Федеральное государственное бюджетное образовательное учреждение высшего образования

«Костромской государственный университет»

(КГУ)

Институт автоматизированных систем и технологий

Кафедра информационных систем и технологий

Направление подготовки: 09.03.02

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту по дисциплине «Базы данных»

(«Управление данными»)

на тему: «Эмоциональный интеллект»



Выполнил: студент группы 15-ИСбо-2

Манукянц А.Э.

Проверила: Прядкина Н. О.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_

Кострома

2018

**Оглавление**

[**Введение**](#_1fob9te) **3**

[**Анализ предметной области и определение требований к системе**](#_3znysh7) **3**

[**Проектирование реляционной БД**](#_2et92p0) **7**

[**ERD**](#_tyjcwt) **7**

[**Схема реляционной БД**](#_1t3h5sf) **8**

[**Разработка серверной части системы**](#_3j2qqm3) **9**

[**Реализация полученной системы в среде выбранной СУБД (PostgreSQL)**](#_1y810tw) **9**

[**Реализация средств обеспечения целостности БД**](#_2xcytpi) **9**

[**Триггеры**](#_1gf8i83) **10**

[**Хранимые процедуры (функции)**](#_upglbi) **12**

[**Заключение**](#_2koq656) **15**

[**Список использованной литературы**](#_3jtnz0s) **15**

# Введение

Тренды на интеллектуальную обработку данных не сбавляют оборотов, а специалисты Data Science пытаются выловить максимум полезной информации из любых данных. Перспективным направлением является анализ биометрических показателей человека. Основываясь на эти тезисы я придумал приложение, которое поможет сделать людей счастливее.

Работа приложения будет основана на сборе информации об эмоциональном состоянии человека во времени, анализе, выведении отчетов и подготовке советов для его улучшения.

В данной курсовой работе была выбрана предметная область “Эмоционального интеллекта” ввиду актуальности проблемы.

# Анализ предметной области и определение требований к системе

**Цель**

Улучшить чувственно-эмоциональную часть жизни людей.

1. Упростить наблюдение за эмоциями и чувствами.
2. Дать советы по управлению эмоциями и чувствами, которые будут работать.
3. Увеличить уровень эмоциональной грамотности населения.

**Предложенное решение**

Реализовать мобильное приложение и сайт с доступом к базе данных, на которой будут храниться записи пользователей.

Первый шаг – создание инструмента мониторинга. Уже это, возможность видеть своё эмоциональное состояние во времени, позволит лучше понимать себя и принимать решения по улучшению своей жизни.  
Зайдя на сайт, и зарегистрировавшись, пользователь сможет, наподобие дневника, оставлять записи о своем самочувствии с комментариями, прикреплением места и времени, а также ссылкой на конкретный объект или событие, вызывающее у него чувство.  
Большинству пользователей будет лениво заполнять такой дневник каждый день. Потому данные будут собираться с малой группы заинтересованных людей. Для реализации приложения необходим удобный минималистичный интерфейс с выбором основных типов чувств и созданием записей “в один клик”. Для этого в базе данных нужны дополнительные сущности с эмоциями, цветами, картинками, их представляющими. Пользователю должно быть доступно создание сущностей, не связанных неразрывно друг с другом. Это позволит сделать инструмент мониторинга за своими эмоциями гибким.

Дальнейшее развитие проекта будет сконцентрировано на разработке методов анализа данных, улучшении пользовательского интерфейса и оптимизации базы данных.

**Критерии оценки эффективности решения**

1. Изменения в результатах тестов на эмоциональную грамотность до и после использования приложения
2. Количество постоянных пользователей приложения (дольше месяца)
3. Соотношение положительных и отрицательных отзывов

**Данные**

|  |  |  |  |
| --- | --- | --- | --- |
| **Категория** | **Сведения** | **Объём** | **Тенденция развития в месяц** |
| Чувство | 1. Название\* 2. Цвет | 40 | 0 |
| Цвет | 1. Название\* 2. Код цвета HEX\* | 20 | 0 |
| Пользователь | 1. Имя 2. Hash-пароля\* 3. Электронная почта\* 4. Дата регистрации\* | 10000 | + 1000 |
| Объект чувства | 1. Пользователь\* 2. Название\* 3. Тег | Количество пользователей \* 100 | + 1000\*100 |
| Испытываемое чувство | 1. Пользователь\* 2. Чувство\* 3. Объект чувства 4. Интенсивность 5. Время\* 6. Тег | Количество пользователей \* 500 | + 1000\*500 |
| Событие | 1. Пользователь\* 2. Название\* 3. Время\* 4. Место 5. Описание 6. Чувство до события 7. Чувство после события 8. Тег | Количество пользователей \* 100 | + 1000\*100 |
| Тег | 1. Пользователь\* 2. Название\* 3. Цвет 4. Описание | Количество пользователей \* 10 | +1000\*10 |

**Пользователи**

|  |  |  |  |
| --- | --- | --- | --- |
| **Группа** | **Задачи** | **Количество** | **Тенденция развития** |
| Администратор | Доступ на создание новых пользователей, чтение и модификацию всех категорий | 1 | 0 |
| Пользователь | Доступ на чтение и модификацию всех записей, принадлежащих данному пользователю | 10000 | + 1000 |

**Ограничения**

**Условия**

* Параметры *интенсивность* у категорий **испытываемое чувство** не может быть меньше 1 и больше 10.
* Параметры *логин* и *почта* уникальны для каждого пользователя.
* Параметр *почта* должен соответствовать почтовому формату.
* У **события** параметр *времени чувства/эмоции до события* не может быть позднее или равно параметру *времени чувства/эмоции* после события.

**Связи**

* **Тег** может быть привязан к любой пользовательской записи.
* У одной пользовательской записи может быть несколько тегов.
* Один тег может быть привязан к нескольким записям.
* У одного **объекта чувства** может быть несколько испытываемых чувств. Даже в одинаковое время.
* У **испытываемого чувства** может быть указан только один **объект чувства**
* Каждому **чувству** соответствует **цвет**.
* **Цвета**, соответствующие **чувствам** не повторяются.

**Обязательность**

* Обязательные поля категорий отмечены символом «\*».

**Действия**

* Параметр *чувство* в категории **испытываемое чувство** может выбираться из выпадающего списка записей категории **чувство.**
* При регистрации нового пользователя параметр *дата регистрации* заполняется текущей датой.
* При создании **испытываемого чувства**, если время не указано, автоматически выставляется настоящее время записи.
* **Теги** пользователя доступны для добавления к любой записи, принадлежащей ему (**объект чувства, испытываемое чувство, событие**). При добавлении тега к записи, пользователь имеет возможность создать новый **тег**.

# Проектирование реляционной БД

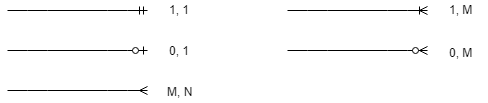
# ERD

# 

*Рисунок 1. ERD*

**Описание нотации**

Для графического представления ERD была выбрана нотация Мартина. Согласно данной нотации, сущность изображается в виде прямоугольника, содержащего её имя, выражаемое существительным. Имя сущности должно быть уникальным в рамках одной модели. Связь изображается линией, которая связывает две сущности, участвующие в отношении. Степень конца связи указывается графически, множественность связи изображается в виде «вилки» на конце связи. Модальность связи также изображается графически — необязательность связи помечается кружком на конце связи.



*Рисунок 2. Нотация Мартина*

# Схема реляционной БД

* Color (id, name, code )
* Feeling (id, *color\_id*, name)
* User (id, login, name, pass\_hash, email, registration\_date)
* FeelingObject (id, *user\_id*, name)
* ActualFeeling (id, *user\_id*, *feeling\_id*, *feeling\_object\_id*, intensity, time)
* Event (id, *user\_id*, *feeling\_before\_id*, *feeling\_after\_id*, name, time, place, description)
* Tag (id, *user\_id*, *color\_id*, name)
* FeelingObjectTag (feeling\_object\_id, tag\_id)
* ActualFeelingTag (actual\_feeling\_id, tag\_id)
* EventTag (event\_id, tag\_id)

Данная схема не является нормализованной, так как *user\_id* дублируется лишний раз в зависимых сущностях. Это сделано для упрощения работы с бизнес-логикой и реализации сценариев: создание пользователем не привязанных ни к чему “чувств”, “объектов”, “событий”, “тегов”.  
Такой механизм создания сущностей, позволит пользователю системы чувствовать гибкость в работе с ней и улучшит пользовательский опыт.

# Разработка серверной части системы

# Реализация полученной системы в среде выбранной СУБД (PostgreSQL)

# 

*Рисунок 3. Диаграмма БД в SQL Server*

# Реализация средств обеспечения целостности БД

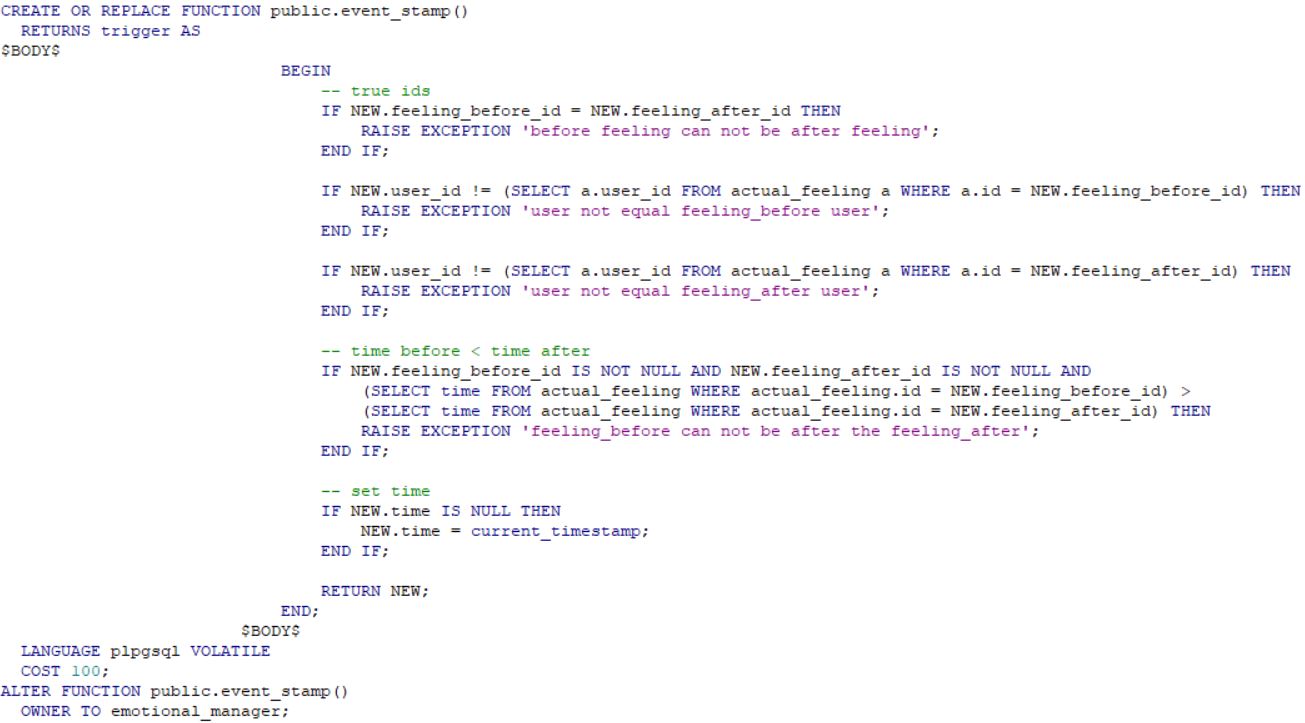
*Ограничения*

|  |  |
| --- | --- |
| Таблица | Ограничения |
| actual\_feeling | * actual\_feeling\_pkey PRIMARY KEY(id) * actual\_feeling\_feeling\_id\_fkey FOREIGN KEY (feeling\_id) * actual\_feeling\_feeling\_object\_id\_fkey FOREIGN KEY (feeling\_object\_id) * actual\_feeling\_user\_id\_fkey FOREIGN KEY (user\_id) * actual\_feeling\_intensity\_check CHECK (intensity > 0 AND intensity <= 10); |
| actual\_feeling\_tag | * actual\_feeling\_tag\_pkey PRIMARY KEY(actual\_feeling\_id, tag\_id); * actual\_feeling\_tag\_id\_fkey FOREIGN KEY (tag\_id) * tag\_actual\_feelingt\_id\_fkey FOREIGN KEY (actual\_feeling\_id) |
| color | * color\_pkey PRIMARY KEY(id) |
| emotional\_user | * emotional\_user\_pkey PRIMARY KEY(id) * emotional\_user\_email\_key UNIQUE(email) |
| event | * event\_pkey PRIMARY KEY(id) * event\_feeling\_after\_id\_fkey FOREIGN KEY (feeling\_after\_id) * event\_feeling\_before\_id\_fkey FOREIGN KEY (feeling\_before\_id) * event\_user\_id\_fkey FOREIGN KEY (user\_id) |
| event\_tag | * event\_tag\_pkey PRIMARY KEY(event\_id, tag\_id) * event\_tag\_id\_fkey FOREIGN KEY (tag\_id) * tag\_event\_id\_fkey FOREIGN KEY (event\_id) |
| feeling | * feeling\_pkey PRIMARY KEY(id) * feeling\_color\_id\_fkey FOREIGN KEY (color\_id) |
| feeling\_object | * feeling\_object\_pkey PRIMARY KEY(id) * feeling\_object\_user\_id\_fkey FOREIGN KEY (user\_id) |
| feeling\_object\_tag | * feeling\_object\_tag\_pkey PRIMARY KEY(feeling\_object\_id, tag\_id) * feeling\_object\_tag\_id\_fkey FOREIGN KEY (tag\_id) * tag\_feeling\_object\_id\_fkey FOREIGN KEY (feeling\_object\_id) |
| tag | * tag\_pkey PRIMARY KEY(id) * tag\_color\_id\_fkey FOREIGN KEY (color\_id) * tag\_user\_id\_fkey FOREIGN KEY (user\_id) |

# Триггеры

В БД используется несколько триггеров, которые осуществляют контроль над вставкой и обновлением данных:

* Триггер, осуществляющий контроль над данными, которые пытаются добавить или изменить в таблице «event». С помощью данного триггера реализованы:
  + проверка принадлежности сущностей одному пользователю
  + проверка того, что время предыдущего чувства меньше времени последующего
  + автоматическая установка времени события настоящим моментом, если пользователь не указал своего времени

**

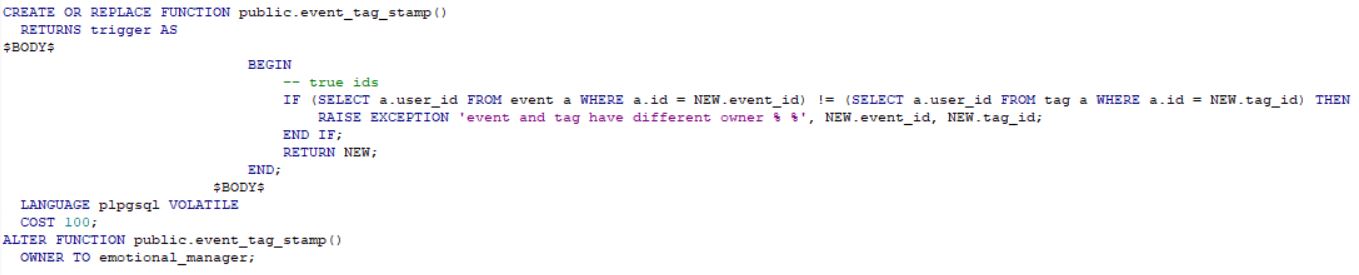
*Рисунок 4. Триггер на вставку/изменение в таблице event*

* Триггер, осуществляющий контроль над данными, которые пытаются добавить или изменить в таблице actual\_feeling. С помощью данного триггера реализованы:
  + проверка принадлежности feeling\_object тому же пользователю, что и создающаяся запись actual\_feeling
  + автоматическая установка времени события настоящим моментом, если пользователь не указал своего времени

# 

*Рисунок 5. Триггер на вставку/изменение в таблице actual\_feeling*

* Триггер, осуществляющий контроль над данными, которые пытаются добавить или изменить в таблицы event\_tag, actual\_feeling\_tag, feeling\_object\_tag. С помощью данного триггера реализована:
  + проверка принадлежности тега и сущности, к которой он прикрепляется одному пользователю

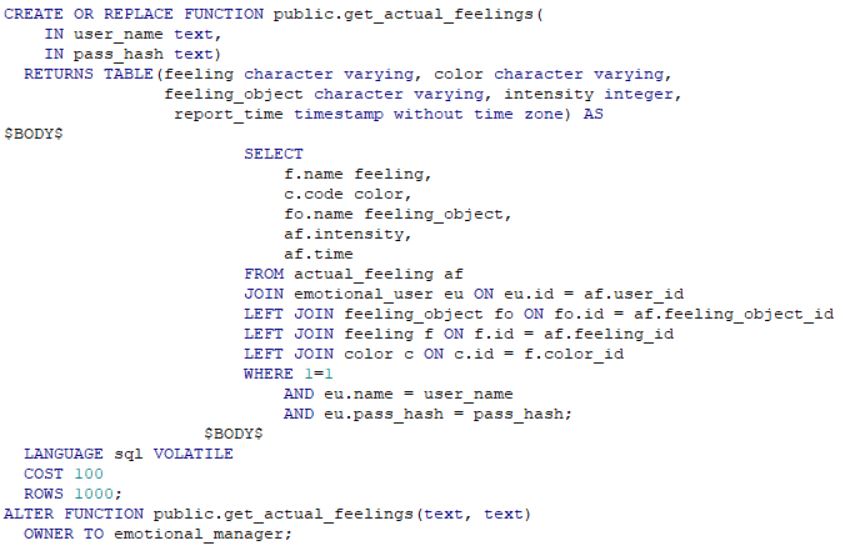


*Рисунок 6. Триггер на вставку/изменение в таблице event\_tag*

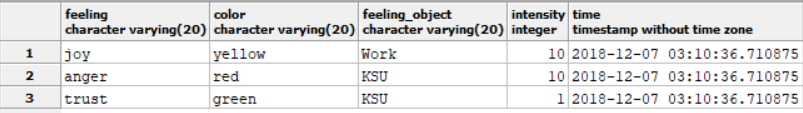
# Хранимые процедуры (функции)

Для упрощения работы бизнес-логики и более быстрого взаимодействия с базой данных были добавлены хранимые процедуры(в PostgreSQL их заменяют функции), реализующие самые распространенные сценарии работы – формирование отчётов.

* Для получения чувств пользователя используется функция «get\_actual\_feelings». Функция принимает два параметра: имя пользователя и хеш его пароля. Происходит объединение всех таблиц, связанных с испытываемым чувством, и отбираются те записи, где имя и пароль совпадают с введенными.



*Рисунок 7. Функция get\_actual\_feeling(name,pass\_hesh)*

**

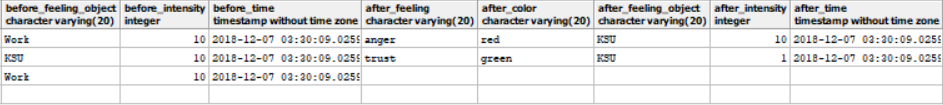
*Рисунок 8. Результат выполнения get\_actual\_feeling(name,pass\_hesh)*

* Для получения событий пользователя используется функция «get\_events». Функция принимает два параметра: имя пользователя и хеш его пароля. Происходит объединение всех таблиц, связанных с событием, и отбираются те записи, где имя и пароль совпадают с введенными.

# 

*Рисунок 9. Функция get\_events(name,pass\_hesh)*

# 



*Рисунок 10. Результат выполнения get\_events(name,pass\_hesh)*

# Заключение

Данный курсовой проект помог закрепить и углубить знания по проектированию и реализации базы данных. Я проанализировал предметную область, разработал ERD и реляционную модель, создал базу данных, ограничения, триггеры и хранимые процедуры.   
Благодаря выполнению данного курсового проекта я так же получил готовую базу данных, которая будет применена в дальнейшем развитии проекта  
“Emotional Intelligence”.

# Список использованной литературы

* Прядкина, Н.О. Учебно-методическое пособие «Базы данных. Методические указания по выполнению курсового проекта», 2008
* Нотация Мартина [Электронный ресурс] / Режим доступа:  
  <https://studopedia.ru/17_52272_notatsiya-martina.html>
* Документация PostgreSQL:   
  https://www.postgresql.org/docs/9.4/index.html
* Триггеры [Электронный ресурс] / Режим доступа:   
  https://postgrespro.ru/docs/postgresql/9.6/plpgsql-trigger