**Bài tập lý thuyết về SQL:**

1. Có mấy loại khoá của bảng? Định nghĩa và ý nghĩa của từng loại? Một bảng có thể có nhiều loại khoá không? Một bảng có thể có nhiều khoá cùng 1 loại hay không? Các giá trị của khoá có thể là NULL hay không?
   * Các loại khóa:

|  |  |  |
| --- | --- | --- |
| Loại khóa | Định nghĩa | Ỹ nghĩa |
| Primary key  (Khóa chính) | * Là một column hoặc một tập các column. | * Xác định tính duy nhất của các rows ở trong bảng. |
| Foreign key  (Khóa ngoại) | * Là một column hoặc một tập các columns có tham chiếu tới chính bảng đó hoặc một bảng khác. | * Xác định mối quan hệ giữa các bảng. |

* + Một bảng có thể có nhiều loại khóa.
  + Trong bảng chỉ có duy nhất một khóa chính, và có thể có nhiều khóa ngoại.
  + Các giá trị của khóa không thể nhận giá trị NULL hay các giá trị không xác định.

1. Có mấy loại Join, ý nghĩa của từng loại, lấy ví dụ minh hoạ.

Các loại JOIN

|  |  |  |
| --- | --- | --- |
| Các loại | Ý nghĩa | Ví dụ |
| CROSS JOIN | * Nối chéo dữ liệu các bảng với nhau. * Cú pháp:   SELECT table1.column, table2.column  FROM table1  CROSS JOIN table2; | SELECT E**.**ENAME**,** D**.**DNAME  FROM EMP E CROSS JOIN DEPT D; |
| NATURAL JOIN | * Phép nối tạo một mệnh đề tham gia ngầm dựa trên các cột chung trong hai bảng. Mặc định tham gia như inner * Cú pháp:   SELECT table1.column, table2.column  FROM table1  NATURAL JOIN table2; | SELECT E**.**ENAME**,** D**.**DNAME  FROM EMP E NATURAL JOIN DEPT D; |
| LEFT OUTER JOIN | * Nối hai bảng A(trái) và B(phải). Trả về tất cả giá trị bảng A và giá trị bảng B tương ứng. Nếu bảng B không có giá trị tương ứng sẽ trả về giá trị NULL. * Cú pháp:   SELECT table1.column, table2.column  FROM table1  LEFT OUTER JOIN table2  ON (table1.column = table2.column); | SELECT E**.**ENAME**,** D**.**DNAME  FROM EMP E LEFT OUTER JOIN DEPT D ON **(**E**.**DEPTNO **=** D**.**DEPTNO**)**; |
| RIGHT OUTER JOIN | * Nối hai bảng A(trái) và B(phải). Trả về tất cả giá trị bảng B và giá trị bảng A tương ứng. Nếu bảng A không có giá trị tương ứng sẽ trả về giá trị NULL. * Cú pháp:   SELECT columns  FROM table1  RIGHT [OUTER] JOIN table2  ON table1.column = table2.column; | SELECT E**.**ENAME**,** D**.**DNAME  FROM EMP E RIGHT OUTER JOIN DEPT D ON **(**E**.**DEPTNO **=** D**.**DEPTNO**)**; |
| FULL OUTER JOIN | * Nối dữ liệu 2 bảng A và B. Trả về tất cả dữ liệu từ hai bảng, trả về giá trị NULL ở vị trí điều kiện nối không đáp ứng. * Cú pháp:   SELECT columns  FROM table1  FULL [OUTER] JOIN table2  ON table1.column = table2.column; | SELECT E**.**ENAME**,** D**.**DNAME  FROM EMP E FULL OUTER JOIN DEPT D ON **(**E**.**DEPTNO **=** D**.**DEPTNO**)**; |

1. Tìm hiểu về từ khoá UNION trong truy vấn dữ liệu? Có mấy loại UNION, cách sử dụng và ví dụ minh hoạ cho từng loại.

* UNION: Kết hợp kết quả của nhiều câu hỏi với nhau, chỉ giữ lại một đại diện cho các mẫu tin trùng nhau.
* UNOIN ALL: Kết hợp kết quả của nhiều câu hỏi với nhau, các mẫu tin trùng nhau cũng được lặp lại
* Các loại UNION

|  |  |  |
| --- | --- | --- |
| Các loại | Cách sử dụng | Ví dụ |
| UNION | SELECT tên cột FROM bảng1  UNION  SELECT tên cột FROM bảng2 | Select job from emp where deptno **=** 10  Union  Select job from emp where deptno **=** 30; |
| UNION ALL | SELECT tên cột FROM bảng1  UNION ALL  SELECT tên cột FROM bảng2 | Select job from emp where deptno **=** 10  Union ALL  Select job from emp where deptno **=** 30; |

1. Tìm hiểu về Nested Query, Correlated Query. Phân biệt, lấy ví dụ minh hoạ.

|  |  |  |
| --- | --- | --- |
|  |  | Ví dụ |
| Nested Query | Truy vấn bên trong chạy đầu tiên, và chỉ một lần. Truy vấn bên ngoài được thực thi với kết quả từ truy vấn bên trong. Truy vấn bên trong được sử dụng trong việc thực hiện truy vấn bên ngoài. | select ename from emp  where job in**(**select job from emp where job like 'SALEMAN'**)** |
| Correlated Query | Truy vấn bên ngoài thực hiện đầu tiên và cho mỗi truy vấn truy vấn bên ngoài được thực thi. Do đó, truy vấn bên trong sử dụng các giá trị từ truy vấn bên ngoài. | select ename from emp  where exists**(**select job from emp where job like 'SALEMAN'**)** |

1. Tìm hiểu về Index? Phân biệt global index và local index? Khi viết câu lệnh truy vấn, muốn các index được sử dụng phải chú ý gì? Lấy ví dụ minh hoạ.

* Là một cấu trúc dữ liệu để tăng hiệu suất truy vấn của cơ sở dữ liệu. Index cho phép cơ sở dữ liệu thực hiện một số câu truy vấn có điều kiện nhanh hơn so với thông thường.

|  |  |
| --- | --- |
| Global index | * Chỉ mục toàn cục là mối quan hệ một-nhiều, cho phép một phân vùng chỉ mục ánh xạ tới nhiều phân vùng bảng. * Có thể được xác định trên bất kỳ loại phân vùng hoặc không phân vùng, bảng |
| Local index | * Chỉ mục cục bộ là ánh xạ 1-1 giữa phân vùng chỉ mục và phân vùng bảng. |

* Muốn các index được sử dụng phải chú ý:
* Khi mà việc lấy dữ liệu <15% số row trong bảng.
* Index column nào dùng để nối giữa các bảng lẫn nhau.
* Không nên dung Index cho các bảng nào chỉ có vài row.
* Primaryvà unique key (khóa chính và khóa duy nhất) tự động có index, nhưng nên có index cho foreign key (khóa ngoại).
* Quá nhiều index sẽ gây ảnh hưởng đến số liệu khi mà dữ liệu trong table bị thay đổi thứ tự theo index

Ví dụ:

CREATE INDEX i-ENAME ON EMP (ENAME);

1. Tìm hiểu về partition của bảng, nên tạo partition cho bảng trong trường hợp nào? Lấy ví dụ minh hoạ?

* Khái niệm: là việc phân chia một table thành những phần nhỏ theo một logic nhất định, được phân biệt bằng key, key này thường là tên column trong table.
* Trường hợp tạo partition cho bảng:
* Bảng lớn hơn 2 GB luôn được coi là ứng viên cho phân vùng.
* Bảng chứa dữ liệu lịch sử, những dữ liệu mới sẽ được thêm vào phân vùng mới.
* Khi nội dung của bảng cần được phân phối trên nhiều loại thiết bị storage khác nhau.

Ví dụ:

create table PARTITION\_ASSIGNMENTS\_HUYEN

**(**

PROJID NUMBER**(**4**)** not null CONSTRAINT FK\_ASSIGNMENTS\_HUYEN REFERENCES SCOTT**.**PROJECT\_HUYEN**(**PROJID**) ,**

EMPNO number **(**4 **)**not null**,**

A\_START\_DATE date **,**

A\_END\_DATE date **,**

BILL\_RATE NUMBER**(**4**,**2**),**

ASSIGN\_TYPE varchar2 **(**2**)**

**)**

PARTITION BY RANGE**(**PROJID**)**

**(**

PARTITION p1 VALUES LESS THAN**(**2**)**

**)**

1. Phân biệt câu lệnh delete và truncate? Cách sử dụng?

|  |  |  |
| --- | --- | --- |
|  |  | Cách sử dụng |
| DELETE | Xóa một hoặc tất cả các hàng từ một bảng dựa trên điều kiện và có thể được phục hồi lại. | DELETE FROM table\_name   WHERE condition; |
| TRUNCATE | Xóa tất cả các hàng từ một bảng bằng cách phân bổ các trang bộ nhớ và không thể phục hồi lại. | TRUNCATE TABLE table\_name |