**SQL QUERY EXECUTION assignment I**

This example I used in executing the query I got it from the internet while trying to learn more about the syntax of query

**1. Codes for creating tables**

-- Create Departments Table

CREATE TABLE Departments (

department\_id NUMBER PRIMARY KEY,

department\_name VARCHAR2(100)

);

-- Create Employees Table with a foreign key relationship to Departments

CREATE TABLE Employees (

employee\_id NUMBER PRIMARY KEY,

first\_name VARCHAR2(100),

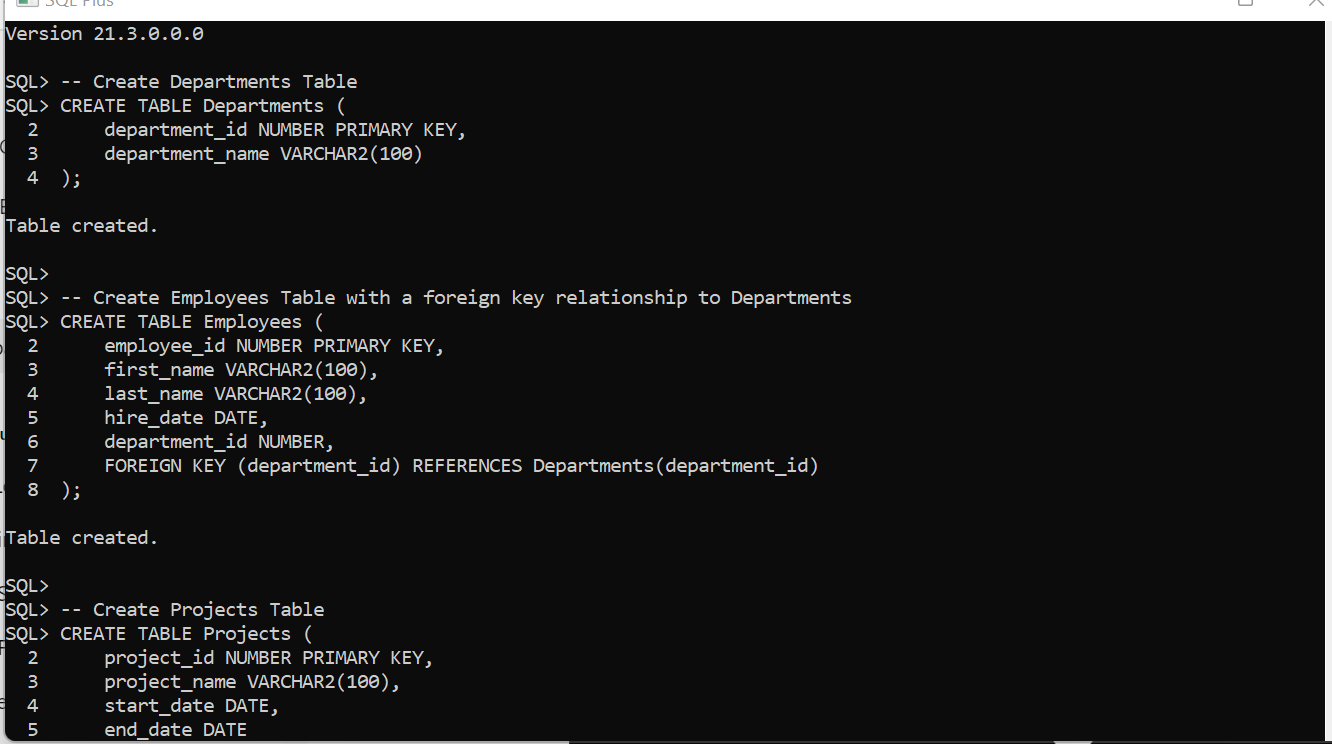
last\_name VARCHAR2(100),

hire\_date DATE,

department\_id NUMBER,

FOREIGN KEY (department\_id) REFERENCES Departments(department\_id)

);



-- Create Projects Table

CREATE TABLE Projects (

project\_id NUMBER PRIMARY KEY,

project\_name VARCHAR2(100),

start\_date DATE,

end\_date DATE

);

-- Create Salaries Table with foreign key to Employees

CREATE TABLE Salaries (

salary\_id NUMBER PRIMARY KEY,

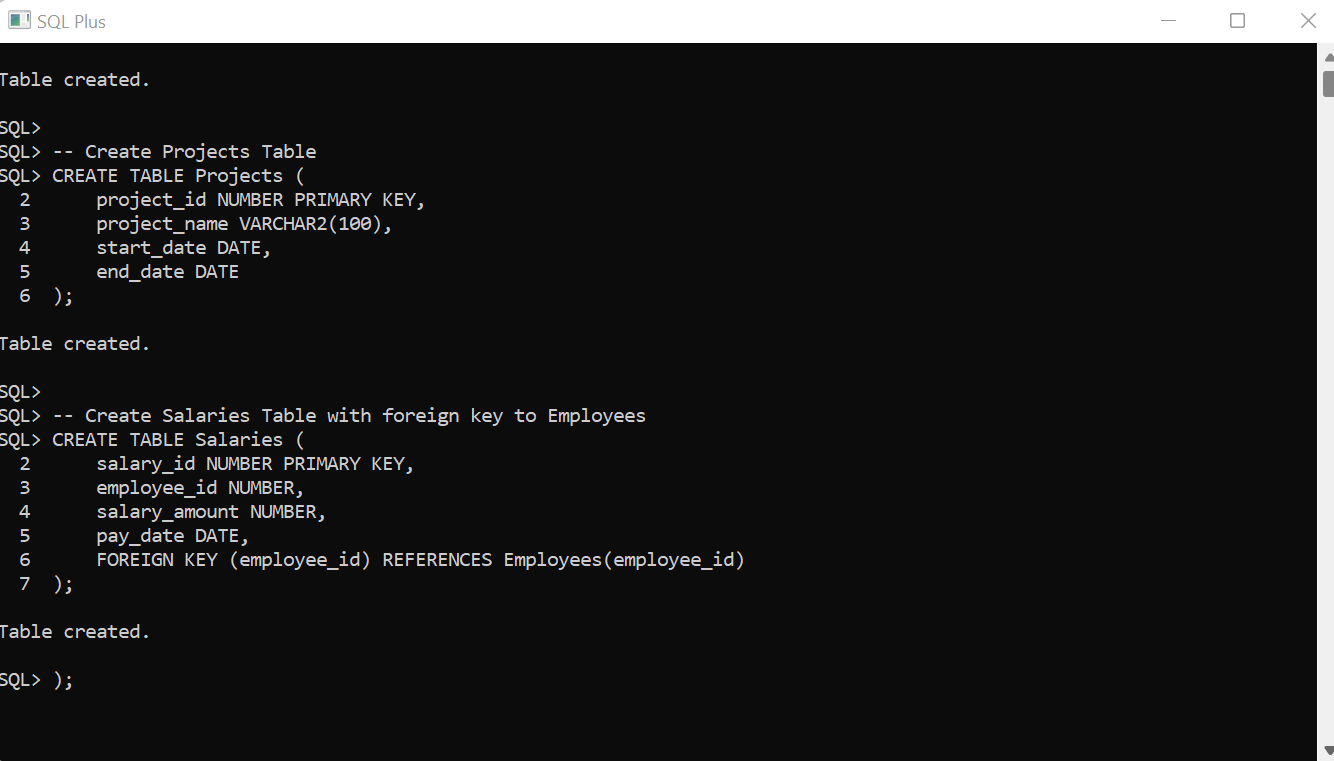
employee\_id NUMBER,

salary\_amount NUMBER,

pay\_date DATE,

FOREIGN KEY (employee\_id) REFERENCES Employees(employee\_id)

);



**2. inserting data**

-- Insert into Departments

INSERT INTO Departments (department\_id, department\_name)

VALUES (1, 'HR');

INSERT INTO Departments (department\_id, department\_name)

VALUES (2, 'Finance');

INSERT INTO Departments (department\_id, department\_name)

VALUES (3, 'Accounting');



**-- Insert into Employees**

-- Insert the first employee

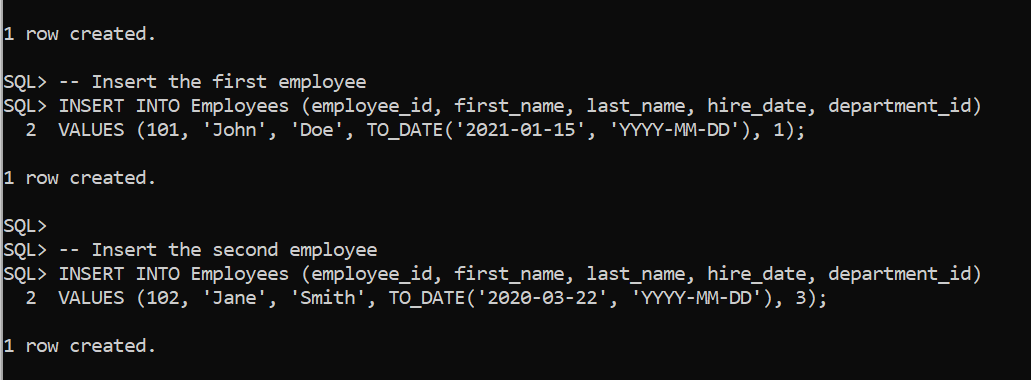
INSERT INTO Employees (employee\_id, first\_name, last\_name, hire\_date, department\_id)

VALUES (101, 'John', 'Doe', TO\_DATE('2021-01-15', 'YYYY-MM-DD'), 1);

-- Insert the second employee

INSERT INTO Employees (employee\_id, first\_name, last\_name, hire\_date, department\_id)

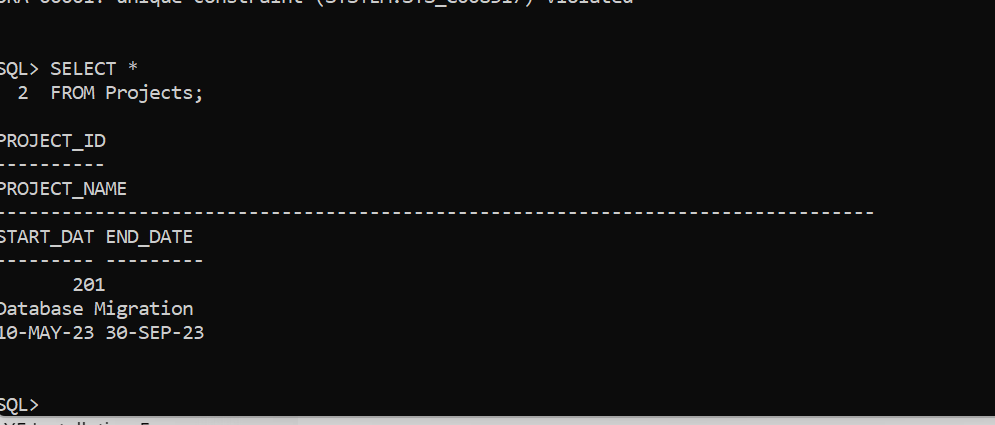
VALUES (102, 'Jane', 'Smith', TO\_DATE('2020-03-22', 'YYYY-MM-DD'), 3);



-- Insert into Projects

INSERT INTO Projects (project\_id, project\_name, start\_date, end\_date)

VALUES (201, 'Database Migration', TO\_DATE('2023-05-10', 'YYYY-MM-DD'), TO\_DATE('2023-09-30', 'YYYY-MM-DD'));



-- Insert into Salaries

-- Insert the first salary

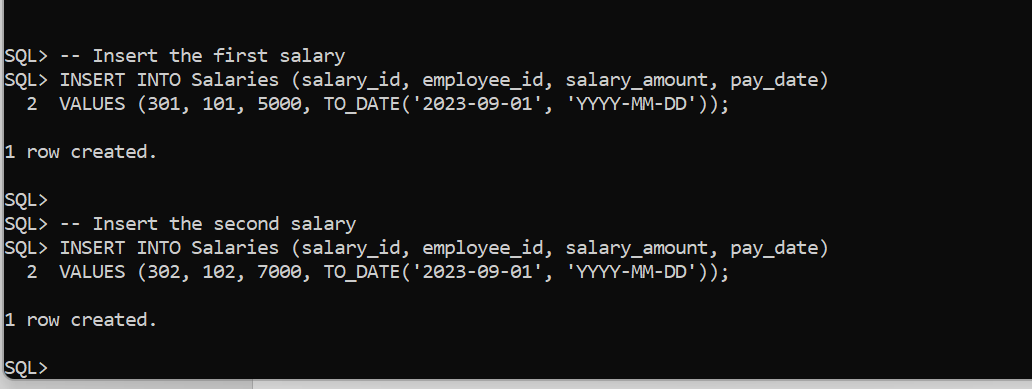
INSERT INTO Salaries (salary\_id, employee\_id, salary\_amount, pay\_date)

VALUES (301, 101, 5000, TO\_DATE('2023-09-01', 'YYYY-MM-DD'));

-- Insert the second salary

INSERT INTO Salaries (salary\_id, employee\_id, salary\_amount, pay\_date)

VALUES (302, 102, 7000, TO\_DATE('2023-09-01', 'YYYY-MM-DD'));



**3.update and delete**

-- Update an employee's salary

UPDATE Salaries

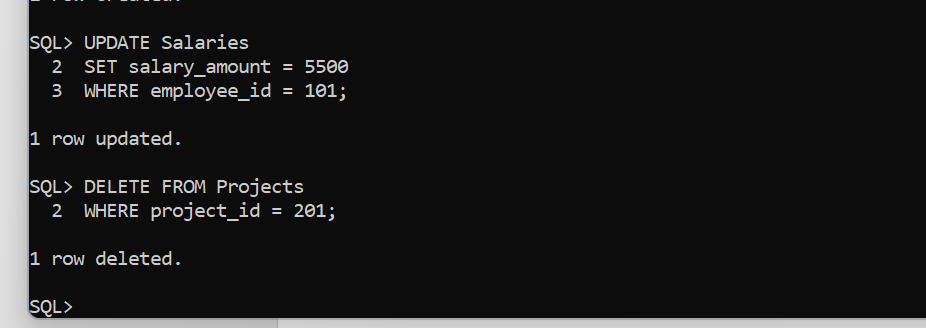
SET salary\_amount = 5500

WHERE employee\_id = 101;

-- Delete a project

DELETE FROM Projects

WHERE project\_id = 201;



**4. perform joins**

1. **Inner join**

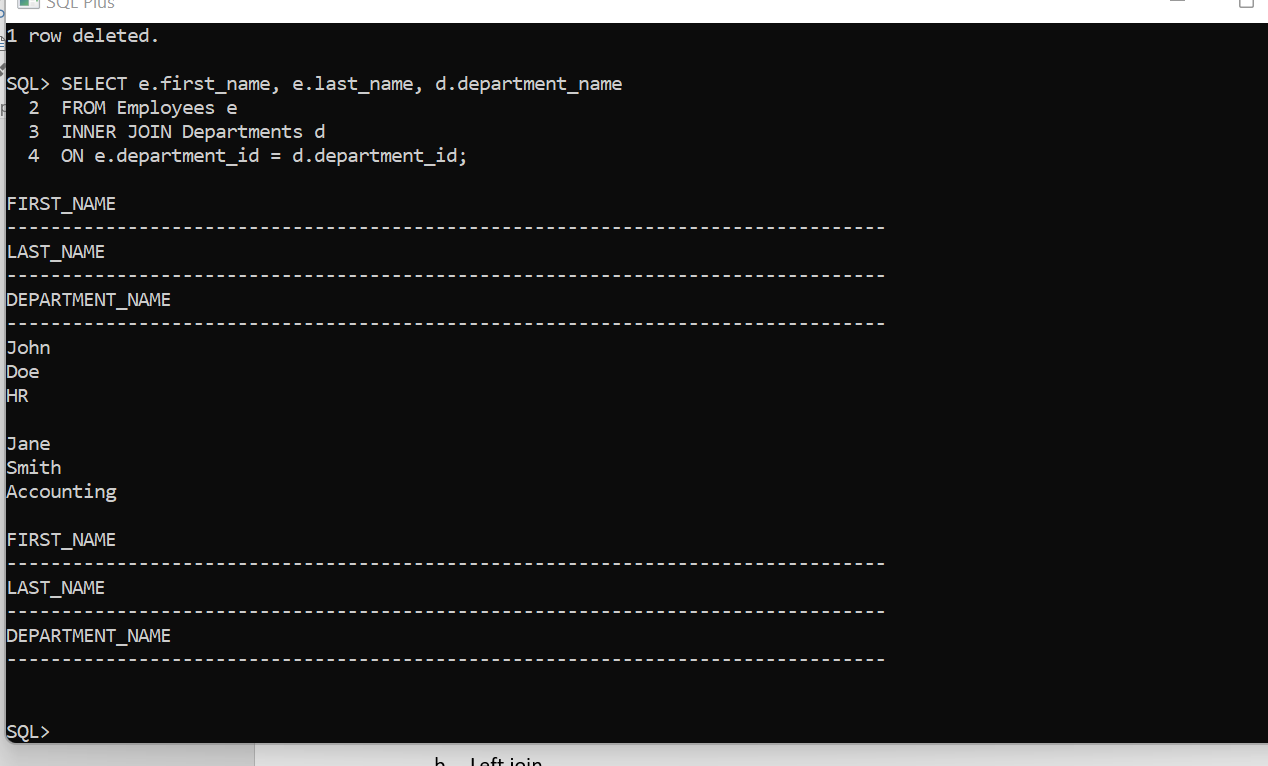
-- Join Employees with Departments to get employee names and their departments

SELECT e.first\_name, e.last\_name, d.department\_name

FROM Employees e

INNER JOIN Departments d

ON e.department\_id = d.department\_id;



1. **Left join**

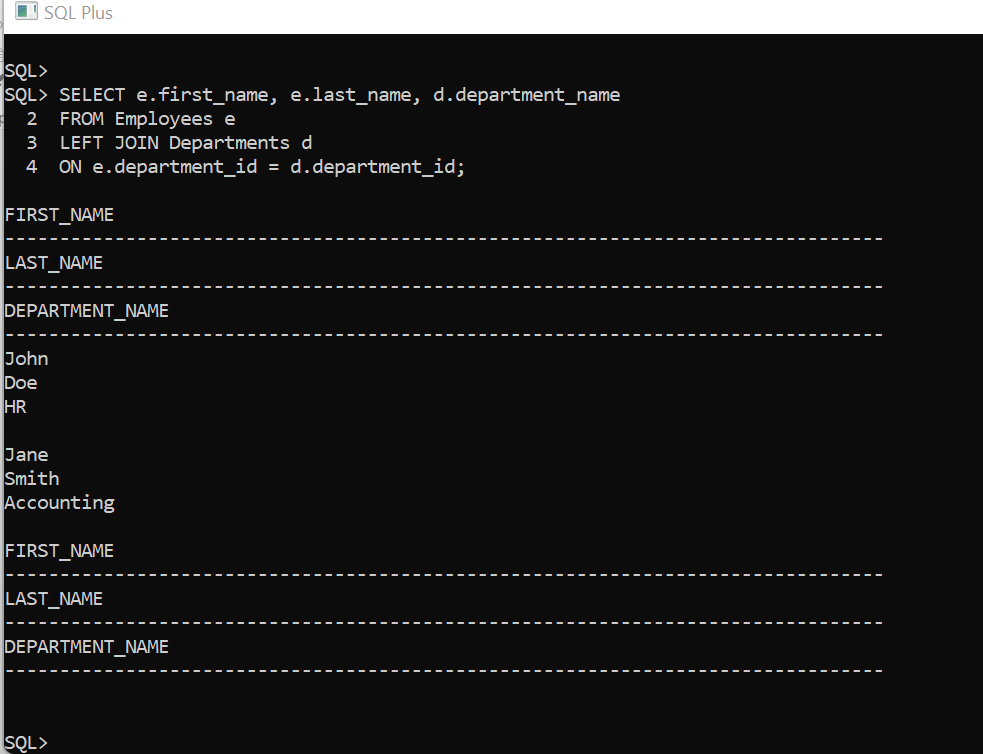
-- Retrieve all employees, including those who may not be assigned to a department

SELECT e.first\_name, e.last\_name, d.department\_name

FROM Employees e

LEFT JOIN Departments d

ON e.department\_id = d.department\_id;



1. **Right join**

-- Retrieve all departments, including those that have no employees assigned

SELECT e.first\_name, e.last\_name, d.department\_name

FROM Employees e

RIGHT JOIN Departments d

ON e.department\_id = d.department\_id;

1. **Full outer join**

-- Retrieve all employees and all departments, showing NULL where there is no match

SELECT e.first\_name, e.last\_name, d.department\_name

FROM Employees e

FULL OUTER JOIN Departments d

ON e.department\_id = d.department\_id;

1. **Join with multiple tables**

-- Join Employees, Departments, and Salaries to show employee details, salary, and department

SELECT e.first\_name, e.last\_name, d.department\_name, s.salary\_amount

FROM Employees e

INNER JOIN Departments d

ON e.department\_id = d.department\_id

INNER JOIN Salaries s

ON e.employee\_id = s.employee\_id;

1. **Subquery with join**

-- Get employees whose salary is above the average salary

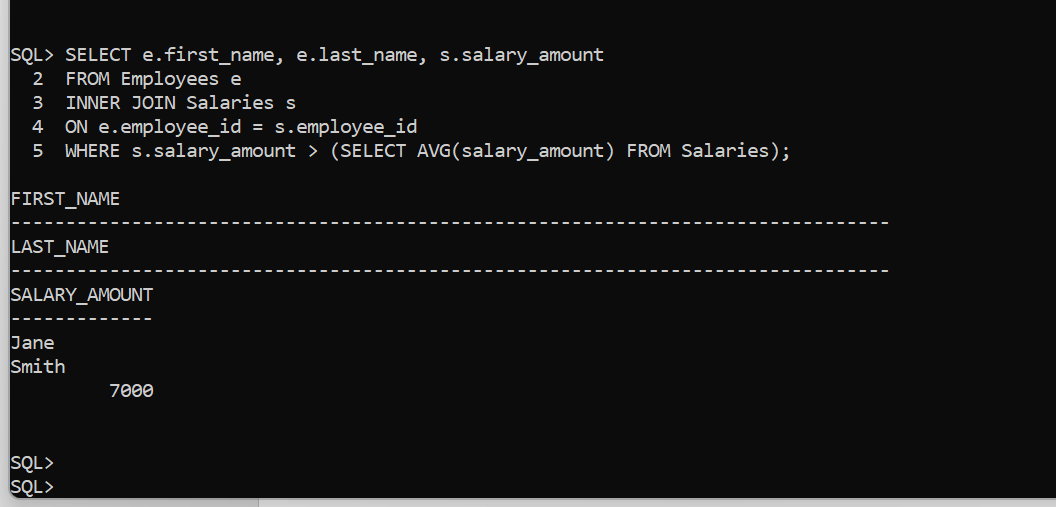
SELECT e.first\_name, e.last\_name, s.salary\_amount

FROM Employees e

INNER JOIN Salaries s

ON e.employee\_id = s.employee\_id

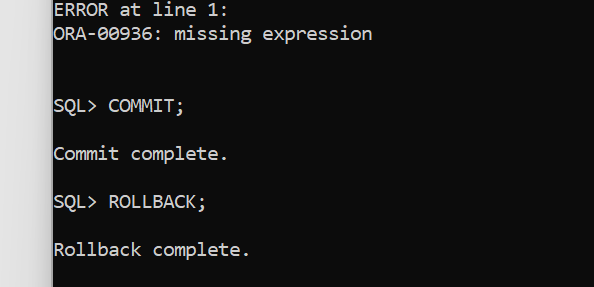
WHERE s.salary\_amount > (SELECT AVG(salary\_amount) FROM Salaries);



COMMIT; -- Save changes

-- or

ROLLBACK; -- Undo changes

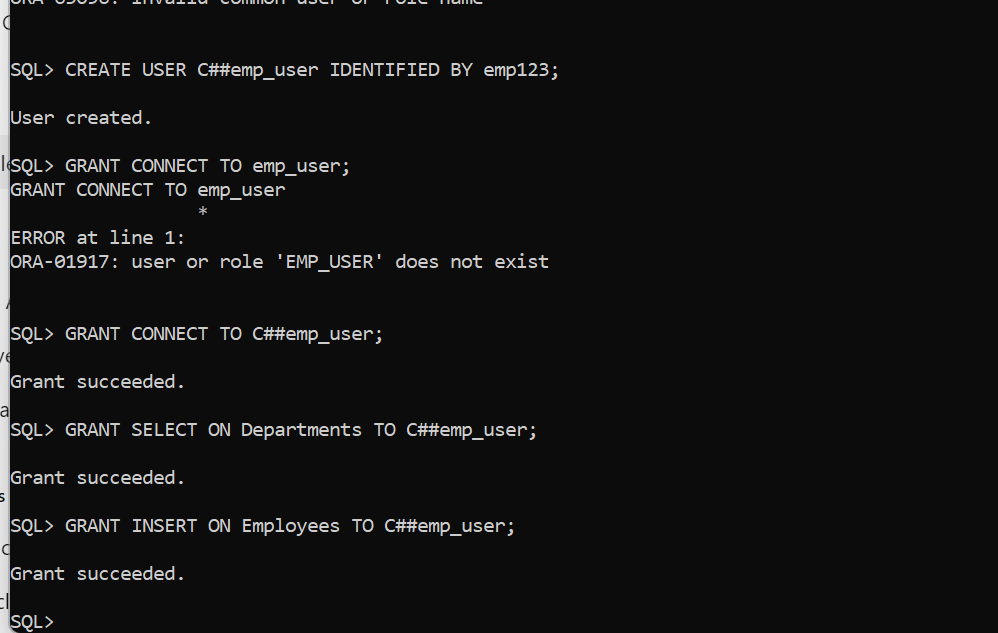


CREATE USER emp\_user IDENTIFIED BY emp123;

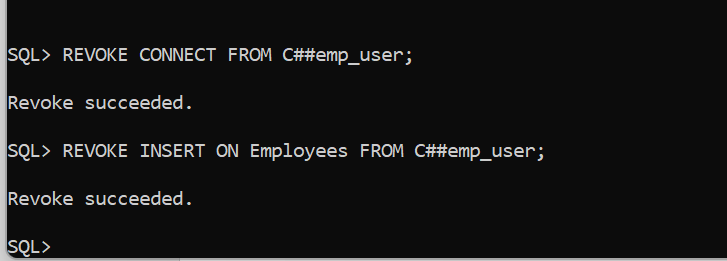
GRANT CONNECT TO emp\_user;

GRANT SELECT ON Departments TO emp\_user;

GRANT INSERT ON Employees TO emp\_user;



REVOKE INSERT ON Employees FROM emp\_user;



REFERENCES

1. Open ai