BHARATIYA VIDYA BHAVANS
SARDAR PATEL INSTITUTE OF TECHNOLOGY
MUNSHI NAGAR, ANDHERI (WEST), MUMBAI - 400 058
AUTONOMOUS INSTITUTE AFFILIATED TO MUMBAI UNIVERSITY

Adwait Purao- 2021300101

TE Comps B - Batch B

DC LAB

DISTRIBUTED COMPUTING EXPERIMENT 8

Aim:

Implementation of a small application using data replication.

Theory:

Data Replication:
In a distributed system data is stored over different computers in a network.
Therefore, we need to make sure that data is readily available for the users. Availability of the data is an important factor often accomplished by data replication. Replication is the practice of keeping several copies of data in different places.

Why do we require replication?
The first and foremost thing is that it makes our system more stable because of node replication.
It is good to have replicas of a node in a network due to following reasons:

- If a node stops working, the distributed network will still work fine due to its replicas which will be there. Thus, it increases the fault tolerance of the system.
- It also helps in load sharing where loads on a server are shared among different replicas.
- It enhances the availability of the data. If the replicas are created and data is stored near to the consumers, it would be easier and faster to fetch data.

❖ Types of Replications:

1. Active Replication
2. Passive Replication

Active Replication:

- The request of the client goes to all the replicas. It is to be made sure that every replica receives the client request in the same order else the system will get inconsistent.
- There is no need for coordination because each copy processes the same request in the same sequence.
- All replicas respond to the client's request.

➢ Advantages:

- It is simple. The codes in active replication are the same throughout.
- It is transparent.
- Even if a node fails, it will be easily handled by replicas of that node.

➢ Disadvantages:

- It increases resource consumption. The greater the number of replicas, the greater the memory needed.
- It increases the time complexity. If some change is done on one replica it should also be done in all others.

Passive Replication:

- The client request goes to the primary replica, also called the main replica.
- There are more replicas that act as backup for the primary replica.
- Primary replica informs all other backup replicas about any modification done.
- The response is returned to the client by a primary replica.
- Periodically the primary replica sends some signal to backup replicas to let them know that it is working perfectly fine.
- In case of failure of a primary replica, a backup replica becomes the primary replica.

➢ Advantages:

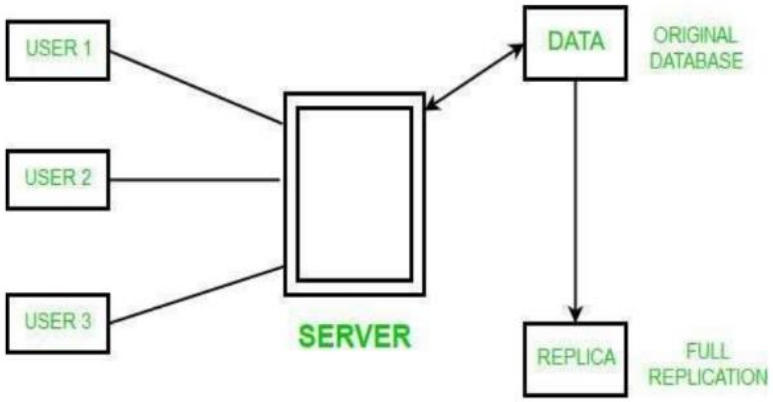
- The resource consumption is less as backup servers only come into play when the primary server fails.

- The time complexity of this is also less as there's no need for updating in all the nodes replicas, unlike active replication.

➢ Disadvantages:

- If some failure occurs, the response time is delayed.

Diagram showing Replication:-



Workflow:

1. The server starts and clients have several options to view or make changes in the Database.

2. If a client makes changes, the server communicates it to the primary database. These changes are then actively replicated to the backup database.

3. If a client views the database, it is fetched from the primary database. The backup database is not used for viewing data in the current implementation.

Fetching data from the backup database (when a client views the database), might not be a good practice because the backup database is typically used for recovery purposes and not for serving read requests. Serving read requests from the backup database might slow down the replication process and make the backup database unavailable in case of a failure in the primary database.

Code:

```
import sqlite3
from multiprocessing import Process, Manager


def create_table(connection):
        cursor = connection.cursor()
   cursor.execute('''CREATE TABLE IF NOT EXISTS users
                  (id INTEGER PRIMARY KEY AUTOINCREMENT,
                   username TEXT,
                   email TEXT)''')
   connection.commit()


def print_replicated_data(shared_data, destination_names):
   print("\nReplicated Data:")
        for i, name in enumerate(destination_names):
       print(f"\nDatabase {i + 1}: {name}")
        for user in shared_data[i::len(destination_names)]:
          print(f"   User: {user[1]}, Email: {user[2]}")


def menu_add_data(source_conn, replicate_to_dest):
        username = input("Enter username: ")
```

```
        email = input("Enter email: ")
    insert_user(source_conn, username, email)
        print("User added to the source database.")

        if replicate_to_dest:
       replicate_data_to_all_dest(source_conn, destination_dbs, shared_data)

def replicate_data(source_conn, destination_conn, shared_data):
        source_cursor = source_conn.cursor()
        destination_cursor = destination_conn.cursor()

    source_cursor.execute('SELECT * FROM users')
        users = source_cursor.fetchall()

    shared_data.extend(users)

        for user in users:
       destination_cursor.execute('INSERT INTO users (username, email) VALUES
(?, ?)', (user[1], user[2]))

    destination_conn.commit()
```

```
def insert_user(connection, username, email):
        cursor = connection.cursor()
    cursor.execute('INSERT INTO users (username, email) VALUES (?, ?)', (username, email))
    connection.commit()


def replicate_data_to_all_dest(source_conn, destination_dbs, shared_data):
        processes = []
        for destination_db in destination_dbs:
       replication_process = Process(target=replicate_data, args=(source_conn, destination_db, shared_data))
       replication_process.start()
       processes.append(replication_process)


        for process in processes:
        process.join()


        print("Data replicated to all destination databases.")


def add_replicas(source_conn, num_replicas):
        for _ in range(num_replicas):
```

```
        destination_db = sqlite3.connect(f'destination_database_{len(destination_dbs) + 1}.db')
        create_table(destination_db)
        destination_dbs.append(destination_db)
        destination_names.append(f'destination_database_{len(destination_dbs)}')

    replicate_data_to_all_dest(source_conn, destination_dbs, shared_data)

    print(f"{num_replicas} replicas added successfully.")

source_db = sqlite3.connect('source_database.db')
manager = Manager()
shared_data = manager.list()

create_table(source_db)

num_files = int(input("Enter the number of destination databases to create: "))
destination_dbs = [sqlite3.connect(f'destination_database_{i + 1}.db') for i in range(num_files)]
destination_names = [f'destination_database_{i + 1}' for i in range(num_files)]

for destination_db in destination_dbs:
```

```
    create_table(destination_db)


while True:
    print("-----------------------------------------")
    print("\nMenu:")
    print("1->Add New Data to the Source Database")
    print("2->Replicate Entire Data to all the Replicated Database")
    print("3->Print All the Replicated Databases")
    print("4->Exit")
    print("-----------------------------------------")


        choice = input("Enter your choice: ")


        if choice == '1':
      replicate_option = input("Do you want to replicate the newly added data to replicated databases? (yes/no): ").lower()
      replicate_to_dest = True if replicate_option == 'yes' else False
      menu_add_data(source_db, replicate_to_dest)
        elif choice == '2':
      replicate_data_to_all_dest(source_db, destination_dbs, shared_data)
        elif choice == '3':
        print_replicated_data(shared_data, destination_names)
```

```
        elif choice == '4':
        break
        else:
      print("Invalid choice. Please enter a valid option.")


source_db.close()
for destination_db in destination_dbs:
    destination_db.close()
```

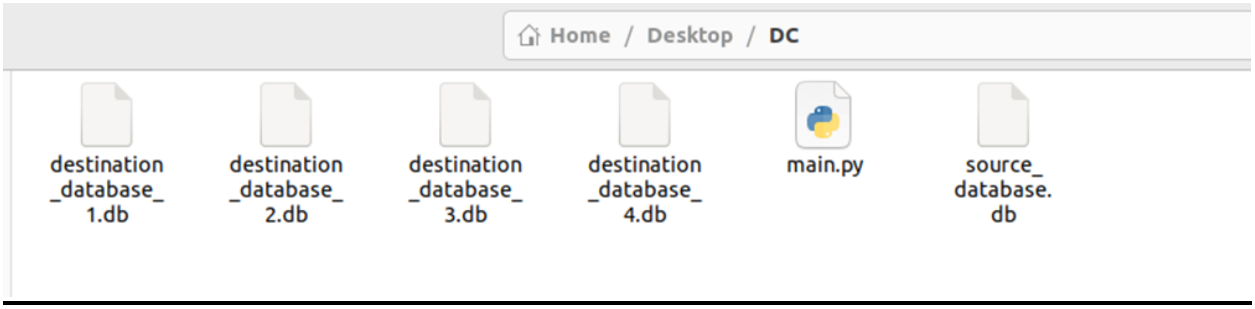
Output:

Implementation 1:

```
Enter the number of destination databases to create: 4
-----------------------------------------

Menu:
1->Add New Data to the Source Database
2->Replicate Entire Data to all the Replicated Database
3->Print All the Replicated Databases
4->Exit
-----------------------------------------
Enter your choice: 1
Do you want to replicate the newly added data to replicated databases? (yes/no): no
Enter username: abc
Enter email: abc@gmail.com
User added to the source database.
-----------------------------------------

Menu:
1->Add New Data to the Source Database
2->Replicate Entire Data to all the Replicated Database
3->Print All the Replicated Databases
4->Exit
-----------------------------------------
Enter your choice: 2
Data replicated to all destination databases.
-----------------------------------------

Menu:
1->Add New Data to the Source Database
2->Replicate Entire Data to all the Replicated Database
3->Print All the Replicated Databases
4->Exit
-----------------------------------------
Enter your choice: 3

Replicated Data:

Database 1: destination_database_1
   User: abc, Email: abc@gmail.com

Database 2: destination_database_2
   User: abc, Email: abc@gmail.com

Database 3: destination_database_3
   User: abc, Email: abc@gmail.com

Database 4: destination_database_4
   User: abc, Email: abc@gmail.com
-----------------------------------------

Menu:
1->Add New Data to the Source Database
2->Replicate Entire Data to all the Replicated Database
3->Print All the Replicated Databases
4->Exit
-----------------------------------------
Enter your choice: 4
```

Folder Before:

Folder After:

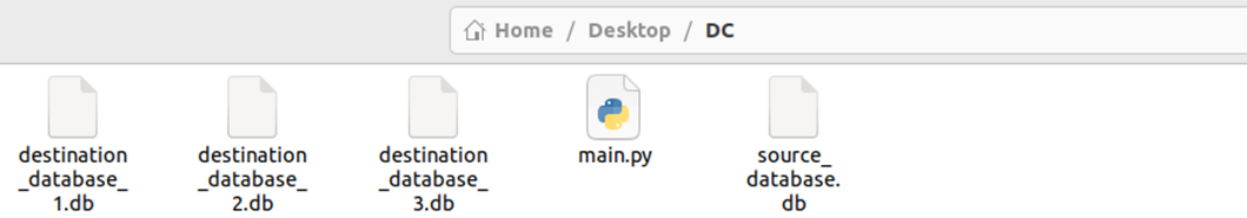


Implementation 2:

```
Enter the number of destination databases to create: 3
----------------------------------------

Menu:
1->Add New Data to the Source Database
2->Replicate Entire Data to all the Replicated Database
3->Print All the Replicated Databases
4->Exit
----------------------------------------
Enter your choice: 1
Do you want to replicate the newly added data to replicated databases? (yes/no): yes
Enter username: xyz
Enter email: xyz@gmail.com
User added to the source database.
Data replicated to all destination databases.
----------------------------------------

Menu:
1->Add New Data to the Source Database
2->Replicate Entire Data to all the Replicated Database
3->Print All the Replicated Databases
4->Exit
----------------------------------------
Enter your choice: 3

Replicated Data:

Database 1: destination_database_1
   User: xyz, Email: xyz@gmail.com

Database 2: destination_database_2
   User: xyz, Email: xyz@gmail.com

Database 3: destination_database_3
   User: xyz, Email: xyz@gmail.com
----------------------------------------

Menu:
1->Add New Data to the Source Database
2->Replicate Entire Data to all the Replicated Database
3->Print All the Replicated Databases
4->Exit
----------------------------------------
Enter your choice: 4
```

Folder Before:

Folder After:



Conclusion:

The experiment concluded that data replication significantly enhances system reliability and fault tolerance. By replicating data across multiple nodes, the application demonstrated improved resilience to node failures and ensured consistent data availability. This approach contributes to a more robust and reliable system architecture, particularly in distributed computing environments.