Swinburne University of Technology
Faculty of Information and Communication Technologies

HIT3304/HIT8304: Database Programming

Spike Assessments

Overview
Spike assessments are part of the core assessment for this subject.

In this assessment you will complete a number of spikes, organised into sets, that demonstrate technologies and tools related to the subject material. Spike sets will be due weekly from Academic Week 4 to Week 9 and must all be passed satisfactorily in order to pass this subject.

Spike work is team based, however each team member will be assessed individually and will need to create individual reports. Each week your team will present their completed spikes in the laboratory and you will be required to answer questions related to ALL spikes.

Teams
The spikes are performed in pairs. You must form your teams in the laboratory sessions. Once you have found a partner you must decide upon a domain or topic area. This needs to be unique to your tutorial group. You will then use this domain for all spikes performed during the semester.

If you need to change partners for any reason you will need to determine which team member will keep the domain and which member will find a new domain.

You can work individually on these spikes. In this case you will need to perform all spikes each week.

Instructions
Complete each weeks spikes and demonstrate the outcomes to your tutor each week from Week 4 to Week 9. Spike plans are contained in this document, organised into numbered sets, programming language and technologies.

Submission Instructions
By Week 3 you must find a partner and submit a domain area that you will be working with.

Each week you will need to prepare the following:
1. Printed spike outcome report for each spike (see Spike Outcome Template)
2. Spike deliverables as expressed on the spike
3. Answer questions related to all spikes

Timetable

Start and Due dates are in reference to academic weeks. See subject outline for specific dates.

Set	Java Topic	C# Topic	Start	Due
1	SQLite		1	3
2	Basic JDBC	Basic ADO.NET	2	4
3	JDBC RowSet & CachedRowSet	ADO.Net DataSet & Relations	3	5
4	Hibernate (ORM)	NHibernate & Linq To SQL (ORM)	4	6
5	Application Layers (packages)	Application Layers (namespaces/dlls's)	5	7
6	Web Applications with Tomcat, JSP's & .war files	Web Applications with ASP.NET	7	8
7	Hibernate Transactions	ADO.NET Transactions	8	9

Additional Spike Notes:

For more information about the concept and purpose of a Spike see:
http://mercury.it.swin.edu.au/swinbrain/index.php/Spike

For a Spike outcome template see the last page of this document, or the word document at
http://mercury.it.swin.edu.au/swinbrain/images/b/bc/SpikeOutcomeTemplate.doc

Spike Plan

Set: 1
Language: NA
Name: Database Creation Spike

Context:
Team needs to be able to work with the SQLite DBMS as an embedded database. Specifically the team needs to be able to install Derby, and create and populate newly created databases.

Knowledge Gap:
Team needs to learn to install and use SQLite.

Goals/Deliverables:
1. Steps needed to install SQLite on a developer machine
2. Steps to create a database manually
3. Steps to populate the database manually

Planned Start:	Week 1
Deadline:	Week 3

Planning notes:
1. Install and configure SQLite – document steps
2. Create a simple database using SQLite – document steps

Spike Plan

Set: 1
Language: NA
Name: Database Creation Automation

Context:
Team needs to use a database to store data for application developed during this subject. Developing the database application will require fresh databases upon which testing can be performed. In order to accelerate the process of creating and populating new databases the team needs to know how to script these processes.

Knowledge Gap:
The team is not familiar with scripting database creation.

Goals/Deliverables:

1. Sample database schema – ERD and creation scripts
2. Command line scripts to create and populate database
3. Command line scripts to clean up database

Planned Start:	Week 1
Deadline:	Week 3

Planning notes:

1. Design database tables for the team's application domain.
2. Read up on command line scripting

Spike Plan

Set: 2
Language: Java
Name: Basic JDBC Spike

Context:

Database programming involves connecting to the database, and issuing commands to retrieve, insert, update, and delete data. JDBC is the fundamental database technology used in Java applications.

Knowledge Gap:

The team is not familiar with the JDBC programming model.

Goals/Deliverables:

1. Command line application that demonstrates
   a. Connecting to an SQLite database (embedded)
   b. Inserting records into the database
   c. Deleting records from the database
   d. Updating records in the database
   e. Selecting records from the database
   f. Selecting records based upon a condition/filter

Planned Start: Week 2
Deadline: Week 4

Planning notes:

1. Create database and populate with initial test data (1 table)
2. Write initial Ant script for building and running application
3. Develop Java class and test connection to database
4. Create statement to read data from database, write results to the console. Use **list** command line argument to perform this test.
5. Write code to insert records into the database. Use the **insert** command line argument to perform this test. Request information from the user via the console.
6. Write code to update a record in the database. Use the **update** command line argument to perform this test. Request additional information from the user via the console.
7. Write code to delete a record from the database. Use the **delete** command line argument to perform this test. Request additional information from the user via the console.
8. Complete the application by writing code to list a selected record. Use the **find** command line argument to perform this test. Request additional information from the user via the console.

Spike Plan

Set: 2
Language: C#
Name: ADO.NET Spike

Context:
Database programming involves connecting to the database, and issuing commands to retrieve, insert, update, and delete data. ADO.NET is the fundamental database technology in the .NET framework.

Knowledge Gap:
The team is not familiar with the ADO.NET programming model.

Goals/Deliverables:
1. Command line application that demonstrates
    a. Connecting to a SQLite database (Embedded)
    b. Inserting records into the database
    c. Deleting records from the database
    d. Updating records in the database
    e. Listing records from the database
    f. Selecting records based upon a condition/filter

Planned Start: Week 2
Deadline: Week 4

Planning notes:
1. Create database and populate with initial test data (1 table)
2. Create a solution and project in Visual Studio to house this program, or create folder structure and write makefile/MSBuild script.
3. Develop C# class and test connection to database
4. Write code to read data from database, get application to write to console. Use **list** command line argument to perform this test.
5. Write code to insert records into the database. Use the **insert** command line argument to perform this test. Request information from the user via the console.
6. Write code to update a record in the database. Use the **update** command line argument to perform this test. Request additional information from the user via the console.
7. Write code to delete a record from the database. Use the **delete** command line argument to perform this test. Request additional information from the user via the console.
8. Complete the application by writing code to list the details of a selected record. Use the **find** command line argument to perform this test. Request additional information from the user via the console.

Spike Plan

Set: 3
Language: Java
Name: RowSet Spike

Context:

While many Java projects opt to use ORM tools, there are those that follow the .NET DataSet style model. This approach takes advantage of the RowSet and CachedRowSet classes from JDBC.

Knowledge Gap:

The team is not familiar with using JDBC RowSet classes.

Goals/Deliverables:

1. Command line application that demonstrates
   a. Selecting records into a RowSet and CachedRowSet
   b. Making changes to a RowSet and persisting to the database (insert, update, delete)

Planned Start:	Week 4
Deadline:	Week 6

Planning notes:

1. Create database and populate with initial test data (1 table)
2. Implement code to perform the following
   a. List records from database
   b. Insert a new record
   c. Update a record
   d. Delete a record

Spike Plan

Set: 3
Language: C#
Name: DataSet Spike

Context:
While Nhibernate provides a means of performing object-relational mapping with .NET, it is not the primary tool used for .NET applications. The .NET framework primarily uses DataSets as a means of interacting with the database.

Knowledge Gap:
The team is not familiar with using ADO.NET's DataSets.

Goals/Deliverables:

1. Command line application that demonstrates
    a. Selecting records into a DataSet
    b. Making changes to a DataSet and persisting to the database (insert, update, delete)
    c. Examine DataRelations and working with related rows

Planned Start:	Week 4
Deadline:	Week 6

Planning notes:

1. Create database and populate with initial test data (2 tables)
2. Implement code to perform the following
    a. List records from database
    b. List one record, and its associated records from a related table (one to many)
    c. Insert a new record
    d. Update a record
    e. Delete a record

Spike Plan

Set: 4
Language: Java
Name: Hibernate Spike

Context:
Object Relational Mapping (ORM) tools provide a convenient way of persisting objects with the database. Hibernate is a popular ORM tool for Java.

Knowledge Gap:
The team is not familiar with the use of Hibernate.

Goals/Deliverables:
1. Command line application that demonstrates
   a. Retrieving an object using Hibernate
   b. Retrieving related objects (one-to-many collection mapping)
   c. Inserting records with Hibernate
   d. Updating records with Hibernate
   e. Deleting records with Hibernate

Planned Start: Week 3
Deadline: Week 5

Planning notes:
1. Create database and populate with initial test data (2 tables)
2. Create hibernate configuration file with database connection details
3. Create classes to map to database
4. Create mapping files
5. Implement code to perform the following
   a. List records from database
   b. List one record, and its associated records from a related table (one to many)
   c. Insert a new record
   d. Update a record
   e. Delete a record

Spike Plan

Set: 4
Language: C#
Name: NHibernate and Linq to SQL Spike

Context:

Object Relational Mapping (ORM) tools provide a convenient way of persisting objects with the database. NHibernate is a .NET port of the popular ORM tool for Java, while Linq to SQL was a Microsoft initiative for ORM with .NET that has now been adopted by the open source community.

Knowledge Gap:

The team is not familiar with the use of NHibernate or Linq To SQL.

Goals/Deliverables:

1. Command line application that demonstrates the following using NHibernate and Linq To SQL
    a. Retrieving an object
    b. Retrieving related objects (one-to-many collection mapping)
    c. Inserting records
    d. Updating records
    e. Deleting records

Planned Start:	Week 3
Deadline:	Week 5

Planning notes:

1. Create database and populate with initial test data (2 tables)
2. Create hibernate configuration file with database connection details
3. Create classes to map to database
4. Create mapping files
5. Implement code to perform the following
    a. List records from database
    b. List one record, and its associated records from a related table (one to many)
    c. Insert a new record
    d. Update a record
    e. Delete a record

Spike Plan

Set: 5
Language: Java
Name: Java Layer Spike

Context:
Database applications typically consist of a number of Layers, usually consisting of a data access layer, a business layer, and a user interface layer.

Knowledge Gap:
The team has not implemented layered applications before.

Goals/Deliverables:

1. Command line application that is constructed from a
    a. User interface layer
    b. Business/Application Layer
    c. Data Access Layer (DAL)
2. Constrains:
    a. UI communicates with Business Layer via an Interface(s)
    b. Business Layer accesses Data Access Layer via Interfaces
    c. Separate layers into different packages

Planned Start:	Week 5
Deadline:	Week 7

Planning notes:

1. Create a Data Access Layer that uses Hibernate to fetch objects
2. Create a Service Interface within the Business Layer code
    a. It fetches objects from the DAL and executes requested functionality
    b. This will include “Manager” objects
3. Implement some basic business logic within your model (Business Layer)
    c. E.g. The code needed to add a Review, with validation etc.
4. Implement a console user interface driven by command line arguments only
    d. Use command line arguments to initiate commands, hard code values as needed

Spike Plan

Set: 5
Language: C#
Name: C# Layer Spike

Context:
Database applications typically consist of a number of Layers, usually consisting of a data access layer, a business layer, and a user interface layer.

Knowledge Gap:
The team has not implemented layered applications before.

Goals/Deliverables:
1. Command line application that is constructed from a
   a. User interface layer
   b. Business/Application Layer
   c. Data Access Layer (DAL)
2. Constrains:
   a. UI communicates with Business Layer via an Interface(s)
   b. Business Layer accesses Data Access Layer via Interfaces
   c. Separate layers into different namespaces and DLLs

Planned Start:	Week 5
Deadline:	Week 7

Planning notes:
1. Create a Data Access Layer that uses ADO.NET to fetch DataSets
   a. This can be implemented as a number of Table Adapters within your DAL
2. Create a Service Interface within the Business Layer code
   a. It fetches DataSets from the DAL and executes requested functionality
   b. This will include “Manager” objects
3. Implement some basic business logic within your model (Business Layer)
   a. E.g. The code needed to add a Review, with validation etc.
4. Implement a console user interface driven by command line arguments only
   a. Use command line arguments to initiate commands, hard code values as needed

Spike Plan

Set: 6
Language: Java
Name: Tomcat JSPs

Context:

We need to build web applications using Java with Java Server Pages (JSPs). Tomcat is an open source web server that is capable of processing JSPs.

Knowledge Gap:

The team has not worked with Tomcat or JSPs.

Goals/Deliverables:

1. Web application deployed on Tomcat
    a. Ant scripts to build the web application
    b. Ant targets to generate war
    c. Web site must show a list of records from the database
    d. Provide the ability to get information from the user and insert into the database

Planned Start:	Week 7
Deadline:	Week 8

Planning notes:

1. Investigate the installation and operation of Tomcat
2. Create a basic JSP to hook to existing backend code for fetching records
3. Investigate Ant scripts for creating war files and deploying to tomcat

Spike Plan

Set: 6
Language: C#
Name: ASP.NET Spike

Context:
We need to build web applications using C# and ASP.NET.

Knowledge Gap:
The team has not developed any ASP.NET applications.

Goals/Deliverables:
1. Web application using ASP.NET
    a. Web site must show a list of records from the database
    b. Supply a page to collect details from the user and insert into the database
    c. Must use existing backend code to interact with database

Planned Start:	Week 7
Deadline:	Week 8

Planning notes:
1. Investigate the creation of ASP.NET pages
2. Hook ASP.NET pages to backend objects which will interact with the database

Spike Plan

Set: 7
Language: Java
Name: Hibernate Transactions

Context:
Multiple users will access our web applications at the same time. We need to ensure that the changes we make to the database work correctly in this environment.

Knowledge Gap:
Need to understand how to use manage concurrent access with Hibernate.

Goals/Deliverables:
1. Console application that can be used to demonstrate concurrent access to the database
    a. Must perform a combination of updates on the database
    b. Transactions ensure all or no updates are performed
    c. Illustrate concurrent access with stale data (read and hold records while changed in a separate process)

Planned Start: Week 8
Deadline: Week 9

Planning notes:
1. Start with transactions surrounding multiple actions (in data access layer)
2. Develop console application to illustrate concurrent access issues

Spike Plan

Set: 7
Language: C#
Name: ADO.NET Transactions

Context:
Multiple users will access our web applications at the same time. We need to ensure that the changes we make to the database work correctly in this environment.

Knowledge Gap:
Need to understand how to use ADO.NET transactions and optimistic concurrency to manage concurrent updates.

Goals/Deliverables:

1. Console application that can be used to demonstrate concurrent access to the database
    a. Must perform a combination of updates on the database
    b. Transactions ensure all or no updates are performed
    c. Illustrate concurrent access with stale data (read and hold records while changed in a separate process)

Planned Start:	Week 8
Deadline:	Week 9

Planning notes:

1. Start with transactions surrounding multiple actions (in data access layer)
2. Develop console application to illustrate concurrent access issues

-- SPIKE OUTCOME TEMPLATE / NOTES --

Spike outcomes

Set: *##*
Languages *#######*
Name: *######*

Goals:

Example:
The goal is to understand better how the performance of Technology-Y degrades as the number of users/transactions increases.

Personnel:

Who is the contact for this spike, and who is the backup?

Example:
primary – Jon   secondary - Mary

Technologies, Tools, and Resources used:

Aim of this section is to provide resource information for other team members to learn from this spike. What tools and techniques are required? What resources, books, learning material, etc. can assist? Try to be specific where possible.

Example:
Java 1.6
Ant 2
JDBC 4
Eclipse 3.2
Hibernate manual, chapter 5, available at:
http://www.hibernate.org/elqNow/elqRedir.htm?ref=http://www.hibernate.org/hib_docs/v3/reference/en/pdf/hibernate_reference.pdf

Tasks undertaken:

Show only the key tasks that are likely to help another developer/reader
Example:
** Started the test with 20 concurrent users running 5 sequential transactions, increased user load to 50 concurrent users.*

What we found out:

Graph/Screen shot/outcome list/notes

Open issues/risks [Optional]:

List out the issues and risks that you have been unable to resolve at the end of the spike. You may have uncovered a whole range of new risks as well
Example:
** Risk xyz (new)*

Recommendations [Optional]:

** You may state that another spike is required to resolve new issues identified (or) indicate that this spike has increased the teams confidence in XYZ and move on.*