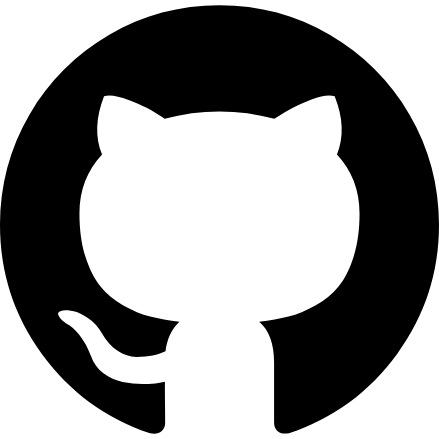


Faculty of Engineering and Applied Science

SOFE 3700U Data Management

**Phase III: Project Report**

**GitHubStats**

****

Due Date: Friday, Nov 29th, 2019

CRN 43512

Group 23:

Anna Safonov - 100601514

Priyadharshini Ramalingam - 100670614

Pranjal Saloni - 100653360

Umar Qureshi - 100591742

**Table of Contents**

[**Abstract**](#_tsia48udotx0) **3**

[**Introduction**](#_1k6i6c8w167e) **3**

[**Goals and Motivations**](#_txmc13yqtneo) **4**

[**Related Applications**](#_1oi3r9fm2ys3) **4**

[**Diagrams**](#_72np398frixl) **6**

[Relational Schema](#_ka05br6hgl5) 6

[ER Diagram](#_gekwtle1jomp) 7

[**Views**](#_q3voh1tglume) **9**

[View 1: Join of at least three tables](#_32afr8n6vsij) 9

[View 2: Nested queries with ANY or ALL operator and GROUP BY clause](#_65juuwtv0ile) 10

[View 3: A correlated nested query](#_blprcjvqzxag) 11

[View 4: FULL JOIN](#_q3vbrq7daxb3) 12

[View 5: Nested queries with any of the set operations UNION, EXCEPT, or INTERSECT](#_hncu622eh9jf) 13

[View 6: Show all users with more than 38 repositories](#_enghosgw7k41) 15

[View 7: View of all users in Toronto](#_4w1sbr83w9qs) 16

[View 8: Show all pull requests made in October](#_h8regeff7pgd) 16

[View 9: Show all issues for repository ID 494431](#_6i8w8nh04izb) 17

[View 10: Show all repositories written in PHP](#_ao0tqrl1lvxd) 17

[City View: Showcases a Pie Chart of which city the Users live](#_obu3qmb3czyi) 19

[Issue Status View: Shows how many issues are open and closed in a bar chart](#_2gvnedfuha5) 20

[Language View: Shows a Doughnut chart of which programs the repositories used](#_kmud1gtehpkh) 20

[**Design and Implementations**](#_irx4c0gamruy) **21**

[**Future Developments**](#_hvy99scodwu0) **23**

[**Conclusion**](#_m0rodjuqokzq) **23**

[**Group contribution**](#_qv99mxhbpsbf) **24**

## 

## **Abstract**

We developed a prototype of a database web application that leverages GitHub REST API to acquire data on the repositories, issues, pull requests, and commits of public GitHub users located in Ontario, Canada, and then uses it to analyze and display visual trends of activity of users for this geographical region.

## **Introduction**

At 40 million users across the world and 100 million repositories, GitHub is one of the most popular version control and project management platforms. Although it provides some general statistics on individual repository pages (new issues, closed issues, pull requests, languages used in the repository, contributors, etc.), currently there is no convenient way to look at real-time commit statistics across repositories of different users at a glance. GitHub does support a well-developed public API to extract information across many repositories for further analysis. For this project, our group will use GitHub API to extract useful statistics on commits for repositories of all public users located in Ontario, Canada as a convenient tool to peek at current GitHub activity in this geographic region.

## **Goals and Motivations**

The goal of this application is to focus on sorting and pulling commit statistics from public repositories of users located in Ontario, Canada from the past year. This data will be analyzed for the following metrics: which months and days of the week and hours of the day have the most commits, most commonly used programming languages, and the categories of software products these commits represent (web applications, system utilities, big data tools, etc.).

Our main motivation for this project is to develop a practical data-driven tool for quantifying trends in software development that will be implemented using tools and technologies covered in this course, such as a database management system, database design process, SQL query language, data flow in RESTful web services etc. Through this, we hope to demonstrate our understanding of the course material and our ability to apply it in a real small-scale solution.

## **Related Applications**

<https://github.com/marketplace/circleci>

CircleCi is an application that uses Github API for project teamwork. It speeds up the test and delivery cycle without running your own infrastructure by showing workflow status, related jobs with the Insights functionality, and performance trends.

<https://github.com/marketplace/zenhub>

ZenHub uses Github API and integrates natively with Githubs user interface. It has a Multi-Repo Task Board that lets the team visualize issues and group them in epics, track dependencies and collaborate on product backlogs. Zenhub can also release reports, use the history of reports to detect trends to improve processes, increase team efficiency and measure the value delivered to end-users.

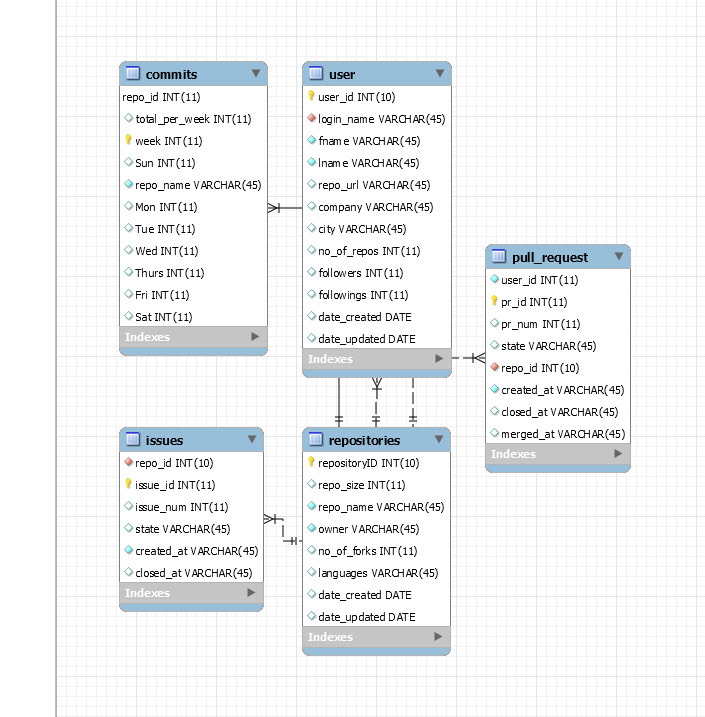
Our project differs from the above applications because it does not focus on the functionality of a single repository. Instead we deliver statistical insights on user activity for many repositories as a quantification tool of trends in development process for various types of software technologies.

## **Diagrams**

### **Relational Schema**

****

### **ER Diagram**



## **Views**

Sample views that can be rendered using the data pulled from GitHub for this project.

### **View 1: Join of at least three tables**

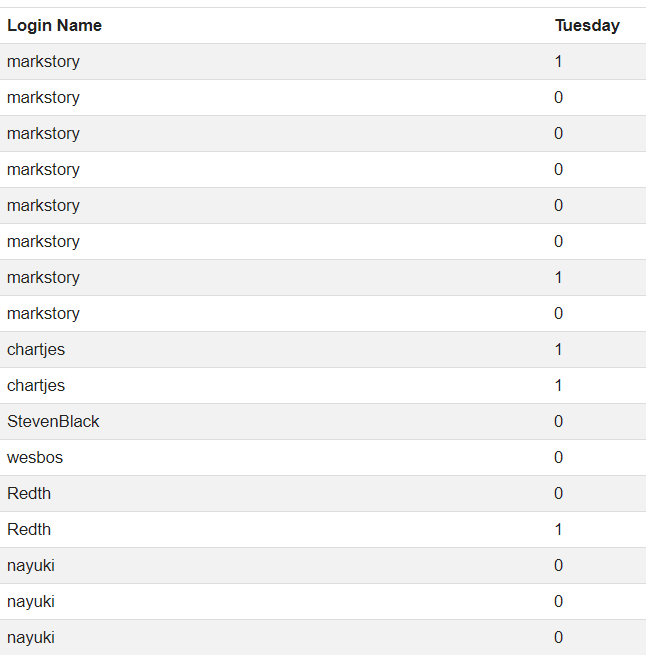
Displays the number of commits made on a Tuesday

SELECT u.login\_name, c.Tue

FROM ((repositories as r

INNER JOIN user AS u ON r.repo\_owner = u.login\_name)

INNER JOIN commits AS c ON r.repositoryID = c.repo\_id);



### **View 2: Nested queries with ANY or ALL operator and GROUP BY clause**

Select users, grouped by last name, that have more followings than all the users living in Toronto.

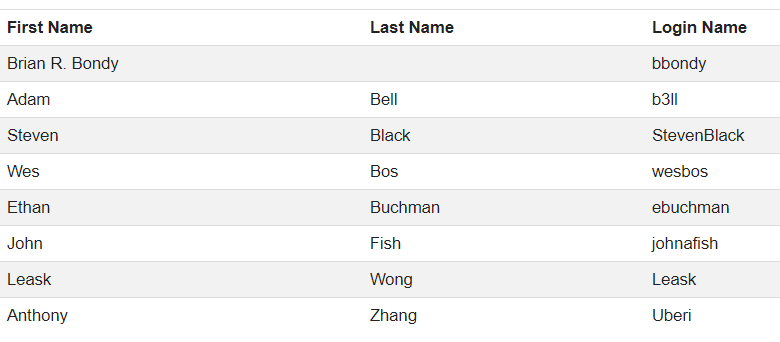
SELECT u.fname, u.lname, u.login\_name

FROM user as u

WHERE u.followings > ALL (SELECT followings

FROM user

WHERE city = 'Toronto')



### **View 3: A correlated nested query**

Showing users who have more followers then user with login name ncschonni

SELECT u.fname, u.lname

FROM user as u

WHERE u.followers >

( SELECT followers

FROM user

WHERE login\_name = "Pahimar");



### **View 4: FULL JOIN**

Select \*

FROM user

FULL OUTER JOIN repositories

ON user.login\_name = repositories.owner;

*Above does not work in MySQL due to Outer Joins not being supported. We did this query instead:*

SELECT \*

FROM user

LEFT JOIN repositories ON user.login\_name = repositories.repo\_owner

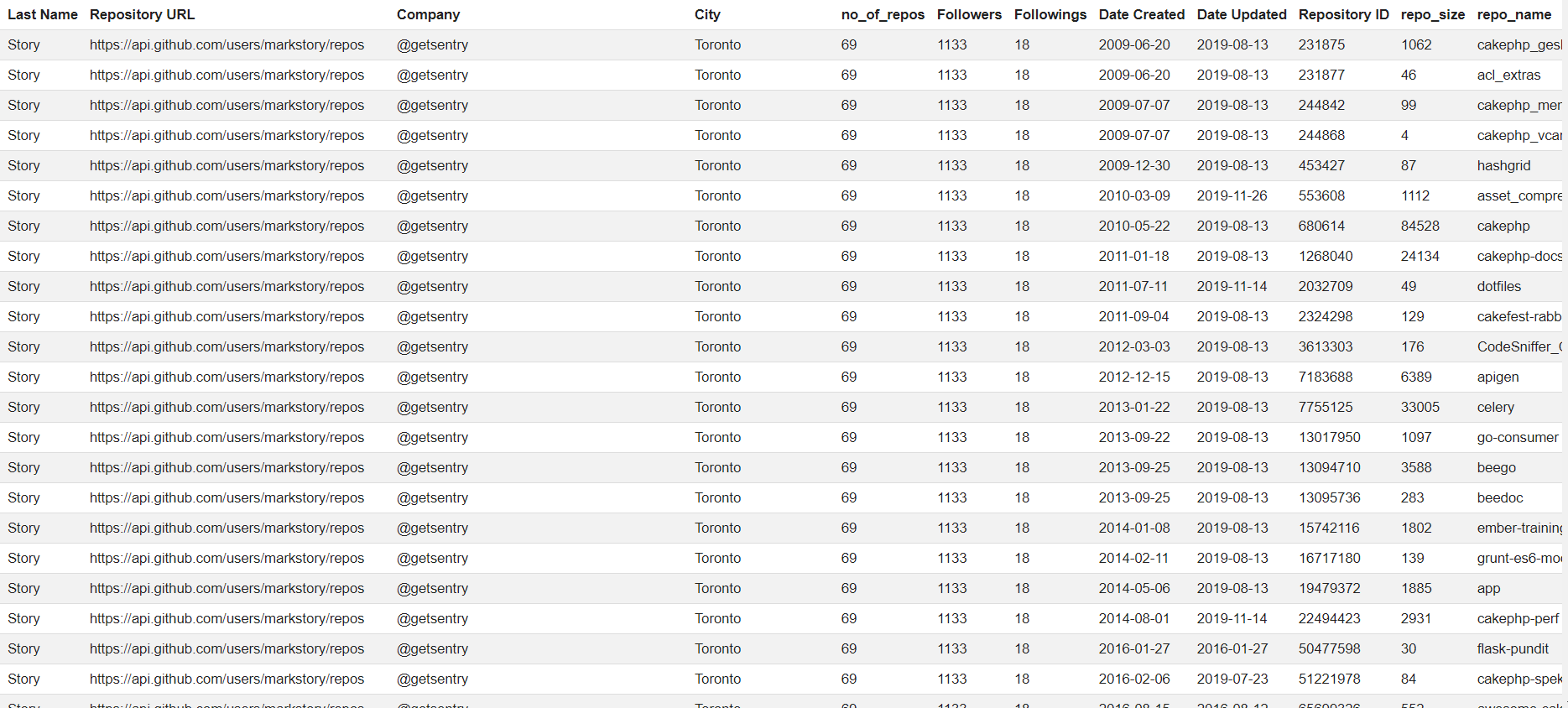
UNION

SELECT \*

FROM user

RIGHT JOIN repositories ON user.login\_name = repositories.repo\_owner;

*Above is just a snippet of the table as it is long, in this snippet it shows the middle of the table where they both have joined. (repo\_url is not it repositories table and repositoryID is not in user table)*

**

### **View 5: Nested queries with any of the set operations UNION, EXCEPT, or INTERSECT**

Shows repo ids of repos that have a smaller size of PHP and if it has an open issue

SELECT r.repositoryID

FROM repositories AS r

WHERE r.repo\_size < ALL (SELECT r.repo\_size

FROM repositories AS r

WHERE languages = 'PHP')

EXCEPT

SELECT i.repo\_id

From issues AS i

where i.state = "open";

*Above doesn’t work (as mySql doesn’t support EXCEPT. So we did this instead. Below shows repo ids of repos that have a smaller size then the PHP repos or shows repo ids that have closed issues.*

SELECT r.repositoryID

FROM repositories AS r

WHERE r.repo\_size < ALL (SELECT r.repo\_size

FROM repositories AS r

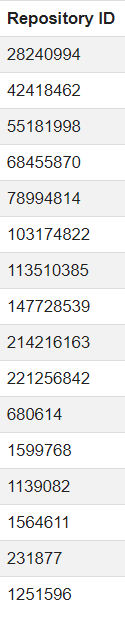
WHERE languages = 'PHP')

UNION

SELECT i.repo\_id

From issues AS i

where i.state\_state = "closed";



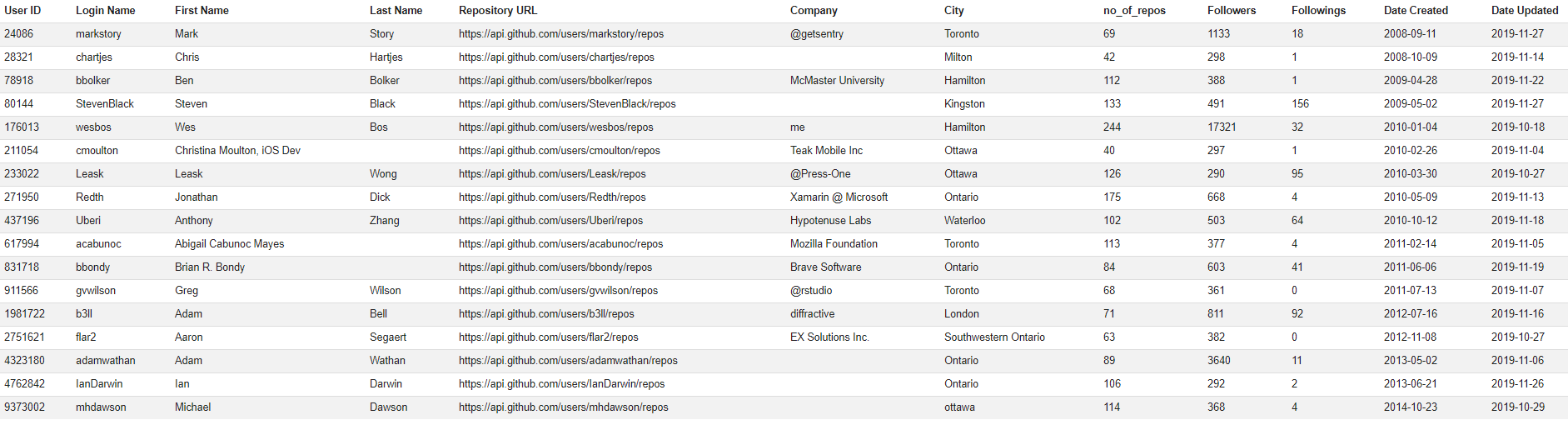
### **View 6: Show all users with more than 38 repositories**

SELECT \*

FROM user

WHERE no\_of\_repos > '38'

ORDER BY user\_id;

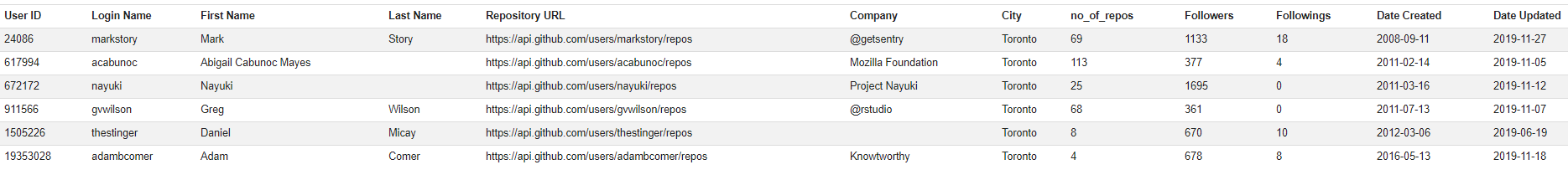


### **View 7: View of all users in Toronto**

SELECT \*

FROM user

WHERE city = 'toronto';



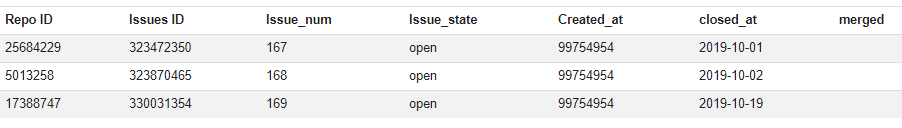
### **View 8: Show all pull requests made in October**

SELECT \*

FROM pull\_request

WHERE (created\_at >= DATE '2019-10-01') AND

(closed\_at <= DATE '2019-10-31');

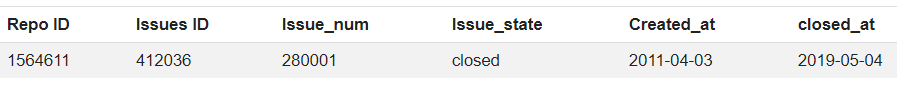


### **View 9: Show all issues for repository ID 494431**

SELECT \*

FROM github\_api.issues

WHERE issue\_id = '412036';

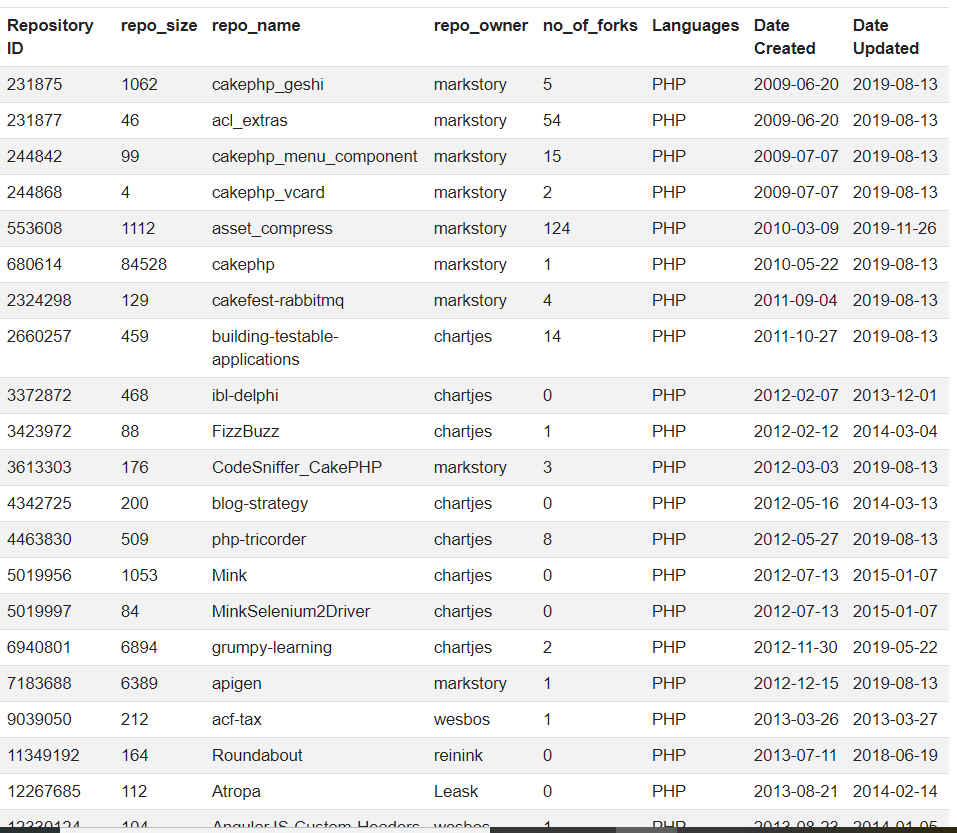


### **View 10: Show all repositories written in PHP**

SELECT \*

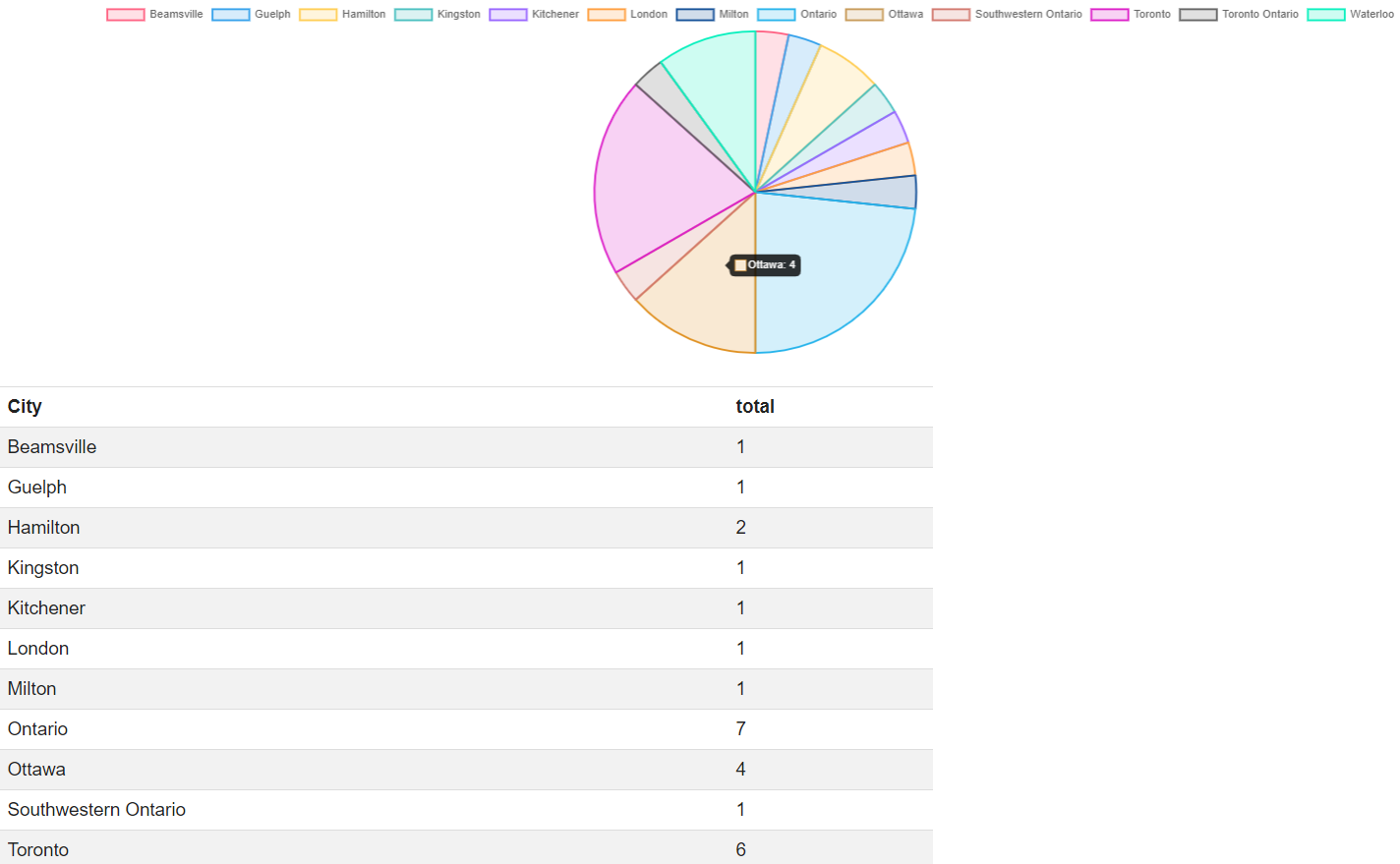
FROM repositories

WHERE languages = 'php';

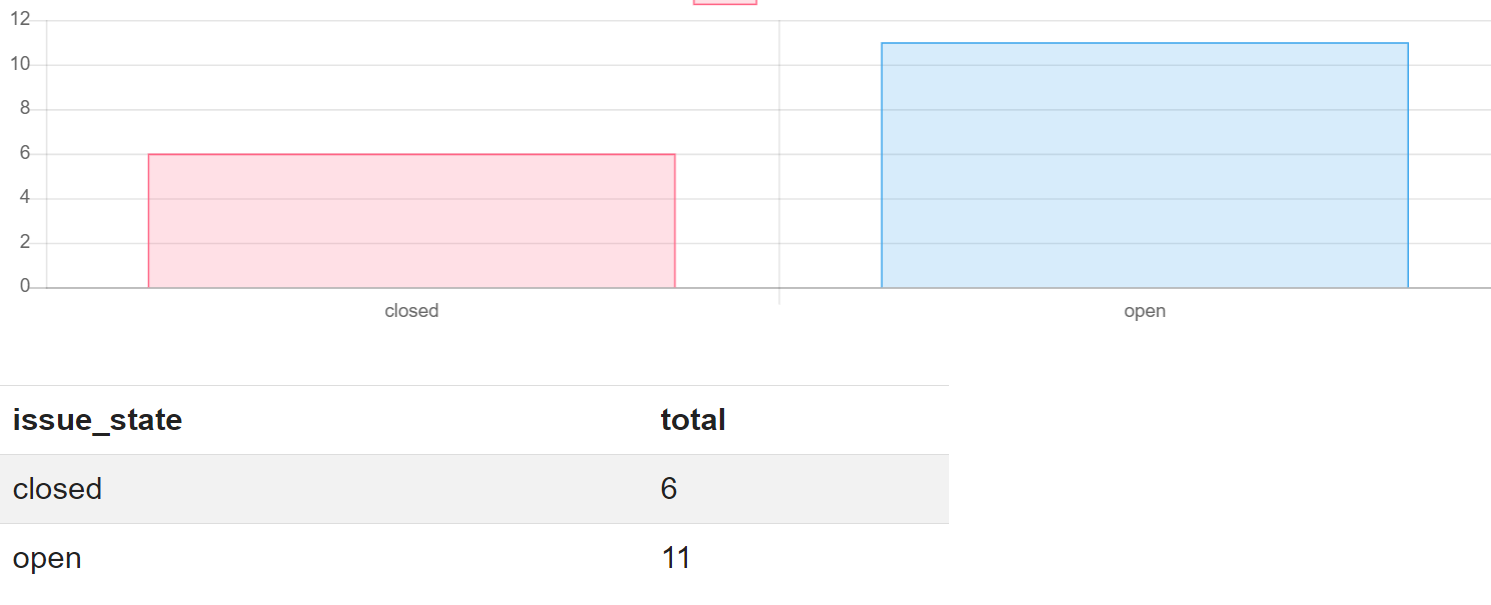


### **City View: Showcases a Pie Chart of which city the Users live**

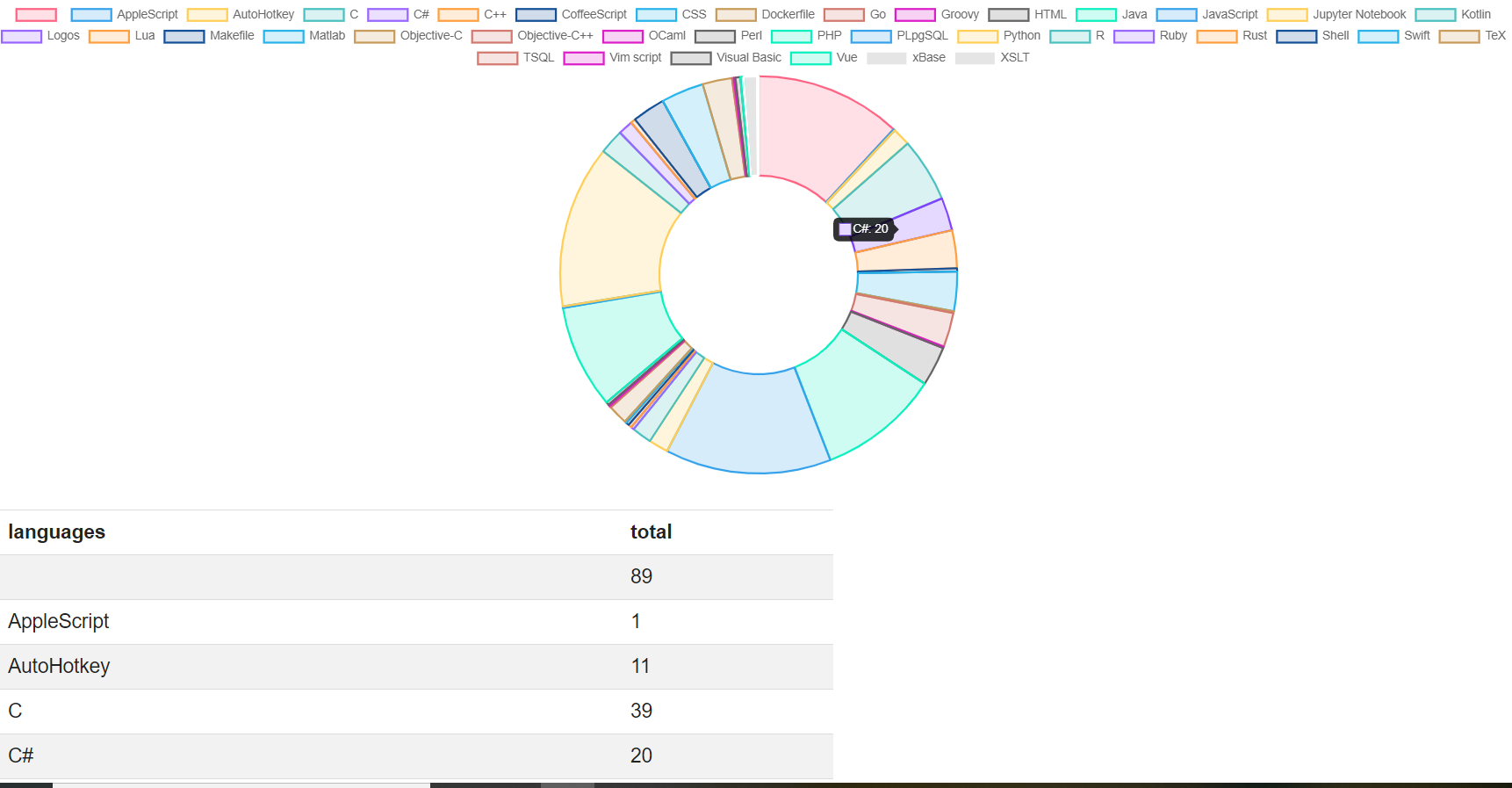
Counts how many times a city occurs in a column

****

### **Issue Status View: Shows how many issues are open and closed in a bar chart**

****

### **Language View: Shows a Doughnut chart of which programs the repositories used**

****

## **Design and Implementations**

Our group approached the project by breaking it down in the following stages:

* Developed the database design,
* Retrieved, parsed and sanitized data from Github using GitHub public REST API,
* Created and populated the database in MySQL Workbench,
* Created a presentation web layer and
* Queried the database for statistics to display, etc.

We used GitHub public REST API vs, MySQL and MySQL Workbench, PHP for REST API queries, and data parsing and sanitation, and Bootstrap HTML and CSS for the web application. We extracted GitHub data on commits as it is the most accurate metric of repository activity. Our project is limited to public repositories of users located in Ontario, Canada and their commit activity in the past year.

Our group has chosen the above-mentioned technologies and frameworks because we had team members with experience using these technologies and languages for previous projects, internships, etc. Also, because all of these technologies are commonly used for various business tech solutions and are common tools in software development currently in the industry.

The web application is a graphical tool to help users navigate statistics for commits, repositories, pull request, user profiles, and issues. The main page contains links to each of these tables and each of these pages contains tables with populated data displayed in a graph.

We wanted to use a live and robust API for this project and GitHub REST API proves to be an excellent choice. This API comes with well written documentation, support and a friendly online community. Github is a commonly used tool for repository and project management.

**Problems and Limitations**

Here we discuss some of the problems and limitations we encountered during the development of this project.

**REST API**: The first difficulty we encountered was specifying the API endpoint incorrectly, so the data returned was not in a proper JSON format.

**Rate Limits**: GitHub only returns the first 1000 requests on any API call. Even though our API calls to users in Ontario have over 14K users, less than 10% of that data is available. GitHub imposes these limitations to prevent scraping of information.

In addition to limiting the response to a query, another limitation is the rate limit for API calls. The maximum rate for an unauthenticated application is 60 hits/hour. Due to this constrained, sleep functions had to be introduced into our PHP script to ensure that we do not hit HTTP 403 error due to hitting the rate limit.

Time delays within the script lead to a long execution time required to populate database tables, but unfortunately this was the easiest approach for a prototype for this project.

**XAMPP:** Xampp was giving many connectivity issues. Had to go through multiple config files and edit them and change the port numbers. When everything was connecting phpMyAdmin would freeze on any action. After researching many users had the same issue with phpMyAdmin freezing. The resolution was to use an older version of Xampp and reconfigure everything again and to let my teammates know the issues and to use another service like Wampp instead.

**Hosting with AWS**: Our original plan was to host our application through the Amazon Web Services platform. However, correct configuration of this cloud service took longer than expected, so we were left with no time to test our PHP script to populate database and render the web application.

## **Future Developments**

GitHubStats is a flexible, adaptable and an easily-modifiable tool that can be used in a variety of settings, including as a team management tool, for data analytics and visualization, and to quantify team member contributions on large projects.

Further developments for this application include a GitHub statistics mobile app, a better frontend interface for the web application, and customizable functionalities for activity tracking and analytics.

## **Conclusion**

The main purpose of this project was to apply our theoretical database knowledge to a real world application and make a web application that makes it easier for users to access useful information and statistics on GitHub API. We successfully accomplished this by using the tools and techniques learned in this course. For this project, we learned to use MySQL workbench which made it very easy to create tables, ER diagram etc. We also successfully learned how to create relational schemas with associated SQL table commands. Furthermore, we populated our database with sample data where we applied different views that the user would find useful and displayed our project database - through reverse engineering - using an ER diagram. We also learned how APIs work, and how to implement one to return a query as a JSON.

## 

## **Group contribution**

We met every week to work on the project, split the work equally and worked on what each of us were assigned to do before the due date. Everyone had an equal share of work. The project was divided between 4 group members, each with 25% of work load.

*Umar Qureshi* - Umar worked on the schema and ER diagram for phase 2. He along with Anna also worked on views 1-3. For phase 3, he was responsible for connecting the front end website to the MySQL database with PHP locally. He also worked on parsing live html tables into Json for char.js to make visual representations of the data.

*Anna Safonov* - For phase 2, Anna worked on populating sample data and alongside with Umar, worked on creating views 3-5. For phase 3, Anna was responsible for populating database, php connections and hosting with AWS.

*Pranjal Saloni* - Worked on creating the 5 sample tables (repositories, commits, user, pull-request and issues) along with Priya using MySQL workbench. For phase 2, was responsible for working on views 6 -8. Also worked on creating the html web pages, css and js.

*Priyadharshini Ramalingam* - Alongside with Pranjal, worked on creating the 5 sample tables, worked on views 8 -10 for phase 2. Also, created the html web pages, css and js.

Everyone worked on the report and presentation together.