Note: For this task I used PostgreSQL

Query for Task 3.1 – Create column transaction\_number with numerating transactions each user:

select \*, row\_number() over (partition by "user id" order by "purchase date") as transaction\_number

into alteredtesttasktable

from (select distinct \* from testtasktable) as new\_table;

Result: deleting all duplicates from main table, creation column transaction\_number with row\_number function in new table alteredtesttasktable with 58094 rows.

Query for Task 3.2 – Get user id of users, with first transaction being not refunded, and second refunded:

select "user id"

from alteredtesttasktable

where transaction\_number=2 and refunded=true and ("user id" in (select "user id" from alteredtesttasktable where transaction\_number=1 and refunded=False));

Result: get column of appropriate user ids with 438 rows.

Query for Task 3.0 – Calculate convert rate from trial to second successful payment:

Remark: In this task I wasn’t clearly confident if the second successful payment is transaction\_number 2 or 3 (should I take into account trial period as payment), and should I include refunded ones into processing, so I calculated all 4 variants here

Count the number of trial periods – Task 3.0.1:

select count(\*)

from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer') as new\_table

where trial=true;

Result: 15.

Count the number of successful second payments (transaction\_number = 2), including refunded, for users with trial – Task3.0.2\_1:

select count(\*)

from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer') as new\_table

where trial=true and refunded=false and check\_trial\_first("user id", "purchase date", transaction\_number);

Result: 4 users, rate – 26,67%.

Count the number of successful second payments (transaction\_number = 2), without refunded, for users with trial - – Task3.0.2\_2:

select count(\*)

from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer') as new\_table

where trial=true and refunded=false and check\_trial\_first\_no\_refund("user id", "purchase date", transaction\_number);

Result: 4 users, rate – 26,67%.

Count the number of successful second payments after the trial (transaction\_number = 3), including refunded, for users with trial – Task3.0.2\_3:

select count(\*)

from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer') as new\_table

where trial=true and refunded=false and check\_trial\_second("user id", "purchase date", transaction\_number);

Result: 3 users, rate – 20%.

Count the number of successful second payments after the trial (transaction\_number = 3), without refunded, for users with trial – Task3.0.2\_4:

select count(\*)

from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer') as new\_table

where trial=true and refunded=false and check\_trial\_second\_no\_refund("user id", "purchase date", transaction\_number);

Result: 2 users, rate – 13,33%.

Functions used in Task 3.0:

Check\_trial\_first function:

CREATE OR REPLACE FUNCTION public.check\_trial\_first(

    userid integer,

    purchasedate date,

    transactionnumber bigint)

    RETURNS boolean

    LANGUAGE 'plpgsql'

    COST 100

    VOLATILE PARALLEL UNSAFE

AS $BODY$

declare

    passed integer;

begin

    select count(\*) into passed

    from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer' and "user id"=userid) as new\_table

    where trial=false and "purchase date"=purchasedate + (7 \* interval '1 day') and transaction\_number>=transactionnumber+1;

    return passed>0;

end;

$BODY$;

Check\_trial\_first\_no\_refund function:

CREATE OR REPLACE FUNCTION public.check\_trial\_first\_no\_refund(

    userid integer,

    purchasedate date,

    transactionnumber bigint)

    RETURNS boolean

    LANGUAGE 'plpgsql'

    COST 100

    VOLATILE PARALLEL UNSAFE

AS $BODY$

declare

    passed integer;

begin

    select count(\*) into passed

    from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer' and "user id"=userid) as new\_table

    where trial=false and refunded=false and "purchase date"=purchasedate + (7 \* interval '1 day') and transaction\_number>=transactionnumber+1;

    return passed>0;

end;

$BODY$;

Check\_trial\_second function:

CREATE OR REPLACE FUNCTION public.check\_trial\_second(

    userid integer,

    purchasedate date,

    transactionnumber bigint)

    RETURNS boolean

    LANGUAGE 'plpgsql'

    COST 100

    VOLATILE PARALLEL UNSAFE

AS $BODY$

declare

    passed integer;

begin

    select count(\*) into passed

    from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer' and "user id"=userid) as new\_table

    where trial=false and "purchase date"=purchasedate + (14 \* interval '1 day') and transaction\_number>=transactionnumber+2;

    return passed>0;

end;

$BODY$;

Check\_trial\_second\_no\_refund function:

CREATE OR REPLACE FUNCTION public.check\_trial\_second\_no\_refund(

    userid integer,

    purchasedate date,

    transactionnumber bigint)

    RETURNS boolean

    LANGUAGE 'plpgsql'

    COST 100

    VOLATILE PARALLEL UNSAFE

AS $BODY$

declare

    passed integer;

begin

    select count(\*) into passed

    from (select \* from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer' and "user id"=userid) as new\_table

    where trial=false and refunded=false and "purchase date"=purchasedate + (14 \* interval '1 day') and transaction\_number>=transactionnumber+2;

    return passed>0;

end;

$BODY$;

Alternative tables:

Table of user ids with successful second payment after trial (result – 4 rows) – AlternativeTable1:

select distinct "user id"

from alteredtesttasktable

where "user id" in (select "user id" from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer' and trial=true and refunded=false) and "product id"='tenwords\_1w\_9.99\_offer'

and trial=False;

Table of all transactions made by users with trials – AlternativeTable2:

select \*

from alteredtesttasktable

where "user id" in (select "user id" from alteredtesttasktable where "product id"='tenwords\_1w\_9.99\_offer' and trial=true) and "product id"='tenwords\_1w\_9.99\_offer';