|  |
| --- |
|  |
| **Зміст**  Введення  1. Створення таблиць  2. Створення запитів  3. Створення звітів  4. Створення форм  5. Створення макросів  Висновок  Список літератури  Введення  Проектована база даних призначена для працівників приватного підприємства, який надає населенню послуги таксі, і дозволяє їм підвищити ефективність своєї роботи за рахунок систематизації і швидкого пошуку потрібної їм інформації. Це сильно спрощує роботу, тому відпадає необхідність переглядати гори паперів в пошуках потрібної інформації, розраховувати вартість подорожнього листа кожному водію та звіти про роботі начальству.  В БД повинні зберігатися відомості про водіїв: ПІБ, позивний, номер водійського посвідчення, дата видачі водійського посвідчення, а також знаходиться водій у відпустці, на лікарняному або звільнений; відомості про автомобілі: державний номер, марка, колір, рік випуску, № свідоцтва про реєстрацію транспортного засобу, власник, а також знаходиться автомобіль в ремонті; відомості про колійних аркушах:  № подорожнього листа, позивний водія, дати відкриття та закриття колійного аркуша, державний номер автомобіля; відомості про замовлення: номер замовлення, дата і час надходження, адреса клієнта, водій виконує замовлення, відмітка про виконанні.  Диспетчер таксі повинен мати можливість отримати такі відомості:  - які автомобілі знаходяться на лінії, а які в ремонті;  - відомості про замовлення з розбивкою по водіям;  - відомості про кількість відпрацьованих годин кожним водієм і, відповідно, вартості його путівки (Вартість путівки розраховується за 20 руб. За годину роботи, інші грошові кошти залишаються водієві).  Для створення бази даних використовуємо СУБД Microsoft Access. Створення будь-якої бази даних починається з вибору структури бази даних. У нашому випадку зручніше використовувати п'ять таблиць з даними. Це будуть дві таблиці з даними по замовленнях: внутрішньоміські і міжміські, таблиця даних по водіях, таблиця даних на автомобілі і таблиця обліку подорожніх листів.  Далі зробимо кілька запитів на вибірку за різними параметрами, а також звіти до них. Наприклад, зробимо запит, видає відомості про водія і автомобілях, що знаходяться на лінії; запит, обчислює час роботи за подорожнім листом, а також його вартість та ін  Для зручності роботи з даними створимо кілька форм і кнопки переходів між ними. Для виклику форми (І не тільки форми) можна також скористатися макросом. Для демонстрації даної можливості створимо макрос виклику головної форми і форми введення відомостей про автомобілі. Також створимо макрос роздруківки вартості шляхових листів.  1. Створення таблиць  Отже, створюємо таблицю. Всі бази даних містять одну або кілька таблиць. Таблиця - сукупність записів. Стовпці в таблиці називаються полями, а рядки - записами. Кількість записів в таблиці обмежується ємністю жорсткого диска. Допустима кількість полів в запису - 255. У кожної таблиці є унікальне ім'я. У базі даних повинна бути представлена вЂ‹вЂ‹окрема таблиця для кожного великого набору елементів, наприклад, для записів про співробітників, про замовлення клієнтів, про способи постачання і постачальниках. Не слід дублювати дані в кількох таблицях. Це поширена помилка, якої легко уникнути, правильно розробивши структуру бази даних. Створювати таблиці можна трьома способами:  - за допомогою майстра (Якщо при створенні таблиці потрібні підказки про елементи, які слід у неї включати);  - в режимі конструктора (якщо ви точно знаєте, які параметри полів вам потрібні). На відміну від майстра, цей спосіб не припускає покрокового процесу і вимагає більше зусиль по налагодженню таблиці;  - допомогою введення даних і визначення полів у режимі таблиці. Цей спосіб застосовується при необхідності налаштувати просту таблицю і швидко ввести дані. Однак в відміну від роботи з майстром або в режимі конструктора при цьому не можна задати багато параметри таблиці.  Створюємо таблицю В«ВодіїВ». Для створення таблиці використовуємо конструктор, як більш докладний метод (Малюнок 1). У таблиці задамо наступні поля: В«ПозивнийВ», В«ПрізвищеВ», В«Ім'яВ», В«По батьковіВ», В«Номер ВУВ», В«Дата видачі ВУВ», В«На лікарняномуВ», В«У відпустціВ», В«ЗвільненийВ». Поле В«ПозивнийВ» має формат лічильника. Наступні поля - текстові, В«Дата видачі ВУВ» має формат дати/часу, а останні три поля - логічні.  Рисунок 1 - Вікно створення таблиці  Малюнок 2 - Введення параметрів поля таблиці  Вводимо назви полів і їх докладний формат (малюнки 2, 3).  Рисунок 3 - вибір типу даних у полі таблиці  Зберігаємо структуру таблиці під ім'ям В«ВодіїВ» і починаємо заповнювати значення полів - записи (рисунок 4).  Малюнок 4 - Заповнення таблиці даними  Запис - Повний набір даних про певний об'єкт: клієнті, книзі, подію і т. п. Запис відображається як рядок таблиці. Кожна запис складається з окремих полів. Кожен запис в кожній таблиці повинна бути унікальна. Щоб відрізнити один запис від іншого, в таблиці можна додати поле первинного ключа. Ключовим полем стане полі В«ПозивнийВ», тому що позивний унікальний для кожного водія. Це поле має формат лічильника. Після завершення введення даних таблиця закривається.  Після створення всіх таблиць, необхідно побудувати схему даних (рис.5).  Рисунок 5 - Схема даних  2. Створення запитів  Переходимо на наступну вкладку - В«ЗапитиВ». Запити - об'єкти, що дозволяють користувачеві отримати потрібні дані з однієї або декількох таблиць. Для створення запиту можна використовувати бланк QBE (Query By Example - запит по зразку) або написати інструкцію SQL. Можна створювати запити на вибірку, оновлення, видалення або додавання даних. За допомогою запитів можна також створювати нові таблиці, використовуючи дані з однієї або декількох таблиць.  Будувати запити також можна за допомогою майстра і за допомогою конструктора. Знову вибираємо конструктор (рисунок 6).  Малюнок 6 - Вікно створення запиту  Для побудови запитів потрібно додати таблиці, за якими буде виконуватися запит. У нашому разі п'ять таблиць, виберемо з них ті, в яких містяться необхідні для запиту дані (малюнок 7).  Малюнок 7 - Вікно додавання таблиці  Після додавання таблиці необхідно вибрати поля, за якими робиться запит, і задати необхідні умови відбору (малюнок 8).  Рисунок 8 - Завдання умов відбору  Наприклад, потрібно, щоб на екран виводився список міжміських замовлень для конкретного водія.  Для створення запитів, коли потрібно подивитися дані для певного водія в стовпці В«ПозивнийВ» можна задати умову для введення позивного водія в квадратних дужках В«[Задайте позивний водія] В». У цьому випадку при відкритті запиту програма попросить ввести потрібний позивний (малюнок 9).  Рисунок 9 - Запуск запиту В«Міжміські замовлення по водіямВ»  Створення запиту В«На лініїВ»:  Створення запиту В«Автомобілі в ремонті В»:  Створення запиту В«Вартість подорожнього листа В»:  3. Створення звітів  Звіт - це гнучке і ефективний засіб для організації даних при виведенні на друк і разом з тим це спосіб виведення даних з ...бази на друк в тому вигляді, в якому вимагається користувачеві. За допомогою звіту можна розташувати інформацію на аркуші в зручному для користувача вигляді з різним оформленням. У звіті дані перетворюються в форматований документ. Конструювання звіту не тільки дозволяє вивести інформацію в зручному і привабливому вигляді, але і комбінувати дані декількох таблиць, а також упорядковувати записи таблиці на підставі даних інших таблиць. При його створенні ви розставляєте спеціальні маркери (звані елементами управління).  За допомогою звітів можна групувати дані, виконувати з ними різні дії, додавати заголовки і інші елементи форматування з метою зробити відображення даних більш зрозумілим.  Створивши звіт, можна зберегти дані в цьому форматі, при цьому кожен раз при друці його структура буде зберігатися, навіть якщо дані були змінені.  Створимо звіти по запитам В«На лініїВ», В«Вартість подорожнього листаВ», В«Автомобілі у власності підприємства В», а також по таблиціВ« Замовлення В»з угрупуванням по водіям.  Створюємо звіт за допомогою майстра звітів, а потім в режимі конструктора відкоригуємо розташування елементів, для більш прийнятного виду і оформлення (малюнки 10, 11).  Рисунок 10 - Вибір елементів, що відображаються у звіті  Малюнок 11 - Звіт в режимі конструктора  4. Створення форм  Форма - об'єкт, призначений в основному для вводу даних, відображення їх на екрані або управління роботою додатка. Зазвичай в цьому режимі виводяться дані з записів, а текстові поля, кнопки, картинки та інші службові об'єкти полегшують перегляд та обробку даних. Інша перевага форм перед режимом таблиці - можливість відображати в одній формі дані з декількох таблиць.  Форми забезпечують найбільш гнучкий спосіб введення, редагування, перегляду і видалення даних і фактично є шаблонами, керуючими відображенням інформації. Форма дозволяє відображати одночасно всі поля однієї або декількох записів. Оптимально побудована форма може вміщати декілька десятків полів на одному екрані, а якщо полів набагато більше, то для кожного запису можна створити багатосторінкову форму. Можна створити форму-меню для виклику інших форм, таблиць, запитів чи звітів. У формі кожне поле можна розмістити в точно заданому місці, вибрати для нього колір або заливку і додати елементи управління текстом для ефективного введення даних.  Для створення форми можна також скористатися конструктором або майстром створення форм. Т.к. конструктор все ж дає більш широкі можливості, скористаємося ним в черговий раз. Створимо, по-перше, титульний аркуш у вигляді форми і зв'яжемо його з головної формою, призначеною для роботи з додатком. За допомогою конструктора і панелі керуючих елементів додамо на заготівлю форми В«ВодіїВ» всі необхідні дані з таблиці В«ВодіїВ», а також, щоб передбачити додавання нових записів в таблицю, кнопки В«Додати записВ», В«Зберегти запис В»іВ« Видалити запис В». Цих кнопок цілком достатньо для того, щоб редагувати записи в таблиці В«ВодіїВ». Крім того, додамо кнопку для пошуку потрібного запису (рисунок 12, 13).  Рисунок 12 - Форма В«ВодіїВ»  Рисунок 13 - Створення кнопок на формі  Звіти попередньо можна створити за допомогою майстра або конструктора на вкладці В«ЗвітиВ», а потім аналогічно створити зручну форму для перегляду звітів (малюнок 14), можна також створити форму для перегляду самих запитів, але звіти дивитися таки зручніше, тим більше, що їх можна відразу і роздрукувати при бажанні за допомогою кнопки В«ДрукВ» на панелі інструментів. Можна було б в принципі створити і на формі кнопку друку звітів, але, по-моєму, ні до чого дуже вже перевантажувати форму кнопками, адже кнопка В«ДрукВ» і так доступна в даному вікні.  Рисунок 14 - Форма В«ЗвітиВ»  Після того, як форма для роботи з базою даних створена можна задати автозапуск форми при відкритті бази даних для зручності роботи. Виконується це в такий спосіб: Сервіс в†’ Параметри запуску в†’ Висновок форми/сторінки в†’ ОК. (Малюнок 15, 16)  Рисунок 15 - Вибір параметрів запуску бази даних  Рисунок 16 - Вікно параметрів запуску  5. Створення макросів  Макрос в Access являє собою структуру, що складається з однієї або декількох макрокоманд, які виконуються або послідовно, або в порядку, заданому певними умовами. Набір макрокоманд в Access дуже широкий, за допомогою макросів можна реалізувати багато. Кожна макрокоманда має певне ім'я і, можливо, один або кілька аргументів, які задаються користувачем. Наприклад, при використанні макрокоманди Відкрити Форму (OpenForm) в якості аргументів необхідно задати, по крайней мірою, ім'я відкривається форми і режим виведення її на екран. Існує ще один дуже простий спосіб введення макрокоманди, яким ми і скористаємося. Припустимо, потрібно створити макрос, який повинен відкривати форму В«ГлавнаяВ» (Customers) в додатку В«Таксі ФортунаВ» (Northwind). Для цього у вікні База даних (Database) необхідно клацнути по ярлику Форми (Forms), потім, використовуючи панель інструментів, відкрити вікно Конструктор макросів (Macro Design) і перетягнути з допомогою миші форму В«ГлавнаяВ» (Customers) в полі В«МакрокомандаВ» (Action) вікна Конструктора макросів.  Для зручності перетягування об'єктів з вікна База даних (Database) у форму можна вибрати команду Вікно, Зліва направо (Window, Tile Vertically), щоб розташувати вікна на екрані поруч.  Тоді в поле В«МакрокомандаВ» (Action) автоматично з'явиться макрокоманда Відкрити Форму (OpenForm) (рисунок 17), а на панелі аргументів автоматично заповняться поля: в полі В«Ім'я формиВ» (Form Name) з'явиться ім'я форми В«ГоловнаВ» (Customers), в полі В«РежимВ» (View) - значення за замовчуванням Форма (Form), у поле В«Режим вікнаВ» (Window Mode) - значення Звичайний (Normal). Інші поля залишаться порожніми.  Рисунок 17 - Створення макросу  Крім того, можна зробити записи в цій формі недоступними для змін, для чого в полі Режим даних (Data Mode) необхідно ввести значення В«тільки читанняВ» (Read Only).  Аналогічно можна створювати макрокоманди, перетягуючи з вікна База даних (Database) таблиці, запити, звіти та інші макроси. При цьому у вікні Конструктора макросів будуть з'являтися відповідні макрокоманди: Відкрити Таблицю (ОрепТаblе), Відкрити Запит (OpenQuery), Відкрити Звіт (OpenReport), запуск макросу (RunMacro).  Отже, база даних Таксі створена. База дуже проста у використанні і зрозуміла для будь-яких користувачів.  Висновок  Сучасні технології розробки прикладних програм роблять побудова додатків фантастично дешевим і швидким. Кваліфікований користувач за допомогою Microsoft Access сьогодні може за один вечір створити на персональному комп'ютері те, що на ранніх ЕОМ вимагало місяців роботи (якщо це взагалі було можливим). Крім того, тепер стало значно легше знаходити помилки, усувати їх і змінювати проект безпосередньо в процесі створення додатку.  Сучасні технології дозволяють створювати дуже складні додатки. До того ж швидкість обчислень по порівнянні навіть з попереднім десятиріччям зросла на кілька порядків. Однак, незважаючи на потужність засобів розробки, якщо не витратити значних зусиль на визначення завдань і принципів роботи програми, то згодом доведеться втратити значно більше часу на всілякі переробки. Якщо проект додатка недостатньо продуманий, то додавання нових функцій або усунення недоліків пов'язано з великими тимчасовими і фінансовими витратами.  Основні етапи розробки:  1. Уточнення завдань.  2. Послідовність виконання завдань.  3. Аналіз даних.  4. Визначення структури даних.  5. Розробка макета додатки і користувальницького інтерфейсу.  6. Створення додатка.  7. Тестування і удосконалення. **Створення програми "Автоматизоване робоче місце диспетчера таксі"** Основна зв'язок яка у програмі - це зв'язку з базою даних із засобам технологіїADO. Основне перевагу як і зв'язку у цьому, що за зв'язок встановлену вADO Connection засобамиADOQuery іADOTable, можна вносити зміни у базі даних, і реально отримувати з її необхідні дані, а як і проводити порівняння даних, і інші операції. Прикладом цього є перевірка логіна і пароля компонентомАDOQuery:  У властивостіSQL додаємо рядок: P.SELECT \*FROMadminsWHERELogin=:Login, і прописуємо код:  >IfCombobox1.Text='Администратор'thenbegin  >ADOQuery2.Active: =>False;  >ADOQuery2.Close;  >ADOQuery2.Parameters.Items [0]. Value: =>Edit1.Text;  >ADOQuery2. Open;  >if (>notADOQuery2.IsEmpty) and (>Edit2.Text=ADOQuery2.FieldByName ('>Password').AsString)then  >begin  >Form2.show;  >Form1.Visible: =>False;  end  >else  >begin  >showmessage (>Неверний ім'я користувача чи пароль)  end;  end;  Зв'язок із базою здійснюється за допомогою багатоступінчастої зв'язку. Багатоступінчаста зв'язку з базою технологієюADO виглядає так:  >КомпонентADOConnection встановлює зв'язок докладання з конкретною базою, адресу якої прописується в рядку ConnectionString, це щабель. До компонентуADOConnection приєднуються чи кілька компонентівADOTable іADOQuery, це кількості атрибутів бази, потрібно відобразити при застосуванні з наступним внесенням змін - у них. Це компоненти другого ступеня.КомпонентADOTable встановлює зв'язку з конкретної таблицею,подсоединенной бази.КомпонентADOQuery необхідний створення запитів, порівняння даних у різних атрибутів однієї бази, або вставка, вибірка та інші дії реляційної алгебри. Третьої щаблем є компоненти з вкладкиdataAccess, зокремаDataSource, як більше часто використовуваний під час створення додатків. Цей компонент свого роду служитьпереходником від компонентів другого ступеня до компонентами четвертого ступеня. Компоненти четвертої системи - це компоненти які відображатимуть вміст бази на формі докладання. Вони яскраві і називаються як компоненти перебувають увкладке Standart, тільки з приставний DB, що з їхньої взаємозв'язок з базою. Це компоненти як:DBEdit,DBGrid іDBCombobox. Крім відображення вмісту атрибутів бази даних, з допомогою можна вставляти, редагувати, видаляти дані з убозівської бази. І тому необхідно компонентADOTable вводити на станInsert чиEdit.  Для друку звітів використовується технологія роботи з Microsoft Word через Delphi. Завдяки встановленню цьому разі можна імпортувати дані з таблиць тестової бази даних як запиту, як і використовується під час упорядкування звітів в режимах ">Администратора" і "Диспетчери". Використовуючи властивістьRecordCount компонентаADOTable, задаємо зміну циклу з параметрів і виконуємо порівняння кожного запису з вимогою, разі результату дані імпортуються в Word.  Крім компонентів через відкликання базою даних використовувалися компоненти:  >Button: Усі кнопки виконують дії закриття, занесення даних до бази, перехід між формами.  >Label: Усі написи, які є підказками чипоявдляющейся інформацією, перевагу - користувач неспроможна змінити змістLabel.  >MainMenu: до створення спливаючого меню вгорі вікна, виконує перемикання між вікнами і дублює дії деяких кнопок.   **Укладання** Ця програма варта автоматизації робочого місця диспетчера таксі, із єдиною метою зменшити частку ручної праці і кількість паперових документів і майже наочно демонструвати зайнятість водіїв.  Програмний продукт " Автоматизоване робоче місце диспетчера таксі" виконує поставлені завдання:  Реєстрація нових водіїв і диспетчерів як адміністратора;  печатку звітів по диспетчерам чи з конкретному водієві;  Формування "Чорного списку" адміністратором.  У режимі диспетчера перед входом робочому місці з'являється ім'я та прізвище працюючого диспетчера.  Прийом замовлень  Реєстрація водіїв що прибули зміну.  Диспетчер зможе закінчити зміну доки надрукує звіт.  Недоліки: Дуже великій ролі грає "Людський чинник" при халатному віданні бази даних замовлень та введеннянеккоректних даних можуть забити базу назвами не існуючих вулиць, іменами водіїв і диспетчерів типу ">ивпрапрваор" тощо.  **Список літератури**  1. Бекаревич Ю., Пушкіна Н. Microsoft Access за 21 заняття. - М.: Олма-Пресс, 2006. - 544с.  2. Лорі Ульріх Фуллер, Кен Кук, Джон Кауфельд. Microsoft Office Access 2007 для "чайників". - М.: Вільямс, 2007. - 384с.  3. Міхеєва В., Харитонова І. Microsoft Access 2003. - М.: Нова, 2005. - 1072с.  4. Напівпровідникові прилади: діоди випрямні, тиристори, оптоелектронні прилади: довідник/під заг. ред. М.М. Горюнова. - 2-ге вид., Перероб. - М.: Вища школа, 1987. - 743с.  5. Хомоненко А., Грідін В. В. Microsoft Access. Швидкий старт. - М.: Фінанси і статистика, 2008. - 304с.  6. Компакт-диск В«Довідник з напівпровідниковим приладам. На допомогу радіоаматори В».  <="" div="" style="width: 600px; height: 30px;"> |