Не забудьте удалить комментарии, помеченные таким цветом

Экзаменационный проект   
по дисциплине  
Проектирование баз данных

студента гр. **M34381**

**Гарипов Эмиль Исмагилович**

исправьте падеж на родительный

по теме

**Фитнес трекер**

Вы можете редактировать формулировку темы, чтобы она лучше отражала суть работы

# Описание проекта

Описание предметной области с кратким описанием неочевидных сущностей и атрибутов.

Есть пользователи, про них известны параметры тела(вес, рост), есть архив записей параметров тела в какие-то дни, есть база продуктов(информация о продукте по типу калорийность, содержание белков, углеводов и жиров), есть список тренировок. Продукты и тренировки создаются пользователями. У каждого продукта/тренировки ровно один владелец, который может менять продукт/тренировку. Так же есть активность пользователя(приемы пищи c информацией о потребленных продуктах, и выполнение тренировок). Пользователи могут создавать фитнес-стратегии(про каждый день есть информация какая еда должна быть съедена в приемах пищи, и необходимые тренировки). Также у пользователей есть список избранных фитнес-стратегий и только одна которой он придерживается в данный момент. Продукт, тренировка и стратегия могут быть выключены создалетем, ??? Хочу ли флаг enabled?

# Построение отношений

В результате предварительного проектирования были выделены следующие отношения:

* **Users -** пользователи, содержит информацию о пользователе - id, имя(ФИО), возраст, пол, вес и рост(на данный момент), email и id фитнес-стратегии которой сейчас придерживается пользователь, у каждого пользователя она только одна. У пользователей не могут совпадать email-ы.
* **Logs** - записи о параметрах тела пользователей. Содержит информацию о теле пользователя(вес, рост), дате записи, и id пользователя.
* **Foods** - продукты, добавленные пользователями. Содержит информацию о названии продукта, калориях, белках, жирах и углеводах(на 100г продукта). У каждого продукта есть владелец - пользователь который его добавил.
* **Meals** - приемы пищи, хранится id о пользователя(который принимал пищу), название приема пищи и время. Прием пищи включает в себя несколько продуктов.
* **Workouts** - тренировки, хранится информация о тренировке - ее название, подробное описание, и id владельца(человека который добавил тренировку).
* **Strategies** - стратегии, у каждой стратегии есть название и id владельца. Каждая стратегия состоит из нескольких дней.
* **Days** - дни(из которых состоят стратегии), хранится информация о названии дня, и его номер по счету в стратегии, и id стратегии которой он принадлежит. В каждом дне есть список тренировок которые необходимо выполнить и список продуктов которые нужно употребить. Про каждый продукт в дне известно в какой по счету прием пищи нужно его употребить
* **Activities** - физические активности, содержит информацию о том кто ее выполнял, какое время, когда и название активности. В ходе активности могла выполняется только одна из существующих тренировок. Хранится id этой тренировки.
* **MealsFoods** - показывает какие продукты входят в приемы пищи. Про продукт в приеме пищи известно его количество и вес продукта(например: творог, 200г, две штуки).
* **DaysFoods** - показывает какие продукты должны быть употреблены в конкретном дне стратегии. Про каждый продукт указан номер приема пищи, в котором продукт должен быть употреблен, количество и вес).
* **DaysWorkouts** - показывает какие тренировки должны быть выполнены в дне стратегии. Про каждую тренировку хранится ее номер в дне.
* users\_strats?

## Отношение **Users**

Атрибуты:

* user\_id
* user\_name - имя пользователя
* age - возраст
* sex - пол
* weight - вес пользователя(указанный пользователем последний раз)
* height - высота пользователя
* email - адрес почты пользователя
* chosen\_strategy - фитнес-стратегия которой в данный момент придерживается пользователь

Функциональные зависимости:

* user\_id -> user\_name
* user\_id -> age
* user\_id -> sex
* user\_id -> weight
* user\_id -> height
* user\_id -> email
* user\_id -> chosen\_strategy
* email -> user\_id

Ключи:

* user\_id
* email

### **Нормализация**

* **1НФ**: нет повторяющихся групп, аттрибуты атормары и есть ключ
* **2НФ**: отношение в 2НФ, так как: оно находится в 1НФ, неключевые атрибуты зависят от ключа в целом(нету составных ключей)
* **3НФ**: отношение в 3 НФ, так как: оно находится 2НФ, неключевые атрибуты непосредственно зависят от ключей
* **НФБК**: отношение в НФБК, так как: оно в 3НФ, и ключи не перекрываются
* **4НФ**: отношение в 4НФ, так как: отношение находится в НФБК и есть простой ключ =>(Теорема Дейта-Фейгина 2) отношение в 4НФ
* **5НФ**: отношение в 5НФ, так как: отношение находится в 3НФ и все ключи простые =>(Теорема Дейта-Фейгина 1) отношение в 5НФ

## Отношение **Logs**

Атрибуты:

* log\_id
* user\_id - id пользователя, информация о котором хранится в записи
* log\_datetime - дата и время в которую была сделана запись
* weight - вес пользователя в момент времени log\_datetime
* height - рост пользователя в момент времени log\_datetime

Функциональные зависимости:

* log\_id -> user\_id
* log\_id -> log\_datetime
* log\_id -> weight
* log\_id -> height

Ключи:

* log\_id

## Отношение **Foods**

Атрибуты:

* food\_id
* food\_title - название продукта
* cals - кол-во калорий на 100г продукта
* carbs - информация о содержании углеводов на 100г продукта
* protein - информация о содержании белков на 100г продукта
* fat - информация о содержании жиров на 100г продукта
* owner\_id - id создателя(пользователь который добавил продукт) продукта

Функциональные зависимости:

* food\_id -> food\_title
* food\_id -> cals
* food\_id -> carbs
* food\_id -> protein
* food\_id -> fat
* food\_id -> owner\_id

Ключи:

* food\_id

## Отношение **Meals**

Атрибуты:

* meal\_id
* meal\_title - название приема пищи
* meal\_datetime - дата и время приема пищи
* user\_id - id пользователя который принимал пищу

Функциональные зависимости:

* meal\_id -> meal\_title
* meal\_id -> meal\_datetime
* meal\_id -> user\_id
* user\_id, meal\_datetime -> meal\_id

Ключи:

* meal\_id
* user\_id, meal\_datetime

### **Нормализация**

* **1НФ**: нет повторяющихся групп, аттрибуты атормары и есть ключ
* **2НФ**: отношение в 2НФ, так как: оно находится в 1НФ, и неключевые атрибуты зависят от ключа в целом
* **3НФ**: отношение в 3 НФ, так как: оно находится 2НФ, неключевые атрибуты непосредственно зависят от ключей
* **НФБК**: отношение в НФБК, так как: оно в 3НФ, и ключи не перекрываются
* **4НФ**: отношение в 4НФ, так как: отношение находится в НФБК и есть простой ключ =>(Теорема Дейта-Фейгина 2) отношение в 4НФ
* **5НФ**: отношение в 5НФ, так как: причина

## Отношение **Workouts**

Атрибуты:

* workout\_id
* workout\_title - название тренировки
* description - подробное описание тренировки
* owner\_id - id владельца

Функциональные зависимости:

* workout\_id -> workout\_title
* workout\_id -> description
* workout\_id -> owner\_id

Ключи:

* workout\_id

## Отношение **Strategies**

Атрибуты:

* strategy\_id
* strategy\_title - название фитнес-стратегии
* owner\_id - id владельца

Функциональные зависимости:

* strategy\_id -> strategy\_title
* strategy\_id -> owner\_id

Ключи:

* strategy\_id

## Отношение **Days**

Атрибуты:

* day\_id
* day\_title - название дня
* number - порядковый номер дня в стратегии
* strategy\_id - стратегия, содержимым которой является день

Функциональные зависимости:

* day\_id -> day\_title
* day\_id -> day\_number
* day\_id -> strategy\_id

Ключи:

* day\_id

## Отношение **Activities**

Атрибуты:

* activity\_id
* activity\_title - название активности
* activity\_datetime - время и дата активности
* duration - длительность активности
* workout\_id - id тренировки, выполняемой в ходе активности
* user\_id - id пользователя, который выполняет активность

Функциональные зависимости:

* activity\_id -> activity\_title
* activity\_id -> activity\_datetime
* activity\_id -> duration
* activity\_id -> workout\_id
* activity\_id -> user\_id
* activity\_datetime, user\_id -> activity\_id

Ключи:

* activity\_id
* activity\_datetime, user\_id

## Отношение **MealsFoods**

Атрибуты:

* meal\_id
* food\_id
* amount - количество употребленного продукта
* weight - вес употребленного продукта в граммах

Функциональные зависимости:

* meal\_id, food\_id -> amount
* meal\_id, food\_id -> weight

Ключи:

* meal\_id, food\_id

### **Нормализация**

* **1НФ**: нет повторяющихся групп, аттрибуты атормары и есть ключ
* **2НФ**: отношение в 2НФ, так как: оно находится в 1НФ, и неключевые атрибуты зависят от ключа в целом
* **3НФ**: отношение в 3 НФ, так как: оно находится 2НФ, неключевые атрибуты непосредственно зависят от ключей
* **НФБК**: отношение в НФБК, так как: оно в 3НФ, и ключи не перекрываются(ключ всего один)
* **4НФ**: отношение в 4НФ, так как: причина
* **5НФ**: отношение в 5НФ, так как: причина

## Отношение **DaysFoods**

Атрибуты:

* day\_id
* food\_id
* amount - количество продукта которое должно быть употреблено
* weight - вес продукта который должен быть употреблен

Функциональные зависимости:

* day\_id, food\_id -> amount
* day\_id, food\_id -> weight

Ключи:

* day\_id, food\_id

## Отношение **DaysWorkouts**

Атрибуты:

* day\_id
* workout\_id
* number - порядковый номер тренировки в дне (чтобы понимать в каком порядке делать тренировки в дне, если их несколько)

Функциональные зависимости:

* day\_id, workout\_id -> number

Ключи:

* day\_id, workout\_id

# Модель сущность-связь

Картинка.

Чем крупнее шрифт, тем лучше. Использование шрифта меньше 10pt не допускается.

Используйте для связей линии, параллельные осям координат и имеющие минимальное число углов. Избегайте пересечения линий.

# Физическая модель

Картинка. Расположение таблиц в физическом модели должно соответствовать расположению сущностей в модели сущность-связь.

Чем крупнее шрифт, тем лучше. Использование шрифта меньше 10pt не допускается.

Используйте для ссылок линии, параллельные осям координат и имеющие минимальное число углов. Избегайте пересечения линий.

При построении физической модели использовалось следующее отображение доменов в типы:

| Домен | Тип |
| --- | --- |
| Домен 1 | Тип 1 |

# Определения таблиц

Для реализации проекта использовалась СУБД название и версия. Определения таблиц и их индексов приведено в файле ddl.sql.

Если ограничения определяются отдельно от таблиц, то приведите обоснование, почему.

Определения индексов должны быть сконцентрированы в конце файла. Для индексов в комментариях указывается запросы, для ускорения которых они предназначены.

Все перечисленные файлы должны быть загружены на Google Drive в тот же каталог, в котором находится этот отчёт.

# Тестовые данные

Скрипт для добавления тестовых данных приведен в файле examples.sql.

Тестовые данные должны быть типичными для предметной области, в том числе, демонстрировать нетривиальные случаи.

# Запросы на получение данных

В рамках проекта были реализованы следующие запросы:

* <<название>> — описание
* <<название>> — описание

Название запроса может быть как на русском, так и на английском языке. В файле указывается в комментарии перед запросом.

Для реализации запросов были созданы вспомогательные представления:

* <<название>> — описание
* <<название>> — описание

Запросы на получение данных и вспомогательные представления приведены в файле selects.sql.

# Запросы на изменение данных

В рамках проекта были реализованы следующие запросы:

* <<название>> — описание
* <<название>> — описание

Название запроса может быть как на русском, так и на английском языке. В файле указывается в комментарии перед запросом.

Запросы на изменение данных, хранимые процедуры и триггеры приведены в файле updates.sql. Для каждого изменяемого запроса и хранимой процедуры в комментарии перед ним должен быть указан минимальный уровень изоляции транзакции для его выполнения.