■스프링데이터베이스 연동 실습문제

프로젝트 준비하기
- 스프링 프로젝트만들기
  pom.xml 변경

스프링버전 및 자바버전 변경
√ java-version : 11변경
√ sprintg-version : 5.0.7 변경

dependency 추가
-- spring db를 위한
spring -jdbc
ojdbc8
-- spring test를 위한
spring-test
junit 4.12 변경

■ 데이터베이스 연동을 위한 DataSource 사용하기
DataSource 사용하기
스프링이 제공하는 DriverManagerDataSource 사용하기 (DataSource 인터페이스를 구혀한 구현체임)
connection pool기법을 사용하여 데이터베이스 연결정보를 미리 여러개 가지고 있다가 연결이 필요할 때 제공함

springdatasource bean생성

```
<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver"></property>
        <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe"></property>
        <property name="username" value="System"></property>
        <property name="password" value="1234"></property>
</bean>
```

2인 1조 과제수행하기

1. datasource 를 이용해 connection 얻어오기 테스트하기
2. MemberDAO 만들기
   - 상품정보등록하기

```
int insertMember ( Member member) ;          //회원등록
int updateMember( Member member);            //회원변경
ArrayList<Member> findAll( ) ;                 //회원목록조회
Member findOne( String id) ;                   // 회원조회
int deleteOne ( String id) ;                          // 회원삭제
```

고객테이블 생성하기

```
create table member_acorn(
    id    varchar2(10) primary key ,
    pwd   varchar2(10) ,
   name  varchar2(20) ,
   email  varchar2(20),
   birth   date   ,
   sns    varchar2(20) ,
   reg_date  timestamp
);
```

고객정보를 담을 Member

```
public class Member {
    private String id;
    private String pwd;
    private String name;
    private String email;
    private String birth;
    private String sns;
    private Date reg_date;

    //setter, getter 작성
    // toString    작성
}
```

```
@Component
public class MemberDAO {
    @Autowired
    DataSource ds;

 public int updateUser(Member member)    {
        System.out.println( member);
        int rowCnt =0;
        String sql =          " update member_acorn " +
                              "   set pwd = ?, name=?, email=?, birth =?, sns=?, reg_date=? " +
                              "   where id = ? ";
        Connection conn =null;
        PreparedStatement pstmt= null;
        System.out.println("sql =" +   sql);


        try {
            conn = ds.getConnection();
            System.out.println( conn );
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, member.getPwd());
            pstmt.setString(2, member.getName());
            pstmt.setString(3, member.getEmail());
            pstmt.setString(4, member.getBirth());
            pstmt.setString(5, member.getSns());
            pstmt.setTimestamp(6, new java.sql.Timestamp(member.getReg_date().getTime()));
            pstmt.setString(7, member.getId());
            rowCnt = pstmt.executeUpdate();
            System.out.println( rowCnt);

        } catch (SQLException e) {
            e.printStackTrace();
            return 0;
        }finally {
           close( pstmt, conn);
        }
        return rowCnt;
    }

    private void close(AutoCloseable... acs) {
        for(AutoCloseable ac :acs)
            try { if(ac!=null) ac.close(); } catch(Exception e) { e.printStackTrace(); }
    }
}
```

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations =
{"file:src/main/webapp/WEB-INF/spring/**/test.xml" ,
"file:src/main/webapp/WEB-INF/spring/**/test2.xml"} )
```

고객테이블 관련 sql 작성, 테스트하기
회원정보 삭제
회원정보 변경
회원정보 등록
회원정보 조회
select id, pw, name, email, birth , sns, to_char(reg_date, 'yyyy-mm-dd hh24:mi:ss') from member_acorn;

3. MemberDAO 테스트하기
-회원등록시 int result = pst.executeUpdate() ; 반환값호 (추가된 레코드 수) 1이 나오지는지 테스트
-회원변경시 변경된 레코드수가 1인지 확인 테스트
-회원삭제시 삭제된 레코드수가 1인지 확인 테스트
-회원조회시 결과가 null이 아니지 확인
-회원one조회시 조회된 고객id가 해당 id가 맞는지 확인 테스트

고객 등록, 변경시 sample data

```
Member member= new Member();
member.setId("test");
member.setPwd("1234");
member.setName("hong");
member.setEmail("hong.email.com");
member.setBirth("2009-02-01");
member.setSns("kakao");
member.setReg_date(new Date());
dao.updateUser(member);
```

테스트클래스작성하기

4. MemberServer 만들기
- 고객전체목록조회하기
ArrayList<Member> getMembers() ;
- 고객정보 조회하기
Member getMember(String id) ;
- 고객정보 변경하기
int chageInfoMember(Member member) ;
- 고객등록하기
int registerMember(Member member)
- 고객정보삭제하기
int deleteMember(String id) ;

5. MemberService 테스트하기