TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC
—— o0o ——

BÁO CÁO MÔN HỌC
CƠ SỞ DỮ LIỆU NÂNG CAO

Sinh viên thực hiện: Hoàng Thanh Lưu
Mã số sinh viên: 20162602
Lớp: Toán Tin K61

Hà Nội - 6/2020

Lời nói đầu

Ngày nay, trong xu thế phát triển không ngừng của công nghệ thông tin, đặc biệt trong giai đoạn Cách mạng công nghiệp 4.0, các công ty, doanh nghiệp đặc biệt quan tâm tới dữ liệu. Khối lượng dữ liệu cần xử lý và phân tích dần dần tích lũy nhiều hơn đòi hỏi ngành khoa học nghiên cứu dữ liệu lớn ra đời. Cùng với đó, các cộng nghệ về khai phá dữ liệu lớn, các hệ quản trị cơ sở dữ liệu được đưa ra nhằm giải quyết các vấn đề về dữ liệu, dần trở thành nhu cầu thiết yếu của nghiên cứu khoa học và xã hội. Một trong số hệ quản trị cơ sở dữ liệu nổi lên đó có thể kể đến như Oracle, SQL Server, ...

Môn học *"Cơ sở dữ liệu nâng cao"* đã cung cấp các kiến thức liên quan đến việc phân tích, xử lý, xây dựng một hệ cơ sở dữ liệu, các vấn đề xử lý tương tranh các giao tác, nghiên cứu cơ sở dữ liệu phân tán và cơ sở dữ liệu song song, dữ liệu lớn – Big Data, kho dữ liệu – Data warehouse, ... Nghiên cứu về hệ quản trị cơ sở dữ liệu Oracle,...

Em xin gửi lời cảm ơn chân thành nhất tới các thầy cô phụ trách học phần: *TS.Nguyễn Thị Thanh Huyền*, *ThS Nguyễn Tuấn Dũng* và *ThS Nguyễn Danh Tú* đã hướng dẫn, chỉ bảo và tạo điều kiện giúp em có thể hoàn thành học phần và báo cáo môn học. Do thời gian, kiến thức và kinh nghiệm còn nhiều hạn chế nên báo cáo còn nhiều thiếu sót, vì vậy em rất mong nhận được những ý kiến góp ý của thầy cô để báo cáo có thể hoàn thiện hơn.

Em xin chân thành cảm ơn!

Mục lục

Chương 1

Cơ sở dữ liệu lớn

1.1 Thế nào là cơ sở dữ liệu lớn?

Theo [3], Big Data là một trong 4 xu hướng công nghệ hiện nay, bao gồm: Dữ liệu lớn (*Big Data*), Đám mây (*Cloud*), Mạng xã hội (*Social Networks*) và Di động (*Mobulity*).

Theo Wikipedia, Dữ liệu lớn (Tiếng Anh: Big data) là một thuật ngữ cho việc xử lý một tập hợp dữ liệu rất lớn và phức tạp mà các ứng dụng xử lý dữ liệu truyền thống không xử lý được. Dữ liệu lớn bao gồm các thách thức như phân tích, thu thập, giám sát dữ liệu, tìm kiếm, chia sẻ, lưu trữ, truyền nhận, trực quan, truy vấn và tính riêng tư. Thuật ngữ này thường chỉ đơn giản đề cập đến việc sử dụng các phân tích dự báo, phân tích hành vi người dùng, hoặc một số phương pháp phân tích dữ liệu tiên tiến khác trích xuất giá trị từ dữ liệu mà ít khi đề cập đến kích thước của bộ dữ liệu.

Dữ liệu lớn thường bao gồm tập hợp dữ liệu với kích thước vượt xa khả năng của các công cụ phần mềm thông thường để thu thập, hiển thị, quản lý và xử lý dữ liệu trong một thời gian có thể chấp nhận được. Kích thước dữ liệu lớn là một mục tiêu liên tục thay đổi. Như năm 2012 thì phạm vi một vài tá terabytes tới nhiều petabytes dữ liệu. Dữ liệu lớn yêu cầu một tập các kỹ thuật và công nghệ được tích hợp theo hình thức mới để khai phá từ tập dữ liệu đa dạng, phức tạp, và có quy mô lớn.

1.2 Đặc trưng cơ bản của cơ sở dữ liệu lớn

Big Data được mô tả bởi những đặc trưng sau:

Hình 1.1: Cơ sở dữ liệu lớn - BigData

1.2.1 Dung lượng (Volume)

Đây là đặc điểm tiêu biểu nhất của dữ liệu lớn, khối lượng dữ liệu rất lớn. Kích cỡ của Big data đang từng ngày tăng lên, và tính đến năm 2012 thì nó có thể nằm trong khoảng vài chục terabyte cho đến nhiều petabyte (1 petabyte = 1024 terabyte) chỉ cho một tập hợp dữ liệu. Dữ liệu truyền thống có thể lưu trữ trên các thiết bị đĩa mềm, đĩa cứng. Nhưng với dữ liệu lớn chúng ta sẽ sử dụng công nghệ đám mây mới đáp ứng khả năng lưu trữ được dữ liệu lớn.

1.2.2 Tốc độ (Velocity)

Tốc độ có thể hiểu theo 2 khía cạnh:

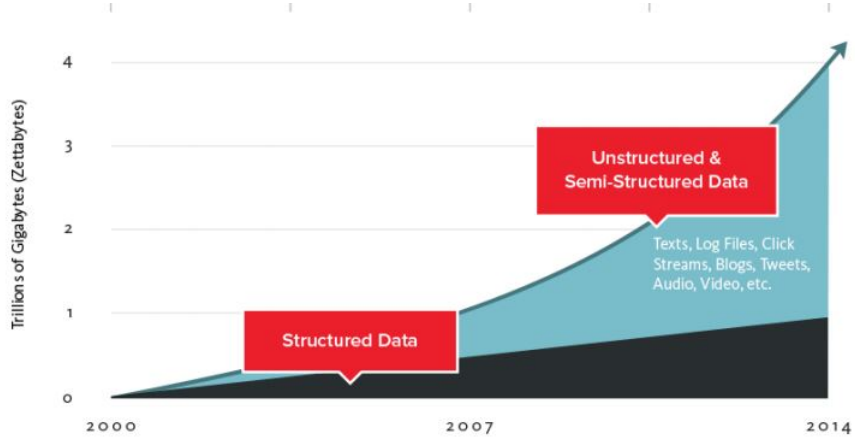
(a) Khối lượng dữ liệu gia tăng rất nhanh (mỗi giây có tới 72.9 triệu các yêu cầu truy cập tìm kiếm trên web bán hàng của Amazon);

(b) Xử lý dữ liệu nhanh ở mức thời gian thực (real-time), có nghĩa dữ liệu được xử lý ngay tức thời ngay sau khi chúng phát sinh (tính đến bằng mili giây). Các ứng dụng phổ biến trên lĩnh vực Internet, Tài chính, Ngân hàng, Hàng không, Quân sự, Y tế- Sức khỏe như hiện nay phần lớn dữ liệu lớn được xử lý real-time. Công nghệ xửxử lý dữ liệu lớn ngày nay đã cho phép chúng ta xử lý tức thì trước khi được lưu trữ vào cơ sở dữ liệu.

1.2.3 Đa dạng (Variety)

Đối với dữ liệu truyền thống chúng ta hay nói đến dữ liệu có cấu trúc, thì ngày nay hơn 80% dữ liệu được sinh ra là phi cấu trúc (tài liệu, blog, hình ảnh, video,

bài hát, dữ liệu từ thiết bị cảm biến vật lý, thiết bị chăm sóc sức khỏe $\cdots$). Big data cho phép liên kết và phân tích nhiều dạng dữ liệu khác nhau. Ví dụ, với các bình luận của một nhóm người dùng nào đó trên Facebook với thông tin video được chia sẻ.



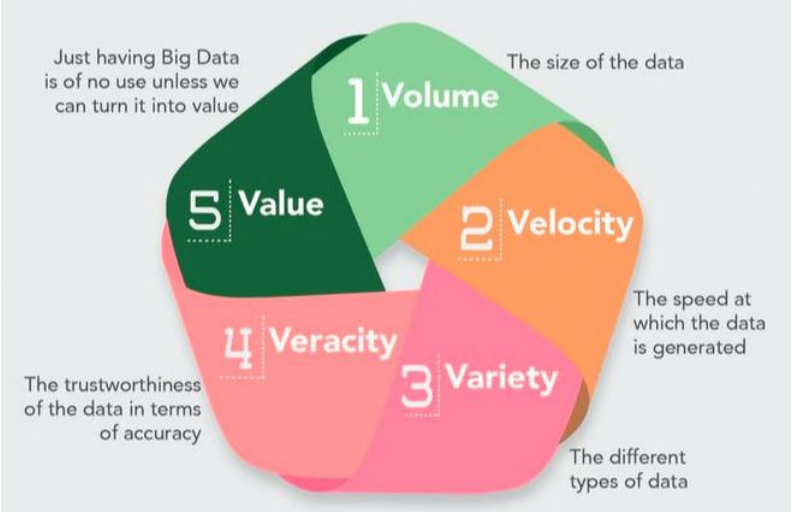
Hình 1.2: Hơn 80% dữ liệu là phi cấu trúc

1.2.4 Độ tin cậy (Veracity)

Một trong những tính chất phức tạp nhất của Dữ liệu lớn là độ tin cậy/chính xác của dữ liệu. Với xu hướng phương tiện truyền thông xã hội (Social Media) và mạng xã hội (Social Network) ngày nay và sự gia tăng mạnh mẽ tính tương tác và chia sẻ của người dùng Mobile làm cho bức tranh xác định về độ tin cậy và chính xác của dữ liệu ngày một khó khăn hơn. Bài toán phân tích và loại bỏ dữ liệu thiếu chính xác và nhiều đang là tính chất quan trọng của Big data.

1.2.5 Giá trị (Value)

Giá trị là đặc điểm quan trọng nhất của dữ liệu lớn, vì khi bắt đầu triển khai xây dựng dữ liệu lớn thì việc đầu tiên chúng ta cần phải làm đó là xác định được giá trị của thông tin mang lại như thế nào, khi đó chúng ta mới có quyết định có nên triển khai dữ liệu lớn hay không. Nếu chúng ta có dữ liệu lớn mà chỉ nhận được 1% lợi ích từ nó, thì không nên đầu tư phát triển dữ liệu lớn. Kết quả dự báo chính xác thể hiện rõ nét nhất về giá trị của dữ liệu lớn mang lại. Ví dụ, từ khối dữ liệu phát sinh trong quá trình khám, chữa bệnh sẽ giúp dự báo về sức khỏe được chính xác hơn, sẽ giảm được chi phí điều trị và các chi phí liên quan đến y tế.



Hình 1.3: Đặc trưng của Big data

1.3 Các ứng dụng của cơ sở dữ liệu lớn

1.3.1 Quản lý chính phủ

Việc sử dụng các dữ liệu lớn trong các quy trình của chính phủ cho phép tăng hiệu quả về mặt chi phí, năng suất và sự đổi mới, nhưng không phải là không có sai sót của nó. Phân tích dữ liệu thường yêu cầu nhiều bộ phận của chính phủ (trung ương và địa phương) hợp tác và tạo ra các quy trình mới và sáng tạo để mang lại kết quả mong muốn.

1.3.2 Sự phát triển quốc tế

Nghiên cứu về việc sử dụng hiệu quả các công nghệ thông tin và truyền thông cho mục đích phát triển (hay còn gọi là ICT4D) cho thấy công nghệ dữ liệu lớn có thể có nhiều đóng góp quan trọng nhưng cũng là thách thức đối với sự phát triển của quốc tế. Những tiến bộ trong phân tích dữ liệu lớn giúp giảm chi phí cho việc ra quyết định trong các lĩnh vực quan trọng như chăm sóc sức khoẻ, việc làm, năng suất kinh tế, tội phạm, an ninh, thiên tai và quản lý tài nguyên. Tuy nhiên, những thách thức đối với các nước đang phát triển như cơ sở hạ tầng công nghệ không đầy đủ và sự khan hiếm về kinh tế và nguồn nhân lực sẽ làm nghiêm trọng thêm các mặt trái của dữ liệu lớn như sự riêng tư hoặc các vấn đề khác.

1.3.3 Tài chính

Việc sử dụng các dữ liệu lớn dưới dạng lịch sử các giao dịch tài chính được gọi là phân tích kỹ thuật. Sử dụng dữ liệu phi tài chính để dự đoán thị trường đôi

khi được gọi là dữ liệu thay thế.

1.3.4 Sản xuất

Theo bài Nghiên cứu xu hướng toàn cầu TCS 2013, sự cải tiến trong kế hoạch sản xuất và chất lượng sản phẩm là lợi ích lớn nhất của dữ liệu lớn cho ngành sản xuất. Dữ liệu lớn cung cấp cơ sở hạ tầng cho ngành công nghiệp sản xuất, đó là khả năng cải thiện năng suất và tính khả dụng. Việc lên kế hoạch sản xuất chính là một cách tiếp cận dữ liệu lớn cho phép giảm thời gian chết về gần như bằng không và cụ thể hóa số lượng lớn dữ liệu và các công cụ dự đoán khác cho phép tạo ra một quá trình nhằm hệ thống hóa dữ liệu thành các thông tin hữu ích. Khái niệm về việc dự báo sản xuất bắt đầu bằng việc thu thập dữ liệu cảm quan khác nhau như âm thanh, chuyển động, áp suất, điện áp... Số lượng lớn các dữ liệu cảm quan cộng với dữ liệu lịch sử sản xuất tạo thành dữ liệu lớn trong sản xuất. Các dữ liệu lớn này như là đầu vào cho các công cụ dự báo và các chiến lược phòng ngừa tương tự như việc dự báo trong lĩnh vực Quản lý Y tế.

1.3.5 Chăm sóc sức khỏe

Phân tích dữ liệu lớn đã giúp cải thiện việc chăm sóc sức khoẻ bằng cách cá nhân hóa các phương pháp trị liệu và chẩn đoán lâm sàng, làm giảm thiểu chi phí và thời gian khám bệnh, tự động báo cáo và lưu trữ thông tin sức khỏe và dữ liệu bệnh nhân trong nội bộ cũng như mở rộng ra bên ngoài, chuẩn hóa các thuật ngữ y học và chống phân mảnh trong lưu trữ dữ liệu và thông tin của bệnh. Một số lĩnh vực có sự cải tiến mang tính hướng dẫn hơn là thực hành. Lượng dữ liệu được tạo ra trong các hệ thống chăm sóc sức khoẻ là không nhỏ. Với sự bổ sung thêm của mHealth, eHealth và các thiết bị công nghệ theo dõi sức khỏe được thì khối lượng dữ liệu sẽ tiếp tục gia tăng. Điều này bao gồm dữ liệu ghi chép sức khoẻ điện tử, dữ liệu hình ảnh, dữ liệu được tạo ra của bệnh nhân, dữ liệu cảm biến và các dạng dữ liệu khó xử lý khác. Hiện nay, nhu cầu lớn hơn đối với các môi trường như vậy là chú ý nhiều hơn đến chất lượng dữ liệu và thông tin. "Dữ liệu lớn rất thường có nghĩa là dữ liệu chưa được xử lý và một phần số liệu không chính xác tăng lên khi có sự tăng trưởng khối lượng dữ liệu." Việc theo dõi bằng con người ở quy mô dữ liệu lớn là không thể và có một nhu cầu cấp thiết về các công cụ thông minh để kiểm soát chính xác và xử lý thông tin bị mất trong dịch vụ y tế. Mặc dù dữ liệu trong lĩnh vực chăm sóc sức khoẻ hiện nay thường được lưu trữ dưới dạng điện tử, nhưng nó nằm ngoài phạm vi của dữ liệu lớn vì hầu hết không có cấu trúc và khó sử dụng.

1.3.6 Giáo dục

Một nghiên cứu của Viện nghiên cứu toàn cầu McKinsey cho thấy, ngành dữ liệu lớn đang thiếu hụt 1,5 triệu chuyên gia cũng như nhà quản lý dữ liệu, và một số trường đại học bao gồm Đại học Tennessee và UC Berkeley đã tạo ra các chương trình thạc sĩ để đáp ứng nhu cầu này. Các khóa huấn luyện tư nhân cũng phát triển các chương trình để đáp ứng nhu cầu đó, bao gồm các chương trình miễn phí như The Data Incubator hoặc chương trình trả tiền như General Assembly.

1.3.7 Truyền thông

Để hiểu cách thức các phương tiện truyền thông sử dụng dữ liệu lớn như thế nào, trước tiên cần hiểu rõ một số ngữ cảnh trong cơ chế sử dụng cho quá trình truyền thông. Nick Couldry và Joseph Turow đề xuất rằng các học viên trong ngành Truyền thông và Quảng cáo cần tiếp cận dữ liệu lớn như là nhiều điểm thông tin về hàng triệu cá nhân. Ngành công nghiệp dường như đang chuyển hướng từ cách tiếp cận truyền thống bằng cách sử dụng các môi trường truyền thông cụ thể như báo chí, tạp chí hoặc chương trình truyền hình và thay vào đó là những người tiêu dùng với công nghệ tiếp cận những người này được nhắm mục tiêu vào những thời điểm tối ưu ở những vị trí tối ưu. Mục đích cuối cùng là để phục vụ hoặc truyền tải, một thông điệp hoặc nội dung (theo cách thống kê) phù hợp với suy nghĩ của người tiêu dùng. Ví dụ, môi trường xuất bản ngày càng làm cho các thông điệp (quảng cáo) và nội dung (bài viết) được cải thiện để thu hút người tiêu dùng đã được thu thập độc quyền thông qua các hoạt động khai thác dữ liệu khác nhau.

- Nhắm đến người tiêu dùng mục tiêu (đối với quảng cáo của các nhà tiếp thị).
- Thu thập dữ liệu
- Dữ liệu trong báo chí: nhà xuất bản và nhà báo sử dụng các công cụ dữ liệu lớn để cung cấp thông tin chi tiết và các bản đồ họa chi tiết độc đáo và sáng tạo.

1.3.8 Công nghệ

Từ năm 2015, dữ liệu lớn trở nên nổi bật trong hoạt động kinh doanh như một công cụ để giúp nhân viên làm việc hiệu quả hơn cũng như tối ưu hóa việc thu thập và chia sẻ thông tin. Việc sử dụng dữ liệu lớn để giải quyết các vấn đề thu thập dữ liệu và CNTT trong một doanh nghiệp được gọi là IT Operations Analytics (ITOA). Bằng cách áp dụng các nguyên tắc dữ liệu lớn vào các khái niệm về trí thông minh của máy móc và tính toán sâu, các bộ phận CNTT có

thể dự đoán các vấn đề tiềm ẩn và đưa ra các giải pháp trước khi vấn đề xảy ra. Vào thời điểm này, các doanh nghiệp ITOA cũng bắt đầu đóng vai trò quan trọng trong việc quản lý hệ thống bằng cách cung cấp các nền tảng mang các dữ liệu cá nhân riêng biệt và tạo ra những hiểu biết sâu sắc từ toàn bộ hệ thống chứ không phải từ các dữ liệu riêng lẻ.

1.3.9 Mạng lưới vạn vật kết nối Internet (IoT)

Dữ liệu lớn có thể kết hợp với công nghệ Mạng lưới vạn vật kết nối Internet. Dữ liệu được chiết xuất từ các thiết bị IoT cung cấp một bản đồ kết nối giữa các thiết bị. Những sự kết nối này đã được ngành công nghiệp truyền thông, các công ty và chính phủ sử dụng để nhắm mục tiêu chính xác hơn đối tượng của họ và tăng hiệu quả của phương tiện truyền thông. IoT cũng ngày càng được chấp nhận như một phương tiện thu thập dữ liệu cảm giác, và dữ liệu cảm giác này đã được sử dụng trong các ngành như y học và sản xuất.
Kevin Ashton, chuyên gia đổi mới kỹ thuật số người được cho là người tạo ra thuật ngữ định nghĩa Internet vạn vật đã phát biểu: "Nếu chúng ta có máy tính biết tất cả mọi thứ - nó sẽ sử dụng dữ liệu mà nó thu thập được mà không có sự trợ giúp từ chúng ta - chúng ta sẽ có thể theo dõi và kiểm soát mọi thứ, giảm đáng kể lượng chất thải, tổn thất và chi phí. Chúng ta sẽ biết khi nào cần thay thế, sửa chữa hoặc thu hồi lại, và liệu rằng thức ăn chúng ta đang ăn có tươi hay không."

Hình 1.4: Ứng dụng của Big data

1.4 NoSQL

Tổng quan về cơ sở dữ liệu NoSQL

Khi làm việc với database, chúng ta đã quá quen với SQL Server, MySQL, PostgreSQL, Oracle,... Điểm chung của những database này là sử dụng ngôn ngữ SQL để truy vấn dữ liệu. Nhưng có 1 dạng database khác với những đặc tính

khác biệt được gọi chung dưới cái tên là NoSQL. Thuật ngữ NoSQL được giới thiệu lần đầu vào năm 1998 sử dụng làm tên gọi chung cho các *"lightweight open source relational database"* (cơ sở dữ liệu quan hệ mã nguồn mở nhỏ) nhưng không sử dụng SQL cho truy vấn. Thuật ngữ NoSQL đánh dấu bước phát triển của thế hệ database mới: *distributed* (phân tán) + *non-relational* (không ràng buộc), đây là 2 đặc tính quan trọng nhất.

Người ta phát triển NoSQL xuất phát từ yêu cầu cần những database có khả năng lưu trữ dữ liệu với lượng cực lớn, truy vấn dữ liệu với tốc độ cao mà không đòi hỏi quá nhiều về năng lực phần cứng cũng như tài nguyên hệ thống và tăng khả năng chịu lỗi. Đây là những vấn đề mà các relational database không thể giải quyết được. Lượng dữ liệu mà các hệ thống cần phải xử lý giờ đây ngày một lớn. Ví dụ như Google, Facebook phải lưu trữ và xử lý một lượng dữ liệu cực lớn mỗi ngày.

Một số đặc điểm của NoSQL

- **High Scalability:** Gần như không có một giới hạn cho dữ liệu và người dùng trên hệ thống.
- **High Availability:**Do chấp nhận sự trùng lặp trong lưu trữ nên nếu một node (commodity machine) nào đó bị chết cũng không ảnh hưởng tới toàn bộ hệ thống.
- **Atomicity:** Độc lập data state trong các operation.
- **Consistency:** chấp nhận tính nhất quán yếu, có thể không thấy ngay được sự thay đổi mặc dù đã cập nhật dữ liệu.
- **Durability:** dữ liệu có thể tồn tại trong bộ nhớ máy tính nhưng đồng thời cũng được lưu trữ lại đĩa cứng.
- **Deployment Flexibility:** việc bổ sung thêm/loại bỏ các node, hệ thống sẽ tự động nhận biết để lưu trữ mà không cần phải can thiệp bằng tay. Hệ thống cũng không đòi hỏi cấu hình phần cứng mạnh, đồng nhất.
- **Modeling flexibility:** Key-Value pairs, Hierarchical data (dữ liệu cấu trúc), Graphs.
- **Query Flexibility:** Multi-Gets, Range queries (load một tập giá trị dựa vào một dãy các khóa).

Chương 2

Cơ sở dữ liệu phân tán

2.1 Các khái niệm cơ bản trong cơ sở dữ liệu phân tán

Cơ sở dữ liệu phân tán

Một tuyển tập dữ liệu có quan hệ logic với nhau, được phân bố trên máy tính của một mạng máy tính.

Hệ quản trị CSDL phân tán

Hệ thống phần mềm cho phép quản lý CSDL phân tán và đảm bảo tính trong suốt về sự phân tán đối với người dùng.

Ứng dụng cục bộ

Là ứng dụng được yêu cầu và thực hiện trên máy tính ở một nút trong hệ CSDL phân tán và chỉ liên quan đến CSDL tại nút đó.

Ứng dụng toàn cục

Là ứng dụng được yêu câu truy nhập dữ liệu ở nhiều nút thông qua hệ thống truyền thông.

2.2 Đánh giá cơ sở dữ liệu phân tán

2.2.1 Ưu điểm và nhược điểm của cơ sở dữ liệu phân tán

Ưu điểm

- Phù hợp với cấu trúc của tổ chức

- Nâng cao khả năng chia sẻ và tính tự trị địa phương
- Nâng cao tính sẵn sàng, tính tin cậy
- Nâng cao hiệu năng
- Dễ mở rộng, phát triển

Nhược điểm

- Thiết kế cơ sở dữ liệu phức tạp hơn
- Khó điều chuyển tính nhất quán dữ liệu
- Khó phát hiện và xử lý lỗi
- Vấn đều bảo mật
- Giá thành cao
- Thiếu chuẩn mực
- Thiếu kinh nghiệm

2.2.2 Một số vấn đề đối với cơ sở dữ liệu phân tán

Vấn đề về thiết kế

- Cách phân bố cơ sở dữ liệu?
- Vấn đề nhân bản dữ liệu
- Vấn đề liên quan trong thư mục quản lý

Vấn đề xử lý truy vấn

- Chuyển các giao tác người dùng thành các chỉ thị thao tác trên dữ liệu
- Vấn đề tối ưu hóa
- Tối thiểu chi phí truyền dữ liệu và xử lý cục bộ
- Không có công thức chung

Vấn đề điều khiển tương tranh

- Đồng bộ hóa các truy xuất tương tranh
- Tính nhất quán và tính riêng biệt của các giao tác
- Quản lý deadlock

Vấn đề độ tin cậy

- Khả năng đáp ứng của hệ thống đối với các lỗi

2.3 Xây dựng kiến trúc cơ sở dữ liệu phân tán

Do sự đa dạng, không có kiến trúc nào được công nhận tương đương với kiến trúc 3 mức ANSI/SPARC. Một kiến trúc tham khảo bao gồm:

- Tập các sơ đồ ngoài toàn cục (Global external schema)
- Sơ đồ khái niệm toàn cục (Global conceptual schema)
- Sơ đồ phân đoạn (Fragmentation schema) và sơ đồ định vị (Allocation schema)
- Tập các sơ đồ cho mỗi hệ CSDL cục bộ tuân theo tiêu chuẩn 3 mức ANSI/SPARC

Sơ đồ tổng thể

Sơ đồ này xác định tất cả các dữ liệu sẽ được lưu trữ trong CSDL phân tán. Sơ đồ tổng thể có thể được định nghĩa một cách chính xác theo cách như trong CSDL không phân tán. Ở đây sẽ sử dụng mô hình quan hệ để hình thành nên sơ đồ này. Sử dụng mô hình này, sơ đồ tổng thể bao gồm định nghĩa của một tập các quan hệ tổng thể.

Sơ đồ phân đoạn

Mỗi quan hệ tổng thể có thể chia thành một vài phần nhỏ hơn không giao nhau được gọi là đoạn (fragments). Có nhiều cách khác nhau để thực hiện việc phân chia này. Sơ đồ tổng thể mô tả các ánh xạ giữa các quan hệ tổng thể và các đoạn được định nghĩa trong sơ đồ phân đoạn. Ánh xạ này là một- nhiều. Có thể có nhiều đoạn liên kết tới một quan hệ tổng thể, nhưng mỗi đoạn chỉ liên kết tới nhiều nhất là một quan hệ tổng thể. Các đoạn được chỉ ra bằng tên của quan hệ tổng thể cùng với tên của chỉ mục đoạn.

Sơ đồ định vị

Các đoạn là các phần logic của một quan hệ tổng thể được định vị trên một hoặc nhiều vị trí vật lý trên mạng. Sơ đồ định vị xác định đoạn nào ở trạm nào. Lưu ý rằng, kiểu ánh xạ được định nghĩa trong sơ đồ định vị quyết định CSDL phân tán là dư thừa hay không. Tất cả các đoạn liên kết với cùng một quan hệ tổng thể R và được định vị tại cùng một trạm j cấu thành ảnh vật lý của quan

hệ tổng thể R tại trạm j. Bởi vậy, có thể ánh xạ một-một giữa một ảnh vật lý và một cặp (quan hệ tổng thể, trạm). Các ảnh vật lý có thể được chỉ ra bằng tên của một quan hệ tổng thể và một chỉ mục trạm.

Sơ đồ ánh xạ địa phương

Ánh xạ các ảnh vật lý tới các đối tượng được các hệ quản trị CSDL địa phương thao tác tại các trạm. Ánh xạ này phụ thuộc vào các hệ quản trị CSDL địa phương.

2.4 Thiết kế cơ sở dữ liệu phân tán

2.4.1 Kỹ thuật phân đoạn

Phân hoạch cơ sở dữ liệu thành các đoạn (fragments): sự phân đoạn cho phép phân chia một đối tượng đơn lẻ thành hai hay nhiều mảnh. Thông tin phân đoạn được lưu trữ trong catalog dữ liệu phân tán. Phần mềm xử lý giao tác sẽ truy nhập thông tin ở đây để xử lý các yêu cầu của người dùng. Việc chia quan hệ tổng thể thành các đoạn có thể thực hiện bằng cách áp dụng các kiểu phân đoạn sau:

- Phân đoạn ngang: Dùng phép chọn để phân đoạn
- Phân đoạn dọc: Dùng phép chiếu để phân đoạn
- Phân đoạn hỗn hợp: Dùng cả phép chọn và phép chiếu để phân đoạn
- Phân đoạn ngang suy diễn: Dùng phép nửa nối để phân đoạn

2.4.2 Các ràng buộc trong thiết kế phân đoạn

Tính đầy đủ

Toàn bộ dữ liệu của quan hệ tổng thể phải được ánh xạ vào các đoạn quan hệ và ngược lại. Điều này có nghĩa là, không tồn tại một mục dữ liệu nào thuộc vào quan hệ tổng thể mà không thuộc vào bất kỳ một đoạn nào.

Tính phục hồi

Quan hệ tổng thể có thể được xây dựng lại từ các đoạn mà nó đã tách ra. Điều kiện này là hiển nhiên, bởi vì trong thực tế chỉ có các đoạn được lưu trữ trong CSDL phân tán, và quan hệ tổng thể phải được xây dựng lại thông qua các đoạn khi cần thiết.

Tính tách biệt

các đoạn được tách ra từ quan hệ tổng thể phải là rời nhau. Vì vậy, việc tạo các bản sao phải rõ ràng với các đoạn được chia. Tuy nhiên, điều kiện này chỉ áp dụng chính vào việc phân đoạn ngang, trong khi việc phân đoạn dọc nhiều khi vẫn được phép vi phạm điều kiện này.

2.5 Tính trong suốt của cơ sở dữ liệu phân tán

2.5.1 Trong suốt phân đoạn (Fragmentation transparency)

Trong suốt phân đoạn: là cấp độ cao nhất của mức độ trong suốt, người sử dụng hoặc chương trình ứng dụng chỉ làm việc trên các quan hệ của cơ sở dữ liệu. Khi dữ liệu đã được phân đoạn thì việc truy cập vào CSDL được thực hiện bình thường như là chưa bị phân tán và không ảnh hưởng tới người sử dụng.

2.5.2 Trong suốt về vị trí (Location transparency)

Người sử dụng không cần biết về vị trí vật lý của dữ liệu mà có quyền truy cập đến cơ sở dữ liệu tại bất cứ vị trí nào. Các thao tác để lấy hoặc cập nhật một dữ liệu từ xa được tự động thực hiện bởi hệ thống tại điểm đưa ra yêu cầu.

2.5.3 Trong suốt ánh xạ địa phương (Local mapping transparency)

Người dùng cuối hoặc người lập trình biết tên các đoạn và vị trí của các đoạn.

2.5.4 Trong suốt nhân bản (Replication transparency)

Mức trong suốt bản sao liên quan chặt chẽ tới mức trong suốt định vị. Mức trong suốt bản sao có nghĩa là người sử dụng không biết bản sao của đoạn đặt ở vị trí nào. Mức trong suốt bản sao tương đương mức trong suốt định vị. Tuy nhiên, trong những trường hợp thực tế người sử dụng không có mức trong suốt định vị nhưng lại có mức trong suốt bản sao.

Chương 3

Hệ quản trị cơ sở dữ liệu Oracle

3.1 Lecture 1

Hướng dẫn cài đặt hệ quản trị cơ sở dữ liệu Oracle.

3.2 Lecture 2

3.2.1 Practice 1

Create the DEPT table based on the following table instance chart. Place the syntax in a script called lab_09_01.sql, then execute the statement in the script to create the table. Confirm that the table is created.

Mã nguồn

```
CREATE TABLE DEPT
  (
    id   NUMBER(7) CONSTRAINT dept_department_id PRIMARY KEY,
    name VARCHAR2(25)
  )
```

Kết quả

```
Table DEPT created.
```

3.2.2 Practice 2

Populate the DEPT table with data from the DEPARTMENTS table. Include only columns that you need.

Mã nguồn

```
INSERT
INTO DEPT
  (
    id,
    name
  )
SELECT department_id, department_name FROM DEPARTMENTS;
```

Kết quả

```
27 rows inserted.
```

3.2.3 Practice 3

Create the EMP table based on the following table instance chart. Place the syntax in a script called lab_09_03.sql, and then execute the statement in the script to create the table. Confirm that the table is created.

Mã nguồn

```
CREATE TABLE EMP
  (
    id         NUMBER(7) NOT NULL,
    last_name  VARCHAR2(25),
    first_name VARCHAR2(25),
    dept_id    NUMBER(7),
    CONSTRAINT emp_employee_id PRIMARY KEY (id),
    CONSTRAINT empdept_fkl FOREIGN KEY (dept_id) REFERENCES dept(id)
  )
```

Kết quả

```
Table EMP created.
```

3.2.4 Practice 4

Create the EMPLOYEES2 table based on the structure of the EMPLOYEES table. Include only the EMPLOYEE_ID, FIRST_NAME, LAST_NAME, SALARY, and DEPARTMENT_ID columns. Name the columns in your new table ID, FIRST_NAME, LAST_NAME, SALARY , and DEPT_ID, respectively.

Mã nguồn

```
CREATE TABLE EMPLOYEES2 AS
SELECT EMPLOYEE_ID, first_name, last_name, salary,
  department_id AS dept_id
FROM EMPLOYEES;
```

Kết quả

```
Table EMPLOYEES2 created.
```

3.2.5 Practice 5

Drop the EMP table.

Mã nguồn

```
DROP TABLE emp;
```

Kết quả

```
Table EMP dropped.
```

3.2.6 Practice 6

Create a nonunique index on the DEPT_ID column in the DEPT table.

Mã nguồn

```
CREATE INDEX emp_dept_id_idx ON emp
  (
    dept_id
  );
```

3.3 Lecture 3

3.3.1 Practice 1

The staff in the HR department wants to hide some of the data in the EMPLOYEES table.They want a view called EMPLOYEES_VU based on the employee numbers, employee names, and department numbers from the EMPLOYEES table. They want the heading for the employee name to be EMPLOYEE.

Mã nguồn

```
CREATE OR REPLACE VIEW EMPLOYEES_VU
AS
  SELECT EMPLOYEE_ID EMPLOYEE_NUMBER, LAST_NAME EMPLOYEE, DEPARTMENT_ID
  FROM EMPLOYEES;
```

Kết quả

```
View EMPLOYEES_VU created.
```

3.3.2 Practice 2

Confirm that the view works. Display the contents of the EMPLOYEES_VU view.

Mã nguồn

```
SELECT * FROM EMPLOYEES_VU;
```

Kết quả

	EMPLOYEE_NUMBER	EMPLOYEE	DEPARTMENT_ID
1	100	King	90
2	101	Kochhar	90
3	102	De Haan	90
4	103	Hunold	60
5	104	Ernst	60
6	105	Austin	60
7	106	Pataballa	60
8	107	Lorentz	60
9	108	Greenberg	100
10	109	Faviet	100

3.3.3 Practice 3

Using your EMPLOYEES_VU view, write a query for the HR department to display all employee names and department numbers.

Mã nguồn

```
SELECT EMPLOYEE, DEPARTMENT_ID FROM EMPLOYEES_VU;
```

Kết quả

	EMPLO...	DEPARTMENT_ID
1	King	90
2	Kochhar	90
3	De Haan	90
4	Hunold	60
5	Ernst	60
6	Austin	60
7	Pataballa	60
8	Lorentz	60
9	Greenberg	100
10	Faviet	100

3.3.4 Practice 4

Department 50 needs access to its employee data. Create a view named DEPT50 that contains the employee numbers, employee last names, and department numbers for all employees in department 50.
You have been asked to label the view columns EMPNO, EMPLOYEE, and DEPTNO. For security purposes, do not allow an employee to be reassigned to another department through the view.
* Display the structure and contents of the DEPT50 view.
* Test your view. Attempt to reassign Mohammed to department 80.

Mã nguồn

```
CREATE OR REPLACE VIEW DEPT50 AS
SELECT EMPLOYEE_ID EMPNO, LAST_NAME EMPLOYEE, DEPARTMENT_ID DEPTNO FROM EMPLOYEES WHERE DEPARTMENT_ID = 50;

SELECT * FROM DEPT50;

CREATE OR REPLACE VIEW DEPT80 AS
SELECT EMPLOYEE_ID EMPNO, LAST_NAME EMPLOYEE, DEPARTMENT_ID DEPTNO FROM EMPLOYEES WHERE DEPARTMENT_ID = 80;
```

Kết quả

	EMPNO	EMPLOYEE	DEPTNO
1	120	Weiss	50
2	121	Fripp	50
3	122	Kaufling	50
4	123	Vollman	50
5	124	Mourgos	50
6	125	Nayer	50
7	126	Mikkilineni	50
8	127	Landry	50
9	128	Markle	50
10	129	Bissot	50

```
View DEPT80 created.
```

3.3.5 Practice 5

* You need a sequence that can be used with the primary key column of the DEPT table. The sequence should start at 200 and have a maximum value of 1,000. Have your sequence increment by 10. Name the sequence DEPT_ID_SEQ. * To test your sequence, write a script to insert two rows in the DEPT table. Be sure to use the sequence that you created for the ID column. Add two departments: Education and Administration. Confirm your additions. Run the commands in your script.

Mã nguồn

```
CREATE SEQUENCE DEPT_ID_SEQ
  INCREMENT BY 10
  START WITH 280
  MAXVALUE 1000
  CYCLE;

INSERT INTO DEPT (ID, NAME) VALUES (DEPT_ID_SEQ.NEXTVAL, 'Education');
INSERT INTO DEPT (ID, NAME) VALUES (DEPT_ID_SEQ.NEXTVAL, 'Administration');
```

Kết quả

	ID	NAME
1	280	Education
2	290	Administration
3	10	Administration
4	20	Marketing
5	30	Purchasing
6	40	Human Resources
7	50	Shipping
8	60	IT
9	70	Public Relations

3.3.6 Practice 6

Create a synonym for your EMPLOYEES table. Call it EMP.

Mã nguồn

```
CREATE SYNONYM EMP FOR EMPLOYEES;
```

3.4 Lecture 4

3.4.1 Practice 1,2

True or False?

Kết quả **1. True** **2. False**

3.4.2 Practice 3

There are four coding errors in the following statement. Can you identify them?

Mã nguồn

```
SELECT employee_id, last_name, (SALARY*12) as SALARY
FROM employees;
```

Kết quả

	EMPLOYEE_ID	LAST_NAME	SALARY
1	100	King	288000
2	101	Kochhar	204000
3	102	De Haan	204000
4	103	Hunold	108000
5	104	Ernst	72000
6	105	Austin	57600
7	106	Pataballa	57600
8	107	Lorentz	50400
9	108	Greenberg	144096
10	109	Faviet	108000

3.4.3 Practice 4

The HR department needs a query to display all unique job codes from the EMPLOYEES table.

Mã nguồn

```
SELECT DISTINCT (JOB_ID) FROM EMPLOYEES;
```

Kết quả

	JOB_ID
1	AC_ACCOUNT
2	AC_MGR
3	AD_ASST
4	AD_PRES
5	AD_VP

3.4.4 Practice 5

The HR department has requested a report of all employees and their job IDs. Display the last name concatenated with the job ID (separated by a comma and space) and name the column Employee and Title.

Mã nguồn

```
SELECT (LAST_NAME || ',' || JOB_ID) AS "EMPLOYEE AND TITLE" FROM EMPLOYEES;
```

Kết quả

	EMPLOYEE AND TITLE
1	Abel,SA_REP
2	Ande,SA_REP
3	Atkinson,ST_CLERK
4	Austin,IT_PROG
5	Baer,PR_REP
6	Baida,PU_CLERK
7	Banda,SA_REP
8	Bates,SA_REP
9	Bell,SH_CLERK
10	Bernstein,SA_REP

3.4.5 Practice 6

The HR departments needs to find high-salary and low-salary employees. Display the last name and salary of employees who earn between 5,000$ and 12,000$ and are in department 20 or 50. Label the columns Employee and Monthly Salary, respectively.

Mã nguồn

```
SELECT LAST_NAME EMPLOYEE, SALARY AS "MONTHLY SALARY" FROM EMPLOYEES WHERE SALARY BETWEEN 5000 AND 12000
AND DEPARTMENT_ID IN (20, 50);
```

Kết quả

	EMPLOYEE	MONTHLY SALARY
1	Weiss	8000
2	Fripp	8200
3	Kaufling	7900
4	Vollman	6500
5	Mourgos	5800
6	Fay	6000

3.4.6 Practice 7

Create a report to display the last name, salary, and commission of all employees who earn commissions. Sort data in descending order of salary and commissions.

Mã nguồn

```
SELECT LAST_NAME, SALARY, COMMISSION_PCT FROM EMPLOYEES
WHERE COMMISSION_PCT IS NOT NULL
ORDER BY SALARY, COMMISSION_PCT DESC;
```

Kết quả

	LAST_NAME	SALARY	COMMISSION_PCT
1	Kumar	6100	0.1
2	Banda	6200	0.1
3	Johnson	6200	0.1
4	Ande	6400	0.1
5	Lee	6800	0.1
6	Sewall	7000	0.25
7	Grant	7000	0.15
8	Tuvault	7000	0.15
9	Marvins	7200	0.1
10	Bates	7300	0.15

3.4.7 Practice 8

Display the last name of all employees who have both an a and an e in their last name.

Mã nguồn

```
SELECT LAST_NAME FROM EMPLOYEES WHERE LAST_NAME LIKE '%a%e%' OR LAST_NAME LIKE '%e%a%';
```

Kết quả

	LAST_NAME
1	Baer
2	Bates
3	Colmenares
4	Davies
5	De Haan

3.4.8 Practice 9

Display the last name, job, and salary for all employees whose job is SA_REP or ST_CLERK and whose salary is not equal to $2,500, $3,500, or $7,000.

Mã nguồn

```
SELECT LAST_NAME, JOB_ID, SALARY FROM EMPLOYEES
WHERE JOB_ID IN ('SA_REP','ST_CLERK') AND SALARY NOT IN (2500,3500,7000);
```

Kết quả

	LAST_NAME	JOB_ID	SALARY
1	Nayer	ST_CLERK	3200
2	Mikkilineni	ST_CLERK	2700
3	Landry	ST_CLERK	2400
4	Markle	ST_CLERK	2200
5	Bissot	ST_CLERK	3300
6	Atkinson	ST_CLERK	2800
7	Olson	ST_CLERK	2100
8	Mallin	ST_CLERK	3300
9	Rogers	ST_CLERK	2900
10	Gee	ST_CLERK	2400

3.5 Lecture 5

3.5.1 Practice 1

Write a query that displays the last name (with the first letter uppercase and all other letters lowercase) and the length of the last name for all employees whose name starts with the letters J, A, or M. Give each column an appropriate label. Sort the results by the employees' last names.

Mã nguồn

```
SELECT INITCAP(LAST_NAME), LENGTH(LAST_NAME) FROM EMPLOYEES
WHERE SUBSTR(LAST_NAME,1,1) IN ('J', 'A', 'M')
ORDER BY LAST_NAME ASC;
```

Kết quả

	INITCAP(LAST_NAME)	LENGTH(LAST_NAME)
1	Abel	4
2	Ande	4
3	Atkinson	8
4	Austin	6
5	Johnson	7
6	Jones	5
7	Mallin	6
8	Markle	6
9	Marlow	6
10	Marvins	7

3.5.2 Practice 2

The HR department wants to find the length of employment for each employee. For each employee, display the last name and calculate the number of months between today and the date on which the employee was hired. Label the column MONTHS_WORKED. Order your results by the number of months employed. Round the number of months up to the closest whole number.

Mã nguồn

```
SELECT LAST_NAME, ROUND(MONTHS_BETWEEN(SYSDATE, HIRE_DATE)) AS MONTHS_WORKED
FROM EMPLOYEES ORDER BY MONTHS_WORKED ASC;
```

Kết quả

	LAST_NAME	MONTHS_WORKED
1	Kumar	146
2	Banda	146
3	Ande	147
4	Markle	148
5	Lee	148
6	Geoni	149
7	Philtanker	149
8	Zlotkey	149
9	Marvins	149
10	Grant	149

3.5.3 Practice 3

Display each employee's last name, hire date, and salary review date, which is the first Monday after six months of service. Label the column REVIEW. Format the dates to appear in the format similar to "Monday, the Thirty-First of July, 2000."

Mã nguồn

```
SELECT LAST_NAME, HIRE_DATE, TO_CHAR(NEXT_DAY(TRUNC(ADD_MONTHS(HIRE_DATE, 6), 'MONTH'),
'MONDAY'), 'DAY," the" ddspth "of" MONTH YYYY') AS REVIEW FROM EMPLOYEES;
```

Kết quả

	LAST_NAME	HIRE_DATE	REVIEW
1	King	17-JUN-03	MONDAY , the eighth of DECEMBER 2003
2	Kochhar	21-SEP-05	MONDAY , the sixth of MARCH 2006
3	De Haan	13-JAN-01	MONDAY , the second of JULY 2001
4	Hunold	03-JAN-06	MONDAY , the third of JULY 2006
5	Ernst	21-MAY-07	MONDAY , the fifth of NOVEMBER 2007
6	Austin	25-JUN-05	MONDAY , the fifth of DECEMBER 2005
7	Pataballa	05-FEB-06	MONDAY , the seventh of AUGUST 2006
8	Lorentz	07-FEB-07	MONDAY , the sixth of AUGUST 2007
9	Greenberg	17-AUG-02	MONDAY , the third of FEBRUARY 2003
10	Faviet	16-AUG-02	MONDAY , the third of FEBRUARY 2003

3.5.4 Practice 4

Create a query that displays the employees' last names and commission amounts. If an employee does not earn commission, show "No Commission." Label the column COMM.

Mã nguồn

```
SELECT LAST_NAME, NVL2(COMMISSION_PCT, TO_CHAR(COMMISSION_PCT, 9.99),
'No Commission') AS COMM FROM EMPLOYEES;
```

Kết quả

	LAST_NAME	COMM
1	King	No Commission
2	Kochhar	No Commission
3	De Haan	No Commission
4	Hunold	No Commission
5	Ernst	No Commission
6	Austin	No Commission
7	Pataballa	No Commission
8	Lorentz	No Commission
9	Greenberg	No Commission
10	Faviet	No Commission

3.5.5 Practice 5

Using the DECODE function, write a query that displays the grade of all employees based on the value of the column JOB_ID, using the following data:

JOB	GRADE
AD_PRES	A
ST_MAN	B
IT_PROG	C
SA_REP	D
ST_CLERK	E

None of the above 0

Mã nguồn

```
SELECT LAST_NAME,
DECODE(JOB_ID, 'AD_PRES', 'A', 'ST_MAN', 'B', 'IT_PROG', 'C',
                'SA_REP', 'D', 'ST_CLERK', 'E' , 0) AS GRADE
FROM EMPLOYEES;
```

Kết quả

	LAST_NAME	GRADE
1	Abel	D
2	Ande	D
3	Atkinson	E
4	Austin	C
5	Baer	0
6	Baida	0
7	Banda	D

3.5.6 Practice 6

Rewrite the query so that the user is prompted to enter a letter that starts the last name. For example, if the user enters H when prompted for a letter, then the output should show all employees whose last name starts with the letter H.

Mã nguồn

```
SELECT LAST_NAME FROM EMPLOYEES WHERE LOWER(LAST_NAME) LIKE 'h%';
```

Kết quả

	LAST_NAME
1	Hall
2	Hartstein
3	Higgins
4	Himuro
5	Hunold
6	Hutton

3.5.7 Practice 7

Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number..

Mã nguồn

```
SELECT MAX(salary) MAXIMUM, MIN(SALARY) MINIMUM, SUM(salary)SUM ,
ROUND(AVG(SALARY)) AVERAGE FROM EMPLOYEES;
```

Kết quả

	MAXIMUM	MINIMUM	SUM	AVERAGE
1	24000	2100	691416	6462

3.5.8 Practice 8

Modify the query in Exercise 1 to display the minimum, maximum, sum, and average salary for each job type.

Mã nguồn

```
SELECT JOB_ID, MAX(salary) MAXIMUM, MIN(SALARY) MINIMUM, SUM(salary)SUM ,
ROUND(AVG(SALARY)) AVERAGE
FROM EMPLOYEES GROUP BY JOB_ID;
```

Kết quả

	JOB_ID	MAXIMUM	MINIMUM	SUM	AVERAGE
1	IT_PROG	9000	4200	28800	5760
2	AC_MGR	12008	12008	12008	12008
3	AC_ACCOUNT	8300	8300	8300	8300
4	ST_MAN	8200	5800	36400	7280
5	PU_MAN	11000	11000	11000	11000
6	AD_ASST	4400	4400	4400	4400
7	AD_VP	17000	17000	34000	17000
8	SH_CLERK	4200	2500	64300	3215
9	FI_ACCOUNT	9000	6900	39600	7920
10	FI_MGR	12008	12008	12008	12008

3.5.9 Practice 9

Determine the number of managers without listing them. Label the column Number of Managers. Hint: Use the MANAGER_ID column to determine the number of managers.

Mã nguồn

```
SELECT COUNT(DISTINCT MANAGER_ID) AS "NUMBER OF MANAGERS" FROM EMPLOYEES;
```

Kết quả

	NUMBER OF MANAGERS
1	18

3.5.10 Practice 10

Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is $6,000 or less. Sort the output in descending order of salary.

Mã nguồn

```
SELECT MANAGER_ID, MIN(SALARY) FROM EMPLOYEES
GROUP BY MANAGER_ID HAVING MIN(SALARY) > 6000 AND MANAGER_ID IS NOT NULL
ORDER BY MIN(SALARY) DESC;
```

Kết quả

	MANAGER_ID	MIN(SALARY)
1	102	9000
2	205	8300
3	145	7000
4	146	7000
5	108	6900
6	147	6200
7	149	6200
8	148	6100

3.6 Lecture 6

3.6.1 Practice 1

The HR department needs a report of all employees. Write a query to display the last name, department number, and department name for all employees.

Mã nguồn

```
SELECT E.LAST_NAME, E.DEPARTMENT_ID, D.DEPARTMENT_NAME
FROM EMPLOYEES E JOIN DEPARTMENTS D
ON E.DEPARTMENT_ID = D.DEPARTMENT_ID;
```

Kết quả

	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
1	Whalen	10	Administration
2	Fay	20	Marketing
3	Hartstein	20	Marketing
4	Tobias	30	Purchasing
5	Colmenares	30	Purchasing
6	Baida	30	Purchasing
7	Raphaely	30	Purchasing
8	Khoo	30	Purchasing
9	Himuro	30	Purchasing
10	Mavris	40	Human Resources

3.6.2 Practice 2

(A) Create a report to display employees' last name and employee number along with their manager's last name and manager number. Label the columns Employee, Emp#, Manager, and Mgr#, respectively.
(B) Modify Part A to display all employees including King, who has no manager. Order the results by the employee number.

Mã nguồn

```
SELECT E.LAST_NAME, E.EMPLOYEE_ID, M.LAST_NAME, M.EMPLOYEE_ID FROM EMPLOYEES
E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID;
SELECT E.LAST_NAME, E.EMPLOYEE_ID, M.LAST_NAME, M.EMPLOYEE_ID FROM EMPLOYEES
E LEFT OUTER JOIN EMPLOYEES M
ON E.MANAGER_ID = M.EMPLOYEE_ID;
```

Kết quả

	LAST_...	EMPLOYEE_ID	LAST_NAME_1	EMPLOYEE_ID_1
1	Ozer	168	Cambrault	148
2	Bloom	169	Cambrault	148
3	Fox	170	Cambrault	148
4	Smith	171	Cambrault	148
5	Bates	172	Cambrault	148
6	Kumar	173	Cambrault	148
7	Hunold	103	De Haan	102
8	Vishney	162	Errazuriz	147
9	Greene	163	Errazuriz	147
10	Marvins	164	Errazuriz	147

	LAST_NAME	EMPLOYEE_ID	LAST_NAME_1	EMPLOYEE_ID_1
1	Ozer	168	Cambrault	148
2	Bloom	169	Cambrault	148
3	Fox	170	Cambrault	148
4	Smith	171	Cambrault	148
5	Bates	172	Cambrault	148
6	Kumar	173	Cambrault	148
7	Hunold	103	De Haan	102
8	Vishney	162	Errazuriz	147
9	Greene	163	Errazuriz	147
10	Marvins	164	Errazuriz	147

3.6.3 Practice 3

The HR department needs to find the names and hire dates for all employees who were hired before their managers, along with their managers' names and hire dates.

Mã nguồn

```
SELECT E.LAST_NAME, E.HIRE_DATE, M.LAST_NAME, M.HIRE_DATE FROM EMPLOYEES E JOIN
EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID AND E.HIRE_DATE < M.HIRE_DATE;
```

Kết quả

	LAST_...	HIRE_DATE	LAST_NAME_1	HIRE_DATE_1
1	De Haan	13-JAN-01	King	17-JUN-03
2	Raphaely	07-DEC-02	King	17-JUN-03
3	Kaufling	01-MAY-03	King	17-JUN-03
4	Greenberg	17-AUG-02	Kochhar	21-SEP-05
5	Whalen	17-SEP-03	Kochhar	21-SEP-05
6	Mavris	07-JUN-02	Kochhar	21-SEP-05
7	Baer	07-JUN-02	Kochhar	21-SEP-05
8	Higgins	07-JUN-02	Kochhar	21-SEP-05
9	Austin	25-JUN-05	Hunold	03-JAN-06
10	Faviet	16-AUG-02	Greenberg	17-AUG-02

3.6.4 Practice 4

Display the employee number, last name, and salary of all employees who earn more than the average salary and who work in a department with any employee whose last name contains a u.

Mã nguồn

```
SELECT E.EMPLOYEE_ID, E.LAST_NAME, E.SALARY FROM EMPLOYEES E
WHERE E.SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES)
AND E.DEPARTMENT_ID IN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE LAST_NAME LIKE '%u%');
```

Kết quả

	EMPLOYEE_ID	LAST_NAME	SALARY
1	103	Hunold	9000
2	120	Weiss	8000
3	121	Fripp	8200
4	122	Kaufling	7900
5	123	Vollman	6500
6	145	Russell	14000
7	146	Partners	13500
8	147	Errazuriz	12000
9	148	Cambrault	11000
10	149	Zlotkey	10500

3.6.5 Practice 5

The HR department needs a report with the following specifications:
- Last name and department ID of all the employees from the EMPLOYEES table, regardless of whether or not they belong to a department
- Department ID and department name of all the departments from the DEPARTMENTS table, regardless of whether or not they have employees working in them Write a compound query to accomplish this .

Mã nguồn

```
SELECT LAST_NAME, DEPARTMENT_ID FROM EMPLOYEES
UNION ALL
SELECT DEPARTMENT_NAME, DEPARTMENT_ID FROM DEPARTMENTS;
```

Kết quả

	LAST_NAME	DEPARTMENT_ID
1	King	90
2	Kochhar	90
3	De Haan	90
4	Hunold	60
5	Ernst	60
6	Austin	60
7	Pataballa	60
8	Lorentz	60
9	Greenberg	100
10	Faviet	100

3.6.6 Practice 6

Create a report that lists the employee IDs and job IDs of those employees who currently have a job title that is the same as their job title when they were initially hired by the company (that is, they changed jobs but have now gone back to doing their original job).

Mã nguồn

```
SELECT EMPLOYEE_ID, JOB_ID FROM EMPLOYEES
INTERSECT
SELECT JOB_HISTORY.EMPLOYEE_ID, JOB_HISTORY.JOB_ID FROM JOB_HISTORY;
```

Kết quả

	EMPLOYEE_ID	JOB_ID
1	176	SA_REP
2	200	AD_ASST

3.6.7 Practice 7

The HR department needs a list of countries that have no departments located in them. Display the country ID and the name of the countries. Use set operators to create this report.

Mã nguồn

```
SELECT COUNTRY_ID, COUNTRY_NAME FROM COUNTRIES
WHERE COUNTRY_ID NOT IN (SELECT L.COUNTRY_ID FROM LOCATIONS L JOIN DEPARTMENTS D
ON D.LOCATION_ID = L.LOCATION_ID);
```

Kết quả

	COUNTRY_ID	COUNTRY_NAME
1	CN	China
2	NL	Netherlands
3	CH	Switzerland
4	HK	HongKong
5	ZM	Zambia
6	AR	Argentina
7	NG	Nigeria
8	EG	Egypt
9	ML	Malaysia
10	DK	Denmark

3.7 Lecture 7

Không có bài tập.

3.8 Lecture 8

3.8.1 Practice 1

Write a query to display the following for those employees whose manager ID is less than 120:
- Manager ID
- Job ID and total salary for every job ID for employees who report to the same manager
- Total salary of those managers Total salary of those managers, irrespective of the job IDs

Mã nguồn

```
SELECT MANAGER_ID, JOB_ID, SUM(SALARY)
FROM EMPLOYEES
WHERE MANAGER_ID < 120
GROUP BY ROLLUP(MANAGER_ID, JOB_ID);
```

Kết quả

	MANAGER_ID	JOB_ID	SUM(SALARY)
1	100	AD_VP	34000
2	100	MK_MAN	13000
3	100	PU_MAN	11000
4	100	SA_MAN	61000
5	100	ST_MAN	36400
6	100	(null)	155400
7	101	AC_MGR	12008
8	101	FI_MGR	12008
9	101	HR_REP	6500
10	101	PR_REP	10000

3.8.2 Practice 2

Observe the output from question 1. Write a query using the GROUPING function to determine whether the NULL values in the columns corresponding to the GROUP BY expressions are caused by the ROLLUP operation.

Mã nguồn

```
SELECT MANAGER_ID, JOB_ID, SUM(SALARY),
GROUPING(MANAGER_ID) MNG, GROUPING(JOB_ID)JOB
FROM EMPLOYEES
WHERE MANAGER_ID < 120
GROUP BY ROLLUP(MANAGER_ID, JOB_ID);
```

Kết quả

	MANAGER_ID	JOB_ID	SUM(SALARY)	MNG	JOB
1	100	AD_VP	34000	0	0
2	100	MK_MAN	13000	0	0
3	100	PU_MAN	11000	0	0
4	100	SA_MAN	61000	0	0
5	100	ST_MAN	36400	0	0
6	100	(null)	155400	0	1
7	101	AC_MGR	12008	0	0
8	101	FI_MGR	12008	0	0
9	101	HR_REP	6500	0	0
10	101	PR_REP	10000	0	0

3.8.3 Practice 3

Write a query to display the following for those employees whose manager ID is less than 120:
- Manager ID
- Job and total salaries for every job for employees who report to the same manager
- Total salary of those managers
- Cross-tabulation values to display the total salary for every job, irrespective of the manager
- Total salary irrespective of all job titles

Mã nguồn

```
SELECT MANAGER_ID, JOB_ID, SUM(SALARY)
FROM EMPLOYEES WHERE MANAGER_ID < 120
GROUP BY CUBE(MANAGER_ID, JOB_ID);
```

Kết quả

	MANAGER_ID	JOB_ID	SUM(SALARY)
1	(null)	(null)	282616
2	(null)	AD_VP	34000
3	(null)	AC_MGR	12008
4	(null)	FI_MGR	12008
5	(null)	HR_REP	6500
6	(null)	MK_MAN	13000
7	(null)	PR_REP	10000
8	(null)	PU_MAN	11000
9	(null)	SA_MAN	61000
10	(null)	ST_MAN	36400

3.8.4 Practice 4

Using GROUPING SETS, write a query to display the following groupings:
- department_id, manager_id, job_id
- department_id, job_id
- manager_id, job_id
The query should calculate the sum of the salaries for each of these groups.

Mã nguồn

```
SELECT DEPARTMENT_ID, MANAGER_ID, JOB_ID FROM EMPLOYEES
GROUP BY GROUPING SETS((DEPARTMENT_ID, MANAGER_ID, JOB_ID),
(DEPARTMENT_ID,JOB_ID),(MANAGER_ID, JOB_ID));
```

Kết quả

	DEPARTMENT_ID	MANAGER_ID	JOB_ID
1	90	(null)	AD_PRES
2	90	100	AD_VP
3	20	100	MK_MAN
4	30	100	PU_MAN
5	80	100	SA_MAN
6	50	100	ST_MAN
7	110	101	AC_MGR
8	100	101	FI_MGR
9	40	101	HR_REP
10	70	101	PR_REP

3.9 Lecture 9, 10

Không có bài tập.

3.10 Lecture 11

3.10.1 Practice 1

Print last names, salaries, and department IDs.

Mã nguồn

```
SELECT LAST_NAME, SALARY, DEPARTMENT_ID FROM EMPLOYEES;
```

Kết quả

	LAST_NAME	SALARY	DEPARTMENT_ID
1	King	24000	90
2	Kochhar	17000	90
3	De Haan	17000	90
4	Hunold	9000	60
5	Ernst	6000	60
6	Austin	4800	60
7	Pataballa	4800	60
8	Lorentz	4200	60
9	Greenberg	12008	100
10	Faviet	9000	100

3.10.2 Practice 2

Create a report that shows the hierarchy of the managers for the employee Lorentz.
* Don't display Lorentz
* Display his immediate manager first.

Mã nguồn

```
SELECT LAST_NAME FROM EMPLOYEES WHERE LAST_NAME <> 'Lorentz'
START WITH LAST_NAME = 'Lorentz'
CONNECT BY PRIOR EMPLOYEE_ID = MANAGER_ID;
```

Kết quả

	LAST_NAME
1	Hunold
2	De Haan
3	King

3.10.3 Practice 3

Produce a company organization chart that shows the management hierarchy. Print the employee's last name, employee ID, manager ID, and job ID.

Mã nguồn

```
SELECT LPAD(LAST_NAME, LENGTH(LAST_NAME) + (LEVEL*2)-2,'_')
AS ORG_CHART FROM EMPLOYEES START WITH LAST_NAME = 'King'
CONNECT BY PRIOR EMPLOYEE_ID = MANAGER_ID;
```

Kết quả

	ORG_CHART
1	King
2	King
3	__Kochhar
4	____Greenberg
5	______Faviet
6	______Chen
7	______Sciarra
8	______Urman
9	______Popp
10	____Whalen

Chương 4

Bài tập kết thúc môn

4.1 Bài 1

Kiểm tra 1 sinh viên đã đủ điều kiện tốt nghiệp chưa biết rằng các điều kiện để một sinh viên tốt nghiệp là:

- Tích luỹ đủ số tín chỉ
- Điểm phẩy tốt nghiệp không nhỏ hơn 1.0, biết bảng đổi điểm như sau:

	Thang điểm 4	
	Điểm chữ	*Điểm số*
ĐẠT	A+	4.5
	A	4.0
	A-	3.5
	B+	3.0
	B	2.5
	B-	2.0
	C+	1.5
	C	1.0
KHÔNG ĐẠT	C-	0.5

4.1.1 Mã nguồn Oracle

Bước 1: Tạo VIEW đổi điểm chữ sang điểm số.

```
CREATE OR REPLACE VIEW VW_TAKES_NUMBER AS
SELECT ID, COURSE_ID, SEC_ID, SEMESTER, YEAR,
  DECODE(GRADE, 'A+', 4.5, 'A ', 4.0, 'A-', 3.5, 'B+', 3.0, 'B ', 2.5, 'B-', 2.0,
                'C+', 1.5, 'C ', 1.0, 'C-', 0.5, 0) POINT
FROM TAKES;
```

Bước 2: Tạo VIEW lấy ra điểm số cao nhất của sinh viên

```
CREATE OR REPLACE VIEW VW_TAKES_MAXPOINT AS
SELECT ID, COURSE_ID, MAX(POINT) POINT
FROM VW_TAKES_NUMBER GROUP BY ID, COURSE_ID;
```

Bước 3: Tạo VIEW chứa tín chỉ tương ứng

```
CREATE OR REPLACE VIEW VW_TAKES_CREDITS AS
SELECT VWM.ID, C.COURSE_ID, VWM.POINT, C.CREDITS
FROM VW_TAKES_MAXPOINT VWM JOIN COURSE C
ON C.COURSE_ID = VWM.COURSE_ID;
```

Bước 4: Tạo VIEW lấy ra tổng số tín chỉ

```
CREATE OR REPLACE VIEW VW_TAKES_MAXPOINT AS
SELECT ID, COURSE_ID, MAX(POINT) POINT
FROM VW_TAKES_NUMBER GROUP BY ID, COURSE_ID;
```

Bước 5: Tạo thủ tục kiểm tra điều kiện tốt nghiệp của sinh viên

```
CREATE OR REPLACE PROCEDURE SP_CHECK_GRADUATING (STUDENT_ID VARCHAR) AS
SUM_CRED NUMBER; CPA NUMBER;
BEGIN
  SELECT SUMCREDITS INTO SUM_CRED FROM VW_TAKES_SUMCREDITS WHERE ID = STUDENT_ID;
  IF SUM_CRED >=  128 THEN
    SELECT ROUND(SUM(T.POINT*C.CREDITS)/SUM(C.CREDITS),2) INTO CPA
    FROM VW_TAKES_MAXPOINT T JOIN COURSE C
    ON T.ID = STUDENT_ID AND T.COURSE_ID = C.COURSE_ID;
    IF CPA < 1.0 THEN
      dbms_output.put_line('Khong du dieu kien tot nghiep. Diem trung binh: '||CPA);
    ELSE
      dbms_output.put_line('Du dieu kien tot nghiep. Diem trung binh: '||CPA);
    END IF;
  ELSE
    dbms_output.put_line('Khong dieu kien tot nghiep. So tin chi: '||SUM_CRED);
  END IF;
  EXCEPTION
    WHEN no_data_found THEN dbms_output.put_line('Ma so sinh vien khong dung');
END;
```

Kết quả thực thi Oracle với ID = 1000

```
Connecting to the database htl.
Khong dieu kien tot nghiep. So tin chi: 44
Process exited.
Disconnecting from the database htl.
```

4.1.2 Mã nguồn SQL

Xây dựng các VIEW và thủ tục tương tự như Oracle.

```
--VIEW DOI DIEM CHU SANG DIEM SO
CREATE VIEW VW_TAKES_NUMBER AS
SELECT ID, COURSE_ID, SEC_ID, SEMESTER, YEAR,
    CASE grade
        WHEN 'A+' THEN 4.5
        WHEN 'A' THEN 4.0
        WHEN 'A-' THEN 3.5
        WHEN 'B+' THEN 3.0
        WHEN 'B' THEN 2.5
        WHEN 'B-' THEN 2.0
        WHEN 'C+' THEN 1.5
        WHEN 'C' THEN 1.0
        WHEN 'C-' THEN 0.5
        ELSE 0
    END AS POINT
FROM TAKES
GO
--VIEW LAY RA DIEM CAO NHAT
CREATE VIEW VW_TAKES_MAXPOINT AS
SELECT ID, COURSE_ID, MAX(POINT) AS POINT
FROM VW_TAKES_NUMBER GROUP BY ID, COURSE_ID;
GO

--VIEW CHUA TIN CHI TUONG UNG
CREATE VIEW VW_TAKES_CREDITS AS
SELECT VWM.ID, C.COURSE_ID, VWM.POINT, C.CREDITS
FROM VW_TAKES_MAXPOINT VWM JOIN COURSE C
ON C.COURSE_ID = VWM.COURSE_ID;
GO
```

```
--VIEW CHUA TONG SO TIN CHI
CREATE VIEW VW_TAKES_SUMCREDITS AS
SELECT ID, SUM(CREDITS) SUMCREDITS FROM VW_TAKES_CREDITS
WHERE POINT > 0.5 GROUP BY ID;
GO

--TAO THU TUC KIEM TRA DIEU KIEN TOT NGHIEP CUA SINH VIEN
CREATE PROCEDURE SP_CHECK_GRADUATING @STUDENT_ID VARCHAR(1000) AS
BEGIN
    DECLARE @SUM_CRED INT;
    DECLARE @CPA FLOAT;
    SELECT @SUM_CRED = SUMCREDITS FROM VW_TAKES_SUMCREDITS WHERE ID = @STUDENT_ID;
    IF @SUM_CRED >=  128
        BEGIN
            SELECT @CPA = ROUND(SUM(T.POINT*C.CREDITS)/SUM(C.CREDITS),2)
            FROM VW_TAKES_MAXPOINT T JOIN COURSE C
            ON T.ID = @STUDENT_ID AND T.COURSE_ID = C.COURSE_ID;
            IF @CPA < 1.0
            PRINT('Khong du dieu kien tot nghiep. Diem trung binh: ' + CONVERT(VARCHAR,@CPA))
            ELSE
            PRINT('Du dieu kien tot nghiep. Diem trung binh: ' +  CONVERT(VARCHAR,@CPA))
        END
    ELSE
        BEGIN
        PRINT('Khong dieu kien tot nghiep. So tin chi: ' +  CONVERT(VARCHAR,@SUM_CRED))
        END
END
GO
```

Kết quả

```
EXEC SP_CHECK_GRADUATING 1000
100 %
Messages
Khong dieu kien tot nghiep. So tin chi: 44
```

4.1.3 Bình luận

Trong Bước 1 Oracle, ta có thể thay hàm Decode bằng Case - When. Việc tạo VIEW cho ta những thuận lợi trong việc tính toán và đưa ra kết quả. Sự khác biệt của SQL và Oracle ở đây là trong SQL không có hàm Decode.

4.2 Bài 2

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào tên trường bất kỳ và một giá trị của trường (Ví dụ: SP_LOC_DU_LIEU 'dept_name', 'Physics'). Kết quả trả về là dữ liệu sau khi lọc theo giá trị của trường dữ liệu đó.

4.2.1 Mã nguồn Oracle

Bước 1

```
--TAO VIEW LAY DU LIEU THEO YEU CAU DE BAI

CREATE OR REPLACE VIEW VW_DATA AS
SELECT ST.ID, ST.NAME STUDENT_NAME, SE.YEAR, SE.SEMESTER, C.TITLE, TS.DAY, TS.START_HR,
TS.START_MIN, TS.END_HR, TS.END_MIN, SE.ROOM_NUMBER, SE.BUILDING, I.NAME INSTRUCTOR_NAME,
ST.DEPT_NAME
FROM SECTION SE
JOIN COURSE C ON SE.COURSE_ID = C.COURSE_ID
JOIN TEACHES TE ON SE.COURSE_ID = TE.COURSE_ID
  AND SE.SEC_ID = TE.SEC_ID
  AND SE.SEMESTER = TE.SEMESTER
  AND SE.YEAR = TE.YEAR
JOIN INSTRUCTOR I ON TE.ID = I.ID
JOIN TAKES TA ON SE.SEC_ID = TA.SEC_ID
  AND SE.SEMESTER = TA.SEMESTER
  AND SE.YEAR = TA.YEAR
  AND SE.COURSE_ID = TA.COURSE_ID
JOIN STUDENT ST ON TA.ID = ST.ID
LEFT JOIN TIME_SLOT TS ON SE.TIME_SLOT_ID = TS.TIME_SLOT_ID;
```

Bước 2

```
--TAO THU TUC LOC DU LIEU
CREATE OR REPLACE PROCEDURE SP_LOC_DU_LIEU (SEARCH_FIELD_NAME IN VARCHAR2,
SEARCH_VALUE IN VARCHAR2, SEARCH_RESULT_CURSOR OUT SYS_REFCURSOR) AS
BEGIN
  IF(SEARCH_FIELD_NAME = 'ID') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE ID = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'STUDENT_NAME') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE STUDENT_NAME = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'YEAR') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE YEAR = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'SEMESTER') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE SEMESTER = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'TITLE') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE TITLE = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'ROOM_NUMBER') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE ROOM_NUMBER = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'INSTRUCTOR_NAME') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE INSTRUCTOR_NAME = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'DEPT_NAME') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE DEPT_NAME = SEARCH_VALUE;
  ELSIF (SEARCH_FIELD_NAME = 'DAY') THEN OPEN SEARCH_RESULT_CURSOR FOR SELECT * FROM VW_DATA
  WHERE DAY = SEARCH_VALUE;
  ELSE dbms_output.put_line('Nhap sai');
  END IF;
END;
```

Kết quả thực thi

ID	STUDENT_N...	YEAR	SEMESTER	TITLE	DAY	START_HR	START_MIN	END_HR	END_MIN	ROOM_NUM...	BUILDING	INSTRUCTO...	DEPT_NAME
1000	Manber	2006	Fall	The Music of...	F	11	0	11	50	183	Taylor	Voronina	Civil Eng.
1000	Manber	2006	Fall	The Music of...	M	11	0	11	50	183	Taylor	Voronina	Civil Eng.
1000	Manber	2006	Fall	The Music of...	W	11	0	11	50	183	Taylor	Voronina	Civil Eng.
1000	Manber	2003	Spring	World History	R	10	30	11	45	261	Rathbone	Mingoz	Civil Eng.
1000	Manber	2003	Spring	World History	T	10	30	11	45	261	Rathbone	Mingoz	Civil Eng.
1000	Manber	2005	Fall	Embedded S...						143	Lamberton	Mingoz	Civil Eng.
1000	Manber	2010	Spring	Music of the ...	W	10	0	12	30	134	Lamberton	Mahmoud	Civil Eng.
1000	Manber	2004	Spring	Plastics						972	Power	Bondi	Civil Eng.
1000	Manber	2004	Fall	Video Gaming	F	13	0	13	50	113	Saucon	DAgostino	Civil Eng.
1000	Manber	2004	Fall	Video Gaming	M	13	0	13	50	113	Saucon	DAgostino	Civil Eng.
1000	Manber	2004	Fall	Video Gaming	W	13	0	13	50	113	Saucon	DAgostino	Civil Eng.
1000	Manber	2005	Spring	Geology	F	13	0	13	50	145	Fairchild	DAgostino	Civil Eng.
1000	Manber	2005	Spring	Geology	M	13	0	13	50	145	Fairchild	DAgostino	Civil Eng.
1000	Manber	2005	Spring	Geology	W	13	0	13	50	145	Fairchild	DAgostino	Civil Eng.
1000	Manber	2002	Spring	Heat Transfer	R	10	30	11	45	180	Saucon	Morris	Civil Eng.
1000	Manber	2002	Spring	Heat Transfer	T	10	30	11	45	180	Saucon	Morris	Civil Eng.
1000	Manber	2003	Fall	Tort Law						180	Saucon	Dale	Civil Eng.

4.2.2 Mã nguồn SQL

```
--TAO VIEW LAY DU LIEU THEO YEU CAU DE BAI

CREATE VIEW VW_DATA AS
SELECT ST.ID, ST.NAME STUDENT_NAME, SE.YEAR, SE.SEMESTER, C.TITLE, TS.DAY, TS.START_HR,
TS.START_MIN, TS.END_HR, TS.END_MIN, SE.ROOM_NUMBER, SE.BUILDING, I.NAME INSTRUCTOR_NAME,
ST.DEPT_NAME
FROM SECTION SE
JOIN COURSE C ON SE.COURSE_ID = C.COURSE_ID
JOIN TEACHES TE ON SE.COURSE_ID = TE.COURSE_ID
  AND SE.SEC_ID = TE.SEC_ID
  AND SE.SEMESTER = TE.SEMESTER
  AND SE.YEAR = TE.YEAR
JOIN INSTRUCTOR I ON TE.ID = I.ID
JOIN TAKES TA ON SE.SEC_ID = TA.SEC_ID
  AND SE.SEMESTER = TA.SEMESTER
  AND SE.YEAR = TA.YEAR
  AND SE.COURSE_ID = TA.COURSE_ID
JOIN STUDENT ST ON TA.ID = ST.ID
LEFT JOIN TIME_SLOT TS ON SE.TIME_SLOT_ID = TS.TIME_SLOT_ID;
GO
```

Bước 2

```
--TAO THU TUC LOC DU LIEU
CREATE PROCEDURE SP_LOC_DU_LIEU (@SEARCH_FIELD_NAME  VARCHAR(50),
@SEARCH_VALUE  VARCHAR(50)) AS
BEGIN
  DECLARE @SEARCH_RESULT NVARCHAR(1000)
  IF(@SEARCH_FIELD_NAME = 'ID') SELECT * FROM VW_DATA
  WHERE ID = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'STUDENT_NAME') SELECT * FROM VW_DATA
  WHERE STUDENT_NAME = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'YEAR') SELECT * FROM VW_DATA
  WHERE YEAR = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'SEMESTER') SELECT * FROM VW_DATA
  WHERE SEMESTER = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'TITLE') SELECT * FROM VW_DATA
  WHERE TITLE = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'ROOM_NUMBER') SELECT * FROM VW_DATA
  WHERE ROOM_NUMBER = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'INSTRUCTOR_NAME') SELECT * FROM VW_DATA
  WHERE INSTRUCTOR_NAME = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'DEPT_NAME') SELECT * FROM VW_DATA
  WHERE DEPT_NAME = @SEARCH_VALUE;
  IF(@SEARCH_FIELD_NAME = 'DAY') SELECT * FROM VW_DATA
  WHERE DAY = @SEARCH_VALUE;
  ELSE PRINT('Nhap sai');
END
```

Kết quả thực thi

```
EXEC dbo.SP_LOC_DU_LIEU @SEARCH_FIELD_NAME = 'ID', -- varchar(50)
    @SEARCH_VALUE = '1000' -- varchar(50)
```

100 %

Results | Messages

	ID	STUDENT_NAME	YEAR	SEMESTER	TITLE	DAY	START_HR	START_MIN	END_HR	END_MIN	ROOM_NUMBER	BUILDING	INSTRUCTOR_NAME
1	1000	Manber	2004	Fall	Video Gaming	F	13	0	13	50	113	Saucon	DAgostino
2	1000	Manber	2004	Fall	Video Gaming	M	13	0	13	50	113	Saucon	DAgostino
3	1000	Manber	2004	Fall	Video Gaming	W	13	0	13	50	113	Saucon	DAgostino
4	1000	Manber	2005	Spring	Geology	F	13	0	13	50	145	Fairchild	DAgostino
5	1000	Manber	2005	Spring	Geology	M	13	0	13	50	145	Fairchild	DAgostino
6	1000	Manber	2005	Spring	Geology	W	13	0	13	50	145	Fairchild	DAgostino
7	1000	Manber	2008	Spring	Animal Behavior	NULL	NULL	NULL	NULL	NULL	45	Nassau	DAgostino
8	1000	Manber	2009	Spring	Greek Tragedy	NULL	NULL	NULL	NULL	NULL	183	Taylor	DAgostino
9	1000	Manber	2004	Spring	Plastics	NULL	NULL	NULL	NULL	NULL	972	Power	Bondi
10	1000	Manber	2002	Spring	Heat Transfer	R	10	30	11	45	180	Saucon	Morris
11	1000	Manber	2002	Spring	Heat Transfer	T	10	30	11	45	180	Saucon	Morris

4.2.3 Bình luận

Bằng cách tạo VIEW chứa tất cả các trường theo yêu cầu của đề bài, chúng ta có thể dễ dàng tạo thủ tục lọc dữ liệu với đầu vào là tên trường và giá trị tương ứng.

4.3 Bài 3

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào một biến kiểu table gồm 2 trường: tên trường và một giá trị của trường. Kết quả trả về là dữ liệu sau khi lọc theo danh sách các giá trị của các trường dữ liệu đó.

4.3.1 Mã nguồn Oracle

Bước 1

```
--TAO OBJECT MOI GOM 2 TRUONG SEARCH_FIELD_NAME VAF SEARCH_VALUE
CREATE OR REPLACE TYPE SEARCH_OBJ FORCE
AS OBJECT
(
  SEARCH_FIELD_NAME VARCHAR(50),
  SEARCH_VALUE VARCHAR(50)
);

--TAO KIEU DU LIEU DANG TABLE CHO OBJECT
CREATE TYPE SEARCH_TAB IS TABLE OF SEARCH_OBJ;
```

Bước 2

```
--TAO THU TUC LOC DU LIEU
CREATE OR REPLACE PROCEDURE SP_LOC_DU_LIEU_3
(
  RESULT_TAB IN SEARCH_TAB,
  SEARCH_RESULT_CURSOR OUT SYS_REFCURSOR
) IS
BEGIN
  FOR i IN 1..RESULT_TAB.COUNT
  LOOP
  IF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'ID') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE ID = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'STUDENT_NAME') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE STUDENT_NAME = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'YEAR') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE YEAR = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'SEMESTER') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE SEMESTER = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'TITLE') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE TITLE = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'ROOM_NUMBER') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE ROOM_NUMBER = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'INSTRUCTOR_NAME') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE INSTRUCTOR_NAME = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'DEPT_NAME') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE DEPT_NAME = RESULT_TAB(i).SEARCH_VALUE;
  ELSIF(RESULT_TAB(i).SEARCH_FIELD_NAME = 'DAY') THEN OPEN SEARCH_RESULT_CURSOR
  FOR SELECT * FROM VW_DATA WHERE DAY = RESULT_TAB(i).SEARCH_VALUE;
  END IF;
  END LOOP;
END;
```

Kết quả thực thi

```
DECLARE
  RESULT_TAB HTL.SEARCH_TAB;
  SEARCH_RESULT_CURSOR SYS_REFCURSOR;
BEGIN
  RESULT_TAB := SEARCH_TAB();
  RESULT_TAB.EXTEND();
  RESULT_TAB(1) := SEARCH_OBJ('ID','1000');

  SP_LOC_DU_LIEU_3(
    RESULT_TAB => RESULT_TAB,
    SEARCH_RESULT_CURSOR => SEARCH_RESULT_CURSOR
  );
```

ID	STUDENT_N...	YEAR	SEMESTER	TITLE	DAY	START_HR	START_MIN	END_HR	END_MIN	ROOM_NUM...	BUILDING	INSTRUCTO...	DEPT_NAME
1000	Manber	2006	Fall	The Music of...	F	11	0	11	50	183	Taylor	Voronina	Civil Eng.
1000	Manber	2006	Fall	The Music of...	M	11	0	11	50	183	Taylor	Voronina	Civil Eng.
1000	Manber	2006	Fall	The Music of...	W	11	0	11	50	183	Taylor	Voronina	Civil Eng.
1000	Manber	2003	Spring	World History	R	10	30	11	45	261	Rathbone	Mingoz	Civil Eng.
1000	Manber	2003	Spring	World History	T	10	30	11	45	261	Rathbone	Mingoz	Civil Eng.
1000	Manber	2005	Fall	Embedded S...						143	Lamberton	Mingoz	Civil Eng.
1000	Manber	2010	Spring	Music of the ...	W	10	0	12	30	134	Lamberton	Mahmoud	Civil Eng.
1000	Manber	2004	Spring	Plastics						972	Power	Bondi	Civil Eng.
1000	Manber	2004	Fall	Video Gaming	F	13	0	13	50	113	Saucon	DAgostino	Civil Eng.
1000	Manber	2004	Fall	Video Gaming	M	13	0	13	50	113	Saucon	DAgostino	Civil Eng.
1000	Manber	2004	Fall	Video Gaming	W	13	0	13	50	113	Saucon	DAgostino	Civil Eng.
1000	Manber	2005	Spring	Geology	F	13	0	13	50	145	Fairchild	DAgostino	Civil Eng.
1000	Manber	2005	Spring	Geology	M	13	0	13	50	145	Fairchild	DAgostino	Civil Eng.
1000	Manber	2005	Spring	Geology	W	13	0	13	50	145	Fairchild	DAgostino	Civil Eng.
1000	Manber	2002	Spring	Heat Transfer	R	10	30	11	45	180	Saucon	Morris	Civil Eng.
1000	Manber	2002	Spring	Heat Transfer	T	10	30	11	45	180	Saucon	Morris	Civil Eng.
1000	Manber	2003	Fall	Tort Law						180	Saucon	Dale	Civil Eng.
1000	Manber	2003	Spring	African Histo...						113	Saucon	Dale	Civil Eng.

4.3.2 Mã nguồn SQL

```
--TAO OBJECT MOI GOM 2 TRUONG SEARCH_FIELD_NAME VAF SEARCH_VALUE
CREATE TYPE SEARCH_TAB AS TABLE
(
  SEARCH_FIELD_NAME VARCHAR(50),
  SEARCH_VALUE VARCHAR(50)
);
GO

--TAO THU TUC LOC DU LIEU

CREATE PROCEDURE SP_LOC_DU_LIEU_3
  @SEARCH_TAB SEARCH_TAB READONLY
AS
BEGIN
  DECLARE @COL_NAME VARCHAR(50)
  DECLARE @VAL VARCHAR(1000)
  DECLARE @i INT = 0
  DECLARE @COUNT INT;
  DECLARE @RESULT_TAB TABLE(ID INT, COL_NAME VARCHAR(50), VAL VARCHAR(50));
```

```
  INSERT INTO @RESULT_TAB SELECT ROW_NUMBER() OVER (ORDER BY SEARCH_FIELD_NAME),
  FROM @SEARCH_TAB;
  SELECT @COUNT = COUNT(SEARCH_FIELD_NAME) FROM @SEARCH_TAB;
  WHILE @i < @COUNT
  BEGIN
    SET @i = @i + 1
    SELECT TOP 1 @COL_NAME = COL_NAME, @VAL = VAL FROM @RESULT_TAB
    WHERE ID = @i;
      IF (@COL_NAME = 'ID') SELECT * FROM VW_DATA
      WHERE ID = @VAL
      ELSE IF(@COL_NAME = 'STUDENT_NAME') SELECT * FROM VW_DATA
      WHERE STUDENT_NAME = @VAL;
      ELSE IF(@COL_NAME = 'YEAR') SELECT * FROM VW_DATA
      WHERE YEAR = @VAL;
      ELSE IF (@COL_NAME = 'SEMESTER') SELECT * FROM VW_DATA
      WHERE SEMESTER = @VAL;
      ELSE IF(@COL_NAME = 'TITLE') SELECT * FROM VW_DATA
      WHERE TITLE =@VAL;
      ELSE IF(@COL_NAME = 'ROOM_NUMBER') SELECT * FROM VW_DATA
      WHERE ROOM_NUMBER = @VAL;
      ELSE IF(@COL_NAME = 'INSTRUCTOR_NAME') SELECT * FROM VW_DATA
      WHERE INSTRUCTOR_NAME = @VAL;
      ELSE IF(@COL_NAME = 'DEPT_NAME') SELECT * FROM VW_DATA
      WHERE DEPT_NAME = @VAL;
      ELSE IF(@COL_NAME = 'DAY') SELECT * FROM VW_DATA
      WHERE DAY = @VAL;
 END
END
GO
```

Kết quả thực thi

```
DECLARE @TABLE SEARCH_TAB
  INSERT INTO @TABLE VALUES ('ID','1000')
  EXEC dbo.SP_LOC_DU_LIEU_3 @SEARCH_TAB = @TABLE -- SEARCH_TAB
```

Results | Messages

ID	STUDENT_NAME	YEAR	SEMESTER	TITLE	DAY	START_HR	START_MIN	END_HR	END_MIN	ROOM_NUMBER
1000	Manber	2003	Fall	Tort Law	NULL	NULL	NULL	NULL	NULL	180
1000	Manber	2003	Spring	African History	NULL	NULL	NULL	NULL	NULL	113
1000	Manber	2010	Spring	Music of the 50s	W	10	0	12	30	134
1000	Manber	2006	Fall	Bacteriology	NULL	NULL	NULL	NULL	NULL	180
1000	Manber	2006	Fall	The Music of the Ramones	F	11	0	11	50	183
1000	Manber	2006	Fall	The Music of the Ramones	M	11	0	11	50	183
1000	Manber	2006	Fall	The Music of the Ramones	W	11	0	11	50	183
1000	Manber	2005	Fall	Embedded Systems	NULL	NULL	NULL	NULL	NULL	143
1000	Manber	2003	Spring	World History	R	10	30	11	45	261
1000	Manber	2003	Spring	World History	T	10	30	11	45	261
1000	Manber	2002	Spring	Heat Transfer	R	10	30	11	45	180

4.3.3 Bình luận

Sử dụng kiểu dữ liệu TYPE OBJECT để lưu giữ các giá trị đầu vào rồi tiến hành lọc tương tự như bài 2.

4.4 Bài 4

Sinh viên *A* muốn học môn 'Mobile Computing' hỏi *A* cần phải học qua những môn gì?

4.4.1 Mã nguồn Oracle

4.4.2 Mã nguồn SQL

4.4.3 Bình luận

4.5 Bài 2

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào tên trường bất kỳ và một giá trị của trường (Ví dụ: SP_LOC_DU_LIEU 'dept_name', 'Physics'). Kết quả trả về là dữ liệu sau khi lọc theo giá trị của trường dữ liệu đó.

4.5.1 Mã nguồn Oracle

4.5.2 Mã nguồn SQL

4.5.3 Bình luận

4.6 Bài 2

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào tên trường bất kỳ và một giá trị của trường (Ví dụ: SP_LOC_DU_LIEU 'dept_name', 'Physics'). Kết quả trả về là dữ liệu sau khi lọc theo giá trị của trường dữ liệu đó.

4.6.1 Mã nguồn Oracle

4.6.2 Mã nguồn SQL

4.6.3 Bình luận

4.7 Bài 2

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào tên trường bất kỳ và một giá trị của trường (Ví dụ: SP_LOC_DU_LIEU 'dept_name', 'Physics'). Kết quả trả về là dữ liệu sau khi lọc theo giá trị của trường dữ liệu đó.

4.7.1 Mã nguồn Oracle

4.7.2 Mã nguồn SQL

4.7.3 Bình luận

4.8 Bài 2

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào tên trường bất kỳ và một giá trị của trường (Ví dụ: SP_LOC_DU_LIEU 'dept_name', 'Physics'). Kết quả trả về là dữ liệu sau khi lọc theo giá trị của trường dữ liệu đó.

4.8.1 Mã nguồn Oracle

4.8.2 Mã nguồn SQL

4.8.3 Bình luận

4.9 Bài 2

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào tên trường bất kỳ và một giá trị của trường (Ví dụ: SP_LOC_DU_LIEU 'dept_name', 'Physics'). Kết quả trả về là dữ liệu sau khi lọc theo giá trị của trường dữ liệu đó.

4.9.1 Mã nguồn Oracle

4.9.2 Mã nguồn SQL

4.9.3 Bình luận

4.10 Bài 2

Viết thủ tục SP_LOC_DU_LIEU cho phép nhập vào tên trường bất kỳ và một giá trị của trường (Ví dụ: SP_LOC_DU_LIEU 'dept_name', 'Physics'). Kết quả trả về là dữ liệu sau khi lọc theo giá trị của trường dữ liệu đó.

4.10.1 Mã nguồn Oracle

4.10.2 Mã nguồn SQL

4.10.3 Bình luận