Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Проектування бази даних для інформаційної системи продажів комп'ютерної техніки

Виконав(ла): студент(ка) 4 курсу, групи СН-41
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

Олещук А.А.
(підпис) (прізвище та ініціали)

Керівник Гром'як Р.С.
(підпис) (прізвище та ініціали)

Нормоконтроль Шимчук Г.В.
(підпис) (прізвище та ініціали)

Завідувач кафедри Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент Кареліна О.В.
(підпис) (прізвище та ініціали)

Тернопіль
2021

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«25» січня 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

студенту Олещук Аліна Андріївна
(прізвище, ім'я, по батькові)

1. Тема роботи **Проектування бази даних для інформаційної системи продажів комп'ютерної техніки**

Керівник роботи **к.т.н., доц. Гром'як Р.С.**
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*02*» *березня* 2021 року № *4/7-171*

2. Термін подання студентом завершеної роботи 19 червня 2021 р.

3. Вихідні дані до роботи Опис предметної області

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ; 1 Теоретична частина ; 1.1 Опис процесу проектування баз даних. 1.2 Аналіз вимог до програмного забезпечення ; 2 Практична частина 2.1 Проектування бази даних 2.2 Опис програмного забезпечення 3 Безпека життєдіяльності та основи охорони праці 3.1 Поняття та об'єкт аналізу технічної безпеки 3.2 Розрахунок захисного заземлення Висновок Перелік використаних джерел

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема. 2. Структура базиданих. 3. Схема бізнес-процесів 4. Алгоритм оформлення замовлення 5. – 7 Зразки інтерфейсу користувача .
11. Висновки3

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Гурик О.Я., к.т.н., доц.		

7. Дата видачі завдання 25 січня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	25.01.21-27.01.21	*Виконано*
2.	Підбір джерел по темі роботи	28.01.21 – 01.04.21	*Виконано*
3.	Оформлення першого розділу	15.04.2021	*Виконано*
4.	Оформлення другого розділу	30.04.2021	*Виконано*
5.	Виконання завдання до підрозділу «Безпека		
	життєдіяльності, основи охорони праці»	15.05.2021	*Виконано*
6.	Оформлення кваліфікаційної роботи	07.06.2021	*Виконано*
7.	Перевірка на плагіат	07.06.2021	*Виконано*
8.	Нормоконтроль	15.06.2021	*Виконано*
9.	Попередній захист кваліфікаційної роботи	18.06.2021	*Виконано*
10.	Захист кваліфікаційної роботи	22.06.2021	

Студент _____ Олещук А.А..
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гром'як Р.С.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Проектування бази даних для інформаційної системи продажів комп'ютерної техніки // Кваліфікеаційна робота бакалавра // Олещук Аліна Андріївна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СН-41 // – Тернопіль, 2021. // с. – , рис. – , табл. – , кресл. – , додат. – , бібліогр. – .

Ключові слова: база даних, інформаційна система, склад, облік, прихід, розхід.

Кваліфікаційна робота стосується проектування інформаційної системи для ведення обліку продажів обчислювальної техінки як реалізація типових бізнес-процесів у цій предметній області.

Програма розроблена за допомогою мови програмування Java. Як наслідок використання вказаної мови програмування розробка програми звелася до вирішення наступних задач:

- розробка структури бази даних;
- реалізація розробленої структури засобами MySQL Workbench;
- побудова інтерфейсу програми;
- написання процедур обробки подій при роботі програми;
- забезпечення різноманітних варіантів взаємодії користувача з програмою.

Програмний код відкомпільовано для роботи на ПЕОМ під керуванням ОС Windows та відтестовано на робочих станціях під керуванням вказаної операційної системи.

ANNOTATION

Data base design for information system of computer equipment sales // Qualification work of a bachelor // Oleshchuk Alina Andriyivna // Ternopil Ivar Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Computer Science Department, Group CH-41// Ternopil, 2021 // p. - , Fig. - , table. - , posters - , references - .

Keywords: database, information system, warehouse, accounting, arrival, expenditure.

Qualification work concerns the design of an information system for accounting of sales of computer equipment as the implementation of typical business processes in this subject area.

The program is developed using the Java programming language. As a result of using the specified programming language, program development was reduced to solving the following tasks:

- development of the database structure;
- implementation of the developed structure by means of MySQL Workbench;
- construction of the program interface;
- writing procedures for processing events during the program;
- providing various options for user interaction with the program.

The program code was compiled to run on a Windows PC and tested on workstations running the specified operating system.

ЗМІСТ

ВСТУП

Нас оточують величезні потоки інформації. З часом їх кількість так і обсяг даних тільки ростуть. Тому в різних закладах, установах, організаціях різного масштабу постає задача організувати ці потоки даних, керуваит ними, приймати рішення н основі цих даних для підвищення ефективності роботи установи чи підприємства. Паперовий докуменгтообіг все більше витісняється електронним з викориистанням автоматизованих систем управління (АСУ). На сьогодні без них неможливо представити роботу більшості будь-яких компаній різних форм власності та масштабів.

Вартість зберігання інформації у файлах значно дешевша, ніж на папері. АСУ дозволяють зберігати, структурувати інформацію та надавати її користувачеві відповідно до його запитів. Використання клієнт/серверних технологій дозволяють зекономити значні ресурси, а, головне, час отримання потрібної інформації, а також спрощують доступ і ведення звітності, оскільки вони ґрунтуються на комплексній обробці даних. Крім того ЕОМ дозволяє зберігати будь-які формати даних, текст, креслення, дані в рукописній формі, фотографії, записи голосу і т.д.

У процесі свого функціонування підприємство вступає у численні і різноманітні відносини з фізичними і юридичними особами, державою. Ці відносини регулюються законодавством України. В умовах реформування економіки нормативно-правова база діяльності установ і підприємств часто змінюється.

Метою дипломної роботи є розробка атоматизованої системи ведення обліку продажів комп'ютерної техніки. Оскільки останнім часом появилися нові правила ведення обліку з урахуванням сучасних реалій, то автоматизація даної ділянки роботи звичайно є актуальним питанням для будь-якої організації подібного роду діяльності.

Відповідно до поставленої мети у проекті програми потрібно розв’язати наступні завдання:

1) навести характеристику об’єкта дослідження;

2) навести характеристику розв’язуваної задачі та дослідити існуючі варіанти її вирішення;

3) розробити поставлені задачі, визначити вхідну та вихідну інформацію;

4) розробити алгоритм розв’язку задачі;

розробити програмне забезпечення для реалізації автоматизованої системи ведення обліку продажів.

1 ТЕОРЕТИЧНА ЧАСТИНА

1.1 Процес проектування баз даних

Вдалий проект бази даних для системи є фундаментальним рішенням, оскільки саме від нього залежить можливість реалізації всіх танзакцій, зручність використання, вибір програмної архітектури і в цілому життєздатність системи та її подальше маштабування і підтримка. Тому питання проектування БД для інформаційнних систем (ІС) виділяють в окремий напрямок.

Проектування баз даних – це покроковий ітераційний процес прийняття рішень на основі аналізу предметної області, вимог до даних з боку прикладних програм. Форми їх подання для користувачів, синтезу логічних моделей даних та їх фізична реалізація у вигляді файлових та програмних структур,і, відповідно, вибір програмно-апаратнохї платформи лдж яствореннята рооти з базою даних. При проектуванні бази даних виділяюь такі етапи: концептуальний проект, логічний проект, фізичний проект. На кожному з них виконують встановлений набір кроків (етапів) проектування, який забезпечує реалізацію наступних кроків.

На концептуальному рівні здійснюється розробка інформаційної моделі для заданої предметної області з виділенням типів сутностей та бізнес-процесів, котрі відображаються у вигляді зв'язків між цими виділеними типами сутностей. На цьому ж етапі виділяють набір характеристик кожного типу сутності (атрибутів) та задають властивості цих атрибутів.

На логічному проектуванні здійснюють перенесення концептуальної моделі у вибрану логічну модель, наприклдад, реляційну. Під час цього етапу здійснюється проектування відношень з їх атрибутами, створюються первинні та звонішні ключі, описуються правила цілісності посилань дл язв'язків. Важливим є також усунення чи перетворення елементів концептуальної моделі, котрі не можуть бути реалізовані напряму у вибраній логічній моделі. Виконується усунення зайвої надлишковості даних шляхом нормалізації набору даних до пвеного рівня. Як правило – до третьохї нормальної форми – 3НФ.

На фізичному рівні здійснюється реалізація розробленої логічної моделі засобами обраної системи керування базами даних – СКБД. Вибираються типи даних для атрибутів, а також виконуєтьяс реалізація певних елементів бізнес-логіки на стороні СКБД, щоби полегшити подальший супровід інформаційної системи і уніфікувати обробку даних. Мається на увазі створення збережуваних процедур, тригерів. Для зручності роботи користувача майбутньої розроблюваної ІС здійснюється проектування набору індексів для таблиць БД, щоби оперативно отримати впорядковані набори даних без виконання сортування, розроьляється набір віртуальинх таблиць – представлень.

Важливим етапом при можливості на етапі физичного проектування є створення системи ролей користувачів БД і СКБД, надання привілеїв користувачам, управління іншими параметрами захисту БД від несанкціонованого доступу та від пошкодження даних (резервне копіювання, вибір структури файлів БД) тощо.

Проектування БД масштабний процес, що вимагає залучення не лише спеціалістів ІТ, але і експертів з предметної облвасті, для якої створюється БД, бізнес-аналітиків, управлінців. Вдалий проект бази даних напряму впливає на продуктивність майбутньої інформаційної системи, її виконання свого функціонального призначення. Можна сказати, що якісний проект БД задає рівень якості всієї проектованої системи.

Помилки в проекті бази даних можуть спричинити потребу в реалізації складної бізнес-логіки засобами прикладних програм, що ускладнює супровід такої системи і негативно впливає на вартість розробки. Крім того вкрай небажаним є зміни в структуру БД, на основі котрої вже працює створена інформаційна система, оскільки це мож вимагати переписування програмного коду, інколи у досить великих обсягах.

1.2 Аналіз вимог до програмного забезпечення

1.2.1 Аналіз предметної області

Головною проблемою у даній предметній області є повільний та складний обмін інформацією між зацікавленими особами. При створенні даної програми можна не тільки систематизувати всі паперові відомості про товар, але й забезпечити оперативні вибірки , виводити відповідні дані .

Програма дозволить здійснювати оперативний моніторинг управління інформацією, виводити продаж комп'ютерної техніки (КТ), оформлювати нові замовлення на поставки КТ, підбивати фінансові підсумки дня, тижня, місяця, шукати техніку по необхідній марці, ціні, підраховувати залишок товарів на складі, виводити список постачальників і постачань КТ.

Програма може бути корисною для всіх ланок підприємства і полегшить пошук всієї потрібної інформації .

Одними із основних проблем на данний момент є:

- недостатньо швидке отримання інформації про даний продукт;
- складність збереження і систематизації інформації;
- складність отримання інформації про поточне положення у виробництві даного товару;
- складність зберігання, обробки та маніпуляції з інформацією про продукт;
- відсутність єдиного систематизованого інформаційного банку;
- відсутність систематизованої інформації про характеристики товарів, наявність їх на складі чи у постачальника;

Тому потрібна система, яка дозволить користувачам отримувати необхідну інформацію самостійно, не відволікаючи від роботи інших працівників.

Тому потрібна система, яка надаватиме користувачам потрібну їм інформацію самостійно, без залучення інших спеціалістів.

Можливості даної системи такі:

- реалізація бізнес-логіки роботи користувачів для зручного подання даних;

- система спростить отримання інформації про кожен продукт чи товар;

- система дасть можливість класифікувати замовлення по терміновості їх виконання та буде реалізовувати різну логіку керування такими замовленнями залежно від їх типу;

- система надаватиме користувачам інформацію про обсяги продажів товарів за певні проміжки часу. Наприклад, за тиждень, місяць і т.д.;

- всі користувачі можуть з використанням свого клієнтського ПЗ відслідковувати свої замовлення, екількість ресурсів, товарів і т.п.;

- система дозволить переглядати списки товарів на складах, їх характеристику, списки покупців та постачальників ;

- система дозволить просто та швидко відшукати необхідний товар за різними критеріями чи замовити його при відсутності.

В системі має бути реалізовано ряд задач, які будуть відповідати конкретному типу користувача:

- облік постачальників і постачань КТ;

- облік продажів КТ;

- підрахунок залишків товарів;

- оформлення замовлень на поставки КТ;

- підбиття фінансових підсумків дня(по магазину);

- аналіз об’єму продажів по днях, тижнях і по місяцях;

- пошук техніки по необхідних критеріях.

Вигоди даної системи такі:

- оперативне отримання інформації про замовлений продукт;

- можливість індивідуального підходу до даного замовлення;

- система дозволить виводити фінансові підсумки дня, тижня, місяця;

- оптимальний розподіл часу виконання замовлення;

- покращиться швидкодія системи;

- дана система дозволить забезпечити надійне зберігання даних.

Система використовуватиметься на конкретному підприємстві і при потребі застосування на іншому підприємстві повинна легко модифікуватись.

1.2.2 Постановка задачі

Для того щоб реалізувати потрібне програмне забезпечення необхідно вирішити наступні задачі:

- встановити СУБД MySql;

- встановити Java конектор до БД.

В базу даних повинна бути закладена наступна інформація:

- товар (Ідентифікатор, ідентифікатор типу товару, назва товару, характеристика товару , ціна товару, гарантійний термін);

- постачальник (Ідентифікатор, ім’я, адреса, телефон);

- клієнт (Ідентифікатор, ім’я, адреса, телефон);

- постачання(Ідентифікатор, ідентифікатори постачальника, товару, дата постачання, кількість);

- склад (Ідентифікатор, ідентифікатор товару, ціна, кількість);

- замовлення (Ідентифікатор, дата, ідентифікатори товару, клієнта, ідентифікатор користувача, кількість);

- користувач (ідентифікотор, логін, пароль);

- тип товару (Ідентифікатор, назва типу товару).

Система повинна підтримувати наступні функції:

- організовувати функції реєстрації і авторизації в системі;

- організовувати функції вводу замовлень, редагування замовлень;

- організовувати функції вводу постачання і замовлення та їх редагування;

- організовувати функції виводу фінансових підсумків дня;

- організовувати функції виводу залишків товарів;

- організовувати функції виводу покупців, замовлень, постачань, товарів.

Система має виконувати наступні запити:

- вивід списку замовлень;
- вивід списку товарів (купівля/продаж);
- додавання замовлення;
- вивід всієї інформації про товар;
- купівля товару на склад;
- додавання товару;
- редагування товару;
- продаж товару із складу;
- вивід фінансових підсумків;
- вивід списку покупців товарів;
- аналіз об’ємів продажів товарів.

Головною задачею автоматизації системи є завдання впорядкування замовлень по групах та вивід про них повної інформації. Така задача не може бути вирішена простим шляхом виконання будь-яких запитів, а потребує розробки різноманітних аналітичних функцій.

Клієнтські частини повинні мати нескладний інтуїтивно зрозумілий інтерфейс. Крім того, клієнтські підсистеми повинні будуватися по відкритій архітектурі для забезпечення можливості автоматичного введення даних з різних пристроїв електронної реєстрації вимірів.

2 ПРАКТИЧНА ЧАСТИНА

2.1 Проектування бази даних

2.2.1 Виявлення набору сутностей для предметної області

При проведенні аналізу предметної області було вирішено обрати головною сутністю предметної області таблицю товар. Для опису даної предметної області було вирішено використати наступні сутності.

На рисунку 2.1 зображено таблицю, яка містить інформацію про товар, а саме: назву товару, його характеристику, ціну, гарантійний термін та зовнішній ключ на таблицю з типами товарів.

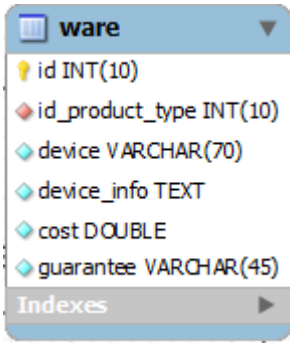

Рисунок 2.1 – Сутність – товар

На рисунку 2.2 зображено таблицю, яка містить інформацію про постачальників комп'ютерного магазину, а саме: назва постачальника, контактна особа, адреса постачальника, телефон постачальника.

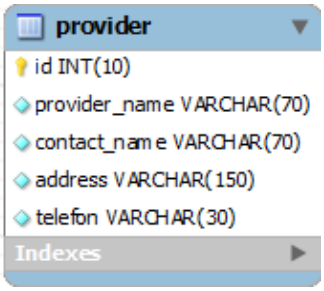

Рисунок 2.2 – Сутність – постачальник

На рисунку 2.3 зображено таблицю, яка містить інформацію про клієнтів комп’ютерного магазину і сюди відноситься: ім’я клієнта, адреса клієнта, телефон клієнта.

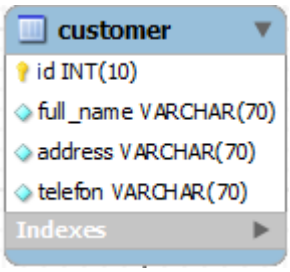

Рисунок 2.3 – Сутність – клієнт

На рисунку 2.4 зображено таблицю, яка містить інформацію про постачання товарів і сюди відносяться: дата приходу товару, кількість, зовнішні ключі на постачальника та на товар.

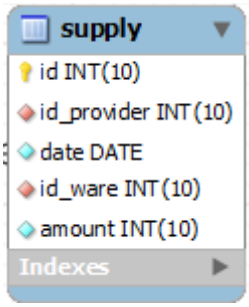

Рисунок 2.4 – Сутність – постачання

На рисунку 2.5 зображено таблицю, яка містить інформацію про склад комп’ютерного магазину і сюди відноситься: оптова та роздрібна ціна товару, кількість, зовнішній ідентифікатор на товар.

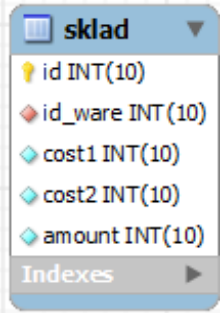

Рисунок 2.5 – Сутність – склад

На рисунку 2.6 зображено таблицю, яка містить інформацію про замовлення товарів і сюди відноситься: дата замовлення товару, кількість, зовнішній ключ на покупця та на товар.

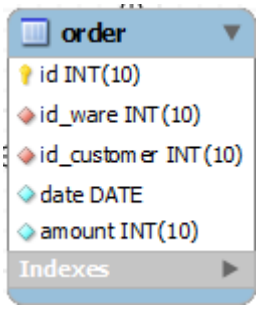

Рисунок 2.6 – Сутність – замовлення

На рисунку 2.7 зображено таблицю, яка містить інформацію про типи товарів і сюди відноситься назва типу товару.

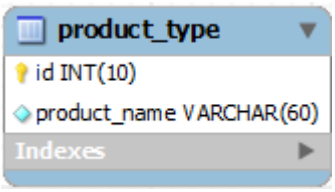

Рисунок 2.7 – Сутність – тип товар

Таким чином, описано всі сутності, які необхідні для подальшого виконання роботи по розробці програмного забезпечення магазину.

2.2.2 Побудова схеми бази даних

Після проведення аналізу сутностей була створена схема бази даних, зображена на рисунку 2.8.

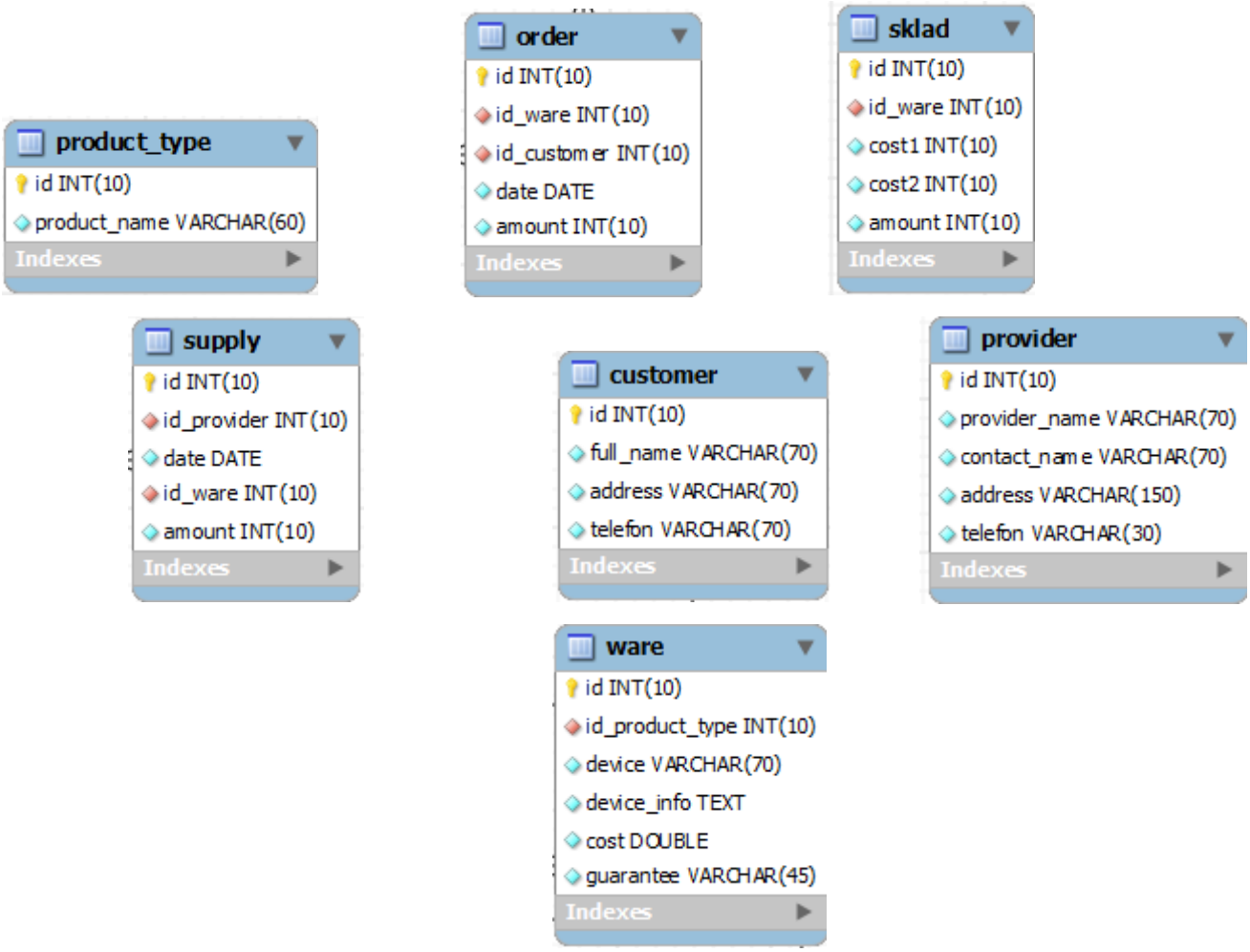

Рисунок 2.8 – Схема бази даних

Дана база даних приведена до третьої нормальної форми, не містить надлишковості та суперечливої інформації.

Нормальна форма (normal form) – властивість відношення в реляційній моделі даних, що характеризує його з погляду надлишковості, котра потенційно може спричинити пошкодження даних, втрату їх цілісності та актуальності.

Нормальна форма є сукупність певних вимог, яким має відповідати кожне відношення реляційної БД. Ці вимоги базуються на поняття функціональних залежностей. На кожному кроці, при переході докожної наступної нормальної

форми у відношеннях БД ідентифікуються функціональні залежності певного типу і усуваються шляхом декомпозиції відношень на більшу кількість. Таким чином отримаємо набір відношень (таблиць), кожне з яких перебуває у потрібній нормальній формі. Зауважимо, що, як правило, нормалізацію виконуютьдо 3НФ, оскільки значне збільшення кількості таблиць у БД спричиняє в майбутньому виконання великої кількості операцій з'єднання, що є вимогливими до обчислювальних ресурсів апаратної платформи, на котрій працює СКБД.

Отже, в базі даних потрібно зробити зміни, тобто створити зв’язки між таблицями.

Таблиця товарів (ware) містить інформацію про товар, а таблиця постачання(supply) містить інформацію про постачання цього товару. Тому потрібно зв’язати ці таблиці створивши в таблиці постачання(supply) зовнішній ключ id_ware, що посилається на первинне поле таблиці товарів(ware) id. Таблиця постачальник(provider) містить інформацію про постачальника. Тому потрібно зв’язати цю таблицю із постачання(supply), створивши в постачання(supply) зовнішній ключ id_provider , що посилається на первинний ключ таблиці постачальник(provider) id. Цей зв'язок можна побачити на рисунку 2.9.

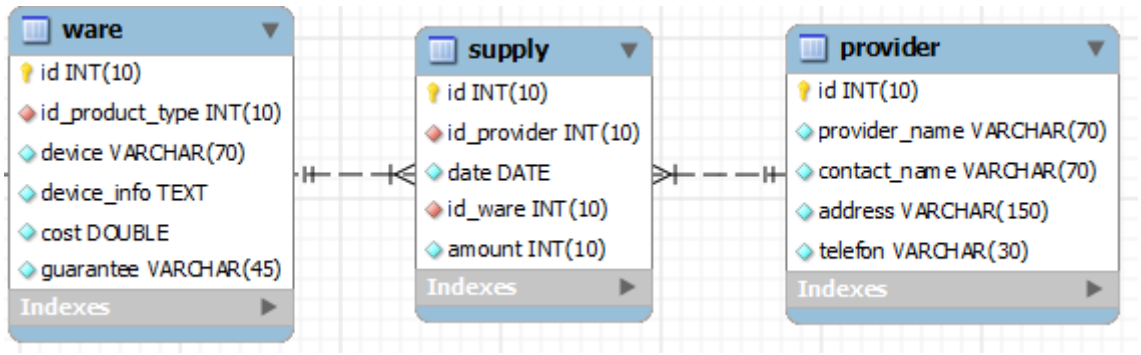

Рисунок 2.9 – Зв'язок між таблицею товарів(ware) , таблицею постачання(supply) та постачальник(provider)

Таблиця склад (sklad) містить інформацію про товари на складі. Тому потрібно зв’язати цю таблицю з таблицею товарів (ware), створивши в таблиці

склад (sklad) зовнішній ключ id_ware, що посилається на первинне поле таблиці товарів (ware) id. Цей зв'язок можна побачити на рисунку 2.10.

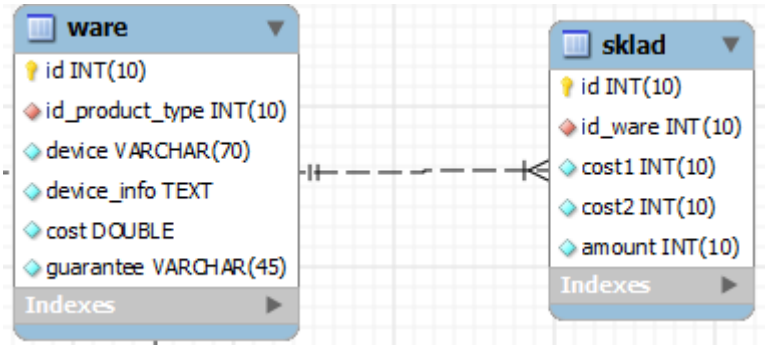

Рисунок 2.10 – Зв'язок між таблицею товарів(ware) та таблицею склад(sklad)

Таблиця замовлення(order) містить інформацію про замовленні товари (ware). Тому потрібно зв’язати цю таблицю з таблицею товарів (ware) створивши в таблиці замовлення (order) зовнішній ключ id_ware, що посилається на первинне поле таблиці товарів (ware) id. Цей зв'язок можна побачити на рисунку 2.11.

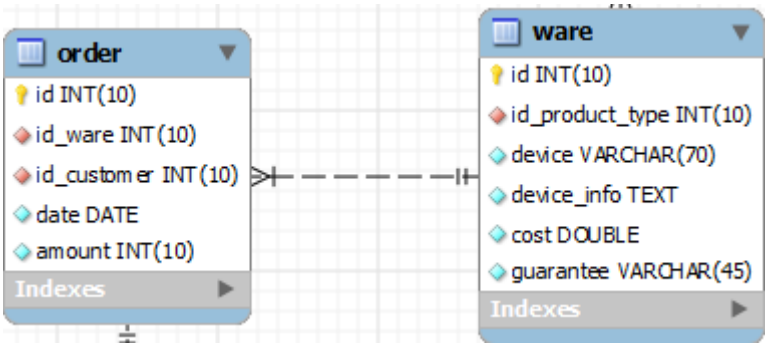

Рисунок 2.11 – Зв'язок між таблицею товарів(ware) та таблицею замовлення(order)

Таблиця замовник (customer) містить інформацію замовника чи покупця. Тому потрібно зв’язати цю таблицю з таблицею замовлення (order) створивши в таблиці замовлення (order) зовнішній ключ id_customer, що посилається на первинне поле замовник (customer) id. Цей зв'язок можна побачити на рисунку 2.12.

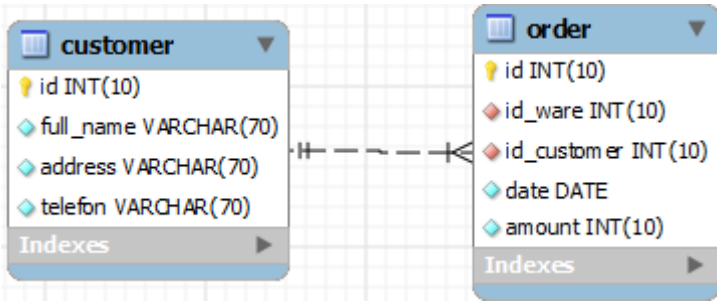

Рисунок 2.12 – Зв'язок між таблицею замовник (customer) та таблицею замовлення (order)

Отже, в результаті приведення бази даних до третьої форми було отримано базу даних, зображену на рисунку 2.13.

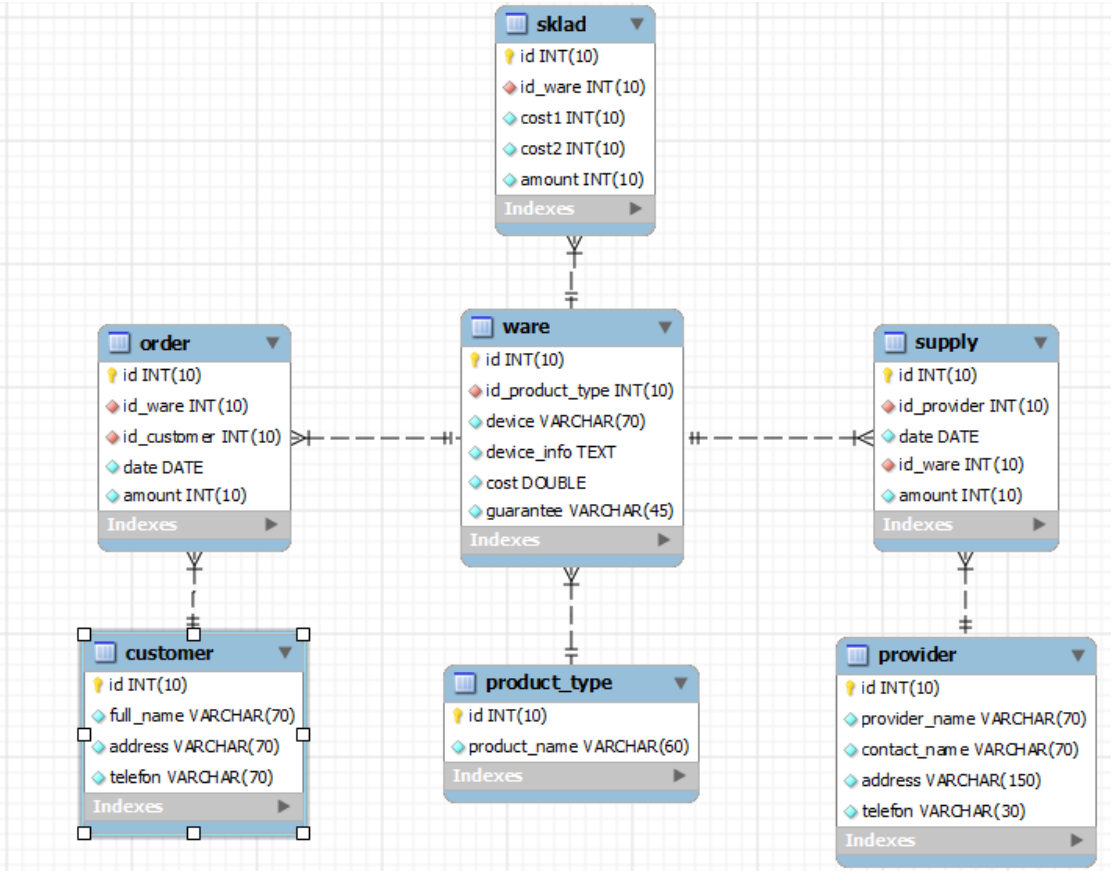

Рисунок 2.13 – Схема бази даних після приведення її до третьої нормальної форми

Дана база даних приведена до третьої нормальної форми. Відношення знаходиться в 1НФ, оскільки всі атрибути відношення є простими і не мають складових компонентів. Домени атрибутів складаються з неподільних значень і не включають множини значень з більш елементарних доменів.

Відношення знаходиться в 2НФ, оскільки воно знаходиться в першій нормальній формі і всі не ключові атрибути функціонально повно залежать від складеного ключа. Відношення не містить часткових функціональних залежностей.

Відношення знаходиться в третій нормальній формі, оскільки воно знаходиться в другій нормальній формі і всі не ключові атрибути відношення залежать лише від первинного ключа. Відношення не містить транзитивних функціональних залежностей не ключових атрибутів від ключа.

2.2.3 Обгрунтування вибору СУБД MySQL

MySQL – реляційна СКБД, що розповсюджується на основі безкоштовної ліцензії GPL від компанії SUN Microsystems. Ця СКБД була створена в якості альтернативи комерційним серверним СКБД і, таким чином, знайшла широке використання для мережі Інтернет в якості СКБД для динамічних порталів (магазинів, форумів, сервісів новин тощо). Початково вона мала мінімальний набір функцій, завдяки чому відрізнялась хорошими показниками продуктивності.

Пізніше СКБД отримала практично весь набір функціональних можливостей, властивих корпоративним серверам баз даних. Починаючи з версії 5, MySQL підтримує збережувані процедури, тригери, курсори, але залишаєтьяс невеликою компактною та швидкою СКБД для Інтернету.

MySQL рекомендується в якості СКБД для малих та средніх інформаційних систем, а розгоннути її можна практично на всіх сучасних апаратно-програмних платформах як для UNIX/Linux-систем, так для Windows, MacOS.

Найкраще можливості сервера проявляються для UNIX/Linux-систем, оскільки початково дана СКБД була створена саме для них.

MySQL є безкоштовним для некомерційного використання.

Перерахуємо основні функціональні можливості MySQL:

− простота установки та супроводу;

− підтримка (теоретично) необмеженої кількості клієнтських підключень, обмежена лише можливостями апаратної платформи сервера баз даних;

− до 50 млн. записів у кожній таблиці;

− ефективне і швидке виконання запитів;

− реалізовано просту та ефективну систему захисту сервера на основі розподілу привілеїв між користувачами.

До основних плюсів MySQL можна віднести високу швидкість роботи, швидкість обробки даних і оптимальну надійність. Важливо й те, що дана СУБД поширюється безкоштовно і представляє собою програмне забезпечення з відкритим кодом. Існує безліч СУБД, які підтримують SQL мову запитів: MySQL, mSQL, PostgreSQL, MSSQL і багато інших. Кожна з них має переваги в певній сфері. І все ж саме MySQL завоювала широке визнання і популярність в Інтернеті завдяки своїй гнучкості та універсальності.

Типово MySQL застосовується у якості сервера, до якого підключаються клієнтські програми. Але дистрибутив даної СКБД включає і локальну реалізацію "движка" БД, що дозволяє включати його в дистрибутиви прогнрамних продуктів, що працюють локально і володіють властивостями портативності. MySQL є гнучким, оскількт від підтримує різні типи таблиць, що відрізняються формою зберігання інформації на диску і, відповідно, найкраще підходять для різних варіантів використання та клієнтської бізнес-логіки. По замовчуванню сервер надає користувачеві таблиці MyISAM, що підтримують можливість повнотекстового пошук даних. В таблицях типу InnoDB реалізована підтримка транзакції на рівні окремих записів таблиці. Через те, що арітектура сервера відкрита та розповсюджується він на основі GPL-ліцензії в цій СКБД з'являються і інші, нові, типи таблиць.

СУБД MySQL реалізує тип таблиць, які постійно зберігаються в ОЗП і тому завжди доступні за мінімальний час. Це MEMORY, ще є синонім HEAP. Друга назва старіша, тому переважно використовується перша. В порівнянні з MyISAM

або INNODB, цей формат сильно обмежений, але із завданням зберігання оперативних даних справляється прекрасно. Плюсами є: будь-які запити виконуються максимально швидко – дані вже в пам'яті, таблиці швидко створюються і швидко знищуються, можливість обмежити об'єм кожної таблиці, підтримуються блокування.

Архітектура зберігання даних в MySQL дозволяє професіоналу бази даних вибирати спеціалізований тип пам'яті для специфічної потреби прикладної програми. Сервер MySQL ізолює прикладного програміста і DBA від всіх подробиць реалізації низького рівня пам'яті, забезпечуючи несуперечливу і просту модель прикладної програми і API. Таким чином, хоча є різні можливості різних типів пам'яті, прикладна програма захищена від цих відмінностей.

Архітектура пам'яті MySQL, що підключається, є компонентом сервера бази даних, який є відповідальним за виконання фактичних операції введення-виведення даних для бази даних, а також надання і розпорядження деяких наборів властивостей, яких потребує специфічна прикладна програма. Головна користь в тому, що у будь-який момент використовується те, що зручне, витрачаючи мінімум зусиль і економлячи багато ресурсів системи сервера. У MySQL 5.1 MySQL AB представила нову знімну архітектуру пам'яті, яка дозволяє завантажувати і вивантажувати типи пам'яті (раніше відомі як драйвери таблиць) у міру потреби, не перезапускаючи сервер. MySQL підтримує наступні типи пам'яті: MyISAM, InnoDB, Memory, Archive, NDB.

Перевагами СКБД MySQL є:

- підтримка виконання декількох запитів одночасно завдяки багатопотоковості;

- оптимізація виконання запитів з прогностичним виконання з'єднання таблиць для отримання інформації з декількох з них одночасно;

- підтримка записів як фіксованої довжини, так і змінної;

- система привілеїв користувачів на основі їх автентифікації на основі паролів;

- Підтримка до 16 унікальних ключів для кожної таблиці; кожен ключ може містити до 15 полів;

- підтримка числових значень різного розміру від 1 до 4 байт, стрічок VARCHAR та часових міток;

- можливість керувати вріхуванням реєстру символів при викоанні операцій над стрічками;

- моживість зміни структури таблиць БД.

Важливо згадати, що СКБД MySQL має набори функцй програмного інтерфейсу (API) для різних сучасних мов програмування, в тому числі C/C++, Java, Lisp, PHP, Python, Ruby, Smalltalk та інші. Для платформи .NET також існують бібліотеки програмного інтерфейсу. Всі вони доступні для вільного завантаження. Реалізована також підтримка ODBC-драйвера MyODBC.

Як згадувалось вище таблиця MySQL може мати до 16 унікальних ключів, в кожен з яких може входити до 15 стовпців таблиці. Слід пам'ятати, що великі ключі можуть привести до зниження ефективності роботи сервера баз даних. Ключам можна давати імена, що є зручним для доступу доних з інших сутностей БД та прикладних програм.

Первинний ключ задається опцією PRIMARY. Якщо ключа створено без імені, то при створенні таблиці сервер БД створить це ім'я на основі імені першго стовпця, що входить до цього ключа з добавленням суфіксу - числа (_2, _3, і т. д.), для забезпечення унікальності цього імені. Такі імена можуть служити дл звертання до ключів, наприклад, при зміні структури таблиці чезе запит ALTER TABLE, щоб його видалити.

Як і в багатьох інших серверних СКБД при потребі отримати ьільки частину даних, що задовільняють усові запиту вибірки в MySQL застосовується опція LIMIT з вказанням кількості рядків, які потрібно включити у результутючий набір даних після виконання запиту SELECT. Як правило, ця опція корисна при тестування роботи прикладної програми або при необхідності виводити інформацію порціями, наприклад, на веб-сторінку. Тоді разом з LIMIT крім

кількості рядків вказують ще одне число, яке означає кількість рядків, які потрібно пропустити від початку.

2.2.4 Установка MySQL

Спочатку сервер MySQL потрібно завантажити з офіційного сайту, його розділу Завантаження. Виберіть версію MySQL Community Server, який є необхідним, а поряд – платформу, на якій ви будете його встановлювати.

Встановлення MySQL на Linux / UNIX.

Рекомендується спосіб, щоб встановити MySQL на в Linux системі – з допомогою RPM. Доступні такі пакети:

– MySQL -сервер – власне сервер, який управляє з базами даних і таблицями, містить елементи управління для користувача і обробляє в SQL запити.

– MySQL-client - програми, які підключаються до сервера і взаємодіють з ним.

– MySQL-devel - Бібліотеки і заголовочні файли, які необхідні для компіляції інших програм, що працюють з MySQL.

– MySQL-shared - Загальні бібліотеки для в MySQL клієнта.

– MySQL-dench - Benchmark для тестування інструментів для в MySQL.

В MySQL Обороти, перераховані тут будуть всі побудовані на більш SuSE Linux системи, але вони будуть, як правило, працюють на інших Linux варіанти з НЕ працею.

Тепер, виконати кроки з установки сервера:

– Ввійти в системі з привілеями root.

– Перейти в в каталог, що містить RPMи.

– Встановити сервер, виконавши команду:

```
[ root @ host ] # rpm -i MySQL-5.0.9-0.i386.rpm
```

Вище команда приймає турботу про встановлення MySQL, створюючи на користувача в MySQL, створення необхідної конфігурації і запускає MySQL сервер автоматично.

Ви можете знайти всі пов'язані з MySQL двійкові файли в / usr / bin та / usr / sbin. Всі ці таблиці і бази даних будуть створені в / var / lib / sbin каталозі.

Наступні команди покуазують, як встановити додаткові елементи сервера при потребі.

```
[ root @ host ] # rpm -i MySQL-client-5.0.9-0.i386.rpm
[ root @ host ] # rpm -i MySQL-devel-5.0.9-0.i386.rpm
[ root @ host ] # rpm -i MySQL-shared-5.0.9-0.i386.rpm
[ root @ host ] # rpm -i MySQL-bench-5.0.9-0.i386.rpm
```

Встановлення MySQL на Windows.

За замовчуванням установка на будь-яку версію в Windows, є в даний час набагато простіше. Просто потрібно завантажити в інсталяційний пакет, розпакувати його в будь-якому місці і запустити в setup.exe файл.

За замовчуванням установник setup.exe проведе користувача через процес установки і по замовчуванням буде встановлювати все в каталозі C: \ MySQL.

Перевірка встановлення MySQL.

Після того, як MySQL, вже була успішно встановлена, то базові таблиці вже були ініційовані і сервер вже працює: ви можете переконатися, що все буде працювати, як це повинно бути з допомогою деяких простих тестів.

Використовуйте в mysqladmin утиліти для того, щоб отримати статус про стан сервера. Використовуйте mysqladmin двійковий файл для перевірки на сервер версії.

Кроки після встановлення.

MySQL поставляється з порожнім паролем для головного користувача MySQL. Звичайно, варто встановити цей пароль.

```
[ root @ host ] # mysqladmin -u root пароль " new_password ";
```

2.2.5 Інтерфейс доступу JDBC

Інтерфейс JDBC фактично стандартним методом доступу до реляційних СУБД з програм, написаних мовою Java. Він був створений по аналогії до специфікації специфікації ODBC (Open Database Connectivity). Але оскільки ODBC написаний на мові C, то його API проблемно адаптувати до програм на Java, оскільки C реалізує арифметику вказівників, а в Java це не працює. Таким чином JDBC підтримує підключення до сервера БД, в тому числі і до MySQL та дозволяє виконувати запити мовою SQL.

Завдяки пакетові бібілотек драйверів JDBC мова Java може застосовуватись длян написання додатків, що потребують підключення до баз даних, без заучення інших мов програмування і, відповідно, додаткових затрат на інтерфейси між програмами на різних мовах програмування.

JDBC API складається з двох частин: API-інтерфейсу виского рівня, який містить бібліотеки функцій, що використовуються для створення прикладних програм мовою Java.

Інший API-ітерфейс – це інтерфейс низького рівня, що служить власне для створення драйверів для підключення до конкретних СКБД. Додатки отримують доступ до БД через драйвери ODBC та наявні API-клієнтські бібліотеки. Або ж з цією метою служить інтерфейс JDBC з драйверами на чистій Java без проміжних елементів ODBC. Все ж перерахуємо основні способи організаціє підключення до БД через JDBC:

- міст JDBC-ODBC. Ця версія засобів доступу була розроблена в середині 1996 року компаніями Sun і Intersolv. Вона передбачає доступ до бази даних через інтерфейс JDBC за допомогою драйверів ODBC. В цьому випадку драйвери ODBC виконують функції посередника між драйвером JDBC і клієнтськими бібліотеками розробника бази даних. Для цього необхідно, щоб двійковий код драйвера ODBC (а у багатьох випадках і деяке клієнтське програмне забезпечення для роботи з базами даних) був завантажений на кожному клієнтському комп'ютері, що використовує цей драйвер, тому драйвер такого типу може застосовуватися в

Internet лише в обмеженій мірі. Цей підхід приводить також до деякого зниження продуктивності через додаткове перетворення даних при прямій і зворотній передачі від драйвера JDBC до драйвера ODBC, і у зв’язку з цим може виявитися неприйнятним для створення великомасштабних застосувань. До того ж цей підхід не забезпечує підтримку всіх засобів мови Java, і можливості програми обмежуються лише тими, що надає відповідний драйвер ODBC. Але даний підхід має також значну перевагу в тому, що в даний час драйвери ODBC поширені буквально повсюди.

- драйвер JDBC, лише частково написаний на мові Java. Драйвер такого типу перетворить виклики JDBC у виклики функцій клієнтських API-інтерфейсів використовуваної СУБД. Драйвер взаємодіє безпосередньо з сервером бази даних і тому на кожному клієнтському комп'ютері має бути завантажене деяке клієнтське програмне забезпечення для роботи з базами даних. Таким чином, цей підхід також характеризується обмеженими можливостями його використання для Internet, але може успішно застосовуватися для створення додатків, які експлуатуються у внутрішніх мережах. Драйвер такого типу забезпечує вищу продуктивність в порівнянні з мостом JDBC-ODBC.

- повністю написаний на мові Java драйвер JDBC для проміжного рівня програмного забезпечення СУБД. Цей драйвер перетворить виклики JDBC в повідомлення протоколу проміжного рівня, які потім перетворяться сервером проміжного рівня в повідомлення протоколу СУБД. Проміжний рівень забезпечує з'єднання з багатьма різними базами даних. В цілому це найбільш гнучке зі всіх альтернативних рішень з використанням пакету JDBC. Мабуть, що всі розробники подібних рішень незабаром нададуть продукти, призначені для доступу до даних у внутрішній мережі. Для підтримки відкритого доступу в мережі Internet вони повинні забезпечити виконання додаткових вимог до захисту (наприклад, доступ через брандмауери). Деякі компанії-розробники додали драйвери JDBC до своїх вже існуючих програмних продуктів проміжного рівня для роботи з базами даних. Драйвери такого типу можуть виявитися найбільш відповідними для середовища, в якому необхідно забезпечити зв'язок з багатьма серверами СУБД і різнотипними

базами даних, а також потрібна підтримка значної кількості одночаснопрацюючих користувачів. У подібному середовищі важливе значення приймають завдання забезпечення високої продуктивності і масштабування.

- написаний повністю на мові Java драйвер JDBC з прямим підключенням до бази даних. Цей драйвер перетворить виклики JDBC в повідомлення мережевого протоколу, який використовується безпосередньо СУБД. При цьому допускається виконання прямих викликів з клієнтського комп'ютера, що передаються на сервер СУБД. Такий метод являє собою зручне практичне рішення для організації доступу до СУБД в Internet і дозволяє динамічно завантажувати необхідні драйвери. Драйвери такого типу реалізуються повністю на мові Java, завдяки чому досягається незалежність від платформи і усуваються проблеми розгортання.

Перевага використання драйверів ODBC полягає в тому, що вони фактично стали стандартом доступу до БД з персональних комп’ютерів і їх варіанти є для більшості найпопулярніших СУБД, причому за досить низькою ціною.

Проте останній підхід має наступні недоліки:

- той драйвер JDBC, який не повністю написаний на мові Java, не завжди працюватиме з будь-яким браузером;

- в даний час (з міркувань захисту) завантажений з мережі Internet аплет може підключатися лише до бази даних, що розміщена на тому ж комп'ютері, з якого цей аплет був отриманий;

- витрати на розгортання додатків зростають через необхідність встановлювати, налаштовувати і супроводжувати на кожному клієнтському комп’ютері деякий набір драйверів, а в разі перших двох підходів – завантажувати і програмне забезпечення бази даних.

З іншого боку, драйвер JDBC, повністю написаний на мові Java, може бути завантажений разом з аплетом.

Хоча в більшості реляційних СУБД для виконання основних функцій використовується стандартна форма мови SQL, в них, як правило, не підтримуються складніші функції, що з'явилися останнім часом. Наприклад, не у всіх реляційних СУБД забезпечується підтримка процедур або виконання

зовнішніх з'єднань, що зберігаються. А якщо ці функції підтримуються СУБД, то найчастіше використовувані методи несумісні один з одним. Інтерфейс JDBC API був задуманий саме для підтримки різних діалектів SQL.

Одним із способів вирішення цієї проблеми за допомогою інтерфейсу JDBC API є передача будь-якого рядка запиту у відповідний драйвер СУБД. Це означає, що додаток може абсолютно вільно використовувати будь-які необхідні функціональні засоби мови SQL, хоча при роботі з деякими типами СУБД це може привести до появи повідомлень про помилку. Насправді запит може бути виражений взагалі не на мові SQL, або це може бути якийсь особливий різновид мови SQL, розроблений спеціально для СУБД деякого конкретного типу. Крім того, пакет JDBC підтримує управляючі послідовності в стилі ODBC. Синтаксис управляючих послідовностей забезпечує стандартний синтаксис пакету JDBC для декількох найпоширеніших різновидів мови SQL. Існують, наприклад, управляючі послідовності для строкових значень дат і викликів процедур, що зберігаються.

У складних застосуваннях пакет JDBC здатний підтримувати сумісність cSQL третім способом. Він надає описову інформацію про використовувану СУБД за допомогою інтерфейсу DatabaseMetaData, що дозволяє додатку адаптуватись до вимог і можливостей кожної конкретної СУБД.

Для вирішення проблеми сумісності з Java компанія Sun ввела позначення "JDBC Compliant ™" (сумісний з JDBC) для опису стандартного набору функціональних засобів JDBC, на наявність якого може розраховувати користувач. Драйвер може отримати таке позначення лише в тому випадку, якщо він підтримує щонайменше початковий рівень специфікації ANSI Sql2. Для перевірки сумісності з інтерфейсом JDBC API розроблено спеціальний тест.

2.3 Опис програмного забезпечення

2.3.1 Загальний опис програмного продукту

Даний програмний продукт створювався на мові Java в середовищі NetBeans IDE 7.3. Мова Java була обрана завдяки її великому функціоналу для роботи з

користувацьким інтерфейсом. В якості системи управління базою даних була вибрана СУБД MySql. Дана СУБД була вибрана завдяки тому, що вона невимоглива до ресурсів системи, некомерційна, володіє достатньою кількістю адміністративних функцій, як для розроблюваної бази даних.

Для нормального функціонування системи необхідні такі параметри персонального комп’ютера:

– ЦП з тактовою частотою не менше 1.8 GHz;

– ОЗП не менше 512 Mb;

– місце на HDD не менше 200 Mb дискового простору.

Мова програмування Java служить для написання програм найрізноманітнішого призначення. Програмний код компілюється в байт-код, що виконується на так званій Java-машині.

Багаторічний досвід використання цієї мови свідчить, що вона проста, потужна і в той же час гнучка, через що здобула широку популярність, в т.ч. і для написання мобільних додатків. Це чисто об'єктно-орієнтована мова без підтримки множинного наслідування, типізована, з пібтримкою мультипотоковості та динамічного зв'язування.

Стинтаксис Java базується на основі синтаксису C і C++. Цілий ряд операцій, котрі в програмах на C/C++ мають виконувати розробники, в програмах на Java, внутрішню кухню по очищенню динамічної пам'яті виконує машина Java. Передусім, Java створена, як незалежна від платформи мова, тому в ній відсутня велика кількість низькорівневих операцій на зразок роботи з вказівниками чи доступ напряму до апаратного забезпечення. При необхідностиі здійснення таких операцій потрібно використовувати інші мови, а з програми на Java викликати ці модулі. Java має такі можливості.

Під поняттям «незалежність від платформи» мається на увазі, що програма, створена мовою Java, буде златна працювати на будь-якій апаратній та/чи програмній платформі без необхідності переписування програмного коду і повтоної компіляції . Це реалізовано шляхом компіляції програмного коду на Java у так званий байт-код, є не чим іншим, як машинними командами. Цей байт код

виконується не безпосередньо на проценсорів , а в Java-машині, котра надає додатковий рівень абстракції та ізолює програму від особливостей апаратного і програмного забезпечення комп'ютера.

Далі байт-код інтерпретується віртуальною машиною Java. Головне, щоби ця машина була встановлена. ЗауважиомЮ що подібну ідею використовує в своїй еко-системі програмних продуктів корпорація Microsoft у платформі .NET.

На сьогодні віртуальні машини Java доступні практично для всіх сучасних процесорів і операційних систем.

Стандартні бібліотеки мови "з коробки" реалізують доступ до графічної системи базової платформиі надають програмістові засоби для опрацювання графіки, реалізації багатопотоковості, передачу даних по мережі. Для збільшення продуктивності деякі програмні платформи дозволяють компілювати байт-код у команди процесора безпосередньо перед запуском програми або в ході її виконання.

Головною перевагою перетворення програми у байт-код є досягення його портативності. Однак є і негативний бік такого підходу. Додаткові затрати часу та обчислювальних ресурсів означають, що програми на мові Java будуть працювати повільніше ніж ті, що компілюються безосередньо в команди процесора. Проте після оптимізації сучасних JVM розрив між такими програмами та програмами на Java суттєво скоротився в плані швидкодії.

Швидкість JVM значно виросла від моменту ранніх версій. Проте не завжди твердження про те, що компілятор шивдкий, означає відповідний виграш в продуктивності програми на Java. Це може підтвердити лише ретельне тестування, оскільки продуктивність програми сильно залежить також від якості написання програмного коду, ефективності реалізації алгоритмів.

У Java через чистий об'єктно-орієнтований підхід всі змінні є об'єктами, успадкованими від головного об'єкта Object, з котрого наслідуються базові дя всіх поведінка (методи) і властивості. Java підтримує тільки однарне наслідування, множинне не підтримується. Тому конфлікти між членами класучерез невизначеність походження від базових класів виключені.

Програмна платформа Java – це ім'я для пакету програм компанії Sun, які дозволяють розробляти і запускати програми, написані на мові програмування Java. Вона не є специфічною для якого-небудь одного процесора або операційної системи, але механізм виконання (так звана віртуальна машина) і компілятор з набором бібліотек, які реалізовані для різного апаратного забезпечення і різних операційних систем, щоб Java-програми могли працювати всюди однаково.

2.3.2 Опис архітектури програмної системи

В даній системі немає ієрархії класів, тобто всі об’єкти функціонують окремо і використовують функції інших класів через їхні об’єкти. Кожен клас відповідає за певний функціонал системи, зокрема :

- MySQL – містить функції для з’єднання та роботою з БД, а саме додавання редагування та видалення записів відповідних БД;
- AutorizationDialog – забезпечує авторизацію в системі ;
- RegistrationDialog – клас для реєстрації нового користувача в системі;
- addDevice – даний клас дозволяє добавляти новий товар в БД;
- addProductType – даний клас дозволяє добавляти новий тип товару в БД;
- addProvider – забезпечує додавання нового постачальника в БД;
- addSupply – дозволяє оформити нову поставку товару;
- customers – додавання нового клієнта;
- providers – дає можливість переглянути список постачальників;
- searchDialog – клас для пошуку товару за приблизними характеристиками;
- sklad – надає можливість переглядати наявний на складі товар;
- sold – забезпечує реалізацію товарів;
- total – клас для підбиття підсумків роботи за введений період.

У кожному класі створені об’єкти:

- db для використання функцій класу MySQL які працюють з базою даних MySQL;
- message класу Message для виводу повідомлень.

Основними функціями даної системи є:

- newSupply () – функція реалізована в класі MySQL. Нове замовлення на постачання обраного товару та обраним постачальником;
- connectToDB () – функція реалізована в класі MySQL. Встановлює зв'язок з базою данних;
- newSold () – функція реалізована в класі MySQL. Здійснює продаж товару;
- getSupply () – функція реалізована в класі MySQL. Повертає інформацію про замовлення;
- dototalActionPerformed() – функція реалізована в класі total. Здійснює підсумок роботи за певний період.

2.3.3 Опис програмної реалізації

Найбільша частина коду реалізації роботи з базою даних знаходиться в класі MySql. В цьому класі знаходяться методи підключення до бази даних, реєстрації користувачів в системі, методи додавання постачаня, замовлення, товару, покупців, постачальників, виведення інформації з БД, підбиття підсумків.

Лістинг 2.1 – Функція підключення до бази даних

```
public void connectToDB() {
        try {

Class.forName("com.mysql.jdbc.Driver").newInstance();
            } catch (Exception ex) {
                ex.printStackTrace();
            }
            String url =
"jdbc:mysql://localhost:3306/computer_shop?useUnicode=true&character
Encoding=UTF-8";
            String login = "root";
```

```
            String passwd = "";
            try {
                connect = DriverManager.getConnection(url, login,
passwd);
            } catch (SQLException ex) {
                ex.printStackTrace();
            }  }
```

Лістинг 2.2 – Функція авторизації в програмі

```
      public int getLogin(String login, String password) {
            int userId = 0;
            String query = "SELECT users.id, usertypes.type FROM
users LEFT JOIN usertypes ON users.userType = usertypes.id WHERE
login = '" + login + "' AND userPassword = '" + password + "'";
            try {
                Statement stm =
connect.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
ResultSet.CONCUR_READ_ONLY);
                result = stm.executeQuery(query);
                while (result.next()) {
userId = Integer.parseInt(result.getString("id")); }


     stm.close();
            } catch (SQLException ex) {
                message.getMessage("Помилка при виборі користувача
з бази даних!\nSqlQuery: " + query, null);
            }
            return userId;
        }
```

Лістинг 2.3 – Функція оформлення продажу товару

```
    public void newSold(String custName, String dev, String
devInfo, String date, String amount) {
            try {
```

```
    result = getQuery("SELECT  * from customer where full_name like
'" + custName + "'");
                result.next();
                String id_customer = result.getString("id");
                result = getQuery("SELECT  * from ware where device
like '" + dev + "' and device_info like '" + devInfo + "' ");
                result.next();
                String id_ware = result.getString("id");
                //враховувати кількість...ціну оптову
                result = getQuery("SELECT  * from sklad where
id="+id_ware );
                result.next();
                String cost = result.getString("cost2");
                stm.executeUpdate("insert into
computer_shop.order(id_ware,id_customer,date,amount,cost) " +
                        " values('" + id_ware + "','" + id_customer
+ "','" + date + "','" + amount + "','"+cost+ "')");
                stm.executeUpdate(" delete from sklad where
id_ware=" + id_ware);
            } catch (SQLException ex) {

Logger.getLogger(MySQL.class.getName()).log(Level.SEVERE, null, ex);
            }
     }
```

Лістинг 2.4 – Функція повертає список постачань товару

```
    public ArrayList<String> getSupply() {
            ArrayList<String> supply = new ArrayList();
            supply.add("-Вибрати-");
            try {
                result = getQuery("SELECT supply.id, supply.date ,
provider.provider_name from supply left join provider on
supply.id_provider=provider.id");
                while (result.next()) {
```

```
                    String supplyName =
result.getString("provider_name") + "-" + result.getString("date");
                    supply.add(supplyName);
                }
            } catch (SQLException ex) {
                JOptionPane.showMessageDialog(null, "SQLException:
" + ex.getMessage() + "\nSQLState: " + ex.getSQLState() +
"\nVendorError: " + ex.getErrorCode());
            }
            return supply;  }
```

2.3.4 Опис задач автоматизації та інтерфейсу користувача

Після запуску програма має вигляд який зображено на рисунку 2.14.

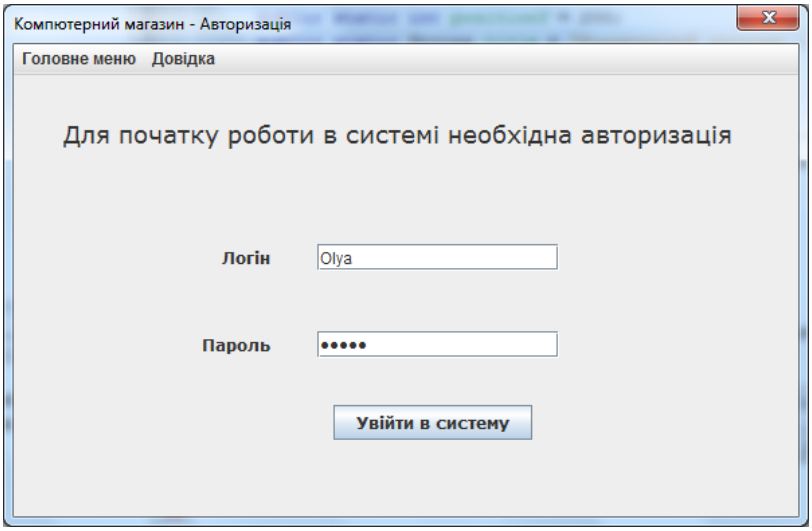

Рисунок 2.14 – Програма після запуску

Зареєстровані користувачі системи авторизуються ввівши логін і пароль. Якщо користувач не зареєстрований, то він може зареєструватись перейшовши на вікно реєстрації, що зображено на рисунку 2.15:

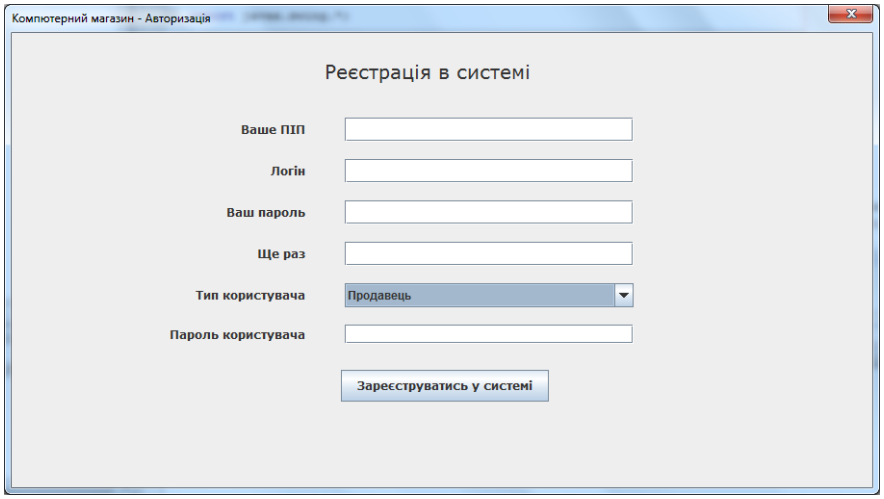

Рисунок 2.15 – Вікно реєстрації користувача в системі

Заповнивши відповідні поля та натиснувши кнопку “Зареєструватись у системі”, зареєстрований користувач матиме змогу увійти в систему, використовуючи введений логін та пароль при реєстрації.

У вікні програми, що зображене на рисунку 2.16, є змога ,заповнивши форму, додати товар до бази даних натиснувши кнопку “Додати”. Обов’язковим є заповнення усіх полів. Якщо будь-яке поле виявиться пустим, виведеться відповідне повідомлення з проханням заповнити пропущене поле.

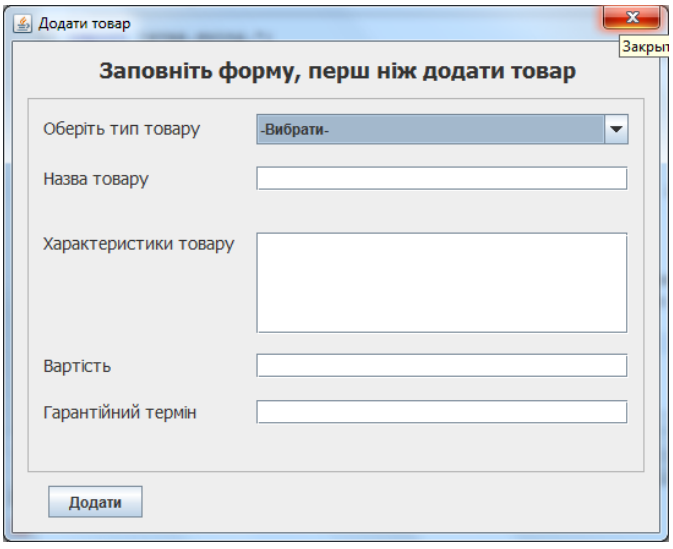

Рисунок 2.16 – Вікно для додавання товару

За допомогою вікна, зображеного на рисунку 2.17, є змога додати до бази данних магазину постачальника товарів. Заповненість всіх полів є також обовязковою і контролюється.

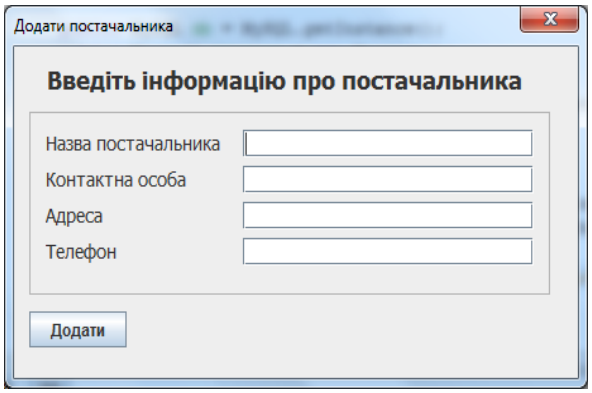

Рисунок 2.17 – Вікно для додавання постачальника

Вікно, зображене на рисунку 2.18, дає можливість оформити замовлення на поставку певного товару. Із списку товарів необхідно обрати той, який є потреба замовити. Також необхідно вказати самого постачальника , кількість і ціни. Після заповнення необхідних полів натиснувши “Додати поставку” оформляємо замовлення.

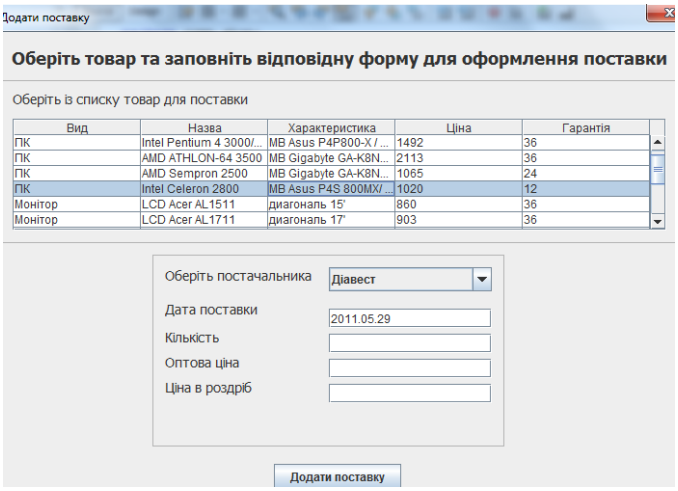

Рисунок 2.18 – Вікно для оформлення поставки товарів

Вікно, зображене на рисунку 2.19, дає можливість переглянути список покупців товарів.

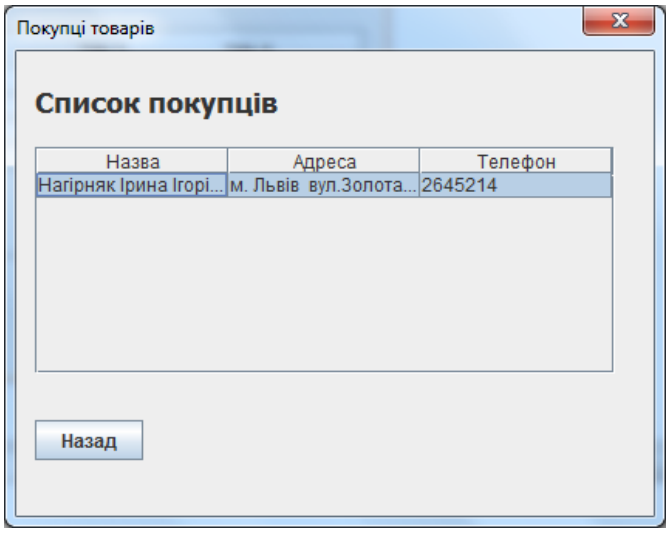

Рисунок 2.19 – Вікно для виведення інформації про покупців

Вікно, зображене на рисунку 2.20, дає можливість переглянути список постачальників товарів.

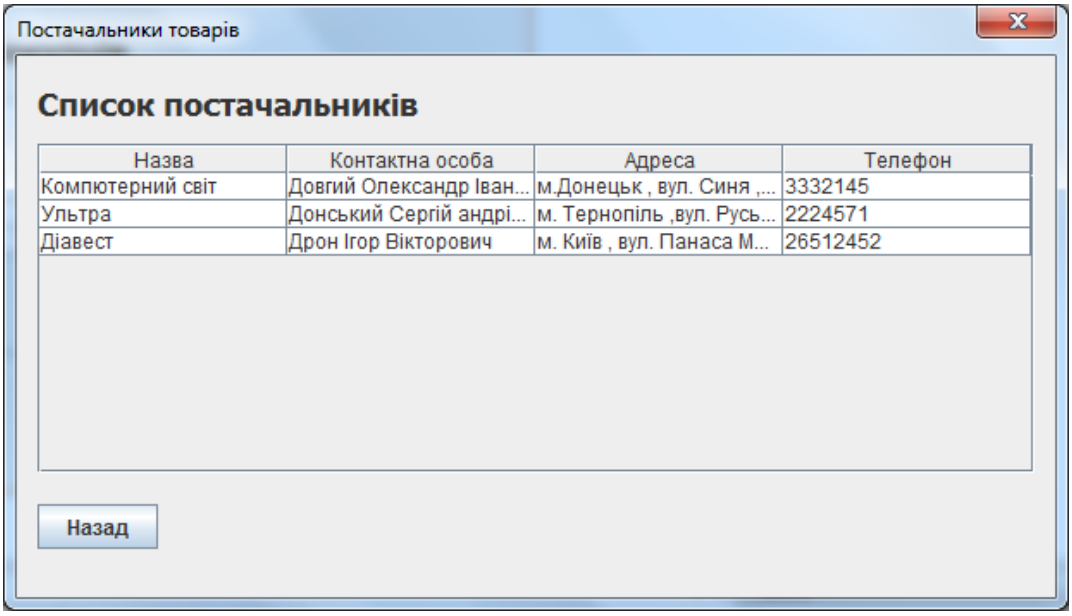

Рисунок 2.20 – Вікно для виведення інформації про постачальників

Вікно, зображене на рисунку 2.21, дає змогу здійснювати пошук товару. Ввівши приблизну назву чи характеристики шуканого товару, натискаємо кнопку “Знайти” для пошуку товару. Користувач має змогу обрати за якими критеріями

здійснювати пошук, поставивши чи забравши відмітку навпроти необхідного критерію пошуку.

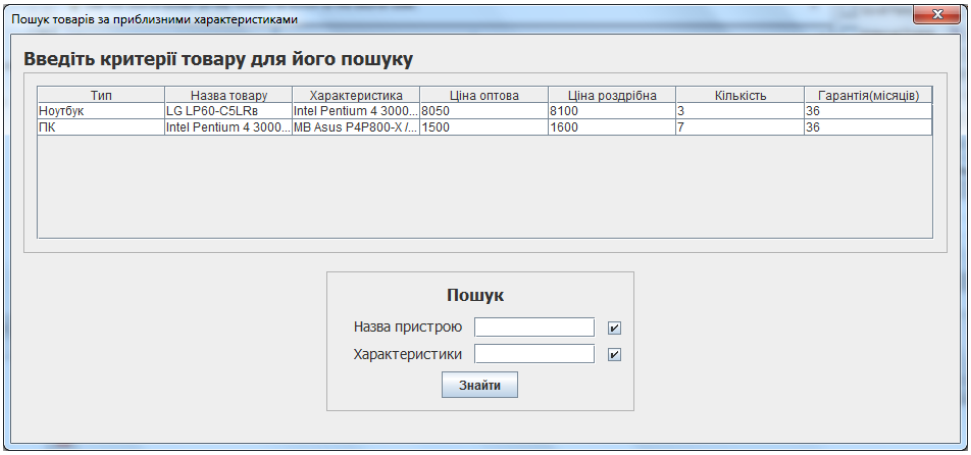

Рисунок 2.21 – Пошук товарів за приблизними критеріями

Вікно, зображене на рисунку 2.22, дає змогу переглядати інформацію про наявний на складі товар та його характеристики. Також користувач має змогу сортувати інформацію за зростанням чи спаданням одного з трьох характеристик товару, а саме назви , оптової ціни та кількості.

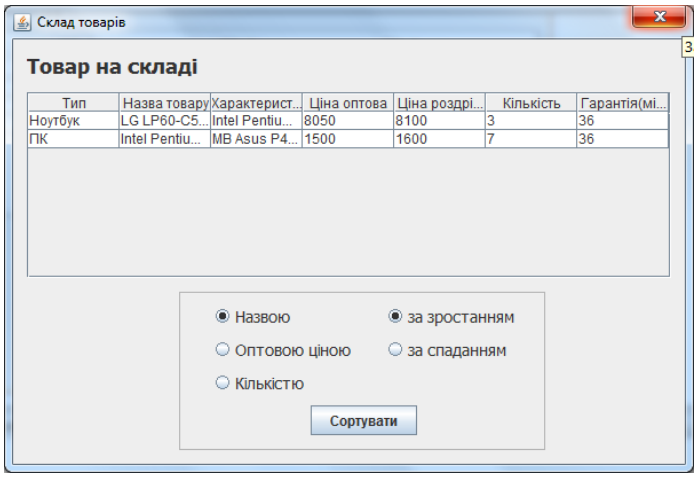

Рисунок 2.22 – Вивід інформації про наявний на складі товар

Вікно, зображене на рисунку 2.23, дає змогу переглянути поступлені та продані товари за певний період і зробити підсумки роботи, обрахувавши вартість поступленого та проданого за цей період товару.

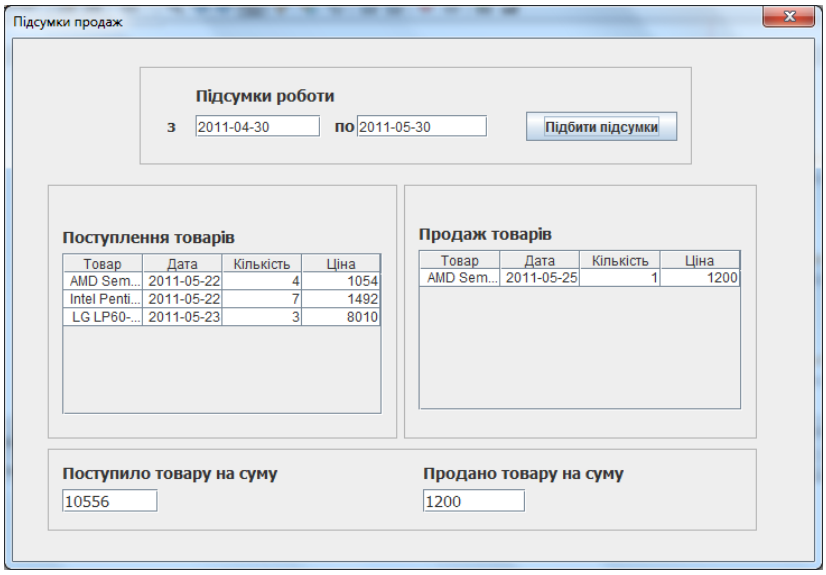

Рисунок 2.23 – Підбиття підсумків за певний період

Вікно, що зображене на рисунку 2.24, дає змогу знайти та реалізувати товар. Знайшовши товар обираємо його у списку. Заповнюємо інформацію про клієнта та натискаємо кнопку “Продати”.

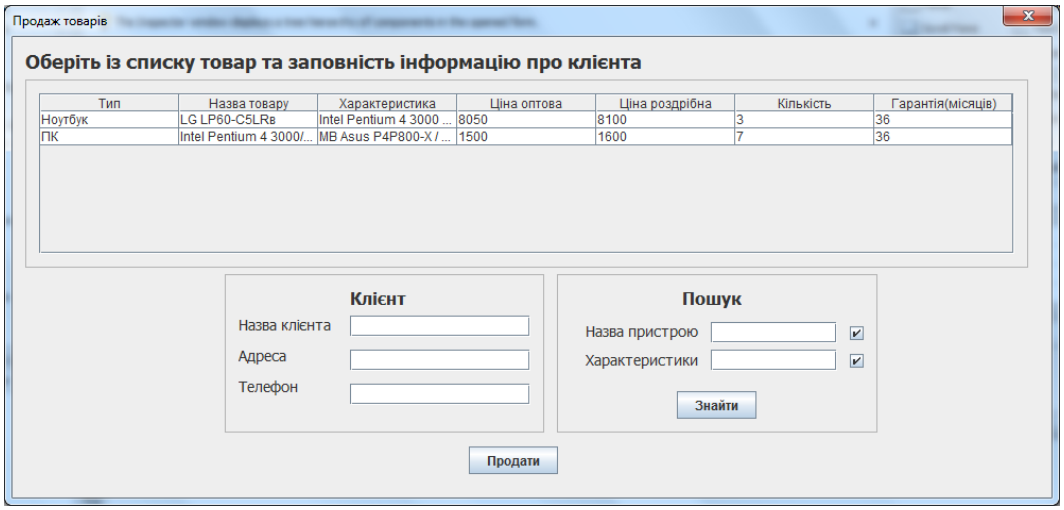

Рисунок 2.24 – Продаж товару

Таким чином, вважаємо що ознайомились із виглядом вікна кожного з етапів.

3 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ТА ОСНОВИ ОХОРОНИ ПРАЦІ

3.1 Поняття та об’єкт аналізу технічної безпеки

Безпеку визначають як стан діяльності людини, за яким з визначе-ною ймовірністю виключено прояв небезпек або ж відсутня надзвичай-на небезпека. Безпека праці – це стан умов праці людини, за яких відсу-тня дія небезпечних і шкідливих факторів.

Об’єктом аналізу безпеки праці є виробнича система "людина – машина – навколишнє середовище" (ЛМС), в якій в єдиній комплекс, створений для виконання певних функцій, поєднані технічні об’єкти, люди і навколишнє середовище, які взаємодіють між собою.

Основними компонентами виробничої системи є людина, машина, навколишнє середовище, взаємодія між якими має ґрунтуватись на дотри-манні відповідних правил, нормативних документів і бути керованою.

Система ЛМС є багаторівневою за ієрархією управління. Ієрархія поділяє людей на особу, яка формує завдання, організовує й управляє вироб-ництвом, й особу, яка разом з технікою безпосередньо виконує це завдан-ня. Таким чином, людина системи ЛМС більш високого рівня розглядає людину і техніку системи ЛМС більш низького рівня як єдиний компонент – своєрідну людину-машину, призначену для здійснення замислу.

Крім рівнів і компонентів в системі ЛМС доцільно виділити окремі стадії її життєвого циклу:

1. Стадія проектування (визначення завдань, формування вимог, розрахунок параметрів).
2. Стадія реалізації (коли у процесі виробництва перша стадія реа-лізується на практиці).
3. Стадія експлуатації (коли система ЛМС здійснює покладені на неї робочі функції).

Вірогідність нещасного випадку зростає, як тільки людина попадає в поле дії небезпечного або шкідливого фактору. Це небезпечні зони, що характеризуються певним видом небезпеки, її інтенсивністю, часом і простором дії.

Таким чином, з точки зору аналізу й управління небезпеками необ-хідно розглядати та аналізувати структурні елементи системи ЛМС – рівні (вищий і нижчий), компоненти і стадії життєвого циклу.

Взаємодія компонентів, що входять до системи ЛМС, може бути штат-ною і нештатною. Нештатна взаємодія може виявлятися у вигляді надзви-чайної події – небажаних, незапланованих випадків, що порушують технологічний процес у відносно короткий відрізок часу. Відмова й інцидент, як пра-вило, передують надзвичайній події, але можуть мати і самостійне значення. До головних моментів аналізу небезпек належить пошук відповідей на такі питання:

1. Які об’єкти є небезпечними;
2. Яким надзвичайним подіям можна запобігти;
3. Які надзвичайні події неможливо усунути і як часто вони мати-муть місце;
4. Яку шкоду не усунуті надзвичайні події можуть спричинити людям, об’єктам, навколишньому середовищу.

Пошук причин надзвичайних подій призводить до аналізу системи управління небезпеками (СУН) на виробництві. Ці системи обов’язково включають такі компоненти, як наявність інформації, зворотних зв’язків та алгоритми функціонування.

Наявність зворотних зв’язків й інформаційної системи дозволяє про-водити збір даних щодо відхилень, відмов, проводити аналіз небезпек, по-рівнювати наслідки функціонування системи ЛМС з програмою управлін-ня небезпеками, приймати рішення. У виробничій системі ЛМС інформаційні функції виконують: рапорти інспекторів, акти розслідування нещасних випадків, аварій, протоколи атестації робочих місць тощо.

3.2 Розрахунок захисного заземлення

Відповідно до ГОСТ 12.1.030-81 захисне заземлення повинне забезпечити захист людей від поразки електричним струмом, при дотику до металевих неструмоведучих частин, які можуть виявитися під напругою. Заземленням називається навмисне з'єднання електроустановок із заземлюючим пристроєм, Заземлювачем називається провідник, що перебуває в зіткненні із землею або її еквівалентом. Заземлюючим провідником називається провідник, що з'єднує заземлені частини із заземлювачем. Сукупність з'єднуючих провідників і заземлювачів називається заземлюючим пристроєм. Для установок потужністю не більше 100 кВт опір заземлюючого пристрою не повинне перевищувати 10 Ом, для установок потужністю більше 100 кВт – 4 Ом.

Розрахунок штучного заземлювального пристрою при відсутності природних заземлювачів.

Вихідні дані:

Захищуваний об’єкт – комп’ютерна мережа.

Захищуваний об’єкт – стаціонарний.

Напруга мережі – 220 В.

Виконання мережі – з глухозаземленою нейтраллю.

Тип заземлювального пристрою – вертикальні труби.

Розміри вертикальних заземлювачів:

Довжина – 6 м.

Діаметр труби – 0,60 м.

Товщина стінки труби – 0,06 м.

Висота труби – 0,6 м.

Відношення відстані між трубами до їхньої довжини:

$$\frac{L_B}{l_B} = 1 \qquad (3.1)$$

(5.2)

Розмір горизонтального заземлювача (з'єднувальної стрічки): довжина $L_Г = L_{З.С.}$ – згідно з розрахунком, м; ширина горизонтальної з'єднувальної стрічки b_c=0,04 м.

Глибина закладання вертикальних заземлювачів $h_В$=0,6.

Розміщення заземлювачів попередньо приймають за чотирикутним контуром при числі стержнів від 4 до 100 та в один ряд при числі стержнів від 2 до 20.

Грунт – супісок; склад – однорідний; вологість – мала; агресивність – нормальна.

Кліматична зона – II.

Розрахунок:

1. Визначаємо характеристику навколишнього середовища в приміщенні організації: за пожежною небезпекою згідно з ПУЕ воно відноситься до класу П-II; за вибухонебезпекою згідно з ПУЕ – до класу В-I; за ступенем ураження електричним струмом – без підвищеної та особливої небезпеки.

2. Визначаємо $R_Д$ – допустиме (нормативне) значення опору розтікання струму в заземлювальному пристрої, $R_Д \leq 4$ *Ом*.

3. Обраховуємо $K_{С.В.}$ – приблизне значення питомого опору ґрунту, що рекомендується для розрахунку – $\rho_{ТАБЛ} = 300$ *Ом* $\cdot$ *м*.

4. Визначаємо $K_{С.В.}$ – коефіцієнт сезонності для вертикальних заземлювачів для денної кліматичної зони. За довідковою інформацією приймаємо $K_{С.В.}$=1,5.

5. Обраховуємо значення $K_{С.Г.}$ – коефіцієнт сезонності для горизонтального заземлювача згідно з кліматичною зоною. За довідковою інформацією приймаємо ; $K_{С.Г.}$=3,5.

6. Визначаємо $\rho_{РОЗР.В.}$ – розрахунковий питомий опір ґрунту для вертикальних заземлювачів:

$$\rho_{РОЗР.В.} = \rho_{ТАБЛ} \cdot K_{С.В.} = 300 \cdot 1{,}5 = 450 \text{ Ом} \cdot \text{м}. \qquad (3.2)$$

7. Розраховуємо $\rho_{РОЗР.Г.}$ – розрахунковий питомий опір грунту для горизонтальних заземлювачів:

$$\rho_{РОЗР.Г.} = \rho_{ТАБЛ} \cdot K_{С.Г.} = 300 \cdot 3{,}5 = 1050\, Ом \cdot м. \tag{3.3}$$

8. Обраховуємо t – відстань від поверхні землі до середини вертикального заземлювача:

$$t = h_B + \frac{l_B}{2} = 0{,}6 + \frac{6}{2} = 3{,}6\ м. \tag{3.4}$$

9. Визначаємо R_B – опір, Ом, розтікання струму в одному вертикальному заземлювачі:

$$R_B = \frac{\rho_{РОЗ.В}}{2\pi l_B} \cdot \left(ln\frac{2l_B}{d} + 0{,}5 \cdot ln\frac{4t + l_B}{4t - l_B} \right) =$$
$$= \frac{450}{2\pi \cdot 6} \cdot \left(ln\frac{2 \cdot 6}{0{,}60} + 0{,}5 \cdot ln\frac{4 \cdot 3{,}6 + 6}{4 \cdot 3{,}6 - 6} \right) = 41{,}05\ Ом \tag{3.5}$$

10. Визначаємо $n_{т.в.}$ – теоретична кількість вертикальних заземлювачів без врахування коефіцієнта використання $\eta_{в.в.}$, тобто $\eta_{в.в.}=1$:

$$n_{Т.В.} = \frac{R_B}{R_Д \cdot \eta_{В.В}} = \frac{41{,}05}{4 \cdot 1} = 10{,}26\ шт. \tag{3.6}$$

11. Визначаємо $\eta_{в.в}$ – коефіцієнт використання вертикальних заземлювачів при розташуванні їх згідно з вихідними даними або за чотирикутним

контуром при числі заземлення, $n_{т.в.}$=11 та при відношенні $\frac{L_B}{l_B}=1$. За довідником приймаємо $\eta_{в.в}$ =0,42.

12. Визначаємо $n_{н.в}$ – необхідна кількість штук, вертикально однакових заземлювачів з врахування коефіцієнта використання:

$$n_{H.B}=\frac{R_B}{R_Д\cdot\eta_{B.B}}=\frac{41{,}05}{4\cdot 0{,}42}=\frac{41{,}05}{1{,}68}=24{,}43\ шт. \qquad (3.7)$$

13. Визначаємо $R_{розр.в}$ – вертикальний опір, Ом, розтіканню струму у вертикальному заземленні при $n_{н.в}$ = 24,43 без врахування з'єднувальної стрічки:

$$R_{РОЗР.B}=\frac{R_B}{n_{H.B}\cdot\eta_{B.B}}=\frac{41{,}05}{10{,}26}=4\ Ом. \qquad (3.8)$$

14. Визначаємо $L_в$ – відстань між вертикальним заземлювачами за відношенням $\frac{L_B}{l_B}=1$, звідси

$$L_B=1\cdot l_B=1\cdot 6=6\ м. \qquad (3.9)$$

15. Визначаємо $L_{з.с}$ – довжину, м, з'єднання стрічки горизонтального заземлювача:

$$L_{З.С}=1{,}05\cdot L_B(n_{H.B}-1)=1{,}05\cdot 6(24{,}43-1)=147{,}60\ м. \qquad (3.10)$$

16. Визначаємо $R_{г.з.с}$ – опір, Ом, розтікання струму в горизонтальному заземлювачі (з'єднувальній стрічці):

$$R_{Г.З.С} = \frac{\rho_{РОЗ.Г}}{2 \cdot \pi \cdot L_{З.С}} \cdot ln \frac{2 \cdot L_{З.С}^{\ 2}}{2 \cdot b \cdot t} = \frac{1050}{2 \cdot 3{,}14 \cdot 147{,}61} \cdot ln \frac{2 \cdot (147{,}61)^2}{2 \cdot 0{,}04 \cdot 3{,}6} = 13{,}50\ Ом. \quad (3.11)$$

17. Визначаємо $\eta_{в.г}$ – коефіцієнт використання горизонтального заземлення при розташуванні вертикальних заземлювачів згідно з вихідними даними або за чотирикутним контуром при відношенні $\frac{L_В}{l_В} = 1$ та необхідній кількості вертикальних заземлювачів $n_{н.в} = 24{,}43$.

За довжину приймаю $\eta_{в.г} = 0{,}19$.

18. Визначаємо $R_{розр.г}$ – розрахунковий опір, Ом, розтікання струму в горизонтальному заземленні (з'єднувальній стрічці) при числі електродів $n_г = 1$:

$$R_{РОЗР.Г} = \frac{R_{Г.З.С}}{n_Г \cdot \eta_{В.Г}} = \frac{13{,}5}{1 \cdot 0{,}19} = 71{,}05\ Ом. \quad (3.12)$$

19. Визначаємо $R_{розр.в.г}$ – розрахунковий теоретичний опір, Ом, розтікання струму у вертикальному та горизонтальному заземленні:

$$R_{РОЗР.В.Г} = \frac{1}{\frac{1}{R_{РОЗ.В}} + \frac{1}{R_{РОЗ.Г}}} = \frac{1}{\frac{1}{4} + \frac{1}{71{,}05}} = 3{,}78 Ом. \quad (3.13)$$

20. Вибираємо матеріал та поперечний перетин з'єднувальних провідників. За довідковою інформацією вибираю голі мідні $S_м$ = 4 мм^2 провідники.

21. Вибираємо матеріал та поперечний перетин магістральної шини. За довідковою інформацією обираємо сталеву шину товщиною $\delta_с$ = 4мм і перетином не менше δ = 100 мм^2.

Схема з'єднання обладнання з магістральною шиною та з'єднання магістральної шини з заземлювальним пристроєм (рисунок 3.1).

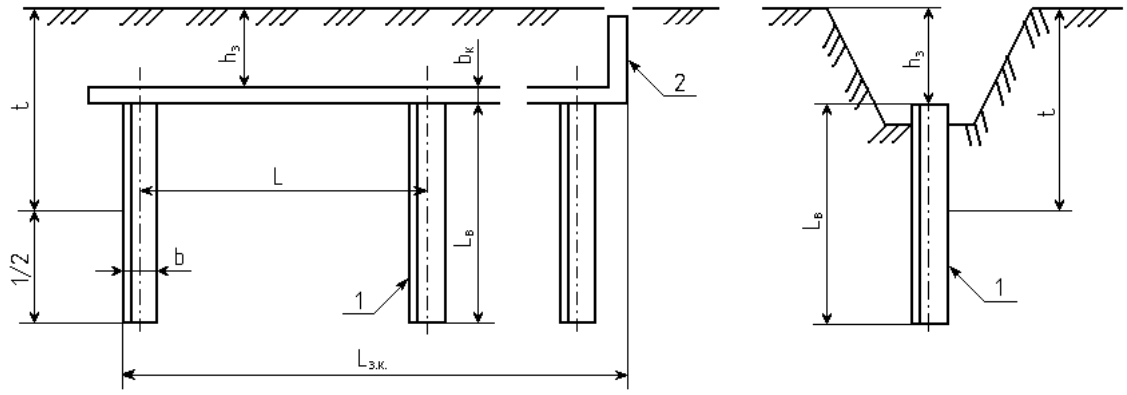

Рисунок 3.1 – Схема заземлювального контуру:

1 – вертикальний заземлювач; 2 – горизонтальний заземлювач;

$h_з$ – глибина закладання заземлювачів; L – відстань між заземлювачами;

$b_К$ – ширина квадрата; t – відстань від середини заземлювача до поверхні грунту;

$L_{З,К}$ довжина горизонтального заземлювача; d – ширина кутника;

$L_{Â}$ – довжина вертикального заземлювача.

ВИСНОВОК

В рамках кваліфікаційної роботи була розроблена автоматизована інформаційна система для контролю та обліку продажів комп'ютерної техніки та обладнання. В результаті виконаної розробки можна зробити наступні висновки:

При розробці системи був пройдений повний цикл проектування програми від постановки завдання замовником до здачі програмного забезпечення в експлуатацію.

Розроблена система дозволяє досягти наступних ефектів:

– зменшення часу, необхідного для обліку операцій, проведених підприємством;

– автоматизація контролю продажів товарів;

– можливість тривалого зберігання інформації про продажі на підприємство великого терміну давності, для можливості повнішого розрахунку ефективності діяльності підприємства;

– своєчасне отримання інформації про терміни оплати за здійснені постачання.

Так само при створенні автоматизованої інформаційної системи були досліджені умови праці інженера-програміста на його робочому місці на підприємстві (у лабораторії по розробці ПЗ) і зроблені конкретні пропозиції по їх поліпшенню.

На підставі вищесказаного можна зробити висновок про те, що розробка такого програмного забезпечення є доцільною і приноситиме реальну користь при використанні її на підприємстві.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Архангельский А.Я. Программирование в Delphi 7 - М.: ООО "Бином-Пресс", 2003, -1152с.

2. Архангельский А.Я. C++ Builder 6. Справочное пособие. Книга 1. Язык C++. - М.: Бином-Пресс, 2002 г. - 544 с.: ил.

3. Архангельский А.Я. C++ Builder 6. Справочное пособие. Книга 2. Классы и компоненты. - М.: Бином-Пресс, 2002 г. - 528 с.: ил.

4. Архангельський А.Я. . Программирование в C++ Builder 6. - М.: ЗАО "Издательство БИНОМ", 2003 г. - 1152 с.: ил.

5. Б.Страуструп. Язык программирования C++, 3-е изд./Пер. с англ.. - СПб.; М.: "Невский диалект" - "Издательство БИНОМ", 1999 г. - 991 с., ил.

6. Грофф Д. Р., Вайнберг Н. П. – SQL: Полн. руководство – пер. с англ., 2-е изд., перераб. и доп. – К.: BHV “Ирина”, 2001.

7. Дейт К. Дж. Введение в системы баз данных -К.; М.; СПб.: Изд. дом "Вильямс", 1999. -848 с.

8. Житецький В.Ц. Охорона праці користувачів комп’ютерів. Навчальний посібник. - Вид. 2-ге, доп. - Львів: Афіша, 2000. - 176с.

9. Карминский А.М., Нестеров П.В. Информатизация бизнеса. М.: Финансы и статистика, 1997.

10. Конноли Томас, Бегг Каротин, Страчан Анна. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 2-е изд.: Пер. с англ. : уч. пос. – М.: Издательский дом "Вильямс", 2000.– 1120 с.

11. Кренкс Д. – Теория и практика построения баз даных: Учебное пособие. пер. с англ. А. Вахитов – 8-е изд. – СПб. И др..: Питер, 2003.

12. Либерти Дж. Освой самостоятельно C++ за 21 день: 3-е изд/Пер. с англ.: Уч. пос. - М.: Издательский дом "Вильямс", 2001. -816 с.: ил.