Software review

Federating data with Information Integrator

Abstract
Information Integrator is an extension to IBM’s relational database DB2, which uses data federation to provide benefits to molecular biology researchers through two unique capabilities: increased flexibility in combining data from disparate sources, and SQL access to non-SQL data, easing the task of automating data analysis.

INTRODUCTION
There is a vast amount of molecular biology data available from a large number of different sites using primarily browsing and searching mechanisms implemented via web interfaces. Answering a particular biological question often requires identifying data distributed among a number of sites and collating information from a variety of formats. Often one uses a series of steps that depend on each other, where one piece of information leads to later pieces of information, rather than gathering all the required information at once.

Types of data that can be accessed
Information Integrator can give you access to data already in relational databases, such as Oracle, SQL Server, Sybase and MySQL. This would primarily refer to databases of locally collected information, but at least one large public database, dbSNP, has agreed to provide its database as a SQL server backup file that can be downloaded and instantiated locally.

Information Integrator can also give you access to non-relational data sources such as SRS, parts of Entrez (PubMed and Nucleotide are currently accessible and more of the NCBI data divisions are promised for the future), some useful programs (BLAST and HMMer) and some other file formats (Documentum, Excel and flat files).

Benefits of data federation
Something of a holy grail of the molecular biology world is the creation of one vast database that integrates all available information. This would be analogous to the creation of a data warehouse within a large organisation, though of greater size and complexity than even the largest organisations use. A data warehouse of molecular biology would enable biologists to find all the information they need from one place, using a small number of queries.

Unfortunately, building data warehouses comes at a huge price in terms of defining how all of the data will be integrated, storing all of the information in one central location, and providing robust procedures for matching the data between the collection points and the warehouse. Data warehouses for molecular biology will be impossible until the field reaches a point where there is general agreement on what types of information are interesting and how they relate to each other. Even assuming such an agreement is possible, it is certainly a significant distance in the future. IBM has positioned ‘data federation’ (see below) as an alternative to data...
warehousing in fields such as molecular biology where the types and uses of data change quickly and dramatically. The use of data federation provides the ability to access lots of databases without going through the costly step of physically integrating them. There is a downside to using federation. Sidestepping the costs of integration pushes the task of integration down to the application that is accessing the data and forces multiple applications to potentially perform the same integration work.

The task of integrating data at a single application level, though, is much less complex than having to integrate across all applications at once, which is what a data warehouse aspires to do. The result of using data federation is a trade-off of reusability (each application does its own integration) vs. a high degree of flexibility in terms of being able to grab any set of data that becomes interesting and changing what data items are joined together as the rapidly developing field changes.

Besides increased flexibility, the data federation available through Information Integrator provides an SQL–interface on top of some of the most widely used software tools in molecular biology. The ability to use the Information Integrator wrappers to get at BLAST databanks, HMMer results and data from PubMed and GenBank not only saves every programmer from having to write their own interfaces for querying and parsing these data sources, but gives these programmers a standard interface for making these queries and joining the results to other data.

**ARCHITECTURAL OVERVIEW**

Information Integrator provides DB2 with the capability of accessing data in other databases. IBM has named the process of enabling a single query against a DB2 database to draw data from multiple other databases ‘data federation’. Information Integrator is the successor to an IBM product called DiscoveryLink. This paper will discuss only those elements of Information Integrator that are descended from DiscoveryLink — the parts of Information Integrator that enable data federation. Information Integrator also includes other extensions to DB2, including functionality for data migration and data replication, but our work has been based solely on the data federation aspects.

Information Integrator works with DB2 version 8. DiscoveryLink worked with DB2 version 7. The data federation components of Information Integrator provide three types of extensions to DB2 that work together to provide data federation. The first extension is to the query engine of DB2. Adding Information Integrator to a DB2 instance extends the DB2 query engine so that it understands the existence of data that are stored in external data sources and so that the query engine knows how to create a query plan that will invoke queries on those external data sources.

The second data federation extension to DB2 provided by Information Integrator is called ‘Relational Connect’. The query engine is able to make queries on other data sources by using a code module known as a wrapper. This wrapper is required to perform the following tasks: receive queries from the DB2 query engine, translate those queries into a query on the external data source, receive the results from the external data source, transform the results into rows and tables the DB2 query engine can understand, and deliver those transformed results to the DB2 query engine. The Relational Connect component of Information Integrator provides wrappers for most of the leading relational databases including Oracle, Sybase, SQL Server and MySQL, though MySQL support is available only from Windows DB2 instances (at the time of writing).

The third data federation extension to DB2 provided by Information Integrator is called ‘Life Sciences Data Connect’. This component consists of a series of wrappers that a federated DB2 query
engine can use to access data in non-relational data sources. This includes wrappers for some non-relational file formats, such as column-separated flat files, Excel and Documentum, as well as wrappers that are able to invoke commonly used molecular biology programs. Bioinformatics program wrappers include BLAST and HMMer for DNA and protein sequence similarity and a wrapper for the ENTREZ application protocol interface (API) that allows direct access to PubMed, Nucleotide and other databases at the NCBI. The SRS wrapper, which would otherwise be expected to be part of the Life Sciences Data Connect, is not actually part of Information Integrator, but is instead distributed by LION Bioscience AG.

Information Integrator is available across a broad range of platforms – we have used each of AIX, Windows, Linux and Solaris – but is notably not available for MAC OSX or MAC OS9.

**USING INFORMATION INTEGRATOR**

Installation of Information Integrator involves using generally straightforward wizards provided by IBM to install DB2, the latest fixpack for DB2, and Information Integrator itself. We have occasionally run into trouble with bugs in the install scripts on AIX and Solaris, but have been able to resolve these with help from a very responsive Life Sciences team at IBM. After installation, you are ready to configure your data sources. What you have to do to make use of Information Integrator depends on what type of information you are interested in querying. Information Integrator’s data federation allows you to query data across multiple relational and non-relational data sources through a single SQL interface. To configure Information Integrator you will need to know which data sources you want to query.

If you want to access data in another relational database, then you would have installed support for the wrapper required for that database while installing Information Integrator. We, for instance, intend to use Information Integrator to allow querying against Oracle databases, so it was necessary to have the oracle client libraries on the same machine that hosts our DB2 instance when installing Information Integrator. If you want to add support for a relational wrapper, it is reasonably convenient to rerun the Information Integrator installation wizard as necessary.

The process of configuring any wrapper is roughly the same, as the following example based on the configuration of the BLAST wrapper.

First, register the wrapper with the DB2 query engine using the `CREATE WRAPPER` command:

```
CREATE WRAPPER blast LIBRARY 'liblsblast.a';
```

The library file ‘liblsblast.a’ is provide as part of Information Integrator and installed into the DB2 package when installing Information Integrator.

Second, register a particular remote server that uses that wrapper:

```
CREATE SERVER blastn TYPE blastn VERSION 2.2.2 WRAPPER blast OPTIONS (NODE 'solar.uits.indiana.edu');
```

The above line tells the DB2 federated query engine that a remote machine, ‘solar.uits.indiana.edu’, can be accessed in order to run the blastn version of BLAST using the BLAST wrapper.

Third, register nicknames that use your server to access particular sets of data at the remote server:

```
CREATE NICKNAME ncbi.blastn_nt (gi_const varchar(2) OPTIONS(INDEX '1', DELIMITER '|'), gi_num varchar(15) OPTIONS(INDEX '2', DELIMITER '|'),
```

...
The above creates a nickname called 'blastn_nt' in the NCBI schema of our DB2 database and specifies that the results that can be returned include the fields 'gb_acc_num', 'description' and others. The BLAST wrapper also returns a large number of default fields representing the normal output of BLAST. The above specifies the name of the BLAST databank to run blastn against (nt) and how long to wait before giving up on a BLAST run (300 seconds).

The BLAST wrapper comes with an accompanying BLAST daemon that must be run on a computer that also has the BLAST application and BLAST databanks. The daemon is configured to know where the databanks are located and then listens on a particular port, waiting for requests from the wrapper. When a federated query engine needs to invoke BLAST, the wrapper accepts the query, translates it into the arguments needed for an invocation, and passes those arguments to the daemon. The daemon accepts the arguments, invokes BLAST, reads in the results, converts the results to columns and rows, and sends the columnar results back to the wrapper. The wrapper then hands the columnar results to the federated query engine that uses them in executing its query plan.

For relational wrappers – if you are accessing data sources such as Oracle, SQL Server and Sybase – a further configuration step is required to define the user that the DB2 federated query engine will use when querying tables in the remote data source. Here is an example from our configuration for a SQL Server instance we use to store dbSNP data:

```
CREATE USER MAPPING FOR db2inst2 SERVER mssql_dbsnp
  OPTIONS (REMOTE_AUTHID '(user)', REMOTE_PASSWORD '(passwd)');
```

In the above, the federated query engine is configured to use the user ‘(user)’ and the password ‘(passwd)’ to query the SQL Server referred to by ‘mssql_dbsnp’ when a query is run by the DB2 user db2inst2. Similar configurations would be required for each user who needs to have access to the remote data source. There is some tedium involved in requiring a configuration for each user, but this method in turn provides a useful level of control as to which users can or cannot access remote data sources. This is particularly useful in the context of restricting access of locally collected laboratory data to specific partners of that laboratory.

A nickname is queried just like any SQL table. A standard SQL query, for instance, on a table called ‘Person’ that had the fields ‘first_name’ and ‘last_name’ might be:

```
SELECT first_name, last_name
FROM Person
WHERE first_name LIKE 'Andy';
```

This would return results similar to:

```
FIRST_NAME  LAST_NAME
----------  ----------
Andy        Arenson
Andy        Simons
Andy        Zucker
```

If the Person table was stored in a remote database, then after configuring the
wrapper, server, nickname and user mapping, the query to get the same set of data from the remote database would be the same as querying it locally.

The BLAST wrapper provides a couple of default pseudo fields that one can use for querying, most notably the ‘blastseq’ pseudo field which one uses to specify a query sequence. Figure 1 is an example of using the BLAST nickname above.

DISCUSSION
Information Integrator versus other methods of accessing data
We have found no errors in the functionality of any of the wrappers. Our experience, though, has been primarily with the life sciences connect wrappers (BLAST, HMMer, XML, flat file, Entrez). We have only recently begun to expand into using the relational wrappers.

In terms of speed, the BLAST, HMMer and Entrez wrappers are perfectly acceptable, running far faster than the applications they invoke. The flat file wrapper performs at a reasonable speed, considering that it reads data out of the file system, but in our tests it ran one hundred times slower than querying similar data that had been loaded into a relational database and indexed. We have not tested the Documentum or Excel...

```sql
SELECT gb_acc_num, E_Value, Score
FROM ncbi.blastn_nt
WHERE blastseq = 'ttgatcttttggcgaagcccttctccatcagtctttagtgccctgtcgtttccctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctctct...'
AND Score > 300
ORDER BY Score
```

<table>
<thead>
<tr>
<th>GB_ACC_NUM</th>
<th>E_VALUE</th>
<th>SCORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>XM_212124</td>
<td>+4.82115E-079</td>
<td>+3.01811E+002</td>
</tr>
<tr>
<td>AC079796</td>
<td>+4.82115E-079</td>
<td>+3.01811E+002</td>
</tr>
<tr>
<td>AC068888</td>
<td>+2.05310E-084</td>
<td>+3.19652E+002</td>
</tr>
<tr>
<td>X55733</td>
<td>+2.05310E-084</td>
<td>+3.19652E+002</td>
</tr>
<tr>
<td>BC001097</td>
<td>+2.05310E-084</td>
<td>+3.19652E+002</td>
</tr>
<tr>
<td>BC047302</td>
<td>+2.05310E-084</td>
<td>+3.19652E+002</td>
</tr>
<tr>
<td>BC013994</td>
<td>+2.05310E-084</td>
<td>+3.19652E+002</td>
</tr>
<tr>
<td>AC087430</td>
<td>+3.32784E-086</td>
<td>+3.25599E+002</td>
</tr>
<tr>
<td>AL049695</td>
<td>+1.41717E-091</td>
<td>+3.43441E+002</td>
</tr>
<tr>
<td>XM_212124</td>
<td>+2.38467E-096</td>
<td>+3.59300E+002</td>
</tr>
<tr>
<td>BC047302</td>
<td>+2.38467E-096</td>
<td>+3.59300E+002</td>
</tr>
<tr>
<td>AC079796</td>
<td>+2.38467E-096</td>
<td>+3.59300E+002</td>
</tr>
<tr>
<td>BC013994</td>
<td>+2.38467E-096</td>
<td>+3.59300E+002</td>
</tr>
<tr>
<td>AL589880</td>
<td>+2.38467E-096</td>
<td>+3.59300E+002</td>
</tr>
<tr>
<td>AC068888</td>
<td>+4.01270E-101</td>
<td>+3.75158E+002</td>
</tr>
<tr>
<td>X55733</td>
<td>+4.01270E-101</td>
<td>+3.75158E+002</td>
</tr>
<tr>
<td>AF119854</td>
<td>+4.01270E-101</td>
<td>+3.75158E+002</td>
</tr>
<tr>
<td>NM_018507</td>
<td>+4.01270E-101</td>
<td>+3.75158E+002</td>
</tr>
<tr>
<td>BC001097</td>
<td>+4.01270E-101</td>
<td>+3.75158E+002</td>
</tr>
</tbody>
</table>
wrappers and have tested the XML wrapper only to a limited degree.

We are just starting our work with relational wrappers and do not know yet how quickly a query to an external relational database will run compared with a query directly on that database. The DB2 federated query engine is supposed to be capable of updating its statistics to take advantage of known statistics from external databases in order to determine an appropriate query plan. For large relational databases, having accurate statistics about the tables and indexes being queried will be crucial for fast performance. As an example, we were able to speed up a query on a local DB2 database from 30 seconds to nearly instantaneous by updating the statistics on the tables affected.

With a couple of exceptions such as the pseudo fields provided with the BLAST wrapper, accessing any of the data sources for which Information Integrator provides wrappers is as simple as following the standard format for accessing a table.

The flat file and Excel wrappers, unfortunately, require that the data in those file formats be very strictly organised compared with what one might expect to find. The flat file and Excel wrappers will work only if they access files where the data have already been organised into rows and columns. It is possible to imagine situations where these wrappers would be useful, but in general our opinion is that any data in a flat file or Excel file that is so organised should probably be put into a relational database in order to speed up the query time.

The SRS wrapper, which is not part of Information Integrator, but is available from LION Bioscience AG, promises to provide access to an SRS implementation, and thus any of the 300+ data sources that have been made accessible to SRS. We are looking forward to testing this wrapper in the near future. Unfortunately, the current version of the wrapper works with DB2 v7 and we do not know if or when a new version will be released for DB2 v8.

### Data that Information Integrator cannot access

If you want to access data from SWISSPROT, KEGG, SGD or dozens of other popular sources, Information Integrator by itself cannot yet help you. Although Information Integrator’s data federation can be quite useful in joining data sources together, much of the data that one might want to join are not yet available to any of its wrappers.

One promising note is that if the SRS wrapper turns out to work well and a version is released for DB2 version 8, then the vast majority of publicly available molecular biology databases will become available to DB2, though this will require maintaining a local SRS implementation. The addition of the Entrez wrapper and the possibility of more of GenBank’s data divisions becoming available through it are also very promising.

In the meantime, here at Indiana University we have written software to download some of these popular databases, parse them into relational format, and upload them into a local relational database so that they can be joined with the other data we access using Information Integrator. We have chartered the IBM Data Federation Users Group and are happy to share our parsers and discuss ways to make the best use of data federation.

### CONCLUSION

Information Integrator can provide a method for centralising data and an interface for programmatically gathering data, rather than relying on web browsing. It can simplify the use of BLAST and HMMer by handling the tasks of launching many runs and parsing the results. It incorporates a few useful data sources, notably PubMed and the Nucleotide division of GenBank through the Entrez wrapper, but unless the SRS wrapper turns out to be useful and supported, is significantly weak in the range of public data that is accessible.

Even the advantage of running BLAST and HMMer comes at the price of...
needing to manage the databanks these applications use locally.

The Indiana Genomics Initiative (INGEN)\textsuperscript{11} IT Core has used Information Integrator successfully as a part of our Centralized Life Sciences Data (CLSD)\textsuperscript{12} service to remove the burden of gathering and managing data from various programmers in the IU School of Medicine. One bioinformatics expert described a program that he had written as accomplishing in 30 seconds what had previously taken researchers 10 hours to gather via web interfaces. The success of CLSD, though, required a significant amount of work beyond using Information Integrator.

If and when Information Integrator is able to access the wide range of molecular biology data required by most researchers, it will become a must-have tool. In the meantime, it is a highly useful tool in specific situations or when a central computing group can augment its capabilities for use by a large group of researchers.

Acknowledgments

This work was supported in part by the Indiana Genomics Initiative (INGEN). The Indiana Genomics Initiative (INGEN) of Indiana University is supported in part by Lilly Endowment Inc. This work was supported in part by Shared University Research Grants from IBM, Inc. to Indiana University.

Andrew D. Arenson,

INGEN IT Core,

Research and Academic Computing Division,
University Information Technology Services,
Indiana University,
Indianapolis, IN 46202, USA
Tel: +1 317 278 1208
E-mail: aarenson@iupui.edu

References


9. Saccharomyces Genome Database (URL: http://www.yeastgenome.org/), Department of Genetics at the School of Medicine, Stanford University (visited 12th September, 2003).

