Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ПОШУКУ І ПРИДБАННЯ АВІАКВИТКІВ
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-18-2

Рубашевська І.О.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник ст. викл. Путятіна О.Є.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2022 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Рубашевській Ірині Олегівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка інформаційної системи пошуку і придбання авіаквитків

затверджена наказом університету від 16 травня 2022 року № 541Ст

2. Термін подання студентом роботи до екзаменаційної комісії 28 травня 2022 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, специфікація застосунку, дані інтернет-мережі.

4. Перелік питань, що потрібно опрацювати в роботі

1. Аналіз принципів проєктування інформаційних систем.
2. Розроблення бази даних застосунку для пошуку і придбання авіаквитків.
3. Розроблення застосунку для пошуку і придбання авіаквитків.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Фізична модель даних, логічна модель даних, візуальне проєктування ERWIN, приклади екранних форм, тестові зображення.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Консультант з дотримання діючих стандартів та норм	Доцент Бєлова Н.В.		

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	18.04.2022	
2	Аналіз завдання, підбір літератури	18.04.22-21.04.22	
3	Аналіз літератури з досліджуваної проблеми	22.04.22-25.04.22	
4	Аналіз принципів проєктування інформаційних систем	26.04.22-30.04.22	
5	Розробка бази даних	01.05.22-12.05.22	
6	Програмна реалізація	13.05.22-21.05.22	
7	Оформлення пояснювальної записки	22.05.22-24.05.22	
8	Перевірка на плагіат	25.05.22	
9	Рецензування	26.05.22	
10	Підготовка презентації та доповіді	27.05.22-28.05.22	
11	Занесення роботи в електронний архів	03.06.22	
12	Попередній захист кваліфікаційної роботи	06.06.22	

Дата видачі завдання 18 квітня 2022 р.

Студент _____

(підпис)

Керівник роботи _____ ст. викл. Путятіна О.Є.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 58 с., 2 табл., 16 рис., 1 дод., 30 джерел.

ІНФОРМАЦІЙНА СИСТЕМА, СУБД, РЕЛЯЦІЙНА БАЗА ДАНИХ, СУТНІСТЬ, НОРМАЛІЗАЦІЯ ДАНИХ, ER-МОДЕЛЬ, МОДЕЛЬ «СУТНІСТЬ – ЗВ'ЯЗОК», МОВА ЗАПИТІВ SQL, CASE-ЗАСІБ ERWIN, MS SQL SERVER, MS SQL SERVER MANAGEMENT STUDIO.

Об'єктом роботи є розробка інформаційної системи пошуку і придбання авіаквитків.

Метою роботи є створення інформаційної системи для вибору клієнтом рейсу та видачі інформації про рейси за запитом клієнтів, яка має 2 рівня доступу: адміністратор та клієнт.

Під час розробки інформаційної системи були використані: теорія проєктування реляційних баз даних на базі будування ER-моделі, основні принципи нормалізації реляційних баз даних, мова запитів SQL, CASE-засіб візуального проєктування даних ERwin, середовище розробки програмних систем MS Visual Studio, мова програмування C#.

INFORMATION SYSTEM, DBMS, RELATIONAL DATABASE, ENTITY, DATA NORMALIZATION, ER-MODEL, ENTITY – RELATIONSHIP MODEL, STRUCTURED QUERY LANGUAGE, CASE-TOOL ERWIN, MS SQL SERVER, MS SQL SERVER MANAGEMENT STUDIO.

The object of the research is to develop an information system for finding and purchasing tickets.

The aim of the research is to create an information system for customer selection of the flight and the issuance of information about flights at the request of customers, which has 2 levels of access: administrator and customer.

During the development of the information system were used: the theory of relational database design based on the construction of the ER-model, basic principles of normalization of relational databases, SQL query language, ERWIN CASE visual data design tool, MS Visual Studio software development environment and C #programming language.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ІС – інформаційна система

Oracle DWM FT – Datawarehouse Method Fast Track (метод створення сховищ даних «високошвидкісна траса»)

DSDM – Dynamic System Development Method (метод розробки динамічних систем)

RAD – Rapid Application Development (швидка розробка аплікацій)

ІПС – інформаційно-пошукова система

ІЗ – інформаційне забезпечення

ПЗ – програмне забезпечення

БД – база даних

СУБД – система управління базами даних

МОД – мова опису даних

SQL – Structured Query Language (структурована мова запитів)

ВСТУП

Використання баз даних є однією з характеристик більшості сучасних інформаційних систем. По суті, бази даних є тим, навколо чого будується інформаційна система будь-якого підприємства. Тому теорії створення та практиці використання баз даних приділяється достатньо уваги в період функціонування інформаційних систем.

Довгий час основним типом були реляційні бази даних, які сьогодні вважаються класичними. Проте розвиток інформаційних систем поставив перед сучасними базами даних проблеми, вирішення яких неможливе за допомогою лише реляційних баз даних. Крім класичних завдань, сучасні бази даних повинні забезпечувати багатомашинну обробку та зберігання великих обсягів інформації, оперативний аналіз даних, інтеграцію із мережею Інтернет, розмежування доступу користувачів, захист інформації під час її передачі по мережі. Хоча на практиці використовується багато різних баз даних, у більшості з них є багато спільного, як щодо розробки, так і використання.

Актуальність роботи полягає у тому, що кожна авіакомпанія будь-якої країни прагне прискорити процеси своєї діяльності і підвищити прибутковість, які безпосередньо залежать від її швидкості та якості обслуговування. Виходячи з цього, існує потреба в дотриманні цієї вимоги, але це призводить до необхідності контролю здійснення рейсів і продажу квитків, а тому найкращим виходом з цієї ситуації є використання баз даних.

1 ПРИНЦИПИ ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЇ ІНФОРМАЦІЙНИХ СИСТЕМ

1.1 Загальні відомості про інформаційні системи

1.1.1 Тенденції розвитку інформаційних технологій

Тенденції розвитку сучасних інформаційних технологій зумовлюють постійне зростання складності інформаційних систем, які створюються в різних сферах людської діяльності. Сучасні великі проєкти ІС характеризуються, як правило, такими особливостями:

– складність опису (велика кількість функцій, процесів, елементів даних і складних зв’язків між ними), що вимагає ретельного моделювання та аналізу даних і процесів;

– наявність сукупності компонентів (підсистем), які перебувають у тісній взаємодії, виконують певні локальні задачі та цілі функціонування;

– відсутність прямих аналогів, що обмежує можливість використання типових проєктних рішень і прикладних систем;

– необхідність узгодження існуючих застосунків з новими розробками;

– робота в неоднорідному середовищі на декількох апаратних платформах;

– неоднорідність рівня кваліфікації та сформованих традицій використання окремих наборів інструментальних засобів у групах розробників;

– значна тривалість проєкту обумовлена, з одного боку, обмеженими можливостями команди розробників; з іншого боку, масштабами організації клієнта та різним ступенем готовності окремих її підрозділів до впровадження ІС.

Для успішної реалізації проєкту об’єкт проєктування спочатку має бути адекватно описаний, побудовані повні і несуперечливі функціональні та інформаційні моделі ІС. Накопичений на даний момент досвід проєктування ІС показує, що це складна, трудомістка і тривала за часом робота, яка вимагає залучення висококваліфікованих спеціалістів [1]. Проте донедавна проєктування ІС здійснювалося переважно на інтуїтивному рівні з використанням неформальних методів, заснованих на мистецтві, практичному досвіді, експертних оцінках та дорогих експериментальних тестах якості роботи ІС. Крім того, в процесі створення та експлуатації ІС інформаційні потреби користувачів можуть бути змінені або уточнені, що ще більше ускладнює розробку та обслуговування таких систем.

Традиційні підходи до побудови інформаційних систем базуються на ідеї, що на початку проєкту важко визначити загальний обсяг даних і які аналітичні завдання будуть вирішувати кінцеві користувачі. Наприклад, методологія Oracle DWM FT (Datawarehouse Method Fast Track – метод створення сховищ даних «високошвидкісна траса») передбачає, що розробники будуть визначати та аналізувати вимоги до зберігання даних протягом усього життєвого циклу інформаційної системи.

Заснована на DSDM (Dynamic System Development Method – метод розробки динамічних систем), ця методологія реалізує підхід RAD (Rapid Application Development – швидка розробка аплікацій).

Відповідно до DSDM і Oracle DWM FT, цикл проєктування проходить через створення ряду прототипів, поки не будуть задоволені вимоги кінцевих користувачів. Щоб цей цикл не став нескінченним, розробка ділиться на 120-денні часові відрізки (timebox), за які можна виконати чітко визначений фіксований набір вимог. Розробники стверджують, що завдяки гнучкості та простоті використання інструментів Business Intelligence, створення прототипів не представляє труднощів. Однак застосування такого підходу виправдано ІТ-фахівцями, які не прагнуть розширити свої знання в цій сфері.

1.1.2 Класифікація інформаційних систем

Інформаційні системи можна класифікувати по цілому ряду різних ознак. У основу даної класифікації покладені найбільш суттєві ознаки, що визначають функціональні можливості і особливості побудови сучасних систем. Залежно від об’єму вирішуваних завдань, використовуваних технічних засобів, організації функціонування, ІС діляться на ряд груп (класів) [2]. Розглянемо найбільш важливі з них.

За типом даних, що зберігаються, ІС діляться на фактографічні і документальні. Фактографічні системи призначені для зберігання і обробки структурованих даних у вигляді чисел і текстів. У документальних системах інформація представлена у вигляді документів, що складаються з найменувань, описів, рефератів і текстів. Пошук серед неструктурованих даних здійснюють із використанням семантичних ознак. Відібрані документи надаються користувачеві, а обробка даних в таких системах практично не проводиться.

За ступенем автоматизації інформаційних процесів в системі управління фірмою, ІС діляться на ручні, автоматичні та автоматизовані.

Ручні ІС характеризуються відсутністю сучасних технічних засобів переробки інформації та виконанням всіх операцій людиною.

В автоматичних ІС всі операції з переробки інформації виконуються без участі людини.

Автоматизовані ІС припускають участь в процесі обробки інформації людини і технічних засобів, при цьому головна роль у виконанні рутинних операцій обробки даних відводиться комп’ютеру [3].

Залежно від характеру обробки даних ІС діляться на інформаційно-пошукові (довідкові) і інформаційно-вирішальні.

Нині створено і успішно функціонує велике число інформаційно-довідкових систем різного призначення, які призначені для задоволення інформаційних запитів користувачів. Характерна особливість

таких систем – інформація, знайдена відповідно до запиту, не використовується безпосередньо у рамках цієї ж системи, а видається користувачеві, який використовує отриману інформацію для будь-яких необхідних йому цілей. Пошук – одна з основних операцій в таких системах, тому вони є також інформаційно-пошуковими системами (ІПС) [4].

Інформаційно-пошукові системи роблять введення, систематизацію, зберігання, видачу інформації за запитом користувача без складних перетворень даних.

Інформаційно-вирішальні системи здійснюють, крім того, операції переробки інформації по певному алгоритму. За характером використання вихідної інформації такі системи прийнято ділити на ті, що керують і радять.

Результуюча інформація керуючих ІС безпосередньо трансформується в прийняті людиною рішення. Для цих систем характерні завдання розрахункового характеру та обробка великих обсягів даних (наприклад, ІС планування виробництва або замовлень, бухгалтерського обліку).

Інформаційні системи, які надають поради, виробляють інформацію, яка приймається людиною до відома і враховується при формуванні управлінських рішень, вона не ініціює конкретні дії. Ці системи імітують інтелектуальні процеси обробки знань, а не даних (наприклад, експертні системи).

Залежно від сфери застосування розрізняють наступні класи ІС.

Інформаційні системи організаційного управління призначені для автоматизації функцій управлінського персоналу як промислових підприємств, так і непромислових об’єктів (готелів, банків, магазинів). Основними функціями подібних систем є: оперативний контроль і регулювання, оперативний облік і аналіз, перспективне і оперативне планування, бухгалтерський облік [5].

ІС управління технологічними процесами (ТПр) служать для автоматизації функцій виробничого персоналу по контролю і управлінню виробничими операціями. У таких системах зазвичай передбачається

наявність розвинених засобів виміру параметрів технологічних процесів (температури, тиску, хімічного складу і тому подібне).

ІС автоматизованого проєктування (САПР) призначені для автоматизації функцій інженерів-проєктувальників, конструкторів, архітекторів, дизайнерів при створенні нової техніки або технології. Основними функціями подібних систем є: інженерні розрахунки, створення графічної документації (креслень, схем, планів), створення проєктної документації, моделювання проєктованих об’єктів.

За масштабом ІС поділяються на однокористувацькі, групові та корпоративні. Інтегровані (корпоративні) ІС використовуються для автоматизації усіх функцій фірми і охоплюють увесь цикл робіт від планування діяльності до збуту продукції. Вони включають ряд модулів (підсистем), працюючих в єдиному інформаційному просторі і виконуючих функції підтримки відповідних напрямів діяльності.

Існує класифікація ІС залежно від рівня управління, на якому система використовується.

Інформаційна система оперативного рівня підтримує виконавців, обробляючи дані про угоди і події (рахунки, накладні, зарплата, кредити, потік сировини і матеріалів). Інформаційна система оперативного рівня є сполучною ланкою між фірмою і зовнішнім середовищем.

Інформаційні системи фахівців підтримують роботу з даними і знаннями, підвищують продуктивність роботи інженерів і проєктувальників. Завдання подібних систем – інтеграція нових відомостей в організацію і допомогу в обробці паперових документів.

Інформаційні системи рівня менеджменту використовуються працівниками середньої управлінської ланки для моніторингу, контролю, ухвалення рішень і адміністрування [6].

Основні функції цих інформаційних систем:

– порівняння поточних показників з минулими;

– складання періодичних звітів за певний час, а не видача звітів по поточних подіях, як на оперативному рівні;

– забезпечення доступу до архівної інформації.

Стратегічна інформаційна система – комп'ютерна інформаційна система, що забезпечує підтримку ухвалення рішень по реалізації стратегічних перспективних цілей розвитку організації.

Інформаційні системи стратегічного рівня допомагають вищій ланці управлінців вирішувати неструктуровані завдання, здійснювати довгострокове планування.

Класифікація по архітектурі.

Архітектура «Файл-сервер» – історично перша архітектура інформаційних систем. Як виконавчі модулі, так і дані розміщуються в окремих файлах операційної системи. Доступ до даних здійснюється через шлях (path) і використання файлових операцій (відкрити, зчитати, записати). Для зберігання даних використовується виділений сервер (окремий комп'ютер), який і є файловим сервером. Виконувані модулі зберігаються або на робочих станціях, або на файловому сервері. В останньому випадку спрощується процедура їх адміністрування, але при цьому зростають вимоги до надійності мережі.

Архітектура «Клієнт-сервер» – це не тільки архітектура, це – нова парадигма, що прийшла на зміну застарілим концепціям. Суть її полягає в тому, що клієнт (виконуваний модуль) запитує ті чи інші сервіси відповідно до визначеного протоколом обміну даними. При цьому, на відміну від ситуації з файловим сервером, немає необхідності у використанні прямих шляхів операційної системи: клієнт їх «не знає», йому «відомі» лише ім'я джерела даних та інші спеціальні відомості, що використовуються для авторизації клієнта на сервері. Сервер, який фізично може перебувати на тому ж комп'ютері, а може – на іншому кінці земної кулі, обробляє запит клієнта і, зробивши відповідні маніпуляції з даними, передає клієнту запитувану порцію даних [7].

В рамках напряму «клієнт-сервер» існують два основних «діалекти»: «тонкий» і «товстий» клієнт.

У системах на основі тонкого клієнта використовується потужний сервер баз даних, це – високопродуктивний комп'ютер і бібліотека так званих процедур, які дозволяють робити обчислення, що реалізують основну логіку обробки даних, безпосередньо на сервері. Клієнтська програма, відповідно, пред'являє невисокі вимоги до апаратного забезпечення робочої станції. Основна перевага таких систем – відносна дешевизна клієнтських станцій [8].

Системи з товстим клієнтом, навпаки, реалізують основну логіку обробки на клієнті, а сервер являє собою в чистому вигляді сервер баз даних, що забезпечує виконання тільки стандартизованих запитів на маніпуляцію з даними (як правило, читання, запис, модифікацію даних в таблицях реляційної бази даних). У системах такого класу вимоги до робочої станції вище, а до сервера – нижче. Всі промислові сервери баз даних реляційного типу підтримують роботу зі стандартизованою мовою маніпулювання даними SQL, але внутрішня вбудована мова обробки даних, необхідна для реалізації логіки обробки у кожного з серверів своя.

1.1.3 Сфери застосування інформаційних систем

В час бурхливого розвитку мереж, інформаційні системи зайняли одне з найголовніших місць у процесах роботи організацій. Інформаційні системи призначені для своєчасного забезпечення та задоволення потреб користувачів в інформації. Наприклад, у діяльності підприємств існує практика створення та функціонування єдиної корпоративної інформаційної системи, що задовольняє інформаційні потреби усіх співробітників, служб та підрозділів організації.

Якщо раніше мало не єдиною областю, в якій застосовувалися інформаційні системи, була автоматизація бухгалтерського обліку, то зараз спостерігається впровадження інформаційних технологій у безліч інших областей. Розглянемо найбільш важливі завдання, що вирішуються за допомогою спеціальних програмних засобів.

Бухгалтерський облік. Це класична сфера застосування інформаційних технологій і завдання, що найчастіше реалізовується на сьогодні. Таке положення цілком з'ясовне. По-перше, помилка бухгалтера може коштувати дуже дорого, тому очевидна вигода використання можливостей автоматизації бухгалтерії. По-друге, завдання бухгалтерського обліку досить легко формалізується, так що розробка систем автоматизації бухгалтерського обліку не представляє технічно складної проблеми.

Управління фінансовими потоками. Впровадження інформаційних технологій в управління фінансовими потоками також обумовлено критичністю цієї області управління підприємства до помилок [9]. Неправильно побудувавши систему розрахунків з постачальниками і споживачами, можна спровокувати кризу готівки навіть при налагодженій мережі закупівлі, збуту і хорошому маркетингу.

Управління складом, асортиментом, закупівлями. Далі, можна автоматизувати процес аналізу руху товару, тим самим відстеживши і зафіксувавши ті двадцять відсотків асортименту, які приносять вісімдесят відсотків прибутку.

Управління виробничим процесом. Основними механізмами тут є планування і оптимальне управління виробничим процесом. Автоматизоване рішення подібної задачі дає можливість грамотно планувати, враховувати витрати, проводити технічну підготовку виробництва, оперативно управляти процесом випуску продукції.

Управління маркетингом. Управління маркетингом має на увазі збір і аналіз даних про фірми конкурентів, їх продукцію і цінову політику, а також моделювання параметрів зовнішнього оточення для визначення

оптимального рівня цін, прогнозування прибутку і планування рекламних кампаній.

Документообіг. Добре налагоджена система облікового документообігу відбиває поточну виробничу діяльність, що реально відбувається на підприємстві, і дає управлінцям можливість впливати на неї [10].

Оперативне управління підприємством. Інформаційна система, вирішальна завдання оперативного управління підприємством, будується на основі бази даних, в якій фіксується уся можлива інформація про підприємство. Така інформаційна система є інструментом для управління бізнесом і зазвичай називається корпоративною ІС.

Надання інформації про фірму. Практично кожне підприємство, що поважає себе, зараз має свій web-сервер підприємства для вирішення завдань, з яких можна виділити дві основні:

– створення іміджу підприємства;

– розвантаження довідкової служби компанії для отримання необхідної інформації про фірму, пропоновані товари, послуги і ціни [11].

1.2 Методології і технології розробки інформаційних систем

1.2.1 Основні процеси життєвого циклу інформаційних систем

Основні процеси життєвого циклу ІС складаються з п’яти процесів, які реалізуються під управлінням основних сторін, залучених в життєвий цикл ПЗ. Основними сторонами є замовник, постачальник, розробник. Основними процесами вважають:

- процес замовлення. Визначає роботи замовника;
- процес постачання. Визначає роботи постачальника;
- процес розробки. Визначає роботи розробника;
- процес експлуатації. Визначає роботи оператора;

– процес супроводу. Визначає роботи супроводжуючої організації, яка надає послуги з супроводу програмного продукту.

Процес створення ІС включає в себе кілька етапів, обмежених певним проміжком часу, які закінчуються випуском конкретного продукт (моделі, програмного продукту, документації тощо) [12]. Зазвичай виділяють наступні етапи створення ІС: формування вимог до системи, проєктування, реалізація, тестування, введення в дію, експлуатація та супровід.

Початковим етапом процесу створення ІС є моделювання бізнес-процесів, що відбуваються на підприємстві, досягають його цілей і завдань. Модель організації описана з точки зору бізнес-процесів і бізнес-функцій, дозволяє сформулювати основні вимоги до ІС. Множина моделей опису вимог до ІС потім перетворюється в систему моделей, які описують концептуальний проєкт ІС. Формуються моделі архітектури ІС, вимог до ПЗ та інформаційного забезпечення. Потім формується архітектура ПЗ та ІЗ, виділяються корпоративні БД та окремі застосунки, формуються моделі вимог до застосунків і проводиться їх розробка, тестування та інтеграція [13].

Метою початкових етапів створення ІС, виконуваних на стадії аналізу діяльності організації, є формування вимог до ІС, які відображають цілі та завдання організації-замовника. Щоб специфікувати процес створення ІС, яка відповідає потребам організації, потрібно з’ясувати і чітко сформулювати сутність цих потреб. Для цього необхідно визначити вимоги замовників до ІС і відобразити їх на мові моделей вимог до розробки проєкту ІС так, щоб забезпечити відповідність цілям і задачам організації. Завдання формування вимог до ІС є одним із найвідповідальніших, важко формалізованих, найдорожчих і важких для виправлення в разі помилки.

Сучасні інструментальні засоби і програмні продукти дозволяють швидко створювати ІС відповідно готовим вимогам. Але найчастіше ці системи не задовольняють замовників, вимагають численних доробок, що призводить до різкого подорожчання вартості ІС. Основною причиною

такого становища є неправильне, неточне або неповне визначення вимог до ІС на етапі аналізу [14].

На етапі проєктування перш за все формуються моделі даних. Проєктувальники в якості вихідної інформації отримують результати аналізу. Побудова логічної і фізичної моделей даних є основною частиною проєктування бази даних. Отримана в процесі аналізу інформаційна модель спочатку перетвориться в логічну, а потім у фізичну модель даних.

Паралельно із проєктуванням схеми бази даних виконується проєктування процесів, щоб отримати специфікації всіх модулів ІС. Головна мета проєктування процесів полягає у відображенні функцій, отриманих на етапі аналізу, в модулі ІС. При проєктуванні модулів визначають інтерфейси програм: розмічають меню, вигляд вікон, гарячі клавіші і пов’язані з ними виклики. Кінцевими продуктами етапу проєктування є: схема бази даних (на підставі моделі, розробленої на етапі аналізу) та набір специфікацій модулів системи (вони будуються на базі моделей функцій).

На етапі проєктування здійснюють також розробку архітектури ІС, яка включає в себе вибір платформи і операційної системи. У неоднорідній ІС можуть працювати кілька комп’ютерів на різних апаратних платформах і під управлінням різних операційних систем. Етап проєктування завершується розробкою технічного проєкту ІС [15].

На етапі реалізації здійснюється створення ПЗ системи, встановлення технічних засобів, розробка експлуатаційної документації.

Етап тестування зазвичай виявляється розподіленим в часі. Після завершення розробки окремого модуля системи виконують автономний тест, який переслідує мету виявити відмови модуля та перевірити відповідність модуля специфікації.

Після того як автономний тест успішно пройдено, модуль включають до складу розробленої частини системи і група згенерованих модулів проходить тестування зв’язків, які повинні відстежити їх взаємний вплив.

Далі група модулів тестується на надійність роботи, тобто проходять тести імітації відмов системи і напрацювання на відмову. Перша група тестів показує, наскільки добре система відновлюється після збоїв ПЗ, відмов апаратного забезпечення. Друга група тестів визначає ступінь стійкості системи при штатній роботі і дозволяє оцінити час безвідмовної роботи системи. У комплект тестів стійкості повинні входити тести, які імітують пікове навантаження на систему.

Потім весь комплект модулів проходить системний тест – тест внутрішнього приймання товару, який показує рівень його якості. Сюди входять тести функціональності і надійності системи.

Останній тест ІС – приймально-здавальні випробування, який передбачає показ ІС замовникові, повинен містити групу тестів, що моделюють реальні бізнес-процеси, щоб показати відповідність реалізації вимогам замовника.

1.2.2 Життєвий цикл програмного забезпечення ІС

Відповідно до базових міжнародних стандартів ISO/IEC 12207 усі процеси життєвого циклу ПЗ розділяють на такі три групи: основні, допоміжні, організаційні.

Основні процеси охоплюють процеси придбання, постачання, розроблення, експлуатації, супроводження ПЗ. Допоміжні процеси призначені для підтримки виконання основних процесів, забезпечення якості проєкту, організації верифікації, перевірки та тестування ПЗ. Організаційні процеси охоплюють процеси створення інфраструктури, управління, навчання, удосконалення. До них примикає процес адаптації, який визначає основні дії, необхідні для адаптації стандарту до умов конкретного проєкту. Організаційні процеси визначають дії і завдання, що виконуються замовником та розробником проєкту для управління їхніми процесами.

Організаційні процеси виконуються на корпоративному рівні або на рівні всієї організації в цілому, створюючи базу для реалізації та постійного вдосконалення інших процесів життєвого циклу ПЗ [16].

Розробка включає, як правило, такі етапи.

Планування і оцінка проєкту. Під час цієї фази виявляються властивості, які повинна мати готова система, тобто, створюється задум системи в остаточному варіанті. Під час цієї фази приймаються рішення відносно розмірів, часу відгуку і інших параметрів системи. Визначення здійснюється за допомогою двох основних категорій – вимог і специфікацій.

Аналіз системних і програмних вимог. Системний аналіз задає роль кожного елементу в комп'ютерній системі, взаємодію елементів один з одним. Аналіз вимог відноситься до програмного елементу – програмного забезпечення. Уточнюються і деталізуються його функції, характеристики і інтерфейс [17].

Проєктування алгоритмів, структур цих і програмних структур. Проєктування полягає в створенні представлень: архітектури ПЗ і модульної структури ПЗ, алгоритмічної структури ПЗ і структури даних, вхідного і вихідного інтерфейсу.

Реалізація (кодування) полягає в перекладі результатів проєктування в текст на мові програмування.

Тестування відповідає за виконання програми для виявлення дефектів у функціях, логіці і формі реалізації програмного продукту.

Введення в дію (супровід) – це внесення змін до експлуатованої ІС. Цілі змін: виправлення помилок, адаптація до змін зовнішньої для ІС середовища, удосконалення ІС за вимогами замовника.

Експлуатація включає роботи по впровадженню компонентів ІС в експлуатацію.

1.3 Архітектура інформаційної системи

Рівень розвитку сучасних технологій настільки високий, що дозволяє побудувати ІС з будь-якою складністю та функціональністю. Проте, враховуючи вимоги бізнесу, ґрунтовані на показниках різних бізнес-оцінок, виникають задачі, розв’язання яких зводиться до забезпечення раціонального підходу до процесу проєктування, реалізації та подальшої експлуатації ІС. Вибрану архітектуру можна розглядати як один із основних показників ефективності створеної ІС.

Відповідно до стандарту ANSI/IEEE 1471-2000, загальноприйнятим є таке визначення архітектури системи – це опис організації системи в термінах компонентів, їх взаємозв’язків між собою і з довкіллям, принципи управління їх розробкою і розвитком. Архітектуру ІС можна описати як концепцію, яка визначає модель, структуру, виконувані функції та взаємозв’язок компонентів. Архітектура є багатовимірною, оскільки різні фахівці працюють з різними її аспектами [18].

Архітектура ПЗ також передбачає різні подання, які служать різним цілям: зображенню функціональних можливостей системи; відображенню логічної організації системи; опису фізичної структури програмних компонентів в середовищі реалізації; відображенню структури потоків управління та аспектів паралельної роботи; опису фізичного розміщення програмних компонентів на базовій платформі [19].

З точки зору програмно-апаратної реалізації можна виділити такі типові архітектури ІС:

– традиційні архітектурні рішення, засновані на використанні виділених файлів-серверів або серверів баз даних;

– архітектури корпоративних ІС, які базуються на технології Internet (Internet-застосунки);

– архітектури ІС, які ґрунтуються на концепції «сховища даних» – інтегрованого інформаційного середовища, що складається із різнорідних інформаційних ресурсів;

– для побудови глобальних розподілених інформаційних застосунків використовується архітектура інтеграції інформаційно-обчислювальних компонентів на основі об’єктно-орієнтованого підходу.

Процес проєктування ІС тісно пов’язаний із її архітектурним описом. Можна виділити п’ять різних підходів до проєктування: календарний підхід; підхід, за основу якого узято процес управління вимогами; підхід, ґрунтований на процесі розробки документації; підхід, в основі якого лежить система управління якістю; архітектурний підхід [20].

Розглянемо їх детально:

– календарний підхід передбачає складання графіку майбутніх робіт з їх поетапним виконанням (при цьому ключові рішення приймаються на підставі локальних завдань і цілей кожного конкретного етапу розробки). Цей підхід не приділяє часу розробці документації, формуванню архітектури і процесам внесення змін (у перспективі через це зростає вартість володіння розробленою на основі цього підходу системою). Він вважається застарілим, проте в деяких компаніях досі застосовується;

– підхід, за основу якого взято процес управління вимогами. Велику частину часу процесу розробки виділяють на функціональні характеристики системи, а нефункціональні (наприклад, масштабованість) практично не розглядають. В ході проєкту усі рішення формуються виходячи з локальних цілей по реалізації конкретного функціонала. Такий підхід може бути ефективний, якщо вимоги до системи, що розробляється, визначені заздалегідь і не змінюються в процесі проєктування. До недоліків можна віднести невідповідність стандарту якості ISO 9126 і нестабільність архітектури системи, що розробляється, оскільки кожна функція, яку реалізують, зв’язується з одним або декількома компонентами [21]. У зв’язку із цим при додаванні до подібних систем додаткових функцій трудомісткість

зростає. Застосування цього підходу в перспективі є нераціональним, проте дозволить успішно управляти вимогами в рамках необхідної функціональності;

– підхід, ґрунтований на процесі розробки документації. Велика кількість часу витрачається на формування пакету документів, який зазвичай не використовує ані замовник, ані користувач. Крім того, із-за нестачі часу страждає якість системи, яку розробляють. Цей підхід використовують в урядових організаціях і великих компаніях;

– підхід, за основу якого взято систему управління якістю, включає велику кількість різнопланових заходів для відстежування параметрів, найбільш значимих для функціонування системи. Вибрані параметри спостерігаються на всіх стадіях розробки системи (в деяких випадках на шкоду іншим). Іноді набір таких параметрів може бути складений неправильно, і оптимізація системи буде проведена помилково. У цьому полягає основний недолік підходу. Окрім цього, при появі нових вимог дуже важко змінювати функціональність, і подальший розвиток системи може бути неможливий у тому числі з причин архітектурних прорахунків [22]. Такий підхід вважається консервативним, а його застосування доцільно при необхідності створити систему з екстремальними характеристиками;

– архітектурний підхід до проєктування ІС можна вважати найбільш зрілим. Його ключовим аспектом є створення фреймворка, тобто каркаса, адаптацію якого під потреби конкретної системи легко здійснити. Відповідно до цього, завдання проєктування розбивається на дві підзадачі: розробка багаторазово використовуваного каркаса та створення системи на його основі. Ці підзадачі можуть вирішуватися різними групами фахівців. При використанні каркасів з’являється можливість швидко змінювати функціональність системи за рахунок ітераційного характеру процесу проєктування.

1.4 Бази даних

1.4.1 Поняття бази даних

Відомі два підходи до організації інформаційних масивів: файлова організація та організація у вигляді бази даних. Файлова організація передбачає спеціалізацію та збереження інформації, орієнтованої, як правило, на одну прикладну задачу, та забезпечується прикладним програмістом. Така організація дозволяє досягнути високої швидкості обробки інформації, але характеризується рядом недоліків.

Характерна риса файлового підходу – вузька спеціалізація як обробних програм, так і файлів даних, що служить причиною великої надлишковості, тому що ті самі елементи даних зберігаються в різних системах. Оскільки керування здійснюється різними особами (групами осіб), відсутня можливість виявити порушення суперечливості збереженої інформації. Розроблені файли для спеціалізованих прикладних програм не можна використовувати для задоволення запитів користувачів, які перекривають дві і більше області. Крім того, файлова організація даних внаслідок відмінностей структури записів і форматів передання даних не забезпечує виконання багатьох інформаційних запитів навіть у тих випадках, коли всі необхідні елементи даних містяться в наявних файлах. Тому виникає необхідність відокремити дані від їхнього опису, визначити таку організацію збереження даних з обліком існуючих зв’язків між ними, яка б дозволила використовувати ці дані одночасно для багатьох застосувань [23]. Вказані причини обумовили появу баз даних.

База даних може бути визначена як структурна сукупність даних, що підтримуються в активному стані та відображає властивості об’єктів зовнішнього світу. В базі даних містяться не тільки дані, але й описи даних, і тому інформація про форму зберігання вже не схована в сполученні «файл-програма», вона явним чином декларується в базі. База даних орієнтована на інтегровані запити, а не на одну програму, як у випадку

файлового підходу, і використовується для інформаційних потреб багатьох користувачів. В зв’язку з цим бази даних дозволяють в значній мірі скоротити надлишковість інформації. Перехід від структури БД до потрібної структури в програмі користувача відбувається автоматично за допомогою систем управління базами даних (СУБД) [24].

1.4.2 Призначення та класифікація СУБД

СУБД – це складна програмна система для накопичення та подальшого маніпулювання даними, що представляють інтерес для користувача. Кожній прикладній програмі СУБД надає інтерфейс з базою даних та має засоби безпосереднього доступу до неї. Таким чином, СУБД відіграє центральну роль в функціонуванні автоматизованого банку даних.

Архітектурно СУБД складається з двох великих компонент. За допомогою мови опису даних (МОД) створюються описи елементів, груп та записів даних, а також взаємозв’язки між ними, які, як правило, задаються у вигляді таблиць. В залежності від конкретної реалізації СУБД мову опису даних підрозділяють на мову опису схеми бази даних (МОС) та мову опису підсхем бази даних (МОП). Слід особливо зазначити, що МОД дозволяє створити не саму базу даних, а лише її опис. Для виконання операцій з базою даних в прикладних програмах використовується мова маніпулювання даними (ММД). Фактична структура фізичного зберігання даних відома тільки СУБД. З метою забезпечення зв’язків між програмами користувачів і СУБД, що особливо важливо при мультипрограмному режимі роботи операційної системи, в СУБД виділяють особливу складову – резидентний модуль системи керування базами даних. Цей модуль значно менший від всієї СУБД, тому на час функціонування автоматизованого банку інформації він може постійно знаходитись в основній пам’яті ЕОМ та забезпечувати взаємодію всіх складових СУБД і програм, які до неї звертаються [25].

Приведена структура притаманна усім СУБД, котрі розрізняються обмеженнями та можливостями по виконанню відповідних функцій. Отже, процес порівняння і оцінки таких систем для одного конкретного застосування зводиться до співставлення можливостей наявних СУБД з вимогами користувачів.

До недавнього часу при організації обробки інформації на ЕОМ застосовувався підхід, при якому на основі інформації одного і того ж об’єкту управління (наприклад, матеріальних ресурсів) в залежності від її вигляду і ступеню постійності формувались масиви лінійної структури двох типів: умовно-постійні (з інформацією, яка використовувалась багато разів протягом довгого часу) і умовно-перемінні (з фактичною або поточною інформацією). Створення і багаторазове використання масивів з умовно-постійною інформацією має ті переваги, які дозволяють значно спростити первинну документацію шляхом виведення з її складу ряду постійних реквізитів, знизити трудомісткість робіт на стадії заповнення первинних документів, підготовки і вводу фактичної або поточної інформації до ЕОМ [26]. Недоліком таких масивів, які мають лінійну структуру, є те, що інформація одного і того ж об’єкту управління розподіляється поміж багатьох різних масивів, що неминуче веде до дублювання деяких реквізитів, ускладненню при спільній їх обробці тощо, а головне – не дає змоги реалізувати принцип незалежності від прикладних програм користувача.

З розвитком інформаційного забезпечення систем автоматизованої обробки інформації, прагненням забезпечити виконання нових режимів обробки даних у реальному часі і з мультидоступом до схованих даних позначилась нова тенденція до складення інформаційного забезпечення розподілених баз даних. В умовах використання таких баз створюються комплексні масиви нелінійної структури, які мають усі дані про ту чи іншу предметну область або про керований об’єкт як постійного, так і перемінного характеру.

Взагалі база даних є сукупність даних на машинних носіях, які використовуються при функціонуванні системи обробки інформації, організовані по визначеним правилам, які передбачають загальні принципи описування збереження і маніпулювання ними, а також які незалежні від прикладних програм. В основі організації бази даних є модель даних, яка визначає правила, відповідно до яких структуруються дані. За допомогою моделі представляється велика кількість даних і описуються взаємні зв'язки між ними. Найбільш поширені такі моделі даних: ієрархічна, мережева, реляційна [27].

В ієрархічній моделі зв'язок даних «один до одного» (1:1) означає, що кожному значенню елемента даних *A* відповідає одне і тільки одне значення, пов'язаного з ним елемента *B*. Наприклад, поміж такими елементами пар даних, як код готової продукції і її найменування, є вищезазначений зв'язок, так як кожному коду продукції відповідає одне її найменування.

Зазначимо, що ієрархічна модель даних будується на основі принципу підпорядкованості поміж елементами даних і представляє собою деревоподібну структуру, яка складається із вузлів (сегментів) і дуг (гілок). Дерево у ієрархічній структурі упорядковане за існуючими правилами розташування його сегментів і гілок: доступ до кожного породженого (крім кореневого) здійснюється через його вихідний сегмент; кожний сегмент може мати по декілька екземплярів конкретних значень елементів даних, а кожний елемент породженого сегменту пов'язаний з екземпляром вихідного і створює один логічний запис; екземпляр породженого сегменту не може існувати самостійно, тобто без кореневого сегменту; при вилученні екземпляру кореневого сегмента також вилучаються усі підпорядковані і взаємопов'язані з ним екземпляри породжених сегментів.

В мережевій моделі зв'язок «один до багатьох» (1:M) означає, що значенню елемента *A* відповідають багато (більше одного) значень, пов'язаних з ним елементів *B*. Наприклад, поміж елементами даних «код виробу» (елемент *A*) і «кодом матеріалів» (елементи *B*) існує такий

взаємозв’язок, бо для виготовлення одного виробу використовується багато різних матеріалів.

Мережева модель даних представляє собою орієнтований граф з поіменованими вершинами і дугами. Вершини графа – записи, які представляють собою поіменовану сукупність логічних взаємозв’язаних елементів даних або агрегатів даних. Під агрегатом даних розуміють сукупність елементів даних, які є усередині запису. Для кожного типу записів може бути кілька екземплярів конкретних значень його інформаційних елементів.

В реляційній моделі зв’язок «багатьох до багатьох» (M:N) указує на те, що декільком значенням елементів даних *A* відповідає декілька значень елементів даних *B*. Наприклад, поміж елементами даних «код операції технологічного процесу» і «табельний номер працівника» існує зазначений взаємозв’язок, так як багато операцій технологічного процесу можуть виконувати різні працівники (табельні номери) і навпаки [28].

Реляційна модель даних являє собою набір двомірних плоских таблиць, що складаються з рядків і стовпців. Первинний документ або лінійний масив являє собою плоску двомірну таблицю. Така таблиця називається відношенням, кожний стовбець – атрибутом, сукупність значень одного типу (стовпця) – доменом, а рядка – кортежем. Таким чином, стовпці таблиці являються традиційними елементами даних, а рядки – записами. Таблиці (відношення) мають імена. Імена також присвоюються і стовпцям таблиці. Кожний кортеж (запис) відношення має ключ. Ключі є прості і складні. Простий ключ – це ключ, який складається з одного атомарного атрибуту, значення якого унікальне. Складний ключ складається з двох і більше атрибутів. Для зв’язків відношень друг з другом в базі даних є зовнішні ключі. Атрибут або комбінація атрибута відношення є зовнішнім ключем, якщо він не є основним (первинним) ключем цього відношення, але являється первинним ключем для другого відношення.

Різновидністю баз даних, з точки зору їх зберігання і використання, є розподіленні бази даних. Розподілена база даних – це сукупність логічно зв’язаних баз даних або частин однієї бази, які розподілені поміж декількома територіально-розподіленими ПЕОМ і забезпечені відповідними можливостями для управління цими базами або їх частинами. Тобто, розподілена база даних реалізується на різних просторово розосереджених обчислювальних засобах, разом з організаційними, технічними і програмними засобами її створення і ведення.

1.4.3 Основні властивості реляційних БД

Основи реляційної моделі даних були вперше викладені у статті Е. Кодда в 1970 році. Ця робота послужила стимулом для великої кількості статей і книг, в яких реляційна модель отримала подальший розвиток.

Найбільш поширене трактування реляційної моделі даних належить К. Дейту. Згідно Дейту, реляційна модель складається з трьох частин:

- структурної;
- цілісної;
- маніпуляційної.

Структурна частина описує, які об’єкти розглядаються реляційною моделлю. Постулюється, що єдиною структурою даних, що використовується в реляційній моделі, є нормалізовані парні відносини.

Цілісна частина описує обмеження спеціального виду, які повинні виконуватися для будь-яких відносин у будь-яких реляційних базах даних. Це цілісність сутностей і цілісність зовнішніх ключів.

Маніпуляційна частина описує два еквівалентних способу маніпулювання реляційними даними – реляційну алгебру і реляційне числення.

У реляційній моделі дані зберігаються у вигляді відношень, або реляцій, кожна з яких описує окремий предмет, ситуацію чи явище у вигляді одного і того ж набору характеристик. Якщо існує кілька однакових об’єктів, вводиться ще одна характеристика – унікальний номер предмету. У найпростішому вигляді ця модель реалізується у вигляді таблиці, стовпці якого містять характеристики, і називаються полями. Кожен об’єкт – це рядок таблиці, або, іншими словами, запис. Предметна область може характеризуватись кількома таблицями, що пов’язані між собою.

Реляційна модель має наступні характеристики:

– кожен рядок таблиці – це один елемент даних;

– кожне поле таблиці має певний тип даних, котрий може мати обмеження;

– кожне поле має унікальне ім’я;

– у таблиці немає однакових записів;

– порядок полів та записів може бути довільним, їх впорядкування – це окрема процедура.

Реляційні бази даних корисні при зберігання однорідної інформації, наприклад фінансових рахунків-фактур, трансакцій, надходжень на рахунки підприємства.

1.4.4 Можливості мови SQL

Історія SQL починається з 70-х років XX століття, коли в дослідницькій лабораторії IBM у штаті Каліфорнія було розроблено першу версію цієї мови. Назва SQL є абревіатурою від Structured Query Language (структурована мова запитів), й іноді її вимовляють як «sequel» (первісна назва). Спочатку ця мова була реалізована в реляційній СУБД DB2 виробництва IBM. На відміну від мов третього покоління (COBOL, C), які з’явилися в той самий час, мова SQL

не є процедурною. Непроцедурна мова – це мова, в якій описується, що потрібно одержати, а не як це зробити.

Особливість реляційних СУБД полягає у тому, що вони надають множинно-орієнтовану мову маніпулювання базами даних, тобто результатом дії мовного оператора є таблиця, яка містить множину даних. Більшість сучасних реляційних СУБД використовують саме мову SQL.

Американський інститут національних стандартів (American National Standards Institute – ANSI) та Міжнародна організація стандартів (International Standards Organization – ISO) займаються описом і підтримкою стандартів цієї мови. Усі сучасні СУБД підтримують певний стандарт, проте є й відхилення, які в кожному конкретному випадку специфікуються в документації програмного продукту. Окрім того, у багатьох системах розроблено розширення SQL, що дають змогу використовувати мову запитів у середовищі програмування [29].

SQL надає такі можливості:

– створювати й видаляти таблиці бази даних, а також змінювати заголовки таблиць;

– вставляти, змінювати й видаляти рядки в таблицях;

– виконувати пошук даних у багатьох таблицях та впорядковувати результати цього пошуку;

– описувати процедури підтримки цілісності;

– визначати та змінювати інформацію про захист даних.

1.4.5 CASE-засоби моделювання структури БД

Термін CASE використовується в даний час у досить широкому сенсі. Первісне значення терміну CASE, обмежене питаннями автоматизації розробки тільки програмного забезпечення, сьогодні набуло нового сенсу, що охоплює процес розробки складних ІС у цілому. Тепер під терміном

CASE-засобу розуміють програмні засоби, що підтримують процеси створення і супроводу ІС, включаючи аналіз і формулювання вимог, проєктування прикладного ПЗ (застосунків) і баз даних, генерацію коду, тестування, документування, забезпечення якості, конфігураційне керування і керування проєктом, а також інші процеси. CASE-засоби разом із системним ПЗ і технічними засобами утворюють повне середовище розробки ІС.

Появі CASE-технології і CASE-засобів передували дослідження в області методології програмування. Програмування набуло рис системного підходу з розробкою і впровадженням мов високого рівня, методів структурного і модульного програмування, мов проєктування і засобів їхньої підтримки, формальних і неформальних мов описів системних вимог і специфікацій [30].

CASE-технологія являє собою методологію проєктування ІС, а також набір інструментальних засобів, що дозволяють у наочній формі моделювати предметну область, аналізувати цю модель на всіх етапах розробки і супроводу ІС і розробляти застосунки відповідно до інформаційних потреб користувачів. Більшість існуючих CASE-засобів засновано на методологіях структурного (в основному) або об’єктно-орієнтованого аналізу і проєктування, що використовують специфікації у вигляді діаграм або текстів для опису зовнішніх вимог, зв’язків між моделями системи, динаміки поводження системи та архітектури програмних засобів.

Для розробки інформаційної моделі була використана така CASE-програма, як ERwin. ERwin не орієнтований на якусь конкретну СУБД і підтримує більше 20 типів СУБД, включаючи СУБД всіх провідних виробників серверів баз даних (Oracle, Sybase, Microsoft, IBM, Informix), а також усі популярні формати настільних СУБД (включаючи dBase, Clipper, FoxPro, Access, Paradox), крім, можливо, самих останніх версій. Справа в тому, що нові версії ERwin не випускалися вже досить давно. Тому при використанні ERwin з останніми версіями деяких СУБД можуть виникнути

проблеми (наприклад, деякі типи даних SQL Server 7.0 цей CASE-засіб підтримує не зовсім коректно). Тим не менш ERwin залишається одним з найпопулярніших в світі продуктів цього класу завдяки підтримці великої кількості платформ, простоті інтерфейсу і, що важливо, підтримці специфічних особливостей організації фізичної пам’яті найбільш популярних серверних СУБД. Наприклад, для СУБД Oracle, Microsoft SQL Server, Sybase цей продукт дозволяє змінювати місце розташування і параметри зберігання індексів, майже для всіх популярних серверних СУБД створювати кластеризовані індекси і для багатьох з них – вказувати характеристики табличних просторів і сегментів відкоту.

1.5 Постановка задачі

Об’єктом роботи є розробка інформаційної системи пошуку і придбання авіаквитків.

Метою роботи є створення інформаційної системи для вибору клієнтом рейсу та видачі інформації про рейси за запитом клієнтів, яка має 2 рівня доступу: адміністратор та клієнт.

Система повинна відображати інформацію про:

- країни, міста звідки та куди здійснюються рейси;
- маршрути рейсів, їх довжина;
- розклад маршрутів;
- типи літаків, на яких здійснюються рейси;
- рейси, їх дату, ціни на квитки різних класів;
- інформацію про пасажирів.

Користувач повинен створити акаунт, з якого він зможе обрати потрібний йому маршрут, день, клас та купити квиток. Акаунти користувачів не прив’язуються до конкретної особи, тому з одного акаунта можна купити квитки для декількох осіб.

Програма повинна надавати доступ для видалення, додавання, редагування наступної інформації:

– країни, міста звідки та куди здійснюються рейси;

– маршрути рейсів, їх довжина;

– розклад маршрутів, їх ціна;

– типи літаків, на яких здійснюються рейси;

– рейси, їх дату.

Для досягнення мети необхідно вирішити наступні завдання:

– вивчити такі теоретичні питання: теорію проєктування реляційних БД на основі побудови ER-моделі, основні принципи нормалізації БД, принципи побудови запитів до БД (мови DQL SQL, DML SQL, DDL SQL);

– ознайомитися з наступними інструментами проєктування і програмування ІС: CASE-засобом візуального проєктування даних ERwin, MS Visual Studio;

– вивчивши опис і вимоги до функціональності ІС, розробити бізнес-правила і глосарій;

– на основі бізнес-правил розробити ER-модель і відобразити її за допомогою ER-діаграми в синтаксисі Чена;

– розробити модель даних за допомогою CASE-засобу візуального проєктування даних ERwin;

– сформувати структуру БД в СУБД;

– розробити дизайн ІС;

– розробити програмний код для забезпечення необхідної функціональності ІС.

Для розробки ІС необхідно використовувати наступні інструменти:

– ERwin – CASE-засіб проєктування БД;

– MS SQL Server – СУБД;

– MS Visual Studio, C # – середа і мова розробки програмного забезпечення.

2 ПРОЄКТУВАННЯ БАЗИ ДАНИХ

2.1 Специфікація вимог

Інформаційна система повинна бути розроблена на мові програмування C# з використанням технології доступу до даних ADO.NET, тому що ця технологія включає до свого складу засоби для роботи з базами даних різних форматів. З їх допомогою можна виконувати будь-які SQL-запити та оброблювати їх результати, отримувати доступ до окремих таблиць бази даних, працювати з транзакціями, а також використовувати особливі моделі для виводу вмісту таблиць або запитів у любому з компонентів-представлень.

Для повноцінної роботи з програмою користувачу потрібно буде мати персональний комп’ютер, на якому він буде запускати систему, монітор, мишу та клавіатуру для вводу даних і взаємодії із застосунком. Рекомендована операційна система – Windows 7 або новіша.

2.2 Розробка бізнес-правил БД

Система займається продажем авіаквитків на різні рейси, веде облік проданих квитків і облік пасажирів, які купили квитки. Проаналізувавши ПО можна виділити такі бізнес-правила:

- 1 пасажир може купити лише 1 квиток на конкретний рейс;
- кожен маршрут може мати 1 пункт призначення;
- 1 літак може мати багато рейсів;
- кількість місць бізнес та економ класу залежить від моделі літака;
- коефіцієнт рейсу залежить від дати вильоту;
- ціна бізнес класу перевищує ціну економ класу у 50 відсотків;
- ціна квитка залежить від класу місця;

– знижка для окремого пасажира залежить від кількості здійснених їм перельотів.

Проаналізувавши ці бізнес правила можна перейти до побудови функціональної структури даної предметної області.

2.3 Розробка концептуальної моделі

Концептуальне моделювання дозволяє врахувати логічне уявлення структури даних у базі даних. Вірно розроблена модель бази даних має підтримувати усі явлення користувачів. Концептуальне моделювання є основою подальшого проєктування бази даних та застосунку для її обробки.

Розробка концептуальної моделі предметної області є найважливішим етапом проєктування бази даних, не орієнтованим на конкретну СУБД. Концептуальна модель предметної області будується першою та полягає у структуризації наочної області: об’єкти реального світу піддаються класифікації, фіксується сукупність тих, що підлягають відображенню в БД. Для кожного об’єкту фіксується сукупність властивостей, за допомогою яких описуватимуться конкретні екземпляри об’єкту, і відношення з іншими об’єктами. Потім вирішуються питання про те, яка інформація про об’єкти повинна бути представлена в БД і як її представити за допомогою даних. Опис кожної сутності показано у таблиці 2.1.

Таблиця 2.1 – Відомості про типи сутностей

Назва сутності	**Опис**
1	**2**
Країна	Назва країни
Місто	Назва міста
Маршрут	Існуючі маршрути

Продовження таблиці 2.1

1	**2**
Розклад	Розклад для конкретних маршрутів
Літак	Назва моделі літака та кількість місць для неї
Рейс	Рейс для конкретних маршрутів на певну дату
Пасажир	Інформація про пасажирів

Далі встановлено типи зв’язків між обраними сутностями в моделі даних (табл. 2.2).

Таблиця 2.2 – Відомості про типи зв’язків між сутностями

Назва сутності	**Назва сутності**	**Зв’язок**
Країна	Місто	1:M
Місто	Маршрут	1:M
Маршрут	Розклад	1:M
Літак	Рейс	1:M
Розклад	Рейс	1:M
Рейс	Пасажир	M:N

На основі цих таблиць побудуємо концептуальну модель даних у синтаксисі Чена (рис. 2.1).

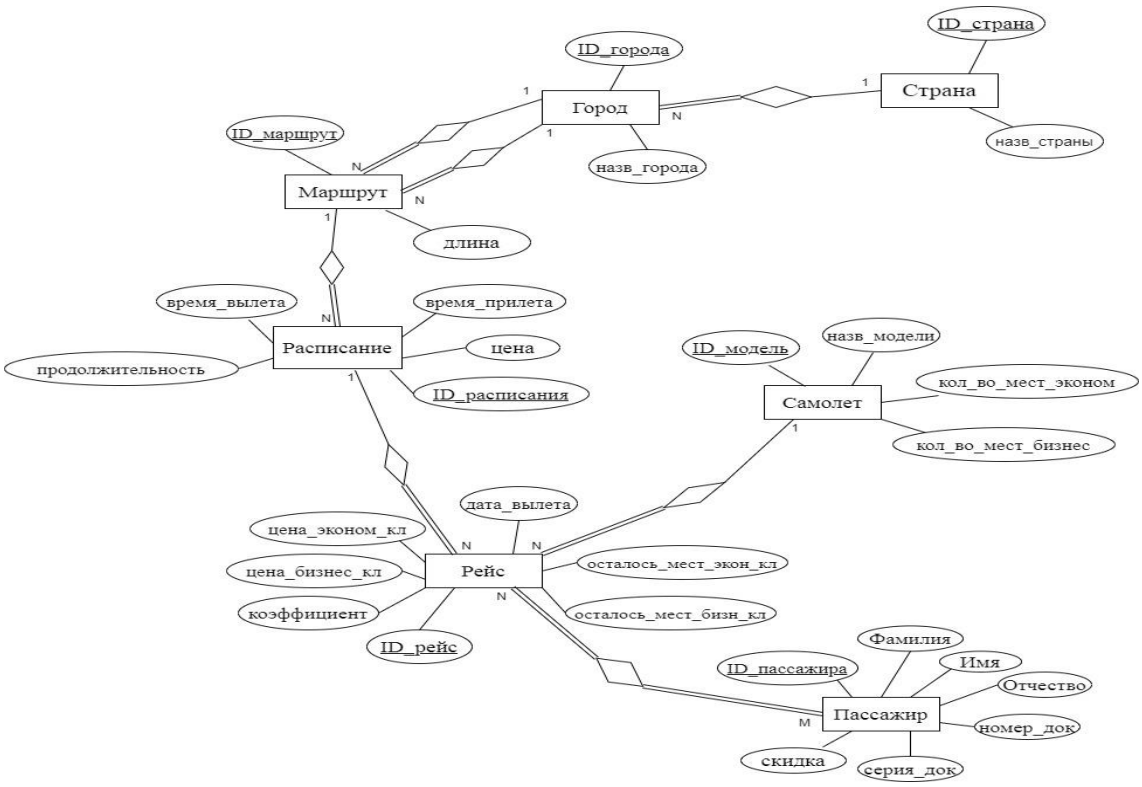

Рисунок 2.1 – Концептуальна модель даних

2.4 Побудова та перевірка логічної моделі

Для переходу з концептуальної моделі даних до логічної потрібно створити слабкі сутності, які будуть поєднувати сутності, що мають зв’язок M:N. Слабка сутність має обов’язково містити первинні ключі сутностей, між якими встановлюється зв’язок.

Логічна модель є основою бази даних, вона повинна відображати взаємозв’язки між реляційними таблицями. Між реляційними таблицями можуть бути наступні типи зв’язків 1:1, 1:M та M:N. Найбільш поширеним зв’язком є зв’язок 1:M. Зв’язок M:N безпосередньо не підтримується в реляційних СУБД.

Сутність знаходиться в 1НФ тоді і тільки тоді, коли всі атрибути містять атомарні значення. Серед атрибутів не повинно зустрічатися повторюваних груп (кілька значень для кожного екземпляра). Сутність знаходиться у 2НФ, якщо вона знаходиться в 1НФ і кожен не ключовий

атрибут повністю функціонально залежить від первинного ключа (не повинно бути залежності від частини ключа). Друга нормальна форма має сенс тільки для сутностей, що мають складовий первинний ключ. Сутність знаходиться в 3НФ формі, якщо вона знаходиться в другій нормальній формі і ніякий не ключових атрибут не залежить від іншого не ключового атрибута (не повинно бути взаємозалежності між не ключових атрибутами). Ця концептуальна модель знаходиться у 3 формі.

Побудуємо логічну модель у синтаксисі Чена (рис. 2.2).

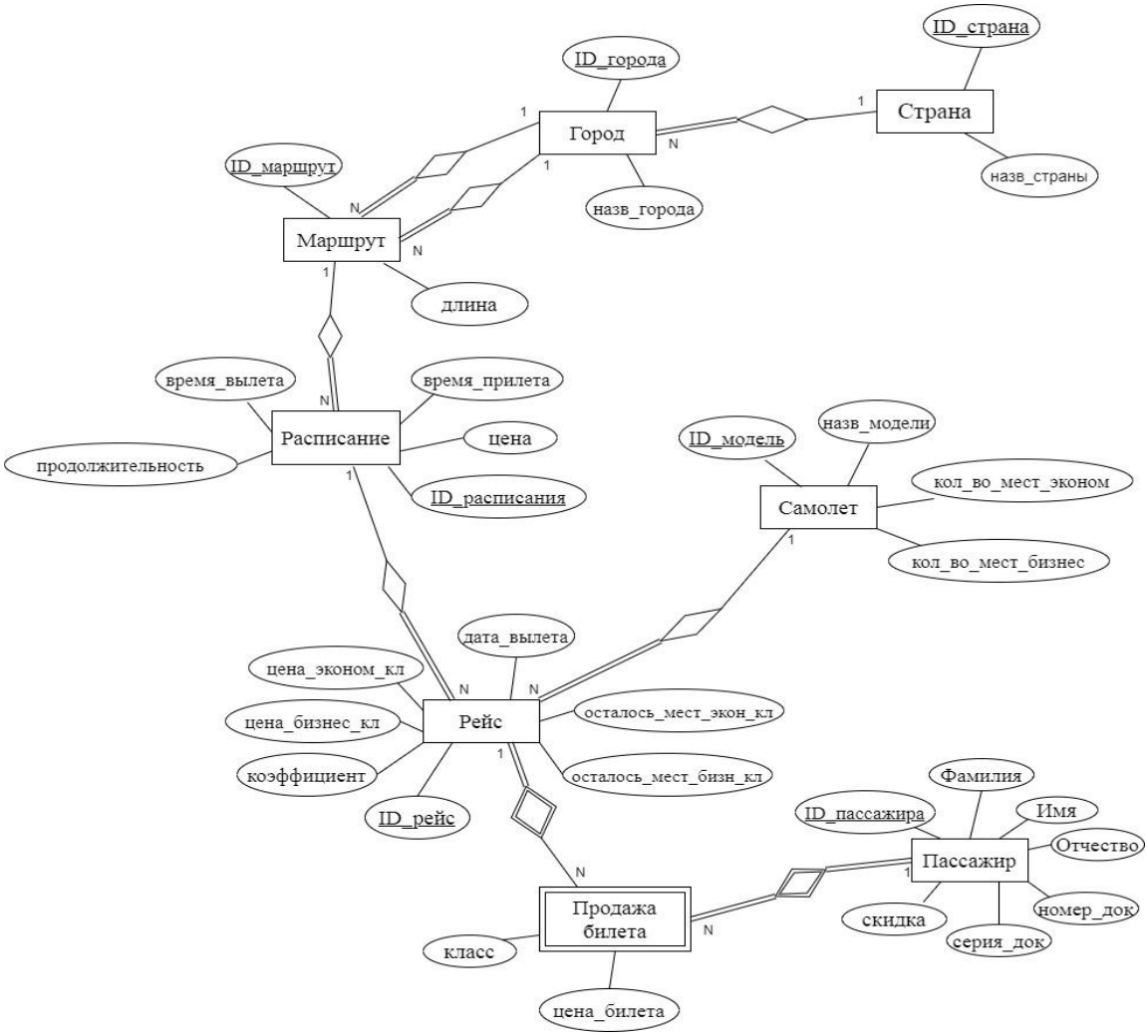

Рисунок 2.2 – Логічна модель даних

У цій ER-діаграмі наявні 8 таблиць. Кожна з таблиць має свій список атрибутів. Зробимо розкриття для цих таблиць.

Страна (ID_страна, назв_страны)

Город (ID_города, назв_города, ID_страна)

Маршрут (ID_маршрут, длина, пункт_отправления, пункт_назначения)

Расписание (ID_расписания, ID_маршрут, время_вылета, время_прилета, продолжительность, цена)

Самолет (ID_модель, назв_модели, кол_во_мест_эконом, кол_во_мест_бизнес)

Рейс (ID_рейс, ID_расписания, ID_модель, дата_вылета, коэффициент, цена_эконом_класса, цена_бизнес_класса, осталось_мест_экон_кл, осталось_мест_бизн_кл)

Продажа билета (ID_ рейс, ID_пассажира, класс, цена_билета)

Пассажир (ID_пассажира, Фамилия, Имя, Отчество, номер_док, серия_док, скидка)

2.5 Побудова моделі даних за допомогою CASE-засобу візуального проєктування ERWIN

На основі концептуальної та логічної моделі побудуємо модель даних за допомогою програми візуального проєктування ERWIN (рис. 2.3).

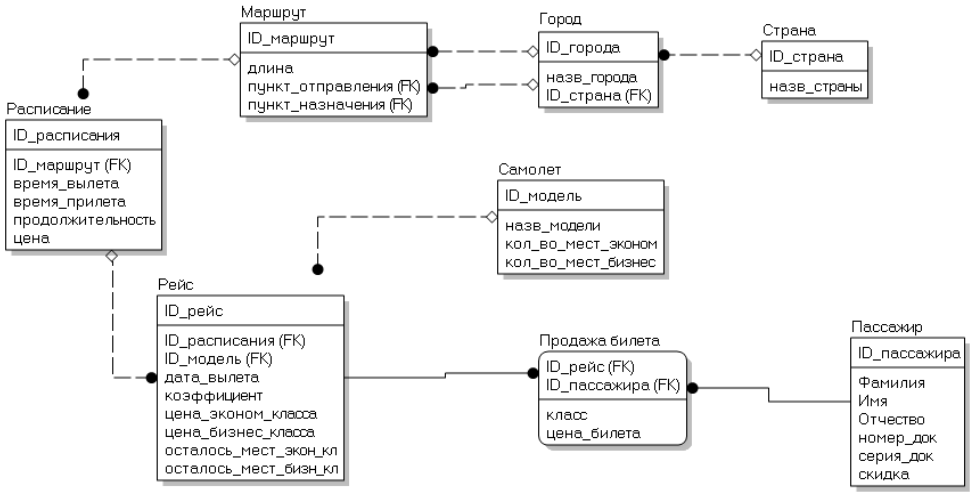

Рисунок 2.3 – Логічна модель в ERWIN

2.6 Фізичне проєктування

Результатом переносу моделі даних з ERWIN у SQL Management Studio є діаграма, яка зображена на рисунку 2.4.

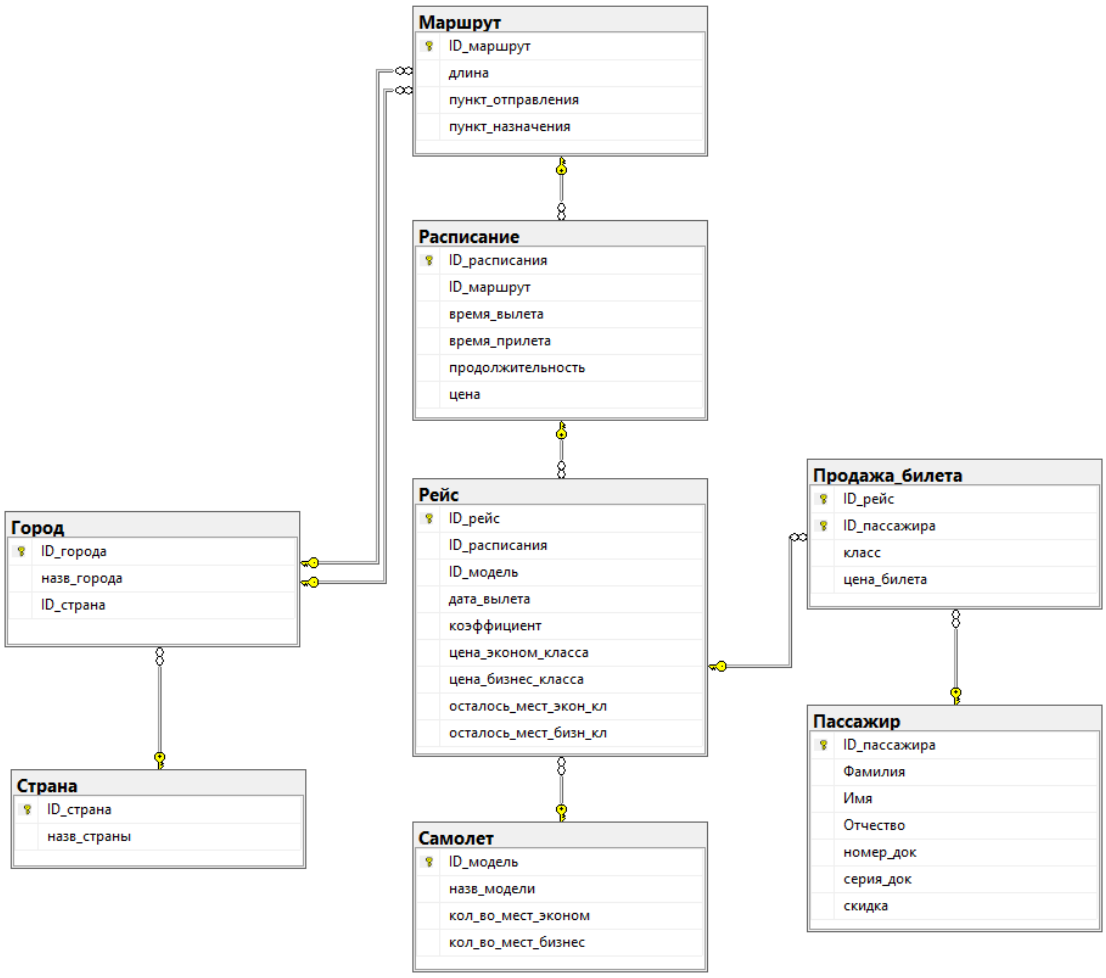

Рисунок 2.4 – Згенерована модель даних у SQL Server

Фізична структура бази даних для інформаційної системи пошуку і придбання авіаквитків наведена нижче за допомогою відповідних команд CREATE TABLE.

Структура таблиці «Страна»:

```
CREATE TABLE Страна
(
ID_страна INT NOT NULL,
```

назв_страны VARCHAR (20) NOT NULL,

CONSTRAINT PK_Страна PRIMARY KEY CLUSTERED (ID_страна ASC)

).

Структура таблиці «Город»:

CREATE TABLE Город

(

ID_города INT NOT NULL,

назв_города VARCHAR (20) NOT NULL,

ID_страна INT NOT NULL,

CONSTRAINT PK_Город PRIMARY KEY CLUSTERED (ID_города ASC),

CONSTRAINT FK_Город_Страна FOREIGN KEY (ID_страна) REFERENCES Страна (ID_страна) ON DELETE CASCADE ON UPDATE CASCADE

).

Структура таблиці «Маршрут»:

CREATE TABLE Маршрут

(

ID_маршрут INT NOT NULL,

длина INT NULL,

пункт_отправления INT NOT NULL,

пункт_назначения INT NOT NULL,

CONSTRAINT PK_Маршрут PRIMARY KEY CLUSTERED (ID_маршрут ASC),

CONSTRAINT FK_Маршрут_Город FOREIGN KEY (пункт_отправления) REFERENCES Город (ID_города) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FK_Маршрут_Город1 FOREIGN KEY (пункт_назначения) REFERENCES Город (ID_города)).

Структура таблиці «Расписание»:

```
CREATE TABLE Расписание
(
   ID_расписания INT NOT NULL,
   ID_маршрут INT NOT NULL,
   время_вылета TIME (0) NOT NULL,
   время_прилета TIME (0) NOT NULL,
   продолжительность TIME (0) NOT NULL,
   цена MONEY NOT NULL,
   CONSTRAINT PK_Расписание PRIMARY KEY CLUSTERED (ID_расписания ASC),
   CONSTRAINT FK_Расписание_Маршрут FOREIGN KEY (ID_маршрут) REFERENCES Маршрут (ID_маршрут) ON DELETE CASCADE ON UPDATE CASCADE
).
```

Структура таблиці «Самолет»:

```
CREATE TABLE Самолет
(
   ID_модель VARCHAR (10) NOT NULL,
   назв_модели VARCHAR (20) NOT NULL,
   кол_во_мест_эконом INT NOT NULL,
   кол_во_мест_бизнес INT NOT NULL,
   CONSTRAINT PK_Самолет PRIMARY KEY CLUSTERED (ID_модель ASC)
).
```

Структура таблиці «Рейс»:

```
CREATE TABLE Рейс
(
   ID_рейс  INT NOT NULL,
   ID_расписания  INT NOT NULL,
```

ID_модель VARCHAR (10) NOT NULL,

дата_вылета DATE NOT NULL,

коэффициент FLOAT (53) NULL,

цена_эконом_класса MONEY NULL,

цена_бизнес_класса MONEY NULL,

осталось_мест_экон_кл INT NULL,

осталось_мест_бизн_кл INT NULL,

CONSTRAINT PK_Рейс PRIMARY KEY CLUSTERED (ID_рейс ASC),

CONSTRAINT FK_Рейс_Самолет FOREIGN KEY (ID_модель) REFERENCES Самолет (ID_модель) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FK_Рейс_Расписание FOREIGN KEY (ID_расписания) REFERENCES Расписание (ID_расписания) ON DELETE CASCADE ON UPDATE CASCADE

).

Структура таблиці «Продажа_билета»:

CREATE TABLE Продажа_билета

(

ID_рейс INT NOT NULL,

ID_пассажира INT NOT NULL,

класс VARCHAR (6) NOT NULL,

цена_билета MONEY NULL,

CONSTRAINT PK__Продажа PRIMARY KEY CLUSTERED (ID_рейс ASC, ID_пассажира ASC),

CONSTRAINT FK_Продажа_билета_Пассажир FOREIGN KEY (ID_пассажира) REFERENCES Пассажир (ID_пассажира) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FK_Продажа_билета_Рейс FOREIGN KEY (ID_рейс) REFERENCES Рейс (ID_рейс) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT CK_Продажа_билета CHECK (класс='бизнес' OR класс='эконом')

).

Структура таблиці «Пассажир»:

```
CREATE TABLE Пассажир
(
   ID_пассажира INT IDENTITY (1, 1) NOT NULL,
   Фамилия  VARCHAR (25) NOT NULL,
   Имя  VARCHAR (20) NOT NULL,
   Отчество VARCHAR (25) NOT NULL,
   номер_док CHAR (9)  NULL,
   серия_док  CHAR (2)  NULL,
   скидка  INT  NULL,
   CONSTRAINT PK_Пассажир PRIMARY KEY CLUSTERED (ID_пассажира ASC)
).
```

3 ПРОЄКТУВАННЯ ТА РОЗРОБКА ІС

3.1 Дизайн ІС

Під час запуску програми відкривається головна форма, зображена на рисунку 3.1, яка представляє собою вікно для авторизації. В залежності від логіну і паролю, який введе користувач, відкриється конкретна форма, яка відповідає статусу акаунта. Усього їх два – клієнт і адміністратор.

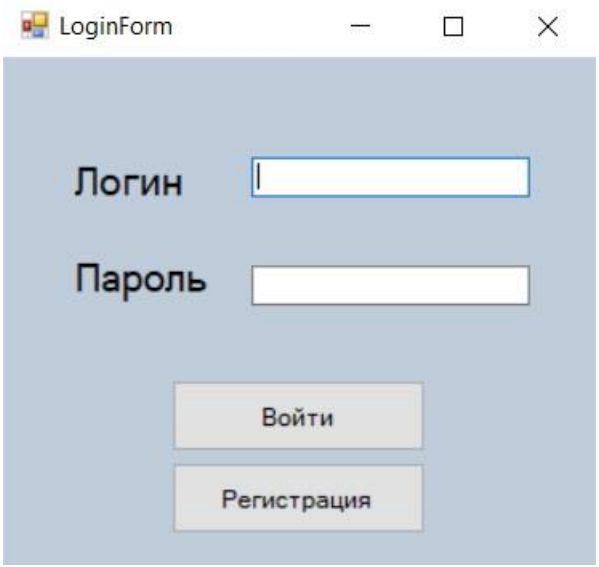

Рисунок 3.1 – Форма авторизації

Розглянемо один з типів доступу, а саме – клієнтський рівень доступу. Для того, щоб зайти на цю форму необхідно правильно ввести логін и пароль, за якими закріплена форма клієнта (рис. 3.2).

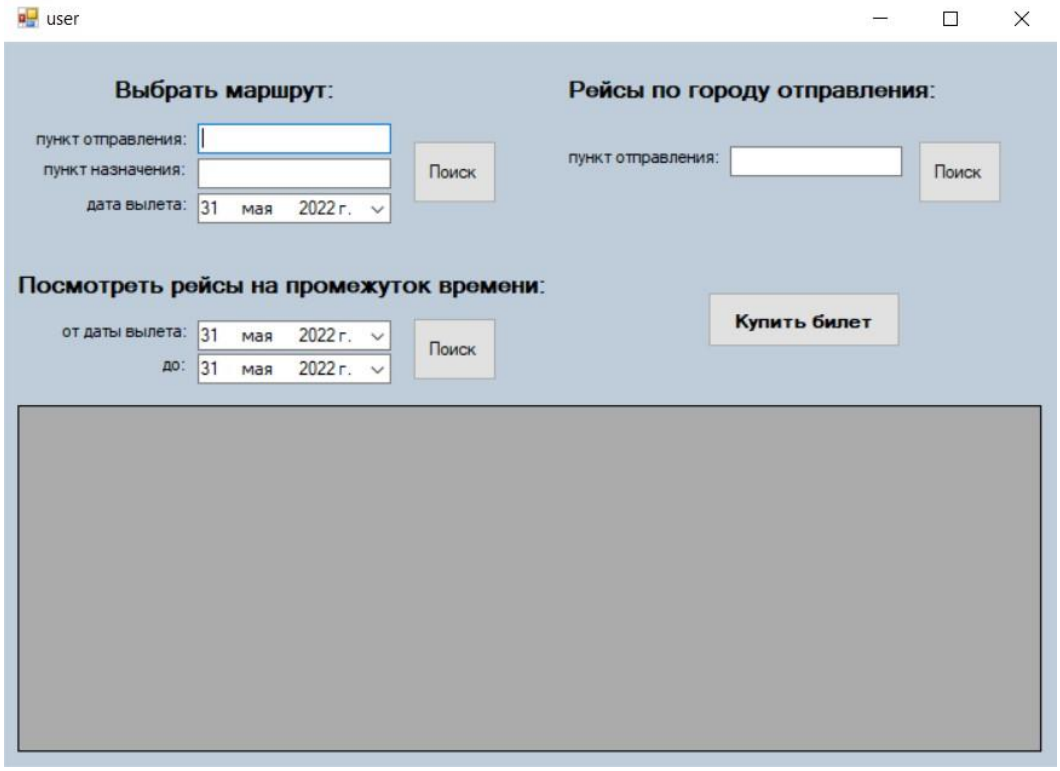

Рисунок 3.2 – Форма клієнта

На цій формі користувач може подивитися рейси на конкретну дату, на потрібний проміжок часу, а також з певного міста. Також він може придбати квиток (рис. 3.3).

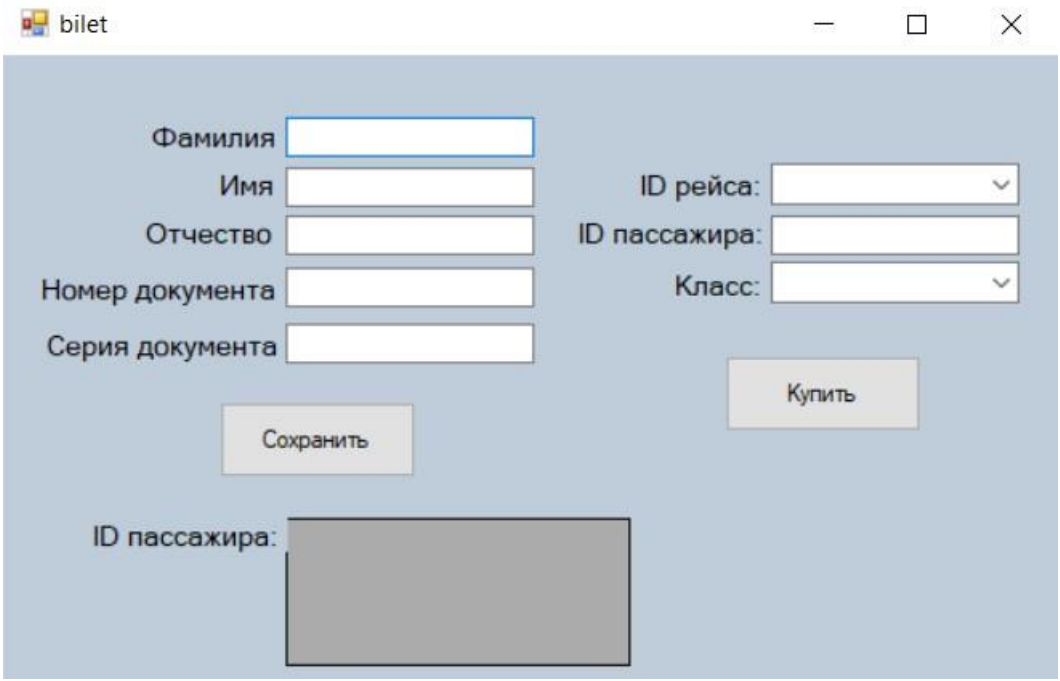

Рисунок 3.3 – Форма для придбання квитків

Додавати дані до таблиць можна лише через форму адміністратора (рис. 3.4).

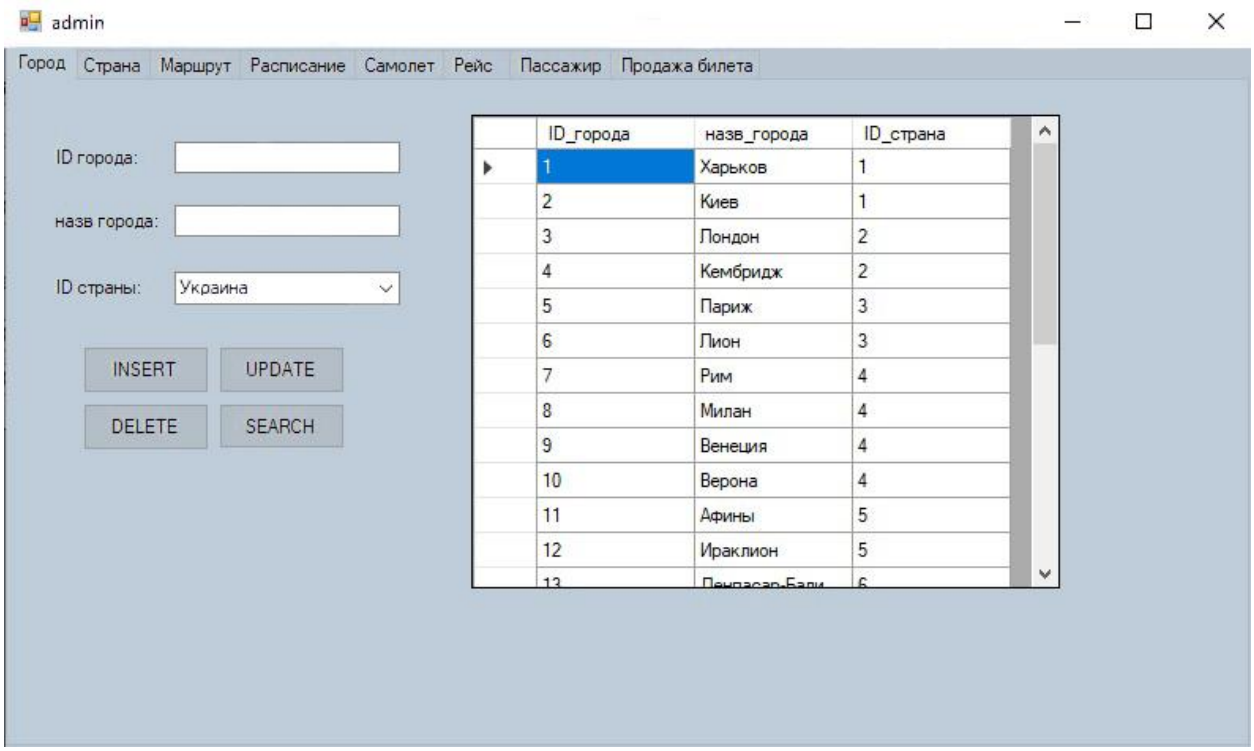

Рисунок 3.4 – Форма адміністратора

3.2 Забезпечення необхідної функціональності ІС

Наступний SQL-запит ілюструє оновлення даних в таблиці «Рейс»:

UPDATE Рейс SET коэффициент =(CASE WHEN дата_вылета between '2018-12-19' and '2019-01-19' THEN 1.3 WHEN дата_вылета between '2019-12-19' and '2020-01-19' THEN 1.3 WHEN дата_вылета between '2019-05-01' and '2019-07-01' THEN 1.3 WHEN дата_вылета between '2019-07-01' and '2019-09-07' THEN 1.4 ELSE 1 END);

UPDATE Рейс SET цена_эконом_класса = (select коэффициент * (select цена from Расписание where Рейс.ID_расписания = Расписание.ID_расписания)from Рейс as P where ID_рейс = Рейс.ID_рейс);

UPDATE Рейс SET цена_бизнес_класса = цена_эконом_класса * 1.5;

UPDATE Рейс SET осталось_мест_экон_кл = (select кол_во_мест_эконом - (select COUNT(*) from Продажа_билета where Продажа_билета.ID_рейс = Рейс.ID_рейс AND класс = 'эконом') from Самолет where ID_модель = Рейс.ID_модель);

UPDATE Рейс SET осталось_мест_бизн_кл = (select кол_во_мест_бизнес - (select COUNT(*) from Продажа_билета where Продажа_билета.ID_рейс = Рейс.ID_рейс AND класс = 'бизнес') from Самолет where ID_модель = Рейс.ID_модель);.

Наступний SQL-запит ілюструє оновлення даних в таблиці «Продажа билета»:

UPDATE Продажа_билета SET цена_билета =(SELECT скидка* (SELECT CASE WHEN класс = 'эконом' THEN(SELECT цена_эконом_класса FROM Рейс WHERE Рейс.ID_рейс = Продажа_билета.ID_рейс) WHEN класс = 'бизнес' THEN(SELECT цена_бизнес_класса FROM Рейс WHERE Рейс.ID_рейс = Продажа_билета.ID_рейс) END FROM Продажа_билета T WHERE Продажа_билета.ID_рейс = T.ID_рейс AND Продажа_билета.ID_пассажира = T.ID_пассажира) FROM Пассажир WHERE ID_пассажира = Продажа_билета.ID_пассажира);.

Цей SQL-запит ілюструє оновлення даних в таблиці «Пассажир»:

UPDATE Пассажир SET скидка =(select (CASE WHEN COUNT(*) > 3 THEN 5 WHEN COUNT(*) > 5 THEN 10 WHEN COUNT(*) > 10 THEN 25 ELSE 1 END) FROM Продажа_билета where ID_пассажира = Пассажир.ID_пассажира);.

3.3 Тестування ІС

Перевіримо необхідну функціональність програми. Для того, щоб знайти рейс на необхідну дату, треба ввести пункт відправлення та призначення у відповідні області та визначити дату вильоту (рис. 3.5).

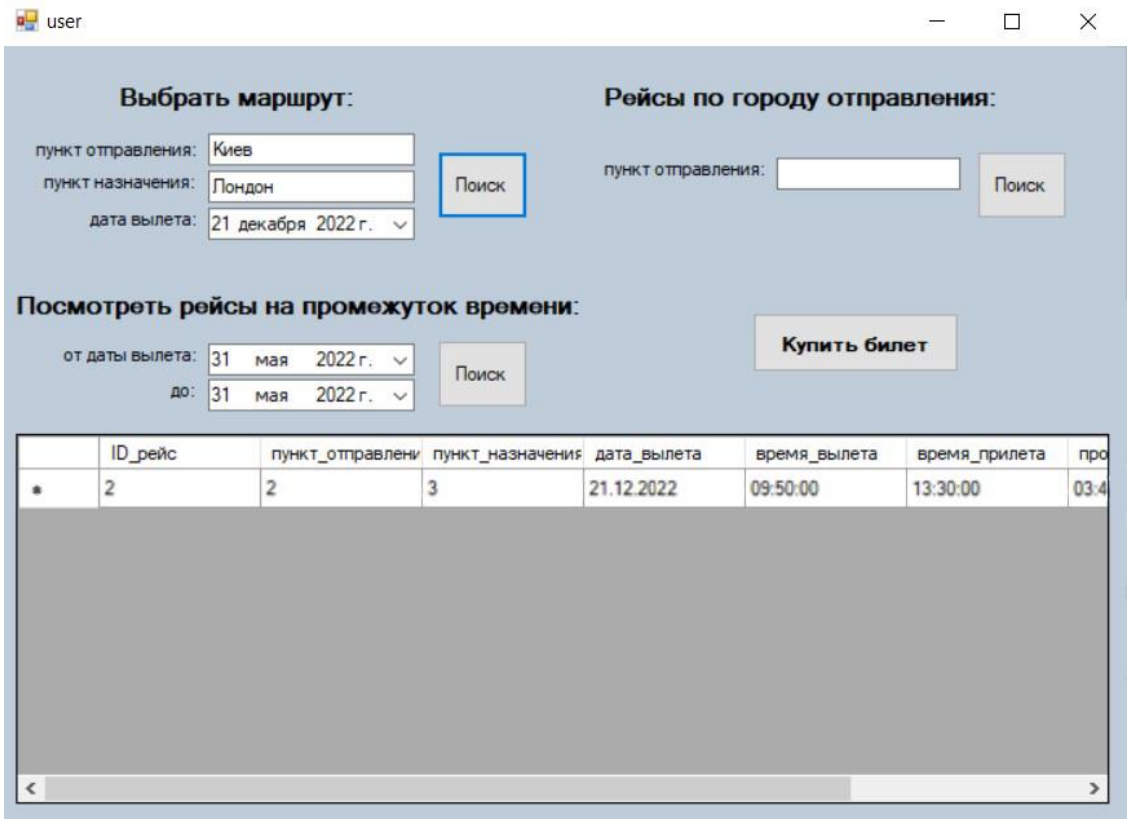

Рисунок 3.5 – Пошук рейсу

Для купівлі квитка треба ввести усі необхідні дані у відповідні області та визначити свій ID пасажира (рис. 3.6). Усі інші основні функціональні можливості застосунку показані на рисунках А.1 – А.6.

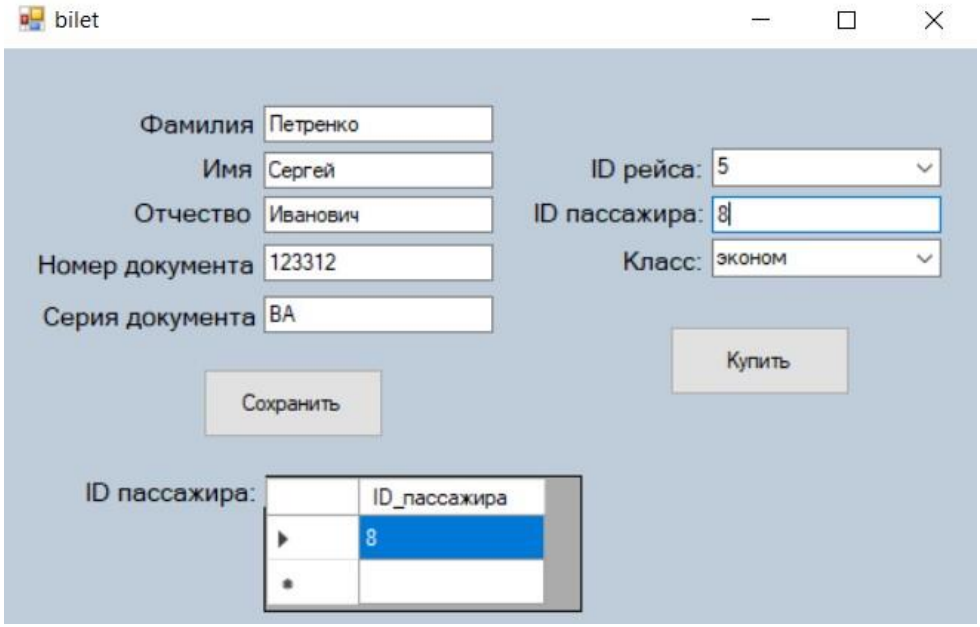

Рисунок 3.6 – Купівля квитка

ВИСНОВКИ

У рамках кваліфікаційної роботи була реалізована інформаційна система пошуку і придбання авіаквитків. Було створено відповідну базу даних та десктопний застосунок. У ході реалізації роботи було вивчено теоретичний матеріал, що стосується баз даних, їх різновидів та логічних моделей, опрацьовано основи роботи з ними, отримано досвід з проєктування баз даних та розробки застосунків, що працюють з базами даних.

Інформаційна система пошуку і придбання авіаквитків передбачає весь функціонал, який потребує користувач застосунку. Для виявлення необхідних користувачу функцій та його потреб було проведено аналіз предметної області та концептуальне моделювання. На основі результатів було сформульовано вимоги до програмного продукту, які були надалі реалізовані в застосунку. Виділені в процесі аналізу об’єкти предметної області та відносини між ними були відображені в логічній моделі бази даних.

Усі поставлені вимоги до інформаційної системи було виконано. Відтак, результатом кваліфікаційної роботи став застосунок для пошуку і придбання авіаквитків, який охоплює всі необхідні об’єкти взаємодії: перегляд, додавання, видалення, редагування інформації, а також сортування, пошук та фільтрацію.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Вендров, А. М. (1998). CASE-технологии. Современные методы и средства проектирования информационных систем. М.: Финансы и статистика, 176, 2000.

2. Грекул, В. И., Денищенко, Г. Н., & Коровкина, Н. Л. (2008). Проектирование информационных систем: учеб. пособие. Москва: Интернет-Университет Информационных Технологий: БИНОМ. Лаборатория знаний.

3. Железко, Б. А., & Подгорная, Г. Н. (2017). Проектирование и эксплуатация информационных систем: учебная программа учреждения высшего образования по учебной дисциплине для специальности 1-25 01 12 «Экономическая информатика».

4. Коцюба, И. Ю., Чунаев, А. В., & Шиков, А. Н. (2015). Основы проектирования информационных систем.

5. Дейт, К. Д. (2008). Введение в системы баз данных. Вильямс.

6. Михалик, Д. М., & Петрик, О. Ю. (2015). Методичні вказівки до виконання курсового проєкту з дисципліни «Бази даних».

7. Швецов, В. И. (2018). Принципы организации электронного курса «Базы данных». Образовательные технологии и общество, 21(1), 449-468.

8. Тарасов, С. В. (2015). СУБД для программиста. Базы данных изнутри.

9. Крамаренко, Т. А., Деменков, И. А., & Михеев, А. И. (2016). Выбор клиент-серверной СУБД для реализации информационной системы. Современные информационные технологии, 24(24), 11-15.

10. Павленко, П. М., Філоненко, С. Ф., Бабіч, К. С., Гавриленко, О. В., & Логачов, Є. Г. (2013). Інформаційні системи і технології.

11. Антоненко, В. М., Мамченко, С. Д., & Рогушина, Ю. В. (2016). Сучасні інформаційні системи і технології: управління знаннями.

12. Ясінецька, І., & Мушеник, І. (2020). ІНФОРМАЦІЙНІ СИСТЕМИ І ТЕХНОЛОГІЇ В УПРАВЛІННІ ДІЯЛЬНІСТЮ ПІДПРИЄМСТВА. Збірник наукових праць ΛΌГОΣ, 66-67.

13. Юрчук, Н. П. (2015). Інформаційні системи в управлінні діяльністю підприємства. Агросвіт, (19), 53-58.

14. Адамик, О. В., & Сисюк, С. В. (2016). Інформаційні системи управління підприємством: вибір базових технологій та програмного забезпечення.

15. Ломачинська, І. М., & Ломачинський, Б. Г. (2020). Роль інформаційної культури у регулюванні соціальних інформаційних систем. Вісник Львівського університету, 94.

16. Адамик, О. В. (2017). Інформаційні системи і технології в обліку й аудиті.

17. Карпенко, М. Ю., Манакова, Н. О., & Гавриленко, І. О. (2017). Технології створення програмних продуктів та інформаційних систем: навчальний посібник.

18. Набойщикова, Є. О., & Осипенко, В. В. (2020). Архітектура інформаційних систем. Електромеханічні та інформаційні системи. Київський національний університет технологій та дизайну.

19. Євланов, М. В. (2018). Удосконалений метод синтезу варіантів опису архітектури створюваної інформаційної системи.

20. Адамик, О. В., & Адамик, К. Б. (2018). Реляційні бази даних як сучасний стандарт накопичення інформації в комп'ютерній системі бухгалтерського обліку.

21. Шаряк, В. (2008). Методи дослідження системних характеристик моделей бази даних. Вісник Тернопільського державного технічного університету, 13(2), 116-121.

22. Melton, J., & Simon, A. R. (1993). Understanding the new SQL: a complete guide. Morgan Kaufmann.

23. Armbrust, M., Xin, R. S., Lian, C., Huai, Y., Liu, D., Bradley, J. K., ... & Zaharia, M. (2015, May). Spark sql: Relational data processing in spark. In Proceedings of the 2015 ACM SIGMOD international conference on management of data (pp. 1383-1394).

24. Lu, J., & Holubová, I. (2019). Multi-model databases: a new journey to handle the variety of data. ACM Computing Surveys (CSUR), 52(3), 1-38.

25. Li, G., Zhou, X., & Cao, L. (2021, October). Machine learning for databases. In The First International Conference on AI-ML-Systems (pp. 1-2).

26. Hejlsberg, A., Wiltamuth, S., & Golde, P. (2003). C# language specification. Addison-Wesley Longman Publishing Co., Inc..

27. Новікова, О. М., Новикова, Е. Н., Паламар, А. Ю., Паламар, А. Ю., Мазикіна, О. Б., Мазыкина, О. Б., ... & Сидоренко, В. Д. (2022). SQL запити для ГІС.

28. Колесник, Л. В., Кириченко, Н. А., & Костоглот, І. В. (2018). Розробка засобу проектування високонавантажених реляційних систем зберігання даних: оптимізація структури та запитів SQL. Проблеми інформаційних технологій, (1), 253-260.

29. Єрмолаєв, О. Д. (2021). Метод оптимізації SQL запитів (Bachelor’s thesis, КПІ ім. Ігоря Сікорського).

30. Коваленко, М. Д. (2021). Методи оптимізації запитів у реляційних БД на основі SQL (Bachelor’s thesis, КПІ ім. Ігоря Сікорського).