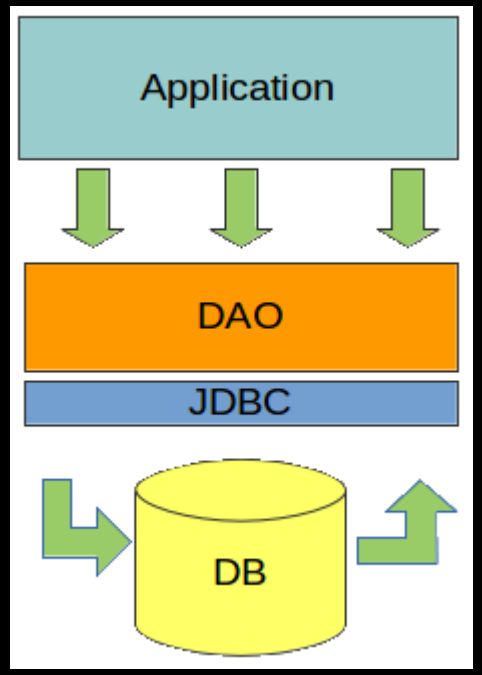
Spring DATA

DAO (Data Access Object)



During the design of some information system, the logic of this system is usually divided into layers. DAO (Data Access Object) – is a layer, that interactive with database and had determined methods for that.

Transaction management

A transaction is associated with Session and instantiated by calling **session.beginTransaction()**

The methods of **Transaction** interface are as follows:

1. **void begin()** - starts a new transaction
2. **void commit() -** ends the unit of work unless we are in FlushMode.NEVER
3. **void rollback()** - forces this transaction to rollback
4. **void setTimeout(int seconds)** - it sets a transaction timeout for any transaction started by a subsequent call to begin on this instance.
5. **boolean isAlive()** - checks if the transaction is still alive
6. **void registerSynchronization(Synchronization s)** - registers a user synchronization callback for this transaction.
7. **boolean wasCommited()** - checks if the transaction is commited successfully
8. **boolean wasRolledBack()** - checks if the transaction is rolledback successfully

Table

```java
package com.example.restcrud.user;

// imports

@Entity
@Table(name = "user")
public class User implements Serializable {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private Long id;

    @Column(name = "email", nullable = false)
    private String email;

    @ManyToOne(fetch = FetchType.LAZY)
    @Fetch(FetchMode.JOIN)
    @JoinColumn(
            name = "role_id",
            foreignKey = @ForeignKey(name = "fk_user_user_role")
    )
    private UserRole role;

    @Column(name = "role_id", insertable = false, updatable = false)
    private Integer roleId;

    @Column(name = "created_at", nullable = false)
    private Date createdAt;

    @Column(name = "name")
    private String name;

    // getters and setters
}
```

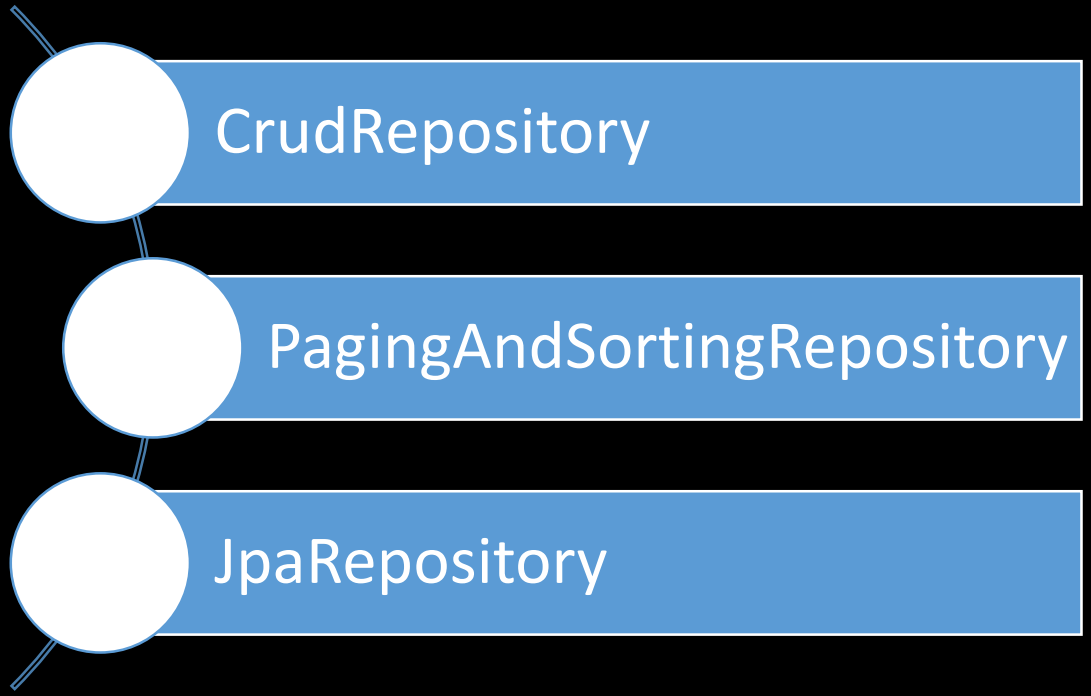
Each entity must have at least two annotations defined: @Entity and @Id.

The @Entity annotation specifies that the class is an entity and is mapped to a database table.

The @Table annotation specifies the name of the database table to be used for mapping.

The @Id annotation specifies the primary key of an entity and the @GeneratedValue provides for the specification of generation strategies for the values of primary keys.

Different kinds of Spring Data repository interfaces



CrudRepository — Interface for generic CRUD (Create, Read, Update, and Delete) operations on a repository for a specific type.

PagingAndSortingRepository — Extension of CrudRepository to provide additional methods to retrieve entities using the pagination and sorting abstraction.

JpaRepository — Extension of PagingAndSortingRepository to provides JPA related methods such as flushing the persistence context and deleting records in a batch. Because of inheritance, it contains all methods of the first two repository interfaces.