Титульный лист

Содержание

[Вариант 3. Внутренняя организация СУБД 3](#_Toc515565881)

[Внутренняя организация СУБД: общие положения. 3](#_Toc515565882)

[Линейный список 4](#_Toc515565883)

[Инвертированный список 4](#_Toc515565884)

[Индексы 5](#_Toc515565885)

[Хеширование 6](#_Toc515565886)

[Вариант 4. Формирование плана выпуска изделий 8](#_Toc515565887)

[Создание таблиц: 9](#_Toc515565888)

[Ввод данных в таблицу. 9](#_Toc515565889)

[Создание форм 10](#_Toc515565890)

[Создание отчета 12](#_Toc515565891)

[Выводы о работе 13](#_Toc515565892)

[Список использованной литературы 14](#_Toc515565893)

# Вариант 3. Внутренняя организация СУБД

Внутренняя организация СУБД: общие положения. Линейный список.

Инвертированный список. Индексы. Хеширование. Кластеризация.

Ответ на вопрос:

## Внутренняя организация СУБД: общие положения.

Данные на внешних носителях информации (например, магнитных или оптических дисках) хранятся в виде файлов. Запись данных во внешнюю память и чтение их из нее реализуются операционной системой компьютера, предоставляющей прикладным программам (в том числе и СУБД) услуги по вводу-выводу информации и управлению памятью. Обработка файлов операционной системой на логическом уровне выполняется с помощью файловой системы, на физическом уровне - системы управления файлами.

Наибольшую производительность работы СУБД обеспечивают физические устройства хранения данных прямого доступа (магнитные и оптические диски и т. д.), обеспечивающие непосредственное считывание необходимой информации, если известно ее местоположение. Оптимальным является размещение данных в файлах прямого доступа, позволяющих выбирать нужные записи базы данных без просмотра всего содержимого файла. Для ускорения поиска информации применяются различные структуры хранения данных (способы их упорядочивания). Существенное внимание специалистов к разработке структур хранения и технологий методов доступа к данным определяется необходимостью сокращения числа дисковых операций ввода-вывода данных из-за относительно большого времени доступа к внешним устройствам.

Данные в процессе работы считываются и записываются страницами - блоками фиксированных размеров (в зависимости от используемой системы обычно 2, 4 или 8 килобайт). Каждая страница, хранящаяся на диске, имеет свой уникальный идентификатор, указывающий на физическое место ее хранения. Этот идентификатор используется системой управления файлами для чтения страницы и ее записи после изменения размещенных на ней данных в то же место на диске, откуда страница была считана.

Файл базы данных представляет собой набор страниц. При создании файла по запросу файловой системы система управления файлами выделяет ему требуемое количество страниц. Каждой странице в полученном наборе страниц присваивается некоторый «логический» (например, порядковый) номер. Обычно система управления файлами ведет каталог, в котором содержатся сведения о наборах страниц и указателях к каждой странице в их пределах. Когда СУБД в рамках решения прикладной задачи взаимодействует с конкретным файлом, логические номера страниц используются файловой системой, не знающей, где физически на диске хранится нужная страница, для обращения к системе управления файлами, осуществляющей чтение и запись данных.

На одной странице могут храниться одна или несколько записей с данными. После считывания из внешней памяти в оперативную память необходимой страницы выполняется выборка и обработка нужной записи. Для этого используется идентификационный номер записи, состоящий из двух частей: номера страницы и параметра, определяющего место расположения записи на странице.

В очень редких случаях запись, размеры которой превышают размеры страницы, может размещаться на двух (но не более) страницах. В таких ситуациях используются страницы переполнения.

## Линейный список

Линейный (последовательный) список - последовательность записей базы данных, сформированная по некоторым логическим принципам.

При поиске информации, соответствующей некоторым критериям (например, товаров с определенным названием или артикулом), линейный список необходимо просмотреть полностью от первой до последней записи. Это приводит к тому, что рассматриваемая структура хранения, обеспечивая оптимальные требования к минимальному объему выделяемой памяти на внешних устройствах, является неэффективной по быстродействию.

## Инвертированный список

Инвертированные списки позволяют существенно ускорить процесс поиска необходимой информации по сравнению с линейными списками. Это достигается с помощью упорядочивания (сортировки) записей исходного списка по значениям данных в одном из не ключевых полей. Инвертирование исходного списка можно выполнить для отдельных (частичное инвертирование) или всех (полное инвертирование) не ключевых полей исходного списка.

Обеспечивая уменьшение времени выполнения запросов, инвертирование приводит к существенному увеличению объемов внешней памяти для хранения информации в результате ее дублирования. Это увеличение прямо пропорционально количеству полей, по которым производится инвертирование. Для частичного устранения данного недостатка применяется создание индексов.

## Индексы

Индексы применяются для ускорения доступа к записям базы данных. Их можно сравнить с предметным указателем книги - упорядоченной последовательностью слов (словосочетаний) с перечнем номеров страниц, на которых встречается это слово (словосочетание).

Индекс базы данных представляет собой структуру, в которой содержатся рассортированные в заданном порядке значения данных в некотором поле и указатели адресов записей (страниц), где находятся эти значения. В отличие от инвертированных списков индексы занимают значительно меньшее место во внешней памяти.

Построение индекса и его обновление выполняется автоматически самой СУБД. Файл базы данных, для которого создан хотя бы один индекс, называется индексированным файлом.

Поиск и выборка нужных записей в базе данных осуществляются в следующей последовательности:

1. Выбирается индекс, соответствующий условию поиска

2. В индексе находится строка с заданным условием.

3. Из найденной строки выбираются номера страниц, где хранятся искомые записи.

4. Полученные номера страниц используются для чтения необходимой информации.

Большинство СУБД реализует этот процесс автоматически, без участия пользователя.

Если для таблицы определены несколько индексов по отдельным полям, при поиске записей в данной таблице по условиям поиска, заданным в этих полях, соответствующие индексы используются совместно.

Если поиск достаточно часто производится по одним и тем же полям, желательно создать составной индекс, включающий несколько полей. При использовании составного индекса сокращается время поиска, так как не приходится сравнивать значения данных из нескольких индексов.

Недостатком индексирования является необходимость выделения во внешней памяти дополнительного места для хранения индексов. Существуют оценки, что файл, в котором созданы индексы для всех полей размещенной в нем таблицы, занимает практически вдвое больше места на внешнем носителе по сравнению с исходным файлом. Кроме того, наличие индексов затрудняет обновление информации, хранящейся в базе данных, и требует дополнительных затрат времени на данный процесс.

Поэтому не рекомендуется индексировать все поля таблиц или поля, данные в которых часто изменяются. Наиболее эффективным является создание индексов для первичных и внешних ключей, для полей, по которым в основном выполняются запросы.

## Хеширование

Хеширование (Hashing) - алгоритмическое преобразование значений некоторого поля записей (первичного ключа или любого другого поля) в адреса их размещения на внешнем носителе.

При вводе данных СУБД вычисляет адрес страницы, на которой будет храниться запись, и заносит запись на эту страницу. При выполнении запросов реализуются аналогичные расчеты адреса страницы с последующим чтением записей с этой страницы. Достоинством такого метода является быстрый прямой доступ к данным. При этом время доступа практически не зависит от количества хранимых записей.

Основной проблемой хеширования является сложность выбора оптимальной хеш-функции. При выполнении данной работы анализируются размеры записей и страниц внешней памяти, вероятное распределение получаемых адресов. Неудачно выбранная хеш-функция может вызвать неравномерное распределение записей в памяти или формирование слишком длинных цепочек синонимов.

Весьма сложной является ситуация, когда количество запоминаемых записей больше, чем предполагалось, и необходимо увеличить место в памяти для их хранения или размеры таблицы хеширования. В этой ситуации потребуется выполнить полное повторное хеширование.

Следует иметь в виду, что хеширование позволяет организовать быстрый поиск информации только по одному полю, данные которого использовались для хеширования. Нельзя хранить записи в порядке, определяемом несколькими хеш-функциями.

Несмотря на отмеченные недостатки хеширования, данный способ организации хранения и быстрого извлечения необходимых данных широко применяется, в первую очередь в объектно-ориентированных базах данных.

Кластеризация

Кластеризация использует принцип близкого размещения во внешней памяти логически связанных данных. Этим обеспечиваются быстрый поиск и извлечение необходимой информации.

Физически кластеризация достигается размещением логически связанных записей на одной странице (если сделать это позволяют размеры страниц и записей) или на страницах, расположенных рядом.

Для кластеризированных таблиц можно создать неплотные индексы с указателями на первые записи, входящие в каждый кластер (блок).

Если кластеризации реализуется для одного логического объекта (файла) базы данных. Такая кластеризация называется внутрифайловой. Иногда применяется межфайловая кластеризация, когда на одной странице во внешней памяти размещаются записи из нескольких логических объектов (файлов) базы данных. Такой принцип хранения данных может существенно ускорить выполнение запросов, включающих критерии отбора для характеристик, хранимых совместно, но он замедляет поиск информации для всех других запросов. Поэтому кластеризация является оправданной, если к базе данных наиболее часто выполняются запросы одного типа. При этом следует иметь в виду, что одновременно можно реализовать только один вариант кластеризации базы данных, так как речь идет о физическом хранении информации.

# Вариант 4. Формирование плана выпуска изделий

1. Создать таблицы:

Таблица 1. Справочник изделий

Структура таблицы: Код изделия, Наименование изделия, Стоимость сборки

Таблица 2. Справочник деталей

Структура таблицы: Код детали, Наименование детали, Цена

Таблица 3. План выпуска

Структура таблицы: Код изделия, Количество изделий

Таблица 4. Состав изделия

Структура таблицы: Код изделия, Код детали, Количество деталей

2. Ввести в таблицу 1 сведения о пяти изделиях, в таблицу 2 - данные о 10ти деталях, которые используются при сборке изделий, в таблицу 3 - данные о плане выпуска пяти изделий (код изделия должен выбираться из поля со списком).

3. Создать форму «Состав изделия» типа главная/подчиненная для просмотра и ввода данных о составе изделия, наименование которого должно выбираться из списка. Главная форма должна содержать поля: Наименование изделия и Код изделия. Подчиненная форма должна иметь табличный вид и содержать строки с полями: Код детали, Количество в изделии. Значение поля Код детали должно выбираться из поля со списком. Каждое изделие должно содержать не менее трех деталей.

4. Создать форму «План выпуска изделия» типа главная/подчиненная.

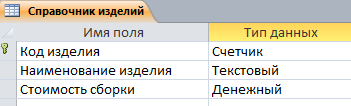
Код изделия должен выбираться из списка. Главная форма должна содержать поля: Код изделия, Наименование изделия, Количество изделий, Стоимость сборки, Стоимость выпуска изделия. Подчиненная форма должна иметь табличный вид и содержать графы: Код детали, Наименование детали, Цена детали, Количество деталей на план выпуска изделия, Стоимость деталей.

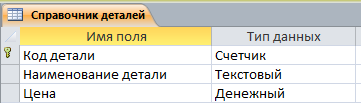
5. Создать итоговый отчет «План выпуска в стоимостном выражении», Отчет должен содержать графы: Код изделия, Наименование изделия, Количество изделий, Стоимость деталей, Стоимость сборки, Стоимость всего.

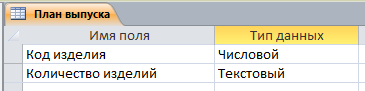
Отчет должен содержать все итоговые стоимости на план выпуска.

Ход выполнения:

## Создание таблиц:







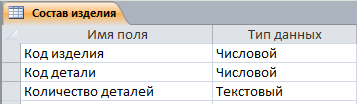
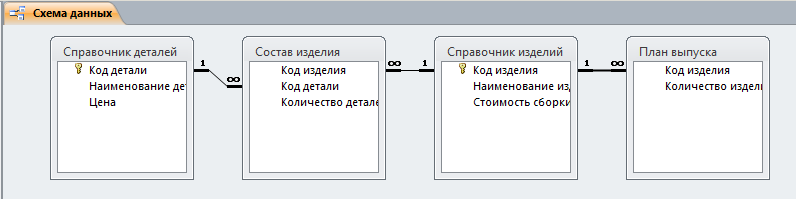
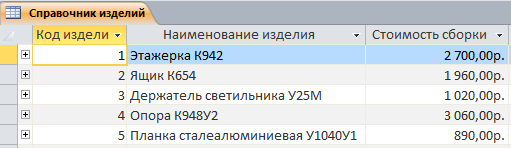


Схема данных:

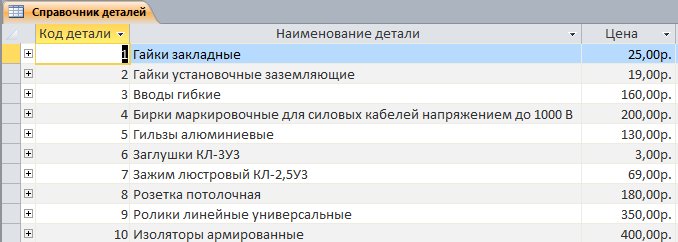


## Ввод данных в таблицу.

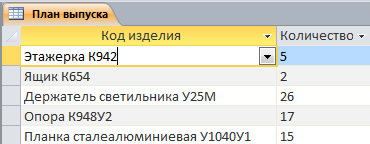
Ввести в таблицу 1 сведения о пяти изделиях:



В таблицу 2 - данные о 10ти деталях, которые используются при сборке изделий:

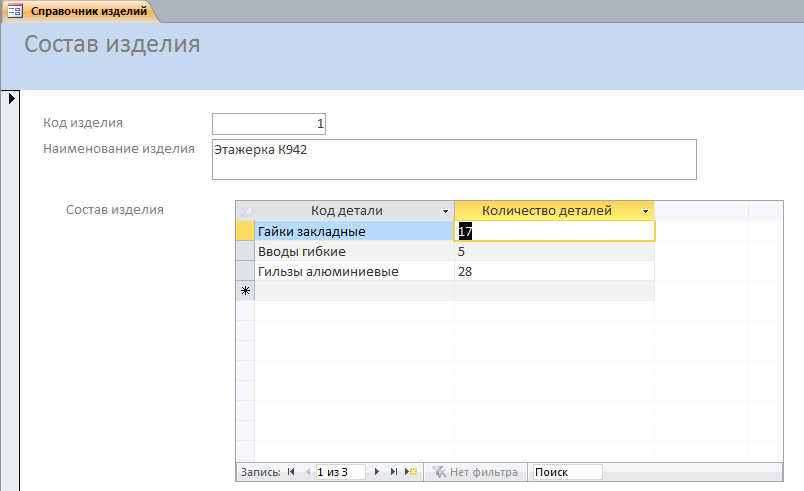


в таблицу 3 - данные о плане выпуска пяти изделий (код изделия должен выбираться из поля со списком):



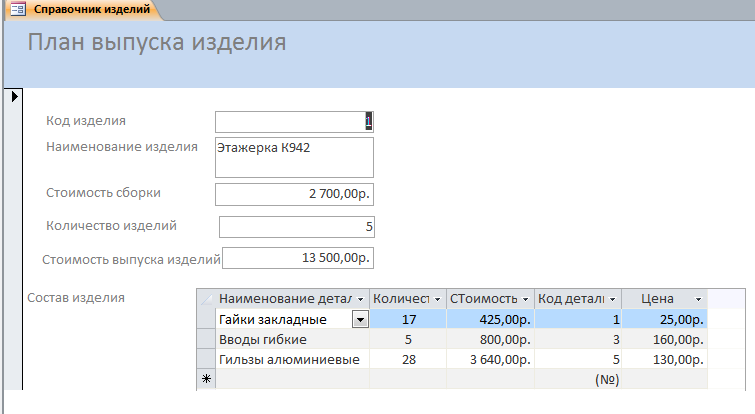
## Создание форм

Создать форму «Состав изделия» типа главная/подчиненная для просмотра и ввода данных о составе изделия, наименование которого должно выбираться из списка. Главная форма должна содержать поля: Наименование изделия и Код изделия. Подчиненная форма должна иметь табличный вид и содержать строки с полями: Код детали, Количество в изделии. Значение поля Код детали должно выбираться из поля со списком. Каждое изделие должно содержать не менее трех деталей.



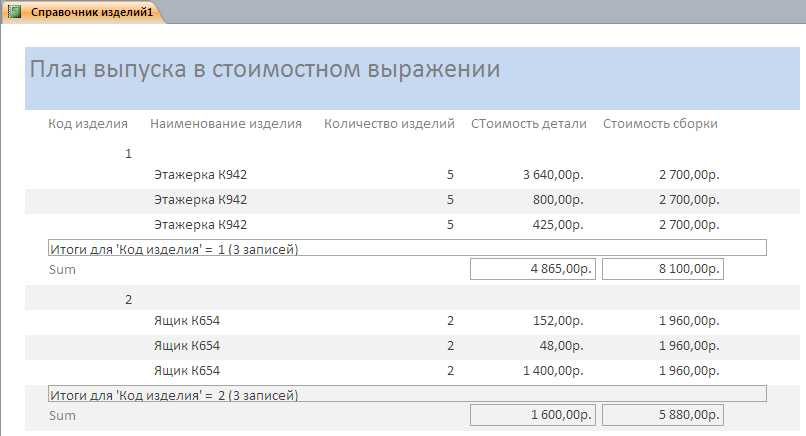
Создать форму «План выпуска изделия» типа главная/подчиненная.

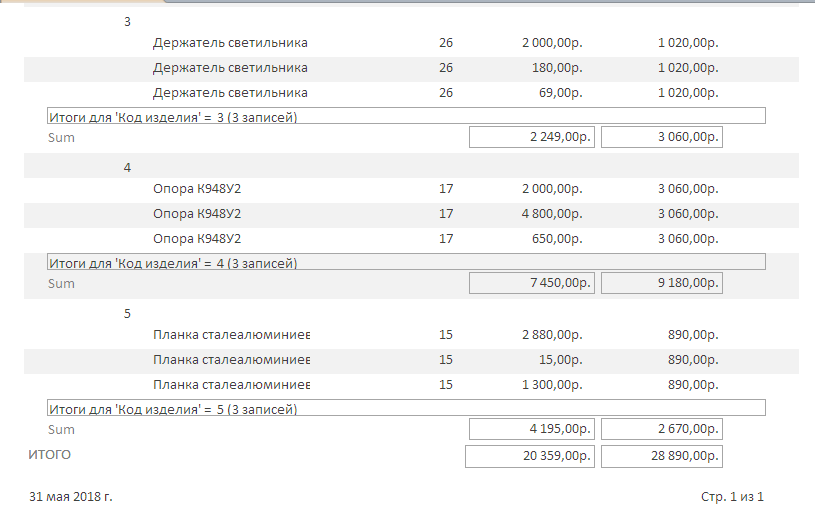
Код изделия должен выбираться из списка. Главная форма должна содержать поля: Код изделия, Наименование изделия, Количество изделий, Стоимость сборки, Стоимость выпуска изделия. Подчиненная форма должна иметь табличный вид и содержать графы: Код детали, Наименование детали, Цена детали, Количество деталей на план выпуска изделия, Стоимость деталей.



## Создание отчета

Создать итоговый отчет «План выпуска в стоимостном выражении», Отчет должен содержать графы: Код изделия, Наименование изделия, Количество изделий, Стоимость деталей, Стоимость сборки, Стоимость всего. Отчет должен содержать все итоговые стоимости на план выпуска.





# Выводы о работе

В ходе выполнения работы было получено представление об основах теории баз данных, в частности – проектирование базы данных. Мы научились создавать таблицы, запросы, формы и отчеты в СУБД Microsoft Access.

Знания по управлению базами данных были сформированы. Цели и задачи, поставленными перед нами, были выполнены в полном объеме.

# Список использованной литературы

1. Гаврилова и др. Базы знаний интеллектуальных систем // Учебник для вузов. – СПб.: Питер, 2000. – 320 с.
2. Загоруйко Н. Г. Прикладные методы анализа данных и знаний. –Новосибирск: ИМ СО РАН, 2005. – 152 с.
3. Базы данных/И.П. Карпова. – И.: Питер, 2013. – 240 с.
4. Википедия. Форма доступа: <https://ru.wikipedia.org>.
5. Объекты БД. Форма доступа: <https://mahara.org/user/sa4ik/ms-access>.