МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»
КРАСНОАРМІЙСЬКИЙ ІНДУСТРІАЛЬНИЙ ІНСТИТУТ

Лесіна Є.В.

КОНСПЕКТ ЛЕКЦІЙ

з нормативної навчальної дисципліни
циклу природничо-наукової підготовки

ІНФОРМАЦІЙНІ СИСТЕМИ ТА ТЕХНОЛОГІЇ

Напрям підготовки: 6.030601 «Менеджмент»
(для студентів всіх форм навчання)

Розглянуто
на засіданні кафедри
загальнонаукової підготовки
Протокол № 4 від 28 жовтня 2015р.

Затверджено на засіданні
навчально-методичного відділу
ДонНТУ
Протокол № 2 від 28 жовтня 2015р.

Красноармійськ-2015

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
«ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»
КРАСНОАРМІЙСЬКИЙ ІНДУСТРІАЛЬНИЙ ІНСТИТУТ

Лесіна Є.В.

Інформаційні системи та технології

Конспект лекцій

Красноармійськ-2015

УДК 681.625.9
ББК 32.973

«Інформаційні системи та технології». Конспект лекцій. / Укл.: канд. фіз.-мат. наук Лесіна Є.В. – Красноармійськ: КІІ ДонНТУ, 2015. – 116 с.

Містить комплекс лекцій з дисципліни «Інформаційні системи та технології». Розкрито базові поняття теорії інформації, види апаратного і програмного забезпечення ПК. Розглянуто основні принципи роботи комп'ютерних мереж, історія і тенденції розвитку комп'ютерних систем. Надається уява про основи побудови баз даних в інформаційних системах.

Призначений для студентів всіх форм навчання напряму підготовки 6.030601 «Менеджмент».

Рецензент: **Вірич С.О.** – кандидат технічних наук, доцент, завідувач кафедри «Гірничі машини і мехатронні системи в машинобудуванні» Донецького національного технічного університету.

ЗМІСТ

Змістовий модуль 1
ОСНОВИ ОРГАНІЗАЦІЇ ІНФОРМАЦІЙНИХ СИСТЕМ

Лекція 1. Основні поняття та роль інформаційних систем в економіці

Мета: надання основних визначень інформатики – *інформація, інформаційна система, інформаційні технології*, виділення властивостей інформації. Надання класифікацій комп'ютерів, програмного забезпечення.

План

1.1. Основні поняття інформатики;
1.2. Пристрої сучасних ПК;
1.3. Програмне забезпечення;
1.4. Операційна система;
1.5. Архівні файли і антивірусний захист.

1.1. Основні поняття інформатики

Інформатика – наука про способи отримання, накопичення, зберігання, перетворення, передачу та використання наукової інформації.

Термін "інформатика" походить від французького слова Informatique і утворений із двох слів: інформація та автоматика. Цей термін почав використовується у Франції в середині 60-х років XX ст., коли розпочалося широке використання обчислювальної техніки. Тоді в англомовних країнах увійшов до вживання термін Computer Science для позначення науки про перетворення інформації, яка базується на використанні обчислювальної техніки. Тепер ці терміни є синонімами.

Предмет інформатики як науки складають:

- апаратне забезпечення засобів обчислювальної техніки;
- програмне забезпечення засобів обчислювальної техніки;
- засоби взаємодії апаратного та програмного забезпечення;
- засоби взаємодії людини з апаратними та програмними засобами.

Засоби взаємодії в інформатиці прийнято називати *інтерфейсом*. Тому засоби взаємодії апаратного та програмного забезпечення інколи називають

також *програмно-апаратним інтерфейсом*, а засоби взаємодії людини з апаратними та програмними засобами – *інтерфейсом користувача.*

Основне завдання інформатики як науки – це систематизація прийомів і методів роботи з апаратними та програмними засобами обчислювальної техніки. *Мета систематизації* полягає в тому, щоб виділяти, впроваджувати і розвивати передові, найбільш ефективні технології автоматизації етапів роботи з даними, а також методично забезпечувати нові технологічні дослідження.

На всіх етапах технічного забезпечення інформаційних процесів для інформатики ключовим питанням є *ефективність*.

- *Для апаратних засобів* під ефективністю розуміють співвідношення продуктивності обладнання до його вартості.
- *Для програмного забезпечення* під ефективністю прийнято розуміти продуктивність користувачів, які з ним працюють.
- *У програмуванні* під ефективністю розуміють обсяг програмного коду, створеного програмістами за одиницю часу.

В інформатиці все жорстко орієнтоване на ефективність. Питання, як здійснити ту чи іншу операцію, для інформатики є важливим, але не основним. *Основним є питання* – як здійснити дану операцію ефективно.

В рамках інформатики як технічної науки можна сформулювати поняття *інформації, інформаційної системи* та *інформаційної технології*.

Інформація – доступна група даних яка зберігається або передається, структурована необхідним для отримання знання чином.

Інформація існує у вигляді документів, креслень, малюнків, текстів, звукових та світлових сигналів, електричних та нервових імпульсів тощо.

Якість інформації – ступінь її відповідності потребам споживачів.

Властивості інформації є відносними, оскільки залежать від потреб споживача інформації. *Найважливіші властивості інформації*:

репрезентативність – правильність відбору інформації з метою адекватного відображення джерела інформації;

достатність (повнота) – мінімальний, але достатній склад даних для досягнення цілей, які переслідує споживач інформації. Ця характеристика схожа на репрезентативність, однак різниця полягає в тому, що в даному випадку враховується мінімальний склад інформації, який не заважає прийняттю рішення;

доступність – простота (або можливість) виконання процедур отримання і перетворення інформації. Ця характеристика може бути застосована не до всієї інформації, а лише до тієї, яка не є закритою;

актуальність – залежить від динаміки зміни характеристик інформації і визначається збереженням цінності інформації для користувача в момент її використання;

своєчасність – надходження не пізніше заздалегідь призначеного терміну;

точність – ступінь близькості інформації до реального стану джерела інформації;

достовірність – властивість інформації відображати джерело інформації з необхідною точністю;

стійкість – здатність інформації реагувати на зміни вихідних даних без порушення необхідної точності.

Під час інформаційного процесу дані перетворюються з одного виду в інший за допомогою методів. Обробка даних включає в себе безліч різних операцій. *Основними операціями* є:

збір даних – накопичення інформації з метою забезпечення достатньої повноти для прийняття рішення;

формалізація даних – приведення даних, що надходять з різних джерел, до єдиної форми;

фільтрація даних – усунення зайвих даних, які не потрібні для прийняття рішень;

сортування даних – впорядкування даних за заданою ознакою з метою зручності використання;

архівація даних – збереження даних у зручній і доступній формі;

захист даних – комплекс заходів, спрямованих на запобігання втрат, відтворення та модифікації даних;

транспортування даних – прийом та передача даних між віддаленими користувачами інформаційного процесу. Джерело даних прийнято називати *сервером*, а споживача – *клієнтом*;

перетворення даних – перетворення даних з однієї форми в іншу, або з однієї структури в іншу, або зміна типу носія.

Будь-яка інформація, яка зберігається на комп'ютері, зберігається у вигляді файлу.

Файл – іменована типізована інформація, збережена на носії за відомою адресою. Файл повинен мати наступні ознаки: *містить інформацію*, *має тип* (вказує тип інформації, яка міститься у файлі), *має ім'я*, *відома адреса файлу*.

Структура будь-якого файлу відповідає вимогам файлової системи.

Файлова система – це набір правил і конструкцій, які описують те, як зберігаються файли на носії.

В інформатиці поняття "система" найчастіше використовують стосовно набору технічних засобів і програм. Системою називають також апаратну частину комп'ютера. Доповнення поняття "система" словом "інформаційна" відображає мету її створення та функціонування.

Інформаційна система – сукупність технічного, програмного та організаційного забезпечення, а також персоналу, яка призначена для того, щоб своєчасно забезпечувати належних людей належної інформацією.

Сучасне розуміння інформаційної системи передбачає використання комп'ютера як основного технічного засобу обробки інформації. Комп'ютери, які оснащені спеціалізованими програмними засобами, є технічної базою та інструментом інформаційної системи.

Переважна більшість інформаційних систем працює в режимі діалогу з користувачем. Типові програмні компоненти інформаційних систем включають: *діалогову підсистему введення-виведення*, *підсистему, яка реалізує*

логіку діалогу, підсистему прикладної логіки обробки даних, підсистему логіки управління даними.

Крім програмної складової інформаційних систем, важливу роль відіграє інформаційна складова, яка задає структуру, атрибутику та типи даних, а також тісно пов'язана з логікою управління даними.

Інформаційні технології – широкий клас дисциплін та галузей діяльності, що відносяться до технологій керування та обробки даних, а також створення даних, в тому числі, із застосуванням обчислювальної техніки.

Останнім часом під інформаційними технологіями найчастіше розуміють *комп'ютерні технології*. Зокрема, ІТ мають справу з використанням комп'ютерів і програмного забезпечення для зберігання, перетворення, захисту, обробки, передачі і отримання інформації.

Самі ІТ вимагають складної підготовки, великих первісних витрат і наукомісткої техніки. Їх впровадження повинно починатися зі створення математичного забезпечення, формування інформаційних потоків у системах підготовки фахівців.

1.2. Пристрої сучасних ПК

Комп'ютер, електронна обчислювальна машина (ЕОМ) – обчислювальний електронний пристрій, призначений для передачі, зберігання та обробки інформації.

Комп'ютер може відображати лише інформацію, представлену в числовій формі. Інформацію в іншій формі для введення в комп'ютер потрібно перетворити на числову форму.

Архітектура ЕОМ включає в себе як структуру, що відбиває склад ПК, так і програмно-математичне забезпечення. Структура ЕОМ – сукупність елементів і зв'язків між ними. Основним принципом побудови всіх сучасних ЕОМ є програмне керування.

Комп'ютер складається з декількох основних пристроїв (арифметико-логічний пристрій, керуючий пристрій, пам'ять, зовнішня пам'ять, пристрої

введення і виведення). Схематично загальна структура комп'ютера зображена на рис.1.1.

Арифметико-логічний пристрій – виконує логічні і арифметичні дії, необхідні для обробки інформації, що зберігається в пам'яті. Дані, які зберігаються в запам'ятовуючому пристрої, представлені в двійковій формі.

Керуючий пристрій – забезпечує управління та контроль всіх пристроїв комп'ютера (керуючі сигнали вказані пунктирними стрілками).

Арифметико-логічний пристрій і *керуючий пристрій* в сучасних комп'ютерах утворюють процесор ЕОМ.

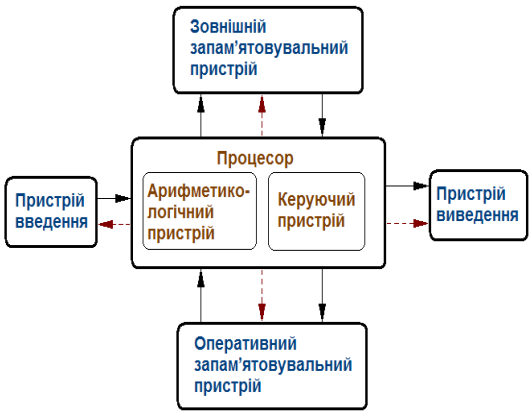


Рис. 1.1. *Загальна структура комп'ютера*

Процесор – пристрій, що відповідає за виконання арифметичних, логічних операцій і операцій керування. Процесор є перетворювачем інформації, яка надходить з пам'яті та зовнішніх пристроїв.

Запам'ятовувальні пристрої забезпечують зберігання вихідних і проміжних даних, результатів обчислень, а також програм. Вони включають: оперативні (ОЗП), надоперативні (Кеш-пам'ять), постійні (ПЗП) і зовнішні (ЗЗП) запам'ятовуючі пристрої.

Оперативні ЗП зберігають інформацію, з якою комп'ютер працює безпосередньо в даний час (резидентна частина операційної системи, дані, які опрацьовуються). У Кеш-пам'яті зберігаються дані, які найбільш часто використовуються процесором. Тільки та інформація, яка зберігається в Кеш-пам'яті і ОЗП, безпосередньо доступна процесору.

Зовнішні запам'ятовуючі пристрої (дискети, жорсткий диск, флеш-пам'ять, CD, DVD диски) з ємністю набагато більшою, ніж ОЗП, але з суттєво більш повільним доступом, використовуються для тривалого зберігання великих обсягів інформації. Наприклад, операційна система (ОС) зберігається

на жорсткому диску, але при запуску комп'ютера резидентна частина ОС завантажується в ОЗУ і знаходиться там до завершення сеансу роботи ПК.

ПЗП (постійні запам'ятовувальні пристрої) і ППЗП (постійні запам'ятовувальні пристрої, які можна перепрограмувати) призначені для постійного зберігання інформації, яка записується туди при її виготовленні, наприклад, ППЗП для BIOS.

БІОС – набір вбудованих програм для старту, перевірки, настроювання пристроїв і завантаження ОС, записаних у ПЗП комп'ютера.

Як *пристрій введення інформації* служить, наприклад, клавіатура. Як пристрій виводу – дисплей, принтер тощо.

1.3. Програмне забезпечення

Програмне забезпечення можна умовно розділити на три категорії: системне ПЗ, прикладне ПЗ, інструментальне ПЗ.

Системне ПЗ. Це програми загального користування, які не пов'язані з конкретним застосуванням ПК і виконують традиційні функції: планування і управління завданнями, управління вводом-виводом тощо. До системного ПЗ відносяться: *операційні системи, програми – оболонки*, *драйвери, утиліти*.

Прикладна програма призначена для виконання певних користувальницьких завдань (редагування текстових документів, створення малюнків, створення електронних таблиць тощо) і розрахована на безпосередню взаємодію з користувачем. У більшості операційних систем прикладні програми не можуть звертатися до ресурсів комп'ютера напряму, а взаємодіють з обладнанням та ін. за допомогою операційної системи.

Інструментальне ПЗ або системи програмування – це системи для автоматизації розробки нових програм на мові програмування.

1.4. Операційна система

Операційна система – комплекс керуючих і оброблювальних програм, які, з одного боку, виступають як інтерфейс між пристроями обчислювальної системи і прикладними програмами, а з іншого – призначені для керування

пристроями, управління обчислювальними процесами, ефективного розподілу обчислювальних ресурсів між обчислювальними процесами і організації надійних обчислень.

У більшості обчислювальних систем ОС є основною, найбільш важливою (а іноді єдиною) частиною системного ПЗ.

Список деяких операційних систем: Mac OS X, MS-Dos, Windows XP, Vista, 7, Novell NetWare, FreeBSD, Linux.

1.5. Архівні файли і антивірусний захист

Стиснення даних – процедура перекодування даних, яка виробляється з метою зменшення їх обсягу. Застосовується для більш раціонального використання пристроїв зберігання та передачі даних.

Стиснення буває *без втрат* (коли можливе відновлення вихідних даних без спотворень) або *з втратами* (відновлення можливе зі спотвореннями, несуттєвими з точки зору подальшого використання відновлених даних). Стиснення без втрат зазвичай використовується при обробці комп'ютерних програм і даних. Стиснення з втратами застосовується для скорочення обсягу звукової, фото- та відеоінформації, воно значно ефективніше за стиснення без втрат.

Зазвичай архівація проводиться в наступних випадках: *коли необхідно створити резервні копії найбільш важливих файлів*, *коли необхідно звільнити місце на диску*, *коли необхідно передати файли по E-mail*.

Архівний файл являє собою набір з декількох файлів або одного файлу, поміщених у стиснутому вигляді в єдиний файл, з якого їх можна при необхідності отримати в первинному вигляді. Архівний файл містить зміст, що дозволяє дізнатися, які файли містяться в архіві.

Архіватори – це програми (комплекс програм), які виконують стиснення і відновлення стиснених файлів у первісному вигляді. Процес стиснення файлів називається *архівуванням*. Процес відновлення стиснених файлів –

розархівування. Сучасні архіватори відрізняються використовуваними алгоритмами, швидкістю роботи (Zip, WinRAR).

Комп'ютерний вірус – різновид комп'ютерних програм, відмітною особливістю якої є здатність до самовідтворення (*самореплікація*). На додаток до цього віруси можуть без відома користувача виконувати інші довільні дії, в тому числі такі, що завдають шкоди користувачу і/або комп'ютеру. Відомі десятки тисяч комп'ютерних вірусів, які поширюються через Інтернет по всьому світу.

Існує багато різних способів класифікації комп'ютерних вірусів. Один із способів класифікації комп'ютерних вірусів – це поділ їх за такими основними ознаками: *середовище знаходження*, *особливості алгоритму*, *способи зараження*, *ступінь впливу* (нешкідливі, небезпечні, дуже небезпечні).

Залежно від середовища проживання основними типами комп'ютерних вірусів є: *програмні* (вражають файли з розширенням .COM и .EXE), *завантажувальні*, *макровіруси, мережеві віруси.*

Програмні віруси – це шкідливий програмний код, який впроваджений всередину виконуваних файлів (програм). Вірусний код може відтворювати себе в тілі інших програм – цей процес називається *самовідтворення*. Після певного часу, створивши достатню кількість копій, програмний вірус може перейти до руйнівних дій – порушення роботи програм і операційної системи, видаляючи інформацію, що зберігається на жорсткому диску. Цей процес називається *вірусною атакою*.

Завантажувальні віруси – вражають не програмні файли, а завантажувальний сектор магнітних носіїв (гнучких і жорстких дисків).

Макровіруси – вражають документи, які створені в прикладних програмах, що мають засоби для виконання макрокоманд. До таких документів відносять, наприклад, документи Word, Excel. Зараження відбувається при відкритті файлу документа у вікні програми, якщо в ній не відключена можливість виконання макрокоманд.

Мережеві віруси пересилаються з комп'ютера на комп'ютер, використовуючи для свого розповсюдження комп'ютерні мережі, електронну пошту та інші канали.

Нефахівці помилково відносять до комп'ютерних вірусів інші види шкідливих програм – *трояни, програми-шпигуни* і навіть *спам*.

Шкідлива програма – зловмисна програма, тобто програма, яка створена зі злим наміром.

Троянська програма – шкідлива програма, яка проникає на комп'ютер під виглядом нешкідливої – *кодека, скрінсейвера, хакерського ПЗ* тощо.

«Троянські коні» не мають власного механізму розповсюдження, і цим відрізняються від вірусів, які поширюються, прикріплюючи себе до нешкідливого ПЗ або документів. Втім, троянська програма може нести вірусне тіло.

Шпигунське програмне забезпечення (Spyware) – програма, яка потайливим чином встановлюється на комп'ютер з метою збору інформації про конфігурацію комп'ютера, користувача, активності користувача без згоди останнього. Також може виконувати інші дії: зміна налаштувань, установка програм без відома користувача, спрямування дій користувача.

Спам – масова розсилка кореспонденції, політичної та іншої реклами (інформації) або іншого виду повідомлень особам, які не висловили бажання її одержувати.

Способи захисту від комп'ютерних вірусів

Одним з основних способів боротьби з вірусами є своєчасна профілактика. Щоб запобігти зараженню вірусами і атаки троянських коней, необхідно виконувати деякі рекомендації:

- Не запускайте програми, отримані з Інтернету або у вигляді вкладення до повідомлення електронної пошти, без перевірки на наявність у них вірусу.

- Необхідно перевіряти всі зовнішні диски на наявність вірусів, перш ніж копіювати або відкривати файли, які містяться на них, або виконувати завантаження комп'ютера з таких дисків.
- Необхідно встановити антивірусну програму і регулярно користуватися нею для перевірки комп'ютерів. Оперативно поповнюйте базу даних антивірусної програми набором файлів сигнатур вірусів, як тільки з'являються нові сигнатури.
- Необхідно регулярно сканувати жорсткі диски в пошуках вірусів. При скануванні антивірусна програма шукає вірус шляхом порівняння коду програм з кодами відомих їй вірусів, що зберігаються в базі даних.
- Створювати надійні паролі, щоб віруси не могли легко підібрати пароль і отримати дозволи адміністратора.

Основний засіб захисту інформації – це резервне копіювання цінних даних, які зберігаються на жорстких дисках.

Існує досить багато програмних засобів антивірусного захисту. До найбільш ефективних і популярних відносяться: Антивірус Касперського, Norton AntiVirus, ESET NOD32, Dr.Web і багато інших.

Висновки

Інформатика – наука про способи отримання, накопичення, зберігання, перетворення, передачі і використання інформації. Вона включає дисципліни, так чи інакше пов'язані з обробкою інформації в обчислювальних машинах і обчислювальних мережах.

Інформаційна система – це система, яка організовує процеси збору, зберігання і обробки інформації про проблемну область. Вона може бути розміщена на одній або декількох комп'ютерних системах. Дані надходять в інформаційну систему і виключаються з неї, і ці взаємодії можуть здійснюватися або людьми, або процесами.

Інформаційні технології – прийоми, способи і методи застосування засобів обчислювальної техніки при виконанні функцій збору, зберігання, обробки, передачі і використання даних.

Структура ЕОМ – сукупність елементів і зв'язків між ними. Архітектура ЕОМ включає в себе як структуру, що відбиває склад ПК, так і програмно-математичне забезпечення.

Комп'ютерний вірус – різновид комп'ютерних програм, відмітною особливістю якої є самореплікація. Віруси можуть без відома користувача виконувати інші довільні дії, в тому числі завдавати шкоди користувачу і/або комп'ютеру. Віруси розповсюджуються, копіюючи своє тіло і забезпечуючи його подальше виконання: впроваджуючи себе у виконуваний код інших програм, замінюючи собою інші програми, прописуючись в автозапуск тощо.

Література [6, 8, 11-13]

Контрольні питання

1. Інформація та її властивості.
2. Призначення інформаційних систем.
3. Інформаційні технології. Основні риси сучасних ІТ.
4. Загальна структура комп’ютера
5. Файли та файлові системи комп’ютеру
6. Операційні системи, їх призначення, приклади.
7. Шкідливі програми та їх різновиди

Лекція 2. Основні відомості та поняття комп’ютерних мереж

Мета: викладення історії розвитку та характеристик комп’ютерних мереж.

План

2.1. Еволюція комп’ютерних мереж.

2.2. Найпростіша мережа з двох комп’ютерів.

2.3. Мережеве програмне забезпечення

2.4. Характеристики фізичних каналів

2.1. Еволюція комп’ютерних мереж

Комп'ютерна мережа (обчислювальна мережа, мережа передачі даних) – система зв'язку комп'ютерів та/або комп'ютерного обладнання (сервери, маршрутизатори та інше обладнання). Для передачі інформації можуть бути використані різні фізичні явища, як правило – різні види електричних сигналів, світлових сигналів чи електромагнітного випромінювання.

Комп'ютерні мережі є логічним результатом еволюції двох найважливіших науково-технічних галузей сучасної цивілізації – комп'ютерних і телекомунікаційних технологій.

З одного боку, мережі являють собою окремий випадок розподілених обчислювальних систем, у яких група комп'ютерів узгоджено вирішує набір взаємопов'язаних завдань, обмінюючись даними в автоматичному режимі. З іншого боку, комп'ютерні мережі можуть розглядатися як засіб передачі інформації на великі відстані, для чого в них застосовуються методи кодування і мультиплексування даних, що отримали розвиток в різних телекомунікаційних системах.

Перші комп'ютери 50-х років – великі, громіздкі і дорогі, які часто займали весь будинок – призначалися для дуже невеликої кількості вибраних користувачів. Такі комп'ютери не були призначені для інтерактивної роботи користувача, а застосовувалися в режимі пакетної обробки.

У міру здешевлення процесорів на початку 60-х років почали розвиватися інтерактивні багатотермінальні системи поділу часу. У таких системах кожен користувач отримував власний термінал, за допомогою якого він міг вести

діалог з комп'ютером. Кількість одночасно працюючих з комп'ютером користувачів визначалася його потужністю.

Термінали, вийшовши за межі обчислювального центру, розосередилися по всьому підприємству. І хоча обчислювальна потужність залишалася повністю централізованою, деякі функції, такі як введення і виведення даних, стали розподіленими. Подібні багатотермінальні централізовані системи зовні вже були дуже схожі на локальні обчислювальні мережі. Користувач міг отримати доступ до загальних файлів і периферійних пристроїв, при цьому у нього підтримувалася повна ілюзія одноосібного володіння комп'ютером, оскільки він міг запустити потрібну йому програму в будь-який момент і майже відразу ж отримати результат.

Потреба підприємств у створенні локальних мереж в цей час ще не дозріла – в одному приміщенні просто нічого було об'єднувати в мережу, так як через високу вартість обчислювальної техніки підприємства не могли собі дозволити розкіш придбання декількох комп'ютерів.

А ось потреба у поєднанні комп'ютерів, що знаходяться на великій відстані один від одного, до цього часу вже цілком назріла. Почалося все з вирішення більш простої задачі – доступу до комп'ютера з терміналів, віддалених від нього на багато сотень, а то й тисячі кілометрів. Термінали сполучалися з комп'ютерами через телефонні мережі за допомогою модемів. Потім з'явилися системи, в яких нарівні з віддаленими з'єднаннями типу *термінал-комп'ютер* були реалізовані і віддалені зв'язки типу *комп'ютер-комп'ютер*.

Комп'ютери отримали можливість обмінюватися даними в автоматичному режимі, що, власне, і є базовою ознакою будь-якої обчислювальної мережі.

На основі такого механізму в перших мережах були реалізовані *служби обміну файлами, синхронізації баз даних, електронної пошти* та інші, що стали тепер традиційними, мережеві служби.

Глобальні мережі. Таким чином, хронологічно першими з'явилися

глобальні мережі (Wide Area Network, WAN), тобто мережі, що об'єднують територіально розосереджені комп'ютери, які, можливо, перебувають у різних містах і країнах. Комп'ютери отримали можливість обмінюватися даними в автоматичному режимі.

Оскільки прокладка високоякісних ліній зв'язку на великі відстані обходиться дуже дорого, то глобальні мережі будувалися на основі телефонних каналів тональної частоти, здатних в кожний момент часу вести передачу лише однієї розмови в аналоговій формі. Оскільки швидкість передачі дискретних комп'ютерних даних по таким каналам була дуже низькою, набір послуг, що надаються в глобальних мережах такого типу, зазвичай обмежувався передачею файлів і електронною поштою.

У 1969 році Міністерство оборони США ініціювало роботи з об'єднання в єдину мережу суперкомп'ютерів оборонних і науково-дослідних центрів. Ця мережа, названа ARPANET, стала відправною точкою для створення першої і найвідомішої нині глобальної мережі – **Інтернет**.

Мережа ARPANET об'єднувала комп'ютери різних типів, які працювали під управлінням різних операційних систем (ОС) з додатковими модулями, що надають комунікаційні протоколи, загальні для всіх комп'ютерів мережі. ОС цих комп'ютерів можна вважати *першими мережевими операційними системами*.

Програмні модулі, що реалізують мережеві функції, з'являлися в операційних системах поступово, у міру розвитку мережевих технологій, апаратної бази комп'ютерів і виникнення нових завдань, що вимагають мережевої обробки.

Прогрес глобальних комп'ютерних мереж багато в чому визначався прогресом телефонних мереж. З кінця 60-х років в телефонних мережах все частіше стала застосовуватися передача голосу в цифровій формі. Це призвело до появи високошвидкісних цифрових каналів, які з'єднують автоматичні телефонні станції (АТС) і дозволяють одночасно передавати десятки і сотні розмов.

До теперішнього часу глобальні мережі за різноманітністю і якістю послуг наздогнали локальні мережі, які довгий час лідирували в цьому відношенні, хоча і з'явилися значно пізніше.

Локальні мережі. На початку 70-х років відбулася важлива подія, що вплинула на еволюцію комп'ютерних мереж – з'явилися великі інтегральні схеми. Їх порівняно невисока вартість і хороші функціональні можливості привели до створення міні-комп'ютерів. Тепер навіть невеликі підрозділи підприємств могли мати власні комп'ютери. Міні-комп'ютери вирішували завдання управління технологічним обладнанням, складом і інші задачі рівня відділу підприємства.

Таким чином, з'явилася концепція розподілу комп'ютерних ресурсів по всьому підприємству, а також необхідність обмінюватися комп'ютерними даними з користувачами інших підрозділів. Відповіддю на цю потребу стала поява перших локальних обчислювальних мереж.

Локальна мережа (Local Area Network, LAN) являє собою комунікаційну систему, яка належить одній організації.

Локальні мережі – це об'єднання комп'ютерів, зосереджених на невеликій території, зазвичай в радіусі не більше 1-2 км, хоча в окремих випадках локальна мережа може мати і великі розміри.

На перших порах для з'єднання комп'ютерів один з одним використовувалися нестандартні мережеві технології.

Мережева технологія – це погоджений набір програмних і апаратних засобів (наприклад, драйверів, мережевих адаптерів, кабелів і роз'ємів), а також механізмів передачі даних по лініях зв'язку, достатній для побудови обчислювальної мережі.

Різноманітні устрої поєднання використовували власні способи представлення даних на лініях зв'язку, свої типи кабелів і т. п., і тому могли з'єднувати тільки ті конкретні моделі комп'ютерів, для яких були розроблені.

У середині 80-х років утвердилися стандартні мережеві технології об'єднання комп'ютерів у мережу – Ethernet, Arcnet, Token Ring, Token Bus та

інші.

Потужним стимулом для їх появи були персональні комп'ютери. З одного боку, вони були достатньо потужними, щоб забезпечувати роботу мережевого програмного забезпечення, а з іншого – явно мали потребу об'єднання своєї обчислювальної потужності для вирішення складних завдань, а також розділення дорогих периферійних пристроїв і дискових масивів. Тому персональні комп'ютери стали переважати в локальних мережах, причому не тільки як клієнтські комп'ютери, але і як центри зберігання і обробки даних, тобто мережевих серверів, потіснивши з цих звичних ролей міні-комп'ютери і мейнфрейми.

Кінець 90-х виявив явного лідера серед технологій локальних мереж – сімейство *Ethernet*, до якого увійшли класична технологія *Ethernet* зі швидкістю передачі 10 Мбіт/с, а також *Fast Ethernet* зі швидкістю 100 Мбіт/с і *Gigabit Ethernet* зі швидкістю 1000 Мбіт/с.

З появою цифрових каналів зв'язку в глобальних мережах істотно підвищилися якість і швидкість передачі даних по ним. Поступово відмінності між локальними та глобальними мережевими технологіями стали згладжуватися. Ізольовані раніше локальні мережі почали об'єднувати один з одним, при цьому для з’єднання їх між собою використовувалися глобальні мережі. Тісна інтеграція локальних і глобальних мереж призвела до значного поєднання відповідних технологій.

Починаючи з 90-х років, комп'ютерні глобальні мережі, що працюють на основі швидкісних цифрових каналів, істотно розширили спектр послуг, які надаються, і наздогнали в цьому відношенні локальні мережі. Стало можливим створення служб, робота яких пов'язана з доставкою користувачеві великих обсягів інформації в реальному часі – зображень, відеофільмів, голосу, загалом всього того, що отримало назву мультимедійної інформації. Найбільш яскравий приклад – гіпертекстова інформаційна служба World Wide Web, що стала основним постачальником інформації в Інтернеті. Її інтерактивні можливості перевершили можливості багатьох аналогічних служб локальних мереж, отже,

розробникам локальних мереж довелося просто запозичити цю службу у глобальних мереж. Процес перенесення технологій з глобальної мережі Інтернет в локальні набув такого масового характеру, що з'явився навіть спеціальний термін - **intranet-технології** (intra - внутрішній).

2.2. Найпростіша мережа з двох комп'ютерів

Головною метою об'єднання комп'ютерів у мережу є поділ ресурсів: *користувачі комп'ютерів, підключених до мережі*, або *програми, які виконуються на цих комп'ютерах*, отримують можливість автоматичного доступу до різноманітних ресурсів інших комп'ютерів мережі, до числа яких належать:

- периферійні пристрої, такі як диски, принтери, сканери та ін.;
- дані, що зберігаються в оперативній пам'яті або на зовнішніх запам'ятовуючих пристроях;
- обчислювальна потужність.

Щоб забезпечити користувачів різних комп'ютерів можливістю спільного використання ресурсів мережі, комп'ютери необхідно оснастити якимись додатковими мережевими засобами.

Розглянемо найпростішу мережу, що складається з двох комп'ютерів, до одного з яких підключений принтер (рис. 2.1). Які додаткові засоби мають бути передбачені в обох комп'ютерах, щоб з принтером міг працювати не тільки користувач комп'ютера В, до якого цей принтер безпосередньо підключений, але і користувач комп'ютера А?

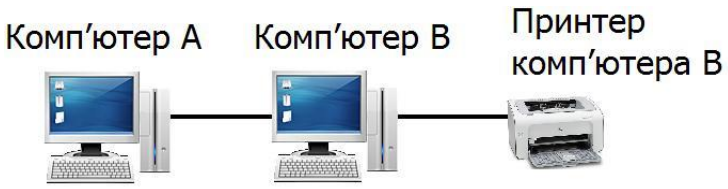


Рис.2.1. Найпростіша мережа.

Для зв'язку пристроїв у них, насамперед, повинні бути передбачені зовнішні інтерфейси.

Інтерфейс – формально визначена логічна та/або фізична межа між

взаємодіючими незалежними об'єктами. Інтерфейс задає параметри, процедури і характеристики взаємодії об'єктів.

Розділяють *фізичний* і *логічний* інтерфейси.

Фізичний інтерфейс (званий також *портом*) – визначається набором електричних зв'язків та характеристиками сигналів. Зазвичай він представляє собою роз'єм з набором контактів, кожен з яких має певне призначення. Пара роз'ємів з'єднується кабелем, що складається з набору проводів, кожен з яких з'єднує відповідні контакти. У таких випадках говорять про створення лінії, або каналу зв'язку між двома пристроями.

Логічний інтерфейс (або *протокол*) – це набір інформаційних повідомлень певного формату, якими обмінюються два пристрої або дві програми, а також набір правил, що визначають логіку обміну цими повідомленнями.

На рис. 2.2 ми бачимо інтерфейси двох типів: *комп'ютер – комп'ютер* і *комп'ютер – периферійний пристрій*.

Інтерфейс *комп'ютер-комп'ютер* дозволяє двом комп'ютерам обмінюватися інформацією. З кожного боку він реалізується парою:

- апаратним модулем, званим *мережевим адаптером*, або мережевою інтерфейсною картою;
- *драйвером* мережевої інтерфейсної карти – спеціальною програмою, що керує роботою мережевої інтерфейсної карти.

Інтерфейс *комп'ютер-периферійний пристрій* (в даному випадку інтерфейс *комп'ютер-принтер*) дозволяє комп'ютеру керувати роботою периферійного пристрою (ПП). Цей інтерфейс реалізується:

- з боку комп'ютера – інтерфейсною картою і драйвером ПП (принтера), подібним мережевій інтерфейсній карті і її драйверу;
- з боку ПП – контролером ПП (принтера), який зазвичай представляє собою апаратний пристрій, що приймає від комп'ютера як дані, наприклад, інформацію, яку потрібно роздрукувати на папері, так і команди, які він відпрацьовує, керуючи електромеханічними частинами

периферійного пристрою, наприклад, виштовхуючи лист паперу з принтера або переміщаючи магнітну головку диска.

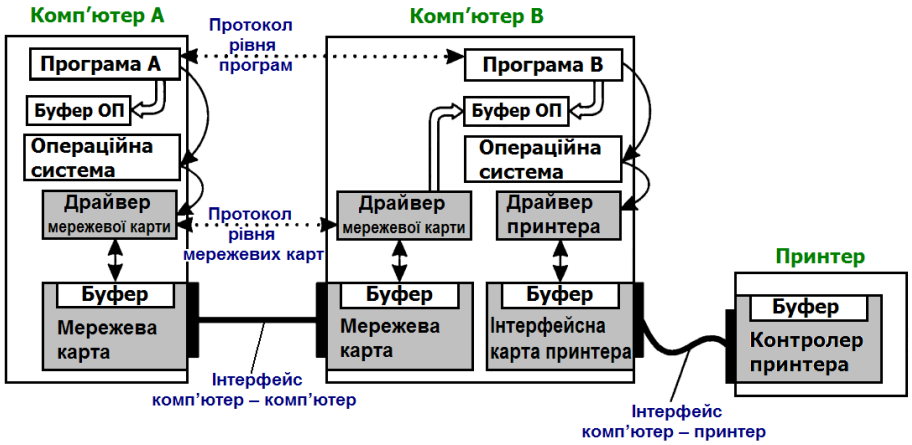


Рис.2.2. Спільне використання принтера в комп'ютерної мережі.

2.3. Мережеве програмне забезпечення.

Ми розглянули випадок спільного використання принтера в найпростішій мережі, що складається тільки з двох комп'ютерів. Однак навіть на цьому початковому етапі ми вже можемо зробити деякі висновки щодо будови мережевого програмного забезпечення: мережевих служб, мережевої операційної системи та мережевих програм.

Мережеві служби та сервіси. Потреба в доступі до віддаленого принтера може виникати у користувачів самих різних програм: текстового редактора, графічного редактора, системи управління базою даних (СУБД). Очевидно, що дублювання в кожній з програм загальних для всіх них функцій з організації віддаленого друку є надлишковим.

Більш ефективним є підхід, при якому ці функції виключаються з програм і оформляються у вигляді пари спеціалізованих програмних модулів – *клієнта* і *сервера* друку. Ця пара клієнт-сервер може бути використана будь-якою програмою, що виконується на комп'ютері А. Узагальнюючи такий підхід стосовно інших типів поділюваних ресурсів, наведемо такі визначення:

Клієнт – це модуль, призначений для формування та передачі повідомлень-запитів до ресурсів віддаленого комп'ютера від різних програм з наступним прийомом результатів з мережі і передачею їх до відповідних

програм.

Сервер – це модуль, який постійно очікує приходу з мережі запитів від клієнтів і, прийнявши запит, намагається його обслужити, як правило, за участі локальної ОС; один сервер може обслуговувати запити відразу декількох клієнтів (по черзі або одночасно).

Пара клієнт-сервер, що надає доступ до конкретного типу ресурсу комп'ютера через мережу, утворює *мережеву службу*.

Кожна служба пов'язана з певним типом мережевих ресурсів. Так, наприклад, модулі клієнта і сервера, що реалізують віддалений доступ до принтера, утворюють *мережеву службу друку*.

Файлова служба дозволяє отримувати доступ до файлів, що зберігаються на диску інших комп'ютерів. Серверний компонент файлової служби називають *файл-сервером*.

Для пошуку та перегляду інформації в Інтернеті використовується *веб-служба*, що складається з веб-сервера і клієнтської програми, званої *веб-браузером*. Ресурсом в даному випадку є *веб-сайт* – певним чином організований набір файлів, які містять пов'язану в смисловому плані інформацію і зберігаються на зовнішньому накопичувачі веб-сервера.

Два комп'ютери можуть бути пов'язані не безпосередньо, а через безліч проміжних комп'ютерів та інших мережевих пристроїв, що входять до складу Інтернету. Обмін повідомленнями між клієнтською і серверною частинами веб-служби виконується за стандартним протоколом HTTP і ніяк не залежить від того, чи передаються ці повідомлення від інтерфейсу одного комп'ютера до інтерфейсу іншого або через велике число посередників – *транзитних комунікаційних пристроїв*. Разом з тим, ускладнення середовища передачі повідомлень призводить до виникнення нових додаткових завдань, на вирішення яких не був розрахований згадуваний раніше найпростіший драйвер мережевої карти. Замість нього на взаємодіючих комп'ютерах повинні бути встановлені більш розвинені програмні транспортні засоби.

Мережевою операційною системою називають операційну систему

комп'ютера, яка, крім управління локальними ресурсами, надає користувачам і програмам ефективний і зручний доступ до інформаційних і апаратних ресурсів інших комп'ютерів мережі.

Сьогодні практично всі операційні системи є мережевими.

З прикладів, розглянутих вище, ми бачимо, що віддалений доступ до мережевих ресурсів забезпечується:

- мережевими службами;
- *засобами транспортування повідомлень по мережі* (в найпростішому випадку – мережними інтерфейсними картами і їх драйверами).

Отже, саме ці функціональні модулі повинні бути додані до ОС, щоб вона могла називатися мережевою.

Крім мережевих служб, мережева ОС повинна включати програмні комунікаційні (транспортні) засоби, що забезпечують спільно з апаратними комунікаційними засобами передачу повідомлень, якими обмінюються клієнтські і серверні частини мережевих служб. Задачу комунікації між комп'ютерами мережі вирішують драйвери і протокольні модулі. Вони виконують такі функції, як формування повідомлень, розбиття повідомлення на частини, перетворення імен комп'ютерів у числові адреси, дублювання повідомлень у разі їх втрати, визначення маршруту в складній мережі і т.д.

І мережеві служби, і транспортні засоби можуть бути невід'ємними компонентами ОС або існувати у вигляді окремих програмних продуктів.

Мережева служба може бути представлена в ОС або обома (клієнтською і серверною) частинами, або тільки однією з них.

У першому випадку операційна система, яка називається *одноранговою*, не тільки дозволяє звертатися до ресурсів інших комп'ютерів, але й надає власні ресурси в розпорядження користувачів інших комп'ютерів.

Операційна система, яка переважно містить клієнтські частини мережевих служб, називається *клієнтською*. Клієнтські ОС встановлюються на комп'ютери, які звертаються із запитами до ресурсів інших комп'ютерів мережі.

За такими комп'ютерами, також званими клієнтськими, працюють рядові користувачі.

До іншого типу операційних систем відноситься *серверна ОС* – вона орієнтована на обробку запитів з мережі до ресурсів свого комп'ютера і включає в себе в основному серверні частини мережевих служб. Комп'ютер з встановленою на ньому серверною ОС, що займається виключно обслуговуванням запитів інших комп'ютерів, називають виділеним сервером мережі. За виділеним сервером, як правило, звичайні користувачі не працюють.

2.4. Характеристики фізичних каналів.

Існує велика кількість характеристик, пов'язаних з передачею трафіку через фізичні канали. Розглянемо основні з них.

Запропоноване навантаження – це потік даних, що надходить від користувача на вхід мережі. Запропоноване навантаження можна характеризувати швидкістю надходження даних в мережу в бітах в секунду.

Швидкість передачі даних – це фактична швидкість потоку даних, що пройшов через мережу. Ця швидкість може бути меншою, ніж швидкість запропонованого навантаження, так як дані в мережі можуть спотворюватися чи губитися.

Ємність каналу зв'язку (пропускна здатність) являє собою максимально можливу швидкість передачі інформації з каналу.

Специфікою цієї характеристики є те, що вона відображає не тільки параметри фізичного середовища передачі, а й особливості обраного способу передачі дискретної інформації по цьому середовищі. Наприклад, ємність каналу зв'язку в мережі *Ethernet* на оптичному волокні дорівнює 10 Мбіт/с. Технологія *Fast Ethernet* забезпечує передачу даних по тому ж оптичному волокну з максимальною швидкістю 100 Мбіт/с, а технологія *Gigabit Ethernet* – 1000 Мбіт/с.

Висновки

Комп'ютерна мережа – система зв'язку комп'ютерів та/або комп'ютерного обладнання.

Глобальні мережі (Wide Area Network, WAN) об'єднують територіально розосереджені комп'ютери, які можливо перебувають у різних містах і країнах.

Локальна мережа (Local Area Network, LAN) являє собою комунікаційну систему, яка належить одній організації. Локальні мережі об'єднують комп'ютери, що зосереджені на невеликій території.

Інтерфейс – формально визначена логічна та/або фізична межа між взаємодіючими незалежними об'єктами. Інтерфейс задає параметри, процедури і характеристики взаємодії об'єктів. Розділяють *фізичний* і *логічний* інтерфейси.

Фізичний інтерфейс (або *порт*) – визначається набором електричних зв'язків та характеристиками сигналів.

Логічний інтерфейс (або *протокол*) – це набір інформаційних повідомлень певного формату, якими обмінюються два пристрої або дві програми, а також набір правил, що визначають логіку обміну цими повідомленнями.

Пара клієнт-сервер, що надає доступ до конкретного типу ресурсу комп'ютера через мережу, утворює *мережеву службу*.

Література [2, 7-11]

Контрольні питання

1. Комп’ютерна мережа.
2. Локальна і глобальна мережі.
3. Мережева технологія.
4. Сімейство мережевих технологій *Ethernet*
5. Мережевий протокол
6. Мережеві служби та сервіси.
7. Які характеристики мають фізичні канали зв’язку?

Лекція 3. Загальні принципи побудови комп’ютерних мереж

Мета: викладення механізмів фізичної реалізації мережних технологій та принципів побудови комп’ютерних мереж.

План

3.1. Топологія фізичних зв’язків

3.2. Адресація вузлів мережі

3.3. Задача комутації

3.4. Модель взаємодії відкритих систем OSI

3.1. Топологія фізичних зв’язків.

При об'єднанні в мережу значної кількості комп'ютерів виникає цілий комплекс нових проблем. Необхідно вирішити, яким чином з'єднати їх один з одним, іншими словами, вибрати конфігурацію фізичних зв'язків, або *топологію*.

Кількість можливих варіантів конфігурації різко зростає при збільшенні числа зв'язувальних пристроїв. Ми можемо поєднувати кожен комп'ютер з кожним або ж зв'язувати їх послідовно, припускаючи, що вони будуть спілкуватися, передаючи повідомлення один одному «транзитом». Транзитні вузли повинні бути оснащені спеціальними засобами, що дозволяють їм виконувати цю специфічну посередницьку операцію. В якості транзитного вузла може виступати як універсальний комп'ютер, так і спеціалізований пристрій.

У сучасних комп'ютерних мережах найпоширенішими є такі топології.

Повнозв'язна топологія (рис. 2.3, а) відповідає мережі, в якій кожен комп'ютер безпосередньо пов'язаний з усіма іншими. Незважаючи на логічну простоту, цей варіант є громіздким і неефективним. Дійсно, в такому випадку кожен комп'ютер в мережі повинен мати велику кількість комунікаційних портів, достатню для зв'язку з кожним з інших комп'ютерів мережі. Для кожної пари комп'ютерів повинна бути виділена окрема фізична лінія зв'язку. Найчастіше цей вид топології використовується в багатомашинних комплексах

або в мережах, які об'єднують невелику кількість комп'ютерів.

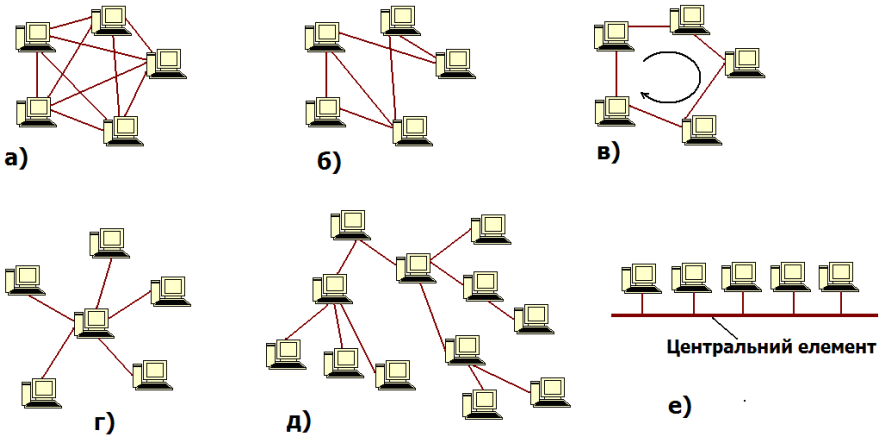


Рис.2.3. Типові топології мереж.

Комірчаста топологія виходить з повнозв'язної шляхом видалення деяких зв'язків (рис. 2.3, б). Комірчаста топологія допускає з'єднання великої кількості комп'ютерів і характерна, як правило, для великих мереж.

У мережах з кільцевою топологією (рис. 2.3, в) дані передаються по кільцю від одного комп'ютера до іншого. Головним достоїнством кільця є те, що воно за своєю природою забезпечує резервування зв'язків, тому будь-яка пара вузлів з'єднана тут двома шляхами – за годинниковою стрілкою і проти неї. У той же час в мережах з кільцевою топологією необхідно приймати спеціальні заходи, щоб у разі виходу з ладу або відключення будь-якого комп'ютера не переривався канал зв'язку між іншими вузлами кільця.

Зіркоподібна топологія (рис. 2.3, г) утворюється у випадку, коли кожен комп'ютер підключається безпосередньо до загального центрального пристрою, що зветься концентратором. У функції концентратора входить напрямок переданої комп'ютером інформації одному або всім іншим комп'ютерам мережі. У якості концентратора може виступати як універсальний комп'ютер, так і спеціалізований пристрій. До недоліків зіркоподібної топології відноситься більш висока вартість мережевого обладнання внаслідок необхідності придбання спеціалізованого центрального пристрою. Крім того, можливості з нарощування кількості вузлів у мережі обмежуються кількістю портів концентратора.

Іноді має сенс будувати мережу з використанням декількох

концентраторів, ієрархічно сполучених між собою зіркоподібними зв'язками (рис. 2.3, д). Одержану в результаті структуру називають ієрархічної зіркою, або деревом. В даний час дерево є найпоширенішою топологією зв'язків, як в локальних, так і в глобальних мережах.

Особливим окремим випадком зірки є загальна шина (рис. 2.3, е). Тут у якості центрального елементу виступає пасивний кабель, до якого підключається декілька комп'ютерів (таку ж топологію мають багато мереж, що використовують бездротовий зв'язок, – роль загальної шини тут відіграє загальна радіосереда). Передана інформація поширюється по кабелю і доступна одночасно всім комп'ютерам, приєднаним до цього кабелю. Основними перевагами такої схеми є її дешевизна і простота приєднання нових вузлів до мережі, а недоліками – низька надійність (будь-який дефект кабелю повністю паралізує всю мережу) і невисока продуктивність (в кожен момент часу тільки один комп'ютер може передавати дані по мережі, тому пропускна здатність ділиться тут між усіма вузлами мережі).

У той час як невеликі мережі, як правило, мають типову топологію – *зірка*, *кільце* або *загальна шина*, для великих мереж характерна наявність довільних зв'язків між комп'ютерами. У таких мережах можна виділити окремі довільно пов'язані фрагменти (підмережі), що мають типову топологію, тому їх називають мережами зі змішаною топологією.

3.2. Адресація вузлів мережі

Ще однією новою проблемою, яку потрібно враховувати при об'єднанні комп'ютерів, є проблема їх адресації, точніше, адресації їх мережевих інтерфейсів. Один комп'ютер може мати декілька мережевих інтерфейсів.

Адреси можуть бути числовими (наприклад, 129.26.255.255) і символьними (site.domen.ru). Символьні адреси (імена) призначені для запам'ятовування людьми і тому звичайно несуть смислове навантаження.

Безліч всіх адрес, які є допустимими в рамках деякої схеми адресації, називається *адресним простором*. Адресний простір може мати *плоску* (лінійну) або *ієрархічну* організацію.

При плоскій організації адресний простір ніяк не структурований. Прикладом плоскої числової адреси є MAC-адреса, призначена для однозначної ідентифікації мережевих інтерфейсів в локальних мережах. Така адреса звичайно використовується тільки апаратурою. При завданні MAC-адреси не потрібне виконання ручної роботи, так як вона зазвичай вбудовується в апаратуру компанією-виробником.

При ієрархічній організації адресний простір структурується у вигляді вкладених одна в одну підгруп, які, послідовно звужуючи адресну область, зрештою визначають окремий мережевий інтерфейс.

Ієрархічна адресація в багатьох випадках виявляється більш раціональною, ніж плоска. У великих мережах, які складаються з багатьох тисяч вузлів, використання плоских адрес призводить до великих витрат – кінцевим вузлам і комунікаційному обладнанню доводиться оперувати таблицями адрес, які складаються з тисяч записів. На противагу цьому ієрархічна система адресації дозволяє при переміщенні даних до певного моменту користуватися тільки старшою складовою адреси, потім для подальшої локалізації адресата – задіяти наступну за старшинством частину, і в кінцевому рахунку – молодшу частину.

Типовими представниками ієрархічних числових адрес є *IP-адреси*. У них підтримується двохрівнева ієрархія, адреса поділяється на старшу частину – номер мережі та молодшу – номер вузла. Такий поділ дозволяє передавати повідомлення між мережами тільки на підставі номера мережі, а номер вузла потрібен вже після доставки повідомлення у потрібну мережу.

На практиці звичайно застосовують відразу декілька схем адресації, так що мережевий інтерфейс комп'ютера може одночасно мати кілька адрес-імен. Кожна адреса задіюється в тій ситуації, коли відповідний вид адресації найбільш зручний. А для перетворення адрес з одного виду в інший використовуються спеціальні допоміжні протоколи, які називають *протоколами дозволу адрес.*

Кінцевою метою даних, що пересилаються по мережі, є не мережеві

інтерфейси або комп'ютери, а програми-процеси, які виконуються на цих пристроях. Тому в адресі призначення поряд з інформацією, що ідентифікує інтерфейс пристрою, повинна вказуватися адреса процесу, якій призначені дані, що посилаються по мережі. Прикладом адрес-процесів є номери портів TCP, які використовуються в стеку TCP/IP.

3.3. Задача комутації.

Нехай комп'ютери фізично пов'язані між собою відповідно до деякої топології і обрана система адресації. Залишається невирішеною ще одна проблема: яким способом передавати дані між кінцевими вузлами? Особливу складність набуває це завдання для неповнозв’язної топології мережі, коли обмін даними між довільною парою кінцевих вузлів (користувачів) повинен йти в загальному випадку через транзитні вузли.

З'єднання кінцевих вузлів через мережу транзитних вузлів називають комутацією. Послідовність вузлів, що лежать на шляху від відправника до одержувача, утворює маршрут.

У самому загальному вигляді задача комутації може бути представлена у вигляді таких взаємопов'язаних окремих завдань.

Визначення інформаційних потоків, для яких потрібно прокладати маршрути.

Маршрутизація потоків.

Просування потоків, тобто розпізнавання потоків і їх локальна комутація на кожному транзитному вузлі.

Мультиплексування і демультиплексування потоків.

Визначення інформаційних потоків. Зрозуміло, що через один транзитний вузол може проходити кілька маршрутів. Транзитний вузол повинен вміти розпізнавати потоки даних, які надходять на нього, для того щоб забезпечувати передачу кожного з них саме на той свій інтерфейс, який веде до потрібного вузла.

Інформаційним потоком, або потоком даних, називають безперервну послідовність даних, об'єднаних набором загальних ознак, що виділяють ці дані

із загального мережевого трафіку. Наприклад, як потік можна визначити всі дані, що надходять від одного комп'ютера; об'єднуючою ознакою в даному випадку служить адресу джерела. Ці ж дані можна представити як сукупність декількох підпотоків, кожен з яких в якості диференціюючої ознаки має адресу призначення. Нарешті, кожен з цих підпотоків, в свою чергу, можна розділити на більш дрібні підпотоки, які породжені різними мережевими програмами – електронною поштою, програмою копіювання файлів, веб-сервером.

Маршрутизація. Завдання маршрутизації включає в себе дві підзадачі:

- визначення маршруту;
- оповіщення мережі про обраний маршрут.

Визначити маршрут означає вибрати послідовність транзитних вузлів та їх інтерфейсів, через які треба передавати дані, щоб доставити їх адресату. Якщо конфігурація мережі така, що між парою взаємодіючих мережевих інтерфейсів існує безліч шляхів, то частіше за все вибір зупиняють на одному оптимальному по деякому критерію маршруті. В якості критеріїв оптимальності можуть виступати номінальна пропускна здатність, завантаженість каналів зв'язку і т.д.

Маршрут може визначатися «вручну» адміністратором мережі. Однак цей підхід до визначення маршрутів мало придатний для великої мережі зі складною топологією. У цьому випадку використовуються автоматичні методи визначення маршрутів. Для цього кінцеві вузли та інші пристрої мережі оснащуються спеціальними програмними засобами, які організовують взаємний обмін службовими повідомленнями, що дозволяє кожному вузлу скласти своє «уявлення» про мережу. Потім на основі зібраних даних програмними методами визначаються раціональні маршрути.

Після того як маршрут визначено, треба сповістити про нього всім пристроям мережі. Кожне повідомлення про маршрут обробляється пристроєм, в результаті створюється новий запис у таблиці комутації. У цій таблиці ознаці потоку ставиться у відповідність номер інтерфейсу, на який пристрій повинен передавати дані, що стосуються цього потоку.

Просування даних. Нехай маршрути визначені, записи про них зроблені в таблицях всіх транзитних вузлів, все готово до виконання основної операції – передачі даних між абонентами.

Для кожної пари абонентів ця операція може бути представлена декількома (по числу транзитних вузлів) локальними операціями комутації. Перш за все, відправник повинен виставити дані на той свій інтерфейс, з якого починається знайдений маршрут, а всі транзитні вузли повинні відповідним чином виконати «перекидання» даних з одного свого інтерфейсу на інший, іншими словами, виконати *комутацію інтерфейсів*. Пристрій, функціональним призначенням якого є комутація, називається комутатором.

Проте перш ніж виконати комутацію, комутатор повинен розпізнати потік. Для цього дані, які надійшли до нього, аналізуються на предмет наявності в них ознак будь-якого з потоків, заданих в таблиці комутації. Якщо стався збіг, то ці дані направляються на інтерфейс, визначений для них у маршруті.

Комутатором може бути як спеціалізований пристрій, так і універсальний комп'ютер з вбудованим програмним механізмом комутації, в цьому випадку комутатор називається програмним.

Демультиплексування – поділ сумарного агрегованого потоку на декілька складових його потоків.

Мультиплексування – складання з декількох окремих потоків загального агрегованого потоку, який передається по одному фізичному каналу зв'язку.

3.4. Модель взаємодії відкритих систем OSI

На початку 80-х років міжнародною організацією зі стандартизації (ISO - International Organization for Standartization) було розроблено модель взаємодії відкритих систем (OSI - Open System Interconnection).

Засоби взаємодії в моделі OSI діляться на сім рівнів: фізичний, канальний, мережевий, транспортний, сеансовий, представницький, прикладний.

Завдяки цьому завдання мережевої взаємодії розбивається на декілька більш дрібних завдань. Це дозволяє при розробці нових способів і інструментів мережевої взаємодії не розробляти їх наново цілком і повністю, а використовувати вже готові рішення, замінивши лише деякі їх частини.

Безпосередньо один з одним взаємодіють тільки фізичні рівні. Всі інші рівні безпосередньо взаємодіють тільки з вище- та нижчележачими рівнями: користуються послугами нижчележачого і надають послуги вищележачим. Один з одним такі рівні контактують у непрямий спосіб, за посередництвом нижчих рівнів.

У деяких випадках мережевої взаємодії фізичний рівень як такий відсутній, при цьому його функції виконує самий нижчележачий рівень.

У міру проходження повідомлення через рівні моделі OSI до даних, які пересилаються, додається службова інформація, яка свідчить про проходження даних через певний рівень.

Взаємодія між комп'ютерами зазвичай здійснюється за допомогою будь-яких програм, які володіють спеціальним набором функцій. Ці програми працюють на найвищому рівні моделі взаємодії – *прикладному*. Тому, коли ви вкажете, що хочете записати певні дані у файл, буде сформовано відповідне повідомлення. У полі даних цього повідомлення і буде утримуватися передана в файл інформація.

Після формування повідомлення з прикладного рівня буде передано на *представницький* рівень. На цьому рівні в заголовок додаються вказівки для представницького рівня комп'ютера-адресата. Потім повідомлення передається *сеансовому* рівню, який додає свою інформацію і т.д. Процес вкладення одного протоколу в інший називається *інкапсуляцією*. У процесі проходження вихідного блоку даних (повідомлення) за рівнями він розбивається на більш дрібні фрагменти для пересилання їх по мережі.

Коли повідомлення надходить на комп'ютер-адресат, воно приймається фізичним рівнем і передається вгору з рівня на рівень. Кожен рівень аналізує вміст заголовка свого рівня, виконує вказівки, які містяться в ньому, потім

видаляє інформацію, яка відноситься до себе, з заголовка і передає повідомлення далі до вищого рівня. Цей процес називається *декапсуляція*.

Висновки

При об'єднанні в мережу великої кількості комп'ютерів виникає цілий комплекс нових проблем. Необхідно вирішити, яким чином з'єднати їх один з одним, іншими словами, вибрати конфігурацію фізичних зв'язків, або *топологію*. У сучасних мережах найпоширенішими є такі топології: *повнозв'язна*, *комірчаста*, *кільцева*, *зіркоподібна*, *дерево*, *загальна шина*.

Адреси інтерфейсів можуть бути числовими (наприклад, 129.26.255.255) і символьними (site.domen.ru). Символьні адреси призначені для запам'ятовування людьми і тому несуть смислове навантаження.

Безліч всіх адрес, які є допустимими в рамках деякої схеми адресації, називається *адресним простором*. Адресний простір може мати *плоску* (лінійну) організацію або *ієрархічну* організацію.

З'єднання кінцевих вузлів через мережу транзитних вузлів називають комутацією. Послідовність вузлів, що лежать на шляху від відправника до одержувача, утворює маршрут. Кожне повідомлення про маршрут обробляється пристроєм, в результаті створюється запис у таблиці комутації.

Засоби взаємодії в моделі OSI діляться на сім рівнів: *фізичний, канальний, мережевий, транспортний, сеансовий, представницький, прикладний.* Це дозволяє при розробці нових способів і інструментів мережевої взаємодії не розробляти їх наново цілком і повністю, а використовувати вже готові рішення, замінивши лише деякі їх частини.

Література [2, 7-11]

Контрольні питання

1. Наведіть найпоширеніші топології фізичних зв’язків.
2. Організація адресного простору.
3. Що таке MAC-адреса?.

4. Чим відрізняється плоска організація адресного простору від ієрархічного?
5. Що таке комутація?
6. Які функції в мережі виконує комутатор?
7. Із яких рівнів складається модель OSI?

Лекція 4. Глобальна мережа Інтернет

Мета: з'ясувати принципи дії та призначення глобальної мережі Інтернет.

План

4.1. Основні поняття.

4.2. Постачальники послуг Інтернету.

4.3. Функціонування Інтернету

4.4. Прикладні сервіси Інтернету

4.1. Еволюція комп'ютерних мереж

Інтернет – найвідоміша реалізація складової мережі, яка охоплює весь світ і побудована на основі технології TCP/IP.

Основні особливості Інтернету.

Це найбільша в світі мережа: за кількістю користувачів, по території покриття, за сумарним обсягом переданого трафіку, за кількістю мереж, які входять до її складу.

Інтернет – це мережа, яка не має єдиного центру управління і в той же час працює за єдиними правилами і надає всім своїм користувачам єдиний набір послуг.

Інтернет – це «мережа мереж», але кожна мережа, яка входить в Інтернет, управляється незалежним оператором – *постачальником послуг Інтернету* (Internet Service Provider, ISP), або *провайдером*. Деякі центральні органи існують, але вони відповідають тільки за єдину технічну політику, за узгоджений набір технічних стандартів, за централізоване призначення таких життєво важливих для гігантської складової мережі параметрів, як імена та адреси комп'ютерів і мереж, що входять в Інтернет, але не за щоденну

підтримку мережі в працездатному стані.

Необмежне інформаційне наповнення та простота доступу до цієї інформації для всіх користувачів Інтернету – сотні тисяч терабайтів інформації, які зберігаються на веб-серверах Інтернету і доступні користувачам Інтернету у формі веб-сторінок. Зручна форма подання взаємозв'язків між окремими інформаційними фрагментами у вигляді гіперпосилань і стандартний графічний браузер, який однаково просто і ефективно працює у всіх популярних операційних системах, здійснили революцію. Інтернет став швидко заповнюватися найрізноманітнішою інформацією у формі веб-сторінок, перетворюючись одночасно в енциклопедію, щоденну газету, рекламне агентство і величезний магазин.

4.2. Постачальники послуг Інтернету

Загальний термін «провайдер», або постачальник послуг Інтернету (Internet Service Provider, ISP) зазвичай відносять до компаній, які виконують для кінцевих користувачів лише транспортну функцію – забезпечують передачу їх трафіка в мережі інших постачальників.

Постачальником інтернет-контенту (Internet Content Provider, ICP) називають такого провайдера, який має власні інформаційно-довідкові ресурси, надаючи їх зміст – контент – у вигляді веб-сайтів. Багато постачальників послуг Інтернету є одночасно постачальниками інтернет-контенту.

Постачальник послуг хостингу (Hosting Service Provider, HSP) – це компанія, яка надає своє приміщення, свої канали зв'язку та сервери для розміщення контенту, створеного іншими підприємствами.

Постачальники послуг з доставки контенту (Content Delivery Provider, CDP) – це підприємства, які не створюють інформаційного наповнення, а займаються доставкою контенту в численні точки доступу, максимально наближені до користувачів, що дозволяє підвищити швидкість доступу користувачів до інформації.

Постачальники послуг з підтримки стосунків (Application Service Provider, ASP) надають клієнтам доступ до великих універсальних програмних

продуктів, які самим користувачам складно підтримувати. Зазвичай це корпоративні користувачі, яких цікавлять програми класу управління підприємством, такі як SAP R3.

4.3. Функціонування Інтернету

У технічному розумінні TCP/IP – це не один мережевий протокол, а два протоколи, що лежать на різних рівнях (стек протоколів). Протокол TCP керує тим, як відбувається передача інформації. Протокол IP – адресний. Він визначає, куди відбувається передача.

Протокол TCP. Згідно з протоколом TCP, дані, що відправляються, «ріжуться» на невеликі пакети, після чого кожен пакет маркується таким чином, щоб у ньому були дані, необхідні для правильного складання документа на комп'ютері одержувача. Два комп'ютери, що пов'язані між собою одним фізичним з'єднанням, можуть підтримувати одночасно кілька TCP-з'єднань. Так, наприклад, два проміжних мережевих сервера можуть одночасно по одній лінії зв'язку передавати один одному в обидві сторони безліч TCP-пакетів від численних клієнтів.

Коли ми працюємо в Інтернеті, то по одній-єдиній телефонній лінії можемо одночасно приймати документи з Америки, Австралії і Європи. Пакети кожного з документів надходять порізно, з поділом у часі, і в міру надходження збираються в різні документи.

Протокол IP. Тепер розглянемо адресний протокол – IP (Internet Protocol). Його суть полягає в тому, що у кожного учасника Всесвітньої мережі повинна бути своя унікальна адреса. Без цього не можна говорити про точну доставку TCP-пакетів на потрібне робоче місце. Ця адреса виражається дуже просто – чотирма байтами, наприклад: 195.38.46.11. Структура IP-адреси організована так, що кожен комп'ютер, через який проходить який-небудь TCP-пакет, може по цих чотирьох числах визначити, кому з найближчих «сусідів» треба переслати пакет, щоб він виявився «ближче» до одержувача. У результаті кінцевого числа перекидань TCP-пакет досягає адресата.

Вирішенням питань, що вважати «ближче», а що «далі», займаються

спеціальні засоби – маршрутизатори. Роль маршрутизатора в мережі може виконувати як спеціалізований комп'ютер, так і спеціальна програма, що працює на вузловому сервері мережі.

4.4. Прикладні сервіси Інтернету

Більшість користувачів мереж не цікавлять подробиці реалізації мереж та різних протоколів. Для кінцевих користувачів інтерес становить те, які переваги може надати дана технологія. Розглянемо деякі мережні служби, які найчастіше використовуються, та їх функції.

4.4.1. DNS

Для ідентифікації хостів у мережі Інтернет використовується 32-розрядна десяткова IP-адреса з роздільними крапками. Це доцільно для мережі та маршрутизаторів, але користувачам зручніше пам'ятати імена. Для системних адміністраторів також краще, коли користувачі використовують імена, а не цифрові адреси, оскільки їм не треба повідомляти кожного користувача про зміни цифрових адрес у разі переміщення системи. Служба імен доменів (DNS, Domain Name System) являє собою розподілену службу імен у мережі Інтернет, що дає змогу хостам перетворювати Інтернет-імена в IP-адреси.

За допомогою служби DNS можна дізнатися IP-адресу хосту за його ім'ям (та навпаки). Розглянемо випадок, коли користувач бажає з'єднатися із сайтом, наприклад, www.dsum.edu.ua. Для цього потрібно відправити запит до локального сервера імен, щоб дізнатися, що IP-адреса цього хосту – 10.0.0.1. Насправді, зазвичай сценарій складніше, ніж просто запит/відповідь. Коли програмі (браузеру, наприклад) потрібно дізнатись адресу деякого хосту, вона відправляє відповідне повідомлення локальному серверу імен. Якщо ім'я, що запитується, невідомо локальному серверу, він зобов'язаний передати запит іншим серверам імен до тих пір, поки запит не буде виконано. Тільки після того, як ім'я буде трансльовано в адресу, локальний хост почне визначати маршрут до хосту призначення.

Усі назви хостів у мережі Інтернет побудовано за принципом загальної

ієрархії, яка читається з права на ліво. Права частина адреси являє собою домен верхнього рівня (TLD, Top Level Domain). Неповний список характерних доменів TLD містить: .com – комерційні організації;

.edu – освітні організації, в основному коледжі та університети;

.gov – урядові організації;

.org – некомерційні організації;

.net – хости мережних операторів;

.biz – бізнесові сайти;

.coop – кооперативні організації.

Інші домени верхнього рівня – це дволітерні коди країн, визначені стандартом 3166 ISO, наприклад, .ua – географічний домен України.

Найближча до останньої частина адреси означає ім'я домену другого рівня та присвоюється уповноваженою установою, пов'язаною з кожним доменом верхнього рівня. При використанні кодів країн імена присвоює відповідна уповноважена організація з надання інтернет-імен у країні. Інші частини адреси присвоює локальний інтернет-адміністратор, який відповідає за призначення імен приватним хостам.

Процес проходження реєстрації імені домену в мережі Інтернет значною мірою змінився за останні декілька років. До середини 1998 р. компанія NSI була єдиним офіційним органом, що надавав імена доменам. Компанія NSI пропонувала реєстрацію імені домену через інформаційний центр мережі Інтернет за 100 дол. У даний час є безліч органів реєстрації імен доменів. Щоб стати таким органом, потрібна акредитація організації ICANN (Internet Corporation for Assigned Names and Numbers).

4.4.2. Прикладні сервіси HTTP, FTP та SNMP FTP

Протокол передачі файлів FTP (File Transfer Protocol) дає змогу користувачам здійснювати доступ до віддалених файлових серверів, переглядати каталоги та переміщувати файли на/або з віддалених хостів. Протокол FTP було розроблено для передачі файлів між двома системами без надання користувачам усіх можливостей сеансу Telnet (наприклад, можливості

виконувати програми на віддаленому хості).

Протокол FTP підтримує окремі TCP-з'єднання для управління і передачі даних. FTP-команди та відповіді сервера передаються управляючим з'єднанням, а для перегляду кожного каталогу або передачі файлів сервер встановлює нове з'єднання передачі даних.

Із TCP/IP-додатків протокол FTP є одним з найчастіше використовуваних звичайним користувачем. Робота FTP з анонімним доступом особливо часто використовується на сайтах з архівами файлів, за допомогою яких користувачі одержують доступ до файлів без створення облікового запису на хості.

Здавалося, що з наступанням століття Всесвітньої павутини (World Wide Web) протокол FTP буде непотрібним, але насправді багато людей продовжують його використовувати, хоча і не завжди усвідомлюючи це. Деякі компанії, наприклад, виробники браузерів, використовують протокол FTP для розповсюдження нових версій своїх програм. Коли користувач обирає необхідний файл, браузер звертається до FTP-сервера, а не до web-сервера. Насправді, використовується анонімний доступ за протоколом FTP, браузер автоматично передає anonymous як ім'я користувача та адресу його електронної пошти - пароль. Перемикаючи передачу файлів на інший сервер, з'єднання з web-сервером лишаються вільними для передачі іншої інформації, що більше потрібна для web-доступу.

WWW та HTTP

На сьогодні існує чотири основних компоненти WWW: 1) покажчики URL (Uniform Resource Locator, уніфікований покажчик інформаційного ресурсу); 2) протокол HTTP (HyperText Transfer Protocol, протокол передачі гіпертекстових файлів); 3) HTTP-транзакції клієнт-сервер; серверна обробка інформації з використанням інтерфейсу CGI (Common Gateway Interface, загальний шлюзовий інтерфейс). Покажчики URL являють собою стандартну схему адреси, прийняту у WWW для визначення місцезнаходження інформації у мережі. Приклад стандартного формату URL – http://www.asus.com. Тут http:// означає, що буде використовуватися протокол HTTP для передачі інформації з

www-сервера, розташованого у домені другого рівня asus, який у свою чергу належить комерційному домену .com.

HTML (HyperText Markup Language) – це мова програмування, використовувана для представлення документів в універсальному, стандартизованому форматі, в якому вони можуть бути завантажені з web-сервера як гіпертекстові документи. Ця мова розмітки складається з команд форматування, які дають дозвіл користувачам встановлювати положення об'єкта на сторінці, розмір, колір, вирівнювання, відстань між заголовками, абзацами, анімацію, цитати, зображення, а також функціональні кнопки, що відкривають інші гіпертекстові документи.

Як і для всіх інших ресурсів WWW, розташування цих документів характеризується адресою URL. Ці URL можуть вказувати не тільки на текст, але також на відео, зображення, музичні файли і т.і. Якщо інформація може бути оцифрованою та збереженою на диску або CD-ROM, то на неї можна дати посилання за допомогою покажчика URL з використанням HTML-форматування і цю інформацію можна буде завантажити за допомогою протоколу HTTP на систему, яка її запитала.

Протокол HTTP є центральним елементом web. Протокол HTTP пропонує синтаксис команд, використовуваний для передачі документів від web-сереера до web-клієнта подібно протоколу FTP. Насправді, протокол HTTP працює за технологією клієнт-сервер. Між клієнтом та сервером здійснюються транзакції, в ході яких запитаний файл переміщується з сервера на комп'ютер клієнта.

4.4.3. Протоколи електронної пошти

Електронна пошта (e-mail) – один з сервісів, який найбільш часто використовується в мережі Інтернет. Насправді, починаючи з 1980-х років електронна пошта є важливішою часткою серед додатків будь-якого хосту, локальної мережі чи операційної системи.

Електронна пошта використовується для різних цілей. У даний час більшість великих компаній працюють з електронною поштою. Підключення системи електронної пошти компанії до такої мережі, як Інтернет, дає великі

переваги: співробітники можуть спілкуватися з колегами-професіоналами всього світу, а клієнти - з компанією. Розглянемо протоколи, потрібні для функціонування електронної пошти у мережі TCP.

SMTP

У протоколі SMTP (Simple Mail Transfer Protocol) об'єкт електронної пошти (лист) відправляється після ідентифікації відправника та одного чи декількох одержувачів. Коли процес SMTP-сервера приймає пошту для певного одержувача, він доставляє пошту користувачеві, якщо одержувач є локальним користувачем, або пересилає пошту відповідному хосту призначення, коли одержувач не є локальним користувачем. Протокол SMTP визначає поштовий сервер призначення даного одержувача по записах ME (Mail Exchange – обмін поштою) у службі DNS. При проходженні повідомлення крізь мережу, воно переносить із собою маршрут зворотного шляху, щоб у випадку невдалої передачі можна було повідомити про це відправника.

Основні функції протоколу SMTP передбачають ідентифікацію відправника, ідентифікацію одержувачів, передачу повідомлення і підтвердження доставки повідомлення.

POP

SMTP визначає протокол обміну поштовими повідомленнями між серверами електронної пошти. Протокол SMTP не надає користувачам прямого механізму доступу до їх пошти та її управління. Наприклад, SMTP не передбачає команд, за допомогою яких користувачі мали змогу відповідати на повідомлення, пересилати повідомлення іншим користувачам або відправляти копії повідомлень. Для цього було розроблено протокол POP (Post Office Protocol, поштовий протокол).

Цей протокол призначено для використання у локальній мережі. POP-сервер підтримує поштові скриньки користувачів. Будь-яка користувацька система має програмне забезпечення POP-клієнта з елементарним інтерфейсом, щоб користувачі мали доступ до поштових скриньок та керували своєю поштою.

Висновки

Інтернет – найвідоміша реалізація складової мережі, яка охоплює весь світ і побудована на основі технології TCP/IP.

Це найбільша в світі мережа: за кількістю користувачів, за територією покриття, за сумарним обсягом переданого трафіку, за кількістю мереж, які входять до її складу.

Інтернет – це мережа, яка не має єдиного центру управління і в той же час працює за єдиними правилами і надає всім своїм користувачам єдиний набір послуг. Кожна мережа, яка входить в Інтернет, управляється незалежним оператором – *постачальником послуг Інтернету* (Internet Service Provider, ISP), або *провайдером*.

Необмежне інформаційне наповнення та простота доступу до інформації для всіх користувачів Інтернету – сотні тисяч терабайтів інформації, які зберігаються на веб-серверах Інтернету і доступні користувачам Інтернету у формі веб-сторінок.

Література [2, 7-11]

Контрольні питання

1. Що таке Internet?
2. На якої технології основана мережа Internet?
3. Хто займається постачанням Internet послуг?
4. Кому належить Internet мережа?
5. Для чого використовується DNS служба?
6. Перелікуйте чотири основних компоненти WWW.

Змістовий модуль 2. Системи обробки інформації

Лекція 5. Системи обробки текстової інформації

Мета: Ознайомитись з основними системами та принципами обробки тексту.

План

5.1. Основні поняття;

5.2. Редактори документів;

5.3. Класифікація засобів обробки текстової інформації;

5.4. Редактори наукових документів;

5.5. Стилі;

5.1. Основні поняття.

Текст – text – форма подання даних у вигляді послідовності символів обраної мови, змістовно розглянутої як єдине ціле. У тексті найчастіше використається природна мова і його алфавіт. Однак, при описі програм застосовуються мови програмування. Зображення виступають як ілюстрації й форми надання додаткової до тексту інформації. Текст є основою документа.

Уведення тексту в систему може здійснюватися за допомогою клавіатури, світлового пера, мікрофона або сканера. Важливе значення в комерції, торгівлі, бухгалтерському обліку й т.д. має подання тексту у вигляді форматизованих бланків. Тексти передаються мережею у вигляді блоків даних. Особливе значення при поданні, обробці, пошуку інформації має методологія, іменована гіпертекстом. Усе більше широке поширення одержують електронні тексти.

Символ – character – те, що служить умовним знаком якого-небудь поняття, явища. Символом є цифра, буква, знак пунктуації або ієрогліф природної мови, розділовий знак, знак пробілу, спеціальний знак, символ операції. Крім цього, широко використаються керуючі символи. Серед наборів символів найбільше поширення одержали ті, які визначаються розширеним Американським стандартним кодом для інформаційного обміну ASCII і розширеним двоїчно-десятковим кодом EBCDIC. Частіше використовується універсальний код. Всі символи відображаються послідовностями бітів,

обумовленими обраним методом кодування.

Абзац – це група рядків, не розділених порожнім рядком. Порожній рядок служить для поділу частин документа (заголовків від змістовної частини, абзаців друг від друга й т.п.).

Зручність й ефективність застосування комп'ютерів для підготовки текстів привели до створення безлічі програм для обробки документів. Такі програми називаються редакторами текстів. Можливості цих програм різні – від програм, призначених для підготовки невеликих документів простої структури, до програм для набору, оформлення й повної підготовки до типографського видання книг і журналів (видавничі системи).

5.2. Редактори документів

Програми для обробки документів орієнтовані на роботу з текстами, що мають структуру документа, тобто складаються з розділів, сторінок, абзаців, речень, слів і т.д. Тому редактори для обробки документів можуть забезпечувати функції, орієнтовані на структуру документа, а саме:

- можливість використання різних шрифтів символів;
- роботу із пропорційними шрифтами;
- завдання довільних міжрядкових проміжків;
- автоматичний перенос слів на новий рядок;
- автоматичну нумерацію сторінок;
- обробку й нумерацію виносок;
- печатка верхніх і нижніх заголовків сторінок (колонтитулів);
- вирівнювання країв абзацу;
- набір тексту в кілька стовпців;
- створення таблиць і побудова діаграм;
- перевірку правопису й підбор синонімів;
- побудова змістів, індексів і т.д.

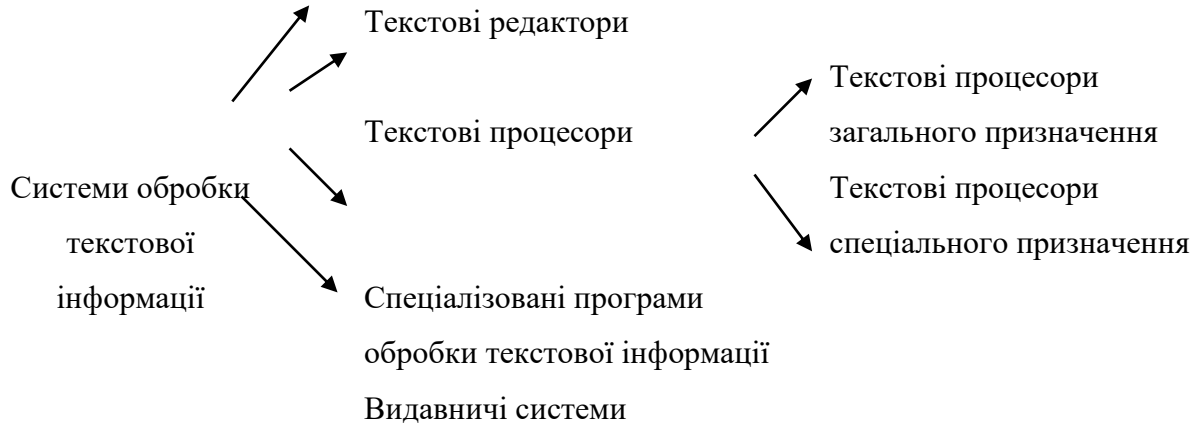
Усього існує кілька сотень редакторів текстів, від найпростіших до досить потужних і складних. Серед найпоширеніших у світі редакторів назвемо

Microsoft Word, WordPerfect, WordStar. Із цих редакторів у США найпоширеніші Microsoft Word для Windows й WordPerfect, у Європі й Росії – Microsoft Word.

5.3. Класифікація засобів обробки текстової інформації

Підготовкою різноманітних документів (текстів) займаються фахівці різних напрямків. Письменник, учений, інженер, економіст, бухгалтер, учитель і багато інших працівників становлять документи різного змісту й виду. Існують і багато інших переваг підготовки документів з використанням ПЭВМ: контекстний пошук і заміна рядків тексту; завдання довільних міжрядкових проміжків; автоматична нумерація сторінок; набір тексту в кілька стовпців; використання операції "відкочування" (відмови від декількох останніх операцій, зроблених з текстом); перевірка правопису й підбор синонімів; побудова змістів і багато чого іншого.

Ці можливості користувачеві ПЕОМ надають різні програми підготовки й редагування текстів. Програм таких дуже багато, і їхні можливості відмінні. У цьому зв'язку доцільно говорити про класифікації засобів обробки текстової інформації.



Текстовий редактор – це інструментальний програмний засіб, призначений для створення й редагування текстів, що не містять складних структур (параграфів, глав й ін.), і мала кількість, що має, функцій.

До текстових редакторів варто віднести редактори текстів програм (EdLin, EdiProf) і вбудовані редактори (NotePad, WordPad). Як правило, текстовий редактор – це реалізована у вигляді функцій у більш складній системі

можливість редагування текстів. Так, наприклад, в оболонці Norton Commander існує функція Edit, що допускає редагування тексту, обсягом не більше 26480 байтів. Відмінною рисою текстових редакторів є обмеженість їхніх можливостей. Як правило, вони не припускають роботу з такими регулярними структурами тексту, як глава, параграф, абзац. Помітимо, що текст програми на якій-небудь алгоритмічній мові сам по собі не містить таких структур. Використають текстові редактори найчастіше для набору тексту програм або екстреного внесення незначних змін у невеликий по обсязі текст. Простота текстових редакторів з погляду їхнього функціонального наповнення спричиняється й простоту роботи з ними.

Безліч додаткових функцій реалізовано в текстових процесорах.

Текстовий процесор – це інструментальний програмний засіб, призначений для створення й редагування текстів складної структури й функціональне наповнення, що має широке застосування. Найчастіше розрізняють текстові процесори загального й спеціального призначення. Відмінною рисою текстових процесорів є можливість обробляти такі регулярні структури документа, як абзац, параграф, сторінка й ін. У світі існує сотні текстових процесорів, різних за своїм функціональним наповненням. Чим більше функцій реалізує той або інший процесор, тим він більше складний для освоєння. Серед найпоширеніших – текстові процесори загального призначення: Лексикон, Фотон, Multe-Edit, Xy Write, Microsoft Word, Word Perfect. Проводячи порівняльну характеристику текстових процесорів, оцінюють, як правило, їхні функції:

- редагування,
- форматування,
- злиття файлів,
- настільне видавництво,
- печатка,
- швидкодія.

Розглядаючи функцію редагування, особлива увага приділялася таким можливостям, як: максимальна кількість вікон, наявність команди Undo (відкочування); пошук помилок; використання перехресних посилань; обробка структурованих текстів; режим, редактор і мова програмування макрокоманд.

Функція форматування оцінюється залежно від наявності наступних можливостей: контроль за висячими заголовками; керування заголовками, завдання таблиці стилю.

Під злиттям файлів мається на увазі можливість зчитування файлів, підготовлених у системах Lotus 1-2-3, dBASE: зчитування ASCII- файлів; можливість використання умовного оператора й математики; створення табличної форми.

Найважливішими можливостями настільного видавництва, які найчастіше реалізовані в текстових процесорах - це імпорт графіки, попередній перегляд сторінок, розміщення тексту навколо графіки, малювання ліній або прямокутників з текстом.

При оцінці функції печатки розглядають можливість організації пропорційної розрядки, постановки документів у чергу на печатку, фонової печатки й підтримки мови PostScript.

Серед текстових процесорів спеціального призначення слід зазначити такі, як Unv Editor (до версії 4,0 цей процесор називався ChiWriter) і Rt-chk, придатних для підготовки наукових текстів, що містять математичні, фізичні або хімічні формули, що допускають можливість використання до 20-ти різних шрифтів одночасно (для одного документа). Крім того, вони дозволяють готовити документи з використанням верхніх і нижніх індексів, готичними, грецькими, латинськими й росіянами буквами й спеціальними знаками. Підвищити ефективність і можливості текстового процесора дозволяють спеціалізовані програми коректування документів. ***Спеціалізовані програми*** обробки текстової інформації – це програмні засоби, що мають вузьку спеціалізацію. Серед таких програм слід зазначити програми ***перевірки правопису й підбора синонімів, формування текстів, кодувальники,*** програми

групового запису текстів, словникові програми. Багато текстових процесорів як функції містять деякі з таких програм. Однак, як правило, можливості убудованих програм обмежені в порівнянні зі спеціалізованими. Як приклад, розглянемо можливість орфографічного контролю тексту (іноді такі програми називають спеллерами - від англійського слова speller).

Програми ***формування текстів*** допомагають скласти (задати) загальний вид документа. Крім того багато хто з них можуть стиснути або розширити весь текст або його частина, а також, що робить документ більш сприятливим для читання; при необхідності здійснити динамічне копіювання заголовків (при стиску або розширенні одного з них відповідно змінюються й всі інші). Деякі програми формування текстів містять зразки форм тексту; глоссарій; спеллер; підтримують сортування заголовків і збір приміток під одним заголовком. Програми можуть завантажуватися в комп'ютер резидентно, автономно або разом з текстовим процесором, базою даних або електронних таблиць. Серед найпоширеніших: Grand Viwe (Symantek), Max-89, PC-Outline (Broun Bag Software).

Проблема сумісності текстових процесорів у деяких випадках може бути вирішена за допомогою так званих ***кодувальників тексту*** або програм перетворення файлів.

Величезну допомогу в підготовці документа, над яким одночасно працюють кілька авторів, можуть зробити програми ***групового запису текстів.*** Вони дозволяють вносити виправлення й коментарі в документ, не знищуючи оригіналу. Деякі із цієї групи програм дозволяють зрівняти два тексти й виділити в них, наприклад кольорами, відмінні частини. Інші використовують різні шрифти для змін, внесених у текст. Однієї з найбільш популярних програм групового запису текстів є програма ForComment, розроблена фірмою Braderbund. З документом можуть працювати до 15 співавторів. Внесення зміни позначаються ім'ям співавтора і датуються. Однак вносити виправлення в оригінал може тільки редактор. Програма дозволяє зберігати до 26 варіантів документа і до 15 зауважень або коментарів до кожного рядка.

Словникові програми орієнтовані на використання фахівцями різних галузей знань і містять визначення слів і фраз. Багато хто з них можуть: містити перехресні посилання; відшукувати синоніми; давати безліч визначень для слів, що мають більше одного значення; перевіряти правильність написання слів. Як приклад можна привести програму Stedmans Medical Dictionary, що є медичним словником на 68 тис. термінів. Програма Chace Wards дозволяє дати визначення 80 тис. слів (у тому числі й з «Короткого словника з електроніки» видавництва Webster") і 40 тис. синонімів. Всі ці програми дозволяють найбільше професійно редагувати різні документи.

Для підготовки рекламних буклетів, оформлення журналів і книг використаються спеціальні ***видавничі системи***. Вони дозволяють підготовляти й друкувати на лазерних принтерах або виводити на фотоскладальні автомати складні документи високої якості.

Є два основних види видавничих систем. Видавничі системи першого виду дуже зручні для підготовки невеликих матеріалів з ілюстраціями, графіками, діаграмами, різними шрифтами в тексті, наприклад газет, рекламних буклетів і невеликих журналів. Ці системи завжди зроблені за принципом WYSIWYG (WYSIWYG — скорочення англійської фрази «що Ви бачите, то й одержите», тобто що на екрані, те й на печатці). Типовий приклад такої системи – Aldus PageMaker.

Видавничі системи другого виду більше підходять для підготовки більших документів, наприклад книг. Вони мають ті ж можливості, що й системи першого напрямку, але для них характерна наявність розвиненого апарата параметрів розміщення тексту. Це дозволяє легко змінювати оформлення документа, зберігаючи єдність стилю, а також автоматизувати процес верстки. Однієї з найпоширеніших систем другого виду є система Ventura Publisher (нині Corel Ventura). Ventura побудована за принципом WYSIWYG, управляється за допомогою меню й може зчитувати тексти, підготовлені за допомогою інших текстових редакторів (наприклад, Microsoft Word), зберігаючи при цьому деякі параметри форматування, установлені цими

редакторами. Зчитавши деякий текст, можна потім установити параметри його розміщення, шрифти для різних частин тексту, вставити малюнки й т.д.

Основна операція, для якої використовуються видавничі системи, — це верстка, тобто розміщення тексту по сторінках документа, вставка малюнків, оформлення тексту різними шрифтами й т.д. А в режимі уведення й редагування тексту Ventura й Aldus PageMaker значно уступають таким редакторам текстів, як Microsoft Word. Вони працюють повільніше, менш зручні й не мають багатьох важливих можливостей редакторів текстів. Тому найчастіше документи підготовляють у два етапи: набирають текст у редакторі типу Microsoft Word, а потім зчитують його системою Aldus PageMaker або Ventura, і здійснюють остаточну підготовку документа.

Останнім часом деякі редактори текстів документів настільки наблизилися по можливостях до видавничих систем, що стали впритул з ними конкурувати. Наприклад, Microsoft Word для Windows також повністю побудований за принципом WYSIWYG і дозволяє виконувати практично всі функції видавничих систем:

- використати сотні різних видів шрифтів (накреслень і розмірів символів тексту), які відображаються на екрані так само, як при печатці;
- розміщати в документі, змінювати й коректувати малюнки й діаграми;
- «розтягувати» букви в тексті (робити розрядку) і «притискати» їх друг до друга;
- зручно підготовляти таблиці;
- вирівнювати нижній край тексту на сторінці на задану границю (щоб сторінки документа мали однаковий вигляд);
- набирати формули і т.д.

Помітимо, що не всі видавничі системи мають такі можливості (наприклад, Aldus PageMaker не вміє підготовляти формули). А в Microsoft Word для Windows до цього додаються ще й всі функції звичайного редактора документів (підтримуються практично все ;можливості Microsoft Word), досить зручний інтерфейс і засоби імпорту документів з Microsoft Word; WordPerfect

практично без втрат у форматуванні. Тому багатьом користувачам можливостей Microsoft Word для Windows виявляється цілком достатньо, а реалізовані ці можливості значно краще, ніж в Ventura або Aldus PageMaker. Зрозуміло, при верстці газет і великих журналів зручність і гнучкість Aldus PageMaker залишаються неперевершеними. А зате до верстки книг і технічної документації Aldus PageMaker пристосований гірше, наприклад, у ньому немає засобів для автоматичної вставки перехресних посилань. Тут більше підходить Ventura, особливо версія Corel Ventura 5.0.

У зв'язку з гострою конкуренцією виробники видавничих систем стали вбудовувати у видавничі системи можливості професійного кольороподілу, що забезпечують підготовку високоякісних кольорових видань, а також засобу графічних редакторів. Раніше такі засоби були тільки на комп'ютерах Macintosh, робочих станціях й ЕОМ більше високого класу.

5.4. Редактори наукових документів

Більшості користувачів не потрібно готовити документи зі скільки-небудь складним набором тексту. Мабуть, для них самий складний з необхідних видів тексту – це таблиця. І більшість редакторів текстів ніяких більше складних видів текстів і не підтримували (принаймні, до останнього часу). Але є й дуже важлива категорія користувачів, яким доводиться читати й писати документи.

Таким чином, для великої кількості фахівців – науковців, інженерів, конструкторів, економістів і т.д. необхідні саме документи з математичними й хімічними формулами, різними спеціальними символами, матрицями й складними діаграмами. Зрозуміло, що їм було вкрай незручно друкувати текст без формул, а потім уписувати формули від руки. Тому для підготовки таких документів були створені спеціальні редактори наукових документів.

Одним із прикладів таких редакторів є ChiWriter. Він потрапив у нашу країну ще в середині 80-х років, коли комп'ютери в основному використалися науковцями, і одержав дуже широке поширення, непорівнянне з його

популярністю на батьківщині, у США. Поширенню ChiWriter досить сприяли легкість його «русифікації» (включення російських шрифтів), важлива в науковому середовищі можливість підготовки документів з математичними й хімічними формулами, а також інтенсивна популяризація (у тому числі й автором цієї книги). Однак потім популярність ChiWriter пішла на спад, тому що його можливості досить обмежені й він не дозволяє домогтися гарного («типографського») якості видрукуваних документів, яку можна одержати на лазерному й високоякісному матричному принтерах. І звичайно, падіння популярності ChiWriter було пов'язане з появою більше привабливих альтернатив.

Крім ChiWriter, на Заході були розроблені й іншої системи для підготовки текстів з формулами. Одні з них (наприклад, T" й MathOr) є самостійними редакторами, інші (MathWord) — засобами для вставки формул у документи, підготовлені за допомогою редакторів текстів загального призначення. Однак вони не одержали широкого поширення, тому що до ' кінця 80-х років можливості підготовки формул стали включатися в редактори текстів загального призначення. Першою «ластівкою» став WordPerfect 5.1, потім з'явилися Microsoft Word для Windows й інші програми. Є такі засоби й у видавничій системі Corel Ventura.

Однак редактори текстів загального призначення не повністю забезпечують підготовку формул. Зокрема, вони не мають багатьох спеціальних символів, погано підтримують обробку «багатоповерхових» :діаграм і т.д. Крім того, уведення формул там виконується дуже повільно — сторінка типового математичного тексту вводиться за 20-30 хвилин, якщо не більше. Наприклад, в Word для Windows для будь-якої формули треба викликати редактор формул, вибирати з піктограм потрібні символи, установлювати курсор миші в потрібне «віконце», щоб Word знав, до якої частини формули ставиться що вводить або видаляє текст, що, і т.д. Весь цей процес дуже схожий на ручне вишивання бісером – і за технологією, і по продуктивності.

Тому використання редакторів текстів загального призначення для

підготовки текстів з формулами має сенс, тільки якщо документ містить відносно небагато формул, так що набір формул становить лише невелику частину загального обсягу роботи. А для документів з інтенсивним використанням формул набагато ефективніше буде спеціалізована система TEX, розроблена математиком Д.Батогом.

Система TEX не так зручна в роботі, як редактори документів і видавничі системи, тому що не побудована за принципом WYSIWYG. В TEX документ набирається як звичайний текстовий файл із убудованими командами форматування. Команди форматування TEX утворюють досить яскраву мову, за допомогою якої можна описати будь-які стилі документів, види форматування, формули й таблиці.

Зрозуміло, що такий набір менш наочний, чим в Word для Windows, і вимагає деякого попереднього навчання. Зате при мінімальній навичці він виконується в п'ять-десять разів швидше, ніж кропітке діалогове визначення формули у звичайних редакторах тексту. Уведення звичайного тексту в TEX виконується з тією же швидкістю, що й у редакторах текстів. Порівняно швидко можна набирати текст і на «екзотичних» мовах — єврейському, арабському, монгольському, китайському й т.д. Звичайно, деякі речі, зокрема включення малюнків, в TEX робити не так зручно, як у видавничій системі. Але в підсумку для документів з більшою кількістю формул загальний час їхньої підготовки до видання скорочується в кілька разів. Тому багато відомих західних видавництв використають для набору книг і журналів по математиці, фізиці, хімії, техніці й т.д. систему TEX, а Американське математичне суспільство приймає для публікації тільки рукопису, підготовлені в TEX.

Підготовлений у системі TEX документ можна переглянути на екрані за допомогою вхідних в TEX програм попереднього перегляду. Документ виглядає на екрані так, як він був би надрукований, однак ніяких виправлень у документі при цьому робити не можна. При виявленні яких-небудь погрішностей у поданні документа необхідно вийти із програми перегляду, знайти в тексті документа помилку й виправити неї. Це, зрозуміло, не так

зручно, як у діалогових редакторах тексту.

Висновки

Підготовкою різноманітних документів (текстів) займаються фахівці різних напрямків. Письменник, учений, інженер, економіст, бухгалтер, учитель і багато інших працівників становлять документи різного змісту й виду. Існують і багато інших переваг підготовки документів з використанням ПЭВМ: контекстний пошук і заміна рядків тексту; завдання довільних міжрядкових проміжків; автоматична нумерація сторінок; набір тексту в кілька стовпців; використання операції "відкочування" (відмови від декількох останніх операцій, зроблених з текстом); перевірка правопису й підбор синонімів; побудова змістів і багато чого іншого.

Текстовий редактор – це інструментальний програмний засіб, призначений для створення й редагування текстів, що не містять складних структур (параграфів, глав й ін.), і мала кількість, що має, функцій.

Текстовий процесор – це інструментальний програмний засіб, призначений для створення й редагування текстів складної структури й функціональне наповнення, що має широке застосування.

Література: [10-12]

Контрольні питання

1. Що таке текст, символ, абзац?
2. Текстовий редактор.
3. Текстовий процесор.
4. Які бувають редактори наукових документів?
5. Призначення і особливості роботи системи TeX.

Змістовий модуль 3.

ПРОЕКТУВАННЯ І РОЗРОБКА ІНФОРМАЦІЙНОЇ БАЗИ

Лекція 6. Уявлення про бази даних

Мета: Отримати основні поняття з проектувань баз даних, їх типів та функціональних особливостей.

План

6.1. Основні поняття (*структурування даних, база даних, предметна область, інформаційна модель, поле, запис, структура запису*)

6.2. Роль БД в інформаційній системі

6.3. Моделі даних (*Ієрархічна, Мережева, Реляційна*)

6.4. Модель «Сутність – зв'язок»

6.5. Поняття о системах управління базами даних (*Способи створення БД, визначення СУБД, види СУБД, інструменти та функції СУБД*).

6.1. Основні поняття

База даних (БД) — це структурована сукупність взаємопов'язаних даних певної предметної області. Щоб оперувати даними, що складають базу, необхідна окрема програма — система управління базами даних (СУБД).

БД – великий упорядкований комплекс інформації, який зберігається на комп'ютерних носіях зазвичай разом із програмою, що дозволяє здійснювати швидкий пошук даних, їх оновлення та друк.

Практично всі сучасні СУБД використовують реляційну модель даних. В основі цієї моделі, якові запропонував Е.Ф. Кодд в 1970 р., лежить поняття відношення (англійською relation). Відношення оформлені у вигляді двовимірних (тобто звичайних) таблиць які складаються з однотипних рядків (записів), і стовпців, що іменуються полями.

Структура таблиці визначається сукупністю полів, типом і розміром даних поля, а також унікальним ключем таблиці. Значення унікального ключа не можуть повторюватись у записах таблиці. Рядки таблиці однозначно ідентифікуються значенням ключа.

Для шкірного поля, крім імені, визначається ще і тип даних, що вказує, якого вигляду дані допускається вводити до полів. Вибір відповідного типу даних забезпечує введення даних у правильній формі для впорядкування, обчислень і виконання інших операцій.

До шкірного поля допускається введення даних тільки одного типу. Дозволяється використовувати такі основні типи даних:

Тип	Значення
Текстовий	Текст або числа, що не потребує проведення розрахунків
Числовий	Числові дані, що використовуються для проведення розрахунків
Дата/година	Дати і година, що належати до років з 100 по 9999, включно
Логічний	Логічні значення, а також поля, які можуть містити одне з двох можливих значень (True/False, Так/Ні)
Грошовий	Грошові значення і числові дані, які використовуються в математичних розрахунках, що проводяться з точністю до 15 знаків у цілій і до 4 знаків у дробовій частині
МЕМО	Довгий текст або комбінація тексту та чисел
Лічильник	Унікальні послідовно зростаючі (на 1) або випадкові числа, що автоматично вводяться при додаванні кожного нового запису в таблицю. Значення полів типу лічильника оновлювати (виправляти) не можна

Структурування даних

У програмуванні та комп'ютерних науках структури даних – це способи організації даних у комп'ютерах. Часто разом зі структурою даних пов'язується і специфічний перелік операцій, що можуть бути виконаними над даними, організованими в таку структуру.

Правильний підбір структур даних є надзвичайно важливим для ефективного функціонування відповідних алгоритмів їх обробки. Добрі побудовані структури даних дозволяють оптимізувати

використання машинного часу та пам'яті комп'ютера для виконання найкритичніших операцій.

Відома формула «Програма = Алгоритми + Структури даних» дуже точно виражає необхідність відповідального ставлення до такого підбору. Тому іноді навіть не обраний алгоритм для обробки масиву даних визначає вибір тої чи іншої структури даних для їх збереження, а навпаки.

Підтримка базових структури даних, які використовуються в програмуванні, включена в комплекти стандартних бібліотек найбільш розповсюджених мов програмування.

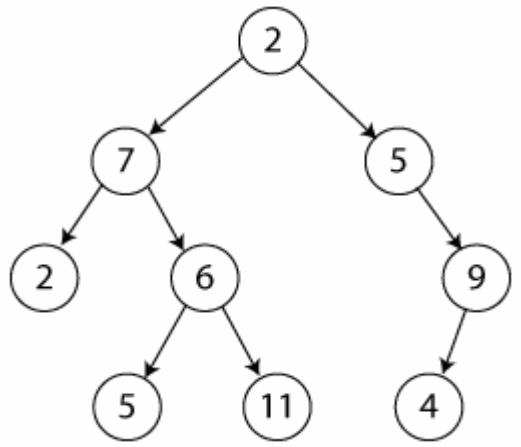


Рис. 6.1. *Бінарне дерево, одна з найпростіших деревовидних структур даних*

Поняття предметної області було введено на початку 80-х років минулого століття, коли вченими в області ИС була усвідомлена необхідність використовувати семантичні моделі для вистави інформації в комп'ютерних системах. Так само як вимоги до комп'ютерної системи формуються коштами природної мови, так і інформація в комп'ютерних системах представляється коштами особливої мови з певною семантикою. Такий підхід уперше був представлений П. Ченом в 1976 році.

Визначення. Предметна область - це цілеспрямована первинна трансформація картини зовнішнього миру в деяку умоглядну картину, певна частина якої фіксується в ІС у якості алгоритмічної моделі фрагмента дійсності.

Інформаційна модель предметної області бази даних

Одним із ключових моментів створення ІС із метою автоматизації інформаційних процесів організації є всебічне вивчення об'єктів автоматизації, їх властивостей, взаємин між цими об'єктами й вистава отриманої інформації у вигляді інформаційної моделі даних.

Інформаційна модель даних призначена для вистави семантики предметної області в термінах суб'єктивних коштів опису - сутностей, атрибутів,ідентифікаторів сутностей, супертипів, підтипів і т.д.

Інформаційна модель предметної області бази даних містить наступні основні конструкції:

- діаграми " сутність-зв'язок" (Entity - Relationship Diagrams);
- визначення сутностей;
- унікальні ідентифікатори сутностей ;
- визначення атрибутів сутностей ;
- відносини між сутностями;
- супертипи й підтипи.

Увага! Елементи інформаційної моделі даних предметної області є вхідними даними для розв'язку завдання проектування бази даних - створення логічної моделі даних.

Сутності, атрибути й ідентифікатори (ключі) сутності, домени атрибутів.

Предметом інформаційної моделі є абстрагування об'єктів або явищ реального миру в рамках предметної області, у результаті якого виявляються сутності (entity) предметної області. Як правило, вони позначаються іменем іменником природної мови.

В Access база даних — це файл з розширенням **mdb**, який містить дані у вигляді однієї чи декількох таблиць. Окрім таблиць, у файлі БД можуть бути такі об'єкти: форми, запити, макроси, модулі, які розглядатимемо далі.

Розглянемо структуру таблиці. Таблиця складається з рядків і стовпців. Рядки називаються записами.

Запис містить інформацію про один елемент бази даних: одну людину, книжку, продукцію, рейс тощо. Він складається з полів.

Поле – це мінімальна (але найважливіша) порція інформації в записі, над якою визначені операції введення, виведення, перетворення тощо. Поле має ім'я, значення, характеризується типом і низкою властивостей. Нижче наведене приклади типів, назв і значень полів, які можуть зустрічатися в різних завданнях.

Назви полям дає користувач, назви типів є стандартні, а значення полів випливають зі змісту конкретної задачі. Розглянемо загальні властивості числового поля: розмір поля (*байт, ціле, довге ціле, 4 байти, 8 байтів*), формат (*основний, грошовий, процентний, експоненціальний* тощо), кількість десяткових знаків після коми, підпис поля, значення за замовчуванням, умова на значення, повідомлення про помилку введення, обов'язкове поле, індексоване поле.

Отже, структура таблиці – це структура запису, тобто сукупність назв полів, їхніх типів та властивостей. Структуру визначає користувач під година аналізу конкретної задачі.

6.2. Роль БД в інформаційній системі

Наприкінці 60-х – на початку 70-х років відбувся перехід від обробки даних до обробки інформації. Це було визнанням того, що інформація не є вибачимо записом даних. У цьому контексті потрібно розрізняти дані та інформацію. Під даними зазвичай розуміють розрізнені факти. Інформація – це організовані та оброблені дані або висновки з їх.

Отже, інформацію отримують у результаті обробки великої кількості даних. Інформація – це ресурс, що має велику цінність, а тому важливо навчитися впорядковувати його та управляти їм. Поступове визнання цього факту призвело до розуміння цінності інформації та потенціалу комп'ютерних систем у процесі підтримання такого значного ресурсу, як інформація. Так, наприкінці 60-х з'явилися інформаційно-управляючі системи. Такі системи

використовували вже наявні в комп'ютері дані та давали відповіді на низку запитань.

Інформаційна система (ІС) – це програмний комплекс, функціями якого є: 1) підтримка надійного зберігання інформації в пам'яті комп'ютера; 2) виконання специфічних для конкретного додатка перетворень інформації й(або) обчислень; 3) надання користувачам зручного інтерфейсу.

Типовим прикладами ІС є банківські системи, системи резервування авіаційних або залізничних квитків, місць у готелях тощо.

Інформаційна база даних повинна задовольняти наступним вимогам:

- структура бази даних винна дозволяти легко розчленити її на складові частини, які розміщуються в окремих вузлах мережі, забезпечувати простоту доступу до будь-якої підбази, захист від несанкціонованого доступу до тихнув чи інших даних і високу продуктивність у роботі з даними;
- структура інформаційної бази винна забезпечувати адекватність змісту зовнішньої (документальної) і внутрішньої (комп'ютерної) форми зберігання інформації про об'єкти чи процеси, з якими працює виконавець;
- структура інформаційної бази й схема її розподілення по вузлах локальної обчислювальної мережі винна забезпечувати можливість єдиного або одночасного процесу корегування декількох однакових баз даних, що зберігаються в різних вузлах, або ж заміну скорегованої бази даних одного вузла на всіх інших, зв'язаних з їм єдиною інформаційною мережею;
- структура інформаційної бази винна бути мінімально надлишковою й одночасно зручною для архівування даних.

6.3. Моделі даних (Ієрархічна, Мережева, Реляційна)

Основою бази даних є модель даних – фіксована система зрозуміти і правил для представлення даних структури, стану і динаміки проблемної області в базі даних. У різні часи послідовне застосування

одержували ієрархічна, мережна і реляційна моделі даних. У сучасні часи все більшого поширення набуває об'єктно-орієнтований підхід до організації баз даних ГІС.

6.3.1. Ієрархічна модель даних

Часто об'єкти перебувають у відношеннях, що називають ієрархічними: відношення «частина – ціле» (наприклад, адміністративна область складається з районів, сільських і міських радий, населених пунктів та ін.); видове відношення (наприклад, будинки бувають житлові, виробничі та ін.); відношення підпорядкованості (наприклад, губернатор – заходів міста).

Об'єкти, що перебувають в ієрархічних відношеннях, утворюють дерево «орієнтований граф», у якого є тільки одна вершина, не підлегла жодній іншій вершині (цю вершину називають коренем дерева); будь-яка інша вершина графа підлегла лише одній іншій вершині (рис. 6.1). Концептуальна схема ієрархічної моделі являє собою сукупність типів записів, пов'язаних типами зв'язків в одне чи кілька дерев. Усі типи зв'язків цієї моделі належати до виду «один до декількох» і зображуються у вигляді стрілок.

Таким чином, взаємозв'язки між об'єктами нагадують взаємозв'язки в генеалогічному дереві, за єдиним винятком: для шкірного породженого (підлеглого) типу об'єкта може бути тільки один вхідний (головний) тип об'єкта. Тобто ієрархічна модель даних допускає тільки два типи зв'язків між об'єктами: «один до одного» і «один до декількох». Ієрархічні бази даних є навігаційними, тобто доступ можливий тільки за допомогою заздалегідь визначених зв'язків.

При моделюванні подій, як правило, необхідні зв'язки типу «багато до декількох». Як одне з можливих рішень зняття цього обмеження можна запропонувати дублювання об'єктів. Однак дублювання об'єктів створює можливості неузгодженості даних.

Достоїнство ієрархічної бази даних полягає в тому, що її навігаційна природа забезпечує швидкий доступ при проходженні вздовж заздалегідь визначених зв'язків. Однак негнучкість моделі даних і, зокрема, неможливість

наявності в об'єкта декількох батьків, а також відсутність прямого доступу до даних роблять її непридатною в умовах частого виконання запитів, не запланованих заздалегідь. Ще одним недоліком ієрархічної моделі даних є ті, що інформаційний пошук з нижніх рівнів ієрархії не можна спрямувати по вище розміщених вузлах.

6.3.2. Мережна модель даних

У мережній моделі даних поняття головних і підлеглих об'єктів дещо розширені. Будь який об'єкт може бути і головним, і підлеглим (у мережній моделі головний об'єкт позначається терміном «власник набору», а підлеглий – терміном «член набору»). Тієї самий об'єкт може одночасно виконувати і роль власника, і роль члена набору. Це означає, що кожний об'єкт може брати доля в будь-якій кількості взаємозв'язків.

Подібно до ієрархічної, мережну модель також можна податі у вигляді орієнтованого графа. Алі в цьому випадку граф може містити циклі, тобто вершина може мати кілька батьківських вершин.

Така структура набагато гнучкіша й виразніша від попередньої й придатна для моделювання більш ширшого класу завдань. У цій моделі вершини є сутностями, а ребра, що їх з'єднують, – відношеннями між ними.

Ієрархічні й мережні бази даних часто називають базами даних з навігацією. Ця назва відбиває технологію доступу до даних, використовувану при написанні програм обробки мовою маніпулювання даними. При цьому доступ до даних по шляхах, не передбачених при створенні бази даних, може потребувати нерозумно тривалого години.

Підвищуючи ефективність доступу до даних і скорочуючи таким чином година відповіді на запитий, принцип навігації разом з цим підвищує і ступінь залежності програм і даних. Програми обробки даних виявляються жорстко прив'язаними до потокового стану структури бази даних і повинні бути переписані при її змінах. Операції модифікації і видалення даних вимагають переустановлення покажчиків, а маніпулювання даними залишається записоорієнтованим. Крім того, принцип навігації не дозволяє істотно

підвищувати рівень мови маніпулювання даними, щоб зробити його доступним користувачу-непрограмісту чи навіть програмісту-непрофесіоналу. Для пошуку запису-мети в ієрархічній або мережній структурі програміст винний спочатку визначити шлях доступу, а потім – крок за кроком переглянути всі записи, що трапляються на цьому шляху.

Наскільки складними є схеми представлення ієрархічних і мережних баз даних, настільки й трудомістким є проектування конкретних прикладних систем на їхній основі. Як показує досвід, тривалі терміни розроблення прикладних систем нерідко призводять до того, що смороду постійно перебувають на стадії розроблення й доопрацювання. Складність практичної реалізації баз даних на основі ієрархічної й мережної моделей визначила створення реляційної моделі даних.

6.3.3. Реляційна модель даних

У реляційній моделі даних об'єкти й взаємозв'язки між ними представляються за допомогою таблиць. Взаємозв'язки також подаються як об'єкти. Кожна таблиця представляє один об'єкт і складається з рядків і стовпців. Таблиця винна мати первинний ключ (ключовий елемент) – поле чи комбінацію полів, що єдиним способом ідентифікують кожний рядок у таблиці.

Назва «реляційна» (relational) пов'язана з тім, що кожен запис у таблиці даних містить інформацію, яка стосується (related) якогось конкретного об'єкта. Крім того, зв'язані між собою (тобто такі, що знаходяться в певних відношеннях – relations) дані навіть різних типів у моделі можуть розглядатися як одне ціле.

Таблиця має такі властивості:

- кожний елемент таблиці являє собою один елемент даних;

- повторювані групи відсутні;

- усі стовпці в таблиці однорідні; це означає, що елементи стовпця мають однакову природу;

- стовпцям присвоєні унікальні імена;

- у таблиці немає двох однакових рядків.

Порядок розміщення рядків і стовпців у таблиці довільний; таблиця такого типу називається відношенням. У сучасній практиці для рядка використовується термін «запис», а для стовпця термін «поле».

Основною відмінністю пошуку даних в ієрархічних, мережних і реляційних базах даних є ті, що ієрархічні і мережні моделі даних здійснюють зв'язок і пошук між різними об'єктами за структурою, а реляційні – за значенням ключових атрибутів (наприклад, можна знайти всі записи, значення яких у полі «номер будинку» дорівнює 3, але не можна знайти 3-й рядок).

Оскільки реляційна структура концептуально проста, вона дозволяє реалізовувати невеликі і прості (і тому легкі для створення) бази даних, навіть персональні, сама можливість реалізації яких ніколи навіть і не розглядалася в системах з ієрархічною чи мережною моделлю.

Недоліком реляційної моделі даних є надмірність по полях (для створення зв'язків між різними об'єктами бази даних).

Практично всі існуючі на сьогоднішній день комерційні бази даних і програмні продукти для їх створення використовують реляційну модель даних.

6.4. Модель «Сутність – зв'язок»

При вивченні моделі даних слід виявити рівні логічної вистави даних, до яких має відношення ця модель. Розширюючи набір положень ми можемо визначити чотири рівні вистави даних:

- інформація, що ставиться до сутностей і зв'язкам, які існують у нашій уяві;
- структура інформації – організація інформації, у якій сутності й зв'язку представляються даними.
- структура даних, незалежна від шляхів доступу, – структури даних, які не зв'язані зі схемами пошуку, індексації і т.д.
- структура даних, залежна від шляхів доступу.

У наступних підрозділах ми будемо крок за кроком розробляти модель сутність-зв'язок для перших двох рівнів. Як ми побачимо пізніше, мережна модель у тому виді, у якому вона існує в цей час, в основному ставиться до

рівня 4, реляційна модель в основному ставиться до рівнів 3 і 2, а модель безлічі сутностей в основному ставиться до рівнів 1 і 2.

На цьому рівні ми розглядаємо сутності й зв'язку. Сутність – це предмет, який може бути ідентифікований деяким способом, що відрізняють його від інших предметів. Прикладами сутності є конкретна людина, компанія або подія. Зв'язок – це асоціація, установлювана між сутностями.

База даних підприємства містить інформацію про сутності й зв'язки, які становлять інтерес для цього підприємства. У базу даних підприємства не може бути занесений повний опис сутності або зв'язки. Неможливо (і, як видно, не обов'язково) зберігати всю потенційно доступну інформацію про сутності й зв'язки. Далі ми будемо розглядати тільки ті сутності й зв'язку (і інформацію про них), які повинні ввійти в проект бази даних.

Нехай e позначає деяку сутність, яка існує в нашій уяві. Кожна сутність ставиться до деякого відмінного від інших безлічі сутностей, такому як EMPLOYEE, PROJECT або DEPARTMENT. З кожною безліччю сутностей зв'язується предикат, що дозволяє перевірити, чи належить дана сутність даній безлічі. Наприклад, якщо ми знаємо, що сутність ставиться до безлічі сутностей EMPLOYEE, те ми також знаємо, що ця сутність має властивості, загальні з іншими сутностями з безлічі сутностей EMPLOYEE. У число цих властивостей входить згаданий вище предикат. Нехай Ei позначає безліч сутностей. Помітимо, що безлічі сутностей не зобов'язано бути непересічними. Наприклад, сутність, що належить безлічі сутностей MALE-PERSON (ЧОЛОВІКА), належить також і безлічі сутностей PERSON (ЛЮДИ). У цьому випадку MALE-PERSON є підмножиною PERSON.

Інформацію про сутність або зв'язок одержують шляхом спостереження або виміру й виражають безліччю пар атрибут-значення. 3, red, Peter і Johnson – це значення. Значення класифікуються в різні безлічі значень (value sets), такі як FEET, COLOR, FIRST-NAME і LAST-NAME. З кожною безліччю значень зв'язується предикат для перевірки того, чи належить значення цій безлічі.

Значення з деякої безлічі значень може бути еквівалентно іншому значенню з іншої безлічі значень.

Атрибут (attribute) може бути формально визначений як функція, що відображає безліч сутностей або безліч зв'язків у безліч значень або декартово добуток безлічей значень: f: Ei or Ri → Vi or Vi1 × Vi2 × ... × Vin

6.5. Поняття про системи управління базами

Організація єдиної бази даних стала можливою лише завдяки тому, що були створені спеціальні програмні продукти – системи управління базами даних (СУБД).

Основне призначення СУБД – створення та підтримка в актуальному стані бази даних, а також зв'язок її з програмами розв'язування економічних завдань (прикладні програми користувачів).

База даних – це комп'ютерний термін, який використовується для позначення сукупності інформації з окремої тими або відомостей, пов'язаних з деякою прикладною задачею. Зберігання інформації у вигляді бази даних полегшує доступ до неї, пошук та вилучення потрібних фрагментів.

На магнітному диску база даних може зберігатись у вигляді одного файлу (бази даних MS Access, Informix та ін.) або у вигляді папки з файлами (бази даних Interbase, Paradox та ін.).

6.5.1. Різновиди систем управління базами даних, їх огляд

Системи управління даними першого покоління

СУБД першого покоління характерні тім, що кожна група користувачів розробляла своє власне програмне забезпечення по управлінню даними.

Наслідками такої сепаратизації стало надмірне дублювання програмних кодів і даних.

Системи управління даними іншого покоління.

Файли взаємопов'язаних даних об'єднуються в бази даних. СУБД створюються для таких досвідчених користувачів, як програмісти.

Системи управління даними третього покоління.

Можливості СУБД розширились. Створені розвинуті інтерфейси, що забезпечують інтерактивний доступ звичайним користувачам.

Переваги СУБД :

Скорочення надлишку даних;

Без баз даних неможливо уникнути зберігання надлишкових даних;

При наявності центрального контролю баз даних деякі надлишкові дані можна усунути.

У світі існує безліч СУБД. Незважаючи на ті, що можуть по-різному працювати з різними об'єктами й надають користувачу різні функції й засоби, більшість СУБД спираються на єдиний устояний комплекс основних зрозуміти. Це дає нам можливість розглянути одну систему й узагальнити її поняття, прийоми й методи на весь клас СУБД.

Система управління базами даних MICROSOFT ACCESS.

Система управління базами даних Microsoft Access входити до складу пакета Microsoft Office. Вона дозволяє розв'язувати широке коло завдань користувачів без програмування й доступна для широкого кола непрофесійних користувачів персональних комп'ютерів.

Система управління базами даних (СУБД) Access розроблена для експлуатації в комп'ютерних мережах у середовищі Windows.

Одна з основних переваг СУБД Access полягає в тому, що вона має прості та зручні засоби обробки кількох таблиць в одній базі даних. Таблиця є основним об'єктом бази даних. В одній базі даних зберігається кілька таблиць та засоби зв'язування таблиць.

У системі Access є різні способи управління даними, а саме:

система меню; панелі інструментів; контекстне меню; укажчик миші; комбінації клавіш.

СУБД Access має значну кількість спеціальних програм – “майстрів”. Є майстер таблиць, майстер кнопок, майстер форм та ін. Майстри здійснюють діалог з користувачем, у процесі якого визначаються дані, необхідні для розв'язування відповідної задачі. Для зручності роботи кожен майстер має певні

етапи (кроки). Будь-Який етап можна пропустити або звернутись до попередніх.

Формою видачі даних на екран користувач може управляти. Важливо правильно конструювати форми, оскільки саме з ними працює користувач при введенні й редагуванні записів бази даних. Крім того, форми можна використовувати для збирання та виведення інформації.

Етапи створення бази даних у середовищі Microsoft Access:

- визначення мети створення бази даних;
- визначення таблиць, які винна містити база даних;
- визначення структури таблиць (полів та їх типів);
- призначення ключів таблиць та створення потрібних індексів;
- визначення зв'язків між таблицями;
- завантаження даних;
- створення інших об'єктів бази даних: запитів, форм, звітів, макросів та модулів;
- аналіз ефективності бази даних за допомогою майстра таблиць та аналізатора швидкодії.

Висновок

Отже, основне призначення системи управління базами даних (скорочено – СУБД) — створення та підтримка в актуальному стані бази даних, а також зв'язок її з програмами розв'язування економічних завдань (прикладні програми користувачів).

Організація єдиної бази даних стала можливою лише завдяки тому, що були створені спеціальні програмні продукти — системи управління базами даних (СУБД).

У світі існує безліч СУБД. Незважаючи на ті, що смороду можуть по-різному працювати з різними об'єктами й надають користувачу різні функції й засоби, більшість СУБД спираються на єдиний устояний комплекс основних зрозуміти. Це дає нам можливість розглянути одну систему й узагальнити її

поняття, прийоми й методи на весь клас СУБД. Найпоширенішими СУБД є Visual Foxpro та Microsoft Access.

Система управління базами даних (СУБД) поєднує відомості з різних джерел в одній реляційній базі даних. Створювані форми, запити й звіти дозволяють швидко й ефективно обновляти дані, отримувати відповіді на питання, здійснювати пошук потрібних даних, аналізувати дані, друкувати звіти, діаграми й поштові наклейки.

Література [6,16]

Контрольні питання

1. Різновиди систем управління БД.
2. Що таке модель даних?
3. В чому відмінності моделі «Сутність-звязок» від «Ієрархічної»?
4. Дайте визначення терміну СУБД.

Питання для самостійного вивчення

1. Проектування баз даних.
2. Інтеграція бази даних з комп’ютерною мережею. Архітектура «Файл-сервер», архітектура «Клієнт сервер»
3. Характеристики і можливості СУБД MS Access
4. Основні принципи роботи з програмою MS Access
5. Створення нового проекту в СУБД MS Access

Лекція 7. Реляційні бази даних

Мета: Визначити основні елементи побудови реляційної бази даних, навчитися використовувати принципи створення реляційних баз даних.

План

7.1. Реляційна модель даних. Переваги і недоліки.

7.2. Поняття ключового елементу. Первинний ключ таблиці.

7.3. Зв’язки між таблицями та цілісність даних.

7.4. Графічне позначення реляційної моделі (*схема даних*)

7.1. Реляційна модель даних. Переваги і недоліки.

Перші системи керування базами даних з'явилися в середині шістдесятих років XX століття і підтримували ієрархічну модель даних. Далі були розроблені сітьові бази даних, в основу яких була закладена значно більш складна сітьова модель. У кожної із цих моделей були свої переваги і недоліки, які зіграли ключову роль у розвитку реляційної моделі.

Реляційна модель вперше була запропонована Е. Ф. Коддом (E. F. Codd) в 1970 році в його основній статті «Реляційна модель даних для більших спільно використовуваних банків даних».

Цілі створення реляційної моделі формулювалися в так:

- Забезпечення більш високого ступеня незалежності від даних. Прикладні програми не повинні залежати від змін внутрішнього представлення даних.
- Створення міцного фундаменту для рішення семантичних питань, а також проблем несуперечності та надмірності даних. Зокрема, у статті Кодда вводиться поняття нормалізованих відносин, тобто відносин без повторюваних груп.
- Розширення мов керування даними за рахунок включення операцій над множинами.

На основі цієї моделі в сімдесяті роки були розроблені перші реляційні бази даних, а в цей час вони розглядаються як стандарт для сучасних комерційних СУБД.

Реляційна модель даних (РМД) - логічна модель даних, прикладна теорія побудови баз даних, яка є додатком до задач обробки даних таких розділів математики як теорії множин і логіка першого порядку.

Кодд у 1985 році сформулював 12 правил, яким повинна задовольняти будь-як база даних, що претендує на тип реляційної.

Реляційна модель даних включає такі 3 компоненти, що описують різні аспекти реляційного підходу:

- структурний аспект - дані в базі даних являють собою набір відносин;
- аспект цілісності - відносини відповідають певним умовам цілісності. У цілісній частині реляційної моделі даних фіксуються дві базових вимоги цілісності, що повинні підтримуватися в будь-якій реляційній СУБД: вимога цілісності сутностей і вимога цілісності по посиланнях;
- аспект обробки (маніпулювання) - РМД підтримує оператори маніпулювання відносинами (реляційна алгебра, реляційне обчислення).

Крім того, до складу реляційної моделі даних включають теорію нормалізації.

Для кращого розуміння РМД слід зазначити три важливі обставини:

- модель є логічною, тобто відносини є логічними (абстрактними), а не фізичними (збереженими) структурами;
- для реляційних баз даних вірний інформаційний принцип: усе інформаційне наповнення бази даних представлено одним і тільки одним способом, а саме - явним завданням значень атрибутів у кортежах відносин;
- наявність реляційної алгебри дозволяє реалізувати декларативне програмування і декларативний опис обмежень цілісності, на додаток до навігаційного (процедурного) програмування та процедурній перевірці умов.

Реляційна модель даних має як потенційні переваги, так і недоліки.

Основні переваги такі.

Спрощення схеми даних для користувача.

Як відомо, деревоподібна та сітьова моделі поєднують в одній схемі поняття логічного та фізичного рівнів, тобто побудова схеми користувачем вимагає гарного знання технічних прийомів, реалізованих у системі, якщо необхідно одержати ефективну у використанні БД. Перевагою реляційної моделі перед іншими моделями є проста і зручна для користувача схема даних, що представляється у вигляді таблиць.

Поліпшення логічної ті фізичної незалежності.

Логічна незалежність допускає можливість застосування однієї концептуальної моделі різними користувачами. Фізична незалежність реляційної моделі полягає в тому, що модель даних не включає ніяких фізичних описів. У дійсності фізичне представлення відносин і шляхів доступу описується незалежно від опису логічної схеми відносин.

Забезпечення користувача мовами високого рівня.

Маніпулювання даними в ієрархічній і сітьовій моделях проводиться за допомогою процедурних мов. Для реляційних моделей недоцільно використовувати процедурну мову, оскільки забезпечена фізична незалежність даних. За допомогою команд процедурної мови програміст будує стратегію доступу до даних, але будь-яка зміна шляху доступу приводить до необхідності модифікації програми.

Оптимізація доступу до БД.

Збільшення фізичної незалежності та використання непроцедурних мов вимагають від системи вибору найкращої стратегії доступу. Оскільки в програмі не визначається стратегія доступу, то система вибирає найбільш ефективну з можливих.

Поліпшення цілісності і захисту даних.

Сучасні СУБД, орієнтовані на ієрархічні та сітьові моделі, мають обмежені засоби для підтримки цілісності і захисту даних. Вимоги цілісності

визначаються логічними термінами на рівні концептуальної схеми. Реляційна модель дозволяє поліпшити вираження вимог цілісності шляхом використання мови високого рівня. Ефективність опису досягається застосуванням непроцедурних мов, оскільки вони здатні ідентифікувати інформацію незалежно від будь-якого шляху доступу.

Можливості різних застосувань.

Використання простої реляційної схеми і мови запитів дозволяє розширити області застосування.

Забезпечення методологічного підходу.

Головною метою моделі БД є можливість опису реального світу. У реляційній моделі визначення першої, другої, третьої нормальних форм ґрунтується на математичній теорії відносин, дозволяє користувачеві структурувати інформацію, точно ідентифікуючи зв'язки, що існують між елементами інформації і обмеженнями, яким ці елементи повинні задовольняти.

Недоліки реляційного підходу, пов'язаного із застосуванням баз даних, такі:

- складність;
- розмір;
- вартість СУБД;
- додаткові витрати на апаратне забезпечення;
- витрати на перетворення;
- продуктивність;
- більш серйозні наслідки при виході системи з ладу;
- складність опису ієрархічних та сітьових зв'язків.

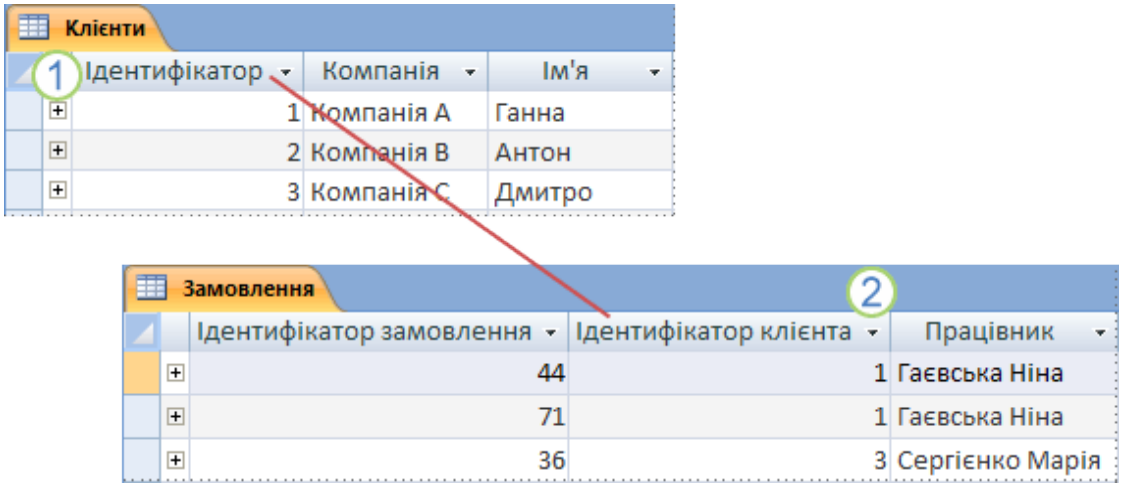
Класична реляційна модель передбачає неподільність даних, які зберігаються в полях записів таблиць. Існує ряд випадків, коли це обмеження заважає ефективній реалізації програм.

7.2. Поняття ключового елементу. Первинний ключ таблиці

Первинний ключ – це поле або набір полів у таблиці, які надають програмі Microsoft Office Access 2007 - Українська версія унікальний

ідентифікатор кожного рядка. У реляційній базі даних, такій як Office Access 2007, відомості розділяються на окремі тематичні таблиці. Відтак зв'язки між таблицями та первинні ключі використовуються програмою Access для об'єднання даних. У програмі Access використовуються поля первинних ключів для швидкого поєднання даних із кількох таблиць й об'єднання цих даних.

Після визначення первинного ключа його можна використовувати в інших таблицях для звертання до таблиці з первинним ключем. Наприклад, поле «Ідентифікатор клієнта» в таблиці «Клієнти» може також відображатися в таблиці «Замовлення». У таблиці «Клієнти» це первинний ключ. У таблиці «Замовлення» це поле називається «зовнішнім ключем». Тобто зовнішній ключ – це первинний ключ іншої таблиці (рис. 7.1).



1 Первинний ключ

2 Зовнішній ключ

***Рис. 7.1** Первинний ключ*

Часто унікальний ідентифікаційний номер, такий як ідентифікаційний номер або серійний номер чи код, є первинним ключем в таблиці. Наприклад, є таблиця «Клієнти», в якій кожний клієнт має унікальний ідентифікатор. Поле ідентифікатора клієнта є первинним ключем.

Поле, яке найкраще підходить для первинного ключа, має кілька характеристик. По-перше, значення поля унікальним чином ідентифікує кожний рядок. По-друге, воно ніколи не буває пустим або нульовим — завжди містить значення. По-третє, поле рідко (найкраще — ніколи) змінюється. У

програмі Access поля первинних ключів використовуються для швидкого зведення даних із кількох таблиць.

Прикладом неправильного вибору первинного ключа може стати поле імені або адреси. Обидва параметри містять відомості, які можуть із часом змінитися.

Для таблиці завжди потрібно вказувати первинний ключ. У програмі Access автоматично створюється індекс первинного ключа, який допомагає прискорити виконання запитів та інших операцій. Також у програмі Access кожен запис має в полі первинного ключа значення, яке є унікальним.

Під час створення нової таблиці в поданні таблиці програмою Access автоматично створюється первинний ключ, якому призначається ім'я поля типу даних «Ідентифікатор» і «Автонумерація». За промовчанням в поданні таблиці поле приховано, але його можна побачити, якщо перейти до подання конструктора.

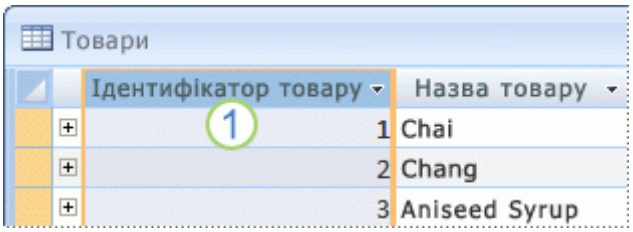


Рис. 7.2 Стовпець з типом даних «Автонумерація»

Якщо важко вибрати поля або набір полів для первинного ключа, можна використати стовпець, який містить тип даних «Автонумерація». Такий ідентифікатор не містить фактичних даних, що описують рядок, якому він відповідає. Доцільно використовувати ідентифікатори без даних, оскільки їхні значення не змінюються. Первинний ключ, який містить факти про рядок, — номер телефону або ім'я клієнта — може змінюватися, оскільки змінитися можуть власне відомості.

Стовпець (рис 7.2) із типом даних «Автонумерація» часто є найкращим вибором для первинного ключа, оскільки всі ідентифікатори продуктів різні.

У деяких випадках як первинний ключ таблиці можна використати кілька полів. Наприклад, для таблиці «Відомості про замовлення», в якій зберігаються

позиції для замовлень, у первинному ключі можна використати два стовпці: «Код замовлення» та «Код товару». Первинний ключ, який використовує кілька стовпців, називається складним ключем.

Додавання первинного ключа авто нумерації.

Під час створення нової таблиці в поданні таблиці програмою Access автоматично створюється первинний ключ, якому призначається поле типу даних «Автонумерація». Якщо вже є наявна таблиця, до якої слід додати поле первинного ключа, потрібно відкрити таблицю в поданні конструктора.

Натисніть кнопку Microsoft Office  і виберіть пункт Відкрити.

У діалоговому вікні Відкрити виберіть і відкрийте базу даних.

В області переходів клацніть правою кнопкою миші таблицю, до якої потрібно додати первинний ключ, і в контекстному меню виберіть пункт Конструктор.

Знайдіть перший доступний пустий рядок у сітці таблиці.

У стовпці Ім’я поля введіть ім’я, наприклад «Ідентифікатор клієнта».

У стовпці Тип даних клацніть стрілку розкривного списку та виберіть пункт Автонумерація.

У розділі Властивості поля для властивості Нові значення виберіть значення Поступово для використання послідовних числових значень для первинного ключа або виберіть Випадково для використання випадкових чисел.

Установлення первинного ключа.

Якщо є таблиця, в якій кожний запис має унікальний номер ідентифікатора (наприклад, ідентифікатор, серійний номер або код), таке поле можна використати як первинний ключ. Найкращий первинний ключ — це поле, яке унікально ідентифікує кожен рядок, не містить пустих значень та рідко (або ніколи) змінюється.

Для встановлення первинного ключа потрібно перейти до подання конструктора.

Натисніть кнопку Microsoft Office  і виберіть пункт Відкрити.

У діалоговому вікні Відкрити виберіть і відкрийте базу даних.

В області переходів клацніть правою кнопкою таблицю, в якій потрібно встановити первинний ключ, і в контекстному меню виберіть пункт Конструктор.

Виберіть поле або поля, які потрібно використати як первинний ключ.

Для вибору одного поля натисніть маркер виділення рядка для потрібного поля.

Для вибору кількох полів утримуйте клавішу CTRL і натисніть маркери виділення рядка для кожного поля.

На вкладці Конструктор у групі Знаряддя виберіть елемент Ключове поле.

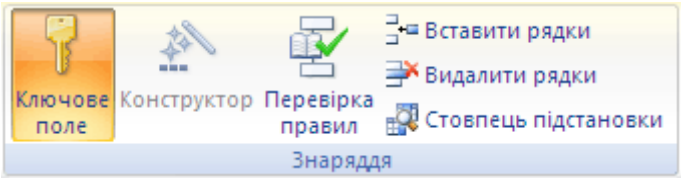


Рис. 7.3 *Піктограма ключа таблиці*

Піктограма ключа додається ліворуч від поля або полів, вибраних як первинний ключ.

Видалення первинного ключа.

У разі видалення первинного ключа поле або поля, які до цього були первинним ключем, більше не є основним засобом ідентифікації запису. Проте видалення первинного ключа не видаляє поля або полів з таблиці. Але з полів видаляється значення первинного ключа.

Видалення первинного ключа призводить також до видалення індексу, створеного для первинного ключа.

Натисніть кнопку Microsoft Office і виберіть пункт Відкрити.

У діалоговому вікні Відкрити виберіть і відкрийте базу даних.

Перед видаленням первинного ключа переконайтеся, що він не входить до зв’язків між таблицями. Якщо спробувати видалити первинний ключ, який є частиною зв’язку, відобразиться повідомлення програми Access про те, що спершу слід видалити зв’язок.

Видалення зв’язку між таблицями.

Якщо відкрито таблиці, які є частинами зв’язку, закрийте їх. Не можна видаляти зв’язок таблиці між відкритими таблицями.

На вкладці Знаряддя бази даних у групі Відобразити або приховати виберіть елемент Зв’язки.

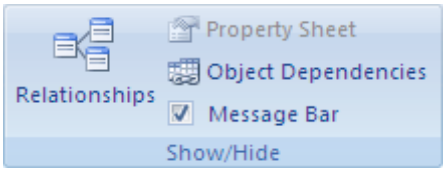


Рис. 7.4 Зв’язки таблиці

Якщо таблиці, які є частинами зв’язку, не відображаються, на вкладці Конструктор у групі Зв’язок виберіть елемент Відобразити таблицю. Відтак виберіть таблиці, які слід додати, у діалоговому вікні Відображення таблиці, натисніть кнопку Додати та кнопку Закрити.

Виберіть зв’язок між таблицями, який слід видалити (лінія стає жирною, якщо її вибрано у разі вибору), і натисніть клавішу DELETE.

На вкладці Конструктор у групі Зв’язок виберіть елемент Закрити.

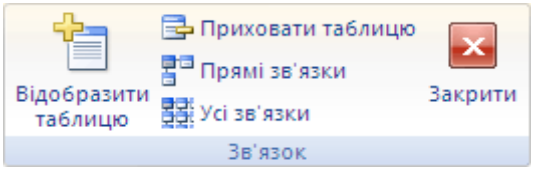


***Рис. 7.4** Вкладка конструктор групи «Звязок»*

В області переходів клацніть правою кнопкою таблицю, в якій потрібно видалити первинний ключ, і в контекстному меню виберіть пункт Конструктор.

Клацніть маркер виділення рядка для поточного первинного ключа.

Якщо первинний ключ складається з одного поля, клацніть маркер виділення рядка для цього поля.

Якщо первинний ключ складається з кількох полів, клацніть маркер виділення рядка для будь-якого поля первинного ключа.

На вкладці Конструктор у групі Знаряддя виберіть елемент Ключове поле (рис. 7.3).

Піктограма ключа видаляється з поля або полів, які було попередньо вибрано як первинний ключ.

7.3. Зв’язки між таблицями та цілісність даних

Сучасні бази даних зазвичай містять велику кількість взаємозв'язаних таблиць, що дозволяє уникнути повторів. Наприклад, крупні фірми можуть зберігати відомості про магазини в одній таблиці, номенклатуру товарів, отриманих цими магазинами в поточному місяці в іншій таблиці, а відомості про оптових покупців в третій таблиці. Access дозволяє працювати одночасно з декількома таблицями, кожна з яких повинна містити записи, присвячені певній темі. Зв'язок між ними встановлюється по загальних для декількох таблиць полях, наприклад, номера магазинів, через які здійснюється реалізація товару. Бажано, аби в одній з таблиць загальне поле було ключовим. Якщо таблиці взаємозв'язані, то зміни, виконані в записі однієї таблиці, можуть впливати на записі в іншій таблиці.

Для збереження повноти і цілісності даних Access накладає певні обмеження на введення і редагування даних, наприклад, неможливо видалити запис з однієї таблиці, якщо існують пов'язані з нею записи в інших таблицях.

Види зв'язків між таблицями

Реляційна база даних може містити велику кількість взаємозв'язаних таблиць. Зв'язки встановлюється між двома загальними полями (стовпцями) двох таблиць. Зв’язуванні поля можуть мати різні імена, але повинні мати однакового типа даних за винятком випадку, коли поле первинного ключа є полем типа Лічильник. Поле лічильника зв'язується з числовим полем, якщо значення властивості Розмір поля (FieldSize) обоє полів збігаються. Наприклад, якщо властивість обоє полів має значення Довге ціле. Навіть у тому випадку, коли зв'язуються поля типа «Числовою», їх властивості Розмір поля (FieldSize) повинні мати однакові значення.

Задавши зв'язки між таблицями, можна створити запити, форми і звіти для відображення відомостей, представлених в декількох таблицях. Між двома таблицями можуть існувати наступні зв'язки:

один до одного – при такому типові зв'язку одного запису в першій таблиці відповідає лише одна запис в іншій таблиці. В цьому випадку слід перевірити можливість розміщення всіх записів в одній таблиці. Проте у ряді випадків можна використовувати декілька простіших таблиць. Відповідність записів встановлюється по полю, яке є первинним ключем в першій таблиці, і полю, званим зовнішнім ключем іншої таблиці;

один до багатьох – в цьому випадку запис однієї таблиці може мати декілька погоджених з нею записів в іншій таблиці. При цьому кожен запис в другій таблиці узгоджується лише з одним записом в першій таблиці. Наприклад, кожен покупець може купити декілька товарів, але кожен проданий товар має лише одного покупця. Поле, що містить первинний ключ нової таблиці, зв'язується із зовнішнім ключем старої. Значення в полі із зовнішнім ключем можуть повторюватися. Це найбільш поширений тип зв’язків.

багато до одного – будь-якому запису таблиці, зв'язок з якою ми розглядаємо, можуть відповідати декілька записів нової таблиці, але не навпаки. Фактично це відношення один до багатьом, що розглядається, в зворотному порядку. В цьому випадку ключове поле нової таблиці є зовнішнім ключем;

багато до багатьох – кожному запису з однієї таблиці може відповідати будь-яка кількість записів в іншій таблиці і навпаки. Наприклад, кожна людина може дзвонити з декількох телефонів. З іншого боку деякими телефонами можуть користуватися декілька чоловік. В цьому випадку поля, по яких встановлюється зв'язок, є зовнішніми ключами. Вони можуть містити значення, що повторюються.

Використання ключових полів для завдання зв'язку між таблицями

В більшості випадків ключове поле однієї таблиці пов'язують із співпадаючим полем (зовнішнім ключем) іншої таблиці. Зовнішній ключ – це одне або декілька полів (стовпців) в таблиці, що містять посилання на поле або поля первинного ключа в іншій таблиці. Поле зовнішнього ключа визначає

спосіб скріплення таблиць – вміст поля зовнішнього ключа повинен збігатися з вмістом ключового поля, хоча імена полів можуть при цьому не збігатися.

Схема даних.

Для управління базою даних використовуються зв'язки між таблицями. Якщо між таблицями бази даних зв'язку не були задані, відкриється діалогове вікно Додавання таблиці (Show Table). Виберіть у вікні необхідні таблиці і запити і натискуйте кнопку Додати (Add). Вони відображуватимуться у вікні Схема даних (Relationships) .

Символи на лініях зв'язку показують тип зв'язку. Символ нескінченності використовується для позначення «багато», і якщо ми бачимо на одній лінії зв'язку символи одиниці і нескінченності, то між таблицями існує зв'язок один до багатьом. Можна пов'язувати поля з різними іменами, а також запити з таблицями або запитами .

Якщо потрібно ввести нову таблицю у вікно Схема даних (Relationships) і встановити зв'язок між таблицями, то слід виконати наступні дії:

закрити всі таблиці і форми;

у вікні бази даних вибрати команду Схема даних (Relationships) у меню Сервіс (Tools) або натиснути однойменну кнопку на панелі інструментів;

якщо у вікні Схема даних (Relationships) буде представлена яка-небудь інформація, то потрібно натиснути спочатку кнопку Очистити макет (Clear Layout) а потім кнопку Так (Yes) аби продовжити виконання наміченої операції;

у меню Зв'язки (Relationships) слід вибрати команду Додати таблицю(Show Table) або натиснути однойменну кнопку на панелі інструментів;

у діалоговому вікні Додавання таблиці (Show Table) на вкладці Таблиці (Tables) слід виділити необхідну таблицю і натиснути кнопку Додати (Add);

у вікні Схема даних (Relationships) відображуватиметься список полів вибраної таблиці з виділеним полем ключа;

у діалоговому вікні Додавання таблиці (Show Table) слід виділити таблицю, з якою встановлюється взаємозв'язок і натиснути кнопку Додати (Add);

у вікні Схема даних (Relationships) перетягнути ключове поле із списку головної таблиці в список зв'язаної таблиці.

7.4. Графічне позначення реляційної моделі

Довільний фрагмент предметної області може бути представлений як множина сутностей, між якими існує деяка множина зв'язків. Графічне відбиття цього представлення й утворює ER-діаграму. У процесі цього графічного представлення сутності різного типу відбиваються прямокутниками різної форми та обрамлення; атрибути - зв'язаними відрізками ліній з прямокутниками сутностей овалами або впровадженими всередину прямокутників записами; зв'язки – відрізками ліній, всередині яких може бути введений графічний елемент з текстом, що описує тип зв'язку, а на кінцях – графічне або текстове позначення кардинальності зв'язку. ER-діаграма задає графічне представлення концептуальної схеми БД, конкретний вигляд якого залежить як від предметної галузі та мети створення БД, так і від обраної нотації, тобто стандартизованого способу графічного представлення елементів ER-діаграми.

Приклад. Розглянемо множину працівників певного підприємства. Кожного з них можна описати за допомогою характеристик табельний номер (number), ім'я (name), вік (age). Відповідно, сутність СПІВРОБІТНИК має атрибути ТАБЕЛЬНИЙ_НОМЕР, ІМ'Я, ВІК.

Сутність фактично становить із себе множину атрибутів, що описують властивості всіх членів даного набору сутностей. Надалі для визначення сутності і її атрибутів будемо використовувати інфологічне представлення, яке для сутності СПІВРОБІТНИК набуває вигляду: СПІВРОБІТНИК (ТАБЕЛЬНИЙ_НОМЕР, ІМ'Я, ВІК). Тоді відділи, на які підрозділяється підприємство, і в який працюють співробітники, можна описати як ВІДДІЛ (НОМЕР_ВІДДІЛУ, НАЙМЕНУВАННЯ).

Множину значень (область визначення) атрибуту називають доменом. Наприклад, для атрибута ВІК домен (назвемо його КІЛЬКІСТЬ_РОКІВ) задається інтервалом цілих чисел більше нуля, оскільки людей з від'ємним віком не буває.

П.Чен визначає атрибут як функцію, що відбиває набір сутностей у набір значень чи у декартів добуток наборів значень. Так, атрибут ВІК здійснює відображення в набір значень (домен) КІЛЬКІСТЬ_РОКІВ. Атрибут ІМ'Я здійснює відображення в декартів добуток наборів значень ІМ'Я, ПРІЗВИЩЕ і ПО_БАТЬКОВІ. Звідси випливає визначення ключа сутності як такої групи атрибутів, що відображення набору сутностей у відповідну групу наборів значень є взаємно однозначним відображенням. Іншими словами, ключ сутності - це один чи більш атрибутів, які унікально визначають дану сутність. У нашому прикладі ключем сутності СПІВРОБІТНИК є атрибут ТАБЕЛЬНИЙ_НОМЕР (звичайно, тільки в тому випадку, якщо всі табельні номери на підприємстві унікальні).

Виходячи з визначення зв'язку (relationship) як асоціації, установлена між кількома сутностями, для розглянутого прикладу можна виділити, наприклад, такі зв'язки:

оскільки кожен співробітник працює в якому-небудь відділі, між сутностями СПІВРОБІТНИК і ВІДДІЛ існує зв'язок "працює у" чи ВІДДІЛ-ПРАЦІВНИК;

тому що один із працівників відділу є його керівником, то між сутностями СПІВРОБІТНИК і ВІДДІЛ є зв'язок "керує" чи ВІДДІЛ-КЕРІВНИК;

можуть існувати і зв'язки між сутностями одного типу, наприклад зв'язок БАТЬКО - НАЩАДОК між двома сутностями ЛЮДИНА;

(Слід зазначити, що в методиці проектування даних є неписане правило, відповідно до якого сутності позначаються за допомогою іменників, а зв'язки - дієслівними формами. Дане правило, однак, не є обов'язковим) .

Не існує загальних правил визначення, що вважати сутністю, а що зв'язком. У розглянутому вище прикладі ми поклали, що "керує" - це зв'язок.

Однак можна розглядати сутність "керівник", що має зв'язки "керує" із сутністю "відділ" і "є" із сутністю "співробітник". Зв'язок також може мати атрибути. Наприклад, для зв'язку ВІДДІЛ-ПРАЦІВНИК можна задати атрибут СТАЖ_РОБОТИ_У_ВІДДІЛІ.

Кожний зв'язок характеризується як зв'язністю (чи типом) зв'язку, так і степенем (потужністю, кардинальністю) зв'язку, і ці два поняття не слід плутати. Можуть існувати наступні зв'язності бінарних зв'язків: 1:1, 1:M, M:M. Графічно зв'язність та кардинальність зв'язків відбиваються по-різному у різних нотаціях; найрозповсюдженіші варіанти розглянуто нижче.

Зв'язок один до одного (позначається 1:1) означає, що в такому зв'язку сутності з однією роллю завжди відповідає не більш однієї сутності з іншою роллю. У розглянутому нами прикладі це зв'язок "керує", оскільки в кожнім відділі може бути тільки один начальник, а співробітник може керувати тільки одним відділі. Даний факт представлений на наступному малюнку, де прямокутники позначають сутності, а ромб - зв'язок (як це прийнято у нотації Чена). Тому що степінь зв'язку для кожної сутності дорівнює 1, то вони з'єднуються одною лінією (рис. 7.5).

Рис. 7.5. *Позначення зв'язку типу 1:1 при кардинальностях 0,1:1,1*

Іншою важливою характеристикою зв'язку крім її степеня є клас приналежності сутностей, що в неї входять, чи обов'язковість зв'язку. Тому що в кожнім відділі обов'язково повинен бути керівник, те кожній сутності "ВІДДІЛ" неодмінно повинна відповідати сутність "СПІВРОБІТНИК". Однак, не кожен співробітник є керівником відділу, отже в даному зв'язку не кожна сутність "СПІВРОБІТНИК" має асоційовану з нею сутність "ВІДДІЛ". Таким чином, сутність "СПІВРОБІТНИК" має обов'язковий клас приналежності (цей факт позначається також вказівкою інтервалу числа можливих входжень сутності в зв'язок, у даному випадку це 1,1), а сутність "ВІДДІЛ" має

необов'язковий клас приналежності (0,1). Тепер даний зв'язок ми можемо описати як 0,1:1,1, і саме цей опис задаватиме кардинальності зв'язку, тобто кількість екземплярів зв'язаної сутності, з якою зв'язаний кожний екземпляр даної сутності.

Зв'язок один до багатьох (1 : n). У даному випадку сутності з однією роллю може відповідати будь-яка кількість сутностей з іншою роллю. Таким є зв'язок ВІДДІЛ-СПІВРОБІТНИК. У кожнім відділі може працювати довільне число співробітників, але співробітник може працювати тільки в одному відділі. Графічно степінь зв'язку n відображається "деревоподібною" лінією, так це зроблено на наступному малюнку. Як видно з рис. 7.6, між двома сутностями може бути визначено кілька наборів зв'язків.

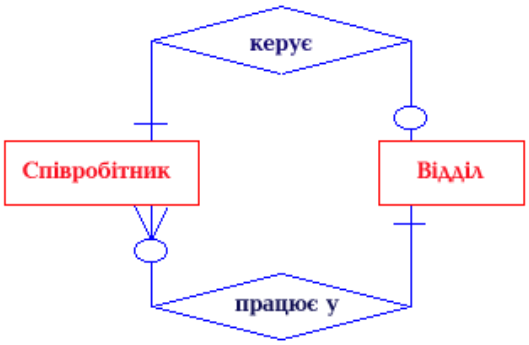


Рис. 7.6. *Позначення зв'язку типу 1* n

(СПІВРОБІТНИК - працює у - ВІДДІЛ), 1,1:0,n (кожний співробітник працює лише у одному відділі, 1:1, кожний відділ може нараховувати скільки завгодно співробітників, у тому числі жодного, 0:n) необхідно також враховувати клас приналежності сутностей. Кожен співробітник повинен працювати в якому-небудь відділі, але не кожен відділ (наприклад, ново сформований) повинен включати хоча б одного співробітника. Тому сутність "ВІДДІЛ" має обов'язковий, а сутність "СПІВРОБІТНИК" необов'язковий класи приналежності.

Зв'язок багато до багатьох (n : n, або M:M, або N:M). У цьому випадку кожна з асоційованих сутностей може бути представлена будь-якою кількістю екземплярів. Нехай на розглянутому нами підприємстві для виконання кожного контракту (сутність КОНТРАКТ) створюється робоча група, у яку входять

співробітники різних відділів. Оскільки кожен співробітник може входити в кілька (у тому числі й у жодну) робочих груп, а кожна група повинна включати не менш одного співробітника, то зв'язок між сутностями СПІВРОБІТНИК і РОБОЧА_ГРУПА має степінь n : n.

Рис. 7.7. *Позначення зв'язку типу n:n*

Якщо існування сутності x залежить від існування сутності y, то x називається залежною сутністю (іноді сутність x називають "слабкою", а "сутність" y – “сильною”). Як приклад розглянемо зв'язок між раніше описаними сутностями РОБОЧА_ГРУПА і КОНТРАКТ. Робоча група створюється тільки після того, як буде підписаний контракт із замовником, і припиняє своє існування по виконанні контракту. Таким чином, сутність РОБОЧА_ГРУПА є залежною від сутності КОНТРАКТ. Залежну сутність будемо позначати подвійним прямокутником, а її зв'язок із сильною сутністю - лінією зі стрілкою:

Рис. 7.8. *Позначення залежних і незалежних сутностей*

Кардинальність зв'язку з боку сильної сутності завжди буде (1,1). Клас приналежності і степінь зв'язку для залежної сутності можуть бути будь-якими. Припустимо, наприклад, що розглянуте нами підприємство користається кількома банківськими кредитами, що представляються набором сутностей КРЕДИТ (НОМЕР_ДОГОВОРУ, СУМА, ТЕРМІН_ПОГАШЕННЯ, БАНК). По кожному кредитові повинні здійснюватися виплати відсотків і платежі в рахунок його погашення. Цей факт представляється набором сутностей ПЛАТІЖ (ДАТА, СУМА) і набором зв'язків "здійснюється по". У тому випадку, коли одержання запланованого кредиту скасовується, інформація про нього повинна бути вилучена з бази даних. Відповідно, повинні бути вилучені і всі

відомості про планові платежі по цьому кредиту. Таким чином, сутність ПЛАТІЖ залежить від сутності КРЕДИТ.

Рис. 7.9. *Сутність ПЛАТІЖ залежна і існує дише у разі існування сутності КРЕДИТ*

Висновки

Створення реляційної бази даних це звичайних процес програмування з графічним зображенням зв'язків та співвідношень.

Література [6,10,13-16]

Контрольні питання

1. Наведіть переваги і недоліки реляційної бази даних.
2. Що таке ключовий елемент? Первинний ключ таблиці.
3. Переваги графічної схеми даних.

Наведіть поняття реляційної алгебри: *множина, кортеж, атрибут*.

Питання для самостійного вивчення

1. Нормалізація даних.
2. Перша, друга, третя нормальні форми.
3. Створення структури таблиць за допомого майстра та конструктора в СУБД MS Access
4. Типи даних в MS Access
5. Налаштування властивостей полів та підстановок в таблицях
6. Створення зв'язків між таблицями. Вікно «Схема даних».
7. Додавання та редагування даних в таблицях MS Access

Лекція 8. Методи відбору даних

Мета: Навчитися оперувати даними при створенні запитів.

План

8.1. Поняття запиту. Типи запитів.

8.2. Основні елементи конструктору запитів в MS Access.

8.3. Запит на вибірку. Технологія створення.

8.4. Запит з параметром.

8.1. Поняття запиту. Типи запитів.

Головне призначення будь-якої бази даних полягає в зберіганні та наданні інформації, яка може знадобиться користувачу або відразу після введення даних, або через роки. Для отримання інформації з бази даних потрібні хоч би поверхневі знання про структуру інформації в базі.

Наприклад, звіти в шафі можуть бути вручну відсортовані по роках і порядкових номерах. Щоб знайти конкретний звіт, потрібно знати рік його складання і номер. У добре організованій картотеці завжди ведеться каталог звітів. У ньому всі звіти згруповані по типу (а не по темі) в алфавітному порядку. Такий каталог, звичайно, дуже корисний, але якщо відома лише тема звіту і приблизна дата його складання, доведеться проглянути всі розділи каталогу, щоб знайти потрібний звіт.

В порівнянні з картотеками, комп'ютерні бази даних володіють явною перевагою. За допомогою спеціальних інструментів можна легко одержати інформацію, що задовольняє практично будь-якому заданому критерію. У можливості перегляданння інформації будь-яким зручним користувачу способом і полягає реальна потужність бази даних. Дані, одержані у відповідь на підготовлений користувачем запит можна використовувати для створення звітів, форм і діаграм.

Запит в Microsoft Access – це вимога надати інформацію, накопичену в таблицях Access. Інформацію можна одержати за допомогою інструментів запиту. Запит може обробляти інформацію одну або декілька зв'язаних

таблиць. При цьому Microsoft Access видає тільки ту інформацію, яку запрошував користувач. Після створення та запуску запиту Microsoft Access відображає дані у вигляді записів, які запрошував користувач. Ці записи називаються *динамічним набором записів.*

Access підтримує різні типи запитів, які можна поділити на сім основних категорій.

Запити на вибірку (простий запит). Найпоширеніший тип запиту. При його виконанні створюється безліч записів, в яких містяться вказані дані з однієї або декількох таблиць. Інформацію, що при цьому відображається в запиті, можна змінювати також, як і при роботі із звичайною таблицею. Крім того запит використовується для угрупування записів, а також для обчислення сум, середніх значень тощо.

Груповий запит. Є спеціальною версією запиту на вибірку. Дозволяє обчислювати суми, підраховувати кількість записів і виконувати розрахунки підсумкових значень. При виборі цього типу запиту Access додає в бланк запиту рядок “Групова операція”.

Запит на зміну. Дозволяє створювати нові таблиці (команда Створення таблиці) або змінювати дані в існуючих таблицях (команди Видалення, Оновлення і Додавання). Якщо в наборі результатів запиту на вибірку можна вносити зміни тільки в один запис за раз, то запит на зміну дозволяє вносити зміни в декілька записів відразу при виконанні однієї операції.

Перехресний запит. Відображає результати статистичних розрахунків (такі як суми, кількість записів і середні значення). Ці результати групуються по двох наборах даних у форматі перехресної таблиці. Перший набір виводиться в стовпці зліва та утворює заголовки рядків, а другий виводиться у верхньому рядку і утворює заголовки стовпців.

Запит SQL. Існують три типу запитів SQL (Structured Query Language): *запит на об’єднання, запит до сервера та керуючий запит,* які використовуються для маніпуляцій з базами даних SQL архітектури

клієнт/сервер. Створюють ці запити за допомогою написання спеціальних інструкцій SQL.

Запит з обмеженням. Цей обмежувач запиту можна використовувати тільки в кон'юнкції з іншими п'ятьма типами запитів. Він дозволяє задавати число перших записів або частину загальної кількості записів у відсотках, яку ви хотіли б одержати у будь-якому вигляді запиту.

8.2. Основні елементи конструктору запитів в MS Access

Простий запит – це різновид запиту, який дозволяє вибрати дані з однієї або декількох таблиць в інтерактивному режимі. При роботі з даним видом запиту користувач тільки указує критерії вибірки записів. Перевага даного виду запиту полягає в тому, що користувачу не потрібно запам’ятовувати спеціальну мову команд – взаємодія між системою і користувачем ведеться в природній (діалогової) формі. У цьому криється причина популярності даного запиту – їм може скористатися будь-який, у тому числі і починаючий користувач.

Загальний алгоритм створення запиту за допомогою Конструктора:

Для створення запиту за допомогою Конструктора запитів користуються наступним алгоритмом.

Перейти у вкладку Запити.

Активізувати кнопку Створити і вибрати в діалоговому вікні “Новий запит” опцію Конструктор.

У діалоговому вікні “Додавання таблиці” вибрати потрібні таблиці, потім закрити це вікно.

У списках полів таблиць задати поля (двічі клацнувши мишею потрібне поле на схемі даних або скориставшись технологією drag-and-drop), які потрібно використовувати як елементи запиту.

У стовпцях всіх полів, які потрібно сортувати, клацнути на рядку Сортування і вибрати варіант “За збільшенням” або “По убуванню”.

Якщо які-небудь поля запиту повинні бути приховані, скинути для них прапорці в рядку “Вивід на екран”.

Внести необхідні вирази як відбір полів.

Зберегти запит.

Вікно Конструктора запитів розбите на дві частини – верхню і нижню.

У верхній частині вікна Конструктора запитів знаходиться схема даних запиту. Ця схема подібна до схеми даних бази даних.

У нижній частині вікна Конструктора запитів розміщується бланк запиту. Кожен рядок цього бланка виконує визначає певну функцію:

Поле. У цьому рядку розміщуються ті поля, які користувач використовує для створення запиту (кожне в своєму елементі таблиці).

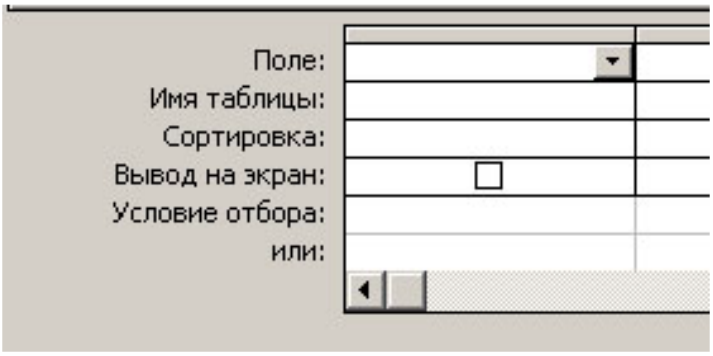


Рис. 8.1. *Рядок запиту поле*

Ім'я таблиці. Цей рядок "повідомляє" користувачу, з якої таблиці (або запиту) вибрано дане поле.

Сортування. У даному рядку користувач задає тип впорядковування інформації, яка повертається в запиті.

Виведення на екран. Прапорець переглядання поля встановлюється, якщо необхідно, щоб інформація з даного поля виводилася на екран. Проте нерідко поле використовується виключно в цілях завдання умови вибірки даних. У такому разі його буде зайвим виводити на екран, і, відповідно, прапорець переглядання поля не встановлюється.

Умова відбору. У цьому рядку і в рядках, розташованих нижче, користувач вводить вирази, що визначають умови відбору полів. Під виразом мається на увазі комбінація імен полів, змінних, арифметичних або інших операторів, які використовуються в даному випадку для перевірки даних.

Умова відбору – це критерій, який вводиться в рядку таблиці Конструктора запиту, і на основі якого здійснюється відбір записів.

8.3. Запит на вибірку. Технологія створення.

Запит - це об'єкт бази даних, що дозволяє відбирати інформацію з однієї або декількох таблиць за вказаними умовами. Запит може не тільки вибирати інформацію (це робиться в запитах «на вибірку»), а й робити обчислення, коригувати поля бази даних, вилучати записи тощо. Запит — найважливіший об'єкт роботи з даними. Розглянемо послідовно основні типи запитів, що працюють в системі Access.

Робота з однією таблицею

Насамперед потрібно навчитися записувати умови відбору даних. Без умов дія запиту не має сенсу.

Для числових полів умови записують як арифметичні або логічні вирази. Наприклад, «<100» , «>40». В умовах можна використати логічні функції «And» , «Or», «Not». Наприклад, умова «>40 And <60» відбирає значення поля, що є більші за 40 і менші за 60.

При формуванні запитів, що мають відбирати значення, перевіряючи їх належність до інтервалу, зручно застосовувати функцію:

«Beetween (НижняяГраница) And (ВерхняяГраница)»

Наприклад, вираз «Beetwen 40 And 60» відбирає значення, що знаходяться в інтервалі від 40 до 60.

При роботі з текстовими полями часто виникає потреба перевіряти інформацію на відповідність певному шаблону. Прикладом такого запиту може бути вибір з базі даних всіх осіб, чиє прізвище починається з «КАР».

Таку дію можна реалізувати з використанням функції «Like "Рядок пошуку із символами шаблона"». Знак «*» у шаблоні заміняє довільну кількість символів, що можуть стояти на місці даної позиції. Знак шаблону «?» заміняє поодинокий символ у позиції, де він знаходиться. Символ «#» вказує, що на його місці має стояти цифра. Наприклад, умова «Like "КАР*"» , будучі

накладеною на поле прізвищ відбирає з бази даних записи, чиє прізвище починається з «КАР».

Запит, що відбирає дані з однієї таблиці за умовою

Побудуємо запит, що відбирає з таблиці «Книги» екземпляри, ціна яких перебільшує 20 гривень і водночас прізвище автора починається або з букви «Г» або «К». Переходимо на вкладку «Запити». Натискаємо «Создание запроса в режиме конструктура» (рис. 8.2).

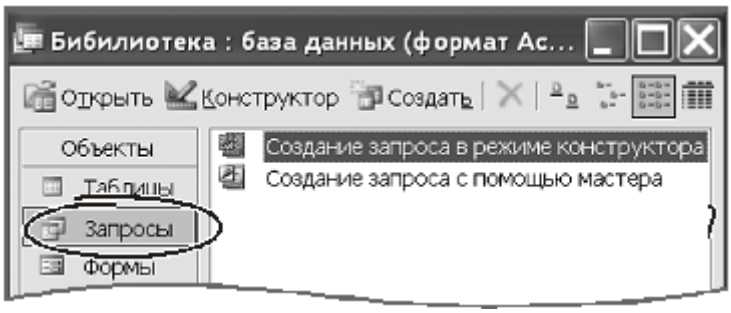


Рис. 8.2. *Створення запита в режимі конструктора*

Як альтернатива — можна натиснути кнопку «Создать» і вибрати варіант «Конструктор» або скористатися правою кнопкою миші. У вікні діалогу «Додавання таблиці» на вкладці «Таблиці» вибираємо таблицю, з якої потрібно відібрати дані (рис. 8.3).

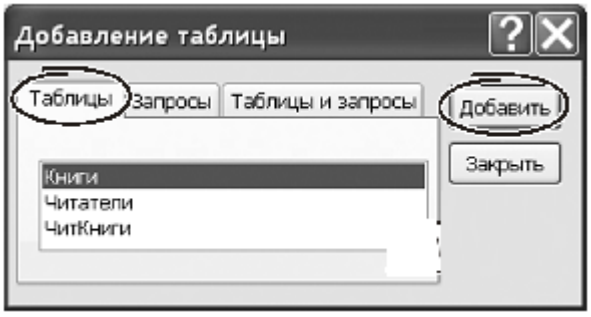


Рис. 8.3. *Додавання таблиці*

У нашому випадку такою таблицею є «Книги». Натисніть кнопку «Добавить». Закрийте вікно «Добавление таблицы». Виберіть поля, які треба включити до запиту, а саме: «Автор», «Название», «Год», «Стоимость». Щоб помістити їх до бланку запиту, двічі клацніть кнопкою миші на імені поля у таблиці. Вибрати поле таблиці можна безпосередньо у вікні запиту, вибравши їх з випадаючого списку (рис. 8.4).

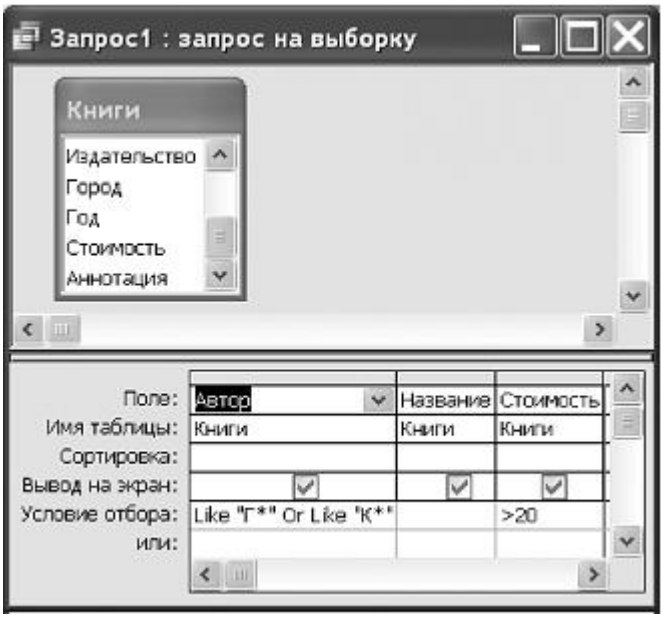


Рис. 8.4. *Вибір поля*

У вікні записуємо умови для відбирання записів. У рядку «Условия отбора» в полі «Стоимость» ставимо умову «>20». Для поля «Автор» ставимо умову «Like "Г*" or Like "R*"». Назву функції «Like» можна не вводити, вона буде додана автоматично. Щоб переглянути результат натисніть кнопку «!» або виберіть пункт меню «Вид → Режим таблицы». Результат роботи запиту показано на рис. 8.5.

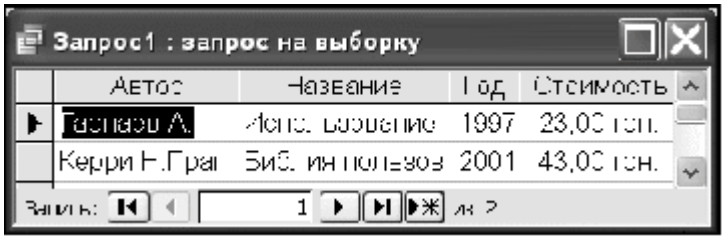


Рис. 8.5. *Результат роботи запиту*

8.4. Запит з параметром.

Процес створення запиту можна розбити на кроки, на яких, зокрема, визначаються:

- поля, які повинні бути включені до запиту;
- порядок сортування, або упорядкування, даних;
- умови відбору, які мають бути використані в запиті.

Покрокові інструкції зі створення за допомогою «Мастера запросов» звичайного запиту на вибірку *Клієнти*, що відображає усі записи

таблиць *Клієнти* і *Реалізація* за полями *Код клієнта*, *НазвФірми*, *Телефон*, *№НаклВитратп*, *Дата*, *Відмітка про оплату*

у вікні бази даних перейти до стрічки створення, обрати Мастер запросов;

у діалоговому вікні Новый запрос вибрати майстра Простой запрос;

натиснути кнопку <ОК>;

вказати ім'я таблиці або запиту, на якому має базуватися створюваний запит, а потім вибрати поля, з яких відбираються дані. Спочатку слід вказати на таблицю *Клієнти* і вибрати з неї поля *Код клієнта*, *НазвФірми*, *Телефон*;

якщо необхідно, вказати додаткові таблиці або запити, а потім вибрати з них поля. Повторювати ці дії доти, доки не будуть відібрані всі необхідні поля. Скажімо, у нашому прикладі слід вказати додатково на таблицю *Реалізація* і відібрати поля *№НаклВитрат*, *Дата*, *Відмітка про оплату*;

якщо серед вибраних до запиту полів є числові, для них можна розрахувати підсумки за функціями SUM(), AVG(), MAX(), MIN() й іншими, а також підрахувати загальну кількість записів, які запит виведе на екран.

в останньому діалоговому вікні користувачеві пропонується надати ім'я запиту, у нашому прикладі - Klienti, і вибір: виконати запит чи переглянути його структуру в режимі Конструктора запросов.

У базі даних не можна, щоб різні об'єкти - таблиці, запити, форми тощо - мали однакові назви.

Результати запиту на вибірку виводяться на екран у вигляді таблиці.

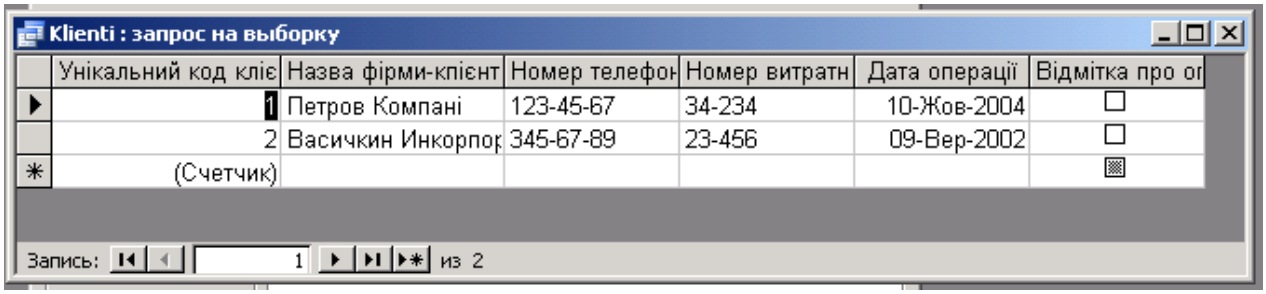


Рис. 8.6. *Результат запиту на вибірку.*

Якщо одержаний запит не відповідає вимогам користувача, можна знову звернутися до «Мастера запросов» або внести зміни до запиту в режимі Конструктора запросов.

Перейти до режиму можна, обравши запит *Клієнти* на вкладці «Запросы» у вікні відкритої бази даних натиснувши кнопку «Конструктор» на панелі інструментів вікна.

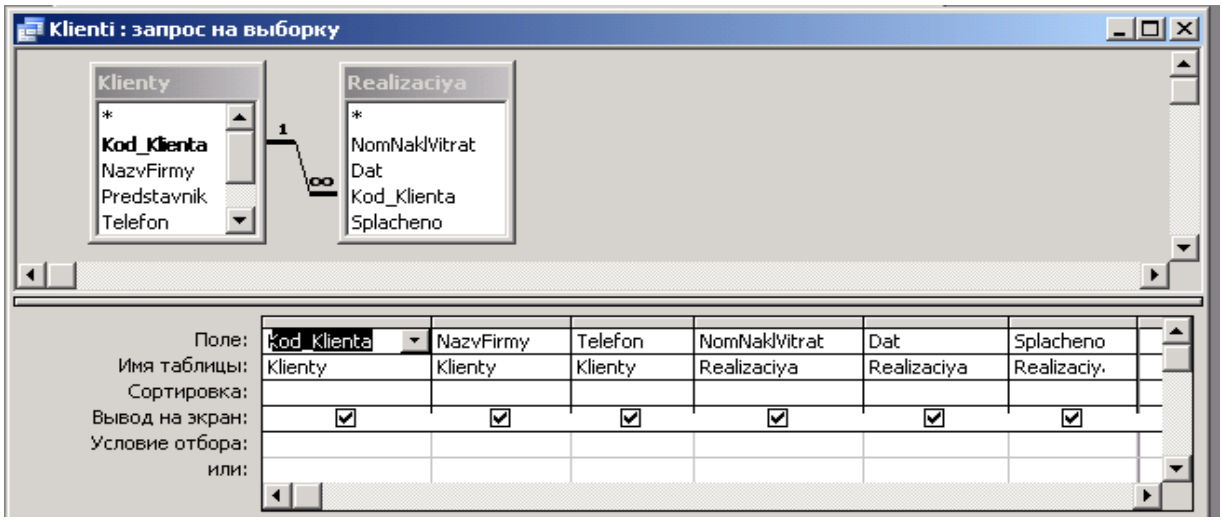


Рис. 8.7. *Створення запиту на виборку.*

При залученні до запиту інформації декількох таблиць або запитів слід переконатися на Схеме данных чи в Конструкторе запросов, що списки їхніх полів з'єднані лінією об'єднання, що дозволяє MS Access визначити його тип. Якщо користувачем раніше вже були створені зв'язки між таблицями, то в бланку Конструктора запросов при підключенні інформації додаткових таблиць до запиту лінії об'єднання виводяться на екран автоматично. Якщо встановлений режим цілісності даних, над лінією об'єднання відображається цифра 1 з боку головної таблиці і знак ∞ з боку підпорядкованої таблиці.

У режимі Конструктора запросов можна доповнити запит розрахунковими полями та сформованими користувачем умовами відбору окремих записів таблиць.

Створивши заздалегідь запит на вибірку Рух товарів з усіма полями однойменної таблиці, слід сформувати розрахункове поле Вартість та вибрати записи для товарів, вартість партії яких перевищує 50 000 грн. Для цього:

у вікні відкритої БД до стрічки створення, обрати запит *Рух товарів*, натиснути кнопку *Конструктор*. З'явиться вікно, у верхній частині якого відображена структура таблиці *Рух товарів*. Нижня частина - бланк запиту - містить опис запиту в табличній формі. Кожний стовпчик у ньому відповідає одному полю. Рядки *Поле* та *Имя таблицы* мають списки, що випадають, за допомогою яких і визначають потрібні для запиту поля;

для створення розрахункового поля Вартість слід:

встановити курсор у бланку запиту після поля *Ціна*, вставити порожній стовпчик через пункти меню *Вставка\Столбцы* та натиснути піктограму *Построитель выражений* на панелі інструментів;

у полі *Построителя выражений* набрати розрахунковий вираз: *=[Ціна]*[Кількість]*, для цього:

- вибрати в переліку полів, вибраних до запиту, поле Ціна та натиснути кнопку Вставить;

- натиснути кнопку знака множення у вікні Построителя выражений;

- вибрати в переліку полів запиту Кількість та натиснути кнопку Вставить;

- натиснути <*ОК*>.

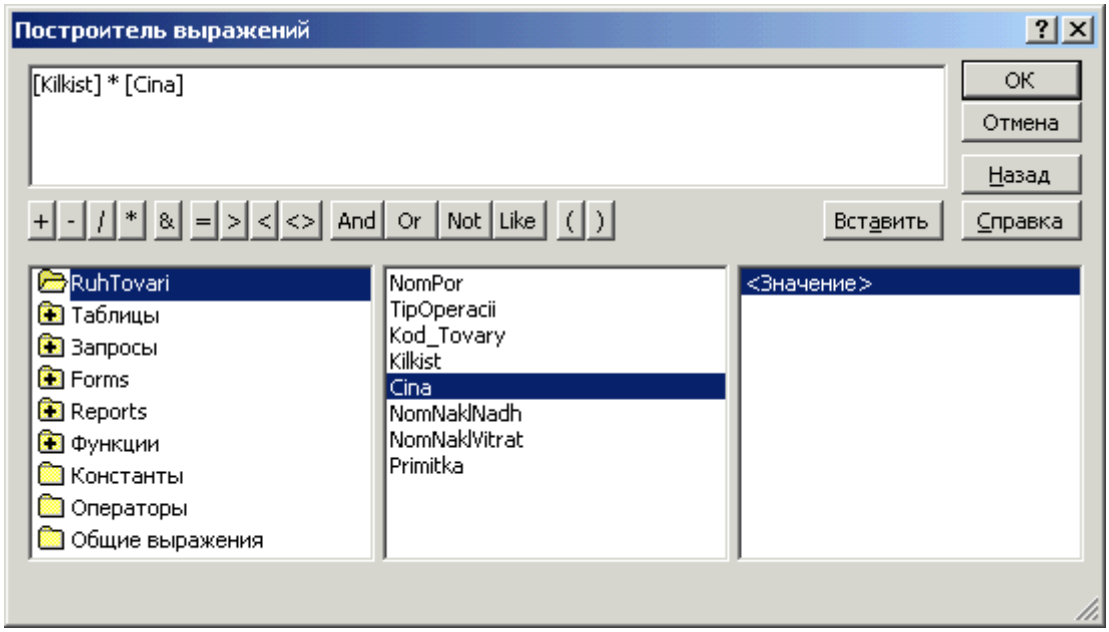


Рис. 8.8. *Вікно побудови виразу*

У бланку запиту замість слова *Выражение!*, запропонованого програмою для підпису розрахункового поля, ввести Вартість;

наприкінці сформувати умову відбору окремих записів, заповнивши рядок Условия отбора для стовпчика Вартість умовою> 50 000.

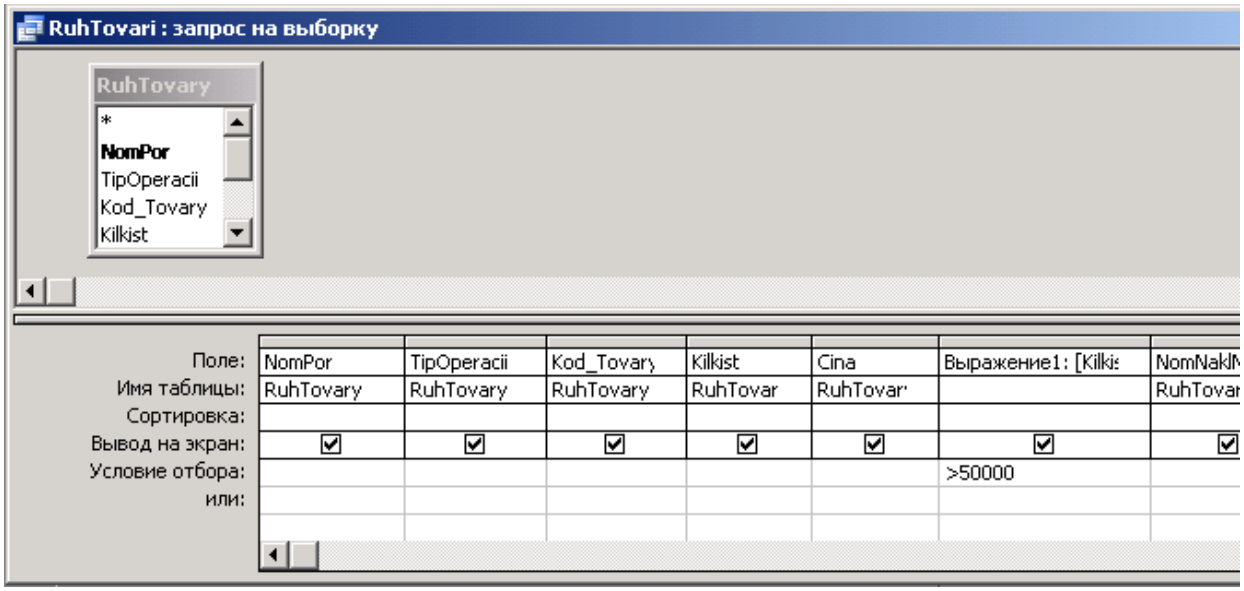


Рис. 8.9. *Вікно запиту на вибірку*

Для перегляду створеного запиту достатньо переключити режим його перегляду за допомогою кнопки зі списком Вид на панелі інструментів Конструктора запросов.

Запити з параметрами. Запит з параметрами - це запит, при виконанні якого в діалоговому вікні відображається пропозиція для користувача ввести певні дані, наприклад, умову для повернення записів з таблиць чи інших запитів. Можна розробити запит, що виводить пропозицію на введення декількох одиниць даних, наприклад, двох дат. Потім СУБД поверне всі записи, які відповідають інтервалу часу між цими датами.

Запити з параметрами також зручно використовувати як основу для форм, звітів і сторінок доступу до даних. Наприклад, на базі запиту з параметрами можна створити звіт про рух товарів за певні періоди часу. При роздрукуванні цього звіту MS Access виводить на екран запрошення ввести початок і кінець періоду, рух товарів за який має бути наведений у звіті. Після введення цих даних MS Access виконає роздрукування відповідного звіту.

Наприклад, слід створити запит з параметрами, що виводить записи про оплачену реалізацію за певний період з усіма полями таблиці Реалізація бази даних Облік товарів з прикладу. Дати початку і кінця періоду повинні вводитися під час виконання запиту у вигляді параметру. Послідовність дій:

створити запит у режимі Мастера запросов, включивши всі поля таблиці Реалізація. Дати запиту назву Oplaty_realizacii (Оплачена реалізація);

відкрити запит у режимі Конструктора запросов. У рядку *Условие отбора* для поля *Відмітка про оплат*у ввести умовуTrue (Истина), тобто дати завдання програмі виводити запити з накинутим прапорцем у полі;

у рядку *Условие отбора* для поля Дата, у якому відображаються дати, ввести запрошення такого вигляду [Введіть дату початку періоду:] і [Введіть дату кінця періоду:], щоб визначити межі діапазону значень, об'єднані операторами BETWEEN(укр. - «між») і AND (укр. - «і»);

переглянути запит у режимі таблиці.

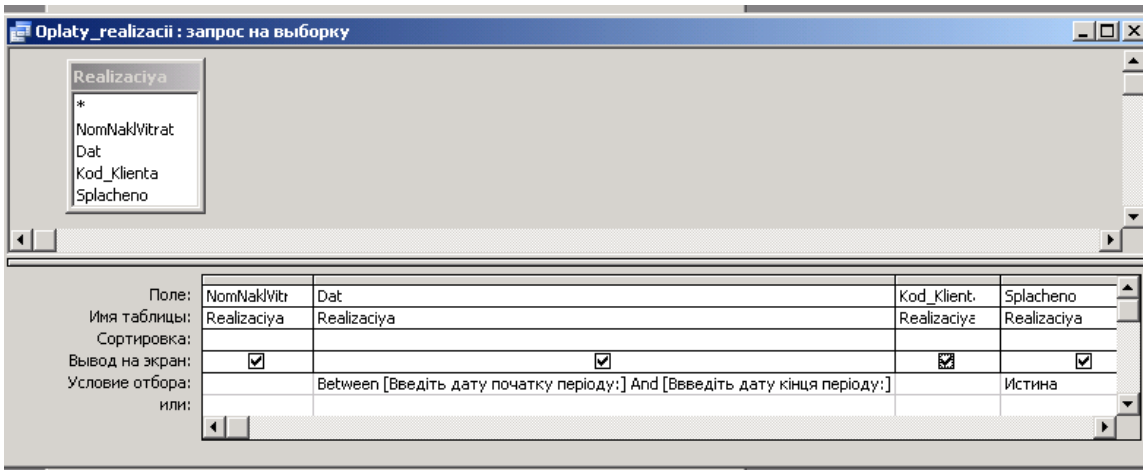


Рис. 8.10. *Створення запрошення по даті*

Таким чином, у цьому прикладі параметр для одного поля складений з двох частин за допомогою з'єднувального оператора AND.
У найпростішому запиті з одним параметром для обраного для параметру поля слід ввести за рядком *Условие отбора* вираз з текстом запрошення у квадратних дужках. Наприклад, у запиті з параметром, який має вивести записи таблиці Рух товарів із записами для товарів, ціна яких менше визначеного

рівня, у полі Ціна за рядком *Условие отбора* вводиться вираз: <[Введіть рівень цін у гривнях:].

Якщо створюється запит з декількома параметрами, для кожного поля, що буде використовуватися як параметр, слід ввести до рядку Условие отбора окремий вираз з текстом пропозиції у квадратних дужках. Наприклад, для виведення відомостей про великі партії дешевих товарів можна ускладнити попередній запит другим параметром, якщо для поля Кількість сформувати параметр: >[Введіть кількість партії товарів:].

Корисним є використання параметрів із символами підстановки. Для кожного поля, яке буде застосовуватися як параметр, слід вводити до рядка *Условие отбора* вираз з текстом запрошення у квадратних дужках.

Висновки

Завдяки функції запитів можливо реалізовувати велику частину питань, по базі даних.

Література [6,10,13-15]

Контрольні питання

1. Навіщо потрібні запити?
2. Переваги графічного інтерфейсу побудови запитів.
3. Запит на вибірку, його призначення?
4. Які елементі входять до запиту з параметром?

Питання для самостійного вивчення

1. Сортування і фільтрація даних в таблицях MS Access.
2. Створення запитів за допомогою майстра в MS Access.
3. Запит на обновлення даних в таблиці
4. Запит на додавання даних в таблицю
5. Запит на видалення даних із таблиці

Лекція 9. Методи обробки даних

Мета: Навчитися складати складні запити.

План

9.1. Підсумкові запити

9.2. Запит з обчислювальним полем.

9.3. Функції, які використовуються в запитах.

9.4. Перехресний запит.

9.1. Підсумкові запити

Щоб вибрати не окремі записи, а підсумкові значення для певної групи даних (наприклад, кількість книг по інформатиці, по кожному видавництву, кількість читачів по кафедрах тощо), треба скористатись підсумковими запитами. Таким запитам треба вказати характер групування даних і тип групової операції. Щоб конкретизувати характер групування входимо до «Груповые операции → Группировка».

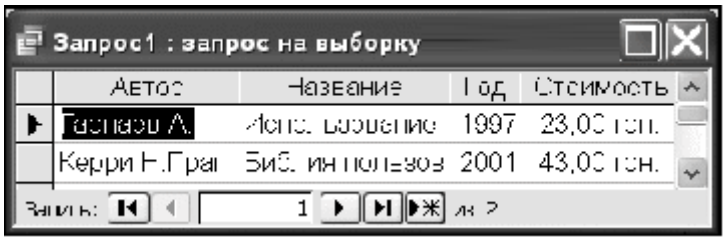


Рис. 9.1. *Підсумковий запит*

Це можна зробити так:

ввійти до режиму «Конструктора» запиту;

сформувати поля, що будуть складати запит;

встановити курсор всередину таблиці із параметрами запиту (у режимі «Конструктора») запитом;

натиснути на піктограму, що знаходиться на панелі інструментів (виглядає як значок «Σ»). Замість піктограми можна клацнути правою кнопкою миші й вибрати з контекстного меню пункт «Групповые операции».

У таблиці з параметрами запиту Access додасть рядок з іменем «Групповая операция». Щоб створити групу записів з однаковими значеннями

по деякому полю, треба в рядку «Групповая операция» цього поля ввести «Группировка». Access вибере множину записів з однаковим значенням поля групування й підрахує для них підсумки. Такі дії Access повторить для кожної групи. Спосіб рахування підсумків вказують у тому ж рядку «Групповая операция». Для цього переходимо до поля, по якому планують визначити підсумкове значення. Клацаємо лівою кнопкою миші на списку «Группировка». Список розкриється, з нього вибираємо тип операції для одержання підсумків. Список операцій має такі функції:

SUM – обчислення суми значень для групи;

AVG – середнє значення поля для даних із групи записів;

MIN – мінімальне значення для даних із групи записів;

MAX – максимальне значення для даних із групи записів;

COUNT – кількість записів, у яких є значення із групи;

STDEV – стандартне відхилення;

VAR – дисперсія;

FIRST – значення в першому записі групи;

LAST – значення в останньому записі групи.

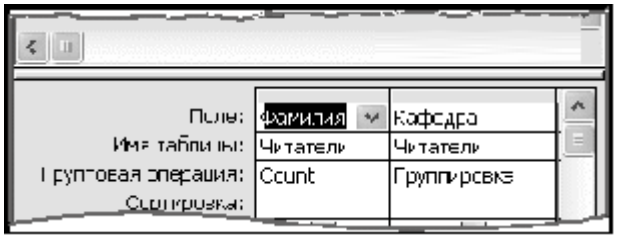


Рис. 9.2. *Операції для одержання підсумків*

Продемонструємо процес будування підсумкового звіту на такому прикладі: порахуємо кількість читачів з кожної кафедри. Спочатку будуємо звичайний запит у режимі «Конструктора», додаємо до нього таблицю «Читатели». Включаємо до результуючого звіту поля «Кафедра» і «Прізвище». Натискаємо на панелі інструментів піктограму «Групповые операции». У полі «Кафедра» вказуємо «Группировка». Переходимо до поля «Фамилия». У рядку

«группировка» для цього поля вибираємо функцію «Count» (рис. 9.2). Результат обробки запиту показано на рис. 9.3.

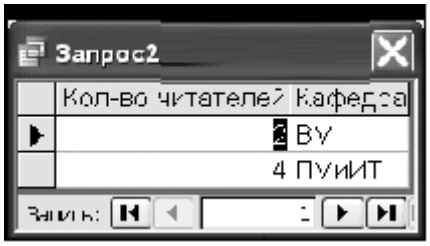


Рис. 9.3. *Результат обробки запиту*

Щоб при перегляді запиту поля мали зрозумілу назву, треба скоригувати їх властивості. Для цього ставимо курсор на колонку поля у вікні «Конструктора запитів», входимо до меню «Вид → Свойства». У полі «Подпись» друкуємо заголовок.

9.2. Запит з обчислювальним полем.

На відміну від електронних таблиць у таблицях баз даних ніколи не зберігають інформацію, яку можна одержати в результаті обчислень. Це пов’язано з тим, що здебільшого такий підхід вимагає значних витрат пам'яті й сповільнює пошук і обробку даних. Тому всі необхідні обчислення виконують у запитах, створюючи для цього спеціальні поля. Щоб створити таке поле у вільному стовпці конструктора запитів записують нове ім'я, знак “:”, а потім вираз, що обчислює необхідне значення. У виразі можна використати знаки операцій, функції, звертатися до значень інших полів. Наприклад, щоб обчислити вартість замовлення, маючи значення «СтоимостьЕдиницы» і «КоличествоЕдиниц» треба у новому стовпці записати формулу «Стоимость Заказа: [Стоимость Единицы]* [Количество Единиц]». Імена полів бази даних записують у квадратних дужках. У формулах можна використати вбудовані функції Access. Ось деякі з них:

iif(умова; вираз1; вираз2) — якщо вираз «умова» виконується , функція обчислює значення «вираз1», якщо умова не виконується — обчислює значення «вираз2». Функція iif() подібна до ЕСЛИ() в Excel.

DateDiff(“d”; дата1; дата2) — знаходить різницю між двома датами, результат представляє у днях. Якщо перший аргумент дорівнює «m», різницю

буде представлено у місяцях, якщо він дорівнює “y” — у роках. Повний перелік функцій та їх параметрів можна побачити у довідковій системі або у вікні «Построитель выражений». «Построитель виражений» — це спеціальний інструмент, що суттєво полегшує набір складних формул. Оскільки основні складові формули при використанні «Построителя…» набираються автоматично, використання цього інструменту суттєво знижує кількість помилок.

Ми проілюструємо використання обчислювальних полів на запиті, що відбирає й поєднує записи з двох таблиць: «Книги» і «ЧитКниги». Мета запиту — показати, які книги були видані читачам. У запиті створимо поле для обчислення з іменем «Пеня». У цьому полі буде нараховуватись пеня на кожну книгу, яка не булла повернута вчасно. Розмір пені дорівнює 1% від вартості книги за кожний прострочений день.

Переходимо на закладку «Запросы». Натискаємо кнопку «Создать». У вікні, що з’явиться, вибираємо варіант «Конструктор». Це означає, що параметри запиту будемо формувати в режимі «Конструктора…». Додаємо до запиту таблиці «Книги» й «ЧитКниги». Оскільки для вказаних таблиць було створено схему даних, її буде відображено у верхній частині вікна запиту. Якщо схеми даних не існує, створіть її. Це можна зробити прямо у вікні запиту. Нагадаємо, що ми маємо зв’язати таблиці за інвентарним номером книги (поле «Инв№»). Тип зв’язку таблиць «Книги» та «ЧитКниги» — «один до багатьох».

До запиту включаємо такі поля: «Автор», «Название», «Стоимость», «Инв№», «Дата выдачи», «Дата возврата», «NB». Останнє поле нам знадобиться для організації зв’язку з таблицею «Читатели». Зберігаємо запит з іменем «Пример_1». У першому вільному стовпчику нижньої частини вікна створіть поле, що обчислюється, з ім'ям «Пеня». Для цього наберіть у верхньому рядку (де розташовано ім'я поля) такий текст: «Пеня: IIf([Дата возврата]< Date();DateDiff("d";[Дата возврата];Date())*0,01* [Стоимость];0)».

Для створення виразу можна використати «Построитель виражений». Натискаємо кнопку «Построить» на панелі інструментів. Відкриється вікно

«Построителя…». У лівій частині вікна перелічені доступні об’єкти Access для будування формули: «Таблицы», «Запроси», «Формы», «Функції», «Отчеты» тощо. Якщо ліворуч від назви об’єкту стоїть позначка «+»(«плюс»), вказаний об’єкт має багато значень. Розкриваючи потрібні закладки, ви одержите доступ до окремих елементів поточної бази даних, з яких можна побудувати формулу. Наприклад, розкриваючи таблиці або звіти, ви можете дістатись до полів, що до них входять. Натискаючи кнопку «Вставить» об’єкт буде перенесено до спеціального вікна, де будується формула. Лишається розставити знаки операцій поміж об’єктами і завершити формулу. Детальне опанування «Посторителя выражений» ми лишаємо для самостійного опрацювання. Збережіть запит з іменем «Список1». Виконайте запит і перегляньте результат. Розмір пені, зрозуміло, залежить від поточної дати. Спробуйте змінити дату повернення книги. Переконайтесь, що сума пені також змінилась.

9.3. Функції, які використовуються в запитах

Можна підрахувати кількість елементів у полі (рядку значень) за допомогою функції Count. Функція Countналежить до набору функцій, які називаються агрегатними. Агрегатні функції використовуються для обчислень у стовпці даних і повертають одне значення. Крім функції Count, у Access є ще кілька агрегатних функцій, а саме:

Sum для обчислення суми стовпця чисел,

Average для обчислення середнього значення для стовпця чисел,

Maximum для пошуку найбільшого значення в полі,

Minimum для пошуку найменшого значення в полі,

Standard Deviation для обчислення того, наскільки значення відрізняються від середнього значення,

Variance для обчислення статистичної дисперсії всіх значень у стовпці.

ФУНКЦІЯ	**ОПИС**	**ВИКОРИСТОВУВАНІ ТИПИ ДАНИХ**
Sum	Обчислює суму елементів у стовпці. Використовується лише з даними в числовому та	Число, Десятковий, Грошова одиниця

	грошовому форматі.	
Average	Обчислює середнє значення для стовпця. Стовпець має містити числові, грошові дані або дані у форматі «Дата/час». Ця функція ігнорує нульові значення.	Число, Десятковий, Грошова одиниця, Дата/час
Count	Підраховує кількість елементів у стовпці.	Усі типи даних, окрім тих, які містять комплексні повторювані скалярні дані (наприклад, стовпець багатозначних списків).
Maximum	Повертає елемент із найбільшим значенням. Для текстових даних найбільше значення — це останнє значення за алфавітом, і в цьому випадку Access не враховує регістр символів. Ця функція ігнорує нульові значення.	Число, Десятковий, Грошова одиниця, Дата/час
Minimum	Повертає елемент із найменшим значенням. Для текстових даних найменше значення — це перше значення за алфавітом, і в цьому випадку Access не враховує регістр символів. Ця функція ігнорує нульові значення.	Число, Десятковий, Грошова одиниця, Дата/час
Standard Deviation	Обчислює, наскільки значення відрізняються від середнього значення.	Число, Десятковий, Грошова одиниця
Variance	Обчислює статистичну дисперсію всіх значень у стовпці. Ця функція може використовуватися лише для числових або грошових даних. Якщо таблиця містить менше двох рядків, Access повертає нульове значення..	Число, Десятковий, Грошова одиниця

9.4. Перехресний запит

Спробуємо обчислити для кожного читача кількість взятих ним книг, їх загальну вартість і пеню, що нарахована. Для цього сформуємо новий запит. Включаємо до нього таблиці «Книги», «Читатели» і запит «Пример_1». З таблиці «Читатели» до запиту включаємо поля «Фамилия», з таблиці «Книги» — поля «Инв№» і «Стоимость», із запиту «Пример_1» — поле «Пеня». На панелі інструментів вибираємо піктограму «Групповые операции», у бланку запиту з'являється рядок «Групповая операция». Для поля «Фамилия» вибираємо «Группировка», для поля «Стоимость» операцію підрахунку суми – «SUM», для поля «Инв№» — операція обчислення кількості «Count» і для поля «Пеня» — операцію «SUM». Параметри запита у режимі «Конструктора» показані на рис. 9.4. Щоб результат мав придатний вигляд, поставимо зрозумілі підписи для кожної колонки. Для цього треба ввійти у коригування властивостей полів (перейти до режиму «Конструктора...», встановити курсор на поле, натиснути праву кнопку миші й вибрати з контекстного меню «Свойства»). У переліку «Свойства» треба поставити зрозумілий текст у параметрі «Подпись».

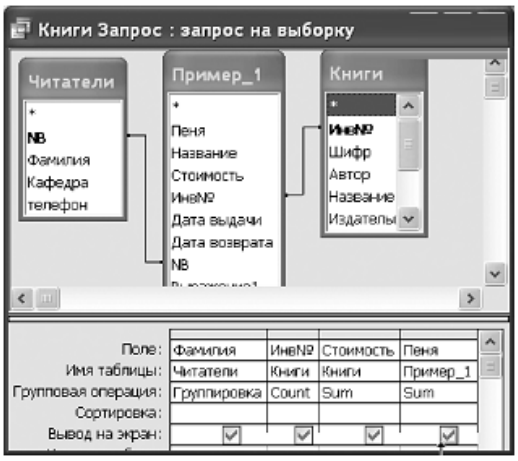


Рис. 9.4 *Параметри запиту у режимі конструктора*

Результат обробки запиту показано на рис. 9.5.

Формування запитів типу «Записи без підлеглих»

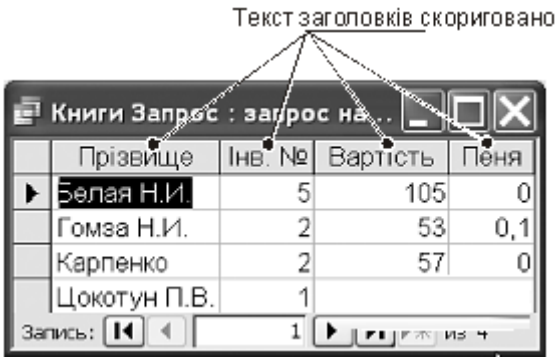


Рис. 9.5. *Результат обробки запиту.*

Ці запити відбирають записи, що не зв’язані із записами в іншому списку. Для роботи з такими запитами треба змінити тип об'єднання таблиць на «зовнішній». При «зовнішньому» зв’язуванні до запиту **обов'язково** потраплять всі записи із першої таблиці. Записи з другої таблиці будуть додані, коли значення зв'язаних полів співпадають. Якщо в другій таблиці такого запису немає, то до запису з першої таблиці додаються поля з порожніми значеннями. У конструкторі таблиць зовнішнє зв’язування зображається лінією зі стрілкою.

Для прикладу побудуємо запит, що знаходить список книг, які не видані на руки читачам. До запиту включимо дві таблиці «Книги» і «ЧитКниги». Нам знадобляться поля «Инв№», «Шифр», «Автор», «Назва» з таблиці «Книги» і поле «Инв№» з таблиці «ЧитКниги». Знайдемо ті записи, які є в таблиці «Книги» і відсутні в таблиці «ЧитКниги». Змінимо параметри об'єднання таблиць. Ставимо курсор на лінію зв’язку між таблицями, клацаємо правою кнопкою миші й обираємо з контекстного меню пункт «Параметры объединения». Із запропонованих варіантів обираємо «Объединение ВСЕХ записей из «Книги» и только тех записей из «ЧитКниги», в которых связанные поля совпадают».(рис. 21).

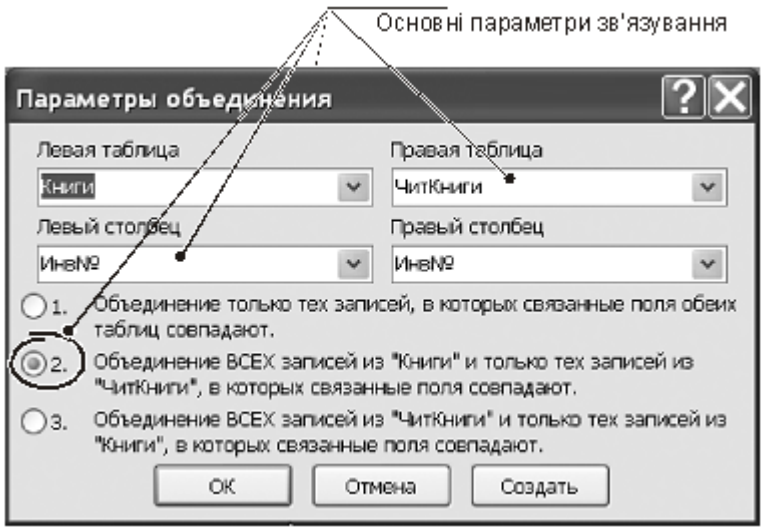


Рис. 9.6. *Параметри об'єднання даних*

Ми створили зовнішнє об'єднання, до якого включені всі записи з таблиці «Книги». Для книг, що не мають записів у таблиці «ЧитКниги» встановлюється значення «Null». Тому для поля «ИнвN» з таблиці «ЧитКниги» встановимо параметр «Условие отбора» в значення «Is Null» і знімемо прапорець «Вывод на экран». Параметри запиту показано на рис. 9.7.

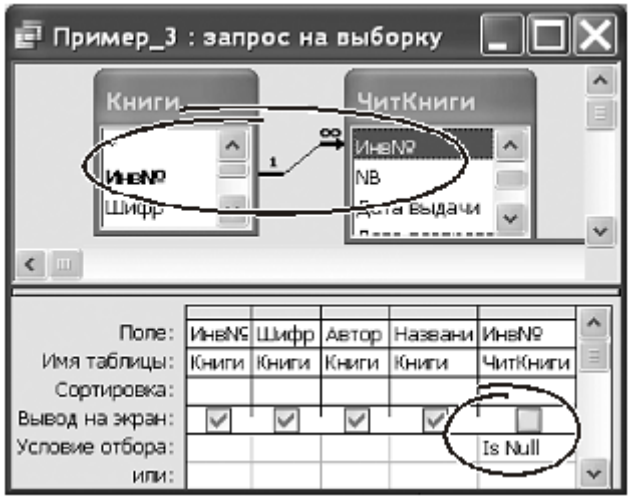


Рис. 9.7 *Параметри запиту на вибір*

Виконуємо запит, результат роботи показано на рис. 9.8.

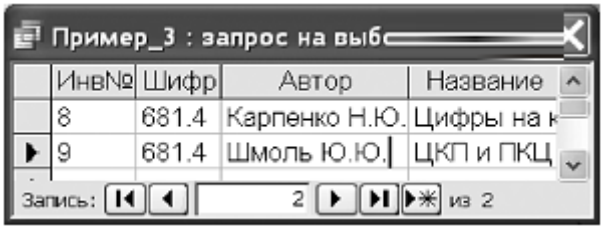


Рис. 9.8 *Результат роботи запиту*

Висновки

Використання складних запитів дозволяє виконувати одночасно з формуванням нових даних різні дії з елементами які входять до бази даних.

Література [6,13-16]

Контрольні питання

1. Що таке підсумкові запити?
2. Чим відрізняється запит з обчислювальним полем від перехресного запиту?
3. Наведіть декілька функції, які використовуються в запитах.

Питання для самостійного вивчення

1. Створення запитів за допомогою язика SQL. Оператор SELECT.
2. Оператор вводу даних INSERT
3. Оператор видалення даних DELETE
4. Оператор оновлення даних UPDATE
5. Запити на додавання та видалення таблиці

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. *Акулов О.А., Медведев Н.В.* Информатика. Базовый курс. – 2-е изд. – М.: 2005. — 552 с.
2. *Грошев А.С.* Информатика. Учебник для вузов. – А.: 2010. — 470 с.
3. *Гаврилов М.В., Климов В.А.* Информатика и информационные технологии. – М.: 2014. — 384 с.
4. *Елович И.В., Кулибаба И.В.* Информатика. – М.: 2011. — 400 с.
5. *Михеева Е.В., Титова О.И.* Информатика. – М.: 2007. — 352 с.
6. *Громов Ю.Ю., Дитрих В.Е. и др.* Информационные технологии. – Тамбов: ТГТУ, 2011. — 152 с.
7. ГОСТ 34.321–96. Информационные технологии. Система стандартов по базам данных. ЭТАЛОННАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДАННЫМИ / Межгосударственный стандарт. – Минск: Межгосударственный совет по стандартизации, метрологии и сертификации.
8. *Олифер В.Г., Олифер Н.А.* Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 4-е изд. – СПб.: Питер, 2011. – 944с.
9. *Косарев В.П.* Экономическая информатика: учебник. 3-е изд., перераб. и доп. –М.: Финансы и статистика, 2006. – 656 с.

10. *Косарев В.П., Мамонтова Е.А.* Информатика. Практикум для экономистов: учеб. пособие. – М.: Финансы и статистика; ИНФА-М. – 2009. – 544 с.
11. *Информатика для экономистов: Учебник* /Под общ. редакцией В.М. Матюшка. – М.: ИНФРА-М, 2007. – 880 с. – (Учебники РУДН).
12. *Симонович С.В.* Информатика. Базовый курс: Учебник для вузов. 3-е изд. – СПб.: Питер, 2012. – 640с.
13. *Макарова Н.В., Волков В.Б.* Информатика: Учебник для вузов. – СПб.: Питер, 2011. – 576с.
14. *Хомоненко А. Д., Цыганков В. М., Мальцев М. Г.* Базы данных. Учебник. 6-е издание. – М.: Бином. – 2007. – 736 с.
15. *Кузнецов С.Д.* Базы данных: языки и модели. Учебник. – М.: Бином. – 2008.
– 720 с.
16. *Пирогов В.Ю.* Информационные системы и базы данных: организация и проектирование: учеб. пособие. – СПб.: БХВ-Петербург, 2009. – 528 с.

Конспект лекцій з нормативної навчальної дисципліни циклу
природничо-наукової підготовки «Інформаційні системи та технології»
для студентів всіх форм навчання
напряму підготовки 6.030601 «Менеджмент»

Укладач: Лесіна Євгенія Вікторівна

Підписано до друку
Умовних друкованих аркушів 7,1
Тираж – 20 примірників

Видавничий центр Красноармійського індустріального інституту
ДВНЗ «ДонНТУ»,
85300, Красноармійськ, площа Шибанкова, 2