**Hibernate教程**

　　Hibernate是一個高效能的物件對應關係（Object/Relational）查詢服務，出自於LGPL組織並免費開放下載。Hibernate不只是JAVA類別和Database之間的對應關係，也提供資料的搜尋或是檢索。此教程提供給想了解Hibernate框架以及其API的使用者們，在開始學習前，有基礎的關聯式資料庫、JDBC、SQL的理解，對學習Hibernate來說是相當有幫助的。

**Hibernate-ORM(物件關聯對映 Object Relational Mapping)導覽**

　　講述ORM之前來看一下JDBC，JDBC的意思是Java Database Connectivity，JDBC提供了一套的JAVA API可以去讀取關聯式資料庫，使它可以去執行SQL在任何支援的資料庫上。

　　JDBC也提供一個彈性的架構讓使用者可以在不同的資料庫系統都能運行。

|  |  |
| --- | --- |
| 贊成JDBC的理由 | 反對JDBC的理由 |
| * 乾淨且簡單的SQL運行方式 * 在大筆資料時有好的表現 * 易於開發小型的應用程式 * 簡單的語法宜於學習 | * 在大型專案上還挺複雜的 * 需要較大量的程式管理 * 不是封裝結構 * 在MVC架構下難以實作 * 本身的搜尋方式是特有的 |

為何使用ORM呢?

　　當我們使用物件導向程式時，在物件導向和關聯式資料庫之間是難以去對映的，關聯式資料庫是用表格的方式，而像是C#、JAVA則是物件互相連結的方式。

　　看看下方的JAVA對此的架構和函式：

|  |
| --- |
| public class Employee {  private int id;  private String first\_name;  private String last\_name;  private int salary;  public Employee() {}  public Employee(String fname, String lname, int salary) {  this.first\_name = fname;  this.last\_name = lname;  this.salary = salary;  }    public int getId() {  return id;  }    public String getFirstName() {  return first\_name;  }    public String getLastName() {  return last\_name;  }    public int getSalary() {  return salary;  }  } |

想像上面的各物件要被儲存並存到下方的關聯性資料庫：

|  |
| --- |
| create table EMPLOYEE (  id INT NOT NULL auto\_increment,  first\_name VARCHAR(20) default NULL,  last\_name VARCHAR(20) default NULL,  salary INT default NULL,  PRIMARY KEY (id)  ); |

　　首先，我們要去思考如何去假如我們在設計幾個頁面之後修改設計的問題，第二物件在關聯式資料庫中也暴露著幾個問題：

|  |  |
| --- | --- |
| Granularity(粒度)  -數量關係 | 有時你會有個物件模型，但它所對應到的資料庫數量兩者是不相同的。 |
| Inheritance(繼承) | 關聯式資料庫沒有去定義之間的繼承關係，但在物件導向這是很常見的。 |
| Identity(身分) | 在關聯式資料庫一定有一個概念，就是RK(Primary Key)，但在JAVA卻可以打破這個概念，例如說{a=b; a.equals(b);} |
| Associations(關聯關係) | 在物件導向是用物件的關聯方式，而關聯式資料庫是用外部鍵(Foreign Key)的方式 |
| Navigation(參考關係) | 很明顯地在物件導向看關聯式資料庫就是截然不同。 |

而ORM就是一個方式取處理這些問題。

**甚麼是ORM呢？**

ORM就是一個物件導向和關聯式資料庫之間的一個橋樑，可以是JAVA、C#的開發方式。

|  |  |
| --- | --- |
| 一個ORM系統相對於JDBC有下列的優點：   * 單純的使用物件而不是資料庫喜統 * 讓SQL語法在物件導向內消失 * 比JDBC更深入的使用 * 不需要處理資料庫的實作 * 完全在於商業性的概念而不是資料庫那方面 * 自動處理資料庫的語法 * 更快的開發應用程式 | 而ORM需要的是下列幾個條件：   * 以物件的方式去處理新增、刪除、修改、查詢 * 查詢語言去滿足物件或是類別 * 對於物件導向和資料庫的資料要有一個對映設施 * 以物件的方式去檢查除錯、簡化關聯搜尋…等 |

**JAVA ORM 框架**

以下是幾種JAVA的ORM框架

|  |  |
| --- | --- |
| * Enterprise JavaBeans Entity Beans * Java Data Objects * Castor * TopLink | * Spring DAO * Hibernate * And many more |

**Hibernate-總覽**

Hibernate是一個提供給JAVA的ORM模型，是由Gavin King在2001年提出的，它可以讓你在JAVA系統中有效的做有關查詢語言的動作。

Hibernate的對應關係主要是JAVA類別<->資料庫表和JAVA物件資料<->資料庫資料，有極大部分的常見資料都可以用這種方式呈現。它建立於JAVA物件和資料庫之間，去聯繫著他們之間。



**Hibernate優勢**

* Hibernate是用XML當去聯繫物件和資料庫的關係，免去一些SQL之類的用法
* 提供簡單的API生成JAVA物件去聯繫資料庫
* 如果資料庫有新增只需更改XML即可
* 對於不熟SQL的人這方法好用
* 只需程式不需要伺服器去啟動
* 可一次對多個數據產生連動
* 可減少對資料庫的負擔
* 提供簡單的資料查詢條件

**支援的資料庫**

Hibernate支援主流的資料庫，例如下列幾種：

* HSQL Database Engine
* DB2/NT
* MySQL
* PostgreSQL
* FrontBase
* Oracle
* Microsoft SQL Server Database
* Sybase SQL Server
* Informix Dynamic Server

**支援技術**

Hibernate也支援下列的技術：

* XDoclet Spring
* J2EE
* Eclipse plug-ins
* Maven

**Hibernate架構**

Hibernate是一層可以讓使用者不須了解底層API就可以使用的架構，Hibernate就像是資料庫和程式之間的介質。

就像是下列此圖



下列是主要的Hibernate架構和類別



Hibernate使用了數種之前的JAVA API，像是JDBC、JTA(JAVA Transaction)、JNDI(Java Naming and Directory Interface)，JDBC提供對於關聯性資料庫的連接，讓支援JDBC的知料庫也能支援Hibernate，JTA和JNDI則提供Hibernate連接伺服器的功能。

下面則要介紹Hibernate裡面主要的幾個類別的功能用法：

**Configuration Object(結構物件)**

這些物件是你在Hibernate裡會使用的第一個物件，通常都在初始化實建立一次。

* 資料庫連接功能-

　　這通常會用一到多個檔案去建立，像是hibernate.properties和hibernate.cfg.xml，用來定義一個設定檔，告知資料庫名稱、使用者名稱、密碼、映射文件位置等。

* 類別對應建立-

　　用來連接JAVA物件和資料庫表。

**SessionFactory Object**

上述的Configuration Object主要是用來初始化並新增SessionFactory物件，而SessionFactory主要是用來創造Session，可以用來開通JAVA類別和資料庫的橋樑。而SessionFactory是一個重要的類別，通常在應用程式開啟創建時啟動，每個資料庫都需要一個SessionFactory來進行溝通，若有多個資料庫較用多個SessionFactory。

**Session Object**

Session是用來連接資料庫的一個橋樑，它是個輕量型的物件能跟資料庫相互作用，提供物件的儲存和檢索工具。但它並不是一個安全的橋樑，所以必須在使用完畢是關閉它。

**Transaction Object**

這個Transaction物件主要是支持和資料之間傳遞的功能，通常都是由JDBC或JTA提供。也可以經由其他方式去轉寫此功能，不一定需要用到物件功能。

**Query Object**

Query物件用到SQL或是HQL去處理資料，這Query也是有所限制和語法規定，境過這些處理會最終在送出Query。

**Criteria Object**

主要用來對Query進行。