**SQL의 개념**

국제 표준 DB 언어, 관계대수와 관계해석을 기초로 한 혼합 데이터 언어, 질의어

2. SQL의 분류 – DDL, DML, DCL

**72. DDL(Data Define Language, 데이터 정의어)**

DB를 구축하거나 수정할 목적으로 사용하는 언어, 번역 결과는 데이터 사전에 테이블로 저장

2. CREATE SCHEMA – CREATE SCHEMA 스키마명 AUTHORIZATION 사용자\_ID;

3. CREATE DOMAIN – CREATE DOMAIN 도메인명 [AS] 데이터\_타입 [DEFAULT 기본값]

[CONSTRAINT 제악조건명 CHECK (범위값)]; 속성값의 범위가 일부분일 때 사용

4. CREATE TABLE – CREATE TABLE 테이블명 (속성명 데이터\_타입 [DEFAULT 기본값] [NOT NULL], …

[, primary KEY(기본키\_속성명, …)][, UNIQUE(대체키\_속성명, …)][FOREIGN KEY(외래키\_속성명, …)

REFERENCES 참조테이블(기본키\_속성명, …)] [ON DELETE 옵션] [ON UPDATE옵션]

[, CONSTRAINT 제약조건명] [CHECK (조건식)];

5. CREATE VIEW – CREATE VIEW 뷰명[(속성명[, 속성명, …])] AS SELECT문;

6. CREATE INDEX – CREATE [UNIQUE] INDEX 인덱스명 ON 테이블명(속성명 [ASC | DESC])

[, 속성명 [ASC \ DESC]]) [CLUSTER]; / ASC-오름차순, DESC-내림차순

7. ALTER TABLE – 테이블에 대한 정의를 변경하는 명령문

ALTER TABLE 테이블명 ADD 속성명 데이터\_타입 [DAFAULT ‘기본값’];

ALTER TABLE 테이블명 ALTER 속성명 [SET DEFAULT ‘기본값’];

ALTER TABLE 테이블명 DROP COLUMN 속성명 [CASCADE];

8. DROP – 스키마, 도메인, 기본/뷰 테이블, 인덱스, 제약 조건 등을 제거하는 명령문

DROP SCHEMA/DOMAIN/TABLE/VIEW/INDEX 이름 [CASCADE | RESTRICTED]; / DROP CONSTRAINT

CASCADE : 제거할 요소를 참조하는 다른 모든 개체를 함께 제거

RESTRICTED : 다른 개체가 제거할 요소를 참조중일 때는 제거를 취소한다.

**73. DCL(Data Control Language, 데이터 제어어)**

데이터의 보안, 무결성, 회복, 병행 제어 등을 정의하는 데 사용하는 언어

2. **GRANT / REVOKE** – 권한 부여를 위한 명령어 / 권한 취소를 위한 명령어

사용자등급 지정 및 해제

GRANT 사용자등급 **TO** 사용자\_ID\_리스트 [IDENTIFIED BY 암호];

REVOKE 사용자등급 FROM 사용자\_ID\_리스트;

테이블 및 속성에 대한 권한 부여 및 취소

GRANT 권한\_리스트 ON 개체 TO 사용자 [WITH GRANT OPTION];

REVOKE [GRANT OPTION FOR] 권한\_리스트 ON 개체 FROM 사용자 [CASCADE];

권한 종류 : ALL, SELECT, INSERT, DELETE, UPDATE, ALTER 등

WITH GRANT OPTION/FOR : 받은 권한을 다른 사용자에게 부여할 수 있는 권한을 부여/취소

3. COMMIT – 트랜잭션 성공 시 변경된 모든 내용을 유에 반영할 때 사용하는 명령어

4. ROLLBACK – 아직 COMMIT되지 않은 변경된 모든 내용들을 취소하고 DB를 이전상태로 되돌림

5. SAVEPOINT – 트랜잭션 내에 ROLLBACK 할 위치인 저장점을 지정하는 명령어, 이름을 부여

**74. DML(Data Manipulation Language, 데이터 조작어)**

DB 사용자가 응용 프로그램이나 질의어를 통해 저장된 데이터를 관리하는데 사용되는 언어

2. 삽입문(INSERT INTO~)

INSERT INTO 테이블명([속성명1, 속성명2, ...)) VALUES (데이터1, 데이터2…);

3. 삭제문(DELETE FROM~) – DELETE FROM 테이블명 [WHERE 조건]; 테이블의 구조는 남아있음

4. 갱신문(UPDATE~SET~)

UPDATE 테이블명 SET 속성명 = 데이터[, 속성명=데이터, …][WHERE 조건];

**75. DML-SELECT-1**

SELECCT [PREDICATE] [테이블명.]속성명 [AS별칭][.[테이블명.]속성명, …] [, 그룹함수(속성명 [별칭])]

[, WINDOW함수 OVER (PARTITION BY 속성명1, 속성명2, … ORDER BY 속성명3, 속성명4, …)]

FROM 테이블명[, 테이블명, …] [WHERE 조건] [GROUP BY 속성명, 속성명, …]

[HAVING 조건] [ORDER BY 속성명 [ASC | DESC]];

Predicate : 불러올 튜플 수를 제한할 명령어를 기술, ALL/DISTINCT/DISTINCTROW

**76. DML-SELECT-2**

그룹함수 : GROUP BY절에 지정된 그룹별로 속성의 값을 집계할 함수를 기술한다. / SUM, AVG 등

WINDOW 함수 : GROUP BY절을 이용하지 않고 속성의 값을 집계할 함수를 기술한다.

-PARTITION BY : WINDOW 함수가 적용될 범위로 사용할 속성 지정

-ORDER BY : PARTITION 안에서 정렬 기준으로 사용할 속성 지정

GROUP BY절 : 특정 속성을 기준으로 그룹화하여 검색할 때 사용, 보통 그룹함수와 함께 쓰임

HAVING절 : GROUP BY와 함께 사용되며, 그룹에 대한 조건을 지정한다.

4. 집합 연산자를 이용한 통합 질의

FROM 테이블명 UNION | UNION ALL | INTERSECT | EXCEPT SELECT FROM 테이블명 …

UNION/UNION ALL : 두 SELECT문 조회 결과 통합하여 모두 출력, 중복행 1번 출력/그대로 출력

INTERSECT : 두 SELECT문의 조회 결과 중 공통된 행만 출력한다.

EXCEPT : 첫 번째 SELECT문의 조회 결과에서 두 번째 SELECT문의 조회 결과를 제외한 행 출력

**77. 프로시저(Procedure)**

절차형 SQL을 활용하여 특정 기능을 수행하는 일종의 트랜잭션 언어, 미리 저장한 작업 수행

DECLARE : 프로시저의 명칭, 변수, 인수, 데이터 타입을 정의하는 선언부이다.

BEGIN / END : 시작과 종료, CONTROL : 조건문 or 반복문이 삽입되어 순차적으로 처리

SQL : DML, DCL이 삽입되어 데이터 관리를 위한 조회, 추가, 수정, 삭제 작업을 수행

EXCEPTION : BEGIN ~ END 안의 구문 실행 시 예외가 발생하면 이를 처리하는 방법을 정의

TRANSACTION : 수행된 데이터 작업을 DB에 적용할지 취소할지를 결정하는 처리부

2. 프로시저 생성

CREATE [OR REPLACE] PROCEDURE 프로시저명(파라미터) [지역변수 선언] BEGIN BODY; END;

OR REPLACE : 선택적인 예약어, 동일한 프로시저 이름이 이미 존재하는 경우 기존 프로시저 대체

파라미터 : IN/OUT/INOUT : 호출 프로그램이 프로시저에게 값을 전달/반환/전달, 실행 후 반환

매개변수명 : 호출 프로그램으로부터 전달받은 값을 저장할 변수의 이름을 지정

프로시저 BODY : 코드 기록 부분, BEGIN에서 시작 END에서 종료, 사이에는 하나이상의 SQL 존재

3. 프로시저 실행 – EXECUTE(EXEC), CALL사용, EXECUTE/EXEC/CALL 프로시저 명;

4. 프로시저 제거 – DROP PROCEDURE 프로시저명;

**78. 트리거(Trigger)**

데이터의 삽입, 갱신, 삭제 등의 이벤트가 발생할 때 관련 작업이 자동으로 수행되는 절차형SQL

2. 트리거의 구성

DECLARE→EVENT(트리거가 실행되는 조건을 명시)→BEGIN→CONTROL→SQL→EXCEPTION→END

3. 트리거의 생성

CREAT [OR REPLACE TRIGGER] 트리거명 [동작시기 옵션][동작 옵션] ON 테이블명

REFFERENCING [NEW | OLD] AS 테이블명 FOR EACH ROW [WHEN 조건식] BEGIN BODY; END;

동작시기 옵션 : 트리거가 실행될 때를 지정, AFTER/BEFORE – 테이블이 변경된 후/되기 전 실행

동작 옵션 : 트리거가 실행되게 할 작업의 종류를 지정

INSERT/DELETE/UPDATE : 새로운 튜플을 삽입/삭제/수정할 때 트리거가 실행

NEW/OLD : 추가되거나 수정에 참여할/수정되거나 삭제전 대상이되는 튜플들의 집합을 의미

FOR EACH ROW : 각 튜플마다 트리거를 적용한다는 의미

WHEN 조건식 : 선택적인(Optional)예약어이다. 트리거를 적용할 튜플의 조건을 지정한다.

**79 사용자 정의 함수**

SQL을 사용하여 일련의 작업을 연속적으로 처리, 종료 시 처리 결과를 단일값으로 반환

2, 사용자 정의 함수의 구성도

DECLARE→BEGIN→CONTROL→SQL→EXCEPTION→RETURN(필, 반환할 값, 변수 정의)→END

3. 사용자 정의 함수 생성 - CREATE [OR REPLACE] FUNCTION 사용자 정의 함수명(파라미터) [지역변수 선언] BIGIN 사용자 정의 함수 BODY; RETURN 반환값; END;

파라미터 : IN(값 전달), 매개변수명, 자료형 / RETURN : 반환 값 or 변수 호출 프로그램으로 반환

4. 사용자 정의 함수 실행

SELECT 함수 FROM 테이블명 / INSERT INTO 테이블명(속성명) VALUES 함수명;

DELETE FROM 테이블명 WHERE 속성명 = 함수명; / UPDATE 테이블명 SET 속성명 = 함수명;

80. 제어문

1. 제어문의 개요 – 절차형 SQL의 진행 순서를 변경하기 위해 사용하는 명령문 ex) if, loop, goto

2. IF문

**IF 조건 THEN**

**실행할 문장1;**

**ELSE IF 조건2 THEN**

**실행할 문장2;**

**ELSE**

**실행할 문장3**

**END IF;**

3. LOOP문

기본형

For 변수 in 초기값..종료값

**Loop**

**실행할 문장;**

**end loop**

FOR LOOP : 초기값부터 종료값까지 1씩 증가하면서 실행할 문장을 반복 수행

FOR 변수 IN 초기값..종료값

**LOOP**

**실행할 문장;**

**END LOOP;**

WHILE LOOP : 조건이 참인 동안 실행할 문장을 반복

4. CONTINUE – 반복문의 실행을 제어하기 위해 사용하는 예약어

**Continue when 조건;**

5. GOTO문 – 원하는 위치로 이동하여 명령문을 수행하기 위해 사용하는 제어문

**GOTO 레이블;**

**<<레이블>>**

**실행할 문장;**

81. 커서(Cursor)

1. 커서의 개념 – 쿼리문의 처리 결과가 저장되어 있는 메모리 공간을 가리키는 포인터

2. 묵시적 커서(Implicit Cursor) – 자동으로 수행, 수행된 쿼리문의 정상적인 수행 여부를 확인하기 위해 사용

DBMS 자체적으로 열리고 패치되어 종료, 커서의 속성을 조회하여 사용된 쿼리 정보를 열람하는 것

커서의 속성

SQL%FOUND : 쿼리 수행의 결과로 패치된 튜플 수가 1개 이상이면 TRUE

SQL%NOTFOUND : 쿼리 수행의 결과로 패치된 튜플 수가 0개이면 TRUE

SQL%ROWCOUNT : 쿼리 수행의 결과로 패치된 튜플 수를 반환

SQL%ISOPEN : 커서가 열린 상태이면 TRUE, 묵시적 커서는 자동으로 생성된 후 자동으로 닫혀 항상 FALSE

3. 명시적 커서(Explicit Cursor) – 직접 구현, 쿼리문의 결과를 저장하여 사용함으로써 동일한 쿼리가 반복 수행되어 데이터베이스 자원이 낭비되는 것을 방지

선언(Declare) 형식

**CURSOR 커서명(매개변수1, 매개변수2, …)**

**IS**

**SELECT문;**

열기(Open)형식

OPEN 커서

패치(Fetch)형식

FETCH 커서명 INTO 변수1, 변수2…;

닫기(Close) 형식

Close 커서명;