**Java Interview Questions- Đề Cương chuẩn bị cho phỏng vấn với doanh nghiệp**

<https://github.com/codegymdanang/CGDN-Java-InterviewQuestion>

**Java Core**

1. Lập trình hướng đối tượng là gì?  
   Trả lời: lập trình hướng đối tượng là 1 kỹ thuật lập trình, cho phép lập trình viên trừu tượng hóa các đối tượng thực tế thành các đối tượng trong code.  
   Ví dụ : Trong thực tế để quản lý 1 khách sạn thì mình có các công việc như quản lý khách tới khách sạn, mình có kế toán để thống kê tài chính, mình có bảo vệ. Như vậy lập trình hướng đối tượng thì trong thực tế có đối tượng khách hàng thì trong lập trình mình cũng có đối tượng khách hàng. Trong thực tế khách hàng có tên , tuổi, địa chỉ thì trong lập trình hướng đối tượng mình cũng có tên tuổi địa chỉ. Hướng đối tượng là ngoài thực tế có cái gì thì mình ánh xạ y chang trong lập trình
2. Các tính chất của lập trình hướng đối tượng trong Java?  
   <https://docs.google.com/presentation/d/1K8f4L0PdSqId76n9ueSDDfRxx9_qQm-AaXs6pCV_yAA/edit?usp=sharing>  
   Trả lời: Có 4 tính chất (tính trừu tượng, tính đóng gói, tính kế thừa, tính đa hình)

* Cho ví dụ minh hoạ cho mỗi tính chất  
  + Ví dụ tính trừa tượng  
    Ví dụ khi mình mua hàng online trên mạng như trang lazada, tiki hay amazon. Mình click vào mua sản phẩm sau đó 1 tuần sau mình nhận được sản phẩm của mình. Trừa tượng ở chổ mình không biết lazada, tiki hay amazon sẽ lấy đơn hàng của mình, đóng gói ra sao và họ vận chuyển như thế nào qua các thành phố và quốc gia. Mình chỉ biết là khi mình bấm vô nút mua hàng thì mình sẽ nhận được hàng mà không cần quan tâm họ làm như thế nào đế ship hàng mình về đúng địa chỉ . Trong lập trình cũng vậy khi mình gọi một phương thức từ một đối tượng mình chỉ quan tâm giá trị cung cấp cho phương thức đó và kết quả của phương thức đó trả về, mình không biết code thực sự bên trong phương thức đó làm gì
  + Ví dụ tính đa hình
  + 1 phương thức có thể được thực hiện theo nhiều cách khách nhau (ovverloading)  
    Ví dụ cũng là phương thức chạy thì con gà chạy bằng 2 chân nhưng con chó chạy bằng 4 chân. Cùng một hành động chạy nhưng ở những ngữ cảnh khác nhau thì hành động khác nhau
  + Ví dụ tính kế thừa
  + Ví dụ như con kế thừa tài sản của cha. Hoặc ví dụ khác như có lớp Animal có các thuộc tính tên, tuổi , và phương thức di chuyển. Nếu con chó kế thừa Animal thì nó sẽ kế thừa các thuộc tính tên , tuổi , và phương thức của lớp cha Animal. Chúng ta không cần phải khai báo lại biến tên, tuổi trong lớp con chó.
  + Lợi ích của tính kế thừa Lớp con (lớp A) có thể tận dụng lại các thuộc tính và phương thức của lớp cha (lớp B) (nghĩa là các thuộc tính và phương thức của lớp B có thể được tái sử dụng bởi lớp A). Lớp A có thể định nghĩa thêm thuộc tính và phương thức mới của riêng nó và có thể định nghĩa lại (hay còn gọi là ghi đè phương thức, overriding) phương thức được kế thừa từ lớp B cho phù hợp với mục đích của nó.
  + Ví dụ tính đóng gói  
    Tính đóng gói hay còn goi là che dấu dữ liệu . Đối với những dữ liêu quan trọng chúng ta không muốn cho các lớp khác truy cập và sử dung thì ta hạn chế không cho các lớp khác truy cập vào biến hay phương thức bảo mật

1. Hỏi về collection Framework (Cái này hầu như 100% ở đâu phỏng vấn cũng hỏi)  
   <https://docs.google.com/presentation/d/1AhJ3XlgEFJIJW8ETvummQBoUkxT2_z5NjJ8XvQoJj7M/edit?usp=sharing>  
   3.1 ArrayList là gì ? Khi nào dùng ArrayList  
   3.2 LinkedList là gì ? Khi nào dùng Linkedlist...  
   3.2 Vector là gì ? Khi nào dùng Vector?  
   3.3 Stack hoạt đông như thế nào ? Khi nào dùng Stack  
   3.4 Queuu hoạt đông như thế nào ? Khi nào dùng Queue
2. Là tập hợp dữ lệu fist in fist out (vào trươc ra trước  
   3.5 Set hoạt đông như thế nào ? Khi nào dùng Set
3. Set là tập hợp dữ liêu mà mỗi phần tử xuất hiện 1 lần duy nhất   
   3.6 Map hoạt đông như thế nào ? Khi nào dùng Map
4. 3.7 Phân biệt TreeSet, HashSet và EnumSet ? Khi nào thì dùng  
   3.8 Phân biệt ArrayList , Linkedlist và Vector ? Khi nào thì dùn 3.9 Sự khác nhau giữa Set với List  
   4.10 Sự khác nhau giữa ArrayList và Array  
   * Array fix cứng giá trị mảng
   * ArrayList có thể co giản được

* Sự khác nhau giữa Linkedlist với Arraylist
* Sự khác nhau của Set và <http://levunguyen.com/2019/08/05/su-khac-nhau-giua-linkedlist-va-arraylist-trong-java/>
* Vector với Arraylist
* HashTable với HashMap  
  <https://viettuts.vn/interview/list-cau-hoi-phong-van-java-collection>  
  3.11 Thuật toán BIG O là gì ?  
  3.12 Trong khi thao tác với các collection (ArrayList,LinkedList, như insert , remove , finding thì cái nào nhanh hơn , cái nào chậm hơn.

1. Sự khác nhau giữa Hashcode và Equals.  
   Trả lời: <https://stackjava.com/java/hashcode-va-equals-trong-java.html>
2. Sự khác nhau giữa override và overload

* Override : Overriding xuất hiện khi trong class con tồn tại một method được định nghĩa với cùng tên và cùng số lượng tham số với method của class cha
* Overload : Overloading xuất hiện khi trong một class tồn tại một method được định nghĩa với cùng tên nhưng khác số lượng tham số đầu vào với method của class cha.

1. Generic là gì? Cho ví dụ minh hoạ? Tại sao dùng Generic  
   <https://viettuts.vn/java-new-features/generics-trong-java>
2. Immutable là gì? Cách để tạo đối tượng immutable?  
   Trả lời: <https://stackjava.com/java/immuable-la-gi.html>
3. Sự khác nhau giữa abstract class và Interface.  
   Trả lời: <http://levunguyen.com/2019/06/18/khac-biet-giua-abstract-va-interface/>
4. Hỏi về thuật toán.

* Tìm kiếm nhị phân
* Thuật toán sắp xếp

1. Phân biệt các kiểu dữ liệu trong java. Khác nhau thế nào ?
2. Phân biệt sự khác nhau của các access modifier : private , default , protected , public  
   Trả lời :

* Khi một phương thức hoặc biến được khai báo là public, có nghĩa là tất cả các class khác, kể cả các class không thuộc cùng package đều có thể truy cập
* Khi một phương thức hoặc biến được khai báo là private nó sẽ không thể truy cập từ class khác, kể cả các class cùng source file hay các class con.
* Khi một phương thức hoặc biến được khai báo là default thì chỉ có các class thuộc cùng package với nó mới có thể truy cập.
* Protected modifier khá giống với default modifier, nó hạn chế khả năng truy cập trong cùng 1 package, tuy nhiên với protected modifier thì nó còn cho phép truy cập từ các class con kể cả khi class con không nằm cùng package với class cha. (truy cập theo trường hợp thừa kế)
* Static : khi mình muốn chia sẽ (dùng chung, và là duy nhất tron cả hệ thống) cái biến đó cho các object khác có thể sử dụng được
* Final : Khi mình muốn giá trị là hằng số và không thể thay đổi được (Ví dụ final double PI = 3.14)

1. Ngoại lệ (Exception là gì) ? <https://docs.google.com/presentation/d/1gJy6TvLc5RR6Rpx8G2JpEvN2GJejrhKwelRSJnikutg/edit?usp=sharing>  
   Khi chúng ta lập trình thì sẽ có những trường hợp có thể xảy ra lỗi . Ví dụ lấy 1 số nguyên chia cho 0. Khi mình thực thi chương trình thì nó sẽ báo ra lỗi Java.lang.ArithmeticException . Trong lập trình mình có rất nhiều Ngoại lệ . Nếu mình đoán được các ngoại lệ có thể xảy ra thì mình có thể dùng try và catch để xử lý ngoại lệ đó. Ngoại lệ rất nguyên hiểm có thể làm chương trình bị đứng . Bắt được ngoại lệ và giải quyết nó giúp chương trình tiếp tục chạy như bình
2. Check va Uncheck la gi?  
   <https://docs.google.com/presentation/d/1gJy6TvLc5RR6Rpx8G2JpEvN2GJejrhKwelRSJnikutg/edit?usp=sharing>
3. Try/Catch/Finally
4. Phân biệt pass-by-value (tham trị) và pass-by-reference (tham chiếu) và bộ nhớ máy tính  
   <https://docs.google.com/presentation/d/1swUT9shoRZk9VZy3wuxL98IIfShYt-g-Sf1fyP9rbVI/edit?usp=sharing>

* CPU xử lý dữ liệu thông qua địa chỉ bộ nhớ nên thứ được truyền vào hàm luôn luôn là địa chỉ bộ nhớ chứ không phải là giá trị.
* Khi chương trình thực thi, dữ liệu trên RAM có thể được lưu trữ trên stack hoặc heap nhưng việc tham chiếu bằng địa chỉ giữa các biến là như nhau nên để cho đơn giản mình sẽ giả sử chúng chỉ được lưu trữ trên stack.
* Pass by value : nghĩa là mình sẽ clone (tạo ra một giá trị mới bằng cách copy (nhân bản) giá trị gốc) giá trị mới từ giá trị gốc và mình chỉ thao táo với bản copy. Khi chúng ta thay đổi các giá trị của đối tượng copy thì không ảnh hưởng đến giá trị gốc. Pass-by-value được hiểu là khi bạn thay đổi biến trong hàm thì ngoài hàm sẽ không bị ảnh hưởng. Nó giống như bạn copy giá trị của biến vào biến khác rồi truyền vào hàm.
* Pass by reference : Ngược lại với Pass by value . Giá trị gốc sẽ bị thay đổi Pass-by-reference là khi bạn thay đổi biến trong hàm cũng làm ngoài hàm bị ảnh hưởng. Nó giống như bạn truyền đúng địa chỉ của biến đó vào hàm.

1. Bộ nhớ máy tính . HEAP và STACK
2. Sự khác nhau giữa Class và Object
3. Enum là gì cho ví dụ ?

**Java Spring**

1.Nguyên lý Inversion of Control là gì?  
IOC : nguyên lý đảo ngược điều khiển. Chúng ta giao cho framework sử lý. Ví dụ như kết nối database, chúng ta giao cho framwork chịu trách nhiệm và điều kiển. <http://acegik.net/blog/java/spring/core/tong-quan-ve-ioc-trong-spring.html> <https://techblog.vn/inversion-of-control-nguyen-ly-cua-cac-nguyen-ly>

1. IOC Container trong Spring IoC Container trong Spring chính là lõi của Spring Framework. IoC Container sẽ tạo ra các đối tượng, nối chúng lại với nhau, cấu hình chúng, và quản lý vòng đời của chúng từ khi tạo ra đến khi bị hủy. IoC Container sử dụng DI (Dependency Injection) để quản lý các thành phần tạo nên một ứng dụng. Những đối tượng này được gọi là Spring Bean. IoC Container được cung cấp thông tin từ các tập tin XML hoặc từ Java Annotation  
   Có hai loại IoC Container, đó là: BeanFactory , ApplicationContext
2. Khái niệm DI (dependency injection) là gì? Cho ví dụ minh hoạ

Là phương thức tiêm sự phụ thuộc thừ các lớp con kế thừa , VD interface A , b extent A , @ Auturaie A A , A chứa các phương thức kế thừa từ A mà k cần khai báo B

1. AOP là gì ? Cho ví dụ minh hoạ

**AOP** là từ viết tắt của **Aspect Oriented Programming**, dịch ra tiếng Việt là “phương pháp lập trình hướng khía cạnh”. AOP là một kỹ thuật lập trình cho phép phân tách chương trình thành cách module riêng rẽ, không phụ thuộc nhau. Khi hoạt động, chương trình sẽ kết hợp các module lại để thực hiện các chức năng nhưng khi sửa đổi chức năng thì chỉ cần sửa đổi trên một module cụ thể.

1. SpringBoot làm gì ?

Là dự án phát triển trên ngôn ngữ JPA trong Spring Framwork ,giúp đơn giản quán quá trình lập trình tập trung vào business ứng dụng

1. Vòng đời của 1 Bean

khởi tạo(init), sử dụng và kết thúc

1. Scope của 1 bean gồm những scope nào ?
2. Giải thích luồng đi của ứng dụng springmvc ?

Spring Web MVC model-view-controller framwrok được thiết kế sung quanh một dispatcherServerlet xử lý tất cả các HTTP yêu cầu và HTTP phản hồi lại.

Công nghệ Servlet được sử dụng để tạo ra ứng dụng web (nằm ở phía máy chủ và tạo ra trang web động).

Sử dụng Servlet, bạn có thể thu thập thông tin đầu vào từ người dùng thông qua các trang web, hiển thị các bản ghi từ một cơ sở dữ liệu hoặc một nguồn khác.

1. Giải thích @Autowire làm gì ?

Là 1 dependence ijection để tạo ta đối tượng con phụ thuộc mà không cần phải khai báo đối tượng đó

1. Giải thích @Controller , @Service , @Repository , @Resfult

Controller là lớp kế thừ từ interface Controller sd @Controller để kế thừ những phương thưc của inerface Controller , để giảm số lượng lớp

@Service để kế thừa những phương thức có sẵn từ Repository của Spring Framwork

1. Giải thích JPA là gì ?

Là 1 bộ đặc tả những tiêu chuẩn của java , mô tả các thao tác quản lý CSDL quan hệtrong các ứng dung java

1. Webservice là gì ?

Một web service là cách giao tiếp giữa hai hoặc nhiều ứng dụng được phát triển cho các nền tảng khác nhau. Các service này độc lập với các trình duyệt và hệ điều hành qua đó dễ dàng giao tiếp và nâng cao hiệu suất để phù hợp với nhiều người dùng. Service này có thể như một hàm, một bộ sưu tập các tiêu chuẩn hoặc các giao thức được triển khai trên server. Nó là một giao tiếp giữa một client và server hoặc thông tin liên lạc giữa hai thiết bị thông qua kết nối mạng.

1. Restful Webservice là gì ?

Nó là một kiểu kiến ​​trúc có thể được sử dụng để thiết kế các dịch vụ web, có thể được sử dụng từ nhiều khách hàng khác nhau. Ý tưởng cốt lõi là, thay vì sử dụng các cơ chế phức tạp như CORBA, RPC hoặc SOAP để kết nối giữa các máy, HTTP đơn giản được sử dụng để thực hiện các cuộc gọi giữa chúng

1. Maven dùng để làm gì ?

là mô hình quản lý dự án

1. Sự khác nhau giữa Session và Cookie ?

Session là 1 phiên làm việc được lưu trong client nếu tắt trình duyệt session saex mất đi

Cookie là 1 biến được lưu trong Server , có tuổi thọ , nếu hết tuổi thị cookie sẽ mất đi

1. Cache nghĩa là gì ?
2. Mô hình MVC là gì ? Nhiệm vụ của M là gì , V là gì ,và C là gì ?

M = Model gửi đi các objiect có trong view ,

V= View trả về trang view nào đó ,

C= Controller sử sụng các phương thức costrong interface Controller

1. Truyền dữ liệu từ Controller tới View thì dùng cái gì ?

Model , ModelAddView;

1. JWT là gì ?  
   <https://docs.google.com/presentation/d/1p_BXFc46BCcJpMUKAOr3vcH41JhakCbnH6_7-qSQjVw/edit#slide=id.p2>

**HTML**

1. HTML là gì ?  
   Trả lời : HTML viết tắt của Hyper Text Markup Language (ngôn ngữ đánh dấu siêu văn bản). Nó là một ngôn ngữ của World Wide Web. Đây là một ngôn ngữ định dạng văn bản chuẩn được sử dụng để tạo và hiển thị các trang trên Web.
2. Khai báo <!DOCTYPE> trong HTML có tác dụng gì?
3. Điểm khác biệt giữa “visibility:hidden” và “display:none”?
4. Thẻ “div” và thẻ “span” khác nhau thế nào?
5. Canvas trong HTML 5 là gì?  
   Trả lời : Canvas là một khoảng trống HTML được sử dụng để vẽ đồ họa.

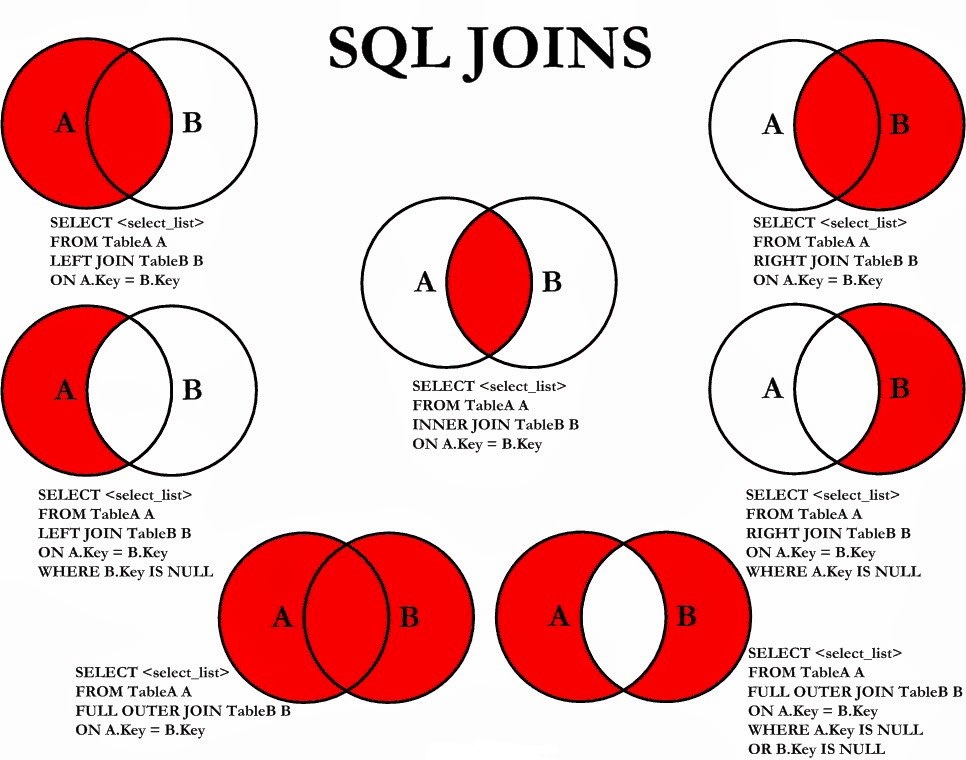
**Java Script**

1. Javascript dùng để làm gì?  
   Trả lời : javascript là ngôn ngữ lập trình kịch bản hướng đối tượng được phát triển bởi Netscape Communications cho các ứng dụng client/server. Javascript là một ngôn ngữ lập trình thông dịch. JavaScript hiển thị các trang web theo cách tương tác và năng động góp phần tạo nên trang web động. Điều này cho phép các trang bắt các sự kiện, hiện thị các hiệu ứng đặc biệt, tự động tạo nội dung HTML, xác thực dữ liệu, tạo cookie, lấy thông tin trình duyệt của người dùng
2. “this” trong Javascript dùng để làm gì?
3. Jquery là gì ? Lợi ích của Jquery trong làm web?

**CSS**

1. CSS framework là gì?
2. Phân biệt Class và ID như thế nào trong CSS?  
   Trả lời : ID là duy nhất, ID hay index (chỉ số) được dùng để xác định một đối tượng duy nhất nào đó. Mỗi đối tượng chỉ có thể có một ID. Mỗi trang chỉ có thể có một đối tượng với một ID nào đó.  
   Class không duy nhất  
   Class dùng để chỉ một lớp các đối tượng có chung các thuộc tính. Nhiều đối tượng có thể thuộc trong cùng một class. Nhiều class có thể được áp dụng lên cùng một đối tượng.  
   Class dùng để chỉ một lớp các đối tượng có chung các thuộc tính. Nhiều đối tượng có thể thuộc trong cùng một class. Nhiều class có thể được áp dụng lên cùng một đối tượng.
3. Phân biệt sự khác nhau của padding và margin
4. Thuộc tính float:left là gì ?
5. Thuộc tính clear:both dùng để làm gì ?
6. Phân biệt position : static , absolute , relative , fixed
7. Bootstrap là gì ? Lợi ích của bootstrap <https://kipalog.com/posts/Tim-hieu-thuoc-tinh-position-trong-CSS>
8. Hỏi về các dự án bạn đã làm, nghiên cứu.  
   Dự án ở đây là các bài tập lớn bạn đã làm, đồ án tốt nghiệp. Để tạo ấn tượng tốt bạn nên có 1 số project trên github đối với các bài tập lớn của mình hoặc tham gia viết bài trên một blog nào đó về lập trình.

**Database**[**https://www.softwaretestinghelp.com/database-interview-questions/**](https://www.softwaretestinghelp.com/database-interview-questions/)

1. Database là gì ?
2. Trong database có các mối quan hệ gì ?
   * Quan hệ 1 1 (One-to-one): One table has the relationship with another table having the similar kind of column. Each primary key relates to only one or no record in the related table.
   * Quan hệ 1 nhiều (One-to-many): One table has a relationship with another table that has primary and foreign key relation. The primary key table contains only one record that relates to none, one or many records in the related table.
   * Quan hệ nhiều nhiều (Many-to-many): Each record in both the tables can relate to many numbers of record in another table.
3. SQL là gì?  
   Trả lời: SQL là viết tắt của cụm từ "Structured Query Language." - ngôn ngữ truy vấn mang tính cấu trúc. Nó được thiết kế để quản lý dữ liệu trong một hệ thống quản lý cơ sở dữ liệu quan hệ (RDBMS). SQL là ngôn ngữ cơ sở dữ liệu, được dùng để tạo, xóa, lấy các hàng và sửa đổi các hàng, … Tất cả hệ quản trị cơ sở dữ liệu (Database Management System - DBMS) như MySQL, Oracle, MS Access, Sybase, Informix, Postgres và SQL Server đều sử dụng SQL như là ngôn ngữ cơ sở dữ liệu chuẩn.
4. Định nghĩa từ khóa "JOIN" và các loại "JOIN" khác nhau?  
   Trả lời: Từ khóa "JOIN" được sử dụng để lấy dữ liệu từ 2 hay nhiều bảng liên quan. Đây là phép kết hợp các dòng dữ liệu từ nhiều bảng lại với nhau. Khi bạn cần truy vấn các cột dữ liệu từ nhiều bảng khác nhau để trả về trong cùng một tập kết quả, bạn cần dùng từ khóa "JOIN". Các loại JOIN như inner join , outer join : leftjoin , right join and fulljoin  
   [](https://user-images.githubusercontent.com/37821007/68453090-22db5100-0227-11ea-8dc0-da75b5934648.jpg)
5. Primary key là gì?  
   Trả lời: Một PRIMARY KEY- Khóa chính là một trường trong một bảng mà nhận diện một cách duy nhất mỗi hàng/bản ghi trong một bảng dữ liệu. Các PRIMARY KEY phải chứa các giá trị duy nhất, không được sử dụng lại. Một cột là PRIMARY KEY thì không được phép có giá trị NULL. Một bảng chỉ cho phép tối đa một PRIMARY KEY, bao gồm một trường đơn hay nhiều trường. Mỗi bảng đều cần có khóa
6. Foreign keys là gì?  
   Trả lời: FOREIGN KEY - Khóa ngoại là một trường trong bảng cơ sở dữ liệu, đó là khóa chính trong một bảng khác được thêm vào tạo ra sự liên quan giữa hai bảng. Trong bảng, giá trị của FOREIGN KEY có thể chấp nhận giá trị NULL, cũng như các giá trị trùng nhau.
7. Composit key là gì ? is a form of the candidate key where a set of columns will uniquely identify every row in the table.
8. Unique key là gì? A Unique key is same as the primary key whose every row data is uniquely identified with a difference of null value i.e. Unique key allows one value as NULL value.
9. CHECK Constraint - Ràng buộc CHECK là gì?  
   Trả lời: Một ràng buộc CHECK được sử dụng để giới hạn các giá trị hoặc kiểu của dữ liệu có thể nhập và lưu trữ trong một trường của bản ghi. Nếu bản ghi không đáp ứng được điều kiện này, thì sẽ không được lưu trữ vào trong bảng.
10. STORED PROCEDURE là gì?  
    Trả lời: STORED PROCEDURE là một tập hợp các câu lệnh SQL dùng để thực thi một nhiệm vụ nhất định. Nó hoạt động giống như một hàm trong các ngôn ngữ lập trình khác. STORED PROCEDURE là một khái niệm khá phổ biến và được hầu hết các hệ quản trị cơ sở dữ liệu (DBMS) hỗ trợ, tuy nhiên không phải tất cả đều hỗ trợ STORED PROCEDURE.
11. NORMALIZATION - Chuẩn hóa dữ liệu là gì?  
    Trả lời: Quá trình thiết kế bảng để giảm thiểu sự dư thừa số liệu, loại bỏ các thay đổi dữ liệu dị thường (update anomaly, insertion anomaly, deletion anomaly) được gọi là quá trình chuẩn hóa dữ liệu. Để thực hiện quá trình này, chúng ta cần phải chia một cơ sở dữ liệu thành hai hay nhiều bảng và xác định các mối quan hệ giữa chúng. <https://docs.google.com/presentation/d/19Q2DYGId0Ns4OjivpGeA8gFNCHtw_Hg8ADHlr-MqWHA/edit?usp=sharing>

1. TRIGGER là gì?  
   Trả lời: TRIGGER được hiểu đơn giản là một thủ tục (một tập các câu lệnh SQL được lưu trữ trong CSDL) được thực thi từ phía máy chủ cơ sở dữ liệu (CSDL) khi có một sự kiện xảy ra như Update, Insert hay Delete. TRIGGER thường dùng để kiểm ra các ràng buộc toàn vẹn trên CSDL, và chúng được thực thi một cách tự động mà không cần sự can thiệp bằng các thao tác thủ công như kiểm tra dữ liệu, đồng bộ hóa dữ liệu
2. Sự khác nhau giữa các câu lệnh TRUNCATE, DELETE và DROP?  
   Trả lời: Lệnh DELETE có tác dụng xóa một số hoặc tất cả các hàng từ một bảng dựa trên những điều kiện được chỉ định. Các bản ghi này cũng có thể được phục hồi lại (Roll back).  
   Lệnh TRUNCATE xóa toàn bộ các bản ghỉ từ bảng bằng cách phân chia lại bộ nhớ các trang. Xử lý này không thể được phục hồi.  
   Lệnh DROP xóa hoàn toàn một bảng từ cơ sở dữ liệu.
3. Transaction là gì ?  
   Sequence of operation performed which changes the consistent state of the database to another is known as the database transaction. After the completion of the transaction, either the successful completion is reflected in the system or the transaction fails and no change is reflected
4. Có bao nhiêu loại câu lệnh SQL được sử dụng many SQL statements are used? Câu lệnh SQL được chia ra thành 3 loại DDL , DML và DCL  
   Data Definition Language (DDL) : Các câu lệnh định nghĩa cấu trúc để lưu trữ dữ liệu.

* CREATE to create a new table or database.
* ALTER for alteration.
* Truncate to delete data from the table.
* DROP to drop a table.
* RENAME to rename a table.  
  Data Manipulation Language (DML): Các câu lệnh thao tác với database
* INSERT to insert a new row.
* UPDATE to update an existing row.
* DELETE to delete a row.
* MERGE for merging two rows or two tables.  
  Data Control Language (DCL): Các câu lệnh dùng để quản lý permission trong cơ sở dử liệu
* COMMIT to permanently save.
* ROLLBACK to undo the change.
* SAVEPOINT to save temporarily.  
  12.Model ERD nghĩa là gì ? Ans: E-R model is an Entity-Relationship model which defines the conceptual view of the database.

E-R model basically shows the real world entities and their association/relations. Entities here represent the set of attributes in the database.

**Devops**

1. Server nghĩa là gì ?
2. Deploy một ứng dụng spring boot lên server như thế nào ?
3. Jenkins dùng để làm gì ?
4. Sonarque dùng làm gì ?
5. Tomcat dùng làm gì ?

**Good Coding**

1. Unit Test là gì ?
2. Lập trình TDD là như thế nào
3. Sonarque có chức năng làm gì ?
4. Bạn có biết design pattern là làm gì ? <https://docs.google.com/presentation/d/1VfKg18KaR_sgct4YJamJLsh86IW44HHdGv1ImhC5-Jo/edit?usp=sharing>  
   Hỏi về các design pattern (1 số design pattern quen thuộc, chọn 1 cái từ category là creation, 1 cái từ structural , 1 cái từ behavior).

* Factory Pattern
* Abstract Factory Patter
* Builder
* Singleton
* Decorator
* State
* Strategy

**Angular**

* Angular là gì ?
* Giải thích Scope trong Angular ?
* Data binding trong angular là gì ?  
  <https://malcoded.com/posts/angular-data-binding/>
* Directive trong angular là gì >
* Vòng đời của một component ? <https://techblog.vn/lifecycle-hooks-in-angular>
* Rounting là gì ?
* Gọi webservice như thế nào ? Sử lý kết quả trả về từ webservice như thê nào?

**Quản lý công việc**

* Kanban dùng để làm gì ?
* Nguyên lý hoạt động ra sao

**Hỏi về qui trình Scrum**

1. Scrum là gì ?
2. Có bao nhiêu roles trong Scrum
3. Có mấy loại meeting trong scrum ?
4. Planing Pocker dùng để làm gì ?
5. Vì sao nên dùng Scrum ?
6. Product Backlog là gì ?
7. Sprint Backlog là gì ?
8. Sprint Retro dùng để làm gì ?
9. 3 câu hỏi trong daily Scrum là gì ?

**Hỏi về khả năng tự học, tiếp cận/giải quyết vấn đề.**

Cái này chủ yếu về cách mà bạn giải quyết vấn đề, tiếp cận cái mới. Thí dụ cách mà bạn search google cũng được đánh giá khá cao.

Ví dụ: nhà tuyển dụng hỏi bạn: “em làm chức năng Login” hết bao lâu thời gian thì đừng trả lời mà phải hỏi lại là chức năng login như nào, áp dụng công nghệ nào, các role, permission ra sao…

Hỏi về những kỹ thuật, ngôn ngữ nào mà bạn tự học, tìm hiểu?

Đặc biệt nếu bạn đọc nhiều sách về lập trình thì sẽ được đánh giá rất cao. Một số sách hay về Java như: Sun Certified Programmer for Java®Platform; Clean Code, Java Effective.

* Ngoài những cái ở trên Bạn biết những kỹ thuật, công nghệ, framework gì nữa ?