МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії

Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
Савченко А.С.

"____"_____20__ р.

ДИПЛОМНИЙ ПРОЕКТ
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СПУПЕНЯ «БАКАЛАВР»

Тема: **«Інформаційна система взаємодії з клієнтами для комерційного підприємства»**

Виконавець: Кушнір Юлія Олександрівна

Керівник: д.т.н. проф. Моржов В.І.

Консультант: Куклінський М.В.

Нормоконтролер: Шевченко О.П.

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп‘ютерної та програмної інженерії

Кафедра комп’ютерних інформаційних технологій

Освітній ступінь: Бакалавр

Галузь знань, спеціальність, спеціалізація: 12 “Інформаційні технології”, 122 “Комп’ютерні науки ”, “Інформаційні управляючі системи та технології”

ЗАТВЕРДЖУЮ
Завідувач кафедри
А.С. Савченко
“_____”_____ 2021 р.

ЗАВДАННЯ

на виконання дипломного проекту студента

Кушнір Юлія Олександрівна

1. Тема проекту: «Інформаційна система взаємодії з клієнтами для комерційного підприємства» затверджена наказом ректора № 636/ст. від 22.04.2021р.

2. Термін виконання проекту: з 11.05.2021 по 14.06.2021р.

3. Вихідні дані до проекту: розробка бази даних для підприємства з онлайн продажу електронних приладів.

4. Зміст пояснювальної записки (перелік питань, що підлягають розробці): вступ, аналітичний огляд і постановка завдання, розгляд завдання інформаційної системи, дослідження технологій та засобів, розробка бази даних, оцінка якості технології, висновки.

5. Перелік обов’язкового графічного матеріалу: загальний перелік існуючих систем та обробка інформації розробленою базою даних.

КАЛЕНДАРНИЙ ПЛАН

№	Етапи виконання дипломного проекту	Термін виконання	Відмітка про виконання
1.	Ознайомлення з постановкою задачі та вивчення літератури.	10.05.21 – 12.05.21	
2.	Розробка та затвердження плану дипломного проекту.	13.05.21	
3.	Проведення консультації з науковим керівником щодо створення першого розділу.	14.05.21- 18.05.21	
4.	Аналітичний огляд і постановка задачі. Написання 2 розділу.	19.05.21 – 22.05.21	
5.	Створення бази даних для підприємства з продажу електронних приладів.	23.05.21 – 27.05.21	
6.	Загальне редагування, написання висновків та оформлення пояснювальної записки.	28.05.21- 31.05.21	
7.	Проходження нормо-контролю, перепліт пояснювальної записки.	01.06.21 – 07.06.21	
8.	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації	07.06.21 – 09.06.21	
9.	Підготовка до захисту та попередній захист дипломного проекту на випусковій кафедрі дипломного проекту	10.06.21- 14.06.21	

Керівник дипломного проекту д.т.н. проф. Моржов В.І.

Студент: Кушнір Ю.О.

РЕФЕРАТ

Пояснювальна записка до дипломного проекту «Інформаційна система взаємодії з клієнтами для комерційного підприємства» містить 86 сторінок, 32 рисунка, 1 таблиця, 16 літературних джерел.

Об’єкт розробки – автоматизація бізнес-процесу комерційного підприємства з продажу електронних приладів.

Предмет розробки – система взаємодії з клієнтами підприємства.

Мета проекту – покращення операційних процесів підприємства шляхом розробки інформаційної системи.

Результати даного проекту можуть використовуватись для автоматизації роботи працівника підприємства з продажу електроприладів в момент оформлення замовлення, продажу товару, чи розрахунку вартості замовлення що міститься в магазині.

CRM-СИСТЕМА, ІНФОРМАЦІЙНА СИСТЕМА, РЕЛЯЦІЙНА БАЗА ДАНИХ, АВТОМАТИЗАЦІЯ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

Інформаційна система	–	система, призначена для зберігання, пошуку та обробки інформації, і відповідні організаційні ресурси, які забезпечують і поширюють інформацію
CRM-система	–	це бізнес-технологія, орієнтована на роботу з клієнтами, що об'єднує в собі безліч інструментів, що дозволяють значно підвищити ефективність роботи компанії
Бізнес-процес	–	сукупність взаємопов'язаних заходів або завдань, спрямованих на створення певного продукту або послуги для споживачів
БД	–	база даних
СУБД	–	система управління базами даних
ІС	–	інформаційна система
SQL	–	(Structured query language) мова структурованих запитів
MS SQL	–	система управління базами даних, яка розроблюється корпорацією Microsoft
DDM	–	набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення
DDL	–	мова визначення даних
DML	–	мова керування даними

ВСТУП

У сучасному світі дуже швидко розвивається ринок IT-технологій та їх застосування в підприємствах та бізнесі. Щороку створюють нові засоби для більш ефективного ведення аналітики та забезпечення програмним продуктом різні комерційні підприємства. Саме тому постає запитання, як саме використовувати підприємствам новітні технології і як краще впровадити нові технології до вже існуючої бази та системи взаємодії з клієнтами.

У наш час дуже багато підприємств працюють з застарілими технологіями ведення даних, які потребують великої кількості часу для обробки запитів та пошуку інформації. Та все більше прагнуть перейти до сучасних засобів ведення баз даних, але серед великого різноманіття не можуть визначитись, яка технологія буде більш застосована саме для їхнього підприємства і задовільнить усі потреби використання.

Актуальність проекту полягає у вирішенні питання , яка ж технологія може підходити до різних комерційних підприємств а також у створенні бази даних для взаємодії між підприємством та клієнтами, що є однією з найнеобхідніших складових ефективної та швидкої взаємодії з клієнтами. Система швидкого пошуку інформації за певними критеріями, збереження, редагування та обробки інформації збільшить ефективність роботи підприємства та знизить навантаження на робітників підприємства.

Метою роботи є розробка нової структури бази даних для комерційного підприємства за для ефективності використання ресурсів.

Для досягнення поставленої мети потрібно вирішити наступні задачі:

- Розглянути засоби ведення клієнтських баз даних;
- Розібрати переваги і недоліки у використанні у різних комерційних підприємствах;
- Розглянути етапи проектування нової бази даних виходячи з потреб підприємства;

- Визначити основні вимоги до інформаційної системи та обрати засоби реалізації;
- Розробити та реалізувати нову інформаційну систему для взаємодії з клієнтами.

Результат дипломного проекту стане система інформаційної взаємодії для комерційного підприємства. Її практичне значення полягає у можливості використанні в управлінні взаємовідносинами з клієнтами. Також цю систему та її окремі модулі можна буде застосовувати та впроваджувати у різні види діяльності та підприємства.

РОЗДІЛ 1
ОСНОВНІ ПОНЯТТЯ ІНФОРМАЦІЙНИХ СИСТЕМ

У сучасному світі неможливо створити успішний бізнес без ефективного управління. Різні підприємства , корпорації та банки є дуже складними системами, які прагнуть вдосконалення і ефективності роботи. Більшість даних, що фіксуються інформаційними системами, стосуються діяльності самої організації, яка служить для отримання внутрішньої інформації. Але на ринку, що стає все більш конкурентоспроможним, фірма повинна отримувати доступ до все більшої кількості зовнішньої інформації. Тому важливо зазначити, що особи, що приймають рішення, потребують як внутрішньої інформації про свою організацію, так і зовнішньої інформації про її оточення.

В організаціях використовується декілька типів інформаційних систем. До них належать:

- Системи обробки транзакцій (TPS);
- Системи управлінської звітності (MRS);
- Системи для прийняття рішень (DSS);
- Виконавчі інформаційні системи (ESS);
- Офісні інформаційні системи (OIS);
- Системи професійної підтримки.

Корпоративні інформаційні системи призначені для автоматизації функціонування та управління фірмою, для підтримки обліку на підприємстві та представляє інформацію для швидкого прийняття рішень.

Головними особливостями корпоративних інформаційних систем є:

- Зв’язаність охоплення функцій управління;
- Послідовність ділових процесів;
- Велика кількість операцій;
- Ефективність споживання і використання програмного забезпечення;
- Можливості розділу системи на частини та їх локальної установки;
- Адаптивність функціоналу та інструментарію системи до особливостей керованого об’єкту;
- Можливість вдосконалення системи після її застосування.

Наведемо приклад структури інформаційної системи корпорації:

Головною місією усіх інформаційних систем є забезпечення організацій необхідною інформацією у найефективніший спосіб, створення такої оболонки, в якій всі елементи будуть функціонувати задля виконання основної мети - управління підприємством.

1.1. Необхідність використання інформаційних систем

Інформаційні системи використовуються для допомоги керівництву шляхом надання зворотного зв'язку про результати діяльності фірми. Зворотній зв'язок відноситься до виходів системи, які перетворюються назад на входи для управління роботою системи. Інформаційні системи використовуються для порівняння даних про фактичні показники роботи зі стандартами, розробленими раніше. На основі інформації про розбіжності менеджери можуть сформулювати коригувальні дії, які потім повертаються в діяльність фірми. . Також не менш важливим є системи обробки інформації. Від них залежить ефективність і швидкість роботи підприємства. Інформаційні системи, швидко обробляючи дані ,перетворюють їх у цільну інформацію яку в подальшому використовують для виконання поставлених завдань. Інформаційні системи використовують дані, що зберігаються в комп'ютерних базах даних, для надання необхідної інформації. База даних - це організований збір взаємопов’язаних даних, що відображають основний аспект діяльності фірми.

Залежно від різних факторів, таких як: галузь в якій функціонує підприємство, ієрархію керівництва, цілі функціонування організації, різний рівень доходу та розміри організації, і розробляють різні інформаційні системи. В залежності від цих всіх факторів необхідно персоналізована розробка структури інформаційної системи яка зможе задовільнити усі потреби організації і збільшити швидкість розвитку бізнес процесів в цілому. Саме тому так необхідно слідкувати за тенденціями розвитку технологій і впроваджувати новинки у свою підприємницьку діяльність.

Усі спеціалісти з розробки інформаційних систем запевняють , що для ефективного функціонування підприємств не є достатнім лише трудових, матеріальних та фінансових ресурсів. Так як наші реалії диктують нам нові правила і головним ресурсом наразі є інформація, бо саме вона вказує нам, що робити з усіма іншими ресурсами які використовує певне підприємство. В порівнянні з минулим, обсяги інформації зараз стали настільки об’ємними, що за допомогою лише людського ресурсу компанії не зможуть ефективно обробляти інформацію і

переробляти для управління. Тому в наслідок цього і була створена інформаційна індустрія, яка збагачується новинками щорічно.

Створення і ведення клієнтської бази. В CRM-системах створюється обліковий запис або картка з контактною інформацією для конкретного замовника, в якій відображаються всі взаємодії з контрагентом. Завдяки цій функції CRM-система вирішує цілий ряд важливих завдань:

Збереження контактів і історії спілкування з клієнтами при звільненні працівника. Його наступник отримає всі необхідні дані про контрагентів і зможе без зволікань приступити до роботи.

Можливість розвивати відносини з потенційними замовниками і нагадувати про себе діючим клієнтам. За допомогою CRM-системи ваші менеджери зможуть розсилати інформацію про нові продукти або послуги, а також привітання з датами, важливими для компанії замовника.

Організація ефективної взаємодії між підрозділами вашої компанії. Спільна робота з єдиної клієнтською базою клієнтів дає можливість вашим співробітникам з різних відділів підходити до завдання комплексно в рамках єдиної маркетингової стратегії. Наприклад, поєднуючи персоналізовані пропозиції щодо товарів і послуг при пошуку нових замовників з PR-акціями або event-заходами.

Підвищення ефективності продажів. Вирішення цього завдання в CRM-системах здійснюється за допомогою додавання в загальну базу даних відомостей про потенційних угодах і супроводу їх на всіх етапах воронки продажів. У картці клієнта відбивається повна інформація про всі його взаємодіях з компанією, починаючи з «холодного» дзвінка або входить заявки. Систематизація роботи менеджера в процесі ведення клієнта по всіх етапах воронки продажів збільшує ймовірність успішної операції.

Підвищення якості обслуговування клієнтів CRM-система надає великі можливості для задоволення потреб замовника. Наприклад, на основі даних, зібраних в CRM по кожному клієнту, можна сформувати для нього найцікавіша пропозиція, не пропустити зручний для клієнта час доставки, а також отримати зворотній зв'язок у вигляді подяк або претензій. Завдяки інформації, що зберігається

в CRM-системі, ваші менеджери не забудуть привітати представників замовника з днем народження або ювілеєм компанії, запросити на корпоративний вечір, що також працює на підвищення лояльності до вашої організації.

Аналітика до даної функції відноситься все, що пов'язано з отриманням і обробкою інформації про ефективність роботи з контрагентами та комерційної діяльності компанії в цілому. CRM-система дає можливість проводити аналіз за різними напрямками і тимчасових проміжків. Керівник отримує відразу кілька зведених звітів: статистику по продажах і операціях, перелік незавершених проектів, інформацію про виконання завдань співробітниками, відгуки від клієнтів. Отримані дані наочно покажуть загальний стан справ в компанії за цікавий для період часу, ефективність роботи кожного співробітника, якість сервісу. На основі цих відомостей керівник зможе спрогнозувати рівень продажів і побачити точки зростання для комерційної діяльності організації. Зведені звіти також будуть корисні при розробці довгострокової стратегії розвитку вашої компанії.

Розмежування прав. Щоб перераховані вище функції могли найбільш ефективно працювати, в CRM-системі підтримується розмежування прав. Керівник зможе особисто давати доступ до ділових операцій певним фахівцям, стежити за реалізацією проектів і контролювати виконання завдань. CRM-система дозволить визначити і зафіксувати зони відповідальності, обов'язки як кожного члена команди, так і цілих підрозділів.

Таким чином актуальність даної теми полягає в тому, що будь-якій компанії, що розвивається, необхідно збільшувати свою конкурентоспроможність, щоб забезпечити розвиток підприємства найбільш ефективним способом і використовують системи керування даними, щоб якомога простіше і швидше надавати клієнтам найактуальнішу інформацію. Інформаційна система яка буде швидко задовольняти потреби ,як керівництва, так і клієнтів ,допоможе підприємству нарощувати темпи розвитку компанії і з цього отримувати набагато більші прибутки.

1.2. Класифікація CRM-систем

На постійно зростаючому конкурентному ринку для бізнесу дуже важливо ділитися правильною інформацією з потрібною людиною в потрібний час, інакше бізнес втратить можливості продавати продукцію чи послуги. Програмне забезпечення управління взаємовідносинами з клієнтами - єдине рішення, яке може допомогти бізнесу правильно спілкуватися з потенційними клієнтами чи клієнтами. Для будь-якої програми CRM основною метою є надання організації змоги зрозуміти потреби та поведінку клієнтів та забезпечити кращу якість обслуговування. Це допомагає утримувати існуючих клієнтів та використовувати нові можливості, будуючи міцні відносини між організацією та клієнтами. CRM може аналізувати дані та створювати звіти, коли це потрібно. В основному існує три типи CRM-додатків - операційний, аналітичний та спільний для виконання всіх цих видів діяльності.

1. Операційна CRM

Операційна CRM впорядковує бізнес-процес, що включає автоматизацію продажів, автоматизацію маркетингу та автоматизацію послуг. Основна мета CRM цього типу - генерувати потенційних клієнтів, перетворювати їх у контакти, фіксувати всі необхідні деталі та надавати послуги протягом усього життєвого циклу клієнта.

Функції автоматизації продажів допомагають організації автоматизувати процес продажів. Основна мета автоматизації продажів - встановити стандарти в організації для залучення нових клієнтів та взаємодії з існуючими клієнтами. Він організовує інформацію таким чином, що бізнес може задовольнити потреби клієнтів та збільшити обсяг продажів більш ефективно та ефективно. Він включає різні модулі продажів CRM, такі як управління потенційними клієнтами, управління контактами, управління замовленням, прогнозування продажів.

Основна мета автоматизації маркетингу - з’ясувати найкращий спосіб запропонувати продукцію та звернутися до потенційних клієнтів. Основним

модулем автоматизації маркетингу є управління кампаніями. Це дозволяє бізнесу вирішувати ефективні канали (наприклад, електронні листи, телефонні дзвінки, очні зустрічі, оголошення в соціальних мережах), щоб охопити потенційних клієнтів.

2. Аналітична CRM

Аналітичний CRM допомагає вищому керівництву, маркетингу, продажам та допоміжному персоналу визначити кращий спосіб обслуговування клієнтів. Аналіз даних є основною функцією цього типу CRM-додатків. Він аналізує дані клієнтів, що надходять з різних точок дотику, щоб отримати кращу інформацію про поточний стан організації. Це допомагає вищому керівництву приймати кращі рішення, керівникам відділів маркетингу зрозуміти ефективність кампанії, керівникам відділів продажів збільшити продажі та допоміжному персоналу для покращення якості підтримки та побудови міцних відносин із клієнтами.

Особливості аналітичного CRM:

- Збирайте інформацію про клієнта, яка надходить з різних каналів, та структуруйте аналіз даних;
- Допоможіть організації встановити методологію ведення бізнесу у сферах продажів, маркетингу та підтримки для поліпшення відносин із клієнтами та лояльності;
- Підвищити ефективність системи управління персоналом та проаналізувати ключові показники ефективності, встановлені бізнесом.

3. Спільна CRM

Спільна CRM, яку іноді називають стратегічною CRM, дозволяє організації ділитися інформацією про клієнтів між різними бізнес-підрозділами, такими як команда з продажу, команда з маркетингу, команда технічної підтримки та підтримки. Наприклад, зворотній зв'язок з командою підтримки може бути корисним для маркетингової команди для звернення до цільових споживачів за допомогою конкретних продуктів чи послуг. У реальному світі кожна бізнес-одиниця працює як незалежна група і досить не часто ділиться даними клієнтів з іншими командами, що часто спричиняє бізнес-збитки. Спільна CRM допомагає об’єднати всі групи для досягнення лише однієї мети - використовувати всю

інформацію для покращення якості обслуговування клієнтів, щоб завоювати лояльність та придбати нових клієнтів для збільшення продажів.

Різні типи CRM-програм мають різні особливості та переваги. Отже, перш ніж впроваджувати CRM-систему, для бізнесу дуже важливо визначитися з майбутньою метою та стратегією. Якщо ви хочете вибрати найкращий CRM для свого бізнесу, прочитайте нашу статтю на тему „Як вибрати найкраще CRM-програмне забезпечення для вашого бізнесу“.

1.3. Застосування бази даних в інформаційних системах

У 1960-1970-х роках комерційні системи управління базами даних були зосереджені на середовищах транзакцій та пакетної обробки з акцентом на мінімізації обчислювальних ресурсів. Реляційні системи управління базами даних з'явилися на комерційній арені в 1980-х рр. І розширив коло застосувань, що піддаються обробці баз даних. Зараз ці програми включають інтерактивне середовище, а також транзакції та пакетні операції. Реляційні СУБД наголошує на функціональності, простоті доступу користувачів до даних та покращеній економіці розробки додатків за рахунок обчислювальних ресурсів. Використання комп'ютерної бази даних, як правило, найголовніше у ефективному управлінні даними. Спільна, інтегрована комп'ютерна структура, база даних зберігає наступне:

- Дані кінцевого користувача, тобто необроблені дані, що стосуються кінцевого користувача;
- Метадані - дані про дані, за допомогою яких інтегруються та управляються даними кінцевих користувачів.

Бази даних відіграють вирішальну роль як у створенні даних, так і в управлінні ними. Без системи управління базами даних неможливо ефективно керувати даними та керувати ними. Служачи посередником між користувачем та базою даних, СУБД надає користувачам доступ до файлів, що зберігаються в базі даних. Він надає кінцевому користувачеві єдиний інтегрований вигляд даних і перетворює всі отримані додатки на складні операції, що виконують ці запити.

Однак значна частина внутрішньої складності бази даних прихована від користувачів та прикладних програм.

Від надання можливості спільного використання даних у базі даних між багатьма програмами або користувачами до надання користувачам єдиного всеохоплюючого сховища даних, СУБД відіграє важливу роль в інформаційних системах. Ось деякі речі, які СУБД дозволяє в інформаційних системах:

- Покращений доступ до даних у межах компанії;
- Посилені взаємозв'язки між даними;
- Покращена безпека даних.

За допомогою СУБД робітники компанії можуть отримувати доступ, оновлювати та видаляти дані в базі даних або інформаційній системі. Ця інформація є легко доступною для користувачів, коли інформаційні системи компанії інтегровані з реляційною СУБД.

1.3.1. Захист інформації та безпека бази даних

У сучасному світі існує дуже багато загроз пов’язаних з цілісністю даних. Все більше компаній прагнуть вдосконалити свої сховища даних за для безпеки комнаї та своїх клієнтів. Зосередження уваги на захисті конфіденційних або критичних даних, таких як інтелектуальна власність та особисті дані, є результатом зростаючих кібер-ризиків та дедалі жорсткіших правил безпеки даних.

Чим більше людей отримує доступ до даних, тим більший ризик порушень безпеки даних. Як правило, компанії вкладають значний час, зусилля та гроші, щоб забезпечити належне використання своїх даних. Але це не завжди дає бажані результати. За допомогою СУБД організації можуть забезпечити кращий контроль за політикою конфіденційності даних та безпеки, що дозволяє їм покращити загальну безпеку даних.

Сьогодні захист конфіденційної інформації вимагає набагато більше, ніж впровадження базових технологій безпеки, таких як антивірусне рішення та брандмауер. На щастя, звіт Netwrix IT-ризиків за 2018 рік показує, що компанії

готові виділяти більше коштів на кібербезпеку: інвестиції в безпеку зросли на 128% за останні 3 роки і, як очікується, зростуть ще на 146% протягом наступних 5 років.

Необхідність пріоритетності інформаційної безпеки походить від ризиків, з якими стикаються підприємства. Фінансові втрати, юридичні проблеми, пошкодження репутації та зриви діяльності є одними з найбільш руйнівних наслідків порушення даних для підприємства.

Ми можемо розділити ризики безпеки даних на дві основні категорії. Ризики, пов'язані з відсутністю видимості - Основою безпеки даних є глибоке розуміння даних, що зберігаються. Компанії часто мають терабайт даних, і ризики їх порушення зростають, коли компанії не знають, де зберігаються критичні та регульовані дані в їхній інфраструктурі - на робочих столах, серверах та мобільних пристроях або в хмарі. Звіт Netwrix виявив, що 44% компаній не знають або не впевнені в тому, як їх працівники мають справу з конфіденційними даними через відсутність видимості їхньої інфраструктури. Це величезний ризик, оскільки це робить виявлення зловживань привілеями або несанкціонованого доступу користувачів до конфіденційних даних майже неможливим, поки не завдасть реальної шкоди.

Ризики, пов’язані з діями людини - Хоча дані можуть бути втрачені або пошкоджені внаслідок стихійних лих, найбільшу загрозу становлять люди, які можуть робити критичні помилки або навмисно спричиняти проблеми багатьом різним. Протягом багатьох років компанії, як правило, довіряють своїм внутрішнім користувачам і зосереджуються на захищаючись від тих, хто отримує доступ до мережі ззовні, і це мислення продовжує панувати. Як показує дослідження Netwrix, більшість компаній продовжують вважати хакерські атаки найнебезпечнішою загрозою, тоді як факти свідчать, що насправді саме інсайдери спричиняють переважну більшість випадків безпеки.

Основними учасниками загроз є:

- Хакери, які можуть встановлювати шкідливе програмне забезпечення, коли користувачі неправильно обробляють фішинг-електронні листи;

- Треті сторони, відсутність достатньої мережевої безпеки може залишити взаємопов’язані системи відкритими для атак, або які можуть скористатися надмірними дозволами та переекспонованими даними;
- Інсайдери зі зловмисними намірами, які можуть вкрасти дані з метою створення конкуруючого бізнесу, продажу інформації на чорному ринку, помсти роботодавцю тощо;
- Співробітники без зловмисного наміру, які можуть скопіювати файли на свої персональні пристрої, щоб використовувати їх для проекту, навіть не підозрюючи, що вони роблять щось незаконне та небезпечне. Вони також можуть випадково прикріпити файл із конфіденційними даними до електронного листа або надіслати його неправильному одержувачу.

Наступні рішення безпеки можуть бути корисними для мінімізації ризиків безпеки даних.

Виявлення та класифікація даних - Технологія виявлення даних сканує сховища даних та звіти про висновки, щоб уникнути зберігання конфіденційних даних у незахищеному місці. Це корисно для зменшення ризику неправильної експозиції даних. Класифікація даних - це процес маркування конфіденційних даних тегами, щоб ви могли захистити корпоративні дані відповідно до їх значення для організації.

Шифрування даних - кодування важливої інформації, щоб зробити її нечитабельною та марною для зловмисних акторів, є важливою технікою комп'ютерної безпеки. Програмне шифрування даних виконується програмним рішенням для захисту цифрових даних перед їх записом на SSD. В апаратному шифруванні окремий процесор присвячений шифруванню та дешифруванню з метою захисту конфіденційних даних на портативному пристрої, такому як ноутбук або USB-накопичувач.

Динамічне маскування даних (DDM) - Ця технологія підтримує маскування даних у режимі реального часу, щоб обмежити вплив конфіденційних даних на непривілейованих користувачів, не змінюючи вихідні дані. Інтерес до DDM особливо великий у проектах великих даних.

Аналітика поведінки користувачів та організацій - це складна технологія для виявлення відхилень від звичайної діяльності та підозрілих або критичних змін до того, як вони вплинуть на безпеку або безперервність бізнесу. Програмне забезпечення для захисту даних цього типу допомагає виявляти різні типи інсайдерських загроз, поганих акторів та хакерів, а також розширені загрози, що включають шкідливе програмне забезпечення та програми-вимагателі.

1.3.2. Переваги розробки нової структури бази даних для комерційного підприємства

Світ кожного дня розвивається і нові більш ефективні технології розробки приходять на заміну старим і неактуальним. Потреба підприємств у розробці нових методів проектування інформаційних систем і баз даних обумовлена багатьма обставинами:

- Швидкий темп зростання і розвитку інформаційних технологій;
- Збільшенням споживчого попиту на програмну продукцію;
- Збільшенням об’ємів інформації, яку необхідно обробляти на підприємствах, необхідність структурування інформації та інтегрування в єдину систему керування;
- Діджіталізація усіх рівнів сучасної економіки, включаючи малі і великі комерційні підприємства;
- Державною політикою у сфері IT-технологій та стандартів інформаційної безпеки.

Тож розробка структурно нових баз даних обумовлена зростаючим попитом на інформаційне та структурне забезпечення, задля швидкого прийняття рішень. Все більше малих бізнесів переходять із простих баз у вигляді Excel документів, до більш розвинених систем упорядкування даних. Зараз не можна уявити успішну компанію без упорядкованої системи даних, тому що створеня правильної, структурованої системи обробки інформації - це запорука ефективності роботи підприємства.

Перевагами переходу з простих сховищ даних до сучасних є:

- Більший рівень безпеки облікової системи;
- Зросте продуктивність інформаційної системи;
- Збільшиться швидкодія виконання поставлених задач;
- Можливість шифрування даних від зовнішніх факторів;
- Зручний сучасний інтерфейс, буде легким у використанні співробітниками підприємства;
- Збільшення обсягу збережених і доступних для обробки даних. Обсяг даних не обмежується пам'яттю. Такі завантаження бази даних загалом прибираються включенням нових аналітичних можливостей;
- Зменшення обсягів даних, які пересилаються. Досягається завдяки тому, що частина даних зберігається і обробляється локально.

Недоліки незначні, але можуть виявитись проблемою для деяких компаній:

- Необхідність підготовки та підвищення кваліфікації співробітників;
- Постійна підтримка баз даних;
- Вартість розробки структури нової бази даних;
- Нестача досвіду. Ще не накопичено достатньо досвіду промислової експлуатації систем керування даними;
- Підвищені вимоги до умов зберігання даних.

1.3.3. Автоматизація роботи підприємств за допомогою бази даних

Автоматизація баз даних має безліч практичних застосувань, корисних для підприємств з різних галузей. Ось деякі загальні переваги автоматизації баз даних SQL:

- Ефективне рішення ,як і найпоширеніші переваги та цілі впровадження будь-якого виду автоматизації, автоматизація баз даних пропонує вам ефективне рішення для створення запланованих запитів SQL тощо;

- Автоматизація баз даних допомагає вашому бізнесу досягти цілодобових термінів, що стало дуже важливим фактором у нинішньому конкурентному бізнесі. Немає шансів на помилки вручну в загальному виконанні запитів. Не лише це, автоматизація баз даних також допомагає вам економити значну частину вашої оплати праці, а отже, також враховує економічну ефективність;
- Паралельне виконання допомагає при роботі зі складними базами даних, тому що бувають випадки, коли для отримання певної категорії даних доводиться писати складні запити. Паралельне виконання робить виконання таких запитів простішим, швидшим та всебічним;
- Автоматизація баз даних дозволяє створювати мініатюрні оператори SQL і забезпечує планування їх виконання відповідно;
- Легке та ефективне в реалізації виконання не обмежується лише паралельним виконанням. Автоматизація баз даних не є ракетною наукою і може легко отримати вигоду від підприємств усіх видів. Агенти SQL Server дозволяють створювати незалежних адміністраторів баз даних і можуть планувати роботу (цілодобово), коли вас немає;
- Покращене бізнес фокусування. Незалежно від того, яку автоматизацію ви плануєте для свого бізнесу, основна увага вашої команди повинна залишатися на послугах або продуктах, які ви пропонуєте;
- Машини надійніші за людей. Після планування автоматизація баз даних буде продовжувати працювати і отримувати результати, без помилок, до втручання. Це дозволяє керівництву вашого бізнесу більше зосередитись на вдосконаленні послуг чи продуктів, які ви пропонуєте.

Є поширеною необхідність в автоматизованій обробці даних, тому виникла необхідність спеціалізованих мовах обробки даних. Пакети СУБД дають можливість управляти даними напряму в інтерактивному режимі, а також дають змогу розробникам реалізовувати більш довершені програмні рішення для їх обробки. Обробляти дані можна по різному, так само як і маніпулювати ними, залежно від середовища керування даними.

Швидкодія доступу до потрібної інформації та результативність її отримання в фінальному результаті визначає вдале ведення бізнесу та скорочують витрати на певні частини заробітку. Розроблені методи, що спрощують використання з великим обсягом даних: створення баз даних, виконання пошуків, редагування, а також аналіз даних та проведення розрахунків, що включають фінальні документи та дані у вигляді таблиць, діаграм та графіки. Діаграми майже в усіх середовищах можна створити автоматично тому не виникає проблем в пошуку додаткових сервісів моделювання діаграм.

Наявність бази даних клієнтів дає змогу: ідентифікувати клієнтів за рівнем доходу, укласти більшу кількість угод, краще пізнати потенційних клієнтів і перетворити їх на реальних покупців, укласти список попередніх клієнтів, які надалі залишаються майбутніми споживачами, виявити послуги, які приносять найбільший дохід, вести відповідну цінову політику, порівнювати витрати і доходи, оцінювати показники обслуговування клієнтів, визначати ефективність управлінських рішень.

Зі сказаного випливає, що підвищення ефективності обліку роботи з клієнтами для підприємств, шляхом розробки CRM-систем є актуальною задачею. Щоб система управління даними автоматизувала роботу на підприємстві вона має відповідати певним критеріям:

- Дотримання бізнес-процесів організації. Приймаючи рішення про впровадження CRM-системи, спочатку необхідно точно визначитись;
- Простота використання. Інтерфейс користувача повинен бути ергономічним, тобто якомога легшим та комфортнішим для роботи;
- CRM-система ускладнює процес взаємодії із замовниками та замовниками
- збільшується кількість дій, які повинні виконувати працівники;
- Наявність засобів аналізу. Вивчіть поведінку споживачів, їх очікування та вимоги, а також аналіз, CRM-система зобов'язана запропонувати можливості реалізації потреб для кожного конкретного замовника;
- Здатність адаптуватися до процесів. Важливою особливістю CRM-системи є можливість гнучко змінювати налаштування за необхідності. Така

можливість дозволити більш повно визначити та автоматизувати кожен конкретний процес;

- Масштабованість. Цей критерій особливо важливий для великих організацій. Необхідно, щоб рішення, що використовуються в CRM-системі, були масштабовані і застосовні до великої кількості користувачів;
- Інтеграція з іншими інформаційними системами та можливість роботи онлайн, можливість інтегрувати та обмінюватися даними між системою CRM та іншими системами автоматизації.

1.4. Висновок до розділу.

Було розглянуто переваги використання інформаційних систем на сучасних підприємствах, та переваги які надають сучасні інформаційні системи. Розглянуто класифікації різних CRM-систем, та області в яких ту чи іншу CRM-систему слід застосовувати, виходячи з потреб комерційного підприємства. Також було досліджено використання баз даних , як складову інформаційної системи. У ході аналізу було визначено головні переваги і недоліки використання баз даних для маленьких, середніх та великих підприємств. Серед переваг визначені такі фактори як: безпека даних, швидкість виконання запитів, зменшення людського ресурсу в веденні конфіденційної інформації, різні рівні доступу унікальні для кожного працівника та інші. Також були дослідженні методи автоматизації роботи підприємства та обробки даних підприємницької діяльності. Також проаналізувавши бізнес процеси комерційних підприємств було визначено, що основною проблемою в сучасних підприємствах є низький ступінь взаємозв’язків з клієнтами та слабка автоматизація функціонального менеджменту. Для вирішення даних проблем у роботі пропонується розробити інформаційну систему для підприємства з онлайн продажу електротехніки.

РОЗДІЛ 2
ТЕХНОЛОГІЇ ТА ЗАСОБИ РОЗРОБКИ БАЗИ ДАНИХ

2.1. Засоби ведення клієнтських баз даних

База даних - це сукупність пов'язаних файлів. Бази даних можуть існувати на папері, наприклад, телефонний довідник, але вони неефективні та дорогі в обслуговуванні. Комп’ютерна база даних пропонує перевагу потужних засобів пошуку, які можна використовувати для пошуку та отримання інформації в рази швидше, ніж ручними методами.

В наш час створено багато систем керування базами даних. Частина з них базуються на основі єдиної концепції понять та значень. Але все-таки в кожній системі керування є свої особливості використання, тож розглянемо найпоширеніші в підприємницькій діяльності засоби ведення баз даних.

MySql – є найпоширенішою системою керування базами даних для розробки веб додатків. СУБД MySql є реляційною базою даних, яка розроблюється і підтримується компанією Oracle. Система має простий для використання інтерфейс і здатна швидко оброблювати великі масиви даних. Ця СУБД дає змогу обрати різноманітні движки для системи зберігання, які дозволяють змінювати функціонал інструменту і обробляти дані, що зберігаються в різних типах таблиць. Еластичність СУБД MySQL полягає у підтримці великої кількості типів таблиць.

За допомогою доступної усім архітектурі, в MySQL повсякчас з'являються нові види таблиць. Мінусом для комерційно великих підприємств може стати те , що лише платні версії сповна вдовольняють усі потреби в розробці структурно важкої бази даних, а безкоштовний пакет MySql буде мати недостатньо інструментів розробки.

MS SQL Server –також відноситься до реляційних баз даних. Перевагою даної системи керування є те що вона входить в пакет Microsoft Office і тісно зв'язана з усіма компонентами Microsoft, що робить її дуже зручною у використанні. Створена ця платформа спеціально для вирішення комерційних задач. Ця СУБД, ядро якої відпрацьовує на хмарі, а також локальних серверах, саме тому це дозволяє суміщати типи використаних серверів разом. Також перевагою є те що одразу після випуску першої версії Microsoft SQL сервер 2016, Microsoft пристосувала продукт для операційної системи Linux. Багатогранний інтерфейс і безліч можливостей реалізації структурно непростих рішень, робить цю платформу як ніколи актуальною і потрібною для ведення бізнес проєктів. Також компанія додала можливість візуалізації бази даних навіть на мобільних пристроях. Ця система ідеально підходить для великих підприємств і бізнесів, які тісно пов'язані з продуктами компанії Microsoft.

PostgreSQL – одна з перших систем керування об'єктно-реляційними базами даних, яка була створена для використання у вільному доступі до програмного забезпечення системи. Саме через це вона часто використовується невеликими компаніями, так як є повністю безкоштовною і ліцензованою. Часто використовується для розробки веб-проектів. Одним з плюсів є те що ядро бази даних може буди розміщено у віртуальних, хмарних і фізичних середовищах. Ідеально підходить для організацій з обмеженим бюджетом, але кваліфікованими фахівцями, коли потрібно можливість вибрати свій інтерфейс.

Microsoft Access – продукт компанії Microsoft, який тісно пов'язаний із Microsoft SQL Server, який ми розглянули вище. Різниця лише в тому що середовище розробки Access є менш гнучким, але більш простим, бо містить дуже

багато шаблонів які можна використати для створення бази даних. Систему управління Acces, якою б простою вона не здавалася, дає змогу виконувати дуже важкі завдання та функції. Перевагою є швидкодія розробки бази даних на цій платформі. Плюсом є те ,що це дешевше і швидше для компанії замовника, але мінус в тому, що універсальні шаблони не завжди можуть задовільнити усі потреби бізнес процесів.

Microsoft 1C – сама по собі не є сучасною базою даних, але і телефонна книга номерів також може нею називатись, просто вона буде проста і не матиме нічого спільного з сучасними технологіями. Тож 1С це програма яка при необхідності може працювати з сучасними базами даних по типу MS SQL Server, але в більшості компаній які працюють на базі 1С не підключені сучасні системи керування даними. З даними які зберігаються всередині 1С дуже важкого працювати і займає багато часу, частіше всього дану програму використовують для бухгалтерського обліку на підприємстві. Тому зараз більшість компаній хочуть переходити на сучасні системи управління даними.

Таким чином, на сьогоднішній день розробник не зв’язаний межами якого-небудь конкретного пакету, а в залежності від поставленої задачі може використовувати самі різні програми. Тому, важливішим є основний вектор прогресування СУБД та інших засобів розробки додатків в даний час.

Тож при виборі системи керування власники компаній звертають свою увагу на вартість платформи, зручність використання, інформаційні потужності системи і звичайно швидкодію виконання запитів.

2.1.1. Переваги та недоліки застосування систем управління базами даних.

СУБД служить проміжним між користувачем та базою даних. Структура бази даних зберігається як колекція файлів. Ці дані можна отримати у цих файлах через

СУБД. СУБД приховує велику частину внутрішньої складності бази даних від прикладних програм та користувачів.

Переваги системи управління базами даних:

- Кращий обмін даними. Головною перевагою СУБД є те, що вона допомагає створити середовище, в якому кінцеві користувачі отримують кращий доступ до більшої кількості та структурованих даних. Цей структурований та простий доступ дає можливість кінцевим споживачам швидко реагувати на зміни у своєму середовищі;
- Краща безпека даних. Легка доступність створює більші ризики порушення безпеки даних. Корпорація інвестує значну кількість грошей, часу та зусиль для забезпечення кращої безпеки. СУБД забезпечує основу для кращого впровадження політики даних та конфіденційності даних;
- Мінімізована невідповідність даних. Результати невідповідності даних обумовлені змінною версією цих самих даних у різних місцях.;
- Поліпшення процесу прийняття рішень. Покращені та краще керовані дані дають змогу отримувати кращу якість інформації, яка допомагає у прийнятті кращих рішень. Якість обробленої інформації залежить від якості базових даних. СУБД не гарантує якості даних, але дає основу для полегшення ініціатив щодо якості даних;
- Підвищена продуктивність кінцевих споживачів. Легка доступність та доступність даних, а також інструменти, що перетворюють дані в корисну інформацію, дозволяють кінцевим користувачам приймати зважені рішення. Це приносить різницю успіху та невдачі у світовій економіці.

Відомі різні переваги СУБД, але вона, безумовно, має деякі недоліки.

- Збільшення витрат. Одним з недоліків СУБД є системи баз даних, що вимагають складного обладнання, програмного забезпечення та висококваліфікованого персоналу. Витрати на підтримку цих реквізитів та

управління системою баз даних можуть бути значними. Витрати на навчання, ліцензування та дотримання правил часто не враховуються, коли використовуються системи баз даних;

- Хитросплетіння управління. Системи баз даних взаємодіють з багатьма різними технологіями та мають значний вплив на ресурси та культуру компанії. Змінами, запровадженими впровадженням системи баз даних, слід належним чином керувати, щоб вони сприяли досягненню цілей компанії. З огляду на той факт, що в системах баз даних зберігаються найважливіші дані компанії, доступ до яких здійснюється з різних джерел, проблеми безпеки повинні постійно оцінюватися;
- Оскільки технологія баз даних швидко розвивається, витрати на навчання персоналу, як правило, значні. Враховуючи значні інвестиції в технології та навчання персоналу, компанії можуть не хотіти змінювати постачальників баз даних. Як наслідок, продавці рідше пропонують переваги в ціноутворенні існуючим клієнтам, і ці клієнти можуть мати обмежений вибір системних компонентів баз даних;
- Часті цикли оновлення товарів. Постачальники СУБД часто оновлюють свої продукти, додаючи нові функції. Такі нові функції часто входять до складу нових версій оновлення програмного забезпечення. Деякі з цих версій потребують оновлення апаратного забезпечення. Мало того, що самі оновлення коштують грошей, але це також коштує, щоб навчити користувачів та адміністраторів баз даних правильно використовувати та керувати новими функціями.

2.2. Обґрунтування вибору середовища Microsoft SQL Server

Роками підготовка бізнес-звітів за допомогою електронних таблиць та інших ручних методів була цілком прийнятною для компаній невеликого розміру. У міру зростання цих організацій стало необхідним переглянути, чи все ще працював такий

підхід, тим більше, що обсяги даних продовжували зростати і команди дедалі більше співпрацювали віртуально.

Оскільки ці обсяги даних неухильно зростають, необхідна нова технологія, щоб підтримувати зв'язок команд і швидко та легко обмінюватися проектною документацією. Як результат, багато компаній звертаються до бази даних, яка використовує технологію SQL, таку як SQL Server від Microsoft. Використовуючи надійну базу даних, яка пропонує вбудовану конфігурацію клієнт / сервер, компанії можуть перетворити свої дані на роботу за допомогою агрегування без особливих зусиль, щоб забезпечити повний огляд стану бізнесу та історичних показників.

Для великих та середніх компаній найважливіше те, що база даних SQL може використовувати великий обсяг даних у міру зростання бізнесу - до великого розміру в один терабайт. Однією з ключових причин продуктивності Microsoft SQL Server є використання передових багатоядерних процесорів, які можуть ефективно масштабуватися з мінімальним впливом на продуктивність. Це перетворюється на, здавалося б, безмежні можливості управління даними, навіть коли бізнес зростає та зростає вимога до бази даних.

У підприємницькій галузі можливість відстеження історичних даних про ефективність є безцінною. Можливості продуктивності та масштабованість Microsoft SQL Server означають, що величезні обсяги даних завжди у користувачів під рукою та на швидкості, що забезпечує стабільну, передбачувану продуктивність. Також незважаючи на можливість додавання кількох користувачів та облікових записів, Microsoft SQL Server забезпечує розумні заходи безпеки для обмеження доступу при необхідності. Адміністратори можуть визначати, які окремі користувачі мають доступ, а також рівень авторизації, який їм надається, для зміни та перегляду різних даних. Доступ користувача може бути додатково налаштований за допомогою конкретних поглядів на дані, що відображають мету цього індивідуального користувача отримати доступ до них.

Не менш важливим для підприємців є надійність платформи , яку вони будуть використовувати для ведення обліку. Поєднання безпеки та надійності є відмінною

рисою SQL Server, завдяки вражаючим можливостям безперебійної роботи, що полегшує розум кінцевих користувачів, зберігаючи доступ - навіть у випадку відключення системи. У разі аварійного завершення роботи або іншого виходу з ладу SQL Server має функції автоматичного відновлення, які набувають чинності автоматично, без необхідності втручання адміністратора. SQL Server автоматично запускає відновлення бази даних до останнього стану узгодженості. Якщо сенс створення бази даних полягає у створенні доступу на вимогу до величезних обсягів даних, ця надійність є ключовою.

Ще однією перевагою використання Microsoft SQL Server є швидкий та зручний аналіз даних. Сьогодні команди підприємницьких проектів більше не знаходяться в одному місці або просто базуються в домашньому офісі. SQL Server поєднує вражаючу надійність із можливістю видобування даних, коли та де це потрібно, та вбудованими інструментами звітування, які підтримують створення діаграм та графіки для насичених, привабливих презентацій. Дані можна легко передати в режими презентації або Excel, що дозволяє менеджерам проектів швидко складати відповідні точки даних у формати, придатні для обміну з клієнтами та партнерами. Завдяки цілодобовому доступу, можливостям інтерактивної звітності та негайному відновленню бази даних у випадку відключення, ваші команди можуть роздумувати і розраховувати на те, що SQL Server виведе здогадки з операцій з базою даних.

2.3. Етапи дизайну та розробки бази даних

Процес розробки баз даних складається з ряду фаз, де кожна з них має власну експертизу. Основними етапами є планування, аналіз потреб та реалізація. Кожен процес розділений на етапи.

Оскільки система баз даних є важливою складовою більшої інформаційної системи підприємства. Розробка бази даних - це процес проектування, впровадження та підтримання системи баз даних для задоволення стратегічних або

оперативних потреб організації чи підприємства, таких як покращена підтримка споживачів та задоволеність споживачів, кращий менеджмент виробництва в організації, кращий менеджмент запасів та більш точний прогнозування продажів. Процес розробки баз даних корисний для розробки різних типів відносин між даними в організації, до яких користувач або адміністратор може легко отримати доступ.

Зазвичай інформаційна система є більшим середовищем, яке оточує базу даних. Можна сказати, що база даних є частиною інформаційної системи. Інформаційна система містить велику кількість даних або інформації, яка включає всі результати, про які клієнт повинен запитати або замовити на сервері. Можна сказати, що інформаційна система є найбільшою платформою в системі управління базами даних.

2.3.1. Вимоги до проектування та аналізу БД

Завжди при замовлені структури бази даних, замовник має дати документацію функціональну та технічну, про те що б він хотів бачити в результаті розробки. Але часті ще за все замовник не володіє професійною мовою виконавця, і тому таку документацію з технічними вимогами дуже не часто зустрінеш. Саме тому виконавець зі своєї сторони має продемонструвати простою мовою як буде виглядати середовище і яку функціональність він може закласти в розробку. Дуже часто показують на прикладі вже створених власних або загальнодоступних проектів, для кращої візуалізації і розуміння. Наступним кроком, коли всі нюанси і недоліки були визначені, розробник має скласти технічну документацію для команди.

При створенні технічної документації заважають на такі фактори як:

- Перечислення вхідних та вихідних ресурсів та даних;
- Список вихідних даних, які не є необхідними для замовника, але які він повинен надати в інші організації (до вищих структур, до органів статистичного обліку, інші адміністративні та контролюючі організації).

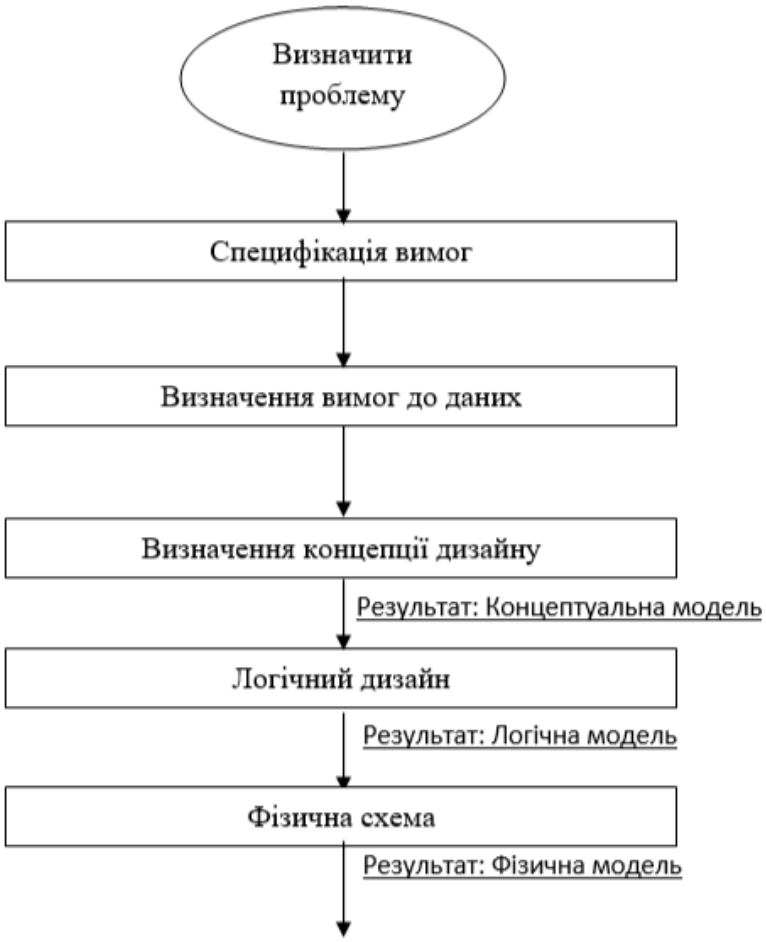


Рис 2.1. Порядок проектування бази даних

Першим кроком є збір вимог. На цьому кроці розробники баз даних повинні опитати клієнтів (користувачів бази даних), щоб зрозуміти запропоноване системи та отримати та задокументувати дані та функціональні вимоги. Результат цього кроку - документ, що включає докладні вимоги, передбачені користувачів.

Встановлення вимог передбачає консультації та узгодження всіх користувачі щодо того, які постійні дані вони хочуть зберігати разом із домовленістю до значення та інтерпретації елементів даних. Адміністратор даних відіграє ключову роль у цьому процесі, оскільки вони оглядають ділові, правові та етичні проблеми в організації, що впливає на вимоги до даних. Документ вимог до даних використовується для підтвердження розуміння вимоги до користувачів. Щоб

переконатися, що це легко зрозуміти, цього не повинно бути надмірно формальний або високо закодований. У документі має бути короткий виклад усіх вимоги користувачів - не просто сукупність вимог фізичних осіб – як намір полягає у розробці єдиної спільної бази даних. Вимоги не повинні описувати спосіб обробки даних, а навпаки що таке елементи даних, які атрибути вони мають, які обмеження застосовуються та взаємозв'язки між елементами даних.

Аналіз даних починається з викладу вимог до даних, а потім виробляє концептуальну модель даних. Метою аналізу є отримання детального опису дані, які відповідатимуть вимогам користувачів, щоб властивості як високого, так і низького рівня обробляються дані та їх використання. Сюди входять такі властивості, як можливе діапазон значень, які можна дозволити для атрибутів (наприклад, у шкільній базі даних. Наприклад, код курсу студента, назва курсу та кредитні бали). Концептуальна модель даних забезпечує спільне, офіційне представлення того, що є спілкування між клієнтами та розробниками під час розробки баз даних - вона орієнтована на дані в базі даних, незалежно від можливого використання цих даних в користувацьких процесах або реалізації даних у спеціальних комп'ютерних середовищах. Отже, концептуальна модель даних стосується значення та структури дані, але не з деталями, що впливають на те, як вони реалізовані. Тоді концептуальна модель даних є формальним поданням, за допомоги якої база даних повинна містити обмеження, яким повинні відповідати дані. Це повинно виражатись в термінах, які не залежать від того, як модель може бути реалізована. Та в результаті аналіз даних фокусується на питаннях: "Що потрібно?", а не “Як це досягнуто? ".

В якості вхідних даних ми можемо використовувати реляційне представлення концептуальної моделі даних процес логічного проектування. Результатом цього етапу є детальний реляційний специфікація, логічна схема, усіх таблиць та обмежень, необхідних для задоволення опис даних у концептуальній моделі даних. Саме під час цієї конструкції діяльність, при якій робиться вибір, які таблиці найбільш підходять представлення даних у базі даних. Цей вибір повинен

враховувати різні критерії проектування, включаючи, наприклад, можливість зміни, контроль дублювання і як найкраще представити обмеження. Це таблиці, визначені логічним схема, яка визначає, які дані зберігаються і як ними можна маніпулювати базою даних.

Проектувальники баз даних, знайомі з реляційними базами даних та SQL, можуть спокуситися перейти безпосередньо до реалізації після того, як вони створили концептуальну модель даних. Однак таке пряме перетворення реляційного подання в таблиці SQL не обов'язково призводить до створення бази даних, яка має всі бажані властивості: повнота, цілісність, ексклюзивність, ефективність та зручність використання. Хороші концептуальні дані - це важливий перший крок до бази даних із цими властивостями, але це не означає, що пряме перетворення в таблиці SQL автоматично виробляє хороша база даних. Цей перший крок точно представлятиме таблиці та обмеження необхідний для задоволення концептуального опису моделі даних, і тому задовольнить вимоги щодо повноти та цілісності, але це може бути нездійсненним або пропонувати погані зручність використання. Потім застосовується перший дизайн для поліпшення якості бази даних.

Після створення логічного дизайну нам потрібна наша база даних відповідно до формулювань, які ми створили. Реалізація реляційної СУБД, це, ймовірно, передбачає використання SQL для створення таблиць і обмежень, які задовольняють опис логічної схеми та вибір відповідної схеми зберігання (якщо СУБД дозволяє такий рівень контролю). Одним із способів досягти цього є запис відповідних операторів SQL DDL запитів що може виконуватися СУБД .Інший метод - це робота інтерактивно, використовуючи інструмент бази даних, такий як SQL Server Management Studio. Який би механізм не використовувався для реалізації логічної схеми, результат є таким що база даних з таблицями та обмеженнями визначена, але не буде містити користувацьких даних.

Коли вже визначено всі вхідні та вихідні данні які необхідні для підприємства, розробник може розпочати етап створення стркутури бази даних, зв’язків та головних таблиць проекту.

Починається розробка з визначення головних полів та атрибутів, які можуть бути досить чисельними. За цими атрибутами вже визначаються і призначаються до таблиць і розподіляється за правилами розробки. Головне на цьому етапі щоб атрибути розподілялись за їх функціональністю, також радять уникати перевантаження даних. Метою такого розподілу є забезпечення коректного введення даних, щоб кожна таблиця відповідала за свій спектр завдань і розподілялася на певні підрозділи. Далі обирають і визначають ключові поля(первинні ключі) кожної з таблиць, варто додати що ключі також можуть бути не один. Ключі необхідні для створення бінарної структурованості даних, що означає принцип дерева, і вони будуть автоматично відсортовані системою управління. Також функцією первинного ключа є повна унікальність, не може бути однакових первинних ключів в таблиці, якщо ваше поле не є унікальним, то воно і не може застосовуватись в якості первинного ключа. Коли вже визначені первинні ключі таблиць, розпочинають етап створення зв’язків між таблицями, тобто обрання зовнішніх ключів, яких може бути теж декілька. Тому і типи зв’язку між таблицями є різні: «один до одного», «один до багатьох» та «багато до багатьох», чим вони різняться і які краще використовувати на практиці я розгляну далі.

Головним фактором в розробці бази даних для підприємства є гнучкість, так як в будь-який момент роботи над проектом, може звернутися замовник з новими побажаннями і доповненнями. Тому якщо ви спроектуєте дуже жорстку структуру, яку не можна буде покращувати, це завдасть вам втрати великої кількості часу і непорозуміння з вашим замовником. Тож кваліфікований розробник завжди зважає на зовнішні фактори розробки і має бути гнучким до будь-яких змін у плані.

2.3.2. Вимоги до розробки БД

Ефективність функціонування інформаційної системи багато в чому залежить від її архітектури. В даний час перспективною вважається архітектура клієнт-сервер. У досить поширеному варіанті вона передбачає наявність комп'ютерної мережі і

розподіленої бази даних, що включає корпоративну базу даних і персональні бази даних, та розміщується на комп'ютері-сервері, а персональні бази даних розміщуються на комп'ютерах співробітників підрозділів, які є клієнтами корпоративних БД.

Правильно розроблена база даних надає вам доступ до актуальної, точної інформації. Оскільки правильний дизайн має важливе значення для досягнення ваших цілей у роботі з базою даних, інвестування часу, необхідного для вивчення принципів належного дизайну, має сенс. Зрештою, ви набагато частіше отримуєте базу даних, яка відповідає вашим потребам і може легко врахувати зміни.

Корпоративна база даних створюється, підтримується і функціонує під управлінням сервера бази даних, наприклад Microsoft SQL Server або Oracle Server. Залежно від розмірів організації і особливостей вирішуваних завдань інформаційна система може мати одну з наступних конфігурацій:

- Комп'ютер-сервер, що містить корпоративну і персональні бази;
- Комп'ютер-сервер і персональні комп'ютери з персональною базою даних;
- Кілька комп'ютерів-серверів і персональних комп'ютерів з персональними базами даних.

Використання архітектури клієнт-сервер дає можливість поступового нарощування інформаційної системи підприємства, по-перше, по мірі розвитку підприємства; по-друге, у міру розвитку самої інформаційної системи. Поділ загальної бази даних на корпоративну БД і персональні БД дає можливість зменшити складність проектування БД в порівнянні з централізованими варіантами, та допомагає знизити ймовірність помилок при проектуванні і вартість проектування.

Найважливішим достоїнством застосування БД в інформаційних системах є забезпечення незалежності даних від прикладних програм. Це дає можливість користувачам не займатися проблемами представлення даних на фізичному рівні: розміщення даних в пам'яті, методів доступу до них та інше. Така незалежність досягається підтримуваним СУБД багаторівневим поданням даних в базі даних на

логічному і фізичних рівнях. Завдяки СУБД і наявності логічного рівня представлення даних забезпечується відділення концептуальної моделі БД від її фізичного представлення в пам'яті.

Встановлені функціональні, нефункціональні та системні вимоги до засобу сканування мережі передачі даних. Серед основних функціональних вимог до засобу можна виділити: забезпечення неперервного сканування мережі передачі даних, визначення заголовків сканованих пакетів, визначення та відображення інформації про мережеві адаптери користувача, визначення та відображення інформації про скановані пакети, сканування вузлів мережі.

Деякі принципи керують процесом проектування бази даних. Перший принцип полягає в тому, що дубльована інформація (також її називають надлишковими даними) є поганою, оскільки вона витрачає простір і збільшує ймовірність помилок та невідповідностей. Другий принцип полягає у важливості правильності та повноти інформації. Якщо ваша база даних містить неправильну інформацію, будь-які звіти, які витягують інформацію з бази даних, також міститимуть неправильну інформацію. Як результат, будь-які прийняті вами рішення, які ґрунтуються на цих звітах, будуть невірними.

Отже, хорошим дизайном бази даних є такий, який:

- Розділяє вашу інформацію на таблиці на основі теми, щоб зменшити зайві дані;
- Забезпечує доступ до інформації, необхідної для об’єднання інформації в таблицях, за потреби;
- Допомагає підтримати та забезпечити точність та цілісність вашої інформації;
- Задовольняє ваші потреби в обробці даних та звітності.

Проектування бази даних зазвичай починається з визначення мети вашої бази даних. Потім відповідні дані збираються та упорядковуються у таблиці. Далі ви вказуєте первинні ключі та аналізуєте зв’язки між різними таблицями для

ефективного проектування даних. Після уточнення таблиць останнім кроком є застосування правил нормування для стандартизації таблиць.

Першим кроком є визначення мети вашої бази даних. Наприклад, якщо ви малий домашній бізнес, ви можете розробляти базу даних клієнтів, яка веде список споживчої інформації для створення електронних листів та звітів. Отже, розуміння важливості бази даних є життєве важливо. Наступним кроком є збір всілякої інформації, яку ви можете захотіти зберегти в базі даних. Починати необхідно із поточної інформації. Обміркувавши усі питання, можна вже вирішувати, які дані потрібно записати.

Після того, необхідно зібрати всі необхідні елементи даних, і наступним кроком є розподіл їх на основні сутності або предметні області. Потім кожна сутність стане окремою таблицею. Вже коли дані розділено на таблиці, так що кожен елемент даних стає полем і відображається у вигляді стовпця, необхідно уточнити набір стовпців і можливо доповнити. Це дозволить вам зручно фільтрувати інформацію.

Наступним кроком для вдосконалення дизайну бази даних є вибір первинного ключа для кожної таблиці. Цей первинний ключ - це стовпець або набір стовпців, який використовується для чіткого визначення кожного рядка. Це дозволить однозначно ідентифікувати кожен рядок на основі ідентифікатора клієнта. Також може існувати більше одного первинного ключа, який називається складеним ключем, включаючи кілька стовпців. Наприклад, у таблиці деталі замовлення первинними ключами можуть бути ідентифікатор замовлення та ідентифікатор товару. Складений ключ можна зробити за допомогою полів із подібними або різними типами даних. Поділивши дані на таблиці, інформацію слід зібрати значущим чином. Отже, необхідно розглянути кожну таблицю та визначте, як дані однієї таблиці пов’язані з даними іншої таблиці. За потреби можна додати поля або сформувати нові таблиці, щоб спростити взаємозв'язок на основі типів інформації.

Далі потрібно використати 3 принципи нормалізації бази даних. Нормалізація - процес організації даних, ціль якого уникнути дублювання та видання. Всього є 6 нормальних форм, але найголовнішими є 3 з них:

- Перша нормальна форма, базується на атомарності значень. Якщо всі атрибути є простими, усі використовувані домени повинні містити лише скалярні значення;
- Друга нормальна форма передбачає, що кожен стовпець, який не є ключем, повинен залежати від первинного ключа;
- Третя нормальна форма полягає в тому, що кожен стовпець, який не є ключем, повинен залежати тільки від первинного ключа.

Якщо підсумувати, то процес розробки бази даних допомагає спростити виконання та обслуговування корпоративної системи управління даними. Хороший дизайн бази даних може допомогти заощадити місце в пам'яті, зменшити надлишковість даних. Поряд із підтримкою точності та надійності даних, це дозволяє отримувати доступ до даних різними способами.Більше того, добре розроблена база даних простіша у використанні та обслуговуванні, що робить інтеграцію простою.

2.4. Висновок до розділу.

Були розглянуті різні системи управління базами даних і досліджені переваги і недоліки використання на різних за розміром комерційних підприємствах. Обґрунтовано вибір середовища Microsoft SQL Server, перевагами якого є: зручність розробки, багатофункціональність системи, взаємодія з іншими компонентами Microsoft Office, ціна використання платформи для комерційного підприємства та кількість професіональних кадрів, що працюють з платформою. Були розглянуті усі етапи дизайну, проектування та розробки бази даних і досліджено ключові моменти роботи з замовником, який представляє інтереси комерційного підприємства, щодо структури і функціональності розроблюваної системи.

РОЗДІЛ 3
ТЕХНОЛОГІЯ РОЗРОБКИ ІНФОРМАЦІЙНОЇ БАЗИ ДАНИХ

3.1. Аналіз підприємства з продажу електротехніки «Electronic Online Store»

Підприємство «Electronic Online Store» займається роздрібною торгівлею різних видів електронної техніки онлайн. Вони мають централізоване керівництво і постачальників продукції. Зважаючи на те що підприємство не має фізичного магазину з продажу продукції потрібно приділити увагу і категоріям продуктів які будуть підтягуватись з бази даних.

Загальні вимоги до бази даних будуть такими:

- Облік товарів в наявності;
- Збереження замовлень оформлених і навіть тих, які не були завершені, за для прогнозування цільового маркетингу для кожного клієнта;
- Облік усієї інформації щодо товарів і надання цієї інформації клієнтам в структурованому, зручному вигляді;
- Можливість швидкого пошуку товару за категоріями та підкатегоріями;
- Підрахунок загальної суми замовлення клієнта;
- Наявність знижкової системи для клієнтів;
- Можливість вибору різних типів доставки за адресою та на пошту;
- Облік постачальників товарів та оформлення поставки;

.

Тож головна задача розробки це створити зручну для клієнта структуровану базу даних, що буде забезпечувати підприємству легку взаємодію з клієнтами і вести облік замовлень та допоможе прогнозувати рекламу для клієнтів по їх запитам.

3.2. Розробка структури бази даних

Першим кроком є проектування основних інформаційних сутностей:

- Клієнт;
- Товар;
- Категорія товарів;
- Замовлення;
- Тип оплати;
- Тип доставки;
- Знижка;
- Постачальник;
- Статус замовлення.

Далі необхідно дослідити зв’язки між основними сутностями. Зв’язок «один до одного» позначується 1:1, «один до багатьох» позначується 1:n або n:1, «багато до багатьох» n:n. Зв’язок сутності з самою собою є неможливим тому у таблиці такий статус позначений як none. Зв’язки загалом визначаються таким чином, щоб визначити чи може бути декілька видів однієї сутності в іншій чи ні, чи вони можуть бути взагалі ніяк не пов’язані одна з одною. В таблиці 3.1 видно що ще є збитковість даних, тому що вона є неповною. Потрібно додати таблиці щоб уникнути зв’язку «багато до багатьох», а також додати таблиці що будуть відповідати за контент на сайті і підкатегорії товарів відсортувати за параметрами які є спільними для багатьох товарів, щоб забезпечити структурованість даних.

Таблиця 3.1.

Зв’язки між сутностями

	Клієнт	Товар	Категорія	Замовлення	Тип оплати	Тип доставки	Знижка	Постачальник	Статус
Клієнт	none	n-n	-	1-n	-	-	-	-	-
Товар	n-n	none	n-n	n-n		-		n-1	-
Категорія	-	n-n	none	-	-		-	-	-
Замовлення	n-1	n-n	-	none	1-1	n-1	1-n	-	1-n
Тип оплати	-	-	-	1-1	none	-	-	-	-
Тип доставки	-	-	-	1-n	-	none	-	-	-
Знижка	-	-	-	n-1	-		none	-	-
Постачальник	-	1-n	-	-	-		-	none	-
Статус	-	-	-	n-1	-	-	-	-	none

Після визначення зв’язків між основними таблицями ми можемо додавати додаткові таблиці щоб уникнути перевантаженості даних і почати розробку таблиць в Microsoft SQL Management Studio.

3.3. Розробка компонентів бази даних

Доповнивши початкову таблицю усіма необхідними компонентами я спроектувала такі таблиці з атрибутами.

Клієнт: Client (ID, name, phone, email, dt_created, dt_updated)

DESKTOP-QG6HK6G.O...tore - dbo.client | DESKTOP-QG6

Имя столбца	Тип данных
ID	int
name	varchar(20)
phone	char(12)
email	char(60)
dt_created	datetime
dt_updated	datetime

Категорія товарів: Product_category (ID, slug, dt_created, dt_updated)

DESKTOP-QG6HK6G.....product_category* | DESKTOP-Q

Имя столбца	Тип данных
ID	int
slug	varchar(50)
dt_created	datetime
dt_updated	datetime
	nchar(10)

Контент категорії: Category_content (product_category_id,[name],[language],[description]).

DESKTOP-QG6HK6G.....category_content | DESKTOP-QG6H

Имя столбца	Тип данных
ID	int
product_category_id	int
name	varchar(50)
language	varchar(25)
dt_created	datetime
dt_updated	datetime
description	varchar(1)

Під категорія товарів: Product_sub_category (product_category_id, slug, dt_created,dt_updated)

DESKTOP-QG6HK6G....uct_sub_category | DESKTOP-QG6H

Имя столбца	Тип данных
ID	int
product_category_id	int
slug	varchar(50)
dt_created	datetime
dt_updated	datetime

Контент підкатегорії товарів: Product_sub_category_content: (product_sub_category_id,[name],[language],[description],dt_created,dt_updated)

DESKTOP-QG6HK6G....category_content | DESKTOP-QG6H

Имя столбца	Тип данных
ID	int
product_sub_category_id	int
name	varchar(50)
language	varchar(25)
dt_created	datetime
dt_updated	datetime
description	varchar(1)

Товар: Product (slug, price, code, count_available, dt_created, dt_updated)

DESKTOP-QG6HK6G....ore - dbo.product | DESKTOP-QG

Имя столбца	Тип данных
ID	int
slug	varchar(50)
price	float
code	int
count_available	int
dt_created	datetime
dt_updated	datetime

Контент товару: Product_content (product_id,[name],[language],[description)

DESKTOP-QG6HK6G.....product_content | DESKTOP-QG6

Имя столбца	Тип данных
ID	int
product_id	int
name	varchar(50)
language	varchar(25)
dt_created	datetime
dt_updated	datetime
description	varchar(1)

Параметри підкатегорії товару: Product_sub_category_param (slug, product_sub_category_id, dt_created, dt_updated)

DESKTOP-QG6HK6G...b_category_param | DESKTOP-QG

Имя столбца	Тип данных
ID	int
product_sub_category_id	int
slug	varchar(50)
dt_created	datetime
dt_updated	datetime

Контент параметрів підкатегорії товару: Product_sub_category_param_content ([name],product_sub_category_param_id,[language],dt_created,dt_updated)

DESKTOP-QG6HK6G....ry_param_content | DESKTOP-QG6

Имя столбца	Тип данных
ID	int
product_sub_category_param_id	int
name	varchar(50)
language	varchar(25)
dt_created	datetime
dt_updated	datetime

Значення параметрів підкатегорії товару: Product_sub_category_param_value (product_sub_category_param_id,product_id,[value],dt_created,dt_updated)

DESKTOP-QG6HK6G...gory_param_value | DESKTOP-QG6

Имя столбца	Тип данных
ID	int
product_sub_category_param_id	int
product_id	int
value	text
dt_created	datetime
dt_updated	datetime

Замовлення:Order(payment_type_id,delivery_type_id,price,price_fee,total_price,status_id,comment,client_id,[address],discount_code_id,discount_amount)

DESKTOP-QG6HK6G....Store - dbo.order | DESKTOP-QG6HK6G...

Имя столбца	Тип данных
ID	int
payment_type_id	int
delivery_type_id	int
status_id	int
client_id	int
price	float
price_fee	float
total_price	float
comment	text
address	varchar(50)
discount_code_id	int
discount_amount	float
dt_created	datetime
dt_updated	datetime

Тип оплати: Payment_type (slug, fee, dt_created, dt_updated)

DESKTOP-QG6HK6G...bo.payment_type

Имя столбца	Тип данных
ID	int
slug	varchar(35)
fee	float
dt_created	datetime
dt_updated	datetime

Контент типу оплати: Payment_type_content (payment_type_id, [name], [language], dt_created,dt_updated)

DESKTOP-QG6HK6G....ent_type_content

Имя столбца	Тип данных
ID	int
payment_type_id	int
name	varchar(35)
language	varchar(25)
dt_created	datetime
dt_updated	datetime

Тип доставки: Delivery_type (slug, fee, dt_created, dt_updated)

DESKTOP-QG6HK6G....dbo.delivery_type

Имя столбца	Тип данных
ID	int
slug	varchar(35)
fee	float
dt_created	datetime
dt_updated	datetime

Контент типу доставки: Delivery_type_content (delivery_type_id, [name], [language], dt_created,dt_updated)

DESKTOP-QG6HK6G....very_type_content — DESKTOP-QG6

Имя столбца	Тип данных
ID	int
delivery_type_id	int
name	varchar(35)
language	varchar(25)
dt_created	datetime
dt_updated	datetime

Статус замовлення: Status (slug, dt_created, dt_updated)

DESKTOP-QG6HK6G....tore - dbo.status — DESKTOP-QG6

Имя столбца	Тип данных
ID	int
slug	varchar(35)
dt_created	datetime
dt_updated	datetime

Контент статусу замовлення: Status_content (status_id, [name], [language], [description], dt_created, dt_updated)

DESKTOP-QG6HK6G....bo.status_content — DESKTOP-QG6

Имя столбца	Тип данных
ID	int
status_id	int
name	varchar(35)
language	varchar(25)
description	text
dt_created	datetime
dt_updated	datetime

Код на знижку: Discount_code (code_is_used,amount, is_persent, dt_created,dt_updated)

DESKTOP-QG6HK6G....bo.discount_code | DESKTOP-QG6

Имя столбца	Тип данных
ID	int
code_is_used	bit
amount	float
is_persent	float
dt_created	datetime
dt_updated	datetime

Таблиця що зв’язує замовлення та товар: Order_product:(order_id, product_id, is_actual, dt_created, dt_updated)

DESKTOP-QG6HK6G....bo.order_product | DESKTOP-QG6

Имя столбца	Тип данных
ID	int
order_id	int
product_id	int
is_actual	bit
dt_created	datetime
dt_updated	datetime

Постачальник: Supplier ([name], [address], phone, email, dt_created, dt_updated)

DESKTOP-QG6HK6G....re - dbo.supplier | DESKTOP-QG6H

Имя столбца	Тип данных
ID	int
name	varchar(35)
address	varchar(50)
phone	char(12)
email	varchar(60)
dt_created	datetime
dt_updated	datetime

Поставка товарів: Supply (supplier_id,price,dt_created,dt_updated)

DESKTOP-QG6HK6G....tore - dbo.supply | DESKTOP-QG6

	Имя столбца	Тип данных
🔑	ID	int
	supplier_id	int
	price	float
	dt_created	datetime
	dt_updated	datetime

Таблиця що зв’язує замовлення на поставку та товар: Supply_product (supply_id,product_id,[count],price,supply_status_id,dt_created,dt_updated)

DESKTOP-QG6HK6G....o.supply_product | DESKTOP-QG6H

	Имя столбца	Тип данных
🔑	ID	int
	supply_id	int
	product_id	int
	count	int
	price	float
	supply_status_id	int
	dt_created	datetime
	dt_updated	datetime

Статус замовлення на поставку: Supply_status (slug,dt_created,dt_updated)

DESKTOP-QG6HK6G....dbo.supply_status | DESKTOP-QG6

	Имя столбца	Тип данных
🔑	ID	int
	slug	varchar(50)
	dt_created	datetime
	dt_updated	datetime

Контент замовлення на поставку: Supply_status_content (supply_status_id,[name],[language],dt_created,dt_updated)

DESKTOP-QG6HK6G....ly_status_content — DESKTOP-QG6H

	Имя столбца	Тип данных
🔑	ID	int
	supply_status_id	int
	name	varchar(35)
	language	varchar(25)
	dt_created	datetime
	dt_updated	datetime

Далі формуємо E/R – діаграму зв’язків між таблицями:

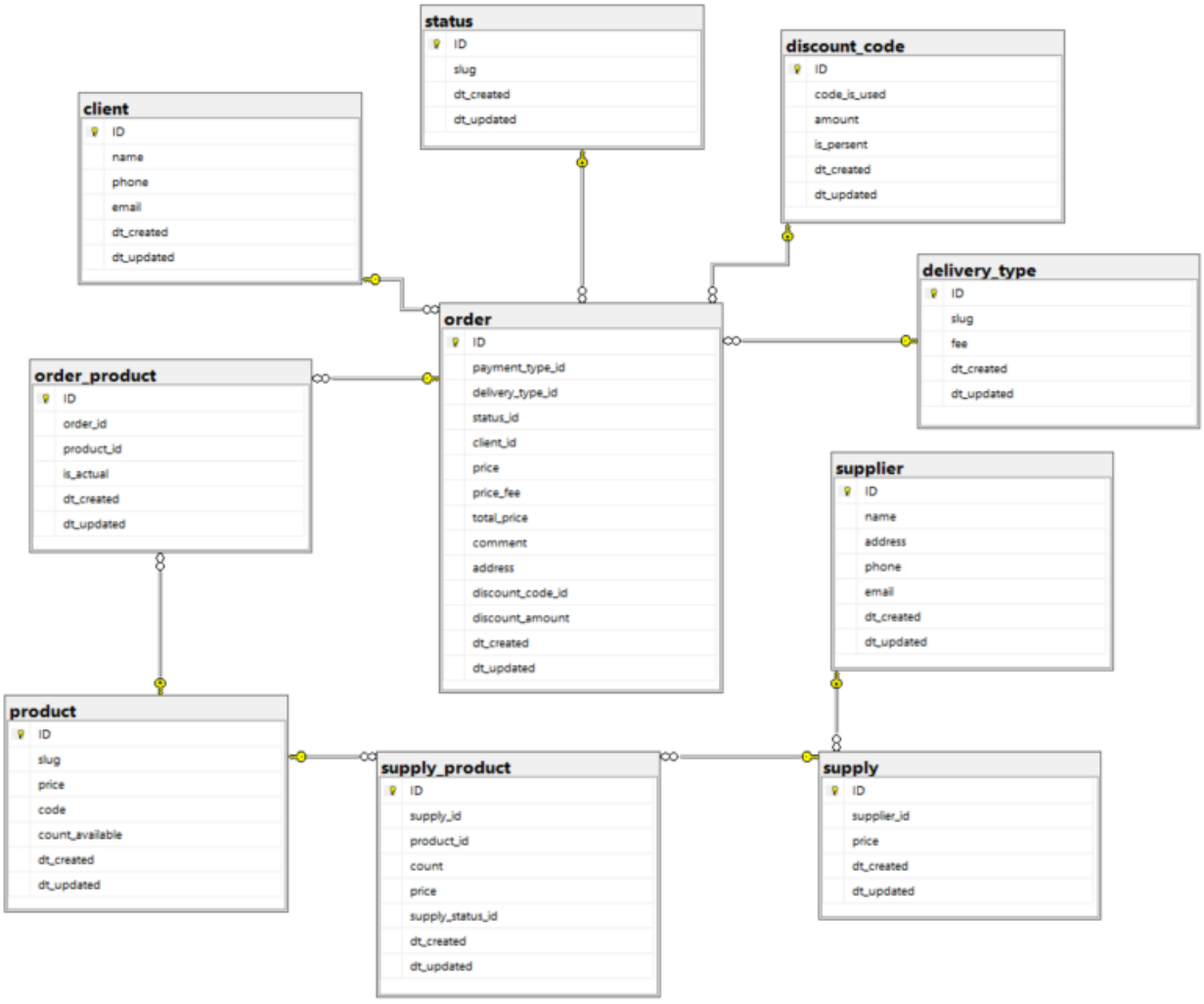


Рис 3.2. E/R – діаграма(не повна)

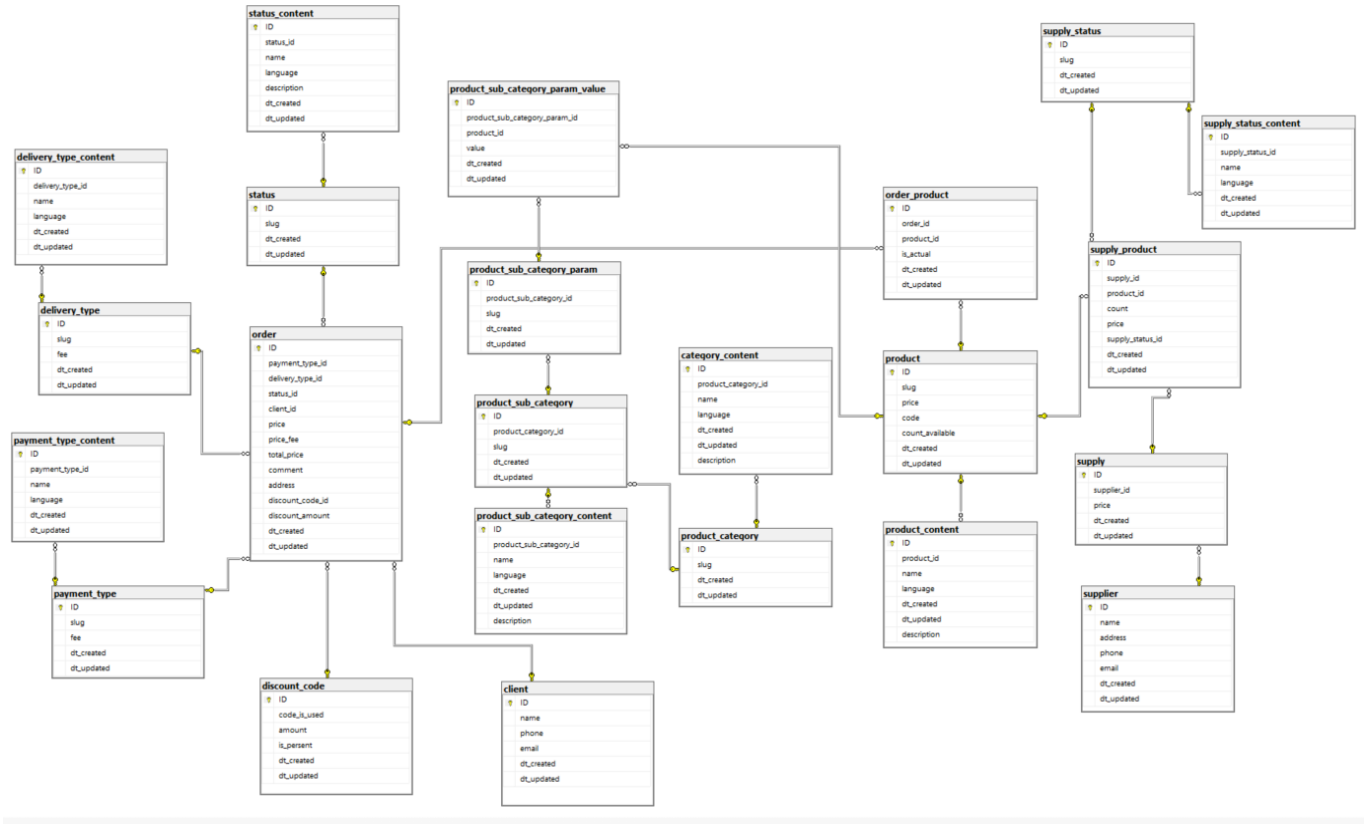


Рис 3.3. E/R – діаграма(повна)

3.4. Додання тригерів і запитів до бази даних

Для того щоб у базі даних, яку я створила розраховувалась загальна сума з усіма знижками та податками, необхідно додати тригери.

Тригер в SQL Server - це особливий тип збережених процедур, який автоматично виконується, коли подія відбувається на конкретному сервері бази даних. SQL Server надає нам два основних типи тригерів: тригери DML та тригери DDL. Тригери DDL спрацьовують у відповідь на різні події мови визначення даних (DDL), такі як виконання операторів CREATE, ALTER, DROP, GRANT, DENY та REVOKE T-SQL. Тригер DDL може реагувати на дії DDL, запобігаючи цим змінам впливати на базу даних, виконувати іншу дію у відповідь на ці дії DDL або реєструючи ці зміни, які виконуються щодо бази даних.

Тригер SQL Server DML - це особливий тип збережених процедур, призначений для виконання послідовності дій над таблицею бази даних, до якої приєднано тригер, під час подій мови маніпуляції даними (DML), таких як INSERT, UPDATE або DELETE дія, відбувається для зміни вмісту таблиць бази даних або подань, незалежно від того, чи впливають на рядки таблиці. Тригери відрізняються від збережених процедур тим, що тригери запускаються автоматично, коли відбувається заздалегідь визначена зміна даних. Тригери DML можна використовувати для підтримання цілісності даних та забезпечення ділових правил компанії, так само, як перевірка таблиці та функціональні обмеження зовнішніх ключів, виконуючи процеси аудиту та інші дії DML після.

Якщо спрацьовує тригер, для збереження значень даних до і після модифікації буде використовуватися спеціальний тип віртуальних таблиць, що називається Вставлені та Видалені таблиці. Оператор тригера працюватиме в межах тієї самої транзакції, яка запускає цей тригер. Це означає, що транзакція не буде здійснена повністю, поки оператор тригера не буде успішно завершено.

Тригер dbo.upd_product_order_trigger створений щоб після додання товару у замовлення з таблиці товарів автоматично додавалася і сума товару за 1 одиницю:

```
CREATE TRIGGER dbo.upd_product_order_trigger
    ON dbo.order_product
    FOR UPDATE
AS
    Declare @product_id int;
    Declare @product_price float;
    Declare @order_id float;

    select @product_id = product_id from inserted;
    select @order_id = order_id from inserted;
    select @product_price = price from product where id = @product_id;

    update "order"
    set price = price - @product_price
        where id = @order_id;
```

Тригер dbo.upd_set_order_confirmed необхідний для розрахунку вартості замовлення зі знижкою та за додаванням податків:

```
CREATE TRIGGER dbo.upd_set_order_confirmed
ON dbo."order"
FOR UPDATE
AS
    Declare @payment_type_fee float;
    Declare @delivery_type_fee float;
    Declare @id int;
    Declare @status_id int;
    Declare @discount_code_id int;
    Declare @discount_amount float;
    select @discount_code_id = discount_code_id from inserted;
    select @status_id = status_id from inserted;
    select @id = id from inserted;
    select @payment_type_fee = fee from payment_type where id = (select
payment_type_id from "order" where id = @id);
    select @delivery_type_fee = fee from delivery_type where id = (select
delivery_type_id from "order" where id = @id);
    select @discount_amount = amount from discount_code where id =
@discount_code_id;
    If @status_id=3
        begin
            update "order"
            set total_price = price + (price * @delivery_type_fee / 100) + (price *
@payment_type_fee / 100) - @discount_amount,
            price_fee = (price * @delivery_type_fee / 100) + (price *
@payment_type_fee / 100),
            discount_amount = @discount_amount
            where id = @id;
            end
```

```
678
679     update "order" set status_id = 3 where id =1;
```

100 %

Результаты | Сообщения

	ID	pay...	deliv...	sta...	c...	c...	address	disc...	dt_created	dt_updated	price	price_fee	total_price	discount_amount
1	1	1	1	3	1		м.Київ,ву...	1	2021-05-24 00:12:03.980	2021-05-25 00:12:03.980	20000	420	20370	50
2	2	2	3	4	2		м.Київ,ву...	2	2021-05-23 00:12:03.980	2021-05-25 00:12:03.980	0	0	0	0

Рис. 3.4. Розрахунок повної вартості замовлення за допомогою тригерів

Тепер можемо побачити розрахунки за допомогою запитів UPDATE на рис. 3.4. та рис. 3.5.Так як тригер спрацьовує на команди UPDATE та INSERT, у вже заповненій таблиці прорахували лише перше замовлення, тому що в запиті ми вказали що хочемо оновити лише замовлення з ID що дорівнює 1. Тепер оновимо таблицю Замовлень повністю і покажемо результат розрахунків замовлень.

Результаты | Сообщения

ID	payment_type...	deliv...	status_id	client_id	address	discount_code...	price	price_fee	total_price	discount_amo
1	1	1	3	1	м.Київ,вул.Жиля...	1	20000	420	20370	50
2	2	3	3	2	м.Київ,вул.Ніжин...	2	23900	717	24667	50
3	2	3	3	2	м.Київ,вул.Ніжин...	2	35000	1050	36000	50

Рис. 3.5. Розрахунок повної вартості замовлення

3.5. Створення системних збережених процедур для бази даних

Системні збережені процедури призначені для виконання особистих адміністративних дій. Практично всі дії по адмініструванню сервера виконуються з їх допомогою. Можна сказати, що системні збережені процедури є інтерфейсом, що забезпечує роботу з системними таблицями. Системні збережені процедури мають префікс sp_, зберігаються в системній базі даних і можуть бути викликані в контексті будь-якої іншої бази даних.

Призначені для користувача процедури, що реалізують ті чи інші дейст- вія. Збережені процедури - повноцінний об'єкт бази даних. Внаслідок цього кожна збережена процедура розташовується в конкретній базі данних, де і виконується.

Тимчасові збережені процедури існують лише деякий час, після чого автоматично знищуються сервером. Вони діляться на ло- Кальне і глобальні. Локальні тимчасові процедури, що можуть бути викликані тільки з того з'єднання, в якому створені. При созда- ванні такої процедури їй необхідно дати ім'я, що починається з одного символу «#». Як і всі тимчасові об'єкти, збережені процедури цього типу автоматично видаляються при відключенні користувача, перезапуску або зупинці сервера. Глобальні тимчасові процедури, що доступні для будь-яких з'єднань сервера, на якому є така ж процедура. Для її визначення достатньо дати їй

ім'я, яке починається з символів «##». Видаляються ці процедури при перезапуску або зупинці сервера, а також при закритті з'єднання, в контексті якого вони були створені.

Створення збереженої процедури передбачає вирішення таких задач: планування прав доступу. При створенні збереженої процедури слід враховувати, що вона буде мати ті ж права доступу до об'єктів бази даних, що і створив її користувач; визначення параметрів храни- мій процедури, процедури, що можуть мати вхідними і вихід-ними параметрами; розробка коду збереженої процедури. Код процедури може містити послідовність будь-яких команд SQL, включаючи виклик інших процедур.

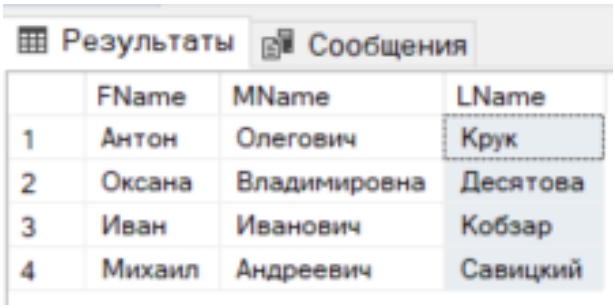
Перший приклад збереженої процедури, наприклад необхідно створити процедуру, яка дозволить знайти покупця за прізвищем і/або містом, при цьому якщо ми не передаємо параметрів для пошуку, необхідно вивести усіх покупців. Також ми можемо знайти всіх покупців, прізвища яких починаються, наприклад на певну літеру. І аналогічно такий самий пошук реалізувати для назви міста.

```
CREATE PROCEDURE ClientByLNameOrCity
        @Word1 nvarchar(25) = NULL,
        @Word2 nvarchar(25) = NULL
AS
IF @Word1 IS NOT NULL AND @Word2 IS NOT NULL
        SELECT [name]
        FROM Client
        WHERE [name] LIKE concat(@Word1 , '%') AND City LIKE concat(@Word2,'%')
 ELSE IF @Word1 IS NOT NULL AND @Word2 IS NULL
        SELECT [name]
        FROM client
        WHERE [name] LIKE concat(@Word1 , '%') OR City LIKE concat(@Word1,'%')
ELSE

        SELECT [name]
                FROM client;
```

Результат виконання процедури:

EXECUTE ClientByNameOrCity '','Киев';


Результаты Сообщения

	FName	MName	LName
1	Антон	Олегович	Крук
2	Оксана	Владимировна	Десятова
3	Иван	Иванович	Кобзар
4	Михаил	Андреевич	Савицкий

Також для спрощення роботи менеджерів додаймо процедуру яка буде зручно додавати новий продукт в базу даних. Процедура має приймати назву товару, колір опис, а кількість може бути не внесена.

```
CREATE PROCEDURE InsertProduct
        @Name varchar(100),
        @Color varchar(15) = NULL,
        @Description varchar(400)=NULL,
        @Qty int = NULL
AS
        Declare @ID int;
        Insert into Products([Name])
        Values (@Name);
        Set @ID = @@Identity;
        IF @Qty IS NOT NULL
                begin
                        insert into Stocks(ProductID,Qty)
                        Values (@ID,@Qty);
                end;
        IF @Color IS NOT NULL AND @Description IS NOT NULL
                begin
                        insert into ProductDetails(ID,Color,[Description])
                        Values (@ID,@Color,@Description);
                end;
```

Далі проведемо тест при доданні 3 продуктів з різними вхідними даними:

EXECUTE InsertProduct 'Product1';
EXECUTE InsertProduct 'Product2','Blue','Some description',5;
EXECUTE InsertProduct 'Product3',Null,Null,10;

	ID	Name
1	1	Ноутбук Asus D345
2	2	Ноутбук HP Pavilion 15-p032er
3	3	Ноутбук Dell Inspiron 5555
4	4	Нетбук Acer Aspire ES1
5	5	Нетбук Lenovo Flex 10
6	6	Нетбук Dell Inspiron 3147
7	7	Смартфон Samsung Galaxy S6 SS 32GB
8	8	Смартфон Apple iPhone 6
9	9	Фотоаппарат Canon PowerShot SX400
10	10	Телевизор LG 55LB631V
11	11	Product1
12	12	Product2
13	13	Product3
14	14	Product1
15	15	Product2
16	16	Product3

Процедура виповнилась і ми мінімізували написання коду з 3 речень до 4 слів і це є найважливішим в автоматизації бізнес-процесів підприємства. І таких збережених процедур можна додати безліч за проханням замовника. Завдяки процедурам підвищується автоматизованість системи та швидкодія виконання запитів, це покращує зручність та процес роботи стає набагато простішим.

3.6. Висновок до розділу.

В результаті виконання дипломної роботи, було спроектована та розроблена інформаційна система підприємства з продажу електроніки «Electronic Online Store».

Актуальність розробки системи полягає в необхідності застосування сучасних систем управління базами даних за для конкурентоспроможності, зручності використання клієнтами підприємства, збільшення ефективності роботи та заощадження коштів.

Застосування даної інформаційної системи дасть змогу автоматизувати бізнес-процеси підприємства, ведення обліку замовлень, створення цільового маркетингу для клієнтів, слідкувати за товарами та розраховувати прибутки компанії.

Результатом проектування стала база даних, яка реалізує логіку взаємодії з клієнтами, та надає необхідну інформацію підприємству. У процесі роботи над проектом була досліджена і вивчена предметна область та виявлені переваги та недоліки розробки власної інформаційної системи.

ВИСНОВКИ

В результаті виконання дипломної роботи, була спроектована та розроблена інформаційна система керування взаємовідносинами з клієнтами на прикладі підприємства з продажу електротехніки. Була спроектована та реалізована база даних системи, яка забезпечує доступ до інформації клієнтів.

Актуальність теми проекту обумовлена необхідністю автоматизації рутинних завдань з метою зниження трудових і тимчасових витрат, створенням єдиної бази знань для зберігання досвіду роботи і готових рішень типових проблем, а також загальної оптимізації роботи підрозділу за рахунок більш ефективного розподілу завдань між співробітниками, з огляду на їх досвід і набір знань. Була досягнута ціль розробки це автоматизація бізнес-процесу комерційного підприємства з продажу електронних приладів та реалізована структура відображення інформації на сайті для зручності клієнтів.

Для реалізації були використані платформи Microsoft SQL Server Management Studio, та написано на мові MSSQL, що дає можливості створення складних структурних запитів та зручний інтерфейс для розробки.

Були вирішені наступні задачі:

– визначені функції, які повинна виконувати система;

– визначені учасники інформаційного обміну у системі;

– проведено вибір архітектури та програмних засобів реалізації клієнтських застосувань для різних категорій користувачів системи;

– спроектована та реалізована база даних системи, яка забезпечує доступ до інформації.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Create Database Microsoft. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/ (дата звернення 10.05.21р) – Назва з екрана.
2. Create Trigger (Transact-SQL). [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/ (дата звернення 10.05.21р) – Назва з екрана.
3. Пасічник В. В. Організація баз даних та знань / В. В. Пасічник, В. А. Резніченко. – К.: Видавнича група BHV, 2006. – 384 с.
4. Пономаренко В. С. Інформаційні системи в управлінні персоналом.. Навчальний посібник / В. С. Пономаренко, І. В. Журавльова, І. Л. Латишева. – Харків: Вид. ХНЕУ, 2008. – 336 с. (Укр. мов.) Електрон. аналог друк. вид.: режим доступу: http://www.repository.hneu.edu.ua/ (дата звернення 18.04.2021 р). – Назва з екрана.
5. Хомоненко А. Д. Базы данных: Учебник для высших учебных заведений. Навчальний посібник / А. Д. Хомоненко, В. М. Цыганков, М. Г. Мальцев - Санкт-Петербург: Вид КОРОНА-Век, 2009. - 736 с. - Електрон. аналог друк. вид.: режим доступу: https://studizba.com/ (дата звернення 01.05.2021 р). – Назва з екрана.
6. Г. Гарсиа-Молина Системы баз данных / Д. Ульман, Д. Уидом. Полный курс. – М.: Вильямс, 2003, 1088 с.
7. Харитонов В. И. Применение CRM-систем при принятии управленческих решений в организации. – 2016. – №10. - режим доступу: http://sisupr.mrsu.ru/ (дата звернення 08.05.2021 р). – Назва з екрана.
8. Кавун С.В. Інформаційна безпека. Навчальний посібник. Ч.1/ С.В. Кавун, В.В. Носов, О.В. Мажай. – Харків: Вид. ХНЕУ, 2008. – 352 с.
9. Т. Коннолли Базы данных: проектирование, реализация и сопровождение. Теория и практика / К. Бегг, А. Страчан М.; Издательский дом «Вильямс». 2000 г. - 761с.

10. Карпенко М. Ю. Інформаційні системи і технології в управлінні організацією. Навчальний посібник /М. Ю. Карпенко, В. Б. Уфимцева – Харків: Вид. ХНАМГ, 2012 – 96 с. Електрон. аналог друк. вид.: режим доступу: https://eprints.kname.edu.ua/ (дата звернення 18.04.2021 р). – Назва з екрана.
11. Watt A. Database Design – 2nd Edition / Eng N. [Electronic resource]: – Publisher: The BCcampus Open Textbook Project, 2014. – 142 s. – Access mode: https://opentextbc.ca/ (lastaccess: 17.05.21.) – Title from the screen.
12. Vessey I. Journal of Management Information Systems / I. Vessey, V. Ramesh, R. L. Glass. [Electronic resource]: – Publisher: Taylor & Francis, Ltd. 2002. – 174 s. – Access mode: https://www.jstor.org/ (lastaccess: 10.05.21.) – Title from the screen.
13. Нечипоренко О.В. Механізми забезпечення захисту баз даних в сучасних СУБД. [Електронний ресурс]: Навчальний посібник - Черкаси: Вид. Вісник ЧДТУ, 2014. – 85 с. (Укр. мов.) Електрон. аналог друк. вид.: режим доступу: http://www.irbis-nbuv.gov.ua/ (дата звернення 02.05.2021 р). – Назва з екрана.
14. Огляд систем управління базами даних для систем контролю і управління доступом. – Режим доступу: https://bumotors.ru/ (дата звернення 03.04.21). – Назва з екрану
15. Базюк О.Д., Михалевич В.М. Використання CRM-системи для управління взаємовідносинами з клієнтами / О.Д. Базюк, В.М. Михалевич [Електронний ресурс] – Режим доступу: http://nauka.mk.ua/ (дата звернення 08.05.2021 р). – Назва з екрана.
16. Можливості використання CRM-систем. [Електронний ресурс]. – Режим доступу: https://www.terrasoft.ua (дата звернення 01.04.2021 р). – Назва з екрана.

ДОДАТКИ

Додаток А

SQL запити створення таблиць бази даних:

```
CREATE DATABASE OnlineElectronicStore
COLLATE Cyrillic_General_CI_AS;

CREATE TABLE client
(
    ID int Identity(1,1) NOT NULL Primary Key,
    [name] varchar(20) NOT NULL,
    phone char(12) NOT NULL,
    email char(60) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );
CREATE TABLE product_category
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    slug varchar(50) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE category_content
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    product_category_id int FOREIGN KEY REFERENCES product_category(ID),
    [name] varchar(50) NOT NULL,
    [language] varchar(25) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    [description] varchar NOT NULL,
```

```
    );
CREATE TABLE product_sub_category
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    product_category_id int FOREIGN KEY REFERENCES product_category(ID),
    slug varchar(50) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE product_sub_category_content
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    product_sub_category_id int FOREIGN KEY REFERENCES product_sub_category(ID),
    [name] varchar(50) NOT NULL,
    [language] varchar(25) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    [description] varchar NOT NULL,
    );

CREATE TABLE product
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    slug varchar(50) NOT NULL,
    price float NOT NULL,
    code int NOT NULL,
    count_available int NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE product_content
```

```
(
        ID int IDENTITY(1,1) NOT NULL Primary Key,
        product_id int FOREIGN KEY REFERENCES product(ID),
        [name] varchar(50) NOT NULL,
        [language] varchar(25) NOT NULL,
        dt_created datetime NOT NULL,
        dt_updated datetime NOT NULL,
        [description] varchar NOT NULL,
        );

CREATE TABLE product_sub_category_param
(
        ID int IDENTITY(1,1) NOT NULL Primary Key,
        product_sub_category_id int FOREIGN KEY REFERENCES product_sub_category(ID),
        slug varchar(50) NOT NULL,
        dt_created datetime NOT NULL,
        dt_updated datetime NOT NULL,
        );

CREATE TABLE product_sub_category_param_content
(
        ID int IDENTITY(1,1) NOT NULL Primary Key,
        product_sub_category_param_id int FOREIGN KEY REFERENCES
product_sub_category_param(ID),
        [name] varchar(50) NOT NULL,
        [language] varchar(25) NOT NULL,
        dt_created datetime NOT NULL,
        dt_updated datetime NOT NULL,
        );

CREATE TABLE product_sub_category_param_value
(
        ID int IDENTITY(1,1) NOT NULL Primary Key,
```

```
product_sub_category_param_id int FOREIGN KEY REFERENCES product_sub_category_param(ID),
        product_id int FOREIGN KEY REFERENCES product(ID),
        [value] text NOT NULL,
        dt_created datetime NOT NULL,
        dt_updated datetime NOT NULL,
        );


CREATE TABLE payment_type
(
        ID int IDENTITY(1,1) NOT NULL Primary Key,
        slug varchar(35) NOT NULL,
        fee float NOT NULL,
        dt_created datetime NOT NULL,
        dt_updated datetime NOT NULL,
        );

CREATE TABLE payment_type_content
(
        ID int IDENTITY(1,1) NOT NULL Primary Key,
        payment_type_id int FOREIGN KEY REFERENCES payment_type(ID),
        [name] varchar(35) NOT NULL,
        [language] varchar(25) NOT NULL,
        dt_created datetime NOT NULL,
        dt_updated datetime NOT NULL,
        );

CREATE TABLE delivery_type
(
        ID int IDENTITY(1,1) NOT NULL Primary Key,
        slug varchar(35) NOT NULL,
        fee float NOT NULL,
        dt_created datetime NOT NULL,
```

```
    dt_updated datetime NOT NULL,
    );

CREATE TABLE delivery_type_content
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    delivery_type_id int FOREIGN KEY REFERENCES delivery_type(ID),
    [name] varchar(35) NOT NULL,
    [language] varchar(25) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE [status]
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    slug varchar(35) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE status_content
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    status_id int FOREIGN KEY REFERENCES [status](ID),
    [name] varchar(35) NOT NULL,
    [language] varchar(25) NOT NULL,
    [description] text NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE discount_code
```

```
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    code_is_used bit NOT NULL,
    amount float NULL,
    is_persent float NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE [order]
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    payment_type_id int FOREIGN KEY REFERENCES payment_type(ID),
    delivery_type_id int FOREIGN KEY REFERENCES delivery_type(ID),
    status_id int FOREIGN KEY REFERENCES [status](ID),
    client_id int FOREIGN KEY REFERENCES client(ID),
    price float NOT NULL,
    price_fee float NOT NULL,
    total_price float NOT NULL,
    comment text NOT NULL,
    [address] varchar(50) NOT NULL,
    discount_code_id int FOREIGN KEY REFERENCES  discount_code(ID),
    discount_amount float NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE order_product
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    order_id int FOREIGN KEY REFERENCES [order](ID),
    product_id int FOREIGN KEY REFERENCES product(ID),
    is_actual bit NOT NULL,
```

```
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE supplier
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    [name] varchar(35) NOT NULL,
    [address] varchar(50) NOT NULL,
    phone char(12) NOT NULL,
    email varchar(60) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE supply
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    supplier_id int FOREIGN KEY REFERENCES supplier(ID),
    price float NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );
CREATE TABLE supply_status
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    slug varchar(50),
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );

CREATE TABLE supply_status_content
```

```
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    supply_status_id int FOREIGN KEY REFERENCES supply_status(ID),
    [name] varchar(35) NOT NULL,
    [language] varchar(25) NOT NULL,
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
    );


CREATE TABLE supply_product
(
    ID int IDENTITY(1,1) NOT NULL Primary Key,
    supply_id int FOREIGN KEY REFERENCES supply(ID),
    product_id int FOREIGN KEY REFERENCES product(ID),
    [count] int NOT NULL,
    price float NOT NULL,
    supply_status_id int FOREIGN KEY REFERENCES supply_status(ID),
    dt_created datetime NOT NULL,
    dt_updated datetime NOT NULL,
        );
```

Додаток Б

Заповнення бази даних інформацією

```
INSERT client
([name], phone, email, dt_created, dt_updated)
VALUES
('Виктор Прокопенко','(093)1416433','vicktorproko6678@gmail.com',DATEADD(DAY,
-85, GETDATE()),DATEADD(DAY, -85, GETDATE())),
('Антон Крук','(050)1416433','antonKruk234@email.com',DATEADD(DAY, -85,
GETDATE()),DATEADD(DAY, -85, GETDATE())),
('Оксана Десятова','(068)0989367','oksghjyh3456@ukr.com',DATEADD(DAY, -85,
GETDATE()),DATEADD(DAY, -85, GETDATE())),
('Антонина
Шевченко','(098)4569111','shevchenkooooo345@gmail.com',DATEADD(DAY, -65,
GETDATE()),DATEADD(DAY, -65, GETDATE())),
('Анатолий Дмитров','(068)2229325','dimitrovvvAnrt@gmai.com',DATEADD(DAY, -45,
GETDATE()),DATEADD(DAY, -45, GETDATE())),
('Иван Кобзар','(063)1119311','kobzartarasivan1898@gmail.com',DATEADD(DAY, -45,
GETDATE()),DATEADD(DAY, -45, GETDATE())),
('Виктор Грачь','(068)4569344','grachhhhvic735@gmail.com',DATEADD(DAY, -35,
GETDATE()),DATEADD(DAY, -35, GETDATE())),
('Ольга Буткова','(050)4569255','olgabut23ols@gmail.com',DATEADD(DAY, -25,
GETDATE()),DATEADD(DAY, -25, GETDATE())),
('Алина Мелова','(050)4539333','alina548Mel@ukr.net',DATEADD(DAY, -15,
GETDATE()),DATEADD(DAY, -15, GETDATE())),
('Михаил Савицкий','(063)9999380','mihail2Savin@gmail.com',DATEADD(DAY, -5,
GETDATE()),DATEADD(DAY, -5, GETDATE())),
('Артем Крава','(067)9995558','artemcdhak@yandex.ru',DATEADD(DAY, -15,
GETDATE()),DATEADD(DAY, -15, GETDATE()));

INSERT product_category
(slug,dt_created,dt_updated)
VALUES
('telephone-smartfon',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -15,
GETDATE())),
('computer',DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -5, GETDATE())),
('for-the-kitchen',DATEADD(DAY, -45, GETDATE()),DATEADD(DAY, -20,
GETDATE())),
```

('home-appliances',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -15, GETDATE())),
('tv-video',DATEADD(DAY, -32, GETDATE()),DATEADD(DAY, -32, GETDATE())),
('gadzhety',DATEADD(DAY, -14, GETDATE()),DATEADD(DAY, -3, GETDATE())),
('tablet-and-ebooks',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -15, GETDATE())),
('photo-and-video',DATEADD(DAY, -18, GETDATE()),DATEADD(DAY, -10, GETDATE())),
('beauty-and-health',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -15, GETDATE()));
INSERT category_content
(product_category_id,[name],[language],[description],dt_created,dt_updated)
VALUES
(1,'Телефони,смартфони','Українська','Телефони, смартфони та аксесуари до них',DATEADD(DAY, -18, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(2,'Ноутбуки та комп'ютери','Українська','Ноутбуки та настільні комп'ютери',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(3,'Техніка для кухні','Українська','Кухонні електорнні прилади',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(4,'Техніка для дому','Українська','Техніка яка використовується в побуті',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(5,'Телевізори та мультимедіа','Українська','Телевізори та пристрої які можуть відтворювати картинку',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(6,'Гаджети','Українська','Смарт-годинники та гаджети',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(7,'Планшети та книги','Українська','Електронні книги та планшети',DATEADD(DAY, -37, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(8,'Фото та відео','Українська','Фото та відео пристрої',DATEADD(DAY, -17, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(9,'Краса та здоров'я','Українська','Електронні прилади для краси',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -10, GETDATE()));

INSERT product_sub_category
(product_category_id,slug,dt_created,dt_updated)
VALUES
(1,'Telephone',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -10, GETDATE())),

(1,'Smartphone',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(1,'Accessories',DATEADD(DAY, -40, GETDATE()),DATEADD(DAY, -18, GETDATE())),
(2,'Notebook',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -19, GETDATE())),
(2,'Computer',DATEADD(DAY, -34, GETDATE()),DATEADD(DAY, -18, GETDATE())),
(3,'Great technique',DATEADD(DAY, -32, GETDATE()),DATEADD(DAY, -17, GETDATE())),
(3,'Small technique',DATEADD(DAY, -36, GETDATE()),DATEADD(DAY, -16, GETDATE())),
(4,'Built-in appliances',DATEADD(DAY, -36, GETDATE()),DATEADD(DAY, -16, GETDATE())),
(5,'TV',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(6,'Smart watches',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -11, GETDATE())),
(6,'Quadcopters',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -18, GETDATE())),
(7,'Tablets',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(7,'E-books',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -13, GETDATE())),
(8,'Cameras',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -14, GETDATE())),
(8,'Video camera',DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(9,'Hair dryer',DATEADD(DAY, -10, GETDATE()),DATEADD(DAY, -14, GETDATE())),
(9,'Curling iron',DATEADD(DAY, -10, GETDATE()),DATEADD(DAY, -5, GETDATE()));

INSERT product_sub_category_content
(product_sub_category_id,[name],[language],[description],dt_created,dt_updated)
VALUES

(1,'Телефон','Українська','Автономний мобільний телефон, призначений для роботи в мережах стільникового зв'язку',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(2,'Смартфон','Українська','Підкатегорія стільникових телефонів, які поєднують функції надання стільникового зв'язку з виконанням широкого спектру додаткових функцій і можливостей',
DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -25, GETDATE())),
(3,'Аксесуари','Українська','Пристрої які працюють разом з телефоном або смартфоном',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -15, GETDATE())),

(4,'Ноутбуки','Українська','Портативний персональний комп'ютер',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(5,'Аксесуари','Українська','Техніка і аксесуари для смартфонів',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(6,'Велика техніка','Українська','Техніка для кухні великого розміру',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(7,'Маленька техніка','Українська','Техніка для кухні не великого розміру',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(8,'Вбудована техніка','Українська','Вбудована техніка для дому',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(9,'Телевізор','Українська','Телевізори різні за розміром і функціоналомр',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(10,'Смарт-годинник','Українська','Годинники які заряжаються від струму та мають безліч функцій',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(11,'Квадрокоптери','Українська','Квадрокоптери на дистанційному управлінні',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(12,'Планшети','Українська','Пленшети разних видів та марок',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(13,'Електронні книги','Українська','Електронні книги в які можна завантажувати книги'
(14,'Фотокамери','Українська','Камери з функцією фото',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(15,'Відеокамери','Українська','Знімати відео',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(16,'Фен','Українська','Прилади для сушки волосся',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
(17,'Техніка для стайлінгу волосяя','Українська','Праски, плойки та стайлери',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE()));

INSERT product
(slug,price,code,count_available,dt_created,dt_updated)
VALUES
('AppleIphone11',20000,12333,11,DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),

('SamsungS10',15000,12344,3,DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -10, GETDATE())),
('AirPods3',15000,12335,10,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('Nokia',15000,12222,5,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('Asus3310',25000,12567,4,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('DellHost',35000,12345,10,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('AppleMacBook',30000,56789,3,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('LenovoComp12',27000,45632,7,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('Holodilnik Samsung',35000,03863,4,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('PiramydaVytagka',25000,12345,2,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('BoschPGP68',15500,84703,1,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('Blendforce',8750,32324,15,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('LGTV',10450,12456,7,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('LenovoTV',11250,12746,5,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('AppleWatch',12000,09843,23,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('SamsungSmartWatch',13450,17656,10,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('Hubsan',13999,09876,3,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('SamsungTablet',12345,76589,4,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),
('PoketBook',5000,23411,2,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11, GETDATE())),

```
 ('Canon',23000,23456,0,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11,
GETDATE())),
('GoPro',15000,33335,9,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11,
GETDATE())),
('Dyson',25000,22112,1,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11,
GETDATE())),
        ('Babyliss',8000,33223,4,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -11,
GETDATE()));

INSERT product_content
(product_id,[name],[language],[description],dt_created,dt_updated)
VALUES
(2,'IPhone 11','Англійська','Сучасний смартфон',DATEADD(DAY, -19,
GETDATE()),DATEADD(DAY, -10, GETDATE())),
(3,'Samsung C10','Англійська','Сучасний смартфон',DATEADD(DAY, -12,
GETDATE()),DATEADD(DAY, -10, GETDATE())),
(4,'AirPods навушники','Українська','Бездротові навушники',DATEADD(DAY, -20,
GETDATE()),DATEADD(DAY, -10, GETDATE())),
(7,'Ноутбук Делл','Українська','Ноутбук',DATEADD(DAY, -25,
GETDATE()),DATEADD(DAY, -10, GETDATE())),
(8,'Apple MacBook','Англійська','Багатофункціональний ноутбук від компанії
Apple',DATEADD(DAY, -28, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(7,'Dell комп'ютер','Українська','Персональний комп'ютер,
стаціонарний',DATEADD(DAY, -28, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(9,'Lenovo','Англійська','Персональний комп'ютер, стаціонарний',DATEADD(DAY, -
28, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(10,'Холодильник','Українська','Зручний компактнийхолодильник від компанії
Samsung',DATEADD(DAY, -28, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(14,'Телевізор LG','Українська','Смарт-телевізор з вайфайем',DATEADD(DAY, -18,
GETDATE()),DATEADD(DAY, -5, GETDATE())),
(16,'Смарт-годинник','Українська','Багатофункціональний смарт годинник від
компанії Apple',DATEADD(DAY, -2, GETDATE()),DATEADD(DAY, -2, GETDATE())),
(19,'Samsung S10+','Англійська','Планшет червоного сірого та зеленого кольору від
компанії Samsung',DATEADD(DAY, -10, GETDATE()),DATEADD(DAY, -1,
GETDATE())),
(20,'PocketBook','Англійська','Електронна книга, білого та чорного
кольору',DATEADD(DAY, -28, GETDATE()),DATEADD(DAY, -15, GETDATE())),
```

```
 (21,'Фотоапарат Canon','Українська','Фотоапарат з 3-х кратним зумом від
компанії Canon',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -10,
GETDATE())),
(22,'Екшн-камера GoPro','Українська','Відеокамера для подоржей,
вологостійка',DATEADD(DAY, -28, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(23,'Фен Дайсон','Українська','Найпопулярніший побутовий пристрій для сушіння
волосся.',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT product_sub_category_param
 (slug,product_sub_category_id,dt_created,dt_updated)
VALUES
('Screen',1,DATEADD(DAY, -27, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Color',1,DATEADD(DAY, -11, GETDATE()),DATEADD(DAY, -1, GETDATE())),
('Memory',1,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -19, GETDATE())),
('Screen',2,DATEADD(DAY, -24, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Color',2,DATEADD(DAY, -26, GETDATE()),DATEADD(DAY, -3, GETDATE())),
('Memory',2,DATEADD(DAY, -11, GETDATE()),DATEADD(DAY, -8, GETDATE())),
('Color',3,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('RAM',4,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Core',5,DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -5, GETDATE())),
('Memory',5,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Parameters',6,DATEADD(DAY, -33, GETDATE()),DATEADD(DAY, -22, GETDATE())),
('Diagonal',6,DATEADD(DAY, -23, GETDATE()),DATEADD(DAY, -17, GETDATE())),
('Function',1,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Whaterproof',10,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Color',10,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Screen',12,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Memory',12,DATEADD(DAY, -3, GETDATE()),DATEADD(DAY, -1, GETDATE())),
('Memory',13,DATEADD(DAY, -31, GETDATE()),DATEADD(DAY, -18, GETDATE())),
('Type screen',13,DATEADD(DAY, -21, GETDATE()),DATEADD(DAY, -10,
GETDATE())),
('Lens',14,DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Power',16,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -15, GETDATE())),
('Function',16,DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT product_sub_category_param_content
```

```
([name],product_sub_category_param_id,[language],dt_created,dt_updated)
VALUES
('Екран',1,'Українська',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Экран',1,'Русский',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Screen',1,'English',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Color',2,'English',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Колір',2,'Українська',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Цвет',2,'Русский',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Пам'ять',3,'Українська',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Память',3,'Русский',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Memory',3,'English',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Водонепроникність',14,'Українська',DATEADD(DAY, -25,
GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Whaterproof',14,'English',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Діагональ',12,'Українська',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Diagonal',12,'English',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
('Lens',20,'English',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
        ('Лінза',2,'Українська',DATEADD(DAY,  -25,  GETDATE()),DATEADD(DAY,  -12,
GETDATE()));

INSERT product_sub_category_param_value
(product_sub_category_param_id,product_id,[value],dt_created,dt_updated)
VALUES
(1,2,'LedHD',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
```

```
(1,3,'OLED',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(1,5,'LCD',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(4,6,'OLED',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(4,7,'IPS',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(4,8,'OLED',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(2,2,'Red',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(2,3,'Blue',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(2,4,'Black',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(2,4,'White',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(3,2,'LPDDR4',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(3,3,'12`',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(3,4,'64GB ',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(3,5,'128GB ',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(4,6,'OLED ',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(4,7,'LCD ',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(5,6,'White ',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(9,8,'SSD 8GB',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(9,7,'HDD 4GB',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
(7,7,'PacificBlue ',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
(7,8,'Grey ',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT payment_type
(slug,fee,dt_created,dt_updated)
VALUES
('Card',0.1,DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Cash',3,DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT payment_type_content
(payment_type_id,[name],[language],dt_created,dt_updated)
VALUES
(1,'Карта','Русский',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE())),
(1,'Картка','Українська',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE())),
```

```
 (1,'Card','English',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE())),
(2,'Cash','English',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE())),
(2,'Готівка','Українська',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE())),
(2,'Наличными','Русский',DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE()));

INSERT delivery_type
(slug,fee,dt_created,dt_updated)
VALUES
('Mail',2,DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35, GETDATE())),
('Courier',3,DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35, GETDATE())),
('Pick up at the office',0,DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE())),
('Delivery service',2,DATEADD(DAY, -35, GETDATE()),DATEADD(DAY, -35,
GETDATE()));

INSERT delivery_type_content
(delivery_type_id,[name],[language],dt_created,dt_updated)
VALUES
(1,'Нова пошта','Українська',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
(1,'Укр пошта','Українська',DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
(1,'Nova Poshta','English',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
(2,'Кур'єр за адресою','Українська',DATEADD(DAY, -25,
GETDATE()),DATEADD(DAY, -12, GETDATE())),
(2,'Курьер','Русский',DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -12,
GETDATE())),
(3,'Самовивіз з точки видачі','Українська',DATEADD(DAY, -25,
GETDATE()),DATEADD(DAY, -12, GETDATE())),
(3,'Self-pickup','English',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -1,
GETDATE())),
```

```
 (4,'DeliveryClub','English',DATEADD(DAY, -20, GETDATE()),DATEADD(DAY, -10,
GETDATE())),
        (3,'Служба доставки','Українська',DATEADD(DAY, -12,
GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT [status]
(slug,dt_created,dt_updated)
VALUES
('Created',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Updated',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
 ('Confirm',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Delivered',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('Closed',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT status_content
(status_id,[name],[language],[description],dt_created,dt_updated)
VALUES
(1,'Створено','Українська','Замовлення було створено клієнтом на
сайті',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(1,'Created','English','Order created by client',DATEADD(DAY, -12,
GETDATE()),DATEADD(DAY, -12, GETDATE())),
(2,'Updated','English','Order updated by the client',DATEADD(DAY, -12,
GETDATE()),DATEADD(DAY, -12, GETDATE())),
(2,'Оновлено','Українська','Замовлення було оновлено клієнтом на
сайті',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(3,'Confirmed','English','Confirm by manager',DATEADD(DAY, -12,
GETDATE()),DATEADD(DAY, -12, GETDATE())),
(3,'Підтверджено','Українська','Замовлення було підтверджене
менеджером',DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(4,'Доставлене','Українська','Замовлення було доставлено клієнту',DATEADD(DAY, -
12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
(5,'Закрито','Українська','Замовлення було виконане і закрите',DATEADD(DAY, -12,
GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT discount_code
(code_is_used,amount,is_persent,dt_created,dt_updated)
VALUES
```

```
 ('true','',5,DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('true',50,'',DATEADD(DAY, -18, GETDATE()),DATEADD(DAY, -2, GETDATE())),
('false','',50,DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE())),
('true','',25,DATEADD(DAY, -12, GETDATE()),DATEADD(DAY, -12, GETDATE()));

INSERT [order]
(payment_type_id,delivery_type_id,price,price_fee,total_price,status_id,comment,client_i
d,[address],discount_code_id,discount_amount,dt_created,dt_updated)
VALUES
 (1,1,'','','',3,'',1,'м.Київ,вул.Жилянська 107',1,'',DATEADD(DAY, -2,
GETDATE()),DATEADD(DAY, -1, GETDATE())),
(2,3,'','','',4,'',2,'м.Київ,вул.Ніжинська 29',2,'',DATEADD(DAY, -3,
GETDATE()),DATEADD(DAY, -1, GETDATE()));
select*from product;

INSERT order_product
(order_id,product_id,is_actual,dt_created,dt_updated)
VALUES
(1,2,'true',DATEADD(DAY, -2, GETDATE()),DATEADD(DAY, -2, GETDATE())),
(1,3,'true',DATEADD(DAY, -2, GETDATE()),DATEADD(DAY, -2, GETDATE())),
(1,6,'true',DATEADD(DAY, -2, GETDATE()),DATEADD(DAY, -2, GETDATE())),
(1,22,'true',DATEADD(DAY, -2, GETDATE()),DATEADD(DAY, -2, GETDATE())),
(2,14,'true',DATEADD(DAY, -3, GETDATE()),DATEADD(DAY, -1, GETDATE())),
(2,15,'true',DATEADD(DAY, -3, GETDATE()),DATEADD(DAY, -2, GETDATE())),
(2,17,'true',DATEADD(DAY, -3, GETDATE()),DATEADD(DAY, -3, GETDATE()));

INSERT supplier
([name],[address],phone,email,dt_created,dt_updated)
VALUES
('ТОВ "Електроніка"','м.Київ вул.Анни Ахматової
37в','(050)3456712','electronikasupplier@gmail.com',DATEADD(DAY, -30,
GETDATE()),DATEADD(DAY, -30, GETDATE())),
('ТОВ "AppleSupplier"','м.Київ вул.Вернадського
130','(050)3234562','applesup123@gmail.com',DATEADD(DAY, -20,
GETDATE()),DATEADD(DAY, -20, GETDATE())),
```

```
 ('ТОВ "SumsungProvider"','м.Полтава вул.Центральна
156','(063)3234562','sumsprovprofi100@gmail.com',DATEADD(DAY, -10,
GETDATE()),DATEADD(DAY, -10, GETDATE()));

INSERT supply
(supplier_id,price,dt_created,dt_updated)
VALUES
(1,150000,DATEADD(DAY, -3, GETDATE()),DATEADD(DAY, -2, GETDATE())),
(2,200000,DATEADD(DAY, -6, GETDATE()),DATEADD(DAY, -3, GETDATE())),
(3,175550,DATEADD(DAY, -7, GETDATE()),DATEADD(DAY, -3, GETDATE()));

INSERT supply_status
(slug,dt_created,dt_updated)
VALUES
('Created',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
('Updated',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
('Paid',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
('Confirmed',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
('Delivered',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
('Closed',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE()));


INSERT supply_status_content
(supply_status_id,[name],[language],dt_created,dt_updated)
VALUES
(1,'Створений','Українська',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30,
GETDATE())),
(1,'Created','English',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30,
GETDATE())),
(2,'Оновлений','Українська',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30,
GETDATE())),
(2,'Updated','English',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30,
GETDATE())),
(3,'Оплачений','Українська',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30,
GETDATE())),
(3,'Paid','English',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30,
GETDATE())),
```

```
 (4,'Підтверджений','Українська',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
(4,'Confirmed','English',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
(5,'Доставлений','Українська',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
(5,'Delivered','English',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
(6,'Закритий','Українська',DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE()));

INSERT supply_product
(supply_id,product_id,[count],price,supply_status_id,dt_created,dt_updated)
VALUES
(1,2,4,100000,5,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
(1,14,3,80000,1,DATEADD(DAY, -25, GETDATE()),DATEADD(DAY, -25, GETDATE())),
(1,21,10,150000,2,DATEADD(DAY, -15, GETDATE()),DATEADD(DAY, -15, GETDATE())),
(2,2,13,200000,4,DATEADD(DAY, -30, GETDATE()),DATEADD(DAY, -30, GETDATE())),
(2,7,1,10000,4,DATEADD(DAY, -10, GETDATE()),DATEADD(DAY, -7, GETDATE())),
      (2,8,5,100500,3,DATEADD(DAY, -10, GETDATE()),DATEADD(DAY, -8, GETDATE()));
```