```
--A
create table Poet ( --User is a reserved keyword, I will use Poet
    pid int primary key identity (1, 1),
    name varchar(100),
    pen_name varchar(100) unique,
    yob int
)

create table Award (
    aid int primary key identity (1, 1),
    pid int references Poet(pid), --an award belongs to a poet
    name varchar(100)
)

create table Competition (
    cid int primary key identity (1, 1),
    year int,
    week int,
    unique (year, week) --a competition is defined by its year and week, thus unique
)

create table Poem (
    poid int primary key identity (1, 1),
    pid int references Poet(pid),
    cid int references Competition(cid), --pid and cid mean a Poem belongs to a poet and is
submitted to a competition
    title varchar(100),
    text varchar(100)
)

create table Judge (
    jid int primary key identity (1, 1),
    name varchar(100)
)

create table Evaluation (
    poid int references Poem(poid), --The poem evaluated
    jid int references Judge(jid), --The judge evaluating
    score int,
    primary key (poid, jid), --A judge cannot evaluate the same poem twice
    constraint score_between_1and10 check(1 <= score and score <= 10) --The score must be 1
<= S <= 10
)

-- B
create or alter procedure remove_judge(@name varchar(100)) as
    declare judge_cursor cursor for --I use a cursor to execute for each judge at a time
        select jid
        from Judge
        where name = @name
    open judge_cursor
    declare @jid int
    fetch judge_cursor into @jid
    while @@FETCH_STATUS = 0 begin
        delete from Evaluation where jid = @jid
        delete from Judge where jid = @jid
        fetch judge_cursor into @jid
    end
    close judge_cursor
    deallocate judge_cursor
```

```sql
-- C
create or alter view show_competitions as
    select C2.year, C2.week
    from (select C.cid as tcid
        from Evaluation E join Poem P on E.poid = P.poid join Competition C on P.cid = C.cid -- I
join the three tables to get the competition of that evaluation
        where P.poid not in (select distinct E2.poid from Evaluation E2 where E2.score >= 5) -- But
discard the Poems that have at least an evaluation of 5
        group by C.cid
        having count(*) >= 10) T join Competition C2 on T.tcid = C2.cid -- And discard
competitions that have less than 10 of these evaluations

-- D
create or alter function list_users(@P int) returns table return
    select P3.name, P3.pen_name
    from (
        select P.pid as tpid -- I build the ids of the Poets with @P poems
        from Poet P join Poem P2 on P.pid = P2.pid
        group by P.pid
        having count(*) >= @P
    ) T join Poet P3 on T.tpid = P3.pid --Then join with the Poet again to have access to name and
penname

-- Inserts and calls
insert into Poet(name, pen_name, yob)
    values ('John', 'Johnnie', 2001),
           ('Jack', 'Jackie', 2002),
           ('Matt', 'Mattie', 2003)

insert into Competition(year, week)
    values (2014, 21),
           (2015, 22),
           (2020, 1)

insert into Poem (pid, cid, title, text)
    values (1, 1, 'a', 'b'),
           (1, 1, 'c', 'd'),
           (1, 1, 'e', 'f'),
           (2, 1, 'g', 'h'),
           (2, 1, 'i', 'j'),
           (2, 1, 'k', 'l'),
           (3, 1, 'm', 'n'),
           (3, 1, 'o', 'p'),
           (3, 1, 'q', 'r'),
           (1, 1, 's', 't'),
           (1, 1, 'u', 'v'),
           (1, 2, 'w', 'x')

insert into Judge (name)
    values ('Ron'), ('Tim'), ('Ron')

insert into Evaluation (poid, jid, score)
    values (1, 1, 4),
           (1, 2, 3),
           (2, 3, 6),
           (2, 2, 2),
           (3, 1, 4),
           (3, 3, 3),
           (3, 2, 2),
           (4, 1, 1),
           (5, 2, 4),
```

```
        (6, 3, 2),
        (7, 3, 1),
        (8, 2, 3),
        (9, 1, 3),
        (10, 2, 3),
        (11, 3, 1)

select * from Poet
select * from Poem
select * from Competition
select * from Judge
select * from Evaluation

select * from list_users(3)
select * from list_users(4)

select * from show_competitions

execute remove_judge 'Ron'

select * from Poet
select * from Poem
select * from Competition
select * from Judge
```