МIНIСТЕРСТВО ОСВIТИ I НАУКИ УКРАЇНИ

НАЦIОНАЛЬНИЙ УНIВЕРСИТЕТ «ОДЕСЬКА ПОЛIТЕХНIКА»

Кафедра iнформацiйних технологiй

КУРСОВИЙ ПРОЕКТ

По курсу: «Організація баз даних»

Варiант №36

Роботу виконав:

Студент 2 курсу групи АД-231

Спеціальності 126 – «Iнформацiйнi системи

та технологiї»

Нерянов Владислав Павлович

Керівник:Гришин Сергій

Національна шкала\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів:\_\_\_\_\_Оцінка:ECTS\_\_\_\_\_

Одеса 2024

ОПИС ЗАВДАННЯ

Обстеження предметної області з залученням інформаційних джерел

Предметна область: Готельне господарство

Готельне господарство охоплює різноманітні аспекти управління готелем: реєстрацію гостей, надання номерів, оплату за проживання, бронювання номерів, оформлення від'їзду гостей, а також організацію обслуговування номерів, у тому числі різні послуги, які можуть бути надані клієнтам. Основними завданнями адміністратора готелю є:

* Реєстрація гостей: збір особистих даних клієнта, включаючи паспортні дані, місце роботи, мету перебування.
* Номерний фонд: кожен номер має свою категорію, кількість кімнат, кількість місць, наявність меблів та техніки (телевізор, холодильник тощо).
* Оплата за проживання: обчислення вартості номеру на основі кількості осіб, категорії номера та тривалості перебування.
* Бронювання номерів: можливість попереднього резервування номерів, а також оформлення та зберігання інформації про бронювання.
* Виписка гостей та перерахунок вартості: облік дострокових від'їздів, перерахунок плати за неочікуваний від'їзд до кінцевої дати.

Інформаційні джерела для розробки цієї системи можуть включати:

* Опис процесів готельного управління з наданими рекомендаціями для оптимізації адміністративних функцій.
* Дані про номери: характеристики номерів, наявність меблів, електроніки, особливості обслуговування (наприклад, наявність балконів або кондиціонерів).
* Інформація про послуги: перелік додаткових послуг, таких як сніданки, трансфери, SPA-послуги тощо.
* Правила реєстрації та оплати: нормативні документи та внутрішні правила готелю, що регулюють взаємодію з клієнтами.

Перелік даних, що зберігаються в системі:

1. Гості: інформація про гостей, їхні персональні дані, паспортні дані, місце роботи, мета приїзду тощо.
2. Номери: інформація про категорії номерів, кількість кімнат, поверхи, наявність телевізора, холодильника, балкона, кількість місць тощо.
3. Бронювання: дані про заброньовані номери, дату приїзду та від'їзду, ПІБ особи, що забронювала.
4. Оплата: інформація про плату за номер, спосіб оплати, квитанції, дату від'їзду та інші фінансові дані.

ЗМІСТ

Вступ……………………………………………………………………………….4

Основна частина…………………………………………………………………..6

* Перелік бізнес-функції, організація БД…………………….…………….6
* Пошук у таблицях SQL……………………………………………………9
* Розрахунок підсумків……………………………………………………..11
* Пошук даних за допомогою з’єднання таблиць………………………...13
* Модифікація даних SQL………………………………………………….15
* Пошук за допомогою підпорядкованих запитів………………………...17
* Використання уявлень і табл. виразів…………………………………...18
* Обробка даних збереженими процедурами……………………………..20
* Обробка даних тригерами………………………………………………..23
* Розробка програми. Форми CRUD………………………………………25

ВСТУП

Проектування та реалізація бази даних для системи управління діяльністю готелю є важливою складовою частиною автоматизації готельного бізнесу. У сучасних умовах ефективне управління готелем вимагає обробки великої кількості даних про гостей, номери, бронювання та фінансові операції. Ці дані повинні бути збережені, оброблені та надані адміністраторам і співробітникам готелю у вигляді, що дозволяє швидко й ефективно виконувати різноманітні функції, такі як реєстрація гостей, бронювання номерів, облік оплат, а також надання різноманітних послуг.

Основною метою цього проекту є розробка бази даних для автоматизації процесів, пов'язаних з управлінням готелем, що включає в себе зберігання інформації про гостей, номери, бронювання та оплату. Система має бути зручним інструментом для адміністраторів, дозволяючи здійснювати оперативну реєстрацію гостей, управління доступними номерами, а також бронювання та оплату.

У результаті виконання цього проекту буде створено ефективну та надійну базу даних, що дозволяє автоматизувати ключові процеси в роботі готелю, забезпечити збереження важливої інформації та спростити доступ до неї для користувачів системи.

ОСНОВНА ЧАСТИНА

2. Визначення переліку бізнес-функцій системи

Для ефективного функціонування системи управління готелем необхідно реалізувати низку бізнес-функцій, що дозволяють адміністратору готелю виконувати основні операції з обробки даних. Перелік основних бізнес-функцій системи включає:

1. Реєстрація гостя: внесення персональних даних нового гостя, створення унікального ідентифікатора та збереження інформації в системі.
2. Бронювання номера: можливість бронювання номера для гостей з певною датою приїзду та від'їзду, визначення наявності вільних номерів.
3. Заселення гостя: прив'язка зареєстрованого гостя до конкретного номера на визначений період.
4. Оплата за проживання: облік фінансових операцій, обчислення вартості проживання, збереження даних про оплату (готівкою чи безготівково).
5. Виписка та від'їзд гостя: процедура завершення перебування, здійснення розрахунків при ранньому виїзді та реєстрація факту від'їзду.
6. Звітність та статистика: можливість перегляду даних про кількість заселених номерів, доходи від проживання та інші показники для аналізу ефективності роботи готелю.
7. Управління номерами: створення, зміна та видалення інформації про номери, включаючи категорії, кількість кімнат, зручності тощо.

Схема бази даних

Схема бази даних буде включати таблиці та їхні зв'язки:

* Таблиця Guests: Зберігає інформацію про гостей, їхні особисті дані та платіжну інформацію.
* Таблиця Rooms: Інформація про номери (кількість кімнат, наявність меблів, TV, холодильників тощо).
* Таблиця Reservations: Зберігає дані про бронювання (ім'я забронювавшої особи, дати прибуття і від'їзду).

1. Зв'язок між таблицею Guests та таблицею Reservations:

* Тип зв'язку: Один до багатьох
* Опис: Кожен гість може мати кілька бронювань, але кожне бронювання належить лише одному гостю.
* Зв'язок: Поле GuestID в таблиці Reservations є зовнішнім ключем, що посилається на поле GuestID в таблиці Guests.

2. Зв'язок між таблицею Rooms та таблицею Reservations:

* Тип зв'язку: Один до багатьох
* Опис: Кожна кімната може бути зарезервована лише один раз в конкретний період, але може бути багато бронювань для різних періодів.
* Зв'язок: Поле RoomNumber в таблиці Reservations є зовнішнім ключем, що посилається на поле Number в таблиці Rooms.

3. Зв'язок між таблицею Rooms та таблицею RoomTypes:

* Тип зв'язку: Багато до одного (N:1)
* Опис: Кожна кімната має певний тип (наприклад, стандарт, люкс і т. д.). В таблиці Rooms є поле Type, яке є зовнішнім ключем до таблиці RoomTypes.
* Зв'язок: Поле Type в таблиці Rooms є зовнішнім ключем, що посилається на поле Type в таблиці RoomTypes.

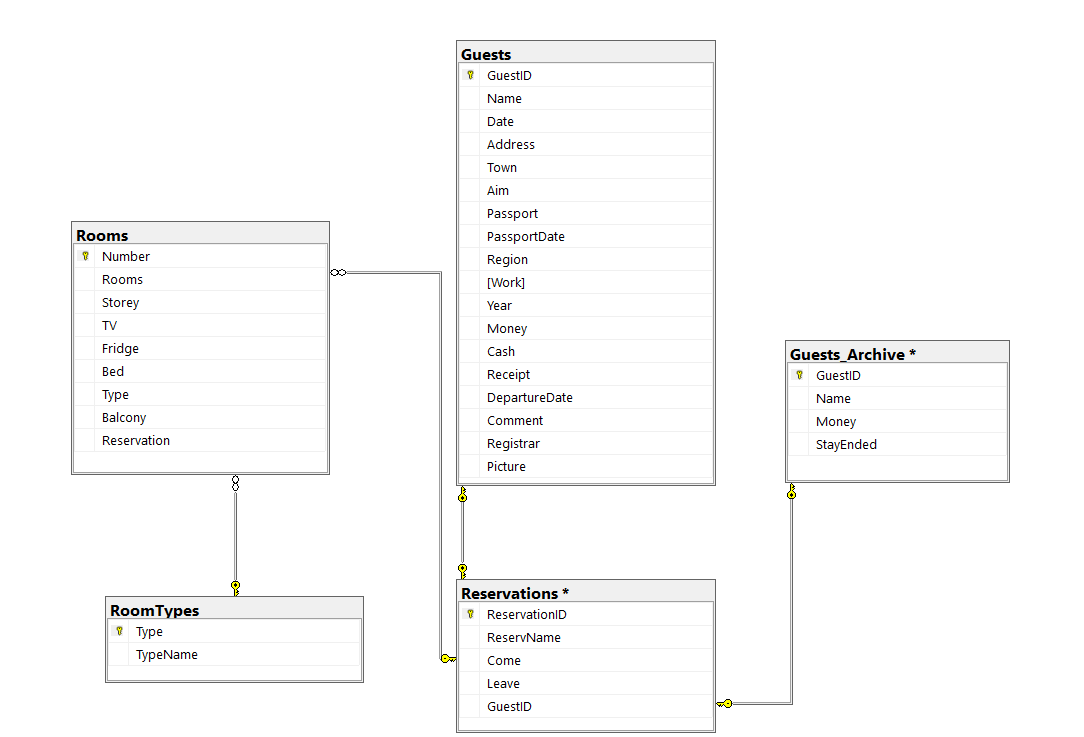


Рис 2.1 Діаграма системи

3. Організація пошуку у таблицях бази даних з допомогою SQL запитів

Опис функції пошуку:

1. Пошук гостей за іменем: Запит дозволяє знайти всіх гостей, чиє ім'я містить вказане значення (наприклад, "Олена"). Це корисно, коли необхідно здійснити пошук за іменем чи частиною імені.
2. Пошук бронювань за ID гостя: Цей запит шукає всі бронювання, які були зроблені певним гостем. Пошук здійснюється через внутрішній підзапит, який знаходить ID гостя в таблиці Guests і потім шукає всі відповідні записи в таблиці Reservations.
3. Пошук номерів за типом номера: Запит дозволяє знайти всі номери, які належать до певного типу (наприклад, супер-люкс). Це дає змогу адміністраторам готелю фільтрувати доступні номери за категоріями.
4. Пошук номерів, що зарезервовані на певну дату: Цей запит шукає всі номери, які мають активні бронювання на певну дату. Це дозволяє перевірити, які номери вже зайняті на заданий день.
5. Пошук гостей, які приїхали після певної дати: Запит використовується для пошуку гостей, які приїхали до готелю після зазначеної дати (наприклад, після 1 листопада 2024 року). Це може бути корисно для визначення гостей, які прибули пізніше, ніж певна дата.
6. Пошук гостей за містом: Запит дозволяє знайти всіх гостей, які проживають у вказаному місті. Це дозволяє адміністраторам готелю отримати список гостей за географічним розподілом.
7. Пошук номерів з певними зручностями: Запит шукає всі номери, що мають певні зручності (наприклад, телевізор і балкон). Це дозволяє гостям або адміністраторам готелю знаходити номери, які відповідають їх вимогам по зручностях.
8. Мета пошукових запитів:  
   Пошукові запити в системі готелю дозволяють здійснювати ефективний доступ до даних про гостей, номери та бронювання на основі різних критеріїв (ім'я, тип номера, дата бронювання, зручності тощо).
9. Типи пошукових функцій:  
   Запити покривають різні аспекти роботи готелю, такі як пошук по імені гостя, пошук по бронюванням, фільтрація номерів за типами та наявністю зручностей, а також пошук по місту проживання гостей.
10. Використання запитів у практиці:  
    Ці запити використовуються адміністраторами готелю для моніторингу бронювань, пошуку потрібних гостей, а також для управління номерами в залежності від запитів клієнтів.

Один із пошуків у запросі:

SELECT \*

FROM Guests

WHERE Town = 'Київ';

Лістинг 3.1 Пошук гостей, що проживають у певному місті

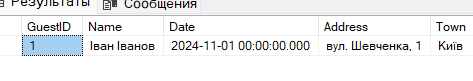


Рис 3.1 Вивід користувача

Ще один пошук:  
SELECT \*

FROM Rooms

WHERE TV = 1 AND Balcony = 1;

Лістинг 3.2 Пошук номерів з певними зручностями(телевізор і балкон)

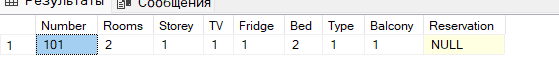


Рис 3.2 Вивід номеру зі зручностями

1. Розрахунок підсумків у таблицях бази даних з допомогою  агрегатних функцій

Основні агрегатні функції:

1. SUM() — ця функція обчислює суму значень у зазначеному стовпці. Вона використовується для підрахунку загального доходу або інших числових підсумків.
2. AVG() — функція обчислює середнє значення для вказаного стовпця. Це корисно, якщо потрібно дізнатися середню суму, яку витрачає кожен гість.
3. MIN() — ця функція повертає мінімальне значення з вибраного стовпця, що дозволяє знайти найменше значення (наприклад, найменший дохід).
4. MAX() — аналогічно функції MIN(), але вона повертає максимальне значення з вибраного стовпця.
5. COUNT() — ця функція підраховує кількість рядків, які задовольняють умови. Вона корисна для підрахунку кількості записів, наприклад, кількості гостей або номерів з певними характеристиками.

Основний запит з агрегованими даними:

SELECT

SUM(g.Money) AS TotalRevenue,

AVG(g.Money) AS AverageSpend,

COUNT(CASE WHEN g.Date = '2024-11-01' THEN 1 END) AS GuestCount,

COUNT(CASE WHEN r.Balcony = 1 THEN 1 END) AS RoomsWithBalcony,

MAX(g.Money) AS MaxMoney,

MIN(g.Money) AS MinMoney,

AVG(g.Money) AS AvgMoney,

COUNT(CASE WHEN r.Reservation IS NULL THEN 1 END) AS AvailableRooms,

COUNT(CASE WHEN r.Reservation IS NOT NULL THEN 1 END) AS ReservedRooms

FROM Guests g

JOIN Rooms r ON r.Reservation = g.Receipt

WHERE g.Date BETWEEN '2024-11-01' AND '2024-11-05';

Лістинг 4.1 Запит для розрахунку підсумків

За результатами запиту було отримано такі підсумки:

* TotalRevenue — загальний дохід від гостей за вказаний період.
* AverageSpend — середня сума витрат одного гостя.
* GuestCount — кількість гостей, які заїхали 1 листопада 2024 року.
* RoomsWithBalcony — кількість номерів із балконом.
* MaxMoney і MinMoney — максимальні та мінімальні витрати серед гостей.
* AvailableRooms і ReservedRooms — кількість вільних і зайнятих номерів відповідно.



Рис 4.1 Вивід даних

1. Пошук даних у складених сутностей з допомогою запитів на з’єднання та об’єднання таблиць
2. Основні функції у запиті:  
   Виведення даних про гостей та їх бронювання:
   * Запит повертає такі дані, як:
     + Ім'я гостя (GuestName),
     + Номер кімнати (RoomNumber),
     + Тип кімнати (RoomType),
     + Назва бронювання (ReservationName),
     + Дата заїзду (CheckInDate),
     + Дата виїзду (CheckOutDate),
     + Сума, витрачена гостем (GuestMoney). Ці дані з'єднуються за допомогою таблиць Guests, Reservations та Rooms.
3. З’єднання таблиць:
   * Таблиці Guests, Reservations і Rooms з'єднуються через відповідні ключі:
     + GuestID в таблицях Guests і Reservations,
     + ReservationID в таблицях Reservations і Rooms. Це дозволяє витягнути повну інформацію про гостей, їх бронювання і відповідні номери.
4. Фільтрація даних по датах:
   * Запит обмежує результати за допомогою умови WHERE, щоб показувати тільки ті записи, де дата заїзду (Come) знаходиться в діапазоні між 1 листопада 2024 року і 30 листопада 2024 року.
   * Це дозволяє відображати тільки актуальні бронювання для зазначеного періоду.
5. Перевірка доступності номерів:
   * Умова AND (r.Reservation IS NULL OR r.Number NOT IN ...) перевіряє, чи є вільні номери:
     + Якщо номер не заброньований (поля Reservation відсутнє або має значення NULL), він відображається в результатах.
     + Якщо номер заброньований, перевіряється, чи не перекривається його бронювання з іншими датами. Якщо заїзд і виїзд перекриваються, цей номер не з’являється у списку.
6. Сортування результатів:
   * Результати сортуються за датою заїзду (CheckInDate), що дозволяє побачити бронювання у хронологічному порядку.

SELECT

g.Name AS GuestName,

r.Number AS RoomNumber,

r.Type AS RoomType,

res.ReservName AS ReservationName,

res.Come AS CheckInDate,

res.Leave AS CheckOutDate,

g.Money AS GuestMoney

FROM

Guests g

JOIN

Reservations res ON g.GuestID = res.GuestID

JOIN

Rooms r ON res.ReservationID = r.Reservation

WHERE

res.Come BETWEEN '2024-11-01' AND '2024-11-30'

ORDER BY

res.Come;

Лістинг 5.1 Запит

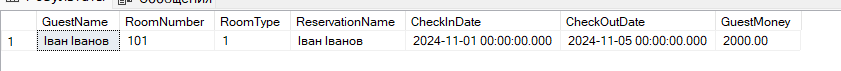


Рис 5.1 Вивід запиту

1. Модифікація даних з допомогою SQL запитів
2. Оновлення даних дозволяє змінювати вже існуючі записи, що корисно для відображення змін у витратах, бронюваннях або інші параметри, що можуть змінюватися протягом часу.
3. Вставка нових даних необхідна для додавання нових записів, таких як нові гості, бронювання або будь-які інші сутності, що з’являються в процесі функціонування готелю.
4. Видалення даних використовується для очищення таблиць від записів, які більше не потрібні, таких як виїзд гостей.
5. Вставка даних за допомогою SELECT дозволяє копіювати або переміщати дані між таблицями на основі певних критеріїв.
6. Модифікація даних через JOIN дає змогу виконувати зміни у таблиці, залежно від даних з інших таблиць, що підвищує ефективність обробки інформації в системі.

Використання в запиті:

1. Оновлення витрат гостя (UPDATE)  
   Змінюється сума витрат гостя з GuestID = 1, додаючи 100 одиниць. Це може бути пов’язано з додатковими послугами, наприклад, харчуванням чи користуванням спа.
2. Додавання нового гостя (INSERT)  
   Додається новий запис для гостя John Doe з початковими витратами 0 та датою приїзду 2024-11-10. Це використовується для реєстрації нового гостя.
3. Видалення гостя (DELETE)  
   Видаляється запис гостя з GuestID = 4, який вже покинув готель. Ця операція дозволяє зберігати актуальність даних.
4. Перенесення архівних даних (INSERT INTO ... SELECT)  
   Додаються записи з архівної таблиці Guests\_Archive для гостей, які завершили перебування в іншому готелі. Це дозволяє зберігати інформацію про попередніх клієнтів для аналізу.
5. Оновлення витрат для спеціальних пропозицій (UPDATE з JOIN)  
   Витрати гостей, які скористалися спеціальною пропозицією, збільшуються на 50 одиниць. Це дає змогу автоматизувати процес нарахування додаткових витрат за акції.

INSERT INTO Guests\_Archive (GuestID, Name, Money, StayEnded)

SELECT GuestID, Name, Money, DepartureDate

FROM Guests

WHERE DepartureDate IS NOT NULL

AND GuestID NOT IN (SELECT GuestID FROM Guests\_Archive);

Лістинг 6.1 Один із запитів



Рис 6.1 Дані з таблиці Guest перенесено у Guest\_Archive

7. Пошук даних у складених сутностей з допомогою підпорядкованих запитів

Дана функція реалізує пошук та вибірку даних із двох таблиць бази даних: Guests (Гості) та Reservations (Бронювання). Вона виконується за допомогою складених запитів, включаючи підпорядковані запити для фільтрації даних, що дозволяє знайти гостей, які витратили більше середнього значення, а також мають хоча б одне бронювання.

Завдання програми:

1. Потрібно вибрати дані з таблиці Guests для гостей, які витратили більше середнього значення грошей серед усіх гостей.
2. Використовуються підзапити для фільтрації даних:

* Підзапит для обчислення середнього значення витрачених грошей усіма гостями.
* Підзапит для вибору гостей, які мають хоча б одне бронювання в таблиці Reservations.

1. Пошук виконано таким чином, щоб:

* Вивести всіх гостей, чий рівень витрат більший або рівний середньому значенню.
* Знайти лише тих гостей, які мають активні бронювання.

Переваги реалізації:

* Оптимізація пошуку: Завдяки використанню підзапитів програма може ефективно працювати з великими обсягами даних, виконуючи фільтрацію та вибірку на основі середніх значень та наявності певних подій (бронювання).
* Гнучкість: Подібний підхід дозволяє гнучко працювати з різними типами запитів і змінювати критерії пошуку відповідно до потреб користувача.
* Універсальність: Вибірка даних із складених сутностей дозволяє ефективно працювати з таблицями, які мають взаємозв'язки, що є важливим для аналізу та звітності в контексті роботи з базами даних.

SELECT GuestID, Name, Money

FROM Guests

WHERE Money >= (SELECT AVG(Money) FROM Guests)

AND GuestID IN (SELECT GuestID FROM Reservations);

Лістинг 7.1 Запит що вибирає гостей які витратили більше середнього значення (серед усіх гостей)

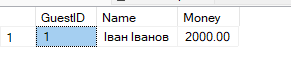


Рис 7.1 Вивід запиту

1. Використання уявлень і табличних виразів для обробки даних з допомогою

Табличні вирази (table expressions) в SQL — це тимчасові результати,

які використовуються для представлення даних, подібно до таблиць. Вони дозволяють спрощувати складні запити та зробити їх більш читабельними. Існує кілька видів табличних виразів, які можуть бути корисні при обробці даних:

1) Звичайний (деривативний) табличний вираз

2) CTE (Common Table Expression — Загальний табличний вираз)

3) Табличні вирази з функцією JOIN

WITH ReservationCount AS (

SELECT GuestID, COUNT(ReservationID) AS NumberOfReservations

FROM Reservations

GROUP BY GuestID

)

SELECT \* FROM ReservationCount;

WITH ReservationCount AS (

SELECT GuestID, COUNT(ReservationID) AS NumberOfReservations

FROM Reservations

GROUP BY GuestID

)

SELECT COUNT(\*) AS GuestsWithMultipleReservations

FROM ReservationCount

WHERE NumberOfReservations > 1;

Лістинг 8.1 Запит

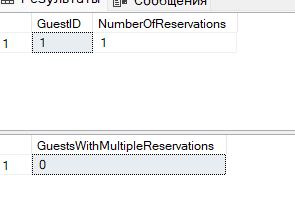


Рис 8.1 На даний момент вивід виглядає так

У запиті їде підрахування кількості гостей, які зробили більше ніж одне бронювання в системі. Це може бути корисно для аналізу повторюваних відвідувачів, планування акцій або спеціальних пропозицій для постійних клієнтів, а також для статистичних цілей.

Як працює запит:

1. Табличне вираження ReservationCount використовується для підрахунку кількості бронювань для кожного гостя. Для цього вибираються всі унікальні GuestID з таблиці бронювань та підраховується кількість бронювань кожного гостя.
2. Потім, за допомогою фільтрації, обираються лише ті записи, де кількість бронювань для гостя більше одного.
3. Нарешті, за допомогою підсумкової функції COUNT(), підраховується загальна кількість таких гостей.

Переваги:

* Аналіз лояльності клієнтів: дає можливість визначити постійних клієнтів, що роблять багато бронювань, що корисно для програм лояльності або спеціальних пропозицій для таких гостей.
* Оптимізація ресурсів: знання, скільки гостей повертаються, може допомогти в оптимізації управління готелем, наприклад, при розподілі номерів або підготовці до високого попиту.
* Статистика та звітність: запит корисний для формування звітів про активність клієнтів, що дозволяє відслідковувати тенденції та зростання числа постійних клієнтів.

1. Використання параметрів при обробці даних з допомогою збережених процедур

Збережені процедури — це попередньо скомпільовані SQL-скрипти, які зберігаються на сервері бази даних і можуть виконуватися багаторазово. Вони дозволяють виконувати складні операції з даними, використовуючи параметри, щоб зробити запити більш гнучкими.

Переваги використання:

1. Оптимізація продуктивності:

* Процедури виконуються швидше, оскільки вони вже скомпільовані.
* Зменшується навантаження на мережу, оскільки передається тільки виклик процедури, а не великий SQL-запит.

1. Безпека:

* Обмежує прямий доступ до таблиць.
* Можна керувати доступом до даних через права на виконання процедур.

1. Повторне використання коду:

* Процедуру можна використовувати в різних частинах програми.
* Зменшує дублювання коду.

1. Гнучкість:

* Збережені процедури дозволяють приймати параметри, що робить їх універсальними для виконання запитів з різними умовами.

Даний запит дозволяє швидко визначити, які номери доступні для бронювання на зазначений користувачем період. Це допомагає уникнути конфліктів між бронюваннями, покращує обслуговування клієнтів та оптимізує управління готелем.

CREATE PROCEDURE GetAvailableRooms

@StartDate DATE,

@EndDate DATE

AS

BEGIN

SELECT Number AS RoomID, Type AS RoomCategory, Bed AS Beds, Storey

FROM Rooms

WHERE Number NOT IN (

SELECT Reservation

FROM Reservations

WHERE (Come <= @EndDate AND Leave >= @StartDate)

);

END;

Лістинг 9.1 Процедура пошуку номерів

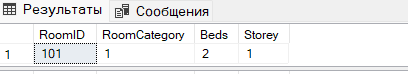


Рис 9.1 Вивід

Ця процедура дозволяє швидко знайти вільні номери на певний період, що спрощує роботу адміністратора готелю.

1. Використання тригерів при обробці даних

Тригери — це спеціальні об'єкти бази даних, які автоматично виконують певні дії у відповідь на події (додавання, оновлення або видалення записів) у зазначених таблицях. Вони дозволяють забезпечити цілісність і узгодженість даних без необхідності ручного втручання.

Основні характеристики тригерів:

1. Автоматичність:

* Виконуються автоматично після або перед вказаними подіями (INSERT, UPDATE, DELETE).

1. Прив'язка до таблиць:

* Кожен тригер прив'язаний до певної таблиці та спрацьовує на основі змін у ній.

1. Можливість модифікації даних:

* Тригери можуть додатково змінювати дані в інших таблицях, вести журнал операцій або перевіряти бізнес-правила.

Переваги тригерів:

* Автоматизація: мінімізують необхідність виконання рутинних завдань вручну.
* Цілісність даних: забезпечують виконання бізнес-правил навіть у разі, якщо дані вводяться з різних джерел.
* Логування: дозволяють вести аудит змін у таблицях.

CREATE TRIGGER trg\_UpdateRoomStatus

ON Reservations

AFTER INSERT, DELETE, UPDATE

AS

BEGIN

UPDATE Rooms

SET Reservation = (SELECT ReservationID FROM INSERTED)

WHERE Number IN (SELECT Reservation FROM INSERTED);

UPDATE Rooms

SET Reservation = NULL

WHERE Number IN (SELECT Reservation FROM DELETED);

END;

GO

Лістинг 10.1 Тригер, що оновлює статус кімнат

Триггер trg\_UpdateRoomStatus автоматично оновлює статус кімнат в системі при зміні даних у таблиці Reservations. Він спрацьовує після додавання, видалення або оновлення записів у таблиці бронювань Reservations і оновлює поле, яке містить інформацію про поточне бронювання кімнат в таблиці Rooms.

Опис роботи тригера:

1. Після вставки нового бронювання (INSERT):  
   Коли додається нове бронювання в таблицю Reservations, триггер оновлює таблицю Rooms, присвоюючи номер бронювання в поле Reservation відповідній кімнаті. Це вказує, що кімната заброньована.
2. Після оновлення існуючого бронювання (UPDATE):  
   Коли змінюються дані в таблиці Reservations (наприклад, дати заїзду або виїзду), триггер оновлює статус кімнати в таблиці Rooms, щоб відобразити нову інформацію про бронювання.
3. Після видалення бронювання (DELETE):  
   Коли бронювання видаляється з таблиці Reservations, триггер звільняє кімнату, видаляючи номер бронювання в таблиці Rooms (встановлюючи значення NULL в поле Reservation).
4. Розробка клієнтської програми. Форми CRUD.

Щоб розробити форми CRUD(создавати, читати, обновляти, видаляти) я зробив так:

На MainForm додав дві кнопки. Одна відповідальна за дії з таблицею Гостей(редагування, додання, видалення). Друга кнопка відповідальна за бронювання номерів.

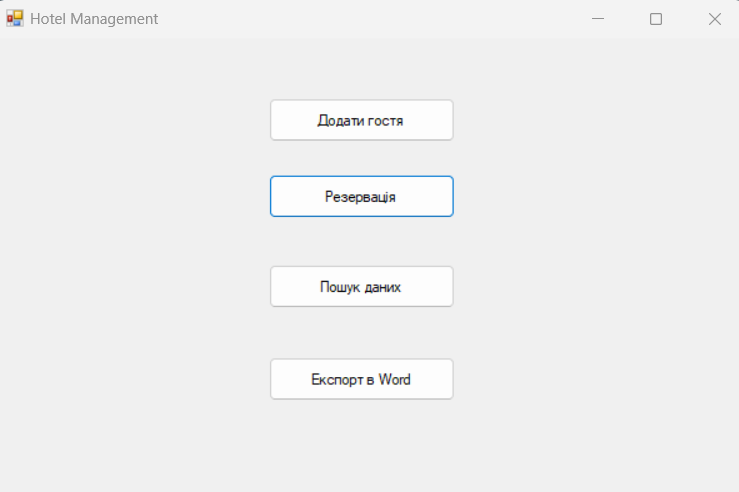


Рис 9.1 Вигляд головного вікна MainForm

Форма Додавання Гостей

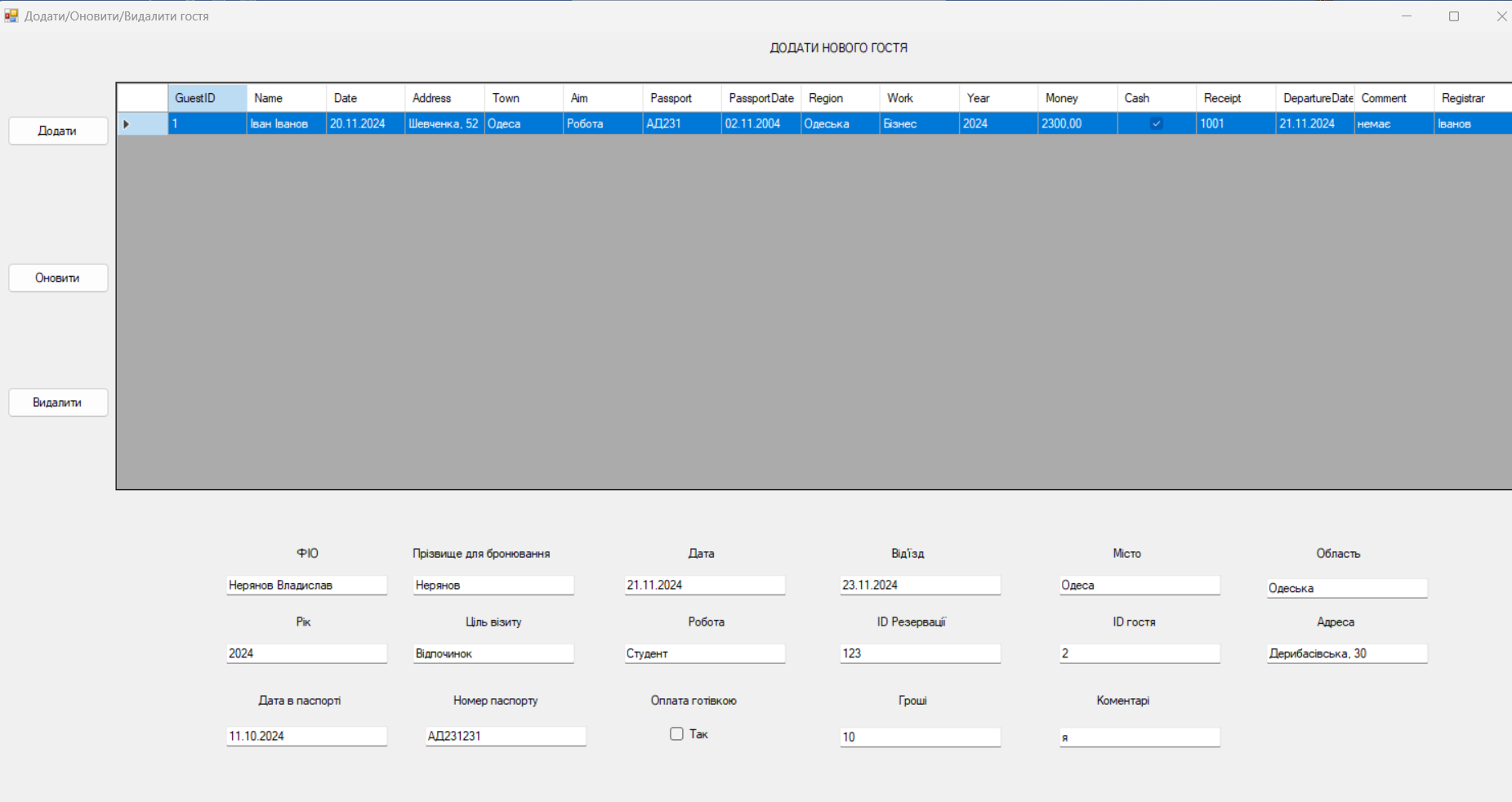


Рис 9.2 Форма до додавання

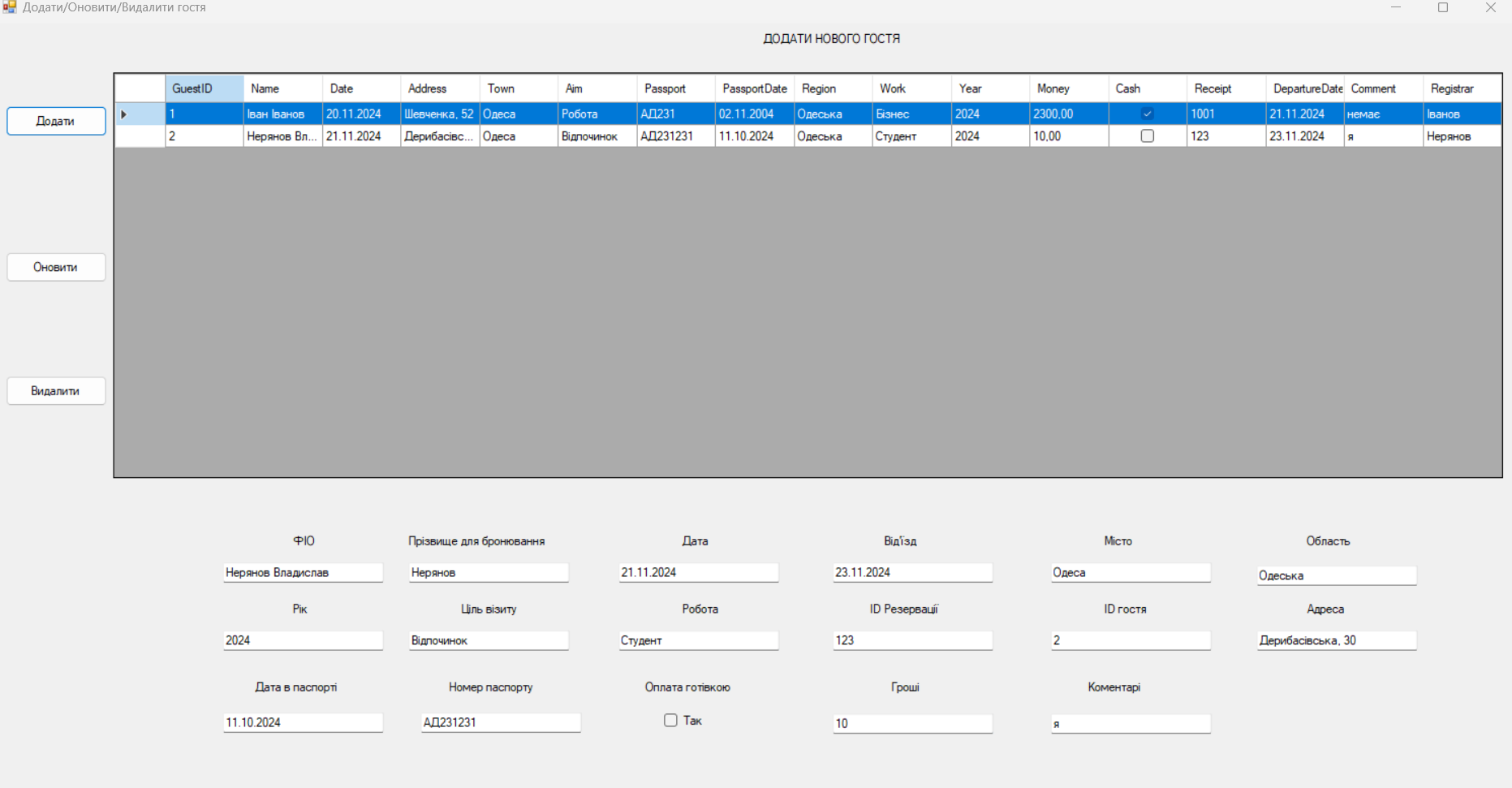


Рис 9.3 Форма після додавання

Основний принцип форми додавання гостя полягає в тому, щоб забезпечити зручний інтерфейс для введення та збереження інформації про нового гостя в базу даних. На формі є текстові поля для введення різних даних, таких як адреса, мета, коментарі, дата приїзду та від'їзду, паспортні дані, а також інші характеристики гостя. Окрім цього, є спеціальні елементи управління, такі як чекбокс для позначення, чи оплачується готель кешем.

Основні етапи роботи форми:

1. Заповнення текстових полів: Користувач вводить інформацію в поля форми, що стосуються гостя. Це дозволяє зберігати таку інформацію як адреса, коментарі, дані паспорта тощо.
2. Вибір оплати (чекбокс Cash): Для зручності, якщо оплата здійснюється кешем, користувач може відмітити відповідний чекбокс. Це важливо для того, щоб зберігати правильну інформацію про метод оплати.
3. Валідність введених даних: Перед додаванням нового гостя до бази даних здійснюється перевірка, чи не залишились порожні поля. Якщо є незаповнені обов'язкові поля, користувач отримає повідомлення про необхідність заповнити всі поля.
4. Додавання в базу даних: Після перевірки даних, кнопка "Додати гостя" викликає функцію, яка формує SQL-запит для вставки введених даних до таблиці Guests в базі даних. Для кожного поля формується параметр, який передається в SQL-запит, і дані записуються в таблицю.
5. Обробка помилок: Якщо під час додавання виникають помилки, наприклад, неправильний формат дати або порожнє поле, з'являється відповідне повідомлення про помилку.

Форма Додавання Резервації

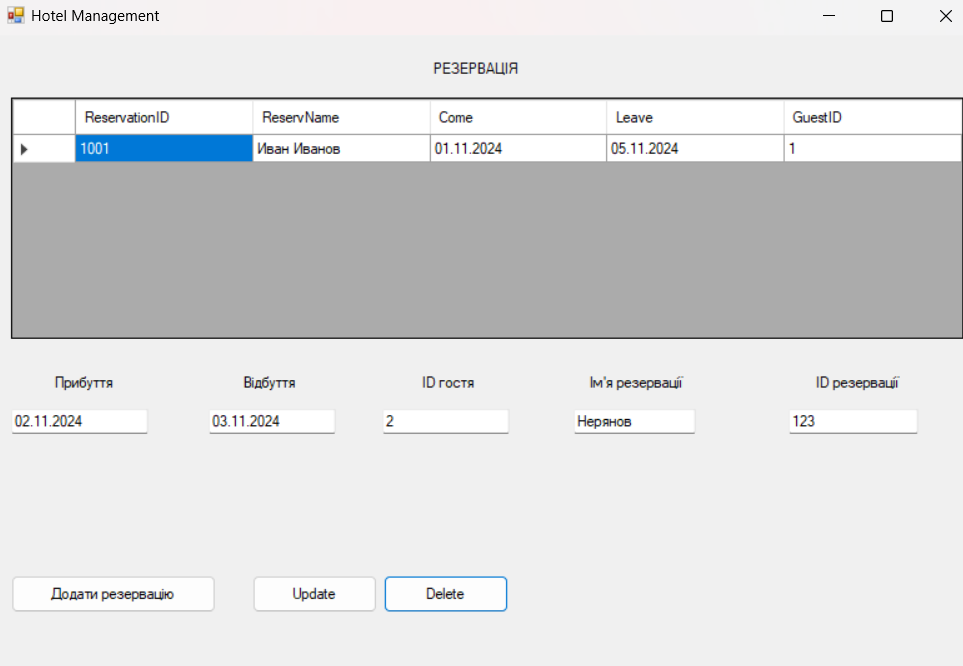


Рис 9.4 Форма до додавання резервації

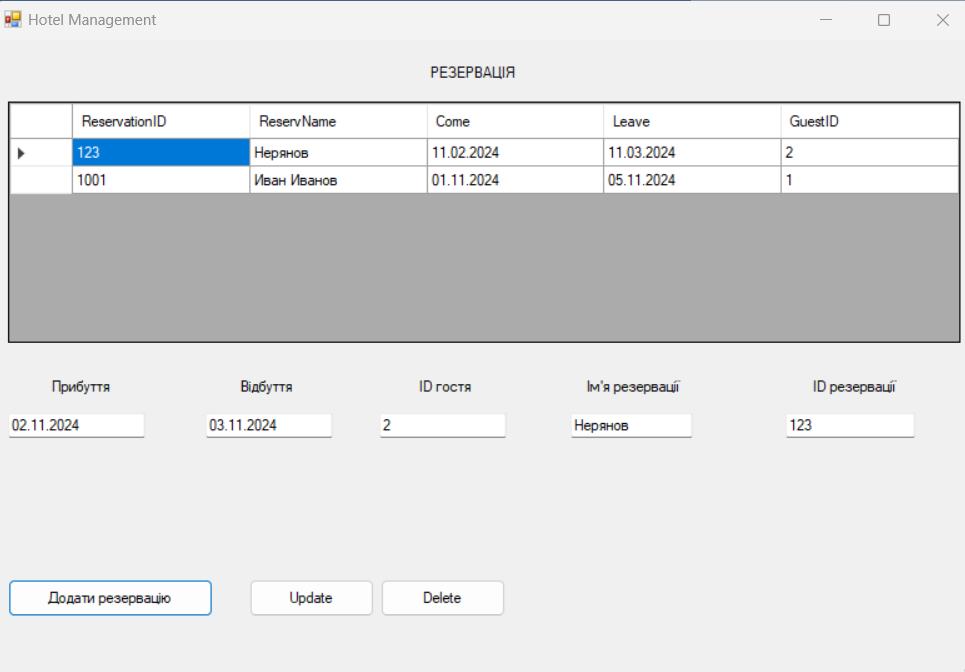


Рис 9.5 Форма після додавання резервації

Основний принцип форми створення резервації полягає в тому, щоб забезпечити ефективне і зручне введення та збереження даних про резервацію в базу даних. Це дозволяє зберігати всю необхідну інформацію про бронювання номерів для гостей, враховуючи дати прибуття та від'їзду, типи номерів, а також інші деталі, пов'язані з резервацією.

Основні етапи роботи форми:

1. Заповнення текстових полів: Користувач вводить інформацію в поля форми, що стосуються гостя.
2. Введення дат: Користувач вводить дату прибуття та дату від'їзду в спеціальні текстові поля або через вибір дати з календаря. Це дозволяє точно визначити період, на який зроблено бронювання, що є ключовим для управління доступністю номерів.
3. Введення додаткових даних: У формі є поля для введення додаткової інформації про бронювання, такі як коментарі чи спеціальні побажання (наприклад, побажання щодо розташування номера або спеціальних умов). Це допомагає персоналу готелю виконувати спеціальні вимоги гостей.
4. Перевірка валідності даних: Перед додаванням резервації в базу даних здійснюється перевірка на коректність введених даних. Якщо які-небудь обов'язкові поля порожні або мають неправильний формат (наприклад, некоректна дата), користувач отримує повідомлення про помилку.
5. Додавання в базу даних: Кнопка "Створити резервацію" викликає функцію, яка формує SQL-запит для вставки введених даних про резервацію в таблицю Reservations в базі даних. Всі введені дані передаються як параметри в SQL-запит, що забезпечує безпеку операції.
6. Обробка помилок: Якщо виникають помилки, наприклад, неправильний формат введених дат або незаповнені поля, з'являється відповідне повідомлення про помилку, щоб користувач міг виправити їх перед повторною спробою.