

Зміст

Анотація.....	3
Annotation.....	4
Аннотация.....	5
Вступ.....	6
Теоретична частина.....	7
Загальні відомості по базам даних.....	7
Основні поняття.....	7
Історія розвитку.....	9
Організація баз даних.....	10
Характеристика БД.....	10
Реалізації баз даних.....	11
Архітектура баз даних.....	11
Таблиці баз даних.....	13
Ключі та індекси.....	14
Зв'язки між таблицями.....	15
Нормалізація реляційної бази даних. Перша нормальна форма. Друга нормальна форма.	
Третя нормальна форма.....	17
База даних «Оптовий магазин».....	18
Загальні відомості по SQL.....	20
Реляційний спосіб доступу до даних.....	20
Основні відомості про мову SQL.....	20
Функції мови SQL.....	22
Визначення даних.....	23
Зміна складу полів таблиці.....	25
Створення та видалення індексу.....	25
Опис оператора SELECT.....	26
Оператори керування полями.....	28
Проста умова відбору записів.....	29
Складні умови відбору записів.....	30
Практична частина (опис роботи програми).....	31
Висновки.....	37
Перелік використаної літератури.....	38
Додатки.....	39

Анотація

Виконання даної курсової роботи передбачає створення бази даних та виконання маніпуляцій з створеною базою. Необхідно створити додаток для користувача бази даних, який за вказівками користувача буде виконувати всі необхідні операції. Додаток не повинен бути громіздким та має мати зручний інтерфейс.

Додаток орієнтований на використання в оптовому магазині, тому має бути врахована специфіка роботи і ведення обліку в торговій сфері. Структура баз даних також має відповідати певним критеріям ведення торгової діяльності. В таблицях повинна міститися інформація про товари на складі (номенклатура), категорії товарів, контрагентів з якими співпрацює оптовий магазин, укладені контракти та договори, контактну інформацію, реквізити контрагентів та відомості по кількості проданого та закупленого товару.

В звіті до курсової роботи міститься загальна інформація про бази даних, ведення баз даних, створення та роботу з ними. А також основна інформація про спеціалізовану мову SQL і перелік основних операндів та алгоритмічних конструкцій мови, що застосовані в курсовій роботі. Наведені приклади з програми (програмного додатку) по роботі з SQL-запитами. Описана загальна структура створеної бази даних та програмного додатку. Наведені приклади роботи програми.

Annotation

Usage of this course work deals with creation of database and performing actions with it. It is necessary to make an application for a user of the database which according to his command will perform all the necessary operations. The application should be convenient for the work with a suitable interface.

The application is oriented on its using in a wholesale store, so the specifics of work and stock-taking in the trade sphere should be taken into account. The structure of database should also meet the demands of certain criteria of carrying on trade business. The information about goods in the warehouse (items), categories of items/goods as well as counterparts with which the wholesale store cooperates, contracts and agreements signed, contact information, counterparts requisites and records on the goods sold and bought — all these should be marked in the tables.

In the course work summary there is common information on the databases and their support, creation and work with databases. There also is the main information on the specialized language SQL and the list of basic/main operands and algorithmic structures of the language which were used in the course work. There are some examples from the programme (programme application) on the dealing with SQL-query. The common structure of the created database and the programme application are both described. There are some examples of the programme in operation.

Аннотация

Исполнение данной курсовой работы предусматривает создание базы данных и выполнение манипуляций с созданной базой данных. Необходимо создать приложение для пользователя базы данных, которое по указаниям пользователя будет исполнять все необходимые операции. Приложение не должно быть громоздким и должно иметь удобный интерфейс.

Приложение ориентировано на использование в оптовом магазине, потому должна быть учтена специфика работы и ведения учета в торговой сфере. Структура баз данных тоже должна отвечать определенным критериям ведения торговой деятельности. В таблицах должна быть информация о товаре на складе (номенклатура), категории товаров, контрагентов с которыми сотрудничает оптовый магазин, составленные контракты и договора, контактную информацию, реквизиты контрагентов, а также ведомости о количестве проданного и закупленного товара.

В отчете к курсовой работе имеется общая информация по базам данных, ведение баз данных, создание и работу с ними. А также основная информация про специализированный язык SQL и перечень основных операндов и алгоритмических структур языка, которые использованы в курсовой работе. Приведены примеры из программы (программного приложения) по работе с SQL-запросами. Описана общая структура созданной базы данных и программного приложения. Приведены примеры работы программы.

Вступ

Інформаційні системи здавна знаходять досить широке застосування в життєдіяльності людства. Це пов'язано з тим, що для існування цивілізації необхідним є обмін інформацією — передача знань, як між окремими членами і колективами суспільства, так і між різними поколіннями.

Найдавнішими і найпоширенішими інформаційними системами можна вважати бібліотеки. І, дійсно, здавна в бібліотеках збирають книжки, зберігають їх, дотримуючись певних правил, створюють каталоги різного призначення для полегшення доступу до книжкового фонду. Видаються спеціальні журнали та довідники, що інформують про нові надходження, ведеться облік видачі.

На великому сучасному підприємстві в тому чи іншому вигляді повинна існувати інформаційна система. Для обробки всіх даних пов'язаних з роботою підприємства потрібні певні організаційні і технічні засоби, тобто ІС.

Сьогодні чи не найбільш актуальними є автоматизовані інформаційні системи, що застосовуються у торговельних закладах — магазинах, кіосках, дрібних комерційних фірмах, тощо. Основною задачею, яку доводиться розв'язувати співробітникам таких підприємств, є облік товарів та матеріалів. При здійсненні обліку особливо важливо забезпечити цілісність та непротиворічність даних.

У даній курсовій роботі ставиться задача розробки автоматизованої інформаційної системи обліку реалізації товарів в оптовому магазині. Для реалізації інформаційної системи застосоване програмне середовище Delphi а також засоби керування базами даних.

Звіт до курсової роботи ділиться на три основні частини: теоретичну, практичну та додатки. В теоретичній частині висвітлені питання по базам даних в цілому та по структурованій мові запитів SQL. В практичній частині наведений опис роботи програми з прикладами. В розділі додатків знаходиться текст всіх модулів програми.

Теоретична частина

Загальні відомості по базам даних

Основні поняття

При найбільш загальному підході інформаційну систему (ІС) можна визначити як сукупність організаційних і технічних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів.

Сьогодні чи не найбільш актуальними є автоматизовані ІС, що застосовуються у торговельних закладах — магазинах, кіосках, дрібних комерційних фірмах, тощо. Основною задачею, яку доводиться розв'язувати співробітникам таких підприємств, є облік товарів та матеріалів. При здійсненні обліку особливо важливо забезпечити цілісність та непротиворічність даних.

На сучасному етапі такі задачі розв'язуються за допомогою систем керування базами даних, а розробка інтерфейсу користувача робиться за допомогою засобів швидкої розробки програм. Прикладом такої системи є Borland Delphi фірми Inprise. Спираючись на об'єктно-орієнтовану мову програмування Object Pascal при використанні бібліотеки візуальних компонентів VCL, це середовище розробки дозволяє швидко створювати прикладні програми для розв'язання найрізноманітніших задач. Особливу цінність мають можливості Delphi щодо роботи з базами даних. Фірма Borland розробила формат зберігання баз даних Paradox, який є стандартним при розробці систем баз даних за допомогою Delphi.

Інформаційна технологія (ІТ) — процес, що використовує сукупність засобів і методів збору, обробки і передачі даних (первинної інформації) для отримання інформації нової якості про стан об'єкта, процесу або явища.

ІС здавна знаходять (в тому чи іншому вигляді) досить широке застосування в життєдіяльності людства. Це пов'язано з тим, що для існування цивілізації необхідним є обмін інформацією — передача знань, як між окремими членами і колективами суспільства, так і між різними поколіннями.

Для успішного функціонування різних організацій необхідна наявність розвинутої інформаційної системи, яка реалізує автоматичний збір, обробку і монтажування даних.

Сучасною формою ІС є банки даних, в склад яких входять:

1. Обчислювальна система
2. СУБД
3. Одна або декілька БД
4. Набір прикладних програм

Бази даних забезпечують зберігання інформації, а також швидкий та зручний доступ до даних. Бази даних являють собою сукупність даних різного характеру, організованих за певними правилами. Інформація в БД повинна бути:

- не суперечливою
- цілісною
- не надлишковою

СУБД являє собою сукупність мовних і програмних засобів, призначених для створення, ведення і використання БД. За характером використання СУБД поділяються на персональні і багатокористувацькі. Персональна СУБД забезпечує можливість створення локальних баз даних, які працюють на одному комп'ютері. До персональних СУБД належать FoxPro, Access, Paradox. Багатокористувацькі СУБД дозволяють створювати системи, які працюють в архітектурі «клієнт-сервер». До них відносять Oracle, InterBase, SyBase, Informix.

Мовні засоби сучасних СУБД включають:

- мову опису даних, яка визначає логічну структуру таблиці;
- мову маніпулювання даними, яка забезпечує виконання основних операцій над даними;
- структуровану мову запитів (SQL), що забезпечує управління структурою БД і маніпулювання даними, а також стандартні засоби доступу;

- мова запитів за зразком (QBE), яка забезпечує візуальне конструювання запитів до БД.

Історія розвитку

- 1960-ті рр. розробка перших БД. CODASYL — мережева модель даних та одночасно незалежна розробка ієрархічної БД фірмою North American Rockwell, котра пізніше взята за основу IMS — власної розробки IBM.
- 1970-ті рр. наукове обґрунтування Едгаром Ф. Коддом основ реляційної моделі, котра на початку зацікавила лише наукові кола. Вперше цю модель було використано у БД Ingres (Берклі) та System R (IBM), що були лише дослідними прототипами анонсованими протягом 1976 року.
- 1980-ті рр. поява перших комерційних версій реляційних БД Oracle та DB2. Реляційні БД починають успішно витіснити мережеві та ієрархічні. Дослідження децентралізованих (розподілених) систем БД, проте вони не відіграють особливої ролі на ринку БД.
- 1990-ті рр. увага науковців зміщується у сторону об'єктно-орієнтованих БД, які знайшли застосування у першу чергу в тих областях де використовуються комплексні дані: інженерні, мультимедійні БД.
- 2000-ні рр. головним нововведенням є підтримка та застосування XML у БД. Розробники комерційних БД котрі панували на ринку у 1990-их рр. отримують все більшу конкуренцію зі сторони руху відкритого програмного забезпечення. Реакцією на це стає поява безкоштовних версій комерційних БД.

Організація баз даних

Структуровані БД використовують структури даних, тобто структурований опис типу фактів за допомогою схеми даних, більш відомої як модель даних. Модель даних описує об'єкти та взаємовідносини між ними. Існує декілька моделей (чи типів) баз даних, основні:

- ієрархічну;
- мережеву;
- реляційну;
- об'єктно-орієнтовану.

Реляційна база даних була запропонована в 70-х роках співробітником фірми IBM Едгаром Кодом. Реляційна база даних являє собою сокупність таблиць зв'язаних певним відношенням. Переваги: гнучкість структури, зручність реалізації на ЕОМ.

Приблизно з 2000 року більше половини БД використовують реляційну модель.

Характеристика БД

Часто зустрічається характеристика БД на основі певних параметрів або необхідних вимог, наприклад:

- значна кількість даних
- незалежність даних
- відкритий доступ до даних
- підтримка транзакцій з гарантією відповідних властивостей
- гарантована відсутність збоїв
- одночасна робота з багатьма користувачами

З подальшим розвитком БД змінюються й ці вимоги та додаються нові, тому однотайності щодо повноти цієї характеристики немає.

Реалізації баз даних

Комерційні:

- DB2
- Informix
- Oracle
- SQL Server

З відкритим кодом:

- MySQL
- Firebird
- PostgreSQL

Архітектура баз даних

В залежності від розташування таблиці і одатків, що їх обробляють розрізняють локальні та віддалені бази даних. Додатки Delphi здійснюють доступ до баз даних через процесор баз даних BDE. BDE представляє собою набір драйверів, які забезпечують доступ до даних. Локальні бази даних розташовані на тому ж комп'ютері, що і додатки, що їх опрацьовують.

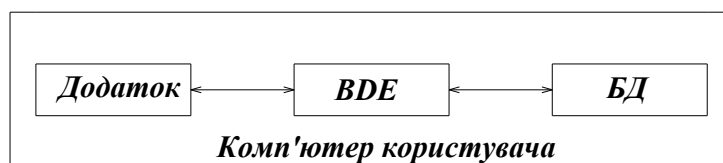


Рис. 1: Схема локальних БД

Переважно робота з локальними БД виконується в однокористувацькому режимі, але при необхідності можна запустити ще один додаток, який буде

здійснювати до тих же даних. В цьому випадку для управління спільним доступом необхідні спеціальні засоби контролю та захисту.

При використанні локальної бази даних в мережі можлива організація багатокористувацького доступу. Такий варіант відповідає архітектурі «файл-сервер».

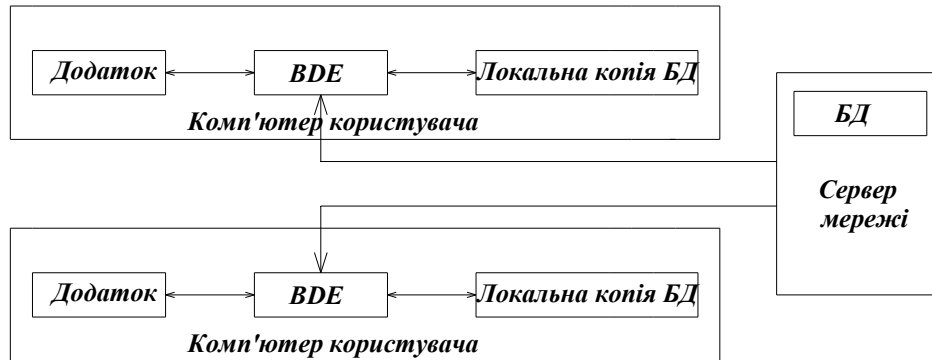


Рис. 2: Схема архітектури «файл-сервер»

Переваги:

1. Простота реалізації;
2. Додаток розробляється фактично на одного користувача і не залежить від комп'ютера, на якому він встановлюється.

Віддалена база даних розташована на комп'ютері-сервері мережі, а додаток, який її опрацьовує розташований на комп'ютері користувача. Це так звана архітектура «клієнт-сервер». В цій архітектурі користувач відсилає запит до даних і отримує тільки ті дані, які його цікавлять на даний момент.

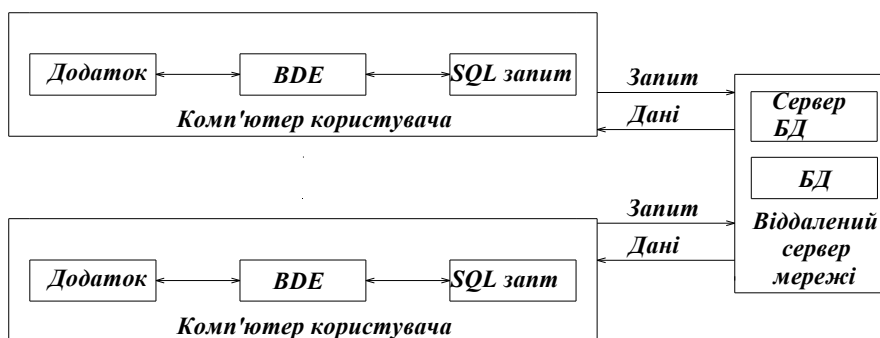


Рис. 3: Схема архітектури «клієнт-сервер»

Переваги архітектури «клієнт-сервер»:

1. Низьке навантаження на мережу, в якій циркулює тільки необхідна інформація;
2. Безпека інформації, оскільки обробка запитів здійснюється єдиною програмою, що розташована на сервері

Таблиці баз даних

Реляційні бази даних складаються з взаємозв'язаних таблиць. Кожна таблиця містить дані про об'єкти одного типу. Сукупність усіх таких таблиць утворюють базу даних. Таблиці, які утворюють базу даних знаходяться в каталозі на жорсткому диску в робочій директрії. Таблиці зберігаються в окремих файлах. До одної таблиці створюються щедекілька файлів, які містять інформацію про ключі, індекси та ін.. Головним є файл з даними, ім'я якого є іменем таблиці, що задається при її створення. Імені інших файлів задаються автоматично. Всі файли мають однакові імена — ім'я таблиці і відрізняються лише розширеннями, які вказують на інформацію, що міститься в даному файлі.

Кожна таблиця складається з рядків і стовпців. Рядок називається записом, а стовпець полем таблиці. Кожне поле повинно мати унікальне ім'я в межах однієї таблиці. Поле містить дані одного з допустимих типів. При введенні значення в поле автоматично здійснюється перевірка на співпадіння типу поля і типу введеного значення. Основу таблиці складає опис її полів. Таблиця обов'язково повинна мати хоча б одне поле.

Поняття структури таблиці включає в себе:

1. Опис полів;
2. Ключі, або задання ключів;
3. Індекси;
4. Обмеження на значення полів;

5. Обмеження зв'язкової цілісності;
6. Паролі.

Всі елементи структури задаються на фізичному рівні і діють для всіх програм, які працюють з обробкою таблиць. Багато з обмежень можна реалізувати програмно, але вони будуть діяти лише в межах власного додатку.

В цілому над таблицею можна виконувати наступні операції:

1. Створення таблиці;
2. Зміна структури (ім'я таблиці не міняється);
3. Перейменування;
4. Видалення таблиці

Ключі та індекси

Ключ представляє собою комбінації полів, дані в яких однозначно визначають кожний запис в таблиці. Простий ключ складається з одного поля, складний відповідно з декількох полів. Поля, по яким будується ключ називаються ключовими. В таблиці може бути визначений тільки один ключ. Ключ забезпечує:

- Однозначну ідентифікацію записів таблиці;
- Попередження повтору значень;
- Прискорення пошуку даних в таблиці;
- Встановлення зв'язку між окремими таблицями БД

Ключ також називається первинним ключем, або первинним (головним) індексом. Інформація про ключі зберігається в окремому файлі або разом з даними таблиці. Значення ключа розташовується в певному порядку. Для кожного ключа існує унікальне посилання, яке вказує на розташування

певного запису. Таблиці різних форматів мають свої особливості побудови ключів, але існують загальні вимоги:

1. Ключ повинен бути унікальним;
2. Ключ не повинен містити поля, які можна видалити без порушення унікальності ключа;
3. В склад ключа не можуть входити поля деяких типів (графічне поле, поле коментарів).

Індекс, як і ключ будується по полям таблиці, однак він може допускати повторення значень складових його полів. Поля, по яким будується індекс називаються індексними. Індеси при створення іменуються. Створення індексів називається індексуванням таблиці.

Використання індексів забезпечує:

- збільшення швидкості доступу до даних;
- сортування таблиці;
- встановлення зв'язків між окремими таблицями;
- використання обмежень зв'язкової цілісності.

Зв'язки між таблицями

Реляційна база даних складається з взаємозв'язаних таблиць. Організація зв'язків між таблицями називається зв'язуванням або об'єднанням таблиць. Зв'язки між таблицями можна встановлювати як при створення БД, так і при створення додатку використовуючи засоби, що представляються обраною СУБД. Для зв'язку таблиць використовуються поля зв'язку. Вони обов'язково повинні бути індексовані. В підпорядковій таблиці для зв'язку з головною таблицею береться індекс, який називається зовнішнім ключем. Склад полів цього індекса повинен частково чи повністю співпадати зі складом полів

індексу головної таблиці. Особливість використання індексів залежить від формату таблиці. Зв'язок між таблицями визначає відношення підпорядкованості. Існують наступні види зв'язку «головний-підпорядкований»:

- відношення «один до одного»;
- «один до багатьох»;
- «багато до одного»;
- «багато до багатьох»

Відношення «один до одного» означає, що одному запису в головній таблиці відповідає один запис в дочірній таблиці. При цьому можливі такі варіанти:

- для кожного запису в головній таблиці є запис в підпорядкованій;
- в підпорядкованій не існує жодного запису, який відповідає запису в головній.

Відношення «один до багатьох» означає, що одному запису з головної таблиці відповідає декілька записів у підпорядкованій таблиці.

«Багато до одного» відрізняється лише напрямком зв'язку від відношення «один до багатьох».

«Багато до багатьох» має місце тоді, коли одному запису головної таблиці відповідають декілька записів підпорядкованої і одночасно одному запису підпорядкованої відповідають декілька записів головної.

Робота зі зв'язаними таблицями має наступні особливості:

1. При зміні поля зв'язку може порушитись зв'язок між записами двох таблиць;
2. При видаленні запису головної таблиці потрібно видаляти відповідні їй записи у підпорядкованій таблиці (каскадне видалення);
3. При доданні запису у підпорядковану таблицю значення її поля

зв'язку повинно бути встановлене рівним значенню поля зв'язку головної таблиці.

4.

Нормалізація реляційної бази даних. Перша нормальна форма. Друга нормальна форма. Третя нормальна форма.

Створення бази даних починається з її проектування. На цьому етапі аналізуються інформаційні потоки з метою визначення набору таблиць та складу її полів. Процес перетворення в основному залежить від досвіду та інтуїції розробника бази даних. Однак, деякі моменти цього процесу є формалізованими. Однією з таких нормалізацій є вимога, згідно якої реляційна база даних повинна бути нормалізована.

Нормалізованою називається база даних, в якій виконуються як мінімум три умови нормалізації. Виконання умов називається приведенням БД до першої форми.

Перша нормальна форма вимагає, щоб кожне поле таблиці було неподільним і не містило груп, що повторюються.

Друга нормальна форма вимагає, щоб усі поля одної таблиці залежали від первинного ключа. Тільки в цьому випадку первинний ключ буде визначати унікальну інформацію.

Третя нормальна форма вимагає, щоб значення будь-якого поля, що не входить в первинний ключ ніяк не залежало від значення інших полів, тобто всі поля в таблиці повинні бути незалежними один від одного і однозначного визначали деякий факт.

База даних «Оптовий магазин»

База даних «оптовий магазин» є локальною реляційною базою даних, яка складається з 8 таблиць. До складу бази даних входять такі таблиці: Goods (перелік товарів), GoodGroups (перелік категорій товарів), Contragent (список контрагентів (покупців/постачальників) оптового магазину з реквізитами контрагентів), Contract (містить в собі перелік договорів, укладених з контрагентами), Income (перелік проведених закупівель товару магазином, та дата закупівлі), IncomeComp (допоміжна таблиця, в якій містяться дані по завезеним партіям товару (вид товару, кількість одиниць завезеного товару)), Realiz (перелік проведених оптових подах), RealizComp (допоміжна таблиця, в якій містяться дані по поданим партіям товару (вид товару, та кількість проданих одиниць товару)).

База даних є нормалізованою. Виконуються три умови нормалізації.

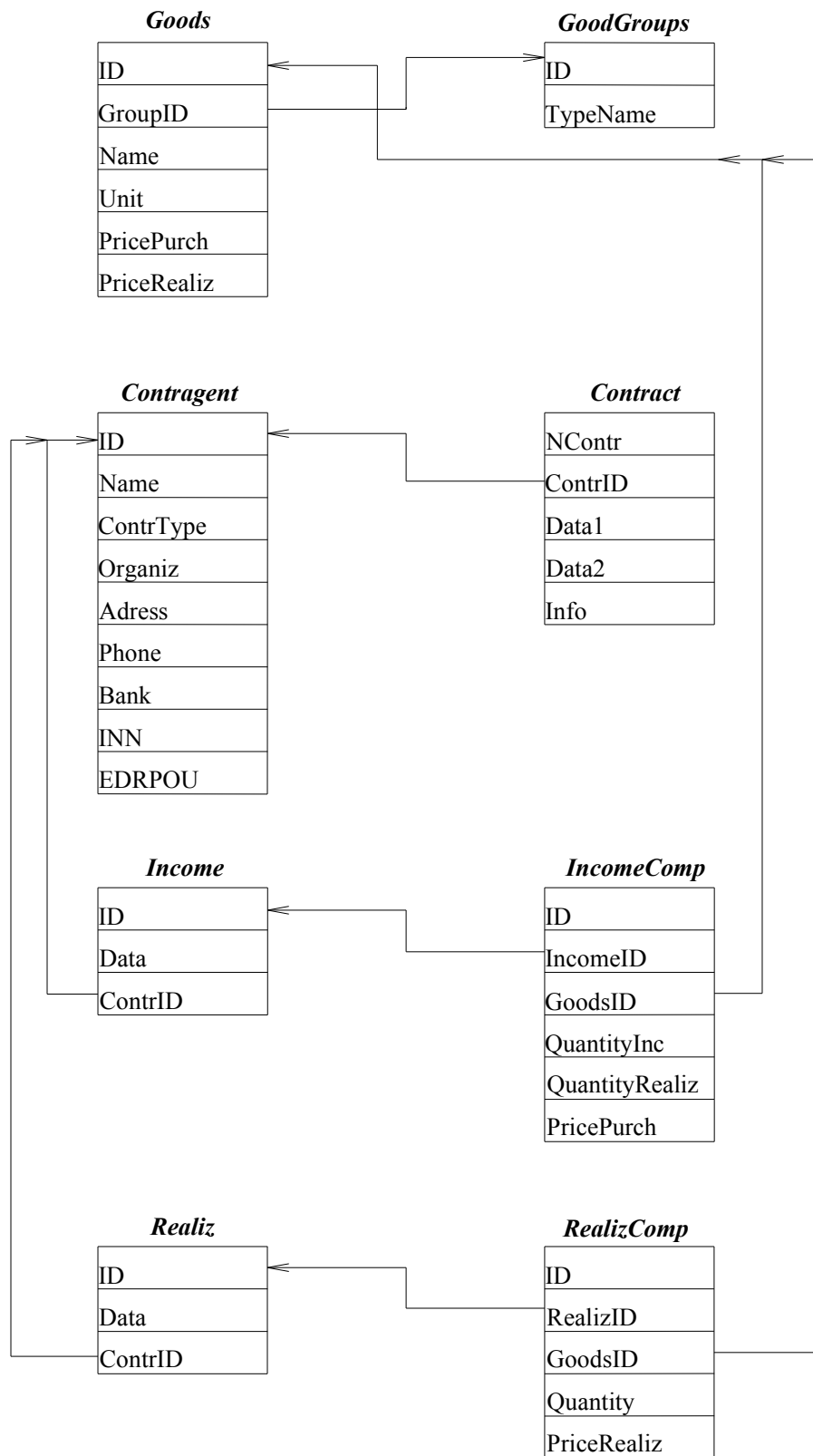


Рис. 4: Структура створеної бази даних

Загальні відомості по SQL

Реляційний спосіб доступу до даних

Реляційний спосіб доступу до даних заключається в операціях з групами записів. Для задання операцій використовуються засоби мови SQL, тому реляційний спосіб доступу до даних називається SQL-орієнтованим.

Засоби SQL застосовуються для операцій з локальними та віддаленими записами. Для локальної бази даних немає переваг при використанні цього методу.

За допомогою SQL-запиту можна:

1. Формувати склад полів набору даних при виконанні додатку;
2. Включати в набір даних поле та записи з декількох таблиць;
3. Відбирати записи за складними критеріями;
4. Сортувати набір даних по будь-якому полю в тому числі неіндексованому;
5. Здійснювати пошук даних.

Основні відомості про мову SQL

SQL (англ. Structured query language - мова структурованих запитів) - декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД. Сам по собі SQL не є ні системою керування базами даних, ні окремим програмним продуктом. Не будучи мовою програмування в тому розумінні, як C або Pascal, SQL може формувати інтерактивні запити або, будучи вбудованою в прикладні програми, виступати в якості інструкцій для керування даними. Стандарт SQL, крім того, вміщує функції для визначення зміни, перевірки і

захисту даних.

Мова SQL орієнтована на виконання дій з таблицями баз даних і даними в цих таблицях, а також для виконання деяких допоміжних дій. На відміну від інших мов програмування в ній відсутні оператори керування обчислювальним процесом (цикли, розгалуження, засоби вводу/виводу). Складену на мові SQL програму називають SQL-запитом. Мова SQL зазвичай інтегрується в інші засоби (оболонки) і використовується в інтерактивному режимі.

Так як SQL не володіє можливостями повноцінної мови програмування, а орієнтована на доступ до даних, то її часто включають в засоби розробки програм. При цьому для роботи з командами SQL використовуються відповідні засоби та компоненти

SQL запити поділяються на статичні та динамічні. Статичний SQL-запит включається в початковий код і на етапі виконання додатку не змінюється. Код динамічного SQL-запиту формується або змінюється в процесі виконання додатку. Такі запити застосовуються у випадку, коли при виконанні запиту слід враховувати дії користувача.

У мові SQL можна виділити наступні підмножини операторів:

1. Визначення даних;
2. Обробка даних;
3. Управління транзакціями;
4. Керування доступом до даних.

В додатках Delphi для виконання операторів SQL використовують компонент Query. Текст SQL-запиту є значенням властивості SQL цього компоненту. Він формується або при розробці додатку або при його виконанні. Формування набору даних виконується при активізації компонента Query, шляхом виклику методу Open, або Active=true. Крім того набрати текст SQL-запиту і виконати його в інтерактивному режимі можна за допомогою інструментальних засобів, таких як:

- DataBaseDesktop

- SQL Builder
- SQL Explorer

Перевірка синтаксису і відлагодження запиту вбудованого в додаток виконується на етапі виконання додатку. В якості результату SQL-запит може повертати набір даних, який складається з записів, що відібрані згідно умов запиту. Такий набір називається результуючим. Регістр букв не впливає на інтерпретацію операторів. В кінці рядка запиту символ «;» не є обов'язковим. Елементи в списках повинні бути розділені комами. Імена таблиць і полів обмежуються одинарними або подвійними лапками. Також допускаються коментарі /*коментар*/.

Функції мови SQL

Мова SQL надає для використання ряд функцій, з яких найбільш використовуваними є наступні:

1. Статистичні функції:

- AVG()
- MAX()
- MIN()
- SUM()
- COUNT()

2. Функції роботи з рядками

- UPPER(Str)
- LOWER(Str)
- TRIM(Str)
- SUBSTRING(Str, FROM n1 to n2) — виділення підрядка
- CAST(<Expr> AS <Type>) - приведення виразу <Expr> до

типу <Type>

- || - конкатенація

3. Функції декодування дати та часу

- `EXTRACT(<Elem> FROM <Expr>)` - з виразу дати або часу виділити значення, що відповідає вказаному елементу.

Визначення даних

Визначення даних — це операції, які виконуються з таблицями. До них відносять:

1. Створення таблиці
2. Визначення таблиці
3. Зміна складу полів
4. Робота з індексами

Для створення таблиці використовується оператор `CREATE TABLE`, який має наступний формат:

```
CREATE TABLE <ім'я таблиці>
(<ім'я поля> <тип даних>,
.....
<ім'я поля> <тип даних>)
```

Створення таблиць в даній курсовій роботі:

```
CREATE TABLE Goods
(ID Autoincrement,
GroupID Number,
Name Alpha,
Unit Alpha,
PricePurch Money,
PriceRealiz Money)
```

Обов'язковими операндами є ім'я таблиці та ім'я хоча б одного поля з відповідним типом даних. Для локальної таблиці її тип автоматично визначається згідно розширення файлу (.db — paradox, dbf — Dbase). Якщо

розширення файлу не вказано, то тип таблиці визначається драйвером, заданим BDE для локальної бази даних.

Файли таблиці роташовані в каталозі бази даних, на який вказує псевдонім (аліас) бази даних. Для компоненти Query псевдонім задає властивість DatabaseName. Порядок слідування рядків з описанням полів визначається порядком розташування полів створеної таблиці. Описи можуть розташовуватися підряд, а не займати окремі рядки оператора.

Типи даних мови SQL та відповідні їм типи для таблиці DBase і Pdox зведено в таблицю:

SQL	Dbase	Paradox
SmallInt	Number(6,10)	Short
Integer	Number(20,4)	LongInteger
Decimal(x,y)	-	BCD
Numeric(x,y)	Number(x,y)	Number
Float(x,y)	Number	Float
Character(N)	Character	Alpha
VarChar(N)	Character	Alpha
Date	Date	Data
Boolean	Logical	Logical
Blob(N,1)	Memo	Memo
Blob(N,2)	Binary	Binary
Blob(N,3)	-	FormattedMemo
Blob(N,4)	OLE	OLE
Blob(N,5)	-	Graphic
Time	-	Time
TimeStamp	-	TimeStamp
Money	Number(20,4)	Money
Autoinc	-	Autoincrement
Bytes(N)	-	Bytes

В таблиці N визначає довжину поля в байтах. X — число цифр, Y —

кількість цифр після десяткової коми.

Для таблиць Paradox можна визначати ключ. Для цього використовується оператор Primary Key, після якого в дужках перераховуються поля, що утворюють ключ.

```
CREATE TABLE Goods
(ID Autoincrement,
GroupID Number,
Name Alpha (250),
Unit Alpha (10),
PricePurch Money,
PriceRealiz Money,
PRIMARY KEY (ID))
```

Для видалення таблиці призначений оператор DROP TABLE <ім'я таблиці>.

Зміна складу полів таблиці

Це можливість додати або вилучити поля таблиці, що приводить до зміни її структури. При зміні складу полів таблиці її не повинні використовувати інші додатки. Для зміни використовується оператор ALTER TABLE <ім'я таблиці>.

```
ALTER TABLE <ім'я таблиці>
ADD <ім'я поля> <тип поля>
DROP <ім'я поля>
.....
ADD <ім'я поля> <тип поля>
DROP <ім'я поля>
```

Створення та видалення індексу

Індекс створюється оператором CREATE INDEX <ім'я індекса> ON <ім'я таблиці> (<ім'я поля>, <ім'я поля>, ... ,<ім'я поля>). Одним оператором можна створити один індекс, при цьому одне поле може входити в склад різних

індексів.

```
CREATE INDEX IndGroupID ON Goods (GroupID)
```

```
CREATE INDEX IndContrID ON Contract (ContrID)
```

Для видалення індексу використовується оператор `DROP INDEX <ім'я таблиці> <ім'я індексу>`.

Опис оператора **SELECT**

Оператор `SELECT` використовується для відбору записів, які задовільняють складному критерію пошуку і має наступний формат:

```
SELECT [DISTINCT] <список полів>
FROM <список таблиць>
[WHERE <умова відбору>]
[ORDER BY <список полів для сортування>]
[GROUP BY <список полів для групування>]
[HAVING <умова відбору>]
[UNION <вкладений оператор SELECT>]
```

Приклад з курсової роботи:

```
SELECT T.ID, G.TypeName, T.Name, T.Unit,
T.PricePurch, T.PriceRealiz
FROM Goods T, GoodGroups G
WHERE T.GroupID=G.ID
```

```
SELECT Income.ID, Income.Data
FROM Income
```

```
SELECT Contract.NContr, Contragent.Name,
Contract.Data1, Contract.Data2, Contract.Info
FROM Contragent, Contract
WHERE Contract.ContrID=Contragent.ID
```

Результат виконання SQL-запиту заданого оператором `SELECT` являє собою вибірку записів, що відповідають заданим умовам. В результуючому

наборі можуть бути записи, що повторюються. Цією можливістю управляє операнда DISTINCT. В операторі SELECT обов'язково включаються списки полів і операнд FROM. В списку операнда FROM перераховуються імена таблиць, з яких вибираються записи. Якщо в набір даних необхідно включити всі поля таблиці, то після оператора SELECT вказується символ «*».

```
SELECT * FROM Realiz
```

Якщо список містить поя декількох таблиць, то для того, щоб вказати належність поля до деякої таблиці використовується його складне ім'я: <ім'я таблиці>.<ім'я поля>

Операнд WHERE задає критерії відбору, яким повинні відповідати записи результуючого набору даних. Вираз, який описує умову є логічним і його елементами можуть бути:

- імена полів;
- операції порівняння;
- логічні та арифметичні операції;
- дужки;
- спеціальні функції.

Операнд GROUP BY дозволяє виділити групи записів в результуючому наборі. Групою є записи з однаковими значеннями в полях, що перераховані за операндом GROUP BY.

Операнд HAVING діє разом з операндом GROUP BY і використовується для відбору записів у групах.

Операнд ORDER BY використовується для сортування записів списку полів, який вказується після цього операнда. За замовчуванням сортування здійснюється в порядку зростання значень. Якщо необхідно здійснити сортування в порядку спадання значень, необхідно вказати операнд DESC.

```
SELECT T.ID, G.TypeName, T.Name, T.Unit,
T.PricePurch, T.PriceRealiz
FROM Goods T, GoodGroups G
WHERE T.GroupID=G.ID
GROUP BY T.Name
```

Оператори SELECT можуть мати складну структуру і бути вкладені один в один. Для об'єднання операторів використовують операнд UNION, в якому записується вкладений оператор SELECT. Результатом виконання запиту будуть записи, що відповідають обома умовам відбору операнду SELECT.

Оператори керування полями

Керування полями полягає в тому, щоб впорядкувати поля, що входять до результуючого набору. Порядок слідування полів в результуючому наборі за замовчуванням відповідає порядку слідування фізичних полів таблиці.

При необхідності отримати дані з деяких полів таблиці, після слова SELECT через кому перераховуються імена цих полів. Порядок слідування полів буде відповідати порядку слідування полів у запиті.

```
SELECT T.ID, G.TypeName, T.Name, T.Unit,
T.PricePurch, T.PriceRealiz
FROM Goods T, GoodGroups G
```

Крім фізичних полів таблиці в набір даних можуть включатися і обчислювальні поля. Для отримання обчислювального поля в списку полів вказується вираз по якому необхідно отримати дані. Вираз складається за загальними правилами і може не містити імена полів.

```
SELECT T.ID, G.TypeName, T.Name,
(Q.QuantityInc-Q.QuantityRealiz), T.Unit,
T.PricePurch, T.PriceRealiz
FROM Goods T, GoodGroups G, IncomeComp Q
WHERE T.GroupID=G.ID
```

```
SELECT Goods.ID, Goods.Name, Goods.Unit,
```

```
IncomeComp.QuantityInc, IncomeComp.PricePurch,  
IncomeComp.QuantityInc*IncomeComp.PricePurch  
FROM Goods, IncomeComp  
WHERE IncomeComp.GoodsID=Goods.ID
```

Ще одною перевагою мови SQL є простота об'єднання таблиць. Для того, щоб результуючий набір даних містив записи з різних таблиць достатньо після операнда FROM перерахувати імена таблиць через кому.

```
SELECT * FROM Goods, IncomeComp
```

Проста умова відбору записів

Набір даних в основному обмежений записами, які задовільняють умовам, що задаються за допомогою оператора WHILE. Критерії відбору можуть варіюватися від найпростіших, в яких порівнюються два значення, до складних, коли враховуються багато факторів. Критерій відбору являє собою логічний вираз, в якому можуть застосовуватися наступні операції:

1. =, >, <, >=, <=, <> або !=, !>, !<
2. LIKE — порівняння по шаблону
3. IS NULL — перевірка на нульове значення
4. IN — перевірка на входження
5. BETWEEN — перевірка на входження в діапазон

Простий критерій відбору складається з одної операції:

```
SELECT Goods.ID, Goods.Name, Goods.Unit,  
IncomeComp.QuantityInc, IncomeComp.PricePurch,  
IncomeComp.QuantityInc*IncomeComp.PricePurch  
FROM Goods, IncomeComp  
WHERE IncomeComp.GoodsID=Goods.ID
```

Умова на часткове співпадіння записів:

```
ELECT Goods.ID, Goods.Name
```

```
FROM Goods
```

```
WHERE Name LIKE "Key"
```

У виразах операції LIKE допускається використання шаблону, в якому дозволено використовувати всі алфавітно-цифрові символи. При цьому два символи мають спеціальне значення.

1. % - заміщення будь-якої кількості символів
2. _ - заміщення одного символу

Перед операндом LIKE може використовуватись NOT. Тоді здійснюється перевірка на неспівпадіння.

Складні умови відбору записів

При відборі записів можна використовувати декілька операндів одночасно, вони утворюють складний критерій відбору, який складається:

1. Простих умов
2. Логічних операцій AND, OR, NOT
3. ()

Практична частина (опис роботи програми)

Основне (вихідне) вікно при запуску програми має вигляд:

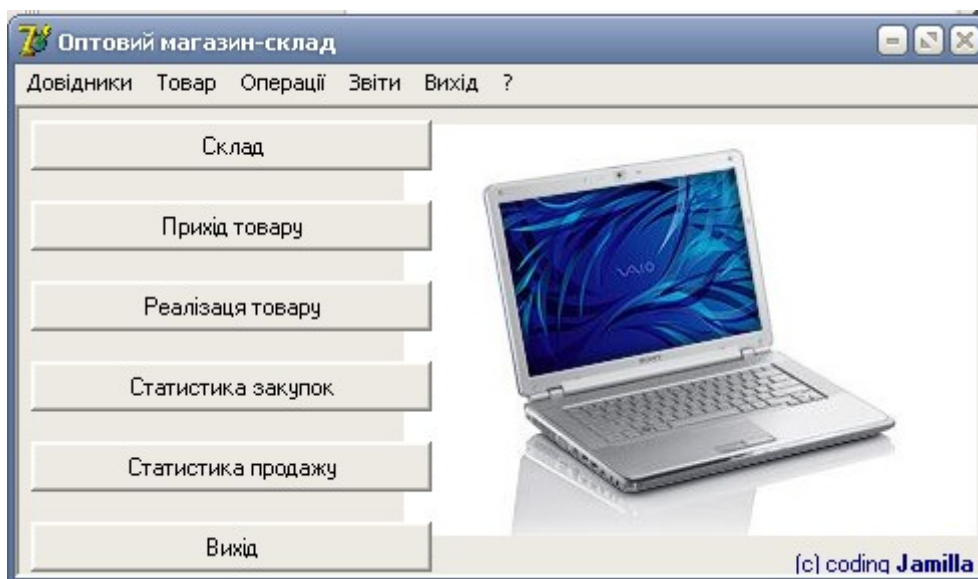


Рис. 5: Основне вікно програми

Воно містить головне меню та шість кнопок для швидшого доступу до основного функціоналу прикладної програми. Навігація по головному меню дає доступ до перегляду наявності товару на складі, номенклатури (товару), категорій товару на складі, перегляду списку контрагентів, що співпрацюють з підприємством, перегляду переліку укладених договорів та інше.

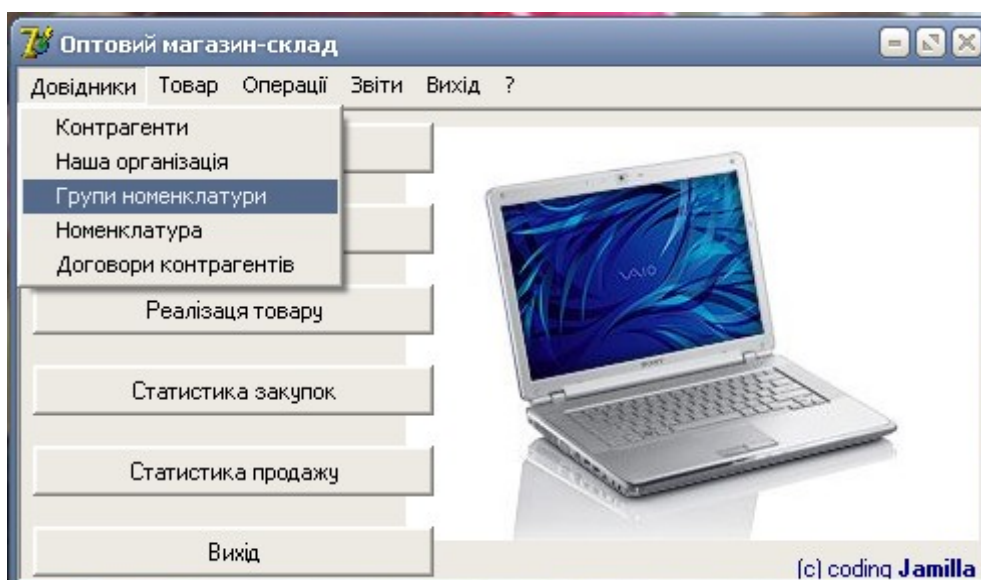
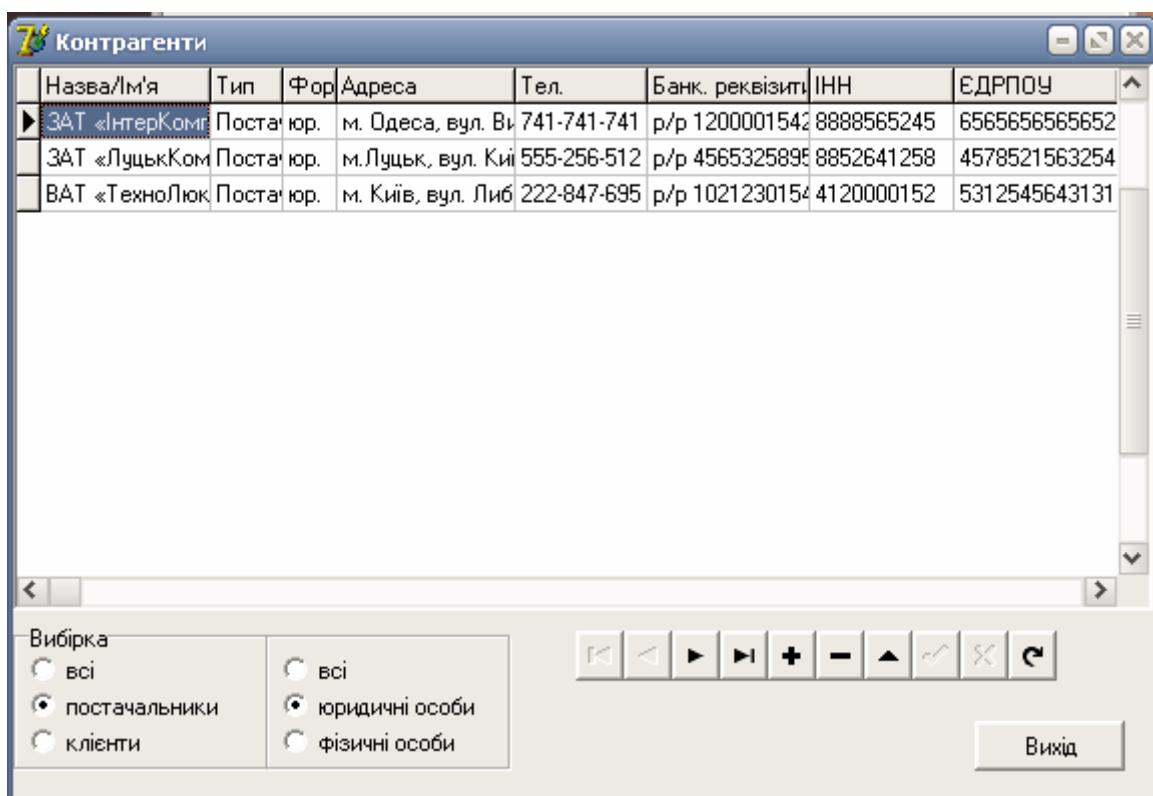


Рис. 6: Навігація головним меню програми

Вікно довідки по контрагентам дає можливість переглянути всіх контрагентів, що співпрацюють з підприємством. Таблиця звіту містить в собі такі поля: Назва/Ім'я контрагента, Тип (контрагент може бути або клієнтом або постачальником), Форма (юридична чи фізична особа), Адреса (юридичне місцезнаходження фірми-представника (окремого представника)), Телефон (контактний), Банківські реквізити, ІНН (Індивідуальний податковий номер), ЄДРПОУ (Єдиний Державний реєстр підприємств та організацій України).

Дане вікно дає можливість робити вибірку по типу контрагентів та по формі, що є дуже зручним при значних розмірах бази даних. Є можливість додавання нових контрагентів до бази даних.



The screenshot shows a software window titled 'Контрагенти' (Counterparties). It contains a table with the following data:

Назва/Ім'я	Тип	Форма	Адреса	Тел.	Банк. реквізити	ІНН	ЄДРПОУ
ЗАТ «ІнтерКом	Поста	юр.	м. Одеса, вул. В	741-741-741	р/р 1200001542	8888565245	6565656565652
ЗАТ «ЛуцькКом	Поста	юр.	м. Луцьк, вул. Ки	555-256-512	р/р 4565325895	8852641258	4578521563254
ВАТ «ТехноЛюк	Поста	юр.	м. Київ, вул. Либ	222-847-695	р/р 1021230154	4120000152	5312545643131

Below the table, there are filter options for 'Вибірка' (Selection):

- Всі (All) - radio button
- постачальники (suppliers) - radio button (selected)
- клієнти (clients) - radio button
- всі (All) - radio button
- юридичні особи (legal entities) - radio button (selected)
- фізичні особи (physical entities) - radio button

Navigation buttons (back, forward, search, etc.) and a 'Вихід' (Exit) button are also present.

Рис. 7: Інформація по контрагентам

Вікно довідки по товарам має наступний вигляд і дає можливість додати нові категорії товарів:

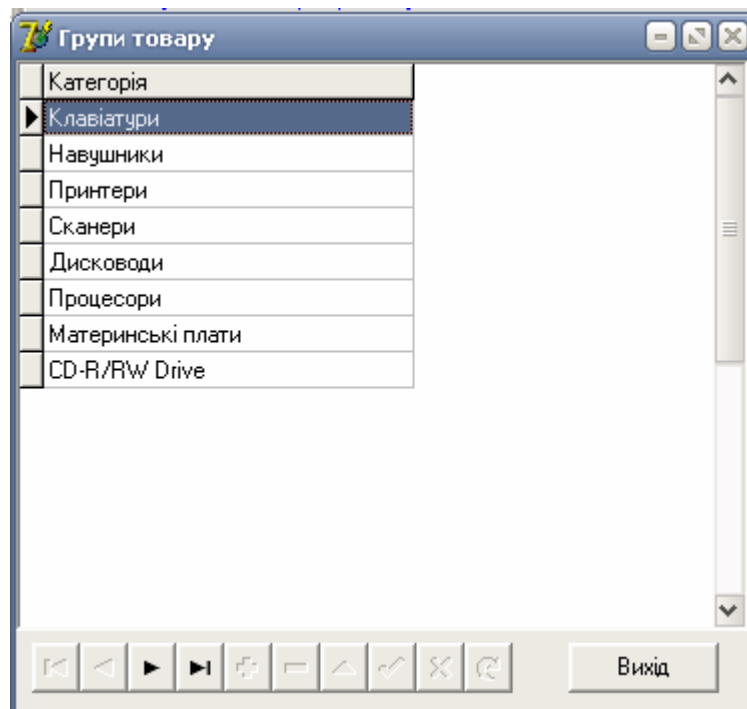


Рис. 8: Групи номенклатури

Вікно номенклатури товару містить таблицю, в якій відображається повний перелік товарів, що може бути в оптовому магазині. таблиця містить такі поля: Артикул (ідентифікатор, унікальний номер товару), Категорія, Назва товару, Одиниці вимірювання, Ціна закупки, Ціна продажу. Аналогічний вигляд має вікно «Товар на складі», що викликається через команду меню Товар→Склад, але воно має дві особливості. В ньому не відображається ціна закупки, але при цьому присутнє поле кількості товару на складі. Кількість товару на складі визначається, як різниця кількості товару що прийшла і товару, що продалась. В вікні перегляду номенклатури товару є можливість робити вибірку за значенням: обирати товари певної категорії і шукати товари за співпадінням назви. З вікна номенклатури товару можна легко перейти до вікна «Товар на складі» і до перегляду категорій номенклатури товару.

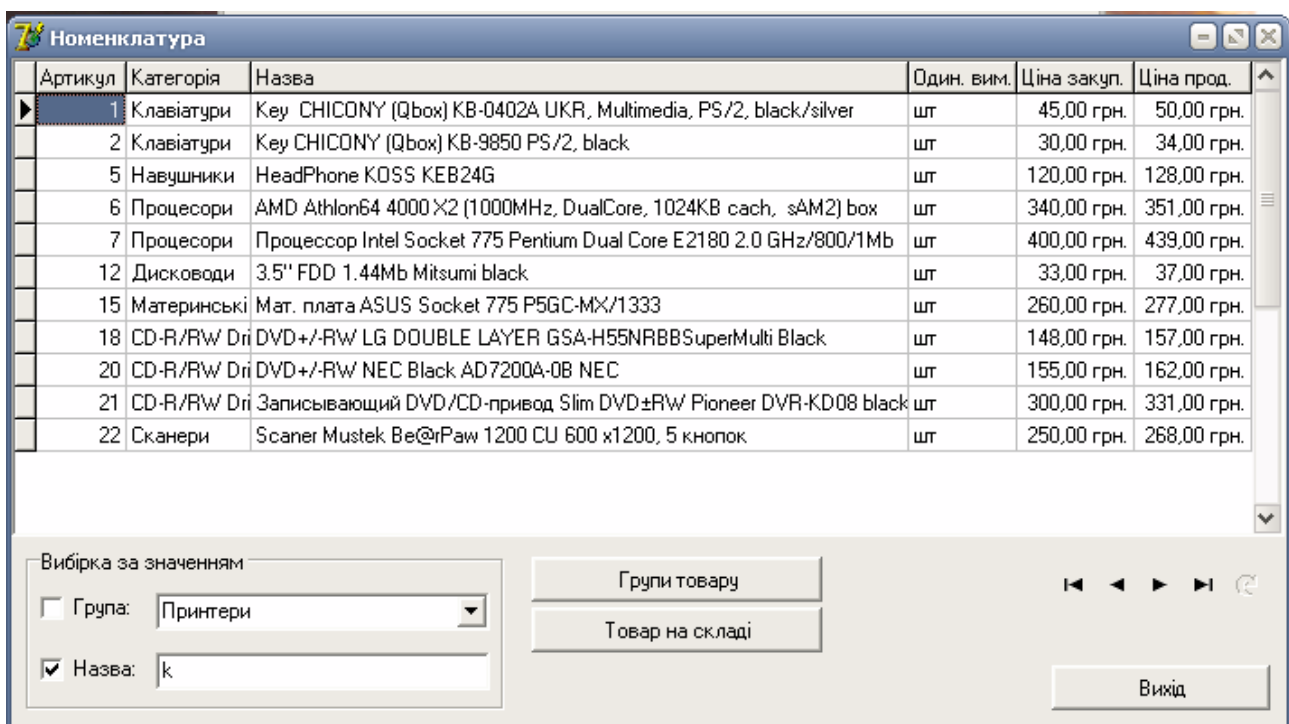


Рис. 9: Номенклатура товару

Договори контрагентів. В цьому вікні міститься таблиця з переліком укладених договорів. Вона містить такі поля: № контр. (індивідуальний ідентифікаційний номер контрагента), Назва/Ім'я, Дату початку дії договору, дату кінця дії договору а також додаткову інформацію про укладений договір.

Є можливість робити вибірку по даті. Вибірка по початку дії договору в певному часовому періоді, вибірка по закінченню дії договору в певному часовому періоді, а також комбінація цих двох варіантів (наприклад дата початку дії з такого-то числа по дату закінчення дії по таке-то число). Дата початку дії договору не повинна бути пізнішою від дати закінчення дії договору. При введенні некоректних даних з'являється повідомлення, яке застерігає користувача від некоректних дій (Рис. 11).

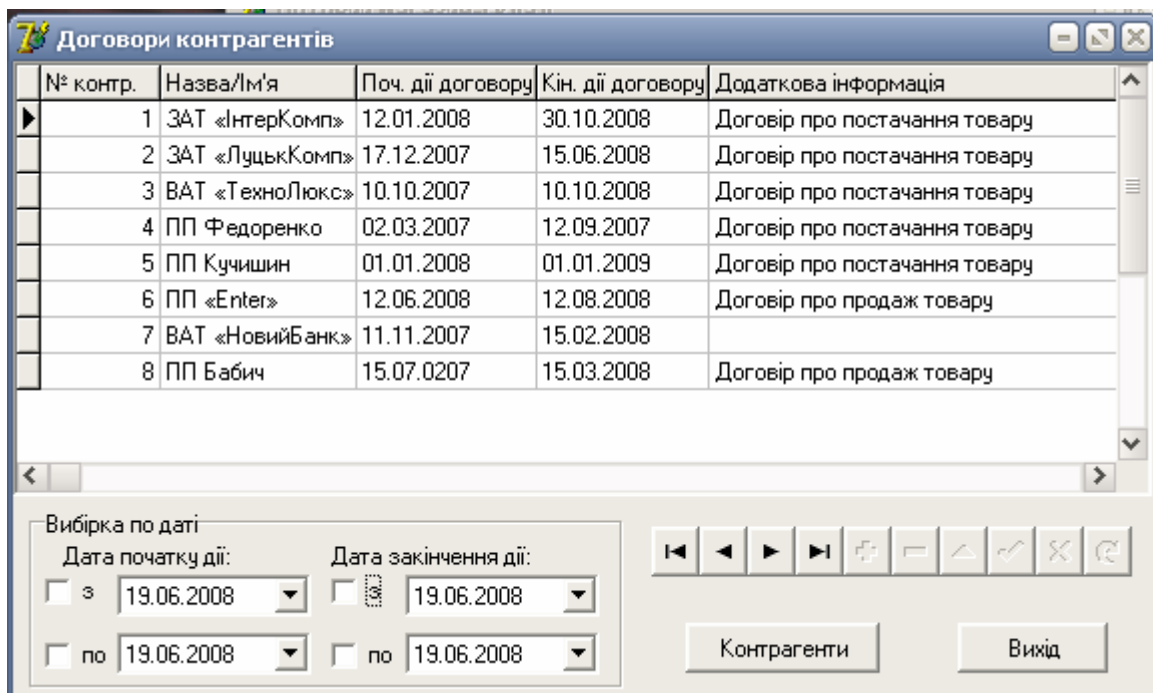


Рис. 10: Договори контрагентів

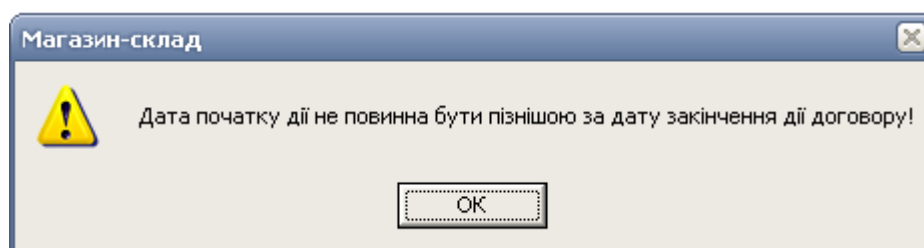


Рис. 11: Повідомлення застереження користувача від некоректних дій

Статистика закупівлі. Вікно викликається через пункт меню Товар→Статистика закупок або через натискання кнопки на головному вікні програми. Таблиця містить відомості про кількість закупленого товару, ціну закупки та вартість. Вартість означається як кількість товару помножена на ціну. Також є можливість робити вибірку за датою. Аналогічний вигляд має вікно «Статистика продажу», яке викликається через пункт меню Товар→Статистика продажу.

Артикул	Дата закуп.	Постачальник	Товар	Кількість	Ціна закуп.	Вартість
2	15.05.2008	ЗАТ «ЛьвцьКомп»	Scanner HP ScanJet G3010 photo	3	543,00 грн.	1 629,00 грн.
2	15.05.2008	ЗАТ «ЛьвцьКомп»	Printer Epson Stylus C 91	1	290,00 грн.	290,00 грн.
5	01.02.2008	ПП Кучишин	AMD Athlon64 4000 X2 (1000MHz, DualCo	4	340,00 грн.	1 360,00 грн.
9	11.11.2007	ПП Кучишин	Key CHICONY (Qbox) KB-0402A UKR, Mu	5	45,00 грн.	225,00 грн.
9	11.11.2007	ПП Кучишин	Key CHICONY (Qbox) KB-9850 PS/2, black	5	30,00 грн.	150,00 грн.
9	11.11.2007	ПП Кучишин	3.5" FDD 1.44Mb Mitsumi black	7	33,00 грн.	231,00 грн.
6	13.05.2008	ЗАТ «ЛьвцьКомп»	Scanner Mustek Be@rPaw 1200 CU 600 x1	2	250,00 грн.	500,00 грн.
6	13.05.2008	ЗАТ «ЛьвцьКомп»	Scanner HP ScanJet 2410G USB2	1	340,00 грн.	340,00 грн.

Рис. 12: Статистика продажу

Артикул	Дата прод.	Клієнт	Товар	Кількість	Ціна	Вартість
2	10.03.2008	ПП Марченко	HeadPhone KOSS KEB24G	2	128,00 грн.	256,00 грн.
2	10.03.2008	ПП Марченко	Процессор Intel Socket 775 Pentium Dual C	1	439,00 грн.	439,00 грн.
6	12.04.2008	БАТ «НовийБан»	Scanner HP ScanJet G3010 photo	2	571,00 грн.	142,00 грн.
4	02.02.2008	БАТ «УкрТелеф»	Мат. плата ASUS Socket 775 P5GC-MX/13	2	277,00 грн.	554,00 грн.
3	05.06.2008	ПП «Enter»	HeadPhone+Mic Cosonic CD-930MV (blister)	7	19,00 грн.	133,00 грн.
5	03.03.2008	ПП Марченко	3.5" FDD 1.44Mb Mitsumi black	3	37,00 грн.	111,00 грн.
5	03.03.2008	ПП Марченко	Мат. плата ASUS Socket 775 P5GC-MX/13	1	277,00 грн.	277,00 грн.
1	11.11.2007	ПП Бабич	DVD+/-RW LG SuperMu LightScribe H55LB	2	175,00 грн.	350,00 грн.
1	11.11.2007	ПП Бабич	Scanner EPSON Perfection V200 Photo	1	588,00 грн.	588,00 грн.
2	10.03.2008	ПП Марченко	DVD+/-RW LG DOUBLE LAYER GSA-H55N	3	157,00 грн.	471,00 грн.

Рис. 13: Статистика закупівлі

Висновки

Поставлене завдання до курсової роботи виконано. Необхідно було створити базу даних та програмний дотак, що опрацьовує створену базу. База даних створена на основі таблиць Paradox та містить вісім таблиць. Створена база є реляційною базою даних і таблиці пов'язані між собою за допомогою полів-ідентифікаторів. Найбільшою за обсягом є таблиця номенклатури товару. На даний момент вона містить 25 записів. Але при подальшому розробці бази даних вона може сягати значних розмірів.

Прикладна програма працює з базою даних та має зручний інтерфейс. Створена за допомогою програмного середовища Borland Delphi. При подальшому вдосконаленні може використовуватись на різних видах підприємств. При незначних змінах в інтерфейс може використовуватись в простих магазинах. При вдосконаленні програми важливим пунктом є реалізація автоматичного створення звітів (прайс-листа, товару на складі, книги покупок, книги продаж, накладних та ін.)

Перелік використаної літератури

1. Баженова И.Ю. Delphi 7. Самоучитель программиста. – М.; Кудиц-образ, 2003. – 436 с.
2. <http://en.wikipedia.org/wiki/SQL>
3. <http://www.delphisources.ru/pages/articles.html>
4. <http://masterdelphi.ucoz.ru/publ/>
5. Дж. Боуман, С. Эмерсон, М.Дарновски. Практическое руководство по SQL. – М.: Нолидж, 2002. — 311 с.
6. В. Фаронов. Delphi 6: учебный курс. - Спб.: Питер, 2002. - 512 с.: - ил.
7. Тихомиров Ю.В. Microsoft ® SQL Server 7.0: разработка приложений. - Спб.: БХВ — Санкт-Петербург, 1999. - 352 с., ил.

Додатки