University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: _4_

Date: _2024-11-29_

Group Number: _Group 106__

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Ankur Bhardwaj	83640458	y8e4o	ankurb75@gmail.com
Divy Patel	82174020	s2x3p	divy07ubc@gmail.com
Oliver Shen	32805475	a3e5k	oliverdsheny@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

1. A short description of the final project, and what it accomplished.

   This project focuses on modeling the various entities and relationships in Valorant such as players, matches, agents, maps, teams and region. The application could be used to track and manage data related to professional players like the matches they participate in and the statistics surrounding their performances.

2. A description of how your final schema differed from the schema you turned in. If the final schema differed, explain why.

   There is no significant difference or major changes between Milestone 3 and Milestone 4 other than correcting syntax errors in the valo.sql file. The only change we made is that we are using ON DELETE CASCADE only and not ON UPDATE CASCADE as Oracle does not support ON UPDATE CASCADE. valo.sql is the name of the set-up SQL script.

3. A list of all SQL queries used to satisfy the rubric items and where each query can be found in the code (file name and line number(s)).

   - **Queries: INSERT Operation:**

     Relations: Works on All Relations

     appService.js/Line 141: insertRow(tableName, rowData)

   - **Queries: DELETE Operation:**

     Relations: Works on All Relations

     appService.js/Line 192: deleteRow(tableName, keyColumn, keyValue)

   - **Queries: UPDATE Operation:**

     Relations: Works on All Relations

     appService.js/Line 160: updateRow(tableName, rowData)

   - **Queries: Selection:**

     Relations: Works on All Relations

appService.js/Line 97: selectRows(tableName, conditions)

- **Queries: Projection**

  Relations: Agent, using abilities

  appService.js/Line 247: projectAgent(abilities)

- **Queries: Join:**

  Relations: WeaponStats and Player; using RiotID

  SELECT w.riotID, w.gunName, w.headshotPercentage, p.rank FROM weaponStats w, Player p WHERE w.riotID = p.riotID;

  appService.js/Line 227: getJoinResults(threshold)

- **Queries: Aggregation with Group By:**

  Relations: ProPlayer, to display totalEarnings group by on team names

  SELECT teamName, SUM(earnings) AS totalEarnings FROM ProPlayer GROUP BY teamName;

  appService.js/Line 206: getGroupByResults()

- **Queries: Aggregation with Having:**

  Relations: PlayedInMatch. Display total kills by group by on agentName having < n deaths

  SELECT agentName, SUM(kills) AS totalKills, SUM(deaths) AS totalDeaths FROM PlayedInMatch GROUP BY agentName HAVING SUM(deaths) < n;

  appService.js/Line 272 getGroupByHavingResults(threshold)

- **Queries: Nested Aggregation with Group By:**

Relation: Round. Display avg number of times spike planted by getting count of spike planted per match

SELECT AVG(spikeCount) AS avgSpikePlanted FROM (SELECT matchId, COUNT(*) AS spikeCount FROM Round WHERE spikePlanted = 1 GROUP BY matchId) spikeCountsPerMatch;

appService.js/Line 334: getAvgSpikePlanted()

- **Queries: Division**

  Relation: Match/Player. Display all matches with a table of players.

  SELECT DISTINCT m.matchId, m.startTime, m.regionName, m.mapName FROM Match m WHERE NOT EXISTS (SELECT 1 FROM temp_riot_ids t WHERE NOT EXISTS (SELECT 1 FROM PlayedInMatch p WHERE p.matchId = m.matchId AND p.riotId = t.riotId))

  appService.js/Line 288: divideMatches(riotIds)

4. In the milestone 4 assignment on Canvas, submit the URL to your group's repository.

   URL: https://github.students.cs.ubc.ca/CPSC304-2024W-T1/project_a3e5k_s2x3p_y8e4o