**Лабораторное занятие №9.**

**Внесение изменений в базу данных.**

**Лабораторное (практическое) занятие** рассчитано на 2 часа, относится к теме «»

**Формируемые компетенции:** ОК.1-9 ПК 2.1 - 2.4

**Цель:** Изучить используемые в реляционных СУБД операторы изменения данных. Получить навыки работы с SQL-операторами INSERT, DELETE и UPDATE. Заполнить разрабатываемую базу данных тестовыми данными для последующего использования.

**Методическое и техническое обеспечение:** IBM PC-совместимый компьютер, SQLiteStudio.

## Исходные данные

Студент получает индивидуальный вариант исходных данных с кратким описанием предметной области, который используется при выполнении всех лабораторных работ. При этом каждая очередная лабораторная работа является продолжением выполненной ранее и поэтому они должны обязательно выполняться последовательно. Вариант задания приведен в лабораторной работе №1.

**Теоретические сведения**

В SQL имеется три оператора, относящиеся к группе операторов DML (Data Manipulation Language), которые предназначены для выполнения запросов на добавление, удаление и обновление данных.

**Добавление новых данных**

Новые данные добавляются оператором **INSERT**. Наименьшей единицей информации, которую можно добавить в реляционную базу данных, является одна строка таблицы.

Немного упрощенный синтаксис оператора INSERT имеет вид:

**INSERT INTO** <имя\_таблицы> [**(**столбец [, столбец ...]**)**]

**{VALUES(**<значение> [, <значение> ...]**)** **|** <оператор **SELECT**>**};**

<величина> = **{**:Переменная | <константа> | <выражение> | <функция> | udf **(**[<величина> [, <величина> ...]]**)** | **NULL** | **USER}**

<константа> = Число | 'Строка'

<функция> = **CAST(**<величина> **AS** <тип данных>**)**

| **UPPER(**<величина>**)**

| **GEN\_ID(**<имя\_генератора>, <величина>**)**

< выражение> = SQL выражение, возвращающее единичное значение

В этом описании можно выделить два варианта оператора:

1. Вставка одной строки. Для этого после ключевого слова **VALUES** в круглых скобках указывают вставляемые величины.

2. Вставка в таблицу нескольких строк, выбранных с помощью оператора SELECT

В этой лабораторной работе рассматривается только первый вариант оператора INSERT.

Пример, когда в качестве вставляемых величин применены константы:

**INSERT INTO** Person**(**Pr\_ID, Pr\_LastName, Pr\_FirstName**)** **VALUES(**15 0, 'Иванов', 'Петр'**);**

**INSERT INTO** REFREG **(**CODCTR, CODREG, CENTER, NAMEREG**)** V**ALUES (**(**SELECT** CODCTR **FROM** REFCTR **WHERE** NAME **=** 'РОССИЯ'), '50', 'МОСКВА', 'Московская область');

Пример использования для вставки значений переменных:

*--объявление локальных переменных*

**DECLARE** Person\_Number **INTEGER**;

**DECLARE** LastName **VARCHAR(30)**;

**DECLARE** FirstName **VARCHAR(30)**;

**BEGIN**

Person\_Number = 150;

LastName = 'Иванов';

FirstName = 'Петр';

**INSERT INTO** Person (Pr\_ID, Pr\_LastName, Pr\_FirstName)

**VALUES(**:Person\_Number, :LastName, :FirstName**)**;

**END;**

**Удаление существующих данных**

Для удаления строк из таблицы используется оператор **DELETE**. Вот его упрощенный синтаксис:

**DELETE FROM** <имя\_таблицы>

[**WHERE** <условие поиска>]**;**

< условие поиска> = как в операторе SELECT

Если не использовать предложение **WHERE**, то будут удалены все строки в таблице.

*-- Удаление всех служащих:*

**DELETE FROM** Employee;

*-- Удаление всех людей с номерами 150 и больше:*

**DELETE FROM** Person **WHERE** Pr\_ID >= 150;

Отбирать строки для удаления не обязательно только на основании содержимого этих строк. Можно составить условие для удаляемых строк, опираясь на данные из других таблиц. Для составления таких условий необходимо сначала изучить оператор SELECT.

**Обновление существующих данных**

Оператор **UPDATE** обновляет значения одного или нескольких столбцов в выбранных строках одной таблицы. Строки для обновления указываются в предложении **WHERE**. Если пропустить предложение **WHERE**, то изменятся все строки таблицы.

**UPDATE** <имя\_таблицы>

**SET** Колонка = <величина> [,

Колонка = <величина> ... ]

[**WHERE** <условие поиска>]

<величина> = **{**Колонка | :Переменная | <константа> | <выражение> | <функция>

| udf**(**[<величина> [, <величина> ...]]**)** | **NULL** | **USER}**

< выражение> = SQL выражение, возвращающее единичное значение

< условие поиска> = как в операторе SELECT

Примеры:

*-- Увеличить зарплату всем служащим на 10%:*

**UPDATE** Employee

**SET** Salary = **1.1\***Salary;

/\* Увеличить зарплату всем служащим, которые имеют зарплату меньше 10000 на 15%: \*/

**UPDATE** Employee

**SET** Salary = **1.15\***Salary;

**WHERE** Salary **<=** 10000;

Отбирать строки для изменения, как и для удаления, можно с использование подчиненного запроса SELECT, который позволит учитывать в условии поиска изменяемых строк данные из других таблиц.

Например, можно выполнить такой запрос: увеличить зарплату на 10% всем служащим, работающим в отделе продаж, которые обслужили за последний месяц клиентов больше чем в полтора раза, чем в среднем по их отделу.

***Ход работы***

Для заполнения созданной базы данных тестовой информацией в диалоговом режиме программы "SQLiteStudio" необходимо выполнить следующие действия:

1. Подключиться к базе данных.

2. Открыть окно с таблицей, в которую будет заноситься информация.

3. Перейти на вкладку "Данные". Заполните таблицу..

***Задание***

Лабораторную работу следует выполнять в следующем порядке:

1. Выполнить по 3 запроса на каждый оператор (INSERT, UPDATE и DELETE) к имеющимся таблицам.
2. Результаты запросов сохранить.

***Отчет о выполнении работы***

Отчет о выполнении необходимо оформить на листах формата A4.

Отчет должен содержать описание и результаты работы, представляемые в следующей последовательности:

1. Распечатка сценария создания базы данных с комментариями, которые должны содержать сведения об авторе, дате создания, всех выполняемых действиях и пояснения к введенным вручную операторам изменения данных.

2. Сведения о количестве введенных в каждую таблицу тестовых данных.