

[기술정의]

1. Data base

: 자료 저장

table

: 행과 열로 구성된 표.

① 구조 - 속성으로 구성됨.

② 예시

속성명	순번	아이디	이름	나이	→ 레이블구조 (= 릴레이션 스키마)
속성값	1	aaa	홍길동	40	→ 튜플 (= 하나의 데이터셋)] → 실제데이터 (= 릴레이션 인스턴스)
의 범위 (= 도메인)	2	bbb	일지매	30	

속성 / 컬럼 (= 하나의 열 전체)

③ 속성명

: 저장할 수 있는 속성을 지정한다.

④ 속성

: 속성명의 데이터 유형. ex) 문자 : Varchar(2), Char

숫자 : Number

날짜 : (in)date, timestamp, interval

2. 명령문

: 쿼리 (= query)

① 예시

```
create table 테이블명 (
    속성명 속성,
    속성명 속성 [제약조건],
    속성명 속성 );
```

=> 테이블은 속성명으로 구성된다.속성명은 속성의 값의 범위를 지정한다.속성은 제약조건을 설정할 수 있다.

create table aaa (

```
no number,
(in)date Varchar2 (숫자),
Point number);
```

한글 : 그네이트
영문:숫자 : 바이트

② 종류

◦ 테이블 생성하기

→ create table 테이블명 (---, ---);

◦ 테이블 찾기

→ desc 테이블명 ;

◦ 속성값 삽입

→ insert into 테이블명 Values (-, -, -);

◦ 컬럼찾기

→ Select * (전체) / 속성명 from 테이블명 ;

24-01-30 [제약조건 1], [오라클 자료형]

[기술정의]

1. 제약조건

: 속성에 조건을 설정한다.

① 종류

- Primary key : 속성의 값이 유니크하다. + index (찾기쉽게. 많으면 느려짐)
- Unique key : 속성의 값이 유니크하다.
- Not null : 속성의 값은 Null이 될 수 없다.

Null - 존재하지만 필요없는 값, 모르겠는 상태 (주의) 숫자 0은 있는 상태

- Check : 속성의 값의 범위를 지정한다.
- default : 속성의 기본값을 설정한다.

[기술 정의]

1. 오라클 자료형 (= 오라클 속성값)

: 오라클의 데이터를 저장하는 타입 정의.

① 종류

분류	나열	
문자	char	- 고정메모리 → CPU 성능빨라짐.
	Varchar2	- 가변메모리 → memory를 효율적 사용
숫자	number (5,2)	- 기본값은 소수점 두자리자리
	timestamp	- 년월일시분초 저장
	date	- 년월일시분초 저장, 년월일만 보여줌
	interval	- 기준날짜에서 년월일 추가해서 계산 가능.

② 컴퓨터구조

- CPU (중앙처리장치) - 결과값 보여줌 ex) 3 ↑
- Memory (RAM) - 저장한 자료 실행 ex) 1+2
- 보조기억장치 (하드디스크) - 모든 자료 저장 ex) 1, 2 저장

24-01-31 [DDL], [DML], [DCL]

[기술정의]

1. DDL (=데이터 정의어)

: 테이블의 스키마, 뷰, 시퀀스, 트리거, 프로시저 등을 만들고, 수정, 삭제 명령어

① alter (alter table 테이블명 [옵션] 속성명 [변경값];)

옵션: modify, add, drop

② Create (Create table 테이블명 (속성명 속성, 속성명 속성);)

③ drop (drop table 테이블명;)

2. DML (=데이터 조작어)

: 투플의 CRUD 연산.

명령어

① insert (insert into 테이블명 values (..., ..., ...));)

☆ ② select (Select 검색할 속성명 from 테이블명 [where] [order by])

③ delete (delete from 테이블명 [where])

④ update (update 테이블명 set 속성명 = 수정값 [where])

3. DCL (=데이터 제어어)

① Commit : CRUD 작업결과 영구저장.

② rollback : CRUD 작업취소. ex) 오류로인해 실행취소 해야할 때.

③ Save point : 중간저장

예제 01기 - 오후과제

1. 모든 학생의 점수를 반올림하여 출력하시오.

순서① ~~DDL~~ / DML

순서② ~~Insert~~ / ~~Update~~ / Select / ~~Deletes~~

순서③ Select 점수 반올림 : 특별한 기능 → 오라클 함수로 해결이 가능한가? → 검색하기 오라클 반올림
→ round (값, 자릿수)

from users

where

group by

order by

[013] 오류과제

Q. 모든 학생의 점수를 반올림하여 출력하시오.

① DDL / DML

② insert / update / select / delete

③ select 점수, 특별한 기능 → 오라클 함수로 해결이 가능할까? → 검색어 : 오라클 반올림 round

from users

where : 모든 학생(튜플)이기 때문에 where은 그룹처럼 보이지만 오라클상의 그룹의 의미는 아님.

group by : 그룹만드는 이유 → 통계를 파악하기 위해 → 결과물 1개 ex) 최대, 최소, 합 ...

order by : 정렬 ●

이어서.

오라클함수 ① 입력

ex) Point ① ~~-----~~ SUM (-----)

② 특별한기능 수행 sum()

③ 결과를 리턴 Point의 형태.

반올림 방법.

Ex) Select round (123.1234, 2) from dual;

-3-2-1 0 1 2 3

Q. 소수점 두번째자리까지 보이도록 반올림하여라

→ round (123.1234, 2)

123.12

Q. 소수점 두번째 자리까지 정수화하여라

trunc (123.1234, 2) from dual;

→ trunc(123.1234, 2)
123.12

Q. 123.1234의 소수점두번째 자리까지 정수화하여라

Select trunc (123.1234, 2) from users;

→ trunc (123.1234, 2) ↳ 뺀하지 않는 값은 상수를 입력하기 때문.

123.12
123.12 → 튜플의 개수만큼 삽입됨.
123.12

Q Point의 소수점 첫번째 자리까지 정수화.

Select trunc (Point, 1) from users;

상수가 아닌 값을 넣었기 때문에 튜플데이터에 맞게 바뀐 값이 나옴.

→ trunc (Point, 1)

99.1

89

88.2

77.6

:

Q. 2학년 중 가장 낮은 점수를 최득한 정수는 몇점입니까?

① DDL / DML

② insert / update / Select / delete

③ Select point, min(point) AS 가장낮은점수.
from users

where grade = 2 ;

group by 하지 않아도 해결할수 있다. → where을 통해 grade를 걸러내고

모든 투플을 대상으로 하는 group보다 더 효율적이다.

이어서.

Point를 같이 출력하면 오류가 난다 이유.

→ point : 단일행 / min() : 단일행.

모든 함수가 단일행인 것은 아니다.

min (입력값 1개) → 출력 : 단일행. → 최소값을 하나만으로	round (__, __ 입력값 2개) → 출력 : 단일행. → 반올림은 결과가 하나가 아님.
---	--

응용 group by를 적어서 풀어보기

답1. select min(point)
from users

where

group by grade having grad = 2 ;

1학년, 2학년, 3학년, 4학년 2학년만 그룹으로 묶어야라.

4개의 그룹을

답2 select min(point)

from users

where grade = 2

group by grade ; → 같은 나오나 group by 절이 불필요하다.

A → B A가 데려온 B보다

A ← B A가 데려온 B보다

응용

2학년 중 가장 낮은 점수를 소수점 첫번째자리로 반올림하여 몇점입니까?

①

②

Select ② (1)

from users ;

where grad = 2

Select round (min (point) , 1)

from users

where "grad = 2".

Q. 이름, 포인트, 학년, 생년월일, 성별을 출력해. 2. 1. 성별은 생년월일로 판단하여 마지막숫자가 1이면 남, 2이면 여라고 표시한다.

① DDL / DML

② INSERT UPDATE SELECT DELETE.

③ SELECT name, Point, Grade, jumin, [성별]

from users.

decode(Substr(jumin, 8, 1), 1, '남', 2, '여')

where

~~group by~~

order by.

이어서

성별은 어떻게 판별하나? → 함수가 가능한가? → 나는 어떻게 해결하나? →

1. 생년월일을 추출한다. → Substr(jumin, 8, 1)

2. 1.의 결과물로 성별을 판별한다. → decode 함수를 사용한다.

★ 1과 2를 세분화해야 한다. 그래야 쿼리를 테스트 할 때 어떤 범위에서 틀린지 알 수 있다.

1과 2를 합쳐서 생각하면 안된다.

↓

decode 함수

: 조건에 대한 출력값을 설정하는 함수.

decode(컬럼명/문자열, 조건1, 조건1 참일때 출력값, 조건2, 조건2 참일때 출력값)

음용) decode(컬럼명/문자열, 조건1, 조건1 참일때 출력값, 조건1 거짓일때 출력값)

음용) decode(컬럼명/문자열, 조건1, 조건1 참 출력값, 조건2, 조건2 참 출력값, 조건2 거짓일때 출력값)

답. SELECT name, Point, Grade, jumin, decode(Substr(jumin, 8, 1), 1, '남', 2, '여')
from users ;

[기술정의]

1. 함수

: 특정 기능을 수행한다.

① 종류

- 집계함수 : 결과값이 1개 ☆

sum() - 총합

avg() - 평균

max() - 최대값

min() - 최소값

count() - 개수

- 문자열 함수 : 모든 투플에 각각 적용된다.

substr (지문문자열, 시작위치, 가져올 문자 개수)

ex) 남자를 구할 때

substr (주민등록번호, 8, 1) → 주민번호 8번째자리부터 한개의 문자.

② TIP

함수를 select, where, from에 다 사용 가능하다.

2. Select문 동작순서

③ Select

① from : 기준 테이블

→ select 절 사용 가능. 테이블 명이 없을 뿐, 테이블 나열하기 가능.

② where : 기준 테이블에서 투플을 하나씩 가져옴.

→ 집계함수는 사용할 수 X → 집계함수는 모든 투플이 대상인데

where은 투플을 하나씩 가져오기 때문.

→ 서브쿼리 select문 사용하면 가능.

③ group by ④ [having]

⑥ Order by

Select 결과값.

: 테이블 or 결과물

3. 서브쿼리 (Subquery)

: 본 쿼리에 Select문이 포함된 것.

→ 서브쿼리가 먼저 실행됨, 실행결과 값을 본 쿼리에 사용.

① 위치

Select 서브쿼리 - 출력할 때 서브쿼리로 보여준다.

from 서브쿼리 - 서브쿼리를 테이블로 활용한다.

where 서브쿼리 - 서브쿼리를 조건으로 활용한다

4. Where의 연산자

① +, -, *, /

: 더하기, 빼기, 곱하기 나누기

② >, <, >=, <=, =, !=, <>

: 초과, 미만, 이상, 이하, 같다, 같지않다, 같지않다

③ in

: 특정 속성값을 모두 출력할 때 사용.

: where 속성명 in (속성값1, 속성값2, ...)

ex) addr in ('서울', '부산') = addr = '서울' or addr = '부산' or ...

④ between

: 특정 범위 내 데이터 조회할 때.

: where 속성명 between 작은 수 and 큰 수

ex) age between 20 and 30 = age >= 20 and age <= 30

⑤ like

: 일부 문자열이 포함된 데이터 조회할 때

: 와일드 카드 - , % → 어떻게 사용하느냐에 따라 속도차이가 낸다.

- : 한 개의 어떤 문자

% : 모든 어떤 문자

ex) addr like '%K%' - K가 어디엔가 들어간 주소를 찾아라.

'%K' - K로 끝나는 주소

'K%' - K로 시작하는 주소

24 - 02 - 01 [Select 절 분해], [별칭 [AS]], 시퀀스

[기술점의]

1. Select 절 분해

Select

from : 대상테이블 ① 실제 존재하는 물리적 테이블

② Select 결과로 나오는 논리적 테이블

★ where : 퍼블 설정조건

[연산자] ex) =, <, >, <=, >=, !=, !=, in, between, like, any, all

★ group by [having] : 그룹에 대한 통계를 피악등하기 위해 그룹화함.

ex) 남학생 중 가장 큰 키

그룹 통계 → group 가능

남학생 중 가장 키 큰 사람의 이름

그룹 통계불가능 → group 불가능.

Select 절을 컴퓨터가 처리하는 순서

no	name	Point	grade
1	a	100	1
2	b	200	1
3	c	300	2
4	d	100	2



③ Select name, Point

① from m

② where point > 100

④ Order by desc

desc : 오름차순

asc : 내림차순

no	name	Point	grade
2	b	200	1
3	c	300	2

↓ ② where

name	Point
b	200
c	300

↓ ③ select

name	Point
b	200
c	300

c 300

↓ ④ Order by

name	Point
c	300
b	200

b 200

2. 별칭 [AS]

: 속성명 뒤 별칭을 붙여 출력값에 별칭을 보이게 한다.

① 예시

Select grade, Count (*) [AS] 명

↓

생략가능

grade	명
:	:
:	:
:	:

3. 시퀀스

: 자동으로 번호를 부여함.

① 예시

Create Sequence User_sq ; → 시퀀스 생성

Select user_sq.nextval from dual ; → 시퀀스 자동생성 기능.

임의의테이블

Insert into m values (user_sq.nextval, — , — , — ...),
→ 순번을 자동으로 매김.

[기술정의]

[데이터 중복]

→

[이상현상]→ **[모델링]**→ **[정규화]**

: 릴레이션에 같은 정보를 저장하면

: 삽입(불필요한)

: 이상현상 최소화.

이상현성이 발생.

삭제(삭제하면 안되는)

갱신(불필요한)

[기본키 PK]

: 유니크

index

not null

[외래키 FK]

: 자식 릴레이션의 키가

부모릴레이션의 키를 참조.

① 이상현상 예시.

고객이 1대 이상 자동차 소유 가능할 때.

table : member.

ID	이름	차량번호	주소	포인트
a	홍길동	1234	서울	100
a	홍길동	5678	서울	100

중복되는 값 때문에 어떤 행에 해야하지
 → 데이터 중복이 발생하기 때문에
 테이블 분리를 통해 정규화를 해야한다.

② 이상현상 해결.

: 테이블 분리하기. - **[정규화]**

table : member

ID (PK)	이름	주소	포인트
a	홍길동	서울	100

table : Car

ID (FK)	차 번호
a	1234
a	5678

→ FK는 부모테이블의 ID가 있어야만 한다.

③ 이상현상 예시

학생과 수강과목 → 주제가 많아서 table 만들기 쉽지 않다.

④ 이상현상 해결

: 하나의 개체에 하나의 주제만 표현하기. - 개체의 독립성 - **[ERD]**

데이터 모델링 순서

1. 요구사항 분석

: 현행 업무 분석, 요구기능 분석 → 요구사항 명세서

2. 논리적 설계

: 개체 도출 → ERD

3. 물리적 설계

: 테이블 설계 → 테이블 명세서 (컬럼, 제약조건, KEY가 들어가 있음)

4. 데이터베이스 구축

: 데이터베이스 생성 → 쿼리를 증빙 (각 table이 CRUD 가능한지 증명)

무결성 제약조건

1. 개체 무결성 - PK

: unique, notnull,

2. 도메인 무결성 - 속성값

: 속성 값의 범위

3. 참조 무결성 - FK

: 참조하는 PK의 속성값
범위 안.

DB에서 중요한 것.

1. 무결성.

① 제약조건을 갖다.

② KEY로 유니크하게 [PK
PK]

③ 정규화 - 테이블 분리.

2. 속도

① index

② Select문을 잘 사용

- Select의 동작순서 기억

논리적 설계 단계

: 테이블 만들기 전 단계, ERD

1. ERD

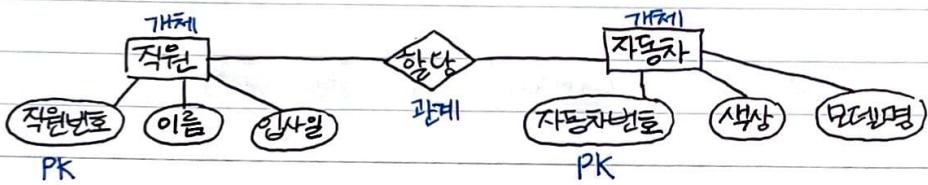
: **[개체][관계]** 모델 (Entity Relationship Diagram)

개체 : 독립적인 투성

관계 : 개체간의 연관성

→ 개체의 기본키 PK를 만드는 것이 중요하다.

① ERD 고려보기

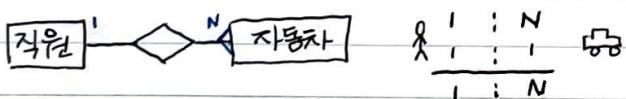


② 몇 : 몇

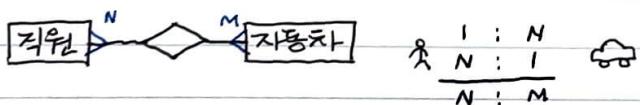
• 한명당 한 대



• 한 명당 여러대



• 한명당 여러대 + 한대당 여러명.



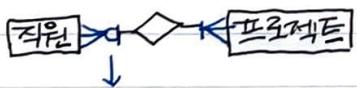
③ 버블

: 반드시 대응되는 것은 아니다. (부정의 의미)

: 동그라미 / ○ 표시.

• 예시

: 직원과 프로젝트가 N:M 일 때.



버블: 프로젝트가 있는 직원이 있을 수 있다.

→ 테이블 만들 때는 의미 없음.

선명 쓰지 말 것.

물리적 설계 단계

: 테이블 만드는 단계

1. 논리적 관계 1:1 이면 테이블 1개

① 예시

작원번호 PK
이름
입사일
자동차번호 <u>unique</u>
자동차색상
자동차모델명

Create table 직원 (

작원번호 Varchar2() Primary key,
 이름 Varchar2(),
 입사일 timestamp default Sysdate,
 자동차번호 Varchar2() unique,
 자동차색상 Varchar2(), 1:1 관계에서는 PK는 올 수 없으니
 자동차모델명 varchar2() unique를 사용한다.
);

2. 논리적 관계 1:N 일때 테이블 N개

: 외래키 FK 참조한다.

부모 table

회원ID (PK)	이름	주소	1 : N	글번호 (PK)	제목	날짜	작성ID (FK)
1	홍길동	서울		1	안녕	0301	1
2	일지매	수원		2	하이	0901	3

N table 쪽에서 1 table의 PK를 참조한다.

N table 쪽에서 FK를 만든다.

FK는 부모PK를 참조한다.

부모 PK에 없는 값은 올 수 없다.

↓

Create table 회원 (

회원ID Varchar2() Primary key
 이름 Varchar2()
 주소 Varchar2()
);

Create table 게시판 (

글번호 Varchar2() Primary key,
 제목 Varchar2(),
 날짜 timestamp,
 작성ID Varchar2()

속성 동일해야 한다.

Constraint FK명 foreign key (작성ID)
 references 회원 (회원ID)

);

참조한 부모 테이블의 PK를 삭제하면 안된다.

- 삭제순서 ① 관계 테이블에서 데이터를 날리고
 ② 학생 테이블에서 데이터를 날린다.

물리적 설계 단계

3. 논리적 관계가 N:M 이면 관계 테이블

: 각 개체가 테이블이 되고 관계도 테이블이 된다.

관계 테이블에는 두 개체의 PK를 참조하는 FK를 만든다.

관계테이블 구조								
회원ID(PK)	이름	주소	N:M	글번호(PK)	제목	날짜	구매ID(FK)	구매번호(FK)
1	홍길동	서울		n01	안녕	03/01	1	n01
2	일지애	수원		n02	하이	02/01	1	n02
							2	n02

FK랑 PK랑 잘 구분하기.



FK는 중복되어도 된다.

Create table 회원 (

회원ID _____ Primary Key,

이름 _____ ,

주소 _____);

Create table 게시판 (

글번호 _____ Primary Key,

제목 _____ ,

날짜 _____);

Create table 구매 (

구매 ID _____ ,

구매번호 _____ ,

foreign key (구매ID)

references 회원 (회원ID) ,

foreign key (구매번호)

references 게시판 (글번호));

관계table은 마지막 줄에 Create 한다.

4. 참조 무결성 위배가능성이 있을까?

	자식 (참조하는)	부모 (참조되는)
insert	O	X
delete	X	O
Select	X	X
update	O	O

5. 외래키 제약조건

① 부모 table의 참조하고 있는 PK가 삭제되면 자식 table의 FK가 null이 된다.

: FK절 맨 뒤에 on delete set null

② 부모 table의 참조하고 있는 PK가 삭제되면 자식 table의 FK가 삭제된다.

: FK절 맨 뒤에 on delete cascade

2024-02-13

[Join]

[기출정의]

1. Join

: 2개 이상의 테이블을 논리적으로 합친다.

① 종류

- Full Join

: 조건없음

- Inner Join

: 조건에 맞는 튜플만 조인

- Outer Join

$\begin{cases} \text{left outer join} \\ \text{right outer join} \end{cases}$

: inner join의 결과 + 조인에 참여하지 않는 것도.

- Self Join

: 하나의 테이블을 2개로 만들어서 조인.

② 예시

Q. 고객님이 등록한 자동차 정보. \rightarrow [고객] 1 : N [자동차]

고객 table (m. left)

ID (PK)	name	addr
1	Kim	수원
2	Lee	서울
3	Park	대전

1 : N 자동차 table (c. right)

C. num	C. name	Id
3125	중형	1
3125	소형	1
3312	중형	3

- Full Join

m.id	m.name	m.addr	c. C-num	c. C-name	c. id
1	Kim	수원	3125	중형	1
2	Lee	서울	3125	소형	1
3	Park	대전	3125	중형	1
1	Kim	수원	3125	중형	1
2	Lee	서울	3125	소형	1
3	Park	대전	3125	중형	1
1	Kim	수원	3312	중형	3
2	Lee	서울	3312	소형	3
3	Park	대전	3312	중형	3
2	Lee	서울	null	null	null

- Inner Join : $\exists m \rightarrow \text{on } m.id = c.id \rightarrow$ 조건에 해당하는 값

- Left Outer Join : 2. Lee. 서울 \rightarrow Join이 되지 않은 값인 데 left 테이블 중에서.

- Right Outer Join : 없음

Q. 배정 자동차의 차번호, 제조사, 자동차명, 가격을 출력하시오.

나의생각.

- Carinfo와 CompanyCar 필요 \rightarrow Join.

1234의 정보 찾아보자.

Carinfo의 C-num 값으로

CompanyCar의 C-num과 \rightarrow inner join.

일치하는 것을 찾음

} 기술명

```
Select      C.C-num, CC.C-com, CC.C-name, CC.C-price  
from       Carinfo C  
innerjoin  CompanyCar CC  
on         C.C-num = CC.C-num
```

~~where~~

~~group by~~

~~order by~~

} 해결답.

Q. 회사에서 구매는 했지만 배정되지 않은 자동차의 차번호, 제조사, 자동차이름 출력.

나의생각.

Carinfo와 CompanyCar 필요 \rightarrow Join

CompanyCar C-num 값과

Carinfo C-num

일치하는 것 제거 \rightarrow inner join.

CompanyCar에서 남은 풀을. \rightarrow outer join.

```
Select      C.C-num, CC.C-com, CC.C-name  
from       CompanyCar CC  
left outer join Carinfo C
```

on CC.C-num = C.C-num

where C.C-name is null

C.C-num

C.C-id

다 null 값으로서 사용된다.

1234	현대아이	소나타	1000	1234	중형	(11)
3344	기아	축제	2000	3344	소형	(11)
11188	기아	레고	800	null	null	null
9900	현대아이	그랜저	2100	null	null	null

Q. 자동차 가격이 1000만원 이상인 자동차의 번호 출력.

Q. 모든 사람의 정보를 출력하시오. 이름, 배정번호는 자동차번호, 자동차이름.

나의생각.

Users 와 Carinfo
left right → 조인 1
조인 CompanyCar → 조인 2
left right

outer join
PEN점의 정보이기 때문.

outer join
하나로 (모든사람)

Select *
from users
left outer join Carinfo C
on U.id = C.id
left outer join CompanyCar CC
on C.C-num = CC.C-num;

답)

U.id	U.name	U.addr	C.C-num	C.chname	C.id	CC.C-num	CC.com	CC.name
1111	Kim	수원	1234	줄령	1111	1234	현대	스파크
1111	Kim	수원	3344	소형	1111	3344	기아	슬레이
3333	Park	대전	5566	중형	3333	null	null	null
2222	Lee	서울	null	null	2222	null	null	null

Q. 배정된 자동차의 제조사와 사용자의 이름을 출력.

나의생각.

Carinfo와 CompanyCar
비교할 제작자

조인1과 Users
→ ~~inner join~~
 inner

Select *
from Carinfo C
left outer join CompanyCar CC
on C.C-num = CC.C-num
left inner join users u
on C.id = u.id;
→ Park null a 나오고

정회
배정번호
→ Carinfo 테이블
inner
 제작자는 같은
 자동차번호.
 Kim 현대
 Kim 기아
 Park null
 Lee null
 Park null

Select U.name, CC.C-com
from users u
left outer join Carinfo C
on U.id = C.id
left outer join CompanyCar CC
on CC.C-num = C.C-num
where CC.C-com is not null;
→ Park null X

Q. 서울사는 사람이 배정받은 자동차번호와 제조사와 이름 충돌.

나의불석

718

କ୍ରି.

Stu			Exam			examresult		
id	name	grade	e-id	name	indate	id	e-id	Point
aaa	홍길동	1	202401	oracle	24/01/09	aaa	202401	80
bbb	일지매	1	202402	java	24/02/10	aaa	202402	90
ccc	김정성	2				bbb	202401	85
ddd	이지태	3				ccc	202402	100

Q. 시험을 볼 학생의 점수 출력. 학생명, 시험ID, 정수

② S S.name, e.e-id, $\frac{stu}{r} e.point$

① f Stus

inner join Examresult e

on S.id = e.id

+

+

+

쿼리 결과 한번에 대비지 말 것. → 틀려도 예측적이 가능.

1. from

: Select * from Stu, Examresult

2. Join

: Select * from Stu [inner join] ~

3. 컬럼명을 넣어서.

: Select 컬럼명 from ~

→ result에도 학생 id가 있으니 2개의 join 가능.

Q. 시험을 볼 과목의 점수 출력. 과목명, 학생ID, 정수.

e s r

② S e.name, r.point, s.id

① f exam, Examresult, Stu → 조인. (1+1)+1

과목별점수 ② 학생별시험점수, 정수 순서는 상관X
모두 같다.
↓

① Stu S inner join Examresult r

on S.id = r.id

② inner join Exam e

on e.e-id = r.e-id

쿼리 대비는 순서.

① Select * from Stu S inner join Examresult r on S.id = r.id

② (↑ 추가) inner join Exam e on e.e-id = r.e-id

③ Select *에 컬럼명 추가.

★ 문제를 발생하는 과정 중심으로 공부하기★

Q. 오라클 과목의 점수 출력 과목명, 학생ID, 점수.

e r r

S e.name, r.point, r.id

f exam e

inner join examresult r

on e.e_id = r.e_id

w e.name = 'oracle';

쿼리운순서

① select * from 이너조인

② select * from 이너조인 + where 절

③ *에 컬럼명 넣기

↓ 다른방법.

exam에서 'oracle'의 정보만 가져오고 examresult 조인한다.

S

f (select * from exam where name = 'oracle') e → 서브쿼리가 from문에 있으므로
inner join examresult r
on e.e_id = r.e_id
컴퓨터 성능이 좋아졌다.

Q 시험을 한 번도 별찍없는 학생의 이름

어떤 table? stu, result S

S s.name

f stu s

left outer join examresult r

on s.id = r.id

w r.id is null

Q 학생별 시험 볼 건수 출력. 학생명, 볼 횟수.

stu. examresult.

S Count(*), S.name. $\rightarrow r.e_id$, r.Point, r.id 등 null인 값을 count한다.

f Stu S
입력하면 이제는 이어된다

left outer join examresult r \rightarrow 시험을 보지 않은 학생까지.

on S.id = r.id

group by	S.name	2 통계	id	name	grade	id	e_id	Point
1	일지애	aaa	홍	1	aaa	202401	80	
1	김정성	aaa	홍	1	aaa	202402	90	
0	*이제는	bbb	월	1	bbb	202401	85	
		ccc	김	2	ccc	202402	100	
		ddd	이	3	null	null	null	

Q 두 과목 모두 시험을 학생의 이름.

S Count(id), S.name

f Stu S

left outer join examresult r

on S.id = r.id

group by S.name having Count(id) >= 2

내란다면? Select Count(*) from exam

\rightarrow 여러 과목을 볼 학생.

① Stu table과 examresult table을 left outer join하여

모든 학생의 시험정보 테이블 만들기.

② ①의 결과에서 학생별 그룹 생성

③ exam의 개수 + 과목의 개수

④ 고목의 개수가 2를 넘지 않으면 조건

⑤ Select 출력.

[Self Join] Q11

Self Join. m1 id name addr m-id m-class Point
m2 id name addr m-id m-class Point
left right
m1 m2.

4 4 아니지 아니지

Select m1.id, m1.name, m2.id, m2.name

from member1 m1 4

inner join member1 m2 아니지.

on m1.m_id = m2.m_id

24 - 02 - 25 [Join] [로드맵]

[기술정리]

1. Join

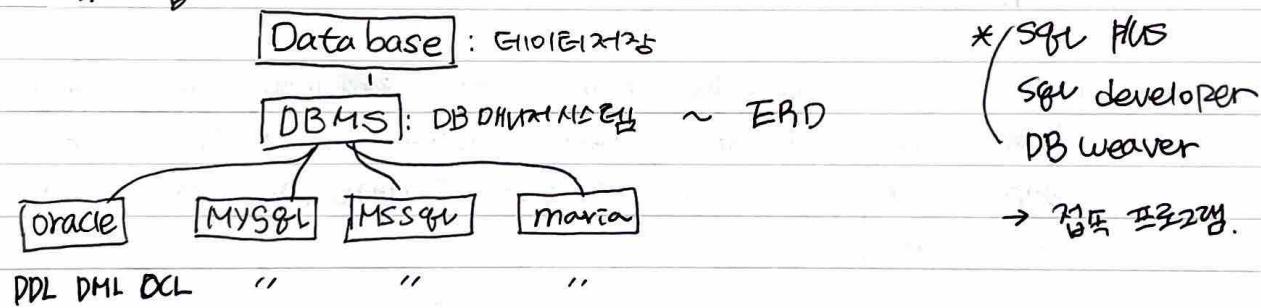
: Join은 2개 이상의 테이블을 논리적으로 합친다. → 질령명을 잘 보자!

① 종류

- Full Join : 모두 다 Join. (= 크로스Join)
- inner Join : 조건에 true 기준으로 조인. ex) 시험을 치른 학생, 자동차 등록 정보
- outer Join : inner Join의 결과 + inner Join 하지 않은 투플까지.
 - Left outer join
 - right outer join
 - Full outer join - left, right 상관없이 양쪽.

ex) 시험을 치루지 않은 학생, 자동차 소유하지 않은 사람.
- Self Join : 자신 table을 2개 이상 만들어서 inner Join 또는 outer Join 한다.
ex) 같은 테이블에서 같은 id 찾기.

2. 로드맵



[MySQL : 서버]

Workbench : 접속프로그램

[Oracle : 1521]

[MySQL : 3306]

[시작 실행 쿼리를 MySQL]

1. Show database;
 2. Create database orcl;
 3. Use orcl;
 4. Create table aaa (
no int Primary key,
id varchar(8));
- MySQL에서는 number는 varchar로 쓰지 않음.

[기술정의]

1. 본쿼리

: 내부에 (Select)로 시작하는 쿼리

: CRUD

2. 서브쿼리

: 본 쿼리가 실행하면서 서브쿼리를 만나면 서브쿼리 먼저 실행 후 결과값을 본 쿼리에 적용

↳ 단일행 (집계함수) / 다중행 고려해야 함. 적용이 어렵다.

ex) IN (-,-,-)

① 종류

Select 스칼라 서브쿼리

from 인라인뷰 서브쿼리 → table로 사용

where 조건

group by having 조건 → 조건으로 사용

User	id	name	Point
a	홍길동	1000	
b	김아첨	800	
c	홍제학	700	

③ Select name 서브쿼리 : select max(point) from u where id = c

① from user 서브쿼리 : select * from user

② where Point > 700 서브쿼리 : select max(point) from user

from 결과			where 결과			Select 결과		
①	id	name	Point	②	id	name	Point	③
	a	홍길동	1000	→	a	홍길동	1000	→
	b	김아첨	800	→	b	김아첨	800	→
	c	홍제학	700	→				홍길동, 김아첨.

(where 절에서 서브쿼리 실행하면)
 (서브쿼리 개수만큼 실행된다.)
ex) 100개가 100개라면 100번 실행됨.

(Select 절에서 서브쿼리 실행하면)
 from - where 후 나머지
 서브쿼리 개수만큼 실행된다

결론 : from 절은 1번만 실행되기 때문에 컴퓨터 성능↑
(where 절, select 절은 신증히.)

단일행인가, 다중행인가?

- Q1 Select name from uu;
- Q2 Select nam from uu where id='d' ;
- Q3 Select min(point) from uu;
- Q4 Select * from (Select * from uu) u where u.point >= 800 ;
- Q5 Select * from (Select * from uu where id = 'c') u;
- Q6 Select name from uu (where point > (Select min(point) from uu));
- Q7 Select name from uu where point > (Select point from uu);
- Q8 Select name from uu where point in (Select point from uu);
- Q9 Select name, (Select max(point) from uu) from uu;
- Q10 Select name, (Select point from uu) from uu;

답변)

- Q1 : 다중행. name의 합을 4개.
- Q2 : 단일행 id='d'인 행은 1개 그가 중복되지 않는다면
- Q3 : 단일행 최소값이자 집계함수의 값은 1개
- Q4 : 다중행 point가 800 이상인 행은 2개
- Q5 : 단일행 id='c'인 행은 1개
- Q6 : 다중행 최소값보다 큰 포인트는 당연히 여러개.
- Q7 : 모든포인트보다 큰 포인트? 무슨소리인지 모르겠음.
- Q8 : 다중행 in 안에 값의 개수만큼 값을 구해야하기 때문에 다중행? → 흥길동 1000
김아침 1000
홍서나 1000
정민우 1000
- Q9 : 단일행 최대값은 1개이기 때문에 이름과 같이 출력해도 행은 1개 → 흥길동 1000
- Q10 : 다중행 이름과 point는 여러행.
name : 단일. 서브쿼리(select point from uu)는 다중.
Select에는 뒤풀을 하나씩 가져와야 하기 때문에.
: 단일 행 하위 질의에 2개 이상의 행이 리턴되었음을.

Q11 이름이 흥으로 시작하면서 포인트가 평균보다 높은 사람의 이름은?

조건1 and 조건2 서브쿼리

→ 조건1이 앞에와서 서브쿼리를 하나라도 더 실행하므로.

Select name

from uu

where name like ('흥%') and Point > (Select avg(point) from uu);

* 서브쿼리: 먼저 실행하고 join을 한다면 컴퓨터 성능이 좋다.

심화)

포인트 기준 등급 선정 기준.

900 ~ 1000 Point	A
800 ~ 899 Point	B
700 ~ 799 Point	C

id	name	Point
a	홍길동	1000
b	김아침	800
c	홍제녁	700
d	추제요일	700

→ uu table

Q. 각 포인트 등급별로 인원수를 구하세요.

문제풀이 순서

- ① 나누기 : 포인트별 등급, 인원수.
- ② 어떤가 먼저 처리 써보자. - 포인트먼저처리하자.

[확인하기 - 포인트별 등급]

Select uu.* , (case when Point >= 900 then 'A'
 MySQL은 엔터도됨.
 when Point >= 800 then 'B'
 else 'C' end) grade
 from uu;

↓

ID	name	Point	grade
a	홍길동	1000	A
b	김아침	800	B
c	홍제녁	700	C
d	추제요일	700	C

[확인하기 - 인원수]

Select u1.grade , Count(*)
 from (Select uu.* , Case when Point >= 900 then 'A'
 when Point >= 800 then 'B'
 else 'C'
 End grade
 from uu) u1
 Group by u1.grade;

↓

grade	Count(*)
A	1
B	1
C	2

24-02-16

[Oracle, MySQL 테이블에서 특정 위치의 랭킹 가져오기]

[기술정의]

1. Oracle

: rownum 사용. → oracle에서 가상으로 제공하는 컬럼 순서.

① 예시

Select id, name, Point, rownum from member

순서 정상

Select id, name, Point, rownum from member

순서 바뀜

Order by Point desc, Id desc

왜 순서가 바뀐가? = 실행순서 → rownum이 Select절에 있기 때문에 orderby보다 먼저 실행됨.

어떻게 해결할까? → order by로 정렬한 테이블 만들기 → r1 table

→ rownum 추가하여 다시 테이블 만들기. → r table

→ r table은 rownum으로 만든 컬럼이 추가됨.

→ 본 쿼리에서 r table의 컬럼을 where 조건으로 선택.

AS Colnum

Select r.Id, r.Name, r.Point, r.Colnum

from r1 (Select r1.* , rownum Colnum

from (Select id, name, point

from member

order by point desc, id desc) r) r1

where r.colnum between 3 and 5;

↓

id	name	Point	Colnum
2	고길동	200	3
1	용길동	100	4
5	광여고	60	5

2. MySQL

: Limit 사용 → order by 절 이후에 실행
(limit 시작위치, 개수를 튜플 수)

① 예시

```
Select *  
from User  
order by Point desc, id desc  
Limit 2, 3 → 3번째 튜플부터 3개 가져옴.  
튜플의 순서가 0부터 시작.
```

id	name
0	a
1	b
2	c
3	d
4	e
5	f
6	g

24 - 02 - 19

[View = 가상 table]

[기본정의]

1. View (= 가상 table)

① 물리적 table

: DB에 실제로 존재함.

→ table : emp

id name addr

table : Car

id num Car

② 임시 table

: Select 절로 임시 만든 것.

① select e.id, e.name, e.addr, e.num, e.com
from emp e, Car c

③ 논리적 table (= 가상 table)

: 물리적 table로 만들어 놓은 가상 table

→ create view emp_view as
() → View 사용

임시 table은 select 출력 후 없어지지만

논리적 table은 가상으로 table을 만들어놓음.

→ select * from emp_view
→ view 퓨플 조회

→ ②, ③ 를 다 고려의 자동화 정보를 조회하는 것.

→ drop view emp_view [Cascade]

view에 대한 세부정보 모두 삭제. table 삭제시에도 사용 가능

→ View 퓨플 조회.