**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?

+ Có 2 loại khóa :

* Primary key: Khóa chính
* Foreign key: Khóa ngoại

|  |  |  |
| --- | --- | --- |
|  | Primary key | Foreign key |
| Định nghĩa và ý nghĩa | * Khóa chính (hay ràng buộc khóa chính) được sử dụng để định danh duy nhất mỗi record trong table của cơ sở dữ liệu. * Ngoài ra, nó còn dùng để thiết lập quan hệ 1-n (hay ràng buộc tham chiếu) giữa hai table trong cơ sở dữ liệu. | * Khóa ngoại của một table được xem như con trỏ trỏ tới khóa chính của table khác. |
| Một bảng có thể có nhiều loại khoá không? | Một bảng có thể có nhiều loại khóa | |
| Một bảng có thể có nhiều khoá cùng 1 loại hay không | Chỉ có 1 khóa chính | Có thể có nhiều khóa ngoại |
| Các giá trị của khoá có thể là NULL hay không? | NOT NULL | NOT NULL |

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

- Có 4 loai Join : INNER JOIN, LEFT JOIN, RIGHT JOINV

|  |  |  |
| --- | --- | --- |
|  | Ý nghĩa | Ví dụ |
| INNER JOIN | Trả về các bản ghi có những giá trị phù hợp trong cả hai bảng. | SELECT E.ENAME, D.DNAME FROM EMP E INNER JOIN DEPT D ON (E.DEPTNO = D.DEPTNO); |
| LEFT JOIN | Trả về tất cả bản ghi từ bảng bên trái và bản ghi trùng với bảng bên phải. | SELECT E.ENAME, D.DNAME FROM EMP E LEFT OUTER JOIN DEPT D ON (E.DEPTNO = D.DEPTNO); |
| RIGHT JOIN | Trả về tất cả bản ghi từ bảng bên phải và bản ghi trùng với bản bên trái. | SELECT E.ENAME, D.DNAME FROM EMP E RIGHT OUTER JOIN DEPT D ON (E.DEPTNO= D.DEPTNO); |
| FULL JOIN | Trả về tất cả bản ghi khi có một kết quả phù hợp trong bảng bên trái hoặc bên phải. | SELECT E.ENAME, D.DNAME FROM EMP E FULL OUTER JOIN DEPT D ON (E.DEPTNO = D.DEPTNO); |

3.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.

|  |  |  |
| --- | --- | --- |
|  | Ý nghĩa | Ví dụ |
| UNION | Kết hợp kết quả của nhiều câu hỏi với nhau, chỉ giữ lại một đại diện cho các mẫu tin trùng nhau. | Select job from emp where deptno **=** 10  Union  Select job from emp where deptno **=** 30; |
| UNION ALL | Kết hợp kết quả của nhiều câu hỏi với nhau, các mẫu tin trùng nhau  cũng được lặp lại | Select job from emp where deptno **=** 10  Union all  Select job from emp where deptno **=** 30; |

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

|  |  |  |
| --- | --- | --- |
|  | Phân biệt | Ví dụ |
| Nested Query  (Truy vấn lồng nhau) | Trong truy vấn lồng nhau, Truy vấn bên trong chạy trước và chỉ một lần. Truy vấn bên ngoài được thực hiện với kết quả từ truy vấn Bên trong, do đó, truy vấn Bên trong được sử dụng để thực hiện truy vấn Bên ngoài. | SELECT DEPTNO**,** DNAME  FROM DEPT  WHERE DEPTNO NOT IN **(**SELECT DEPTNO FROM EMP**)** |
| Correlated Query | Trong Truy vấn Tương quan, Truy vấn Bên ngoài thực hiện đầu tiên và đối với mọi hàng Truy vấn Bên ngoài Truy vấn Bên trong được thực thi. Do đó, truy vấn Bên trong sử dụng các giá trị từ truy vấn Bên ngoài. | SELECT DEPTNO**,** DNAME  FROM DEPT D  WHERE not exists **(**SELECT **\*** FROM EMP E  WHERE D**.**DEPTNO **=** E**.**DEPTNO**)** |

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.

-Index là một cấu trúc cơ sở dữ liệu, được sever sử dụng để tìm một row trong bảng một

cách nhanh chóng. Index bao gồm một key value (một cột (column) trong hàng (row)) và ROWID.

|  |  |
| --- | --- |
| Global index | Local index |
| Chỉ mục được phân vùng toàn cục được sử dụng cho tất cả các chỉ mục khác ngoại trừ chỉ mục được sử dụng làm khóa phân vùng bảng. Các ứng dụng OLTP (xử lý giao dịch trực tuyến) phân vùng chỉ mục toàn cầu trong đó yêu cầu ít đầu dò chỉ mục hơn so với các chỉ mục được phân vùng cục bộ. Trong lược đồ phân vùng chỉ mục chung, chỉ mục khó duy trì hơn vì chỉ mục có thể kéo dài các phân vùng trong bảng cơ sở. | Các chỉ mục được phân vùng cục bộ cho phép DBA lấy các phân vùng riêng lẻ của bảng và các chỉ mục ngoại tuyến để bảo trì (hoặc tổ chức lại) mà không ảnh hưởng đến các phân vùng và chỉ mục khác trong bảng.  Trong chỉ mục được phân vùng cục bộ, các giá trị khóa và số lượng phân vùng chỉ mục sẽ khớp với số lượng phân vùng trong bảng cơ sở. |
| CREATE INDEX item\_idx trên all\_fact (item\_nbr) TOÀN CẦU (PARTITION city\_idx1 VALUES ÍT HƠN (100)), (PARTITION city\_idx1 VALUES ÍT HƠN (200)), (PARTITION city\_idx1 VALUES ÍT HƠN (300)), (PARTITION city\_idx1 VALUES ÍT HƠN (400 )) )), (PARTITION city\_idx1 GIÁ TRỊ ÍT HƠN (500)); | CREATE INDEX year\_idx trên all\_fact (order\_date) LOCAL (PARTITION name\_idx1), (PARTITION name\_idx2), (PARTITION name\_idx3); |

+Lưu ý:

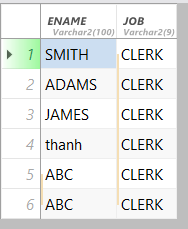
* Khi lựa chọn index, nên xem xét xem liệu phần performance đạt được khi query có đáng với phần performance mất đi khi INSERT, UPDATE, DELETE, và phần tài nguyên phải dùng để lưu trữ index. Bạn nên so sánh perfomance giữa trường trường hợp có/không có index để đưa ra được phương án tốt nhất.
* Khi lựa chọn composite index (index được tạo nên bởi 2 hoặc nhiều column khác nhau), tổ hợp của các column có selectivity thấp có thể tạo nên index có selectivity cao. Ngoài ra cũng nên xem xét lựa chọn các column thường được chọn làm điều kiện query ở phần WHERE, hoặc các column dùng để JOIN, để làm composite key.

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

- 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.

- Tạo partition cho bảng trong trường hợp

* Table có dung lượng từ 2GB trở lên nên cân nhắc được partition.
* Table có chứa dữ liệu lịch sử. Dữ liệu cũ, ít được sử dụng sẽ đưa vào các partition cũ, những dữ liệu mới sẽ được đưa vào partition mới.
* Table có nội dung có thể được chia thành các khu vực có mức độ ưu tiên khác nhau (VD: khu vực dành cho dữ liệu thường xuyên được truy vấn,hoặc ít được truy vấn, …)

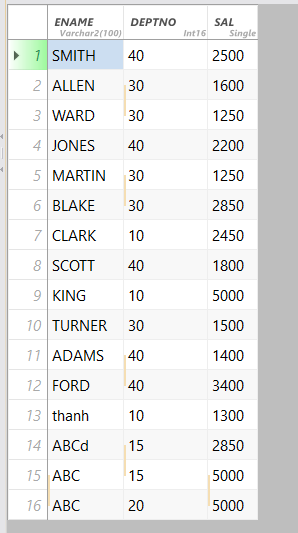
-Ví dụ

* TH tạo partition

SELECT ENAME**,** JOB

FROM EMP

WHERE JOB **=** 'CLERK'

* TH không nên tạo partition

SELECT ENAME**,** DEPTNO **,**SAL

FROM EMP

Where SAL**>** 1000 ;

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

|  |  |
| --- | --- |
| Delete | Truncate |
| Xóa bỏ các bản ghi – **Record**khỏi cơ sở dữ liệu theo hàng – **Row** | sẽ thực hiện việc phân bổ lại trang chứa dữ liệu trong bảng, và toàn bộ phần dữ liệu đó sẽ bị xóa bỏ hoàn toàn sau khi thực hiện |
| Những bản ghi nào đó vô tình bị xóa bỏ thì sẽ khôi phục lại được | những bản ghi nào đó vô tình bị xóa bỏ thì sẽ không thể khôi phục lại được |
| Không thay đổi việc tự động tăng hoặc giảm số lượng cột – **Column**trong bảng chứa dữ liệu đó. | Nếu 1 bảng dữ liệu sau khi thực hiện **TRUNCATE**có tính năng tăng số lượng cột thì sẽ tự động được sắp xếp lại theo phần định nghĩa ban đầu |