МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА
ФАКУЛЬТЕТ УПРАВЛІННЯ ФІНАНСАМИ ТА БІЗНЕСУ

Кафедра цифрової економіки та бізнес-аналітики

КУРСОВА РОБОТА

з дисципліни «Проектування та адміністрування БД і СД»

на тему:

«Інформаційна система страхової компанії»

Науковий керівник:

к.ф.-м.н., доц. Депутат Б.Я
(науковии ступінь, посада, прізвище, ініціали)
_____ "____" _____ 2020 р.
(підпис)

Виконавець:

студент групи УФЕ-31с

Кріль П.А.
(прізвище, ініціали)
_____ "____" _____ 2020 р.
(підпис)

Загальна кількість балів _____ _____
(підпис, ПІП членів комісії)

Львів 2020

ЗМІСТ

ВСТУП

Ефективне управління підприємством в сучасних умовах неможливе без використання комп'ютерних технологій. Правильний вибір програмного продукту і фірми-розробника - це перший і визначальний етап автоматизації будь-якої діяльності. В даний час проблема вибору інформаційної системи (ІС) з специфічної задачі перетворюється на стандартну процедуру.

Основними факторами, які впливають на впровадження інформаційних систем, є потреби організацій та користувачів, а також наявність відповідних засобів для їх формування. Найсуттєвіше на розвиток інформаційних систем вплинули досягнення в галузі комп'ютерної техніки та телекомунікаційних мереж.

Причини, що спонукають організації впроваджувати інформаційні системи, з одного боку обумовлюються прагненням збільшити продуктивність повсякденних робіт чи усунути їх повторне проведення, а з іншого боку бажанням підвищити ефективність управління діяльністю організації за рахунок прийняття оптимальних та раціональних управлінських рішень.

Інформаційна система - це взаємозв'язана сукупність засобів, методів і персоналу, використовуваних для зберігання, обробки та видачі інформації в інтересах досягнення поставленої мети. Інформаційні системи включають в себе: технічні засоби обробки даних, програмне забезпечення і відповідний персонал.

Метою курсової робити є з’ясування змісту та ролі інформаційної системи для того чи іншого виду діяльності, у моєму випадку для страхової компанії. Для досягнення поставленої мети були поставлені наступні завдання:

— проаналізувати предметну область;

— спроектувати базу даних;

Об’єктом дослідження даної роботи є використання засобів розробки для інформаційної системи, зокрема MySQL.

РОЗДІЛ 1. АНАЛІЗ ВИМОГ

1.1 Постановка завдання

У сучасних умовах динамічно розвивається ринок комплексних інтегрованих систем автоматизації страхових підприємств і установ страхового бізнесу самого різного профілю (автомобільного страхування, майнового, фінансового, соціального і так далі). Всі ці організації можуть мати самі різні розміри з різноманітними схемами ієрархії, починаючи від малих підприємств з чисельністю в кілька десятків людей і закінчуючи великими корпораціями чисельністю в десятки тисяч співробітників. Такі системи призначені для вирішення завдань як підприємства в цілому (управління фінансовими ресурсами, управління актуальною інформацією, планування та облік, пошук клієнтів і реклама), так і рівня його підрозділів та організаційних відділів.

Головна особливість індустрії систем автоматизації підприємств та установ страхування, що характеризуються широкою номенклатурою вхідних даних з різними маршрутами обробки цих даних, полягає в концентрації складності на початкових етапах аналізу вимог і проектування специфікацій системи при відносно невисокої складності і трудомісткості наступних етапів. Фактично тут і відбувається розуміння того, що робитиме майбутня система, і яким чином вона буде працювати, щоб задовольнити пред'явлені до неї вимоги. А саме нечіткість і неповнота системних вимог, невирішені питання й помилки, допущені на етапах аналізу і проектування, породжують на наступних етапах важкі, часто нерозв'язні проблеми і, в кінцевому рахунку, приводять до неуспіху всієї роботи в цілому.

Перший кроком у розробці інформаційної системи є аналіз предметної області. На даному етапі аналізуються запити користувачів, відповідно до них визначається мета створення інформаційної системи, вибираються інформаційні об'єкти та їх характеристики, які визначають зміст проектованої бази даних.

Метою створення інформаційної системи страхової компанії є облік клієнтів страхової компанії.

Завершену інформаційну систему можуть використовувати фірми, що займаються страхуванням громадян та їх майна. З точки зору користувача, інформаційна система страхової компанії повинна зберігати всю необхідну інформацію для страхування громадян та їх майна, а так само видавати її на вимогу в зрозумілому і наочному вигляді, і повинна бути проста у використанні.

Завданням страхової компанії є укладання договорів. Компанія має різні філії по всій країні. Кожна філія характеризується назвою, адресою та телефоном. Діяльність компанії організована таким чином: в компанію звертаються різні особи з метою укладення договору про страхування. Залежно від прийнятих на страхування об'єктів і страхуються ризиків, договір укладається за певним видом страхування (наприклад, страхування автотранспорту від викрадення, страхування домашнього майна, добровільне медичне страхування). При укладанні договору фіксується дата укладення, страхова сума, вид страхування, тарифна ставка і філія, в якому укладався договір.

В даний час для проектування БД активно використовуються CASE-засоби, в основному орієнтовані на використання ERD (Entity - Relationship Diagrams, діаграми «сутність-зв'язок»). З їх допомогою визначаються важливі для предметної області об'єкти (сутності), відносини один з одним (зв'язку) та їх властивості (атрибути). Слід зазначити, що засоби проектування ERD в основному орієнтовані на реляційні бази даних (РБД), і якщо існує необхідність проектування іншої системи, скажімо об'єктно-орієнтованої, то краще обрати інші методи проектування.

У рамках заданої предметної області можна побудувати наступний список сутностей: «договір», «вид страхування», «філія», «клієнт», «страховий агент» .

РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ

2.1 Опис моделі даних

Модель даних – це засіб для визначення логічного зображення фізичних даних, що відносяться до деякого додатку. В процесі розробки засад створення баз даних вибір моделі в основному залежить від об'єкту впровадження (від регіонального представництва до центрального органу управління), а також від етапу впровадження.

Модель даних включає три компоненти:

— структура даних, що складають зміст бази даних;

— допустимі операції, що виконуються на структурі даних. Вони складають основу мови даних моделі, що розглядається, і дають можливість працювати з цією структурою за допомогою різних операцій мови опису даних (МОД) та мови маніпулювання даними (ММД);

— обмеження для контролю цілісності. Модель даних повинна бути забезпечена засобами, що дозволяють зберігати її цілісність і захищати її.

Термін “база даних” говорить про те, що йдеться про дані, тобто про інформацію, яка характеризує певний об’єкт, та, що ці дані є базовими, основними. З погляду користувача, який експлуатує базу даних, вона є моделлю предметної області програмного забезпечення, об’єкта, наприклад, підприємства або його частини, підрозділу. Найважливішою вимогою, яка ставиться до моделі, є її адекватність, тобто вірне відображення зв’язків і відношень між елементами того об’єкта, який вона описує. Як правило, кожний об’єкт з плином часу розвивається, змінюється. Тому БД, як модель об’єкта, являє собою живий організм, який перебуває в постійному русі, вона теж змінюється в часі.

В сучасних базах даних практично не існує обмежень на вигляд даних. Це можуть бути числа (облік кількості товарів, працівників, коштів), літерні або

символьні дані (назва та опис властивостей товару, прізвище працівника), календарна дата протікання події, текст великого розміру, малюнок, кіно– та відеофільм, адреса мережі Internet, математичний або логічний вираз і т. д. Тобто даними є будь-яка інформація, яка відображає стан об’єкта протягом заданого проміжка часу і яку можна зберігати в пам’яті комп’ютера.

З погляду програміста, який будує нову або веде (забезпечує супровід у ході експлуатації) готову базу даних, і в межах цього посібника ми будемо її розглядати саме з цього погляду, база даних являє собою сукупність даних та програм, призначених для їх обробки.

Обробка даних передбачає виконання таких робіт:

— ввід, занесення нових даних;
— забезпечення цілісності БД. Це такі роботи, як контроль і попередження помилок під час вводу, наприклад, температура води при нормальному тиску не може бути меншою за 0 °С. Може виникнути потреба у відновленні частини даних, втрачених через технічні неполадки та ін.;
— вивід на екран монітора, друк тощо;
— вилучення з бази уже непотрібних, зайвих даних;
— редагування, наприклад, виправлення помилок, зроблених під час вводу, або зміна даних у зв’язку зі зміною стану об’єкта;
— маніпуляція даними, їх підготовка до вигляду, вигідного для подальшого використання. Це, наприклад, сортування даних;
— вибір потрібної інформації та її представлення користувачеві у зручному для нього вигляді;
— пошук даних. Як правило, сучасні промислові БД громіздкі, тому операції пошуку є найбільш трудомісткими під час відбору даних для запитів. Ця робота виконується переважно програмним способом за заданою умовою.

Моделювання сутностей і зв'язків може використовуватися не тільки на етапі концептуального моделювання, але і на етапах розробки стратегії і аналізу, й і ставить основною метою створення точної й адекватної моделі інформаційних потреб організації.

На початку проектування баз даних, як правило, розробляється модель предметної області, для якої створюється ця БД. У ній указуються типи об’єктів, що будуть включені до бази даних, і зв’язки між ними. Для наочності таку модель можна подати у графічному вигляді.

У даній базі даних створено 5 таблиць:

— *client*- таблиця, яка містить дані про клієнта;

— *contract*– таблиця, яка містить інформації про умови укладання договору і вимоги;

— *branch*- таблиця, яка містить інформацію про філію компанії;

— *insurance_type*- таблиця, у якій вказано інформацію про наявні види страхування.

— *insurance_agent*- таблиця, яка містить дані про страхового агента.

Тип об’єкта предметної області називають сутністю. Сутностями таблиці contract розглянутої на рис. 2.1 предметної області є:

— contract_id – ідентифікаційний код договору;

— start_date- дата укладання договору;

— name- назва договору;

— client_id- зовнішній ключ;

— type_id- зовнішній ключ;

— agent_id- зовнішній ключ.

Сутності таблиці branch рис. 2.1:

— branch_id- ідентифікаційний код категорії;

— branch_name- найменування філії;

— lacation- адреса філії;

— phone- контактний номер філії;

— agent_id- зовнішній ключ.

Сутності таблиці insurance_type рис. 2.1:

— type_id- ідентифікаційний код виду страхування;

— available- наявність виду страхування

— type_name – вид страхування;

— insurance_premium- страховий внесок;

— insurance_payment- страхові виплати.

Сутності таблиці client рис. 2.1:

— client_id- ідентифікаційний код клієнта;

— firstname- ім’я клієнта;

— lastname- прізввище клієнта;

— phone- контактний номер клієнта;

— city- місце проживання клієнта;

— contract_id- зовнішній ключ.

Сутності таблиці insurance_agent рис. 2.1:

— agent_id- ідентифікаційний код страхового агента;

— agent_name- ім’я і прізвище страхового агента;

— branch_id- зовнішній ключ.

Сутності є сукупностями однотипних об’єктів. Наприклад, сутність firstname_insured_person і lastname_insured_person може складатися з об’єктів з з однаковими іменами і прізвищами. Окремі об’єкти сутності називають екземплярами сутності. Сутність має як мінімум один екземпляр.

У даній курсовій роботі для всіх сутностей визначено первинні ключі, які ми можемо розглянути на рис. 2.1

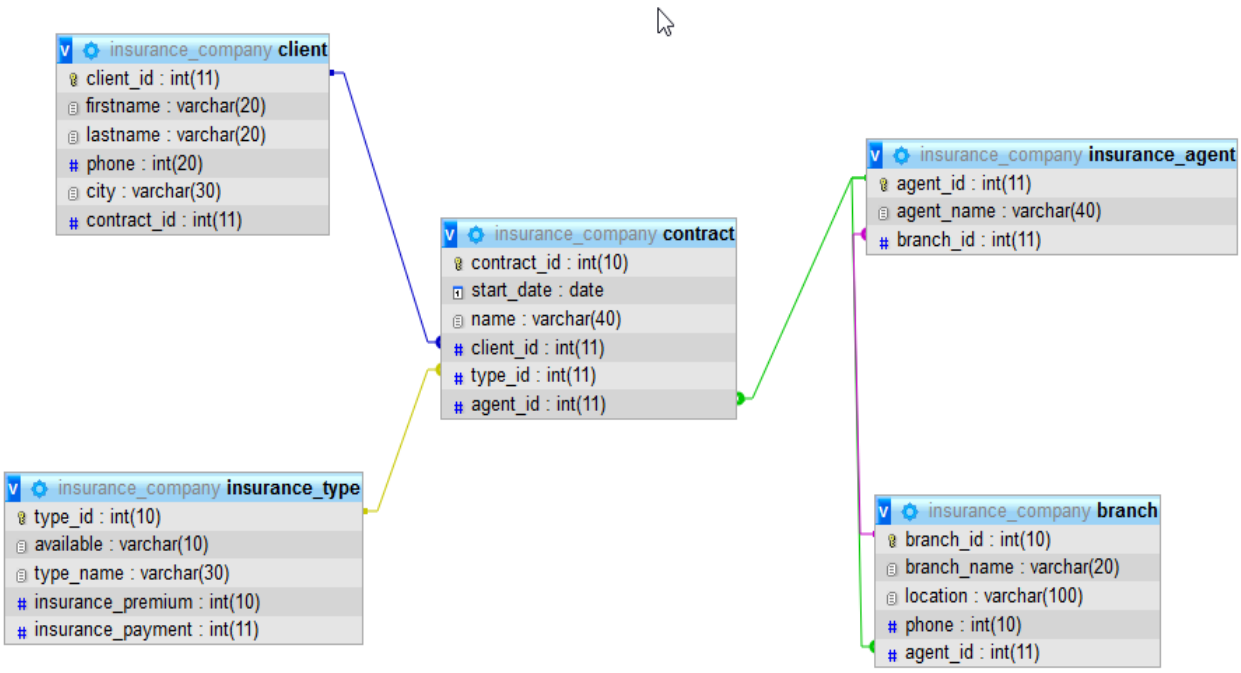

Рис. 2.1- Діаграма сутностей і зв'язків

2.2 Нормалізація реляційних відношень

Нормалізація відношень- покроковий процес розділення (декомпозиції) початкових відношень БД на простіші. Кроки цього процесу переводять схему відношення БД в послідовні нормальні форми. Кожна наступна форма володіє кращими властивостями ніж попередня. Кожній нормальній формі відповідає певний набір обмежень. При переведенні структури відношення у форми вищого порядку досягають видалення з таблиць надмірної описової інформації.Процес нормалізації заснований на понятті функціональної залежності атрибутів.

Принципи нормалізації

— в кожній таблиці БД не повинно бути повторюваних полів;

— в кожній таблиці повинен бути унікальний ідентифікатор (первинний ключ);

— кожному значенню первинного ключа повинна відповідати достатня інформація про тип суті або про об’єкт таблиці (наприклад, інформація про успішність, про групу або студентах);

— зміна значень в полях таблиці не повинна впливати на інформацію в інших полях (крім змін у полях ключа).

У теорії реляційних баз даних прийнято виділяти таку послідовність нормальних форм:

- перша нормальна форма, 1НФ;
- друга нормальна форма, 2НФ;
- третя нормальна форма, 3НФ;
- нормальна форма Бойса – Кодда, БКНФ*;*
- четверта нормальна форма, 4НФ;
- п'ята нормальна форма, 5НФ.

Перша нормальна форма (1НФ) передбачає, що зберігаються дані на перетині рядків і стовпців повинні представляти скалярний значення, а таблиці не повинні містити повторюваних рядків.

Друга нормальна форма (2НФ) передбачає, що кожен стовпець, який не є ключем, повинен залежати від первинного ключа.

Третя нормальна форма (3НФ) передбачає, що кожен стовпець, який не є ключем, повинен залежати тільки від первинного ключа.

Нормальна форма Бойса – Кодда, БКНФ - нормальна форма використана в нормалізації баз даних. Це сильніша версія третьої нормальної форми (3НФ). Таблиця знаходиться в БКНФ тоді і тільки тоді, коли для кожної її нетривіальної функціональної залежності X → Y, X це суперключ — тобто, X або потенційний ключ, або його надмножина.

Четверта нормальна форма (4НФ) застосовується для усунення багатозначних залежностей (multivalued dependencies) - таких залежностей, де стовпчик з первинним ключем має зв'язок один-до-багатьох зі стовпцем, який не є ключем. Ця нормальна форма усуває некоректні відносини багато-до-багатьох.

П'ята нормальна форма (5НФ) розділяє таблиці на менші таблиці для усунення надмірності даних. Розбиття йде до тих пір, поки не можна буде відтворити оригінальну таблицю шляхом об'єднання малих таблиць.

Кожній нормальній формі відповідає певний набір обмежень. При переведенні структури відношення у форми вищого порядку досягають видалення з таблиць надмірної описової інформації.

Процес нормалізації заснований на понятті функціональної залежності атрибутів.

Атрибут B функціонально залежить від атрибуту A (позначають A → B), якщо у будь-який момент часу кожному значенню атрибуту A відповідає не більше одного значення атрибуту B.

Якщо описовий атрибут залежить від усього складеного ключа і не залежить від його частин, то мова йде про повну функціональну залежність атрибуту від складеного ключа.

2.3 Визначення типів даних

При визначенні стовпців таблиці для них необхідно вказати тип даних. Кожен стовпець повинен мати тип даних. Тип даних визначає, які значення можуть зберігатися в стовпці, скільки вони будуть займати місця в пам'яті.

MySQL містить велику кількість типипів даних які можна розподілити на ряд груп: символьні типи(CHAR, VARCHAR, TEXT), числові типи(INT, FLOAT, DOUBLE), типи для роботи з датою і часом(DATE, TIME, YEAR), складові типи(ENUM, SET), бінарні типи(TINYBLOB, BLOB).

Найчастіше в базі даних використовуються такі типи даних:

- INT - представляє цілі числа від -2147483648 до 2147483647, займає 4 байта:
- FLOAT - зберігає дробові числа з плаваючою точкою одинарної точності від $-3.4028 * 10^{38}$ до $3.4028 * 10^{38}$, займає 4 байта:

— TEXT – преставляє текст довжиною до 65КБ;

— VARCHAR - представляє рядок змінної довжини. Довжина збереженої рядки також указиватся в дужках, наприклад, VARCHAR(10). Однак на відміну від CHAR збережена рядок буде займати саме стільки місця, як необхідна. Наприклад, якщо визначення довжини в 10 символів, але в стовпець зберігається рядок в 6 символів, то збережена рядок так і буде займати 6 символів плюс додатковий байт, який зберігає довжину рядка.

— DATE - зберігає дати з 1 січня 1000 року до 31 грудня 9999 року (с "1000-01-01" до "9999-12-31"). За замовчуванням для зберігання використовується формат РРРР-ММ-ДД. Займає 3 байта.

У даній роботі було при створенні таблиць було використано такі типи даних: INT, VARCHAR, DATE табл. 2.1, табл. 2.2, табл. 2.3, табл. 2.4, табл. 2.5

Таблиця 2.1 – Склад таблиці “branch”

Атрибут	Тип даних	Розмір	Додаткові обмеження цілісності
branch_id	INT	10	первинний ключ
branch_name	VARCHAR	20	обов’язковий
location	VARCHAR	100	обов’язковий
phone	INT	10	обов’язковий
agent_id	int	11	зовнішній ключ

Таблиця 2.2 – Склад таблиці “contract”

Атрибут	Тип даних	Розмір	Додаткові обмеження цілісності
contract_id	INT	10	первинний ключ
start_date	DATE	-	обов’язковий
name	VARCHAR	40	обов’язковий
client_id	INT	11	зовнішній ключ
type_id	INT	11	зовнішній ключ
agent_id	INT	11	зовнішній ключ

Таблиця 2.3 – Склад таблиці “insurance_type ”

Атрибут	Тип даних	Розмір	Додаткові обмеження цілісності
type_id	INT	10	первинний ключ
available	VARCHAR	10	обов’язковий
type_name	VARCHAR	30	обов’язковий
insurance_premium	INT	10	обов’язковий
insurance_payment	INT	11	обов’язковий

Таблиця 2.4 – Склад таблиці “client ”

Атрибут	Тип даних	Розмір	Додаткові обмеження цілісності
client_id	INT	11	первинний ключ
firstname	VARCHAR	20	обов’язковий
lastname	VARCHAR	20	обов’язковий
phone	INT	20	обов’язковий
city	VARCHAR	30	обов’язковий
contract_id	INT	11	зовнішній ключ

Таблиця 2.5 – Склад таблиці “insurance_agent ”

Атрибут	Тип даних	Розмір	Додаткові обмеження цілісності
agent_id	INT	11	первинний ключ
agent_name	VARCHAR	40	обов’язковий
branch_id	INT	11	зовнішній ключ

2.4 Обмеження цілісності даних

При створенні баз даних велика увага має бути приділена засобам підтримки даних в цілісному стані. Розглянемо передбачені стандартом мови SQL функції, які призначені для підтримки цілісності даних. Ця підтримка включає засоби завдання обмежень, вони вводяться з метою захисту бази даних від порушення узгодженості що зберігаються в ній даних. До таких типів підтримки цілісності даних відносяться:

— обов'язкові дані;

— обмеження для доменів полів;

— цілісність сутностей;

— посилальна цілісність ;

— вимоги конкретного підприємства.

Велика частина перерахованих обмежень задається в операторах CREATE TABLE і ALTER TABLE.

Створення таблиці. У стандарті SQL дано декілька варіантів визначення оператора створення таблиці, проте його базовий формат має наступний вигляд:

<визначення_таблиці> ::=

CREATE TABLE ім'я_таблиці

{(ім'я_стовпця тип_даних [NOT NULL][UNIQUE]

[DEFAULT <значення>]

[CHECK (<умова_вибору>)][,..n]}

[CONSTRAINT ім'я_обмеження]

[PRIMARY KEY (ім'я_стовпця [,..n])

{[UNIQUE (ім'я_стовпця [,..n])}

[FOREIGN KEY (ім'я_стовпця_зовнішнього_ключа

[,..n])

REFERENCES ім'я_рід_таблиці

[(ім'я_стовпця_рід_таблиці [,..n])],

[MATCH {PARTIAL | FULL}]

[ON UPDATE {CASCADE| SET NULL |SET DEFAULT

|NO ACTION}]

[ON DELETE {CASCADE| SET NULL |SET DEFAULT

|NO ACTION}]

{[CHECK(<умова_вибору>)][,..n]})

Обмеження. Представлена версія оператора створення таблиці включає засоби визначення вимог цілісності даних, а також інші конструкції. Є дуже великі варіації в наборі функціональних можливостей цього оператора,

реалізованих в різних діалектах мови SQL. Розглянемо призначення параметрів команди, використовуваних для підтримки цілісності даних.

Обов'язкові дані. Для деяких стовпців потрібно наявність в кожному рядку таблиці конкретного і допустимого значення, відмінного від опущеного значення або значення NULL. Для завдань обмежень подібного типу стандарт SQL передбачає використання специфікації NOT NULL.

Обмеження для доменів полів. Кожен стовпець має власний домен - деякий набір допустимих значень. Стандарт SQL передбачає два різні механізми визначення доменів. Перший полягає у використанні пропозиції CHECK, що дозволяє задати необхідні обмеження для стовпця або таблиці в цілому, а другою припускає застосування оператора CREATE DOMAIN.

Цілісність сутностей. Первинний ключ таблиці повинен мати унікальне не порожнє значення в кожному рядку. Стандарт SQL дозволяє задавати подібні вимоги підтримки цілісності даних за допомогою фрази PRIMARY KEY. В межах таблиці вона може вказуватися тільки один раз. Проте існує можливість гарантувати унікальність значень і для будь-яких альтернативних ключів таблиці, що забезпечує ключове слово UNIQUE. Крім того, при визначенні альтернативних ключів рекомендується використовувати і специфікатори NOT NULL.

Посилальна цілісність. Зовнішні ключі є стовпцями або наборами стовпців, призначеними для зв'язування кожної з рядків дочірньої таблиці, що містить цей зовнішній ключ, з рядком батьківської таблиці, що містить відповідне значення потенційного ключа. Стандарт SQL передбачає механізм визначення зовнішніх ключів за допомогою пропозиції FOREIGN KEY, а фраза REFERENCES визначає ім'я батьківської таблиці, тобто таблиці, де знаходиться відповідний потенційний ключ. При використанні цієї пропозиції система відхилить виконання будь-яких операторів INSERT або UPDATE, за допомогою яких буде зроблена спроба створити в дочірній таблиці значення зовнішнього ключа, що не відповідає одному із вже існуючих значень потенційного ключа батьківської таблиці. Коли дії системи виконуються при вступі операторів

UPDATE і DELETE, що містять спробу відновити або видалити значення потенційного ключа у батьківській таблиці, якому відповідає одна або більше за рядки дочірньої таблиці, то вони залежать від правил підтримки посилальної цілісності, вказаних у фразах ON UPDATE і ON DELETE пропозиції FOREIGN KEY. Якщо користувач робить спробу видалити з батьківської таблиці рядок, на який посилається одна або більше за рядки дочірньої таблиці, мова SQL надає наступні можливості:

• CASCADE - виконується видалення рядка з батьківської таблиці, що супроводжується автоматичним видаленням усіх рядків дочірньої таблиці, що посилаються на неї;

• SET NULL - виконується видалення рядка з батьківської таблиці, а в зовнішні ключі усіх рядків дочірньої таблиці, що посилаються на неї, записується значення NULL ;

• SET DEFAULT - виконується видалення рядка з батьківської таблиці, а в зовнішні ключі усіх рядків дочірньої таблиці, що посилаються на неї, заноситься значення, що приймається за умовчанням;

• NO ACTION - операція видалення рядка з батьківської таблиці відміняється. Саме це значення використовується за умовчанням в тих випадках, коли в описі зовнішнього ключа фраза ON DELETE опущена.

Ті ж самі правила застосовуються в мові SQL і тоді, коли значення потенційного ключа батьківської таблиці оновлюється.

Визначник MATCH дозволяє уточнити спосіб обробки значення NULL в зовнішньому ключі.

При визначенні таблиці пропозиція FOREIGN KEY може вказуватися довільна кількість разів.

У операторові CREATE TABLE використовується необов'язкова фраза DEFAULT, яка призначена для задання по замовчуванні значення, коли в операторі INSERT значення в цьому стовпці буде відсутнє.

Фраза CONSTRAINT дозволяє задати ім'я обмеженню, що дозволить згодом відмінити те або інше обмеження за допомогою оператора ALTER TABLE.

Зміна і видалення таблиці. Для внесення змін до вже створених таблиць стандартом SQL передбачений оператор ALTER TABLE, призначений для виконання наступних дій :

• додавання в таблицю нового стовпця;

• видалення стовпця з таблиці;

• додавання у визначення таблиці нового обмеження;

• видалення з визначення таблиці існуючого обмеження;

• завдання для стовпця значення за умовчанням;

• відміна для стовпця значення за умовчанням.

Оператор зміни таблиці має наступний узагальнений формат:

<зміна_таблиці> ::=

ALTER TABLE ім'я_таблиці

[ADD [COLUMN]ім'я_стовпця тип_даних

[NOT NULL][UNIQUE]

[DEFAULT <значення>][CHECK (<умова_вибору>)]]

[DROP [COLUMN] ім'я_стовпця [RESTRICT | CASCADE]]

[ADD [CONSTRAINT [ім'я_обмеження]]

[{PRIMARY KEY (ім'я_стовпця [,..n])

|[UNIQUE (ім'я_стовпця [,..n])}

|[FOREIGN KEY (ім'я_стовпця_зовнішнього_ключа [,..n])

REFERENCES ім'я_рід_таблиці

[(ім'я_стовпця_рід_таблиці [,..n])],

[MATCH {PARTIAL | FULL}

[ON UPDATE {CASCADE| SET NULL |

SET DEFAULT | NO ACTION}]

[ON DELETE {CASCADE| SET NULL |

SET DEFAULT | NO ACTION}]

|[CHECK(<умова_вибору>)][,..n]}]

[DROP CONSTRAINT ім'я_обмеження

[RESTRICT | CASCADE]]

[ALTER [COLUMN] SET DEFAULT <значення>]

[ALTER [COLUMN] DROP DEFAULT]

Тут параметри мають те ж саме призначення, що і у визначенні оператора CREATE TABLE.

Оператор ALTER TABLE реалізований не в усіх діалектах мови SQL. У деяких діалектах він підтримується, проте не дозволяє видаляти з таблиці вже існуючі стовпці.

Для видалення таблиці використовується команда DROP TABLE.

2.5 Реалізація SQL-скрипту

У базі даних реалізовуються такі запити:

— SELECT type_name FROM insurance_type WHERE available = 'yes' – використовується для перевірки наявності виду страхування в компанії;

— SELECT location, phone FROM branch WHERE branch_name LIKE '%Lviv Brok%' – використовується для отримання адреси і контактного номеру одної з філій в країні;

— SELECT validity_period FROM contract GROUP BY validity_period – використовується для групування строку дії договорів;

— DELETE FROM `branch` WHERE `branch`.`branch_id` = 10- використовується для видалення рядка з id, яке дорівнює 10;

— ALTER TABLE `contract` DROP `firstname`, DROP `lastname`, DROP `lastname_insured_person`- використовується для видалення всієї сутності ім’я і прізвище;

— UPDATE insurance_type SET insurance_premium = insurance_premium * 1.2 WHERE type_id = 158- використовується для збільшення страхового внеску в 1.2 раза;

— UPDATE client SET city = "Kharkiv" WHERE type_id = 1212- використвоються для редагування місця проживання клієнта з id-1212:

— UPDATE insurance_type
SET insurance_premium = CASE
WHEN insurance_premium < 400
THEN insurance_premium * 1.3
ELSE insurance_premium * 1.1
END- збільшує суму страхових внесків в 1.3 раза, які коштують менше 400 і в 1.1 раза, які коштують більше 400.

ВИСНОВКИ

Після написання курсової роботи можна зробити висновок, що інформаційна система повинна виконувати збір та відбір в базі даних, а так само стежити за тим, щоб введені дані були точні і правильно оформлені.

У даному випадку була використана, за основу засобів розробки проекту, програмний продукт MySQL.

У ході роботи були виконані наступні основні етапи проектування інформаційної системи для страхових компаній: планування, збір необхідних даних, Перед проектування вмісту в різних структурованих образах, створення самого проекту засобами MySQL.

Створена інформаційна система страхової компанії може вільно застосовуватися виключно в навчальних цілях. Інформаційна система дозволяє відстежувати інформацію стосовно філій компанії, укладання договорів та наявних видів страхування.

Мета курсової роботи досягнута - створена інформаційна система страхової компанії.

Використання інформаційної системи також могла б дозволити підвищити якість обслуговування клієнтів, що дасть страхової компанії переваги перед конкурентами на ринку страхових послуг. А так само використання прикладного програмного продукту значно скоротить обсяг ручного рутинної праці при підготовці адміністраторами звітних документів.

Автоматизація інформаційної системи, на сьогоднішній день, стали не просто засобом оптимізації внутрішніх процесів організації, а нагальною необхідністю в умовах жорсткої конкуренції. Саме автоматизація системи дає нові можливості будь-якої організації щодо прискорення роботи, дозволяє випередити конкурентів при прийнятті як оперативних, так і стратегічних рішень.

Одним із найкращих методів побудови діаграм для дослідження зв’язків і способів використання даних, подій є мова ER-моделювання (від англ. (Entity-

relationship model або Entity-relationship diagram)- це мова визначення інформаційних потреб організації. Мова базується на концепції, відповідно до якої інформаційне забезпечення будь-якої предметної області представляється як сукупність взаємозалежних об'єктів. Процес моделювання полягає у виділенні сутностей програмного забезпечення, установлення властивостей виділених сутностей і виявлення існуючих між ними зв’язків.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Гайдаржи В., Ізварін І. Бази даних в інформаційних системах: - Університет "Україна", 2018.- 418.

2. StudFiles[Електронний ресурс]. Режим доступу- https://studfile.net/preview/5129956/page:15/

3. Лоусон,Б. Вивчаємо MySQl [Текст]/ Б. Лоусон.:- Бібліотека спеціаліста, 2005.– 272 с.

4. О.Н. Романюк. Організація баз даних і знань. – Вінниця: ВДТУ, 2001. – 103с.

5. OpenNET[Електронний ресурс]. Режим доступу- https://www.opennet.ru/docs/RUS/mysqlcli/glava01.html

6. Грабер М. SQL, опис SQL 92, SQL99 і SQL j, 2016.- 643.

7. Bookwu.net [Електронний ресурс]. Режим доступу- http://bookwu.net/book_organizaciya-baz-danih-i-znan_997/13_1.2.4-proces-proektuvannya-bd

8. Інформаційна система [Електронний ресурс]. Режим доступу- https://uk.wikipedia.org/wiki/Інформаційна_система.

9. DO.NTUDP [Електронний ресурс]. Режим доступу- https://do.nmu.org.ua/course/info.php?id=1015