University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 4

Date: November 29, 2024

Group Number: 41

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Terence Yin	47554431	n6p7i@ugrad.cs.ubc.ca	terence.yin76@gmail.com
Fegico Chen	14033468	b9c8r@ugrad.cs.ubc.ca	jakeyeozh@gmail.com
Jake Yeo	86759529	k3c3v@ugrad.cs.ubc.ca	fegicochen@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

CPSC 304 Milestone 4

1) The final project is a user matchmaking system that stores users, profiles and personalities. It includes features such as user registration, personality assessments, and matching algorithms to find other users based on compatibility scores derived from their personality traits and sexuality.

2) Previously, personality traits were distributed across multiple tables, such as PToIntro, PToObservant, PToProspecting, PToFeeling, and PToTurbulent. Relationships between traits were managed in separate mapping tables like ExtroToIntro and IntuitiveToObservant. In the final schema, all personality traits have been merged into a single Personality table, significantly simplifying the structure and reducing complexity. Other than the changes of personality traits into a single Personality table, the schema remained largely the same. The only additional changes involved adjustments to data types, such as converting CHAR to VARCHAR2 and DECIMAL to NUMBER for storing personality traits.

3) All SQL queries are in appService.js.

- INSERT - 530-716
- UPDATE - 503 - 505
- DELETE - 358-373
- Selection - 865-870
- Projection - 835-842
- Join - 453-473

4)

- Aggregation with GROUP BY

```
`SELECT Sexuality, COUNT(*) AS HomosexualCount
 FROM Profile
 WHERE Sexuality = :sexuality
 GROUP BY Sexuality`,
['Homosexual']

`SELECT Sexuality, COUNT(*) AS HeterosexualCount
 FROM Profile
 WHERE Sexuality = :sexuality
 GROUP BY Sexuality`,
['Hetero']
```

  Counts the amount of users by sexuality.

- Aggregation with HAVING - 996-999

```
SELECT upc.PostalCode, COUNT(*) as UserCount
        FROM UserPostalCode upc
        GROUP BY upc.PostalCode
        HAVING upc.PostalCode = :postalCode
```

Counts the amount of users in a postal code.

- Nested aggregation with GROUP BY

```
`SELECT u.Email, u.Name, upc.PostalCode
        FROM Users u
        JOIN UserPostalCode upc ON u.Email = upc.Email
        JOIN Users u ON u.Email = upc.Email
        WHERE upc.PostalCode IN (
          SELECT upc2.PostalCode
          FROM UserPostalCode upc2
          JOIN Users u2 ON upc2.Email = u2.Email
          GROUP BY upc2.PostalCode
          HAVING COUNT(u2.Email) >= :input
          )
          `,
        [input]
```

Finds Postal Codes where the number of users is greater than or equal to the chosen number.

- Division

```
SELECT DISTINCT pc.PostalCode, pc.City
  FROM PostalCodeCity pc
  WHERE NOT EXISTS (
    (SELECT up.Email
    FROM UserPostalCode up
    WHERE up.PostalCode = pc.PostalCode)
    MINUS
    (SELECT p.PersonalityID
    FROM Personality p
    WHERE p.Extrovertedness > 5)
  )`);
```

Find Postal Codes with All Extroverted Users.