문장 실행 순서

1. FROM > WHERE > GROUP BY > HAVING > SELECT > ORDER BY

DDL(CRATE, ALTER, DROP, RENAME, TRUNCATE)

1. CRATE
   A. CREATE TABLE 테이블이름 (속성이름 자료형 제약조건, 반복) ;
      i. CONSTRAINT : 기본키를 지정할 때 사용
         - CONSTRAINT 제약조건이름 PRIMARY KEY (PK할속성이름)
   B. CREATE INDEX 인덱스이름 ON 테이블이름 (속성이름) ;
2. ALTER
   A. ALTER TABLE 테이블이름 ALTER COLUMN 속성이름 변경자료형 변경제약조건 ;
   B. ALTER TABLE 테이블이름 ADD 추가속성이름 추가제약조건 ;
   C. ALTER TABLE 테이블이름 ADD CONSTRAINT 그냥이름 PRIMARY KEY (PK할속성이름) ;
3. DROP
   A. DROP TABLE 테이블이름 ;
   B. DROP COLUMN 속성이름 ;
4. RENAME
   A. RENAME 테이블이름 TO 변경테이블이름 ;
5. TRUNCATE
   A. TRUNCATE
   B. TRUNCATE TABLE 테이블이름 ; / 테이블의 모든 데이터를 삭제하는 명령문
   C. DROP과 차이점 = DROP은 테이블을 완전 삭제, TRUNCATE은 테이블을 완전초기상태로 되돌림
   D. TRUNCATE TABLE 테이블이름 ; / 테이블의 모든 데이터를 삭제하는 명령문

DML(SELECT, INSERT, DELETE, UPDATE, MERGE)

1. SELECT
    A. SELECT * FROM 테이블이름 ;
2. INSERT
    A. INSERT INTO 테이블이름 (속성이름1, 속성이름2, 속성이름3) VALUES (튜플1, 튜플2, 튜플3);
3. DELETE
    A. DELETE FROM 테이블이름 ;
4. UPDATE
    A. UPDATE 테이블이름 SET 속성이름 = 튜플 ;
5. MERGE

DCL(GRANT, REVOKE)

1. ROLE
    A. DBMS 사용자를 생성하면 기본적으로 많은 권한을 부여해야 한다
    B. 많은 DBMS에서는 DMBS 관리자가 사용자 별로 권한을 관리해야 하는 부담과 복잡함을 줄이기 위하여 다양한 권한을 그룹으로 묶어 관리할 수 있도록 사용자와 권한 사이에서 중개 역할을 수행
2. GRANT
    A. GRANT 권한1, 권한2 ... ON 테이블 TO 사용자 ;
    B. GRANT 권한1, 권한2 ... TO 사용자 ;
    C. ROLE을 DBMS USER에게 부여하기 위해서 사용
3. REVOKE
    A. REVOKE 권한1, 권한2, ... ON 객체 FROM 사용자 ;
    B. REVOKE 권한1, 권한2, ... FROM 사용자 ;
    C. ROLE을 회수하기 위해서 사용

TCL(COMMIT, ROLLBACK, SavePoin)

1. COMMIT
2. ROLLBACK
3. SavePoint

AS

1. SELECT
    A. AS 생략 가능
    B. 컬럼 명에 띄어쓰기 "변경속성"
2. FROM
    A. AS 사용불가

VIEW(독립성, 편리성, 보안성)

1. 단지 정의만을 가지고 있으며, 실행 시점에 질의를 재작성하여 수행
2. 보안을 강화하기 위한 목적으로 활용할 수 있다.
3. 실제 데이터를 저장하고 있는 VIEW를 생성하는 기능을 지원하는 DBMS도 있다.

옵션

CONCAT

1. (인수1, 인수2) : 인수가 무조건 2개임 (3개 안됨)
2. + : SQL Server
3. || : Oracle

속성 BETWEEN 범위1 AND 범위2 : 날짜1 <= 날짜속성 <= 날짜2

LIKE

1. _ : 단일 임의의 문자를 나타냄
2. % : 0개 이상의 임의의 문자열을 나타냄

LIKE '김!%' ESCAOE '!' -> 김!민

속성 IN (1, 2, 3) -> 속성=1 OR 속성=2 OR 속성=3

ROWNUM : Oracle

1. WHERE RWONUM <= N : 1 ~ N (1포함) : ORDER BY 전에 테이블 상위 N개의 행을 반환

TOP : SQL Server

1. (SELECT) TOP (N) (컬럼명) : 상위속성 N개

CASE WHEN 조건1 THEN "값1" WHEN 조건2 THEN "값2" ELSE "모두 아니면 값"

GROUP BY

1. 집약기능
2. 그룹수

JOIN

1. NATNANL JOIN
    A. 중복된 컬럼 하나
2. USING JOIN
    A. 중복된 컬럼 하나
3. LEFT OUTER JOIN

ORDER BY

1. 가장 마지막에 실행
2. 성능이 느려질 가능성
3. NULL 값과의 관계

컬럼번호 정렬

1. 출력되는 컬럼의 수보다 큰값은 불가능

인수 2개 정렬

1. 인수1이 같으면 인수2의 순서에 따른다

논리 연산자

1. AND : A and B
2. OR : A or B
3. NOT : not A, B

논리 연산자 순서

1. NOT > AND > OR

집합 연산자

1. UNION 합집합 : 결과에서 모든 중복된 행은 하나의 행으로 만든다 / 정렬 발생
2. UNION ALL 합집합 : 중복된 행도 그대로 결과를 표시 / 정렬 안 함 / 빠르다
3. INTERSECT 교집합 : 중복된 행은 하나의 행으로 만든다
4. EXCEPT / MINUS 차집합 : 중복된 행은 하나의 행으로 만든다 / NOT IN, NOT EXISTS 대체 가능

NULL

1. 값의 부재
2. 모르는 값
3. NULL과 모든 비교(IS NULL 제외)는 알 수 없음(Unknown)을 반환
4. Oracle : NULL값을 가장 큰 값으로 간주하여 오름차순으로 정렬했을 경우에는 가장 마지막에 출력
5. SQL Server : NULL값을 가장 작은 값으로 간주하여 오름차순으로 정렬했을 경우 가장 먼저 출력

NVL (표현식, 대체값) : 널뛰기 : 표현식이 NULL인 경우 반환될 대체값

NVL2 (값1, 값2, 값3) : 널뛰기 : 값1 IS NULL 값3, IS NOT NULL 값2

IS NULL (값1, 값2) : 널뛰기 : 값1 IS NULL = 값2, != 값1

NULL IF (값1, 값2) : 같이 놀자 : 값1, 값2 같으면 NULL, 다르면 값1

COALESCE (값1, 값2, ...) : 널 아닌 첫번째 값

함수

숫자 함수

1. ROUND (실수형, 원하는 반올림 위치) : 반올림
2. Ceil : Oracle
3. Ceiling : SQL Server

문자 함수

1. UPPER : 대문자로 변환
2. LOWER : 소문자로 변환
3. LPAD : 지정한 길이 만큼 왼쪽부터 특정 문자로 채워준다
4. RPAD : 지정한 길이 만큼 오른쪽부터 특정 문자로 채워준다
5. LTRIM : 문자의 앞 공백 제거
6. RTRIM : 문자의 뒤 공백 제거
7. SUBSTR : 문자의 일부분을 추출
8. INSTR : 특정문자가 출현하는 위치를 알려준다

날짜 함수

1. TO_CHAR : 날짜 값을 문자로 변환
2. TO_DATE : 문자를 날짜 값으로 변환
3. SYSDATE : Oracle에서 현재시간 반환
4. GETDATE ( ) : SQL Server
5. 날짜 데이터 + day

집계 함수

1. NULL 관계

그룹 함수

1. ROLL UP
    A. 인수의 위치가 다르면 결과가 다르다

B. 총합 속성 있음 (행의 수가 적다)

2. CUBE

A. 인수의 위치가 달라도 결과는 같다

B. 총합 속성 있음 (행의 수가 많다)

3. GROUPING SETS

A. 총합 속성 없음

윈도우 함수

1. ROWS : 동일한 값 != 동일한 순위, = 고유한 순위
2. RANGE
3. RANK : 중복 건너 뛴다 : 1, 1, 3, 4 ...
4. DEASE RANK : 중복 건너 뛰지 않는다 : 1, 1, 2, 3 ...

서브쿼리

1. SELCT : SCALAR
2. FROM : Inline View, Dynamin View
3. WHERE : 중첩 서브쿼리
4. GROUP BY : 안들어감
5. HAVING : 중첩 서브쿼리
6. ORDER BY : SCALAR

NL Join (Nested Loop Join) 특징

1. 조인 칼럼에 적당한 인덱스가 있어서 자연조인이 효율적일 때 유용
2. Driving Table의 조인 데이터 양이 큰 영향을 주는 조인 방식이다
3. 유니크 인덱스를 활용하여 수행시간이 적게 걸리는 소량 테이블을 온라인 조회하는 경우 유용
4. 선택도가 낮은 테이블이 선행 테이블로 선택되는 것이 일반적

SMJ (Sort Merge Join) 특징

1. 조인 칼럼에 적당한 인덱스가 없어서 NL Join가 비효율적일 때 사용할 수 있다
2. Driving Table의 개념이 중요하지 않는 조인 방식
3. 조인 조건의 인덱스의 유무에 영향을 받지 않는다
4. ~~비 동등에서도 사용할 수 있다~~

해시조인 Hash Join

1. 조인 컬럼에 적당한 인덱스가 없어서 자연조인이 비효율적일 때
2. 자연 조인 시 드라이빙 집합 쪽으로 조인 액세스 량이 많아 Random 액세스 부하가 심할 때
3. 소트 머치 조인을 하기에는 두 테이블이 너무 커서 소트 부하가 심할 때
4. ~~행의 수가 작은 테이블을 선행 테이블로 선택하는 것이 유리~~
5. ~~SMJ보다 일반적으로 더 우수한 성능을 보임~~
6. ~~Join 대상 테이블이 Join Key 컬럼으로 정렬되어 있을 때는 SMJ가 더 우수한 성능을 낼 수 있다.~~

계층형 질의

1. Prior 자식데이터 = 부모데이터
2. 부모 -> 자식 가는 경우 순방향

절차형 PL/SQL

1. 변수와 상수 등을 사용하여 일반 SQL 문장을 실행할 때 WHERE절의 조건 등으로 대입할 수 있다
2. Procedure, User Defined Function, Trigger 객체를 PL/SQL로 작성 가능
3. Procedure 내부에 작성된 절차적 코드는 PL/SQL엔진이 처리
4. 일반적인 SQL 문장은 SQL실행기가 처리

SQL 모듈

1. 저장형 프로시져는 SQL을 로직과 함께 데이터베이스 내에 저장해 놓은 명령문의 집합
2. 저장형 함수(사용자 정의 함수)는 단독적으로 실행되기 보다는 다른 SQL문을 통하여 호출되고 그 결과를 리턴하는 SQL의 보조적인 역할을 한다
3. 트리거는 특정한 테이블에 INSERT, UPDATE, DELETE와 같은 DML문이 수행되었을 때 데이터베이스에서 자동으로 동작하도록 작성된 프로그램이다.

Trigger 트리거

1. 데이터베이스에 의해서 자동으로 호출되고 수행
2. 특정 테이블에 대해서 INSERT, UPDATE, DELETE문이 수행되었을 때 호출되도록 정의
3. 데이터베이스에 로그인하는 작업에도 정의
4. TCL 사용 불가

데이터모델링

1. 업무를 데이터모델화 시킨다

엔터티 Entity

1. 속성이 없는 엔터티는 있을 수 없다. (엔터티는 반드시 속성을 가져야 한다)
2. 엔터티는 다른 엔터티와 관계가 있을 수 밖에 없다. (단, 통계성 코드성 엔터티의 경우 생략 가능)
3. 데이터로서 존재하지만 업무에서 필요로 하지 않으면 해당 업무의 엔터티로 성립될 수 없다.
4. ~~반드시 해당 업무에서 필요하고 관리하고자 하는 정보이어야 한다.~~
5. 유일한 식별자에 의해 식별이 가능해야 한다
6. ~~영속적으로 존재하는 (2개 이상의)인스턴스의 집합이어야 한다~~
7. 엔터티는 업무 프로세스에 의해 이용되어야 한다

기본 엔터티(키엔터티)

1. 다른 엔터티로부터 주식별자를 상속받지 않고 자신의 고유한 주식별자를 가진다

엔터티의 이름을 부여하는 방법

1. 현업의 업무 용어를 사용하여 업무상의 의미를 분명하게 한다
2. 모든 엔터티에서 유일한 이름을 부여
3. 엔터티가 생성되는 의미대로 자연스럽게 부여

속성 (ATTRIBUTE)

1. 업무에서 필요로 하는 인스턴스에 관리하고자 하는 의미상 더 이상 분리되지 않는 최소의 데이터 단위

속성의 특성에 따른 분류

2. 기본속성 : ~~기본적인 베이스~~
3. 설계속성 : ~~계산된 속성~~
4. 파생속성 : ~~정의된 속성~~

도메인

1. 가질 수 있는 값의 범위
2. 엔터티 내에서 속성에 대한 데이터타입과 크기, 제약사항을 지정

주식별자의 특징

1. 유일성 : 주식별자에 의해 엔터티내에 모든 인스턴스들을 유일하게 구분
2. 최소성 : 주식별자를 구성하는 속성의 수는 유일성을 만족하는 최소의 수가 되어야 함
3. 불변성 : 주식별자가 한 번 특정 엔터티에 지정되면 그 식별자의 값은 변하지 않아야 함
4. 존재성 : 주식별자가 지정되면 반드시 데이터 값이 존재 (Null 안됨)

실행계획

읽는 순서

1. 안에서 밖으로
2. 위에서 아래로

인덱스 INDEX

1. 기본 인덱스에 널 값들이 나타날 수 없다
2. 테이블의 전체 데이터를 읽는 경우는 인덱스가 거의 불필요
3. B트리는 관계형 데이터베이스의 주요 인덱스 구조이다.
4. 규칙기반 옵티마이저는 적절한 인덱스가 존재하면 항상 인덱스를 사용하려고 한다
5. 인덱스 범위 스캔은 결과가 없으면 한 건도 반환하지 않을 수 있다
6. 인덱스의 목적은 조회 성능을 최적화
7. INSERT, UPDATE, DELETE 등의 DML 처리 성능을 저하시킬 수도 있다 (부하를 줄 수 있다)
8. INSERT, DELETE와 다르게 UPDATE는 부하가 없을 수 있다
9. B-트리 인덱스는 일치 및 범위 검색에 적절한 구조

B-TREE 인덱스 :

1. 브랜치 블록과 리프 블록으로 구성
2. 브랜치 블록 = 분기를 목적
3. 리프블록 = 인덱스를 구성하는 컬럼의 값으로 정렬
4. OLTP 시스템에서 가장 많이 사용

CLUSTERED 인덱스 :

1. 인덱스의 리프 페이지 = 데이터 페이지
2. 리프 페이지의 모든 데이터는 인덱스 키 컬럼 순으로 물리적으로 정렬되어 저장

BITMAP 인덱스 :

1. DW 및 AD-HOC 질의 환경을 위해서 설계
2. 하나의 인덱스 키 엔트리가 많은 행에 대한 포인터를 저장하고 있는 구조