دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

گزارشکار فاز اول پروژه اصول طراحی پایگاه داده

دکتر پوربهمن

1403/03/14

بردیا صباغ کرمانی

کد این پروژه، با زبان جاوا و پایگاه داده postgresql انجام شد. ابتدا برای اتصال پایگاه داده به کد جاوا باید از درایور jdbc استفاده کرد. یک کلاس به نام Query ساخته شد که در آن چهار متد insert, update, delete و read پیاده سازی شد.

سپس باید connection را ساخت.

```
public static Connection getConnection() {
    String url = "jdbc:postgresql://localhost:5432/postgres";
    String username = "myuser";
    String password = "mypassword";
    try {
        Class.forName( className: "org.postgresql.Driver");

        return DriverManager.getConnection(url, username, password);
    } catch (Exception ex) {
        System.out.println(ex.toString());
    }
    return null;
}
```

این متد را بصورت static تعریف کردم که هر جای کلاس Query بتوانیم از آن استفاده کنیم.

متد insert بدین صورت است که یک نام جدول و یک Hash map از مقدار هایی که قرار است insert شوند را دریافت میکند. دقت کنید که HashMap را بصورت Object به Object نوشته ایم که در آینده بتواند هر نوع ورودی، از جمله media را هم بتواند اضافه کند. این Hashmap نام attribute را به value مربوطه map میکند.

در ابتدا بجای مقدار ها در رشته ی query، علامت سوال میگذاریم (برای جلوگیری از sql injection) و سپس علامت سوال هارا با متد setObject جایگذاری میکنیم.

کوئری را مرحله به مرحله با StringBuilder میسازیم.

```
public static void insert(String table, HashMap<Object, Object> queries) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    try {
        connection = getConnection();

        StringBuilder pedaret = new StringBuilder();
        pedaret.append("INSERT INTO ").append(table).append(" (");
        for (Object o : queries.keySet()) {
            pedaret.append(o).append(", ");
        }
        pedaret.delete(pedaret.length() - 2, pedaret.length() - 1);
        pedaret.append(") VALUES (");
        pedaret.append("?, ".repeat(queries.size()));
        pedaret.delete(pedaret.length() - 2, pedaret.length() - 1);
        pedaret.append(")");

        assert connection != null;
        preparedStatement = connection.prepareStatement(pedaret.toString());
        int k = 1;
        for (Object o : queries.values()) {
            preparedStatement.setObject(k, o);
            k++;
        }
        System.out.println(pedaret);
        System.out.println(preparedStatement);
        int rowsAffected = preparedStatement.executeUpdate();
        if (rowsAffected > 0) {
            System.out.println("Data inserted successfully.");
```

```java
        } else {
            System.out.println("Failed to insert data.");
        }
    } catch (SQLException e) {
        System.err.println("SQL exception occurred: " + e.getMessage());
    } catch (Exception e) {
        System.err.println("An error occurred: " + e.getMessage());
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }
            if (connection != null) {
                connection.close();
            }
        } catch (SQLException e) {
            System.err.println("Error closing resources: " + e.getMessage());
        }
    }
```

یک نمونه از insert که بصورت دستی در Main نوشته شده:

```java
HashMap<Object, Object> queries = new HashMap<>();
String tableName = "users";
queries.put("username", "bardia112");
queries.put("firstname", "Bardia");
queries.put("lastname", "Sabbagh");
queries.put("phonenumber", "09130412486");
queries.put("birthDate", Date.valueOf( s: "2004-06-06"));
Query.insert(tableName,queries);
```

و خروجی آن:

```
INSERT INTO users (firstname, phonenumber, birthDate, username, lastname ) VALUES (?, ?, ?, ?, ? )
INSERT INTO users (firstname, phonenumber, birthDate, username, lastname ) VALUES (('Bardia'), ('09130412486'), ('2004-06-06 +04:30'), ('bardia112'), ('Sabbagh') )
Data inserted successfully.
```

دو رشته اول صرفا برای بررسی صحت Query نوشته شده است.

زمانی که insert با موفقیت انجام شد، پیغام Data inserted successfully چاپ میشود.

متد update نیز دقیقا مانند insert کار میکند صرفا با این تفاوت که دو Hashmap ورودی میگیرد زیرا ابتدا باید مقادیری که باید بروز شوند را بگیرد و یک HashMap برای شروطی که سطر مورد نظر را پیدا میکند بگیرد.

```
public static void update(String table, HashMap<Object, Object> updates, HashMap<Object, Object> conditions) {
    Connection connection = null;
    PreparedStatement preparedStatement = null;
    try {
        connection = getConnection();

        StringBuilder pedaret = new StringBuilder();
        pedaret.append("UPDATE ").append(table).append(" SET ");
        for (Object key : updates.keySet()) {
            pedaret.append(key).append(" = ?, ");
        }
        pedaret.delete(pedaret.length() - 2, pedaret.length());
        if (conditions != null && !conditions.isEmpty()) {
            pedaret.append(" WHERE ");
            for (Object key : conditions.keySet()) {
                pedaret.append(key).append(" = ? AND ");
            }
            pedaret.delete(pedaret.length() - 5, pedaret.length());
        }

        assert connection != null;
        preparedStatement = connection.prepareStatement(pedaret.toString());
        int k = 1;
        for (Object value : updates.values()) {
            preparedStatement.setObject(k, value);
            k++;
        }
        if (conditions != null) {
            for (Object value : conditions.values()) {
                preparedStatement.setObject(k, value);
```

```
            }
        }
        System.out.println(pedaret);
        System.out.println(preparedStatement);
        int rowsAffected = preparedStatement.executeUpdate();
        if (rowsAffected > 0) {
            System.out.println("Data updated successfully.");
        } else {
            System.out.println("Failed to update data.");
        }
    } catch (SQLException e) {
        System.err.println("SQL exception occurred: " + e.getMessage());
    } catch (Exception e) {
        System.err.println("An error occurred: " + e.getMessage());
    } finally {
        try {
            if (preparedStatement != null) {
                preparedStatement.close();
            }
            if (connection != null) {
                connection.close();
            }
        } catch (SQLException e) {
            System.err.println("Error closing resources: " + e.getMessage());
        }
    }
}
```

نحوه ورودی گرفتن و خروجی دقیقا مانند متد insert است.

متد های Delete و Read هم دقیقا مانند دو متد بالا کار میکنند، صرفا خروجی متد Read در کنسول چاپ میشود.

اضافه کردن Tom Kane

```
HashMap<Object, Object> queries = new HashMap<>();
queries.put("username", "Tom_kane");
queries.put("firstname", "Tom");
queries.put("lastname", "Kane");
queries.put("phonenumber", "+44796268462");
queries.put("birthdate", Date.valueOf( s: "2003-03-24"));
Query.insert( table: "users", queries);
```

خروجی:

```
INSERT INTO users (firstname, birthdate, phonenumber, username, lastname ) VALUES (?, ?, ?, ?, ? )
INSERT INTO users (firstname, birthdate, phonenumber, username, lastname ) VALUES (('Tom'), ('2003-03-24 +04:30'), ('+44796268462'), ('Tom_kane'), ('Kane') )
Data inserted successfully.
```

اضافه کردن کاربر Tom به یک گروه با آیدی 1 و نوشتن یک پیام

```
HashMap<Object, Object> queries2 = new HashMap<>();
queries2.put("userid", 27);
queries2.put("chatid", 1);
Query.insert( table: "chatmembers", queries2);
HashMap<Object, Object> queries3 = new HashMap<>();
queries3.put("chatid", 1);
queries3.put("senderid", 27);
queries3.put("context", "Tom sent a message");
queries3.put("ts", Timestamp.valueOf( s: "2024-06-02 12:38:56"));
Query.insert( table: "messages", queries3);
```

اضافه شدن این دو به جدول های chatmembers و message

```
INSERT INTO chatmembers (chatid, userid ) VALUES (?, ? )
INSERT INTO chatmembers (chatid, userid ) VALUES (('1'::int4), ('27'::int4) )
Data inserted successfully.
INSERT INTO messages (senderid, chatid, context, ts ) VALUES (?, ?, ?, ? )
INSERT INTO messages (senderid, chatid, context, ts ) VALUES (('27'::int4), ('1'::int4), ('Tom sent a message'), ('2024-06-02 12:38:56+03:30') )
Data inserted successfully.
```

و در نهایت تغییر تاریخ تولد این کاربر و نشان دادن پیامش با متد read

```
HashMap<Object, Object> conditions = new HashMap<>();
conditions.put("username", "Tom_kane");
HashMap<Object, Object> updates = new HashMap<>();
updates.put("phonenumber", "+447342780080");
Query.update( table: "users", updates, conditions);
HashMap<Object, Object> conditions2 = new HashMap<>();
conditions2.put("messageid", 9);
Query.read( table: "messages", conditions2);
```

خروجی این:

```
UPDATE users SET phonenumber = ? WHERE username = ?
UPDATE users SET phonenumber = ('+447342780080') WHERE username = ('Tom_kane')
Data updated successfully.
SELECT * FROM messages WHERE messageid = ?
SELECT * FROM messages WHERE messageid = ('10'::int4)
messageid: 10, chatid: 1, senderid: 28, context: Tom sent a message, ts: 2024-06-02 12:38:56
```

در فایل **create.sql** علاوه بر ساختن جدول ها (همان فایل فاز قبل)، تغییرات جدید هم اعمال شده است

```
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('mcrocumbe0', 'Mariann', 'Crocumbe', '+86 543 182 8
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('chaggard1', 'Corrine', 'Haggard', '+86 972 653 315
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('csaintepaul2', 'Cecilius', 'Sainte Paul', '+62 718
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('lworsam3', 'Leontine', 'Worsam', '+223 914 737 633
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('cnightingale4', 'Calli', 'Nightingale', '+51 992 3
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('tpessler5', 'Thaxter', 'Pessler', '+380 305 417 06
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('hfairman6', 'Hilarius', 'Fairman', '+880 557 283 0
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('qmadelin7', 'Quintus', 'Madelin', '+52 226 901 120
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('ofranzoli8', 'Orren', 'Franzoli', '+389 914 472 77
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('atixier9', 'Alaster', 'Tixier', '+234 159 984 5158
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('tvondrysa', 'Trevor', 'Vondrys', '+63 396 949 6921
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('bbettb', 'Buddy', 'Bett', '+62 909 833 9338', '200
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('yhawardc', 'Yard', 'Haward', '+47 612 722 2069', '
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('gberrickd', 'Godiva', 'Berrick', '+86 714 765 6567
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('vguinnesse', 'Vida', 'Guinness', '+351 952 745 155
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('ycalderonf', 'Yankee', 'Calderon', '+81 875 163 89
Run | New Tab
insert into USERS (username, firstName, lastName, phoneNumber, birthDate) values ('ehartegang', 'Elwood', 'Hartegan', '+30 375 900 76
Run | New Tab
```

به عنوان مثال اضافه کردن مقداری کاربر **fake**

و اضافه کردن آن ها به گروه ها و نوشتن پیام برای آن ها و ساختن لیست مخاطبین برای آن ها

```sql
insert INTO chat (chatname, chattype) VALUES ('chate jadid', 'gp');
insert INTO chat (chatname, chattype) VALUES ('my groupchat', 'gp');
insert INTO chat (chatname, chattype) VALUES ('university', 'gp');
insert INTO chat (chatname, chattype) VALUES (NULL, 'pv');
insert into chatmembers (userid, chatid) values (1,4);
insert into chatmembers (userid, chatid) values (3,4);

insert into chatmembers (userid, chatid) values (1,1);
insert into chatmembers (userid, chatid) values (3,1);
insert into chatmembers (userid, chatid) values (4,1);

insert into chatmembers (userid, chatid) values (4,2);
insert into chatmembers (userid, chatid) values (5,2);
insert into chatmembers (userid, chatid) values (6,2);

insert into contact (userid, contactid) values (1,3);
insert into contact (userid, contactid) values (1,4);
insert into contact (userid, contactid) values (1,5);
insert into contact (userid, contactid) values (1,6);

insert into contact (userid, contactid) values (8,9);
insert into contact (userid, contactid) values (8,10);
```

```sql
insert into chatmembers (userid, chatid) values (4,2);
insert into chatmembers (userid, chatid) values (5,2);
insert into chatmembers (userid, chatid) values (6,2);

insert into contact (userid, contactid) values (1,3);
insert into contact (userid, contactid) values (1,4);
insert into contact (userid, contactid) values (1,5);
insert into contact (userid, contactid) values (1,6);

insert into contact (userid, contactid) values (8,9);
insert into contact (userid, contactid) values (8,10);
insert into contact (userid, contactid) values (8,11);
insert into contact (userid, contactid) values (8,7);
insert into messages (chatid, senderid, context, ts) values (1,16,'hello this is a message from user 16 to chat 30 as another message
insert into messages (chatid, senderid, context, ts) values (1,1,'hello this is a message from user 1 to chat 1', '2024-06-02 12:34:5
insert into messages (chatid, senderid, context, ts) values (1,3,'hello this is a message from user 3 to chat 1', '2024-06-02 12:35:5
insert into messages (chatid, senderid, context, ts) values (2,4,'hello this is a message from user 4 to chat 1', '2024-06-02 12:36:5
insert into messages (chatid, senderid, context, ts) values (2,4,'hello this is a message from user 4 to chat 2', '2024-06-02 12:37:5
insert into messages (chatid, senderid, context, ts) values (2,5,'hello this is a message from user 5 to chat 2', '2024-06-02 12:38:5
insert into messages (chatid, senderid, context, ts) values (1,4,'hello this is a message from user 4 to chat 1 as another message',
```

و در نهایت کوئری های خواسته شده

کوئری اول برای پاک کردن کاربر هایی که در هیچ گروهی نیستند و کوئری دوم برای نشان دادن گروه های کاربران مختلف

```sql
--Query that deletes users who are not in a group.
▷ Run | New Tab | Copy
DELETE FROM USERS
WHERE id NOT IN (
    SELECT DISTINCT userID
    FROM CHATMEMBERS
    WHERE chatID IN (
        SELECT chatID
        FROM CHAT
        WHERE chatType = 'gp'
    )
);
```

```sql
--Query that checks What groups does each user (with complete information) belong to
▷ Run | New Tab | JSON | Copy
SELECT
    USERS.id AS userID,
    USERS.username,
    USERS.firstName,
    USERS.lastName,
    USERS.phoneNumber,
    CHAT.chatID,
    CHAT.chatName
FROM
    USERS
JOIN
    CHATMEMBERS ON USERS.id = CHATMEMBERS.userID
JOIN
    CHAT ON CHATMEMBERS.chatID = CHAT.chatID
WHERE
    CHAT.chatType = 'gp';
```

کوئری برای نشان دادن کاربران یک گروه (با اطلاعات کامل) به ترتیب تعداد پیام های ارسال شده در آن گروه

```
--Query that returns Users of a group (with complete information) in order of the number of messages sent in that group.
▷ Run | New Tab | JSON | Copy
SELECT
    USERS.id AS userID,
    USERS.username,
    USERS.firstName,
    USERS.lastName,
    USERS.phoneNumber,
    CHAT.chatID,
    CHAT.chatName,
    COUNT(MESSAGES.messageID) AS messageCount
FROM
    USERS
JOIN
    CHATMEMBERS ON USERS.id = CHATMEMBERS.userID
JOIN
    CHAT ON CHATMEMBERS.chatID = CHAT.chatID
LEFT JOIN
    MESSAGES ON USERS.id = MESSAGES.senderID AND CHAT.chatID = MESSAGES.chatID
WHERE
    CHAT.chatID = 1 --Arbitrary chatID
GROUP BY
    USERS.id, USERS.username, USERS.firstName, USERS.lastName, USERS.phoneNumber, CHAT.chatID, CHAT.chatName
ORDER BY
    messageCount DESC;
```

کوئری برای نشان دادن هر گروه (با اطلاعات کامل) چه تعداد کاربر دارد که با یکدیگر چت خصوصی دارند

```
--This query returns Each group (with complete information) has how many users who have private chats with each other
▷ Run | New Tab | JSON | Copy
SELECT
    u1.id AS user1_id,
    u1.username AS user1_username,
    u1.firstName AS user1_firstName,
    u1.lastName AS user1_lastName,
    u1.phoneNumber AS user1_phoneNumber,
    u1.dateJoined AS user1_dateJoined,
    u2.id AS user2_id,
    u2.username AS user2_username,
    u2.firstName AS user2_firstName,
    u2.lastName AS user2_lastName,
    u2.phoneNumber AS user2_phoneNumber,
    u2.dateJoined AS user2_dateJoined
FROM
    USERS u1
JOIN
    CONTACT c ON u1.id = c.userid
JOIN
    USERS u2 ON c.contactID = u2.id
JOIN
    CHATMEMBERS cm1 ON u1.id = cm1.userID
JOIN
    CHATMEMBERS cm2 ON u2.id = cm2.userID AND cm1.chatID = cm2.chatID
JOIN
    CHAT ch ON cm1.chatID = ch.chatID
WHERE
    ch.chatType = 'pv';
```

و دو کوئری آخر خواسته شده:

```sql
--This query returns Users who have entered our messengers in one day and have sent more than one message in at least 2 groups
▷ Run | New Tab | JSON | Copy
SELECT u.*
FROM USERS u
JOIN MESSAGES m ON u.id = m.senderID
JOIN CHAT c ON m.chatID = c.chatID
WHERE u.dateJoined = '2024-06-03' --Arbitrary date
AND c.chatType = 'gp'
GROUP BY u.id, u.username
HAVING COUNT(DISTINCT m.chatID) >= 2
AND COUNT(m.messageID) > 1;

--This query returns Users (with complete information) who share one or more groups with a specific user
▷ Run | New Tab | JSON | Copy
SELECT DISTINCT u.*
FROM USERS u
JOIN CHATMEMBERS cm1 ON u.id = cm1.userID
JOIN CHAT c ON cm1.chatID = c.chatID
JOIN CHATMEMBERS cm2 ON c.chatID = cm2.chatID
WHERE cm2.userID = 1 --Arbitrary id
AND u.id != 1 --Arbitrary id
AND c.chatType = 'gp';
```