



Term: Fall 2023 **Subject:** Computer Science & Engineering (CSE) **Number:** 512

Course Title: Distributed Database Systems (CSE 512)

Project Deliverable - 4

1. Implement ACID-compliant distributed transactions using the chosen database system (e.g., PostgreSQL, CockroachDB).
2. Propose concurrency control mechanisms to handle simultaneous transactions.

– Explore any tools for distributed coordination and managing distributed transactions.

Summary of the Python Script

The Python script is designed to interact with a Citus-based PostgreSQL database cluster. It provides a set of functions to manage supplier data within a Supply Chain Management (SCM) system. Here are the key functionalities implemented in the script:

- **Database Connection (connect function):** Establishes a connection to the PostgreSQL database, specifically to a database named scm running on localhost, accessed with the username postgres and password pass1.
- **Adding Suppliers (add_supplier function):** Adds a new supplier to the Suppliers table. It generates a unique UUID for each supplier, and takes the supplier's name, contact information, and product categories as inputs.
- **Selecting Suppliers (select_suppliers function):** Retrieves and displays all entries from the Suppliers table, showing the current list of suppliers in the database.
- **Updating Suppliers (update_supplier function):** Updates the name of a supplier based on a given UUID. It demonstrates the ability to modify existing records in the database.
- **Deleting Suppliers (delete_supplier function):** Removes a supplier from the database using their UUID, showcasing the script's capability to handle data deletion.
- **Viewing Data Distribution (get_data_distribution_details function):** A special function to get insights into how data is distributed across the Citus cluster. It queries Citus metadata tables to provide information about data sharding and placement across nodes.

Summary of the Docker Compose File

The Docker Compose file sets up a Citus cluster with one master (coordinator) node and two worker nodes. Here's an overview of its configuration:

- **Citus Cluster Configuration:** Defines a service for the master node and two services for worker nodes, all using the citusdata/citus Docker image.
- **Environment Setup:** Each node is configured with the environment variables POSTGRES_PASSWORD (set to pass1) and POSTGRES_HOST_AUTH_METHOD (set to trust) for authentication and access control.
- **Port Mapping:** The master node's PostgreSQL port (5432) is mapped to the host machine, enabling direct database interactions from the host.
- **Data Initialization:** The Compose file specifies volumes to mount insert_data.sql and schema.sql files into the master node. These SQL files are used to initialize the database schema and insert initial data.
- **Labels:** Uses labels such as "com.citusdata.role=Master" and "com.citusdata.role=Worker" to distinguish between master and worker nodes.

Conclusion

In summary, the Python script and Docker Compose file together establish a functional SCM database system with a Citus cluster backend. The script provides essential database operations for supplier management, while the Docker Compose file ensures an easy and consistent setup of the distributed database environment. This setup demonstrates key concepts in distributed databases, like data sharding and management, within a containerized environment.

Screenshots:




















<input type="checkbox"/>	▼	 documents	Running (3/3)	24.78%			
<input type="checkbox"/>		 worker2-1 113d3d05bcef  citusdata/citus:latest	Running	0%			
<input type="checkbox"/>		 worker1-1 782ef507dc06  citusdata/citus:latest	Running	24.16%			
<input type="checkbox"/>		 master-1 466e735e87b0  citusdata/citus:latest	Running	0.62% 5432:5432			

Fig. 1 : All the nodes containers, master and workers, running on docker

```
Supplier added with ID: a4a21efa-434e-4ced-bd7a-3239a7745e12
After Addition:
('00000000-0000-0000-0001-000000000001', 'ElectroTech Suppliers', '123-456-7890', ['Electronics'])
('a4a21efa-434e-4ced-bd7a-3239a7745e12', 'Supplier-1', '+1-9900220033', ['Electronics'])
('00000000-0000-0000-0002-000000000002', 'Fashion Trends Supplier', '987-654-3210', ['Clothing and Fashion'])
('00000000-0000-0000-0003-000000000003', 'BookWorld Publishers', '555-123-4567', ['Books'])
Supplier updated with ID: a4a21efa-434e-4ced-bd7a-3239a7745e12
After Update:
('00000000-0000-0000-0001-000000000001', 'ElectroTech Suppliers', '123-456-7890', ['Electronics'])
('a4a21efa-434e-4ced-bd7a-3239a7745e12', 'Electronic Supplier', '+1-9900220033', ['Electronics'])
('00000000-0000-0000-0002-000000000002', 'Fashion Trends Supplier', '987-654-3210', ['Clothing and Fashion'])
('00000000-0000-0000-0003-000000000003', 'BookWorld Publishers', '555-123-4567', ['Books'])
Supplier deleted with ID: a4a21efa-434e-4ced-bd7a-3239a7745e12
After Deletion:
('00000000-0000-0000-0001-000000000001', 'ElectroTech Suppliers', '123-456-7890', ['Electronics'])
('00000000-0000-0000-0002-000000000002', 'Fashion Trends Supplier', '987-654-3210', ['Clothing and Fashion'])
('00000000-0000-0000-0003-000000000003', 'BookWorld Publishers', '555-123-4567', ['Books'])
```

Fig. 2 : The data stored in the supplier database, fetched using the python script.

```
('orders', 102008, 'localhost', 5432, '-2147483648', '-2013265921')
('orders', 102009, 'localhost', 5432, '-2013265920', '-1879048193')
('orders', 102010, 'localhost', 5432, '-1879048192', '-1744830465')
('orders', 102011, 'localhost', 5432, '-1744830464', '-1610612737')
('orders', 102012, 'localhost', 5432, '-1610612736', '-1476395009')
('orders', 102013, 'localhost', 5432, '-1476395008', '-1342177281')
('orders', 102014, 'localhost', 5432, '-1342177280', '-1207959553')
('orders', 102015, 'localhost', 5432, '-1207959552', '-1073741825')
('orders', 102016, 'localhost', 5432, '-1073741824', '-939524097')
('orders', 102017, 'localhost', 5432, '-939524096', '-805306369')
('orders', 102018, 'localhost', 5432, '-805306368', '-671088641')
('orders', 102019, 'localhost', 5432, '-671088640', '-536870913')
('orders', 102020, 'localhost', 5432, '-536870912', '-402653185')
('orders', 102021, 'localhost', 5432, '-402653184', '-268435457')
('orders', 102022, 'localhost', 5432, '-268435456', '-134217729')
```

Fig 3: Data that is stored in the pg_dist_shard table, which holds the sharding details of each table (Orders table in this case)