ИСПОЛЬЗОВАНИЕ СТРУКУТРИРОВАННОГО ЯЗЫКА ЗАПРОСОВ

Лабораторный практикум по Базам данных, занятие 1

МГУ им. А.А. Кулешова, ФМЕ, 1 курс 2 семестр
заочная форма получения образования
специальность ИПОВС, 2020-2021 учебный год
доцент Сидоренко И.Н.

1 Цель работы

Ознакомиться с возможностями клиентской программы **MySQL**, которая представляет собой утилиту командной строки. Создать с ее помощью базу данных, набор таблиц в ней, заполнить таблицы данными для последующей работы, провести модификацию таблиц.

Содержание работы.

1. Для работы с базой данных вначале необходимо запустить сервер MySQL. После того как сервер MySQL запущен, к нему можно подключиться. Открыть окно командной строки. Вывести список имеющихся баз данных. Команда **SHOW DATABASES**.

2. Изучить набор команд языка SQL, связанный с созданием базы данных, созданием, модификацией структуры таблиц и их удалением, вставкой, модификацией и удалением записей таблиц.

SHOW DATABASES, CREATE DATABASE, SELECT DATABASE(), CREATE TABLE, SHOW TABLES, DESCRIBE, ALTER TABLE, DROP TABLE, DROP DATABASE, INSERT INTO, UPDATE, DELETE

3. Создание базы данных.

Создать *базу данных для хранения сведений о животных для ветеринарной клиники с именем **A_XX_ГГ_YY** , (где XX-номер группы, ГГ-год группы, YY-номер в журнале).*

Убедитесь, что она создана (**SHOW DATABASES**). Сделайте ее текущей.

4. Создание таблиц в базе данных.

*Создать таблицу Животные с именем **A_XX_ГГ_YY_PetName** (XX-номер группы, ГГ-год группы, YY-номер в журнале) со структурой:*

Pet_id (int) unsigned not null primary key	*Pet_name (varchar20)*	*Owner (varchar15)*	*Ves (int)*	*PYear (date)*

5. Сделать активной созданную БД ***A_XX_ГГ_YY*** и посмотреть структуру созданной таблицы ***A_XX_ГГ_YY_PetName***.

6. Изменить структуру таблицы.

Добавить столбец Vid (varchar10) (вид – собака, кошка, птица, змея и т.п.) слева от PYear.

Переименовать (изменить) столбец PYear в столбец PetBirth.

Добавить столбец ID (int) слева от Pet_id (первый).

Посмотреть структуру таблицы.

Удалить столбец ID.

7. Внести данные в таблицу (7-9 записей).

Используйте не менее 3-х различных видов и владельцев (owner) животных. Вводить дату в разном формате: как строка "2012-10-01" или как последовательность чисел 20121001.

Пример:

Pet_id (int)	*Pet_name (varchar20)*	*Owner (varchar15)*	*Ves (int)*	*Vid (varchar10)*	*PYear (date)*
1	*БобикФИО**YY***	*ИвановФИОYY*	*250*	*собакаФИОYY*	*1980*
2	*БарбосФИО**YY***	*ПетровФИОYY*	*230*	*собакаФИОYY*	*2005*

8. Проверить результат заполнения таблиц, написав и выполнив простейший запрос: ***SELECT * FROM имя_таблицы***

9. Выполнить ряд изменений в таблице.

Удалить данные из строки 2.

Добавить столбец Town (varchar5).

Изменить имя и тип столбца Town(alter table).

Переименовать столбец Ves.

Изменить значение Вес в строке 1.

10. Проверить результаты выполнения изменений. *(SELECT)*

Методические указания к выполнению лабораторной работы №1.

База данных –набор сведений, хранящихся упорядоченным образом. Мы рассматриваем реляционные базы данных, которые состоят из таблиц.

Для работы с базой данных необходима **СУБД (система управления базами данных), т.е. программа, которая берет на себя все заботы, связанные с доступом к данным. Она содержит команды, позволяющие создавать таблицы, вставлять в них записи, искать и даже удалять записи**.

Взаимодействие с базой данных осуществляется на **языке**, называемом **SQL** (Structured Query Language — язык структурированных запросов).

MySQL - это открыто распространяемая СУБД, функционирует по модели "клиент/сервер", доступна для загрузки на сайте MySQL.com.

В MySQL для каждой базы данных создается отдельный каталог, а каждой таблице соответствуют три файла.

Клиентская программа MySQL представляет собой утилиту командной строки. Эта программа подключается к серверу по сети.

Для работы с базой данных, вначале необходимо **запустить сервер MySQL**. Если сервер MySQL был сконфигурирован как сервис Windows то запускать и останавливать его можно с помощью компонента Службы панели управления. После того как сервер MySQL запущен**, к нему можно подключиться**.

Просмотр Баз данных.

Для просмотра имеющихся баз данных существует команда **SHOW DATABASES.** Введите ее в строке приглашения и сервер выведет список баз данных, которые были созданы MySQL во время установки.

Примечание. Изначально в MySQL присутствуют только две базы данных: mysql и test. В базах данных information_schema и mysql хранится информация об учетных записях, системный каталог привилегий пользователей СУБД MySQL, региональные настройки и т.д. База данных test является пустой и создается при установке MySQL вместе с системными базами данных information_schema и mysql. База данных mysql является реальной базой данных, а information_schema — виртуальной, именно поэтому в каталоге данных нет подкаталога с соответствующим именем.

Выход из системы.

Для выхода введите **"QUIT"** в приглашении mysql, снова появится приглашение MS-DOS. По окончании работы остановите сервер MySQL.

Создание базы данных в Windows. Команда CREATE DATABASE

Синтаксис команды **CREATE DATABASE** имеет вид:

CREATE DATABASE [IF NOT EXISTS] имя_базы_данных [спецификация_create[,спецификация_create]...]

Команда CREATE DATABASE создает базу данных с указанным именем. Для использования команды необходимо иметь привилегию CREATE для базы данных. Если база данных с таким именем существует, генерируется ошибка.

Спецификация_create:

[DEFAULT] CHARACTER SET имя_набора_символов

[DEFAULT] COLLATE имя_порядка_сопоставления

Опция спецификация_create может указываться для определения характеристик базы данных. Характеристики базы данных сохраняются в файле db.opt, расположенном в каталоге данных. Конструкция CHARACTER SET определяет набор символов для базы данных по умолчанию. Конструкция COLLATION задает порядок сопоставления по умолчанию.

Поскольку изначально в базе нет никаких таблиц, оператор **CREATE DATABASE** только создает подкаталог в каталоге данных MySQL.

(Примечание: Команда заканчивается символом точки с запятой).

Создадим базу данных с именем DB1.

CREATE DATABASE DB1;

Сервер ответит, что запрос обработан, изменилась 1 строка (0.00 сек). База данных была успешно создана.

Определение текущей базы данных.

База данных DB1 уже создана. Для работы с ней, необходимо её "активировать" или "выбрать". Всякий раз при работе с клиентом MySQL необходимо определять, **какая *база данных будет использоваться***.

Определить текущую базу данных можно несколькими способами:

при запуске: mysql> **MySQL DB1;**

- с помощью оператора **USE**: mysql> **USE DB1**;
- с помощью **\u**: mysql**> \u DB1**;

После определения базы данных вводим команду **SELECT DATABASE();** и видим нашу БД.

Таблицы. Создание. Команда CREATE TABLE

Базы данных хранят данные в таблицах. Проще всего таблицы можно представлять себе, как состоящие из строк и столбцов. Каждый столбец определяет данные определенного типа. Строки содержат отдельные записи.

Синтаксис команды **CREATE TABLE** таков:

CREATE [TEMPORARY] TABLE [IF NOT EXISTS] имя
[(спецификация, ...)] [опция, ...]
[[IGNORE | REPLACE] запрос]

Флаг TEMPORARY задает *создание временной таблицы*, существующей в течение текущего сеанса. По завершении сеанса таблица удаляется. Временным таблицам можно присваивать имена других таблиц, делая последние временно недоступными. Спецификатор **IF NOT EXIST** подавляет вывод сообщений об ошибках в случае, если таблица с указанным именем уже существует. Имени таблицы может предшествовать имя базы данных, отделенное точкой. Если это не сделано, таблица будет создана в базе данных, которая установлена по умолчанию.

Разрешается создавать таблицы без столбцов, однако в большинстве случаев спецификация хотя бы одного столбца все же присутствует. Спецификации столбцов и индексов приводятся в круглых скобках и разделяются запятыми.

Формат спецификации следующий:

имя тип[NOT NULL | NULL][DEFAULT значение]
[AUTO_INCREMENT][KEY][ссылка]

Спецификация типа включает название типа и его размерность. По умолчанию столбцы принимают значения NULL. Спецификатор NOT NULL запрещает подобное поведение.

У любого столбца есть значение по умолчанию. Если оно не указано, программа MySQL выберет его самостоятельно. Для столбцов, принимающих значения NULL, значением по умолчанию будет NULL, для строковых столбцов — пустая строка, для численных столбцов — нуль. Изменить эту установку позволяет предложение **DEFAULT**.

Поля-счетчики, создаваемые с помощью флага AUTO_INCREMENT, игнорируют значения по умолчанию, так как в них записываются порядковые номера. Тип счетчика должен быть беззнаковым целым. В таблице может присутствовать лишь одно поле-счетчик. Им не обязательно является первичный ключ. Когда MySQL встречается со столбцом с атрибутом auto_increment, то генерируется новое значение, которое на единицу больше чем наибольшее значение в столбце. Поэтому мы не должны задавать для этого столбца значения, MySQL генерирует их самостоятельно. Из этого также следует, что каждое значение в этом столбце будет уникальным.

Примечание: в MySQL команды и имена столбцов не различают регистр символов, однако имена таблиц и баз данных могут зависеть от регистра в связи с используемой платформой (как в Linux).

Ключевой столбец (primary key)необходим для того, чтобы исключить возможность совпадения данных. Если имеется столбец с уникальными значениями, то можно легко различить две записи. Лучше поручить присваивание уникальных значений самой системе MySQL

После создания таблицы убедимся, что она существует командой

SHOW TABLES;

Просмотр структуры таблицы. DESCRIBE.

Оператор **DESCRIBE ИмяТаблицы**;» позволяет посмотреть структуру таблицы:

Изменение структуры таблицы. ALTER TABLE.

Эта команда позволяет добавлять и удалять столбцы, создавать и уничтожать индексы, переименовывать столбцы и саму таблицу.

ALTER TABLE table_name alter_spec;

Основные преобразования, выполняемые оператором ALTER TABLE

· **ADD create definition [FIRST | AFTER column_name]** Добавление нового столбца. Параметр **create_definition** представляет собой название нового столбца и его тип. Конструкция FIRST добавляет новый столбец перед столбцом column_name. Конструкция AFTER добавляет новый столбец после столбца column_name. По умолчанию столбец добавляется в конец таблицы.

· **ADD INDEX [index_name] (index_col_name, ...)** Добавление индекса index_name для столбца index_col_name. Если имя индекса index_name не указывается, ему присваивается имя, совпадающее с именем столбца index_col_name.

· **ADD PRIMARY KEY (index_col_name, ...)** Делает столбец index_col_name или группу столбцов первичным ключом таблицы.

· **CHANGE old_col_name new_col_name type** Изменение столбца с именем **old_col_name** на столбец с именем **new_col_name** и типом **type**.

· **DROP col_name** Удаление столбца с именем col_name.

· **DROP PRIMARY KEY** Удаление первичного ключа таблицы.

· **DROP INDEX index_name** Удаление индекса index_name.

1. Добавление в таблицу new_table нового столбца test с размещением его после столбца name можно выполнить следующим SQL-запросом

mysql> ***ALTER TABLE new_table ADD test INT(10) AFTER name***;

2. Переименование столбца test в текстовый столбец new_test можно осуществить следующим образом.

mysql> ***ALTER TABLE new_table CHANGE test new_test text;***

Удаление таблиц. Команда DROP TABLE

Для того, чтобы удалить таблицу, убедитесь сперва, что она существует. Это можно проверить с помощью команды SHOW TABLES.

Инструкция DROP TABLE имеет следующий синтаксис:

DROP TABLE [IF EXISTS] таблица [RESTRICT | CASCADE]

Спецификация IF EXISTS подавляет вывод сообщения об ошибке, выдаваемого в случае, если заданная таблица не существует. Можно указывать несколько имен таблиц, разделяя их запятыми.

Флаги RESTRICT и CASCADE предназначены для выполнения сценариев, созданных в других СУБД.

Удаление баз данных. DROP DATABASE

Команда удаляет базу данных со всеми таблицами, входящими в ее состав. Синтаксис команды:

DROP DATABASE database_name;

Запись данных в таблицу. Оператор INSERT

Оператор **INSERT** заполняет таблицу данными.

INSERT into table_name (column1, column2, ...)values (value1, value2...);

где table_name является именем таблицы, в которую надо внести данные; column1, column2 … являются именами столбцов, а value1, value2 … являются значениями для соответствующих столбцов.

Несколько важных моментов:

- Значениями столбцов, которые являются текстовыми строками записываются в кавычках.
- Значение для столбца со свойством auto_increment задает система MySQL, которая находит в столбце наибольшее значение, увеличивает его на единицу, и вставляет новое значение.

Если приведенная выше команда правильно введена в приглашении клиента mysql, то программа выведет сообщение об успешном выполнении. Создание дополнительных записей требует использования отдельных операторов INSERT. Чтобы облегчить эту работу можно поместить все операторы INSERT в файл. Это должен быть обычный текстовый файл с оператором INSERT в каждой строке.

Ввод данных из файла. LOAD DATA LOAD DATA.

Команда **LOAD DATA INFILE** читает строки из текстового файла и вставляет их в таблицу с очень высокой скоростью.

```
LOAD DATA [LOW_PRIORITY | CONCURRENT] [LOCAL] INFILE
'file_name.txt'
    [REPLACE | IGNORE]    INTO TABLE tbl_name
    [FIELDS    [TERMINATED BY '\t']
      [[OPTIONALLY] ENCLOSED BY ''']    [ESCAPED BY '\\' ]    ]
    [LINES TERMINATED BY '\n']    [IGNORE number LINES]
    [(col_name,...)]
```

Если задано ключевое слово LOCAL, то файл читается с клиентского хоста. Если же LOCAL не указывается, то файл должен находиться на сервере. Если текстовые файлы, которые нужно прочитать, находятся на сервере, то из соображений безопасности эти файлы должны либо размещаться в директории

базы данных, либо быть доступными для чтения всем пользователям. Кроме того, для применения команды LOAD DATA INFILE к серверным файлам необходимо обладать привилегиями FILE для серверного хоста.

Изменение записей. UPDATE

Команда **UPDATE** выполняет **изменение данных** в таблицах. Она имеет очень простой формат.

UPDATE имя_таблицы SET имя_столбца_1 = значение_1, имя_столбца_2 = значение_2, имя_столбца_3 = значение_3,

...[WHERE условия];

Как и все другие команды SQL можно вводить ее на одной строке или на нескольких строках.

Предположим, в таблице Sotrudniki рабочим увеличили зарплату (поле Salary) на 1000 и надбавки (Premiya) на 500. Их предыдущая зарплата была 20000, а надбавки были 500.

UPDATE Sotrudniki SET Salary=220000, Premiya=55000 WHERE title='рабочий';

Можно воспользоваться арифметическими операторами. Следующий оператор сделает то же самое, при этом исходные данные знать заранее не требуется.

UPDATE Sotrudniki SET Salary= Salary + 1000, Premiya= Premiya +500 WHERE title='рабочий';

В качестве другого примера можно попробовать изменить название должности "прораб" на "мастер".

UPDATE Sotrudniki SET title='мастер' WHERE title='прораб';

Оператор UPDATE без условий изменит все данные столбца во всех строках. Надо быть очень осторожным при внесении изменений.

Удаление записей из таблицы. DELETE

Оператор DELETE требует задания имени таблицы и необязательных условий. Если никакие условия не заданы, то удаляются все данные в таблице.

DELETE from имя_таблицы [WHERE условия];

Предположим, сотрудник Иванов уволился из компании. Надо удалить его запись из таблицы Sotrudniki.

DELETE from Sotrudniki WHERE firstname = 'Иванов';

Содержание отчета (в электронном виде загрузить в Moodle)

1. Тема работы
2. Цель работы
3. Выполненное задание с кодом и скриншотами.
4. Ответы на контрольные вопросы

Оформление отчета должно удовлетворять требованиям предъявляемым к курсовым работам (правила оформления курсовых смотрите на сайте факультета)

Контрольные вопросы

1. Как просмотреть список имеющихся баз данных?
2. Принципы физического хранения данных в СУБД MySQL?
3. Какова архитектура СУБД MySQL?
4. Для чего предназначен язык SQL?
5. Назовите команды для создания БД и таблиц.
6. Можно ли создать таблицу до создания базы данных?
7. Как изменить значение по умолчанию столбца?
8. Сколько полей -счетчиков, создаваемые с помощью флага AUTO_INCREMENT, может быть в таблице?
9. Может ли поле счетчик не быть первичным ключом?
10. Какие типы данных допустимы при создании таблицы?
11. Как вставить строки данных в таблицу средствами SQL?
12. Как изменить строки таблицы средствами SQL?
13. Каким образом выполнить просмотр таблицы?
14. Как получить информацию о структуре таблицы?
15. Какими способами можно заполнить таблицу БД?