

ADBMS

Exp6

Dhruv Bheda
60004200102
B/B1

ADBMS
Exp-6

Aim : Implementation of 2-Phase Protocol

Theory :

- The two-phase commit protocol breaks a database commit into two phases to ensure correctness & fault tolerance in a distributed database system.
- Phase 1: Prepare Phase
 - After each slave has locally completed its transaction, it sends a "DONE" message to the controlling site.
 - When the controlling site has received "DONE" message from all slaves, it sends a "Prepare" message to slaves.
 - The slaves vote on whether they still want to commit or not.
 - If a slave wants to commit it sends "Ready" message.
 - Otherwise, it sends "Not Ready" message.
- Phase 2: Commit/Abort Phase
 - After the controlling site has received "Ready" message from all slaves:-
 - The controlling site sends "Global Commit" msg to slaves.
 - i) The slaves apply the transaction & send a "Commit ACK" message to controlling site.

Sundaram®
FOR EDUCATIONAL USE

Disadvantages:-

- a) The major disadvantage of the two-phase commit protocol is faced when Coordinator site failure may result in blocking, so a decision either to commit or abort Transaction (T) may have to be postponed until Coordinator recovers.
- b) Consider a scenario, if a Transaction (T) holds lock on data items of active sites, but - amid the execution, if coordinator fails & the active sites keep no additional log-record except $\langle read\ T \rangle$ like $\langle abort \rangle$ or $\langle commit \rangle$. So, it becomes impossible to determine what decision has been made to $\langle commit \rangle / \langle abort \rangle$.

Conclusion:-

Thus, we successfully studied and implemented the 2-Phase Protocol.

Code:

Client

```
import java.io.*;
import java.net.*;
public class Client implements Runnable
{
    static Socket clientSocket = null;
    static PrintStream os = null;
    static DataInputStream is = null;
    static BufferedReader inputLine = null;
    static boolean closed = false;
    public static void main(String[] args)
    {
        int port_number=1111;
        String host="localhost";
        try {
            clientSocket = new Socket(host, port_number);
            inputLine = new BufferedReader(new
InputStreamReader(System.in));
            os = new PrintStream(clientSocket.getOutputStream());
            is = new DataInputStream(clientSocket.getInputStream());
        } catch (Exception e)
        {   System.out.println("Exception occurred : "+e.getMessage());
        }

        if (clientSocket != null && os != null && is != null)
        {
            try
            {
                new Thread(new Client()).start();
                while (!closed)
                {
                    os.println(inputLine.readLine());
                }
                os.close();
                is.close();
                clientSocket.close();
            } catch (IOException e)
            {
                System.err.println("IOException: " + e);
            }
        }
    }
}
```

```

    }
}
}
@SuppressWarnings("deprecation")
public void run() {
    String responseLine;
    try
    {
        while ((responseLine = is.readLine()) != null)
        {
            System.out.println("\n"+responseLine);
            if (responseLine.equalsIgnoreCase("GLOBAL_COMMIT")==true
|| responseLine.equalsIgnoreCase("GLOBAL_ABORT")==true )
            {
                break;
            }
        }
        closed=true;
    }
    catch (IOException e)
    {
        System.err.println("IOException:  " + e);
    }
}
}

```

Server

```

import java.io.*;
import java.net.*;
import java.util.*;

public class Server {
    boolean closed = false, inputFromAll = false;
    List<ClientThread> thread;
    List<String> data;
    List<String> decision;

    Server() {

```

```

        thread = new ArrayList<ClientThread>();
        data = new ArrayList<String>();
        decision = new ArrayList<String>();
    }

    public static void main(String args[])
    {
        Socket clientSocket = null;
        ServerSocket serverSocket = null;
        int port_number = 1111;
        Server server = new Server();
        try
        {
            serverSocket = new ServerSocket(port_number);
        } catch (IOException e) {
            System.out.println(e);
        }
        while (!server.closed)
        {
            try {
                clientSocket = serverSocket.accept();
                ClientThread clientThread = new ClientThread(server,
clientSocket);

                (server.thread).add(clientThread);
                System.out.println("\nNow Total clients are : " +
(server.thread).size());
                (server.data).add("NOT_SENT");
                (server.decision).add("NOT_SENT");
                clientThread.start();
            } catch (IOException e) { }
        }
        try {

            serverSocket.close();
        } catch (Exception e1) { }
    }
}

class ClientThread extends Thread
{
    DataInputStream is = null;
    String line;
    String destClient = "";

```

```

String name;
PrintStream os = null;
Socket clientSocket = null;
String clientIdentity;
Server server;

public ClientThread(Server server, Socket clientSocket)
{
    this.clientSocket = clientSocket;
    this.server = server;
}

@SuppressWarnings("deprecation")
public void run()
{
    try {
        is = new DataInputStream(clientSocket.getInputStream());
        os = new PrintStream(clientSocket.getOutputStream());
        os.println("Enter your name.");
        name = is.readLine();
        clientIdentity = name;
        os.println("Welcome " + name + " to this 2 Phase
Application.\nYou will receive a vote Request now...");
        os.println("Send Ready or Not Ready after local
transaction..");
        while (true)
        {
            line = is.readLine();
            if (line.equalsIgnoreCase("NOT READY"))
            {
                System.out.println("\nFrom '" + clientIdentity
+ "' : NOT READY\n\nSince NOT READY we will
not wait for inputs from other clients.");
                System.out.println("\nAborted....");

                for (int i = 0; i < (server.thread).size(); i++) {
                    ((server.thread).get(i)).os.println("GLOBAL_ABORT"
);
                    ((server.thread).get(i)).os.close();
                    ((server.thread).get(i)).is.close();
                }
                break;
            }
        }
    }
}

```



```

        if (line.equalsIgnoreCase("READY"))
        {
            System.out.println("\nFrom '" + clientIdentity + "' :
READY");
            if ((server.thread).contains(this))
            {
                (server.data).set((server.thread).indexOf(this),
"READY");
                for (int j = 0; j < (server.data).size(); j++)
                {
                    if
(!(((server.data).get(j)).equalsIgnoreCase("NOT_SENT")))
                    {
                        server.inputFromAll = true;
                        continue;
                    }
                    else{
                        server.inputFromAll = false;
                        System.out.println("\nWaiting for inputs
from other clients.");
                        break;
                    }
                }
                if (server.inputFromAll)
                {
                    System.out.println("All Ready..");
                }
            }
        }

        os.println("VOTE_REQUEST\nPlease enter COMMIT or ABORT to
proceed : ");
        line = is.readLine();
        if (line.equalsIgnoreCase("ABORT"))
        {
            System.out.println("\nFrom '" + clientIdentity
+ "' : ABORT\n\nSince ABORT we will not wait
for inputs from other clients.");
            System.out.println("\nAborted....");

            for (int i = 0; i < (server.thread).size(); i++) {
                ((server.thread).get(i)).os.println("GLOBAL_ABORT"
);
            }
        }
    }
}

```

```

        ((server.thread).get(i)).os.close();
        ((server.thread).get(i)).is.close();
    }
    break;
}
if (line.equalsIgnoreCase("COMMIT")){
    System.out.println("\nFrom '" + clientIdentity + "' :
COMMIT");
    if ((server.thread).contains(this))
    {
        (server.decision).set((server.thread).indexOf(this
), "COMMIT");
        for (int j = 0; j < (server.decision).size(); j++)
        {
            if
(!(((server.decision).get(j)).equalsIgnoreCase("NOT_SENT")))
            {
                server.inputFromAll = true;
                continue;
            }
            else{
                server.inputFromAll = false;
                System.out.println("\nWaiting for inputs
from other clients.");
                break;
            }
        }
        if (server.inputFromAll){
            System.out.println("\n\nCommitted....");
            for (int i = 0; i < (server.thread).size();
i++)
            {
                ((server.thread).get(i)).os.println("GLOBA
L_COMMIT");

                ((server.thread).get(i)).os.close();
                ((server.thread).get(i)).is.close();
            }
            break;
        }
    }
}
server.closed = true;

```



```

        clientSocket.close();
    } catch (IOException e) { }

}
}

```

Output:

Scenario: One "not ready"

```

C:\Windows\System32\cmd.e x + v
Now Total clients are : 2
Now Total clients are : 3
From 'Rishabh' : READY
Waiting for inputs from other clients.
From 'Abhishek' : READY
Waiting for inputs from other clients.
From 'Yash' : NOT READY
Since NOT READY we will not wait for inputs from other clients.
Aborted....

C:\Windows\System32\cmd.e x + v
C:\Users\Admin\OneDrive\Desktop\2phasescommit>java Client
Enter your name.
Rishabh
Welcome Rishabh to this 2 Phase Application.
You will receive a vote Request now...
Send Ready or Not Ready after local transaction..
Ready
VOTE_REQUEST
Please enter COMMIT or ABORT to proceed :
GLOBAL_ABORT

C:\Windows\System32\cmd.e x + v
C:\Users\Admin\OneDrive\Desktop\2phasescommit>java Client
Enter your name.
Abhishek
Welcome Abhishek to this 2 Phase Application.
You will receive a vote Request now...
Send Ready or Not Ready after local transaction..
Ready
VOTE_REQUEST
Please enter COMMIT or ABORT to proceed :
GLOBAL_ABORT

C:\Windows\System32\cmd.e x + v
C:\Users\Admin\OneDrive\Desktop\2phasescommit>java Client
Enter your name.
Yash
Welcome Yash to this 2 Phase Application.
You will receive a vote Request now...
Send Ready or Not Ready after local transaction..
Not ready
GLOBAL_ABORT

```

Scenario: all ready but one “Abort”

```
C:\Windows\System32\cmd.e x + v
From 'Yash' : READY
All Ready..

From 'Rishabh' : COMMIT
Waiting for inputs from other clients.

From 'Abhishek' : COMMIT
Waiting for inputs from other clients.

From 'Yash' : ABORT
Since ABORT we will not wait for inputs from other clients.
Aborted....
|

C:\Windows\System32\cmd.e x + v
Enter your name.
Rishabh

Welcome Rishabh to this 2 Phase Application.
You will receive a vote Request now...

Send Ready or Not Ready after local transaction..
Ready

VOTE_REQUEST

Please enter COMMIT or ABORT to proceed :
Commit

GLOBAL_ABORT
|

C:\Windows\System32\cmd.e x + v
Enter your name.
Abhishek

Welcome Abhishek to this 2 Phase Application.
You will receive a vote Request now...

Send Ready or Not Ready after local transaction..
Ready

VOTE_REQUEST

Please enter COMMIT or ABORT to proceed :
Commit

GLOBAL_ABORT
|

C:\Windows\System32\cmd.e x + v
Enter your name.
Yash

Welcome Yash to this 2 Phase Application.
You will receive a vote Request now...

Send Ready or Not Ready after local transaction..
Ready

VOTE_REQUEST

Please enter COMMIT or ABORT to proceed :
Abort

GLOBAL_ABORT
|
```

Scenario: all ready and all commit

```
C:\Windows\System32\cmd.e x + v
Waiting for inputs from other clients.

From 'Yash' : READY
All Ready..

From 'Rishabh' : COMMIT
Waiting for inputs from other clients.

From 'Abhishek' : COMMIT
Waiting for inputs from other clients.

From 'Yash' : COMMIT
Committed....
|

C:\Windows\System32\cmd.e x + v
Enter your name.
Rishabh

Welcome Rishabh to this 2 Phase Application.
You will receive a vote Request now...

Send Ready or Not Ready after local transaction..
Ready

VOTE_REQUEST

Please enter COMMIT or ABORT to proceed :
Commit

GLOBAL_COMMIT
|

C:\Windows\System32\cmd.e x + v
Enter your name.
Abhishek

Welcome Abhishek to this 2 Phase Application.
You will receive a vote Request now...

Send Ready or Not Ready after local transaction..
Ready

VOTE_REQUEST

Please enter COMMIT or ABORT to proceed :
Commit

GLOBAL_COMMIT
|

C:\Windows\System32\cmd.e x + v
Enter your name.
Yash

Welcome Yash to this 2 Phase Application.
You will receive a vote Request now...

Send Ready or Not Ready after local transaction..
Ready

VOTE_REQUEST

Please enter COMMIT or ABORT to proceed :
Commit

GLOBAL_COMMIT
|
```

Conclusion:

Thus, we successfully studied and implemented 2PL protocol.