Data Engineer Associate Exam - Virtual Reality Fitness

ActiVR provides a virtual reality device designed for exercise and fitness.

ActiVR offers a range of products, including VR devices and subscription-based fitness programs through their apps.

The sales team at ActiVR wants to analyze user data to enhance their marketing strategy and evaluate their products. For this, it is crucial that the data is clean, accurate, and available for reporting.

They need your assistance in preparing the data before launching a new promotional campaign.

Database Schema

The data schema for ActiVR's database is outlined as follows:

- **events:** Contains records of events registered in different games.
- **games:** Stores information about various games available on the platform.
- **devices:** Holds data about the virtual reality devices used by the users.
- **users:** Contains details about the users utilizing the ActiVR platform.

Task 1

ActiVR's sales team wants to use the information it has about users for targeted marketing.

However, they suspect that the data may need to be cleaned before.

The expected data format and types for the `users` table according to the sales team's requirements is shown in the table below.

Write an SQL query that returns the `users` table with the specified format. Ensure that your query does not modify the `users` table.

Column Name	Description
user_id	Unique integer (assigned by the database, cannot be altered). Missing values are not possible due to the database structure.
age	Integer representing the age of the customer. Missing values should be replaced with the average age.
registration_date	Date when the user made an account first (YYYY-MM-DD). Missing values should be replaced with January 1st, 2024.
email	Email address of the user. Missing values should be replaced with Unknown.
workout_frequenc	Workout frequency as a lowercase string, one of: *minimal*, *flexible*,

Column Name	Description
y	*regular*, *maximal*. Missing values must be replaced with *flexible*.

```
WITH cte as (SELECT
  user_id,age,registration_date,email,lower(workout_frequency) as
workout_frequency
FROM users),
cte1 as(SELECT
  user_id,age,registration_date,email,
     CASE
           WHEN workout_frequency IN ('minimal', 'flexible',
'regular', 'maximal')
                THEN workout_frequency
           ELSE NULL
       END AS workout_frequency
      FROM cte
)

SELECT
user_id,
COALESCE(age,AVG(AGE)OVER(ORDER BY user_id)) as age,
COALESCE(registration_date,'2024-01-01')::Date as registration_date,
COALESCE(email,'Unknown') as email,
COALESCE(workout_frequency,'flexible') as workout_frequency
FROM cte1
```

Task 2

It seems like there are missing values in the `events` table for the column `game_id` for all events before the year 2021.

However, we know that before 2021 there were only games where the `game_type` is `running`. The `game_id` for these games can be found in the `games` table.

Write a query so that the `events` table has a `game_id` for all events including those before 2021.

```
SELECT event_id,
           COALESCE(public.events.game_id,4)as game_id,
           device_id, user_id, event_time
FROM public.events
```

Task 3

ActiVR's sales team plans to launch a promotion for upgrades to virtual reality devices.

They aim to target customers who have participated in events related to specific game types.

Write a SQL query to provide the `user_id` and `event_time` for users who have participated in events related to `biking` games.

```
SELECT user_id, event_time
FROM public.events e
JOIN public.games g ON e.game_id = g.game_id
WHERE g.game_type = 'biking'
```

Task 4

After running their promotion, the sales team at ActiVR wants to investigate the results.

To do so, they require insights into the number of users who participated in events for each `game_type`.

Write a SQL query that returns the count of unique users for each game type `game_type` and `game_id`. The user count should be shown in a column `user_count`.

```
SELECT COUNT(DISTINCT user_id)AS user_count , g.game_type, g.game_id
FROM public.events e
JOIN public.games g ON e.game_id = g.game_id
GROUP BY 2,3
```