

DataDirect XQuery™ FAQs

- What language can I use to query XML? 2
- What is XQuery? 2
- What is the current status of XQuery? 2
- What is XQJ?..... 3
- Why do I need XQJ? 3
- Why would I use XQuery to query relational data? Why would I use XQuery for XML reporting and Web publishing? 3
- Why would I use XQuery in a Java application? 3
- Why would I use XQuery for Web Services? 4
- How can I query both XML sources and relational databases with a single query? 4
- What is DataDirect XQuery™ ?..... 6
- What benefits does DataDirect XQuery™ provide over other XQuery products? 7
- What is the relationship between DataDirect XQuery™ and Saxon? 7
- Why can't I just use a free XQuery implementation, such as Saxon, instead of DataDirect XQuery™? 7
- How is DataDirect XQuery™ different from the XQuery implementations provided by the big database vendors? 8
- Can I embed DataDirect XQuery™ into a commercial application? 8
- Which platforms does DataDirect XQuery™ run on?..... 8
- Which relational databases does DataDirect XQuery™ support? 8
- What types of XML does DataDirect XQuery™ support access to? 9
- Which version of the XQuery specification does DataDirect XQuery™ support?..... 9
- Can I use DataDirect XQuery™ to update data?..... 10
- How easy is DataDirect XQuery™ to configure?..... 10
- How do I develop a Java application that executes an XQuery using DataDirect XQuery™? 10
- Where can I learn more about XQuery and XQJ? 11
- Where can I learn more about DataDirect XQuery™? 11

What language can I use to query XML?

You can use XQuery, an XML query language, to query XML data sources. In addition, you can use XQuery to query any type of data that can be virtually represented as XML.

What is XQuery?

XQuery is a query language that was designed as a native XML query language. Because most types of data can be represented as XML, XQuery can also be used to query other types of data. For example, XQuery can be used to query relational data using an XML view of a relational database. This is important because many Internet applications need to integrate information from multiple sources, including data found in web messages, relational data, and various XML sources. XQuery was specifically designed for this kind of data integration.

For example, suppose your company is a financial institution that needs to produce reports of stock holdings for each client. A client requests a report with a Simple Object Access Protocol (SOAP) message, which is represented in XML. In most businesses, the stock holdings data is stored in multiple relational databases, such as Oracle, Microsoft SQL Server, or DB2. XQuery can query both the SOAP message and the relational databases, creating a report in XML.

XQuery is based on the structure of XML and leverages that structure to make it possible to perform queries on any type of data that can be represented as XML, including relational data. In addition, XQuery API for Java (XQJ) lets your queries run in any environment that supports the J2EE platform.

What is the current status of XQuery?

XQuery is currently under development at the W3C, which is a standards body for the World Wide Web. The XQuery specification is currently a W3C Candidate Recommendation. The W3C maintains a home page for XQuery, including pointers to the XQuery specifications, tutorials, and a variety of products, at <http://www.w3.org/XML/Query.html>.

What is XQJ?

The XQuery API for Java (XQJ) is designed to support the XQuery language, just like the JDBC API supports the SQL query language. XQJ is based on the XQuery Data Model rather than the relational model. It allows a Java application to submit XQuery queries to data sources and process the results.

The XQJ standard (JSR 225) is being developed under the Java Community Process. For more information, see <http://www.jcp.org/en/jsr/detail?id=225>.

Why do I need XQJ?

You need the XQuery API for Java (XQJ) to allow a Java application to submit XQuery queries to XML and relational data sources and process the results. XQJ allows a Java program to configure connections, issue XQuery queries, and obtain results. XQJ is to XQuery what JDBC is to SQL.

The XQJ standard (JSR 225) is being developed under the Java Community Process. For more information, see <http://www.jcp.org/en/jsr/detail?id=225>.

Why would I use XQuery to query relational data?

Why would I use XQuery for XML reporting and Web publishing?

Many applications need to process relational data to create complex reports, and they need these reports in XML. In addition, most commercial web pages use relational data, but it rarely looks like relational data when it is presented to the user. These sites combine information from many sources, creating complex web pages that are user friendly. For relational data, XQuery can be thought of as an XML-based report writing language. Similarly, SOAP messages are hierarchical XML, but the data found in them frequently comes from relational databases, and XML is also used in many print-based database publishing applications.

Why would I use XQuery in a Java application?

The benefits of Java are well-known. It lets you write applications that work on any operating system, and the J2EE platform provides robust support for XML, security, web services, and database connectivity. Embedding XQuery in Java applications can significantly reduce the amount of Java code that is needed, because XQuery often is easier to use and more efficient than Java for processing XML or querying relational data to create XML.

The XQuery for Java API (XQJ) is being developed to allow Java applications to use XQuery much like JDBC allows Java applications to use SQL. Using DataDirect XQuery, a Java application that embeds or references an XQuery query can run on most operating systems, can use data from most relational databases, and can access XML through the standard Java XML APIs (SAX and DOM, for example).

Why would I use XQuery for Web Services?

Web Services depend heavily on XML for both its definition language, Web Services Description Language (WSDL), and its messaging protocol, Simple Object Access Protocol (SOAP). XQuery provides a language that can interact with XML data and other types of data as long as they can be represented as XML. For example, XQuery can be used to access the content of messages or to construct new messages to be passed to a Web service. In addition, you often need data to process Web Services results. XQuery can process SOAP messages and create the XML needed for result messages. See

http://blogs.datadirect.com/jonathan_robie/2006/05/calling_an_amazon_web_service.html for more information.

How can I query both XML sources and relational databases with a single query?

The best way to explain how a single XQuery query can query both XML and relational data sources is to show an example. Suppose your company is a financial institution that needs to produce reports of stock holdings for each client. A client requests a report with a Simple Object Access Protocol (SOAP) message, which is represented in XML. In most businesses, the stock holdings data is stored in multiple relational databases, such as Oracle, Microsoft SQL Server, or DB2. XQuery can query both the SOAP message and the relational databases, creating a report in XML. The XQuery query is processed using an XQuery implementation, such as DataDirect XQuery.

The following example joins an XML document (our SOAP message) named "request.xml" to two relational database tables named "holdings" and "statistical". The request.xml XML document is joined to the "holdings" table by the userId attribute in the XML file and the userId column of the "holdings" table. The two tables are joined by the ticker column of the "statistical" table and the stockticker column of the "holdings" table.

```
let $request := doc('request.xml')/request
for $user in $request/performance/UserId
return
  <portfolio UserId="{ $user }">
    { $request }
```

```

{
  for $st in collection('holdings')/holdings,
    $stats in collection('statistical')/statistical
  where $st/userid = $user
    and $stats/ticker = $st/stockticker
  return
    <stock>
      { $stats/companyname }
      { $st/stockticker }
      { $st/shares }
      { $stats/annualrevenues }
    </stock>
}
</portfolio>

```

Result

The result of this query is an element named portfolio. The first child of this element contains the original request from request.xml. After that, the query provides the stock information for a given user, taken from two tables.

```

<portfolio UserId="Jonathan">
  <request>
    <performance>
      <UserId>Jonathan</UserId>
      <start>2003-01-01</start>
      <end>2004-06-01</end>
    </performance>
  </request>
  <stock>
    <companyname>Amazon.com, Inc.</companyname>
    <stockticker>AMZN</stockticker>
    <shares>3000</shares>
    <annualrevenues>7780</annualrevenues>
  </stock>
  <stock>
    <companyname>eBay Inc.</companyname>
    <stockticker>EBAY</stockticker>
    <shares>4000</shares>
    <annualrevenues>22600</annualrevenues>
  </stock>
  <stock>
    <companyname>Int'l Business Machines C</companyname>
    <stockticker>IBM</stockticker>
    <shares>2500</shares>
    <annualrevenues>128200</annualrevenues>
  </stock>
  <stock>
    <companyname>Progress Software</companyname>
    <stockticker>PRGS</stockticker>
    <shares>23</shares>
    <annualrevenues>493.4</annualrevenues>
  </stock>
</portfolio>

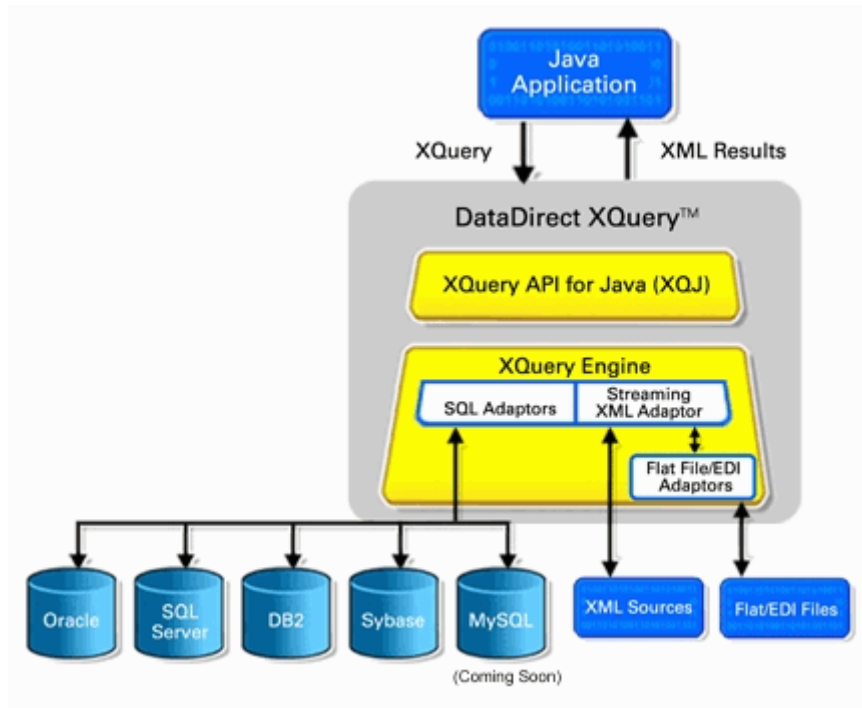
```

What is DataDirect XQuery™?

DataDirect XQuery is the first embeddable component for XQuery that implements the XQuery API for Java (XQJ). It supports all major relational databases on any Java platform. DataDirect XQuery allows you to query XML, relational databases, or a combination of the two, integrating the results for XML-based data exchange, XML-driven Web sites, and other applications that require or leverage the power of XML.

DataDirect XQuery is designed for software developers and independent software vendors (ISVs) who need to manage heterogeneous data sources in XML applications.

The following architectural diagram shows the components of DataDirect XQuery and the flow involved in accessing both XML and relational data sources.



What benefits does DataDirect XQuery™ provide over other XQuery products?

DataDirect XQuery is ideal for any company that needs to access and integrate data stored in XML and relational sources. Unlike other XQuery products, DataDirect XQuery provides the following benefits:

- DataDirect XQuery is database and platform independent.
- DataDirect XQuery provides good performance when querying relational databases because it allows the XQuery query to be executed as efficiently as possible (decomposing the query into SQL statements) and moving out of the database just the minimum amount of data. Other XQuery products return large chunks of data and then perform additional steps to retrieve the required data, which slows processing.
- DataDirect XQuery supports XML streaming to drastically reduce the amount of memory required when processing large XML documents. Other XQuery products do not support XML streaming.
- DataDirect XQuery does not require its own server infrastructure, which makes it easy to embed into other commercial applications.

What is the relationship between DataDirect XQuery™ and Saxon?

DataDirect XQuery embeds Saxon. It is used by DataDirect XQuery to query XML documents.

Why can't I just use a free XQuery implementation, such as Saxon, instead of DataDirect XQuery™?

If you want to query and integrate both XML and relational data, DataDirect XQuery provides substantial benefits in performance over a free implementation such as Saxon:

- DataDirect XQuery provides support for all major databases on any platform. In contrast, Saxon only has minimal support for data stored in relational databases.
- DataDirect XQuery provides good performance when querying relational databases because it allows the query to be executed as efficiently as possible (decomposing the query into SQL statements) and moving out of the database just the minimum amount of data. Other XQuery products return large chunks of data and then perform additional steps to retrieve the required data, which slows processing.
- DataDirect XQuery supports XML streaming to drastically reduce the amount of memory required when processing large XML documents. Saxon and many other XQuery products do not support XML streaming.

- DataDirect Technologies provides comprehensive, award-winning technical support. Technical support is not usually available with free implementations.

How is DataDirect XQuery™ different from the XQuery implementations provided by the big database vendors?

Microsoft and Oracle support XQuery in their products. Unlike the XQuery implementations from database vendors, DataDirect XQuery is database and platform independent. Using DataDirect XQuery, you can write applications that work for all major databases as well as XML data sources.

Can I embed DataDirect XQuery™ into a commercial application?

Yes. DataDirect XQuery does not require its own server infrastructure, which makes it easy to embed into other commercial applications.

Which platforms does DataDirect XQuery™ run on?

DataDirect XQuery is implemented in Java and runs on any J2SE 1.4 or higher Java platform.

Which relational databases does DataDirect XQuery™ support?

DataDirect XQuery supports the following databases:

- Oracle 10g (R1 and R2) and Oracle 9i (R1 and R2)
- DB2 Universal Database (UDB) v8.1 and v8.2 for Linux/UNIX/Windows
- DB2 UDB v8.1 for z/OS
- DB2 UDB V5R2 and V5R3 for iSeries
- Microsoft SQL Server 2005 and Microsoft SQL Server 2000 (including SP1, SP2, SP3a, and SP4)
- Sybase Adaptive Server Enterprise 15 and 12.5.x

In addition, support for MySQL 5.0 is planned.

What types of XML does DataDirect XQuery™ support access to?

DataDirect XQuery can access XML data sources that have the following physical formats:

- XML text files/streams. These files/streams can be accessed using `fn:doc()`, which supports the `http:`, `ftp:`, and `file:` URI schemes and Stylus Studio XML Deployment Adapter URI schemes.

For example:

```
let $request := doc('file:///c:/request.xml')/request
...
```

Here is an example of a Stylus Studio XML Deployment Adapter URI scheme in which the name of the Adapter is Base64, the properties set for the conversion are newline and encoding, and the file to convert is `base_to_xml.bin`:

```
let $request := doc('adapter:Base64:newline=crlf:
encoding=utf-8?file//w:/myfiles/base_to_xml.bin')/
request
...
```

See <http://www.stylusstudio.com/deployment/datadirectxquery> for more information about the Stylus Studio XML deployment adapters.

- XML contained in a Java application such as DOM trees. This type of XML can be bound to external variables in XQJ and used in XQuery queries.
- XML stored in columns of any supported relational database using an XML data type.
- XML stored in character columns of any supported relational database.

Which version of the XQuery specification does DataDirect XQuery™ support?

DataDirect XQuery supports the W3C Candidate Recommendation 3 November 2005 of the XQuery specification located at:

<http://www.w3.org/TR/2005/CR-xquery-20051103>.

Can I use DataDirect XQuery™ to update data?

XQuery updates are still in the planning stage. Using a combination of DataDirect XQuery and JDBC offers a simple solution to working with both XML and relational data, and providing updates to relational data. See http://www.datadirect.com/developer/xquery/xquery_updates/index.ssp for more information.

How easy is DataDirect XQuery™ to configure?

Configuring DataDirect XQuery for your environment is straightforward. Once installed, you configure a source configuration file to specify the connection information for XML and relational data sources that are used by XQuery queries. To make this configuration easy, DataDirect XQuery ships with an example source configuration file for each database that the product supports, so you can start with one of these example files and modify the values as needed.

How do I develop a Java application that executes an XQuery using DataDirect XQuery™?

Java applications use the XQuery for Java API (XQJ) to execute XQuery, which is analogous to using the JDBC API to execute SQL statements. The following sample Java code illustrates the basic steps that an application would perform to execute an XQuery expression using DataDirect XQuery.

```
// import the XQJ classes
import com.ddtek.xquery.*;
import com.ddtek.xquery.xqj.mediator.DDXQDataSource;

// establish a connection to a data source
DDXQDataSource ds = new DDXQDataSource();
ds.setJdbcUrl("jdbc:xquery:sqlserver://server1:1433;
  databaseName=stocks");
XQConnection conn = ds.getConnection("myuserid", "mypasswd");

// create an expression object that is used to execute a query
XQExpression expr = conn.createExpression();

// the query
String es = "for $h in collection('holdings')/holdings " +
  "where $h/stockticker='AMZN' " +
  "return $h";

// execute the query
XQResultSequence result = expr.executeQuery(es);
System.out.println(result.getSequenceAsString());
```

```
// free all resources
result.close();
expr.close();
conn.close();
```

Additional XQJ examples covering different scenarios are shipped with DataDirect XQuery.

Where can I learn more about XQuery and XQJ?

- The W3C maintains a home page for XQuery, including pointers to the XQuery specifications, tutorials, and a variety of products, at <http://www.w3.org/XML/Query.html>.
- The XQJ standard (JSR 225) is being developed under the Java Community Process. For more information, see <http://www.jcp.org/en/jsr/detail?id=225>.
- Information about XQuery and DataDirect XQuery can be found at <http://www.datadirect.com/products/xquery/index.ssp>.
- Learn the basics of XQuery. See the XQuery tutorial at http://www.datadirect.com/download/docs/ddxquery/tutorial_query.html.
- Learn the basics of using XQuery in a Java program. See the XQJ tutorial at http://www.datadirect.com/developer/xquery/topics/xqj_tutorial/index.ssp.
- You'll find valuable developer advice and programming resources in Jonathan Robie's XQuery blog at http://blogs.datadirect.com/jonathan_robie/. Jonathan Robie is an editor of several of the XQuery specifications and a well-known speaker at XML conferences.

Where can I learn more about DataDirect XQuery™?

Information about DataDirect XQuery can be found at <http://www.datadirect.com/products/xquery/index.ssp>.

We welcome your feedback! Please send any comments concerning documentation, including suggestions for other topics that you would like to see, to:

docgroup@datadirect.com

FOR MORE INFORMATION

800-876-3101

Worldwide Sales

Belgium (French).....0800 12 045
Belgium (Dutch).....0800 12 046
France.....0800 911 454
Germany0800 181 78 76
Japan0120.20.9613
Netherlands.....0800 022 0524
United Kingdom.....0800 169 19 07
United States.....800 876 3101

Copyright © 2006 DataDirect Technologies Corp. All rights reserved. DataDirect Connect is a registered trademark of DataDirect Technologies Corp. in the United States and other countries. DataDirect XQuery is a trademark of DataDirect Technologies Corp. in the U.S. and other countries. Java and all Java based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. Other company or product names mentioned herein may be trademarks or registered trademarks of their respective companies.



DataDirect Technologies is the software industry's only comprehensive provider of software for connecting the world's most critical business applications to data and services, running on any platform, using proven and emerging standards. Developers worldwide depend on DataDirect® products to connect their applications to an unparalleled range of data sources using standards-based interfaces such as ODBC, JDBC™ and ADO.NET, XQuery and SOAP. More than 300 leading independent software vendors and thousands of enterprises rely on DataDirect Technologies to simplify and streamline data connectivity for distributed systems and to reduce the complexity of mainframe integration. DataDirect Technologies is an operating company of Progress Software Corporation (Nasdaq: PRGS). For more information, visit www.datadirect.com.