1. Dữ liệu là gì?

Là các mô tả về các sự vật,hiện tượng,sự kiện…được biểu diễn dưới dạng tài liệu,chữ số,hình ảnh,âm thanh…Xử lí dữ liệu sẽ có thông tin ->dữ liệu và thông tin sẽ cho chúng ta tri thức

1. Cơ sở dữ liệu là gì?

Database : nhóm dữ liệu có tổ chức

1. Csdl quan hệ?

Là csdl được xây dựng dựa trên mô hình dữ liệu quan hệ (relational model of data)

Dữ liệu được tổ chức trong các bảng

Bảng bao gồm cột (column) và dòng (row)

Cột : thuộc tính ( attribute) / trường (field)

Dòng : bản ghi (record) / hang (tuple)

1. Hệ quản trị csdl?

Database management system – DBSM là 1 ứng dụng phần mềm máy tính được sử dụng để quản lí csdl. Chức năng : định nghĩa,cập nhật,truy vấn,phân quyền

MySQL là 1 hệ quản trị csdl quan hệ miễn phí,mã nguồn mở,chạy đa nền tảng

Có 2 thành phần chính:

* My SQL server : máy chủ dịch vụ quản trị csdl
* My SQL client : có thể là MySQL workbench,phpMyadmin

1. Sql? Structured Query Language \_ Ngôn ngữ truy vấn có cấu trúc

Nhóm lệnh : DDL (Data Definition Language \_ định nghĩa) : create , altel , drop

DML (Data Manipulation Language \_ thao tác) : insert,select,update,delete

DCL ( Data Control Language \_ phân quyền) : grant , revoke

TCL (Transaction Control Language \_ giao dịch) : commit , rollback , savepoint

1. ERD ( entity relationship diagram) các thành phần :

(Xây dựng quy ước chung để mọi người có thể hiểu)

Entity : thực thể

Attribute : thuộc tính

Relationship : là mối quan hệ giữa các thực thể

* Thực thể : là đối tượng trong thế giới thực,tồn tại độc lập và phân biệt được với các đối tượng khác -> KH : hình vuông,hình chữ nhật
* Thuộc tính : là những đặc tính,thông tin riêng của đối tượng

phân loại : + loại thuộc tính : thuộc tính đơn\_ko thể chia nhỏ & thuộc tính phức hợp \_ có thể chia nhỏ

+ giá trị của thuộc tính :

. Thuộc tính đơn trị : có giá trị duy nhất cho 1 thực thể

. thuộc tính đa trị : các thuộc tính có 1 tập giá trị cho cùng 1 thực thể -> elip lồng nhau

. thuộc tính suy diễn : có thể tính toán được từ các thuộc tính khác

. thuộc tính khóa : là tt dùng để xác định 1 thực thể trong 1 tập thực thể -> hình elip có gạch chân

* Quan hệ : sự liên kết giữa 2 hay nhiều thực thể : one to one 1-1 , one to many 1-n, many to many n-m

1. Chuyển đổi từ requirement sang erd

B 1 : Từ requi xác định các thực thể và thuộc tính

B 2: Xác định các mối quan hệ giữa các thực thể

1. Chuyển đổi từ erd sang csdl qh ( table)

B1: Biến đổi các thực thể thành table : thuộc tính khóa là khóa chính,thuộc tính thông thường là các thuộc tính của table

B2: Biến đổi thuộc tính đa trị : tách thuộc tính đa trị thành table mới,đặt khóa ngoại vào table hiện tại

B3: biến đổi mối quan hệ 1-1 : C1 \_ đặt thuộc tính khóa ngoại ở phía tùy chọn sang phía bắt buộc, C2\_thêm table xác định bởi các thuộc tính là khóa của các thực thể liên quan

B4:biến đổi mối qh 1-n : đặt thuộc tính khóa ở phía 1 sang phía n

B5:biến đổi mối qh n-m:tạo ra 1 table mới có khóa chính là tập hợp của 2 khóa chính của 2 thực thể tham gia vào mối qhe

1. Ràng buộc

Constraint là quy tắc (rule) được áp dụng cho table,cột -> mục đích giúp cho dữ liệu được chính xác,toàn vẹn,tin cậy

* Primary key : khóa chính sd để xác định 1 bản ghi duy nhất trong table(not null và unique,2 loại:khóa chính đơn và khóa chính phức hợp,mỗi table chỉ có 1 khóa chính)
* Foreign key : khóa ngoại là cơ chế tạo ra liên kết giữa 2 table
* Not null
* Unique : giá trị duy nhất
* Default : mặc định
* Check : ktra điều kiện
* Index : tăng tốc độ truy vấn

1. Chuẩn hóa dữ liệu Nomalization Form

Mục đích : giảm bớt sự dư thừa dữ liệu,đảm bảo tính độc lập,giảm bớt lỗi xảy ra khi thực hiện các câu lệnh truy vấn -> chuẩn hóa là bước cuổi cùng để review lại sản phẩm sau khi đã chuyển đổi sang toàn bộ table

1NF : loại bỏ thuộc tính đa trị và suy diễn

2NF: loại bỏ thuộc tính phụ thuộc bộ phận

3NF: loại bỏ thuôc tính bắc cầu

4NF : boyee codd

1. Delete&Truncate

Delete : xóa theo điều kiện,không reset trường tự tăng

Truncate : xóa hết dữ liệu, reset trường tự tăng

1. Đường đi của câu lệnh SQL:

From-> Where -> group ->order -> select

1. Join : inner join + outer join ( left join,right join,full join)
2. Where : lọc kết quả theo dòng
3. Having : quy định điều kiện trong trường hợp sử dụng các hàm
4. Group by : nhóm các hàng dữ liệu có giá trị giống nhau thành 1 nhóm
5. Order by : sắp xếp asc tăng dần,desc giảm dần
6. Union : gộp kết quả của 2 câu lệnh select khác nhau ( đk : cùng số lượng column)

Union : loại bỏ record trùng lặp \_tốc độ chậm hơn

Union all : không loại bỏ record trùng lặp

1. Limit : sd để hạn chế số lượng record được trả về bởi câu lệnh select

Limit offset,row count

Offset = index , ko bắt buộc,mặc định là 0

Row count : số lượng record trả về,tính từ offset

1. Hàm gộp/hàm tổng hợp : Là những hàm hay được sử dụng trong mệnh đề group by

Count,max,min,sum.avg

1. In vs exit: toán tử kiểm tra sự tồn tại của record nào đó ở trong query hoặc tập dữ liệu

+ In sẽ tìm ra tất cả các bộ thỏa yêu cầu câu truy vấn con rồi mới  
so sánh với câu truy vấn mẹ  
+ Exists sẽ quét từng bộ ở câu truy vấn con và so sánh trực tiếp  
với yêu cầu của mẹ, nếu không thỏa là loại trừ ngay (trả về true/false)  
vậy nếu câu truy vấn con quét ở một quan hệ có số bộ từ ít đến trung bình thì nên dùng IN VÀ NOT IN  
Ngược lại nếu câu truy vấn con đòi hỏi phải quét nhiều quan hệ có số bộ lớn thì EXISTS và NOT EXITST lại tốt hơn

1. Truy vấn lồng/ sub-query : là thực hiện truy vấn bên trong 1 truy vấn khác
2. Hàm thông dụng : concat(),ucase(),lcase(),now(),curdate(),year(),timestampdiff()…–Hàm thao tác với chuỗi : cộng chuỗi có 2 cách: dùng dấu cộng và concat , chuỗi có null cộng lại thì concat sẽ bỏ null còn dấu cộng sẽ không bỏ và trả về giá trị null
3. Index ---Google search

Tăng tốc độ truy vấn,tìm kiếm

2 loại : clustered Index (index mặc định với khóa chính) và non-clustered Index (index do người dùng tự định nghĩa \_ các trường còn lại trừ khóa chính)

Cơ chế : binary tree và hash index

Dựa vào index ta sẽ tìm kiếm nhanh hơn (chỉ những column nào luôn tìm kiếm và dữ liệu lớn thì mới cần đánh index)

Create unique/rỗng index name\_index on table table/name(namecolumn)

1. View

Là khung nhìn được tạo từ 1 hoặc nhiều bảng

Mục đích : giảm sự phức tạp của câu lệnh query,bảo mật che dấu dữ liệu,có thể phân quyền cho ai xem, có thể tùy biến kết quả trả về

Lưu ý : Khi update hoặc insert view để thay đổi các table thì các column nằm trong view phải không có ràng buộc ; select không được dùng từ khóa distinct,hàm tổng,hàm tập hợp,order by,from nhiều bảng,không có truy vấn con,group by,having

Create view nameview as select…;

Không phải view nào cũng insert và update được(muốn update phải view đơn giản trừ các trường hợp trong mục lưu ý ở trên)

1. Stored procedure

Giống vs hàm void trong java : là tập hợp các câu query để thực hiện 1 nhiệm vụ nào đó

Tham số: 3 loại :

In : tham số truyền vào (giống tham trị)

Out : tham số có mục đích lấy kết quả ( đưa biến out vào và nhận lại khi xong thủ tục)

InOut : tương tự tham chiếu

* Sd call để gọi sp
* - từ khóa khai báo biến : declare và từ khóa intro để gán dữ liệu

1. Trigger

Cũng là 1 tập hợp các câu lệnh query để thực hiện 1 nghiệp vụ nào đó.Sd tự động nhưng chỉ đối vs các câu lệnh insert,update,delete

Create trigger nametrigger

Before/after insert/update/delete on tablename

For each row

Begin query end

1. Function

Cũng là 1 tập hợp các câu query để thực hiện 1 tác vụ nào đó nhưng luôn có giá trị trả về

Create function namefunction (param…)

Returns typereturn

Begin query end;

+ read sql data : xác định chỉ đọc dữ liệu từ sql mà không thay đổi trong table

+modifies sql data : có thể sd lệnh update/delete/insert/alter trong function

+deterministic : xác định là trong function có các tác vụ liên quan đến thao tác chuỗi và xử lí toán học

+function có thể gọi sp nhưng ngược lại thì không

1. Phân biệt

|  |  |  |  |
| --- | --- | --- | --- |
|  | Trigger | Sp | Function |
| Kiểu dữ liệu trả về | không | Không | có |
| Tham số truyền vào | không | Có (in,out,inout) | Có(in) |
| Cách sử dụng | Sẽ tự động chạy khi có 1 sự kiện xảy ra | Call | Sử dụng kèm query  (có thể gọi sp nhưng ngược lại thì không) |
| xử lí ngoại lệ try/cactch |  | Có thể sử dụng | (không thể sử dụng trong các hàm do người dùng tự định nghĩa) |

1. CSS

Cascading Style Sheets là ngôn ngữ được sử dụng để tìm và định dạng lại các phần tử được tạo ra bởi ngôn ngữ đánh dấu html

3 cách nhúng :

+ Inline style : khai báo trực tiếp trong thẻ html

+ internal style sheet : sd thẻ <style> trong head

+ external style sheet : khai báo file css riêng biệt

Ưu tiên : inline -> internal vs external tùy vị trí đặt

Dùng từ khóa impotant để xét độ ưu tiên

1. CSS selector

+ tag selector : chọn ptu html

+ id selector (#id) : chọn id của phần tử html (id không được trùng nhau)

+ class selector (.class) : chọn tất cả phần tử html có cùng class

Ưu tiên : id->class->tag

1. Box model

Mỗi phần tử html được xem như là 1 chiếc hộp (box model).Box model là khái niệm để nói về việc design layout.

4 thành phần : content : nội dung của thẻ html

Padding : khoảng cách từ nội dung đến border

Border : viền của thẻ html

Margin :khoảng cách từ box morder này đến bm khác

1. Float & Clear trong CSS:

Float : sử dụng để chuyển 1 phần tử sang góc trái hoặc phải của không gian bao quanh nó

Clear (ngược lại vs float) : ngăn chặn thành phần a chiếm vùng không gian của thành phần b (b sử dụng float)

1. Display style

Thuộc tính display trong css cho phép chúng ta xác định kiểu hiển thị của thẻ html

+ inline : chiều rộng của thẻ sẽ phụ thuộc vào nội dung trong thẻ ; không thể sd được thuộc tính width,height; gt md của nhóm thẻ span,strong,b,a

+block:chiều rộng của thẻ chiếm 100% độ rộng của trang;dùng đc thuộc tính width height,dtmd của nhóm thẻ block : div,p,h..

+inline-block: kết hợp đặc tính của 2 kiểu trên

1. position

Dùng để định vị trí của 1 phần tử html

+Static : giá trị mặc định

+Relative : căn chỉnh so vs static,có thể dùng kết hợp vs left,right,top

+absolute : căn chỉnh so vs thẻ cha chứa nó (thẻ cha cần có position khác static)

+fixed : căn chỉnh so với khung của trình duyệt , không bị ảnh hưởng bởi scroll

+sticky : khi scroll đến vị trí nào đó thì phần tử sticky sẽ được dán cố định trên trang

1. RWD

Responsive Web Design (Thiết kế web thích ứng) là kĩ thuật được sử dụng để trang web có thể hiển thị tốt (không bị mất nội dung) trên nhiều loại thiết bị với kích thước trình duyệt khác nhau

1. Grid View

Phân bố nội dung theo dạng lưới,nội dung của trang được chia thành các cột/hang (thường 12 cột)

1. RWD Media Queries

Media Queries (truy vấn phương tiện) là 1 kĩ thuật của css 3,cho phép kết xuất nội dung để thích ứng với điều kiện kích thước và độ phân giải màn hình.

1. Phân biệt framework với Library

+ Lybrary là công cụ,tập hợp các chức năng (function) class được viết sẵn để có thể tái sử dụng

+ Framework là 1 tập hợp các library được đóng gói để phát triển dựa theo framework đó. Framwork cung cấp tài nguyên,nguyên tắc,cấu trúc của ứng dụng mà ltv cần phải tuân thủ theo nó

1. BootStrap

Là 1 front end frameword miễn phí

Gồm các mẫu thiết kế dựa trên html,css,javascript

Cung cấp khả năng tạo các thiết kế responsive

2 cách nhúng : CDN(nhược không bảo mật)

và tải về dùng trực tiếp(nhược tốn dung lượng)

1. Hệ thống grid của bootstrap

Chia 1 trang web thành 12 cột

.col- : dành cho thiết bị rất nhỏ extra small (rộng <576px)

.col-sm : thiết bị nhỏ small (rộng >=576px)

.col-md : tbt rung bình medium ( rộng >=768px)

.col-lg : tbt lớn large (rộng >=992px)

.col-xl : tb rất lớn extra large (rộng >=1200px)

1. Boostrap 1 số thành phần thông dụng:

* Màu sắc : primary( xanh lam) , secondary (xám),success(xanh lá cây),danger(đỏ),warning(vàng),info(xanh lục),light(trắng),dark(đen)
* Button
* Table
* Carousel (chạy hình ảnh)
* Navbar
* Modal(xác nhận)
* Card(ảnh và nội dung đi kèm)
* ….

1. Website

+Tập hợp các trang web ( web page ) chứa văn bản,âm thanh,hình ảnh,video

+Một trang web có thể lưu trữ ở máy chủ hoặc máy tính cá nhân của người dùng

+Phân loại :

* Static website : là loại web mà phần nội dung không thay đổi theo thời gian,thường được viết bằng mã html,css,js
* Dynamic website : là loại website mà phần nội dung có thể thay đổi theo thời gian ngữ cảnh hoặc tương tác của người dùng . Thường được viết bằng ngôn ngữ back-end như java,php,c#... Dynamic website thông thường được lưu trữ trên 1 máy chủ web ( web server )

1. Giao thức HTTP , HTTPs

+ HTTP : HyperText Transfer Protocol : Giao thức truyền tải siêu văn bản được sử dụng ở trong www . Dùng để truyền tải dữ liệu giữa web service và web browser

+ HTTPs : HyperText Transfer Protocol Secure : giao thức http có thêm chứng chỉ SSL ( Secure Socket Layer ) để mã hóa dữ liệu trong lúc truyền tải dữ liệu giữa web browser và web server

1. Mô hình Client – Server

Là mô hình thông dụng trong mạng máy tính.chia gồm 2 thành phần là máy chủ server và máy khách client

Ý tưởng : client gửi request đến server.Server xử lí yêu cầu của client rồi trả về kết quả reponse cho client

Status code : mã code gồm 3 chữ số 1xx-2xx-3xx-4xx-5xx

200 : thành công ;

404 : không tìm thấy trang \_ nhập url sai,điều hướng sai,nhập sai link trên thanh địa chỉ

500:code ở sever bị sai

1. Phân biệt POST và GET

Điểm chung: là các HTTP method dùng để trao đổi dữ liệu giữa client và server.

Điểm khác nhau:

+POST: Bảo mật hơn GET vì dữ liệu được gửi ngầm, không xuất hiện trên URL

+GET: Dữ liệu được gửi tường minh, chúng ta có thể nhìn thấy trên URL, đây là lý do khiến nó không bảo mật so với POST.

-GET thực thi nhanh hơn POST vì những dữ liệu gủi đi luôn được webbrowser cached lại.

-Khi dùng phương thức POST thì server luôn thực thi và trả về kết quả cho client, còn phương thức GET ứng với cùng một yêu cầu đó webbrowser sẽ xem trong cached có kết quả tương ứng với yêu cầu đó không và trả về ngay không cần phải thực thi các yêu cầu đó ở phía server.

-Đối với những dữ liệu luôn được thay đổi thì chúng ta nên sử dụng phương thức POST, còn dữ liệu ít thay đổi chúng ta dùng phương thức GET để truy xuất và xử lý nhanh hơn

1. Phân biệt request vs response

request là yêu cầu từ client lên server và response là server trả kết quả về cho client

1. Java Servlet

Là 1 công nghệ được sử dụng để tạo ra các ứng dụng web (được triển khai ở sever để tạo trang web động)

Vòng đời :

1. Tải servlet class vào bộ nhớ

2. Tạo đối tượng servlet

3. Gọi phương thức init() \_ chạy khi lần đầu tiên gọi servlet (thương dùng để mở file,mở kết nối..)

4. Gọi phương thức service() \_ thể hiện ở get và post

5. gọi phương thức destroy() \_ ngay trước khi xóa servlet,khi tắt sever (dùng vs các công việc luôn cần làm trước khi đóng sever:đóng file,dọn dẹp vùng nhớ)

Điều hướng :

|  |  |
| --- | --- |
| SendRedirect | Forward |
| Không kèm data | Kèm data |
| Giá trị url bị cập nhật thành url đích(gọi lại sever 1 lần nữa) | Không thay đổi giá trị url(nằm trong phiên request của người dùng nên ko gọi lại sever) |
| Tốc độ chậm hơn | Tốc độ nhanh hơn |
| Có thể dùng để gọi các url ngoài hệ thống ( google..) | Không thể |

1. JSP

Java Server Pages là ngôn ngữ kịch bản phía server,cho phép người dùng tạo ra các trang web động.

Ý nghĩa các loại scope trong jsp

…

Session và cookki

|  |  |
| --- | --- |
| **Cookie** | **Session** |
| **Cookie** được lưu trữ trên trình duyệt của người dùng. | **Session** không được lưu trữ trong trình duyệt của người dùng. |
| Dữ liệu **cookie** được lưu trữ ở phía máy khách. | Dữ liệu **session** được lưu trữ ở phía máy chủ. |
| Dữ liệu **cookie** dễ dàng sửa đổi khi chúng được lưu trữ ở phía khách hàng. | Dữ liệu **session** không dễ dàng sửa đổi vì chúng được lưu trữ ở phía máy chủ. |
| Dữ liệu **cookie** có sẵn trong trình duyệt của chúng ta đến khi hết hạn. | Dữ liệu **session** có sẵn cho trình duyệt chạy. Sau khi đóng trình duyệt sẽ mất thông tin **session**. |

1. So sánh JSP vs Servlet

+ Giống : đều là ngôn ngữ phía server,xử lí dữ liệu động. Bản chất JSP cũng chính là servlet ,vòng đời của jsp cũng giống như servet

+Khác:

* Servlet mạnh xử lí logic-điều phối,yếu về xử lí giao diện còn jsp mạnh về xử lí giao diện nhưng yếu về điều phối
* ở Servlet thì mã html nằm trong mã java,còn jsp thì mã java nằm trong mã html

1. JSTL

JSP Standard Tag Library : bộ thư viện thẻ chuẩn được bổ sung để hỗ trợ lập trình trong jsp

Nhóm thẻ :

* Core : chứa những thẻ điều khiển cơ bản
* Format : chứa các thẻ định dạng và đa ngôn ngữ
* XML : chứa các thẻ xử lí tài liệu xml
* SQL : chứa các thẻ làm việc vs CSDL
* Function : chứa các thẻ cung cấp các hàm hỗ trợ cho Expression Language

1. Một số thẻ core cơ bản:

* C:out - hiển thị kết quả của 1 biểu thức

<c:out value="${'This is true: 10 > 1 '}" />

* C:set – đánh giá 1 biểu thức và sử dụng các kết quả để đặt giá trị của 1 đối tượng javabean hoặc java.until.map

<c:set scope="request" var="greeting" value="Hello every body" />

Greeting: <c:out value="${greeting}"/>

* C:remove – xóa 1 biến khỏi phạm vi được chỉ định hoặc phạm vi đầu tiên nơi tìm thấy biến

<c:set scope="request" var="greeting" value="Hello every body" />

Greeting: <c:out value="${greeting}"/>

<c:remove scope="request" var="greeting" />

* C:catch – bắt ngoại lệ xảy ra trong phạm vi xử lí của nó

<c:catch var ="ex"><%int a = 100/0;%></c:catch>

<c:if test = "${ex != null}">Exception : ${ex}Message: ${ex.message}</c:if>

* C:if – kiểm tra điều kiện

<c:if test = "${salary > 2000}">

         <p>My salary is:  <c:out value = "${salary}"/><p>      </c:if>

* C:choose + c:when + c:otherwise : =switch case
* C:forEach : - lặp

<c:forEach var = "i" begin = "1" end = "5">

         Item <c:out value = "${i}"/><p>    </c:forEach>

* C:forTokens : ngắt chuỗi = split

<c:forTokens items="Tom,Jerry,Donald" delims="," var="name">

<c:out value="${name}"/><p></c:forTokens>

1. MVC

+ Model : biểu diễn dữ liệu,xử lí logic nghiệp vụ ,xử lí kết nối database

+View : hiển thị dữ liệu và giao diện tương tác với người dùng

+Controller : xử lí đón nhận request từ người dùng,kết nối giữa tầng model và view

* Lợi ích :

- Dễ tái sử dụng

* Dễ mở rộng
* Tách phần view và phần nghiệp vụ riêng biệt
* Cho phép các lập trình viên làm việc trên các thành phần khác nhau trong cùng 1 thời điểm
* Dễ bảo trì

1. Cách tổ chức Package theo mô hình MVC

Controller ---------(bean)------🡪 service (BO) -------(bean)-----🡪 Repository(DAO) 🡪database

+Tầng controller : servlet

+Tầng model:

* Bean : chứa dữ liệu (class)
* Service (BO) xử lí logic nghiệp vụ
* Repository (DAO) biểu diễn dữ liệu và kết nối database

+Tầng view : jsp

1. JDBC

Java Database Connectivity là 1 API ( Application Program Interface ) cho phép kết nối các chương trình viết bằng ngôn ngữ java với các hệ qtCSDL ( My SQL,SQL service,Oracle,DB2…)

+ Thành phần của JDBC :

* Driver : dùng để liên kết các kết nối tới csdl ( mỗi hqtdl có 1 driver riêng )
* Driver Manager : dùng quản lí danh sách các driver
* Connection : đối tượng biểu thị có việc kết nối tới csdl
* Statement,PrepareStatement,CallabeStatement chứa câu lệnh sql , gửi tới csdl để thực thi

(statement st = connection.createStatement() ; PrepareStatement ps = connection.prepareStatement() ; CallableStatement cs = prepareCall() )

* ResultSet : chứa tập kết quả trả về từ CSDL ( thông thường được dùng vs câu lệnh select )
* SQLException : class xử lí ngoại lệ liên quan đến việc truy cập csdl

+ Trình tự làm việc :

* Tạo connection bởi các thông tin : host name (localhost) ,database name,username ( root ),password,port (3306)
* Sử dụng statement,prepareStatement,CallableStatement để tạo câu lệnh query
* Sử dụng ResultSet để xử lí kết quả ở bước 2 đối vs câu lệnh select
* Đóng connection

1. Phân biệt các loại statement

Cả 3 đều là interface

|  |  |  |
| --- | --- | --- |
| Statement | Prepare Statement | Callable Statement |
| Dùng để tạo câu truy vấn tĩnh | Dùng để tạo câu truy vấn động | Thường được dùng để gọi Procedure |
| Hiệu suất thấp | Hiệu suất cao hơn | Hiệu suất cao hơn |
| Không bảo mật |  |  |

1. Transaction

Giao dịch/giao tác : là 1 tiến trình xử lí có điểm bắt đầu và điểm kết thúc

Transaction gồm nhiều phép thực thi nhỏ,trong đó mỗi phép thực thi sẽ thực hiện 1 cách tuần tự và độc lập,theo nguyên tắc là tất cả thành công thì transaction được xem là thành công,1 phép thực thi thất bại thì tất cả các phép thực thi còn lại là thất bại (transaction thất bại)

setAutoCommit(false) =>commit / rollback

* Dùng savePoint để quay lại bước nào đó chứ không quay về ngay ban đầu với nhiều tiến trình nhỏ

1. ACID

+ Atomicity : tính nguyên tử : tất cả thành công hoặc không (1 tiến trình thất bại thì sẽ không có tiến trình nào thành công)

+ Consictency : tính nhất quán : đảm bảo tính đồng nhất của dữ liệu

+ Isolation : tính độc lập : đảm bảo transaction này độc lập với transaction khác

+ Durability : tính bền vững : nghĩa là 1 transaction đã thực hiện thành công thì kết quả đó sẽ được đảm bảo tồn tại dù có xảy ra các lỗi (về thiên tai,kĩ thuật..)

1. Đóng gói phần mềm

Sản phẩm phần mềm khi mà chuyển giao cho người dùng thì bắt buộc phải được đóng gói,để đảm bảo sản phẩm luôn ở trạng thái sẵn sàng hoạt động

* Các hình thức đóng gói

+ Standalone : chỉ có mã nguồn,không có môi trường thực thi JRE

+ Self-contanined : mã nguồn+JRE+thực thi với khả năng tự khởi động ứng dụng

1. Sản phẩm của việc đóng gói

* JAR file : sản phẩm đóng gói phần mềm / thư viện java
* WAR file : sản phẩm đóng gói ứng dụng web application

1. Các công cụ quản lí thư viện phụ thuộc

Depedence management tool : phần mềm giúp quản lí tích hợp những thư viện/module phụ thuộc vào chương trình/dự án đang phát triển

+ Apache Ant : ra đời 2000

+ Maven : 2004 cho phép trả về các dependence thông qua internet

Nhược điểm : quản lí dependence theo cú pháp XML tuân thủ theo tiêu chuẩn Maven standart layout -> triển khai file POM dài dòng phức tạp

+ Gradle : 2012 : Build script ngắn gọn và trực quan \_> file build gradle