Филиал Московского Государственного Университета
имени М.В. Ломоносова в городе Ташкенте

Факультет прикладной математики и информатики
Кафедра прикладной математики и информатики

Абдуллаева Евгения Гасановна

КУРСОВАЯ РАБОТА
на тему: «Тема курсовой работы»

по направлению 01.03.02 «Прикладная математика и информатика»

Научный руководитель
к.ф.-м.н., в.н.с. Алисейчик Павел Александрович

«_____» _____ 2020 г.

Ташкент 2020 г.

Тема курсовой работы

Аннотация на русском

Abstract in English (не нужно)

Содержание

Введение **4**

Исследование структуры баз данных систем **5**

Методы и организация работы **6**

Подготовка к переносу сообщений **7**
Выделение сообщений необходимых для переноса 7
Разделение совмещенных сообщений 9
Создание пользователей в удобном для переноса виде 13
Стандартизация имен пользователей 14
Получение идентификаторов курсов, разделов и задач 14
Распределение сообщений по курсам, разделам и задачам 14
Получение сообщений в удобном для переноса виде 14

Перенос сообщений **15**
Добавление комментариев 15
Редактирование комментариев 15

Заключение **16**

Список использованных источников и литературы **17**

Приложения **18**

Введение

За годы сопровождения курса «Практикум на ЭВМ» системой «Форум МГУ» накоплена база сообщений, включающая вопросы студентов и ответы преподавателей по различным этапам и темам данного курса, являющаяся ценным образовательным ресурсом. В связи с созданием системы «МГУ Контест» возникла необходимость переноса базы сообщений в новую систему. Сохранение накопленной базы сообщений ведет как к экономии времени преподавателей, освобождая их от необходимости многократно отвечать на повторяющиеся вопросы, так и студентов, которым не придется ожидать ответа на заданный вопрос, а следовательно способствует развитию обучающей системы.

В настоящий момент качество образования является одним из ключевых факторов определяющих развитие общества, именно благодаря образованию человек приобретает способность свободно и независимо мыслить, имплементировать собственные идеи и эффективно использовать интеллектуальные ресурсы для решения множества практических задач. Преподавание в наши дни представляет собой нить, по которой ученик продвигается посредством исследований и открытий. Следовательно, актуальной задачей является повышение эффективности процесса обучения, а в частности помощь преподавателям в предоставлении студентам обучающего материала.

Исследование структуры баз данных систем

Текст про структуру БД систем «Форум МГУ» и «МГУ Контест»

Методы и организация работы

Текст про Python/Django, MySQL

Подготовка к переносу сообщений

Выделение сообщений необходимых для переноса

Первая задача, которую необходимо выполнить для подготовки к переносу сообщений – определить, какие сообщения следует переносить, так как обязательно соответствие между темами, обсуждаемыми в сообщениях, и содержанием курсов системы «МГУ Контест». Решение задачи осуществляется в два этапа: выделим доски, необходимые для переноса; затем выделим темы и принадлежащие им сообщения, которые будут перенесены. Для этого удобно иметь возможность просматривать сообщения, относящиеся к различным темам и доскам.

Средства управления администратора, предоставляемые Django, не позволяют просматривать текст нескольких сообщений, сгруппированных по какому-либо признаку, поэтому для просмотра сообщений из базы данных системы «Форум МГУ» с указанием их принадлежности к определенной доске и теме используется следующее решение.

Реализована функция `get_messages_view`, которая позволяет получить представление списка сообщений, разделенных по доскам и темам в виде текста с HTML-форматированием. Данная функция принимает в качестве аргументов список досок `all_boards`, список тем `all_topics` и список сообщений `all_messages`, которые необходимо отобразить. В теле функции объявляется переменная `html` типа `str`, изначально пустая. Для каждой доски из списка досок `all_boards` к переменной `html` прибавляется идентификатор и имя доски, заключенные в HTML-теги заголовка первого уровня; переменной `topics` присваивается список тем из `all_topics`, принадлежащих рассматриваемой в текущий момент доске. Для каждой темы из `topics` к переменной `html` прибавляется идентификатор и название темы (определяется как заголовок первого сообщения в теме), заключенные в HTML-теги заголовка второго уровня; переменной `messages` присваивается список сообщений из `all_messages`, относящихся к рассматриваемой в текущий момент теме; сообщения в списке `messages` сортируются по возрастанию даты их написания, определяемой полем `postertime` модели сообщения. Для каждого сообщения из `messages` к переменной `html` прибавляется идентификатор, заголовок, имя автора сообщения, имя изменявшего текст сообщения пользователя, дата написания и непосредственно текст сообщения. Функция возвращает переменную `html`.

Описанная функция вызывается как возвращаемое значение function-based view (представления в функциональном виде) `all_messages`. В качестве аргументов в функцию передаются списки всех существующих в базе досок, всех

тем и всех сообщений. Таким образом данное view предоставляет возможность просмотра сообщений, сгруппированных по принадлежности к доскам и темам в браузере по URL-адресу `<host>:<port>/all_messages`.

База данных системы «Форум МГУ» содержит следующие доски:

1. Практикум на ЭВМ: общие положения
2. Практикум по программированию, 1 курс, 1 семестр
3. Практикум по программированию, 1 курс, 2 семестр
4. Практикум по программированию, 2 курс, 1 семестр
5. Практикум по программированию, 2 курс, 2 семестр
6. Практикум по программированию, 3 курс, 1 семестр
7. Практикум по программированию, 3 курс, 2 семестр
8. Практикум по программированию, 4 курс, 1 семестр
9. 4 курс, 2 семестр
10. Магистратура мех-мат ф-та, практикум
11. Прочее
12. Ассемблер
13. Семинар «Программирование интеллектуальных систем»
14. Системы программирования
15. Спортивное программирование
16. Курсовые и дипломные работы
17. Служебный раздел

База данных системы «МГУ Контест» содержит курсы, соответствующие этапам «Практикума на ЭВМ» начиная от первого семестра первого курса до первого семестра четвертого курса, перечисленные ниже:

1. Простые алгоритмы
2. Дискретная математика
3. Операционные системы

4. Основы ООП
5. Дискретная оптимизация
6. Численные методы, часть 1
7. Численные методы, часть 2

Перенос осуществляется лишь для тех сообщений из базы данных системы «Форум МГУ», которые принадлежат доскам, соответствующим этапам курса «Практикум на ЭВМ» или содержат общие положения проведения «Практикума на ЭВМ». Данному условию соответствуют сообщения, относящиеся к доскам: «Практикум на ЭВМ: общие положения», «Практикум по программированию, 1 курс, 1 семестр», «Практикум по программированию, 1 курс, 2 семестр», «Практикум по программированию, 2 курс, 1 семестр», «Практикум по программированию, 2 курс, 2 семестр», «Практикум по программированию, 3 курс, 1 семестр», «Практикум по программированию, 3 курс, 2 семестр», «Практикум по программированию, 4 курс, 1 семестр», «Прочее». Присвоим выделенное множество досок константе `NECESSARY_BOARDS`. Далее необходимо определить, какие темы, принадлежащие выделенным доскам, будут перенесены.

В результате просмотра сообщений определяются темы, для которых перенос осуществлен не будет. Таковыми являются темы с условиями задач, так как условия задач уже присутствуют в системе «МГУ Контест»; темы, содержащие устаревшую информацию: объявления и распределение задач. Множество тем, включающих сообщения с полезной информацией, присвоим константе `NECESSARY_TOPICS`. Множество сообщений, принадлежащих темам из `NECESSARY_TOPICS` запишем в константу `NECESSARY_MESSAGES`. Таким образом получены необходимые для переноса доски, темы и сообщения.

Разделение совмещенных сообщений

В системе «Форум МГУ» преподаватели вставляли некоторые ответы на вопросы студентов непосредственно в текст вопроса и выделяли текст ответа одним из следующих цветов: голубым, синим или зеленым. При этом форматирование текста осуществлялось с помощью тегов языка разметки BBCode, имеющих следующий вид: `[color=<color_name>]`. В системе «МГУ Контест» существует возможность привязывать ответ к конкретному комментарию, соответственно, возникает задача выделить отдельные сообщения из совмещенных, сохраняя при этом цепочки вопросов и ответов.

Чтобы приступить к решению данной задачи, найдем для начала все совмещенные сообщения. Воспользуемся для поиска модулем re языка Python, предоставляющим функции для работы с регулярными выражениями. Составим шаблон на языке регулярных выражений для поиска вхождений тегов языка разметки BBCode, отвечающих за голубой, синий или зеленый цвет текста:

```
\[color=blue]|\[color=navy]|\[color=green]
```

В константу с именем `COMBINED_MESSAGES` сохраним множество сообщений из `NECESSARY_MESSAGES`, удовлетворяющих условию:

```
if re.search(r'\[color=blue]|\[color=navy]|\[color=green]',
             message.body))
```

Множество же несовмещенных сообщений, полученное вычитанием множества совмещенных сообщений из множества всех необходимых для переноса сообщений, присвоим константе `UNCOMBINED_MESSAGES`.

Для сообщений из `NECESSARY_MESSAGES`, полученных в результате разделения, а также для сообщений, которые в разделении не нуждаются, создадим новую модель данных Message со следующими полями:

- id – содержит положительное целое число – уникальный идентификатор сообщения, является первичным ключом модели;
- parent_msg_id – содержит положительное целое число – id первого сообщения в цепочке, к которой относится данное сообщение;
- author – содержит внешний ключ, указывающий на автора сообщения – объект типа User, где User является встроенной моделью Django;
- text – содержит текст сообщения;
- date_created – содержит положительное целое число – время создания сообщения в формате POSIX-времени, где время определяется как количество секунд, прошедших с полуночи (00:00:00 UTC) 1 января 1970 года;
- subject – текстовое поле, содержащее заголовок сообщения;
- topic – содержит внешний ключ, указывающий на тему, к которой относится данное сообщение;
- board – содержит внешний ключ, указывающий на доску, к которой относится данное сообщение.

Создание промежуточной модели для сообщений позволяет включить в модель сообщения только те поля, которые будут сохранены при переносе, а также сохранить изначальные сообщения.

Реализуем функцию `create_new_messages`, позволяющую разделить совмещенные сообщения и создать разделенные и не нуждавшиеся в разделении сообщения из `NECESSARY_MESSAGES`, как объекты модели Message. Объявим переменную `new_messages` типа `set`, изначально пустую. В это множество будем помещать создаваемые сообщения. Для задания значений уникальных идентификаторов сообщений введем переменную `id_counter` типа `int` с начальным значением равным `1`.

Для каждого сообщения из `UNCOMBINED_MESSAGES` создадим новый объект модели Message, присваивая значения полям создаваемого сообщения следующим образом:

```
Message(
    id=id_counter,
    parent_msg_id=id_counter,
    author=message.id_member,
    text=message.body,
    date_created=message.postertime,
    subject=message.subject,
    topic=message.id_topic,
    board=message.id_board
)
```

и добавим созданное сообщение во множество `new_messages`. При каждой итерации к значению переменной `id_counter` будем прибавлять `1`.

Теперь необходимо разделить совмещенные сообщения `COMBINED_MESSAGES`. Введем специальную переменную-маркер, куда сохраним шаблон на языке регулярных выражений для поиска ответов преподавателей (текста, выделенного одним из упомянутых выше цветов):

```
teacher_message_marker = r'(?<=\[color=blue]).+?(?=\[/color])|' \
                         r'(?<=\[color=navy]).+?(?=\[/color])|' \
                         r'(?<=\[color=green]).+?(?=\[/color])'
```

Для каждого сообщения из `COMBINED_MESSAGES` выясним, начинается ли текст совмещенного сообщения со слов студента (текста, не выделенного цветом). Для этого введем переменную `student_begins` булевского типа, которая принимает значение `True` в случае, если начало сообщения совпадает с шаблоном:

```
\[color=blue]|\[color=navy]|\[color=green]
```

и принимает значение `False` иначе. Затем получим из текста сообщения все части, соответствующие сообщениям студента (таких частей может быть больше одной), используя функцию `split` из модуля re следующим образом:

```
re.split(teacher_message_marker, message.body)
```

В данном случае разделителями для функции `split` являются сообщения преподавателя. Результирующий список сообщений студента сохраним в переменную `student_messages`. В текст сообщений студента, помимо самого сообщения будут записаны так же закрывающие и открывающие теги языка разметки BBCode, заключавшие в себя текст сообщений преподавателя. Для каждого сообщения в `student_messages` удалим из текста сообщения эти теги. Список сообщений преподавателя получим используя функцию `findall` модуля re следующим образом:

```
re.findall(teacher_message_marker, message.body)
```

Сохраним список сообщений преподавателя в переменную `teacher_messages`. Теперь необходимо объединить сообщения студента и преподавателя в один список, сохраняя изначальную последовательность сообщений. Введем переменную `splitted_messages` типа `list`, изначально пустую и переменную `last_index` типа `int` с начальным значением равным `0`. Для каждого индекса списка сообщений студента или списка сообщений преподавателя (выбор списка осуществляется в зависимости от длины списков, выбирается список с минимальной длиной) добавим в конец списка `splitted_messages` сначала сообщение студента с текущим индексом в списке `student_messages`, затем сообщение преподавателя из `teacher_messages` с тем же индексом, если значение переменной `student_begins` равно `True`, иначе добавим сначала сообщение преподавателя из `teacher_messages` с текущим индексом, а затем сообщение студента из `student_messages` с тем же индексом. После добавления двух сообщений в список `splitted_messages` переменной `last_index` присвоим значение текущего индекса. По завершении этого цикла, если в каком-либо из списков `student_messages` или `teacher_messages` есть сообщения с индексами больше `last_index`, то добавим все эти сообщения в конец списка `splitted_messages`. Удалим из списка `splitted_messages` все сообщения, текстом которых является пустая строка, если такие присутствуют. Для каждого индекса из полученного списка сообщений `splitted_messages` создадим объект модели Message для сообщения из `splitted_messages` с этим индексом. Для этого получим уникальный идентификатор автора сообщения и сохраним в переменную `id_member`. Если сообщение с текущим индексом является сообщением преподавателя и поле `modifiedname` данного сообщения не является пустой строкой, то значение `id_member` будет получено как уникальный идентификатор объекта модели TashkentMember, у которого значение поля `realname` равно значению поля

`modifiedname` данного сообщения. Иначе переменной `id_member` будет присвоено значение поля `id_member` текущего сообщения. Создавать новые объекты типа Message будем следующим образом:

```
Message(
    id=id_counter,
    parent_msg_id=id_counter - i,
    author=id_member,
    text=splitted_messages[i],
    date_created=message.postertime + i,
    subject=message.subject,
    topic=message.id_topic,
    board=message.id_board
)
```

Созданное сообщение добавим во множество `new_messages`. При каждой итерации к значению переменной `id_counter` будем прибавлять `1`.

Мы получили множество сообщений `new_messages`, содержащее все сообщения из `NECESSARY_MESSAGES`, где совмещенные сообщения разделены и сохранены как цепочки нескольких последовательных сообщений. Вставим объекты множества `new_messages` в базу данных, используя стандартный метод Django QuerySet API `bulk_create` следующим образом:

```
Message.objects.bulk_create(new_messages)
```

Создание пользователей в удобном для переноса виде

Стандартизация имен пользователей

Получение идентификаторов курсов, разделов и задач

Распределение сообщений по курсам, разделам и задачам

Получение сообщений в удобном для переноса виде

Перенос сообщений

Добавление комментариев

Редактирование комментариев

Заключение

Выводы

Список использованных источников и литературы

[1] Документация Python
https://docs.python.org/3

[2] Документация Django
https://docs.djangoproject.com/en/3.0

[3] Документация MySQL
https://dev.mysql.com/doc/

Приложения

- Инструменты для работы с базой данных системы «Форум МГУ» https://github.com/eugeuie/forum TODO MAKE IT PUBLIC AND DELETE SETTINGS.PY WITH SECRET KEY !!
- Репозиторий проекта «МГУ Контест» https://github.com/ruslanbektashev/contest