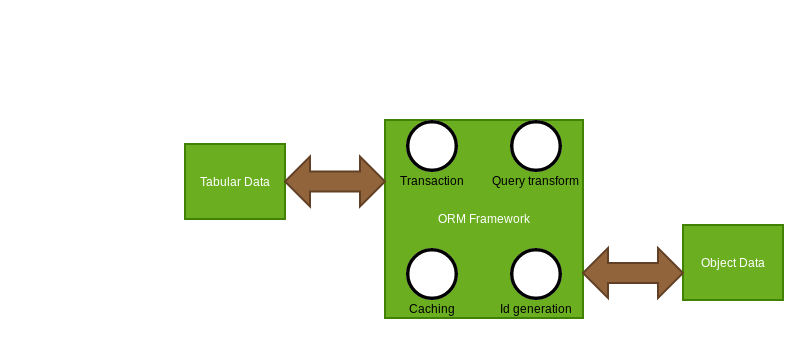
Introduction

**What is ORM?**

ORM stands for Object Relational Model, We can think ORM as a transformer which changes the table data structure to a corresponding Object Data Structure or Vice Versa. So we can think it is a programming technique which takes the data as tabular form and convert them to an Object form , mapping each column with Object property and present to Program developers, and vice versa.



**ORM Framework as a Transformer**

So, The definition of ORM is very easy to understand but it is important to know

**Why we need ORM ?**

**What are the advantages we /developers get from ORM?**

To answer the first question let see how we fetch data from database using JDBC,

1. Load database driver.
2. Create connection Object.
3. Get Statement Object from Connection.
4. Structure the query, RDBMS specific.
5. Execute the query using Statement.
6. Get the ResultSet.
7. Iterate Over the Resultset.
8. Manually fetch column value from Resultset for each row.
9. Close the statement Object.
10. Close the connection Object.

A long procedure to maintain isn’t it?

JDBC has exposed an API to the developer , Developers are calling different methods on that API to get the job done.

**Disadvantages of the JDBC** :

1. The developer has too many responsibilities, where the main focus of the developer should be concentrate on business logic , to be specifically developers responsibility should be step 4 to Step 7. But developers have to perform all the steps to get the job done efficiently.
2. Another big disadvantage is fetched data from resultset Object, Developer has to iterate over result set and fetch the columns either by index position or by column name which is very error prone a slight error in column name or case sensitivity or error in index position can give you a fatal error , Not only that think about a scenario if two columns has same data type but when we fetch these columns say position has been interchanged then JDBC will not throw an error as data type is same, but we get a wired result. It would be nice if framework takes care of the mapping rather than a developer.
3. JDBC has no encapsulation, It does not hide the implementation details from the developer, so the developer has to provide a raw DBMS specific SQL query , so if in the future underlying database has been changing developers has to change the JDBC layer accordingly as they are strongly coupled. It would be nice if an abstraction is present there which will take DSL from Developer and convert it to RDBMS specific queries.
4. Another problem is in JDBC, developer has to close the connection and manage the transaction, which should be handled by the framework. This is not developers responsibility at all.

ORM framework addresses these problems and solves this problem so developers only concentrate on the business logic.

**Basic 4 pillars of ORM**

1. Provide an API by which we can perform CRUD operations on Domain Objects or Persistence class we called them Entity.
2. Provide a DSL(Domain specific language) by which we can do the query on Entity properties, and framework will convert it to RDBMS specific queries.
3. Provide and mapping techniques by XML or Annotation by which a Domain
4. Object/Entity can map with an underlying database table.

The framework provides Transaction,close connection optimizes query techniques.

**Advantages of ORM:**

1. Developers work on entity or POJO , ao POJO hides the Database table from Developers a perfect encapsulation. So developers code is independent of the underlying database.
2. As ORM provides DSL(Domain specific languages), Developer does not have to bother about SQL query.The framework generates it from DSL and optimizes the same accordingly.
3. Automated Transaction management.
4. Automated Key generation.
5. Caching query result or Domain to enhance performance and reduce the cost of database hitting.

Hibernate is one of the Popular ORM frameworks there are other ORM frameworks like EclipseLink,Toplink,Castor etc.