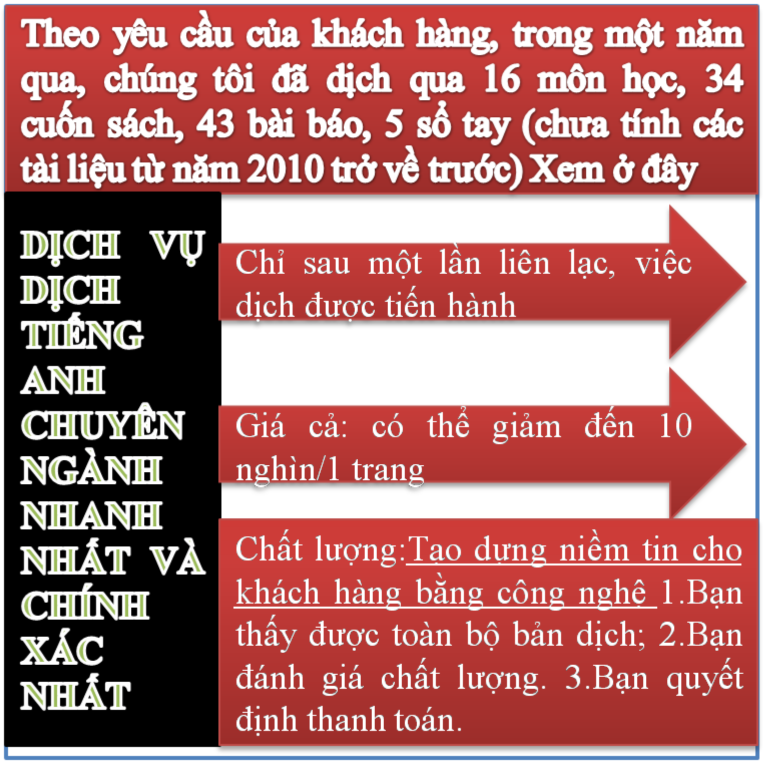

Describe : check 12/8	**Mô tả:**
-Two-Tier : The Traditional client/server database architecture implements a two-tier approach. The application logic exists in either the user interface or within the database server or both. The location of the partitioned processes gave rise to terms such as “fat client” and “fat server” depending on the location of the bulk of the application processing. Most client/server systems implement the fat client approach. Regardless of the version of two-tier implemented the system can face scalability, performance and flexibility problems. For larger systems it is better to adopt a three-tier or multi-tier approach.	**Hai mức:** Kiến trúc cơ sở dữ liệu client/server truyền thống sử dụng mô hình hai mức. Các xử lý đặc trưng của ứng dụng có thể nằm trong giao diện người dùng hoặc trong server cơ sở dữ liệu hoặc cả hai. Vị trí của các quá trình được phân vùng tạo ra những thuật ngữ mới, chẳng hạn như “fat client” và “fat server” phụ thuộc vào vị trí của khối xử lý ứng dụng. Phần lớn các hệ thống client/server sử dụng mô hình fat client. Dù phiên bản mô hình hai mức được áp dụng, hệ thống vẫn gặp vấn đề về khả năng mở rộng, hiệu suất hoạt động và độ linh hoạt. Với những hệ thống lớn hơn, tốt hơn là áp dụng phương pháp ba mức hoặc đa mức. Client: máy khách hay máy trạm Server: máy chủ
-Three-Tier: There are many benefits to three-tier systems, they are more scalable and easier to control. The middle tier enables the system to handle more client connections, implement better security and provide easier maintenance. Connectivity to heterogeneous data sources is greatly simplified as the required database drivers are contained at a single location, requiring fewer client licenses. It is with three tier and multi-tier architectures that the future of client/server database applications lie, true database independence will not be possible without three tier architectures.	[redacted]
-Peer-Peer “DOM” The Distributed Object Model does not	

distinguish between clients and servers. Each object can be a server to a client and likewise each object can be a client of another object. This promotes high levels of scalability and flexibility. Within the Distributed Object Model, objects provide interfaces for the services that they offer to middleware. Clients then makes requests to the middleware which in turn acts as an intermediary and passes messages between suitable objects that can meet the client object's requests.	[redacted]
Discuss the Transparency , Scalability, Performance , Flexibility , Reliability of two-tier ,Three-Tier :	[redacted]
- ***Transparency*** **:** The two-tier model offers a lower degree of transparency because the logical components are more tightly coupled than in the three-tier model. In contrast the 3-tier model decouples presentation/logic/data. As such the data layer can be changed without needing to change the presentation (applications). For example, a different data storage technology may be employed or data storage may be replicated / distributed. Transparency within the distributed object model is enhanced as object services are decoupled through the use of interfaces. Any object could potentially be a server to another object through its exposed interfaces Clients do not necessarily need to know which individual server objects will be used but instead can make a request to	[redacted]

middleware. The middleware can then choose from suitable available objects.	[redacted]
- ***Scalability*** : The three-tier model is potentially more scalable and reliable, as replication can be performed at either the middle or back-end tiers. The full separation of business logic, UI and data management / storage enables the appropriate processing power to be provided for each tier, incrementally added if necessary to avoid performance bottlenecks as the system is scaled up. Conversely, the tight coupling of clients to server within the 2-tier model inhibit easily adding additional resources without alerting large portions of the enterprise application .The Distributed Object Model is highly scalable as new resources can be added as required without the need to alter the application. New resources would simply need to make themselves available to the middleware.	[redacted]
- ***Performance*** : The two-tier model has lower performance when dealing with enterprise applications. This is due to the business processing tasks having to be performed by either the server or the client. When business processing effort is substantial the 2-tier approach can prove a limitation on performance. In contrast the 3-teir model has business processing performed by 1 or more separate resources (within the middle tier). Thus the total effort is distributed	[redacted]

among elements improving performance. The distributed object model offers similar increases in performance as objects can be created within the DIS as required. Objects can also easily be migrated within the DIS to enhance performance. - ***Reliability :*** Reliability is reduced in the 2-tier approach as clients are tightly coupled to server. In the 3-teir architecture the existence of multiple servers provides more potential redundancy and increases the number of points of failure. Reliability can be improved in the distributed object model providing enough objects exist to provide redundancy **Benefit and limitation of 2-Tier ,3-Tier ,DOM in regard to following characteristics Transparency , Scalability, Performance , reliability** ***-2-tier*** Low degree of scalability. [1] Good performance with few users but suffers with large user pools due to its lack of scalability, [2] Low scalability[1] Lack of redundancy negatively impacts reliability and no-transparency is provided hence additional resources need to be tightly coupled.[2] ***-3-tier*** Much more scalable, adding additional servers to layers is simple.[1] Performance is better with large numbers of users due mostly to the	[redacted]

enhanced scalability. [2] More reliable due to redundancy of resources. [1] High transparency inherent in the model. [1]	[redacted]
-DOM Much more scalable, additional objects can be created as required. [1] Performance is better due to the enhanced scalability. [1] Highly transparent. depending on the nature of the technique, RMI v SOAP etc. [2]	[redacted]
Some problems with reliability may exist with respect to possible due to the ad-hoc construction of dependency chains, otherwise reliability is good.[2] The main circumstances that effect these issues are the size of the system large/small – these should be included where	[redacted]
Network , Failure , Implementation , Location , Access , Concurrency Transparency (Explain the meaning of Access, Location, Failure, Concurrency and Implementation transparency giving suitable examples of how it is achieved and why it is required in a Distributed Information System.)	[redacted]
-Location transparency [4 marks] i) Location of resources can vary without the user needing to know. ii) Can be achieved using directory service techniques. iii) Useful in many ways, can change location for example without the need to inform all users of the resource.	[redacted]

-Failure transparency [4 marks] i) Failures are not visible to users. ii) achieved through replicated services. iii) Important as failures in a DIS will likely be partial and can be made transparent with careful management. ***-Implementation transparency [4 marks]*** i) The implementation of the resource is not important. instead its interfaces and behaviour are important. ii) One method is through the use of middleware, whereby common access / location transparency is achieved and resources can be altered / swapped without the need to update user processes. iii)Important as it promotes flexibility / portability / scalability. Processes can be altered without impacting on other parts of the DIS ***-Network transparency*** i) This refers to a combination of access transparency (i.e. access local and remote resources through similar methods) and location transparency (i.e. access to a resource without knowledge of its actual physical location). Network transparency is achieved when a user does not need knowledge of the network, instead accessing resources using similar access methods and without the need to know the location of the resource.	[redacted]

ii)VFS example, VFS uses similar access method (access transparency) and a directory or recourse locations (location transparency) iii) Useful as it reduces complexity, improves scalability and flexibility.	[redacted] truy cập [redacted]
-Access transparency	**-Tính trong suốt truy cập**
i) access to remote and local resources using same operations ii) VFS example iii) Useful as it removes complexity as users and software have a single method rather than potentially lots of different methods. Improves scalability.	i)truy cập các nguồn tài nguyên cục bộ và từ xa bằng các thao tác như nhau ii)Ví dụ VFS iii)Hữu dụng vì nó loại bỏ tính phức tạp khi người dùng và phần mềm có một phương pháp duy nhất thay vì nhiều phương pháp khác nhau. Cải thiện khả năng mở rộng
-Concurrency transparency -Users of a resource can access resources seemingly concurrently while the resources themselves might maintain single write/read access at various points. -This is achieved through read/write locking and exclusion. Updates can be written to be applied once locks are released. -This is important as multiple access to resources will occur and ideally process should not be delayed while waiting for access to shared resources.	[redacted]
a) What are the main advantages and disadvantages of replication within a Distributed Information	[redacted] **Ưu và nhược điểm** [redacted]

System? **[6 marks]** (a) Advantages: Fault tolerance, scalability, performance Disadvantages: Consistency and communication Depending on the type of access (read or write) communication overheads involved with maintaining consistency could become expensive. This is especially so if there are many write updates and a strict consistency model is required.	[redacted] [redacted]
b) Briefly compare and contrast the features of passive and active replication in a Distributed Information System.	[redacted]
(b) The passive replication uses a primary replication manager and secondary replica managers back-ups or slaves, the main concept is that the primary replication manager sends updates to slaves and should it fail a slave can then become the primary manager Replicas have the same role and organize themselves, the front ends multicast requests to the replica group and the replicas are then process independently and reply, as such the front end receives many responses.	[redacted]
Passive mainly provides fault tolerance, whereas active replication can enhance performance as many replicas each vie for a fastest response.	Sự sao chép bị động chủ yếu cung cấp khả năng kháng lỗi trong khi sự sao chép chủ động có thể nâng cao hiệu suất hoạt động vì mỗi bản sao cạnh tranh để nhận hồi đáp nhanh nhất.