README

Distributed Data Systems Project

Name	Roll Number
Yoogottam Khandelwal	2018101019

Application Description

A group chat application where users can create multiple groups with other users and send messages.

Features

- Set your status
- Check when your friends were last active on the application
- Create groups of users and send messages

Tables

1. `user`
2. `group`
3. `message`
4. `group_members`

Fragmentation Plan

`Message` (4 Fragments)

Messages will be derived horizontally fragmented. The parent fragment will be group. Since we expect the application to generate a lot of messages, we want this table to be horizontally fragmented to even out the load. Also, the application will want messages from the same group (when the user opens a group), so, it makes sense to keep all messages from the same group together.

`Group` (4 Fragments)

Group will be horizontally fragmented based on the group id (% 4). This allows me to derived horizontally fragment messages, the reasons for which are stated above.

`User` (3 Fragments)

User will be vertically fragmented.

- The columns [`username` , `last_seen`] will be needed for UI in the group chat (to show who sent the message and when they were last active)
- The columns [`name` , `status`] will be needed for viewing the profile of a user.
- The columns [`phone` , `email`] will be needed when someone wants to see additional details about the user.

The vertical fragmentation has been made carefully such that for a query, we'll only need to contact a single site. Different types of queries have been stored at different sites.

`GroupMembers` (1 Fragment)

No fragmentation plan, the full table will exist on a single site.

Allocation Plan

fragment	site
user_1	1
user_2	2
user_3	3
group_1	1
group_2	2
group_3	3
group_4	4
message_1	1
message_2	2
message_3	3
message_4	4
group_members	4