TASK 13 - Realtime Database Concepts and Implementation

TASK 12 - Timestamping Messages

TASK 14 - Handling a Closed Connection

NECESSARY KNOWLEDGE

- Subscription and Publishing logic.

I. AIM

The aim of this task is to understand the concepts behind a realtime database, then we need to implement the logic which controls the subscription of the **RT** to the **TP** and allows the **TP** to publish to the **RT**.

II. INFORMATION

On start the **RT** will subscribe to the **TP**. When it has done this the **TP** will reply with schema and sets up the tables **RT** side. Finally **RT** starts replaying the log from disk.

III. INSTRUCTIONS

You will need to edit tp.q and rt.q to complete this task. Locate the following lines in tp.q:

```
tp_sub:{[]

sub:{[h]

pub:{[t]
```

And the the following lines in rt.q:

```
sub_tp:{[tp]

init:{[]
```

It will be necessary to modify these functions such that on connection and publishing in the simulator causes subscribers to the **TP** to print "Hello World". Subscribers to the **TP** should be stored in the list *SUBS* and should all receive the message when the simulator publishes.

IV. TESTING

To test your code in one session start the tickerplant process:

```
q tp.q -p 5000 -tp_path /tmp
```

In a second session start the feedhandler:

```
q fh.q -p 4000 -tp localhost:5000
```

In a third session start the simulator:

```
q simu.q -fh localhost:4000 -data data/msgs
```

Then start three more sessions, each is a realtime database, with the following commands:

```
q rt.q -p 5001 -tp localhost:5000 -hdb /tmp/tick

q rt.q -p 5002 -tp localhost:5000 -hdb /tmp/tick

q rt.q -p 5003 -tp localhost:5000 -hdb /tmp/tick
```

In the simulator session type:

```
do[10;pub[]]
```

Now return to each of your realtime database sessions, each one should have the following:

```
"Hello World"
q)"Hello World"
"Hello World"
"Hello World"
"Hello World"
"Hello World"
"Hello World"
"Hello World"
"Hello World"
"Hello World"
"Hello World"
```
