Практическая работа №4
Исследование возможностей СУБД Access по созданию базы данных с помощью SQL-запросов.

Цель работы – исследовать возможности СУБД Access и получить практические навыки по созданию базы данных с помощью SQL-запросов.

1. Теоретические сведения

Для работы с базами данных, которые используют реляционную модель данных, предназначен язык SQL (Structured Query Language – структурированный язык запросов). С помощью SQL осуществляется реализация всех возможностей, которые представляются пользователям разработчиками СУБД:

- выборка данных (извлечение из базы данных содержащейся в ней информации);
- организация данных (определение структуры базы данных и установление отношений между ее элементами);
- обработка данных (добавление/изменение/удаление);
- управление доступом (ограничение возможностей ряда пользователей на доступ к некоторым категориям данных, защита данных от несанкционированного доступа);
- обеспечение целостности данных (защита базы данных от разрушения);
- управление состоянием СУБД.

Существуют две формы языка:

- интерактивный SQL - используется для создания запросов и получения результатов в интерактивном режиме;
- встроенный SQL - включает команды SQL, которые встроены внутрь программ, написанных на другом языке программирования. Это позволяет наиболее эффективно разрабатывать приложения, которые используют данные, хранящиеся в базе.

1.1. Общая характеристика языка

В SQL существует множество инструкций, каждая из которых требует выполнить определенное действие, например, извлечь данные, создать таблицу и т. д.

Все инструкции SQL имеют одинаковую структуру и заканчиваются они точкой с запятой(;).

Каждая инструкция SQL начинается с команды, т. е. ключевого слова, описывающего действие, выполняемое инструкцией. Командами, например, являются CREATE, INSERT, SELECT и т.д.

После команды идет одно или несколько предложений, описывающих данные, с которыми работает инструкция, либо содержится уточняющая информация о действии, выполняемом инструкцией. Каждое предложение также начинается с ключевого слова (например, WHERE, FROM и т. д.). Конкретная структура и содержимое предложения могут изменяться.

У каждого объекта в базе данных имеется уникальное имя (идентификатор), которое однозначно идентифицирует объект в системе баз данных, например, базу данных, таблицу или поля. Поддержка имен в различных СУБД реализована по-разному.

Если в инструкции SQL указано имя таблицы, СУБД предполагает обращение к таблице текущей базы данных. Для обращения к таблицам из других баз данных надо использовать полное имя таблицы, которое состоит из имени владельца (или имени схемы) и собственно имени таблицы, разделенных точкой.

Если в инструкции используется имя столбца, то СУБД сама определяет, в какой из указанных в этой же инструкции таблиц содержится данный столбец. Если необходимо включать столбцы с одинаковыми названиями из различных таблиц, то следует указывать полные имена столбцов, которые состоят из имени таблицы, содержащей столбец, и имени столбца (короткого имени), разделенных точкой. Если столбец находится в другой

базе данных, следует в полном имени столбца указывать полное имя таблицы. Полное имя столбца можно использовать во всех инструкциях SQL вместо короткого имени.

В языке SQL можно выделить несколько подмножеств языка SQL, в которых содержатся команды, использующиеся для различных целей. Перечень основных команд представлен в таблице 1.

Таблица 1. Перечень основных команд языка SQL

Подмножество языка SQL	Команда	Описание
DDL (Data Definition Language, язык определения данных) - позволяет создавать различные объекты базы данных и переопределять их структуру.	**CREATE TABLE**	Создание новой таблицы в базе данных
	ALTER TABLE	Изменение структуры существующей таблицы в базе данных, включая ограничения, заданные декларативно
	DROP TABLE	Удаление таблицы из базы данных
	CREATE VIEW	Создание представления
	ALTER VIEW	Изменение представления, созданного ранее
	DROP VIEW	Удаление представления
	CREATE INDEX	Создание индекса для определенной таблицы с целью обеспечения быстрого доступа по атрибутам, которые определены этим индексом
	DROP INDEX	Удаление индекса
DML (Data Manipulation Language, язык манипулирования данными) - позволяет пользователю манипулировать данными внутри объектов реляционных баз данных.	**INSERT**	Вставка одной строки в таблицу
	UPDATE	Обновление значения одного либо нескольких столбцов в одной или нескольких строках таблицы
	DELETE	Удаление одной либо нескольких строк из таблицы
DQL (Data Query Language, язык запросов к данным) - позволяет выполнить выборку данных из базы в соответствии с заданными критериями.	**SELECT**	Конструирование запросов к базе данных любой сложности. Данная команда имеет много опций и необязательных параметров
DCL (Data Control Language, язык управления данными либо команды администрирования данных) - позволяет осуществлять контроль над возможностью доступа к данным внутри базы данных. Команды DCL обычно используются для создания объектов, относящихся к управлению доступом пользователем к базе данных, а также для	**ALTER DBAREA**	Изменение созданной области хранения данных
	ALTER PASSWORD	Изменение пароля всей базы данных
	CREATE DATABASE	Создание новой базы данных
	CREATE DBAREA	Создание области хранения базы данных
	DROP DATABASE	Удаление базы данных
	DROP DBAREA	Удаление области хранения базы данных
	CREATE SYNONYM	Создание синонима (псевдонима) базы данных
	GRANT	Предоставление прав доступа для

назначения пользователям подходящих уровней привилегий (прав) доступа Команды администрирования данных - предоставляют возможность выполнять аудит и анализ операций внутри базы данных. Эти команды могут также помочь при анализе производительности системы данных в целом. Команды управления транзакциями - позволяют выполнить обработку информации, объединенной в транзакцию.		действий над заданными объектами базы данных
	REVOKE	Лишение прав доступа к объекту либо некоторым действиям над заданными объектами базы данных
	START AUDIT	Начало аудита и анализа операций внутри базы данных
	STOP AUDIT	Завершение аудита и анализа операций внутри базы данных
	COMMIT	Сохранение транзакции, т. е. завершение взаимосвязанной обработки информации, и тем самым изменение состояния базы данных
	ROLLBACK	Отмена транзакции, т. е. отменить изменения, которые выполнялись в ходе транзакции
	SAVE POINT	Создание точки отката внутри групп транзакций, к которым отсылает команда ROLLBACK
	SET TRANSACTION	Назначение имени транзакции

1.2. Типы данных в MS Access

В базах данных тип данных означает классификацию хранимых данных, будь то числа, символы или даты. Набор доступных типов данных зависит от СУБД, которые не всегда полностью и согласованно реализуются различными производителями в соответствиями со стандартами ANSI SQL (например, в SQL-92, SQL-99 и SQL-2003).

Таблица 2 содержит подмножество типов данных ANSI SQL и соответствующие названия, используемые в MS Access.

Таблица 2. Типы данных MS Access

ANSI SQL	MS Access	Описание	Пример
character(n)	**char(n)**	Хранит текстовые данные. Символ может быть любой буквой, цифрой или знаком пунктуации. Число хранимых символов следует указать заранее. Если реальное число сохраняемых символов окажется меньше заданного, СУБД заполнит оставшиеся позиции пробелами	char (8) выделяет пространство для восьми символов
character varying(n)	**varchar(n)**	Хранит текстовые данные. Подобен character, но длина текста оказывается переменной. Объем используемой памяти будет соответствовать действительному числу сохраняемых символов	varchar(8) выделяет пространство для хранения текстовых данных длиной до восьми символов. Но для хранения только одного символа потребуется лишь 1 байт памяти,

			двух символов — 2 байта памяти и т.д., вплоть до выделенных 8 байтов
text	**text**	TEXT (другое название — MEMO) Хранит текстовые данные от 0 до 2,14 Гбайт.	Символы в полях, сохраняются в формате представления символов Юникод.
integer	**int (integer)**	Хранит целое число из диапазона между -2147483648 и 2147483647	integer требует 4 байта памяти независимо от значения хранимого числа
smallint	**smallint**	Хранит целое число из диапазона между -32768 и 32767	smallint требует 2 байта памяти независимо от значения хранимого числа
real	**float (double)**	Хранит число с плавающим десятичным разделителем из диапазона от -3.40E+38 до 3.40E+38. Может иметь до восьми цифр после десятичного разделителя (например, 87.12342136)	float требует 4 байта памяти независимо от значения хранимого числа
date	**date**	Хранит значения даты	например, 1 Dec 2006 или 12/01/2006. Существуют разные форматы представления даты.
time	**time**	Хранит значения времени	например, 17:54:45

1.3. Создание базы данных

В случае MS Access создать базу данных можно только с помощью этой программы. Многие СУРБД позволяют создать базу данных с помощью SQL. Каждая СУРБД предлагает свои способы входа в систему и выполнения операторов SQL.

Для создания новой базы данных используется следующий оператор SQL, имеющий формат вида:

CREATE DATABASE <имя базы данных>;

Например, для создания базы данных myDatabase необходимо использовать оператор:

CREATE DATABASE myDatabase;

База данных названа здесь myDatabase, но для нее можно было бы выбрать любое другое название. Некоторые ограничения относительно имени базы данных все же существуют, например, относительно длины имени. В DB2 длина имени не должна превышать восемь символов, а в SQL Server — 123 символа. Для имен безопаснее всего

ограничиться использованием букв, цифр и символов подчеркивания, а также избегать использования знаков пунктуации.

А что делать, если потребуется удалить базу данных? Опять же, большинство СУРБД предлагают простую и удобную консоль, позволяющую сделать это, но для удаления базы данных можно использовать и непосредственно SQL.

Для удаления базы данных используется следующий оператор SQL, имеющий формат вида:

DROP DATABASE <имя таблицы>;

Например, удаления myDatabase необходимо использовать оператор:

DROP DATABASE myDatabase;

Эта команда требует предельной аккуратности, так как результатом этой команды является удаление базы данных из СУБД.

В случае MS Access создать базу данных и удалить её можно только с помощью этой программы.

1.4. Работа с таблицами

Для работы с таблицами и данными, опираясь на его реализацию в стандартном интерфейсе ODBC (Open Database Connectivity - совместимость открытых баз данных), используются операторы языка SQL, которые можно условно разделить на два подъязыка: язык определения данных DDL и язык манипулирования данными DML. Основные операторы языка SQL представлены в таблице 1.

1.4.1. Создание таблиц

Для создания таблиц используется оператор SQL CREATE TABLE, который имеет формат вида:

CREATE TABLE <имя таблицы>
(<имя столбца> <тип данных> [<ограничения>]
[,<имя столбца> <тип данных> [<ограничения>]] ...);

Обязательными операндами оператора являются имя создаваемой таблицы и имя хотя бы одного столбца (поля) с указанием типа данных, хранимых в этом столбце.

При создании таблицы для отдельных полей могут указываться некоторые дополнительные правила контроля вводимых в них значений (NOT NULL, UNIQUE, CHECK, PRIMARY KEY, FOREIGN KEY).

Конструкция NOT NULL (не пустое) для столбца таблицы означает, что в этом столбце должно быть определено значение.

Ограничение UNIQUE не позволяет двум различным записям таблицы иметь одинаковые значения в соответствующем столбце.

Ограничение CHECK задает условие, которое должно проверяться при вводе записи. Если условие принимает значение false, запись нарушает установленное ограничение и поэтому не вводится в таблицу.

Среди всех ограничений PRIMARY KEY является наиболее важным и наиболее часто используемым. Каждая таблица должна иметь первичный ключ, который обеспечивает связь между таблицами. Для того чтобы связь работала, первичный ключ должен уникальным образом идентифицировать записи, а значит, для него допускаются только уникальные значения и не допускается NULL.

Ограничение FOREIGN KEY позволяет указать, что столбец в таблице является внешним ключом из другой таблицы — тогда этот столбец на самом деле является только ссылкой на строки другой таблицы.

Пример 1. Пусть требуется создать таблицу «Производитель», имеющую поля: comp_id - идентификатор компании-производителя, name - название автомобиля, model - модель автомобиля, year - год выпуска и price - цена.

Для создания таблицы в MS Access необходимо:

1. Нажать кнопку «Конструктор запросов» на вкладке «Создание». В результате откроется окно шаблона запроса (рис. 1).

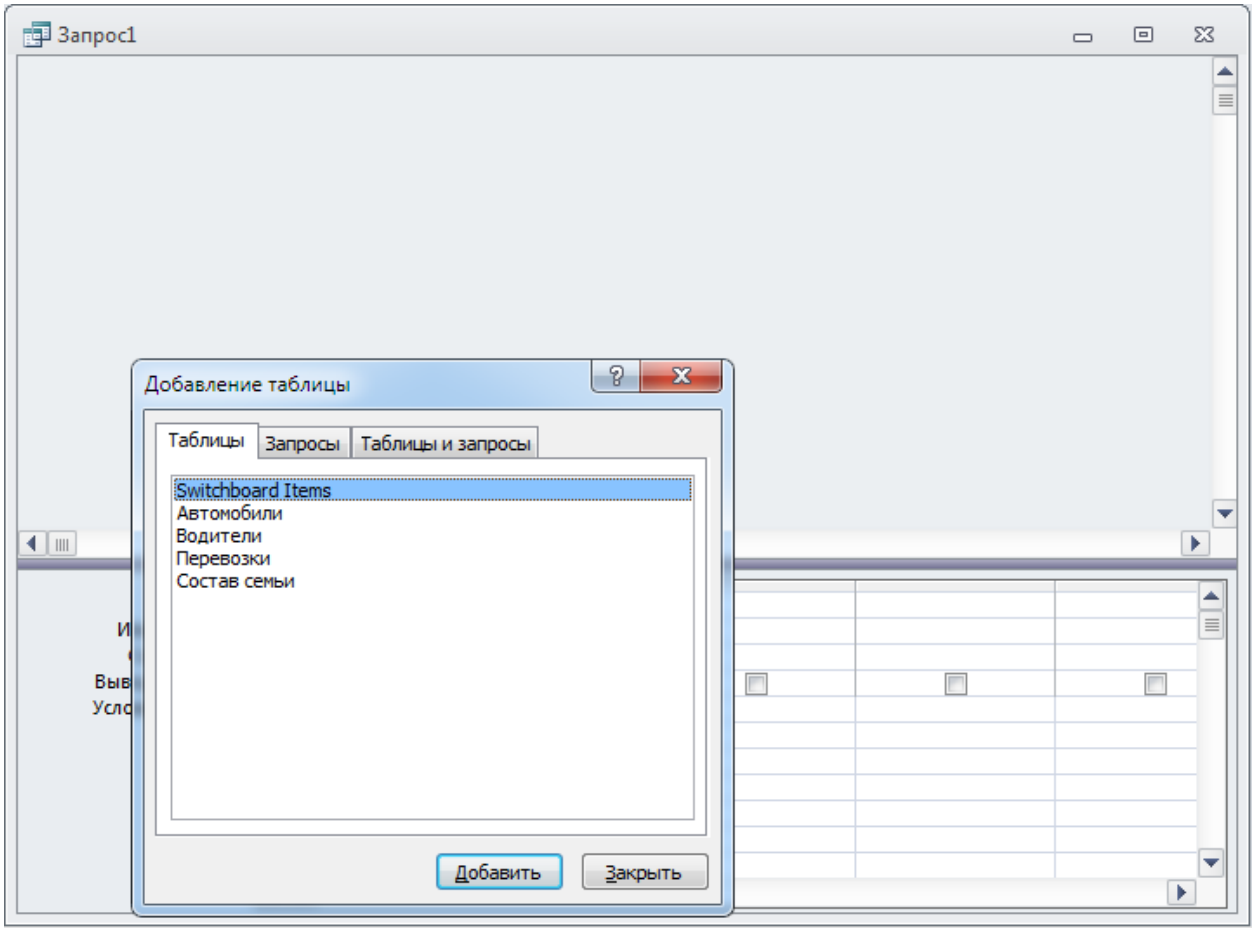

Рис. 1. Вид окна шаблона «Запрос1»

2. Закрыть диалоговое окно «Добавление таблицы». В результате появится вкладка «Работа с запросами. Конструктор».

4. В группе «Результаты» нажать кнопку «SQL» или в группе «Тип запроса» нажать кнопку «Управление», что приводит к переходу в режим набора операторов языка SQL.

Если была нажата кнопка «SQL», то возвратиться в режим «Конструктор» можно с помощью нажатия кнопки «Конструктор», которая появляется после выбора кнопки «Режим» в группе «Результаты».

Если была нажата кнопка «Управление», то возвратиться в режим «Конструктор» можно с помощью нажатия кнопки «Удаление» в группе «Тип запроса».

В результате выполненных действий откроется диалоговое окно «Запрос1», в котором необходимо записать соответствующий оператор.

Оператор определения таблицы может иметь следующий вид:

```
CREATE TABLE Производитель (comp_id char(10) PRIMARY KEY,
                    name varchar(20) NOT NULL, model char(15) NOT NULL,
                    year smallint, price double);
```

5. После записи оператора необходимо запустить запрос на выполнение с помощью кнопки «Выполнить» вкладки «Работа с запросами. Конструктор» или быстрой кнопки «Запуск». Если синтаксис оператора записан правильно, то по данному запросу создаётся таблица «Производитель», в противном случае – выдаётся сообщение об ошибке.

6. Сохранить запрос под соответствующим именем.

1.4.2. Изменение существующих таблиц

Для изменения существующих таблиц используется оператор SQL ALTER TABLE, который имеет формат вида:

ALTER TABLE <имя таблицы>
({ADD | MODIFY | DROP} <имя столбца> [<тип данных>] [NOT NULL]
[,{ADD, MODIFY, DROP } <имя столбца> [<тип данных>] [NOT NULL]] …);

Изменение структуры таблицы может состоять в добавлении (ADD), изменении (MODIFY) или удалении (DROP) одного или нескольких столбцов таблицы. Правила записи оператора ALTER TABLE такие же, как и оператора CREATE TABLE. При удалении столбца указывать <тип данных> не нужно.

Стандартный оператор ALTER TABLE ANSI SQL не позволяет изменять тип данных существующего столбца. Однако многие СУБД предлагают свою расширенную версию оператора ALTER TABLE, позволяющую изменение определения столбцов.

Для изменения типа существующего поля в MS Access необходимо использовать инструкцию ALTER COLUMN, указав имя поля, его тип и для текстовых и двоичных полей размер.

Пример 2. Необходимо в таблице «Производитель» добавить поле name_comp, отводимое для хранения названия компании. Для этого следует записать оператор вида:

ALTER TABLE Производитель ADD name_comp char(25);

Пример 3. Необходимо в таблице «Производитель» изменить размер поля model до 30 символов. Для этого следует записать оператор вида:

ALTER TABLE Производитель ALTER COLUMN model char(30);

Пример 4. Необходимо из таблицы «Производитель» удалить поле name_comp. Для этого следует записать оператор вида:

ALTER TABLE Производитель DROP name_comp;

1.4.3. Удаление существующих таблиц

Для удаления существующих таблиц используется оператор SQL DROP TABLE, который имеет формат вида:

DROP TABLE <имя таблицы>;

1.5. Работа с данными

1.5.1. Вставка данных

Вставка новых данных в БД выполняется с помощью оператора INSERT INTO. При этом требуется только указать таблицу, в которую необходимо вставить данные, соответствующий столбец и данные для вставки.

Оператор вставки новых записей имеет форматы двух видов:

INSERT INTO <имя таблицы> [(<список столбцов>)] VALUES (<список значений>);
INSERT INTO <имя таблицы> [(<список столбцов>)] <предложение SELECT>;

В первом формате оператор INSERT предназначен для ввода новых записей с заданными значениями в столбцах. Порядок перечисления имен столбцов должен соответствовать порядку значений, перечисленных в списке операнда VALUES. Если <список столбцов> опущен, то в <списке значений> должны быть перечислены все значения в порядке столбцов структуры таблицы.

Во втором формате оператор INSERT предназначен для ввода в заданную таблицу новых строк, отобранных из другой таблицы с помощью предложения SELECT.

Для вставки данных следует указать список имен столбцов (разделив их запятыми) в скобках после имени таблицы. В скобках после ключевого слова VALUES нужно указать список данных, предназначенных для соответствующих столбцов, тоже разделив их запятыми. Символьные данные и значения дат должны быть заключены в одиночные кавычки. Для числовых данных не требуются никакие ограничивающие символы.

Пример 5. Необходимо в таблицу «Производитель» ввести следующие данные:
- идентификатор компании: 00056;
- название автомобиля: Toyota;
- модель: Toyota Auris;
- год выпуска: 2011;
- цена: 21185$;

Для этого следует записать оператор вида:

INSERT INTO Производитель (comp_id, name, model, year, price)
VALUES ('00056', 'Toyota', 'Toyota Auris', 2011, 21185);

Чтобы проверить результат, используются либо средства управления СУБД, предназначенные для просмотра данных таблицы, либо оператор SQL:

SELECT * FROM Производитель;

Этот оператор выводит список всех записей таблицы «Производитель».

Имена столбцов можно указывать в любом порядке, поэтому аналогичный оператор может выглядеть и так:

INSERT INTO Производитель (comp_id, model, name, year, price)
VALUES ('00056', 'Toyota Auris', 'Toyota', 2011, 21185);

Независимо от порядка столбцов, указанного в операторе, SQL-код выполняется одинаково, важно только, чтобы порядок данных во втором наборе в скобках соответствовал порядку имен столбцов в первом.

Имена столбцов можно вообще опустить, записав оператор следующим образом:

INSERT INTO Производитель VALUES ('00056', 'Toyota Auris', 'Toyota', 2011, 21185);

1.5.2. Обновление данных

Для изменения записей используется оператор UPDATE. При этом главным отличием от операции вставки новых данных является необходимость указания того, какие записи требуется изменить. Записи, требующие обновления, указываются с помощью директивы WHERE, которая позволяет указать, что обновлять следует только записи, удовлетворяющие заданному условию.

Оператор изменения записей имеет формат вида:

UPDATE <имя таблицы> SET <имя столбца> = {<выражение> | NULL}
[, SET <имя столбца> = {<выражение> | NULL}...]
[WHERE <условие>];

Выполнение оператора UPDATE состоит в изменении значений в определенных операндом SET столбцах таблицы для тех записей, которые удовлетворяют условию, заданному директивой WHERE.

Новые значения полей в записях могут быть пустыми (NULL), либо вычисляться в соответствии с арифметическим выражением. Правила записи арифметических и логических выражений аналогичны соответствующим правилам оператора SELECT.

Пример 6. Необходимо в таблице «Производитель» изменить в записях об автомобилях Toyota год выпуска с 2011 на 2012. Для этого следует записать оператор вида:

UPDATE Производитель SET year = 2012 WHERE name = 'Toyota';

В директиве WHERE кроме операции сравнения можно использовать и другие типы сравнения (<> - не равно; > - больше; < - меньше; >= - больше или равно; <= - меньше или равно).

Операции сравнения работают с числовыми и текстовыми полями, а также полями даты. На самом деле они работают с большинством типов данных, но при этом имеются исключения. Например, некоторые СУБД поддерживают тип данных Boolean, который предполагает хранение только одного из двух значений — true (истина) или false (ложь). С таким типом данных нет смысла использовать операции сравнения, отличные от "равно" и "не равно".

С текстовыми типами данных операции >, <, >= и <= обеспечивают сравнение, эквивалентное алфавитному порядку.

Тот же принцип применяется и к датам: 1 января 2005 года будет меньше, чем 1 февраля 2005 года.

С помощью директивы WHERE можно проверять несколько условий сразу, воспользовавшись логическими операциями AND и OR. Логические операции AND ("И") и OR ("ИЛИ") позволяют проверять более одного условия в предложении WHERE.

Операция AND означает, что, если условие слева от этой операции и условие справа имеют значение true (истина), то в соответствующее поле устанавливается требуемое значение.

Операция OR, означающая, что, если хотя бы одно из условий будет равным true, то и соответствующая запись обновится.

Пример 7. Необходимо в таблице «Производитель» изменить в записях данные о ценах автомобилей Toyota 2011 года выпуска. Для этого следует записать оператор вида:

UPDATE Производитель SET price = 20150 WHERE name = 'Toyota' AND year = 2011;

1.5.3. Удаление данных

Для изменения записей используется оператор DELETE, который имеет формат вида:

DELETE FROM <имя таблицы> [WHERE <условие>];

Результатом выполнения оператора DELETE является удаление из указанной таблицы строк, которые удовлетворяют условию, определенному операндом WHERE. Если необязательный операнд WHERE опущен, т. е. условие отбора удаляемых записей отсутствует, удалению подлежат все записи таблицы.

Пример 8. Необходимо из таблицы «Производитель» удалить все записи об автомобилях 2009 года выпуска. Для этого следует записать оператор вида:

DELETE FROM Производитель WHERE year = 2009;

2. Рабочее задание

2.1. Исследование возможностей MS Access по созданию, модификации и удалению таблиц с помощью операторов SQL

Задание 1. Создайте запрос, в результате выполнения которого создается таблица «Ремонт», в которой хранится информация об автомобилях, проходивших ремонт или находящихся в ремонте (гос. номер, название системы, где обнаружена неисправность, название неисправности, причина неисправности, дата начала и окончания ремонта). Запрос сохранить под именем **Создание_таблицы**.

Под системой, где обнаружена неисправность, следует понимать системы автомобиля: двигатель, трансмиссия, ходовая часть, рулевое управление, тормозная система, бортовое электрооборудование и др.

Задание 2. Создайте запрос, в результате выполнения которого в таблицу «Ремонт» добавляются столбцы, содержащие информацию о коде неисправности, мерах по устранению неисправности, а также сведения об автомастере (фамилия), проводившего ремонт. Запрос сохранить под именем **Добавление_столбцов**.

Задание 3. Создайте запрос, в результате выполнения которого в таблицу «Водители» добавляется столбец, содержащий информацию о государстве, в котором родился водитель. Запрос сохранить под именем **Государство_рождения**. В данный столбец ввести соответствующую информацию.

Задание 4. Создайте запрос, в результате выполнения которого изменяется размер поля, в котором хранится информация о причине неисправности. Запрос сохранить под именем **Изменение_размера**.

Задание 5. Создайте запрос, в результате выполнения которого изменяется тип поля «Год выпуска» в таблице «Автомобили». Запрос сохранить под именем **Изменение_типа**.

2.2. Исследование возможностей MS Access по вводу, модификации и удалению данных с помощью операторов SQL

Задание 6. Создайте запросы, в результате выполнения которых в таблицу «Ремонт» записывается информация о 5-и ремонтируемых автомобилях. Данные о системе, причинах неисправностей, мерах по устранению выбрать самостоятельно, используя данные из таблицы 1. Запросы сохранить под именем **Ввод_данных_по_ремонту_1, … , Ввод_данных_по_ремонту_5**.

Таблица 1. Таблица неисправностей

Система	Неисправность	Причина	Меры по устранению
Двигатель	Двигатель проворачивается, но не запускается	Закупорка топливного фильтра	Замена фильтра
		Низкая компрессия в цилиндрах	Проверка компрессии в цилиндрах
	Черный дым при движении	Закупорка воздушного фильтра	Замена фильтра
		Топливные форсунки	Чистка форсунок
	Детонация двигателя	Низкий уровень масла	Добавление масла
		Чрезмерное отложение нагара	Очистка деталей от нагара
Трансмиссия	Проскальзывание сцепления	Недостаточный свободный ход педали	Регулировка хода педали
		Изношен ведомый диск	Замена диска
		Поврежден нажимной диск	Замена диска
	Неполное выключение сцепления	Зазор в педали сцепления	Регулировка зазора
		Замасливание накладки диска сцепления	Чистка накладок
Ходовая часть	Вибрация автомобиля	Разбалансировка колес	Балансировка колес
		Деформация дисков	Замена дисков
		Изношены подшипники	Замена подшипников
	Увод в сторону	Повреждение рулевого механизма	Ремонт элементов рулевого механизма
		Износ подшипников передних колес	Замена подшипников
Тормозная система	Увод при торможении	Повреждены тормозные	Замена тормозных

		колодки	колодок
		Заедание тормозных цилиндров	Замена тормозных цилиндров
		Деформация диска переднего тормоза	Замена диска переднего тормоза
Электрооборудование	При включении стартера тяговое реле многократно срабатывает и отключается	Разряжен аккумулятор	Заряд аккумулятора
		Обрыв в обмотке тягового реле	Замена тягового реле
		Сильное окисление наконечников	Чистка наконечников
	Не горят фары	Перегорели предохранители	Замена предохранителей
		Перегорели нити ламп	Замена ламп
		Окисление наконечников проводов	Чистка наконечников

Задание 7. Создайте запрос, в результате выполнения которого в таблицу «Водители» записывается информация о 2-х автомастерах. Данные об автомастерах выбрать самостоятельно. Запрос сохранить под именем **Автомастер**.

Задание 8. Создайте запрос, в результате выполнения которого обновляется информация о местожительстве (адрес и номер телефона) Царева В.Н. Запрос сохранить под именем **Простое_обновление**.

Задание 9. Создайте запрос, в результате выполнения которого обновляется информация о протяженности маршрута между Киевом и Харьковом. По уточненным данным протяженность маршрута Киев-Харьков составляет 497 км. Запрос сохранить под именем **Обновление_с_условием**.

Задание 10. Самостоятельно создать запрос, демонстрирующий использование операций сравнения совместно с логическими операциями. Запрос сохранить под именем **Самостоятельный**.

Задание 11. Создайте запрос, в результате выполнения которого из базы данных «Автоперевозки» будет удалена информация об одном из автомастеров. Запрос сохранить под именем **Удаление_данных**.

3. Контрольные вопросы

Литература

1. Бекаревич Ю.Б., Пушкина Н.В. Самоучитель Microsoft Access 2010. – СПБ.: БХВ-Петербург, 2011. – 752 с.: ил.
2. Вейскас Дж. Эффективная работа: Microsoft Office Access 2007. - СПб.: Питер, 2010. – 1168 с.: ил.
3. Гурвиц Г.А. Microsoft Access 2010. Разработка приложений на реальном примере. – СПБ.: БХВ-Петербург, 2010. – 462 с.: ил.
4. Информатика. Базовый курс. 2-е издание / Под ред. С.В. Симоновича. – СПб.: Питер, 2005. – 640 с.: ил.
5. Одиночкина С.В. Разработка баз данных в Microsoft Access 2010. Учебное пособие. – СПб.:НИУ ИТМО, 2012. – 83 с.: ил.
6. Рудикова Л.В. Базы данных. Разработка приложений. - СПб.: БХВ-Петербург, 2006. – 496 с.: ил.

7. Фуллер Л., Кук К. Microsoft Access 2010 для чайников. – М.:Диалектика, 2011. – 384 с.: ил.