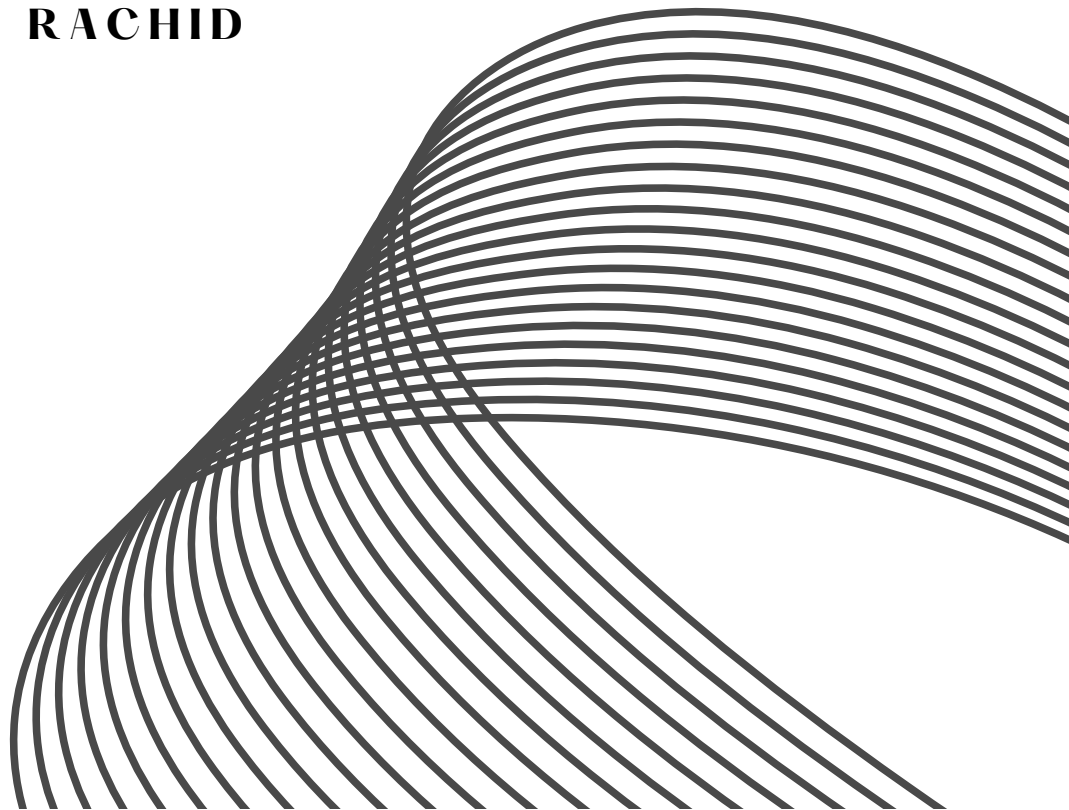


02/02/2023

Projet oracle

Préparé pour
Pr.Meryem El Mouhtadi

PRÉPARÉ PAR
SAKASSA RACHID



Introduction

Notre projet consiste à créer une plateforme web sécurisée avec une connexion simple, comprenant un bouton de connexion et deux zones de texte pour saisir le nom d'utilisateur et le mot de passe. La plateforme sera connectée à une base de données Oracle.

Nous avons choisi de se concentrer sur une base de données de type étudiant, contenant des informations sur les étudiants et leurs notes. Nous disposons de deux utilisateurs dans notre base de données Oracle : un administrateur avec des droits d'accès, de modification et de gestion complets, et un utilisateur normal avec un accès limité à la base de données et une restriction sur la visualisation de certaines tables (que nous pouvons préciser).

En utilisant Node.js, nous allons tester les accès à notre plateforme en connectant simultanément les deux utilisateurs pour vérifier que la connexion respecte les droits d'accès appropriés en fonction du type d'utilisateur. L'utilisateur administrateur devra pouvoir visualiser toutes les tables de la base de données, tandis que le deuxième utilisateur n'aura accès qu'à certaines tables.

sommaire

- Creation des tables
- Creation des utilisateurs admin et normal
- Concession des privilèges
- test sous oracles
- test sous la plateforme

Creation des tables sous oracle

Dans ce projet on va travaillé sur une base de données crée avec l'utilisateur systeme sys.

Création de la table "student" avec 3 colonnes: "student_id", "name" et "email". La colonne "student_id" est définie comme clé primaire et ne peut pas être nulle.

```
CREATE TABLE student (  
  student_id number(10) PRIMARY KEY,  
  name varchar2(50) NOT NULL,  
  email varchar2(50) NOT NULL  
);
```

Création de la table "notes" avec 4 colonnes: "note_id", "student_id", "subject" et "grade". La colonne "note_id" est définie comme clé primaire et la colonne "student_id" est définie comme clé étrangère en référence à la colonne "student_id" de la table "student"

```
CREATE TABLE notes (  
  note_id number(10) PRIMARY KEY,  
  student_id number(10) NOT NULL,  
  subject varchar2(50) NOT NULL,  
  grade number(3,2) NOT NULL,  
  FOREIGN KEY (student_id) REFERENCES student(student_id)  
);
```

Creation des utilisateurs admin et normal

Après la connection à sys as sysdba , on crée deux utilisateur sous sql developer

Deux utilisateurs: "ad" et "u1" avec leurs mots de passe respectifs. Les deux utilisateurs sont affectés au tablespace "users".

```
create user ad identified by ad default tablespace users;  
create user u1 identified by u1 default tablespace users;
```

On va montrer avec ces deux utilisateur qu'on a crée un exemple de concession de privilège sous oracle

si on a utilisé sql plus on va chaque fois se connecter à l'utilisateur pour tester les concession des privilèges, pour cela on a travaillé sur sql developer.

Concession des privilèges et test sous oracle

Concession de toutes les privilèges à l'utilisateur "ad".

Concession des privilèges "create session" et "select" sur la table "student" à l'utilisateur "u1".

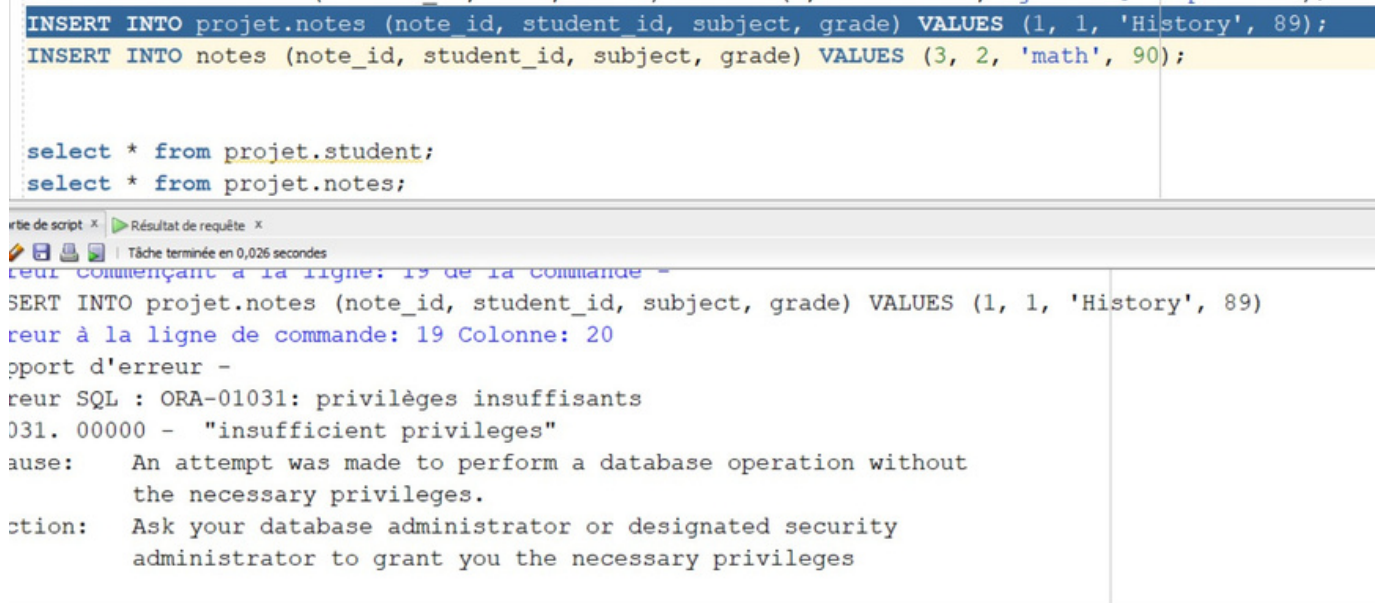
```
create user ad identified by ad default tablespace users;
create user u1 identified by u1 default tablespace users;

grant all privileges to ad;
grant create session to u1;
grant select on student to u1;
```

On va tester avec notre utilisateur normal u1 pour afficher la table notes ou bien insérer dans la table student

```
INSERT INTO projet.notes (note_id, student_id, subject, grade) VALUES (1, 1, 'History', 89);
INSERT INTO notes (note_id, student_id, subject, grade) VALUES (3, 2, 'math', 90);

select * from projet.student;
select * from projet.notes;
```



Erreur de script x Résultat de requête x

Tâche terminée en 0,026 secondes

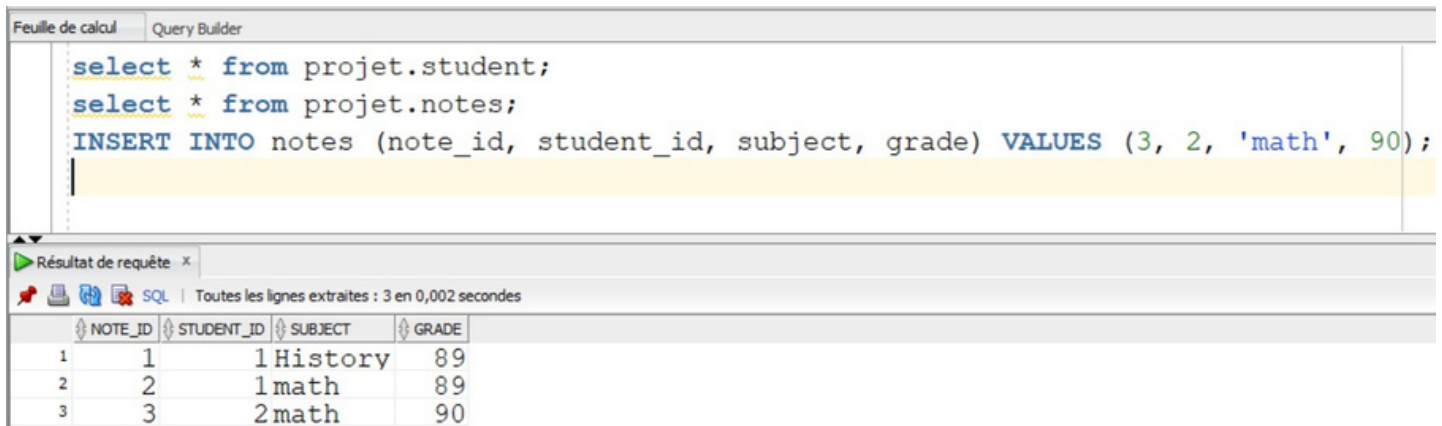
Erreur commençant à la ligne: 19 de la commande -

```
INSERT INTO projet.notes (note_id, student_id, subject, grade) VALUES (1, 1, 'History', 89)
Erreur à la ligne de commande: 19 Colonne: 20
Rapport d'erreur -
Erreur SQL : ORA-01031: privilèges insuffisants
031. 00000 - "insufficient privileges"
Cause:      An attempt was made to perform a database operation without
            the necessary privileges.
Action:     Ask your database administrator or designated security
            administrator to grant you the necessary privileges
```

on obtient une erreur car on n'a pas donné l'accès à l'utilisateur u1 pour voir la table notes ou d'insérer à la table student, ou notes.

Concession des privilèges et test sous oracle

On peut afficher et insérer et supprimer avec l'utilisateur admin qui a tous les privileges



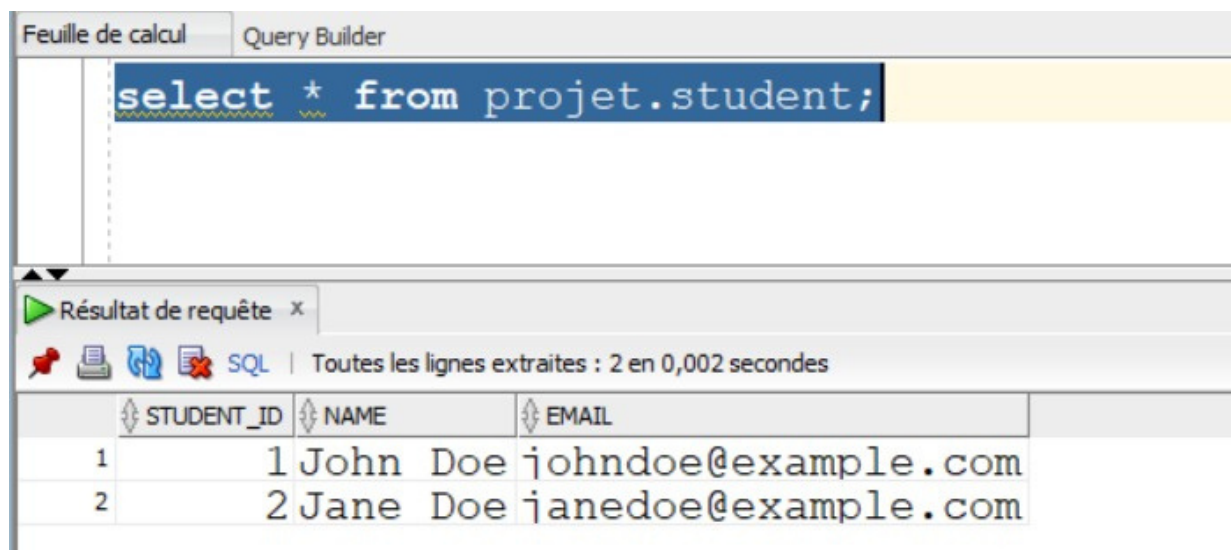
The screenshot shows the Oracle SQL Developer interface. The 'Query Builder' tab is active, displaying the following SQL query:

```
select * from projet.student;  
select * from projet.notes;  
INSERT INTO notes (note_id, student_id, subject, grade) VALUES (3, 2, 'math', 90);
```

Below the query, the 'Résultat de requête' (Query Results) window shows the output of the query. It indicates that 3 lines were extracted in 0.002 seconds. The results are displayed in a table with the following columns: NOTE_ID, STUDENT_ID, SUBJECT, and GRADE.

	NOTE_ID	STUDENT_ID	SUBJECT	GRADE
1	1	1	History	89
2	2	1	math	89
3	3	2	math	90

mais on peut afficher la table student avec u1



The screenshot shows the Oracle SQL Developer interface. The 'Query Builder' tab is active, displaying the following SQL query:

```
select * from projet.student;
```

Below the query, the 'Résultat de requête' (Query Results) window shows the output of the query. It indicates that 2 lines were extracted in 0.002 seconds. The results are displayed in a table with the following columns: STUDENT_ID, NAME, and EMAIL.

	STUDENT_ID	NAME	EMAIL
1	1	John Doe	johndoe@example.com
2	2	Jane Doe	janedoe@example.com

C'est un simple exemple, en utilisant ad comme admin et u1 comme utilisateur normal on peut faire plusieurs privilèges sur des autres types d'utilisateur par exemple : professeur, directeur, étudiant...

Sous plateforme web

Notre code est un exemple de serveur Web express qui implémente une fonctionnalité de login et d'affichage de tableaux.

Lorsque l'utilisateur accède à la page d'accueil à l'aide de la méthode HTTP GET, un formulaire de connexion est affiché, où l'utilisateur peut entrer son nom d'utilisateur et son mot de passe. Lorsque l'utilisateur soumet le formulaire, la méthode HTTP POST / login est appelée. Dans cette méthode, les informations d'identification de l'utilisateur sont extraites du corps de la requête et comparées à la liste des utilisateurs autorisés. Si l'utilisateur n'est pas trouvé dans la liste, un message de login échoué est envoyé. Sinon, la connexion est enregistrée dans un fichier de journal et l'utilisateur est redirigé vers la page appropriée en fonction de son nom d'utilisateur.

Lorsque l'utilisateur accède à la page admin à l'aide de la méthode HTTP GET, un ensemble de requêtes SQL sont exécutées sur une base de données Oracle. Le résultat de chaque requête est converti en HTML pour une affichage en tableau sur la page.

Sous plateforme web

```
app.get('/admin', async(req, res) => {
  let connection;
  try {
    connection = await oracledb.getConnection({
      user: "ad",
      password: "ad",
      connectionString: "localhost/orcl"
    });
    const sqlQueries = [ "select * from projet.student", "select * from projet.notes"];
    // const sqlQueries = [ "select * from projet.car"];
    const tables = [];
    for (const sql of sqlQueries) {
      const result = await connection.execute(sql);
      const tableHTML = result.rows.map(row => {
        let rowHTML = '';
        for (let i = 0; i < result.metaData.length; i++) {
          rowHTML += `<td>${row[i]}</td>`;
        }
        return `<tr>${rowHTML}</tr>`;
      }).join('');
      tables.push({ metaData: result.metaData, tableHTML });
    }
    res.send(`
```

une connexion à une base de données Oracle est créée en utilisant les informations d'identification fournies (utilisateur : "ad" et mot de passe : "ad"). La connexion est établie avec le connecteur oracledb en spécifiant le connectionString "localhost/orcl".

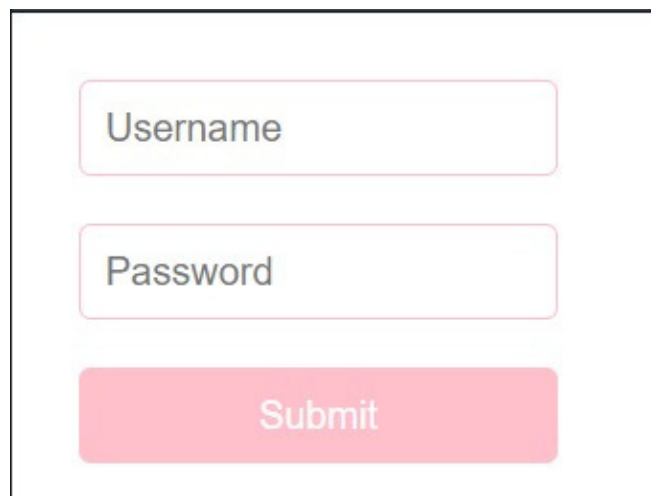
Ensuite, le code exécute deux requêtes SQL (sélectionnez * de projet.student et sélectionnez * de projet.notes) sur la base de données en utilisant la connexion et stocke les résultats dans des tables HTML. Les tables HTML sont ensuite renvoyées à la page HTML en tant que réponse à la requête.

En cas d'erreur pendant l'exécution du code, un message d'erreur est affiché. La connexion à la base de données est finalement fermée.

Meme cas pour l'utilisateur normal.

Sous plateforme web

Lorsque l'utilisateur se connecte en entrant son nom d'utilisateur et son mot de passe dans un formulaire sur la page d'accueil, le code vérifie si les informations de connexion correspondent à l'un des utilisateurs autorisés. Si les informations sont correctes, l'utilisateur sera redirigé vers une page correspondant à son type d'utilisateur (admin ou normal). Le code enregistre également le moment de la connexion dans un fichier de journal.



A login form with three input fields: 'Username', 'Password', and a 'Submit' button. The form is enclosed in a light gray border.

Si l'utilisateur est un administrateur, il peut voir une page affichant les tables student et notes de la base de données. Le code utilise la bibliothèque OracleDB pour exécuter des requêtes SQL sur ces tables et générer une table HTML pour chaque résultat.

Ici l'exemple de connexion avec u1(normal utilisateur)

Tables

STUDENT_ID	NAME	EMAIL
1	John Doe	johndoe@example.com
2	Jane Doe	janedoe@example.com

logout

Sous plateforme web

Si l'utilisateur est un administrateur, il peut voir une page affichant les tables student et notes de la base de données. Le code utilise la bibliothèque OracleDB pour exécuter des requêtes SQL sur ces tables et générer une table HTML pour chaque résultat.

Tables

STUDENT_ID	NAME	EMAIL
1	John Doe	johndoe@example.com
2	Jane Doe	janedoe@example.com

NOTE_ID	STUDENT_ID	SUBJECT	GRADE
1	1	History	89
2	1	math	89
3	2	math	90

logout

Mais si on se connect avec normal user et on veut afficher par exemple la table notes qu'on a pas donné au normal utilisateur pour la voir on va obtenir une erreur.

Error Occurred

```
nodemon] starting `node app.js`  
server Started  
the log was saved!  
error: ORA-00942: Table ou vue inexistante
```

meme chose si on veut inserer ou modifier sur une table avec l'utilisateur normal(u1)

Sous plateforme web

On peut accéder au fichier logs.txt pour voir notre journal, comme on peut ajouter pour chaque commande exécuter pour chaque utilisateur un log enregistrer dans notre journal, (ceci just un exemple).

```
logs.txt
1 [Wed Feb 01 2023 04:01:41 GMT+0100 (UTC+02:00)] User ad logged in
2 [Wed Feb 01 2023 04:01:42 GMT+0100 (UTC+02:00)] User ad logged out
3 [Wed Feb 01 2023 04:01:59 GMT+0100 (UTC+02:00)] User u1 logged in
4 [Wed Feb 01 2023 04:02:01 GMT+0100 (UTC+02:00)] User u1 logged out
5 |
```

Pour améliorer ce code, plusieurs modifications peuvent être apportées:

Sécurité: La vérification des informations d'identification de l'utilisateur n'est pas sécurisée et pourrait être compromise par des attaques de type "brute force" ou "rainbow table". Il serait préférable d'utiliser un système d'authentification plus sécurisé, comme OAuth2 ou JSON Web Tokens (JWT).

Gestion des erreurs: Il n'y a pas de gestion des erreurs dans le code, ce qui pourrait entraîner des comportements inattendus lors de la survenue d'erreurs. Il est donc important d'ajouter des blocs try-catch pour traiter les erreurs potentielles.

Abstraction de la base de données: Le code n'utilise pas de couche d'abstraction pour accéder à la base de données Oracle, ce qui peut rendre difficile le remplacement de la base de données par une autre technologie. Il peut être utile d'utiliser un ORM (Object-Relational Mapping) ou un système de gestion de bases de données de niveau supérieur pour gérer les interactions avec la base de données.

Testabilité: Le code n'est pas facile à tester et il serait préférable d'utiliser des tests unitaires pour vérifier les différentes parties du code.