Training Java DAC

Dokumentasi Sesi Ke-2

Dedi Suryadi - Kamis, 15 Desember 2015

Sesi ke-2 pada training kali ini membahas bagaimana cara untuk menambah dependency menggunakan Maven pada sebuah project Spring Boot, menggunakan Java Persistence API (JPA), melakukan koneksi database, membuat entity di class-class java yang secara otomatis terkait dengan table dan column di database dan melakukan test pada class terhadap masing-masing entity-nya di database. Database yang digunakan adalah MySQL dan IDE nya Eclipse. Konteks training: membuat aplikasi pelatihan.

Eclipse Tips:

- Tekan Ctrl+1 untuk mengakses dropdown import
- Tekan Ctrl+Space untuk dropdown autocomplete
- Tekan Shift+Alt+X B untuk run sebagai aplikasi Spring Boot
- Tekan Shift+Alt+X T untuk melakukan unit testing di Test Class yang sedang di edit

1. Menambah Dependency MySQL
   - Di *Package Explorer* double click file *pom.xml*
   - Di window yang baru muncul, pilih tab *Dependecies*
   - Klik tombol *Add…* di sub-kolom *Dependencies*
   - Pada Window *Select Dependency* silahkan ketik mysql
   - Tunggu beberapa saat, lalu muncul daftar library di kolom *Search Result*
   - Klik library *mysql mysql-connector-java (managed)*
   - Lalu tekan tombol *OK*

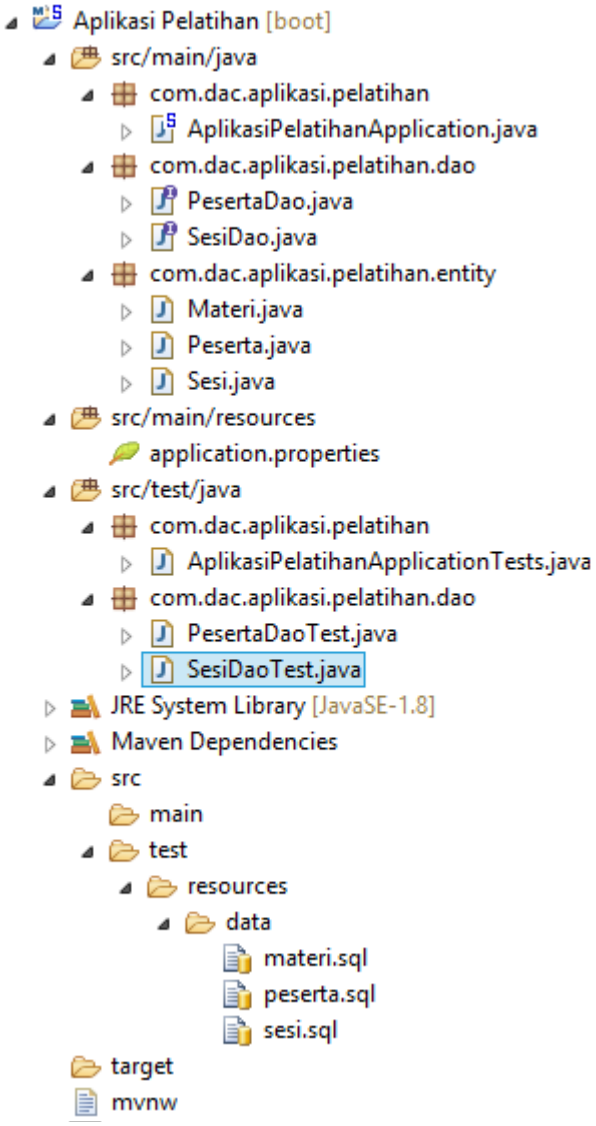


Figure 1. Tampilan Project Eclipse

2. Koneksi Database
   - Jalankan MySQL server, buat database, misalnya: *pelatihan*
   - Catat nama database dan nomor port pada server MySQL yang baru dijalankan
   - Buka *application.properties* yang terletak di *src/main/resources*
   - Masukan settingan sebagai berikut

```
spring.datasource.driverClassName=com.mysql.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/pelatihan
spring.datasource.username=root
spring.datasource.password=
spring.jpa.generate-ddl=true
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

Perhatikan field *spring.datasource.url* silahkan diisi berdasarkan settingan database masing-masing. Pada contoh, port MySQL nya adalah *3306* dan database nya adalah *pelatihan.*
Pada field *spring.datasource.username* dan *spring.datasource.password* silahkan diisi sesuai dengan settingan database masing-masing.

3. Membuat Package Entity

Package *Entity* berisi class-class yang akan digunakan untuk memanage (CRUD) data di database yang sudah kita setting di *application.properties*. Setiap class di dalam package ini mewakili data (table, column, join table dll) di database berdasarkan class dan class property yang sudah di anotasi. Jika class tidak dianotasi, maka maka class tersebut tidak akan tersambung dengan database.

- Buat package baru dengan cara klik kanan folder *src/main/java* di *Package Explorer*
- Nama package bebas, namun usahakan nama package memiliki nama akhir entity untuk memudahkan kita nantinya, Karena di dalam package ini khusus berisi class yang memiliki entity di database. Contoh nama: *com.dac.aplikasi.pelatihan.entity*

4. Membuat Class Peserta.java di Package Entity
   - Klik kanan *com.dac.aplikasi.pelatihan.entity*
   - Klik *New > Class*
   - Namai *Peserta*
   - Klik *OK*
   - Buat property sbb:

```
private String id;
private String nama;
private String email;
private Date tanggalLahir;
```

5. Menambahkan Anotasi pada Class Peserta.java
   - Tambahkan anotasi pada class dan property yang telah dibuat sebelumnya.

```
// Anotasi @Entity menandai kalau class Peserta adalah sebuah tabel di database
// JPA akan mengubah class Peserta menjadi tabel peserta di database (p huruf kecil)
// untuk override nama table, gunakan anotasi:
// @Entity @Table(name = "peserta_pelatihan")
@Entity
public class Peserta {

	// Anotasi @Id menandai kalau properti ini adalah primary key
	@Id
	private String id;

	// Kolom nama tidak boleh null
	@Column(nullable = false)
	private String nama;

	// email tidak boleh kosong dan harus unik
	@Column(nullable = false, unique = true)
	private String email;

	// overide nama kolom di database
	// dan overide Date value pada kolom di database
	// sehingga value yang tersimpan database hanya berisi
	// tanggal, bulan dan tahun, tanpa jam menit dan detik
	@Column(name = "tanggal_lahir", nullable = false)
	@Temporal(TemporalType.DATE)
	private Date tanggalLahir;
}
```

6. Tampilan Import *Peserta.java*

```
import java.util.Date;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
```

7. Generate Getters and Setters
   - Klik kanan di dalam *Peserta.java* editor
   - Klik *Source* > *Generate Getters and Setters*
   - Klik *Select All*
   - *Insertion Point* nya adalah *Last Member*
   - *OK*

8. Membuat Package Data Access Object (DAO)
   Apa itu DAO? Intinya DAO adalah objek atau interface yang menyediakan fungsi untuk melakukan akses ke database atau media penyimpanan. Dengan melakukan extends pada interface DAO maka class atau objek yang kita miliki bisa mengakses database.
   - Klik kanan folder *src/main/java* di *Package Explorer*
   - Klik *New* > *Package*
   - Nama package bebas, namun usahakan nama package memiliki nama akhir *dao* untuk memudahkan kita. Contoh nama: *com.dac.aplikasi.pelatihan.dao*

9. Membuat Interface *PesertaDao.java*
   - Klik kanan *com.dac.aplikasi.pelatihan.dao*
   - Klik *New* > *Interface*
   - Namai *PesertaDao*
   - Klik *OK*

10. Extends PagingAndSortingRepository di *PesertaDao.java Interface*

```
package com.dac.aplikasi.pelatihan.dao;

import org.springframework.data.repository.PagingAndSortingRepository;
import com.dac.aplikasi.pelatihan.entity.Peserta;

public interface PesertaDao extends PagingAndSortingRepository<Peserta, String> {

}
```

11. Membuat *Unit Test* untuk *PesertaDao.java Entity*
    - Buat package baru
    - Klik kanan folder *src/test/java* di *Package Explorer*
    - Klik *New* > *Package*
    - Namai *com.dac.aplikasi.pelatihan.dao*
    - Klik kanan package *com.dac.aplikasi.pelatihan.dao*
    - Klik *New* > *Class*
    - Namai *PesertaDaoTest.java*

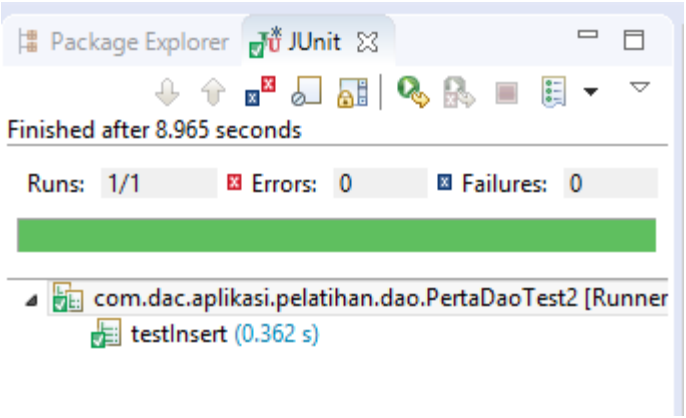
12. Ubah Class *PesertaDaoTest.java* menjadi sbb:

```
@RunWith(SpringRunner.class)
@SpringBootTest
public class PertaDaoTest2 {

	@Autowired
	private PesertaDao pd;

	@Autowired
	private DataSource ds;

	@Test
	public void testInsert() throws SQLException{
		Peserta p = new Peserta();
		p.setId("19");
		p.setNama("Peserta 0019");
		p.setEmail("peserta0019@gmail.com");
		p.setTanggalLahir(new Date());
		pd.save(p);

		String sql = "select count(*) as jumlah "
				+ "from peserta "
				+ "where email = 'peserta0019@gmail.com'";

		try (Connection c = ds.getConnection()) {
			ResultSet rs = c.createStatement().executeQuery(sql);
			Assert.assertTrue(rs.next());
			Long jumlahRow = rs.getLong("jumlah");
			Assert.assertEquals(1L, jumlahRow.longValue());
		}
	}

	@After
	public void hapusData() throws Exception {
		String sql = "delete from peserta where email = 'peserta001@gmail.com'";
		try (Connection c = ds.getConnection()) {
			c.createStatement().executeUpdate(sql);
		}
	}
}
```

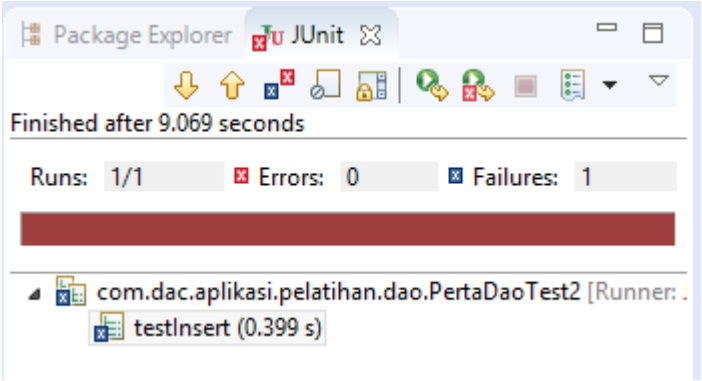
13. Tampilan import di Class *PesertaDaoTest.java*

```
package com.dac.aplikasi.pelatihan.dao;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Date;

import javax.sql.DataSource;

import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.test.context.junit4.SpringRunner;

import com.dac.aplikasi.pelatihan.entity.Peserta;
```

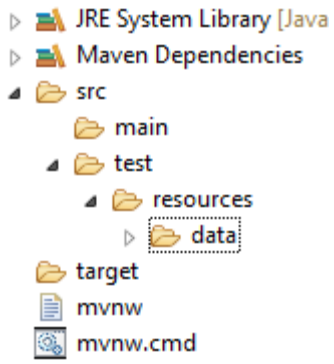
Jalankan dengan cara, Klik kanan di editor *Run As* > *Junit Test* atau dengan shortcut Shift+Alt+X T. Jika berhasil tampilan di sidebar sebelah kiri sbb:



Jika ada error maka tampilan menjadi seperti di samping, pastikan tidak ada error sebelum melanjutkan.



14. Menambah test case pada Class *PesertaDaoTest.java*
    - Buat file *peserta.sql* di folder *src/test/resources/data*
    - Klik kanan di folder *data*
    - *New > File*
    - Namakan *peserta.sql*
    - Isi dengan statement sql sbb:




```
delete from peserta_pelatihan;
delete from peserta;

insert into peserta (id, nama, email, tanggal_lahir)
values ('aa1', 'Peserta Test 001',
'peserta.test.001@gmail.com', '2011-01-01');

insert into peserta (id, nama, email, tanggal_lahir)
values ('aa2', 'Peserta Test 002',
'peserta.test.002@gmail.com', '2011-01-02');

insert into peserta (id, nama, email, tanggal_lahir)
values ('aa3', 'Peserta Test 003',
'peserta.test.003@gmail.com', '2011-01-03');
```

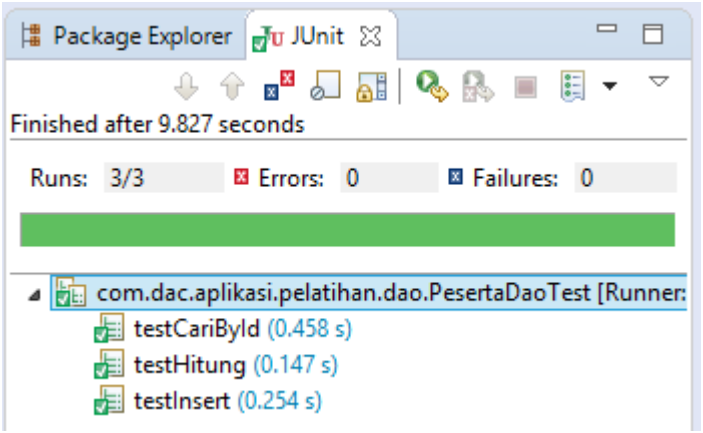
- Ubah anotasi pada Class *PesertaDaoTest.java* menjadi sbb:

```
@RunWith(SpringRunner.class)
@SpringBootTest
@Sql(
                executionPhase =
Sql.ExecutionPhase.BEFORE_TEST_METHOD,
                scripts = "/data/peserta.sql"
)
public class PesertaDaoTest {
```

- Sebelum anotasi *@After* masukan tambahan test sebagai berikut:

```
@Test
public void testHitung(){
        Long jumlah  = pd.count();
        Assert.assertEquals(3L, jumlah.longValue());
}

@Test
public void testCariById() {
        Peserta p = pd.findOne("aa1");
        Assert.assertNotNull(p);
        Assert.assertEquals("Peserta Test 001", p.getNama());
        Assert.assertEquals("peserta.test.001@gmail.com", p.getEmail());

        Peserta px = pd.findOne("xx");
        Assert.assertNull(px);
}
```

- Jika tidak ada error maka tampilan sidebar Junit akan menjadi sebagai berikut:



15. Membuat Entity Materi
    - Klik kanan *com.dac.aplikasi.pelatihan.entity*
    - Klik *New* > *Class*
    - Namakan *Materi*
    - Klik *OK*
    - Isi file *Materi.java* menjadi sebagai berikut:

```
@Entity
public class Materi {

	@Id @GeneratedValue(generator = "uuid")
	@GenericGenerator(name = "uuid", strategy = "uuid2")
	private String id;

	@Column( nullable = false, unique = true, length = 10)
	private String kode;

	@Column(nullable = false)
	private String nama;

	@OneToMany(
		cascade = CascadeType.ALL,
		orphanRemoval = true,
		mappedBy = "materi"
	)
	private List<Sesi> daftarSesi = new ArrayList<>();

}
```

- Berikut tampilan importnya

```
package com.dac.aplikasi.pelatihan.entity;

import java.util.ArrayList;
import java.util.List;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.OneToMany;
import org.hibernate.annotations.GenericGenerator;
```

- Jangan lupa untuk generate *Getters and Setters* nya juga

16. Membuat Entity Sesi
    - Klik kanan *com.dac.aplikasi.pelatihan.entity*
    - Klik *New > Class*
    - Namakan *Sesi*
    - Klik *OK*
    - Isi file *Sesi.java* menjadi sebagai berikut:

```
@Entity
public class Sesi {
	@Id @GeneratedValue(generator="uuid")
	@GenericGenerator(name = "uuid", strategy = "uuid2")
	private String id;

	@Temporal(TemporalType.DATE)
	private Date mulai;

	@Temporal(TemporalType.DATE)
	private Date sampai;

	@ManyToOne
	@JoinColumn(name = "id_materi", nullable = false)
	private Materi materi;

	@ManyToMany
	@JoinTable(
		name = "peserta_pelatihan",
		joinColumns = @JoinColumn(name = "id_sesi"),
		inverseJoinColumns = @JoinColumn(name = "id_peserta")
	)
	private List<Peserta> listPeserta = new ArrayList<>();
}
```

- Berikut ini tampilan import nya

```
package com.dac.aplikasi.pelatihan.entity;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.ManyToOne;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;
import org.hibernate.annotations.GenericGenerator;
```

- Seperti biasa jangan lupa untuk generate *Getters and Setters* di class ini.

17. Membuat Interface *SesiDao.java*
    - Klik kanan *com.dac.aplikasi.pelatihan.dao*
    - Klik *New* > *Interface*
    - Namakan *SesiDao*
    - Klik *OK*
    - Extends PagingAndSortingRepository di *SesiDao.java Interface*
    - Lalu tambahkan Custom method untuk unit testing

```
package com.dac.aplikasi.pelatihan.dao;

import java.util.Date;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.PagingAndSortingRepository;
import org.springframework.data.repository.query.Param;
import com.dac.aplikasi.pelatihan.entity.Materi;
import com.dac.aplikasi.pelatihan.entity.Sesi;

public interface SesiDao extends PagingAndSortingRepository<Sesi, String>{
	public Page<Sesi> findByMateri(Materi m, Pageable page);

	@Query("select x from Sesi x where x.mulai >= :m "
			+ "and x.mulai < :s "
			+ "and x.materi.kode = :k "
			+ "order by x.mulai desc ")
	public Page<Sesi> cariBerdasarkanTanggalMulaiDanKodeMateri(
			@Param("m") Date mulai,
			@Param("s") Date sampai,
			@Param("k") String kode,
			Pageable page);
}
```

18. Membuat file yang berisi sql statement untuk keperluan unit testing Entity Sesi dan Materi
    - Klik kanan folder *data* yang terletak di *src/test/resources/data*
    - *New > File*
    - Namakan *materi.sql*
    - Isi dengan statement sql sbb:

```
delete from sesi;
delete from materi;

insert into materi (id, kode, nama)
values ('aa6', 'JF-001', 'Java Fundamental');

insert into materi (id, kode, nama)
values ('aa7', 'JF-002', 'Java Web');

insert into materi (id, kode, nama)
values ('aa8', 'MB-001', 'IOS Fundamental');

insert into materi (id, kode, nama)
values ('aa9', 'MB-002', 'Android Fundamental');
```

    - Klik kanan folder *data*
    - *New > File*
    - Namakan *sesi.sql*
    - Isi dengan statement sql sbb:

```
delete from peserta_pelatihan;
delete from sesi;

insert into sesi (id, id_materi, mulai, sampai)
values ('aa', 'aa6', '2015-01-01', '2015-01-05');

insert into sesi (id, id_materi, mulai, sampai)
values ('ab', 'aa7', '2015-01-08', '2015-01-15');

insert into sesi (id, id_materi, mulai, sampai)
values ('ac', 'aa8', '2015-01-01', '2015-01-25');

insert into peserta_pelatihan (id_sesi, id_peserta) values ('aa', 'aa1');
insert into peserta_pelatihan (id_sesi, id_peserta) values ('aa', 'aa2');
insert into peserta_pelatihan (id_sesi, id_peserta) values ('aa', 'aa3');
insert into peserta_pelatihan (id_sesi, id_peserta) values ('ab', 'aa2');
insert into peserta_pelatihan (id_sesi, id_peserta) values ('ab', 'aa3');
insert into peserta_pelatihan (id_sesi, id_peserta) values ('ac', 'aa2');
```

19. Membuat *Unit Test* untuk *SesiDao.java Entity*
    - Klik kanan package *com.dac.aplikasi.pelatihan.dao*
    - Klik *New* > *Class*
    - Namai *SesiDaoTest.java*
    - Masukan kode berikut

```
package com.dac.aplikasi.pelatihan.dao;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.sql.DataSource;
import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.test.context.jdbc.Sql;
import org.springframework.test.context.junit4.SpringRunner;
import com.dac.aplikasi.pelatihan.dao.SesiDao;
import com.dac.aplikasi.pelatihan.entity.Materi;
import com.dac.aplikasi.pelatihan.entity.Peserta;
import com.dac.aplikasi.pelatihan.entity.Sesi;

@RunWith(SpringRunner.class)
@SpringBootTest
@Sql(
        executionPhase = Sql.ExecutionPhase.BEFORE_TEST_METHOD,
        scripts = {"/data/peserta.sql", "/data/materi.sql", "/data/sesi.sql"}
)
public class SesiDaoTest {

        @Autowired
        private SesiDao sd;

        @Autowired
        private DataSource ds;

        @Test
        public void testCariByMateri() {
                Materi m = new Materi();
                m.setId("aa6");

                PageRequest page = new PageRequest(0, 5);

                Page<Sesi> hasilQuery = sd.findByMateri(m, page);
                Assert.assertEquals(1L, hasilQuery.getTotalElements());

                Assert.assertFalse(hasilQuery.getContent().isEmpty());
                Sesi s = hasilQuery.getContent().get(0);
                Assert.assertNotNull(s);
                Assert.assertEquals("Java Fundamental", s.getMateri().getNama());
        }

}
```

- Tambahkan method unit testing berikut ke Class *SesiDaoTest.java*

```
@Test
public void testCariBerdasarkanTanggalMulaiDanKodeMateri() throws Exception {
	PageRequest page = new PageRequest(0, 5);
	SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
	Date sejak = formatter.parse("2015-01-01");
	Date sampai = formatter.parse("2015-01-05");

	Page<Sesi> hasil = sd.cariBerdasarkanTanggalMulaiDanKodeMateri(
			sejak, sampai, "JF-001", page);

	Assert.assertEquals(1L, hasil.getTotalElements());
	Assert.assertFalse(hasil.getContent().isEmpty());
	Sesi s = hasil.getContent().get(0);
	Assert.assertEquals("Java Fundamental", s.getMateri().getNama());

}

@Test
public void testSaveSesi() throws Exception {
	Peserta p1 = new Peserta();
	p1.setId("aa1");
	Peserta p2 = new Peserta();
	p2.setId("aa2");
	Materi m = new Materi();
	m.setId("aa8");

	SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
	Date sejak = formatter.parse("2015-02-01");
	Date sampai = formatter.parse("2015-02-03");

	Sesi s = new Sesi();
	s.setMateri(m);
	s.setMulai(sejak);
	s.setSampai(sampai);
	s.getListPeserta().add(p1);
	s.getListPeserta().add(p2);
	sd.save(s);

	String idSesiBaru = s.getId();
	Assert.assertNotNull(idSesiBaru);
	System.out.println("ID Baru : " + s.getId());
	String sql = "select count(*) from sesi where id_materi='aa8'";
	String sqlManyToMany = "select count(*) from peserta_pelatihan "
			+ "where id_sesi = ?";

	try (Connection c = ds.getConnection()) {
		ResultSet rs = c.createStatement().executeQuery(sql);
		Assert.assertTrue(rs.next());
		Assert.assertEquals(2L, rs.getLong(1));

		PreparedStatement ps = c.prepareStatement(sqlManyToMany);
		ps.setString(1, idSesiBaru);
		ResultSet rs2 = ps.executeQuery();

		Assert.assertTrue(rs2.next());
		Assert.assertEquals(2L, rs2.getLong(1));
	}
}
```

20. Run unit testing di class *SesiDaoTest.java*, jika sukses maka akan muncul tampilan sebagai berikut: