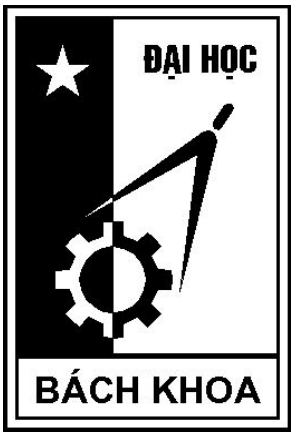
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC



TIỂU LUẬN CƠ SỞ DỮ LIỆU

NÂNG CAO

Sinh viên thực hiện : Nguyễn Thế Hùng

Mã số sinh viên : 20142102

Lớp : Toán Tin 02 – K59

Hà Nội 2019

Mục lục

Lời mở đầu

Trong những năm gần đây, khái niệm "dữ liệu lớn" (Big Data) ra đời, kéo theo sự phát triển về khoa học dữ liệu nhằm đáp ứng các nhu cầu cho việc kinh doanh, nghiên cứu khoa học,... Các công nghệ về khai phá dữ liệu lớn được ra đời nhằm đã và đang trở thành nhu cầu tất yếu của xã hội.

Học phần “Cơ sở dữ liệu nâng cao” cung cấp cho sinh viên kiến thức nền tảng trong việc xây dựng và xử lý dữ liệu lớn. Trong báo này, em xin được trình bày một số kiến thức về xử lý dữ liệu lớn. Báo cáo gồm 4 chương chính:

- Chương 1: Cơ sở dữ liệu lớn
- Chương 2: Cơ sở dữ liệu phân tán
- Chương 3: Hệ quản trị cơ sở dữ liệu Oracle
- Chương 4: Bài tập kết thúc môn

Do kiến thức, thời gian và kinh nghiệm còn hạn chế nên không tránh khỏi những thiếu sót. Vì vậy em rất mong nhận được sự cảm thông và góp ý của thầy cô để bài báo cáo có thể hoàn thiện hơn.

Em xin chân thành cảm ơn!

Chương 1: Cơ sở dữ liệu lớn

1.1. Các khái niệm

Big Data là thuật ngữ dùng để chỉ một tập hợp dữ liệu rất lớn và rất phức tạp đến nỗi những công cụ, ứng dụng xử lý dữ liệu truyền thống không thể nào đảm đương được. Kích thước dữ liệu lớn ngày càng tăng theo thời gian. 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

Big Data được mô tả bởi những đặc trưng sau:

Volume (Dung lượng): Số lượng dữ liệu được tạo ra và lưu trữ. Kích thước của dữ liệu xác định giá trị và tiềm năng insight- và liệu nó có thể thực sự được coi là dữ liệu lớn hay không.

Variety (Tính đa dạng): Các dạng và kiểu của dữ liệu. Dữ liệu được thu thập từ nhiều nguồn khác nhau và các kiểu dữ liệu cũng có rất nhiều cấu trúc khác nhau.

Velocity (Vận tốc): Trong trường hợp này nghĩa là tốc độ các dữ liệu được tạo ra và xử lý để đáp ứng các nhu cầu và thách thức trên con đường tăng trưởng và phát triển.

Veracity (Tính xác thực): Chất lượng của dữ liệu thu được có thể khác nhau rất nhiều, ảnh hưởng đến sự phân tích chính xác.

Nhà máy và các hệ thống không thực-ảo có thể có một hệ thống 6C bao gồm:

- Kết nối (cảm biến và mạng)
- Đám mây (tính toán và dữ liệu theo yêu cầu)
- Nội dung ảo (mẫu và bộ nhớ)
- Nội dung / ngữ cảnh (ý nghĩa và tương quan)
- Cộng đồng (chia sẻ và cộng tác)
- Tùy chỉnh (cá nhân hoá và giá trị)

Dữ liệu phải được xử lý bằng các công cụ tiên tiến (phân tích và thuật toán) để cho ra các thông tin có ý nghĩa. Ví dụ, để quản lý một nhà máy phải xem xét cả hai vấn đề hữu hình và vô hình với các thành phần khác nhau. Các thuật toán tạo thông tin phải phát hiện và giải quyết các vấn đề không nhìn thấy được như sự xuống cấp của máy, mài mòn linh kiện,... trong nhà máy.

1.3. Ứng dụng

Cơ sở dữ liệu lớn được ứng dụng trong các lĩnh vực như:

- Quản lý chính phủ
- Chăm sóc sức khỏe
- Sản xuất
- Giáo dục

- Truyền thông
- IoT
- Công nghệ
- ...

1.4. Một số khái niệm khác

Dữ liệu giao dịch/tác nghiệp (Online Transaction Processing - OLTP): là hệ thống được sử dụng nhằm mục đích kiểm soát và thực thi các nghiệp vụ cơ bản, thiết yếu xảy ra hàng ngày, là một phần quan trọng không thể thiếu trong quá trình tác nghiệp của tổ chức.

Dữ liệu phân tích (Online Analysis Processing - OLAP): là hệ thống xây dựng nhằm giúp nhà quản lý lập kế hoạch, giải quyết vấn đề và ra các quyết định liên quan của tổ chức.

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 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) và non-relational (không ràng buộc).

Kho dữ liệu (Data Warehouse): là một tập hợp dữ liệu hướng chủ đề, tích hợp, có tính thời gian và không thay đổi để hỗ trợ quá trình tạo quyết định quản lý.

Data Mart: là một phiên bản thu nhỏ của kho dữ liệu tập trung vào một chủ thể nhất định.

Chương 2: Cơ sở dữ liệu phân tán

2.1. Định nghĩa

Cơ sở dữ liệu phân tán (Distributed Database - DDB): là một tuyển tập dữ liệu có quan hệ logic với nhau, được phân bố trên các 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 cung cấp các cơ chế truy xuất đảm bảo tính trong suốt (transparent) về sự phân tán đối với người dùng.Tính trong suốt của hệ phân tán được thể hiện như sau:

- Tính trong suốt phân đoạn (fragmentation transparency): mức cao nhất của tính trong suốt, người dùng cuối và người lập trình không cần biết về sự phân tán của cơ sở dữ liệu (không biết tên đoạn và vị trí phân bố các đoạn).
- Tính trong suốt định vị (location transparency): người dùng cuối hoặc người lập trình biết cơ sở dữ liệu phân chia thành các đoạn, tên của các đoạn nhưng không biết vị trí phân bố của các đoạn.
- Tính trong suốt đối với ánh xạ địa phương (local mapping transparency): người dùng cuối hoặc người lập trình biết tên các đoạn và vị trí của các đoạn.

Hệ cơ sở dữ liệu phân tán (Distributed Database System - DDBS) bao gồm: DDB và D_DBMS (Distributed Database Management System).

Hệ cơ sở dữ liệu phân tán được chia ra làm 2 loại ứng dụng:

- Ứng dụng cục bộ: được yêu cầu và thực hiện trên máy tính ở một nút trong hệ CSDLPT và chỉ liên quan đến CSDL tại nút đó.
- Ứng dụng toàn cục: yêu cầu truy nhập dữ liệu ở nhiều nút thông qua hệ thống truyền thông.

2.2. Đánh giá CSDL 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.
- Nâng cao tính tin cậy.
- Nâng cao hiệu năng.
- Dễ mở rộng.

Hạn chế:

- Thiết kế CSDL phức tạp hơn.
- Khó điều khiển tính nhất quán dữ liệu.
- Khó phát hiện và khử lỗi.
- Giá thành.
- Bảo mật.
- Thiếu chuẩn mực.

- Thiếu kinh nghiệm.

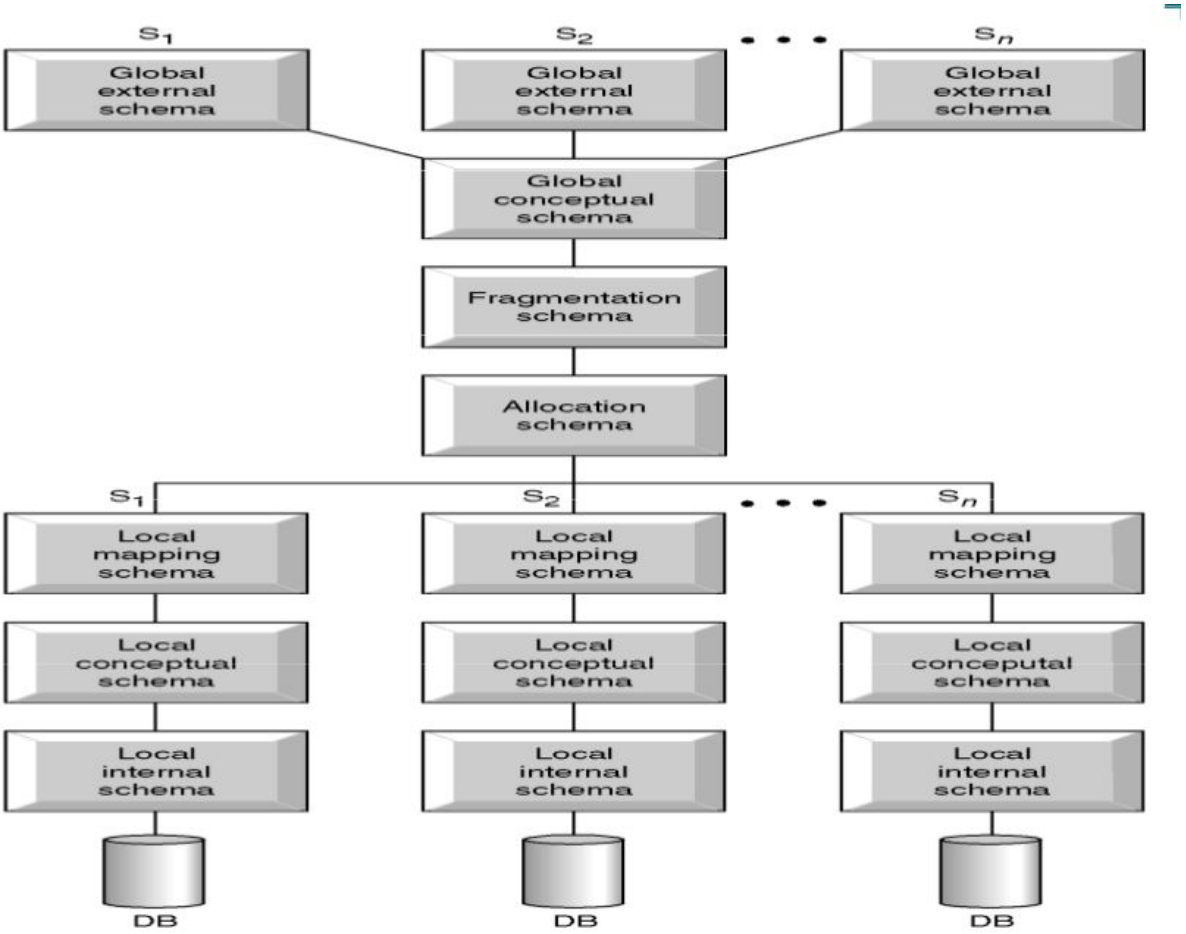
2.3. Kiến trúc của CSDL phân tán

Do sự đa dạng, không có kiến trúc nào được công nhận tương đương với kiến trúc 3 mức ANSI/SPARC, vì vậy khi thiết kế kiến trúc CSDL phân tán ta có thể lược bỏ một số yếu tố sau:

- Có thể khuyết một số mức, phụ thuộc vào các mức trong suốt được cung cấp.
- Có thể thuần nhất hoặc (homogeneous) hoặc hỗn tạp (heterogeneous).

Một kiến trúc tham khảo:

- Tập các lược đồ ngoài toàn cục (Global external schemas).
- Lược đồ khái niệm toàn cục (Global conceptual schema −GCS).
- Lược đồ phân đoạn (Fragmentation schema) và lược đồ định vị (allocation schema).
- Tập các lược đồ cho mỗi hệ CSDL cục bộ tuân theo tiêu chuẩn 3 mức ANSI/SPARC .



Hình 2.1: Ví dụ về kiến trúc CSDL phân tán

2.4. Thiết kế CSDL phân tán

Quan hệ: CSDL gồm một tập các quan hệ, trong đó các khung nhìn là các tập con của một quan hệ. Các quan hệ cần nhiều chi phí truyền thông.

Đoạn/mảnh (fragment) của quan hệ (sub-relations): Thi hành tương tranh nhiều giao tác truy xuất trên các phần khác nhau của quan hệ, cùng với đó là điều khiển ngữ nghĩa dữ liệu (đặc biệt là các ràng buộc toàn vẹn) khó hơn. Đặc biệt với những khung nhìn không được định nghĩa trên một mảnh/đoạn riêng lẻ thì sẽ đòi hỏi nhiều xử lý hơn

Phân đoạn dữ liệu là 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 khúc hay mảnh. Thông tin phân đoạn dữ liệu 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.

Các loại phân đoạn:

- 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 chiếu để phân đoạn.
- Phân đoạn ngang suy diễn: Dùng phép nửa nối để phân đoạn.

Trong mọi loại phân đoạn, một đoạn được xác định bởi một biểu thức quan hệ, trong đó toán hạng là các quan hệ tổng thể, kết quả là các đoạn.

Các quy tắc phân đoạn:

- Điều kiện phân đoạn đầy đủ (Completeness condition)
- Điều kiện khôi phục phân đoạn (Reconstruction condition).
- Điều kiện không giao nhau (Disjointness condition) với phân đoạn ngang.

Mục tiêu của phân đoạn:

- Tăng tính cục bộ.
- Nâng cao độ tin cậy và tính sẵn sàng.
- Nâng cao hiệu năng.
- Cân bằng khả năng lưu trữ và chi phí.
- Tối thiểu chi phí truyền thông.

Chương 3: Hệ quản trị cơ sở dữ liệu Oracle

3.1. 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 );
```

Chương 4: Bài tập kết thúc môn

4.1. Bài 1

Đề bài:
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 lũy 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, A+	4
	A-	3.5
	B, B+	3
	B-	2.5
	C, C+	2
KHÔNG ĐẠT	C-	1

Bài làm:

Bước 1: Tạo 1 view convert từ điểm chữ sang điểm hệ số 4:

```
CREATE VIEW takes_number AS
SELECT id,
      course_id,
      sec_id,
      semester,
      year ,
      CASE
         WHEN grade = 'A ' OR grade = 'A+' THEN 4
         WHEN grade = 'A-' THEN 3.5
         WHEN grade = 'B ' OR grade = 'B+' THEN 3
         WHEN grade = 'B-' THEN 2.5
         WHEN grade = 'C ' OR grade = 'C+' THEN 2
         WHEN grade = 'C-' THEN 1
         ELSE 0
      END AS point
```

```
FROM takes;
```

Bước 2: Tạo view lấy điểm cao nhất của sinh viên của một môn học:

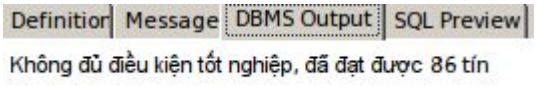
```
CREATE VIEW takes_bester AS
SELECT id,
      course_id,
      MAX(point) count
FROM takes_number
GROUP BY id,
      course_id;
```

Bước 3: Tạo store procedure kiểm tra điều kiện tốt nghiệp của sinh viên:

```
CREATE OR REPLACE PROCEDURE SP_BAI1(student_id IN VARCHAR)
AS tot_cred NUMBER;
       cpa NUMBER;
BEGIN
      SELECT tot_cred INTO tot_cred
      FROM student
      WHERE ID = student_id;
      IF tot_cred > 128 THEN
            SELECT SUM(t.point * c.credits) / SUM(c.credits)
                  INTO cpa
            FROM takes_bester 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('Không đủ điều kiện tốt
nghiệp, điểm trung bình ' || cpa);
            ELSE DBMS_OUTPUT.PUT_LINE('Đã đủ điều kiện tốt nghiệp,
điểm trung bình ' || cpa);
            END IF;
      ELSE DBMS_OUTPUT.PUT_LINE('Không đủ điều kiện tốt nghiệp, đã
đạt được ' || tot_cred || ' tín');
      END IF;
  EXCEPTION
      WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('Mã sinh viên không đúng: ' ||
student_id);
END;
```

Kết quả:

Với mã sinh viên 123



Bình luận:

```
SELECT s.ID,
      p.TOTAL_CRED Tong_tin_chi_thuc_hoc,
      s.TOT_CRED Tong_tin_chi_tich_luy
FROM
      (SELECT SUM(c.CREDITS) total_cred,
              t.ID
      FROM TAKES_NUMBER t
      JOIN COURSE c ON c.COURSE_ID = t.COURSE_ID
      GROUP BY t.ID) p
JOIN STUDENT s
ON p.ID = s.ID;
```

Sau khi thực hiện truy vấn trên, em phát hiện ra là một số sinh viên có tổng số tín chỉ thực học nhỏ hơn tổng số tín chỉ tích lũy do vậy rất khó để xác định sinh viên này đủ điều kiện tốt nghiệp hay chưa.

ID	TONG_TIN_CHI_THUC_HOC	TONG_TIN_CHI_TICH_LUY
94836	38	13
49073	57	74
33460	39	48
792	64	102
52945	33	13
43032	71	84
17924	51	97

4.2. Bài 2

Đề bài:

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.

Bài làm:

Bước 1: Tạo view để lấy dữ liệu theo yêu cầu:

```
CREATE OR REPLACE VIEW schedule AS
SELECT st.id,
      st.name studnet_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: Viết thủ tục lấy dữ liệu sử dụng execute string trong Oracle:

```
CREATE OR REPLACE PROCEDURE SP_LOC_DU_LIEU_BAI2(field_name IN
VARCHAR2, field_value IN VARCHAR, mycursor OUT SYS_REFCURSOR)
AS str_query VARCHAR (1000 );
BEGIN
   str_query := 'SELECT * FROM SCHEDULE WHERE ' || field_name || '
LIKE ''%' || field_value || '%''';
     OPEN mycursor FOR str_query;
  EXCEPTION
    WHEN OTHERS THEN
     DBMS_OUTPUT.PUT_LINE('Trường không tồn tại: ' || field_name);
END;
```

Kết quả:

Lọc với sinh viên mã số 12326

ID	STUDNET_...	YEAR	SEMESTER	TITLE	DAY	START_HR	START_MIN	END_HR	END_MIN	ROOM_NU...	BUILDING	INSTRUCT...	DEPT_NAME
12326	Watson	2001	Spring	Biostatistics						547	Alumni	Mingoz	Finance
12326	Watson	2002	Spring	Manufactu...						145	Fairchild	Mingoz	Finance
12326	Watson	2010	Fall	Environme...						145	Fairchild	Lembr	Finance
12326	Watson	2008	Spring	World Hist...	R	14	30	15	45	707	Gates	Jaekel	Finance
12326	Watson	2008	Spring	World Hist...	T	14	30	15	45	707	Gates	Jaekel	Finance
12326	Watson	2003	Fall	Astronautics	W	10	0	12	30	808	Polya	Ullman	Finance
12326	Watson	2003	Spring	Bankruptcy						183	Taylor	Ullman	Finance
12326	Watson	2002	Fall	The Music...	F	13	0	13	50	375	Chandler	Ullman	Finance
12326	Watson	2002	Fall	The Music...	W	13	0	13	50	375	Chandler	Ullman	Finance
12326	Watson	2002	Fall	The Music...	M	13	0	13	50	375	Chandler	Ullman	Finance
12326	Watson	2009	Fall	Rock and ...	F	11	0	11	50	145	Fairchild	Morris	Finance
12326	Watson	2009	Fall	Rock and ...	W	11	0	11	50	145	Fairchild	Morris	Finance
12326	Watson	2009	Fall	Rock and ...	M	11	0	11	50	145	Fairchild	Morris	Finance
12326	Watson	2010	Spring	Greek Tra...	F	13	0	13	50	183	Taylor	Mahmoud	Finance
12326	Watson	2010	Spring	Greek Tra...	W	13	0	13	50	183	Taylor	Mahmoud	Finance
12326	Watson	2010	Spring	Greek Tra...	M	13	0	13	50	183	Taylor	Mahmoud	Finance
12326	Watson	2003	Spring	Greek Tra...	F	13	0	13	50	757	Drown	Mahmoud	Finance
12326	Watson	2003	Spring	Greek Tra...	W	13	0	13	50	757	Drown	Mahmoud	Finance
12326	Watson	2003	Spring	Greek Tra...	M	13	0	13	50	757	Drown	Mahmoud	Finance
12326	Watson	2008	Spring	Elastic Str...	F	13	0	13	50	812	Taylor	Dale	Finance
12326	Watson	2008	Spring	Elastic Str...	W	13	0	13	50	812	Taylor	Dale	Finance
12326	Watson	2008	Spring	Elastic Str...	M	13	0	13	50	812	Taylor	Dale	Finance

4.3. Bài 3

Đề bài:

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.

Bài làm:

Tương tự như bài 2 ta thay đổi đầu vào bằng dạng bảng.

Bước 1: Tạo object mới gồm 2 trường theo yêu cầu:

```
CREATE OR REPLACE TYPE my_object
AS OBJECT(field_name VARCHAR(50),
          field_value VARCHAR(50));
```

Bước 2: Tạo kiểu dữ liệu bảng theo object đã tạo:

```
CREATE TYPE my_table
AS TABLE OF my_object;
```

Bước 3: Tạo thủ tục lọc theo kiểu bảng:

```
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 schedule 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;
```

Kết quả:

Chạy với mã sinh viên 123 và year 2008.

```
DECLARE
 row_1 hust.my_object;
 row_2 hust.my_object;
 MYTABLE hust.MY_TABLE;
 MYCURSOR SYS_REFCURSOR;
BEGIN
      row_1 := hust.my_object('id', '123');
      row_2 := hust.my_object('year', '2008');
      MYTABLE := hust.MY_TABLE(row_1, row_2);
  HUST.SP_LOC_DU_LIEU_BAI3(
    MYTABLE  => MYTABLE,
    MYCURSOR => MYCURSOR
  );
  :MYCURSOR := MYCURSOR;
END;
```

ID	STUDNET_...	YEAR	SEMESTER	TITLE	DAY	START_HR	START_MIN	END_HR	END_MIN	ROOM_NU...	BUILDING	INSTRUCT...	DEPT_NAME
7123	Holn	2008	Spring	Race Car ...	M	8	0	8	50	183	Taylor	Ullman	Math
7123	Holn	2008	Spring	Race Car ...	W	8	0	8	50	183	Taylor	Ullman	Math
7123	Holn	2008	Spring	Race Car ...	F	8	0	8	50	183	Taylor	Ullman	Math
12326	Watson	2008	Spring	Elastic Str...	M	13	0	13	50	812	Taylor	Dale	Finance
12362	Zhanr	2008	Spring	Elastic Str...	M	13	0	13	50	812	Taylor	Dale	Civil Eng.
12326	Watson	2008	Spring	Elastic Str...	W	13	0	13	50	812	Taylor	Dale	Finance
12362	Zhanr	2008	Spring	Elastic Str...	W	13	0	13	50	812	Taylor	Dale	Civil Eng.
12326	Watson	2008	Spring	Elastic Str...	F	13	0	13	50	812	Taylor	Dale	Finance
12362	Zhanr	2008	Spring	Elastic Str...	F	13	0	13	50	812	Taylor	Dale	Civil Eng.
51238	Kran	2008	Spring	World Hist...	T	14	30	15	45	707	Gates	Jaekel	Civil Eng.
12326	Watson	2008	Spring	World Hist...	T	14	30	15	45	707	Gates	Jaekel	Finance
12362	Zhanr	2008	Spring	World Hist...	T	14	30	15	45	707	Gates	Jaekel	Civil Eng.
11237	Rokhs	2008	Spring	World Hist...	T	14	30	15	45	707	Gates	Jaekel	Physics
61232	Fukui	2008	Spring	World Hist...	T	14	30	15	45	707	Gates	Jaekel	Pol. Sci.
75123	Chowdhury	2008	Spring	World Hist...	T	14	30	15	45	707	Gates	Jaekel	Physics
51238	Kran	2008	Spring	World Hist...	R	14	30	15	45	707	Gates	Jaekel	Civil Eng.
12326	Watson	2008	Spring	World Hist...	R	14	30	15	45	707	Gates	Jaekel	Finance
12362	Zhanr	2008	Spring	World Hist...	R	14	30	15	45	707	Gates	Jaekel	Civil Eng.
11237	Rokhs	2008	Spring	World Hist...	R	14	30	15	45	707	Gates	Jaekel	Physics
61232	Fukui	2008	Spring	World Hist...	R	14	30	15	45	707	Gates	Jaekel	Pol. Sci.
75123	Chowdhury	2008	Spring	World Hist...	R	14	30	15	45	707	Gates	Jaekel	Physics
51238	Kran	2008	Spring	Transactio...						134	Lamberton	DAgostino	Civil Eng.

4.4. Bài 4

Đề bài:

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

Bài làm:

Tạo thủ tục với vòng lặp để lấy những môn học trước đó đến khi không tồn tại mã môn học nào trước đó thì dừng.

```
CREATE OR REPLACE PROCEDURE SP_BAI4(course_name IN VARCHAR)
AS
      CURSOR mycursor IS
            SELECT DISTINCT p.prereq_id,
                  p.course_id
            FROM prereq p
            JOIN course c ON p.course_id = c.course_id
                  AND c.title = course_name;
    preID  mycursor%ROWTYPE;
      TYPE pre_ID IS TABLE OF preID%TYPE;
  mypreID  pre_ID;
      objectTitle VARCHAR2(50);
  pre_value VARCHAR2(50);
  recent_value VARCHAR2(50);
  mycount NUMBER;
BEGIN
      OPEN mycursor;
      FETCH mycursor BULK COLLECT INTO mypreID;
      CLOSE mycursor;
      FOR indx IN 1.. mypreID.COUNT LOOP
            SELECT c.TITLE INTO objectTitle
            FROM COURSE c
            WHERE c.COURSE_ID = mypreID(indx).PREREQ_ID;
            DBMS_OUTPUT.PUT_LINE('Những học phần học trước cho môn
'|| course_name ||
                  ' mã học phần:  '||mypreID(indx).Course_ID);
            DBMS_OUTPUT.PUT_LINE(mypreID(indx).PREREQ_ID ||'  ' ||
objectTitle);
            recent_value := mypreID(indx).PREREQ_Id;
            SELECT COUNT(p.PREREQ_ID) INTO mycount FROM PREREQ p
WHERE p.COURSE_ID = recent_value;
            IF(mycount>0) THEN
                  SELECT p.PREREQ_ID INTO pre_value
```

```
                FROM PREREQ p
                WHERE p.COURSE_ID = recent_value;
            END IF;
            IF(mycount>0) THEN
                WHILE mycount>0 LOOP
                    recent_value:=pre_value;
                    SELECT c.TITLE INTO objectTitle
                    FROM COURSE c
                    WHERE c.COURSE_ID = recent_value;
                    DBMS_OUTPUT.PUT_LINE(pre_value||' ' ||
objectTitle);
                    SELECT COUNT(p.PREREQ_ID) INTO mycount FROM
PREREQ p WHERE p.COURSE_ID = recent_value;
                    IF(mycount>0) THEN
                        SELECT p.PREREQ_ID INTO pre_value
                        FROM PREREQ p
                        WHERE p.COURSE_ID = recent_value;
                    END IF;
                END LOOP;
            END IF;
        END LOOP;
END;
```

Kết quả:

```
Những học phần học trước cho môn Mobile Computing mã học phần:  810
966  Sanitary Engineering
Những học phần học trước cho môn Mobile Computing mã học phần:  612
123  Differential Equations
```

4.5. Bài 5

Đề bài:

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 làm:

Đối với trigger ta sẽ kiểm tra

- Sinh viên đã đăng ký lớp hay chưa?
- Lớp đã đầy hay chưa?

Nếu không thỏa mãn 2 điều kiện trên thì ta thực hiện insert dữ liệu nếu không thì đưa a thông báo và không insert dữ liệu.

```
CREATE OR REPLACE TRIGGER check_student_per_classroom
    BEFORE INSERT ON takes FOR each ROW
DECLARE
```

```
  recent_capacity NUMBER;
  max_capacity NUMBER;
  registed_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO  registed_count
  FROM TAKES t
  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 (registed_count >0) THEN
           RAISE_APPLICATION_ERROR(-20201,'Sinh viên đã đăng kí
lớp. ');
  ELSE
           SELECT c.CAPACITY INTO max_capacity
           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 recent_capacity
           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(recent_capacity+1 > max_capacity) THEN
                RAISE_APPLICATION_ERROR(-20202,'Lớp học đã đầy '
|| max_capacity || '/' || recent_capacity);
           END IF;
   END IF;
END;
```

Kết quả:

Insert với query sau:

```
INSERT INTO takes ( id, course_id, sec_id, year, semester )
VALUES ( 69730, 313, 1, 2010, 'Fall' );
```

Đăng ký thành công

```
INSERT INTO takes ( id, course_id, sec_id, year, semester )
VALUES ( 69730, 313, 1, 2010, 'Fall' )
> Affected rows: 1
> Time: 0.003s
```

Lỗi khi sinh viên đã đăng ký

```
INSERT INTO takes ( id, course_id, sec_id, year, semester )
VALUES ( 69730, 313, 1, 2010, 'Fall' )
> ORA-20201: Sinh viên đã đăng kí lớp.
ORA-06512: at "HUST.CHECK_STUDENT_PER_CLASSROOM", line 14
ORA-04088: error during execution of trigger 'HUST.CHECK_STUDENT_PER_CLASSROOM'
```

Lỗi khi lớp đã đầy

```
INSERT INTO takes ( id, course_id, sec_id, year, semester )
VALUES ( 69730, 313, 1, 2010, 'Fall' )
> ORA-20202: Lớp học đã đầy 11/269
ORA-06512: at "HUST.CHECK_STUDENT_PER_CLASSROOM", line 30
ORA-04088: error during execution of trigger 'HUST.CHECK_STUDENT_PER_CLASSROOM'

> Time: 0.001s
```

4.6. Bài 6

Đề bài:

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.

Điều 23. Điểm trung bình học kỳ và điểm trung bình tích lũy

1. Điểm trung bình học kỳ (TBHK) và điểm trung bình tích lũy (TBTL) được tính theo công thức sau (làm tròn đến hai chữ số thập phân):

$$A = \frac{\sum_{i=1}^{N} a_i \times n_i}{\sum_{i=1}^{N} n_i}$$

trong đó:

A là điểm trung bình học kỳ hoặc điểm trung bình tích lũy

a_i là điểm học phần thứ i

n_i là số tín chỉ của học phần thứ i

N là số học phần tính điểm trung bình.

Bài làm:

Sử dụng view đã tạo ở bài 2.

```
CREATE OR REPLACE PROCEDURE SP_SHOW_RESULT_STUDENT_BAI6 (
student_id NUMBER, mycursor OUT sys_refcursor )
IS
BEGIN
      OPEN mycursor FOR
            SELECT t.id,
                  s.name,
                  t.semester,
                  t.year,
                  ROUND( SUM( t.point * c.credits ) / SUM( c.credits
), 2 ) gpa,
                  ROUND((
                        SELECT SUM(MAX( t1.point) * c1.credits) /
SUM( c1.credits )
                        FROM takes_number t1,
                              course c1
                        WHERE t1.id = student_id
                              AND t1.course_id = c1.course_id
                              AND t1.year <= t.year
                              AND t1.semester <= t.semester
                        GROUP BY t1.course_id,
                              c1.credits), 2) AS cpa,
                  (SELECT SUM(c1.credits)
                        FROM takes_number t1,
                              course c1
                        WHERE t1.id = student_id
                              AND t1.course_id = c1.course_id
                              AND t1.year <= t.year
                              AND t1.semester <= t.semester
                        GROUP BY t1.id) total_cred
      FROM student s
      JOIN takes_number t ON s.id = t.id
            AND t.id = student_id
      JOIN  course c ON c.course_id = t.course_id
      GROUP BY t.id,
            s.name,
            t.semester,
            t.year
      ORDER BY t.year,
            t.semester DESC;
END;
```

Kết quả:

ID	NAME	SEMESTER	YEAR	GPA	CPA	TOTAL_CR...
1232	Marcus	Fall	2001	2.5	2.5	3
1232	Marcus	Spring	2001	1	1.75	6
1232	Marcus	Fall	2003	2	2.15	10
1232	Marcus	Spring	2003	2.5	2.08	19
1232	Marcus	Spring	2004	2.43	2.17	26
1232	Marcus	Fall	2005	4	2.68	14
1232	Marcus	Fall	2006	3.36	2.9	21
1232	Marcus	Fall	2007	3.5	3	25
1232	Marcus	Spring	2008	3	2.71	45
1232	Marcus	Fall	2009	4	3.22	32
1232	Marcus	Fall	2010	1.64	2.94	39
1232	Marcus	Spring	2010	2.5	2.73	62

4.7. Bài 7

Đề bài:

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:

Điều 25. Xếp hạng trình độ và học lực cho sinh viên

1. Căn cứ vào số tín chỉ tích lũy, Nhà trường xếp hạng trình độ cho sinh viên sau mỗi học kỳ như trong Bảng 2.

Bảng 2: Xếp hạng trình độ của sinh viên

Trình độ	Số tín chỉ tích lũy		
	Cao đẳng 3 năm	Đại học 4 năm	Đại học 4,5-5 năm
Sinh viên năm thứ nhất	dưới 32 TC		
Sinh viên năm thứ hai	32 đến dưới 64 TC		
Sinh viên năm thứ ba	từ 64 TC	64 đến dưới 96 TC	
Sinh viên năm thứ tư	–	từ 96 TC	96 đến dưới 128 TC
Sinh viên năm thứ năm	–	–	từ 128 TC

3. Sau mỗi học kỳ, sinh viên được xếp hạng học lực căn cứ vào điểm trung bình tích lũy theo phân loại trong Bảng 3.

Bảng 3: Xếp hạng học lực sinh viên

Học lực	Loại	Điểm trung bình tích lũy
Bình thường	Xuất sắc	từ 3,60 đến 4,00
	Giỏi	từ 3,20 đến 3,59
	Khá	từ 2,50 đến 3,19
	Trung bình	từ 2,00 đến 2,49
Yếu kém	Yếu	từ 1,00 đến 1,99
	Kém	dưới 1,0

Bài làm:

Bước 1: Tạo function chuyển đổi trình độ của sinh viên

```
CREATE OR REPLACE FUNCTION func_convert_level_student ( no_credits
NUMBER )
RETURN VARCHAR2
IS
grade_point VARCHAR2 ( 20 ) := 'SV năm nhất';
BEGIN
      grade_point :=
            CASE
                  WHEN no_credits < 32 THEN 'SV năm nhất'
                  WHEN no_credits BETWEEN 32 AND 63 THEN 'SV năm
hai'
                  WHEN no_credits BETWEEN 64 AND 95 THEN 'SV năm ba'
                  WHEN no_credits BETWEEN 96 AND 127  THEN 'SV năm
```

```
bốn'
                WHEN no_credits >= 128 THEN 'SV năm năm'
                ELSE 'Lỗi chuyển đổi trình độ!'
          END;
     RETURN grade_point;
     EXCEPTION
          WHEN others THEN
                dbms_output.put_line ( 'ERR: ' || SQLERRM );

END;
```

Bước 2: Function chuyển đổi học lực của sinh viên:

```
CREATE OR REPLACE FUNCTION func_convert_student_capacity (
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 'Xuất sắc'
                WHEN 3.2 <= avg_point AND avg_point < 3.6 THEN
'Giỏi'
                WHEN 2.5 <= avg_point AND avg_point < 3.2 THEN
'Khá'
                WHEN 2.0 <= avg_point AND avg_point < 2.5 THEN
'Trung Bình'
                WHEN 1.0 <= avg_point AND avg_point < 2.0 THEN
'Yếu'
                WHEN avg_point < 1.0 THEN 'Kém'
                ELSE 'Lỗi chuyển đổi trình độ!'
          END;
     RETURN grade_point;
     EXCEPTION
          WHEN others THEN
                dbms_output.put_line ( 'Err: ' || SQLERRM );

END;
```

Bước 3: 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 t.id,
                  s.name,
                  t.semester,
                  t.year,
                  func_convert_student_capacity (
                        ROUND((
                        SELECT SUM(MAX( t1.point) * c1.credits) /
SUM( c1.credits )
                        FROM takes_number t1,
                              course c1
                        WHERE t1.id = student_id
                              AND t1.course_id = c1.course_id
                              AND t1.year <= t.year
                              AND t1.semester <= t.semester
                        GROUP BY t1.course_id,
                              c1.credits), 2)) AS capacity,
                  func_convert_level_student(SELECT SUM(c1.credits)
                        FROM takes_number t1
                        JOIN course c1 ON c1.COURSE_ID =
t1.COURSE_ID
                        WHERE t1.id = student_id
                              AND t1.course_id = c1.course_id
                              AND t1.year <= t.year
                              AND t1.semester <= t.semester
                        GROUP BY t1.id) level_student
      FROM student s
      JOIN takes_number t ON s.id = t.id
            AND t.id = student_id
      JOIN  course c ON c.course_id = t.course_id
      GROUP BY t.id,
            s.name,
            t.semester,
            t.year
      ORDER BY t.year,
            t.semester ASC;
END;
```

4.8. Bài 8

Đề bài:

Đánh chỉ mục các bảng takes, student, advisor. So sánh tốc độ truy vấn sau khi đã thực hiện đánh chỉ mục

Bài làm:

- takes đã có các khóa: *id, course_id, sec_id, semester, year* nên không cần đánh index, còn trường *grade* rất ít truy vấn và là trường text nên việc đánh index không cải thiện được nhiều cho việc truy vấn dữ liệu.

```
CREATE INDEX take_index ON takes(grade);
```

- student có khóa: *id,* đánh index cho các trường *name, dept_name, tot_cred* việc đánh index với những trường này sẽ cải thiện tốc độ truy vấn tuy nhiên do bảng này khá ít dữ liệu (2000) nên việc đánh index không cải thiện nhiều về truy vấn dữ liệu.

```
CREATE INDEX student_index ON student(name, dept_name, tot_cred);
```

- advisor có khóa *s_id* nên chỉ đánh index cho trường *i_id* tương tự như student bảng này có khá ít dữ liệu nên việc đánh index không cải thiện nhiều về tốc độ truy vấn.

```
CREATE INDEX advisor_index ON advisor(i_id);
```

4.9. Bài 9

Đề bài:

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**.**

Bài làm:

```
CREATE OR REPLACE PROCEDURE SP_REGISTER_BAI9(student_id VARCHAR2,
    course_id VARCHAR2,
    sec_id VARCHAR2,
    semester VARCHAR2,
    year NUMBER)
AS
BEGIN
  SAVEPOINT pre_register;
  INSERT INTO TAKES (ID, COURSE_ID, SEC_ID, SEMESTER, YEAR)
    VALUES (student_id,course_id,sec_id, semester, year);
  EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Đã có lỗi, trạng thái về thời điểm trước
```

```
đăng kí!');
    ROLLBACK TO pre_register;
    RAISE;
END;
```

4.10. Bài 10

Kết luận

Qua môn học này, em đã thu được:

- Một số khái niệm cơ bản về Big Data, Kho dữ liệu, xử lý và khai phá dữ liệu.
- Các khái niệm và cơ chế xử lý tiến trình, giải quyết tương tranh trong các hệ quản trị cơ sở dữ liệu.
- Các khái niệm, kiến trúc của mô hình cơ sở dữ liệu phân tán và song song.
- 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.
- Có thể tạo được một ứng dụng đơn giản về truy vấn dữ liệu trong Oracle.

Một lần nữa em xin chân thành cảm ơn thầy cô đã giúp đỡ em hoàn thành môn học Cơ sở dữ liệu nâng cao!

Tài liệu tham khảo