TẠP CHÍ KHOA HỌC TRƯỜNG ĐẠI HỌC HOA LƯ
ISSN 2615-9538
Website: http://hluv.edu.vn/vi/tckh

XÂY DỰNG HỆ THỐNG CƠ SỞ DỮ LIỆU CHO BÀI TOÁN BỎ PHIẾU TRÊN NỀN TẢNG THIẾT BỊ DI ĐỘNG BẰNG HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU MYSQL

Đào Sỹ Nhiên[1]*, Nguyễn Thị Thu Hà[1], Bùi Thị Tuyết[2], Phạm Xuân Nguyện[3], Hoàng Cao Minh[4]

Ngày nhận bài: 25/09/2024 Ngày chấp nhận đăng: 13/12/2024

Tóm tắt: *Hệ thống cơ sở dữ liệu là yếu tố tạo cho phần mềm hoạt động hiệu quả, bảo mật và an toàn. Bài viết thực hiện việc khảo sát các giải pháp để thiết kế cơ sở dữ liệu đối với bài toán bỏ phiếu trên nền tảng thiết bị di động, có so sánh các đặc điểm giữa các hệ quản trị cơ sở dữ liệu phổ biến như Microsoft SQL Server, Oracle Database, PostgreSQL và MySQL và triển khai xây dựng hệ thống cơ sở dữ liệu cho bài toán trên MySQL trong đó đưa ra những giải pháp về một số View (bảng ảo), Procedure (thủ tục), Index (chỉ mục) và Trigger trong MySQL nhằm hệ thống được xây dựng đảm bảo các yêu cầu về hiệu suất, bảo mật, tính toàn vẹn dữ liệu và giải quyết các thách thức trong việc tối ưu hóa truy vấn, đặc biệt trong môi trường bỏ phiếu trực tuyến.*

Từ khóa: *MySQL (Relational Database Management System - RDBMS), hệ thống cơ sở dữ liệu, App (Application - ứng dụng), View (bảng ảo), Procedure (thủ tục), Index (chỉ mục) và Trigger.*

BUILDING A DATABASE SYSTEM FOR MOBILE-BASED VOTING PROBLEM USING MYSQL DATABASE MANAGEMENT SYSTEM

Abstract: *The database system is a key factor enabling software to operate efficiently, securely, and safely. This article surveys solutions for designing a database for the voting problem on mobile platforms, comparing the characteristics of popular database management systems such as Microsoft SQL Server, Oracle Database, PostgreSQL, and MySQL. It then implements a database system for the voting problem using MySQL, providing solutions involving several Views (virtual tables), Procedures, Indexes, and Triggers in MySQL to ensure the system meets requirements for performance, security, data integrity, and addresses challenges in query optimization, especially in an online voting environment.*

Keyword: *MySQL (Relational Database Management System - RDBMS), database system, App (Application), View, Procedure, Index, Trigger.*

1. Đặt vấn đề

Ngày nay, các ứng dụng di động đã nổi lên như một thành phần thiết yếu trong nỗ lực chiến lược của bất kỳ tổ chức nào nhằm tiếp cận và tương tác với đối tượng mục tiêu. Với điện thoại thông minh và máy tính bảng đã trở thành phương tiện thống trị để liên lạc, giải trí và năng

[1] Khoa Ngoại ngữ - Công nghệ thông tin, Trường Đại học Hoa Lư, *Email: dsnhien@hluv.edu.vn
[2] Phòng Đào tạo - Quản lý khoa học, Trường Đại học Hoa Lư
[3] Trung tâm Ngoại ngữ - Tin học, Trường Đại học Hoa Lư
[4] Phòng Hành chính - Quản trị, Trường Đại học Hoa Lư

suất. Do đó, việc phát triển các ứng dụng dành cho thiết bị di động phục vụ cho các nhu cầu và sở thích khác nhau là một nỗ lực đầy thách thức và bổ ích. Thiết kế ứng dụng trên thiết bị di động (Mobile App) giúp tối ưu hóa quy trình vận hành và quản lý, nâng cao hiệu suất công việc, từ đó tăng trải nghiệm người dùng là cần thiết.

Ứng dung (App) bỏ phiếu trên Mobile mang lại nhiều tiện lợi, dễ dàng cho người sử dụng, khắc phục khó khăn về địa lý, tiết kiệm thời gian, chi phí. Hơn nữa nó còn giúp các nhà quản lý có thể thu thập dữ liệu ý kiến người dùng một cách nhanh chóng, hiệu quả chính xác, dễ dàng xem xét, theo dõi và phân tích kết quả.

Khi nghiên cứu về các hệ quản trị cơ sở dữ liệu phổ biến như Microsoft SQL Server, Oracle Database, PostgreSQL và MySQL, tác giả nhìn nhận việc ứng dụng MySQL để xây dựng và triển khai một hệ thống cơ sở dữ liệu cho bài toán bỏ phiếu trên nền tảng thiết bị di động là đảm bảo, vì đó là hệ quản trị cơ sở dữ liệu mạnh mẽ, an toàn và linh hoạt. MySQL được thiết kế để xử lý các truy vấn và thao tác dữ liệu nhanh chóng và ổn định, phù hợp với các hệ thống cần xử lý lượng lớn dữ liệu và phù hợp với bài toán đã nêu. Vấn đề cần thực hiện chính là phát triển một hệ thống cơ sở dữ liệu tối ưu cho các bài toán bỏ phiếu trên thiết bị di động, đáp ứng được các yêu cầu về tốc độ, độ chính xác và bảo mật.

2. Nội dung

2.1. Cơ sở lý thuyết để lựa chọn MySQL để triển khai

- Thực hiện khảo sát các giải pháp hệ thống bỏ phiếu điện tử hiện nay, nhận thấy rất đa dạng về quy mô, tính năng và công nghệ sử dụng. Tuy nhiên, thường có các thành phần chính sau:

+ Giao diện người dùng: Cho phép người bỏ phiếu đăng nhập, xác thực danh tính và thực hiện việc bỏ phiếu.

+ Cơ sở dữ liệu: Lưu trữ thông tin về người bỏ phiếu, ứng viên, phiếu bầu và kết quả.

+ Máy chủ ứng dụng: Xử lý các yêu cầu từ giao diện người dùng, tương tác với cơ sở dữ liệu và đảm bảo tính toàn vẹn của dữ liệu.

+ Hệ thống xác thực: Đảm bảo tính bảo mật và xác thực danh tính của cử tri.

- So sánh các hệ quản trị cơ sở dữ liệu [1], [2], [6], [7]:

Bảng so sánh các hệ quản trị cơ sở dữ liệu

Đặc điểm	Microsoft SQL Server	Oracle Database	PostgreSQL	MySQL
Nguồn mở	Không	Không	Có	Có
Chi phí	Thương mại	Thương mại	Miễn phí, có bản thương mại	Miễn phí, có bản thương mại
Hiệu suất	Cao	Rất cao	Cao	Cao
Tính năng	Rất nhiều	Rất nhiều	Nhiều	Nhiều
Cộng đồng	Lớn	Lớn	Lớn	Rất lớn
Dễ sử dụng	Tốt	Tốt	Tốt	Rất tốt
Phổ biến	Rất phổ biến	Rất phổ biến	Phổ biến	Rất phổ biến

- Lý do lựa chọn MySQL cho hệ thống bỏ phiếu điện tử [6].

Dựa trên bảng so sánh trên tác giả lựa chọn MySQL để xây dựng và triển khai một hệ thống cơ sở dữ liệu cho bài toán chính vì:

+ Chi phí: MySQL là một lựa chọn miễn phí, phù hợp với các dự án có ngân sách hạn chế.

+ Nguồn mở: Tính mở nguồn cho phép tùy chỉnh và mở rộng hệ thống một cách dễ dàng.

+ Cộng đồng lớn: Cộng đồng người dùng MySQL rất lớn, giúp việc tìm kiếm tài liệu, hỗ trợ và giải quyết vấn đề trở nên dễ dàng hơn.

+ Hiệu suất cao và phổ biến: MySQL đáp ứng được yêu cầu về hiệu suất xử lý lượng lớn dữ liệu trong các cuộc bỏ phiếu và có đặc điểm là rất phổ biến.

+ Dễ sử dụng: MySQL có cú pháp SQL đơn giản và dễ học, giúp giảm thời gian phát triển hệ thống.

+ Linh hoạt: MySQL có thể chạy trên nhiều hệ điều hành khác nhau và tích hợp với nhiều ngôn ngữ lập trình.

2.2. Thiết kế hệ thống

2.2.1 Mô hình cơ sở dữ liệu

* Phân tích chức năng của bài toán bỏ phiếu trên nền tảng thiết bị di động, nhóm tác giả đã mô tả về người dùng, Use Case và mối quan hệ của bài toán như sau:

- Actors: Người dùng (Voter); Quản trị viên (Admin); Hệ thống xác thực (Authentication System).

- Use Cases: Đăng ký tài khoản; Đăng nhập; Xem danh sách cuộc bỏ phiếu; Tham gia bỏ phiếu; Xem kết quả bỏ phiếu; Quản lý tài khoản; Quản lý người dùng; Tạo cuộc bỏ phiếu; Quản lý cuộc bỏ phiếu; Xem và xuất kết quả.

- Mối quan hệ:

+ Người dùng ↔ Đăng ký tài khoản, Đăng nhập, Xem danh sách cuộc bỏ phiếu, Tham gia bỏ phiếu, Xem kết quả bỏ phiếu, Quản lý tài khoản

+ Quản trị viên ↔ Quản lý người dùng, Tạo cuộc bỏ phiếu, Quản lý cuộc bỏ phiếu, Xem và xuất kết quả.

+ Hệ thống xác thực ↔ Đăng ký tài khoản, Đăng nhập

* Từ kết quả phân tích chức năng của bài toán ở trên, ta xác định 07 Bảng gồm: Users (Người dùng); Polls (Cuộc bỏ phiếu); Options (Các lựa chọn); Votes (Bỏ phiếu); Candidate (Ứng cử viên); User_Permissions (Quyền của người dùng), Polls_candidate.

* Chi tiết về mô hình ERD (Entity-Relationship Diagram) với bảng dữ liệu chính và mối quan hệ giữa chúng như sau:

- Users có quan hệ 1-N với Polls, Votes, và User_Permissions.
- Polls có quan hệ 1-N với polls_candidate và Options.
- Votes có quan hệ N-1 với Users và Options.
- Polls_candidate có quan hệ N-1 với Polls và Candidate.
- Options có quan hệ 1-N với Votes và N-1 với Polls_candidate.

2.2.2. Thiết kế các Bảng

Là cách tổ chức bảng và tạo các chỉ mục để tối ưu hóa truy vấn. Thực hiện trên hệ thống cơ sở dữ liệu MySQL.

* Tạo các Bảng [3], [5] thuộc cơ sở dữ liệu tên là "voting_hluv".

(1) Bảng `Users` (Người dùng), tạo bảng bằng câu lệnh như sau:

```
CREATE TABLE Users (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL UNIQUE,
    password_hash VARCHAR(255) NOT NULL,
    fullname_user VARCHAR(100),
    email VARCHAR(100) UNIQUE,
    role ENUM('admin', 'user') NOT NULL,
```

```
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP);
```

(2) Bảng `User_Permissions` (Quyền của người dùng)

```
CREATE TABLE User_Permissions (
    user_id INT,
    permission VARCHAR(100),
    granted_by INT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    PRIMARY KEY (user_id, permission),
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE
CASCADE,
    FOREIGN KEY (granted_by) REFERENCES Users(user_id) ON DELETE SET NULL);
```

(3) Bảng `Polls` (Cuộc bỏ phiếu)

```
CREATE TABLE Polls (
    poll_id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255) NOT NULL,
    description TEXT,
    user_id INT,
    start_date DATE,
    end_date DATE,
    status ENUM('chưa bắt đầu', 'đang diễn ra', 'đã kết thúc') NOT NULL,
    poll_type ENUM('yes_no', 'choose_x_from_y') NOT NULL,
    max_selections INT DEFAULT 1,
    is_public BOOLEAN DEFAULT 1,
    visibility ENUM('public', 'private', 'specific_group') DEFAULT 'public',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE);
```

(4) Bảng 'Candidate' (Ứng cử viên)

```
    CREATE TABLE Candidate (
    candidate_id INT AUTO_INCREMENT PRIMARY KEY,
    fullname_user VARCHAR(100) NOT NULL,
    birthday DATE,
    picture VARCHAR(255),
    description TEXT,
```

work_unit VARCHAR(100),
updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP);

(5) Bảng 'Polls_candidate'

```
CREATE TABLE polls_candidate (
    poll_candidate_id INT AUTO_INCREMENT PRIMARY KEY,
    candidate_id INT,
    poll_id INT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (candidate_id) REFERENCES Candidate(candidate_id) ON DELETE CASCADE,
    FOREIGN KEY (poll_id) REFERENCES Polls(poll_id) ON DELETE CASCADE);
```

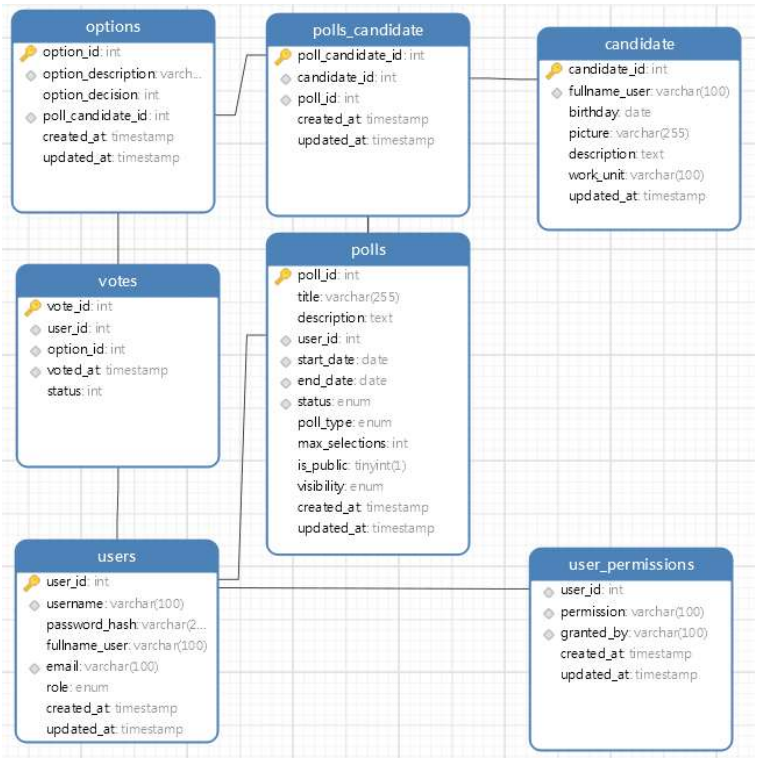
(6) Bảng `Options` (Các lựa chọn)

```
CREATE TABLE Options (
    option_id INT AUTO_INCREMENT PRIMARY KEY,
    option_description VARCHAR(255),
    option_decision INT,
    poll_candidate_id INT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (poll_candidate_id) REFERENCES polls_candidate(poll_candidate_id) ON DELETE CASCADE) ;
```

(7) Bảng `Votes` (Bỏ phiếu)

```
CREATE TABLE Votes (
    vote_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    option_id INT,
    voted_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    status INT DEFAULT 1,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE,
    FOREIGN KEY (option_id) REFERENCES Options(option_id) ON DELETE CASCADE);
```

* Mô hình ERD (Entity-Relationship Diagram) với bảng dữ liệu ở trên và mối quan hệ giữa chúng theo hình sau:



Hình 1: Mô hình cơ sở dữ liệu "vote_hluv" dưới dạng ERD

2.2.3. Xây dựng một số View (bảng ảo), Procedure (thủ tục), Index (chỉ mục) và Trigger trong MySQL.

Để xây dựng hệ thống bỏ phiếu trên ứng dụng di động hiệu quả, an toàn, tối ưu hóa cơ sở dữ liệu và truy xuất nhanh, việc sử dụng View (bảng ảo), Procedure (thủ tục), Index (chỉ mục) và Trigger có thể giúp tối ưu cơ sở dữ liệu, tự động hóa một số quy trình như kiểm tra dữ liệu đầu vào, ghi log thay đổi hoặc cập nhật thông tin tự động…nhằm đảm bảo hệ thống hoạt động hiệu quả, an toàn thông tin, đặc biệt đối với bài toán bỏ phiếu trên thiết bị di động đó là điều rất cần thiết.

(1) Tạo một *View* [3], [4], [5] để hiển thị kết quả bầu cử:

* Mục tiêu của View:

- Hiển thị thông tin về các cuộc bỏ phiếu (từ bảng `Polls`).
- Hiển thị danh sách các ứng viên (từ bảng `Candidate`).
- Hiển thị số lượng phiếu bầu mà mỗi ứng viên nhận được (dựa trên bảng `Votes` và `Options`).

* Dưới đây là một truy vấn MySQL để tạo View "election_results" nhằm liệt kê kết quả bầu cử, bao gồm số phiếu bầu của từng ứng viên trong từng cuộc bỏ phiếu:

```
CREATE VIEW election_results AS
SELECT
        p.title AS poll_title,
        c.fullname_user AS candidate_name,
        COUNT(v.vote_id) AS vote_count,
        p.start_date AS poll_start_date,
        p.end_date AS poll_end_date
FROM
        Votes v
```

```
        JOIN Options o ON v.option_id = o.option_id
        JOIN polls_candidate pc ON o.poll_candidate_id = pc.poll_candidate_id
        JOIN Candidate c ON pc.candidate_id = c.candidate_id
        JOIN Polls p ON pc.poll_id = p.poll_id
GROUP BY
        p.poll_id,
        c.candidate_id
ORDER BY
        p.poll_id,
        vote_count DESC
```

* Xem dữ liệu từ View: SELECT * FROM election_results;

poll_title	candidate_name	vote_count	poll_start_date	poll_end_date
Đại hội chi bộ	Nguyễn Thị Hồng Tuyên	18	2024-09-25	2024-09-25
Đại hội chi bộ	Nguyễn Thị Hoàng Huế	18	2024-09-25	2024-09-25
Đại hội chi bộ	Đồng Thị Thu	18	2024-09-25	2024-09-25
Đại hội chi bộ	Đào Sỹ Nhiên	18	2024-09-25	2024-09-25

Hình 2: Kết quả của khi thực hiện View "election_results"

(2) Xây dựng thủ tục lưu trữ (stored procedure)

Với bài toán trên để đạt được mục tiêu sắp xếp kết quả bầu cử, ta cần tổng hợp số phiếu (votes) cho từng ứng viên (candidate) trong một cuộc bỏ phiếu cụ thể, sau đó sắp xếp chúng từ cao xuống thấp. Ta xây dựng các thủ tục [3], [4], [5] (stored procedure) để thực việc này. Cụ thể, ta cần một truy vấn để lấy tổng số phiếu của mỗi ứng viên trong một cuộc bỏ phiếu và sắp xếp chúng từ cao xuống thấp. Câu truy vấn này sẽ kết nối bảng `Votes`, `Options`, `polls_candidate`, và `Candidate` để lấy thông tin.

- Dưới đây là một truy vấn MySQL để tạo thủ tục lưu trữ "GetElectionResults":

```
DELIMITER //
CREATE PROCEDURE GetElectionResults(IN poll_id INT)
BEGIN
        SELECT
                c.fullname_user AS candidate_name,
                COUNT(v.vote_id) AS vote_count
        FROM
                Votes v
        JOIN
                Options o ON v.option_id = o.option_id
        JOIN
                polls_candidate pc ON o.poll_candidate_id = pc.poll_candidate_id
        JOIN
                Candidate c ON pc.candidate_id = c.candidate_id
        WHERE
                pc.poll_id = poll_id
        GROUP BY
                c.candidate_id
        ORDER BY
```

vote_count DESC;
END //
DELIMITER;
- Gọi thủ tục này và lấy kết quả bầu cử của một cuộc bỏ phiếu có `poll_id` cụ thể: CALL GetElectionResults(300);

candidate_name	vote_count
Nguyễn Thị Hoàng Huế	18
Đào Sỹ Nhiên	18
Đồng Thị Thu	18
Nguyễn Thị Hồng Tuyên	18

Hình 3: Kết quả khi thực hiện thủ tục lưu trữ "GetElectionResults"

(3) Tạo Index

Để tối ưu hóa cơ sở dữ liệu hệ thống bỏ phiếu trên thiết bị di động, việc đánh "Index" [3], [4], [5] trong MySQL là cần thiết nhằm tăng tốc các truy vấn tìm kiếm và lọc dữ liệu. Dưới đây là cách đánh "Index" cho các bảng theo cấu trúc cơ sở dữ liệu đã được thiết kế.

Với bảng "User", các truy vấn thường xuyên sẽ bao gồm việc tra cứu theo "username", "email", và "user_id" (khóa chính) khi đăng nhập, tìm kiếm hoặc quản lý người dùng.

CREATE INDEX idx_username ON Users(username);
CREATE INDEX idx_email ON Users(email);…
Tương tự, với các bảng khác, chúng ta cũng thực hiện đánh "Index" để tối ưu hóa cơ sở dữ liệu.
Khóa chính "user_id" đã tự động được đánh index khi được định nghĩa là "PRIMARY KEY".

(4) Xây dựng các TRIGGER

* Trigger [3], [4], [5]: Kiểm tra trạng thái cuộc bỏ phiếu trước khi bỏ phiếu. Khi người dùng bỏ phiếu, ta cần đảm bảo cuộc bỏ phiếu vẫn đang diễn ra (status = 'đang diễn ra') trong bảng `Votes`. Cụ thể thực hiện như sau:

```
CREATE TRIGGER check_poll_status_before_vote
BEFORE INSERT ON Votes
FOR EACH ROW
BEGIN
    DECLARE poll_status VARCHAR(255);
    SELECT status INTO poll_status FROM Polls WHERE poll_id = NEW.poll_id;
    IF poll_status != 'đang diễn ra' THEN
    SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Cuộc bỏ phiếu này đã kết thúc hoặc chưa bắt đầu.';
    END IF;
END;
```

* Trigger [3], [4], [5]: Cập nhật số lượng phiếu khi có người bỏ phiếu. Khi người dùng thực hiện bỏ phiếu thành công, hệ thống sẽ tự động cập nhật số lượng phiếu cho cuộc bỏ phiếu liên quan thông qua Trigger.

```
CREATE TRIGGER update_vote_count
AFTER INSERT ON Votes
FOR EACH ROW
BEGIN
    UPDATE Polls
```

```
        SET vote_count = vote_count + 1
        WHERE poll_id = NEW.poll_id;
    END;
```

* Trigger [3], [4], [5]: Kiểm tra số lượng lựa chọn khi bỏ phiếu. Đối với các cuộc bỏ phiếu thuộc loại "choose_x_from_y", hệ thống sẽ sử dụng Trigger để kiểm tra xem số lượng lựa chọn của người dùng có vượt quá giới hạn tối đa đã được thiết lập hay không, đảm bảo tính hợp lệ trước khi lưu phiếu bầu.

```
    CREATE TRIGGER check_max_selections_before_vote
    BEFORE INSERT ON Votes
    FOR EACH ROW
    BEGIN
        DECLARE max_selections INT;
        DECLARE current_votes INT;
        -- Lấy số lượng lựa chọn tối đa cho cuộc bỏ phiếu
        SELECT max_selections INTO max_selections FROM Polls WHERE poll_id =
    NEW.poll_id;
        -- Đếm số lần người dùng đã bỏ phiếu cho cuộc bỏ phiếu này
        SELECT COUNT(*) INTO current_votes FROM Votes WHERE user_id =
        NEW.user_id AND poll_id = NEW.poll_id;
        IF current_votes >= max_selections THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Bạn đã bỏ tối đa số
phiếu cho cuộc bỏ phiếu này.';
        END IF;
    END;
```

2.2.4. Đánh giá giữa việc áp dụng View, Procedure, Index và Trigger trong MySQL và các hệ quản trị Microsoft SQL Server, Oracle Database, PostgreSQL

Khi áp dụng View, Procedure, Index và Trigger đối với hệ quản trị Microsoft SQL Server, Oracle Database, PostgreSQL đối với bài toán trên, tác giả nhìn nhận về những ưu nhược điểm của các hệ quản trị cơ sở dữ liệu này so với áp dụng hệ quản trị MYSQL như sau:

- Hệ quản trị cơ sở dữ liệu MYSQL cơ ưu điểm là cung cấp khả năng sử dụng ORDER BY trong VIEW và hỗ trợ tùy chọn ALGORITHM (giải thuật) để kiểm soát cách xử lý VIEW, là một lợi thế so với các hệ quản trị khác. Procedure, INDEX và TRIGGER có cú pháp đơn giản, giúp người dùng nhanh chóng tạo và sử dụng cho các tác vụ, truy vấn với yêu cầu của bài toán bỏ phiếu trên thiết bị di động.

- Hệ quản trị cơ sở dữ liệu khác (Microsoft SQL Server, Oracle Database, PostgreSQL) có ưu điểm là khả năng tối ưu hóa PROCEDURE, INDEX và TRIGGER cao, với các tính năng bảo mật và kiểm soát lỗi tốt hơn, phù hợp cho các hệ thống lớn. Điều này cũng là hạn chế của hệ quản trị cơ sở dữ liệu MYSQL, nhưng với yêu cầu của bài toán bỏ phiếu trên thiết bị di động mà nhóm tác giả xây dựng đảm bảo tốt yêu cầu về hiệu suất, bảo mật, tính toàn vẹn dữ liệu và giải quyết được các thách thức trong việc tối ưu hóa truy vấn, đặc biệt trong môi trường bỏ phiếu trực tuyến, đồng thời giúp ích cho công tác đào tạo sinh viên năm thứ 3 ngành công nghệ thông tin.

3. Kết luận

Qua quá trình thiết kế và triển khai, kết quả cho thấy MySQL là lựa chọn phù hợp cho các ứng dụng đòi hỏi khả năng xử lý dữ liệu linh hoạt, đồng thời giải quyết tốt các thách thức trong môi trường bỏ phiếu trực tuyến trên thiết bị di động.

Với việc sử dụng MySQL trong hệ thống bỏ phiếu trên thiết bị di động, cụ thể là việc áp dụng các kỹ thuật tối ưu hóa như sử dụng các view, stored procedure, index và trigger. Điều này giúp cải thiện đáng kể hiệu suất truy vấn và khả năng mở rộng của hệ thống, đáp ứng các yêu cầu về hiệu suất, tính bảo mật và toàn vẹn dữ liệu. Hệ thống có thể xử lý hiệu quả với nhiều người dùng đồng thời, đảm bảo sự an toàn và tính bảo mật.

TÀI LIỆU THAM KHẢO

[1] Dusan Petkovic (2020). Microsoft SQL Server 2019 A Beginner's Guide Edition 7. *McGraw-Hill Education eBooks*.

[2] Iggy Fernandez (2008). Beginning Oracle Database 11g Administration. Berkeley, CA

[3] Navicat (2024). Cách thức sử dụng navicat-premium để xây dựng cơ sở dữ liệu trong MySQL. Navicat Premium | Connects to multiple databases on a single GUI

[4] Freetuts (2024). Tài liệu MySQL nâng cao. https://freetuts.net/hoc-mysql/mysql-nang-cao.

[5] Sourcee (2024). Cách áp dụng XAMPP để thực hiện tạo lập cơ sở dữ liệu với công cụ phpMyAdmin. https://sourceforge.net/projects/xampp/

[6] Ray Harris - Joel Murach (2019). Lập Trình Cơ Bản PHP và MySQL. Bách khoa Hà Nội.

[7] Regina Obe và Leo Hsu (2012). PostgreSQL: Up and Running. O'Reilly.