Database for Messaging System

Designed by:-

Shubham Rathee

Jain University, Banglore

Requirements of Messaging system: -

The main functionality of a messaging system on a device is to keep track of the messages sent and received and notifying the respective individuals. Our system will allow a person to send messages to a user or a user group. Each user is then checked with the necessary privileges. The system also stores the received messages and allows the receivers to respond to the messages. It notifies the user regarding the messages with a notification frequency which can be hourly, daily or weekly.

Entities and Relationships: -

The Most important Relation of all the relations is user relation.

1)User

User entity stores the details of users. It has following attributes.

User_id:-

Each user in the user entity is uniquely identified by the user_id and hence user_id acts a primary key.

first_name and last_name

It will store first name and last name of user respectively. Multiple users can have the same first name and last name. first_name and last_name can be combinely taken as composite key.

phone_numer:

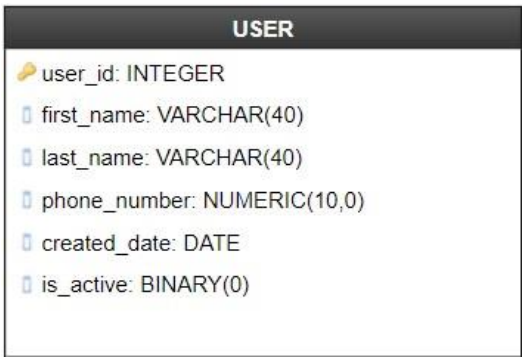
It will store the phone number of each user .

created_date

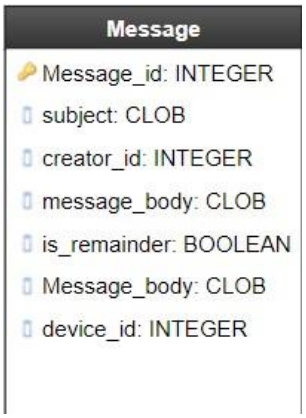
It will show the date on which user created account on platform. It will have data type as date. It can be timestamp also.

is_active:-

It will tell if a user is active or not .The is_active attribute is of domain Boolean and hence it can accept true or false only.



2)Message:



It is the next important entity from the database. It will store all the necessary information for given message. It will identify each message uniquely.

A message created and sent will result in the addition of a new tuple in the following message entity.

Message_id:-

It will be the primary key for given table as it will identify each message uniquely. It has a domain type integer. Message entity is very important and will be in relation with many other entities and hence this key will be a foreign key in many other entities.

Subject:-

It will store the subject of each message It is chosen to be of clob datatype.

Creator_id:-

It is id of user of who created the message. So, it is a foreign key referencing user entity.

Message_body:-

It will store the main message written by user. It will be the main part of the message. It will be of type clob.

Is_reminder:-

It is a Boolean type attribute. It is used to remind the user in case of an important message, else it will be false.

Device_id:-

Device_id will be foreign key and references device_details entity.

3)User_Contacts:-

The table 'user_contacts' gives information of contact of each user.

The information from this table can be updated time to time as user's phone number or contact id will change time to time. It will have attributes as follows:- **User_id:-**

It is foreign key for given table and along with contact_id works as a primary key as it will define each tuple uniquely .

Contact_id:-

This field contains the user_id of the contact of a particular user.

First_name:-

It stores first name of user. It will be of type varchar.

Last_name:-

It will describe last of each user. It will also of type varchar.

Creation_date:-

It stores date on which user has been added as the contact of the particular user. It will be of type DATE.

Modified_date:-

The last date on which user data is modified will be stored in the modified_date attribute.

Phone_number:

Phone_number hold the phone number of the contact.

4)Message Recipient:

It will store information about the receiver of message.

Message_id:-

Each message will have its recipient, so a tuple of message recipient entity can be uniquely identified using message_id. And this key reference message entity. This attribute in combination with other attributes can be used as a primary key.

Recipient_id:-

Recipient_id is nothing but user_id to which message is sent .

Is_read:-

It is a boolean value which tells if a message has been read or not. The value would be true if the message was read, else it would be false.

Sending Message to Groups:

5)User Group

This table will register the user against their groups. This table tells which user belong to which group and also if a user is active or not.

User_id:-

User_id will be foreign key for given table referencing the user table . Same user may be present in different groups and hence this cannot be a primary key.

Group_id:

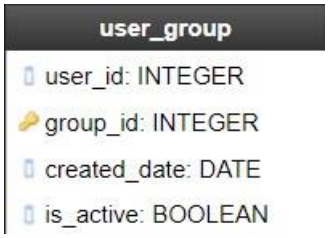
This attribute uniquely identifies the tuples in user_group along with user_id. Therefore, this combination could be used as a primary key.

Created_date:-

Created_date tells the date of creation of a particular group. Domain could be date or timestamp.

Is_active:-

Is_active will be Boolean data type which will provide the information about the user if the user is active or not in the group.



6)Group:

As messaging apps should have broadcast messaging service i.e. a particular message can be shared with particular group of people. So, group entity stores such information.

Group_id:-

Each group should be identified uniquely. So, Group_id attribute functions as a primary key for group table.

Group_name:-

Every group will have it's name to be identified by the user as searching on group_id could be a difficult task. A user cannot belong to the same group twice.

Created_date:-

Created_date will store the date on which particular group created.

Is_active:-

By setting particular interval of time to check the activity of group, we can check wheter the group is active or not, Depending on which priority can be given to the group operations. It is of type boolean.

7)Storage_Constraints

Since, storage of messages is important, we have included an entity for storage purpose. Storage constraints table stores information about memory used by message and free memory available in storage for every message of the user. The attributes are as follows:

User_id:-

It is foreign key referencing user table which tells us the owner of the message.

Message_id:-

It is foreign key referencing message table which gives unique message reference for each tuple.

Storage_name:-

Storage name stores the name of storage where message is stored.

Allocated:-

Allocated attribute tells the memory aquired by given message in storage.

Available:-

Available attribute describes remaining or free space in storage after some space aquired by message.

8)Deleted Message:

This table holds the details regarding the deleted messages. As deleted message body and subject will not be useful, we just save the traces of it with the use of Message_id. It has following attributes:

Message_id :-

It will store the id of deleted message. It will be the primary key for given message. It will be of integer type.

User_id:

It will tell us the owner of the message.

created_date:-

This date will have information on which user sent the message to the other user. It will of type date.

deletion_date:-

It will be having info on which user deleted the message from the system or his account. It will of type of date.

Blocking Messages:

This is very important because many a times we don't want to receive unnecessary messages. So, we add the particular sender in the blocklist so that he cannot further send any messages.

9)Permission entity:

It is the entity which stores the devices from which user logged in platform. So, it will store the list of devices from which user logged in.

It will have attribute as follows:-

User_id:-

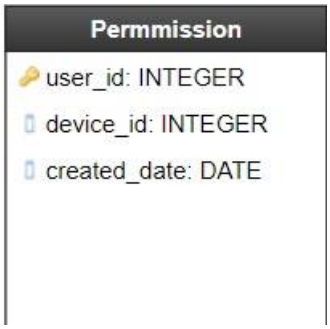
User_id references user table as foreign key and it will of domain integer.

Device_id:-

It will be foreign key referencing device _details table. user_id and device_id combinedly make primary key for given table.

Created_date:-

It will have the information about date on which user logged in from given device . It will of data type of date.



10)Block_list

It will have the id's that blocked by a particular user. By this, a user is refrained from sending messages. It has following attributes: -

user_id: -

It is the foreign key for given table and references user table. It signifies the details regarding the blocked or spam contacts. It will be of type integer.

device_id: -

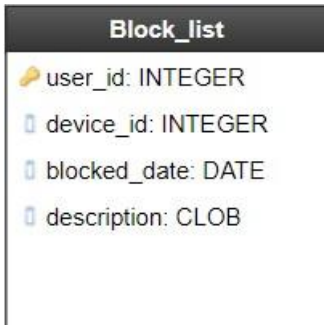
This will store details of the device of the blocked messenger. It will shoe device from which user is blocked it is the foreign key referencing device details.

blocked_date: -

As the name suggests, it will store the date on which the contact was blocked by the user. It is of type date.it can also be of type timestamp.

Description: -

This will store the reason for which the contact was blocked by user. Since, it is descriptive, it has domain of CLOB.



11)Device_details:-

The user can use multiple devices to send and read messages as each device will have different screen ratios and also different softwares to show messages. It is important to have the information of device from which user is currently logged in.

Device_id:-

Each device can be identified by unique id , so we can have device_id as the attribute which identifies each device uniquely and hence it is a primary key.

Device_name:-

Device will have their name or their company name so we should have attribute to store their name .The domain is varchar.

Processor:-

Device's processor info can be stored in processor attribute of type varchar.

Battery:-

Battery attribute will give information about battery details of the device.

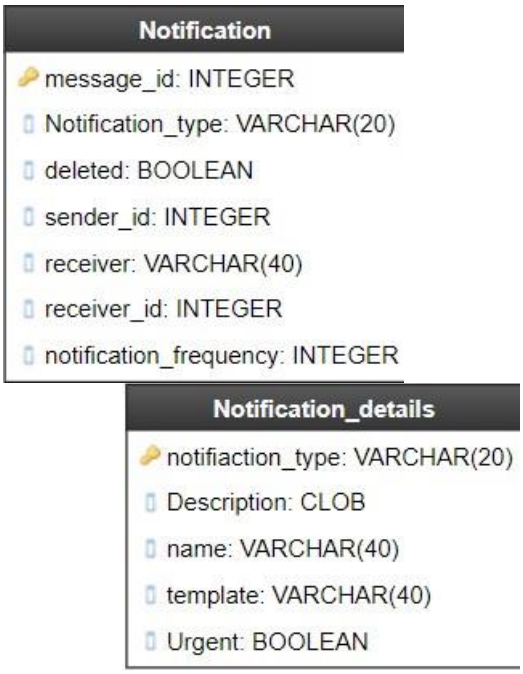
Reminding Mechanism

13) Notifications and Notification_details entities:-

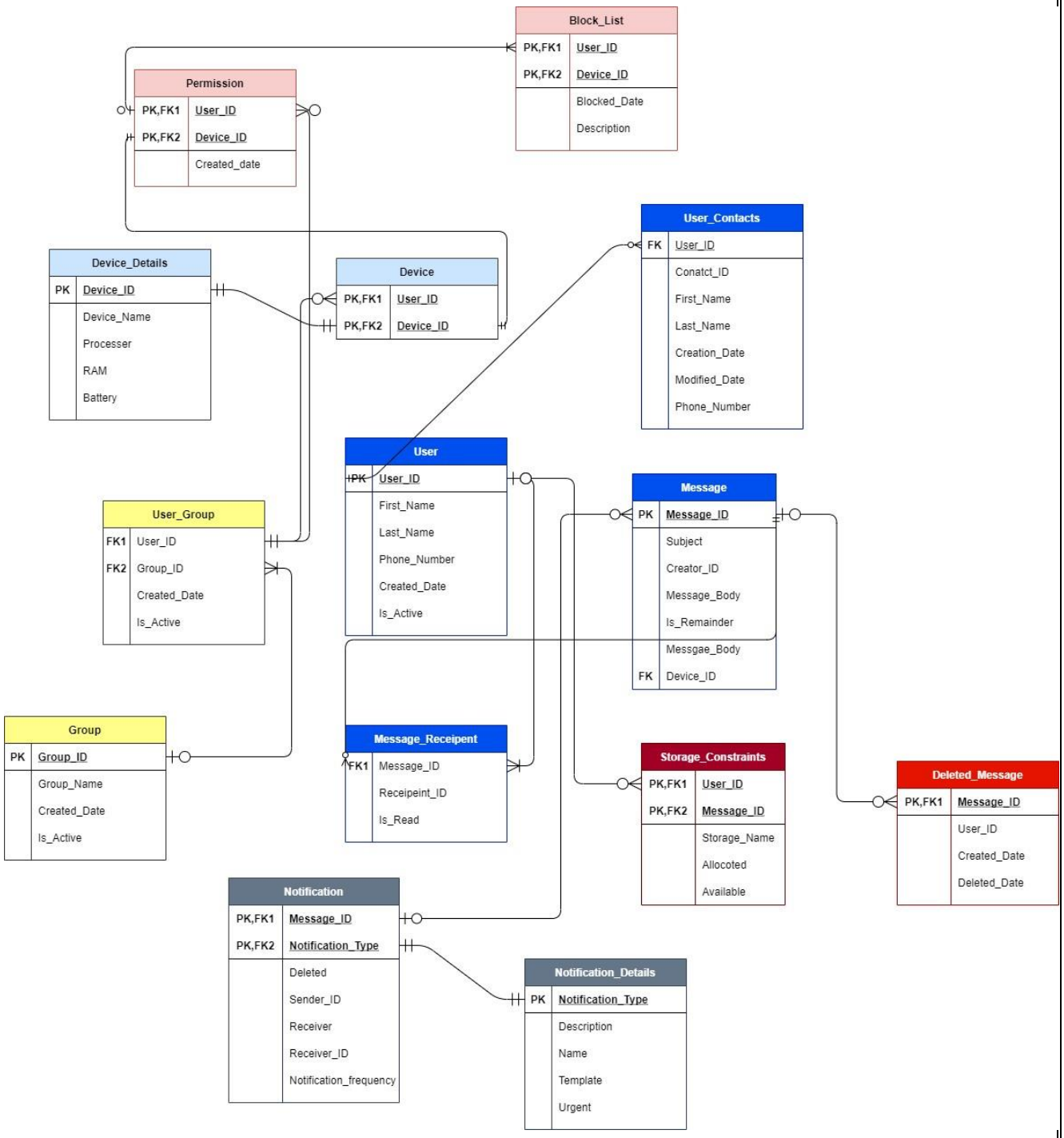
The messages received on a device are notified to the user with the help of notifications. We can do this by implementing the following two relations Notification and Notification_details.

Each message as mentioned above is identified with a unique message_id and each of these messages will have a notification type. Each Message has a

sender and a receiver. Each of the sender and receiver are identified with their ID's. There is also another attribute notification_frequency which tells us the frequency of the reminder (on daily or weekly basis). This table is connected with another table Notification_details where classifies messages according to different types. Since some of the messages may be urgent, special care is provided for them by including an attribute Urgent int the second table. It has a domain of Boolean. It is 1 if the message is urgent and 0 if the message is not urgent.

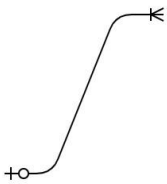


Final Model:-

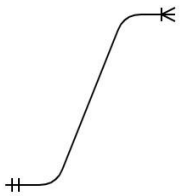


Different symbols used in the data model:-

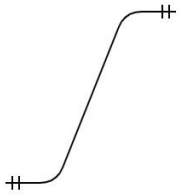
One optional to many mandatory



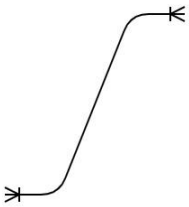
One mandatory to many optional



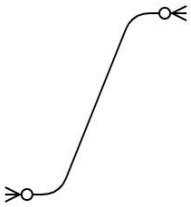
One to One (Complete participation)



Many mandatory to many mandatory



Many optional to many optional



Sql Code:

```
create table user_details(
user_id number,
first_name varchar2(40),
last_name varchar2(40),
phone_number number,
created_date date, is_active
```

```
binary_double, primary
key(user_id));

create table device_details(
device_id number,
device_name varchar2(20),
processor varchar2(20),
ram varchar2(20), battery
number, primary
key(device_id));

create table message(
message_id number, subject
clob, creator_id number,
message_body clob,
is_remainder binary_double,
device_id number, primary
key(message_id),
foreign key (device_id) references device_details, foreign key(creator_id)
references user_details);

create table user_contacts( user_id number,
contact_id number, first_name varchar2(20),
last_name varchar2(20), creation_date date,
modified_date date, phone_number number,
primary key(user_id,contact_id), foreign
key(user_id) references user_details);
```

```
create table message_receipent(
message_id number,
receipent_id number, is_read
binary_double, primary
key(message_id));


create table user_group(
user_id number, group__id
number, created_date
date, is_active
binary_double,
foreign key(user_id) references user_details); create table group_( group__id
number, group_name varchar2(40), created_date date, is_active
binary_double, primary key(group__id));

create table storage_constraints(
user_id number, message_id
number, storage_name
varchar2(20), allocated number,
availabe number,
foreign key(user_id) references user_details, foreign
key(message_id) references message);

create table deleted_message( message_id
number, user_id number, created_date date,
```

```
deleted_date date, primary key(message_id),
foreign key(message_id) references message,
foreign key(user_id) references user_details);

create table permission_( user_id
number, device_id number,
created_date date, primary
key(user_id,device_id), foreign
key(user_id) references
user_details, foreign
key(device_id) references
device_details);

create table block_list( user_id number, device_id
number, blocked_date date, description_ clob,
primary key(user_id,device_id), foreign
key(user_id) references user_details, foreign
key(device_id) references device_details);

create table notification_details(
notification_type varchar2(20),
description_ clob, name_
varchar2(40), template_
varchar2(40), urgent
binary_double, primary
key(notification_type));
```

```
create table notification_( message_id
number, notification_type
varchar2(20), deleted binary_double,
sender_id number, receiver
varchar2(40), reciever_id number,
notification_frequency number,
foreign key(message_id) references
message,
foreign key(notification_type) references notification_details);
```

Inserting tuples into the relations:

```
INSERT INTO USER_details VALUES(21, 'Rohit', 'Verma',
1234567890, '1-3-20', 1.0);
INSERT INTO USER_details VALUES(22, 'Sneha', 'Shukla',
2345678901, '12-3-20', 1.0);
INSERT INTO USER_details VALUES(23, 'Andy', 'Robbin', 3456789012,
'7-4-20', 0);
INSERT INTO USER_details VALUES(24, 'Tom', 'Stark', 4567890123,
'15-4-20', 1.0);
INSERT INTO USER_details VALUES(25, 'Abhijeet', 'Shah', 5678901234,
'24-7-20', 0);
INSERT INTO USER_details VALUES(26, 'Suresh', 'Kumar', 6789012345,
'10-8-20', 1.0)
```

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.005 seconds

	USER_ID	FIRST_NAME	LAST_NAME	PHONE_NUMBER	CREATED_DATE	IS_ACTIVE
1	21	Rohit	Verma	1234567890	01-03-20	1.0
2	22	Sneha	Shukla	2345678901	12-03-20	1.0
3	23	Andy	Robbin	3456789012	07-04-20	0.0
4	24	Tom	Stark	4567890123	15-04-20	1.0
5	25	Abhijeet	Shah	5678901234	24-07-20	0.0
6	26	Suresh	Kumar	6789012345	10-08-20	1.0

```
INSERT INTO MESSAGE VALUES(51, 'Wishes', 21, 'Happy
Birthday', 0, 71);
INSERT INTO MESSAGE VALUES(52, 'Request', 22, 'Please  extend
the date of Submission', 1.0, 72);
INSERT INTO MESSAGE VALUES(53, 'Reminder', 23, 'Hey u ve  exam
2day', 1.0, 73);
INSERT INTO MESSAGE VALUES(54, 'Demand', 24, 'can u send  money',
0, 74);
INSERT INTO MESSAGE VALUES(55, 'Notice', 25, 'Hostel should  be
vacated in 2 weeks', 1.0, 75);
INSERT INTO MESSAGE VALUES(56, 'Selection', 26, 'Hey u  selected
for interview', 1.0, 76);
```

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.045 seconds

	MESSAGE_ID	SUBJECT	CREAT...	MESSAGE_BODY	IS_REMAINDER	DEVICE_ID
1	51	Wishes	21	Happy Birthday	0.0	71
2	52	Request	22	Please extend the date of Submission	1.0	72
3	53	Reminder	23	Hey u ve exam 2day	1.0	73
4	54	Demand	24	can u send money	0.0	74
5	55	Notice	25	Hostel should be vacated in 2 weeks	1.0	75
6	56	Selection	26	Hey u selected for interview	1.0	76

```
INSERT INTO Device_details VALUES(71, 'Hp-Notebook',
```

'Intel I-5', 8, 30);

INSERT INTO Device_details VALUES(72, 'Android',

'Snapdragon 865', 4, 80);

INSERT INTO Device_details VALUES(73, 'Macbook', 'M1', 6,

60);

INSERT INTO Device_details VALUES(74, 'iPhone', 'A13

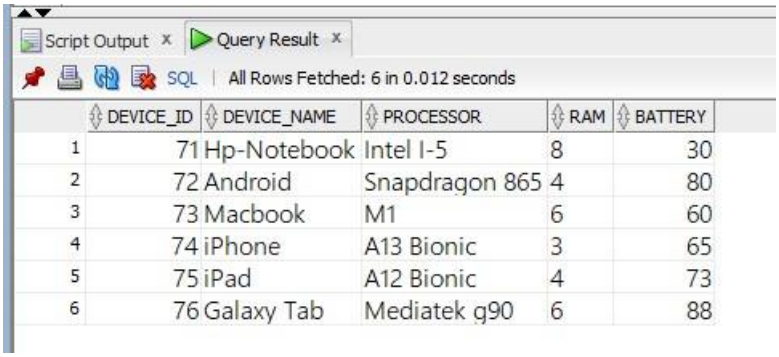
Bionic', 3, 65);

INSERT INTO Device_details VALUES(75, 'iPad', 'A12

Bionic', 4, 73);

INSERT INTO Device_details VALUES(76, 'Galaxy Tab',

'Mediatek g90', 6, 88);


Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.012 seconds

	DEVICE_ID	DEVICE_NAME	PROCESSOR	RAM	BATTERY
1	71	Hp-Notebook	Intel I-5	8	30
2	72	Android	Snapdragon 865	4	80
3	73	Macbook	M1	6	60
4	74	iPhone	A13 Bionic	3	65
5	75	iPad	A12 Bionic	4	73
6	76	Galaxy Tab	Mediatek g90	6	88

INSERT INTO user_contacts VALUES(21, 24, 'Tom', 'Stark',

'11-4-20', '12-6-20', 4567890123);INSERT INTO user_contacts VALUES(21, 25, 'Abhijeet', 'Shah',

'1-3-20', '12-3-20', 5678901234);

INSERT INTO user_contacts VALUES(21, 23, 'Andy', 'Robbin',

'12-3-20', '7-4-20', 6789012345);

INSERT INTO user_contacts VALUES(22, 21, 'Rohit', 'Verma',

'7-4-20', '11-4-20', 1234567890);

INSERT INTO user_contacts VALUES(22, 24, 'Tom', 'Stark',

'11-4-20', '4-7-20', 4567890123);

INSERT INTO user_contacts VALUES(23 , 24, 'Tom', 'Stark',

'4-7-20', '10-8-20', 4567890123);

INSERT INTO user_contacts VALUES(23, 25, 'Abhijeet', 'Shah',

'10-8-20', '1-9-20', 5678901234);

INSERT INTO user_contacts VALUES(24, 21, 'Rohit', 'Verma',

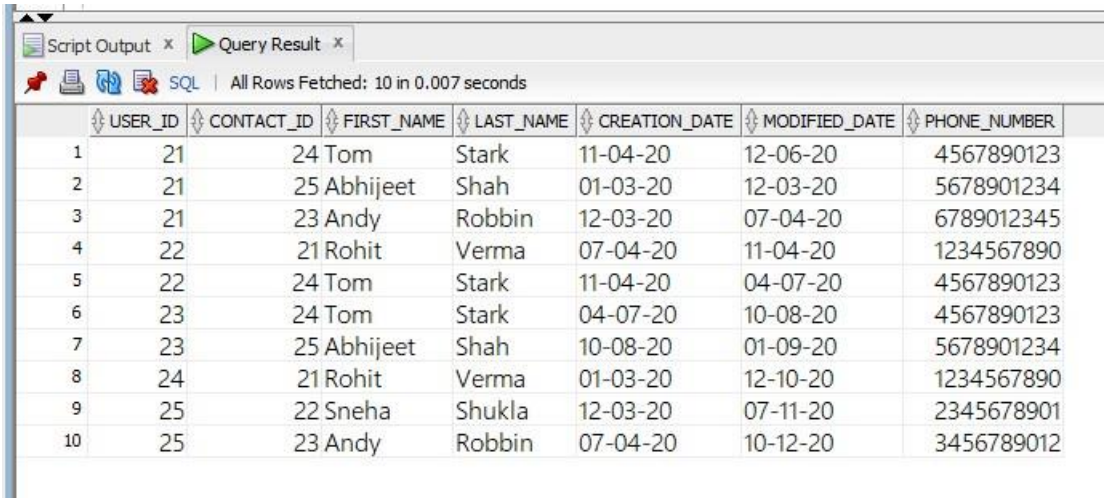
'1-3-20', '12-10-20', 1234567890);

INSERT INTO user_contacts VALUES(25, 22, 'Sneha', 'Shukla',

'12-3-20', '7-11-20', 2345678901);

INSERT INTO user_contacts VALUES(25, 23, 'Andy', 'Robbin',

'7-4-20', '10-12-20', 3456789012);


Script Output × Query Result ×

SQL | All Rows Fetched: 10 in 0.007 seconds

	USER_ID	CONTACT_ID	FIRST_NAME	LAST_NAME	CREATION_DATE	MODIFIED_DATE	PHONE_NUMBER
1	21	24	Tom	Stark	11-04-20	12-06-20	4567890123
2	21	25	Abhijeet	Shah	01-03-20	12-03-20	5678901234
3	21	23	Andy	Robbin	12-03-20	07-04-20	6789012345
4	22	21	Rohit	Verma	07-04-20	11-04-20	1234567890
5	22	24	Tom	Stark	11-04-20	04-07-20	4567890123
6	23	24	Tom	Stark	04-07-20	10-08-20	4567890123
7	23	25	Abhijeet	Shah	10-08-20	01-09-20	5678901234
8	24	21	Rohit	Verma	01-03-20	12-10-20	1234567890
9	25	22	Sneha	Shukla	12-03-20	07-11-20	2345678901
10	25	23	Andy	Robbin	07-04-20	10-12-20	3456789012

INSERT INTO message_receipent VALUES(51, 25, 1.0);

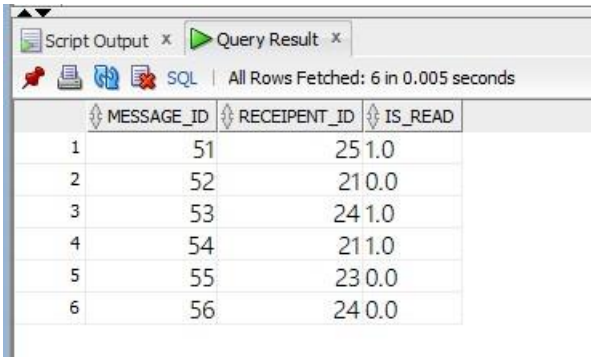
INSERT INTO message_receipent VALUES(52, 21, 0);

INSERT INTO message_receipent VALUES(53, 24, 1.0);

INSERT INTO message_receipent VALUES(54, 21, 1.0);

INSERT INTO message_receipent VALUES(55, 23, 0); INSERT

INTO message_receipent VALUES(56, 24, 0);


Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.005 seconds

	MESSAGE_ID	RECEIPENT_ID	IS_READ
1	51	25	1.0
2	52	21	0.0
3	53	24	1.0
4	54	21	1.0
5	55	23	0.0
6	56	24	0.0

INSERT INTO user_group VALUES(21, 1, '10-8-20', 1.0);

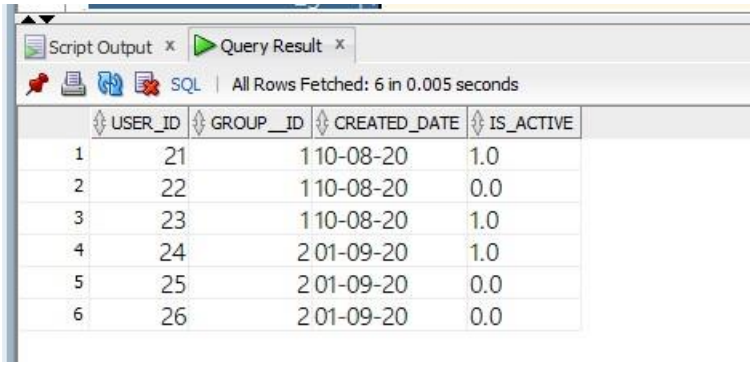
INSERT INTO user_group VALUES(22, 1, '10-8-20', 0);

INSERT INTO user_group VALUES(23, 1, '10-8-20', 1.0);

INSERT INTO user_group VALUES(24, 2, '1-9-20',1.0);

INSERT INTO user_group VALUES(25, 2, '1-9-20', 0);

INSERT INTO user_group VALUES(26, 2, '1-9-20', 0);


Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.005 seconds

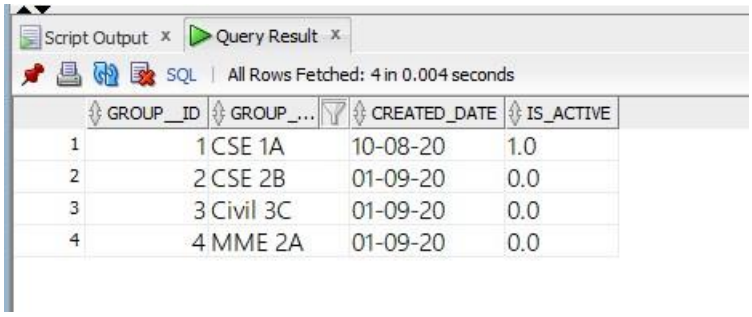
	USER_ID	GROUP__ID	CREATED_DATE	IS_ACTIVE
1	21	1	10-08-20	1.0
2	22	1	10-08-20	0.0
3	23	1	10-08-20	1.0
4	24	2	01-09-20	1.0
5	25	2	01-09-20	0.0
6	26	2	01-09-20	0.0

INSERT INTO Group_ VALUES(1, 'CSE 1A', '10-8-20', 1.0);

INSERT INTO Group_ VALUES(2, 'CSE 2B', '1-9-20', 0);

INSERT INTO Group_ VALUES(3, 'Civil 3C', '1-9-20', 0);

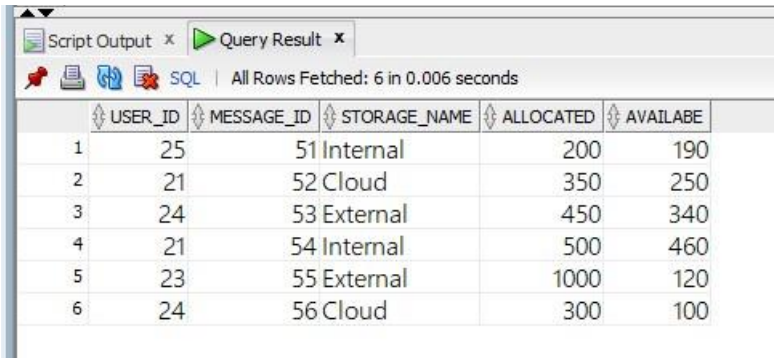
INSERT INTO Group_ VALUES(4, 'MME 2A', '1-9-20', 0);


Script Output | Query Result

SQL | All Rows Fetched: 4 in 0.004 seconds

	GROUP__ID	GROUP_...	CREATED_DATE	IS_ACTIVE
1	1	CSE 1A	10-08-20	1.0
2	2	CSE 2B	01-09-20	0.0
3	3	Civil 3C	01-09-20	0.0
4	4	MME 2A	01-09-20	0.0

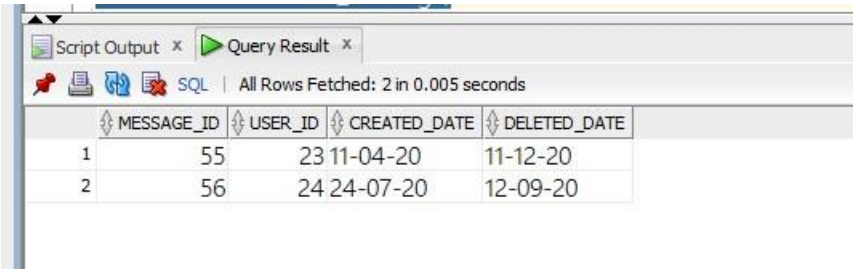
```
INSERT INTO Storage_constraints VALUES(25, 51, 'Internal',
200 , 190 );
INSERT INTO Storage_constraints VALUES(21, 52, 'Cloud',
350 , 250 );
INSERT INTO Storage_constraints VALUES(24, 53,
'External',450, 340);
INSERT INTO Storage_constraints VALUES(21, 54, 'Internal',
500, 460);
INSERT INTO Storage_constraints VALUES(23, 55, 'External',
1000, 120);
INSERT INTO Storage_constraints VALUES(24, 56,'Cloud',  300,
100);
```


Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.006 seconds

	USER_ID	MESSAGE_ID	STORAGE_NAME	ALLOCATED	AVAILABE
1	25	51	Internal	200	190
2	21	52	Cloud	350	250
3	24	53	External	450	340
4	21	54	Internal	500	460
5	23	55	External	1000	120
6	24	56	Cloud	300	100

```
INSERT INTO deleted_message VALUES(55, 23, '11-4-20',
'11-12-20');
INSERT INTO deleted_message VALUES(56, 24, '24-7-20',  '12-9-20');
```


Script Output | Query Result | SQL | All Rows Fetched: 2 in 0.005 seconds

	MESSAGE_ID	USER_ID	CREATED_DATE	DELETED_DATE
1	55	23	11-04-20	11-12-20
2	56	24	24-07-20	12-09-20

INSERT INTO Permission_ VALUES(21, 71, '11-4-20');

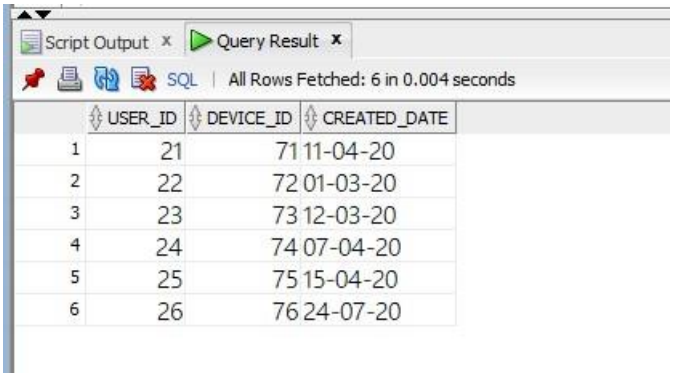
INSERT INTO Permission_ VALUES(22, 72, '1-3-20');

INSERT INTO Permission_ VALUES(23, 73, '12-3-20');

INSERT INTO Permission_ VALUES(24, 74, '7-4-20');

INSERT INTO Permission_ VALUES(25, 75, '15-4-20');

INSERT INTO Permission_ VALUES(26, 76, '24-7-20');


Script Output | Query Result | SQL | All Rows Fetched: 6 in 0.004 seconds

	USER_ID	DEVICE_ID	CREATED_DATE
1	21	71	11-04-20
2	22	72	01-03-20
3	23	73	12-03-20
4	24	74	07-04-20
5	25	75	15-04-20
6	26	76	24-07-20

INSERT INTO Block_list VALUES(21, 71, '11-4-20', 'Spam');

INSERT INTO Block_list VALUES(22, 72, '1-3-20', 'Advertisements');

INSERT INTO Block_list VALUES(23, 73, '12-3-20', 'Fraud');

INSERT INTO Block_list VALUES(24, 74, '7-4-20', 'Ex');

INSERT INTO Block_list VALUES(25, 75, '15-4-20', 'Threat'); INSERT INTO Block_list VALUES(26, 76, '24-4-20', 'Clash');

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.006 seconds

	USER_ID	DEVICE_ID	BLOCKED_DATE	DESCRIPTION_
1	21	71	11-04-20	Spam
2	22	72	01-03-20	Advertisements
3	23	73	12-03-20	Fraud
4	24	74	07-04-20	Ex
5	25	75	15-04-20	Threat
6	26	76	24-04-20	Clash

```
INSERT INTO Notification_ VALUES(51, 'pop-up', 0, 21,
'Abhijeet',25,5);
INSERT INTO Notification_ VALUES(51, 'Ring', 0, 22,
'Rohit',21,3);
INSERT INTO Notification_ VALUES(51, 'silent', 0, 23,
'Tom',24,0);
INSERT INTO Notification_ VALUES(51, 'alert sound', 0, 24,
'Rohit',21,6);
INSERT INTO Notification_ VALUES(51, 'No notification',
1.0, 25, 'Andy',23,0);
INSERT INTO Notification_ VALUES(51, 'pop-up', 1.0, 26,  'Suresh',26,2);
```

Script Output | Query Result

SQL | All Rows Fetched: 6 in 0.006 seconds

	MESSAGE_ID	NOTIFICATION_TYPE	DELETED	SENDER_ID	RECEIVER	RECIEVER_ID	NOTIFICATION_FREQUENCY
1	51	pop-up	0.0	21	Abhijeet	25	5
2	51	silent	0.0	23	Tom	24	0
3	51	alert sound	0.0	24	Rohit	21	6
4	51	pop-up	1.0	26	Suresh	26	2
5	51	Ring	0.0	22	Rohit	21	3
6	51	No notification	1.0	25	Andy	23	0

```
INSERT INTO Notification_Details VALUES('pop-up', 'Portion  of
message', 'Popping', 'Message with Rings', 1.0);
INSERT INTO Notification_Details VALUES('Ring', 'Name of
```

Sender', 'Ringing', 'Just rings without screen on', 1.0);

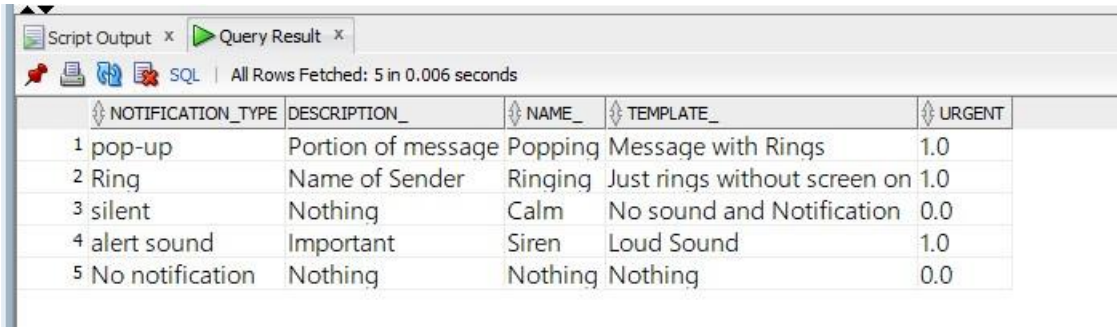
INSERT INTO Notification_Details VALUES('silent',

'Nothing', 'Calm', 'No sound and Notification', 0);

INSERT INTO Notification_Details VALUES('alert sound',

'Important', 'Siren', 'Loud Sound', 1.0);

INSERT INTO Notification_Details VALUES('No notification','Nothing', 'Nothing', 'Nothing', 0);


Script Output | Query Result

SQL | All Rows Fetched: 5 in 0.006 seconds

	NOTIFICATION_TYPE	DESCRIPTION_	NAME_	TEMPLATE_	URGENT
1	pop-up	Portion of message	Popping	Message with Rings	1.0
2	Ring	Name of Sender	Ringing	Just rings without screen on	1.0
3	silent	Nothing	Calm	No sound and Notification	0.0
4	alert sound	Important	Siren	Loud Sound	1.0
5	No notification	Nothing	Nothing	Nothing	0.0

Conclusion:-

We have intended to built a fully functional messaging model which can fit into a variety of modern day system to send messages/notifications.