The Active Record Design Pattern

* The Active Record design pattern was named by Martin Fowler in 2003 in his book Patterns of Enterprise Application Architecture.
* Active Records are commonly used in Ruby as a means of persisting data, i.e., of saving data persistently, so that it can be recalled for later use, even after you’ve closed the program that created the data, and turned off your computer.
* More specifically, the Active Record Pattern is used to access data stored in relational database – it allows us to perform CRUD operations without worrying about the specific underlying database technology (e.g., SQLite, MySQL, PostgreSQL, SQL Server, Oracle etc.).
* Most software applications today are written using some OO language, and often it is necessary to persist (i.e., save and later retrieve) the objects associated with these applications.
* There’s a big problem: The classes and objects associated with an OO language are incompatible with the structure of relational databases.
* Active Records to the Rescue: This design pattern encapsulates that notion an object-relational mapping (ORM), i.e., a mapping between OO language constructs and relational database constructs.
* The ORM provided by Active Records automatically converts object into constructs that can be stored in a database (and converts them back upon retrieval).
* This creates, effect, a “virtual object database” that can be used form within an OO language.

ORM – How it’s done

Active Records use the following ORM:

|  |  |
| --- | --- |
| OO Language Feature | Relations DB Item |
| Classes | Tables |
| Objects | Records (Rows in a Table) |
| Attributes | Record Values (Columns in a Table) |

Active Records in Ruby

* The Active Record design pattern is provided in a Ruby module called ActiveRecord.
* Using the functionality provided by this module you can:
* Establish a connection to a database.
* Create database tables.
* Specify associations between tables that correspond to associations between the Ruby classes.
* Establish an ORM between Ruby classes/objects/attributes and the tables/rows/columns in the underlying database.
* Perform CRUD operations on Ruby ActiveRecord objects.
* The ActiveRecord module is built into Rails – the functionalities above are utilized when we create a Rails app and run scaffold and model generators.