```
--#set delimiter !

begin atomic
declare changed_ smallint default 1;
while changed_ <> 0 do
    set changed_ = 0;

    for S as
with
    -- HAUL
    -- Per quest, what each actor hauled in # prizes and net worth
    -- of the prizes (such that actor got more than one prize).
    Haul (day, realm, theme, login, #prize, worth) as (
        select L.day, L.realm, L.theme, L.login,
               count(*),
               sum(T.sql)
        from Loot L, Treasure T
        where L.treasure = T.treasure
          and login is not null
        group by day, realm, theme, login
        having count(*) > 1
    ),
    -- DONOR
    -- Per quest, Actor with the most prizes.
    -- In case of a tie for most prizes, then the actor whose prizes
    -- are, in sum, worth the most.
    -- In case of a tie in # prizes and worth, (unfairly) just target
    -- the Actor whose login would come last in the list ordered by
    -- dictionary (e.g., 'Franck' < 'Parke').
    Donor (day, realm, theme, login, #prize, worth) as (
        select day, realm, theme, login, #prize, worth
        from Haul H
        where not exists (
                select *
                from Haul J
                where H.day   = J.day
                  and H.realm = J.realm
                  and H.theme = J.theme
                  and ((J.#prize > H.#prize)
                        or
                       (J.#prize = H.#prize and J.worth > H.worth)
                        or
                       (J.#prize = H.#prize
                            and J.worth = H.worth
                            and J.login > H.login))
            )
    ),
    -- DONATION
    -- Per quest, pick the prize given to the Donor that is worth the
    -- least.  In case of ties, pick the prize by name (treasure)
    -- that comes last in dictionary order (e.g., 'left sandal'
    -- < 'right sandal').  If we stil have a tie (because the Donor
    -- received more than one of such prizes, pick the one with the
    -- greatest loot#.
    Donation (day, realm, theme, login, loot#, treasure, sql) as (
        select D.day, D.realm, D.theme, D.login,
               L.loot#, L.treasure, T.sql
        from Donor D, Loot L, Treasure T
        where D.day   = L.day    -- join of D and L
          and D.realm = L.realm
          and D.theme = L.theme
          and D.login = L.login
          and L.treasure = T.treasure -- join of L and T
          and not exists (
                select *
                from Loot M, Treasure S
                where M.day   = L.day    -- join of M and L
                  and M.realm = L.realm
                  and M.theme = L.theme
                  and M.login = L.login
                  and M.treasure = S.treasure
                  and ((S.sql < T.sql)
                        or
                       (S.sql = T.sql and S.treasure > T.treasure)
                        or
                       (S.sql = T.sql
                            and S.treasure = T.treasure
                            and M.loot# > L.loot#))
              )
    ),

   Recipient as (select day, realm, theme, min(login) as login
        from Actor
        where (day, realm, theme, login) not in (
            select day, realm, theme, login
            from Loot
        )
        group by day, realm, theme
        order by day, realm, theme)
   select a.day, a.realm, a.theme, a.loot#, a.treasure, b.login from
Donation a, Recipient B where a.day=b.day and a.realm=b.realm and
a.theme=b.theme


    do
        update loot set login=s.login where loot.day=s.day and
loot.realm=s.realm and loot.theme=s.theme and loot.loot#=s.loot#;

        set changed_ = 1;
    end for;
    -- handle each swap

end while;
-- loop until no changes remain to be done
-- we're done

end!
```