

ACCA7002 Database Systems

Tutorial Tasks

Tutorial 8: Perform transactions with sample data

The aim of this exercise is to help you practice starting new transactions, committing, and rolling back in PostgreSQL

Task 8.1: Create a table

```
CREATE TABLE Orders (  
    order_id serial PRIMARY KEY,  
    customer_name varchar(255),  
    order_date date  
);  
  
CREATE TABLE OrderItems (  
    item_id serial PRIMARY KEY,  
    order_id integer REFERENCES Orders(order_id),  
    product_name varchar(255),  
    quantity integer,  
    price numeric(10, 2)  
);
```

Task 8.2: Insert Sample Data:

```
-- Start a transaction  
BEGIN;  
  
-- Insert orders  
INSERT INTO Orders (customer_name, order_date) VALUES  
('Customer A', '2023-01-15');  
INSERT INTO Orders (customer_name, order_date) VALUES  
('Customer B', '2023-01-16');  
  
-- Insert order items  
INSERT INTO OrderItems (order_id, product_name, quantity,  
price) VALUES (1, 'Product X', 5, 10.00);  
INSERT INTO OrderItems (order_id, product_name, quantity,  
price) VALUES (1, 'Product Y', 3, 8.00);  
INSERT INTO OrderItems (order_id, product_name, quantity,  
price) VALUES (2, 'Product Z', 2, 15.00);  
  
-- Commit the transaction  
COMMIT;
```

Task 8.3: Start a New Transaction and Rollback:

```
-- Start a new transaction
BEGIN;

-- Attempt to insert data, but then decide to rollback
INSERT INTO Orders (customer_name, order_date) VALUES
('Customer C', '2023-01-18');
-- You realize you made a mistake and want to undo this action
-- Rollback the transaction to remove the inserted data
ROLLBACK;
```

Task 8.4: Start Another Transaction and Commit:

```
-- Start a new transaction
BEGIN;

-- Insert a new order
INSERT INTO Orders (customer_name, order_date) VALUES
('Customer D', '2023-01-18');

-- Commit the transaction to save the new order
COMMIT;
```

Task 8.5:

- a) Create a PostgreSQL database and connect to it.
- b) Create a new table called "Employees" with the following columns:

EmployeeID (Primary Key)
FirstName
LastName
DepartmentID (Foreign Key referencing a "Departments" table)
Salary

- c) Create a table called "Departments" with the following columns:

DepartmentID (Primary Key)
DepartmentName

- d) Insert 10 sample records into the "Employees" table.
- e) Define an index on the "EmployeeID" column of the "Employees" table.
- f) Write an SQL query to retrieve all records from the "Employees" table.
- g) Write an SQL query to retrieve the first and last names of all employees who earn more than £50,000.

h) Group the employees by their department and calculate the average salary for each department.

i) Perform a LEFT JOIN between the "Employees" and "Departments" tables, showing all employees and their corresponding department names.

j) Perform an INNER JOIN between the "Employees" and "Departments" tables, showing only the employees who belong to a department.

k) Update the salary of an employee with a specific EmployeeID.

l) Delete an employee record based on the EmployeeID.

m) Start a new transaction. Insert a new employee record and make sure it is saved only if the transaction is committed.

n) Perform a ROLLBACK to undo the changes made in the previous transaction.

o) Perform a COMMIT to save the changes made in the current transaction.