**List of object-relational mapping software**

* [ActiveJDBC](https://en.wikipedia.org/wiki/ActiveJDBC), Java implementation of [Active record pattern](https://en.wikipedia.org/wiki/Active_record_pattern), inspired by Ruby on Rails
* [ActiveJPA](https://en.wikipedia.org/wiki/ActiveJPA), open-source Java ORM JPA-like implementation of [Active record pattern](https://en.wikipedia.org/wiki/Active_record_pattern)
* [Apache Cayenne](https://en.wikipedia.org/wiki/Apache_Cayenne), open-source for Java
* [Apache Gora](https://en.wikipedia.org/wiki/Apache_Gora), open-source software framework provides an in-memory data model and persistence for big data focused on [NoSQL](https://en.wikipedia.org/wiki/NoSQL) and SQL stores
* [Athena Framework](https://en.wikipedia.org/wiki/Athena_Framework), open-source Java ORM, native support for multitenancy SaaS and remoting to [Adobe Flex](https://en.wikipedia.org/wiki/Adobe_Flex)
* [Carbonado](https://en.wikipedia.org/wiki/Carbonado_(Java)), open-source framework, backed by [Berkeley DB](https://en.wikipedia.org/wiki/Berkeley_DB) or [JDBC](https://en.wikipedia.org/wiki/Java_Database_Connectivity)
* [DataNucleus](https://en.wikipedia.org/wiki/DataNucleus), open-source JDO and JPA implementation (formerly known as JPOX)
* [Ebean](https://en.wikipedia.org/wiki/Ebean), open-source ORM framework
* [EclipseLink](https://en.wikipedia.org/wiki/EclipseLink), Eclipse persistence platform
* [Enterprise JavaBeans](https://en.wikipedia.org/wiki/Enterprise_JavaBeans) (EJB)
* [Enterprise Objects Framework](https://en.wikipedia.org/wiki/Enterprise_Objects_Framework), Mac OS X/Java, part of Apple [WebObjects](https://en.wikipedia.org/wiki/WebObjects)
* [Fast Java Object Relation Mapping](https://en.wikipedia.org/wiki/Fjorm_(software)) (Fjorm)
* [Hibernate](https://en.wikipedia.org/wiki/Hibernate_(framework)), open-source ORM framework, widely used
* [Java Data Objects](https://en.wikipedia.org/wiki/Java_Data_Objects) (JDO)
* [Java Object Oriented Querying](https://en.wikipedia.org/wiki/Java_Object_Oriented_Querying) (jOOQ)
* [Kodo](https://en.wikipedia.org/wiki/Apache_OpenJPA), commercial implementation of both [Java Data Objects](https://en.wikipedia.org/wiki/Java_Data_Objects) and [Java Persistence API](https://en.wikipedia.org/wiki/Java_Persistence_API)
* [Kundera](https://en.wikipedia.org/wiki/Kundera), open-source framework, JPA compliant, [polyglot](https://en.wikipedia.org/wiki/Polyglot) object-datastore mapping library for [NoSQL](https://en.wikipedia.org/wiki/NoSQL) datastores
* [MyBatis](https://en.wikipedia.org/wiki/MyBatis), free open-source, formerly named [iBATIS](https://en.wikipedia.org/wiki/IBATIS)
* [QuickDB ORM](https://en.wikipedia.org/wiki/QuickDB_ORM), open-source ORM framework
* [TopLink](https://en.wikipedia.org/wiki/TopLink) by Oracle
* [Torque](https://en.wikipedia.org/wiki/Apache_Torque), an object-relational mapper for Java

Some of the frameworks support the required features by default. For others, I had to write some kind of interlayer code. Obsolete frameworks or frameworks that made it impossible to use any of the features have been excluded from this test.

* [ActiveJDBC](http://javalite.io/activejdbc) is the implementation of the active record pattern. The library is lightweight and easy to use. Unfortunately, it does not integrate with spring transaction management, so I had to write some code for it. It's author, Igor Polevoy, reacts really quickly to new requests in GitHub's issue tracker.
* [Apache Cayenne](http://cayenne.apache.org/) is an ORM that generates domain objects (DO) from an XML model that can be managed by bundled a GUI application called a modeler. Same as ActiveJDBC, I had to write a custom integration into Spring Transaction Management (STA).
* [Apache OpenJPA](http://openjpa.apache.org/) is the [JPA](https://dzone.com/refcardz/getting-started-with-jpa) implementation made by Apache. Even though Spring provides light support for the integration of OpenJPA, it is harder to integrate it than Hibernate. Due to this, I think it doesn't make too much sense to use OpenJPA in the Spring world. The project uses bytecode enhancement of compiled DO classes.
* [DataNucleus Access Platform](http://www.datanucleus.org/) (formerly known as JPOX) is a rich framework that provides implementations of JPA, JDO, and REST APIs. For me, the JPA was the easiest way to implement it. The framework provides some handy mappings, like Java 8 time API mapping. Unfortunately, the mappings can't be used in native query calls, so it is necessary to write an extra set of mappings.
* [Ebean](http://ebean-orm.github.io/) is flexible and lightweight ORM that is actually used by the [Play Framework](https://www.playframework.com/). Even though the ORM is not shipped with Spring, it was really easy to integrate it. The framework uses bytecode enhancement of DO classes.
* [EclipseLink](http://www.eclipse.org/eclipselink/) is the JPA implementation for Eclipse. It has some support in Spring but, like with OpenJPA, it doesn't make too much sense to use it. To be able to use user-defined types, I had to write some mappers. The framework uses a runtime weaver to enhance DOs.
* [jOOQ](http://www.jooq.org/) is one of the three frameworks that can be chosen on the [Spring Initializr](http://start.spring.io/) page. This means a good integration with Spring. The ORM generates code not only for DOs but also for user-defined types and stored procedures. It still has some flaws, like problems with combining [UDTs with POJOs](https://github.com/jOOQ/jOOQ/issues/5401) or DAO insert methods that [don't return autogenerated primary key values](https://github.com/jOOQ/jOOQ/issues/2536). On the other hand, [Lukas Eder](https://dzone.com/articles/dev-week-lukas-eder), the author of the framework, is doing a really good job of improving the framework and also propagating it on this site.
* [MyBatis](http://mybatis.org/) is the third ORM option available on Spring Initializr page. MyBatis is a mature, lightweight framework with good integration with Spring. Unfortunately, due to its lightweightness, it lacks native support for a lot of PostgreSQL's advanced features, so I had work around that limitation to implement it.
* [Hibernate](http://hibernate.org/) is used as the default implementation of JPA in the [Spring Data JPA](http://projects.spring.io/spring-data-jpa/) project. Thanks to this, it has perfect integration with Spring. The ORM is quite heavy, and I think it is quite hard to use.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **JSON** | **Composite or enum type** | **Array** | **Stored procedures** | **Auditing** | **Pagination** | **Time API** | **STA** |
| **ActiveJDBC** | Custom mapper | Custom mapper | Custom mapper | Pure JDBC call | Custom code | Partial support | Custom mapper | Custom code |
| **Cayenne** | Custom mapper | Custom mapper | Custom mapper | Partial support | Custom code | Custom code | Custom mapper | Custom code |
| **OpenJPA** | Custom mapper | Custom mapper | Custom mapper | Partial support | Custom code | Custom code | Custom mapper | Default |
| **DataNucleus** | Custom mapper | Custom mapper | Default | Partial support | Custom code | Custom code | Default | Default |
| **Ebean** | Default | Custom mapper | Default | Partial support | Default | Custom code | Default | Default |
| **EclipseLink** | Custom mapper | Custom mapper | Custom mapper | Partial support | Custom code | Custom code | Custom mapper | Default |
| **jOOQ** | Default | Default | Default | Default | Custom code | Custom code | Custom mapper | Default |
| **MyBatis** | Custom mapper | Custom mapper | Custom mapper | Partial support | Custom code | Custom code | Custom mapper | Default |
| **Hibernate** | Custom mapper | Custom mapper | Custom mapper | Partial support | Default | Default | Default | Default |

By custom code, I mean that an ORM allows us to use a feature by providing some kind of low-level API, but a lot of work is still up to the developer. Custom mapper or partial support means that an ORM provides higher-level API. And default is used for features that are supported by ORM natively via a method or use an interface.

**Flex [**[**edit**](https://en.wikipedia.org/w/index.php?title=List_of_object-relational_mapping_software&action=edit&section=2)**]**

* [Athena Framework](https://en.wikipedia.org/wiki/Athena_Framework), open source Flex ORM, native support for [multitenancy](https://en.wikipedia.org/wiki/Multitenancy)