тестовое_задание-циан-sql

Задание 1

```
-- Используется диалект PostgreSQL.

SELECT id_user, foo.id_экземпляр, дата_взяли, cnt
FROM
-- Ранжирование взятых произведений
  (SELECT id_user, id_экземпляр, дата_взяли, ROW_NUMBER()
   OVER (PARTITION BY id_user ORDER BY дата_взяли DESC) AS pos
   FROM operations) AS foo
LEFT JOIN
-- Кол-во взятий каждого произведения
  (SELECT id_экземпляр, COUNT(id_экземпляр) AS cnt
   FROM operations
   GROUP BY id_экземпляр) AS bar
ON foo.id_экземпляр = bar.id_экземпляр
WHERE pos < 4;
```

Задание 2

2.1

```
-- Используется диалект PostgreSQL.

WITH t1
AS
-- ID клиента, Дата и сумма первой покупки, Дата и сумма повторной покупки, Дата последней покупки
(
    SELECT DISTINCT
           ClientID,
           FIRST_VALUE(OperationTime) OVER w AS FirstTime,
           FIRST_VALUE(Amount) OVER w AS FirstAmount,
           NTH_VALUE(OperationTime, 2) OVER w AS SecondTime,
           NTH_VALUE(Amount, 2) OVER w AS SecondAmount,
           LAST_VALUE(OperationTime) OVER w AS LastTime
    FROM tblClientBalanceOperation
    WHERE ClientID NOT IN (SELECT CLientID FROM tblTestClients)
    WINDOW w AS (
                 PARTITION BY ClientID ORDER BY OperationTime
                 RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING
                )
)
,
t2
AS
-- Сумма покупок, совершенных в течение месяца после первой покупки
(
    SELECT ClientID,
           SUM(Amount) AS MonthSum
    FROM
         (
          SELECT BalanceOperationID,
                 ClientID,
                 Amount,
                 OperationTime,
                 FIRST_VALUE(OperationTime) OVER w AS FirstTime
           FROM tblClientBalanceOperation
           WINDOW w AS (
                        PARTITION BY ClientID ORDER BY OperationTime
                        RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING
                       )
        ) AS foo

    WHERE OperationTime < FirstTime + 31
    GROUP BY ClientID
)
,
t3
AS
-- Среднее время (кол-во дней) между покупками
```

```
(
    SELECT ClientID,
           AVG(OperationTime - LagTime) AS AvgDeltaTime
    FROM
       (
        SELECT BalanceOperationID,
               ClientID,
               OperationTime,
               LAG(OperationTime) OVER w As LagTime
        FROM tblClientBalanceOperation
        WINDOW w AS (PARTITION BY ClientID ORDER BY OperationTime)
       ) AS foo
    GROUP BY ClientID
)

SELECT t1.ClientID,
       FirstTime,
       FirstAmount,
       SecondTime,
       SecondAmount,
       LastTime,
       MonthSum,
       (SecondTime-FirstTime) AS DeltaFirstSecond,
       AvgDeltaTime
FROM t1 LEFT JOIN t2 USING(ClientID)
    LEFT JOIN t3 USING(ClientID)
```

2.2

```
-- Используется диалект PostgreSQL.

WITH t1
AS
-- Кол-во потерянных клиентов (по месяцам)
(
    SELECT MonthLosted, COUNT(MonthLosted) AS cntLosted
    FROM
    (
         SELECT DISTINCT ClientID,
                         date_part('month', OperationTime + interval '1 month') AS MonthLosted
         FROM tblClientBalanceOperation AS foo
         WHERE OperationTime <= date(now()) - interval '1 month' AND
               OperationTime >= date(to_char(now() - interval '1 year', 'YYYY') || '-12-01') AND
               NOT EXISTS (
                           SELECT ClientID
                           FROM tblClientBalanceOperation
                           WHERE OperationTime <= date(now()
                             AND OperationTime >= date(to_char(now(), 'YYYY') || '-01-01')
                             AND date_part('month', OperationTime) = date_part('month', foo.OperationTime + interval '1 month')
                             AND ClientID = foo.ClientID
                          )
    ) AS bar
    GROUP BY MonthLosted
    ORDER BY MonthLosted
)
,
t2
AS
-- Кол-во клиентов потерянных и находящихся в статусе 'Deleted' (по месяцам)
(
    SELECT MonthLosted,
           COUNT(MonthLosted) AS cntLostedAndDeleted
    FROM
      (
       SELECT DISTINCT ClientID,
                       date_part('month', OperationTime + interval '1 month') AS MonthLosted
       FROM tblClientBalanceOperation AS foo
       WHERE OperationTime <= date(now()) - interval '1 month'
         AND OperationTime >= date(to_char(now() - interval '1 year', 'YYYY') || '-12-01')
         AND NOT EXISTS (
                         SELECT ClientID
                         FROM tblClientBalanceOperation
                         WHERE OperationTime <= date(now())
                           AND OperationTime >= date(to_char(now(), 'YYYY') || '-01-01')
                           AND date_part('month', OperationTime) = date_part('month', foo.OperationTime + interval '1 month')
                           AND ClientID = foo.ClientID
                         )
         AND EXISTS (
```

```
                    SELECT ClientID
                    FROM tblclients
                    WHERE ClientID = foo.ClientID
                    )
       ) AS bar
       GROUP BY MonthLosted
       ORDER BY MonthLosted
)

SELECT t1.MonthLosted,
       COALESCE(cntLostedAndDeleted, 0) AS cntLostedAndDeleted,
       cntLosted,
       COALESCE((cntLostedAndDeleted::float / cntLosted::float) * 100, 0) AS PercentDeleted
FROM t1 LEFT JOIN t2 USING(MonthLosted)
```

2.3

```
-- Используется диалект PostgreSQL.

WITH foo
AS
-- Первая покупка
(
    SELECT DISTINCT ClientID,
                    FIRST_VALUE(OperationTime)
                    OVER(PARTITION BY ClientID ORDER BY OperationTime
                    RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING)
                    AS FirstOperation
    FROM tblClientBalanceOperation
)
,
bar
AS
-- Регистрация на сайте
(
    SELECT DISTINCT ClientID,
                    FIRST_VALUE(OnlineTime)
                      OVER (PARTITION BY ClientID ORDER BY OnlineTime
                      RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING)
                      AS RegTime
    FROM tblOnlineSessions_mini
)

SELECT DISTINCT numWeek,
     COUNT(ClientID) OVER (PARTITION BY numWeek) AS cntClients,
     COUNT(ClientID) OVER (PARTITION BY numWeek)::float /
        COUNT(ClientID) OVER ()::float
        AS propClients
FROM
(
    SELECT ClientID,
           CASE WHEN FirstOperation - RegTime >= 0
                 AND FirstOperation - RegTime <= 7  THEN 1

                WHEN FirstOperation - RegTime >  7
                 AND FirstOperation - RegTime <= 14 THEN 2

                WHEN FirstOperation - RegTime >  14
                 AND FirstOperation - RegTime <= 21 THEN 3

                WHEN FirstOperation - RegTime >  21 THEN 4

                ELSE NULL
           END AS numWeek
    FROM foo LEFT JOIN bar USING (ClientID)
) AS foobar
```