Chương 1

hệ quản trị cơ sở dữ liệu quan hệ

1. **Hệ quản trị cơ sở dữ liệu là gì?**

Một cơ sở dữ liệu (database) là một tập hợp các thông tin được tổ chức hợp lý để có thể truy xuất nhanh khi cần sử dụng. Một hệ quản trị cơ sở dữ liệu (DataBase Management System – DBMS) là một công cụ cho phép quản lý và tương tác với cơ sở dữ liệu.

1. **Các chức năng cơ bản của hệ quản trị cơ sở dữ liệu?**

Các chức năng cơ bản của hệ quản trị cơ sở dữ liệu:

* Lưu trữ dữ liệu
* Tạo ra và duy trì cấu trúc dữ liệu
* Cho phép nhiều người dùng truy xuất đồng thời
* Hỗ trợ tính bảo mật và riêng tư
* Cho phép lấy ra và xử lý các dữ liệu lưu trữ
* Cho phép nhập liệu và nạp dữ liệu
* Cung cấp một cơ chế chỉ mục (index) hiệu quả để lấy nhanh các dữ liệu lựa chọn.
* Cung cấp tính nhất quán giữa các bản ghi khác nhau.
* Bảo vệ dữ liệu khỏi mất mát bằng các quá trình sao lưu (backup) và phục hồi (recovery).

1. **Kể ra ba loại hệ quản trị cơ sở dữ liệu chính?**

Ba loại hệ quản trị cơ sở dữ liệu chính là:

***Hệ quản trị cơ sở dữ liệu phân cấp chứa dữ liệu trong một cấu trúc như cây.*** Hệ này thiết lập một quan hệ cha-con trong dữ liệu. Ở đỉnh của cây, còn gọi là root, có thể có một số nút bất kỳ. Các nút này lại có thể có một số nút con bất kỳ, v.v... Các hệ cơ sở dữ liệu phân cấp ngày nay đã lạc hậu.

***Một hệ quản trị cơ sở dữ liệu mạng chứa dữ liệu ở dạng các bản ghi và các liên kết***. Hệ thống này cho phép nhiều quan hệ nhiều-nhiều hơn hệ cơ sở dữ liệu phân cấp. Một hệ quản trị cơ sở dữ liệu mạng lưu trữ có hiệu quả và rất nhanh, cho phép quản lý các cấu trúc dữ liệu phức tạp. Tuy nhiên, chúng lại rất cứng nhắc và yêu cầu nặng nề về thiết kế. Một hệ thống đăng ký vé máy bay là một ví dụ của loại này.

***Hệ quản trị cơ sở dữ liệu quan hệ có lẽ là một cơ sở dữ liệu có cấu trúc đơn giản nhất***. Trong một hệ quản trị cơ sở dữ liệu quan hệ, dữ liệu được tổ chức thành các bảng. Các bảng lại bao gồm các bản ghi và các bản ghi chứa các trường. Mỗi trường tương ứng với một mục dữ liệu. Hai hay nhiều bảng có thể liên kết nếu chúng có một hay nhiều trường chung.

1. **Thế nào là cơ sở dữ liệu phẳng?**

Cơ sở dữ liệu phẳng gồm một bảng với các bản ghi có đủ cột để chứa tất cả các dữ liệu cần thiết. Thuật ngữ dữ liệu phẳng bắt nguốn từ sự kiện là cơ sở dữ liệu có hai chiều – số trường (field) của bảng xác định chiều rộng của cơ sở dữ liệu, và số lượng các bản ghi của bảng xác định chiều cao.

1. **Các tính chất của một bảng quan hệ trong mô hình quan hệ?**

Một bảng quan hệ trong mô hình quan hệ cần có thỏa các tính chất sau:

* Dữ liệu chứa trong các ô cần phải là nguyên tố. Mỗi ô chỉ có thể chứa một mẩu dữ liệu. Đó chính là yếu tố cơ bản của thông tin.
* Dữ liệu trong một cột nào đó phải có cùng kiểu dữ liệu.
* Mỗi hàng là duy nhất (không có các hàng trùng nhau).
* Các cột không có thứ tự.
* Các hàng không có thứ tự.
* Các cột có một tên duy nhất.

Chương 2

oracle và công nghệ khách/chủ

1. **Nêu ba ưu điểm của kiến trúc công nghệ khách/chủ so với kiến trúc công nghệ máy lớn?**

Cấu trúc khách/chủ đưa ra một số ưu điểm so với các kiến trúc cũ:

* Nó hỗ trợ việc sử dụng nhiều chủng loại máy client khác nhau. Trong môi trường ngày nay, người sử dụng có thể dùng Windows 3.11, Windows 95, Windows NT, một máy Mac, hay một trạm UNIX, tất cả đều có thể nối với cùng một server.
* Trách nhiệm tính toán được phân bổ một cách hợp lệ giữa client và server. Máy client có trách nhiệm điều khiển giao diện người dùng – hiển thị các thông tin, hợp lệ hóa đầu vào và cung cấp sự phản hồi (feedback) hợp lý – trong khi máy server được dùng như một server cơ sở dữ liệu.
* Nó độc lập với thủ tục mạng, hệ điều hành ở máy server và hệ điều hành ở máy client. Sự độc lập này cho ta sự linh động lớn trong việc lựa chọn các thành phần phần cứng và phần mềm của một hệ thống mới khi hay khi hiện thực một ứng dụng khách/chủ trên một cơ sở phần cứng đã tồn tại.

1. **Ưu điểm của stored procedure so với một khối lệnh PL/SQL?**

Ưu điểm của stored procedure so với một khối lệnh PL/SQL là:

* Gọi một stored procedure từ một ứng dụng client giảm đáng kể mật độ lưu thông mạng. Thay vì ứng dụng gởi một khối chương trình PL/SQL từ client, chỉ cần một lệnh gọi đơn đến thủ tục hay hàm với một danh sách tham số tùy ý.
* Các stored procedure cung cấp một cơ chế bảo mật hiệu quả và tiện lợi. Một trong những đặc trưng của stored PL/SQL là nó luôn thực hiện với đặc quyền (privilege) của chủ nhân thủ tục (procedure). Điều này ngăn các user không đặc quyền dùng mã thủ tục truy xuất các đối tượng đặc quyền. Đặc tính này thường dùng để giảm lượng giám sát cấp phát quyền mà DBA phải làm.
* Cả hai dạng text và đã biên dịch của các stored procedure đều nằm trong cơ sở dữ liệu. Do dạng biên dịch của thủ tục có sẵn và sẵn sàng thực hiện, nhu cầu phân tích cú pháp và biên dịch PL/SQL lúc thi hành (run time) được giảm nhẹ.

1. **Chức năng của Net8?**

Net8 là một phần mềm kết nối cơ sở dữ liệu dùng để cung cấp các thông tin cơ sở dữ liệu xác thực, tối ưu trên mọi thủ tục mạng thông dụng. Net8 được thiết kế để cung cấp tính trong suốt của server cho bất kỳ nút nào bên trong một ứng dụng mạng và sử dụng các thành phần nằm trên cả hai phía client và server của một ứng dụng.

Ngoài việc cung cấp sự kết nối giữa máy trạm và server trong môi trường khách/chủ, server cũng sử dụng Net8 để giao dịch các chuyển tác phân tán (distributed transaction), các phép gọi thủ tục từ xa và sao lưu bảng (replication) với các server khác. Các server tham chiếu đến các server khác bằng cách sử dụng các liên kết cơ sở dữ liệu (*database link*) để định nghĩa tên của các cơ sở dữ liệu từ xa. Các liên kết cơ sở dữ liệu đơn giản hóa quá trình phân tán bằng cách cung cấp sự truy xuất trong suốt các đối tượng từ xa như các bảng và thủ tục, cho phép một ứng dụng tham chiếu chúng như thể chúng nằm ở cơ sở dữ liệu cục bộ của ứng dụng.

Chương 3

cấu trúc của oracle

1. **Những thành phần phần cứng chính ảnh hưởng đến hiệu năng?**

Các thành phần phần cứng chính ảnh hưởng đến hiệu năng là: CPU, cache, bộ nhớ (memory), bus, đĩa cứng và mạng.

1. **Điều gì xảy ra nếu các tập tin redo log bị hư hỏng?**

Nếu thông tin tập tin redo log bị mất, ta không thể phục hồi lại hệ thống nếu nó bị hư hỏng.

1. **Điều gì xảy ra nếu các tập tin dữ liệu bị hư hỏng?**

Nếu ta có một bộ đầy đủ các tập tin offline redo log từ lần cuối backup cơ sở dữ liệu, khi các tập tin dữ liệu bị hư ta có thể dùng bản backup và các tập tin redo log để gán lại những thay đổi cho bản backup.

1. **Cấu thành của một thể hiện của Oracle?**

Mỗi lúc một cơ sở dữ liệu được tạo ra, một system global area (SGA) được định vị và các quá trình chạy nền (background) của Oracle được khởi động. Sự kết hợp giữa các quá trình background và các bộ nhớ đệm gọi là một thể hiện của Oracle.

Chương 4

sql

1. **Điều này đúng hay sai: “Cần phải đưa một cột vào mệnh đề SELECT nếu ta muốn sắp thứ tự các kết quả trả về theo cột đó?**

Sai. Bạn có thể chỉ định một cột trong mệnh đề **ORDER** không nằm trong các cột chọn của mệnh đề **SELECT** để sắp thứ tự các kết quả trả về theo cột đó

1. **Lệnh này đúng hay sai:**

select First\_Name

from student

order by Last\_Name

where

Last\_Name like ‘%ỊN’

Sai. Mệnh đề **order by** phải đứng sau mệnh đề **where**.

1. **Bảng DUAL là gì? Tại sao nó được sử dụng?**

Oracle cho phép user thực hiện một số lệnh SQL đặc biệt để hoàn thành những bài toán số học mà không cần truy xuất dữ liệu. Đặc trưng này liên quan đến một bảng đặc biệt gọi là DUAL. DUAL là một bảng rỗng được dùng để đáp ứng cấu trúc của lệnh **select**.

**SELECT 64+36 FROM DUAL;**

Kết quả sẽ cho ta:

64+36

-----

100

Trong bảng DUAL không có dữ liệu thực sư, nó chỉ tồn tại như là một cấu trúc SQL để hỗ trợ yêu cầu chỉ định bảng trong mệnh đề **FROM**.Bảng DUAL chứa chỉ một cột và một hàng với giá trị NULL.

1. **Hàm nvl() dùng để làm gì? Cách sử dụng nó?**

Hàm **nvl()** được dùng để biến đổi một cột sao cho khi giá trị của cột này là NULL, nó sẽ trả vể một chuỗi mặc định. Cú pháp của nó là:

**NVL**(tên\_cột, giá\_trị\_nếu\_null)

1. **Bí danh (Alias) cột là gì? Nó hữu dụng trong những trường hợp nào?**

Bí danh cột được sử dụng để chỉ định rõ nghĩa các heading cho dữ liệu trong output từ các query SQL. Các alias có thể chỉ định theo hai cách, hoặc đặt alias sau tên cột cách bằng một khoảng trống, hoặc sử dụng từ khóa **as** để đánh dấu alias rõ ràng hơn cho các người đọc khác.

1. **Ký tự gì dùng để nối các cột trong mệnh đề select?**

Phương pháp được sử dụng để nối output của một số cột với nhau nhằm tạo ra một dữ liệu mới được thực hiện bằng một toán hạng đặc biệt gọi là concatenation (ghép). Toán hạn ghép có ký hiệu như hai ký tự **|** ghép với nhau: **||**.

1. **Mục đích của lệnh decode()? Nó chấp nhận đối số nào?**

Hàm **decode( )** cho phép chuyển đổi dữ liệu từ dạng này sang dạng khác một cách linh hoạt. Nó có thể sử dụng thay thế hàm **nvl()**, tuy nhiên nó mạnh hơn nhiều ở khả năng chấp nhận bất cứ đối số nào.

1. **Thế nào là một hàm group? Kể ra vài hàm group?**

Một hàm nhóm cho phép user thực hiện một thao tác dữ liệu trên một vài giá trị trong một cột dữ liệu như thể cột là một nhóm tập hợp dữ liệu. Những hàm này cũng được gọi là các hàm group, bởi vì chúng thường được sử dụng trong một mệnh đề đặc biệt của lệnh **select** gọi là mệnh đề **group by**.

Một số hàm group thông dụng là:

|  |  |
| --- | --- |
| **avg(*x*)** | Trung bình của tất cả các giá trị của cột x trả về bởi lệnh **select** |
| **count(*x*)** | Tổng tất cả các hàng của cột x trả về bởi lệnh **select** |
| **max(*x*)** | Giá trị lớn nhất trong cột x của tất cả các hàng trả về bởi lệnh **select** |
| **min(*x*)** | Giá trị nhỏ nhất trong cột x của tất cả các hàng trả về bởi lệnh **select** |

1. **ROWID là gì?**

Một ROWID là một giá trị đặc biệt chỉ định duy nhất mỗi hàng. Mỗi hàng trong một bảng có một gía trị ROWID duy nhất. ROWID không phải là một phần thực sự của bảng mà là một mẫu thông tin trữ trong Oracle. Vì vậy, nó được gọi là một cột giả.

1. **Từ khóa distinct được sử dụng để làm gì?**

Từ khoá distinct dùng để lọc lấy một giá trị duy nhất khi bộ lọc trả về nhiều kết quả giống nhau.

1. **Ký tự đặc biệt nào được sử dụng để chỉ định một biến runtime?**

Ký tự **&** được sử dụng để chỉ định một biến runtime.

1. **Nêu ra hai ràng buộc tạo ra index?**

Hai ràng buộc tạo ra index là primary key và unique key.

1. **Cách tạo ra các bảng có sẵn dữ liệu?**

Để tạo ra các bảng có sẵn dữ liệu từ các bãng khác, ta dùng mệnh đề **CREATE TABLE** ... **AS SELECT**

1. **Tự điển dữ liệu là gì?**

Tự điển dữ liệu Oracle bao gồm các bảng chứa thông tin về cơ sở dữ liệu được lưu trữ, giúp người dùng theo dõi cơ sở dữ liệu.

1. **Lệnh nào được sử dụng để thay đổi định nghĩa một bảng?**

Để thay đổi định nghĩa một bảng, ta dùng lệnh **ALTER TABLE**

Chương 5

PL/SQL

1. **Nêu một số ưu điểm khi sử dụng PL/SQL để truy xuất dữ liệu?**

Các ưu điểm khi sử dụng PL/SQL để truy xuất dữ liệu

* PL/SQL được quản lý tập trung trong cơ sở dữ liệu Oracle. DBA quản lý source code và đặc quyền thực hiện với cú pháp tương tự khi quản lý các đối tượng cơ sở dữ liệu khác.
* PL/SQL giao tiếp tự nhiên với các đối tượng cơ sở dữ liệu khác.
* PL/SQL dễ đọc và có nhiều đặc trưng module hóa code và quản lý lỗi.

1. **Kể ra ba phần của một khối mã PL/SQL. Bốn loại khối mã trong Oracle?**

Bất kỳ một khối PL/SQL nào cũng có ba thành phần. Những thành phần này là *phần khai báo biến, phần thực thi* và *trình xử lý exception.* Phần khai báo biến chứa phần định danh tất cả các cấu trúc biến sẽ được sử dụng trong khối code. Một biến có thể là một kiểu dữ liệu bất kỳ có sẵn trong cơ sở dữ liệu Oracle, cũng như một số kiểu đặc biệt của PL/SQL. Phần thực thi của một khối PL/SQL bắt đầu với từ khóa **begin** và kết thúc hoặc với từ khóa **end** đối với toàn bộ khối hoặc với từ khóa **exceptions**. Thành phần cuối cùng của một khối PL/SQL là trình xử lý exception. Phần code này định nghĩa tất cả các lỗi có thể xảy ra trong khối và chỉ định chúng nên được xử lý như thế nào. Trình xử lý exception là một tùy chọn trong PL/SQL.

Bốn loại khối mã trong Oracle là:

* *Procedure –* một chuỗi các lệnh nhận và/hay trả về zero hay nhiều biến.
* *Function* - một chuỗi các lệnh nhận zero hay nhiều biến và trả về một giá trị.
* *Package* – Một tập các thủ tục và các hàm và có hai phần, một danh sách đặc tả các hàm và các thủ tục có sẵn cùng các thông số của nó, và một phần thân chứa code thực sự của các thủ tục và các hàm.
* *Trigger* – Một chuỗi các lệnh PL/SQL gắn với một bảng cơ sở dữ liệu mà được thực hiện khi một event (**select**, **update**, **insert**, **delete**) xảy ra.

1. **Kể ra một số thuộc tính con trỏ (cursor) ngầm.**

Một số thuộc tính của con trỏ (cursor) ngầm:

* **%notfound** chỉ định không tìm ra hàng nào. Giá trị trả về là TRUE hay FALSE, ngược với **%found**.
* **%rowcount** chỉ định số hàng đã được xử lý bởi cursor. Trả về một giá trị số.
* **%found** chỉ định có tìm được hàng nào không. Giá trị trả về là TRUE hay FALSE, ngược với **%notfound**.
* **%isopen**  chỉ định cursor đề cập đến có mở và sẳn sàng sử dụng hay không. Trả về là TRUE hay FALSE

1. **Có bao nhiêu loại vòng lặp trong PL/SQL?**

Các kiểu vòng lặp có sẵn trong PL/SQl là:

* Lệnh **loop-exit**.
* Lệnh **while-loop**.
* Lệnh **for-loop**.

1. **Phân biệt con trỏ (cursor) ngầm và con trỏ tường minh?**

Cursor ngầm là một địa chỉ không đặt tên của một lệnh SQL được xử lý bởi Oracle và/hay cơ chế thực thi PL/SQL. Mọi lệnh SQL đều thực hiện trong một cursor ngầm, bao gồm các lệnh **update**, **insert**, **delete và** các lệnh **select** không thực hiện trong một cursors tường minh.

Một con trỏ tường minh được đặt tên bởi developer. Nó chính là một lệnh **select** được đặt tên. Bất kỳ lệnh **select** nào cũng có thể sử dụng trong một cursor tường minh bằng cách sử dụng cú pháp **cursor** *cursor\_name* **is**. Khi một lệnh **select** được đặt trong một cursor tường minh, developer có toàn quyền điều khiển trên sự thực hiện lệnh.

**DECLARE**

**CURSOR employee\_cursor IS**

**SELECT \* FROM employee;**

Chương 6

sql\*PLUS

1. **Các cách kết thúc một lệnh SQL trong SQL\*Plus?**

Bạn có thể kết thúc một lệnh SQL theo một trong ba cách:

* với một dấu chấm phẩy (;)
* với một dấu slash (/) nằm một mình trên một hàng
* với một hàng trống

Một dấu chấm phẩy (;) báo cho SQL\*Plus biết bạn muốn chạy lệnh. Đánh dấu chấm phẩy ở cuối hàng cuối cùng của lệnh và nhấn [Enter], SQL\*Plus sẽ xử lý lệnh và đưa nó vào bộ đệm của SQL (SQL buffer). Nếu bạn vô ý nhấn [Enter] trước khi gõ dấu chấm phẩy, SQL\*Plus sẽ nhắc bạn bằng một hàng kế có đánh số. Gõ dấu chấm phẩy và nhấn [Enter] lần nữa để chạy lệnh.

Một dấu slash (/) nằm một mình trên một hàng cũng báo cho SQL\*Plus biết rằng bạn muốn chạy lệnh. Khi nhấn [Enter] ở hàng cuối cùng, SQL\*Plus sẽ nhắc bạn bằng một hàng kế có đánh số, nhấn slash (/) và nhấn [Enter] một lần nữa, SQL\*Plus sẽ thực hiện lệnh đồng thời đưa nó vào SQL buffer.

Một hàng trống báo cho SQl\*Plus biết rằng bạn đã chấm dứt đưa lệnh vào, nhưng chưa muốn chạy nó. Khi nhấn [Enter] ở hàng cuối cùng, SQL\*Plus sẽ nhắc bạn bằng một hàng kế có đánh số, nhấn [Enter] một lần nữa, SQL\*Plus sẽ hiện trở lại dấu nhắc lệnh. Lệnh bạn vừa đánh vào không được thực hiện mà chỉ đưa vào SQL buffer. Nếu lúc này bạn lại đưa vào một lệnh khác, SQL\*Plus sẽ ghi chồng lên lệnh vừa rồi.

1. **Cách thêm một hàng mới vào bộ đệm của SQL\*Plus?**

Để chèn một hàng mới vào hàng hiện hành, sử dụng lệnh INPUT.

Để chèn một hàng trước hàng 1, nhập vào số không ("0") và theo sau là chuỗi text. SQL\*Plus chèn hàng vào đầu buffer và hàng này trở thành hàng 1.

SQL> 0 SELECT EMPNO

Giả sử bạn muốn thêm một hàng thứ tư vào lệnh SQL bạn đã sửa ở ví dụ trước. Do hàng thứ 3 là hàng hiện hành, nhập vào INPUT (có thể viết tắt là I) và nhấn [Enter]. SQL\*Plus nhắc bạn hàng mới:

SQL> INPUT

4

Nhập vào hàng mới rồi nhấn [Enter]. SQL\*Plus lại nhắc bạn lần nữa với hàng mới:

4 ORDER BY SAL

5

Nhấn [Enter] lần nữa để chỉ định rằng bạn sẽ không nhập vào thêm hàng nào nữa, và sau đó sử dụng RUN để kiểm tra và chạy lại query.

1. **Biến liên kết REFCURSOR là gì? Cách sử dụng?**

Các biến liên kết REFCURSOR của SQL\*Plus cho phép SQL\*Plus fetch và format các kết quả của một lệnh SELECT chứa trong một khối PL/SQL.

Các biến liên kết REFCURSOR cũng có thể sử dụng để tham trỏ các biến cursor PL/SQL trong các stored procedure. Điều này cho phép bạn lưu các lệnh SELECT trong cơ sở dữ liệu và tham trỏ chúng từ SQL\*Plus.

Một biến liên kết REFCURSOR cũng có thể trả về từ một function.

Để tạo ra, tham trỏ và hiển thị một biến liên kết REFCURSOR, đầu tiên khai báo một biến liên kết cục bộ kiểu dữ liệu REFCURSOR:

SQL> VARIABLE dept\_sel REFCURSOR

Kế đó, nhập vào khối PL/SQL sử dụng biến liên kết trong một lệnh OPEN ... FOR SELECT. Lệnh này mở một biến cursor và thực hiện một query.

Chương 7

sql\*LOADER

1. **Tập tin bad là gì? Chúng được tạo ra như thế nào? Tập tin bad chứa những gì? Được ghi theo khuôn mẫu nào?**

Các tập tin bad dùng để SQL\*Loader có thể đặt các record dữ liệu không thể nạp vào cơ sở dữ liệu.

Nếu trong khi thực hiện việc load dữ liệu, SQL\*Loader gặp các record không thể nạp, record bị từ chối và đưa vào *bad file*. Record sau đó có thể xem xét để tìm ra vấn đề. Các điều kiện có thể khiến một record bị từ chối là xâm phạm ràng buộc toàn vẹn, ghép sai kiểu dữ liệu, và các lỗi khác trong xử lý trường.

SQL\*Loader ghi các record bad vào file theo cùng format khi được nạp vào SQL\*Loader từ data file. Đặc trưng này cho phép sửa chữa dễ dàng và nạp lại, để sử dụng lại file control gốc.

1. **Chức năng và nội dung của tập tin log?**

Việc ghi lại sự thực thi của SQL\*Loader diễn ra trong log file. Nếu vì bất kỳ lý do nào SQL\*Loader không thể tạo ra một *log file*, sự thực thi chấm dứt. Log file chứa sáu thành phần. Phần header chứa phiên bản (version) SQL\*Loader và ngày chạy. Phần thông tin toàn cục chứa tên tất cả các file input và output, các thông số hàng lệnh và một đặc tả ký tự liên tục nếu có yêu cầu. Phần thông tin *bảng* liệt kê tất cả các bảng đang được nạp hiện hành, các điều kiện nạp, dữ liệu được insert, append hay replace. Phần datafile chứa chi tiết về bất cứ record nào bị từ chối. Phần thông tin *nạp bảng* liệt kê số bảng được nạp và và số record nạp không được, ví dụ do ràng buộc toàn vẹn. Cuối cùng, phần thống kê tổng hợp mô tả không gian sử dụng cho mãng liên kết, thống kê nạp, thời gian chấm dứt, thời gian tiêu hao và thời gian CPU.

1. **Tập tin discard là gì? Mệnh đề nào xác định nội dung của nó?**

SQL\*Loader cho user các tùy chọn để từ chối dữ liệu dựa trên một tiêu chí đặc biệt. Tiêu chí này được định nghĩa trong một file control như là một phần của mệnh đề **when**. Nếu SQL\*Loader gặp một record không thỏa mệnh đề **when**, record được đặt vào một file đặc biệt gọi là *discard file*.

1. **Ba cách sử dụng thông số hàng lệnh trong nạp dữ liệu?**

Ba cách sử dụng thông số hàng lệnh trong nạp dữ liệu:

* Chỉ định các giá trị của từng tùy chọn mà không cần gọi tên tùy chọn, miễn là các giá trị tương ứng theo vị trí trong danh sách.
* Đặt các thông số hàng lệnh trong một parfile. Parfile sau đó có thể được tham trỏ trong hàng lệnh.
* Chỉ định các thông số hàng lệnh trong file control. Để đặt các thông số hàng lệnh vào file control, mệnh đề **options** cần sử dụng trong file control. Mệnh đề này nên đặt ở đầu file, trước mệnh đề **load**

1. **Tại sao nạp đường dẫn trực tiếp thường tốn ít thời gian hơn nạp qui ước?**

Trong một lần load qui ước, SQL\*Loader đọc nhiều record dữ liệu từ file input vào một mãng liên kết. Khi mãng đầy, SQL\*Loader đưa dữ liệu cho cơ chế xử lý của Oracle SQL hay bộ tối ưu hóa để chèn. Trong load trực tiếp, SQL\*Loader đọc các record từ datafile, chuyển trực tiếp vào các khối dữ liệu Oracle và ghi chúng lên đĩa, bỏ qua hầu hết việc xử lý cơ sở dữ liệu Oracle. Vì vậy thời gian xử lý của tùy chọn này thường nhanh hơn cách nạp (load) quy ước.

Chương 14

thiết kế cơ sở dữ liệu

1. **Quá trình chuẩn hóa dữ liệu là gì?**

Quá trình chuẩn hoá (normalization) là một kỹ thuật được sử dụng để nhóm các thuộc tính theo cách loại trừ dư thừa dữ liệu.

1. **Nêu 4 dạng chuẩn hóa của cơ sở dữ liệu?**

Bốn dạng chuẩn hóa của cơ sở dữ liệu

* Dạng chuẩn thứ nhất rất dễ đạt. Yêu cầu duy nhất là mỗi “ô” trong bảng có một giá trị đơn. Nói cách khác, dạng chuẩn đầu tiên sẽ bị xâm phạm nếu có nhiều giá trị được lưu trong một sự kết hợp hàng và cột của bảng.
* Dạng chuẩn thứ hai được đạt đến khi tất cả các phụ thuộc một phần bị thu hồi. Nếu không có các khóa phức trong bảng, mức chuẩn hóa này cũng dễ đạt được.
* Dạng chuẩn thứ ba yêu cầu rằng tất cả các phụ thuộc transitive được thu hồi. Trong các tương quan mẫu đã đưa ra trước đây, có ít nhất hai phụ thuộc transitive. Trong bảng Addresses, cả thành phố và bang đều phụ thuộc vào zip code. (Cũng nói rằng có sự phụ thuộc nhiều giá trị giữa zip code và thành phố, bang).
* Dạng chuẩn thứ tư đạt được khi tất cả các sự phụ thuộc nhiều giá trị được thu hồi.

1. **Khi nào thì ta bỏ chuẩn hóa?**

Việc bỏ chuẩn hóa, ngược với quá trình chuẩn hóa, có thể sử dụng khi cần cải thiện hiệu năng.

# Chương 16 SQL động

1. Tại sao phải sử dụng SQL động, so với các lệnh SQL thông thường SQL động có những điểm gì cần lưu ý?

2. Để sử dụng được lệnh SQL động cần phải có điều kiện gì?

3. Hãy cho ví dụ về sử dụng các lệnh DDL (như DROP, CREATE) trong PL/SQL.

4. Hãy nêu cách sử dụng các lệnh SELECT, INSERT, UPDATE trong SQL động.

## Đáp án

1. SQL động cho phép bạn xây dựng các thủ tục mang tính tổng quát cao, xử lý được những thao tác mà tên của bảng hay đối tượng muốn truy xuất chỉ xác định được vào lúc chạy chương trình.

Chương trình có thể phân tích và thực thi các lệnh DML SQL (như INSERT, UPDATE …) hay các lệnh DDL SQL (như CREATE TABLE, DROP …). SQL động cho phép khắc phục hạn chế của ngôn ngữ PL/SQL, PL/SQL không cho phép bạn thực thi các lệnh DDL SQL trong khối lệnh nhưng sử dụng lệnh SQL động bạn có thể thực hiện được khả năng này

2.Muốn sử dụng SQL động bạn phải được cấp quyền EXECUTE để gọi các hàm trong Package DBMS\_SQL Để tạo và sử dụng được các hàm trong Package DBMS\_SQL bạn cần kết nối vào user SYS để chạy các Script DBMSSQL.SQL và PRVTSQL.PLB. Hai script này sẽ được tự động gọi khi bạn chạy script tổng quát CATPROC.SQL.

3. Ví dụ về thực thi lệnh DROP trong PL/SQL

cursor\_name INTEGER;

ret INTEGER;

SQLstring VARCHAR2(100);

BEGIN

cursor\_name := DBMS\_SQL.OPEN\_CURSOR;

SQLString:=’DROP TABLE EMP’;

--Lệnh DDL được thực thi bởi thủ tục PARSE

--Sau quá trình thực thi lệnh COMMIT ngầm định sẽ được thực hiện

DBMS\_SQL.PARSE(cursor\_name, SQLstring, DBMS\_SQL.NATIVE);

ret := DBMS\_SQL.EXECUTE(cursor\_name);

DBMS\_SQL.CLOSE\_CURSOR(cursor\_name);

END;

4. Xem lý thuyết mục 4 (chương SQL động).

# *Chương 19*

**TỔNG QUAN VỀ LẬP TRÌNH ORACLE OLE**

1. Oracle OLE Object là gì? Một chương trình chạy trên hệ điều hành Windows muốn sử dụng được OO40 cần phải có điều kiện như thế nào?

Oracle OLE Server (OO4O) là phần mềm trung gian (middleware) do Oracle xây dựng cho phép truy cập cơ sở dữ liệu một cách trực tiếp (native access). OO4O chỉ sử dụng được trong môi trường Windows (9.x/NT/2000) của Microsoft với kiến trúc COM (hay OLE) của hệ điều hành. Bạn phải cài đặt các thư viện cần thiết do Oracle cung cấp . Mỗi phiên bản OO4O có thể cung cấp thêm những chức năng mới do đó khi sử dụng OO4O bạn nên xem xét kỹ những thuộc tính và phương thức dành cho mỗi phiên bản. Không nên đem những chương trình sử dụng chức năng của OO4O trong Oracle 7 sang OO4O của Oracle 8 hay ngược lại (mặc dù điều này cũng không hẳn luôn luôn gây ra lỗi).

2. OO4O so với ODBC có những ưu và khuyết điểm gì?

So sánh OO4O và ODBC: OO4O cho phép truy cập trực tiếp đến cơ sở dữ liệu Oracle, tốc độ truy cập do đó sẽ nhanh hơn ODBC. Tuy nhiên ODBC là cầu nối được dùng chung cho tất cả cơ sở dữ liệu trên Windows trong khi OO4O chỉ là một đối tượng OLE đơn thuần dành riêng cho việc truy xuất đến môi trường cơ sở dữ liệu Oracle. OO4O do Oracle cung cấp trong khi Oracle ODBC có thể được sự hỗ trợ từ nhiều hãng phần mềm khác.

3. Có thể sử dụng OO4O từ các ứng dụng của Microsoft (như Word, Excell, Access …) được không?

OO4O được xây dựng dựa trên mô hình COM của Microsoft nên bạn có thể sử dụng OO4O trong hầu hết các ứng dụng của Microsoft như Word, Excell, Access hay SQL Server.

4. Làm thế nào để kết nối và truy xuất được cơ sở dữ liệu của Oracle bằng OO4O từ môi trường lập trình Visual Basic hay từ các ứng dụng của Microsoft Office, cho ví dụ.

Truy xuất cơ sở dữ liệu Oracle bằng OO4O trong môi trường lập trình Visual Basic cũng như trong hầu hết các ứng dụng Microsoft Office khác hoàn toàn tương tự nhau (các ứng dụng Microsoft Office như Word, Excell sử dụng các Macro với ngôn ngữ Visual Basic). Ví dụ để lấy thông tin về một nhân viên trong bảng EMP truy xuất bằng OO4O được thực hiện như sau:

Sub Form\_Load ()

Dim OraSession As Object  *'Khai báo đối tượng OLE Objects*

Dim OraDatabase As Object

Dim OraDynaset As Object

‘Tạo đối tượng OO4O

Set OraSession = CreateObject(

"OracleInProcServer.XOraSession")

‘Kết nối với cơ sở dữ liệu Oracle

Set OraDatabase = OraSession.DbOpenDatabase(

"ExampleDB", "scott/tiger", 0&)

MsgBox "Connected to " & OraDatabase.Connect & "@"

& OraDatabase.DatabaseName

'Tạo đối tượng OraDynaset và lấy kết quả trả về

Set OraDynaset = OraDatabase.DbCreateDynaset("select

empno, ename from emp", 0&)

MsgBox "Employee " &

OraDynaset.Fields("empno").value

& ", #" & OraDynaset.Fields("ename").value

End Sub

5. Làm thế nào để kết nối và truy xuất cơ sở dữ liệu Oracle từ các môi trường lập trình C++ hay Delphi bằng OO4O, cho ví dụ.

Kết nối với Oracle bằng OO4O từ môi trường Delphi gần như hoàn toàn tương tự với môi trường Visual Basic ví dụ:

Var OraSession :OleVariant; *// Khai báo đối tượng OLE*

OraDatabase:OleVariant;

OraDynaset:OleVariant;

Begin

*//Tạo đối tượng OLE*

OraSession:= CreateOLEObject(

‘OracleInProcServer.XoraSession’);

*//Kết nối với cơ sở dữ liệu*

OraDatabase:=OraSession.OpenDatabase

(‘ExampleDB’, ‘username’,’password’);

*//Truy xuất dữ liệu*

OraDynaset:= OraDatabase.DbCreateDynaset(

‘select empno, ename from emp’, 0)

ShowMessage(‘Employee ‘ +

OraDynaset.Fields(‘empno’).value +

OraDynaset.Fields(‘ename’).value);

End;

Kết nối với Oracle từ môi trường lập trình C++ bằng OO4O được sử dụng thông qua các lớp Oracle Class Library như sau:

#ifndef ORACL\_ORACLE

#include "oracl.h"

#endif

int main(void){

//Thực hiện kết nối với cơ sở dữ liệu

OStartup();

ODatabase m\_database(“ExampleDB”,”scott”,”tiger”);

if (!m\_database.IsOpen()) {

printf("Connect error \n");

return 0;

}

/\* Kết nối thành công \*/

//Truy xuất dữ liệu của bảng EMP

ODynaset m\_dynaset;

m\_dynaset.Open(m\_database,

"select empno, ename from emp");

//In dữ liệu ra màn hình

OValue data;

m\_dynaset.GetFieldValue(1,&data);

printf("%s \n",(const char\*)data);

return 0;

}

# *Chương 21*

# Lập trình oracle oci với c++

**1. Oracle OCI là gì? Lập trình và truy xuất cơ sở dữ liệu Oracle bằng OCI có những ưu và khuyết điểm gì so với OO4O và ODBC.**

Lập trình truy xuất cơ sở dữ liệu Oracle bằng các hàm trong thư viện OCI nhanh hơn OO4O và ODBC (các giao tiếp OO4O và ODBC hầu hết đều gọi đến các hàm OCI) nhưng mã lệnh phức tạp và rất khó bảo trì (Xem thêm lý thuyết).

2. Một chương trình OCI cần phải có những thư viện gì khi biên dịch và thực thi?

Một chương trình OCI viết bằng C++ sau khi biên dịch ra tập tin thực thi .EXE cần phải có thêm thư viện các file DLL hỗ trợ thư viện OCI kèm theo (những thư viện này được cài đặt chung với phần mềm Oracle nếu bạn chọn Option hỗ trợ OCI trong quá trình cài đặt).

3. Handle và Descriptor là gì? Thư viện OCI sử dụng chúng như thế nào?

Xem lý thuyết.

4. Kết nối với cơ sở dữ liệu Oracle bằng các hàm trong thư viện OCI được thực hiện như thế nào? cho ví dụ.

Kết nối và làm việc với cơ sở dữ liệu Oracle thông qua các hàm trong thư viện OCI thường thực hiện những thao tác sau:

* Cấp phát các handle và Descriptor cần thiết.
* Kết nối với một cơ sở dữ liệu bằng hàm OCILogon (hoặc nhiều cơ sở dữ liệu bằng hàm OCIServerAttach).
* Mở cursor và thực thi lệnh SQL.
* Đóng cursor.
* Ngắt kết nối với cơ sở dữ liệu.
* Giải phóng Handle và các Decscriptor đã cấp phát.

Ví dụ:

*//Con trỏ đến các cấu trúc handle*

static OCIEnv \*envhp;

static OCIError \*errhp;

static OCIServer \*srvhp;

*//Chiều dài các biến chuỗi*

int namelen=5;

int passwdlen=5;

int dbnamelen=10;

*//Kết nối với cơ sở dữ liệu*

OCILogon(envhp, errhp, &srvhp,

”scott”, namelen,”tiger”, passwdlen,

”ExampleDB”,dbnamelen);

5. Điều khiển lệnh SQL hay khối lệnh PL/SQL được thực hiện như thế nào trong các chương trình OCI? cho ví dụ.

Hãy gọi hàm OCIStmtExecute() để thực thi lệnh SQL cũng như các khôí lệnh PL/SQL của Oracle (xem thêm lý thuyết).

# Chương Oracle Object Type

1. Kiểu đối tượng (Object type) là gì? So với các kiểu dữ liệu khác kiểu đối tượng có gì đặc biệt?

2. Hãy tạo đối tượng là một đơn đặt hàng PO (Purchase Order) với các thuộc tính sau:

item\_id NUMBER

line\_no NUMBER (3)

part\_no VARCHAR2 (20)

price NUMBER (15,5)

qty NUMBER (9,2)

received NUMBER (9,2)

Xây dựng các phương thức cho biết số lượng hàng cần đặt, số lượng hàng đã nhận …

3. Tạo một bảng (table) lưu đối tượng PO (đơn đặt hàng) theo cột.

4. Tạo một bảng (table) lưu đối tượng PO (đơn đặt hàng) theo dòng.

5. Thực hiện lệnh INSERT chèn các PO mới vào bảng, SELECT xuất các PO đã nhập ra màn hình, UPDATE cập nhật thông tin về PO, DELETE xóa các PO cũ …

6. So sánh hai đối tượng được thực hiện như thế nào? Phân biệt phương thức ánh xạ (map method) và phương thức phân ngôi (order method).

7. Tham chiếu và kết nối giữa các đối tượng từ hai bảng khác nhau được thực hiện như thế nào, cho ví dụ.

8. Có thể ràng buộc một thuộc tính của một đối tượng bằng các từ khóa NOT NULL, CHECK hoặc PRIMARY KEY hay không?

## Đáp án

1. Không như các kiểu dữ liệu khác, kiểu dữ liệu Object là một kiểu dữ liệu phức hợp nó có thể đóng gói những kiểu dữ liệu khác (được biểu diễn dưới dạng thuộc tính) và cả những phương thức để xử lý dữ liệu mà đối tượng đóng gói. Bạn có thể tạo ra một kiểu đối tượng và chỉ định áp dụng kiểu đối tượng cho một cột của bảng hoặc toàn bộ bảng. Việc truy xuất các đối tượng lưu trong bảng cũng thông qua các lệnh SQL thông thường. Sử dụng kiểu đối tượng bạn sẽ mô tả dữ liệu gần với thế giới thực hơn.

2. Kiểu đối tượng PO có thể được tạo ra như sau:

*/\* Phần đặc tả \*/*

CREATE TYPE po AS OBJECT(

item\_id NUMBER,

line\_no NUMBER (3),

part\_no VARCHAR2 (20),

price NUMBER (15,5),

qty NUMBER (9,2),

received NUMBER (9,2),

*-- Phương thức dùng để lấy về lượng hàng chưa nhận*

MEMBER FUNCTION Backlog RETURN NUMBER,

*-- Phương thức dùng để lấy về số tiền của PO*

MEMBER FUNCTION GetAmount NUMBER

RETURN NUMBER

);

*/\* phần thân của đối tượng \*/*

CREATE TYPE BODY po AS

MEMBER FUNCTION Backlog RETURN NUMBER

IS

BEGIN

RETURN (qty-received)

END Backlog;

MEMBER FUNCTION GetAmount RETURN NUMBER

IS

BEGIN

RETURN (qty\*price)

END GetAmount;

END;

3. Tạo một bảng (table) lưu đối tượng PO theo cột:

CREATE TABLE ALLPO(d DATE,SaleOrder PO);

Bảng ALLPO sẽ được tạo ra với nội dung như sau:

SQL> DESC ALLPO

…

4. Tạo một bảng (table) lưu đối tượng PO theo dòng:

CREATE TABLE ALLPO OF PO;

Bảng ALLPO sẽ được tạo ra với nội dung như sau:

SQL> DESC ALLPO

…

5. Chèn PO (đơn đặt hàng) mới vào bảng ALLPO

6. Xem lý thuyết.

7,8. Xem lý thuyết.

# *cHƯƠNG 26*

ORACLE và cấu hình mạng   
(Oracle network configuration)

1. Listener là gì? Các Service chính nào được dùng để phục cho việc truy xuất cơ sở dữ liệu Oracle?

Listener đóng vai trò tiếp nhận các kết nối từ máy client gửi đến Server, muốn máy client và Server kết nối được với nhau bạn phải START (hay khởi động) chương trình listener trên Server. Một khi chương trình listener đã hoạt động kết nối từ máy client của bạn gửi đến đúng địa chỉ máy Server sẽ được chuyển cho listener kiểm tra nếu đạt yêu cầu máy Server sẽ chấp nhận cho client thực hiện nối kết vào các dịch vụ của cơ sở dữ liệu.

Listener chỉ là chương trình thực hiện kết nối, bạn phải khởi động thêm các Service khác của cơ sở dữ liệu, thường mỗi cơ sở dữ liệu sẽ có hai Service cần phải khởi động là Oracle<instanceName>Service và Oracle<SID>Startup.

2. Net Service Name là gì? Có mấy cách cấu hình Net Service Name?

Để kết nối với Server máy client sẽ dùng *Net Service Name* là một chuỗi thông tin được lấy từ tập tin TNSNAMES.ORA (trên máy client cục bộ tập tin này thường được Oracle lưu trong thư mục *ORACLE\_HOME\Net80\Admin*) hoặc trên một máy Server khác (Oracle Names Server) trong trường hợp bạn có nhiều cơ sở dữ liệu và cần truy xuất trên mạng diện rộng với quy mô cơ sở dữ liệu phân tán lớn. Khi cấu hình cho máy client (hay Server đóng vai trò client) chúng ta cần phải quan tâm đến những thông số kết nối này.

Tổng quát ta có hai mô hình thiết lập cấu hình cho *Net Service Name* ở máy client là:

* Cấu hình quản lý cục bộ (Localized management)
* Cấu hình quản lý tập trung (Centralized management)

3. Cấu hình để kết nối giữa máy client và cơ sở dữ liệu trên Server được thực hiện như thế nào? Giả sử Server của bạn có địa chỉ IP là 172.16.11.10 cho ví dụ về cách cấu hình file TNSNAMES.ORA trên máy client và LISTENER.ORA trên máy Server.

#### A. Cấu hình Server

Cấu hình cho service listener (file LISTENER.ORA) như sau:

LISTENER =

(ADDRESS\_LIST =

(ADDRESS = (PROTOCOL = TCP)

(HOST = 172.16.11.12)

(PORT = 1521))

(ADDRESS = (PROTOCOL = TCP)

(HOST = 127.0.0.1)

(PORT = 1521))

)

SID\_LIST\_LISTENER =

(SID\_LIST =

(SID\_DESC =

(SID\_NAME = ORCL)

)

)

**B. Cấu hình client**

Thêm vào file TNSNAMES.ORA nội dung sau:

EXAMPLEDB.WORLD =

(DESCRIPTION =

(ADDRESS =

*(PROTOCOL = TCP)*

*(HOST = 172.16.11.12)*

*(PORT = 1521)*

)

(CONNECT\_DATA = (*SID = ORCL*))

)

4. Oracle Name Server (ONS) là gì? Cấu hình cho ONS được thực hiện như thế nào (giả sử máy ONS của bạn có địa chỉ IP là 172.16.11.2)?

ONS là một chương trình (hay Name Service) chạy riêng trên một máy (được gọi là ONS Server). Các listener hoạt động trên máy chủ chứa cơ sở dữ liệu khi khởi động sẽ tự tìm đến máy ONS để đăng ký các thông tin cần thiết Xem lý thuyết về cấu hình ONS

5. Làm thế nào để chương trình listener trên server (chứa cơ sở dữ liệu Oracle) có thể tự động đăng ký các dịch vụ giao tiếp với máy ONS?

Xem chi tiết cấu hình ONS ở mục 6 (chương cấu hình mạng).

**6. Bộ quản lý kết nối Oracle Connection Manager (OCM) được dùng để làm gì? Cách cấu hình và thiết lập proxy hạn chế các máy client truy xuất vào cơ sở dữ liệu.**

Oracle 8 cung cấp mô hình kết nối tập trung thông qua chương trình quản lý Oracle Connection Manager. Oracle Connection Manager (OCM) được cài trên một máy riêng biệt, giả sử bạn có nhiều cơ sở dữ liệu đặt trên các máy chủ khác nhau, các máy client khi kết nối với cơ sở dữ liệu đều phải thông qua máy quản lý OCM này. OCM sẽ đóng vai trò như một cầu nối giữa máy client với các máy chủ chứa cơ sở dữ liệu. Bạn có thể cấu hình và sử dụng OCM như một Proxy dùng để cho phép hay cấm máy client truy xuất vào một cơ sở dữ liệu nào đó. Xem thêm cấu hình cho OCM ở mục 7 (chương Cấu hình mạng).