План :

1. Основные функции ИС.
2. Примеры ИС и способов их организации
    a. Каталог библиотеки (ручной подход)
    b. Агентство недвижимости (файловый подход)
3. СУБД , функции СУБД
4. Сравнительные характеристики способов.
5. Состав ИС

1 Основные функции ИС – ввод, хранение и извлечение информации.

2. Способы организации ИС

2.1 Ручной способ. Организация библиотечного каталога.
Хранятся следующие сведения : название книги, фамилии авторов, название издания, год издания, количество страниц

Носитель – бумага.
Рассмотреть, каким образом производится ввод данных, корректировка, резервное копирование, архивирование, уничтожение.
Достоинства сравнительная простота представления данных, отсутствие необходимости наличия особых технических средств, относительная низкая стоимость .
Недостатки – чрезмерное время поиска, практическая сложность упорядочения данных, невозможность резервирования данных, сложность модифицирования,

2.2 На основе файловой системы

Файловые системы были первой попыткой компьютеризировать известные всем ручные картотеки. Ручные картотеки позволяют успешно справляться с поставленными задачами, если количество хранимых информационных объектов невелико.
В качестве примера рассмотрим агентство недвижимости, которое состоит из двух отделов – отдела реализации и отдела контрактов. Отдел реализации собирает информацию о сдаваемых с аренду объектах недвижимости, их собственниках и потенциальных арендаторах.
Вся информация сохраняется в файлах, которые содержат записи с одинаковой структурой.

Пусть ИС отдела реализации состоит из двух файлов
НЕДВИЖИМОСТЬ
ВЛАДЕЛЬЦЫ

КЛИЕНТЫ

НЕДВИЖИМОСТЬ

№ объекта	Адрес	Кол-во комнат	Стоимость	Паспорт владельца
1	Ленина 10	3	500	69 01
2	Кирова 40	2	400	69 01
3	Усова 50	2	300	69 02
4	Иркутский 110	1	100	69 03

ВЛАДЕЛЬЦЫ

Паспорт владельца	ФИО	Телефон	Адрес
69 01	Иванов	123456	Ленина 15
69 02	Петров	234567	Кирова 90
69 03	Сидоров	678990	Кирова 50

КЛИЕНТЫ

Паспорт клиента	ФИО	Телефон	Кол-во комнат	Максимум оплаты
79 01	Гончаров	41-00-56	2	500
79 02	Васильев	41-00-57	2	300
79 03	Скворцов	41-00-48	1	150

Допустим, что отдел контрактов на основе информации о владельцах, арендаторах и арендодателях занимается составлением и регистрацией договоров. Информация хранится в трех файлах
НЕДВИЖИМОСТЬ
КЛИЕНТЫ
ДОГОВОРА

НЕДВИЖИМОСТЬ

№ объекта	Адрес	Кол-во комнат	Стоимость	Паспорт владельца
1	Ленина 10	3	500	69 01
2	Кирова 40	2	400	69 01
3	Усова 50	2	300	69 02
4	Иркутский 110	1	100	69 03

КЛИЕНТЫ

Паспорт клиента	ФИО	Телефон	Кол-во комнат	Максимум оплаты
79 01	Гончаров	41-00-56	2	500

79 02	Васильев	41-00-57	2	300
79 03	Скворцов	41-00-48	1	150

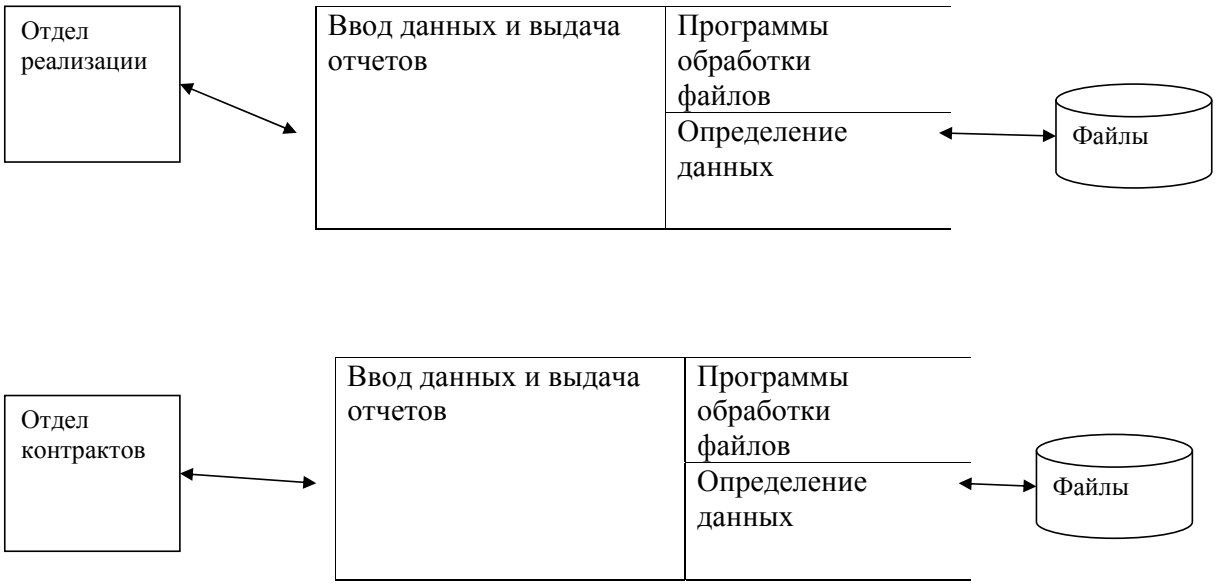
ДОГОВОРА

Паспорт клиента	№ объекта	Срок аренды, мес	Плата в месяц	Номер договора
79 01	1	6	500	4501
79 02	2	12	300	4502
79 03	3	24	100	4503

В целом эта ситуация схематически может быть представлена на рис. 1.3.

Примечание [M1]: Сделать рисунок

На этой схеме показано, что каждый отдел обращается к своим собственным данным с помощью специализированных приложений. Набор приложений каждого отдела позволяет вводить данные, работать с файлами и генерировать некоторый фиксированный набор специализированных отчетов. Самым важным является то обстоятельство, что физическая структура и методы хранения записей файлов с данными жестко определены в коде программ приложений.



Приведем пример небольшого приложения на языке Pascal, при помощи которого производится формирование структуры и ввод данных для файла ВЛАДЕЛЬЦЫ.

```
PROGRAM Client;
TYPE
    Man = Record of
        Pass    : Integer;
        FIO     : String(30);
        Phone   : String(6);
        Address : String(25);
    End;
VAR
    vMan     : Man;
    vPass    : Integer;
    vFIO     : String(30);
    vPhone   : String(6);
    vAddress : String(25);
    f        : File of Man;
    c        : Char;
BEGIN
    Assign(f, 'C:\owner.dat');
    Reset(f);
    REPEAT
        Write('Введите номер паспорта');  read(Man.Pass);
        Write ('Введите фамилию');        read(Man.FIO);
        Write ('Введите номер телефона');
    read(Man.Phone);
        Write('Введите адрес');
    read(Man.Address);
    Write(f, Man);
        Write('Продолжить ввод ?');       Read(c);
    UNTIL c='n'
    Close(f);
END.
```

Такой подход обладает еще рядом недостатков, а именно.

- Разделение и изоляция данных
- Дублирование данных
- Зависимость от данных
- Несовместимость файлов
- Фиксированные запросы/быстрое увеличение количества приложений

Разделение и изоляция данных

Когда данные изолированы в отдельных файлах, доступ к ним весьма затруднителен. Например, для создания списка всех объектов, отвечающих требованиям потенциальных арендаторов, предварительно нужно создать

временный файл со списком арендаторов, желающих арендовать недвижимость с 2 комнатами. Затем в файле НЕДВИЖИМОСТЬ следует осуществить поиск объектов с 2 комнатами и арендной платой ниже установленного арендатором максимума. Выполнять подобную обработку данных в файловых системах достаточно сложно. Для извлечения соответствующей поставленным условиям информации программист должен организовать синхронную обработку двух файлов. Трудности существенно возрастают, когда необходимо извлечь данные более чем из двух файлов.

Дублирование данных

Из-за децентрализованной работы с данными, проводимой в каждом отделе независимо от других отделов фактически допускается бесконтрольное дублирование данных, и это, в принципе, неизбежно. Видно, что в отделе реализации и отделе контрактов дублируется информация об объектах недвижимости и арендаторах. Бесконтрольное дублирование данных нежелательно по следующим причинам.

• Дублирование данных сопровождается неэкономным расходованием ресурсов, поскольку на ввод избыточных данных требуется затрачивать дополнительное время и деньги.

• Более того, для их хранения необходимо дополнительное место во внешней памяти, что связано с дополнительными накладными расходами. Во многих случаях дублирования данных можно избежать за счет совместного использования файлов.

• Дублирование данных может привести к нарушению их целостности. Иначе говоря, данные в разных отделах могут стать противоречивыми. Например, предположим, что у арендатора сменился номер телефона. Если этот изменение будет зафиксировано в одном отделе, но незафиксировано в другом, то невозможно будет узнать, какой же на самом деле телефон у арендатора.

Зависимость от данных

Как уже было показано, физическая структура и способ хранения записей файлов данных жестко зафиксированы в коде приложений. Это значит, что изменить существующую структуру данных достаточно сложно. Например, пусть в городе нумерация изменилась с 6-значной на 7-значную. Данный факт приведет к необходимости изменения формата хранимых данных. Чтобы осуществить это, необходимо переписать код программы и помимо этого – перековертировать уже имеющиеся данные. Приведем упрощенный алгоритм.

• открыть исходный файл `owner.dat для` чтения;

• открыть временный файл с новой структурой записи;

• считать запись из исходного файла, преобразовать данные в новый формат и записать их во временный файл. Эти действия следует выполнить для всех записей исходного файла;

• удалить исходный файл `owner.dat`;

• присвоить временному файлу имя `owner.dat`

Помимо этого, все обращающиеся к файлу `owner.dat` программы должны быть изменены с целью соответствия новой структуре файла. А таких программ может быть очень много. Следовательно, программист должен прежде всего выявить все программы, нуждающиеся в доработке, а затем их перепроверить и изменить. Обратите внимание, что многие подлежащие изменению программы могут обращаться к файлу `owner.dat`, но при этом вообще не использовать поле телефона. Ясно, что выполнение всех этих действий требует больших затрат времени и может явиться причиной появления ошибок. Данная особенность файловых систем называется *зависимостью программ от данных* (program-data dependence).

Несовместимость форматов файлов

Поскольку структура файлов определяется кодом приложений, она также зависит от языка программирования этого приложения. Например, структура файла, созданного программой на языке Pascal, может значительно отличаться от структуры файла, создаваемого программой на языке C. Прямая несовместимость таких файлов затрудняет процесс их совместной обработки.

Фиксированные запросы/быстрое увеличение количества приложений

С точки зрения пользователя возможности файловых систем намного превосходят возможности ручных картотек. Соответственно возрастают и требования к реализации новых или модифицированных запросов. Однако файловые системы требуют больших затрат труда программиста, поскольку все необходимые запросы и отчеты должны быть созданы именно им.

1.3. Системы с использованием баз данных

Все перечисленные выше ограничения файловых систем являются следствием двух факторов.

1. Определение данных содержится внутри приложений, а не хранится отдельно и независимо от них.
2. Помимо приложений не предусмотрено никаких других инструментов доступа к данным и их обработки.

Для повышения эффективности работы необходимо использовать новый подход, а именно *базу данных* (database) и *систему управления базами данных,* или *СУБД* (Database Management System — DBMS). В этом разделе представлено формальное определение этих терминов, а также рассмотрены компоненты среды СУБД.

1.3.1. База данных

База данных. *Совместно используемый набор логически связанных данных]*

;(и описание этих данных), предназначенный для удовлетворения информацион-

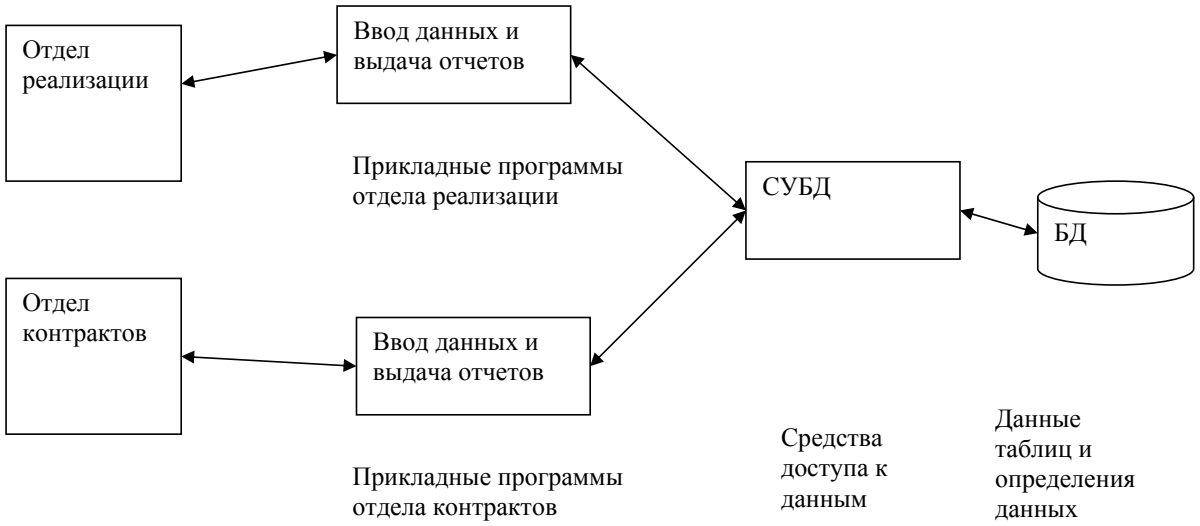
ных потребностей организации.

База данных — это единое, большое хранилище данных, которое *однократно* определяется, а затем совместно используется многими пользователями — представителями разных подразделений. Вместо разрозненных файлов с избыточными данными здесь все данные собраны вместе с минимальной долей избыточности. База данных уже не принадлежит какому-либо единственному отделу, а является общим корпоративным ресурсом. Причем база данных хранит не только рабочие данные этой организации, но и их описания.

По этой причине базу данных еще называют *набором интегрированных записей с самоописанием.* В совокупности описание данных называется *системным каталогом* (system catalog), или *словарем данных* (data dictionary), а сами элементы описания принято называть *метаданными* (meta-data), т.е. "данными о данных". Именно наличие самоописания данных в базе данных обеспечивает в ней *независимость программ от данных* (program-data independence).

Система управления базами данных (*СУБД*) - это комплекс программных и языковых средств, необходимых для создания баз данных, их поддержания в актуальном состоянии и организации в них поиска необходимой информации.

Рассмотрим принцип организации ИС при помощи СУБД на примере того же агентства.



Рассмотрим языковые средства СУБД. Ими являются :
DDL – data definition language, или язык описания данных. Содержит набор операторов, при помощи которых производится описание данных.

DML – data manipulating language, или язык манипулирования данными данных. Содержит набор операторов, при помощи которых производится ввод, обновление, удаление и выборка данных.

DCL – data control language, или язык контролирования данных. Содержит набор операторов, при помощи которых регламентирование доступа к данным.

Приведем пример реализации ИС для вышеупомянутого агентства.

Определение данных

```
CREATE TABLE CLIENT (
       ClientPass           INTEGER NOT NULL,
       ClientFIO            CHAR(25) NOT NULL,
       ClientPhone          INTEGER NULL,
       ClientFlat           SMALLINT NULL,
       ClientMaxRent        NUMBER(4) NOT NULL
);


ALTER TABLE CLIENT
       ADD  ( PRIMARY KEY (ClientPass) ) ;


CREATE TABLE LEASE (
       ClientPass           INTEGER NOT NULL,
       LeaseNum             INTEGER NOT NULL,
       PropertyNun          INTEGER NOT NULL,
       LeaseDuration        SMALLINT NOT NULL,
       LeaseRent            NUMBER(4) NOT NULL
);


ALTER TABLE LEASE
       ADD  ( PRIMARY KEY (LeaseNum) ) ;


CREATE TABLE OWNER (
       OwnerPass            INTEGER NOT NULL,
       OwnerFIO             CHAR(25) NOT NULL,
```

```
       OwnerPhone           INTEGER NULL,
       OwnerFlat            SMALLINT NOT NULL
);


ALTER TABLE OWNER
       ADD  ( PRIMARY KEY (OwnerPass) ) ;


CREATE TABLE PROPERTY (
       PropertyNun          INTEGER NOT NULL,
       PropertyAddress      VARCHAR(25) NOT NULL,
       PropertyRent         NUMBER(4) NOT NULL,
       PropertyFlat         SMALLINT NOT NULL,
       OwnerPass            INTEGER NOT NULL
);


ALTER TABLE PROPERTY
       ADD  ( PRIMARY KEY (PropertyNun) ) ;


ALTER TABLE LEASE
       ADD  ( FOREIGN KEY (PropertyNun)
                           REFERENCES PROPERTY ) ;


ALTER TABLE LEASE
       ADD  ( FOREIGN KEY (ClientPass)
                           REFERENCES CLIENT ) ;


ALTER TABLE PROPERTY
       ADD  ( FOREIGN KEY (OwnerPass)
                           REFERENCES OWNER ) ;
```

Ввод данных

```
INSERT INTO PROPERTY VALUES(1,Ленина 10,3,500,69 01);
INSERT INTO PROPERTY VALUES(2,Кирова 40,2,400,69 01)
INSERT INTO PROPERTY VALUES(3,Усова 50,2,300,69 02)
INSERT INTO PROPERTY VALUES(4,Иркутский 110,1,100,69 03)
```

```
INSERT INTO OWNER VALUES(6901,'Иванов',123456,Ленина 15)
INSERT INTO OWNER VALUES(6902,'Петров',234567,Кирова 90)
INSERT INTO OWNER VALUES(6903,'Сидоров',678990,Кирова 50)

INSERT INTO CLIENT VALUES(79 01,'Гончаров',41-00-56,2,500)
INSERT INTO CLIENT VALUES(79 02,'Васильев',41-00-57,2,300)
INSERT INTO CLIENT VALUES(79 03,'Скворцов',41-00-48,1,150)


INSERT INTO LEASE VALUES(79 01,1,6,500,4501)
INSERT INTO LEASE VALUES(79 02,2,12,300,4502)
INSERT INTO LEASE VALUES(79 03,3,24,100,4503)
```

Выборка данных

Все двухкомматные квартиры

```
SELECT *  FROM Property WHERE PropertyFlat=2
```

Преимущества ИС с использованием СУБД

Контроль за избыточностью данных
Непротиворечивость данных
Совместное использование данных
Поддержка целостности данных
Повышенная безопасность
Развитые службы резервного копирования и восстановления

Контроль за избыточностью данных

Традиционные файловые системы неэкономно расходуют внешнюю память, сохраняя одни и те же данные в нескольких местах. При использовании базы данных, наоборот, предпринимается попытка исключить избыточность данных за счет интеграции файлов. Это позволяет исключить необходимость хранения нескольких копий одного и того же элемента информации. Однако полностью избыточность информации в базах данных не исключается, а лишь ***ограничивается ее степень***. В одних случаях необходимо дублировать ключевые поля для связи таблиц. В других случаях некоторые данные требуется дублировать из соображений повышения производительности системы.

Непротиворечивость данных

Устранение избыточности данных или контроль над ней позволяет уменьшить риск возникновения противоречивых состояний. Если элемент данных хранится в базе только в одном экземпляре, то для изменения его значения потребуется выполнить только одну операцию обновления, причем новое значение станет доступным сразу всем пользователям базы данных.

Совместное использование данных

Файлы обычно принадлежат отдельным лицам или целым отделам, которые используют их в своей работе. В то же время база данных принадлежит всей организации в целом и может совместно использоваться всеми зарегистрированными пользователями. При такой организации работы большее количество пользователей может работать с большим объемом данных.

Поддержка целостности данных

Целостность базы данных означает корректность и непротиворечивость хранимых в ней данных. Целостность обычно описывается с помощью *ограничений,*т.е. правил поддержки непротиворечивости, которые не должны нарушаться в базе данных. Ограничения можно применять к элементам данных внутри одной
записи или к связям между записями. Например, ограничение целостности может гласить, что арендная плата за квартиру не может превышать 600.

Повышенная безопасность

Безопасность базы данных заключается в защите базы данных от несанкционированного доступа со стороны пользователей. Без привлечения соответствующих мер безопасности интегрированные данные становятся более уязвимыми . Система обеспечения безопасности может быть выражена в форме имен и паро-
лей для идентификации пользователей, которые зарегистрированы в этой базе данных. Доступ к данным со стороны зарегистрированного пользователя может быть ограничен только некоторыми операциями (извлечением, вставкой, обновлением и удалением). Например, АБД может быть предоставлено право доступа ко всем данным в базе данных, менеджеру отделения компании — ко всем данным, которые относятся к его отделению, а инспектору отдела реализации — лишь ко всем данным о недвижимости, в результате чего он не будет иметь доступа к другим данным.

Недостатки
Высокая сложность
Большой размер
Высокая стоимость СУБД
Дополнительные затраты на аппаратное обеспечение

СОСТАВ и окружение ИС

Пользователи ИС.
С ИС в процессе создания и эксплуатации взаимодействуют пользователи различных категорий, а именно:

1. **Конечные пользователи**. Это специалисты предметных областей, для которых собственно и создаются ИС. Конечные пользователи различаются сферой своих интересов, широтой информационных потребностей, квалификацией, мобильностью и пр. Понятием конечный пользователь определяются не только отдельное лицо или группа лиц, но и вычислительные процессы, задачи, а иногда и целые системы, взаимодействующие с ИС.
2. **Прикладные программисты**. Они играют роль посредников между ИС и конечным пользователем. Принимают непосредственное участие при разработке информационной системы обработки данных.
3. **Администраторы.** Службой администратора БнД называется структурное подразделение организации, вычислительных центров или АСУ, которое несет ответственность за создание БнД и его надежное функционирование, за соблюдение регламента доступа к хранимым данным и целостности данных.

Состав Бнд

База данных. База данных является ядром банка данных. Базой данных называется поименованная структурированная совокупность взаимосвязанных данных, относящихся к конкретной предметной области. В базу данных обычно не входят файлы входной и выходной информации, архивные файлы, вводимые запросы, временные файлы.

Программные средства. Программные средства БнД представляют собой сложный комплекс, который обеспечивает взаимодействие всех частей информационной системы в процессе ее создания и существования.
В состав программных средств обычно входят

1. Система управления базами данных (СУБД)
2. Информационная система управления данными,
3. программы, обеспечивающие взаимодействие пользователей и технических средств.
4. В состав программных средств Бнд входит также и операционная система.

Технические средства БнД. К техническим средствам относятся устройства ввода-вывода, внешние запоминающие устройства для хранения массивов данных, каналы связи и аппаратура передачи данных. В каждом конкретном случае в зависимости от используемой СУБД состав технических средств

может быть различным. В технической документации СУБД указывается минимальная конфигурация технических средств , необходимых для организации банка данных, а также различные ограничения на состав и количество технических средств.

Организационно – методические средства БнД состоят из нормативно-технологических методических материалов по организации и использованию БнД.

Администраторы БнД.
Администраторы БнД – это коллектив специалистов, которые обеспечивают создание и поддержание целостности баз данных, организуют и контролируют доступ к данным различных пользователей.