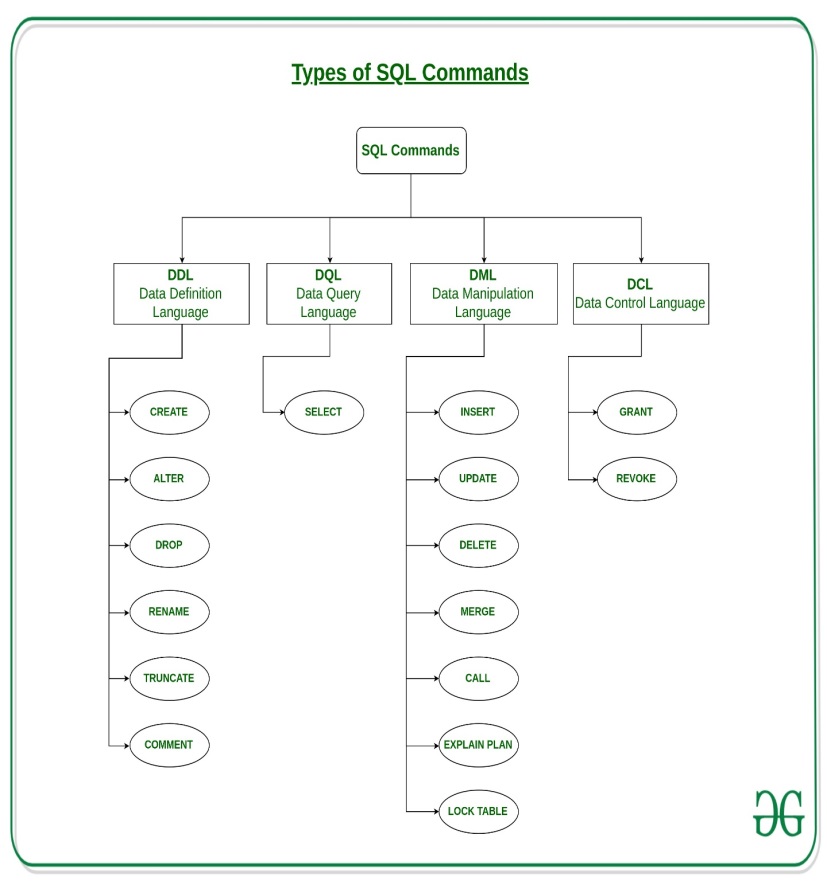
1. **Phân biệt các câu lệnh DML & DDL? DML, DDL bao gồm các câu lệnh nào?**

* DML là viết tắt của Ngôn ngữ Thao tác Dữ liệu ( Data Manipulation Language): INSERT, UPDATE và DELETE là các câu lệnh DML.
* DDL là viết tắt của Ngôn ngữ Định nghĩa Dữ liệu (Data Definition Language): CREATE, ALTER, DROP, RENAME là các câu lệnh DDL.



1. **Cơ sở dữ liệu (database) là gì?**

Cơ sở dữ liệu (database) nhóm dữ liệu có tổ chức

1. **Trình bày về mô hình thực thể mối quan hệ (ERD - Entity-Relationship Diagram)? Trong mô hình thực thể mối quan hệ có những thành phần nào?**

ERD là một sơ đồ, thể hiện các thực thể có trong CSDL, và mối quan hệ giữa chúng với nhau.

* ***Thực thể (Entity):*** là đối tượng cụ thể hoặc trừu tượng mà CSDL quản lý
* ***Thuộc tính (Attributes***): Là các đặc trưng mô tả thực thể.
* ***Mối quan hệ/ liên kết (Relationship):*** Là sự liên kết giữa 2 hoặc nhiều thực thể. Ví dụ: Một sinh viên tham gia nhiều môn học, mỗi một môn học có nhiều sinh viên
* ***Khóa***: Là thuộc tính mà giá trị của nó khác nhau trên 2 thực thể. Dùng để phân biệt 2 thực thể đó trong 1 nhóm. Ví dụ: mỗi sinh viên có 1 mã sinh viên riêng biệt.

1. **Các bước chuyển đổi từ ERD -> Table?**

* B1:Xác định thực thể, thuộc tính, mối quan hệ.
* B2.Tạo table, thêm khóa chính, khóa ngoại(Ưu tiên các table ít liên quan nhất)

1. **Các bước chuẩn hóa dữ liệu?**

Mục đích:

* Giảm thiểu dư thừa dữ liệu
* Loại bỏ các bất thường khi cập nhật cơ sở dữ liệu

Có 3 dạng chuẩn hóa cơ bản đó là:

1. ***First Normal Form (1NF): dạng chuẩn 1NF***

* Các thuộc tính của bảng phải là nguyên tố
* Giá trị của các thuộc tính trên các hàng phải là đơn trị, không chứa nhóm lặp
* Không có một thuộc tính nào có giá trị có thể tính toán được từ một thuộc tính khác

\

1. ***Second Normal Form (2NF): dạng chuẩn 2NF***

* Bước 1: Loại bỏ các thuộc tính không khóa phụ thuộc vào một bộ phận khóa chính và tách ra thành một bảng riêng, khóa chính của bảng là bộ phận của khóa mà chúng phụ thuộc vào.
* Bước 2: Các thuộc tính còn lại lập thành một quan hệ, khóa chính của nó là khóa chính ban đầu.

1. ***Third Nomal Form (3NF): dạng chuẩn 3NF***

* Phải đạt chuẩn 2NF-
* Mọi thuộc tính không khóa phụ thuộc bắc cầu vào thuộc tính khóa (nghĩa là tất cả các thuộc tính không khóa phải được suy ra trực tiếp từ thuộc tính khóa)

1. **Ràng buộc (constraint) là gì? Các loại ràng buộc trong MySQL?**

* Constraint là các quy tắc (rule) được quy định cho bảng
* Sử dụng constraint, chúng ta có thể hạn chế những dữ liệu có thể đưa vào trong bảng
* Constraint giúp cho dữ liệu chính xác, tin cậy, toàn vẹn
* Dữ liệu đưa vào trong bảng cần tuân thủ các constraint...

+ NOT NULL: Không cho phép giá trị NULL

+ UNIQUE: Mỗi giá trị là duy nhất

+ PRIMARY KEY: Khóa chính (Không NULL và là UNIQUE)

+ FOREIGN KEY: Khóa ngoại (tham chiếu sang bảng khác)

+ CHECK: Kiểm tra dựa vào một điều kiện

+ DEFAULT: Quy định giá trị mặc định cho trường (nếu không có giá trị

nào được nhập vào)

+ INDEX: Giúp tăng tốc độ truy vấn dữ liệu

1. **Khóa chính (Primary key) là gì? “Khóa chính chứa một hoặc nhiều thuộc tính khóa” nghĩa là gì?**

Khóa chính là một cột (hoặc nhiều cột) được sử dụng để xác định một bản ghi duy nhất trong bảng

* Khóa chính là UNIQUE và NOT NULL
* Mỗi bảng chỉ có thể có 1 khóa chính.Khóa chính phức hợp (composite) là khóa được tạo nên từ 2 hoặc nhiều cột

1. **Khóa ngoại (foreign key) là gì?**

* Khóa ngoại là cơ chế để tạo liên kết giữa 2 bảng trong cùng CSDL
* Khóa ngoại được đặt trên một cột của bảng này và tham chiếu đến khóa chính của một bảng khác
* Kiểu dữ liệu của khóa chính và khóa ngoại phải giống nhau

1. **Phân biệt DELETE và TRUNCATE?**

|  |  |
| --- | --- |
| **DELETE** | **TRUNCATE** |
| Có thể xóa từng hàng | Xóa toàn bộ record của bảng |
| Khi DELETE, SQL sẽ log lại các dòng đã xóa trong transaction log | Xóa record của bảng và reset transaction log |
| Trong trường hợp id số nguyên tự tăng, khi tạo record mới id sẽ là id n+1 với n là id mới nhất đã được khởi tạo trước đó | Id sẽ bắt đầu lại từ số 1 |
| Có thể khôi phục dữ liệu đã xóa | Không thể khôi phục dữ liệu đã xóa |
|  | Không thể chạy được khi bảng định xóa có foreign\_key (nếu bảng đó có foreign\_key trỏ đến chính nó thì có thể xóa bình thường) |
| Tốc độ xóa của DELETE chậm hơn TRUNCATE |  |

1. **Làm sao để truy vấn dữ liệu từ nhiều bảng?**

Sử dụng Join

1. **Các loại JOIN? Phân biệt?**

INNER JOIN,LEFT JOIN,RIGHT JOIN,FULL JOIN,CROSSJOIN, SELFJOIN

1. **Phân biệt AND và OR?**
2. **Phân biệt GROUP BY và ORDER BY?**

* GROUP BY: sẽ tổng hợp các bản ghi theo các cột được chỉ định cho phép bạn thực hiện các hàm tổng hợp trên các cột không được nhóm (chẳng hạn như SUM, COUNT, AVG, v.v.).
* ORDER BY: sắp xếp các bản ghi theo các cột được chỉ định

1. **Muốn nối chuỗi trong MySQL thì sử dụng gì?**

CONCAT( expression1, expression2, ... expression\_n )

CONCAT\_WS( separator, expression1, expression2, ... expression\_n )

1. **Yêu cầu tính điểm trung bình của tất cả các sinh viên thì làm thế nào?**

SELECT AVG(diem) from bangdiem;

1. **Yêu cầu tính điểm trung bình của từng bạn sinh viên thì làm thế nào?**

SELECT AVG(diem1, diem2, diem3) FROM bangdiem GROUP BY id\_sinhvien;

1. **Yêu cầu hiển thị tên sinh viên và điểm trung bình tương ứng, chỉ hiển thị các bạn có điểm trung bình lớn hơn 5 thì làm thế nào?**

SELECT ten\_sinhvien, avg(diem1, diem2, diem3) as diemTB FROM bangdiem GROUP BY id\_sinhvien HAVING diemTB > 5;

1. **Hiển thị danh sách sinh viên và lớp học tương ứng thì làm thế nào (hai bảng Student và Class có cột chung ClassID)?**

SELECT \* FROM student LEFT JOIN class ON student.ClassID = class.ClassID;

1. **Hiển thị danh sách sinh viên theo thứ tự tăng dần của điểm thi?**

SELECT \* FROM sinhvien ORDER BY diemthi;

1. **Hiển thị danh sách sinh viên theo thứ tự giảm dần của họ tên, nếu họ tên trùng nhau thì sắp xếp theo tăng dần của tuổi?**

SELECT \* FROM sinhvien ORDER BY hoten\_sinhvien DESC, tuoi\_sinhvien;

1. **Hiển thị các sinh viên có họ là ‘Le’ và tên là ‘Hai’?**

SELECT \* FROM sinhvien WHERE ten\_sinhvien like “Le%” and “%Hai”;

1. **Hiển thị các sinh viên có tên bắt đầu là ‘C’?**

SELECT \* FROM sinhvien WHERE ten\_sinhvien like “c%”;

1. **Phân biệt JOIN và UNION?**

|  |  |
| --- | --- |
| **JOIN** | **UNION** |
| cho phép kết nối dữ liệu liên quan trong các bảng khác nhau | hợp dữ liệu của 2 bảng mà có các cột loại data giống nhau. |
| JOIN cho phép chúng ta "tra cứu" các bản ghi trong các table khác nhau dựa trên các điều kiện kết nối giữa hai bảng. | UNION cho phép chúng ta thêm các bộ dữ liệu tương tự nhau để tạo ra bộ dữ liệu kết quả có chứa tất cả dữ liệu từ các bộ dữ liệu nguồn. Union không yêu cầu bất kỳ điều kiện join nào. |

1. **Phân biệt UNION và UNION ALL?**

UNION loại bỏ các bản ghi trùng lặp trước khi trả lại kết quả còn UNION ALL giữ lại tất cả các bản ghi từ hai tập ban đầu.

1. **Phân biệt EXIST và IN?**

|  |  |
| --- | --- |
| **IN** | **EXIST** |
| Ví dụ: – **Tìm những sản phẩm của Singapore mà SoHD ‘1001’ chưa mua**.   1. SELECT \* 2. FROM SANPHAM 3. WHERE NUOCSX='SINGAPORE' AND MASP **NOT IN**(--Câu truy vấn con 4. SELECT MASP 5. FROM CTHD 6. WHERE SOHD='1001') | 1. SELECT \* 2. FROM SANPHAM A 3. WHERE NUOCSX='SINGAPORE' AND **NOT EXISTS**(--Câu truy vấn con 4. SELECT \* 5. FROM CTHD B 6. WHERE SOHD='1001' AND A.MASP=B.MASP) |
| 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 |
| 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 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. **Làm sao để có thể tăng tốc độ truy vấn?**

* Chỉ SELECT những trường mà bạn cần
* Sử dụng Indexs và cố gắng truy vấn dữ liệu thông qua các điều kiện xác lập trên chỉ số.
* Sử dụng EXPLAIN để phân tích truy vấn: Sử dụng EXPLAIN để phân tích truy vấn
* Chấp nhận dư thừa dữ liệu: Truy vấn trên một bảng sẽ nhanh hơn rất nhiều lần khi truy vấn trên nhiều bảng quan hệ

1. **Các dạng tham số ở SP?**

IN: Đây là chế độ mặc định, không bị thay đổi nếu như trong Procedure có tác động đến

OUT: Chế độ này nếu như trong Procedure có tác động thay đổi thì nó sẽ thay đổi theo, mặc định nó luôn hiểu giá trị truyền vào là NULL.

INOUT: Đây là sự kết hợp giữa IN và OUT, có thể gán giá trị trước và có thể bị thay đổi nếu trong Procedure có tác động tới

1. **Phân biệt SP, Trigger, Function?**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nội dung** | **SP** | **Function** | **Trigger** |
| Khái niệm | Đối tượng cơ sở dữ liệu chứa một tập các câu lệnh SQL để hoàn thành một tác vụ.  Thực thi khi được gọi | | Là một đối tượng được định danh trong CSDL và tự động thực thi khi xảy ra những giao tác làm thay đổi dữ liệu trong các bảng. |
| Giá trị trả về | Thủ tục lưu trữ có thể trả về giá trị zero, một hoặc nhiều giá trị. | Trong khi hàm phải trả về một giá trị duy nhất (có thể là bảng). | Khi xảy ra ra một sự kiện thao tác dữ liệu một bản ghi trong CSDL nó sẽ lưu ra một bản ghi trong Trigger nó có tên là inserted đối với các thao tác Insert hay Update và deleted đối với Delete |
| Tham số | tham số đầu vào hoặc đầu ra. | chỉ có thể có các tham số đầu vào | Không tham số |
| Gọi | Thủ tục gọi được hàm | Hàm không goi được thủ tục |  |
|  |  |  |  |

* 1. **CSS là gì? Các cách nhúng CSS vào tài liệu html? Độ ưu tiên?**
* Cascading Style Sheets mô tả cách các phần tử HTML sẽ hiển thị trên màn hình,  
  trang web, hoặc các phương tiện khác
* Nhúng CSS:

+ Khai báo trực tiếp ở thẻ HTML (Inline style) : <a style = “…> </a>

+ Khai báo trong thẻ <style> của tài liệu HTML (Internal style sheet): <style>….</style>

+ Khai báo trong file .css riêng biệt (External style sheet): <link rel = “stylesheet” type=”text/css” href =”mystyle.css”>

* Đô ưu tiên: inline > internal style sheet > external style sheet
  1. **Phân biệt giữa bộ chọn class và bộ chọn id?**

.class selector chọn các phần tử với class xác định

#id selector chọn các phần tử với id xác định

Tag selecter chọn tất cả các phần tử

Kết hợp nhiều bộ chọn bằng dấu phẩy

* 1. **Trình bày mô hình hộp (box model) trong CSS?**
* Margin - là phần nằm bên ngoài của border. Margin trong suốt.
* Border - là đường viền bao xung quanh của padding và content.
* Padding - là phần bao xung quanh content (nội dung của phần tử), nằm giữa padding và content.
* Content - là nội dung thực tế của phần tử
  1. **Trình bày thuộc tính float, clear?**

Thuộc tính Float sử dụng để chuyển một phần tử sang góc trái hoặc phải của không gian bao quanh nó, rất cần thiết trong việc định dạng bố cục trang.

Theo mặc định, tất cả các phần tử HTML là không float.

Thuộc tính Float có thể có một trong các giá trị sau:

* left: Cố định phần tử về bên trái.
* right: Cố định phần tử về bên phải.
* none: Nằm tại chính vị trí của nó (trạng thái bình thường).
* inherit (kế thừa): Phần tử kế thừa giá trị từ float cha.

Clear gần như là ngược lại với float. Thuộc tính Clear ngăn chặn thành phần A chiếm vùng không gian của thành phần B (với thành phần B là thành phần sử dụng float). Đôi khi không muốn float ở một số tình huống nào đó ta sẽ dùng clear để khắc chế.

Nói một cách đơn giản, clear sử dụng để giải quyết vấn đề trong mục chú ý của thuộc tính float phía trên.

Thuộc tính clear có thể có một trong các giá trị sau:

* left: Tràn về phía bên trái.
* right: Tràn về phía bên phải.
* none: Cho phép tràn lên cả hai phía (mặc định).
* both: Không cho phép tràn về bên nào
* inherit (kế thừa): Phần tử kế thừa giá trị từ float cha.
  1. **Thuộc tính Position**

Thuộc tính position của CSS quy định cách thức xếp đặt vị trí của một thành phần trong trang web, hay nói cách khác position chỉ định phương thức định vị sẽ áp dụng cho một phần tử HTML

* Static: Khi một phần tử là static điều đó có nghĩa là phần tử này sẽ được sắp xếp đúng theo vị trí tự nhiên của nó, theo luồng bình thường của trang. Không bị ảnh hưởng bởi thuộc tính top left right bottom.
* Relative: Lấy vị trí bình thường của nó làm gốc tọa độ. Thiết lập cho các thuộc tính định vị top, right, bottom, left cho một phần tử là relative là nguyên nhân khiến nó bị điều chỉnh ra khỏi vị trí bình thường của nó.
* Absolute: lấy thành phần cha làm gốc tọa độ
* Fixed: lấy chế độ xem (view port) làm gốc tọa độ
  1. **Responsive Web Design là gì?**
* RWD (Thiết kế Web Thích ứng) là kỹ thuật được sử dụng để trang web có thể hiển thị tốt trên nhiều loại thiết bị với kích thước của trình duyệt khác nhau.
* Grid-view phân bố nội dung theo dạng một lưới, nghĩa là các trang được chia thành các cột/hàng. Điều đó giúp cho việc thiết kế trở nên dễ dàng hơn, dễ đặt nội dung lên các vùng phù hợp của trang web
* Meria Queries (Truy vấn phương tiện) là một kỹ thuật của CSS, cho phép kết xuất nội dung để thích ứng với các điều kiện như kích thước và độ phân giải màn hình
  1. **Làm thế nào để tạo 1 trang web responsive?**

Để áp dụng Responsive ta cần thực hiện 2 thao tác.

* Khai báo meta viewport
* Viết CSS cho từng loại thiết bị.
  1. **Bootstrap là gì?**

Bootstrap là một front-end framework miễn phí để phát triển web nhanh hơn và dễ dàng hơn.

Bootstrap gồm các mẫu thiết kế dựa trên HTML và CSS

Bootstrap cung cấp khả năng tạo các thiết kế responsive

* 1. **Một số thành phần thông dụng ở Bootstrap?**

Container, container-fluid

Form

Button

GlyphiIcons

Navbar

*(Nếu có thắc mắc, Hãy gửi teamview cho Phattrap – A0720i1 để được giải đáp)*

* 1. **Phân biệt Static web và Dynamic web?**
* Static web: Là loại website cơ bản được tạo mà không cần phải biết đến các  
  ngôn ngữ lập trình web như Java, PHP, … hoặc thiết kế cơ sở dữ  
  liệu. Những trang web trong website tĩnh được viết bằng mã HTML hoặc  
  thêm CSS, JavaScript để thêm các hiệu ứng.
* Website động là website có nội dung thay đổi. Sự thay đổi có thể là tùy theo thời gian, tùy theo người dùng, tùy theo ngữ cảnh. Để tạo được website động, chúng ta thường sử dụng đến các ngôn ngữ phía server (server-side), chẳng hạn như Servlet, JSP, PHP, Python, C#...

Một website động thường được đặt trên một máy chủ dịch vụ web (Web Server)

* 1. **Phân biệt 2 phương thức GET/POST**

|  |  |
| --- | --- |
| Get | Post |
| Nút quay lại/Tải lại | Không có vấn đề gì cả | Dữ liệu sẽ bị gửi lại(Trình duyệt sẽ hỏi ý kiến của bạn) |
| Đánh dấu(bookmarked) | Có thể | Không thể |
| Lưu trữ(cached) | Có thể | Không thể |
| Kiểu mã hóa(encoding type) | application/x-www-form-urlencoded | application/x-www-form-urlencoded hoặc multipart/form-data sử dụng các multipart encoding cho các tập dữ liệu nhị phân(binary data) |
| Lịch sử | Các thông số được lưu lại trong trình duyệt. | Các thông số không được lưu lại. |
| Độ dài giới hạn | Có giới hạn, khi sử dụng phương thức GET thì sẽ nhập dữ liệu lên thanh URL(độ dài giới hạn là 2048 kí tự) | Không giới hạn |
| Kiểu dữ liệu giới hạn | Chỉ có các kí tự ASCII | Không giới hạn, mã nhị phân cũng được. |
| Bảo mật | GET ít bảo mật hơn so với Post do các dữ liệu gửi được hiển thị trên URL.  Không bao giờ sử dụng GET để gửi các thông tin như mật khẩu hay các dữ liệu nhạy cảm khác | POST thì an toàn hơn GET một ít do các dữ liệu không bị lưu lại trên trình duyệt hay trên server Web log. |
| Hiển thị | Dữ liệu hiển thị trên URL | Không show dữ liệu |

* 1. **Phân biệt request/response**

Request là thông tin từ client gửi tới server. Gồm:

* Request line: Chứa method (get/post), URL và HTML version
* Request header:

Respond là thông tin từ server phản hồi lại client

* 1. **Phân biệt forward/redirect**

|  |  |
| --- | --- |
| **FORWARD** | **REDIRECT** |
| Chuyển đến tài nguyên khác trong cùng 1 máy chủ để xử lý | Có thể chuyển đến tài nguyên ở máy chủ khác xử lý |
| Không thấy địa chỉ chuyển tiếp | Thấy địa chỉ chuyển tiếp trong thanh địa chỉ |
| Tốc độ chuyển nhanh hơn | Tốc độ chuyển chậm hơn |
| Gửi dữ liệu sang một trang khác bằng requestSetAttribute() | Lưu dữ liệu trong phiên làm việc hoặc đi cùng URL |

* 1. **JSP là gì? Nói: “Về bản chất JSP cũng chính là Servlet” đúng hay sai? Giải thích? So sánh giống và khác nhau giữa JSP và Servlet?**

JSP (JavaServer Pages) là một ngôn ngữ kịch bản phía server, cho phép người dùng tạo ra các trang web động.

Về bản chất, JSP cũng chính là Servlet. Vì trong quá trình biên dịch, JSP sẽ được chuyển thành Servlet rồi Servlet này sẽ được biên dịch. Servlet của trang JSP có vòng đời giống như các Servlet thông thường.

*Giống nhau*

* Đều nằm ở phía server.
* Đều xử lý dữ liệu “động”.
* Đều chạy với web container.
* Đều là những công nghệ quan trọng của Java EE.

*Khác nhau*

* Nếu như Servlet mạnh về “controlling” và “dispatching” nên trong kiến trúc MVC, Servlet đóng vai trò Controller, thì JSP lại mạnh về “displaying” và đóng vai trò View.
* Ở Servlet, mã HTML nằm trong mã Java. Còn ở JSP, mã Java nằm trong mã HTML.
  1. **JSTL là gì? Giới thiệu một số thẻ bạn đã sử dụng JSTL?**

JSP Standard Tag Library (JSTL) là thư viện thẻ chuẩn cung cấp các thẻ để kiểm soát hành vi trang, lặp đi lặp lại và các lệnh điều khiển, các thẻ quốc tế hóa, và các thẻ SQL.

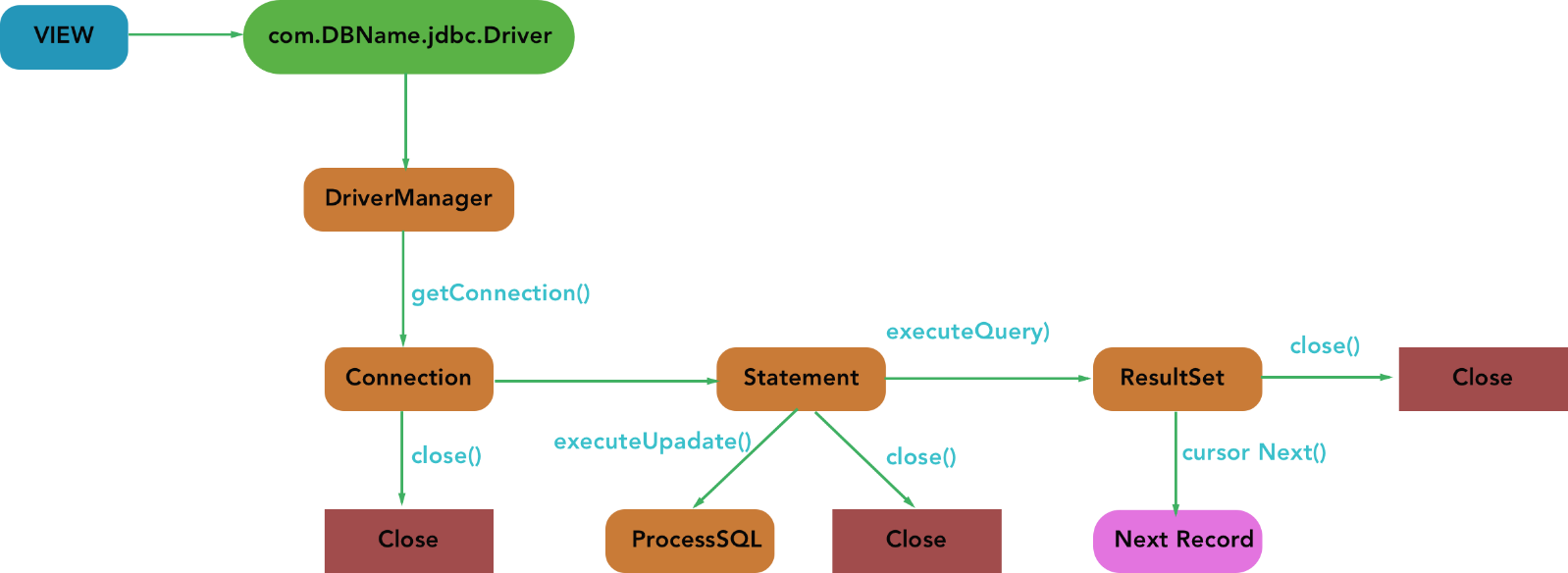
* 1. **Trình bày kiến trúc MVC? Vai trò của các thành phần trong kiến trúc MVC?**

MVC (Model – View – Controller) – MVC Pattern Design là một mô hình kiến trúc phần mềm được tạo ra với mục đích quản lí và xây dựng dự án phần mềm có hệ thống hơn, nói cụ thể thì nó là một mẫu thiết kế nhằm chia tách phần giao diện và phần code của ứng dụng để dễ dàng quản lí, bảo trì và phát triển. MVC được áp dụng trên hầu hết các ngôn ngữ lập trình hướng đối tượng hiện nay như C++, C#, Java, PHP, …

MVC chia ứng dụng phần mềm ra thành 3 phần có tương tác với nhau:

* Model: thường là các class chứa thông tin đối tượng (dữ liệu), tương tác truy xuất database.
* View: là nơi nhận dữ liệu từ model, database và sắp xếp chúng chính xác nhờ yêu cầu từ controller và truyền về client.
* Controller: Đóng vai trò trung gian giữa Model và View, nó có nhiệm vụ nhận yêu cầu từ client sau đó xử lý request, load model và gửi data qua view tương ứng với nhau rồi trả kết quả về cho client.
  1. **Mô tả cách thực hiện tạo và lấy kết quả 1 câu query bất kỳ bằng JDBC?**

JDBC, là viết tắt của Java Database Connectivity, là một Java API chuẩn để kết nối giữa ngôn ngữ lập trình Java và các cơ sở dữ liệu đa dạng. Sử dụng JDBC, bạn có thể thực hiện nhiều tác vụ đa dạng khi làm việc với cơ sở dữ liệu như tạo, xóa cơ sở dữ liệu; tạo và thực thi các lệnh SQL hoặc MySQL; tạo, xóa các bản ghi; …



* Khởi tạo connection

Đăng ký driver: Class.forName("com.mysql.cj.jdbc.Driver");

connection = DriverManager.getConnection(jdbcURL, jdbcUsername, jdbcPassword);

* Tạo đối tượng Statement

Statement (liên lạc chỉ để lấy hàng, không có thêm điều kiện): sử dụng cho các câu SQL query không kèm tham số.

PreparedStatement (liên lạc lấy một mặt hàng nhất định): sử dụng tốt cho các câu SQL lặp lại nhiều lần, chấp nhận tham số truyền vào.

CallableStatement (liên lạc để thay đổi một số mặt hàng trong kho): thướng sử dụng với storeProcedure, chấp nhận tham số truyền vào.

* ExecuteStatement method

Execute (làm việc đa năng): phương thức này sử dụng khi ta không rõ kiểu của statement.

ExecuteQuery (chỉ lấy hàng trong kho): sử dụng chủ yếu cho các câu query select.

ExecuteUpdate (chỉ thêm hàng, hủy hàng, cập nhật hàng): phương thức này sử dụng cho các câu insert, update, delete, chỉ trả về giá trị row count hoặc 0 cho các câu SQL không trả về kết quả gì.

* 1. **Kỹ thuật sử dụng transaction bằng JDBC là như thế nào?**

Transaction (giao tác) là một tiến trình xử lý, có điểm bắt đầu vàđiểm kết thúc

Transaction gồm nhiều phép thực thi nhỏ, trong đó mỗi phép thực thi được thực thi một cách tuần tự và độc lập theo nguyên tắc là tất cả thành công hoặc một phép thực thi thất bại thì cả tiến trình thất bại.

Trong JDBC, Connection Interface cung cấp các phương thức sau để

quản lý transaction:

• void setAutoCommit(boolean status)

• void commit(): Thực hiện thay đổi đối với CSDL và muốn ký thác nó

• void rollback(): Xóa các thay đổi và quay lại trạng thái ban đầu khi thấy có lỗi xảy ra

• setSavepoint(String ten\_cua\_savepoint)

• releaseSavepoint(Savepoint ten\_cua\_savepoint)

• rollback (String ten\_cua\_savepoint)

* 1. **Các tính chất của Transaction?**

Các thuộc tính ACID miêu tả rõ ràng nhất về Transaction. 4 thuộc tính này bao gồm Atomicity, Consistency, Isolation và Durability, trong đó:

• Atomicity nghĩa là tất cả thành công hoặc không.

• Consistency bảo đảm rằng tính đồng nhất của dữ liệu.

• Isolation bảo đảm rằng Transaction này là độc lập với Transaction khác.

• Durability nghĩa là khi một Transaction đã được ký thác thì nó sẽ vẫn tồn tại như thế cho dù xảy ra các lỗi, …

* 1. **Phương pháp gọi 1 SP ở JDBC?**

**Sử dụng CallableStatement ()**

* 1. **Phân biệt Ant, Maven, Gradle?**

|  |  |  |
| --- | --- | --- |
| **Ant** | **Maven** | **Gradle** |
| Một công cụ viết bằng Java, được xây dựng với mục đích làm một build tool uyển chuyển cho các ứng dụng Java. Sử dụng định dạng XML cho tập tin cấu hình. | một công cụ quản lý thư viện phụ thuộc kiêm build tool cho các ứng dụng Java. Khả năng quản lý tốt hơn so với Ant, nhưng kém uyển chuyển hơn. Sử dụng định dạng XML cho tập tin cấu  hình. | Một công cụ quản lý thư viện phụ thuộc kiêm build tool cho  các ứng dụng Java. Được xây dựng trên concept của Ant và Maven. Sử dụng ngôn ngữ Groovy cho tập tin cấu hình. |
|  |  |  |