**Propósito**

DAO es un patrón de diseño utilizado para crear una capa de persistencia

**Problema**

Una de las grandes problemáticas al momento de acceder a los datos, es que la implementación y formato de la información puede variar según la fuente de los datos. Implementar la lógica de acceso a datos en la capa de lógica de negocio puedes ser un gran problema, pues tendríamos que lidiar con la lógica de negocio en sí, más la implementación para acceder a los datos, adicional, si tenemos múltiples fuentes de datos o estas pueden variar, tendríamos que implementar las diferentes lógicas para acceder a las diferentes fuentes de datos, como podrían ser: bases de datos relacionales, No SQL, XML, archivos planos, Webservices, etc).

**Solución**

Dado lo anterior**, el patrón DAO propone separar por completo la lógica de negocio de la lógica para acceder a los datos,** de esta forma, el DAO proporcionará los métodos necesarios para insertar, actualizar, borrar y consultar la información; por otra parte, la capa de negocio solo se preocupa por lógica de negocio y utiliza el DAO para interactuar con la fuente de datos.

**Aplicabilidad**

* DAO encapsula el acceso a la base de datos. Por lo que cuando la capa lógica de negocio necesite interactuar con la base de datos, va a hacerlo a través de la API que le ofrece DAO.
* Generalmente esta API consiste en métodos CRUD (Create, Read, Update, y Delete). Entonces por ejemplo cuando la capa de lógica de negocio necesite guardar un dato en la base de datos va a llamar a un método créate().
* DAO Consiste básicamente en una clase que es la que interactúa con la base de datos.
* Los métodos de esta clase dependen de la aplicación y de lo que queramos hacer.

**Pros**

* Los Objetos de Acceso a Datos son un Patrón de los subordinados de Diseño Core [J2EE](http://es.wikipedia.org/wiki/J2EE) y considerados una buena practica. La ventaja de usar objetos de acceso a datos es que cualquier objeto de negocio (aquel que contiene detalles específicos de operación o aplicación) no requiere conocimiento directo del destino final de la información que manipula.
* Los Objetos de Acceso a Datos pueden usarse en Java para aislar a una aplicación de la tecnología de persistencia Java subyacente (API de Persistencia Java), la cual podría ser JDBC, JDO, Enterprise JavaBeans, TopLink, Hibernate, iBATIS, o cualquier otra tecnología de persistencia. Usando Objetos de Acceso de Datos significa que la tecnología subyacente puede ser actualizada o cambiada sin cambiar otras partes de la aplicación.

## Contras

* La flexibilidad tiene un precio. Cuando se añaden DAOs a una aplicación, la complejidad adicional de usar otra capa de persistencia incrementa la cantidad de código ejecutado durante tiempo de ejecución. La configuración de las capas de persistencia requiere en la mayoría de los casos mucho trabajo.
* Las aplicaciones críticas con el rendimiento no deberían usar DAOs.