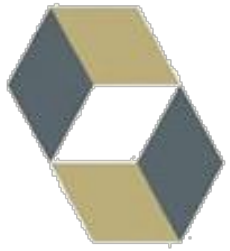


HIBERNATE

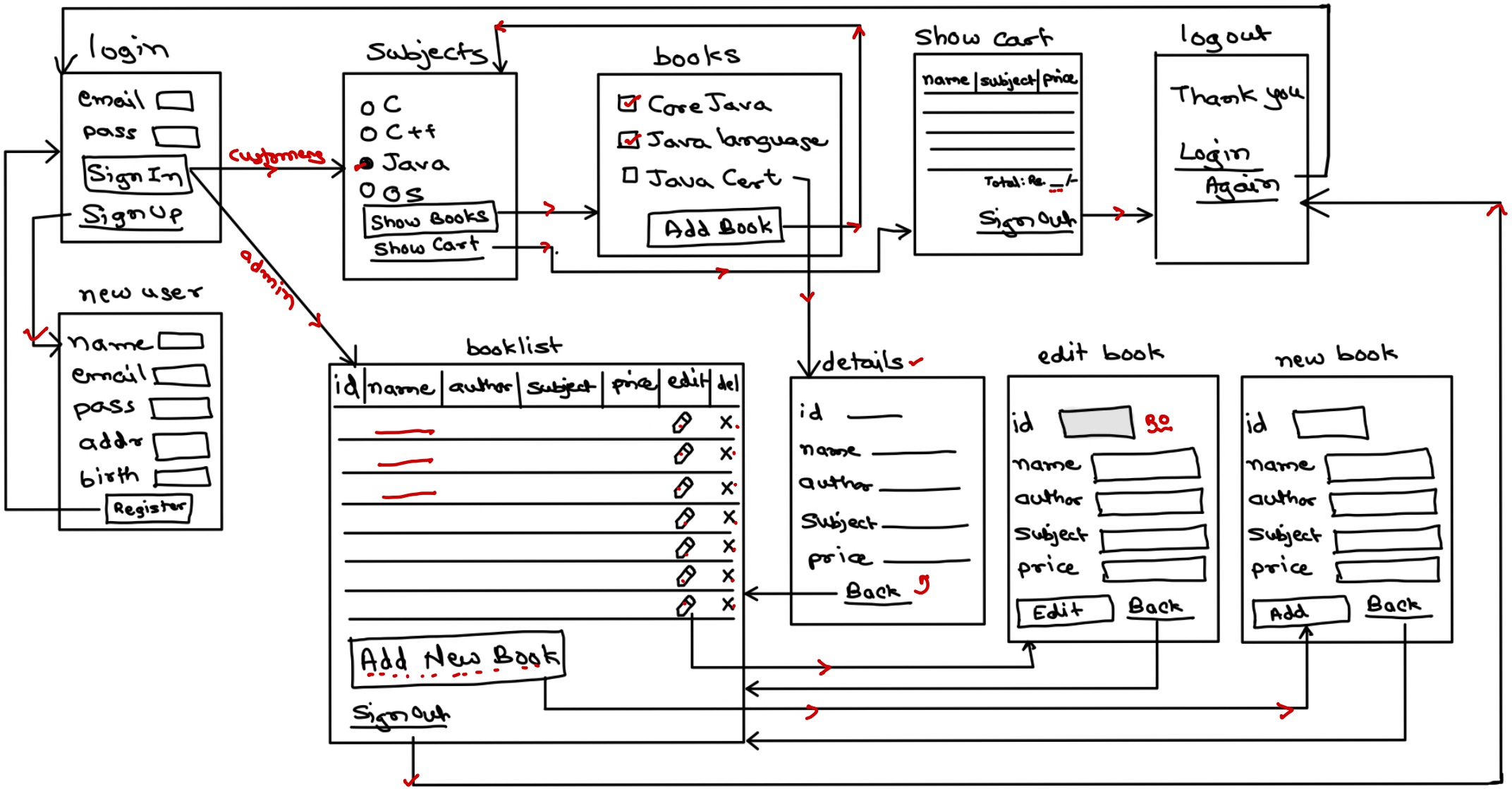
Trainer: Nilesh Ghule

Wake up from Hibernate, Spring up!!!

Agenda

- HQL
- Named Queries
- Calling stored procedure
- Auto-generated primary key
- Composite primary key
- Hibernate relations
- @OneToMany
- @ManyToOne

Project requirement



Project requirement

CustomerDao

1. findByEmail()
2. addCustomer()
3. editCustomer()

BookDao

1. getSubjects()
2. findBySubject()
3. findById()
4. findAll()
5. editBook()
6. deleteBook()
7. addBook()

Native Queries and HQL

- Ad-hoc SQL queries (on tables) can be executed in hibernate directly in hibernate.
  - NativeQuery q = session.createSQLQuery(sql);
- Hibernate recommends using HQL for ad-hoc queries.
- These queries are on hibernate entities (not on tables).
  - Query q = session.createQuery(hql);
- HQL supports SELECT, DELETE, UPDATE operation.
- INSERT is limited to INSERT INTO ... SELECT ...;

Hibernate – HQL

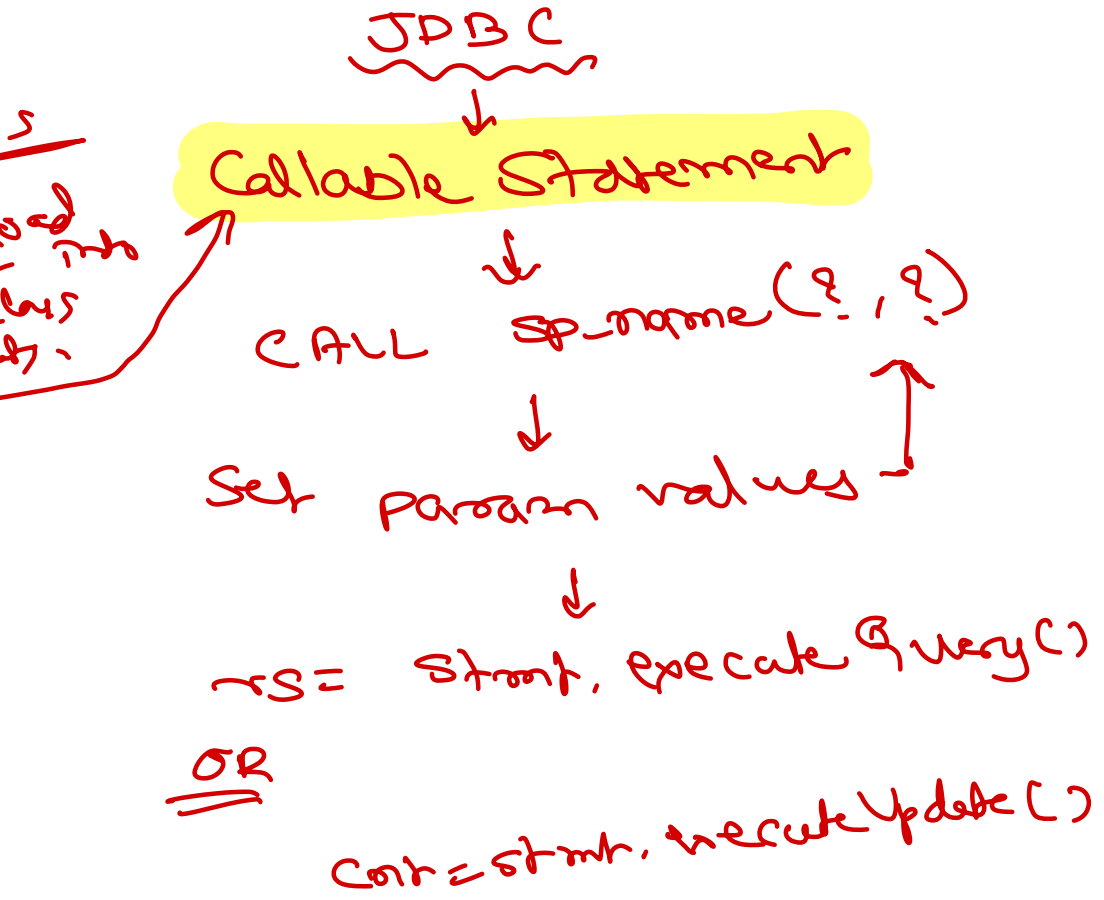
- SELECT
  - from Book b
  - from Book b where b.subject = :p_subject
  - from Book b order by b.price desc
  - select distinct b.subject from Book b
  - select b.subject, sum(b.price) from Book b group by b.subject
  - select new Book(b.id, b.name, b.price) from Book b
- DELETE
  - delete from Book b where b.subject = :p_subject
- UPDATE
  - update Book b set b.price = b.price + 50 WHERE b.subject = :p_subject
- INSERT
  - insert into Book(id, name, price) select id, name, price from old_books

Named queries

- Using multiple HQL queries in project will scatter them into multiple files and thus application maintenance become complicated.
- All queries related to an entity can be associated with the class using @NamedNativeQuery or @NamedQuery.
- NamedNativeQuery represent SQL query.
  - Use session.getNamedNativeQuery() to access NativeQuery.
  - Invoke methods on NativeQuery to perform appropriate operations.
- NamedQuery represent HQL query.
  - Use session.getNamedQuery() to access NativeQuery.
  - Invoke methods on NativeQuery to perform appropriate operations.
- To associate multiple queries use @NamedNativeQueries or @NamedQueries.

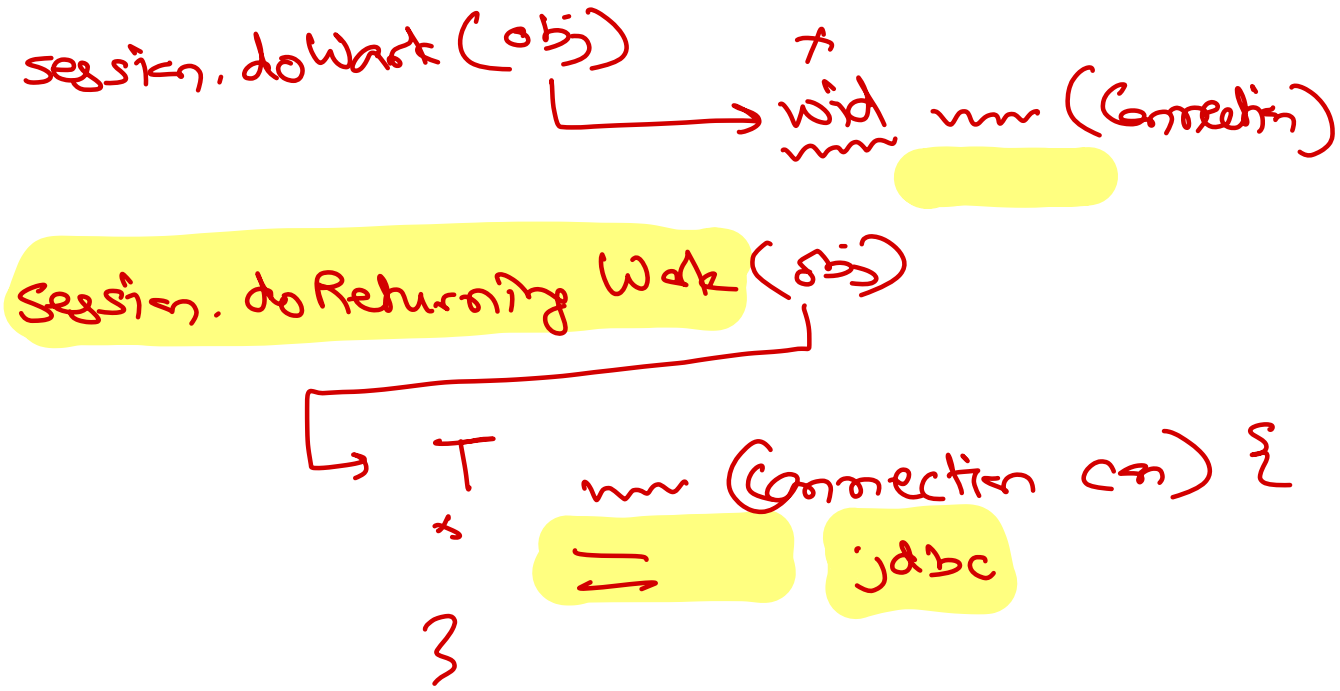
Hibernate 3 – Calling Stored Procedure

- Use @NamedNativeQuery to define stored procedure call statement.
  - @NamedNativeQuery(name="qry", query="CALL books_in_range(:p1,:p2)", hints = {@QueryHint (name = "org.hibernate.callable", value = "true") })
- Simple SP (with no out params) can be executed like named queries.
- If SP returns rows (due to SELECT), resultClass can be set in @NamedNativeQuery.



Hibernate 3 – Calling Stored Procedure

- Hibernate3 doesn't have support for out params, so JDBC code can be embedded using doWork() or doReturningWork() method.
- They respectively use Work (returns void) and ReturningWork<> (returns value) functional interfaces having jdbc Connection as arg.



Hibernate 5 – Calling Stored Procedure

- Hibernate5 provides ProcedureCall object to deal with SP.
  - Way1: session.createStoredProcedureCall()
  - Way2: use @NamedStoredProcedureQuery on entity class and session.getNamedProcedureCall() similar to named queries.
- Procedure call deal with out params using registerStoredProcedureParameter() and setting param mode as IN or OUT.
- This is JPA compliant.

Auto-generate Primary keys

- @GeneratedValue annotation is used to auto-generate primary key.
- This annotation is used with @Id column.
- There are different strategies for generating ids.
  - AUTO: Depends on database dialect.
    - MySQL 8: id will be taken from "next_val" column of "gen" table.
  - IDENTITY: RDBMS AUTO_INCREMENT / IDENTITY
    - MySQL 8: id will be AUTO_INCREMENT
  - TABLE: Dedicated table for PK generation
    - MySQL 8: @TableGenerator(name="gen", initialValue=1000, pkColumnName="book_ids", valueColumnName="id", table="id_gen", allocationSize=1)
  - SEQUENCE: RDBMS sequence using @SequenceGenerator
    - MySQL 8: Emulated with table.

hibernate_sequence or name = "gen"

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>