**관계 데이터베이스와 언어**

[데이터 모델](http://www.cworldlab.com/CandCplus/DB-01.htm#데이터 모델) | [관계데이터 모델](http://www.cworldlab.com/CandCplus/DB-01.htm#관계 데이터 모델)| [질의어(SQL)](http://www.cworldlab.com/CandCplus/DB-01.htm#질의어) | [검색](http://www.cworldlab.com/CandCplus/DB-01.htm#검색) | [삽입](http://www.cworldlab.com/CandCplus/DB-01.htm#삽입) | [삭제](http://www.cworldlab.com/CandCplus/DB-01.htm#삭제) | [생성](http://www.cworldlab.com/CandCplus/DB-01.htm#생성) | [갱신](http://www.cworldlab.com/CandCplus/DB-01.htm#갱신)

**데이터 모델**데이터 모델

* + 데이터 모델의 개념
    - 현실 세계를 데이터베이스에 표현하는 중간 과정
    - 데이터 베이스의 구조를 묘사하기 위해 사용되는 개념
    - 논리적 데이터 모델  
      (데이터 필드로 기술된 데이터 타입과 이 데이터 타입들간의 관계를 이용하여 현실 세계를 표현하는 방법)
    - 개념적 데이터 모델  
      (속성들로 기술된 개체 타입과 이 개체 타입들가늬 관계를 이용하여 현실셰를 표현하는 방법)
    - 데이터 모델의 구성요소
      * 구조
      * 연산
      * 제약조건

1. 개체-관계 모델(E-R모델)
   * 개체-관계 모델이란?
     + 현실세계를 개념적으로 표현
     + 1976년 P.Chen이 제안
     + E-R 다이어그램은 E-R 모델을 그래프 방식으로 표현한 것
   * 구성요소
     + 사각형 : 개체 집합(Entity set)
     + 타원 : 속성(Attribute)
     + 마름모 : 개체 집합간의 관계
     + 선 : 연결
   * 관계의 유형
     + 1 : 1 (부부관계)
     + 1 : N (사람 - 도시 거주관계, 교수 - 학생 관계)
     + N : N (학생 - 과목 관계)
2. 논리적 데이터 모델
   * 관계 데이터 모델(테이블 구조, 정규화)
   * 망 데이터 모델(그래프 구조, 오너 - 멤버 관계)
   * 계층 데이터 모델(트리구조,부모 - 자식 관계)
   * 객체지향 데이터 모델(객체, 객체의 특성, 객체의 연산)

[위로](http://www.cworldlab.com/CandCplus/DB-01.htm#관계 데이터베이스와 언어)

**관계 데이터 모델**

1. 릴레이션
   * 특성
     + 릴레이션에서 모든 튜플은 상이함.
     + 릴레이션에서 튜플 사이에는 순서 정의된지 않음
     + 릴레이션을 구성하는 속성 사이에는 순서 없음
     + 릴레이션 내의 모든 값은 원자값(atomic value)지님
   * 용어

|  |  |
| --- | --- |
| 튜플(Tuple) | 릴레이션의 행(row) |
| 속성(Attribute) | 릴레이션의 열(column) |
| 영역(Domain) | 한 속성이 가질 수 있는 값의 집합 |
| 차수(Degree) | 한 릴레이션에서 속성의 개수 |
| 카디날리티(Cardinality) | 한 릴레이션에서 튜플의 수 |

|  |  |
| --- | --- |
| 기본키 | 하난의 속성만으로 한 릴레이션내의 모든 튜플구분 |
| 후보키 | 유일한 식별성을 갖는 속성의 조합이 한 릴레이션에 여러 개 있을 경우 |
| 대체키 | 후보키 중 하나를 임의로 선택하여 기본 키로 정하면 기본키가 아니 키를 대체키 |
| 외래키 | 참조의 무결성 지님 |
| 슈퍼키 | 릴레이션을 구성하는 속성의 전체 부분집합에서 같은 튜플이 발생하지 않는 부분집합 |

1. 관계 데이터 언어
   * 관계 대수
     + 특징  
       - 릴레이션에 관한 연산의 집합이다.  
       - 절차식 언어의 특성을 지닌다.  
       - Codd가 최초로 연산의 집합을 정의하였다.  
       - 일반 집합 연산자와 특별 관계 연산자의 두 그룹으로 구성되어 있다.
     + 특별 관계 연산자
       - 프로젝션(projection)   
         (릴레이션의 특정 속성만으로 구성된 새로운 릴레이션을 구하기 위한 연산으로,수직적 부분집합)  
         예) 구입자[이름]=(이름)

|  |
| --- |
| 이름 |
| 이화중 김현일 김동진 |

* + - * 셀렉션(selection)  
        (한 릴레이션으로부터 조건에 만족된 튜플을 선택하는 연산으로, 수평적 부분 집합)  
        예)생산품[판매가<=800]=(생산품 코드,생산비,판매가)

|  |  |  |
| --- | --- | --- |
| 생산품 코드 | 생산비 | 판매가 |
| A | 500 | 800 |
| B | 400 | 400 |

* + - * 예)생산품[판매가=생산비]=(생산품 코드,생산비,판매가)

|  |  |  |
| --- | --- | --- |
| 생산품 코드 | 생산비 | 판매가 |
| B | 400 | 400 |

* + - * 조인(join)  
        - 일방조인(equi join) : 카티션 프러덕트와 셀렉션을 결합한 연산  
        - 자연조인(natural) : 카티션 프로덕트,셀렉션,프로젝션을 결합한 연산으로, 속서의 중복을 허용하지 않는다.
      * 디비전(division)  
        (이항 릴레이션과 단항 릴레이션의 두 릴레이션이 존재하는 경우, 단항 릴레이션에 있는 모든 값과 부합되는 이항 릴레이션의 속성의 모든 값으로 구성된 릴레이션을 구하기 위한 연산)
      * 연산자의 특성에 따른 분류

|  |  |
| --- | --- |
| 반드시 합병가능이어야 하는 연산자(두릴레이션의 차수와 대응속성별로 도메인이 같은것) | 합집합,교집합,차집합 |
| 기본(근원)연산자 | 합집합,차집합,카티션 프로덕트,프로젝션,셀렉션 |
| 복합연산자 | 교집합,조인,디비전 |
| 한개의 릴레이션만으로 연산이 가능한 연산자 | 프로젝션,셀렉션 |

* + 관계 해석
    - 특징  
      - 원하는 릴레이션에 대한 형태와 내용을 결정하는 조건들로 구성된다.  
      - 비절차적 언어의 특성을 지닌다.  
      - 수학의 predicate calculus에 기반을 두고 있다.  
      - 튜플 관계 해서과 도메인 관계 해것으로 구분한다.
    - 튜플 관계 해석
  + 질의어(Query Language)
    - SQL
      * 특징  
        - 관계 사상(relation mapping)에 기초한 언어이다.  
        - SELECT - FROM - WHERE 구조이다.  
        - 단순한 검색 기능뿐만 아니라, 완전한 종류의 갱신 연산 기능을 제공한다.
      * DML명령어

|  |  |  |  |
| --- | --- | --- | --- |
| **검색(Search)** | | | |
| |  |  |  | | --- | --- | --- | | 형식 | **SELECT FROM [WHERE]** [GROUP BY [HAVING [ORDER BY | [DISTINCT] 속성 LIST 테이블  [속성 LIKE]조건]          속성 LIST 조건] 속성 LIST [ASC | DESC]]; |   Select c\_Name, c\_PassWord FROM tblGameUser where Address = 서울 서초구 \_동 ; | | | |
| * + - * + SELECT절         + FROM 절         + WHERE 절         + DISTINCT 절         + LIKE 절         + ORDER BY 절         + GROUP 절         + HAVING 절 | 질의의 결과에 나타나기를 원하는 속성들을 열거 테이블의 리스트로 구성 관계형 대수의 선택 술어로 구성 검색 결과의 중복 레코드 제거 속성 이름과 함께 스트링 상수를 명세한 검색 조건 오름차순(ASC) 또는 내림차순(DESC)으로 정렬 검색 명세된 속서의 값별로 분할 각 그룹에 대한 검색조건을 명세 | | |
| **예) 학생 테이블에서 학생명이 '최'인 학번과 학년을 검색하는 경우** | | | |
| SELECT 학번, 학년 FROM    학생 WHERE  학생명=’최’; | | 결과 | |  |  | | --- | --- | | 학번 | 학년 | | 903 | 3 | |
| **예) 점수 테이블에서 중복된 코드번호를 제거하고 검색하는 경우** | | | |
| SELECT DISTINCT 코드번호 FROM     점수; | | 결과 | |  | | --- | | 코드번호 | | C-101 C-102 D-103 | |
| **예)점수테이블에서 중간고사 점수가 70점보다 큰 코드번호와 중간고사를 검색하되,**  **코드번호는 내림차순으로, 또 코드간은 코드번호에 대해서는**  **중간고사를 오름차순으로 검색하는 경우** | | | |
| SELECT 코드번호, 중간고사 FROM    점수 WHERE  중간고사 > 70 ORDER  BY 코드번호 DESC, 중간고사 ASC; | | 결과 | |  |  | | --- | --- | | 코드번호 | 중간고사 | | C-102 | 75 | | C-102 | 95 | | C-101 | 80 | | C-101 | 90 | |
| **예)코드번호가 'C-102'에 등록한 학생명을 검색하는 경우** | | | |
| SELECT 학생명 FROM 학생 WHERE 학번 IN  (SELECT 학번 FROM 점수  WHERE 코드번호='C-102'); | | 결과 | |  | | --- | | 학생명 | | 박 | | 조 | |
| **예)점수 테이블에서 학번별 중간고사의 평균을 검색하는 경우** | | | |
| SELECT 학번,AVG(중간고사) AS 중간평균 FROM 점수 GROUP BY 학번; | | 결과 | |  |  | | --- | --- | | 학번 | 중간평균 | | 901 | 77.5 | | 902 | 81.5 | | 903 | 65 | |
| 예)점수 테이블에서 코드번호 D로 시작되는 코드번호와 기말고사를 검색하는 경우 | | | |
| SELECT 코드번호 ,기말고사 FROM 점수 WHERE 코드번호 LIKE 'D%' ; | | 결과 | |  |  | | --- | --- | | 코드번호 | 기말고사 | | D-103 | 90 | | D-103 | 70 | |
| **예)점수 테이블에서 세사람 이상 등록한 학번의 중간고사의 평균을 검색하는 경우** | | | |
| SELECT 학번,AVG(중간고사) AS 중간평균 FROM 점수 GROUP BY 학번 HAVING COUNT(\*) >= 3 ; | | 결과 | |  |  | | --- | --- | | 학번 | 중간평균 | | 902 | 81.5 | |
| **집단 함수(aggregate function)** | | |  |
| 함수 | 의미 |  |  |
| COUNT | 값의 개수 |  |  |
| SUM | 총계 |  |  |
| AVG | 평균값 |  |  |
| MAX | 최대값 |  |  |
| MIN | 최소값 |  | [위로](http://www.cworldlab.com/CandCplus/DB-01.htm#관계 데이터베이스와 언어) |
| **삽입(Insertion)** | | | |
| 형식 | INSERT INTO 테이블 VALUE (속성값 리스트); INSERT INTO 테이블 SELECT 문 ; | | |
| * + - * + INSERT INTO ~ VALUE : 특성 속성에 대해 특정값을 갖는 하나의 튜플을 테이블에 삽입         + INSERT INTO ~ SELECT문 : 여러 개의 튜플을 테이블에 한 번에 삽입 | | | |
| 예) INSERT INTO 학생      VALUE ( 904,'송',2); | | 결과 | |  |  |  | | --- | --- | --- | | 학번 | 학생명 | 학년 | | 901 | 박 | 2 | | 902 | 조 | 4 | | 903 | 최 | 3 | | 904 | 송 | 2 | |
| **삭제(Deletion)** | | | |
| 형식 | DELETE FROM 테이블 [WHERE 조건]; | | |
| * + - * + WHERE 절의 조건을 만족하는 튜플을 테이블에서 삭제         + 테이블명은 반드시 하나만 명세해야 함. | | | |
| 예)DELETE FROM 학생      WHERE 학번 =901; | | 결과 | |  |  |  | | --- | --- | --- | | 학번 | 학생명 | 학년 | | 902 | 조 | 4 | | 903 | 최 | 3 | | 904 | 송 | 2 | |
| **갱신(Update)** | | | |
| 형식 | UPDATE 테이블 SET 속성=값 [WHERE 조건]; | | |
| * + - * + WHERE의 조건을 만족하는 각 튜플에 대하여 SET절의 지시에 따라 갱신 | | | |
| 예)UPDATE 점수     SET 중간고사=중간고사 +5     WHERE 학번 = 902; | | 결과 | |  |  |  |  | | --- | --- | --- | --- | | 학번 | 코드번호 | 중간고사 | 기말고사 | | 901 | C-101 | 80 | 95 | | 901 | C-102 | 75 | 85 | | 902 | C-101 | 95 | 80 | | 902 | C-102 | 100 | 75 | | 902 | D-103 | 65 | 90 | | 903 | D-103 | 65 | 70 | |
| **생성** | | | |
| **베이스테이블** | |  | [위로](http://www.cworldlab.com/CandCplus/DB-01.htm#관계 데이터베이스와 언어) |
| 형식 | CREATE TABLE 테이블 (속성이름 TYPE [NOT NULL], PRIMARY KEY (속성 LIST), UNIQUE          (속성 LIST), FOREIGN KEY (속성 LIST), CHECK           (조건식)); | | |
| * + - * + PRIMARY KEY : 기본키를 나타냄.         + UNIQUE : 대체키를 나타냄         + FOREIGN KEY : 참조 무결성을 나타냄         + CHECK : 무경성 제약조건 | | | |
| **데이터형** |  |  |  |
| DECIMAL(m,n) INTEGER SMALLINT FLOAT CHAR(n) VARCHAR(n) | 십진소수 4바이트 정수(31비트) 2바이트 정수(15비트) 부동 소수 문자의 수가 n인 스트링 문자의 개수가 최대 n인 스트링 | | |
| 예) 학생,점수,점수 테이블을 SQL문으로 정의하시오. | | | |
| CREATE | TABLE 학생 (학번 SMALLINT NOT NULL, 학생명 CHAR(10), 학년    SMALLINT); | | |
| CREATE | TABLE 과정 (코드번호 CHAR(6) NOT NULL, 과목 VARCHAR(20), 교수명 CHAR(20)); | | |
| CREATE | TABLE 점수 (학번  SMALLINT NOT NULL, 코드번호 CHAR(6) NOT NULL, 중간고사 INTEGER, 기말고사 INTEGER); | | |
| **속성의 첨가** | |  | [위로](http://www.cworldlab.com/CandCplus/DB-01.htm#관계 데이터베이스와 언어) |
| 형식 | ALTER TABLE 테이블 이름 ADD                  속성이름 TYPE; | | |
| * + - * + 기존 테이블에 새로운 속성 이름을 첨가 | | | |
| 예) ALTER TABLE 학생                  ADD   주소 VARCHAR(30); | | | |
| **삭제** | | | |
| 베이스 테이블 인덱스 뷰 | DROP TABLE 테이블 이름 DROP INDEX 인덱스 이름 DROP VIEW 뷰 이름 | | |

* + - * QUEL(튜플관계해석)
      * QBE(도메인관계해석)
  + 시스템 카탈로그  
    - 데이터베이스에 관한 정보를 모아 놓은 집합체  
    - 데이터 사전 또는 메타 데이터라 한다.  
    - 시스템 데이터베이스
  + 뷰
    - 개념  
      - 하나 또는 그 이사의 테이블로부터 유도되는 가상적 테이블  
      - 뷰는 다른 뷰를 가질수 있다.  
      - 터미널 사용자는 데이터 검색시 복잡한 명령을 간단하게 처리  
      - 뷰의 정의 내용에 따랄 일부 연산의 제한을 받음.
    - 장점  
      - 데이터베이스 재구성 면에서 논리적인 데이터 독립성을 제공  
      - 동시에 같은 데이터를 여러 사용자에게 상이한 방법으로 볼 수 있게 함.  
      - 사용자의 인식 단순화  
      - 숨겨진 데이터에 대해 자동적 보안 제공
  + 질의 최적화
    - 개요  
      -질의가 시스템에 제기되었을 때, 처리 시간을 단축시켜 사용자에게 빠르게 결과를 돌려준다.  
      - 질의의 실행하는 전략을 개선시킨다.
    - 최적화 과정  
      - 사용자 질의를 내부 표현 형식으로 변환한다.  
      - 변형된 질의 구문을 표준형으로 만들어 준다.  
      - 하위 단계의 후보 프로시저들을 선택한다.  
      - 질의 계획을 세우고, 가장 비용이 적은 것을 선택한다.