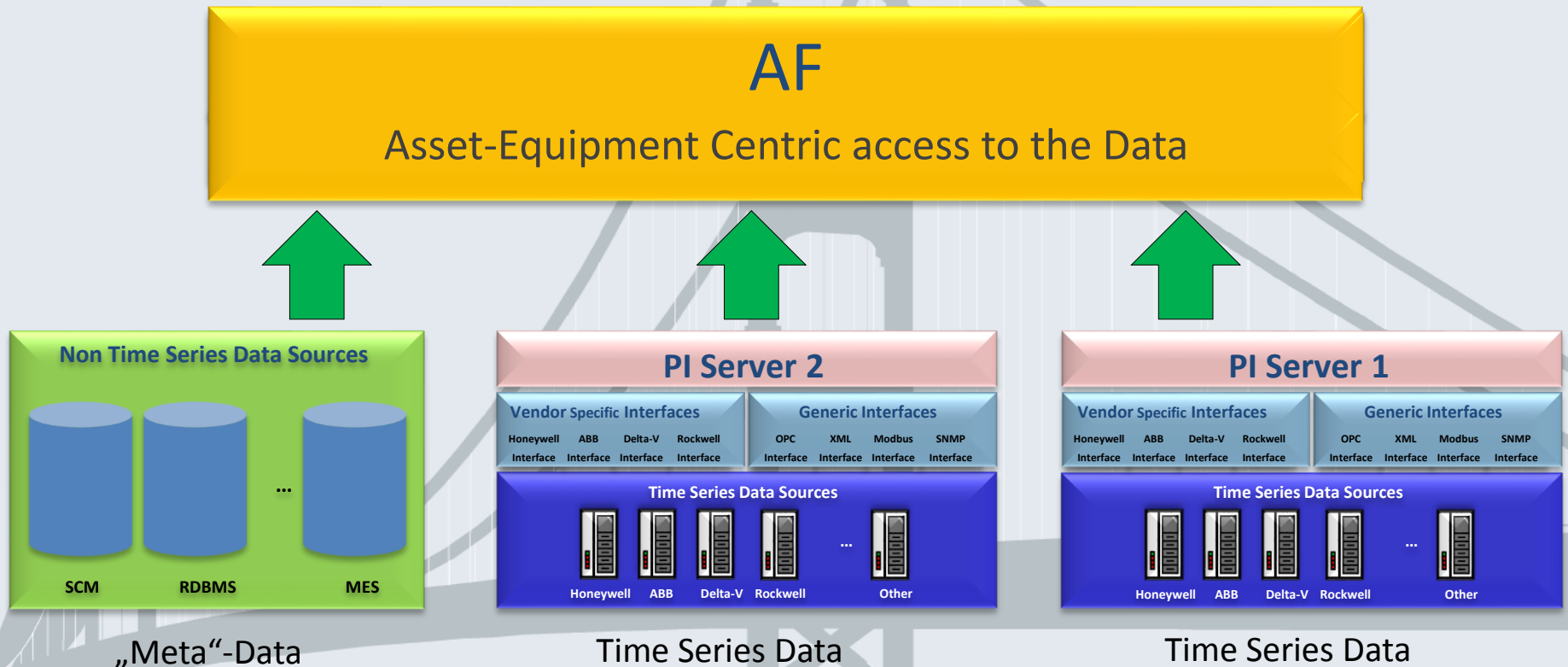


Next Generation – Motivation:

- renew PI OLEDB Architecture
- introduce JDBC and other data provider standards
- support „PI System“

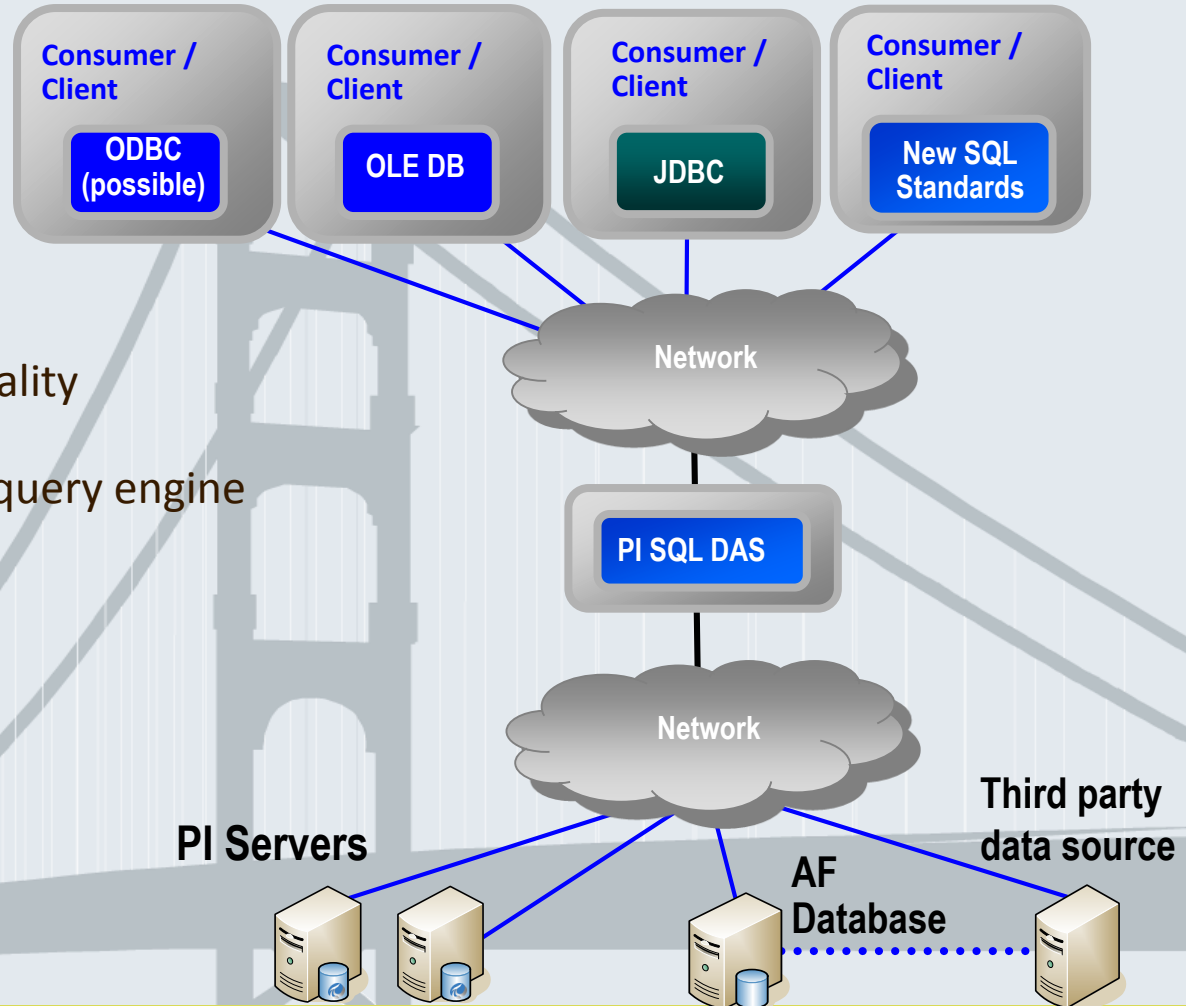
AF in the PI System

- Data structured and organized by Assets
- Spans multiple PI Systems
- Incorporates non time series Data



New Architecture

- Multi-standard and **multi-platform** architecture
One connection allows querying multiple data sources
- Standard implementation separated from SQL functionality
- Prepared for **heterogeneous** query engine



New Resource Management

❑ Paging

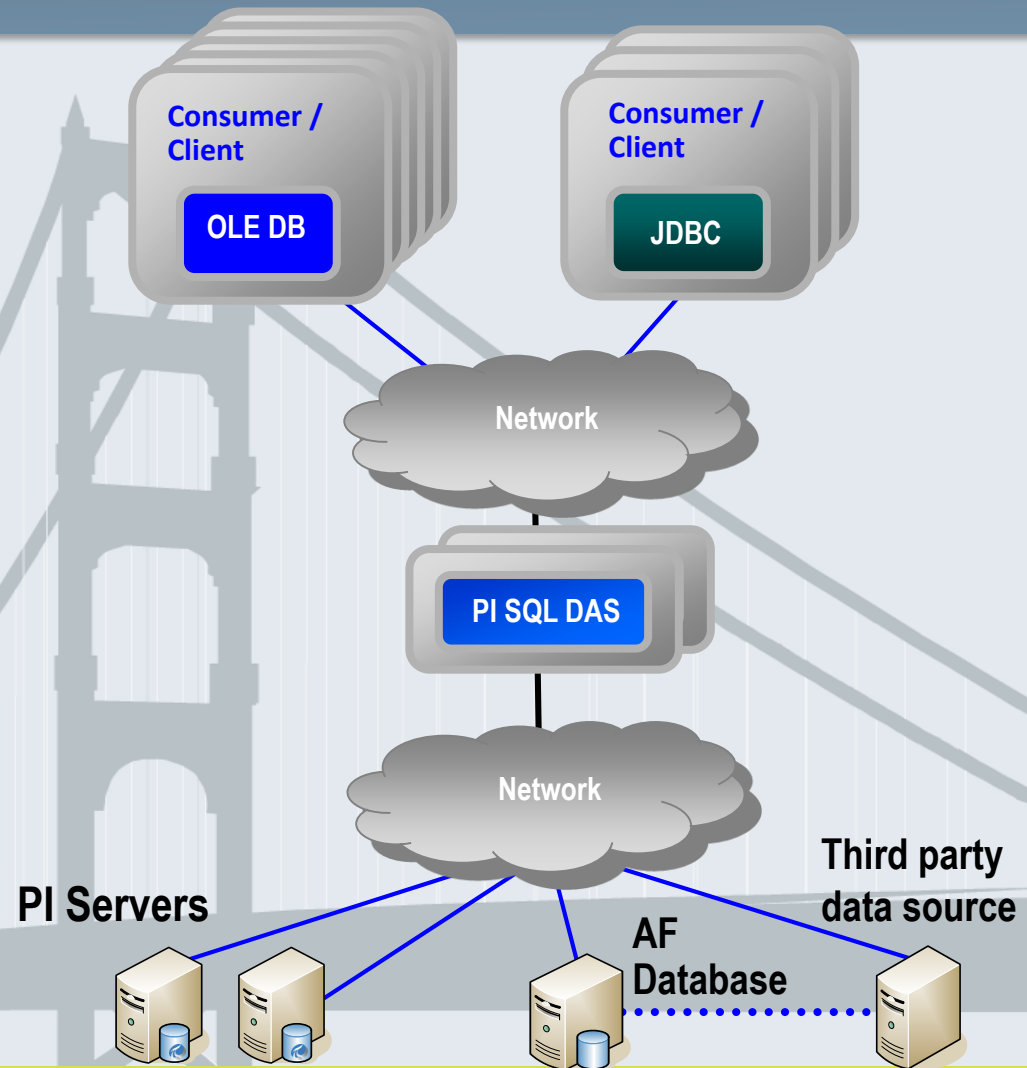
- Protects server from huge data requests
- Decreases memory requirements

❑ Caching

- Smart memory management

❑ PI SQL DAS Deployment options for example:

- 1x PI SQL DAS for Web Farm
- 1x PI SQL DAS for Reporting
- 1x PI SQL DAS for Clients

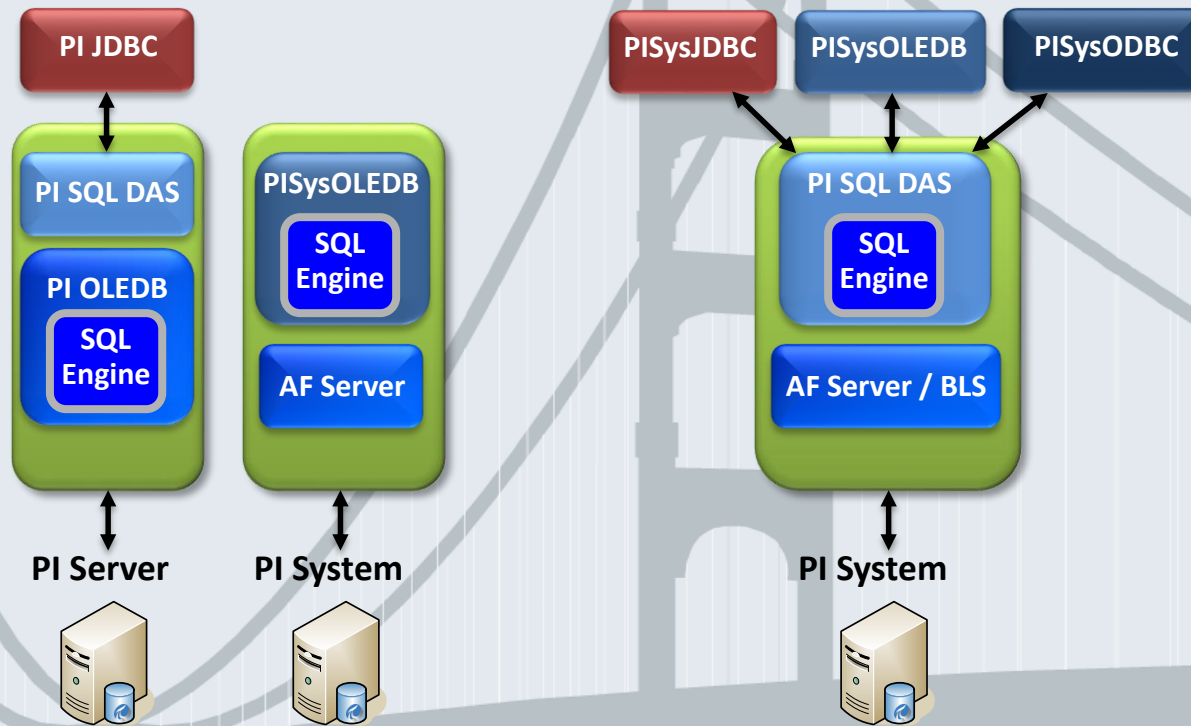


Roadmap

2008

Use of new Architecture Components

2011



PI JDBC

Architecture Details

Client Application

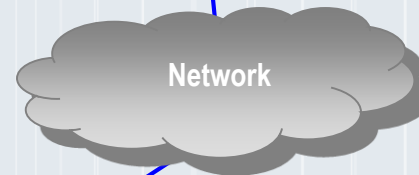
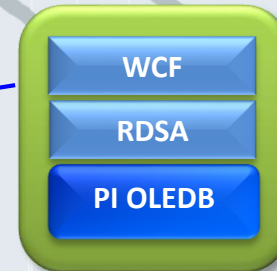


PI JDBC

- Windows + Linux
- requires PI OLEDB
- meant to support Java server apps



https



PI Servers



PI JDBC

□ PI JDBC Details

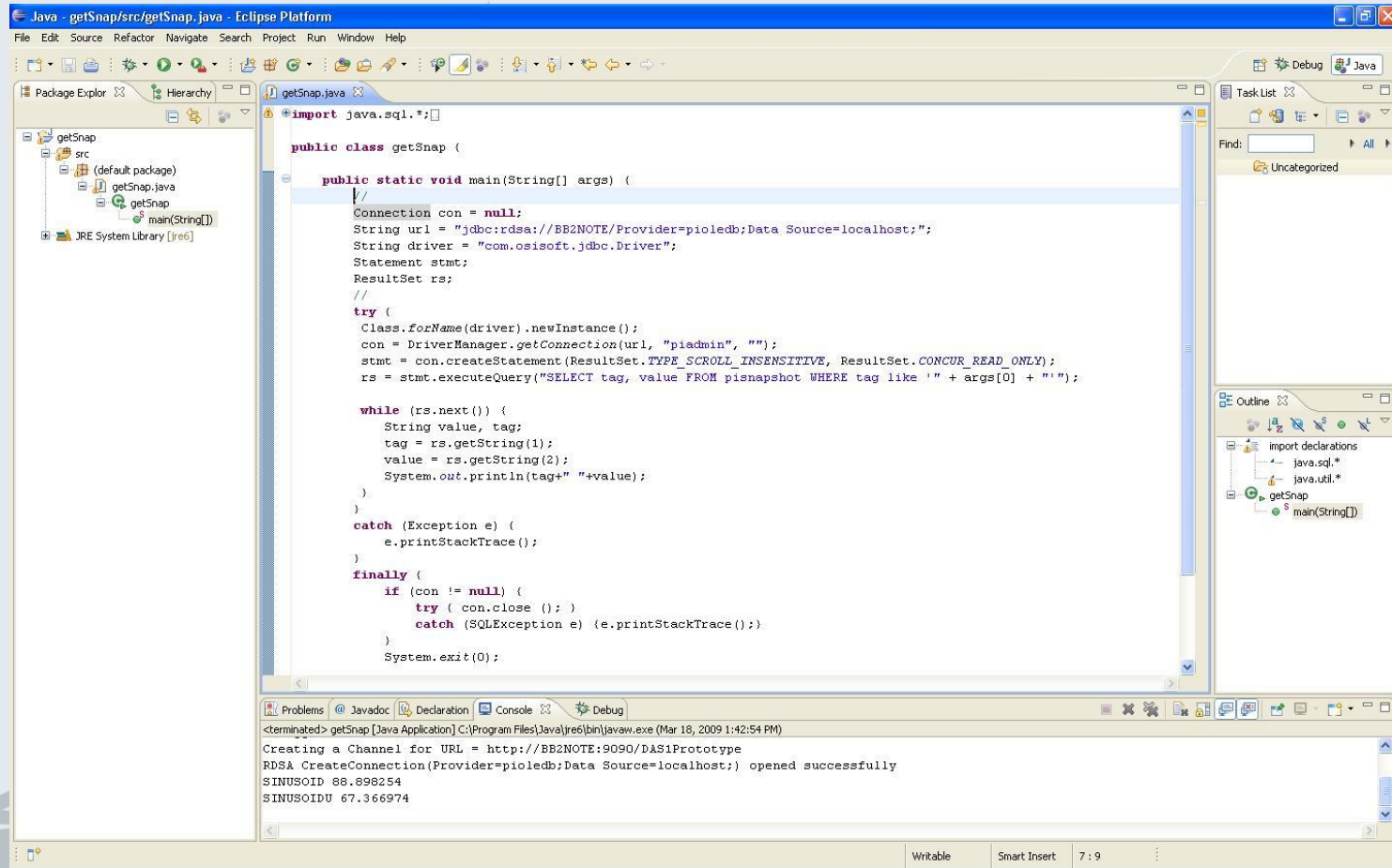
- bridge to PI OLEDB (Type 1 JDBC driver)
- based on JDBC 4.0 API (Java Platform SE 6)
- same table structure, same SQL functionality as PI OLEDB
- Multiplatform
 - » Will work on many Linux Distributions
 - » Testing concentrates on Windows, Novell Suse Linux Enterprise Server and Red Hat Enterprise Linux
- best option for JAVA based development

□ Limitations

- PI OLEDB is multithreaded but not designed as backend
- additional communication infrastructure overhead
- amount of supported OS limited because network component is OS specific

PI JDBC Demo

- **getSnap** JDBC application
- Compile in Windows version of **Eclipse**
- Run on **Windows and Linux** without additional work



The screenshot displays the Eclipse IDE interface. The main editor window shows the `getSnap.java` file with the following code:

```
import java.sql.*;

public class getSnap {

    public static void main(String[] args) {
        //
        Connection con = null;
        String url = "jdbc:rdsai://BB2NOTE/Provider=pioledb;Data Source=localhost;";
        String driver = "com.osisoft.jdbc.Driver";
        Statement stmt;
        ResultSet rs;
        //
        try {
            Class.forName(driver).newInstance();
            con = DriverManager.getConnection(url, "piadmin", "");
            stmt = con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);
            rs = stmt.executeQuery("SELECT tag, value FROM pisinapshot WHERE tag like '" + args[0] + "'");

            while (rs.next()) {
                String value, tag;
                tag = rs.getString(1);
                value = rs.getString(2);
                System.out.println(tag+" "+value);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (con != null) {
                try { con.close(); }
                catch (SQLException e) { e.printStackTrace(); }
            }
            System.exit(0);
        }
    }
}
```

The Package Explorer on the left shows the project structure with `getSnap` and `main(String[])`. The Outline view on the right shows the class structure. The Console window at the bottom displays the execution output:

```
<terminated> getSnap [Java Application] C:\Program Files\Java\jre6\bin\javaw.exe (Mar 18, 2009 1:42:54 PM)
Creating a Channel for URL = http://BB2NOTE:9090/DAS1Prototype
RDSA CreateConnection(Provider=pioledb;Data Source=localhost;) opened successfully
SINUSOID 88.898254
SINUSOIDU 67.366974
```

PI JDBC Demo

- **Linux (ubuntu)**
- **DBVisualizer**
allows to access and explore any jdbc Driver
- **PI JDBC (Bridge)**
talking to PI OLEDB

The screenshot shows the DBVisualizer Free 6.0.7 application window. The 'Connections' pane on the left lists the 'PI JDBC (current version)' connection. The 'SQL Commander' pane shows a query: `select * from piproductversion`. The 'Result Set [1]' pane displays a table with 7 columns: item, descriptor, version, major, minor, build, and revision. The table contains 3 rows of data. The 'Monitor' pane on the right shows a list of 26 rows of data, including timestamps and numerical values. The status bar at the bottom indicates 'Auto Reload Seconds: 20' and '00:00:18'.

item	descriptor	version	major	minor	build	revision
1	PI Server	3.4.375.38	3	4	375	38
2	PIOLEDB PI OLE DB Provider	3.2.2.7	3	2	2	7
3	PISDK PI Software Development Kit	1.3.5.343	1	3	5	343

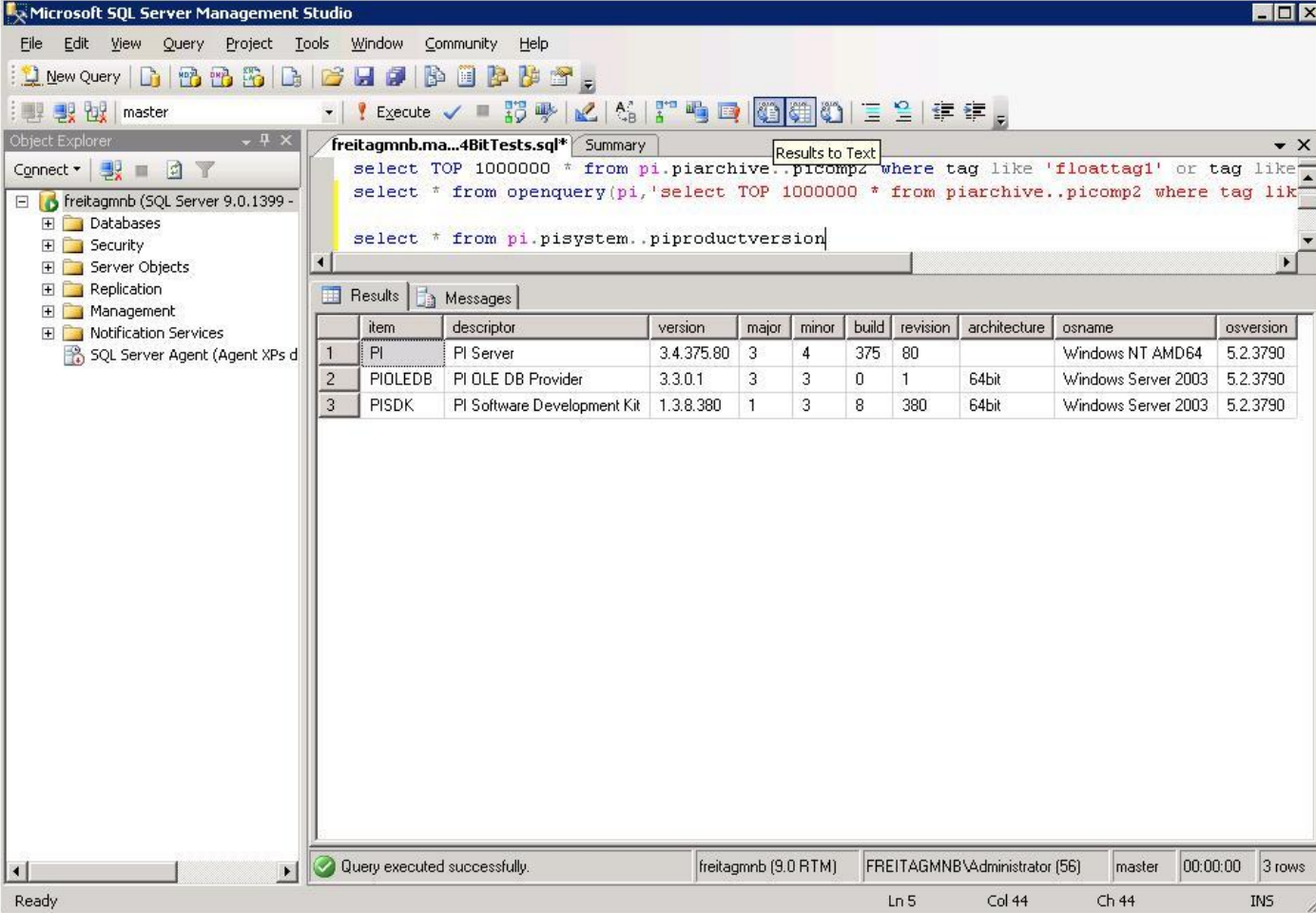
item	descriptor	version	major	minor	build	revision
13	2008-02-21 11:44:34.0	24.34698				
14	2008-02-21 12:22:34.0	85.43672				
15	2008-02-21 12:37:34.0	98.32598				
16	2008-02-21 12:55:04.0	96.945114				
17	2008-02-21 13:13:04.0	77.86411				
18	2008-02-21 13:53:04.0	13.953162				
19	2008-02-21 14:09:34.0	0.8968167				
20	2008-02-21 14:26:04.0	3.684022				
21	2008-02-21 14:44:34.0	24.346743				
22	2008-02-21 15:22:34.0	85.43652				
23	2008-02-21 15:37:34.0	98.325905				
24	2008-02-21 15:55:04.0	96.94521				
25	2008-02-21 16:13:04.0	77.86434				
26	2008-02-21 16:43:04.0	27.978298				

PI OLEDB 64bit

- ❑ can coexist with 32bit PI OLEDB version on 64bit Windows (x86-x64)
- ❑ transparent to applications if both versions installed
- ❑ required for 64bit SQL Server Linked Server
- ❑ requires 64bit PI SDK
- ❑ comes as seperate setup kit

PI OLEDB 64bit

- check version via architecture column in **piproductversion** table



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the server 'freitagmb (SQL Server 9.0.1399 -)'. The Query window on the right contains the following SQL query:

```
select TOP 1000000 * from pi.piarchive..picomp2 where tag like 'floattag1' or tag like  
select * from openquery(pi,'select TOP 1000000 * from piarchive..picomp2 where tag lik  
select * from pi.pisystem..piproductversion
```

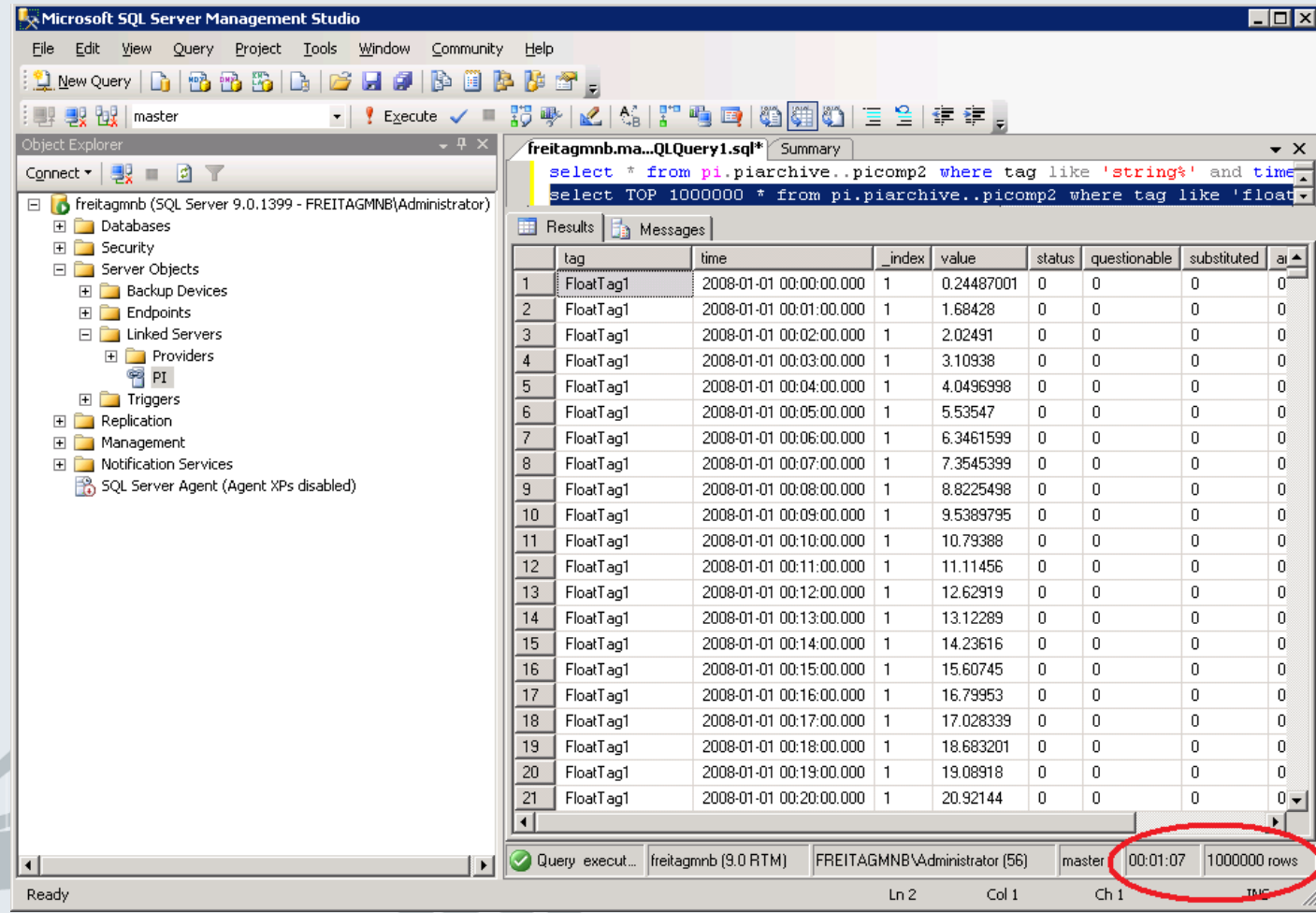
The Results pane shows the output of the query, which is a table with the following data:

	item	descriptor	version	major	minor	build	revision	architecture	osname	osversion
1	PI	PI Server	3.4.375.80	3	4	375	80		Windows NT AMD64	5.2.3790
2	PIOLEDB	PI OLE DB Provider	3.3.0.1	3	3	0	1	64bit	Windows Server 2003	5.2.3790
3	PISDK	PI Software Development Kit	1.3.8.380	1	3	8	380	64bit	Windows Server 2003	5.2.3790

The status bar at the bottom indicates 'Query executed successfully.' and 'freitagmb (9.0 RTM) FREITAGMNB\Administrator (56) master 00:00:00 3 rows'.

PI OLEDB 64bit

- Increased performance for large resultsets



The screenshot displays the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the server structure for 'freitagmnb (SQL Server 9.0.1399 - FREITAGMNB\Administrator)', including Databases, Security, Server Objects, and Providers. The 'PI' provider is highlighted under the Providers folder. The Query window on the right shows a query titled 'freitagmnb.ma...QLQuery1.sql*'. The query is a SELECT statement filtering data from 'pi.piarchive..picomp2' based on the 'tag' column, specifically selecting rows where 'tag' is like 'string%' and 'time' is like 'float%'. The Results tab shows a table with 21 rows of data. The table has columns: tag, time, _index, value, status, questionable, substituted, and ai. The data shows a sequence of 'FloatTag1' entries with timestamps from 2008-01-01 00:00:00.000 to 2008-01-01 00:20:00.000. The status column is consistently 0, and the questionable and substituted columns are also 0. The 'ai' column is consistently 0. The bottom status bar indicates 'Query execut...' and 'freitagmnb (9.0 RTM)'. The 'FREITAGMNB\Administrator (56)' session is shown, and the 'master' database is selected. The execution time is 00:01:07, and the number of rows returned is 1000000, which is circled in red.

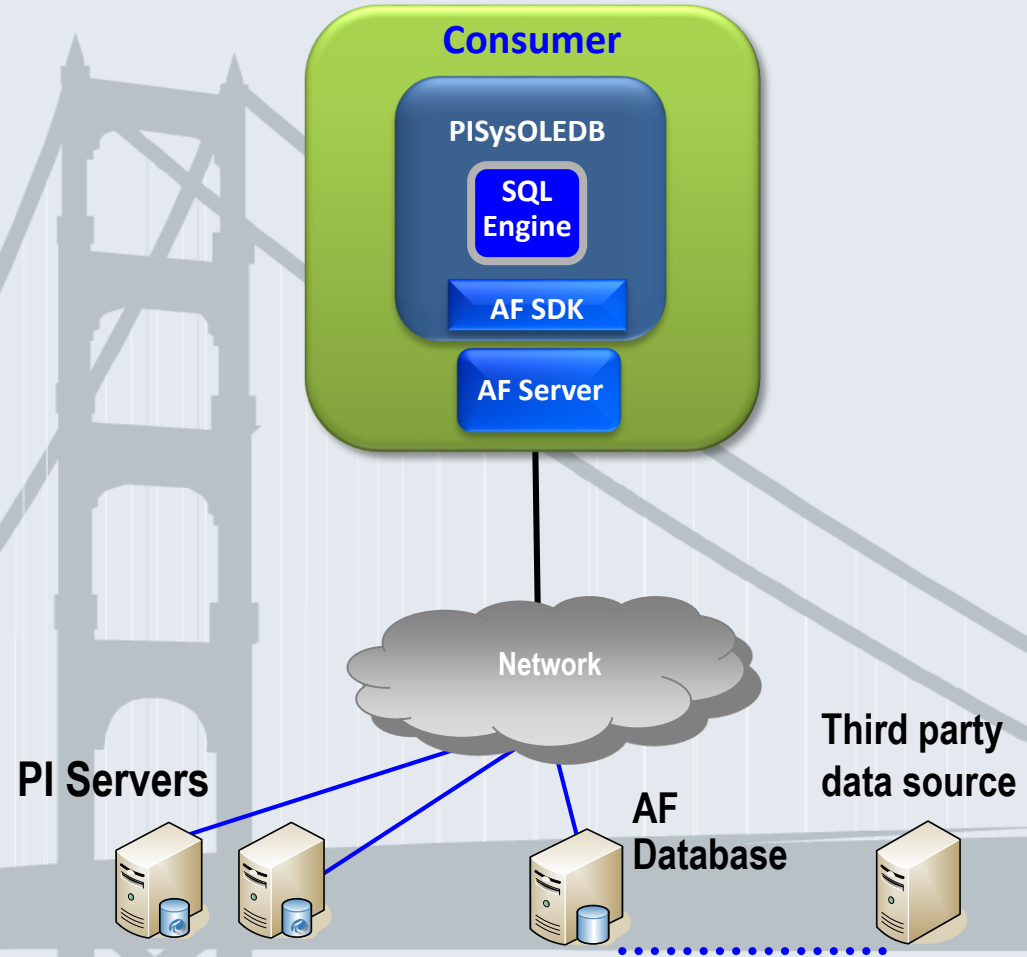
	tag	time	_index	value	status	questionable	substituted	ai
1	FloatTag1	2008-01-01 00:00:00.000	1	0.24487001	0	0	0	0
2	FloatTag1	2008-01-01 00:01:00.000	1	1.68428	0	0	0	0
3	FloatTag1	2008-01-01 00:02:00.000	1	2.02491	0	0	0	0
4	FloatTag1	2008-01-01 00:03:00.000	1	3.10938	0	0	0	0
5	FloatTag1	2008-01-01 00:04:00.000	1	4.0496998	0	0	0	0
6	FloatTag1	2008-01-01 00:05:00.000	1	5.53547	0	0	0	0
7	FloatTag1	2008-01-01 00:06:00.000	1	6.3461599	0	0	0	0
8	FloatTag1	2008-01-01 00:07:00.000	1	7.3545399	0	0	0	0
9	FloatTag1	2008-01-01 00:08:00.000	1	8.8225498	0	0	0	0
10	FloatTag1	2008-01-01 00:09:00.000	1	9.5389795	0	0	0	0
11	FloatTag1	2008-01-01 00:10:00.000	1	10.79388	0	0	0	0
12	FloatTag1	2008-01-01 00:11:00.000	1	11.11456	0	0	0	0
13	FloatTag1	2008-01-01 00:12:00.000	1	12.62919	0	0	0	0
14	FloatTag1	2008-01-01 00:13:00.000	1	13.12289	0	0	0	0
15	FloatTag1	2008-01-01 00:14:00.000	1	14.23616	0	0	0	0
16	FloatTag1	2008-01-01 00:15:00.000	1	15.60745	0	0	0	0
17	FloatTag1	2008-01-01 00:16:00.000	1	16.79953	0	0	0	0
18	FloatTag1	2008-01-01 00:17:00.000	1	17.028339	0	0	0	0
19	FloatTag1	2008-01-01 00:18:00.000	1	18.683201	0	0	0	0
20	FloatTag1	2008-01-01 00:19:00.000	1	19.08918	0	0	0	0
21	FloatTag1	2008-01-01 00:20:00.000	1	20.92144	0	0	0	0

PI System OLEDB v1

Architecture Details

PISysOLEDB version 1

- Windows based
- Linked to AF SDK (bypass object model)
- Access via AF Server
- Read-only



PI System OLEDB v1 Demo

The screenshot displays the PI System OLEDB Client application. On the left, the 'Server Explorer' shows a tree view of the database structure, including 'Servers', 'Catalogs', 'Configuration', 'PISystem', 'Test', and 'Asset'. The 'Asset' folder is expanded, showing 'Tables' and 'Views'. The 'Tables' folder is further expanded, listing various tables like 'CurElement', 'CurElementAttribute', 'CurElementHierarchy', etc.

The main window, titled 'PISysOleDbQueryWindow 5', contains a SQL query:

```
SELECT e.Path+e.Name Element, a.Path+a.Name Attribute, r.Time, r.Value FROM Test..ElementHierarchy e
INNER JOIN Test.Asset.ElementAttribute a
ON a.ElementVersionID = e.VersionID
INNER JOIN Test.Data.Recorded r
ON r.ElementAttributeID = a.ID
WHERE r.time BETWEEN '06-Oct-2008' AND '06-Oct-2008 12:00:00'
OPTION (FORCE ORDER)
```

Below the query, the 'Result' tab shows a table with 17 rows of data. The columns are 'Element', 'Attribute', 'Time', and 'Value'.

	Element	Attribute	Time	Value
01	\E_1	\A_1	10/6/2008	49.14205
02	\E_1	\A_1	10/6/2008 1:31 AM	85.66759
03	\E_1	\A_1	10/6/2008 2:40 AM	99.24164
04	\E_1	\A_1	10/6/2008 3:45 AM	96.19122
05	\E_1	\A_1	10/6/2008 4:58 AM	75.74577
06	\E_1	\A_1	10/6/2008 7:31 AM	14.17995
07	\E_1	\A_1	10/6/2008 7:42 AM	11.00134
08	\E_1	\A_1	10/6/2008 7:42 AM	System.__ComObject
09	\E_1	\A_1	10/6/2008 7:46 AM	9.941768
10	\E_1	\A_1	10/6/2008 8:51 AM	0.1380453
11	\E_1	\A_1	10/6/2008 9:57 AM	6.160055
12	\E_1	\A_1	10/6/2008 11:16 AM	31.46513
13	\E_1	\A_1	10/6/2008 12:00 PM	49.02984
14	\ET_3_1	\AT_1	10/6/2008	91.82214
15	\ET_3_1	\AT_1	10/6/2008 12:31 AM	98.46332
16	\ET_3_1	\AT_1	10/6/2008 1:31 AM	98.12081
17	\ET_3_1	\AT_1	10/6/2008 2:41 AM	81.79839

The status bar at the bottom indicates 'Query executed successfully' and shows '25 rows'.

Summary

❑ Upcoming releases

- PI JDBC
- PI OLEDB 64bit
 - » Release schedule ~ Q2/2009
- PISysOLEDB v1
 - » Beta scheduled ~ Q2/2009
 - » Release schedule ~ Q4/2009

❑ PI SQL DAS v2 based products in development

- PISysOLEDB v2
- PISysJDBC