|  |
| --- |
| **TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**  **VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC**  BLACK AND WHITE    **Báo cáo kết thúc môn học**  **Cơ sở dữ liệu nâng cao**        **Sinh viên thực hiện: LÊ THỊ DUYÊN**  **Lớp: Toán Tin 02 K60**    HÀ NỘI – 2018 |

**PHẦN MỞ ĐẦU**

Trong cuộc cách mạng công nghiệp 4.0, công nghệ thông tin phát triển như vũ bão. Xuất phát từ sự bùng nổ dữ liệu trong những năm gần đây thông qua mạng internet, khái niệm “dữ liệu lớn” ra đời, kéo theo sự phát triển mạnh mẽ về khoa học dữ liệu với mục đích trích rút thông tin, tri thức nguồn dữ liệu vô hạn nhằm đáp ứng các nhu cầu từ xã hội. Các công nghệ mới về khai phá dữ liệu lớn được ra đời trong thời gian gần đây và đang trở thành một xu hướng tất yếu trong tương lai

Học phần “Cơ sở dữ liệu nâng cao” cung cấp cho sinh viên các kiến thức nề tảng trong việc phát triển và xử lý dữ liệu lớn. Trong tiểu luận này, em xin được trình bày một số vấn đề về công nghệ xử lý dữ liệu, Bố cục của báo cáo gồm phần mở đầu, 4 chương, phần kết luận và tại liệu tài khảo:

* Chương I: Cơ sở dữ liệu lớn
* Chương II: Cơ sở dữ liệu phân tán
* Chương III: Hệ quản trị cơ sở dữ liệu Oracle
* Chương IV: Bài tập kết thúc môn

Em xin chân thành cảm ơn Thầy Nguyễn Danh Tú, Thầy Nguyễn Tuấn Dũng, Cô Nguyễn Thị Thanh Huyền - các thầy cô đã trực tiếp giảng dạy chúng em trong học phần này. Nhờ có kiến thức chuyên sâu và sự tâm huyết, nhiệt tình của các thầy cô, chúng em mới có thể tiếp thu được những kiến thức về công nghệ xử lý dữ liệu trong thời đại mới.

Mặc dù em đã cố gắng hoàn thành bài báo cáo trong phạm vi và khả năng cho phép, do kiến thức, thời gian và kinh nghiệm có hạn nên không tránh khỏi những thiếu sót. Vì vậy em rất mong nhận được sự cảm thông và ý kiến đóng góp của quý thầy cô để bài báo cáo có thể hoàn thiện hơn.

Em xin chân thành cảm ơn!

**KẾT LUẬN**

Thông qua báo cáo kết thúc môn học, em đã nắm được các kiến thức nền tảng về CSDL lớn cũng như CSDL phân tán, từ đó làm nền tảng cho các công việc sau này. Đồng thời, em cũng nắm được các kiến thức về truy vấn dữ liệu cũng như nắm được cách khởi tạo các thủ tục, cảnh báo,... trong quá trình thực hành với các hệ CSDL gồm Oracle, SQL Server. Cụ thể, em đã làm được những công việc như sau

* Nắm được khái niệm về dữ liệu lớn, hệ cở dữ liệu NoSQL.
* Nắm được các khái niệm cở bảo về một hệ CSDL phân tán
* Nắm được kiến trúc, các khái niệm, đặc tính cơ bản của hệ quản trị cơ sở dữ liệu Oracle.
* Các kỹ năng thực hành cơ bản với cơ sở dữ liệu Oracle, MySQL. Có thể tạo được một ứng dụng đơn giản về xử lý dữ liệu trong Oracle, MySQl.
* Có được một số kỹ năng như tìm kiếm, đọc hiểu tài liệu, viết báo cáo.

Một lần nữa, em xin chân thành cảm ơn thầy cô và các bạn đã giúp đỡ em hoàn thành học phần “Cơ sở dữ liệu nâng cao”.

**PHẦN 1: CƠ SỞ DỮ LIỆU LỚN**

* 1. **Dữ liệu lớn (Big data)** 
     1. **Định nghĩa**

Big data thường được định nghĩa như là một tập hợp dữ liệu lớn, bao gồm các dữ liệu có cấu trúc, không có cấu trúc hoặc được cấu trúc không đầy đủ, mà mỗi dữ liệu trong đó đều có thể được sử dụng để khai thác thành các thông tin chi tiết. Những tập dữ liệu này thường lớn và phức tạp đến nỗi mà các phần mềm xử lý dữ liệu bình thường khó có thể thu thập, lưu trữ và xử lý nó trong một khoảng thời gian phù hợp, bởi khối lượng của nó có thể lên đến petabyte (hàng triệu gigabyte) hoặc thậm chí là Exabyte.

Trong năm 2012 thì Gartner đã cập nhật định nghĩa big data như sau: "Dữ liệu lớn là khối lượng lớn, tốc độ cao và/hoặc loại hình thông tin rất đa dạng mà yêu cầu phương thức xử lý mới để cho phép tăng cường ra quyết định, khám phá bên trong và xử lý tối ưu".

Theo đó các định nghĩa để mô ta dữ liệu lớn theo mô hình '3Vs' của Gartner vẫn được sử dụng rộng rãi, và có định nghĩa phù hơp là: "Dữ liệu lớn tiêu biểu cho tập thông tin mà đặc điểm như khối lượng lớn(Volume), tốc độ cao(Velocity) và đa dạng(Variety) để yêu cầu phương thức phân tích và công nghệ riêng biệt để biến nó thành có giá trị". Thêm nữa, vài tổ chức đã thêm vào tính xác thực(Veracity) để mô tả về nó. 3Vs đã được mở rộng để bổ sung đặc tính của dữ liệu lớn.

* + 1. **Đặc trưng**

Big Data được mô tả bởi những đặc trưng sau:

* **Volume (Dung lượng):** là sự tăng trưởng về mặt khối lượng. Dữ liệu trong các hệ thống thông tin luôn luôn và không ngừng tăng lên về mặt kích thước (khối lượng). Chúng ta có thể tìm thấy dữ liệu trong các định dạng video, music, image lớn trên các kênh truyền thông xã hội. Khối lượng dữ liệu của một hệ thống thông tin có thể lên đến hàng Terabyte và Petabyte.
* **Variety (Tính đa dạng):** là sự tăng lên về tính đa dạng của dữ liệu. Dữ liệu không chỉ ở dạng có cấu trúc, mà còn bao gồm rất nhiều kiểu dữ liệu phi cấu trúc nữa như video, hình ảnh, dữ liệu cảm biến, cũng như các file log. Dữ liệu của một doanh nghiệp hay một hệ thống thông tin ngày nay không còn đơn giản chỉ có một hoặc một vài loại dữ liệu nữa, mà tính đa dạng của nó cũng đang ngày càng tăng lên làm cho tính phức tạp của dữ liệu ngày càng phức tạp hơn.
* **Velocity (Vận tốc):** là sự tăng trưởng về mặt tốc độ. Bên cạnh sự tăng trưởng về khối lượng, tốc độ tăng trưởng của dữ liệu cũng tăng lên một cách chóng mặt. Một ví dụ đơn giản là trên các mạng xã hội đôi khi các thông báo cách đó vài giây (tweet, status,….) đã là cũ và không được người dùng quan tâm. Người dùng thường loại bỏ các tin nhắn cũ và chỉ chú ý đến các cập nhật gần nhất. Sự chuyển động của dữ liệu bây giờ hầu như là thực tế (real time) và tốc độ cập nhật thông tin đã giảm xuống đơn vị hàng mili giây.
* **Veracity (Tính xác thực):** là tính xác thực của dữ liệu. Với xu hướng Social 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 & chính xác của dữ liệu ngày một khó khăn hơn. Bài toán phân tích và loại bỏ dữ liệu thiếu chính xác và nhiễu đang là tính chất quan trọng của Big Data.

Big data được tạo thành từ các nguồn bao gồm các trang web, phương tiện truyền thông xã hội, máy tính để bàn và ứng dụng di động, thí nghiệm khoa học, các cảm biến và các thiết bị khác trên internet (IoT).

* + 1. **Những vấn đề liên quan**

**1.1.3.1. Phân tích dữ liệu Big Data**

Khi đã có dữ liệu, điều quan trọng nhất là phải phân tích những dữ liệu đó để chúng thực sự mang lại lợi ích như tăng doanh thu, cải thiện dịch vụ khách hàng, nâng cao hiệu suất và tăng sức cạnh tranh tổng thể.

Việc phân tích dữ liệu bao gồm việc kiểm tra các tập dữ liệu, từ đó đưa thông tin chi tiết hoặc rút ra kết luận về xu hướng và dự đoán về hoạt động trong tương lai. Nhờ đó, dữ liệu có thể áp dụng vào công việc như lựa chọn thời gian, địa điểm để quảng cáo cho sản phẩm và dịch vụ, điều rất có ích cho các công ty cần những chiến dịch quảng bá sản phẩm theo thời điểm.

Phân tích dữ liệu bao gồm:

* + - Phân tích dữ liệu thăm dò (để xác định các mẫu và mối quan hệ trong dữ liệu)
    - Phân tích dữ liệu xác nhận (áp dụng các kỹ thuật thống kê để tìm hiểu xem giả định về một tập dữ liệu cụ thể có đúng không)
    - Phân tích dữ liệu định lượng (so sánh thống kê)
    - Phân tích dữ liệu định tính (tập trung vào dữ liệu phi số như video, hình ảnh và văn bản).

**1.1.3.2. Xây dựng cơ sở hạ tầng Big Data**

Muốn sử dụng và khai thác Big Data một cách hiệu quả, điều cần thiết là phải xây dựng cơ sở hạ tầng đủ để thu thập và lưu trữ dữ liệu, cung cấp quyền truy cập và bảo mật thông tin trong khi lưu trữ và chuyển tiếp, bao gồm các hệ thống lưu trữ và máy chủ, phần mềm quản lý, tích hợp dữ liệu, phần mềm phân tích dữ liệu và một số thành phần khác. Điều này có thể được hiện thực hóa bởi các data center lớn, các dịch vụ điện toán đám mây cũng góp phần giải quyết bài toán này.

Dữ liệu có thể đến từ nhiều nguồn khác nhau như các ứng dụng web, kênh truyền thông xã hội, ứng dụng dành cho thiết bị di động và lưu trữ email, ngoài ra khi IoT trở nên bùng nổ, các cảm biến trong sản phẩm cũng góp phần tạo dựng và chuyển hóa dữ liệu. Để lưu trữ tất cả dữ liệu đến, có một số lựa chọn phổ biến là kho dữ liệu truyền thống, xây dựng hồ dữ liệu hoặc tận dụng lưu trữ trên đám mây.

Ngoài ra, còn cần xây dựng cơ sở hạ tầng bảo mật, bao gồm việc mã hóa dữ liệu, xác thực người dùng và các quyền truy cập, giám sát hệ thống, xây dựng tường lửa, quản lý doanh nghiệp và các sản phẩm khác để bảo vệ hệ thống và dữ liệu.

**1.1.3.3. Tầm quan trọng của Big Data**

Big data là công nghệ thu thập thông tin quy mô lớn từ các website. Các doanh nghiệp thường vận dụng công cụ này nhằm phục vụ công việc dự đoán xu hướng thị trường, nâng cao chất lượng sản phẩm hoặc dịch vụ hiện có, tạo ra sản phẩm mới hoặc tìm hiểu về hành vi khách hàng.

Phân tích dữ liệu cũng có thể giúp các doanh nghiệp thích nghi, tạo ra nội dung website thu hút nhiều khách hàng hơn, có được cái nhìn sâu sắc vào hành vi mua hàng. Dữ liệu càng nhiều thì càng tốt cho công ty. Để làm được như vậy, doanh nghiệp nên cung cấp nội dung trên nhiều nền tảng social media, nhằm thu thập được nhiều thông tin từ những điểm tiếp xúc với khách hàng.

Bằng cách tìm hiểu qua hệ thống cơ sở dữ liệu, công ty có thể tạo ra nội dung có liên quan hơn với người đọc. Xa hơn một chút Big Data có thể giúp các cơ quan Chính phủ dự đoán được tỉ lệ thất nghiệp, xu hướng nghề nghiệp của tương lai để tập trung đầu tư cho các hạng mục đó, cắt giảm chi phí, kích thích tăng trưởng kinh tế, thậm chí là ra phương án phòng ngừa trước một dịch bệnh nào đó.

Tóm lại, Big Data chính là thách thức đặt ra cho các doanh nghiệp trong thời đại công nghệ số. Một khi tận dụng được tối đa nguồn dữ liệu lớn thì có hội thành công sẽ lớn hơn nhiều lần. Tuy vẫn còn những chỉ trích về Big Data nhưng đây là một lĩnh vực còn rất mới nên chúng ta hãy chờ đón sự tiến hóa của Big Data trong tương lai.

**1.2. Cơ sở dữ liệu lớn**

Thuật ngữ Big Data cũng đề cập đến 3 khía cạnh (3V): Dung lượng (Volume), sự đa dạng (Variety), và tốc độ (Velocity). Dung lượng là phạm vi dữ liệu, sự đa dạng là các cơ sở dữ liệu không đồng nhất, và tốc độ chính là tốc độ mà các phân tích được thực hiện, thường với mục tiêu là phải xử lý được ở thời gian thực.

Các cơ sở dữ liệu liên quan cực lớn – lên đến hàng terabyte đến zettabyte (1ZB tương đương 909.494.701 TB). Ngoài phạm vi cơ sở dữ liệu này, thì dữ liệu cũng có nhiều loại khác nhau: từ dữ liệu cấu trúc, bán cấu trúc và phi cấu trúc, các dữ liệu đó có thể xuất phát từ nhiều nguồn khác nhau. Điều này không nói đến vấn đề tất cả dữ liệu đó đều được tạo ra từ đâu. Dữ liệu lớn xuất phát từ tất cả các mọi nơi, trong đó có từ web, truyền thông xã hội, mạng, các tệp tin sổ ghi, tệp tin video, các bộ cảm biến và từ các thiết bị di động. Trong đó nguồn dữ liệu từ các thiết bị di động là đặc biệt quan trọng bởi hầu hết chúng ta cứ kè kè điện thoại bên mình 24/7, và mỗi chiếc điện thoại thì có hàng loạt bộ cảm biến, trong đó có GPS, camera, microphone, và cảm biến nhận biết sự di chuyển. Hơn nữa, phần lớn thời gian người dùng sử dụng diện thoại không phải để gọi mà là vào các ứng dụng như là email, chơi game, lướt web và các ứng dụng mạng xã hội – tỷ lệ vào các ứng dụng chiếm đến 90% thời gian người sử dụng điện thoại. Một nguồn dữ liệu rất lớn tạo nên Big Data là từ dữ liệu di động, và nguồn dữ liệu này liên tục được tạo ra với tốc độ chóng mặt.

**1.2.1. Cơ sở dữ liêu NoSQL**

Khi làm việc với database, chúng ta đã quá quen với SQLServer, 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. Vào năm 2009, Eric Evans, nhân viên của Rackspace giới

thiệu lại thuật ngữ NoSQL trong một hội thảo về cơ sở dữ liệu nguồn mở phân tán. 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.

Cơ sở dữ liệu NoSQL 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. Và cơ sở dữ liệu NoSQL dược sử dụng rộng rãi hiện nay vì lượng dữ liệu mà các hệ thống cần phải xử lý giờ đây ngày 1 lớn. Ví dụ như Google, Facebook phải lưu trữ và xử lý một lượng dữ liệu cực lớn mỗi ngày.

**1.2.2. Đặc điểm của sơ sở dữ liệu NoSQL**

Một số đặc điểm chung của sơ sở dữ liệu NoSQL bao gồm:

* High Scalability: Gần như không có một giới hạn cho dữ liệu và người dùng trên hệ thống.
* High Scalability: Gần như không có một giới hạn cho dữ liệu và người dùng trên 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).

**1.2.3. Phân loại cơ sở dữ liệu NoSQL**

Có bốn loại chung (loại phổ biến nhất) của cơ sở dữ liệu NoSQL. Mỗi loại đều có các thuộc tính và giới hạn riêng. Không có một giải pháp duy nhất nào tốt hơn tất cả các giải pháp khác, tuy nhiên có một số cơ sở dữ liệu tốt hơn để giải quyết các vấn đề cụ thể. Để làm rõ cơ sở dữ liệu NoSQL, hãy thảo luận các loại phổ biến nhất:

* Key-value stores
* Column-oriented databases (column-family).
* Graph databases
* Document Oriented databases

**PHẦN 2: CƠ SỞ DỮ LIỆU PHÂN TÁN**

**2.1. Khái niệm cơ sở dữ liệu phân tán (Distributed Database)**

Cơ sở dữ liệu phân tán là một tuyển tập dữ liệu có quan hệ logic vói nhau, được phân bố trên các máy tính của một mạng máy tính. Hệ quản trị cơ sở dữ liệu phân tán là hệ thống phần mềm cho phép quản lý cơ sở dữ liệu phân tán và đảm bảo tính trong suốt về sự phân tán đối với người dùng.

Hệ quản trị cơ sở dữ liệu có thể được yêu cầu và thực hiện trên máy tính ở một nút trong hệ cơ sở dữ liệu phân tán và chỉ liên quan đế cơ sở dữ liệu tại nút đó hoặc yêu cầu truy nhập dữ liệu ở nhiều thông qua hệ thống truyền thông

**Ưu điểm của cơ sở dữ liệu phân tán:**

* Phù hợp với cấu trúc tổ chức
* Nâng cao khả năng chia sẻ và tính tự trị địa phương
* Nâng cao tính sẵn sàng
* Nâng cao tính tin cậy
* Nâng cao hiệu năng
* Dễ mở rộng

**Nhược điểm của sơ sở dữ liệu phân tán:**

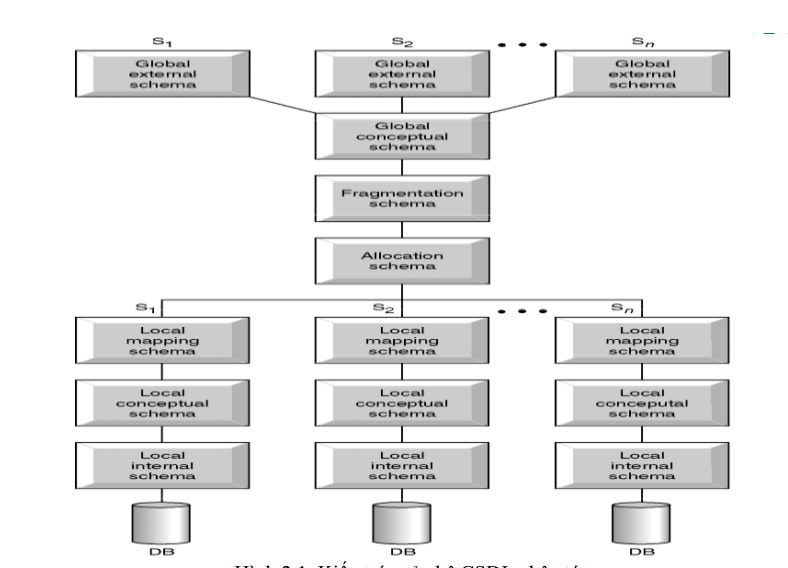
* Thiết kế cơ sở dữ liệu phức tạp hơn
* Khó điều khiển tính nhất quán dữ liệu
* Khó phát hiện lỗi và sửa lỗi
* Giá thành cao
* Thiếu chuẩn mực
* Thiếu kinh nghiệm
* Vấn đề về bảo mật

**2.2. Kiến trúc của một hệ cơ sở dữ liệu phân tán**

Do sự đa dạng và phức tạp nên không có kiến trúc nào được công nhận là tương đương với kiến trúc 3 mức ANSI/SPARC. Do đó chúng ta có thể tham khảo kiến trúc với:

* Tập lược đồ ngoài toàn cục (Global external schema): định nghĩa tất cả dữ liệu được chứa trong CSDLPT
* Lược đồ khái niệm toàn cục (Global conceptual schema – GCS): mỗi quan hệ toàn cục có thể được phân chia thành nhiều phần nhỏ không giao nhau.
* Lược đồ phân đoạn (Fragmentation schema): ánh xạ giữa quan hệ toàn cục và các đoạn.
* Lược đồ định vị (allocation schema): định vị các nút chứa một đoạn
* Tập lược đồ của mỗi hệ CSDL cục bộ tuần theo tiêu chuẩn 3 mức ANSI/SPARC.

Có thể khuyết một số mức tùy thuộc vào mức trong suốt được cung cấp.



**Hình 2.1. Kiến trúc của hệ CSDL phân tán**

**2.3. Phân loại cơ sở dữ liệu phân tán**

Hệ quản trị cơ sở dữ liệu phân tán cung cấp công cụ như tạo lập và quản lý cơ sở dữ liệu phân tán. Hệ quản cơ sở dữ liệu phân tán có chức năng hỗ trợ việc tạo và bảo trì cơ sở dữ liệu phân tán, chúng có các thành phần tương tự như một hệ quản trị cơ sở dữ liệu tập trung và các thành phần hỗ trợ trong việc chuyền tải dữ liệu đến các trạm và ngược lại.

Hệ quản trị cơ sở dữ liệu phải đảm bảo tính nguyên tố của các giao tác con, đồng bộ giữa các giao tác con với các giao tác cục bộ khác được thi hành tương tranh tại một nút và đồng bộ các giao tác con với giao tác toàn cục đang thi hành đồng thời tại cùng/hoặc khác nút.

Bộ quản lý giao tác(hay bộ điều phối giao tác – transaction coordinator) tại mỗi nút phải điều phối các giao tác cục bộ và toàn cục được khởi tạo tại nút đó. Và các môdun chính trong hệ quản trị CSDL phân tán bao gồm

* Phần mềm xử lý dữ liệu: Chịu trách nhiệm quản lý dữ liệu cục bộ tại các trạm (giống với phần mềm CSDL tập trung)
* Phần mềm xử lý ứng dụng: Chịu trách nhiệm với phần lớn chức năng phân tán
* Phần mềm truyền thông (thường liên kết với một hệ điều hành phân tán): Cung cấp các hàm nguyên thủy về truyền thông được các phần mềm xử lý ứng dụng dùng để truyền lệnh và dữ liệu giữa các trạm

Cơ sở dữ liệu phân tán được phân thành hai loại:

|  |  |
| --- | --- |
| Hệ CSDL phân tán thuần nhất  Homogeneous DB | Hệ CSDL phân tán hỗn tạp  Heterogeneous DB |
| - Tất cả các nút cùng sử dụng một loại hệ quản  trị cơ sở dữ liệu  - Thuận lợi cho việc trưởng và cho phép nâng  cao hiệu năng  - Có được bằng cách chia một tập CSDL  thành một tập CSDL cục bộ | - Các nút có thể thực hiện trên các hệ quản trị  cơ sở dữ liệu khác nhau  - Xảy ra khi các nút đã cài đặt hệ quản trị cơ  sở dữ liệu khác nhau  - Có được bằng cách tích hợp các CSDL cục  bộ đã có sẵn |

**2.4. Độ trong suốt phân tán(Distributed transparency)**

Tính trong suốt của một hệ phân tán được hiểu như là việc che khuất đi các thành phần riêng biệt của hệ đối với người sử dụng và những người lập trình ứng dụng.

Có các loại trong suốt trong hệ phân tán bao gồm:

* Trong suốt phân đoạn(fragment transparency): là mức trong suốt cao nhất, người sử dụng hoặc chương trình ứng dụng chỉ làm việc trên các quan hệ. Khi dữ liệu được phân đoạn thì việc truy cập vào CSDL được thực hiện bình thường như là chưa bị phân tán và không ảnh hưởng đến người sử dụng.
* Trong suốt định vị(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.
* Trong suốt ánh xạ địa phương(local mapping transparency): Ứng dụng tham chiếu đến các đối tượng có các tên độc lập từ các hệ thống cục bộ địa phương hoặc được cài đặt trên một hệ thống không đồng nhất nhưng được sử dụng như một hệ thống đồng nhất.
* Trong suốt nhân bản(replication transparency): Có mức trong suốt tương đương với mức trong suốt định vị, có nghĩa là người dùng không biết bản sao của đoạn đặt ở vị trí nào.
* Không trong suốt (no transparency)

**PHẦN 3: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU ORACLE**

**3.1. Oracle lecture 1**

Phần này giới thiệu về hệ quản trị cơ sở dữ liệu Oracle và cách cài đặt Oracle 10G

**3.2. Oracle lecture 2**

**Ex1:** Tạo bảng DEPT theo câu truy vấn cho trước, xác nhận truy vấn thành công

Create Table dept

(ID number(7) CONSTRAINT dept\_department\_id PRIMARY KEY,

NAME varchar2(25));

**Ex2:** Insert vào bảng DEPT với dữ liệu từ bảng Department. Chỉ bao gồm các cột cần tương ứng của bảng.

INSERT INTO dept

SELECT dept\_department\_id,

department\_name

FROM department;

**Ex2**: Insert vào bảng DEPT với dữ liệu từ bảng Department. Chỉ bao gồm các cột cần tương ứng của bảng

CREATE TABLE employees2 AS

SELECT employee\_id id,

first\_name,

last\_name,

salary,

department\_id dept\_id

FROM employees;

**3.2. Lecture 3**

**Ex1**: Các nhân viên trong phòng nhân sự muốn ẩn một số dữ liệu trong bảng EMPLOYEES. Họ muốn có một View “EMPLOYEES\_VU” bao gồm mã nhân viên, tên nhân viên và mã phòng ban từ bảng EMPOLYEES. Họ muốn tiêu đề cho tên nhân viên là EMPLOYEE.

CREATE OR REPLACE VIEW employees\_vu AS

SELECT employee\_id,

last\_name employee,

department\_id

FROM

employees;

**Ex2**: Department 50 cần truy cập vào dữ liệu nhân viên của mình. Tạo view có tên DEPT50 chứa mã nhân viên, tên nhân viên và mã phòng ban cho tất cả nhân viên trong Department 50, Tên các cột tương ứng là EMPNO, EMPLOYEE và DEPTNO, vì lý do bảo mật, không cho phép nhân viên chuyển sang Department khác

CREATE OR REPLACE VIEW dept50 (empno,

employee,

deptno) AS

SELECT employee\_id,

last\_name,

department\_id

FROM employees

WHERE department\_id = 50

WITH READ ONLY;

SELECT \*

FROM dept50;

**Ex3**: Tạo Sequence sử dụng cho cột khóa chính của bảng DEPT. Sequence bắt đầu từ 200 và có giá trị tối đa là 1000. Bước nhảy là 10. Đặt tên cho. Đặt tên cho Sequence là DEPT\_ID\_SEQ. Để kiểm tra Sequence vừa tạo, hãy viết một câu lệnh để chèn hai hàng vào bảng DEPT, Education and Administration

CREATE SEQUENCE dept\_id\_seq

INCREMENT BY 10

START WITH 200

MAXVALUE 1000

NOCACHE

NOCYCLE;

INSERT INTO dept

VALUES (dept\_id\_seq.nextval, 'Education');

INSERT INTO dept

VALUES (dept\_id\_seq.nextval, 'Administration');

**Ex4**: Tạo một Synonyms cho bảng EMPLOYEES, đặt tên là EMP

CREATE Synonym EMP

FOR employees;

**3.3. Lecture 4**

**Ex1**: Bộ phận nhân sự cần một truy vấn để hiển thị tất cả các mã công việc không bị lặp từ bảng EMPOYEES.

SELECT DISTINCT job\_id

FROM employees;

**Ex2**: Bộ phận nhân sự đã yêu cầu một báo cáo của tất cả nhân viên và jobs\_ID của họ. Hiển thị last\_name cùng với jobs\_ID (được phân tách bằng dấu phẩy và dấu cách) và đặt tên cho cột Employee and Title.

SELECT last\_name || ', ' || job\_id AS "Employee and Title"

FROM employees;

**Ex3**: Phòng nhân sự cần các tìm nhân viên lương theo mức lương. Hiển thị tên last\_name và salary của nhân viên kiếm được từ 5.000$ đến 12.000$ và có mã phòng là 20 hoặc 50. Tên cột tương ứng là Employee and Monthly Salary.

SELECT last\_name Employee,

salary

FROM employees

WHERE (department\_id = 20 OR department\_id = 50)

AND (salary BETWEEN 5000 AND 12000);

**Ex4**: Tạo một báo cáo để hiển thị tên,lương và hoa hồng của tất cả các nhân viên. Sắp xếp dữ liệu theo thứ tự giảm dần của lương và hoa hồng.

SELECT last\_name,

salary,

commission\_pct

FROM employees

WHERE commission\_pct IS NOT NULL

ORDER BY commission\_pct DESC;

**Ex5**: Hiển thị tên của tất cả các nhân viên có cả chữ cái a và e trong họ của họ.

SELECT last\_name

FROM employees

WHERE last\_name LIKE '%a%'

AND last\_name LIKE '%e%';

**Ex6**: Hiển thị tên, vị trí, and lương cho tất cả nhân viên có công việc là SA\_REP hoặc ST\_CLERK và có mức lương không bằng 2.500$, 3.500$ hoặc 7.000$.

SELECT last\_name,

job\_id,

salary

FROM employees

WHERE (job\_id = 'SA\_REP' OR job\_id = 'ST\_CLERK')

AND salary NOT IN (2500, 3500, 7000);

**3.4. Lecture 5**

**Ex1**: Viết một truy vấn hiển thị tên (Chữ cái đầu viết hoa) và độ dài tương ứng cho tất cả nhân viên có tên bắt đầu bằng các chữ cái J, A hoặc M, Sắp xếp kết quả theo tên nhân viên.

SELECT INITCAP(last\_name),

LENGTH(last\_name) length\_name

FROM employees

WHERE last\_name LIKE 'J%'

OR last\_name LIKE 'A%'

OR last\_name LIKE 'M%'

ORDER BY last\_name DESC;

**Ex2**: Bộ phận nhân sự muốn tìm thời gian làm việc của mỗi nhân viên. Đối với mỗi nhân viên, hiển thị tên và tính số tháng giữa ngày hôm nay và ngày mà nhân viên được thuê. Dán nhãn cột MONTHS\_WORKED, Sắp xếp kết quả của bạn theo số tháng làm việc. Làm tròn số tháng cho đến số nguyên gần nhất.

SELECT last\_name,

ROUND(MONTHS\_BETWEEN(sysdate, hire\_date)) Months\_Worked

FROM employees

ORDER BY Months\_Worked;

**Ex3**: Hiển thị mỗi nhân viên tên, họ, ngày thuê và ngày xem xét lương, là ngày thứ Hai đầu tiên sau sáu tháng phục vụ. Dán nhãn cột REVIEW. Định dạng ngày xuất hiện theo định dạng tương tự như “Monday, the Thirty-First of July, 2000”.

SELECT last\_name,

hire\_date,

TO\_CHAR(NEXT\_DAY(ADD\_MONTHS(hire\_date, 6), 'MONDAY'),

'"Monday, the" fmddspth "of" Month"," YYYY') REVIEW

FROM employees;

**Ex4**: Tạo một truy vấn hiển thị tên và hoa hồng của nhân viên. Nếu một nhân viên không kiếm được hoa hồng, hãy hiển thị “No commission”. Đặt tên cột là COMM

SELECT last\_name,

NVL2(commission\_pct,TO\_CHAR(commission\_pct),

'No Comission') COMM

FROM employees;

**Ex5**: Sử dụng hàm DECODE, viết truy vấn hiển thị cấp bậc của tất cả nhân viên dựa trên giá trị của cột JOB\_ID, sử dụng dữ liệu sau:

Cấp bậc công việc:

AD\_PRES A

ST\_MAN B

IT\_PROG C

SA\_REP D

ST\_CLERK E

SELECT job\_id,

DECODE (job\_id, 'AD\_PRES', 'A',

'ST\_MAN', 'B',

'IT\_PROG', 'C',

'SA\_REP', 'D',

'ST\_CLERK', 'E',

0 ) GRADE

FROM employees;

**Ex6**: Tìm mức lương cao nhất, thấp nhất, tổng và trung bình của tất cả nhân viên theo mỗi loại công việc. Dán nhãn các cột tương ứng là MAXIMUM, MINIMUM, SUMMARY và AVERAGE. Làm tròn kết quả của bạn đến số nguyên gần nhất.

SELECT ROUND(MAX(salary)) MAXIMUM,

ROUND(MIN(salary)) MINIMUM,

ROUND(SUM(salary)) SUMMARY,

ROUND(AVG(salary)) AVERAGE

FROM employees

GROUP BY job\_id;

**Ex7**: Xác định số lượng người quản lý mà không liệt kê chúng. Dán nhãn cột Number of Managers. Gợi ý: Sử dụng cột Manager\_ID để xác định số lượng người quản lý.

SELECT COUNT(DISTINCT manager\_id) "Number of Managers"

FROM employees;

**Ex8**: Tạo một báo cáo để hiển thị mã quản lý và mức lương của nhân viên được trả lương thấp nhất thuộc người quản lý đó. Loại trừ các nhận viên không có người quản lý. Không bao gồm bất kỳ nhóm nào có mức lương tối thiểu là 6.000 đô la trở xuống. Sắp xếp đầu ra theo thứ tự giảm dần của tiền lương

SELECT manager\_id, MIN(salary)

FROM employees

WHERE manager\_id IS NOT NULL

GROUP BY manager\_id

HAVING MIN(salary) > 6000

ORDER BY MIN(salary) DESC;

**3.5. Lecture 6**

**Ex1**: Phòng nhân sự cần báo cáo của tất cả nhân viên. Viết một truy vấn để hiển thị tên , số phòng ban và tên bộ phận cho tất cả nhân viên

SELECT e.last\_name,

d.department\_id,

d.department\_name

FROM employees e,

departments d

WHERE e.department\_id = d.department\_id;

**Ex2**: Tạo một báo cáo để hiển thị họ tên nhân viên và số nhân viên cùng với người quản lý của họ. Dán nhãn các cột tương ứng Employee, Emp#, Manager, and Mgr#. Sau đó hiển thị tất cả nhân viên, kể cả King(người không có người quản lý). Sắp xếp kết quả theo số nhân viên.

SELECT e.last\_name "Employee",

e.employee\_id "Emp#",

m.last\_name "Manager",

m.employee\_id "Mgr#"

FROM employees e

LEFT OUTER JOIN employees m

ON (e.manager\_id = m.employee\_id)

ORDER BY e.employee\_id;

**Ex3**: Bộ phận nhân sự cần tìm tên và ngày thuê cho tất cả nhân viên được thuê trước người quản lý của họ, cùng với họ tên của người quản lý và ngày thuê.

SELECT e.last\_name "Employee",

e.hire\_date "Emp Hire Date",

m.last\_name "Manager",

m.hire\_date "Mng Hire Date"

FROM employees e

JOIN employees m

ON (e.manager\_id = m.employee\_id)

WHERE e.hire\_date < m.hire\_date;

**Ex5:** Bộ phận nhân sự cần một báo cáo với các thông số sau:

● Tất cả họ và tên bộ phận của tất cả các nhân viên trong bảng EMPLOYEES

● Tất cả ID phòng ban và tên bộ phận của tất cả các phòng ban từ bảng DEPARTMENTS

SELECT last\_name,

department\_id,

TO\_CHAR(NULL)

FROM employees

UNION

SELECT TO\_CHAR(NULL), department\_id, department\_name

FROM departments;

**Ex6**: Tạo một báo cáo liệt kê ID nhân viên và job\_ID của những nhân viên hiện đang có chức danh công việc giống như chức danh công việc của họ khi ban đầu được công ty thuê (nghĩa là họ đã thay đổi công việc nhưng giờ đã quay lại làm công việc ban đầu của họ).

SELECT employee\_id,

job\_id

FROM employees

INTERSECT

SELECT employee\_id,

job\_id

FROM job\_history;

**Ex7**: Bộ phận nhân sự cần một danh sách các vùng không có bộ phận nào trong đó. Hiển thị ID của vùng và tên của các vùng. Sử dụng bộ toán tử để tạo báo cáo này.

SELECT country\_id,

country\_name

FROM countries

MINUS

SELECT l.country\_id,

c.country\_name

FROM locations l JOIN countries c

ON (l.country\_id = c.country\_id);

**3.6. Lecture 8**

**Ex1**: Viết truy vấn để hiển thị thông tin sau cho những nhân viên mà người quản lý của họ có ID dưới 120:

● Manager ID

● Job\_ID và tổng tiền lương cho mỗi job\_ID cho nhân viên báo thuộc cùng quản lý

● Tổng tiền lương của những người quản lý

● Tổng tiền lương của những người quản lý, không phân biệt job\_ID

SELECT department\_id,

job\_id,

SUM(salary) sum\_salary

FROM employees

WHERE department\_id < 60

GROUP BY ROLLUP(department\_id, job\_id);

**Ex2**: Viết truy vấn bằng hàm GROUPING để xác định xem các giá trị NULL trong các cột tương ứng với các biểu thức GROUP BY có phải do hàm ROLLUP gây ra hay không từ EX1.

SELECT department\_id,

job\_id,

SUM(salary) sum\_salary

FROM employees

WHERE department\_id < 60

GROUP BY CUBE (department\_id, job\_id);

**Ex3**: Viết truy vấn để hiển thị thông tin sau cho những nhân viên mà người quản lý có ID dưới 120:

● Manager ID

● Công việc và tổng tiền lương cho mọi công việc cho nhân viên có cùng một người quản lý

● Tổng tiền lương của những người quản lý

● Các giá trị lập bảng chéo để hiển thị tổng tiền lương cho mọi công việc, không phân biệt quản lý

● Tổng tiền lương không phân biệt tất cả các chức danh công việc.

SELECT department\_id,

job\_id,

manager\_id,

avg(salary) avg\_salary

FROM employees

GROUP BY GROUPING SETS < 129

GROUP BY ROLLUP ( manager\_id, job\_id );

**Ex4**: Sử dụng GROUPING SET, viết truy vấn để hiển thị các nhóm sau:

● department\_id, manager\_id, job\_id

● department\_id, job\_id

● manager\_id, job\_id Truy vấn sẽ tính tổng tiền lương cho mỗi nhóm này

SELECT manager\_id,

job\_id,

SUM(salary) sum\_salary

FROM employees

WHERE manager\_id < 100

GROUP BY CUBE ( manager\_id, job\_id );

**PHẦN 4: BÀI TẬP KẾT THÚC MÔN**

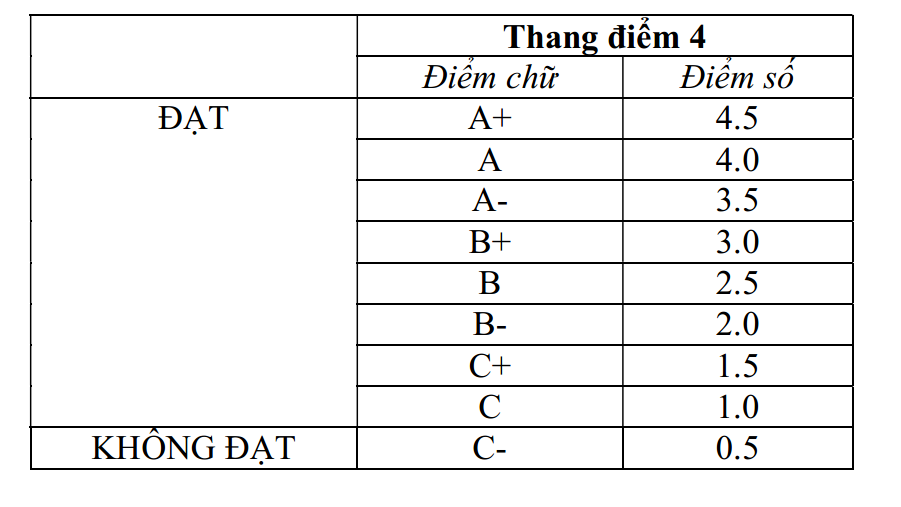
**4.1. Bài tập**

**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:

1. Tích lũy đủ số tín chỉ

2. Điểm phẩy tốt nghiệp không nhỏ hơn 1.0. Biết bảng điểm quy đổi như sau:



**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.

Bảng kết quả trả về gồm các trường: Mã sinh viên, Họ tên sinh viên, Năm học, Kỳ học, Khóa

học, Thời gian học, Phòng học, Giảng viên, Khoa viện.

**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.

Bảng kết quả trả về gồm các trường: Mã sinh viên, Họ tên sinh viên, Năm học, Kỳ học, Khóa

học, Thời gian học, Phòng học, Giảng viên, Khoa viện.

**Bài 4:** Sinh viên A muốn học môn ‘Mobile Computing’ hỏi A cần phải học qua những môn

gì?

**Bài 5:** Cài đặt Trigger kiểm tra số lượng sinh viên đăng ký vượt quá sức chứa của phòng.

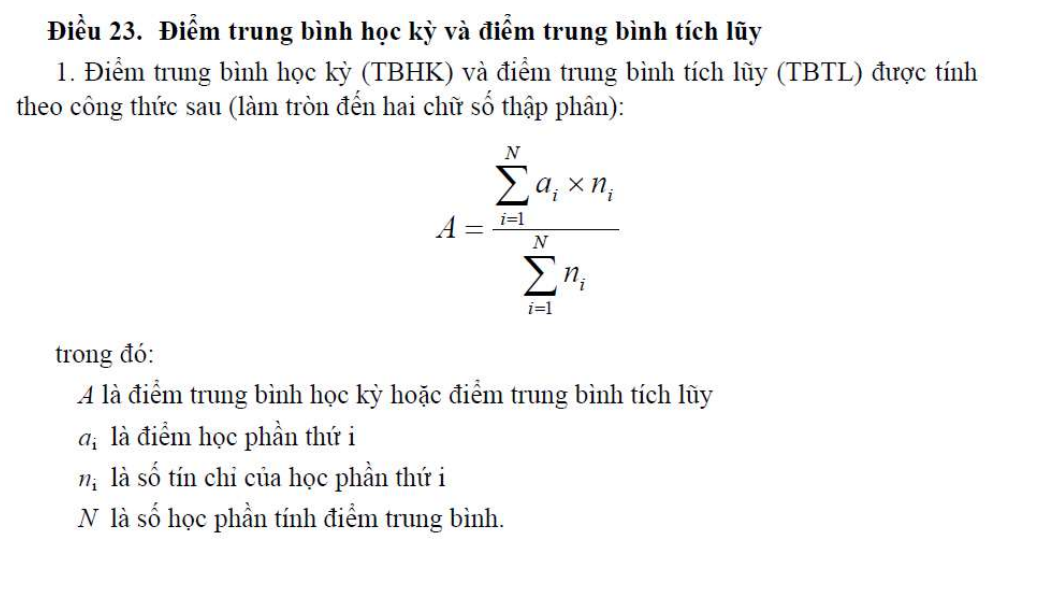
Đưa ra thông báo không thành công khi sinh viên đăng ký môn học. Rollback khi có lỗi xảy ra.

**Bài 6:** Viết thủ tục cho biết kết quả học tập của một sinh viên với:

Đầu vào: Mã sinh viên

Đầu ra: Mã sinh viên, Tên sinh viên, Số tín chỉ tích lũy, Điểm trung bình học kỳ và điểm

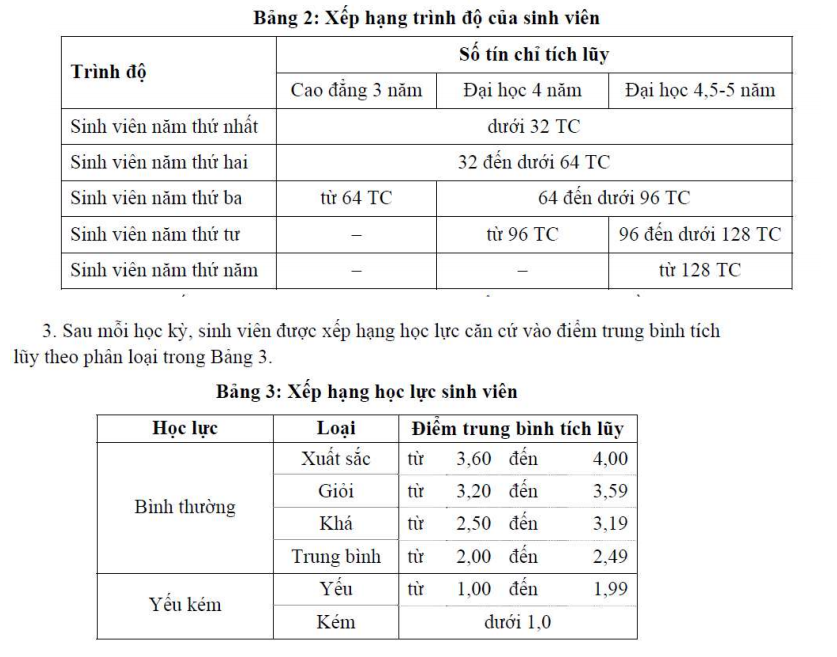
trung bình tích lũy theo từng học kỳ.



**Bài 7:** Viết thủ tục đánh giá kết quả học tập của một sinh viên với:

Đầu vào: Mã sinh viên

Đầu ra: Xếp hạng trình độ sinh viên và xếp hạng học lực của sinh viên, biết rằng:



**Bài 8:** Đánh chỉ mục các bảng takes, student, advisor. So sánh tốc độ truy vấn sau khi đã thực

hiện đánh chỉ mục.

**Bài 9:** Viết thủ tục cho phép sinh viên đăng ký khóa học với lựa chọn phòng và thời gian nào

đó. Cài đặt các TRANSACTION để đảm bảo toàn vẹn dữ liệu và đưa ra thông báo lỗi khi có lỗi

xảy ra.

**4.2. SQL Server**

**4.3. Oracle**

**Bài 1**

**Bước 1:**  Tạo view chuyển điểm từ chữ sang hệ số 4 theo bảng đã cho

*--tạo view chuyển điểm số sang chữ*

CREATE VIEW Diem\_chu AS

SELECT id,

course\_id,

sec\_id,

semester,

year ,

CASE WHEN grade = 'A+' THEN 4.5

WHEN grade = 'A ' THEN 4

WHEN grade = 'A-' THEN 3.5

WHEN grade = 'B+' THEN 3

WHEN grade = 'B ' THEN 2.5

WHEN grade = 'B-' THEN 2

WHEN grade = 'C+' THEN 1.5

WHEN grade = 'C ' THEN 1

WHEN grade = 'C-' THEN 0.5

ELSE 0

END AS point

FROM takes;

*--test*

select \* from takes where id = 10033 order by course\_id

select \* from takes\_number where id = 10033 order by course\_id

select \* from takes\_bester where id = 10033 order by course\_id

**Bước 2:**

*--view hiển thị những môn đạt có điểm cao nhất của sv( bỏ những môn có điểm C- và môn có điểm đạt thấp hơn)*

CREATE VIEW diem\_dat AS

SELECT id,

course\_id,

MAX(point) count

FROM Diem\_chu

Where point != 0.5

GROUP BY id,

course\_id;

**Bước 3:** Thủ tục tính TCTL và CPA để xét xem sinh viên đã đủ điều kiện tốt ngiệp chưa

CREATE OR REPLACE PROCEDURE BAI1(MSSV IN VARCHAR)

AS TCTL NUMBER;

cpa NUMBER;

BEGIN

SELECT sum(c.credits) INTO TCTL

FROM diem\_dat t

JOIN course c ON t.id = MSSV

AND t.course\_id = c.course\_id;

IF TCTL > 128 THEN

SELECT SUM(t.count \* c.credits) / SUM(c.credits)

INTO cpa

FROM diem\_dat t

JOIN course c ON t.id = MSSV

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('Đa du dieu kien tot nghiep, diem trung binh ' || cpa);

END IF;

elsif 0 < TCTL and TCTL < 128

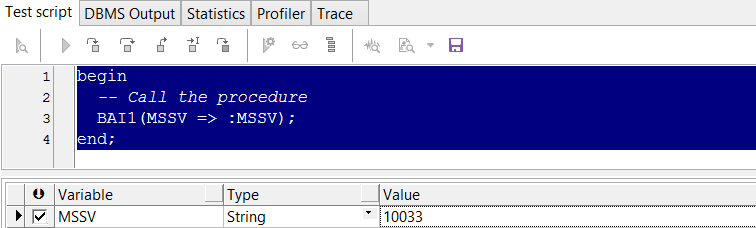
then DBMS\_OUTPUT.PUT\_LINE('Khong du dieu kien tot nghiep, da dat duoc ' || TCTL || ' tin');

ELSE DBMS\_OUTPUT.PUT\_LINE('Khong ton tai ma sinh vien nay: ' || MSSV);

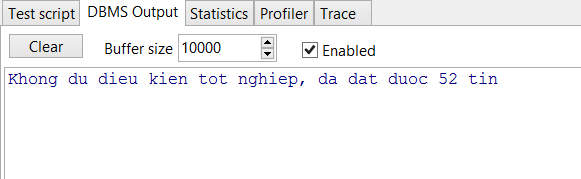
END IF;

END;

Chạy thử với id = 10033



Kết quả hiển thị



**Bài 2**

**Bước 1: Tạo view lấy các thông tin ta cần hiển thị:** Mã sinh viên, Họ tên sinh viên, Năm học, Kỳ học, Khóa học, Thời gian học, Phòng học, Giảng viên, Khoa viện.

*--Bài 2*

CREATE OR REPLACE VIEW Thong\_tin\_bai2 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;

*--thutuc*

CREATE OR REPLACE PROCEDURE SP\_LOC\_DU\_LIEU\_BAI2(truong IN

VARCHAR2, dulieu IN VARCHAR, mycursor OUT SYS\_REFCURSOR)

AS str\_query VARCHAR (1000 );

BEGIN

str\_query := 'SELECT \* FROM Thong\_tin\_bai2 WHERE ' || truong || '

LIKE ''%' || dulieu || '%''';

OPEN mycursor FOR str\_query;

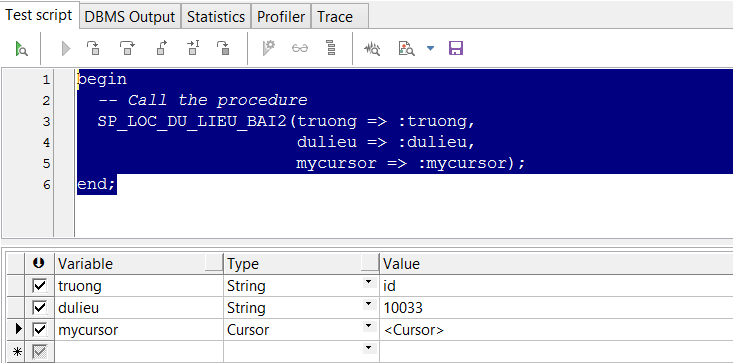
EXCEPTION

WHEN OTHERS THEN

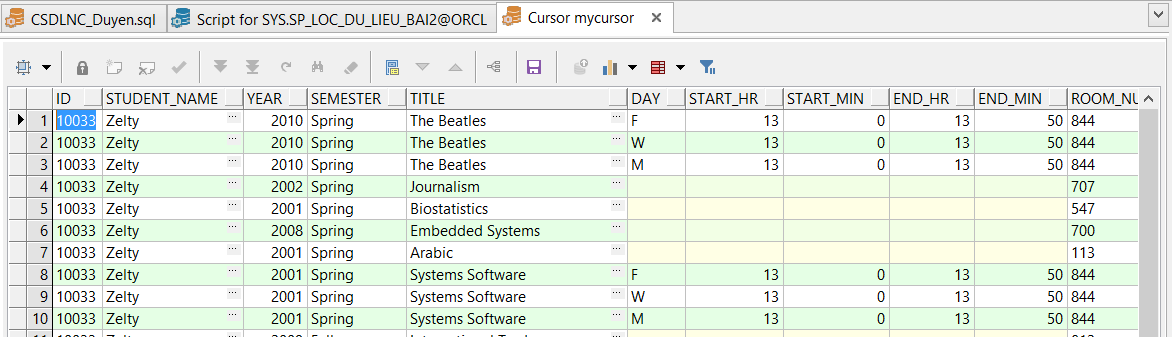
DBMS\_OUTPUT.PUT\_LINE('Du lieu khong ton tai: ' || truong);

END;

Chạy chương trình với trường: id, giá trị: 10033

****

**Kết quả:**

****

**Bài 3**

**Bước 1: Tạo kiểu dữ liệu bảng gồm 2 trường: tên trường và một giá trị của trường để input.**

CREATE OR REPLACE TYPE my\_object

AS OBJECT(field\_name VARCHAR(50),

field\_value VARCHAR(50));

CREATE TYPE my\_table

AS TABLE OF my\_object;

**Bước 2: Tạo thủ tục lọc dữ liệu theo kiểu bảng sử dụng lại view ở câu 2:**

CREATE OR REPLACE PROCEDURE SP\_LOC\_DU\_LIEU\_BAI3 (mytable IN

my\_table, mycursor OUT SYS\_REFCURSOR)

AS str\_query VARCHAR (1000);

BEGIN

str\_query := 'SELECT \* FROM Thong\_tin\_bai2 WHERE 1=1';

FOR indx IN mytable.FIRST .. mytable.LAST LOOP

str\_query := str\_query || ' AND ' ||

mytable(indx).field\_name ||' LIKE ''%' ||

mytable(indx).field\_value || '%'' ';

END LOOP;

OPEN mycursor FOR str\_query;

EXCEPTION

WHEN OTHERS THEN

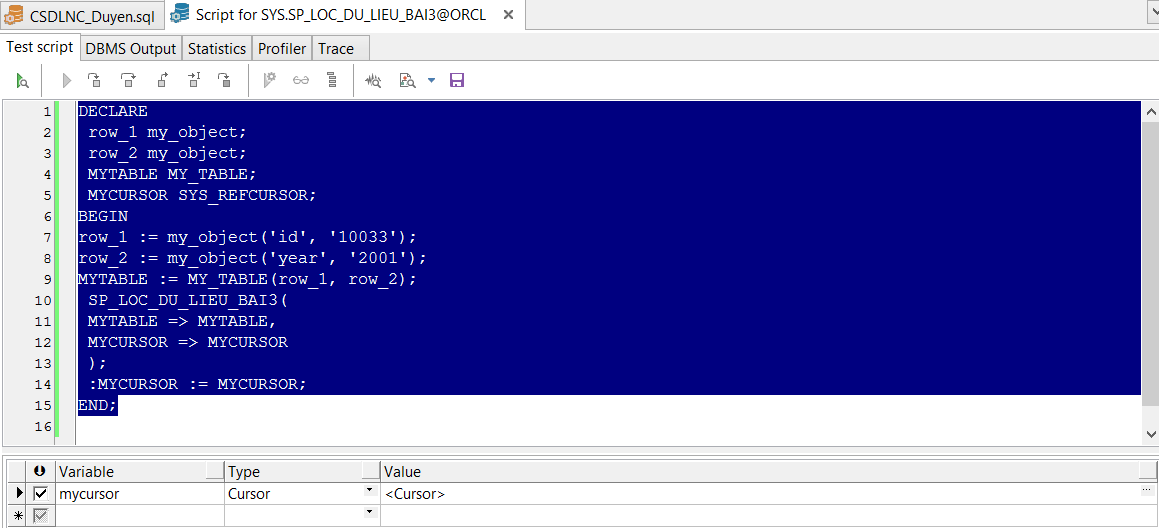
DBMS\_OUTPUT.PUT\_LINE('Điều kiện đầu vào không đúng!');

END;

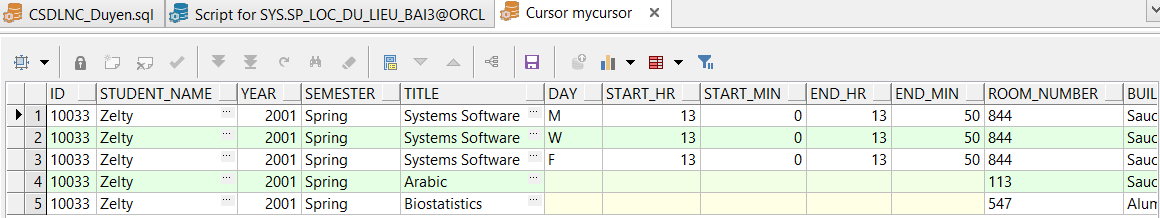
**Test thủ tục với trường dữ liệu và giá trị tương ứng :**

Id: 10033

Year: 2001

****

**Kết quả:**

****

**Bài 4:**

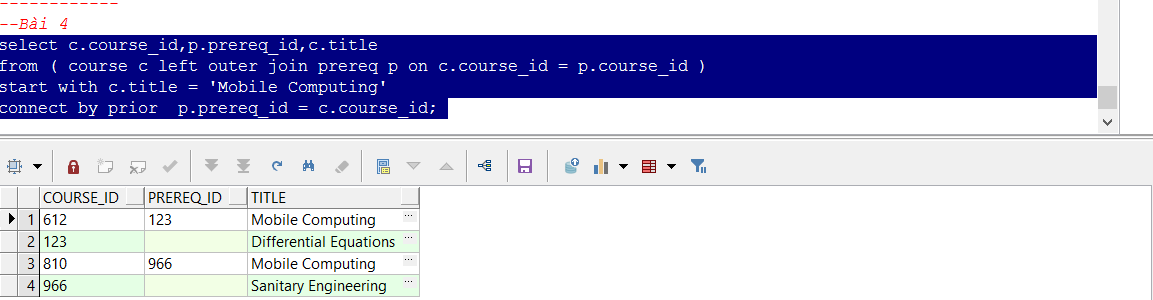
Để truy vấn những môn cần học trước “Mobile Computing” ta sử dụng truy vấn phân cấp.

select c.course\_id,p.prereq\_id,c.title

from ( course c full outer join prereq p on c.course\_id = p.course\_id )

start with c.title = 'Mobile Computing'

connect by prior p.prereq\_id = c.course\_id;



**Bài 5:**

Ta tạo Before Trigger để kiểm tra các trường hợp có thể xảy ra khi đăng kí học:

* Nếu sinh viên đã đk học: báo lỗi 'sinh vien da dang ky'
* Quá số lượng đăng kí lớp: báo lỗi 'lop da day'
* Nếu không thuộc 2 trường hợp trên thì in ra đăng kí thành công và cập nhật thông tin sv vào bảng takes

CREATE OR REPLACE TRIGGER dang\_ki\_hoc

BEFORE INSERT ON takes FOR each ROW

DECLARE

so\_sv\_dk NUMBER;

so\_sv\_max NUMBER;

kiem\_tra NUMBER;

BEGIN

SELECT COUNT(\*) INTO kiem\_tra

WHERE (t.ID = :NEW.ID)

AND (t.COURSE\_ID =:NEW.COURSE\_ID)

AND (t.SEC\_ID = :NEW.SEC\_ID)

AND (t.SEMESTER = :NEW.SEMESTER)

AND (t.YEAR = :NEW.YEAR);

IF (kiem\_tra >0) THEN

RAISE\_APPLICATION\_ERROR(-20191,'sinh vien da dang ky. ');

ELSE

SELECT c.CAPACITY INTO so\_sv\_max

FROM SECTION s

JOIN CLASSROOM c using (building, room\_number)

WHERE (s.COURSE\_ID = :NEW.COURSE\_ID)

AND (s.SEC\_ID = :NEW.SEC\_ID)

AND (s.SEMESTER = :NEW.SEMESTER)

AND (s.YEAR = :NEW.YEAR);

SELECT COUNT(\*) INTO so\_sv\_dk

FROM TAKES t

WHERE (t.COURSE\_ID = :NEW.COURSE\_ID)

AND (t.SEC\_ID = :NEW.SEC\_ID)

AND (t.SEMESTER = :NEW.SEMESTER)

AND (t.YEAR = :NEW.YEAR);

IF(so\_sv\_dk +1 < so\_sv\_max) THEN

DBMS\_OUTPUT.PUT\_LINE('DK thanh cong ');

ELSE

RAISE\_APPLICATION\_ERROR(-20192,'lop da day' || so\_sv\_dk|| '/' ||

so\_sv\_max);

END IF;

END IF;

END;

**Bài 6:** -- Tạo view conver học kì:

CREATE OR REPLACE VIEW CONVER\_HOC\_KI AS

SELECT id,

course\_id,

sec\_id,

semester,

year ,

point,

CASE

WHEN semester = 'Fall' THEN CONCAT( year, '2' )

WHEN semester = 'Spring' THEN CONCAT( year, '1')

END AS semester\_number

FROM diem\_chu;

*-- Tạo view TONG\_HOP\_DIEM*

create or replace view tong\_hop\_diem as

select hk.id, st.name, hk.course\_id, hk.Sec\_id, hk.semester, hk.year, hk.point,

c.credits, hk.semester\_number

from conver\_hoc\_ki hk

join course c on c.course\_id = hk.course\_id

join student st on st.id = hk.id;

*-- Tạo hàm tính số tín chỉ tích lũy.*

create or replace function func\_tich\_luy(student\_id NUMBER, semester\_max

Number)

return float is FunctionResult float;

begin

select sum(pn.credits) into FunctionResult

from

(select th.id, th.name, th.course\_id, th.credits, max(point) as max\_point

from Tong\_hop\_diem th

where TO\_NUMBER(th.semester\_number) <=

TO\_NUMBER(semester\_max)

group by th.course\_id, th.name, th.credits, th.id) pn

where pn.id = student\_id

and pn.max\_point >= 1

group by pn.id, pn.name;

return (FunctionResult);

end func\_tich\_luy;

*-- Tạo hàm tính số tín chỉ tích lũy.*

create or replace function func\_tich\_luy(student\_id NUMBER, semester\_max

Number)

return float is FunctionResult float;

begin

select sum(pn.credits) into FunctionResult

from

(select th.id, th.name, th.course\_id, th.credits, max(point) as max\_point

from Tong\_hop\_diem th

where TO\_NUMBER(th.semester\_number) <=

TO\_NUMBER(semester\_max)

group by th.course\_id, th.name, th.credits, th.id) pn

where pn.id = student\_id

and pn.max\_point >= 1

group by pn.id, pn.name;

return (FunctionResult);

end func\_tich\_luy;

*--Hàm tính CPA*

create or replace function func\_cpa(student\_id NUMBER, semester\_max

Number)

return float is FunctionResult float;

begin

select round(sum(pn.max\_point\*pn.credits)/sum(pn.credits), 2) into

FunctionResult

from

(select th.id, th.name, th.course\_id, th.credits, max(point) as

max\_point

from Tong\_hop\_diem th

where TO\_NUMBER(th.semester\_number) <=

TO\_NUMBER(semester\_max)

group by th.course\_id, th.name, th.credits, th.id) pn

where pn.id = student\_id

group by pn.id, pn.name;

return (FunctionResult);

end func\_cpa;

*-- Tạo thủ tục show kết quả học tập của học sinh theo từng kì.*

CREATE OR REPLACE PROCEDURE

SP\_SHOW\_RESULT\_STUDENT\_BAI6 ( student\_id NUMBER, mycursor OUT

sys\_refcursor )

IS

BEGIN

OPEN mycursor FOR

select

th.id,

th.name,

th.semester,

th.year,

th.semester\_number,

round(sum(th.point\*th.credits)/sum(th.credits) , 2) as gpa,

sum(th.credits),

Func\_CPA(student\_id,th.semester\_number) as cpa,

Func\_tich\_luy(student\_id,th.semester\_number) as Tong\_tich\_luy

from Tong\_hop\_diem th

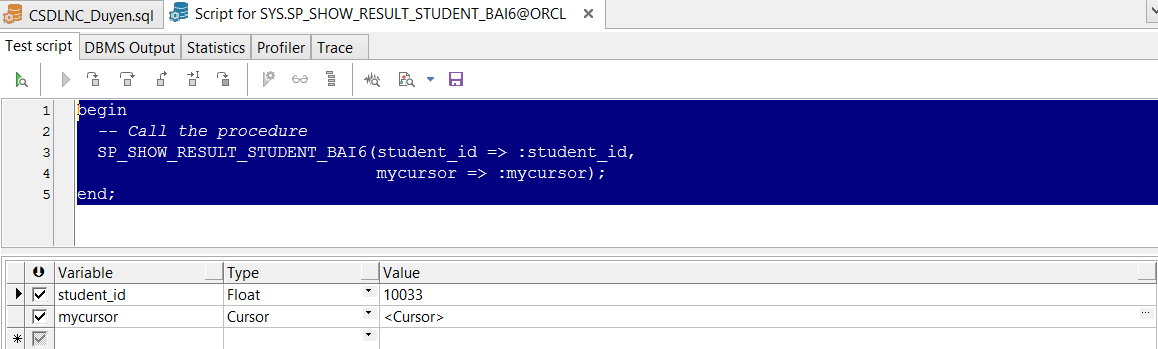
where id = student\_id group by th.semester\_number, th.name, th.semester,

th.year, th.id

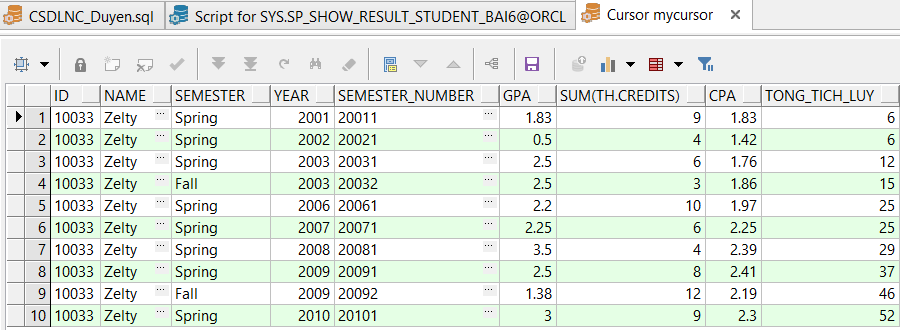
ORDER BY th.semester\_number ASC;

END;

Kiếm tra thủ tục với id = 10033.



Kết quả trả về:



**Bài 7**

*-- Tạo function chuyển đổi trình độ của sinh viên*

CREATE OR REPLACE FUNCTION func\_trinh\_do\_sv ( no\_credits NUMBER )

RETURN VARCHAR2

IS

grade\_point VARCHAR2 ( 20 ) := 'SV nam nhat';

BEGIN

grade\_point :=

CASE

WHEN no\_credits < 32 THEN 'SV nam nhat'

WHEN no\_credits BETWEEN 32 AND 63 THEN 'SV nam hai'

WHEN no\_credits BETWEEN 64 AND 95 THEN 'SV nam ba'

WHEN no\_credits BETWEEN 96 AND 127 THEN 'SV nam bon'

WHEN no\_credits >= 128 THEN 'SV nam nam'

ELSE 'Loi chuyen doi'

END;

RETURN grade\_point;

EXCEPTION

WHEN others THEN

dbms\_output.put\_line ( 'ERR: ' || SQLERRM );

END;

*-- Function chuyển đổi học lực của sinh viên:*

CREATE OR REPLACE FUNCTION func\_hoc\_luc\_sv ( avg\_credit\_point NUMBER )

RETURN VARCHAR2

IS avg\_point NUMBER := 0;

grade\_point VARCHAR2 ( 20 ) := '';

BEGIN

avg\_point := round( avg\_credit\_point, 2 );

grade\_point :=

CASE

WHEN 3.6 <= avg\_point THEN 'Xuat xac'

WHEN 3.2 <= avg\_point AND avg\_point < 3.6 THEN 'gioi'

WHEN 2.5 <= avg\_point AND avg\_point < 3.2 THEN 'kha'

WHEN 2.0 <= avg\_point AND avg\_point < 2.5 THEN 'trung binh'

WHEN 1.0 <= avg\_point AND avg\_point < 2.0 THEN 'yeu'

WHEN avg\_point < 1.0 THEN 'Kem'

ELSE 'Loi chuyen doi!'

END;

RETURN grade\_point;

EXCEPTION

WHEN others THEN

dbms\_output.put\_line ( 'Err: ' || SQLERRM );

END;

*-- Tạo store proceduce kiểm tra:*

CREATE OR REPLACE PROCEDURE SP\_SHOW\_RESULT\_STUDENT\_BAI7 (

student\_id NUMBER, mycursor OUT sys\_refcursor )

IS

BEGIN

OPEN mycursor FOR

select

th.id,

th.name,

th.semester,

th.year,

th.semester\_number,

round(sum(th.point\*th.credits)/sum(th.credits) , 2) as gpa,

sum(th.credits),

Func\_CPA(student\_id,th.semester\_number) as cpa,

Func\_tich\_luy(student\_id,th.semester\_number) as Tong\_tich\_luy,

func\_trinh\_do\_sv(Func\_tich\_luy(student\_id,th.semester\_number)),

func\_hoc\_luc\_sv(Func\_CPA(student\_id,th.semester\_number))

from Tong\_hop\_diem th

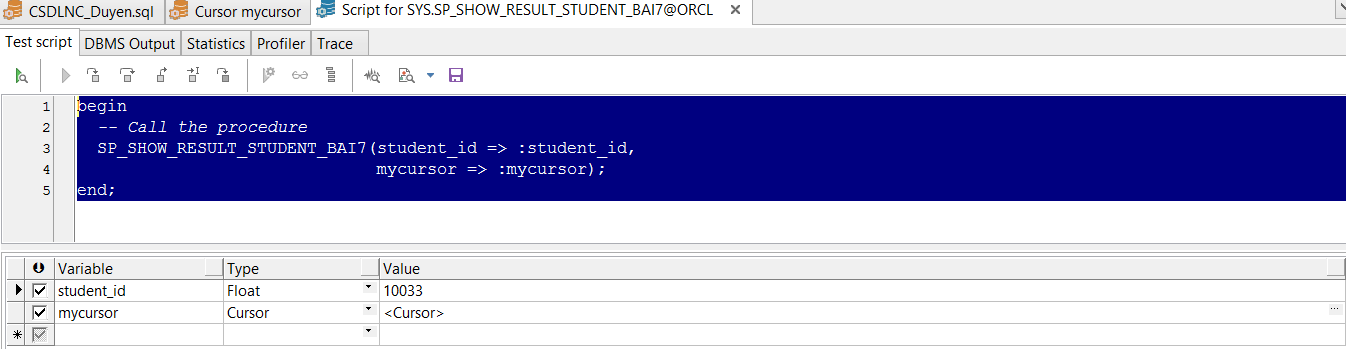
where id = student\_id group by th.semester\_number, th.name, th.semester, th.year,

th.id

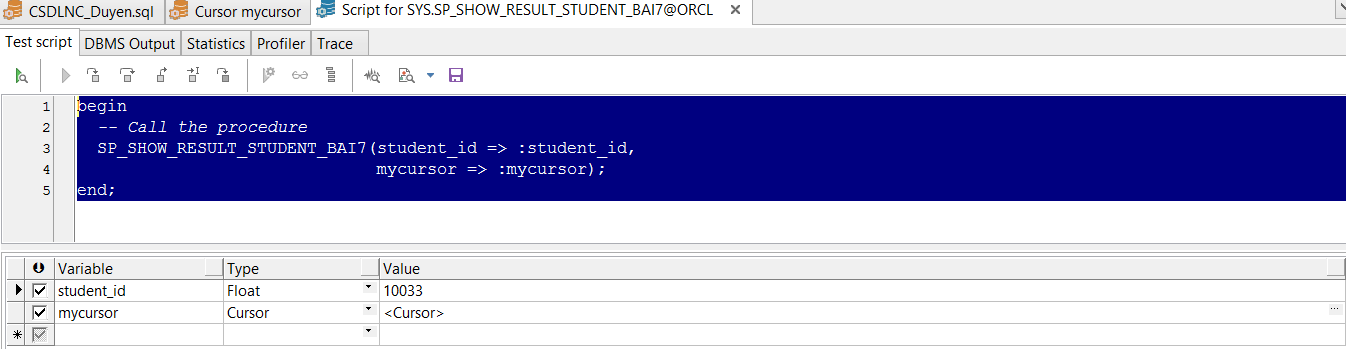
ORDER BY th.semester\_number ASC;

END;

Chạy thủ tục với id = 10033



Ta có kết quả:



Câu 8:

CREATE INDEX take\_index ON takes(grade);

CREATE INDEX student\_index ON student(name, dept\_name, tot\_cred);

CREATE INDEX advisor\_index ON advisor(i\_id);

Cả 3 bảng điều khá ít dữ liệu, nên việc index không cải thiện tốc độ truy vấn nhiều.