

<SQL 테이블 구성>

```
drop table if exists update_date;  
create table update_date(  
    match_date date primary key  
);
```

```
drop table if exists User;  
create table User(  
    Id varchar(20) primary key,  
    Name varchar(20),  
    Age int,  
    Money dec(10) );
```

```
drop table if exists HomeGround;  
create table HomeGround(  
    Number int primary key,  
    Name varchar(20),  
    Location varchar(20) );
```

```
drop table if exists Team;  
create table Team(  
    Number int primary key,  
    Name varchar(20),  
    HomeGround_number int,  
    foreign key(HomeGround_number) references  
HomeGround(Number)  
);
```

```
drop table if exists Matches;
create table Matches(
    Number int primary key,
    Date date,
    Home_Rate float,
    Away_Rate float,
    Home_Team_Number int,
    Away_Team_Number int,
    Recommend_Team_Number int,

    foreign key(Home_Team_Number) references Team(Number),
    foreign key(Away_Team_Number) references Team(Number),
    foreign key(Recommend_Team_Number) references
Team(Number)
);
```

```
drop table if exists MatchHistory;
create table MatchHistory(
    Number int primary key,
    Match_Number int,
    Win_Team_Number int,
    Loss_Team_Number int,

    foreign key(Match_Number) references Matches(Number),
    foreign key(Win_Team_Number) references Team(Number),
    foreign key(Loss_Team_Number) references Team(Number)
);
```

```
drop table if exists Bet;
create table Bet(
    Match_Number int,
    User_Id varchar(20),
    User_Choice_Team_Number int,
    Money Dec(10),
    foreign key(Match_number)
references Matches(Number),
    foreign key(User_Id) references User(Id),
    foreign key(User_Choice_Team_Number) references
Team(Number),
    primary key(Match_Number, User_Id)
);
```

<SQL 퀴리문 및 프로시저문 정리>

##날짜별로 경기결과 업데이트(배팅 결과 업데이트)

```
DROP PROCEDURE IF EXISTS SP_UPDATE;
```

```
DELIMITER $$
```

```
CREATE PROCEDURE SP_UPDATE(IN cdate varchar(20))
```

```
BEGIN
```

```
    DECLARE chk_count int;
```

```
    set chk_count = 0;
```

```
    select count(*) into chk_count from update_date where match_date = cdate;
```

```
    if chk_count = 0 then
```

```
        start transaction;
```

```
            insert into update_date values (cdate);
```

```
            update user as A inner join
```

```
            ( select user.id, user.money as tot_money, ttt.away_rate, ttt.money from
```

```
            (select tt.match_number, tt.user_id,tt.money, tt.away_team_number, tt.away_rate  from
```

```
            (( select t.match_number, t.user_id, t.money, m.away_team_number, m.away_rate from
```

```
            ( select b1.match_number,b1.user_id, b1.money,
```

```
            b1.user_choice_team_number,d1.win_team_number from
```

```
            (select m1.date, m1.number, mh1.win_team_number from (select * from matches where date =  
            cdate) as m1
```

```
            left join matchhistory as mh1 on m1.number = mh1.number ) as d1
```

```
            left join bet as b1 on b1.match_number = d1.number where user_choice_team_number =
```

```
            win_team_number ) as t
```

```
            left join matches as m on m.number = t.match_number and
```

```
            t.win_team_number=m.away_team_number ) as tt ) where tt.away_rate is not  NULL) as ttt
```

```
            left join user on user.id = ttt.user_id) as B on  A.id =B.id set A.money = A.money +
```

```
            B.money*B.away_rate;
```

```
        update user as A inner join
```

```
        ( select user.id, user.money as tot_money, ttt.home_rate, ttt.money from
```

```
        (select tt.match_number, tt.user_id,tt.money, tt.home_team_number, tt.home_rate  from
```

```
        (( select t.match_number, t.user_id, t.money, m.home_team_number, m.home_rate from
```

```
        ( select b1.match_number,b1.user_id, b1.money,
```

```
        b1.user_choice_team_number,d1.win_team_number from
```

```
        (select m1.date, m1.number, mh1.win_team_number from (select * from matches where date =  
        cdate) as m1
```

```
        left join matchhistory as mh1 on m1.number = mh1.number ) as d1
```

```
        left join bet as b1 on b1.match_number = d1.number where user_choice_team_number =
```

```
        win_team_number ) as t
```

```
        left join matches as m on m.number = t.match_number and
```

```
        t.win_team_number=m.home_team_number ) as tt ) where tt.home_rate is not  NULL) as ttt
```

```
        left join user on user.id = ttt.user_id) as B on  A.id =B.id set A.money = A.money +
```

```
        B.money*B.home_rate;
```

```
        commit;
```

```
        end if;
```

```
        select chk_count;
```

```
    END $$
```

```
DELIMITER ;
```

설명 :

해당 날짜의 경기를 찾고

matchhistory 에서 해당경기의 이긴 팀을 찾음

date number win_team_number 가 나오면

경기번호가 같고, 승리팀을 맞춘 유저의정보를 뽑음

match_number, user_id, money-user_choice_team_number, win_team_number 뽑음

승리팀이 어웨이 팀(홈팀)인 경우 나눠서

match_number, user_id, money, away_team_number, away_rate 를 뽑아옴

다시 조인으로 업데이트 해야할 user 의

id tot_money, away_rate, 배팅한 money 불러옴

그리고 업데이트

<사용자가 배팅을 하는 기능>

```
DROP PROCEDURE IF EXISTS SP_BET;
```

```
DELIMITER $$
```

```
CREATE PROCEDURE SP_BET(IN mnumber int, IN uid varchar(20), IN cteam varchar(20), IN bmoney  
dec(10) )
```

```
    BEGIN
```

```
        DECLARE chkmoney dec(10);
```

```
        DECLARE team_name varchar(20);
```

```
        DECLARE team_number int;
```

```
        DECLARE avail int;
```

```
        DECLARE chk_count int;
```

```
        DECLARE chk_team int;
```

```
        DECLARE chk_date int;
```

```
        DECLARE chk_name date;
```

```
        set chk_count = 0;
```

```
        set chk_date = 0;
```

```
        set avail = 0;
```

```
        set team_name = cteam;
```

```
        set chk_team = 0;
```

```
        select date into chk_name from matches where number = mnumber;
```

```
        select count(*) into chk_date from update_date where chk_name;
```

```
        if chk_date = 0 then
```

```
            select count(*) into chk_team from team where name = cteam;
```

```
            if chk_team = 1 then
```

```
                start transaction;
```

```
                update user set money = money - bmoney where id =
```

```
uid;
```

```
                select money into chkmoney from user where id = uid;
```

```

select number into team_number from team where name
= cteam;

select count(*) into chk_count from bet where
match_number=mnumber and user_id=uid;

if chk_count = 0 then
    insert into Bet(Match_number, user_id,
user_choice_Team_Number, Money) values
(mnumber,uid, team_number,
bmoney);
end if;

if chkmoney >= 0 and chk_count = 0 then
    set avail = 1;
    commit;
elseif chk_count > 0 then
    set avail = 2;
    rollback;
elseif chkmoney < 0 then
    set avail = 3;
    rollback;
end if;

else
    set team_number = 0;
    set avail=4;
end if;

else
    set avail=5;
end if;

select team_number,avail;

end $$
delimiter ;

```

<회원가입>

```
DROP PROCEDURE IF EXISTS SP_JOIN;
DELIMITER $$
CREATE PROCEDURE SP_JOIN( IN inputid varchar(20), IN inputname varchar(20), IN inputage int,
IN inputpwd varchar(20) )
BEGIN
    DECLARE count int;
    select count(*) into count from user where id = inputid;

    IF count = 0 THEN
        insert into User(Id, Name, Age, Passwd) values (inputid, inputname, inputage,
md5(inputpwd));
    END IF;
END $$
DELIMITER ;
```

<로그인>

```
DROP PROCEDURE IF EXISTS SP_LOGIN;
DELIMITER $$
CREATE PROCEDURE SP_LOGIN(IN inputid varchar(20), inputpasswd char(32) )
BEGIN
    DECLARE chk int;
    DECLARE user_name varchar(20);
    select count(*),user_name into chk,user_name from user where id = inputid and
passwd = md5(inputpasswd);

    select chk,user_name;
end $$
delimiter ;
```

<현재 세션의 사용자 정보를 보여줌>

```
DROP PROCEDURE IF EXISTS SP_MY_INFO;
DELIMITER $$
CREATE PROCEDURE SP_MY_INFO( IN inputid varchar(20) )
BEGIN
    DECLARE user_name VARCHAR(20);
    DECLARE user_age int;
    DECLARE user_money int;
    DECLARE chk int;

    select count(*) into chk from user where id = inputid;
    if chk = 1 then
        select name,age,money into user_name,user_age,user_money from user where id =
inputid;
    end if;
    select chk,user_name,user_age,user_money;
END $$
DELIMITER ;
```

<특정 경기에 대한 추천팀 선정>

```
DROP PROCEDURE IF EXISTS SP_RECOMMEND;
```

```
DELIMITER $$
```

```
CREATE PROCEDURE SP_RECOMMEND(IN matchnumber INT)
```

```
BEGIN
```

```
    DECLARE home_win_count INT;
```

```
    DECLARE away_win_count INT;
```

```
    DECLARE home_tot_win_count INT;
```

```
    DECLARE away_tot_win_count INT;
```

```
    DECLARE HOME INT;
```

```
    DECLARE AWAY INT;
```

```
    DECLARE recommend_team_name varchar(20);
```

```
    DECLARE win_count INT;
```

```
    DECLARE loss_count INT;
```

```
    DECLARE tot_win_count INT;
```

```
    DECLARE tot_loss_count INT;
```

```
    declare team_number int;
```

```
    select home_team_number into HOME from matches where number = matchnumber;
```

```
    select away_team_number into AWAY from matches where number = matchnumber;
```

```
    select count(*) into home_win_count from matchhistory where win_team_number =  
HOME and loss_team_number = AWAY;
```

```
    select count(*) into away_win_count from matchhistory where win_team_number =  
AWAY and loss_team_number = HOME;
```

```
    select count(*) into home_tot_win_count from matchhistory where win_team_number  
= HOME;
```

```
    select count(*) into away_tot_win_count from matchhistory where win_team_number  
= AWAY;
```

```
    IF home_win_count > away_win_count then
```

```
        select name into recommend_team_name from team where number = HOME;
```

```
        set win_count = home_win_count;
```

```
        set loss_count = away_win_count;
```

```
        set tot_win_count = home_tot_win_count;
```

```
        select count(*) into tot_loss_count from matchhistory where loss_team_number =  
HOME;
```

```
        SELECT RECOMMEND_TEAM_number into team_number from matches where number =  
matchnumber;
```

```
        IF team_number IS NULL then
```

```
            update matches set recommend_team_number = HOME where number = matchnumber;
```

```
        end if;
```

```
    ELSEIF home_win_count < away_win_count then
```

```
        select name into recommend_team_name from team where number = AWAY;
```

```
        set win_count = away_win_count;
```

```
        set loss_count = home_win_count;
```

```
        set tot_win_count = away_tot_win_count;
```

```
        select count(*) into tot_loss_count from matchhistory where loss_team_number  
= AWAY;
```

```
        SELECT RECOMMEND_TEAM_number into team_number from matches where number =  
matchnumber;
```

```
        IF team_number IS NULL then
```

```

        update matches set recommend_team_number = AWAY where number =
matchnumber ;
    end if;
    ELSE
        IF home_tot_win_count > away_tot_win_count then
            select name into recommend_team_name from team where number = HOME;
            SELECT RECOMMEND_TEAM_number into team_number from matches where number
= matchnumber;
            if team_number IS NULL then
                update matches set recommend_team_number = HOME where number = matchnumber;
                end if;
                set tot_win_count = home_tot_win_count;
                select count(*) into tot_loss_count from matchhistory where
loss_team_number = HOME;
                set win_count = home_win_count;
                set loss_count = away_win_count;
            else
                select name into recommend_team_name from team where number = AWAY;
                SELECT RECOMMEND_TEAM_number into team_number from matches where number =
matchnumber;
                if team_number IS NULL then
                    update matches set recommend_team_number = AWAY where number = matchnumber;
                    end if;
                    set tot_win_count = away_tot_win_count;
                    select count(*) into tot_loss_count from matchhistory where loss_team_number
= HOME;
                    set win_count = home_win_count;
                    set loss_count = away_win_count;
                end if;
            END IF;
            select          win_count,          loss_count,          tot_win_count,
tot_loss_count,RECOMMEND_TEAM_name;

        end $$
delimiter ;
DROP PROCEDURE IF EXISTS SP_SHOW_TEAM;
DELIMITER $$
CREATE PROCEDURE SP_SHOW_TEAM( IN inputid varchar(20) )
BEGIN
    DECLARE user_name VARCHAR(20);
    DECLARE user_age int;
    DECLARE user_money int;
    DECLARE chk int;

    select count(*) into chk from user where id = inputid;
    if chk = 1 then
        select name,age,money into user_name,user_age,user_money from user where id =
inputid;
    end if;
    select chk,user_name,user_age,user_money;
END $$
DELIMITER ;

```

<특정 팀의 정보 보기>

```
select t.name as team_name, g.name as ground_name, g.location as location from (select *  
from team where name  
= '특정팀') as t  
inner join homeground as g on t.homeground_number = g.number;
```

<오늘의 경기 정보 보기>

```
( select M1.number, M1.date,t1.name As Home,t2.name As Away, M1.home_rate, M1.away_rate from  
(select number, date, Home_Team_Number, Away_Team_Number, home_rate,away_rate from matches  
where date = '오늘 날짜') As M1 inner join team as t1 on M1.Home_Team_Number = t1.number  
inner join team as t2 on M1.Away_Team_Number = t2.number) order by M1.number"
```

<성능 개선 사항>

matches 테이블은 date 로 index

->

```
Create index index_date on matches(date)
```

Match_history 테이블은 win_team_number 로 index

->

```
Create index index_win_team_number on  
match_history(win_team_number);
```

user 테이블은 id 로 index

->

```
Create index index_id on user(id);
```

date 테이블은 match_date 로 index

->

```
Create index index_match_date on date(match_date);
```

쿼리문의 효율성 늘림

잘게 쪼갤 수 있는 쿼리문은 잘게 쪼개서 성능을 높임

<dummy data 파일 생성 프로시저>

```
DROP PROCEDURE IF EXISTS SP_DUMMY2;
```

```
DELIMITER $$
```

```
CREATE PROCEDURE SP_DUMMY2()
```

```
BEGIN
```

```
    declare k int;
```

```
    set k = 1;
```

```
    WHILE k < 10000 DO
```

```
        INSERT INTO matches VALUES(k, date_sub('2014-06-22', interval k day),  
2, 2,floor(rand()*100), floor(rand()*900) ,NULL);
```

```
        INSERT INTO matches VALUES(k+1, date_sub('2014-06-22', interval k  
day), 2, 2,floor(rand()*100), floor(rand()*900) ,NULL);
```

```
        INSERT INTO matches VALUES(k+2, date_sub('2014-06-22', interval k  
day), 2, 2,floor(rand()*100), floor(rand()*900) ,NULL);
```

```
        INSERT INTO matches VALUES(k+3, date_sub('2014-06-22', interval k  
day), 2, 2,floor(rand()*100), floor(rand()*900) ,NULL);
```

```
        SET k = k + 4;
```

```
    end while;
```

```
END $$
```

```
DELIMITER ;
```

```
DROP PROCEDURE IF EXISTS SP_DUMMY;

DELIMITER $$

CREATE PROCEDURE SP_DUMMY()
BEGIN
    DECLARE i int;
    SET i = 1;

    WHILE i < 10000 DO

        INSERT INTO team VALUES(i, CONCAT('test-team#',i), 1);
        SET i = i + 1;

    END WHILE;

END $$

DELIMITER ;
```

```
DROP PROCEDURE IF EXISTS SP_DUMMY3;

DELIMITER $$

CREATE PROCEDURE SP_DUMMY3()
BEGIN
    declare k int;
    set k = 1;
    WHILE k < 10000 DO
        INSERT INTO user VALUES(k, concat('name#',k), k, 10000,md5('1234') );
        SET k = k + 1;
    end while;
END $$

DELIMITER ;
```

```

DROP PROCEDURE IF EXISTS SP_DUMMY4;

DELIMITER $$

CREATE PROCEDURE SP_DUMMY4()
BEGIN
    declare k int;
    declare home int;
    declare away int;

    set k = 1;
    WHILE k < 10000 DO
        select home_team_number, away_team_number into home,away from matches where
number = k;
        if (k mod 2) = 1 then
            INSERT INTO matchhistory VALUES(k, k, home, away,NULL );
        else
            INSERT INTO matchhistory VALUES(k, k, away, home,NULL );
        end if;
        SET k = k + 1;
    end while;
END $$

DELIMITER ;

```



```

DROP PROCEDURE IF EXISTS SP_DUMMY5;

DELIMITER $$

CREATE PROCEDURE SP_DUMMY5()
BEGIN
    declare k int;
    declare home int;
    declare away int;

    set k = 1;
    WHILE k < 10000 DO
        select home_team_number, away_team_number into home,away from matches where
number = k;

        if (k mod 2) = 1 then
            INSERT INTO bet VALUES(k, concat('',k), home, 1,0 );
        else
            INSERT INTO bet VALUES(k, concat('',k), away, 1,0 );
        end if;
        SET k = k + 1;
    end while;
END $$

DELIMITER ;

```