**Spring JPA Batch Insert and Update - 2022**

**Entity Layer**

@Entity(name = "StarCast")

@Table(name = "starcast")

@Getter @Setter

**public** **class** StarCast {

@Id @GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

}

@Entity(name = "Movie")

@Table(name = "movie")

@Getter @Setter

**public** **class** Movie {

@Id @GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

@OneToMany(cascade = CascadeType.***ALL***)

**private** Set<StarCast> starCasts =

**new** HashSet<>();

}

**application.properties**

## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)

spring.datasource.url=jdbc:postgresql://localhost:5432/dev

spring.datasource.username=postgres

spring.datasource.password=postgres

# The SQL dialect makes Hibernate generate better SQL for the chosen database

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.PostgreSQLDialect

# Hibernate ddl auto (create, create-drop, validate, update)

spring.jpa.hibernate.ddl-auto = update

**spring.jpa.properties.hibernate.generate\_statistics=true**

**spring.jpa.properties.hibernate.jdbc.batch\_size=5**

**spring.jpa.properties.hibernate.order\_inserts=true**

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

**\*\* remembering that Hibernate also has an order\_updates property**

**Repository Layer**

@Repository

**public** **interface** One2ManyUniRepository **extends** CrudRepository<Movie, Long> {

}

**Service layer**

@Service

**public** **class** One2ManyUniServiceImpl **implements** One2ManyUniService {

**public** **interface** One2ManyUniService {

**void** saveAllMovies(List<Movie> movies);

}

@Autowired

**private** One2ManyUniRepository repo;

@Override

**public** **void** saveAllMovies(List<Movie> movies) {

repo.saveAll(movies);

}

}

@Component

**public** **class** One2ManyBatchInsert {

@Autowired

**private** One2ManyUniService service;

**public** **void** save() {

Instant start = Instant.*now*();

List<Movie> movieList = **new** ArrayList<>();

**for**(**int** i = 0; i < 100; i++) {

Movie movie = **new** Movie();

movie.setName("Movie Name "+(i+1));

**for**(**int** j = 0; j < 5; j++) {

StarCast cast = **new** StarCast();

cast.setName("Cast Name "+ (j+1));

movie.getStarCasts().add(cast);

}

movieList.add(movie);

}

service.saveAllMovies(movieList);

Instant end = Instant.*now*();

Duration timeElapsed = Duration.*between*(start, end);

System.***out***.println("Time taken: "+ timeElapsed.toMillis() +" milliseconds");

// Without Batch Insert - 551 milliseconds

// With Batch Insert - 518 milliseconds

}

}

**Spring Boot Auto Start Application**

@Component

**public** **class** AppAutoStart {

@Autowired

**private** One2ManyBatchInsert mapping;

@EventListener(ApplicationReadyEvent.**class**)

**public** **void** afterStartup() {

// Save One To One Unidirectional

mapping.save();

}

}

**Spring Boot Main Application**

@SpringBootApplication

@EnableJpaRepositories

**public** **class** SpringOne2OneMain {

**public** **static** **void** main(String[] args) {

SpringApplication.*run*(SpringOne2OneMain.**class**, args);

}

}

Multiple Datasource Configuration

#first db

spring.datasource.url = [url]

spring.datasource.username = [username]

spring.datasource.password = [password]

spring.datasource.driverClassName = oracle.jdbc.OracleDriver

#second db ...

spring.secondDatasource.url = [url]

spring.secondDatasource.username = [username]

spring.secondDatasource.password = [password]

spring.secondDatasource.driverClassName = oracle.jdbc.OracleDriver

Add in any class annotated with @Configuration the following methods:

@Bean

@Primary

@ConfigurationProperties(prefix="spring.datasource")

public DataSource primaryDataSource() {

return DataSourceBuilder.create().build();

}

@Bean

@ConfigurationProperties(prefix="spring.secondDatasource")

public DataSource secondaryDataSource() {

return DataSourceBuilder.create().build();

}