Unable to use Spring Data JPA to extract users that have a byte[] of a profile Image

Asked 1 year, 9 months ago Modified 1 year, 9 months ago Viewed 2k times

0

I have this AppUser entity:

```
@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
public class AppUser implements UserDetails {
@Id
@SequenceGenerator(
        name = "appUser_sequence",
        sequenceName = "appUser_sequence",
        allocationSize = 1
)
@GeneratedValue(
        strategy = GenerationType.SEQUENCE,
        generator = "appUser_sequence"
)
private Integer id;
private String firstname;
private String lastname;
private String email;
private String password;

@Lob
private byte[] profileImage;
private Role role;
// rest of the code
```

When i try to use this method to extract the user in the AppUserService:

```
@Override
public AppUser loadUserByUsername(String email) throws UsernameNotFoundException
{
    return repository.searchByEmail(email)
            .orElseThrow(()-> new UsernameNotFoundException("Email not found"));
}
```

I get this error: `org.springframework.orm.jpa.JpaSystemException: Unable to extract JDBC value for position 6`

The error is refering to profileImage, it said that JPA is unable to extract the byte array from the user, (the user is successfully saved in the database, i just can't extract it from my postgres database)

this is the service that receive a 64 string rappresenting the image and encode it in a byte array:

```
public AuthenticationResponse register(RegistrationRequest registerRequest) {
    byte[] profileImageBytes = null;

    if (registerRequest.getProfileImageData() != null)
        profileImageBytes =
Base64.getDecoder().decode(registerRequest.getProfileImageData());



AppUser appUserToRegister = AppUser.builder()
            .firstname(registerRequest.getFirstname())
            .lastname(registerRequest.getLastname())
            .email(registerRequest.getEmail())
            .password(passwordEncoder.encode(registerRequest.getPassword()))
            .profileImage(profileImageBytes)
            .role(Role.USER)
            .build();

    appUserService.saveUser(appUserToRegister);

    String token = jwtService.generateToken(appUserToRegister);
    return new AuthenticationResponse(token);
}
```

java postgresql spring-boot rest spring-data-jpa

Share Edit Follow

asked Apr 11, 2023 at 12:05
leo__
51 9

Which Hibernate version? What database in use? What Hibernate dialect? – Mar-Z Apr 11, 2023 at 15:11

What is the type of the column profileImage in the database? – Mar-Z Apr 11, 2023 at 15:35

I'm using a local postgres database, Hibernate version is 6.1.7.Final – leo__ Apr 11, 2023 at 16:38

that is the log at the table creation: create table app_user (id integer not null, email varchar(255), firstname varchar(255), lastname varchar(255), password varchar(255), profile_image oid, role smallint, primary key (id)) – leo__ Apr 11, 2023 at 16:40

OK. It means the image is stored in the database as oid type. and not as bytea. In this case please remove the annotation @Lob – Mar-Z Apr 11, 2023 at 19:27

1 Answer

Sorted by: Highest score (default)

4

You are using Postgres. In this case there are two options to store a byte array in the database.

One is OID type which is a composite type where the content is stored in a separate table and the original column contains only an integer id pointing to it. It can take up to 4 TB of data (older versions up to 2 GB).

The other one is the BYTEA type which is exactly what you wanted to use. It can take up to 1 GB of data and is more then enough for a profile picture.

You are using Hibernate 6. It will map the byte array type of the entity field to the BYTEA type of Postgres database without issue. Don't use the @Lob annotation as it would cause Hibernate to use the OID type which need to be read as a stream.

Solution

Remove the @Lob annotation from the entity.

Disclaimer: the solution is for Postgres database only. Other JDBC drivers can handle it different.

Share Edit Follow

answered Apr 12, 2023 at 8:09

Mar-Z
4,778 2 8 21