http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:rte2:brte:batch_example:db_jdbcpaging

JdbcPaging Examples

Outline

This Section describes how the data are processed by reading DB in by unit of page. In Spring the examples of JdbcPagingItemReader and JpaPagingItemReader, the paging query interface PagingQueryProvider, and such implementation classes as OraclePagingQueryProvider, HsqlPagingQueryProvider, MySqlPagingQueryProvider, SqlServerPagingQueryProvider and SybasePagingQueryProvider are available.

Description

Settings

Configuring Jobs

Check out jdbcPagingIoJob.xml, the job configuration file for JdbcPaging example.

The following configurations are available in JdbcPagingItemReader:

- dataSource : Database
- rowMapper : Mapper of ResultSet, the result of SQL execution
- queryProvider : Provides queries for paging
- pageSize : Page size in terms of data
- parameterValues : Parameter values of query designated as Map.

```
<bean id="itemReader" class="org.springframework.batch.item.database.JdbcPagingItemReader" scope="step">
        <property name="dataSource" ref="dataSource" />
        <property name="rowMapper">
                <bean class="egovframework.brte.sample.common.domain.trade.CustomerCreditRowMapper" />
        </property>
        <property name="queryProvider">
                <bean
class="org.springframework.batch.item.database.support.SqlPagingQueryProviderFactoryBean">
                        <property name="dataSource" ref="dataSource" />
                        <property name="sortKey" value="ID" />
                        <!-- Intentionally put sort key second in the query list as a test -->
                        <property name="selectClause" value="select NAME, ID, CREDIT" />
                        <property name="fromClause" value="FROM CUSTOMER" />
                        <property name="whereClause" value="WHERE CREDIT > :credit" />
                </bean>
        </property>
        <property name="pageSize" value="2" />
        <property name="parameterValues">
                <map>
                        <entry key="statusCode" value="PE" />
                        <entry key="credit" value="#{jobParameters[credit]}" />
                        <entry key="type" value="COLLECTION" />
                </map>
        </property>
</bean>
```

Composition and Implementation of JunitTest

Composition of JunitTest

Implement the Jdbcpaging example and verify the result of batch by comprising @Test as follows:

✔ See Junit Test Description for Batch Execution for more information.
✔ The parameter information required for querying of JobParameter in getUniqueJobParameters is available.
✔ Pre-batch data and post-batch data are to be compared to each other in EgovAbstractIoSampleTests.
✔ assertEquals(BatchStatus.COMPLETED, jobExecution.getStatus()): Check out the batch implementation is COMPLETED.
✔ Note that your Altibase version should be 6.1.1 or newer.

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(locations = "/egovframework/batch/jobs/jdbcPagingIoJob.xml")
public class EgovJdbcPagingFunctionalTests extends EgovAbstractIoSampleTests {
	// SimpleJdbcTemplate
	private SimpleJdbcTemplate simpleJdbcTemplate;

	/**
	 * Pre-test DB works
	 */
	@Before
	public void setUp() {
		simpleJdbcTemplate.update("DELETE from CUSTOMER");

		simpleJdbcTemplate.update("INSERT INTO CUSTOMER (ID, VERSION, NAME, CREDIT)
VALUES	(1, 0, 'customer1', 100000)");
		simpleJdbcTemplate.update("INSERT INTO CUSTOMER (ID, VERSION, NAME, CREDIT)
VALUES	(2, 0, 'customer2', 100000)");
		simpleJdbcTemplate.update("INSERT INTO CUSTOMER (ID, VERSION, NAME, CREDIT)
VALUES	(3, 0, 'customer3', 100000)");
		simpleJdbcTemplate.update("INSERT INTO CUSTOMER (ID, VERSION, NAME, CREDIT)
VALUES	(4, 0, 'customer4', 100000)");

	}

	@Override
	protected JobParameters getUniqueJobParameters() {
		return new JobParametersBuilder(super.getUniqueJobParameters())
				.addDouble("credit", 10000.).toJobParameters();
	}

}
@ContextConfiguration(locations = { "/egovframework/batch/simple-job-launcher-context.xml",
"/egovframework/batch/job-runner-context.xml"})
@TestExecutionListeners( { DependencyInjectionTestExecutionListener.class,
StepScopeTestExecutionListener.class })
public abstract class EgovAbstractIoSampleTests {

	// JobLauncherTestUtils to test the batches.
	@Autowired
	@Qualifier("jobLauncherTestUtils")
	private JobLauncherTestUtils jobLauncherTestUtils;

	// Reader for batches
	@Autowired
	private ItemReader<CustomerCredit> reader;

	/**
	 * Batch Testing
	 */
	@Test
	public void testUpdateCredit() throws Exception {
```

```
            open(reader);
            List<CustomerCredit> inputs = getCredits(reader);
            close(reader);

            JobExecution jobExecution = jobLauncherTestUtils.launchJob(getUniqueJobParameters());
            assertEquals(BatchStatus.COMPLETED, jobExecution.getStatus());

            pointReaderToOutput(reader);
            open(reader);
            List<CustomerCredit> outputs = getCredits(reader);
            close(reader);

            assertEquals(inputs.size(), outputs.size());
            int itemCount = inputs.size();
            assertTrue(itemCount > 0);

            for (int i = 0; i < itemCount; i++) {

        assertEquals(inputs.get(i).getCredit().add(CustomerCreditIncreaseProcessor.FIXED_AMOUNT).intValue(),
                                                    outputs.get(i).getCredit().intValue());
            }

        }

    ...
}
```

Implementation of JunitTest

See [Implementation of JunitTest](#) for more information.

Verification of Result

1. You can check out the paging for DB in the query of console log. With the pagesize set '2', a pair of data are read out of DB.

```
Tasks | Properties | Servers | Coverage | Bookmarks | Console | Javadoc | SVN Repositories | SQL Results | Execution Plan | DBIO Search | Query Result
<terminated> EgovJdbcPagingFunctionalTests [JUnit] C:₩eGovFrame-2.0₩bin₩jdk1.5.0_22₩bin₩javaw.exe (2012. 9. 24. 오후 2:33:56)
14:34:00,232 DEBUG main RepeatTemplate:463 - Starting repeat context.
14:34:00,232 DEBUG main RepeatTemplate:365 - Repeat operation about to start at count=1
14:34:00,232 DEBUG main JdbcPagingItemReader:104 - Reading page 1
14:34:00,232 DEBUG main JdbcPagingItemReader:214 - SQL used for reading remaining pages: [SELECT TOP 2 NAME, ID, CREDIT FROM CUSTOMER
14:34:00,232 DEBUG main JdbcPagingItemReader:289 - Using parameterMap:{statusCode=PE, credit=10000.0, type=COLLECTION, _sortKey=2}
14:34:00,232  INFO main audit:156 - 2. Connection.prepareStatement(SELECT TOP 2 NAME, ID, CREDIT FROM CUSTOMER WHERE CREDIT > ? AND I
14:34:00,232  INFO main audit:156 - 2. PreparedStatement.setObject(1, 10000.0) returned
14:34:00,232  INFO main audit:156 - 2. PreparedStatement.setObject(2, 2) returned
14:34:00,248  INFO main audit:156 - 2. PreparedStatement.setMaxRows(2) returned
14:34:00,248  INFO main sqlonly:191 - SELECT TOP 2 NAME, ID, CREDIT FROM CUSTOMER WHERE CREDIT > 10000.0 AND ID > 2 ORDER BY ID ASC
```

2. You can check out the job is implemented and modified in the value Credit in the table Customer of DB.

Status | Result1

	ID	VERSION	NAME	CREDIT
1	1	0	customer1	100005
2	2	0	customer2	100005
3	3	0	customer3	100005
4	4	0	customer4	100005

References

- JdbcPagingItemReader