

Using a different data store

- [The options](#)
- [Different database type](#)
- [Other types of triple stores](#)
 - [Using TDB](#)
 - [Using web-based triple stores](#)
 - [Using Virtuoso](#)

VIVO can work with databases like Oracle or SQL Server, or with other types of triple stores.

The options

The default configuration of VIVO uses the Jena SDB triple-store to hold its content. SDB requires a relational database, and the default configuration specifies MySQL as the storage for SDB.

You can configure VIVO to use a different database as the basis for Jena SDB. You can also configure VIVO to use a different triple store.

Different database type

To use a database other than MySQL, set these values in `runtime.properties`:

Property name	<code>VitroConnection.DataSource.dbtype</code>
Description	Change the dbtype setting to use a database other than MySQL. Otherwise, leave this value unchanged. Possible values are DB2, derby, HSQLDB, H2, MySQL, Oracle, PostgreSQL, and SQLServer. Refer to http://openjena.org/wiki/SDB/Databases_Supported for additional information. Earlier releases of VIVO used Jena version 2.6.4, which does not work correctly with Microsoft SQL Server. Since release 1.7, VIVO uses Jena version 2.10.1, which should fix this problem.
Default value	MySQL
Example value	Oracle

Property name	<code>VitroConnection.DataSource.driver</code>
Description	Specify a driver class name to use a database other than MySQL. Otherwise, leave this value unchanged. This JAR file for this driver must be added to the <code>webapp/lib</code> directory within the Vitro installation directory, as specified by <code>vitro.core.dir</code> in the <code>build.properties</code> file.
Default value	<code>com.mysql.jdbc.Driver</code>
Example value	<code>com.mysql.jdbc.Driver</code>

Property name	<code>VitroConnection.DataSource.validationQuery</code>
Description	Change the validation query used to test database connections only if necessary to use a database other than MySQL. Otherwise, leave this value unchanged.
Default value	<code>SELECT 1</code>
Example value	<code>SELECT 1</code>

Depending on your database, you might also need to change the basic database properties:

- `VitroConnection.DataSource.url`
- `VitroConnection.DataSource.username`
- `VitroConnection.DataSource.password`

Other types of triple stores

To use a triple store other than Jena SDB, you will need to edit `applicationSetup.n3`, in the `config` sub-directory of your VIVO home directory. In the standard configuration, this file specifies that VIVO's content triples will be handled by a Java class called `ContentTripleSourceSDB`

You will need to be familiar with the syntax of N3 files before editing this file.

Here are the lines that establish that configuration:

```
:application
  :hasContentTripleSource :sdbContentTripleSource ;
```

```
:sdbContentTripleSource
  a    <java:edu.cornell.mannlib.vitro.webapp.triplesource.impl.sdb.ContentTripleSourceSDB> ,
        <java:edu.cornell.mannlib.vitro.webapp.modules.triplesource.ContentTripleSource> .
```

The file also contains templates for using other triple stores. Uncomment the one you want to use, provide the parameters as described, and change the `:hasContentTripleSource` property to point to the template you chose.

`ContentTripleSourceSDB` looks at `runtime.properties` for the details of its configuration. This maintains compatibility with older releases of VIVO. Other triple store implementations must be configured in `applicationSetup.n3`, as shown. They ignore `runtime.properties`.

Using TDB

TDB is another triple store from the Jena project, but TDB keeps its data in directory of flat files, while SDB uses a relational database. To use TDB, you can specify the `ContentTripleStoreTDB` class, and provide the path to a directory.

Change the `:hasContentTripleSource` property:

```
:application
  :hasContentTripleSource :tdbContentTripleSource ;
```

Uncomment the template for TDB and provide a directory path. For example:

```
:tdbContentTripleSource
  a    <java:edu.cornell.mannlib.vitro.webapp.triplesource.impl.tdb.ContentTripleSourceTDB> ,
        <java:edu.cornell.mannlib.vitro.webapp.modules.triplesource.ContentTripleSource> ;
  # May be an absolute path, or relative to the Vitro home directory.
  :hasTdbDirectory "tdbContentModels" .
```

Using web-based triple stores

VIVO can use an external triple store, if the triple store supports Web-based use of the SPARQL language to query and modify its data. In order to connect VIVO to an external triple store, you will need to know two URIs: the store's endpoint URI for issuing SPARQL queries that read data, and its URI for issuing SPARQL UPDATE commands. These URIs are typically kept separate in order to make it easier to secure the triple store against unauthorized edits. With [Sesame](#), for example, the update URI is usually the query endpoint URI with `/statements` appended.

Change the `:hasContentTripleSource` property:

```
:application
  :hasContentTripleSource :sparqlContentTripleSource ;
```

Uncomment the template for SPARQL triple stores, and provide the two URIs, as described above. For example:

```
:sparqlContentTripleSource
  a    <java:edu.cornell.mannlib.vitro.webapp.triplesource.impl.sparql.ContentTripleSourceSPARQL> ,
        <java:edu.cornell.mannlib.vitro.webapp.modules.triplesource.ContentTripleSource> ;
  # The URI of the SPARQL endpoint for your triple-store.
  :hasEndpointURI "http://example.org/sesame" ;
  # The URI to use for SPARQL UPDATE calls against your triple-store.
  :hasUpdateEndpointURI "http://example/sesame/statements" .
```

Using Virtuoso

[Virtuoso](#) is a multi-purpose server engine from OpenLink Software, Inc. It operates as a web-based triple store, as described in the previous section. However, it has some minor peculiarities, which are encapsulated in the `ContentTripleSourceVirtuoso` Java class.

Change the `:hasContentTripleSource` property:

```
:application
  :hasContentTripleSource :virtuosoContentTripleSource ;
```

Uncomment the template for Virtuoso triple stores, provide the base URI for accessing Virtuoso, provide the username and password of an authorized account on Virtuoso. For example:

```
:virtuosoContentTripleSource
  a    <java:edu.cornell.mannlib.vitro.webapp.triplesource.impl.virtuoso.ContentTripleSourceVirtuoso> ,
        <java:edu.cornell.mannlib.vitro.webapp.modules.tripleSource.ContentTripleSource> ;
  # The URI where Virtuoso can be accessed: don't include the /sparql path.
  :hasBaseURI "http://localhost:8890" ;
  # The name and password of a Virtuoso account that has the SPARQL_UPDATE role.
  :hasUsername "USERNAME" ;
  :hasPassword "PASSWORD" .
```