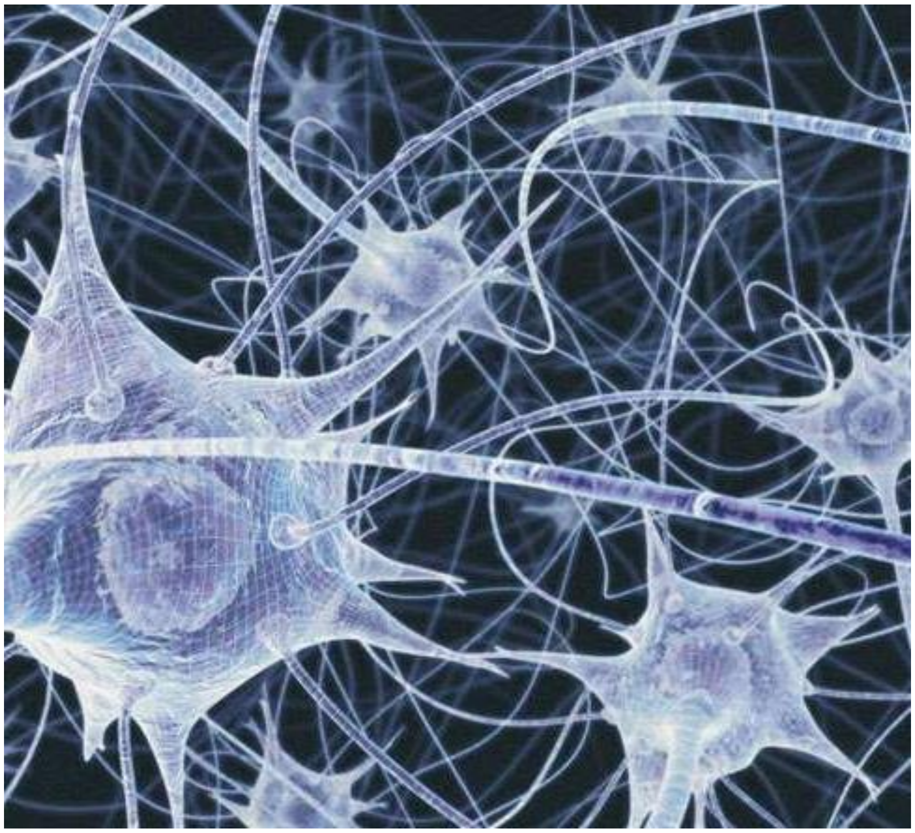
MIND MODELLING



- Mandalapu Venkata Lakshmi Mahitha
- Venkatesh Kumar
- Aisswarya

Table of Contents

1. Abstract: *3*

2. Problem Statement: *3*

3. Introduction: *3*

4. Novelty of our Project(Approach/Solution) *3*

5. Collecting Emotion *4*

6. Connectome - Neural Dust *8*

7. Retriggering the emotion - Optogenetics *12*

8. Visualization of the Database *12*

8.1. Entity Relationship Diagram 13

8.2. Activity Diagram 13

8.3. Use Case Diagram 15

8.4. Visual representation of Database Implementation 15

7. Tables created in the Database *16*

8. Functions *18*

9. Views *26*

10. Stored Procedures *28*

11. Triggers *29*

12. Privileges For Users *31*

17. Schedulers *34*

18. Conclusion *36*

1. Abstract:

As the evolution progresses human race tries to learn the theory of nature and achieve their goals by bending the science. One such example of nature is emotions and memories that happen in Human Brain. Human Brain is said to have 85 Billion of Neurons which are responsible for human memory and human activities. Although, human can remember their past and cherish about the things happened, still they miss to feel the exact amount of emotions they went through during that time-period. We are proposing a model which could capture and store the neuron connections happening at a particular time-period which are responsible for memories and emotions and retrigger them when the person want to recollect and relive the moment he went through.

2. Problem Statement:

Emotions such as Happy, Sad, Fear, Anger are some special occasions that the human store in their brain and cherish about the moments later. Though it could be remembered, it is hard to feel the same way it happened. For example, occurrence of a first kiss, a lady giving birth to her child, first break-up or even the moment when you are pushed hard and the rage that outburst. These are some special moments that people think about it, and cherish thought their life time.

3. Introduction:

Human Brain is a complex and huge storage system that stores the data in neurons. While many are trying to understand the neuron connections by anatomy or scan some set of researcher studies mouse brain and they retrigged the fear using Optogenetics. Recollecting the events happened in our life is a big task and not everyone will be able to do the task perfectly. Even though when someone could recollect the events and happenings, still they will miss to feel the same way how they felt at that point of time.

4. Novelty of our Project (Approach/Solution)

In our system to recollect and re-live the moments the user is fitted with emotion detection devices which captures the real time emotion the user feels. The neuron connections happening during the time is detected by the neural dust that are injected in the human brain. They track the neurons which gets activated and saves the collection of neurons in the database. Whenever the user wishes to feel and re-live those moments, he has to select the time he wish to feel back, the system will pass the connectome to the optogenetics device which in turn triggers those neurons which has details of those memory.

5. Collecting Emotion

Detecting emotions is a tricky and confusing part. Some person is highly expressive while some is less expressive. Hence detecting emotions based on physical emotion evidence is tedious we took the non-conventional way to detect the human emotion. The devices such as EEG, ECG, GSR, EMG analyses the biological changes happening in the body and hence it detects accurate emotions a person is going through. Instead of using any one of the devices mentioned above, we are calculating the output of all those devices to increase the accuracy of emotion detection.

EEG (electroencephalogram**):**

It is a test to record electrical signals happening in the brain. The electrical signals are recorded in terms of amplitude and frequency. Electrodes are placed on the scalp of the human head so that the device can measure the electrical signals happening in the brain. In our system the electrodes are placed in such a way that asymmetry of electrode in different brain regions can be calculated in ordet to detect the human emotions.

The EEG device values cannot be directly used it has to be preprocessed where the amplitude and frequency are increased, and noise is removed. The resulting data is then used for further calculations.

In this method of calculating the asymmetry of electrodes two electrodes are placed on the brain lobe, one on left brain and one on right brain. Then the ration of two electrodes are calculated which gives the alpha symmetry value of the lobe. By calculating for different brain lobe, we could find the percentage of match with the predefined set and judge whether it’s a match for emotion.

ECG (Electrocardiogram):

Electrocardiogram (ECG) signals are among the most important sources of diagnostic in healthcare. The ECG device output signal waves is standardly classified as PQRST interval which could be used to recognize the emotions a person is going through. There are several techniques to find the emotions based on ECG signals. In our system we are applying the EMD (Empirical Model Decomposition) technique which extracts the ECG features from an instantaneous frequency and local oscillation of Intrinsic Model Function(IMF).

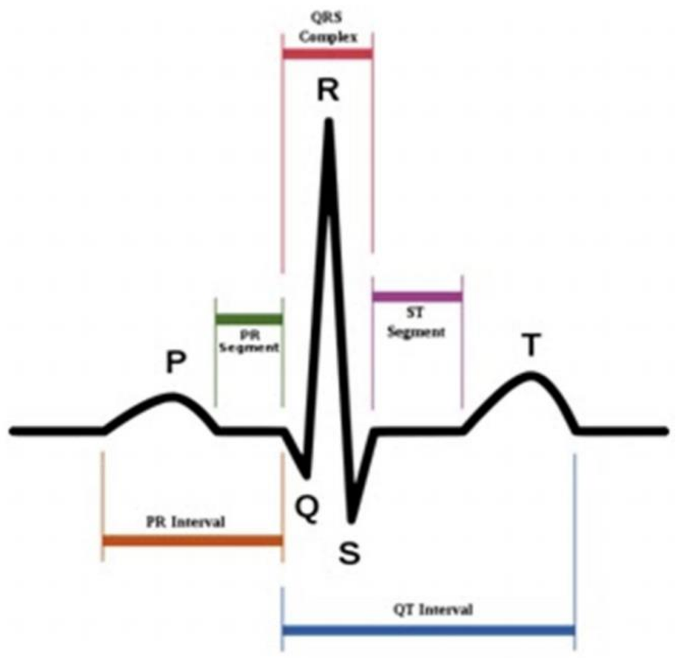


Fig. 1 Normal persons ECG signal

Methodology of ECG:
The proposed framework is comprised of four independent steps as shown in the fig. 2.
1. ECG synthesis, wherein an ECG signal x(t) is generated. From the data base the ECG data's were collected. From the ECG raw data's, the synthetic ECG signal was generated.
2. The fast fourier transform can be used to remove the noise from the synthetic ECG signal for accurate emotion detection.
3. Estimation of the oscillatory modes, called Intrinsic Mode Functions (IMF) which can be found by using EMD algorithm.
4. Extraction of features associated with the instantaneous frequency and the local oscillation of the IMFs and classification among predefined affect states.

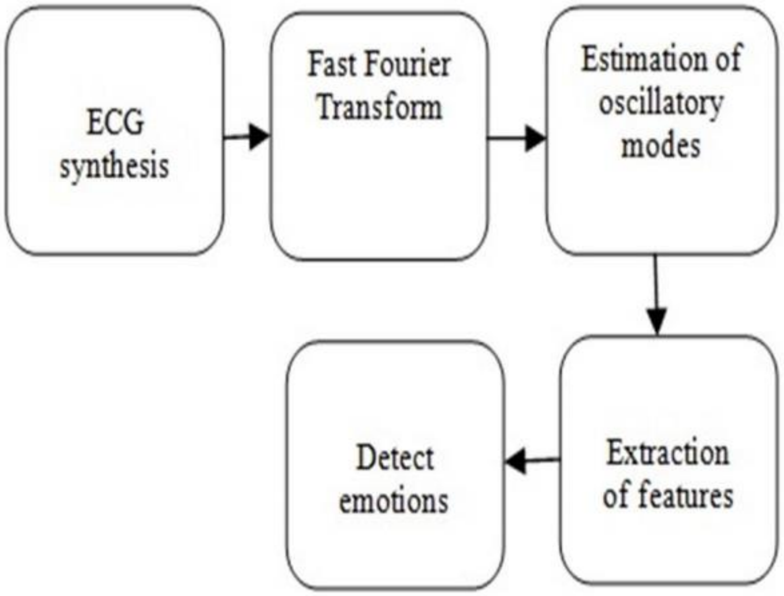


Fig.2 Block diagram of proposed work

Results:

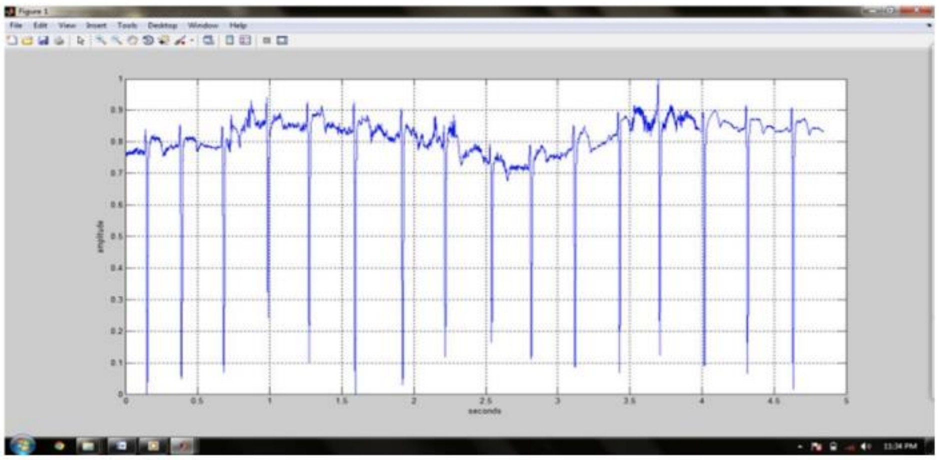


Fig. 3 Generated synthetic ECG signal.

From the synthetic ECG signal the noise are removed by using the fast fourier transform technique. The noise removed ECG signal is shown in the Fig. 4

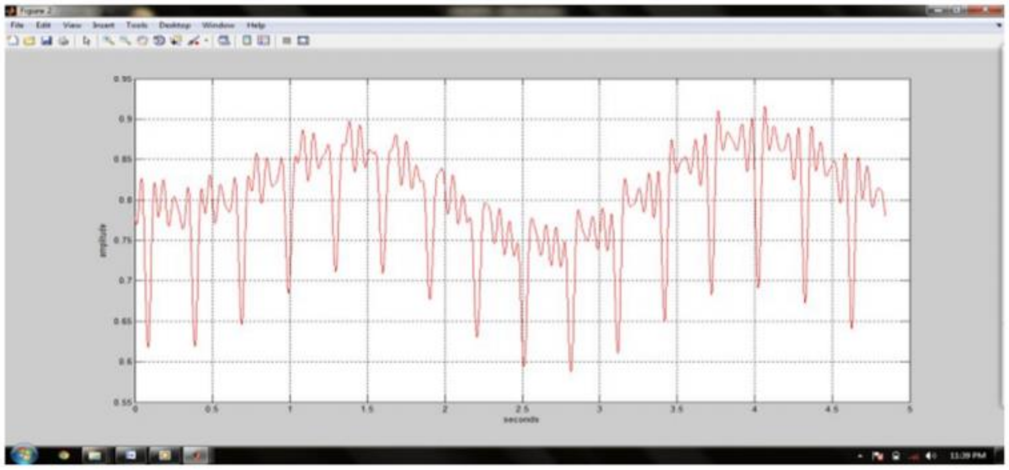


Fig. 4 Noise removed ECG signal

After noise removing process, the noise removed signal is converted into raw data's, and it is given to the EMD algorithm to generate the IMF signals.

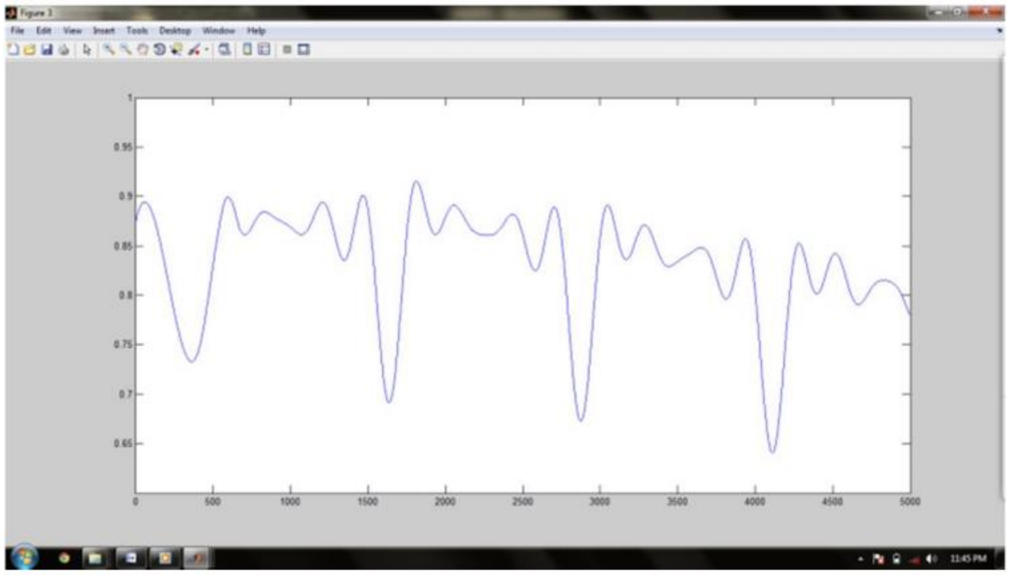


Fig. 5 First IMF signal

From the IMF signals we will consider any one of the first three IMF signals, because other IMF signals have less oscillatory activity, they will not prevent the instantaneous frequency and local oscillation activity.

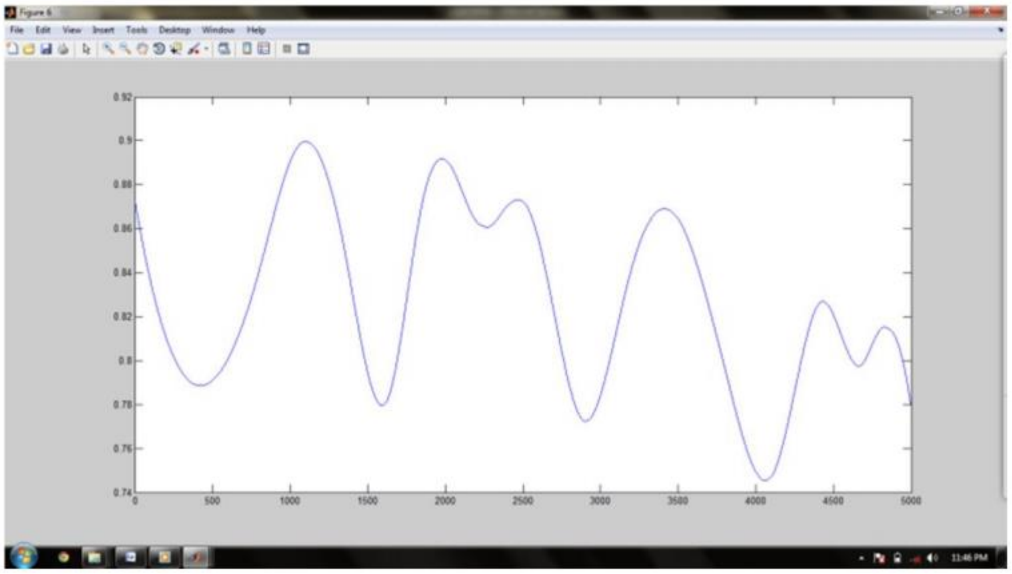


Fig. 6 Second IMF signal

From the IMF signal we want to find the amplitude and instantaneous frequency. Based on the frequency and amplitude we can classify the emotion of human beings.

For the joy persons ECG signal the instantaneous frequency is between (10 Hz-40Hz) and the amplitude is in between (0.2mv-0.25mv). When the amplitude and instantaneous frequency fall above and below the particular limit, it shows the fear, angry and sad emotional states. For the sad emotional state the instantaneous frequency is below (10-40) Hz, and the amplitude is below (0.2-0.25) mv, and for the anger emotional state the instantaneous frequency and the amplitude is beyond the particular value, ie., the instantaneous frequency is above (40-100) Hz and the amplitude is above 1.5 mv. For the fear emotional state, the instantaneous frequency is in between (40-100) Hz and the amplitude is, between the limit (0.3-1.5) mv.

Emotional States	**Instantaneous Frequency (Hz)**	**Amplitude (mv)**
Joy	(10-40)	(0.2-0.25)
Sad	Below 10	Below 0.2
Fear	(40-100)	(0.3-1.5)
Anger	Beyond 100	Above 1.5

These result values are used for predefined set of ECG.

6. Connectome - Neural Dust

A connectome is a comprehensive map of neural connections in the brain, and may be thought of as its "wiring diagram". More broadly, a connectome would include the mapping of all neural connections within an organism's nervous system. The production and study of connectomes, known as connectomics, may range in scale from a detailed map of the full set of neurons and synapses within part or all of the nervous system of an organism to a macro scale description of the functional and structural connectivity between all cortical areas and subcortical structures. The term "connectome" is used primarily in scientific efforts to capture, map, and understand the organization of neural interactions within the brain. Research has successfully constructed the full connectome of one animal: the roundworm Caenorhabditis elegans. Partial connectomes of a mouse retina and mouse primary visual cortex have also been successfully constructed.

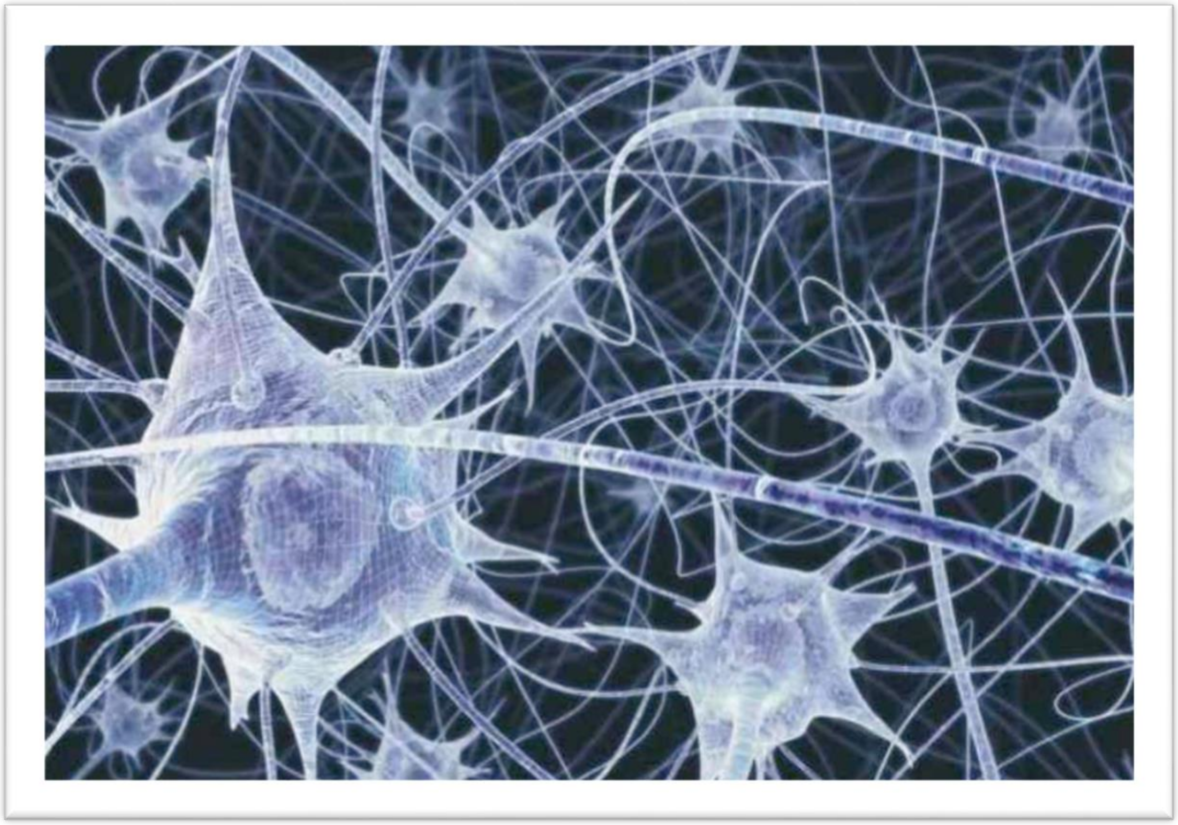


Figure 1 Neural Nework

Neural dust is a wireless, battery less, ultrasonic neural interface platform. Ultrasound enables wireless power and communication with sub-mm implants. The neural dust platform uses ultrasonic power and communication to enable a scalable, wireless, and battery less system for interfacing with the nervous system. Ultrasound offers several advantages over alternative wireless approaches, including a safe method for powering and communicating with sub mm-sized devices implanted deep in tissue. Early studies demonstrated that neural dust motes could wirelessly transmit high-fidelity electrophysiological data in vivo, and that theoretically, this system could be miniaturized well below the mm-scale. Future developments are focused on further minimization of the platform, better encapsulation methods as a path towards truly chronic neural interfaces, improved delivery mechanisms, stimulation capabilities, and finally refinements to enable deployment of neural dust in the central nervous system.

Figure 2 Neural Dust

Our project assumes that the connectome of every action of the brain is clearly recorded using the neural dust technology. The process can be divided conceptually into two parts: recording and reporting. Recording involves sensing and coding for transmission neural activity including membrane potentials, protein expression levels, calcium concentrations and their correlates. Reporting involves conveying the coded information from the locus of the recording typically deep within the neural tissue of an awake subject to some external computing or storage device.

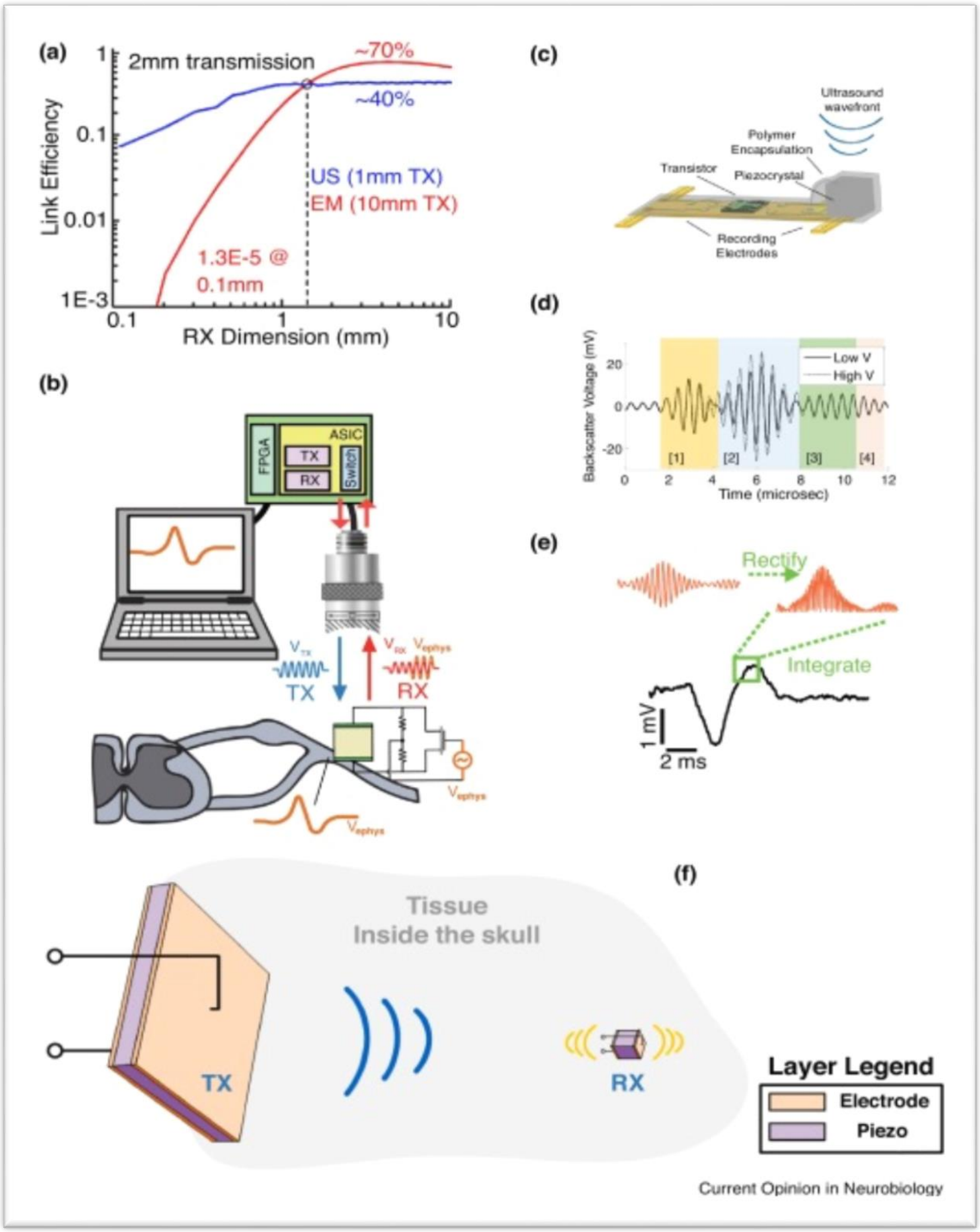


Figure 3 Neural Dust Working

7.Retriggering the emotion - Optogenetics

Optogenetics is a method that uses light to modulate molecular events in a targeted manner in living cells or organisms. It relies on the use of genetically-encoded proteins that change conformation in the presence of light to alter cell behavior for example, by changing the membrane voltage potential of excitable cells.

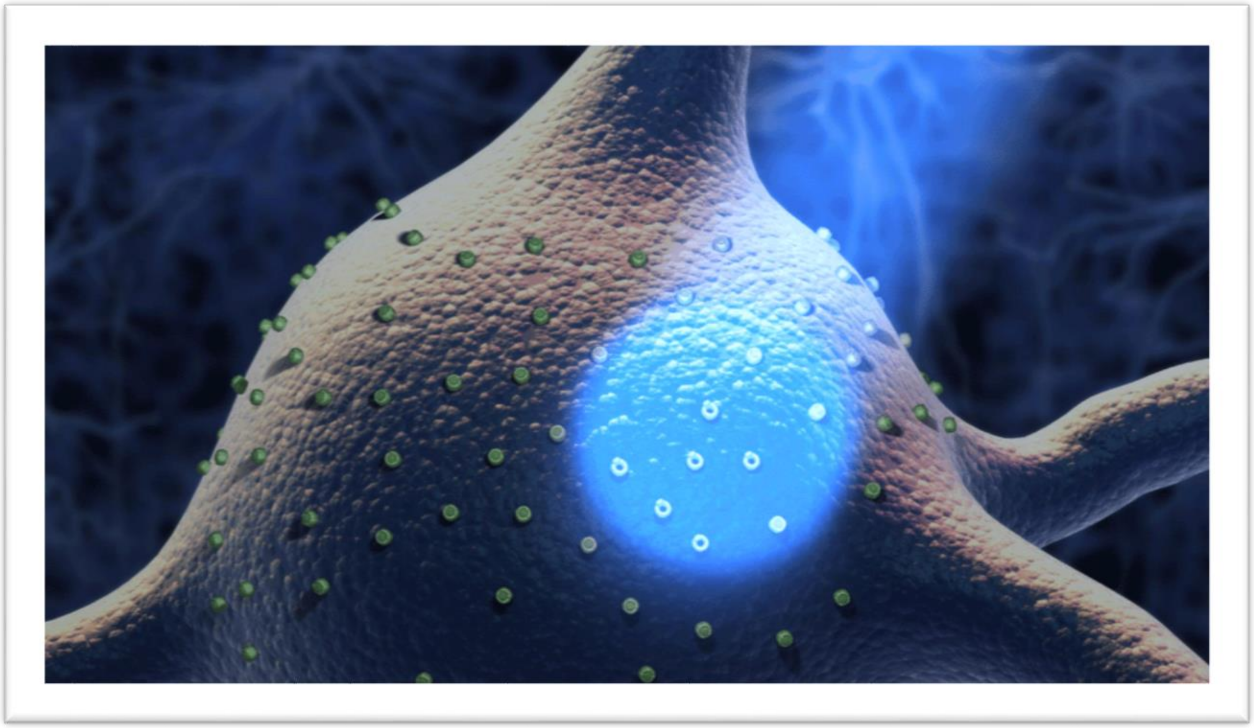


Figure 4 Optogenetics

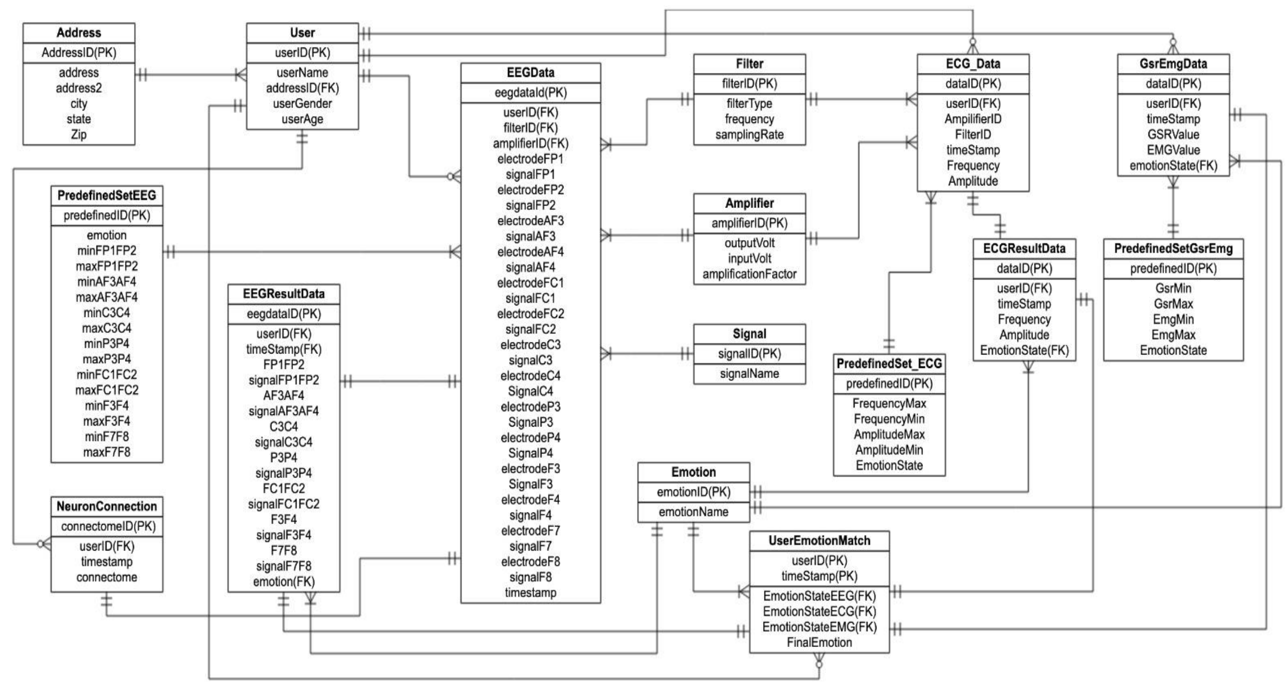
Comprehension of the brain function can be helpful for therapy of neurodegenerative diseases. The brain consists of various types of neuron sets, which organize in three-dimensional complex networks and form neural circuits underlying different behaviors. The circuits act based on the patterns that encode the brain functions. Recognition of the neural patterns requires methods to manipulate the neurons. Electrical stimulation may be the most common method. However, it has significant drawbacks including failure to identify specific neurons in experiments. As an alternative, optical stimulation is a new method that acts in combination with genetic approaches. The novel, optogenetic technology makes it feasible to manipulate either the specific cell types or the neural circuits. This is associated with minimum tissue damages as well as side effects.

8.Visualization of the Database

This project assumes that the User saves all his memories and thoughts of brain into a database in the form of neural points (Connectome) and uses those data to retrigger his/her emotions. The User’s emotions are analyzed using three techniques: EEG, ECG, GSR-EMG and finalizes the emotion at any time by comparing the results of predefined data sets of all the three techniques. Also, the pattern of neurons connected obtained from Neural Dust technique for each emotion at each timestamp is stored in the database which represents the neural points where his memories are stored. Now as all the data to retrigger an emotion is stored, whenever user wish to retrigger the emotion using Optogenetics.

8.1. Entity Relationship Diagram

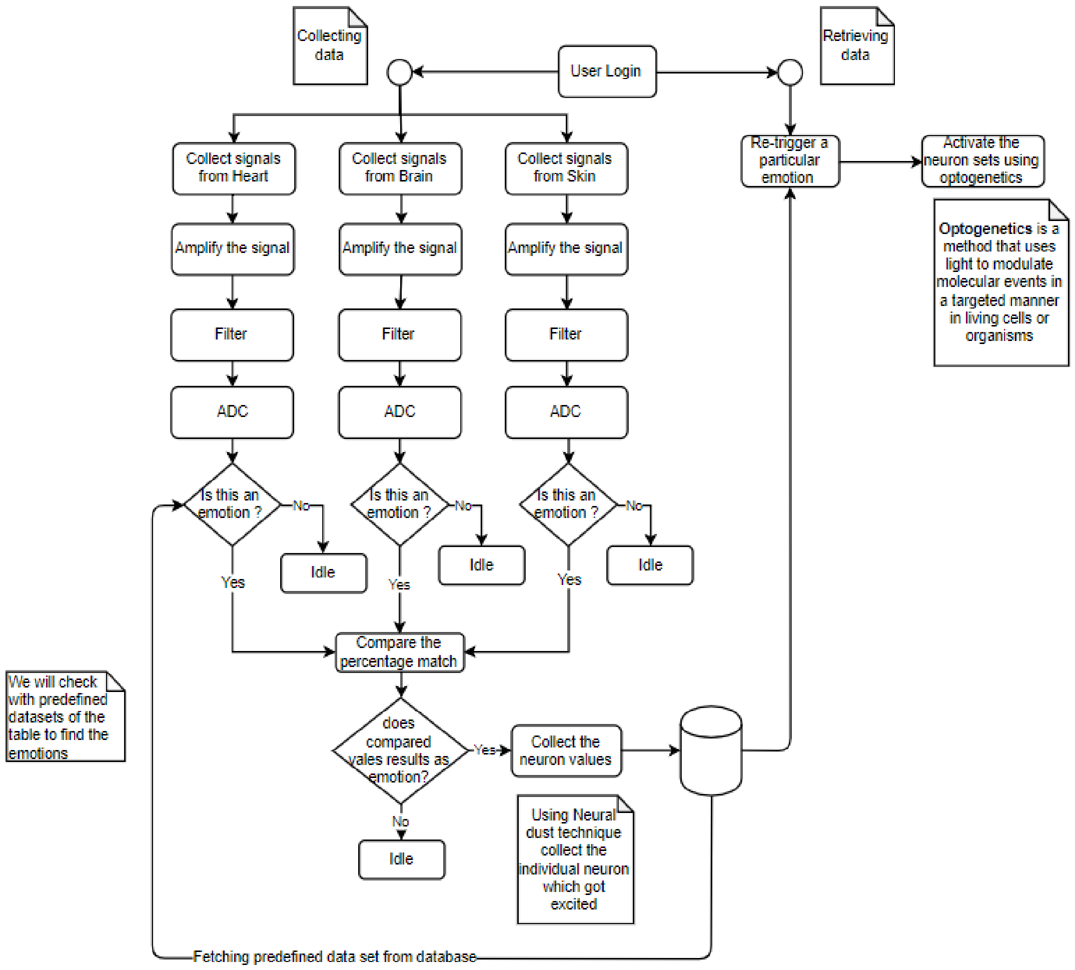
The Entity Relationship Diagram (ERD) is a visual representation showing the relationship between the multiple tables of data in our project. The Crow’s Foot Notation used in our ERD clearly defines the cardinality between the tables as well. This is a vital step that is conducted for a User to understand how all the information a system produces is related.



Some of the relationships that can be observed from the diagram is that one eegDataID(EEGResultData) have only one FinalEmotion(UserEmotionMatch) and one FinalEmotion(UserEmotionmatch) belongs to only one eegDataID(EEGResultData hence suggesting a one-to-one relationship between the EEGResultData and the UserEmotionMatch. Similarly, one emotion hence describing a one-to-many relationship. This information becomes useful to the User to link the tables and populating other tables in a sequence.

8.2. Activity Diagram

The Activity diagram gives a visual representation of the sequence in which activities that the DBMS system is designed to do occurs. The chronological order of the activities gives insight into the output of the data management system that is designed. Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of the DBMS system, but they are also used to construct the executable system by using forward and reverse engineering techniques.

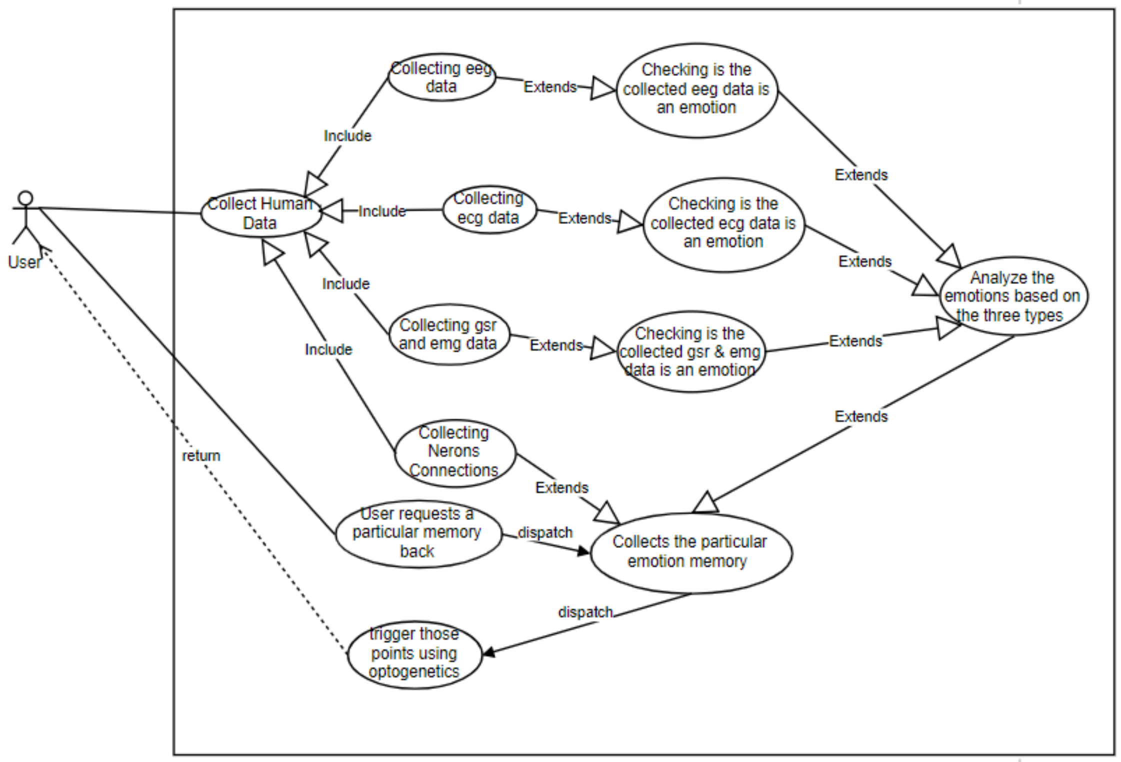


Collecting data
User Login
Retrieving data
Collect signals from Heart
Collect signals from Brain
Collect signals from Skin
Amplify the signal
Filter
ADC
Is this an emotion ?
No
Idle
Yes
Compare the percentage match
does compared vales results as emotion?
Yes
Collect the neuron values
No
Idle
Using Neural dust technique collect the individual neuron which got excited
We will check with predefined datasets of the table to find the emotions
Fetching predefined data set from database
Re-trigger a particular emotion
Activate the neuron sets using optogenetics
Optogenetics is a method that uses light to modulate molecular events in a targeted manner in living cells or organisms

The diagram enumerates the steps in which the database is going to achieve our goal. The User for collecting the data uses three techniques and collect signals from Heart, Brain and Skin. Next the collected signals are sent to an amplifier for amplification and next are filtered using filter. Now the signal data is converted to digital form using an Analog to Digital Converter (ADC). Now to check whether the collected signal is an emotion, each data is compared with a predefined set in all the three techniques. Next, along with emotion neuron connections are needed which are found using Neural Dust technique. All the values are stored in database.

Now the User to retrigger an emotion, finds the emotion and corresponding neuron connection values from the database for any particular timestamp. Using Optogenetics, the user can activate the neuron sets. Hence finally achieved retriggering of emotion.

8.3. Use Case Diagram



User
Collect Human Data
Collecting eeg data
Collecting ecg data
Collecting gsr and emg data
Collecting Nerons Connections
Include
Include
Include
Include
Extends
Checking is the collected eeg data is an emotion
Checking is the collected ecg data is an emotion
Checking is the collected gsr & emg data is an emotion
Analyze the emotions based on the three types
Extends
User requests a particular memory back
dispatch
Collects the particular emotion memory
dispatch
trigger those points using optogenetics
return

The Use case diagram represents the set of actions and functions the system must perform for the given user. In our system the User’s emotions and neuron connections are saved in the database. The generic notation of collecting human data includes collecting EEG,ECG,GSR and EMG data which are then further extended to find whether the user has undergone predefined human emotions. When the user does an action of retriggering some memory back, the collected neuron patterns are returned to the Optogenetics for the human to get the memory back.

8.4. Visual representation of Database Implementation

---------------- Need to draw -----------

7. Tables created in the Database

Information is stored in databases in the form of tables. Each of the tables contain several valuable information that are stored as attributes. Different tables are related to each other in one-to-one, one-to-many or many-to-many relationships which are all enumerated in the ERD. A total of 16 tables exist in this database. All the tables present in our database is in the 3NF form. The following table lists all the tables created in our database including indication of the primary key, PK and the foreign key, FK.

Table 1: Tables created in the Database

S. no.	*Table Name*	*Attributes*
1	***Address***	***(PK) AddressID*** *Address* *Address2* *City* *State* *Zip*
2	***User***	***(PK) User_ID*** *User_Name* *AddressID(**FK**)* *UserGender* *UserAge*
3	***EEGData***	***(PK) EEGDataID*** *userID(**FK**)* *filterID(**FK**)* *amplifierID(**FK**)* *electrodeFP1* *signalFP1* *electrodeFP2* *signalFP2* *electrodeAF3* *signalAF3* *electrodeAF4* *signalAF4* *electrodeFC1* *signalFC1* *electrodeFC2* *signalFC2* *electrodeC3* *signalC3* *electrodeC4* *SignalC4* *electrodeP3* *SignalP3* *electrodeP4* *SignalP4* *electrodeF3* *SignalF3* *electrodeF4* *signalF4* *electrodeF7* *signalF7* *electrodeF8* *signalF8* *timestamp*
4	***PreDefinedSetEEG***	***(PK) predefinedID*** *emotion*

		minFP1FP2 *maxFP1FP2* *minAF3AF4* *maxAF3AF4* *minC3C4* *maxC3C4* *minP3P4* *maxP3P4* *minFC1FC2* *maxFC1FC2* *minF3F4* *maxF3F4* *minF7F8* *maxF7F8*
5	***EEGResultData***	***(PK,FK) eegDataID*** *userID**(FK)*** *timeStamp* *FP1FP2* *signalFP1FP2* *AF3AF4* *signalAF3AF4* *C3C4* *signalC3C4* *P3P4* *signalP3P4* *FC1FC2* *signalFC1FC2* *F3F4* *signalF3F4* *F7F8* *signalF7F8* *emotion**(FK)***
6	***NeuronConnection***	***(PK) ConnectomeID*** *userID**(FK)*** *timestamp* *connectome*
7	***Filter***	***(PK) FilterID*** *filterType* *frequency* *samplingRate*
8	***Amplifier***	***(PK) AmplifierID*** *outputVolt* *inputVolt* *amplificationFactor*
9	***Signal***	***(PK) SignalID*** *SignalName*
10	***Emotion***	***(PK) EmotionID*** *EmotionName*
11	***ECG_Data***	***(PK) dataID*** *userID**(FK)*** *AmpilifierID* *FilterID* *timeStamp* *Frequency* *Amplitude*
12	***PredefinedSetECG***	***(PK) PredefinedID*** *FrequencyMax* *FrequencyMin* *AmplitudeMax* *AmplitudeMin* *EmotionState*
13	***ECGResultData***	***(PK) dataID*** *userID(FK)* *timeStamp* *Frequency* *Amplitude* *EmotionState**(FK)***
14	***GsrEmgData***	***(PK,FK) dataID*** *userID(FK)*

		timeStamp *GSRValue* *EMGValue* *emotionState(**FK**)*
15	***PredefinedSetGsrEmg***	***(PK,FK) predefinedID*** *GsrMin* *GsrMax* *EmgMin* *EmgMax* *EmotionState*

16	***UserEmotionMatch***	***(PK) UserID, timeStamp***
		EmotionStateEEG(FK)
		EmotionStateECG(FK)
		EmotionStateEMG(FK)
		FinalEmotion

8. Functions

FUNCTION 1: RETURN EMOTION ID

```
DELIMITER $$
CREATE FUNCTION RETURNEMOTIONID (VAL VARCHAR(25))
RETURNS INT
BEGIN
        RETURN (SELECT EMOTIONID FROM EMOTION WHERE EMOTIONNAME = VAL);
END
$$;
```

***DESCRIPTION*:**
This function is used to return emotion id for a given emotion name for a given emotion name from emotion table.

FUNCTION 2: RETURN EMOTION NAME

```
DELIMITER $$
CREATE FUNCTION RETURNEMOTIONNAME (VAL INT)
RETURNS VARCHAR (25)

BEGIN
         RETURN (SELECT EMOTIONNAME FROM EMOTION WHERE EMOTIONID = VAL);
END
$$;
```

***DESCRIPTION*:**
This function is used to return emotion name for a given emotion id for a given emotion name from emotion table.

FUNCTION 3: GT_SIGNAL

```
DELIMITER $$
CREATE FUNCTION GT_SIGNAL (INPUT_SIGNAL VARCHAR (20))
```

```
RETURNS INT

BEGIN
            RETURN (SELECT SIGNALID FROM SIGNAL WHERE SIGNALNAME = INPUT_SIGNAL);
END
$$;
```

***DESCRIPTION*:**
This function is used to determine the signal id for a given signal name from signal table.

FUNCTION 4: GT_DIV

```
drop function IF EXISTS gt_div;
delimiter $$
create function gt_div (input_num1 decimal (6,4),input_num2 decimal(6,4))
            returns decimal (6,4)
begin
            return input_num1/input_num2;
end;
$$
delimiter;
```

***DESCRIPTION*:**
This function is written to calculate the Asymmetry Index which is the ratio of the signals collected from EEG.
Let x1 and x2 be two input signals from EEG. Then Asymmetry Index is calculated as:

Asymmetry Index = x1/x2

The function returns the Asymmetry Index value.

FUNCTION 5: EMOTIONNAMECALC

```
drop function IF EXISTS emotionNameCalc;
delimiter $$
CREATE function emotionNameCalc(FP1FP2 decimal,AF3AF4 decimal,c3c4 decimal,p3p4 decimal,fc1fc2 decimal,
F3F4 decimal, f7f8 decimal,
SignalFP1FP2 int, SignalAF3AF4 int,Signalc3c4 int,Signalp3p4 int,Signalfc1fc2 int,SignalF3F4 int, Signalf7f8 int)
returns int
BEGIN

  DECLARE HAPPY INT DEFAULT 0;
    DECLARE SAD INT DEFAULT 0;
    DECLARE FEAR INT DEFAULT 0;
    DECLARE ANGER INT DEFAULT 0;
    Declare val varchar(80);
    DECLARE EMOTIONID INT DEFAULT 0;

            if FP1FP2 > (select minFP1FP2 from PredefinedSetEEG where emotion = 'HAPPY') AND FP1FP2 <(select
maxFP1FP2 from PredefinedSetEEG where emotion = 'HAPPY') AND SignalFP1FP2 = gt_signal('Alpha')
                        Then set HAPPY = HAPPY+14.25;

     else if FP1FP2 > (select minFP1FP2 from PredefinedSetEEG where emotion = 'SAD') AND FP1FP2 <(select
maxFP1FP2 from PredefinedSetEEG where emotion = 'SAD') AND SignalFP1FP2 = gt_signal('Alpha')
                                    Then set SAD = SAD+14.25;
            else if FP1FP2 > (select minFP1FP2 from PredefinedSetEEG where emotion = 'FEAR') AND FP1FP2
<(select maxFP1FP2 from PredefinedSetEEG where emotion = 'FEAR') AND SignalFP1FP2 = gt_signal('Alpha')
                                                Then set FEAR = FEAR + 14.25;

            else if FP1FP2 > (select minFP1FP2 from PredefinedSetEEG where emotion = 'ANGER') AND FP1FP2
<(select maxFP1FP2 from PredefinedSetEEG where emotion = 'ANGER') AND SignalFP1FP2 = gt_signal('Alpha')
                                                Then set ANGER = ANGER + 14.25;
             end if;
             end if;
             end if;
             end if;

            if AF3AF4 > (select minAF3AF4 from PredefinedSetEEG where emotion = 'HAPPY') AND AF3AF4
<(select maxAF3AF4 from PredefinedSetEEG where emotion = 'HAPPY') AND SignalAF3AF4 = gt_signal('Alpha')
                        Then set HAPPY = HAPPY+14.25;

        else if AF3AF4 > (select minAF3AF4 from PredefinedSetEEG where emotion = 'SAD') AND AF3AF4 <(select
maxAF3AF4 from PredefinedSetEEG where emotion = 'SAD') AND SignalAF3AF4 = gt_signal('Alpha')
                                    Then set SAD = SAD+14.25;

            else if AF3AF4 > (select minAF3AF4 from PredefinedSetEEG where emotion = 'FEAR') AND AF3AF4
<(select maxAF3AF4 from PredefinedSetEEG where emotion = 'FEAR') AND SignalAF3AF4 = gt_signal('Alpha')
                                                Then set FEAR = FEAR + 14.25;

            else if AF3AF4 > (select minAF3AF4 from PredefinedSetEEG where emotion = 'ANGER') AND AF3AF4
<(select maxAF3AF4 from PredefinedSetEEG where emotion = 'ANGER') AND SignalAF3AF4 = gt_signal('Alpha')
                                                Then set ANGER = ANGER + 14.25;
             end if;
             end if;
             end if;
             end if;

            if c3c4 > (select minc3c4 from PredefinedSetEEG where emotion = 'HAPPY') AND c3c4 <(select maxc3c4
from PredefinedSetEEG where emotion = 'HAPPY') AND Signalc3c4 = gt_signal('Alpha')
                        Then set HAPPY = HAPPY+14.25;

        else if c3c4 > (select minc3c4 from PredefinedSetEEG where emotion = 'SAD') AND c3c4 <(select maxc3c4 from
PredefinedSetEEG where emotion = 'SAD') AND Signalc3c4 = gt_signal('Alpha')
                                    Then set SAD = SAD+14.25;

            else if c3c4 > (select minc3c4 from PredefinedSetEEG where emotion = 'FEAR') AND c3c4 <(select
maxc3c4 from PredefinedSetEEG where emotion = 'FEAR') AND Signalc3c4 = gt_signal('Alpha')
                                                Then set FEAR = FEAR + 14.25;

            else if c3c4 > (select minc3c4 from PredefinedSetEEG where emotion = 'ANGER') AND c3c4 <(select
maxc3c4 from PredefinedSetEEG where emotion = 'ANGER') AND Signalc3c4 = gt_signal('Alpha')
                                                Then set ANGER = ANGER + 14.25;
             end if;
             end if;
             end if;
             end if;

            if p3p4 > (select minp3p4 from PredefinedSetEEG where emotion = 'HAPPY') AND p3p4 <(select maxp3p4
from PredefinedSetEEG where emotion = 'HAPPY') AND Signalp3p4 = gt_signal('Alpha')
                        Then set HAPPY = HAPPY+14.25;
```

```
    else if p3p4 > (select minp3p4 from PredefinedSetEEG where emotion = 'SAD') AND p3p4 <(select maxp3p4
from PredefinedSetEEG where emotion = 'SAD') AND Signalp3p4 = gt_signal('Alpha')
                            Then set SAD = SAD+14.25;

          else if p3p4 > (select minp3p4 from PredefinedSetEEG where emotion = 'FEAR') AND p3p4 <(select
maxp3p4 from PredefinedSetEEG where emotion = 'FEAR') AND Signalp3p4 = gt_signal('Alpha')
                                      Then set FEAR = FEAR + 14.25;

          else if p3p4 > (select minp3p4 from PredefinedSetEEG where emotion = 'ANGER') AND p3p4 <(select
maxp3p4 from PredefinedSetEEG where emotion = 'ANGER') AND Signalp3p4 = gt_signal('Alpha')
                                      Then set ANGER = ANGER + 14.25;
           end if;
           end if;
           end if;
           end if;

          if fc1fc2 > (select minfc1fc2 from PredefinedSetEEG where emotion = 'HAPPY') AND fc1fc2 <(select
maxfc1fc2 from PredefinedSetEEG where emotion = 'HAPPY') AND Signalfc1fc2 = gt_signal('Alpha')
                  Then set HAPPY = HAPPY+14.25;

    else if fc1fc2 > (select minfc1fc2 from PredefinedSetEEG where emotion = 'SAD') AND fc1fc2 <(select maxfc1fc2
from PredefinedSetEEG where emotion = 'SAD') AND Signalfc1fc2 = gt_signal('Alpha')
                            Then set SAD = SAD+14.25;

          else if fc1fc2 > (select minfc1fc2 from PredefinedSetEEG where emotion = 'FEAR') AND fc1fc2 <(select
maxfc1fc2 from PredefinedSetEEG where emotion = 'FEAR') AND Signalfc1fc2 = gt_signal('Alpha')
                                      Then set FEAR = FEAR + 14.25;

          else if fc1fc2 > (select minFP1FP2 from PredefinedSetEEG where emotion = 'ANGER') AND FP1FP2
<(select maxfc1fc2 from PredefinedSetEEG where emotion = 'ANGER') AND Signalfc1fc2 = gt_signal('Alpha')
                                      Then set ANGER = ANGER + 14.25;
           end if;
           end if;
           end if;
           end if;

          if F3F4 > (select minF3F4 from PredefinedSetEEG where emotion = 'HAPPY') AND F3F4 <(select
maxF3F4 from PredefinedSetEEG where emotion = 'HAPPY')  AND SignalF3F4 = gt_signal('Alpha')
                  Then set HAPPY = HAPPY+14.25;

    else if F3F4 > (select minF3F4 from PredefinedSetEEG where emotion = 'SAD') AND F3F4 <(select maxF3F4
from PredefinedSetEEG where emotion = 'SAD') AND SignalF3F4 = gt_signal('Alpha')
                            Then set SAD = SAD+14.25;

          else if F3F4 > (select minF3F4 from PredefinedSetEEG where emotion = 'FEAR') AND F3F4 <(select
maxF3F4 from PredefinedSetEEG where emotion = 'FEAR') AND SignalF3F4 = gt_signal('Alpha')
                                      Then set FEAR = FEAR + 14.25;

          else if F3F4 > (select minF3F4 from PredefinedSetEEG where emotion = 'ANGER') AND F3F4 <(select
maxF3F4 from PredefinedSetEEG where emotion = 'ANGER') AND SignalF3F4 = gt_signal('Alpha')
                                      Then set ANGER = ANGER + 14.25;
           end if;
           end if;
           end if;
           end if;
```

```
            if f7f8 > (select minf7f8 from PredefinedSetEEG where emotion = 'HAPPY') AND f7f8 <(select maxf7f8
from PredefinedSetEEG where emotion = 'HAPPY') AND Signalf7f8 = gt_signal('Alpha')
                        Then set HAPPY = HAPPY+14.25;

            else if f7f8 > (select minf7f8 from PredefinedSetEEG where emotion = 'SAD') AND f7f8 <(select maxf7f8
from PredefinedSetEEG where emotion = 'SAD') AND Signalf7f8 = gt_signal('Alpha')
                        Then set SAD = SAD+14.25;

            else if f7f8 > (select minf7f8 from PredefinedSetEEG where emotion = 'FEAR') AND f7f8 <(select maxf7f8
from PredefinedSetEEG where emotion = 'FEAR') AND Signalf7f8 = gt_signal('Alpha')
                        Then set FEAR = FEAR + 14.25;

            else if f7f8 > (select minf7f8 from PredefinedSetEEG where emotion = 'ANGER') AND f7f8 <(select
maxf7f8 from PredefinedSetEEG where emotion = 'ANGER') AND Signalf7f8 = gt_signal('Alpha')
                        Then set ANGER = ANGER + 14.25;
                        end if;
        end if;
        end if;
        end if;

          if HAPPY>SAD AND HAPPY> FEAR And HAPPY > ANGER
          then set val ='HAPPY';
          else if SAD>HAPPY and SAD>FEAR and SAD > ANGER
          then set val = 'SAD';
          else if FEAR>HAPPY and FEAR>SAD and FEAR>ANGER
          then set val = 'FEAR';
          else if ANGER>HAPPY and ANGER>FEAR and ANGER>SAD
          then set val = 'ANGER';
          else set val = 'UNIDENTIFIED';
          end if;
          end if;
          end if;
          end if;

           RETURN RETURNEMOTIONID (val);

END
$$ ;
```

***DESCRIPTION*:**
This function calculates the emotion from the input signals of EEG.
The function calculates the Asymmetry indices of all the signals and compare them with the predefind set of EEG signals and returns the Emotion.

FUNCTION 6: EMOTIONSTATEDETECTECG

```
DROP FUNCTION IF EXISTS emotionStateDetectECG
DELIMITER $$
CREATE FUNCTION emotionStateDetectECG(frequency decimal(6,4),amplitude decimal(6,4))
RETURNS INT
BEGIN
Declare state varchar(256);
```

```
IF (frequency>(SELECT frequencyMin FROM PredefinedSet_ECG WHERE EmotionState='HAPPY') &&
frequency<(SELECT frequencyMax FROM PredefinedSet_ECG WHERE EmotionState='HAPPY')
        &&
            amplitude>(SELECT amplitudeMin FROM PredefinedSet_ECG WHERE EmotionState='HAPPY') &&
amplitude<(SELECT amplitudeMax FROM PredefinedSet_ECG WHERE EmotionState='HAPPY'))
  THEN SET state = 'HAPPY';
ELSE IF (frequency>(SELECT frequencyMin FROM PredefinedSet_ECG WHERE EmotionState='SAD') &&
frequency<(SELECT frequencyMax FROM PredefinedSet_ECG WHERE EmotionState='SAD')
        &&
            amplitude>(SELECT amplitudeMin FROM PredefinedSet_ECG WHERE EmotionState='SAD') &&
amplitude<(SELECT amplitudeMax FROM PredefinedSet_ECG WHERE EmotionState='SAD'))
  THEN SET state = 'SAD';
ELSE IF (frequency>(SELECT frequencyMin FROM PredefinedSet_ECG WHERE EmotionState='FEAR') &&
frequency<(SELECT frequencyMax FROM PredefinedSet_ECG WHERE EmotionState='FEAR')
        &&
            amplitude>(SELECT amplitudeMin FROM PredefinedSet_ECG WHERE EmotionState='FEAR') &&
amplitude<(SELECT amplitudeMax FROM PredefinedSet_ECG WHERE EmotionState='FEAR'))
  THEN SET state = 'FEAR';
ELSE IF (frequency>(SELECT frequencyMin FROM PredefinedSet_ECG WHERE EmotionState='ANGER') &&
frequency<(SELECT frequencyMax FROM PredefinedSet_ECG WHERE EmotionState='ANGER')
        &&
            amplitude>(SELECT amplitudeMin FROM PredefinedSet_ECG WHERE EmotionState='ANGER') &&
amplitude<(SELECT amplitudeMax FROM PredefinedSet_ECG WHERE EmotionState='ANGER'))
  THEN SET state = 'ANGER';
ELSE
 SET state = 'UNIDENTIFIED';
END IF;
END IF;
END IF;
END IF;

RETURN RETURNEMOTIONID(state);

END
$$
DELIMITER ;
```

***DESCRIPTION*:**
This function calculates the emotion from the input signals of ECG.
The function compares each signal frequency and amplitude with the predefined set of ECG values and returns the Emotion.

FUNCTION 7: EMOTIONSTATEDETECT

```
DELIMITER $$
CREATE FUNCTION emotionStateDetect(gsr decimal(6,4),emg decimal(6,4))
 RETURNS INT
BEGIN
Declare state varchar(80);

IF (gsr>(SELECT GSRMin FROM PredefinedSetGsrEmg WHERE EmotionState='HAPPY') && gsr<(SELECT
GSRMax FROM PredefinedSetGsrEmg WHERE EmotionState='HAPPY')
        &&
            emg>(SELECT EMGMin FROM PredefinedSetGsrEmg WHERE EmotionState='HAPPY') &&
emg<(SELECT EMGMax FROM PredefinedSetGsrEmg WHERE EmotionState='HAPPY'))
```

```
  THEN SET state = 'HAPPY';
ELSE IF (gsr>(SELECT GSRMin FROM PredefinedSetGsrEmg WHERE EmotionState='SAD') && gsr<(SELECT GSRMax FROM PredefinedSetGsrEmg WHERE EmotionState='SAD')
        &&
            emg>(SELECT EMGMin FROM PredefinedSetGsrEmg WHERE EmotionState='SAD') && emg<(SELECT EMGMax FROM PredefinedSetGsrEmg WHERE EmotionState='SAD'))
  THEN SET state = 'SAD';
ELSE IF (gsr>(SELECT GSRMin FROM PredefinedSetGsrEmg WHERE EmotionState='FEAR') && gsr<(SELECT GSRMax FROM PredefinedSetGsrEmg WHERE EmotionState='FEAR')
        &&
            emg>(SELECT EMGMin FROM PredefinedSetGsrEmg WHERE EmotionState='FEAR') && emg<(SELECT EMGMax FROM PredefinedSetGsrEmg WHERE EmotionState='FEAR'))
  THEN SET state = 'FEAR';
ELSE IF (gsr>(SELECT GSRMin FROM PredefinedSetGsrEmg WHERE EmotionState='ANGER') && gsr<(SELECT GSRMax FROM PredefinedSetGsrEmg WHERE EmotionState='ANGER')
        &&
            emg>(SELECT EMGMin FROM PredefinedSetGsrEmg WHERE EmotionState='ANGER') && emg<(SELECT EMGMax FROM PredefinedSetGsrEmg WHERE EmotionState='ANGER'))
  THEN SET state = 'ANGER';
ELSE
   SET state = 'UNIDENTIFIED';
END IF;
END IF;
END IF;
END IF;

RETURN RETURNEMOTIONID(state);

END
$$
DELIMITER ;
```

***DESCRIPTION*:**
This function calculates the emotion from the input values of GSR and EMG.
The function compares each GSR and EMG value with the predefined set of GSR, EMG values and returns the Emotion.

FUNCTION 8: FINALEMOTIONDETECT

```
DELIMITER $$
CREATE FUNCTION FinalemotionDetect(EEGemotion varchar(256),ECGemotion varchar(256),GSREMGemotion varchar(256))
 RETURNS INT
BEGIN
Declare state varchar(256);
DECLARE HAPPY int;
DECLARE SAD int;
DECLARE ANGER int;
DECLARE FEAR int;
DECLARE UNIDENTIFIED int;

SET HAPPY =0,SAD=0,ANGER=0,FEAR=0,UNIDENTIFIED=0;

IF (EEGemotion='HAPPY')
  THEN SET HAPPY=HAPPY+1;
```

```
ELSE IF (EEGemotion='SAD')
  THEN SET SAD=SAD+1;
ELSE IF (EEGemotion='FEAR')
  THEN SET FEAR=FEAR+1;
ELSE IF (EEGemotion='ANGER')
  THEN SET ANGER=ANGER+1;
ELSE
   SET UNIDENTIFIED = UNIDENTIFIED+1;
END IF;
END IF;
END IF;
END IF;

IF (ECGemotion='HAPPY')
  THEN SET HAPPY=HAPPY+1;
ELSE IF (ECGemotion='SAD')
  THEN SET SAD=SAD+1;
ELSE IF (ECGemotion='FEAR')
  THEN SET FEAR=FEAR+1;
ELSE IF (ECGemotion='ANGER')
  THEN SET ANGER=ANGER+1;
ELSE
   SET UNIDENTIFIED = UNIDENTIFIED+1;
END IF;
END IF;
END IF;
END IF;

IF (GSREMGemotion='HAPPY')
  THEN SET HAPPY=HAPPY+1;
ELSE IF (GSREMGemotion='SAD')
  THEN SET SAD=SAD+1;
ELSE IF (GSREMGemotion='FEAR')
  THEN SET FEAR=FEAR+1;
ELSE IF (GSREMGemotion='ANGER')
  THEN SET ANGER=ANGER+1;
ELSE
   SET UNIDENTIFIED = UNIDENTIFIED+1;
END IF;
END IF;
END IF;
END IF;

IF(HAPPY>SAD && HAPPY>FEAR && HAPPY>ANGER && HAPPY>UNIDENTIFIED)
  THEN RETURN RETURNEMOTIONID('HAPPY');
ELSE IF(SAD>HAPPY && SAD>FEAR && SAD>ANGER && SAD>UNIDENTIFIED)
  THEN RETURN RETURNEMOTIONID('SAD');
ELSE IF(FEAR>HAPPY && FEAR>SAD && FEAR>ANGER && FEAR>UNIDENTIFIED)
  THEN RETURN RETURNEMOTIONID('FEAR');
 ELSE IF(ANGER>HAPPY && ANGER>SAD && ANGER>FEAR && ANGER>UNIDENTIFIED)
  THEN RETURN RETURNEMOTIONID('ANGER');
 ELSE IF(UNIDENTIFIED>HAPPY && UNIDENTIFIED>SAD && UNIDENTIFIED>FEAR &&
UNIDENTIFIED>ANGER)
  THEN RETURN RETURNEMOTIONID('UNIDENTIFIED');
ELSE
   RETURN RETURNEMOTIONID('UNIDENTIFIED');
```

```
END IF;
END IF;
END IF;
END IF;
END IF;

END
$$
DELIMITER ;
```

***DESCRIPTION*:**
This function calculates the emotion from averaging the emotion from all 3 techniques: EEG, ECG and GSR-EMG.

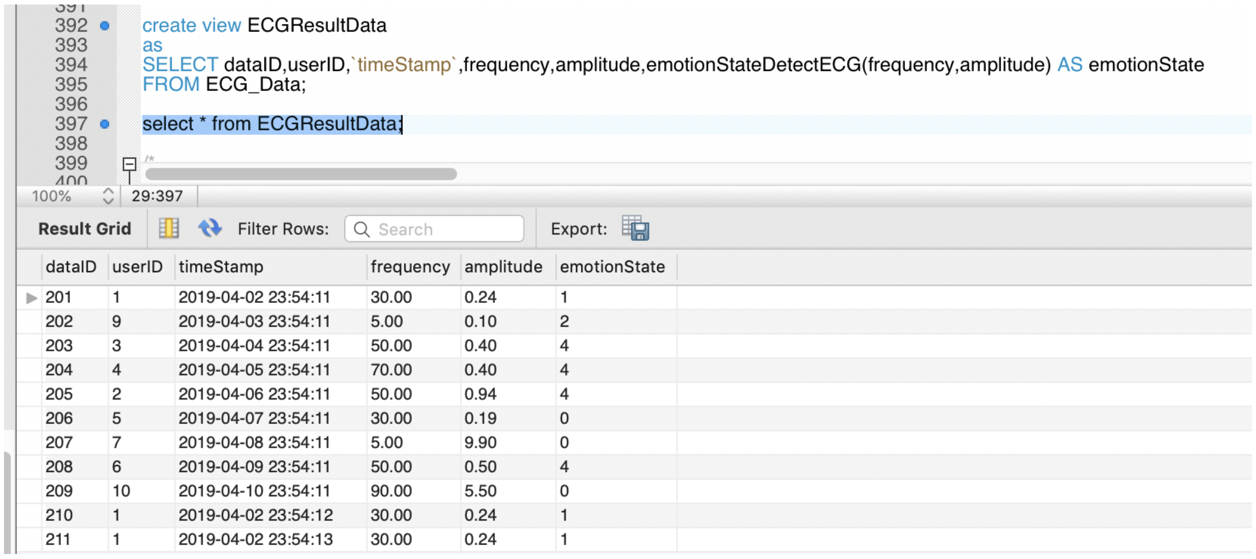
9. Views

Views are created to reduce the complexity of a large table and display only appropriate and relevant columns or data at a specific stage for users. The following views are created in our database. All these views serve different purposes and require specific information to be pulled from different tables and displayed together. Since the information is present on other tables, views are used to pull all relevant information required for the users' benefit. The code displayed below is writen for the implementation of views.

VIEW1: ECGResultData

```
create view ECGResultData
as
SELECT dataID, userID,`timeStamp`,frequency,amplitude,emotionStateDetectECG(frequency,amplitude) AS emotionState FROM ECG_Data;
```

OUTPUT:



```
create view ECGResultData
as
SELECT dataID,userID,`timeStamp`,frequency,amplitude,emotionStateDetectECG(frequency,amplitude) AS emotionState
FROM ECG_Data;

select * from ECGResultData;
```

100% 29:397

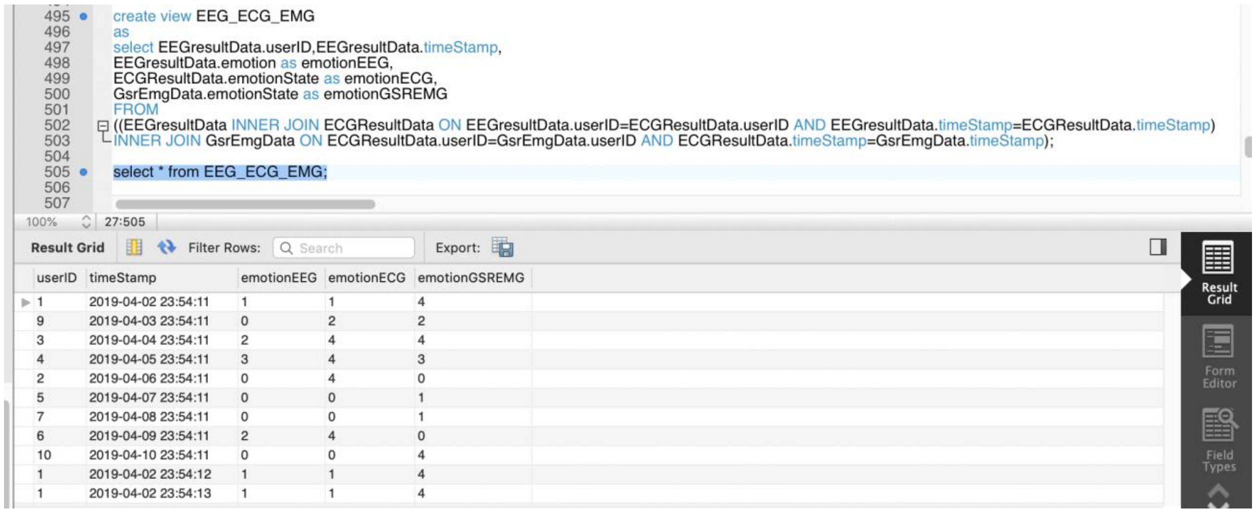
Result Grid Filter Rows: Search Export:

dataID	userID	timeStamp	frequency	amplitude	emotionState
201	1	2019-04-02 23:54:11	30.00	0.24	1
202	9	2019-04-03 23:54:11	5.00	0.10	2
203	3	2019-04-04 23:54:11	50.00	0.40	4
204	4	2019-04-05 23:54:11	70.00	0.40	4
205	2	2019-04-06 23:54:11	50.00	0.94	4
206	5	2019-04-07 23:54:11	30.00	0.19	0
207	7	2019-04-08 23:54:11	5.00	9.90	0
208	6	2019-04-09 23:54:11	50.00	0.50	4
209	10	2019-04-10 23:54:11	90.00	5.50	0
210	1	2019-04-02 23:54:12	30.00	0.24	1
211	1	2019-04-02 23:54:13	30.00	0.24	1

VIEW2: EEG_ECG_EMG

create view EEG_ECG_EMG
as
select EEGresultData.userID,EEGresultData.timeStamp,
EEGresultData.emotion as emotionEEG,
ECGResultData.emotionState as emotionECG,
GsrEmgData.emotionState as emotionGSREMG
FROM
((EEGresultData INNER JOIN ECGResultData ON EEGresultData.userID=ECGResultData.userID AND EEGresultData.timeStamp=ECGResultData.timeStamp)
INNER JOIN GsrEmgData ON ECGResultData.userID=GsrEmgData.userID AND ECGResultData.timeStamp=GsrEmgData.timeStamp);

OUTPUT:



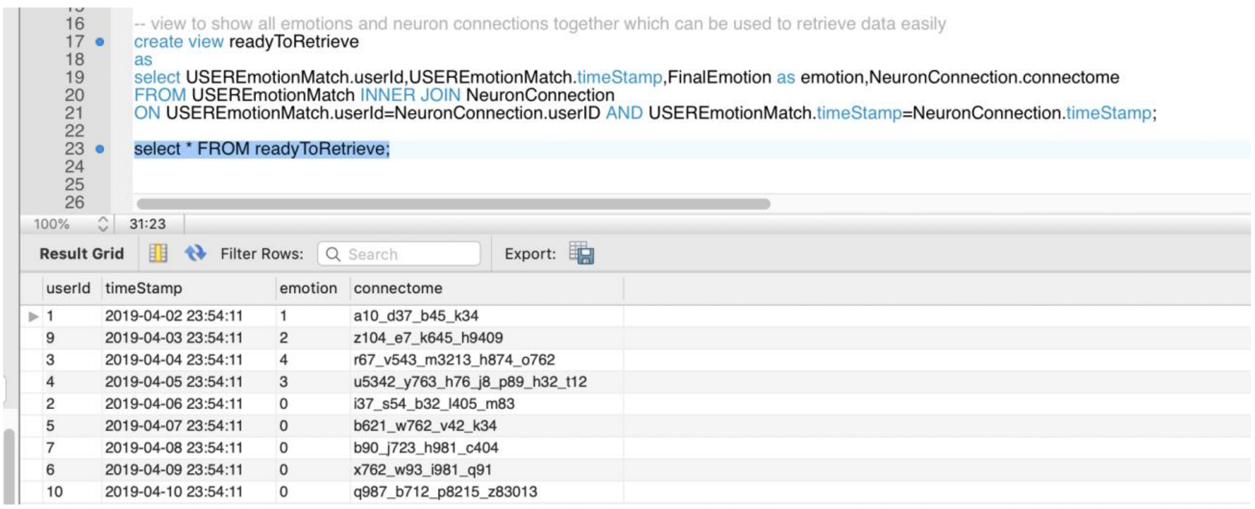
```
create view EEG_ECG_EMG
as
select EEGresultData.userID,EEGresultData.timeStamp,
EEGresultData.emotion as emotionEEG,
ECGResultData.emotionState as emotionECG,
GsrEmgData.emotionState as emotionGSREMG
FROM
((EEGresultData INNER JOIN ECGResultData ON EEGresultData.userID=ECGResultData.userID AND EEGresultData.timeStamp=ECGResultData.timeStamp)
INNER JOIN GsrEmgData ON ECGResultData.userID=GsrEmgData.userID AND ECGResultData.timeStamp=GsrEmgData.timeStamp);

select * from EEG_ECG_EMG;
```

userID	timeStamp	emotionEEG	emotionECG	emotionGSREMG
1	2019-04-02 23:54:11	1	1	4
9	2019-04-03 23:54:11	0	2	2
3	2019-04-04 23:54:11	2	4	4
4	2019-04-05 23:54:11	3	4	3
2	2019-04-06 23:54:11	0	4	0
5	2019-04-07 23:54:11	0	0	1
7	2019-04-08 23:54:11	0	0	1
6	2019-04-09 23:54:11	2	4	0
10	2019-04-10 23:54:11	0	0	4
1	2019-04-02 23:54:12	1	1	4
1	2019-04-02 23:54:13	1	1	4

VIEW3: readyToRetrieve

create view readyToRetrieve
as
select USEREmotionMatch.userId,USEREmotionMatch.timeStamp,FinalEmotion as emotion,NeuronConnection.connectome
FROM USEREmotionMatch INNER JOIN NeuronConnection
ON USEREmotionMatch.userId=NeuronConnection.userID AND USEREmotionMatch.timeStamp=NeuronConnection.timeStamp;

OUTPUT:



```
-- view to show all emotions and neuron connections together which can be used to retrieve data easily
create view readyToRetrieve
as
select USEREmotionMatch.userId,USEREmotionMatch.timeStamp,FinalEmotion as emotion,NeuronConnection.connectome
FROM USEREmotionMatch INNER JOIN NeuronConnection
ON USEREmotionMatch.userId=NeuronConnection.userID AND USEREmotionMatch.timeStamp=NeuronConnection.timeStamp;

select * FROM readyToRetrieve;
```

userId	timeStamp	emotion	connectome
1	2019-04-02 23:54:11	1	a10_d37_b45_k34
9	2019-04-03 23:54:11	2	z104_e7_k645_h9409
3	2019-04-04 23:54:11	4	r67_v543_m3213_h874_o762
4	2019-04-05 23:54:11	3	u5342_y763_h76_j8_p89_h32_t12
2	2019-04-06 23:54:11	0	i37_s54_b32_l405_m83
5	2019-04-07 23:54:11	0	b621_w762_v42_k34
7	2019-04-08 23:54:11	0	b90_j723_h981_c404
6	2019-04-09 23:54:11	0	x762_w93_i981_q91
10	2019-04-10 23:54:11	0	q987_b712_p8215_z83013

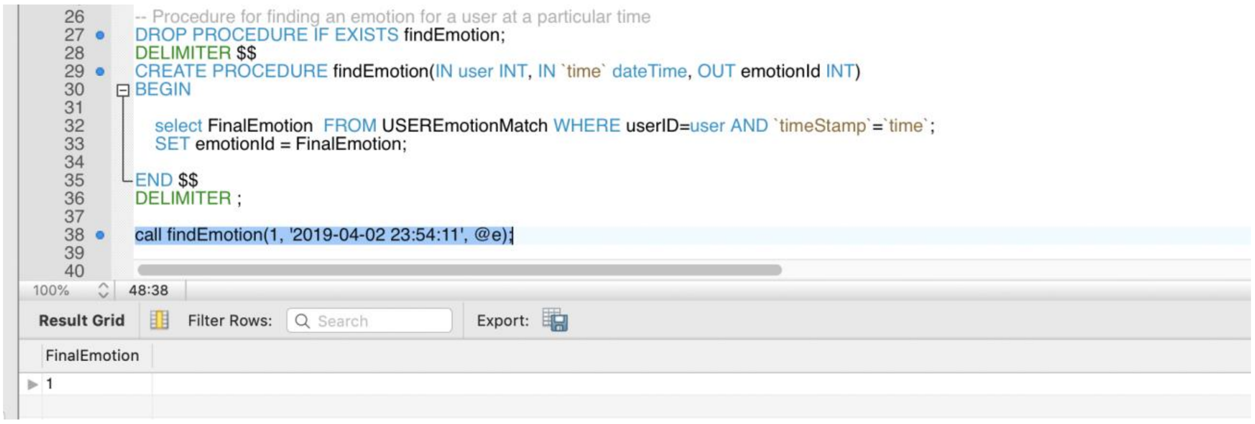
10. Stored Procedures

Stored Procedure 1: findEmotion

This Procedure is used for finding an emotion for a user at a particular time

```
DELIMITER $$
CREATE PROCEDURE findEmotion (IN user INT, IN `time` dateTime, OUT emotionId INT)
BEGIN
        select FinalEmotion FROM USEREmotionMatch WHERE userID=user AND `timeStamp`=`time`;
        SET emotionId = FinalEmotion;

END $$
DELIMITER;
```

OUTPUT:



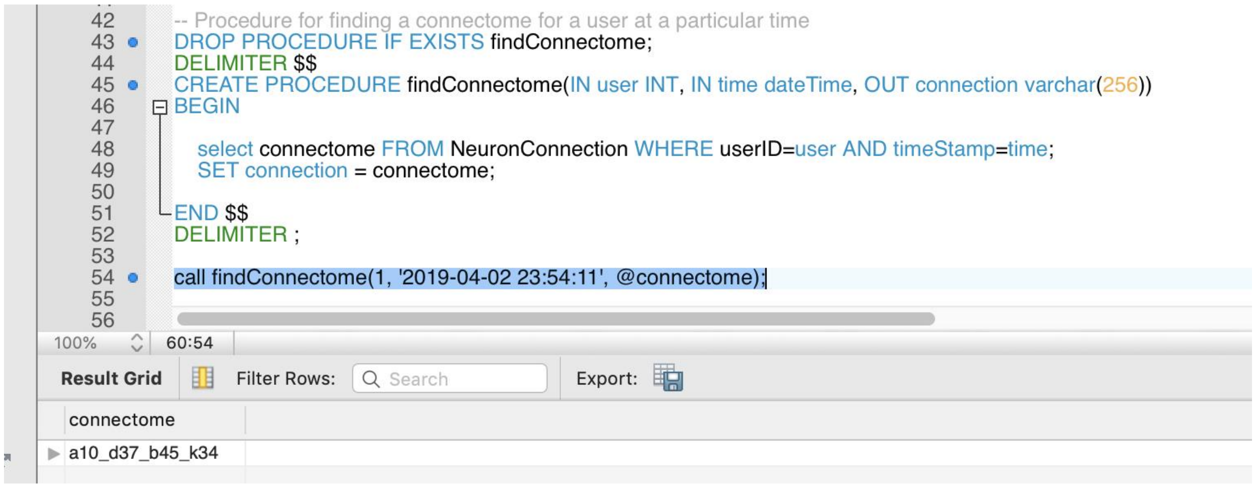
```
-- Procedure for finding an emotion for a user at a particular time
DROP PROCEDURE IF EXISTS findEmotion;
DELIMITER $$
CREATE PROCEDURE findEmotion(IN user INT, IN `time` dateTime, OUT emotionId INT)
BEGIN

   select FinalEmotion  FROM USEREmotionMatch WHERE userID=user AND `timeStamp`=`time`;
   SET emotionId = FinalEmotion;

END $$
DELIMITER ;

call findEmotion(1, '2019-04-02 23:54:11', @e);
```

FinalEmotion
1

Stored Procedure 2: findConnectome
This Procedure is used for finding an emotion for a user at a particular time

```
DELIMITER $$
CREATE PROCEDURE findConnectome (IN user INT, IN time dateTime, OUT connection varchar (256))
BEGIN
            select connectome FROM NeuronConnection WHERE userID=user AND timeStamp=time;
            SET connection = connectome;

END $$
DELIMITER;
```

OUTPUT:



```
-- Procedure for finding a connectome for a user at a particular time
DROP PROCEDURE IF EXISTS findConnectome;
DELIMITER $$
CREATE PROCEDURE findConnectome(IN user INT, IN time dateTime, OUT connection varchar(256))
BEGIN

    select connectome FROM NeuronConnection WHERE userID=user AND timeStamp=time;
    SET connection = connectome;

END $$
DELIMITER ;

call findConnectome(1, '2019-04-02 23:54:11', @connectome);
```

connectome
a10_d37_b45_k34

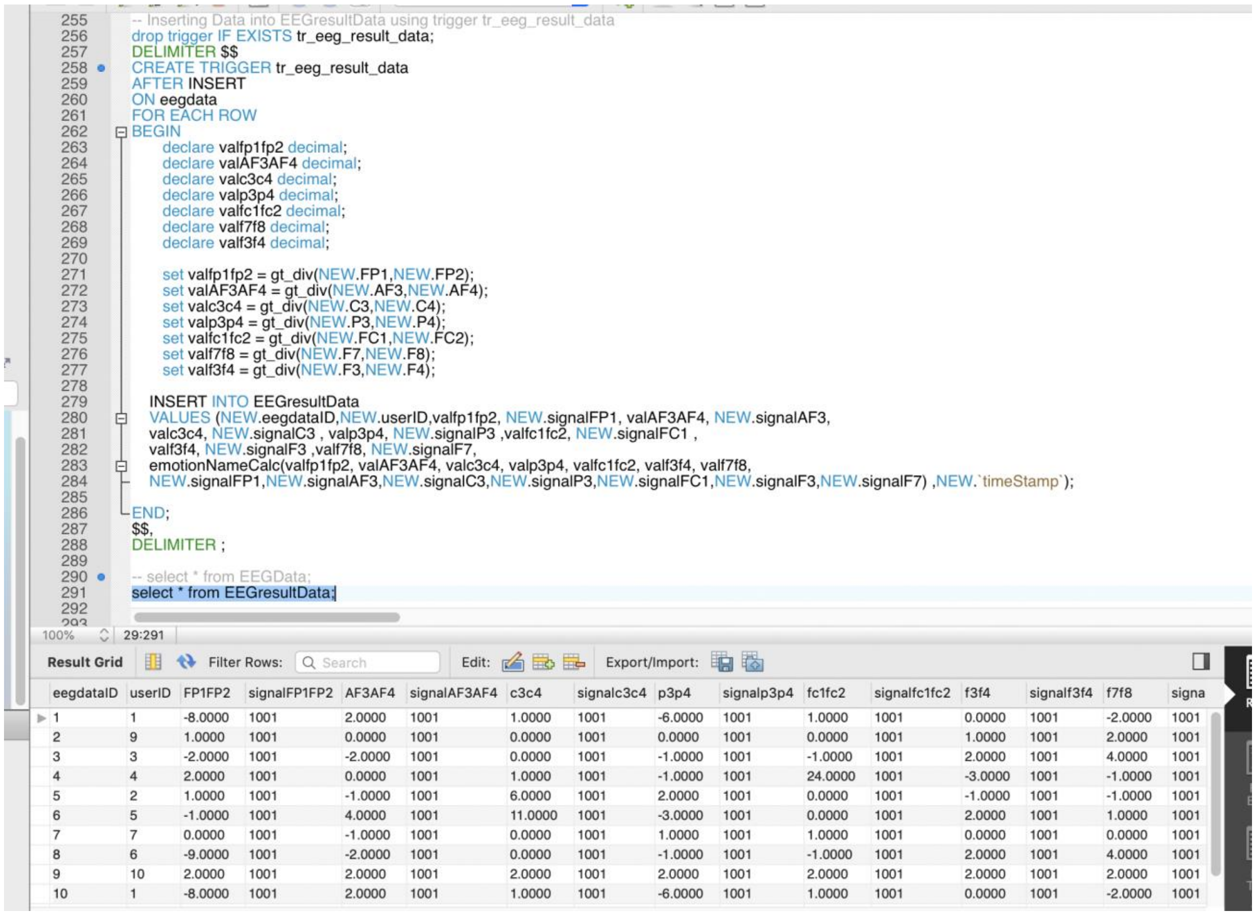
11. Triggers

Trigger 1: This trigger(tr_eeg_result_data) is used to insert data into EEGresultData table and this is triggered after each row is inserted in EEGData.
Functions used in this trigger: gt_div, emotionNameCalc

```
DELIMITER $$
CREATE TRIGGER tr_eeg_result_data
AFTER INSERT
ON eegdata
FOR EACH ROW
BEGIN
      declare valfp1fp2 decimal;
      declare valAF3AF4 decimal;
      declare valc3c4 decimal;
      declare valp3p4 decimal;
      declare valfc1fc2 decimal;
      declare valf7f8 decimal;
      declare valf3f4 decimal;

      set valfp1fp2 = gt_div(NEW.FP1,NEW.FP2);
      set valAF3AF4 = gt_div(NEW.AF3,NEW.AF4);
      set valc3c4 = gt_div(NEW.C3,NEW.C4);
      set valp3p4 = gt_div(NEW.P3,NEW.P4);
```

```
        set valfc1fc2 = gt_div(NEW.FC1,NEW.FC2);
        set valf7f8 = gt_div(NEW.F7,NEW.F8);
        set valf3f4 = gt_div(NEW.F3,NEW.F4);

            INSERT INTO EEGresultData
    VALUES (NEW.eegdataID,NEW.userID,valfp1fp2, NEW.signalFP1, valAF3AF4, NEW.signalAF3,
    valc3c4, NEW.signalC3 , valp3p4, NEW.signalP3 ,valfc1fc2, NEW.signalFC1 ,
    valf3f4, NEW.signalF3 ,valf7f8, NEW.signalF7,
    emotionNameCalc(valfp1fp2, valAF3AF4, valc3c4, valp3p4, valfc1fc2, valf3f4, valf7f8,
    NEW.signalFP1,NEW.signalAF3,NEW.signalC3,NEW.signalP3,NEW.signalFC1,NEW.signalF3,NEW.signalF7)
,NEW.`timeStamp`);

END;
$$,
DELIMITER ;
```

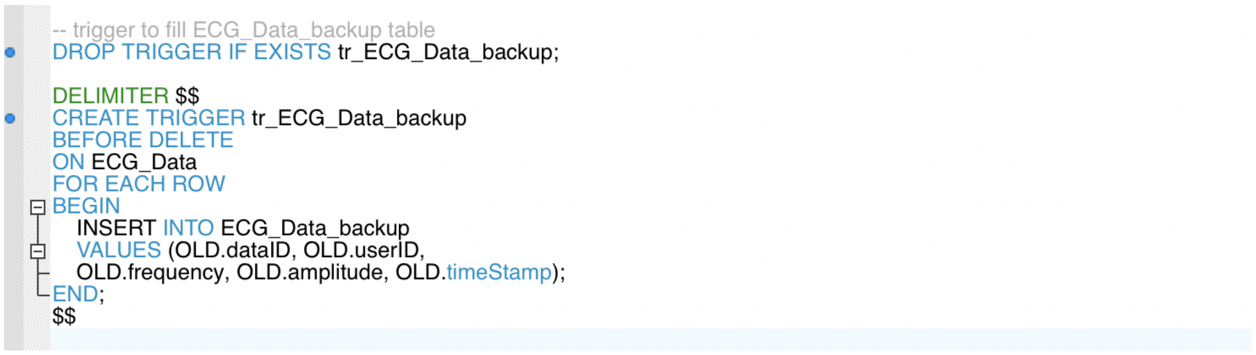
OUTPUT:



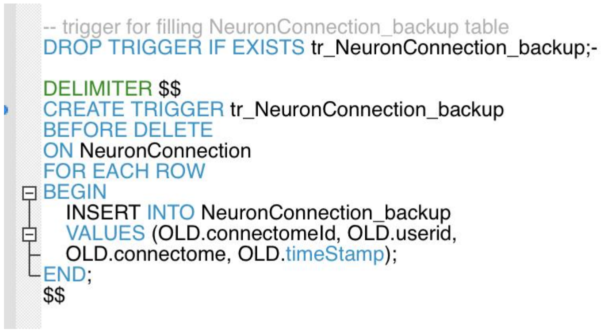
```
-- Inserting Data into EEGresultData using trigger tr_eeg_result_data
drop trigger IF EXISTS tr_eeg_result_data;
DELIMITER $$
CREATE TRIGGER tr_eeg_result_data
AFTER INSERT
ON eegdata
FOR EACH ROW
BEGIN
      declare valfp1fp2 decimal;
      declare valAF3AF4 decimal;
      declare valc3c4 decimal;
      declare valp3p4 decimal;
      declare valfc1fc2 decimal;
      declare valf7f8 decimal;
      declare valf3f4 decimal;

      set valfp1fp2 = gt_div(NEW.FP1,NEW.FP2);
      set valAF3AF4 = gt_div(NEW.AF3,NEW.AF4);
      set valc3c4 = gt_div(NEW.C3,NEW.C4);
      set valp3p4 = gt_div(NEW.P3,NEW.P4);
      set valfc1fc2 = gt_div(NEW.FC1,NEW.FC2);
      set valf7f8 = gt_div(NEW.F7,NEW.F8);
      set valf3f4 = gt_div(NEW.F3,NEW.F4);

   INSERT INTO EEGresultData
   VALUES (NEW.eegdataID,NEW.userID,valfp1fp2, NEW.signalFP1, valAF3AF4, NEW.signalAF3,
   valc3c4, NEW.signalC3 , valp3p4, NEW.signalP3 ,valfc1fc2, NEW.signalFC1 ,
   valf3f4, NEW.signalF3 ,valf7f8, NEW.signalF7,
   emotionNameCalc(valfp1fp2, valAF3AF4, valc3c4, valp3p4, valfc1fc2, valf3f4, valf7f8,
   NEW.signalFP1,NEW.signalAF3,NEW.signalC3,NEW.signalP3,NEW.signalFC1,NEW.signalF3,NEW.signalF7) ,NEW.`timeStamp`);

END;
$$,
DELIMITER ;

-- select * from EEGData;
select * from EEGresultData;
```

eegdataID	userID	FP1FP2	signalFP1FP2	AF3AF4	signalAF3AF4	c3c4	signalc3c4	p3p4	signalp3p4	fc1fc2	signalfc1fc2	f3f4	signalf3f4	f7f8	signa
1	1	-8.0000	1001	2.0000	1001	1.0000	1001	-6.0000	1001	1.0000	1001	0.0000	1001	-2.0000	1001
2	9	1.0000	1001	0.0000	1001	0.0000	1001	0.0000	1001	0.0000	1001	1.0000	1001	2.0000	1001
3	3	-2.0000	1001	-2.0000	1001	0.0000	1001	-1.0000	1001	-1.0000	1001	2.0000	1001	4.0000	1001
4	4	2.0000	1001	0.0000	1001	1.0000	1001	-1.0000	1001	24.0000	1001	-3.0000	1001	-1.0000	1001
5	2	1.0000	1001	-1.0000	1001	6.0000	1001	2.0000	1001	0.0000	1001	-1.0000	1001	-1.0000	1001
6	5	-1.0000	1001	4.0000	1001	11.0000	1001	-3.0000	1001	0.0000	1001	2.0000	1001	1.0000	1001
7	7	0.0000	1001	-1.0000	1001	0.0000	1001	1.0000	1001	1.0000	1001	0.0000	1001	0.0000	1001
8	6	-9.0000	1001	-2.0000	1001	0.0000	1001	-1.0000	1001	-1.0000	1001	2.0000	1001	4.0000	1001
9	10	2.0000	1001	2.0000	1001	2.0000	1001	2.0000	1001	2.0000	1001	2.0000	1001	2.0000	1001
10	1	-8.0000	1001	2.0000	1001	1.0000	1001	-6.0000	1001	1.0000	1001	0.0000	1001	-2.0000	1001

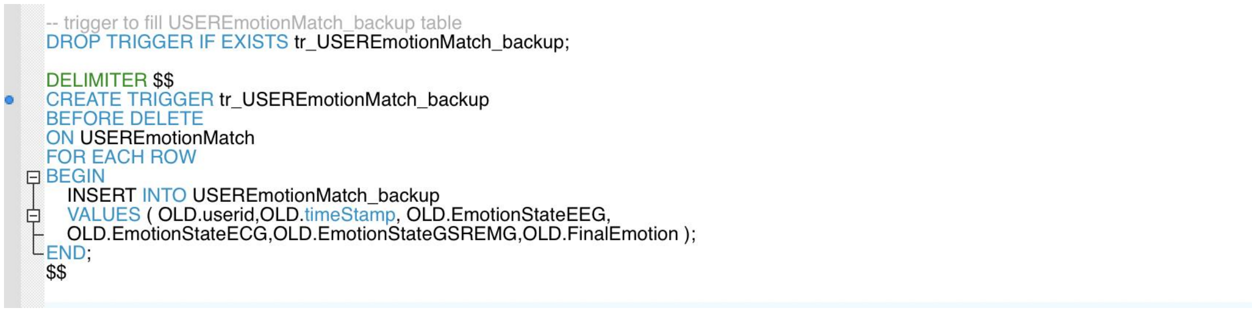
Trigger 2: This trigger (tr_ECG_Data_backup) is used to insert data into ECG_Data_backup table and this is triggered after each row is inserted in ECG_Data.


```
-- trigger to fill ECG_Data_backup table
DROP TRIGGER IF EXISTS tr_ECG_Data_backup;

DELIMITER $$
CREATE TRIGGER tr_ECG_Data_backup
BEFORE DELETE
ON ECG_Data
FOR EACH ROW
BEGIN
    INSERT INTO ECG_Data_backup
    VALUES (OLD.dataID, OLD.userID,
    OLD.frequency, OLD.amplitude, OLD.timeStamp);
END;
$$
```

Trigger 3: This trigger (tr_NeuronConnection_backup) is used to insert data into NeuronConnection_backup table and this is triggered after each row is inserted in NeuronConnection.

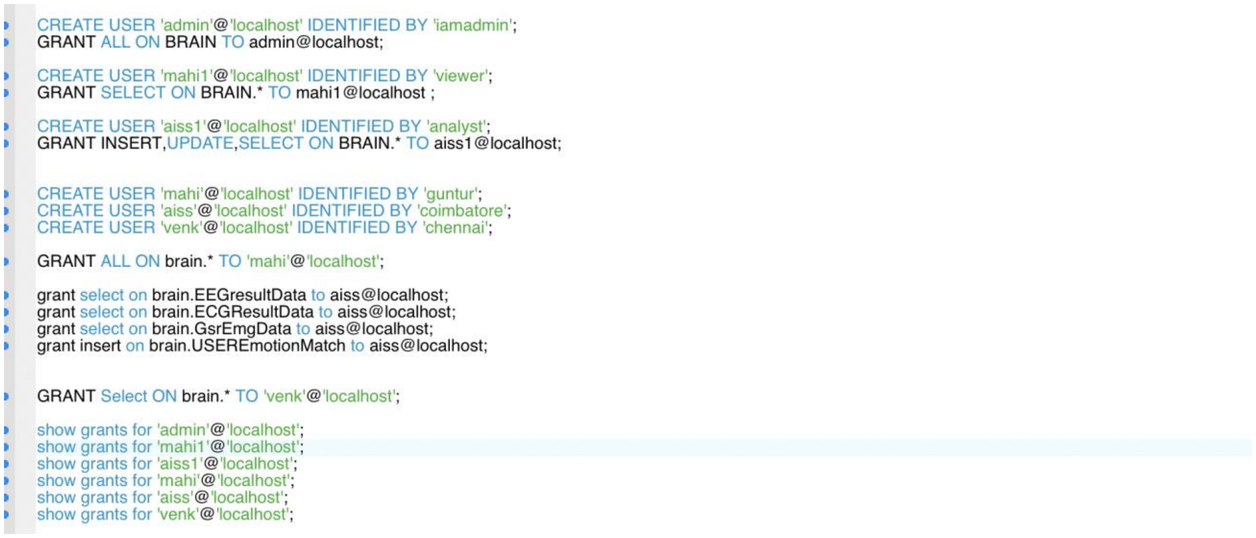

```
-- trigger for filling NeuronConnection_backup table
DROP TRIGGER IF EXISTS tr_NeuronConnection_backup;-

DELIMITER $$
CREATE TRIGGER tr_NeuronConnection_backup
BEFORE DELETE
ON NeuronConnection
FOR EACH ROW
BEGIN
    INSERT INTO NeuronConnection_backup
    VALUES (OLD.connectomeId, OLD.userid,
    OLD.connectome, OLD.timeStamp);
END;
$$
```

Trigger 4: This trigger (tr_USEREmotionMatch_backup) is used to insert data into USEREmotionMatch_backup table and this is triggered after each row is inserted in USEREmotionMatch.


```
-- trigger to fill USEREmotionMatch_backup table
DROP TRIGGER IF EXISTS tr_USEREmotionMatch_backup;

DELIMITER $$
CREATE TRIGGER tr_USEREmotionMatch_backup
BEFORE DELETE
ON USEREmotionMatch
FOR EACH ROW
BEGIN
    INSERT INTO USEREmotionMatch_backup
    VALUES ( OLD.userid,OLD.timeStamp, OLD.EmotionStateEEG,
    OLD.EmotionStateECG,OLD.EmotionStateGSREMG,OLD.FinalEmotion );
END;
$$
```

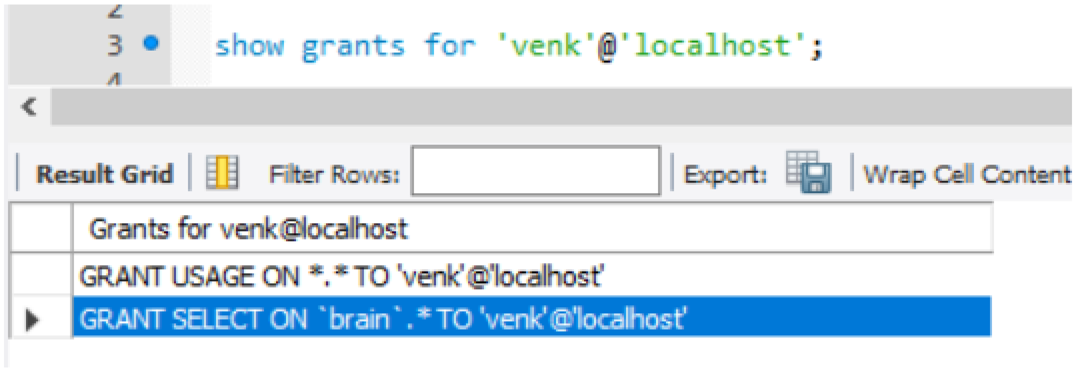
12. Privileges For Users

In the database we have created many users. Admin is a user who have all the access to all the tables on database. Also there is a user mahi1 who have only select privilege on database. There is a user aiss1 who have insert, update, select access on database. Also three different users are created who have only limited access to some tables.



```
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'iamadmin';
GRANT ALL ON BRAIN TO admin@localhost;

CREATE USER 'mahi1'@'localhost' IDENTIFIED BY 'viewer';
GRANT SELECT ON BRAIN.* TO mahi1@localhost ;

CREATE USER 'aiss1'@'localhost' IDENTIFIED BY 'analyst';
GRANT INSERT,UPDATE,SELECT ON BRAIN.* TO aiss1@localhost;


CREATE USER 'mahi'@'localhost' IDENTIFIED BY 'guntur';
CREATE USER 'aiss'@'localhost' IDENTIFIED BY 'coimbatore';
CREATE USER 'venk'@'localhost' IDENTIFIED BY 'chennai';

GRANT ALL ON brain.* TO 'mahi'@'localhost';

grant select on brain.EEGresultData to aiss@localhost;
grant select on brain.ECGResultData to aiss@localhost;
grant select on brain.GsrEmgData to aiss@localhost;
grant insert on brain.USEREmotionMatch to aiss@localhost;


GRANT Select ON brain.* TO 'venk'@'localhost';

show grants for 'admin'@'localhost';
show grants for 'mahi1'@'localhost';
show grants for 'aiss1'@'localhost';
show grants for 'mahi'@'localhost';
show grants for 'aiss'@'localhost';
show grants for 'venk'@'localhost';
```

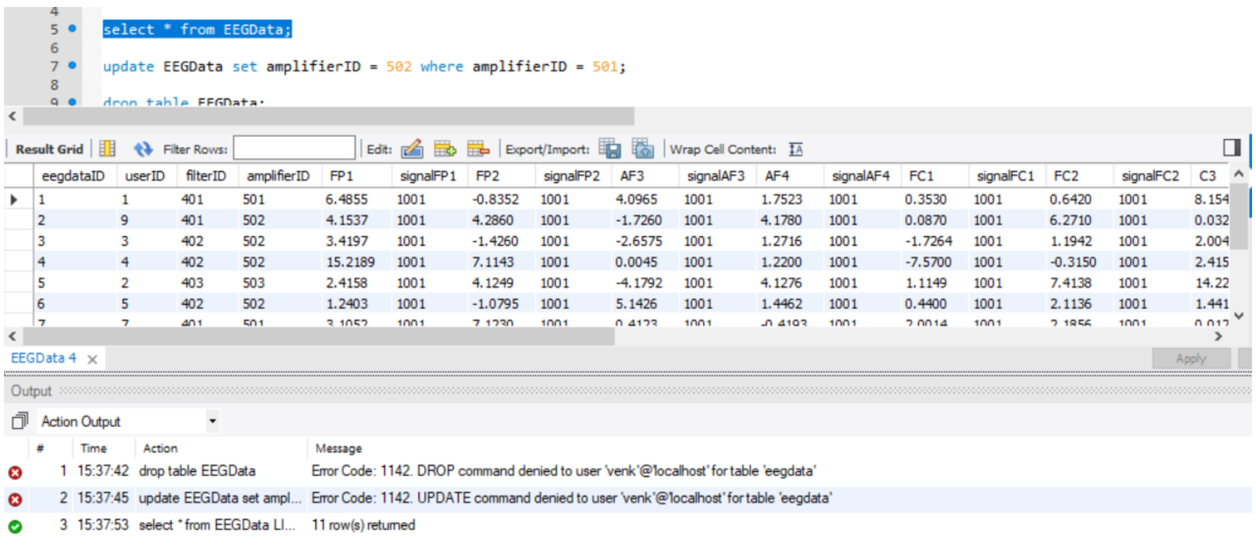
Show grants displays the grants available for a particular database user



```
show grants for 'venk'@'localhost';
```

```
Grants for venk@localhost
GRANT USAGE ON *.* TO 'venk'@'localhost'
GRANT SELECT ON `brain`.* TO 'venk'@'localhost'
```

When logged in with database user as ‘Venk’ and when he tries to Drop OR Update table then the system shows message that he is not authorized to do that action while allows him to use the Select query which he is allowed to do.



```sql
select * from EEGData;

update EEGData set amplifierID = 502 where amplifierID = 501;

drop table EEGData;
```

eegdataID	userID	filterID	amplifierID	FP1	signalFP1	FP2	signalFP2	AF3	signalAF3	AF4	signalAF4	FC1	signalFC1	FC2	signalFC2	C3
1	1	401	501	6.4855	1001	-0.8352	1001	4.0965	1001	1.7523	1001	0.3530	1001	0.6420	1001	8.154
2	9	401	502	4.1537	1001	4.2860	1001	-1.7260	1001	4.1780	1001	0.0870	1001	6.2710	1001	0.032
3	3	402	502	3.4197	1001	-1.4260	1001	-2.6575	1001	1.2716	1001	-1.7264	1001	1.1942	1001	2.004
4	4	402	502	15.2189	1001	7.1143	1001	0.0045	1001	1.2200	1001	-7.5700	1001	-0.3150	1001	2.415
5	2	403	503	2.4158	1001	4.1249	1001	-4.1792	1001	4.1276	1001	1.1149	1001	7.4138	1001	14.22
6	5	402	502	1.2403	1001	-1.0795	1001	5.1426	1001	1.4462	1001	0.4400	1001	2.1136	1001	1.441
7	7	401	501	3.1052	1001	7.1230	1001	0.4123	1001	-0.4193	1001	2.0014	1001	2.1856	1001	0.012

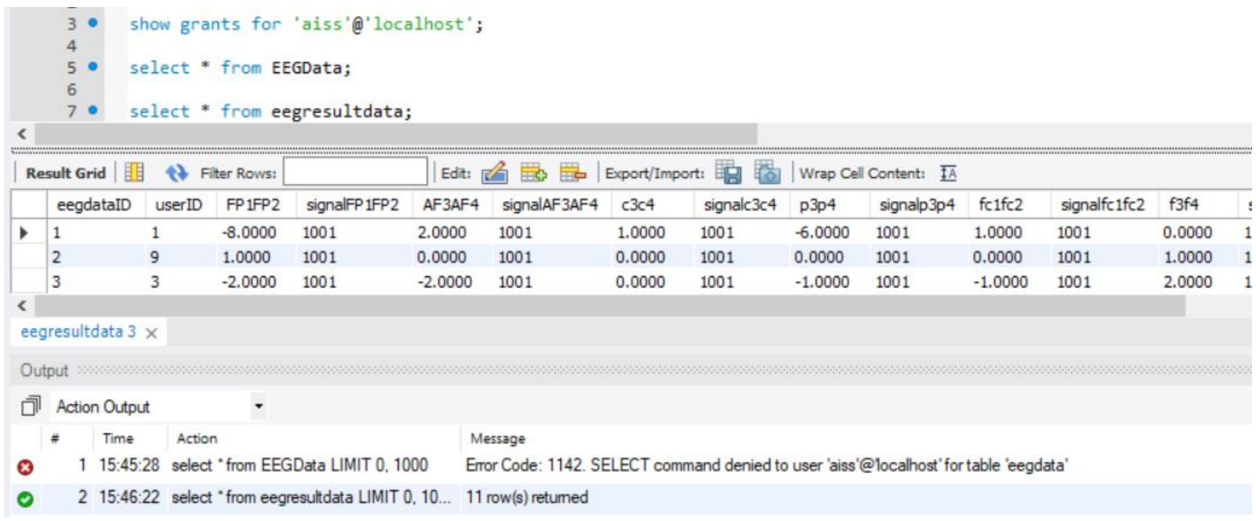
EEGData 4

Output

Action Output

#	Time	Action	Message
1	15:37:42	drop table EEGData	Error Code: 1142. DROP command denied to user 'venk'@'localhost' for table 'eegdata'
2	15:37:45	update EEGData set ampl...	Error Code: 1142. UPDATE command denied to user 'venk'@'localhost' for table 'eegdata'
3	15:37:53	select * from EEGData LI...	11 row(s) returned

When logged in with database user as 'aiss' and when she tries to select a table which she is not authorized, the system restricts the user from doing that action and displays a message that the command is denied while allows to run select command on the table she is authorized to.



```sql
show grants for 'aiss'@'localhost';

select * from EEGData;

select * from eegresultdata;
```

eegdataID	userID	FP1FP2	signalFP1FP2	AF3AF4	signalAF3AF4	c3c4	signalc3c4	p3p4	signalp3p4	fc1fc2	signalfc1fc2	f3f4
1	1	-8.0000	1001	2.0000	1001	1.0000	1001	-6.0000	1001	1.0000	1001	0.0000
2	9	1.0000	1001	0.0000	1001	0.0000	1001	0.0000	1001	0.0000	1001	1.0000
3	3	-2.0000	1001	-2.0000	1001	0.0000	1001	-1.0000	1001	-1.0000	1001	2.0000

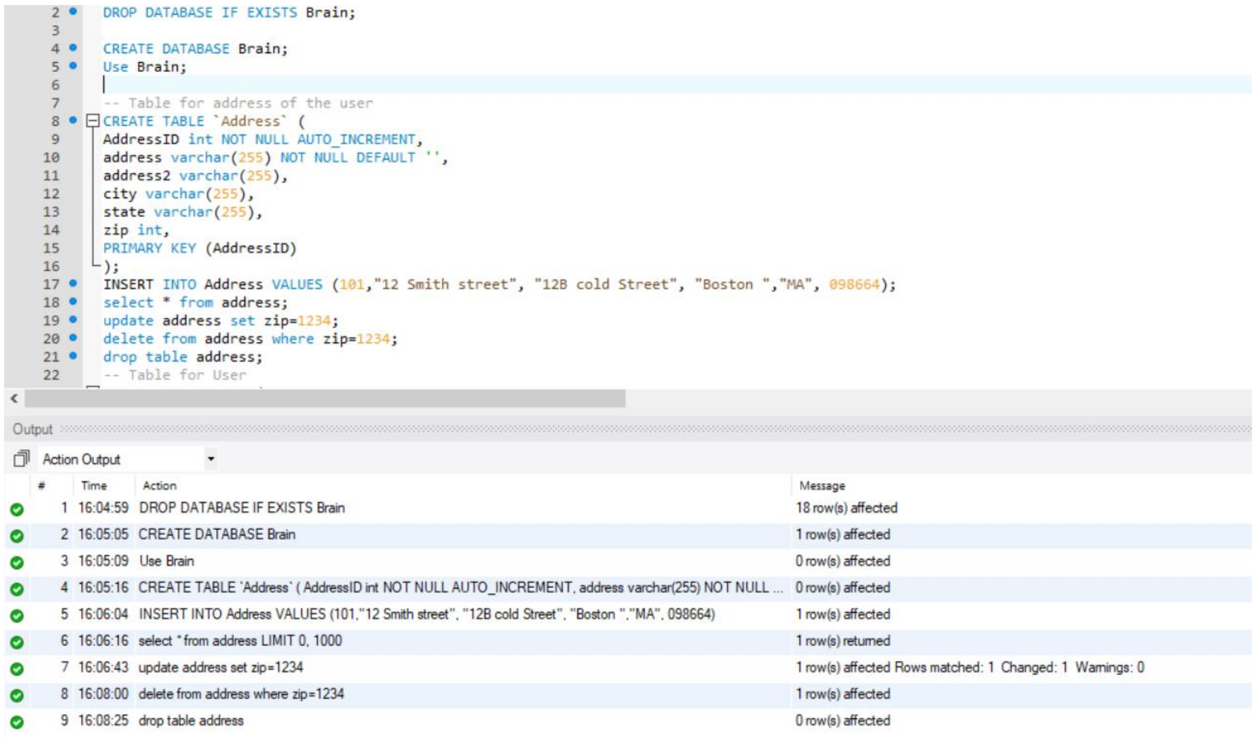
eegresultdata 3

Output

Action Output

#	Time	Action	Message
1	15:45:28	select * from EEGData LIMIT 0, 1000	Error Code: 1142. SELECT command denied to user 'aiss'@'localhost' for table 'eegdata'
2	15:46:22	select * from eegresultdata LIMIT 0, 10...	11 row(s) returned

While the system allows all kinds of commands when logged in with user as mahi.



```
DROP DATABASE IF EXISTS Brain;

CREATE DATABASE Brain;
Use Brain;

-- Table for address of the user
CREATE TABLE `Address` (
AddressID int NOT NULL AUTO_INCREMENT,
address varchar(255) NOT NULL DEFAULT '',
address2 varchar(255),
city varchar(255),
state varchar(255),
zip int,
PRIMARY KEY (AddressID)
);
INSERT INTO Address VALUES (101,"12 Smith street", "12B cold Street", "Boston ","MA", 098664);
select * from address;
update address set zip=1234;
delete from address where zip=1234;
drop table address;
-- Table for User
```

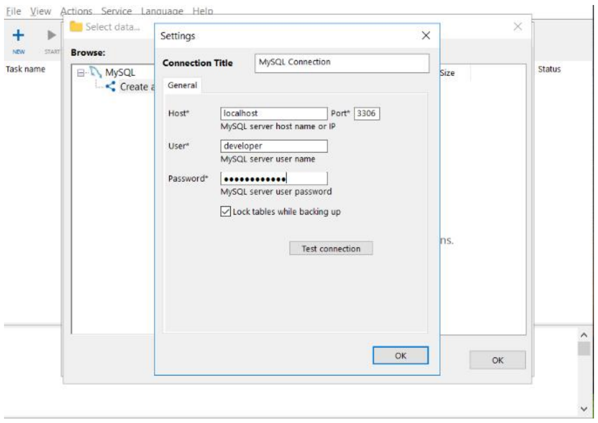
```
#  Time      Action                                                                                         Message
1  16:04:59  DROP DATABASE IF EXISTS Brain                                                                  18 row(s) affected
2  16:05:05  CREATE DATABASE Brain                                                                          1 row(s) affected
3  16:05:09  Use Brain                                                                                      0 row(s) affected
4  16:05:16  CREATE TABLE `Address` ( AddressID int NOT NULL AUTO_INCREMENT, address varchar(255) NOT NULL ...  0 row(s) affected
5  16:06:04  INSERT INTO Address VALUES (101,"12 Smith street", "12B cold Street", "Boston ","MA", 098664)  1 row(s) affected
6  16:06:16  select * from address LIMIT 0, 1000                                                            1 row(s) returned
7  16:06:43  update address set zip=1234                                                                    1 row(s) affected Rows matched: 1  Changed: 1  Warnings: 0
8  16:08:00  delete from address where zip=1234                                                             1 row(s) affected
9  16:08:25  drop table address                                                                             0 row(s) affected
```

17. Schedulers

The below table shows the number of off-peak hours available for performing Full back-up

Night	Off-Peak Start	Off-Peak End	Off-Peak Hours
Monday	12:00 am	5 :00 am	5
Tuesday	12:00 am	5 :00 am	5
Wednesday	12:00 am	5 :00 am	5
Thursday	12:00 am	5 :00 am	5
Friday	12:00 am	5 :00 am	5
Saturday	10:00 pm	8:00 am	10
Sunday	8:00pm	5:00 am	9

For our project the database should be backed up every day whenever the brain examination is performed. So the back ups are performed at the end of the day from 1:00 am to 5:00 am on week days and from 10:00 pm to 8:00 on Saturday and from 8:00pm to 5:00 am on Sundays.



Settings

Connection Title: MySQL Connection

General

Host*: localhost Port*: 3306
MySQL server host name or IP

User*: developer
MySQL server user name

Password*: ••••••••••••
MySQL server user password

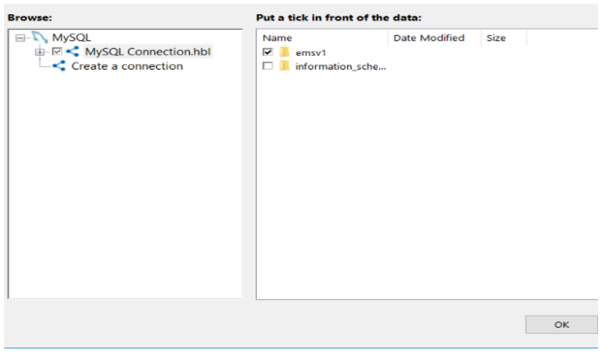
☑ Lock tables while backing up

Test connection

OK

Figure 5 Backup_User login

+—


Browse:

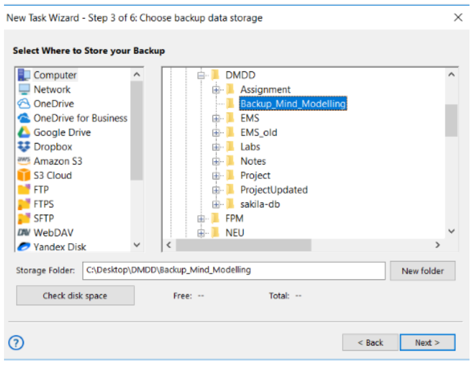
- MySQL
  - ☑ MySQL Connection.hbl
  - Create a connection

Put a tick in front of the data:

Name	Date Modified	Size
☑ emsv1		
☐ information_sche...		

OK

Figure 6 Selecting Database


New Task Wizard - Step 3 of 6: Choose backup data storage

Select Where to Store your Backup

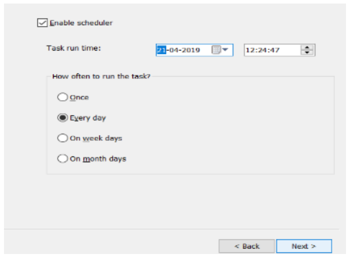
Storage Folder: C:\Desktop\DMDD\Backup_Mind_Modelling

New folder

Check disk space Free: -- Total: --

< Back Next >

Figure 7Selecting Destination


☑ Enable scheduler

Task run time: 21-04-2019 12:24:47

How often to run the task?

○ Once
● Every day
○ On week days
○ On month days

< Back Next >

Figure 8 scheduler

Incremental Back-up

Since the observation of the living human is continuous an incremental backup for every 2 hours is performed.

18. Conclusion

Thus, we are able to collect all the memories and store them in a database and are able to retrieve when needed.