```
# package to handle thread-base parallelism
import threading
# package to handle time related functions
import time

print("imported some packaged")

# represents a distrunted database system
class DatabaseNode:
  def __init__(self, node_id):
    # unique indentifier for each node
     self.node_id = node_id
    #data stored locally within the node
     self.data = {}
    # list of replica nodes
     self.replica_nodes = []

# stimulates a write  opereration on the datanase node
  def write_data(self, key, value) :
      print(f"Node {self.node_id} : Wrote Opertation - Key : {key}, Value : {value}")
      self.data[key] = value

      for replica_node in self.replica_nodes:
        replica_node.recieve_replication(key, value)

# recieve replicated data from other nodes
  def recieve_replication(self, key, value) :
      print(f"Node {self.node_id} : Replication - Key : {key} , Value : {value}")
      self.data[key] = value

# stimulates a read operation on the database node
  def read_data(self, key) :
    print(f"Node {self.node_id}: Read Operation - Key: {key} , Value: {self.data.get(key, 'Not found')}")
    return self.data.get(key, None)

print("created the node that represents the distributed database system")

# simulates a continous stream of wrote operations on a database node
def simulated_writes(nodes) :
     # used to generate unique keys for write operatin
      i = 0
      # continoius loop
      while True:
          nodes.write_data(f" k - {i}" , f" v - {i}")
          # ensure unique key-valuse pair
          i =+ 1
          # pause executino for 2 seconds before the next iteration
          time.sleep(2)

print("defineed the methods to handle simulating a continous stream of write operations")

# create two node instances
node1 = DatabaseNode(1)
node2 = DatabaseNode(2)

# set up replication bewtween the two nodes
```

```
node1.replica_nodes.append(node2)
node2.replica_nodes.append(node1)

print("initiialized the node_instances and setup mode relication")

# start write operations for the nodel in a seprate thread
threading.Thread(target=simulated_writes, args=(node1,)).start()

# intiates a read opertion on nodel
node1.read_data("ket0")
# pause 5 seconds to all write poerations to be replaced bewtweeen the nodes before reading again
time.sleep(5)
# performs a simliar rea poeration on node allow for replication of write operations bewtwwen the nodes
node2.read_data("key0")
```

```
imported some packaged
created the node that represents the distributed database system
defineed the methods to handle simulating a continous stream of write operations
initiialized the node_instances and setup mode relication
Node 1 : Wrote Opertation - Key :  k - 0, Value :  v - 0Node 1: Read Operation - Key: ket0 , Value:

Node 2 : Replication - Key :  k - 0 , Value :  v - 0
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 1 : Wrote Opertation - Key :  k - 1, Value :  v - 1
Node 2 : Replication - Key :  k - 1 , Value :  v - 1
Node 2: Read Operation - Key: key0 , Value: Not found
```

Start coding or generate with AI.