**Book Sharing 서비스 DB 설계 프로젝트**

**[주제]**

Book Sharing 도서 공유 서비스 플랫폼

**[주제 선정 이유]**

팀원 간 공통점이 독서라는 점에서 현재 사회의 독서 과정에 대한 문제를 인식하여 이를 해결할 수 있는 방안을 모색해 해당 주제를 선택하게 되었음

**[개발 동기]**

**<구체적 문제>**

* 2014년 11월, 도서를 신구〮 관련 없이 도서 원가의 15% 이내에서만 재판매 할 수 있도록 하는 ‘도서정가제’ 법안이 개편되어 도서 소비자의 비용적 부담이 커짐
* 최근 독서 인구가 줄어듬과 동시에 코로나19로 인해 영업을 종료하는 오프라인 서점이 증가하거나 도서관 방문이 어려워짐
* 이에 대한 대처방안으로 최근 E-book과 오디오북이 증가하고 있지만, 소비자들 사이에서 불편함을 호소하는 이들이 존재함
* 새 도서를 구매하는 소비자는 비용의 부담을 느끼고, 구매했다 하더라도 완독 후의 처리가 곤란함

**<해결방안 제시>**

* 소비자의 비용적 부담을 줄일 수 있도록 도서 판매가 아닌 도서 대출 혹은 도서 공유 서비스가 필요
* 방문이 아닌 도서를 직접 배송해주는 서비스가 필요
* E-book과 오디오북이 아닌 실물의 도서를 접할 수 있도록 하는 서비스가 필요
* 이미 완독한 도서를 유의미하게 처리할 수 있는 서비스가 필요
* 현재 위와 같은 서비스를 제공하는 프로그램이 미흡함으로 해당 서비스를 개발할 필요가 있음

**[기대 효과]**

* 완독한 도서 공유 시 훼손도를 검토하여 서비스 내 도서 목록에 저장될 때, 해당 회원에게 도서 대출 시 사용되는 포인트를 지급하는 방식으로 회원의 완독 도서를 유의미하게 처리할 수 있음
* 회원이 서비스에서 제공하는 도서를 대출 신청할 시, 직접 배송을 통해 실물의 도서를 접할 수 있음
* 포인트 충전 외 다양한 방식과 이벤트로 지급되는 포인트로 인해 소비자의 비용적 부담을 줄일 수 있음

**[주요 기능]**

* **도서공유**

회원이 보유한 도서를 공유할 수 있도록 돕는 시스템. 공유 신청 시, 공유한 도서를 7일 내로 수거하여 내부적 기준(도서의 훼손도) 으로 도서의 등급(A,B,C,D)을 결정한 후 각 등급에 적합한 식으로 회원에게 포인트 지급

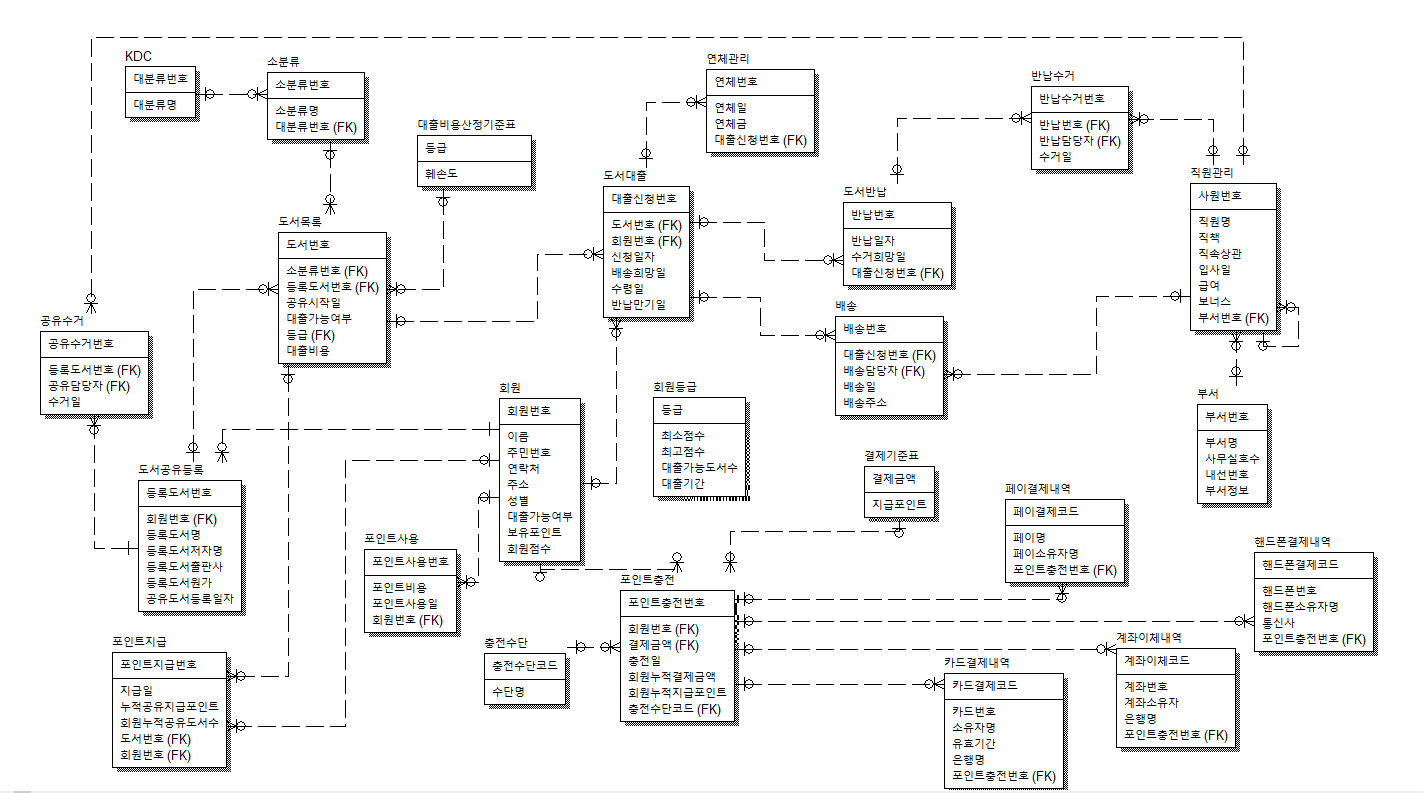
* **도서대출**

회원이 보유한 포인트를 사용하여 원하는 도서를 대출하는 시스템. 대출 비용은 각 도서의 원가와 등급에 따라 서로 상이함. 대출 신청 시 2주 내로 배송되며 반납 만기일은 회원의 등급에 따라 서로 상이함

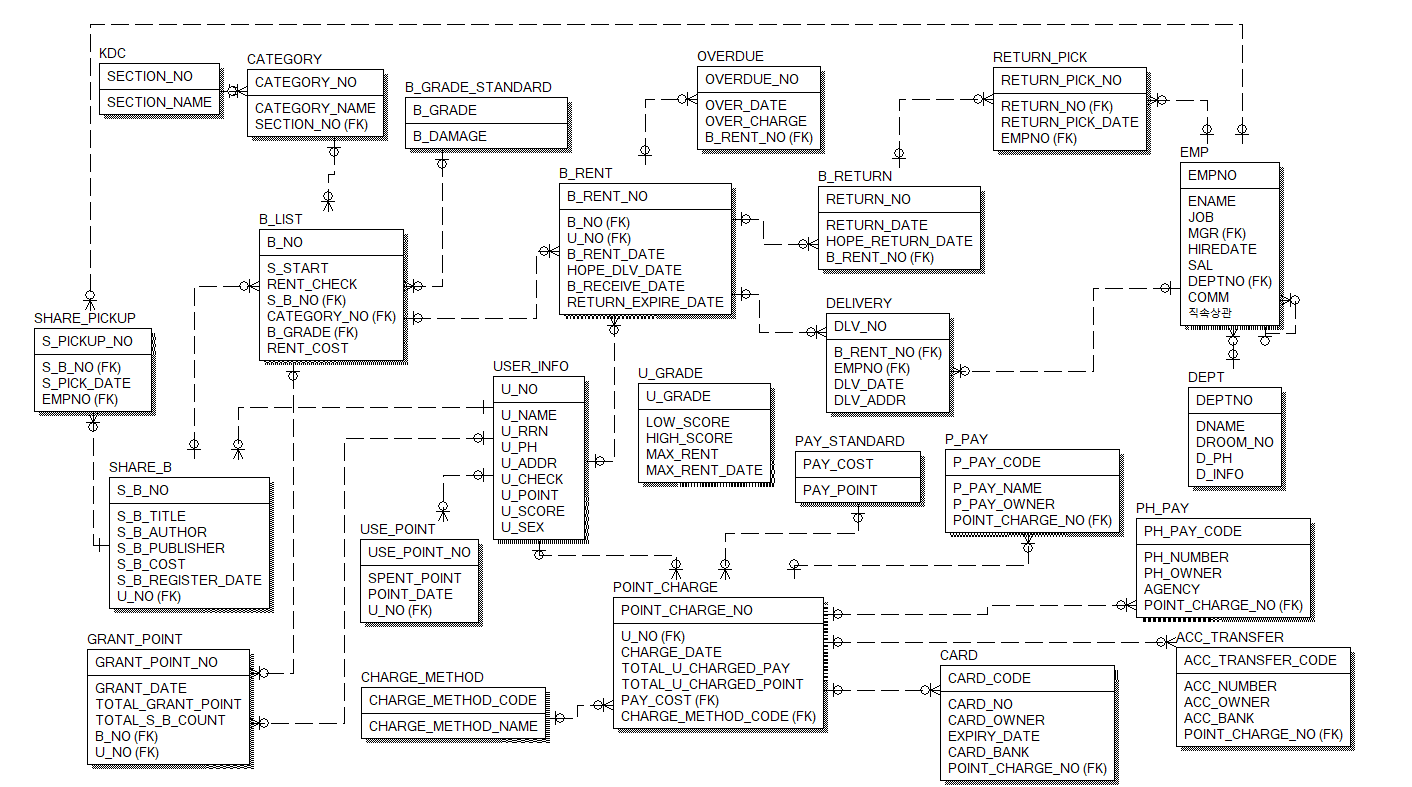
* **도서반납**

반납 신청과 연체 관리 시스템. 반납 신청 시 7일 내로 수거하며, 회원 등급에 따른 도서 반납 만기일을 초과할 시, 연체일 1일 당 연체금 100원이 발생함. 연체금은 기본적으로 회원의 보유 포인트에서 차감됨

**[논리적(Logical) DB 설계]**

****

**[물리적(Physical) DB 설계]**

****

**< SQL >**

**1. 도서 소분류별 누적 대출수**

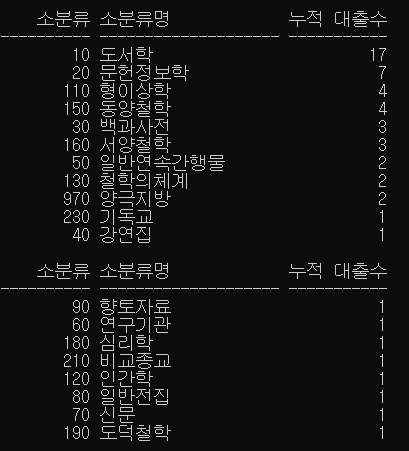
select b.category\_no "소분류", c.category\_name "소분류명", count(\*) "누적 대출수"

from b\_list b, b\_rent r, category c

where b.b\_no=r.b\_no and b.category\_no=c.category\_no

group by b.category\_no, c.category\_name

order by count(\*) desc;



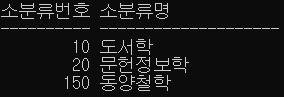
**2. 대출이 가장 많이 발생한 소분류 3순위까지 추출**

select cate\_no "소분류번호", cate\_name "소분류명"

from (select b.category\_no as cate\_no, c.category\_name as cate\_name from b\_list b, b\_rent r, category c where b.b\_no=r.b\_no and b.category\_no=c.category\_no

group by b.category\_no, c.category\_name order by count(\*) desc)

where rownum<4;

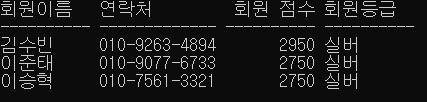


**3. 3월 한달 동안 책을 빌린 회원의 이름, 연락처, 회원등급을 출력해라**

select u.u\_name "회원이름", u.u\_ph "연락처", u.u\_score "회원 점수", case when u\_score>4501 then '다이아' when u\_score>3500 then '골드' when u\_score>2500 then '실버' else '브론즈' end "회원등급"

from user\_info u, b\_rent r

where u.u\_no=r.u\_no and to\_char(b\_rent\_date, 'mm')='03';



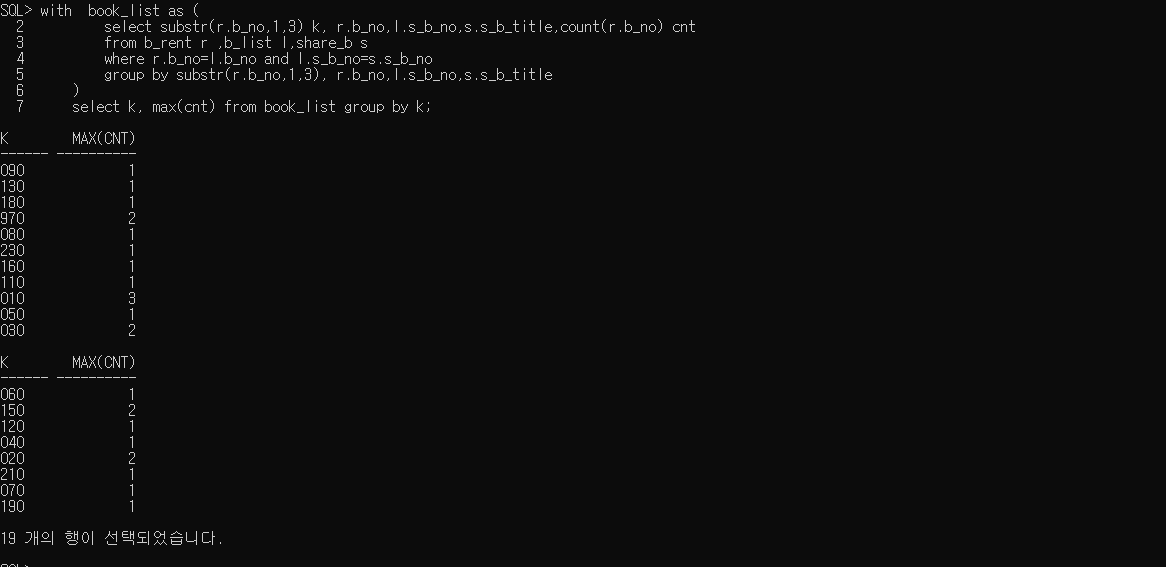
**4. 도서 소분류별 가장 많이 대출된 도서의 정보 추출**

with book\_list as ( select substr(r.b\_no,1,3) k, r.b\_no,l.s\_b\_no,s.s\_b\_title,count(r.b\_no) cnt

from b\_rent r ,b\_list l,share\_b s where r.b\_no=l.b\_no and l.s\_b\_no=s.s\_b\_no

group by substr(r.b\_no,1,3), r.b\_no,l.s\_b\_no,s.s\_b\_title)

select k, max(cnt) from book\_list group by k;



**5. 회원등급이 다이아인 사람의 월별 대출 권수를 조회하시오**

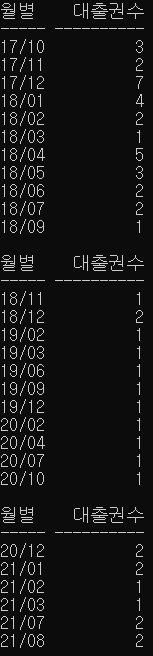
select to\_char(b\_rent\_date,'yy/mm') "월별", count(\*) "대출권수"

from b\_rent r, user\_info u

where u.u\_no=r.u\_no and u\_score>4500

group by to\_char(b\_rent\_date,'yy/mm')

order by 1;

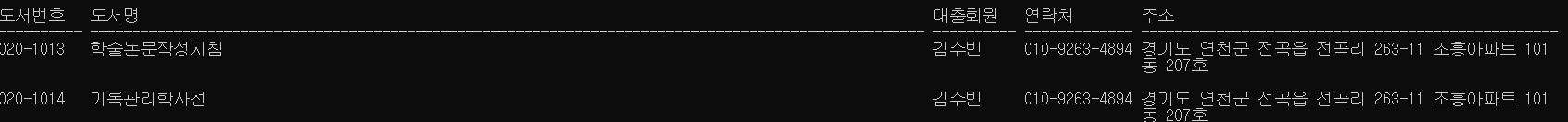


**6. 배송 희망일이 오늘인 도서번호, 도서명, 이름, 연락처, 주소 추출**

select b.b\_no "도서번호", s.s\_b\_title "도서명", u.u\_name "대출회원", u.u\_ph "연락처", u.u\_addr "주소"

from b\_rent r, b\_list b, share\_b s, user\_info u, b\_return ret

where(r.b\_rent\_no=ret.b\_rent\_no and b.b\_no=r.b\_no and s.s\_b\_no=b.s\_b\_no and u.u\_no=r.u\_no) and ret.hope\_return\_date = '18/02/07';

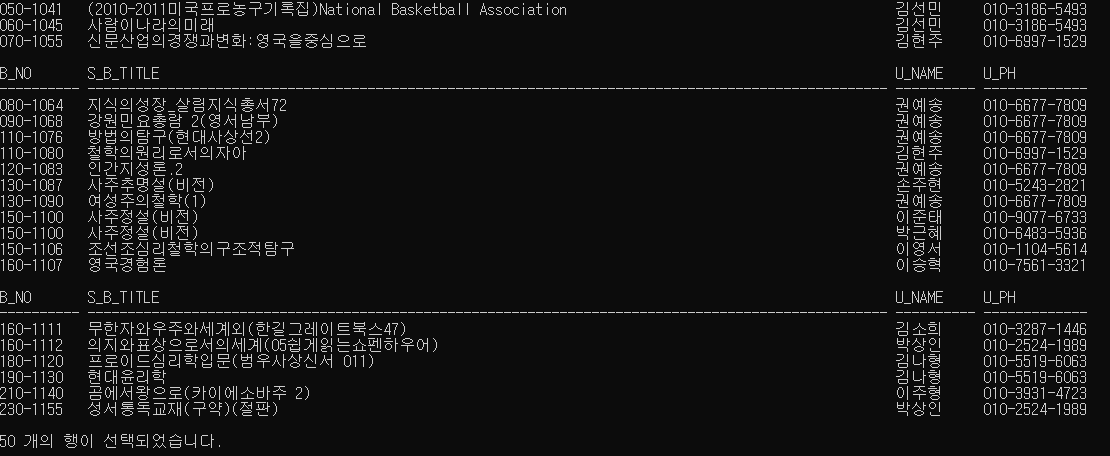


**7. 연체일이 14일 지난 도서번호, 도서명, 회원이름, 연락처를 추출**

select r.b\_no, s.s\_b\_title, u.u\_name, u.u\_ph

from overdue o, b\_rent r, share\_b s, b\_list b, user\_info u

where r.b\_rent\_no=o.b\_rent\_no and b.b\_no=r.b\_no and s.s\_b\_no=b.s\_b\_no and u.u\_no=r.u\_no and o.over\_date>=14;



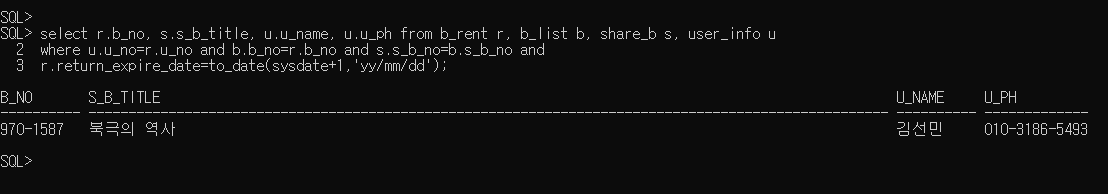
**8. 도서 반납만기일이 하루 남은 도서의 도서번호, 도서명, 회원이름, 연락처 추출**

select r.b\_no, s.s\_b\_title, u.u\_name, u.u\_ph

from b\_rent r, b\_list b, share\_b s, user\_info u

where u.u\_no=r.u\_no and b.b\_no=r.b\_no and s.s\_b\_no=b.s\_b\_no and

r.return\_expire\_date=to\_date(sysdate+1,'yy/mm/dd');

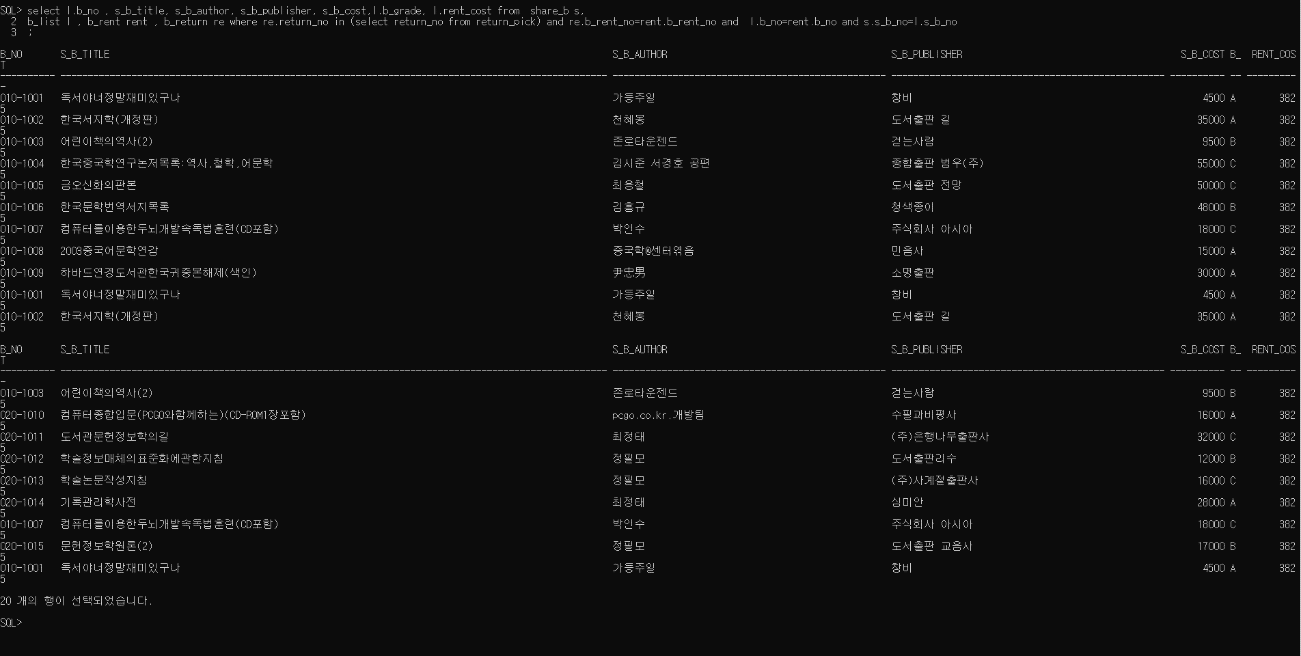
****

**9. 수거된 도서의 도서정보(도서번호,도서명,저자명,출판사,원가,등급,대출비용)을 추출**

select l.b\_no , s\_b\_title, s\_b\_author, s\_b\_publisher, s\_b\_cost,l.b\_grade, l.rent\_cost

from share\_b s, b\_list l , b\_rent rent , b\_return re

where re.return\_no in (select return\_no from return\_pick) and re.b\_rent\_no=rent.b\_rent\_no and l.b\_no=rent.b\_no and s.s\_b\_no=l.s\_b\_no;



**10. 반납율을 계산하라**

select distinct((select count(return\_no) from b\_return)/(select count(b\_rent\_no) from b\_rent))

from b\_return r, b\_rent b;



**11. 현재 보유 책 수량이 얼마나 되는지 소분류별로 출력**

select c.category\_name, count(b.b\_no)

from b\_list b, category c

where c.category\_no=b.category\_no group by c.category\_name;



**12. 8월 한 달 동안 고객이 각 결제 수단을 얼마만큼 사용했는지,**

**총 금액은 얼마인지 사용 빈도수가 높은 순서대로 출력**

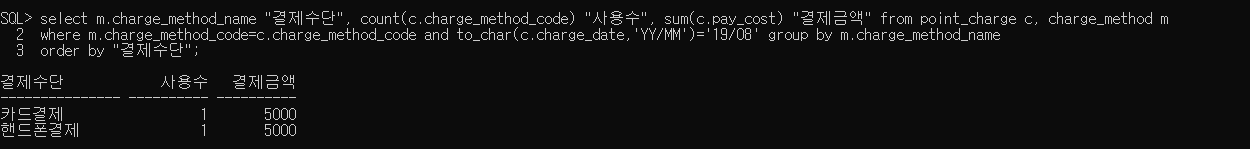
select m.charge\_method\_name "결제수단", count(c.charge\_method\_code) "사용수", sum(c.pay\_cost) "결제금액"

from point\_charge c, charge\_method m

where m.charge\_method\_code=c.charge\_method\_code and to\_char(c.charge\_date,'YY/MM')='19/08'

group by m.charge\_method\_name

order by "결제수단";

****

**13. 성별, 연령대 별 사용한 결제 수단에 따른 인원수 추출**

select Age as 연령대, Gender as 성별, Method as 수단, count(\*) as 인원수

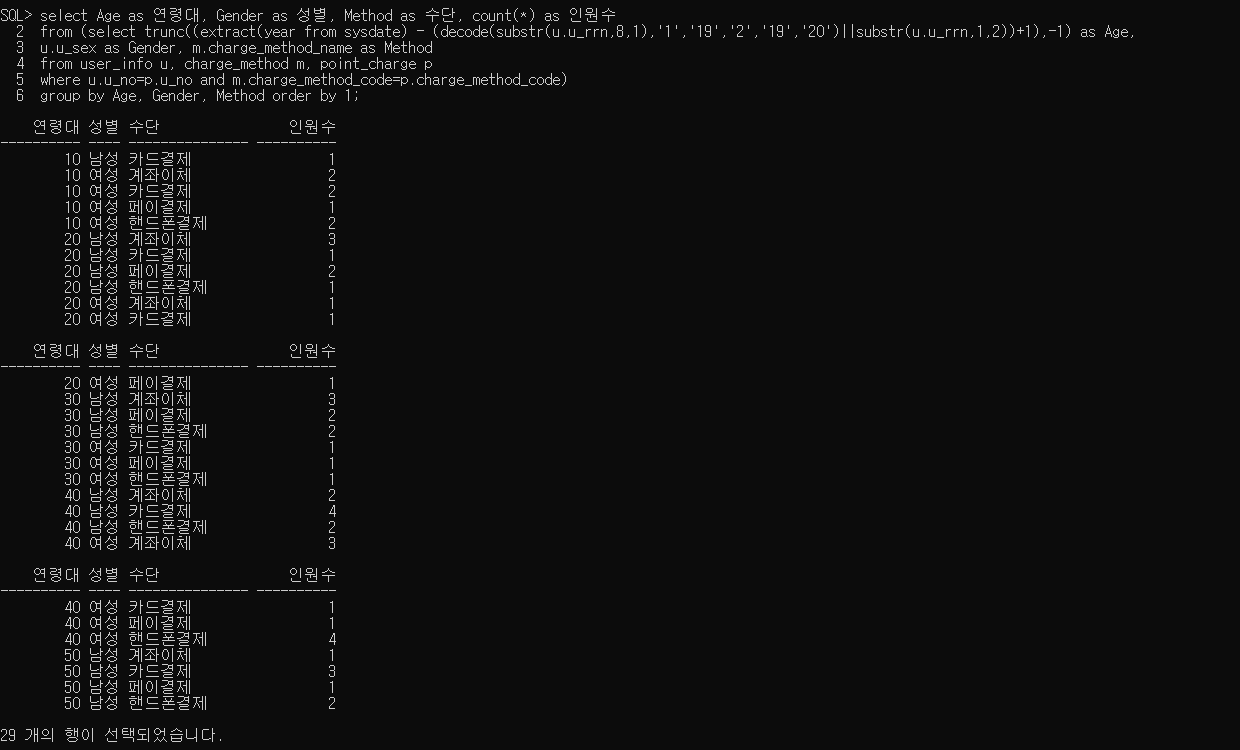
from (select trunc((extract(year from sysdate) - (decode(substr(u.u\_rrn,8,1),'1','19','2','19','20')||substr(u.u\_rrn,1,2))+1),-1) as Age,

u.u\_sex as Gender, m.charge\_method\_name as Method

from user\_info u, charge\_method m, point\_charge p

where u.u\_no=p.u\_no and m.charge\_method\_code=p.charge\_method\_code)

group by Age, Gender, Method order by 1;



**14. 최근 3개월 간 회원들의 공유로 지급된 포인트 총 합계와**

**결제를 통해 지급된 포인트의 총 합계 비례식을 추출**

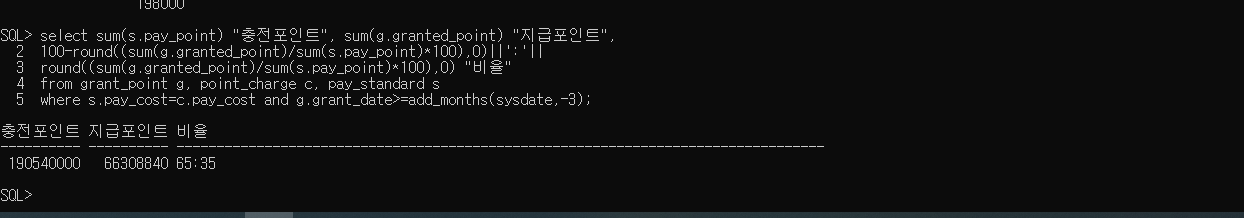
select sum(s.pay\_point) "충전포인트", sum(g.granted\_point) "지급포인트",

100-round((sum(g.granted\_point)/sum(s.pay\_point)\*100),0)||':'||

round((sum(g.granted\_point)/sum(s.pay\_point)\*100),0) "비율"

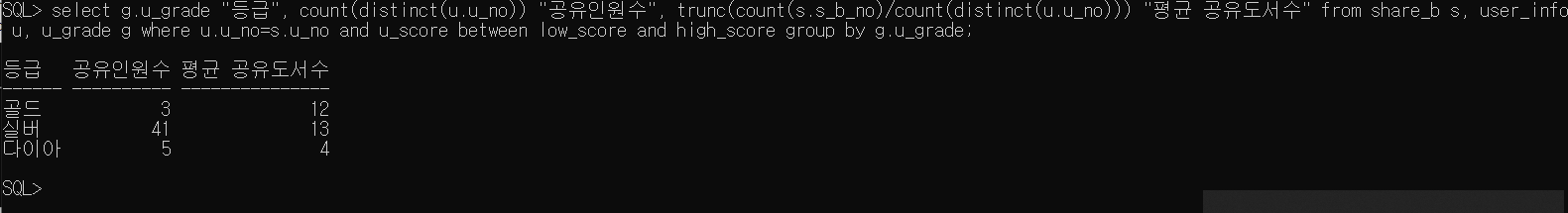
from grant\_point g, point\_charge c, pay\_standard s

where s.pay\_cost=c.pay\_cost and g.grant\_date>=add\_months(sysdate,-3);



**15. 회원의 등급별 각 공유 인원수와 평균 공유도서수 출력**

select g.u\_grade "등급", count(distinct(u.u\_no)) "공유인원수", trunc(count(s.s\_b\_no)/count(distinct(u.u\_no))) "평균 공유도서수" from share\_b s, user\_info u, u\_grade g where u.u\_no=s.u\_no and u\_score between low\_score and high\_score group by g.u\_grade;



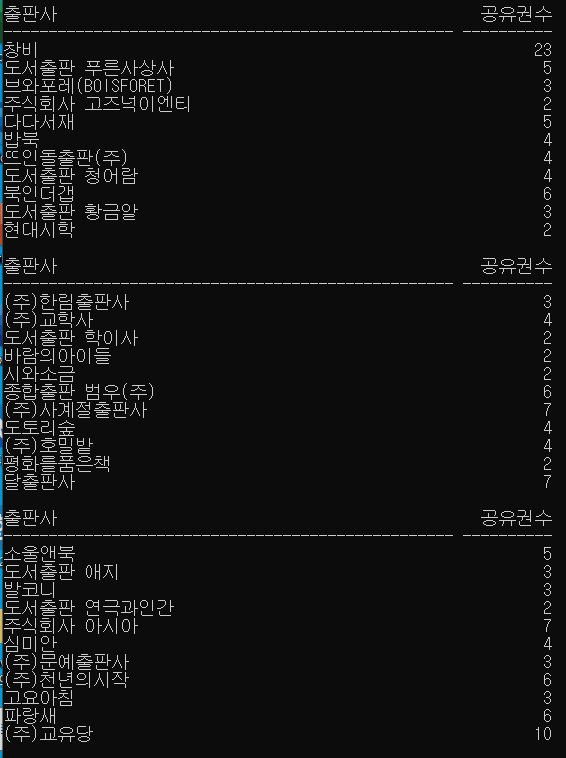
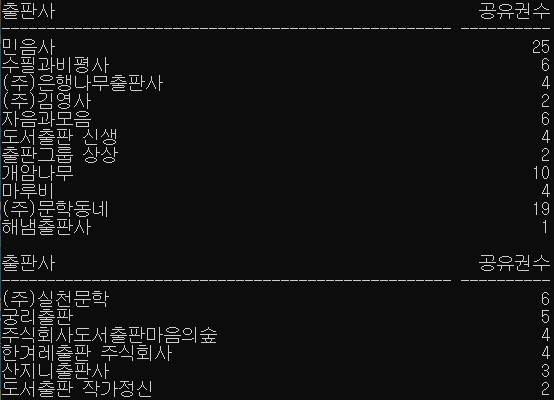
**16. 출판사별 공유현황**

select s.s\_b\_publisher "출판사", count(\*) "공유권수"

from share\_b s, b\_list t

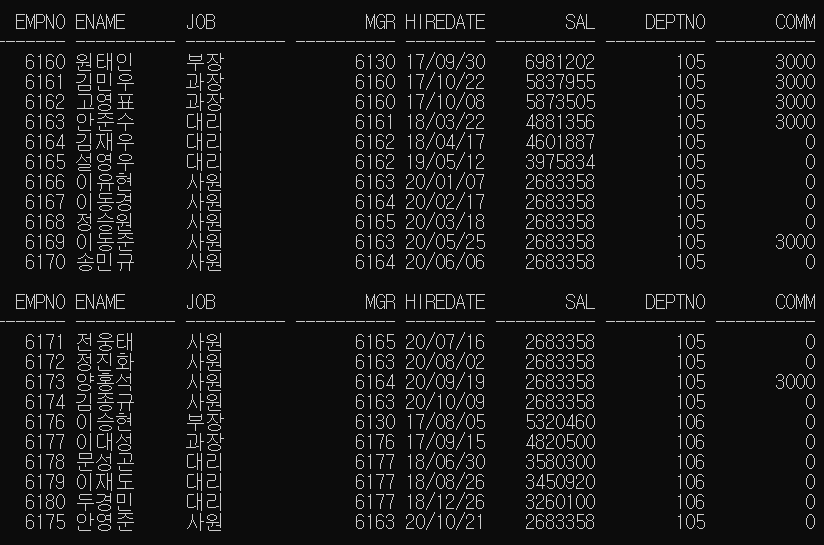
where s.s\_b\_no=t.s\_b\_no

group by s\_b\_publisher;

****

**17. 수령일이 주말이라면 해당 배송 담당자에게 건당 3천원 보너스 지급**

update emp set comm = comm+3000 where empno in (select empno from (select empno, to\_char(dlv\_date, 'day') day from delivery) where day = '토요일' or day ='일요일');



**< PL/SQL >**

**1. 회원 점수가 0 이하인 회원의 대출가능여부를 false로 변환하는 프로시저 생성**

create or replace procedure p\_rent\_ck

(v\_u\_no user\_info.u\_no%type, v\_u\_score user\_info.u\_score%type)

is

begin

if v\_u\_score < 0 then

UPDATE user\_info SET u\_check ='F' where u\_no=v\_u\_no;

end if;

end;

/

**2. 도서목록 table 내 모든 도서의 대출비용을 10% 차감하는 커서 정의**

declare

cursor mycur is select b\_no, rent\_cost from b\_list;

cnt number(5);

begin

for i in mycur loop

update b\_list set rent\_cost=rent\_cost\*0.9 where b\_no=i.b\_no;

cnt := mycur%rowcount;

end loop;

commit;

end;

/

**3. 회원이 도서대출 신청시 해당 도서의 대출가능 여부를 N으로 변경하고,**

**대출비용만큼 회원의 보유포인트를 차감하는 트리거 생성**

create or replace trigger t\_book\_up

after insert on b\_rent

for each row

declare

u\_point user\_info.u\_point%type;

rent\_cost b\_list.rent\_cost%type;

v\_u\_point user\_info.u\_point%type;

begin

UPDATE b\_list SET rent\_check='N' WHERE b\_no=:new.b\_no;

v\_u\_point := u\_point - rent\_cost;

UPDATE user\_info SET u\_point=v\_u\_point WHERE u\_no=:new.u\_no;

end;

/

**4. 현재 대출가능한 도서 중 등록일이 3개월 지난 도서의 대출 비용을 15% 삭감**

declare

cursor sale\_cur is select rent\_cost, rent\_check, s\_start, s\_b\_no from b\_list;

v\_rent\_cost b\_list.rent\_cost%type;

v\_rent\_check b\_list.rent\_check%type;

v\_s\_start b\_list.s\_start%type;

v\_s\_b\_no b\_list.s\_b\_no%type;

begin

if sale\_cur%isopen then

close sale\_cur;

end if;

open sale\_cur;

loop

fetch sale\_cur into v\_rent\_cost, v\_rent\_check, v\_s\_start, v\_s\_b\_no;

exit when(sale\_cur%notfound);

if v\_rent\_check='Y' and v\_s\_start<(add\_months(sysdate,-3)) then

v\_rent\_cost := v\_rent\_cost \* 0.85;

end if;

update b\_list set rent\_cost=v\_rent\_cost where s\_b\_no=v\_s\_b\_no;

end loop;

close sale\_cur;

end;

/

**5. 현재 대출가능한 도서 중 등록일이 12개월 지난 도서 삭제**

declare

cursor del\_cur is select rent\_cost, rent\_check, s\_start, s\_b\_no from b\_list;

v\_rent\_cost b\_list.rent\_cost%type;

v\_rent\_check b\_list.rent\_check%type;

v\_s\_start b\_list.s\_start%type;

v\_s\_b\_no b\_list.s\_b\_no%type;

begin

if del\_cur%isopen then

close del\_cur;

end if;

open del\_cur;

loop

fetch del\_cur into v\_rent\_cost, v\_rent\_check, v\_s\_start, v\_s\_b\_no;

exit when(del\_cur%notfound);

if v\_rent\_check='Y' and v\_s\_start<(add\_months(sysdate,-12)) then

delete from b\_list where s\_b\_no=v\_s\_b\_no;

end if;

end loop;

close del\_cur;

end;

/

**6. 대출 신청시 대출비용보다 해당 회원의 보유포인트가 적다면**

**대출 신청목록에서 신청한 도서를 삭제하고, 알림 메시지 출력**

create or replace trigger t\_book\_rent

before insert on b\_rent

for each row

declare

v\_point user\_info.u\_point%type;

v\_cost b\_list.rent\_cost%type;

begin

select u\_point into v\_point from user\_info where u\_no=:new.u\_no;

select rent\_cost into v\_cost from b\_list where b\_no=:new.b\_no;

dbms\_output.put\_line(v\_point|| ' ' ||v\_cost);

if v\_point < v\_cost then

raise\_application\_error(-20001, '보유포인트가 부족합니다.');

end if;

end;

/

**7. 오늘이 문화의 날(10월16)일 이라면, 모든 도서 대출 비용을 10%할인해주는**

**커서 정의**

declare

cursor mycur is select b\_no, rent\_cost from b\_list;

v\_rent\_cost b\_list.rent\_cost%type;

rent\_cost b\_list.rent\_cost%type;

cnt number(5);

begin

for i in mycur loop

if (to\_char(sysdate,'mm/dd')='10/16') then

v\_rent\_cost := i.rent\_cost\*0.9;

end if;

update b\_list set rent\_cost=v\_rent\_cost where b\_no=i.b\_no;

cnt := mycur%rowcount;

end loop;

commit;

end;

/

**8. 도서대출 table에서 반납만기일을 넘긴 도서를 연체관리 table에 입력하는**

**커서 정의**

create sequence over\_s1

increment by 1

start with 1

minvalue 401

maxvalue 99999

nocycle

nocache;

declare

cursor over\_cur is select return\_expire\_date, b\_rent\_no from b\_rent;

v\_r\_expire b\_rent.return\_expire\_date%type;

v\_b\_rent\_no b\_rent.b\_rent\_no%type;

v\_o\_date b\_rent.b\_rent\_no%type;

v\_o\_charge overdue.over\_charge%type;

cnt number(10) := 1;

begin

if over\_cur%isopen then

close over\_cur;

end if;

open over\_cur;

loop

fetch over\_cur into v\_r\_expire, v\_b\_rent\_no;

exit when(over\_cur%notfound);

select to\_date(to\_char(sysdate, 'yyyymmdd')) - to\_date(to\_char(return\_expire\_date, 'yyyymmdd')) into v\_o\_date from b\_rent where b\_rent\_no=cnt;

cnt:=cnt+1;

v\_o\_charge := v\_o\_date\*100;

if v\_r\_expire<sysdate and v\_o\_date > 1 then

insert into overdue values(over\_s1.nextval, v\_o\_date, v\_o\_charge, v\_b\_rent\_no);

end if;

end loop;

close over\_cur;

end;

/

**9. 반납수거 시 해당 도서를 도서반납 table 목록에서 삭제하고,**

**도서목록 table에서 해당 도서의 대출가능 여부를 Y로 변경하는 트리거 생성**

create or replace trigger t\_return\_pick

after insert on return\_pick

for each row

begin

update b\_list set rent\_check='Y' where b\_no=(select r.b\_no from b\_rent r, b\_return br where (r.b\_rent\_no = br.b\_rent\_no) and (br.return\_no=:new.return\_no));

delete from b\_return where return\_no=:new.return\_no;

end;

/

**10. 도서공유등록 table에 새로운 도서를 입력하는 프로시저 생성**

**등록도서번호는 시퀀스 이용**

create sequence b\_no\_s1

increment by 1

start with 1601

minvalue 1601

maxvalue 99999

nocycle

nocache;

create or replace procedure new\_b\_insert

( p\_s\_b\_title share\_b.s\_b\_title%type,

p\_s\_b\_author share\_b.s\_b\_author%type,

p\_s\_b\_publisher share\_b.s\_b\_publisher%type,

p\_s\_b\_cost share\_b.s\_b\_cost%type,

p\_u\_name user\_info.u\_name%type )

is

s share\_b%rowtype;

u user\_info%rowtype;

begin

insert into share\_b(s\_b\_no, s\_b\_title, s\_b\_author, s\_b\_publisher, s\_b\_cost, s\_b\_register\_date, u\_no)

values (b\_no\_s1.nextval, p\_s\_b\_title, p\_s\_b\_author, p\_s\_b\_publisher, p\_s\_b\_cost, sysdate, (select u\_no from user\_info where u\_name=p\_u\_name));

end;

/

**11. 회원이 공유한 도서가 도서목록에 입력될 시 회원의 보유포인트를 증가하는**

**트리거 생성**

create sequence g\_p\_no\_s1

increment by 1

start with 601

minvalue 1

maxvalue 99999

nocycle

nocache;

create or replace trigger t\_b\_list

after insert on b\_list

for each row

begin

update user\_info set u\_point=(u\_point+(:new.rent\_cost\*3))

where u\_no=(select u\_no from share\_b where s\_b\_no=:new.s\_b\_no);

insert into grant\_point(grant\_point\_no, grant\_date, granted\_point, total\_s\_b\_count, b\_no, u\_no)

values (g\_p\_no\_s1.nextval, sysdate, (:new.rent\_cost\*3),

(nvl((select total\_s\_b\_count from grant\_point where grant\_date=(select max(grant\_date) from grant\_point where u\_no=(select u\_no from share\_b where s\_b\_no=:new.s\_b\_no))),0)+(:new.rent\_cost\*3)),

:new.b\_no, (select u\_no from share\_b where s\_b\_no=:new.s\_b\_no));

end;

/

**12. 입력받은 기간(숫자)으로부터 오늘 날짜까지 등록된 도서 출력하는**

**프로시저 생성**

create or replace procedure d\_b\_list

( v\_date number = 1; )

is

v b\_list%rowtype;

cursor b\_list\_get is

select b\_no, s\_start, rent\_check, b\_grade, rent\_cost from b\_list where s\_start >= sysdate - v\_date;

begin

open b\_list\_get;

loop

fetch b\_list\_get into v.b\_no, v.s\_start, v.rent\_check, v.b\_grade, v.rent\_cost;

exit when b\_list\_get%notfound;

dbms\_ouput.put\_line('도서번호 : '||v.bno||' 공유시작일 : '||v.s\_start||' 대출가능여부 : ' ||v.rent\_check||' 도서등급 : '||v.b\_grade||' 도서대출비용 : '||v.rent\_cost);

end loop;

end;

/

**13. 결제 발생하면 포인트 변경하는 트리거 생성**

create or replace trigger t\_charge\_point

after insert on point\_charge

for each row

declare

v\_point pay\_standard.pay\_point%type;

begin

update user\_info set u\_point=(u\_point+(select pay\_point from pay\_standard

where pay\_cost=:new.pay\_cost)) where u\_no=:new.u\_no;

end;

/

**14. 입력한 회원 테이블의 회원점수를 변경할 때, 그 기준 테이블의 범위를 벗어날 경우 메세지와 회원정보를 출력하는 프로시저 생성**

create or replace procedure p\_u\_u\_u\_score

( v\_u\_no user\_info.u\_no%type,

v\_u\_score user\_info.u\_score%type )

is

user\_info\_score\_ck EXCEPTION;

v\_score user\_info.u\_score%type;

v\_grade u\_grade.u\_grade%type;

v\_u\_grade u\_grade.u\_grade%type;

begin

select u\_score into v\_score from user\_info where u\_no=v\_u\_no;

select u\_grade into v\_grade from u\_grade where v\_score between low\_score and high\_score;

select u\_grade into v\_u\_grade from u\_grade where v\_u\_score between low\_score and high\_score;

if v\_grade<>v\_u\_grade then

raise user\_info\_score\_ck;

else

update user\_info set u\_score = v\_u\_score where u\_no=v\_u\_no;

end if;

EXCEPTION

when user\_info\_score\_ck then dbms\_output.put\_line('범위 안의 점수가 아닙니다!');

when no\_data\_found then dbms\_output.put\_line('해당 회원이 존재하지 않습니다!');

when too\_many\_rows then dbms\_output.put\_line('해당 회원은 중복된 회원입니다!');

when others then dbms\_output.put\_line('문의바랍니다!');

end;

/

**15. 회원정보의 회원점수를 자동으로 계산해주는 트리거 생성**

**(도서공유등록테이블, 포인트충전, 도서대출, 연체관리) <트리거 4개 필요>**

(1) 도서공유등록테이블

create or replace trigger t\_u\_u\_score\_s\_b

after insert on share\_b

for each row

begin

if inserting then

update user\_info set u\_score=u\_score+100

where u\_no=:new.u\_no;

end if;

end;

/

(2) 포인트 충전

create or replace trigger t\_u\_u\_score\_p\_c

after insert on point\_charge

for each row

begin

if inserting then

update user\_info set u\_score=u\_score+100

where u\_no=:new.u\_no;

end if;

end;

/

(3) 도서 대출

create or replace trigger t\_u\_u\_score\_b\_r

after insert on b\_rent

for each row

begin

if inserting then

update user\_info set u\_score=u\_score+50

where u\_no=:new.u\_no;

end if;

end;

/

(4) 연체 관리

create or replace trigger t\_u\_u\_score\_overdue

after insert on overdue

for each row

begin

if inserting then

update user\_info set u\_score=u\_score-50

where u\_no=(select u\_no from b\_rent where b\_rent\_no=:new.b\_rent\_no);

end if;

end;

/

**16. 수령 및 수거일이 주말(토, 일)이라면 해당하는 배송 담당자의 커미션을 3000원씩 누적하여 더하는 커서 정의**

declare

cursor w3c is select distinct r.empno, e.comm from emp e, return\_pick r where r.empno=e.empno and (to\_char(r.return\_pick\_date, 'day') = '토요일' or to\_char(r.return\_pick\_date, 'day') = '일요일');

v\_r\_no return\_pick.empno%type;

v\_comm emp.comm%type;

v\_comm1 number(5);

cnt number(3);

begin

if w3c%isopen then

close w3c;

end if;

open w3c;

loop

fetch w3c into v\_r\_no, v\_comm;

exit when(w3c%notfound);

v\_comm1 := v\_comm+3000;

update emp set comm = v\_comm1 where empno = v\_r\_no;

dbms\_output.put\_line( '반납담당자번호: ' || v\_r\_no || ', 커미션: ' || v\_comm1);

end loop;

cnt := w3c%rowcount;

commit;

close w3c;

dbms\_output.put\_line('처리된 사원의 수는 '||cnt||'명 입니다!');

end;

/

**17. 오늘이 1일이라면, 모든 사원의 보너스를 0으로 변경하는 커서 정의**

declare

cursor mycurt1c0 is select comm from emp;

v\_comm emp.comm%type;

v\_dd number(2);

cnt number(3);

begin

if mycurt1c0%isopen then

close mycurt1c0;

end if;

open mycurt1c0;

loop

fetch mycurt1c0 into v\_comm;

exit when(mycurt1c0%notfound);

v\_dd := to\_char(sysdate, 'dd');

if v\_dd=01 then

v\_comm := 0;

end if;

update emp set comm = v\_comm;

dbms\_output.put\_line('커미션: '|| v\_comm);

end loop;

cnt := mycurt1c0%rowcount;

commit;

close mycurt1c0;

dbms\_output.put\_line('처리된 사원의 수는 '||cnt||'명 입니다!');

end;

/

**18. 부서번호를 입력 받아 해당 부서의 사원 리스트를 출력하는 커서**

declare

cursor mycurdee is select e.ename, d.dname from emp e, dept d where e.deptno=d.deptno;

v\_ename emp.ename%type;

v\_dname dept.dname%type;

begin

if mycurdee%isopen then

close mycurdee;

end if;

open mycurdee;

loop

fetch mycurdee into v\_ename, v\_dname;

exit when(mycurdee%notfound);

dbms\_output.put\_line('이름: '||v\_ename || ', 부서:' || v\_dname);

end loop;

close mycurdee;

end;

/

**19. 새로운 사원정보 입력하는 프로시저**

create or replace procedure p\_emp\_in(

v\_no emp.empno%type,

v\_name emp.ename%type,

v\_job emp.job%type,

v\_mgr emp.mgr%type,

v\_sal emp.sal%type,

v\_deptno emp.deptno%type,

v\_comm emp.comm%type)

is

e emp%rowtype;

begin

insert into emp(empno, ename, job, mgr, hiredate, sal, deptno, comm)

values(v\_no, v\_name, v\_job, v\_mgr, sysdate, v\_sal, v\_deptno, v\_comm);

commit;

end;

/

**20. 급여값 입력받아 세금(3.3%)을 제한 실수령액 계산하는 함수**

CREATE OR REPLACE FUNCTION f\_s\_t\_l (v\_sal number)

RETURN number

IS

v\_v\_sal number(10);

BEGIN

v\_v\_sal := v\_sal\*0.967;

RETURN v\_v\_sal;

END;

/

**21. 최저시급을 받는 사원이 존재할때 매년 사원의 급여가**

**당시 년도의 최저시급에 맞게 변경되는 트리거 생성**

<최저시급 표>

create table l\_pay\_list(

l\_no number(5) constraint l\_l\_pk primary key,

l\_sal number(10),

l\_date date default sysdate);

<트리거>

create or replace trigger t\_l\_pay

after insert or update of l\_sal on l\_pay\_list

for each row

begin

if :old.l\_sal<:new.l\_sal then

update emp set sal = :new.l\_sal where sal<:new.l\_sal;

elsif inserting then

update emp set sal = :new.l\_sal where sal<:new.l\_sal;

end if;

end;

/

**22. 2021년 8월 10일부터 8월 17일기간 동안 결제 금액 50000원 발생시,**

**해당 회원에게 포인트 10% 추가 지급하는 트리거 생성**

create or replace trigger point\_plus\_event

after insert on point\_charge

for each row

begin

update user\_info set u\_point=u\_point+

((select pay\_point from pay\_standard where pay\_cost=:new.pay\_cost) + (:new.pay\_cost\*0.1))

where (:new.pay\_cost=50000) and (:new.charge\_date>= to\_date('19/08/10'))

and (:new.charge\_date<= to\_date('21/08/17'));

end;

/

**23. 생일인 회원에게 등급별로 (2000,1500,1000,1000) 포인트 지급하는**

**프로시저 생성**

create or replace procedure pro\_birth

is

begin

update user\_info U set u\_point = (select case when u\_grade='다이아' then 2000

when u\_grade='골드' then 1500 else 1000 end

from u\_grade G where U.u\_point between G.low\_score and G.high\_score)

where to\_date(substr(u\_rrn,1,6),'YYMMDD')=sysdate;

end;

/

**24. 입력된 부서 사원들 급여 7% 인상**

create or replace procedure p\_emp\_pay\_update

( v\_dname dept.dname%type )

is

v\_dno dept.deptno%type;

begin

select deptno into v\_dno from dept where dname=v\_dname;

update emp set sal = sal\*1.07 where deptno=v\_dno;

end;

/