

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
(повна назва інституту/факультету)

КАФЕДРА інформатики та програмної інженерії
(повна назва кафедри)

КУРСОВА РОБОТА

з дисципліни «Бази даних»
(назва дисципліни)

на тему: База даних для підтримки діяльності call-центру

Студента (ки) 2 курсу ІП-24 групи
спеціальності 121 «Інженерія програмного
забезпечення»

Піддубного Б.С.
(прізвище та ініціали)

Керівник Ліщук Катерина Ігорівна, доц., к.т.н.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____ Оцінка ECTS _____

Члени комісії

_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)
_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)
_____	_____
(підпис)	(вчене звання, науковий ступінь, прізвище та ініціали)

Київ – 2024 рік

**Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет Інформатики та обчислювальної техніки
(повна назва)

Кафедра Інформатики та програмної інженерії
(повна назва)

Дисципліна Бази даних

Курс 2 Група ІІІ-24 Семестр 3

**З А В Д А Н Н Я
НА КУРСОВУ РОБОТУ СТУДЕНТУ**

Піддубному Борису Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи База даних для підтримки діяльності call-центру

керівник роботи Ліщук Катерина Ігорівна, к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 29.12.2023

3. Вихідні дані до роботи завдання на розробку бази даних для підтримки діяльності call-центру

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

- 1) Аналіз предметного середовища
 - 2) Побудова ER-моделі
 - 3) Побудова реляційної схеми з ER-моделі
 - 4) Створення бази даних, у форматі обраної системи управління базою даних
 - 5) Створення користувачів бази даних
 - 6) Імпорт даних з використанням засобів СУБД в створену базу даних
 - 7) Створення мовою SQL запитів
 - 8) Оптимізація роботи запитів
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Дата видачі завдання 04.11.2023

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання курсового проекту	Строк виконання етапів проекту	Примітка
1	Аналіз предметного середовища	12.11.2023	
2	Побудова ER-моделі	12.12.2023	
3	Побудова реляційної схеми з ER-моделі	13.12.2023	
4	Створення бази даних, у форматі обраної системи управління базою даних	17.12.2023	
5	Створення користувачів бази даних	19.12.2023	
6	Імпорт даних з використанням засобів СУБД в створену базу даних	20.12.2023	
7	Створення мовою SQL запитів	24.12.2023	
8	Оптимізація роботи запитів	07.01.2023	
9	Оформлення пояснювальної записки	20.01.2023	
10	Захист курсової роботи	24.01.2023	

Студент

(підпис) Піддубний Б.С
(прізвище та ініціали)

Керівник роботи

(підпис) Ліщук К.І
(прізвище та ініціали)

АНОТАЦІЯ

Пояснювальна записка до курсової роботи: 67 сторінки, 52 рисунків, 12 таблиць, 7 посилань.

Об'єкт дослідження: база даних для підтримки діяльності call-центру.

Мета роботи: розробка та реалізація бази даних для підтримки діяльності call-центру.

Проведено аналіз предметного середовища, побудована ER-модель та реляційна схема, створена база даних у форматі MySQL. Створені користувачі та ролі. Були імпортовані дані в створену базу даних. Створені функції, процедури, тригери, представлення та SQL запити для роботи з базою даних. Також була проведена оптимізація роботи запитів.

Виконана реалізація бази даних для підтримки діяльності call-центру у форматі MySQL згідно з варіантом завдання.

ЗМІСТ

ВСТУП	5
1 АНАЛІЗ ПРЕДМЕТНОГО СЕРЕДОВИЩА	6
1.1 Опис предметного середовища	6
1.2 Аналіз існуючих програмних продуктів	6
2 ПОСТАНОВКА ЗАВДАННЯ	9
3 ПОБУДОВА ER-МОДЕЛІ	10
3.1 Бізнес-правила	10
3.2 Створення сутностей	10
3.3 Опис сутностей	11
4 РЕАЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ	15
5 РЕАЛІЗАЦІЯ БАЗИ ДАНИХ	20
5.1 Створення бази даних.....	20
5.2 Імпортування даних.....	23
6 СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ	28
6.1 Адміністратор call-центру.....	28
6.2 Наглядач call-центру.....	28
6.3 Агент call-центру	28
6.4 Користувач call-центру.....	29
7 РОБОТА З БАЗОЮ ДАНИХ	30
7.1 Тексти генераторів.....	30
7.2 Тексти збережених процедур/функцій.	31
7.3 Тригери	42
7.4 Представлення.....	48
7.5 SQL-запити	51
7.6 Індeksi та результати оптимізації.....	62
ВИСНОВКИ	66
СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ	67

ВСТУП

У сучасному світі, важливість ефективного управління та обробки даних надзвичайно висока. Однією з ключових складових цього процесу є використання баз даних, які забезпечують систематизоване зберігання та зручний доступ до інформації.

У даній курсовій роботі розробляється база даних для підтримки діяльності call-центру, на прикладі компанії з надання технічної підтримки та IT послуг, з використанням системи управління базами даних MySQL[1]. Діяльність call-центрів на пряму пов'язана із взаємодією з величезними обсягами даних про клієнтів, статистикою, журналами викликів тощо. Призначенням розробки є поліпшення ефективності роботи call-центру, скорочення часових затрат на обробку інформації, а також забезпечення надійності та цілісності даних. Сфера використання розробки охоплює всі аспекти, що пов'язані із обслуговування клієнтів, наданням послуг та управління великим обсягом інформації.

Вибір MySQL для реалізації бази даних обґрунтований його широким застосуванням, надійністю та ефективністю. Також важливим аспектом у виборі MySQL відіграє його відкритість та вільна ліцензія, що забезпечує доступність для різних користувачів та розробників.

Отже, дана курсова робота націлена на створення бази даних для оптимізації діяльності call-центру, яка відповідатиме сучасним вимогам та сприятиме підвищенню ефективності обслуговування клієнтів.

1 АНАЛІЗ ПРЕДМЕТНОГО СЕРЕДОВИЩА

1.1 Опис предметного середовища

База даних call-центру, повинна включати в себе велику кількість інформації про: клієнтів, робітників, дзвінків, наявних послуг, відгуків про роботу агентів call-центру тощо.

Вхідними даними є інформація про: клієнтів (контактна інформація, історія дзвінків на гарячу лінію, відгуки), робітників call-центру (контактна інформація, зарплата, дата найму), також індекси для наглядачів та агентів, дзвінки (час початку, час закінчення, інформація про клієнта та агента call-центру), предмет дискусії (опис), послуги (назва, ціна) та відгуки (рейтинг, коментарі)

Вихідними даними виступають: звіти про продуктивність агентів від наглядачів. У звітах вказана статистика дзвінків, аналіз надання послуг, відгуки клієнтів тощо.

1.2 Аналіз існуючих програмних продуктів

Five9[2] – це сучасна хмарна платформа для call-центрів, яка надає повний спектр можливостей для ефективної комунікації з клієнтами. Five9 пропонує обробку дзвінків та чатів, електронну пошту, соціальні мережі та інші канали в одному інтерфейсі. Можна легко переключатися між різними каналами, а також використовувати функції, такі як переведення дзвінків, конференц-зв'язок, утримання дзвінків, запис голосових повідомлень, тощо

Наявна інтелектуальна маршрутизація дзвінків, автоматизація вихідних дзвінків, CRM-інтеграція (Управління відносинами з клієнтами), аналітика та звітність. Також, Five9 підтримує більше 100 мов та діалектів і використовує штучний інтелект для покращення роботи агентів та задоволення клієнтів. Наприклад, Five9 може автоматично маршрутизувати дзвінки до найбільш підходящих агентів, а також надавати їм підказки та рекомендації під час розмови. Використовується IVA (інтелектуальний віртуальний асистент).

На відміну від Five9, що є хмарною платформою, моя база даних може бути розміщена на власному сервері, що надає більший контроль над безпекою та доступом до даних.

Наступна програма – VoIPTime[3]. Вона створена для call-центрів та пропонує найкраще співвідношення ціна-якість. VoIPTime також дозволяє обробляти дзвінки, чати, електронну пошту, соціальні мережі та інші канали в одному інтерфейсі. Серед наявних функції є:

- Автоматичне нагадування
- Запис дзвінків
- Сценарії розмов (дозволяє створювати та використовувати готові сценарії для різних типів дзвінків, таких як привітання, продажі, консультації, підтримка)
- Аналітика та звітність (схожа за принципом дії на таблицю з відгуками про агентів call-центру в моїй базі даних).

Також, VoIPTime надає функції використання IVR (інтерактивна голосова відповідь) та чатботів для самообслуговування клієнтів, що суттєво зменшує навантаження на агентів та підвищує задоволення клієнтів.

В сучасному світі, де бізнес залежить від задоволення клієнтів, важливо мати ефективні та надійні інструменти для аналітики та звітності, які допомагають краще зрозуміти бізнес та взаємодію з клієнтами. Обидва продукти, Five9 та VoIPTime, надають такі інструменти, які дозволяють отримувати цінну інформацію про продуктивність, якість, задоволеність та поведінку клієнтів. Використання наявних функцій допоможе забезпечити високу якість обслуговування, зменшити витрати та відрізнитися від конкурентів.

Однак, не всі продукти та програмне забезпечення однаково підходять для різних call-центрів. У підсумку можна зазначити, що вибір конкретного

продукту та програмного забезпечення повинен ґрунтуватися на відповідних потребах call-центру, його розміру, специфіки та цілей. Повинні враховуватися питання швидкості обробки даних, можливості реструктуризації, збільшення бази даних. Також значну роль відіграє актуальність ПЗ у сучасних реаліях, де технології швидко розвиваються та впливають на очікування клієнтів. Call-центр може використовувати хмарні рішення, які дозволяють легко масштабувати, інтегрувати та оновлювати свої системи, або застосовувати штучний інтелект, який допомагає автоматизувати та оптимізувати процеси.

Тому, на підґрунті цих фактів виникає потреба створення власної бази даних для підтримки діяльності call-центру із власними встановленими правилами та можливостями. Така база даних дозволить call-центру керувати своїми даними, адаптуватися до змін, використовувати свої ресурси ефективно та забезпечити високий рівень задоволення клієнтів. Розробка та впровадження такої бази даних стане важливим кроком у напрямку оптимізації та автоматизації процесів у сфері обслуговування клієнтів, сприяючи підвищенню ефективності та якості роботи call-центрів.

2 ПОСТАНОВКА ЗАВДАННЯ

Метою даної курсової роботи є розробка та реалізація бази даних для call-центра компанії з надання технічної підтримки та ІТ послуг. Основними напрямками роботи є поліпшення ефективності роботи call-центру, скорочення часових затрат на обробку інформації, а також забезпечення надійності та цілісності даних .

Етапи розробки курсової роботи:

1. Аналіз предметного середовища
2. побудова ER-моделі
3. Побудова реляційної схеми бази даних
4. Створення бази даних з використання СУБД MySQL
5. Імпорт даних з використанням засобів СУБД
6. Автоматизація ключових бізнес-процесів за допомогою запитів мовою SQL
7. Оптимізація роботи запитів
8. Додання користувачів у систему

База даних повинна відповідати стандартам ефективного управління роботи в кол-центрі, бути стійкою до навантажень, мати можливість впровадити нові функції та/або сутності, забезпечити швидкий доступ до інформації. Дані в базі повинні зберігатися відповідно до політики конфіденційності компанії.

Серед присутніх таблиць у базі даних слід визначити наступні сутності:

- Користувач – людина, що звернулася до кол-центру з метою отримання послуг, консультації або вирішення проблеми.
- Агент – людина та працівник компанії, що проводить консультативні розмови з користувачом, пропонує послуги.
- Наглядач (супервайзер) – людина та працівник компанії, що наймає в свою команду агентів, слідкує за їхньою успішністю.

3 ПОБУДОВА ER-МОДЕЛІ

3.1 Бізнес-правила

До основних бізнес-правил належить:

1. Наглядач може не мати підпорядкованих працівників на даний момент часу
2. Кожному працівнику та користувачу відповідають унікальні контактні дані
3. Користувач зареєстрован у базі під конкретними даними, для їх зміни потрібно звертатися до кол-центру
4. Номер телефону повинен бути унікальним
5. Пошта повинна бути унікальною
6. Зарплата робітників компанії не може бути від'ємною або більше за 99999.99 доларів
7. Користувач може залишити відгук про агента та оцінити його від 1 до 5
8. Дзвінок не може тривати більше 2 годин
9. Ціна послуг не може бути від'ємною або більше за 9999.99 доларів

3.2 Створення сутностей

Для виконання поставленої задачі було створено наступні сутності:

1. Contact_info
2. Customer
3. Agent
4. Supervisor
5. Staff
6. Issue
7. Feedback
8. Call
9. Service

3.3 Опис сутностей

Після детального аналізу опису предметного можна точно виділити наступні атрибути сутностей (таблиця 3.1).

Таблиця 3.1 – Опис сутностей

Назва сутності	Атрибути	Опис
contact_info	id	Унікальний ідентифікатор контактної інформації
	name	Повне ім'я людини
	number	Номер телефона
	email	Електронна пошта
customer	id	Унікальний ідентифікатор користувача
	contact_info_id	Ідентифікатор контактної інформації
agent	id	Унікальний ідентифікатор агента
	staff_id	Ідентифікатор працівника
	supervisor_id	Ідентифікатор наглядача
supervisor	id	Унікальний ідентифікатор наглядача
	staff_id	Ідентифікатор працівника

Продовження таблиці 3.1

Назва сутності	Атрибути	Опис
staff	id	Унікальний ідентифікатор працівника
	salary	Зарплата працівника
	hire_date	Дата найму
	contact_info_id	Ідентифікатор контактної інформації

issue	id	Унікальний ідентифікатор предмету дискусії
	description	Опис
feedback	id	Унікальний ідентифікатор відгука
	rating	Оцінка роботи агента
	comments	Коментар до відгуку
	customer_id	Ідентифікатор користувача
	agent_id	Ідентифікатор агента
call	id	Унікальний ідентифікатор дзінка
	start_at	Початок дзвінка
	end_at	Кінець дзвінка
	customer_id	Ідентифікатор користувача
	agent_id	Ідентифікатор агента
service	id	Унікальний ідентифікатор послуги
	name	Назва
	price	Ціна

Опис зв'язків між сутностями:

1. contact_info – customer: один до одного (1:1). Для кожного користувача існує відповідна контактна інформація
2. contact_info – staff: один до одного (1:1). Для кожного працівника існує відповідна контактна інформація.
3. staff – agent: один до одного (1:1). Для кожного агента існує відповідний запис працівника в компанії.
4. staff – supervisor: один до одного (1:1). Для кожного наглядача існує відповідний запис працівника в компанії.
5. customer – feedback: один до багатьох (1:N). Користувач може залишити багато відгуків.

6. agent – feedback: один до багатьох (1:N). Агент може мати багато відгуків.
7. customer – call: один до багатьох (1:N). Користувач може здійснити багато дзвінків.
8. agent – call: один до багатьох (1:N). Агент може відпрацювати багато дзвінків.
9. call – service: багато до багатьох (M:N). Під час дзвінка можна замовити багато послуг. Послугу можуть замовити в багатьох дзвінках. Зв'язок виконан з використання асоціативної таблиці.
10. call – issue: багато до багатьох (M:N). Під час дзвінка можна порушити багато питань та провести багато дискусій. Предмет дискусії та порушені питання можуть обговорюватися в багатьох дзвінках.

Побудована ER-модель зображена на рис. 3.1. Кожна таблиця має свої ключові атрибути та логічні зв'язки з іншими таблицями. Для зв'язків з іншими сутностями використовуються FK (зовнішні ключі).



Рис. 3.1 – ER-модель бази даних для підтримки діяльності call-центру

4 РЕАЛЯЦІЙНА МОДЕЛЬ БАЗИ ДАНИХ

Для ефективної роботи call-центру необхідно мати базу даних, яка зберігає, обробляє і надає доступ до великої кількості інформації про клієнтів, співробітників, статистику тощо. Саме для такої цілі було обрано систему управління базами даних (СУБД) MySQL1. MySQL – Це популярна, відкрита і надійна СУБД, яка має багато переваг для цієї мети. Ось деякі з них:

- Open source[4]. MySQL є однією з найпоширеніших та високопродуктивних реляційних СУБД, а відкритий вихідний код дозволяє безкоштовно використовувати та модифікувати систему відповідно до конкретних потреб проєкту.
- Швидкість і продуктивність. MySQL володіє високою швидкістю обробки запитів і може ефективно працювати з великими обсягами даних. Це дозволяє call-центру забезпечити швидку і якісну обслуговування клієнтів, а також аналізувати і оптимізувати свою діяльність.
- Безпека і надійність. MySQL має різні рівні безпеки, які захищають дані від несанкціонованого доступу, змін або втрат. Він дозволяє встановлювати паролі, обмеження та інші заходи, які гарантують конфіденційність і цілісність даних. MySQL також підтримує резервне копіювання і відновлення даних, що знижує ризик втрати важливої інформації.

Після детального обґрунтування вибору СУБД було реалізовано та описано структури бази даних у табличному вигляді. Таблиці 4.1 – 4.11.

Таблиця 4.1 – «contact_info» містить контактну інформацію про працівників та користувачів call-центру. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор контактної інформації
name	VARCHAR	100		Повне ім'я людини
number	VARCHAR	15		Номер телефону
email	VARCHAR	100		Електронна пошта

Таблиця 4.2 – «customer» містить інформацію про користувачі call-центру. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор користувача
contact_info_id	INT		FK	Ідентифікатор контактної інформації

Таблиця 4.3 – «agent» містить інформацію про агентів call-центру. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор агента
staff_id	INT		FK	Ідентифікатор працівника
supervisor_id	INT		FK	Ідентифікатор наглядача

Таблиця 4.4 – «supervisor» містить інформацію про наглядачів call-центру. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор наглядача
staff_id	INT		FK	Ідентифікатор працівника

Таблиця 4.5 – «staff» містить інформацію про працівників call-центру. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор працівника
salary	DECIMAL	7, 2		Заробітна плата

hire_date	DATE			Дата найму на роботу
contact_info_id	INT		FK	Ідентифікатор контактної інформації

Таблиця 4.6 – «feedback» містить інформацію про відгуки користувачі.
Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор відгука
rating	ENUM			Оцінка роботи агента
comments	TEXT	1000		Коментар до відгуку
customer_id	INT		FK	Ідентифікатор користувача
agent_id	INT		FK	Ідентифікатор агента

Таблиця 4.7 – «issue» містить інформацію про теми проведених дискусій, можливі порушені питання під час дзвінків, предмети дискусії.
Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор предмету дискусії
description	VARCHAR	255		Опис

Таблиця 4.8 – «service» містить інформацію про послуги, що може надати компанії користувачеві. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор послуги
name	VARCHAR	100		Назва
price	DECIMAL	6, 2		Ціна

Таблиця 4.9 – «call» містить інформацію про дзвінки в call-центрі.
Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
id	INT		PK	Унікальний ідентифікатор дзінка
start_at	TIMESTAMP			Початок дзвінка

end_at	TIMESTAMP			Кінець дзвінка
customer_id	INT		FK	Ідентифікатор користувача
agent_id	INT		FK	Ідентифікатор агента

Таблиця 4.10 – «call_issue» асоціативна, містить інформацію про зв'язок між дзвінками та предметами дискусії в call-центрі. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
call_id	INT		FK	Ідентифікатор дзвінка
issue_id	INT		FK	Ідентифікатор предмету дискусії

Таблиця 4.11 – «call_service» асоціативна, містить інформацію про зв'язок між дзвінками та послугами в call-центрі. Структура таблиці наступна:

Ім'я поля	Тип даних	Розмір	Ключ	Опис
call_id	INT		FK	Ідентифікатор дзвінка
service_id	INT		FK	Ідентифікатор послуги

Після наведення опису структури в табличному вигляді була побудована реляційна схема за допомогою СУБД MySQL (рис. 4.1)

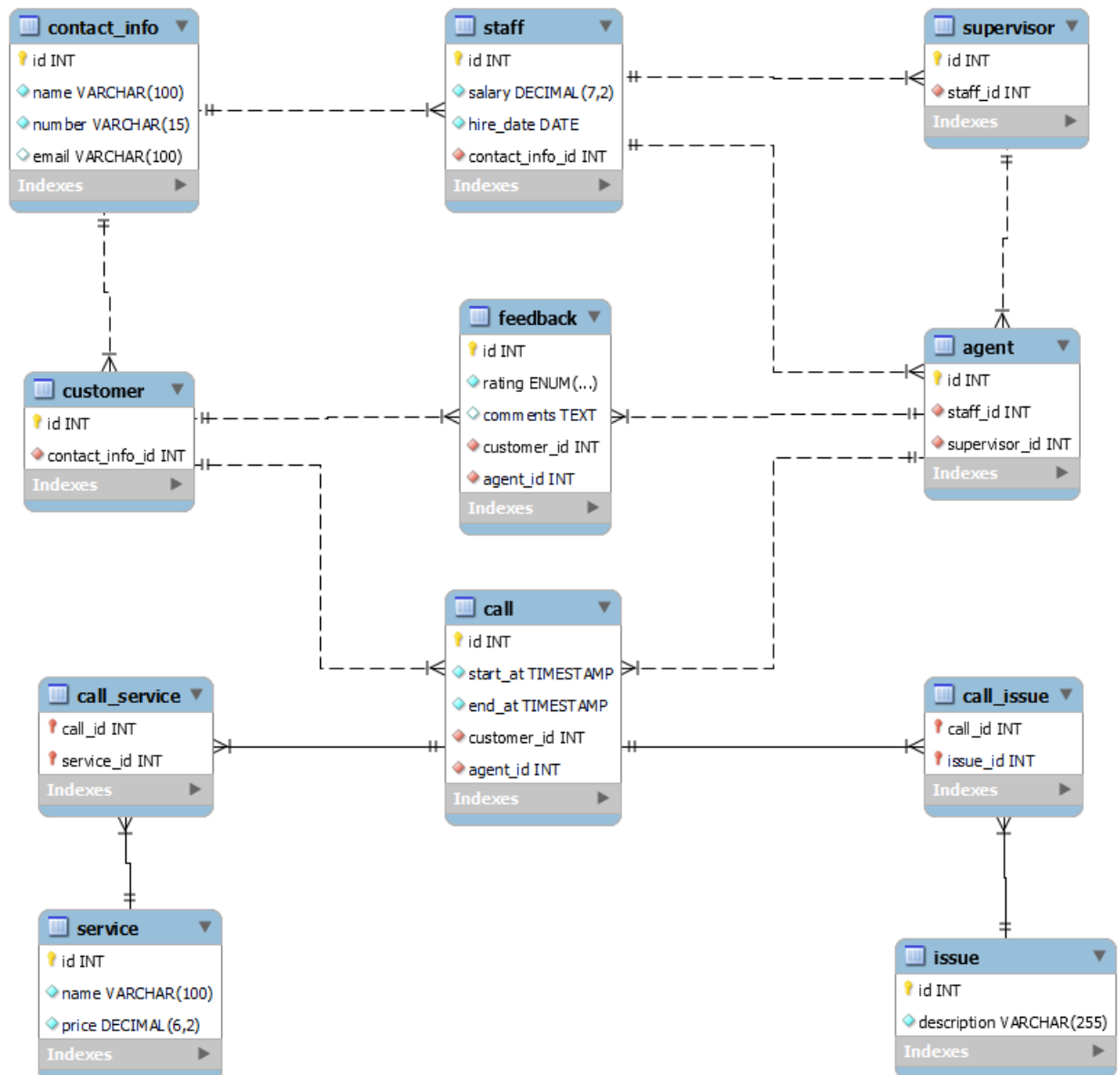


Рис. 4.1 – Реляційна схема бази даних згенерована в додатку MySQL Workbench[5]

Підводимо підсумки розділу: було створено сутності, побудовано логічні зв'язки між ними, та на їх основі розроблено ER-модель бази даних для підтримки діяльності call-центру. Також, було створено реляційну схему за допомогою MySQL Workbench.

5 РЕАЛІЗАЦІЯ БАЗИ ДАНИХ

5.1 Створення бази даних

Результатом проєктування бази даних є сформований SQL-скрипт, який використовується для створення таблиць, які були наведені в ER-моделі та в табличних описах:

```
CREATE TABLE `contact_info` (  
  `id` int PRIMARY KEY AUTO_INCREMENT,  
  `name` varchar(100) NOT NULL,  
  `number` varchar(15) UNIQUE NOT NULL,  
  `email` varchar(100) UNIQUE  
);
```

```
CREATE TABLE `customer` (  
  `id` int PRIMARY KEY AUTO_INCREMENT,  
  `contact_info_id` int UNIQUE NOT NULL,  
  
  FOREIGN KEY (`contact_info_id`) REFERENCES `contact_info` (`id`)  
);
```

```
CREATE TABLE `staff` (  
  `id` INT PRIMARY KEY AUTO_INCREMENT,  
  `salary` DECIMAL(7, 2) NOT NULL,  
  `hire_date` DATE NOT NULL,  
  `contact_info_id` INT UNIQUE NOT NULL,  
  
  CHECK (`salary` > 0),  
  
  FOREIGN KEY (`contact_info_id`) REFERENCES `contact_info` (`id`)  
);
```

```
CREATE TABLE `supervisor` (  
  `id` int PRIMARY KEY AUTO_INCREMENT,  
  `staff_id` int UNIQUE NOT NULL,  
  
  FOREIGN KEY (`staff_id`) REFERENCES `staff` (`id`)  
);
```

```
CREATE TABLE `agent` (  
  `id` int PRIMARY KEY AUTO_INCREMENT,  
  `staff_id` int UNIQUE NOT NULL,  
  `supervisor_id` int NOT NULL,  
  
  FOREIGN KEY (`staff_id`) REFERENCES `staff` (`id`),  
  FOREIGN KEY (`supervisor_id`) REFERENCES `supervisor` (`id`)  
);
```

```
CREATE TABLE `feedback` (  
  `id` int PRIMARY KEY AUTO_INCREMENT,  
  `rating` enum('1','2','3','4','5') NOT NULL,  
  `comments` text(1000),  
  `customer_id` int NOT NULL,  
  `agent_id` int NOT NULL,  
  
  FOREIGN KEY (`customer_id`) REFERENCES `customer` (`id`),  
  FOREIGN KEY (`agent_id`) REFERENCES `agent` (`id`)  
);
```

```
CREATE TABLE `issue` (  
  `id` int PRIMARY KEY AUTO_INCREMENT,
```

```
`description` varchar(255) NOT NULL
);
```

```
CREATE TABLE `service` (
  `id` int PRIMARY KEY AUTO_INCREMENT,
  `name` varchar(100) NOT NULL,
  `price` decimal(6, 2) NOT NULL,
```

```
  CHECK (`price` >=0 )
);
```

```
CREATE TABLE `call` (
  `id` int PRIMARY KEY AUTO_INCREMENT,
  `start_at` timestamp NOT NULL,
  `end_at` timestamp NOT NULL DEFAULT current_timestamp,
  `customer_id` int NOT NULL,
  `agent_id` int NOT NULL,
```

```
  CHECK (`end_at` <= `start_at` AND `start_at` = `end_at` <= 7200),
```

```
  FOREIGN KEY (`customer_id`) REFERENCES `customer` (`id`),
```

```
  FOREIGN KEY (`agent_id`) REFERENCES `agent` (`id`)
```

```
);
```

```
CREATE TABLE `call_issue` (
  `call_id` int NOT NULL,
  `issue_id` int NOT NULL,
```

```
  PRIMARY KEY (`call_id`, `issue_id`),
```

```
  FOREIGN KEY (`call_id`) REFERENCES `call` (`id`),
```

```

FOREIGN KEY (`issue_id`) REFERENCES `issue` (`id`)
);

CREATE TABLE `call_service` (
  `call_id` int NOT NULL,
  `service_id` int NOT NULL,

  PRIMARY KEY (`call_id`, `service_id`),
  FOREIGN KEY (`call_id`) REFERENCES `call` (`id`),
  FOREIGN KEY (`service_id`) REFERENCES `service` (`id`)
);

```

5.2 Імпортування даних

Для імпортування згенерованих даних, що зберігаються в текстових файлах у форматі csv (дані розділені комою), треба використати команду `LOAD DATA INFILE` та вказати шлях до даних. Дані повинні зберігатися в спеціальній теці `MySQL\MySQL Server 8.0\Uploads`.

1. Імпорт даних у таблицю `contact_info`:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\contact-info.csv'
INTO TABLE `contact_info`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(`id`, `name`, `number`, `email`);

```

2. Імпорт даних у таблицю `customer`:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\customer.csv'

```



```

INTO TABLE `customer`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(`id`, `contact_info_id`);

```

3. Імпорт даних у таблицю staff:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\staff.csv'
INTO TABLE `staff`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(`id`, `salary`, `hire_date`, `contact_info_id`);

```

4. Імпорт даних у таблицю supervisor:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\supervisor.csv'
INTO TABLE `supervisor`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(`id`, `staff_id`);

```

5. Імпорт даних у таблицю agent:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\agent.csv'

```

```

INTO TABLE `agent`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(`id`, `staff_id`, `supervisor_id`);

```

6. Імпорт даних у таблицю feedback:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\feedback.csv'
INTO TABLE `feedback`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(`id`, `rating`, `comments`, `customer_id`, `agent_id`);

```

7. Імпорт даних у таблицю issue:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\issue.csv'
INTO TABLE `issue`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS
(`id`, `description`);

```

8. Імпорт даних у таблицю service:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\service.csv'

```

```

INTO TABLE `service`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS
(`id`, `name`, `price`);

```

9. Імпорт даних у таблицю call:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\call.csv'
INTO TABLE `call`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS
(`id`, `start_at`, `end_at`, `customer_id`, `agent_id`);

```

10. Імпорт даних у таблицю call_issue:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\call-issue.csv'
INTO TABLE `call_issue`
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\r\n'
IGNORE 1 ROWS
(`call_id`, `issue_id`);

```

11. Імпорт даних у таблицю call-service:

```

LOAD DATA INFILE 'C:\\ProgramData\\MySQL\\MySQL Server
8.0\\Uploads\\call-service.csv'

```

```
INTO TABLE `call_service`  
FIELDS TERMINATED BY ','  
ENCLOSED BY ''''  
LINES TERMINATED BY '\r\n'  
IGNORE 1 ROWS  
(`call_id`, `service_id`);
```

Імпорт даних у таблиця відрізняється через їхній формат запису. Деякі дані були згенеровані штучно, за допомогою застосунків або інтернет додатків (наприклад сайт [cobbl\[6\]](#) та [filldb\[7\]](#)), інша – створена та описана вручну.

6 СТВОРЕННЯ КОРИСТУВАЧІВ БАЗИ ДАНИХ

6.1 Адміністратор call-центру

```
CREATE USER 'callcenter_administrator'@'localhost' IDENTIFIED BY '12345678';  
  
GRANT ALL PRIVILEGES ON callcenter.* TO 'callcenter_administrator'@'localhost';
```

6.2 Наглядач call-центру

```
CREATE USER 'callcenter_supervisor'@'localhost' IDENTIFIED BY '12345678';  
GRANT SELECT ON callcenter.feedback TO 'callcenter_supervisor'@'localhost';  
GRANT SELECT, INSERT, DELETE, UPDATE ON callcenter.issue TO 'callcenter_supervisor'@'localhost';  
GRANT SELECT, INSERT, DELETE, UPDATE ON callcenter.service TO 'callcenter_supervisor'@'localhost';  
GRANT SELECT, INSERT, DELETE, UPDATE ON callcenter.agent TO 'callcenter_supervisor'@'localhost';  
GRANT SELECT, INSERT, DELETE, UPDATE ON callcenter.customer TO 'callcenter_supervisor'@'localhost';  
GRANT SELECT, INSERT, DELETE, UPDATE ON callcenter.call_issue TO 'callcenter_supervisor'@'localhost';  
GRANT SELECT, INSERT, DELETE, UPDATE ON callcenter.call_service TO 'callcenter_supervisor'@'localhost';  
GRANT SELECT, INSERT, DELETE, UPDATE ON callcenter.`call` TO 'callcenter_supervisor'@'localhost';
```

6.3 Агент call-центру

```
CREATE USER 'callcenter_agent'@'localhost' IDENTIFIED BY '12345678';  
GRANT SELECT ON callcenter.feedback TO 'callcenter_agent'@'localhost';  
GRANT SELECT ON callcenter.issue TO 'callcenter_agent'@'localhost';  
GRANT SELECT ON callcenter.service TO 'callcenter_agent'@'localhost';
```

```
GRANT SELECT ON callcenter.customer TO 'callcenter_agent'@'localhost';
GRANT SELECT, INSERT ON callcenter.call_issue TO
'callcenter_agent'@'localhost';
GRANT SELECT, INSERT ON callcenter.call_service TO
'callcenter_agent'@'localhost';
GRANT SELECT, INSERT, UPDATE ON callcenter.`call` TO
'callcenter_agent'@'localhost';
```

6.4 Користувач call-центру

```
CREATE USER 'callcenter_customer'@'localhost' IDENTIFIED BY '12345678';
GRANT SELECT, INSERT ON callcenter.feedback TO
'callcenter_customer'@'localhost';
```

Результат виконання команд для створення користувачів командами SQL на рис. 6.1.

	user	host
►	callcenter_administrator	localhost
	callcenter_agent	localhost
	callcenter_customer	localhost
	callcenter_supervisor	localhost

Рис. 6.1 – Створені користувачі в базі даних call-центру

7 РОБОТА З БАЗОЮ ДАНИХ

7.1 Тексти генераторів

Було встановлено значення AUTO_INCREMENT для унікальних ідентифікаторів таблиць у базі даних за допомогою наступних SQL команд:

```
ALTER TABLE contact_info  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE customer  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE staff  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE supervisor  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE agent  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE feedback  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE issue  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE service  
MODIFY COLUMN id INT AUTO_INCREMENT;
```

```
ALTER TABLE call
```

```
MODIFY COLUMN id INT AUTO_INCREMENT;
```

7.2 Тексти збережених процедур/функцій.

Створено різноманітні процедури та функції для підтримки діяльності call-центру та впровадження функціоналу зазначеного в аналізі предметного середовища

7.2.1 Функція calculate_last_month_earnings

Функція для отримання заробітку call-центру на продажах послуг за останні місяці. Параметром функції виступає кількість місяців. Створена для допомоги керівникам call-центру відслідковувати успішність роботи агентів та call-центру загалом. Результат виконання функції на рис. 7.1.

```
DELIMITER //
```

```
CREATE FUNCTION calculate_last_month_earnings(num_previous_months INT)
```

```
RETURNS DECIMAL(10,2)
```

```
DETERMINISTIC
```

```
BEGIN
```

```
    DECLARE total_price DECIMAL(10,2);
```

```
    SELECT SUM(service.price) INTO total_price
```

```
    FROM callcenter.call_service
```

```
    JOIN callcenter.service ON call_service.service_id = service.id
```

```
    JOIN callcenter.`call` ON call_service.call_id = `call`.id
```

```
    WHERE `call`.end_at >= DATE_SUB(CURRENT_DATE, INTERVAL
num_previous_months MONTH);
```

```
    RETURN total_price;
```

```
END //
```

```
DELIMITER ;
```


Result Grid		Filter Rows:
	calculate_last_month_earnings(3)	
▶	13124.58	

Рис. 7.1 – Результат виконання функції calculate_last_month_earnings для останніх трьох місяців

7.2.2 Процедура agent_productivity_report

Дана процедура повертає кількість дзвінків, які обробив агент впродовж зазначеного терміну. Створена для відслідковування продуктивності агента. Результати виконання процедури на рис. 7.2.

DELIMITER //

```
CREATE PROCEDURE get_agent_total_calls(IN start_date DATE, IN end_date DATE)
```

```
BEGIN
```

```
SELECT
```

```
  `call`.agent_id,  
  ci.`name` AS agent_name,  
  COUNT(*) AS total_calls
```

```
FROM
```

```
  callcenter.`call`
```

```
JOIN
```

```
(
```

```
  SELECT
```

```
    agent.id AS agent_id,  
    staff.contact_info_id
```

```
FROM
```

```
  agent
```

```
JOIN
```

```
  staff ON agent.staff_id = staff.id
```

```
) a ON `call`.agent_id = a.agent_id
```

JOIN

contact_info ci ON a.contact_info_id = ci.id

WHERE

`call`.start_at >= start_date AND `call`.end_at <= end_date

GROUP BY

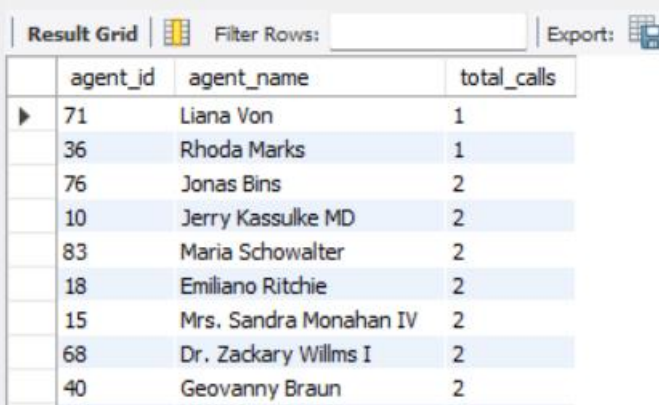
a.agent_id, ci.`name`

ORDER BY

total_calls;

END //

DELIMITER ;



	agent_id	agent_name	total_calls
▶	71	Liana Von	1
	36	Rhoda Marks	1
	76	Jonas Bins	2
	10	Jerry Kassulke MD	2
	83	Maria Schowalter	2
	18	Emiliano Ritchie	2
	15	Mrs. Sandra Monahan IV	2
	68	Dr. Zackary Willms I	2
	40	Geovanny Braun	2

Рис. 7.2 – Результат виконання процедури get_agent_total_calls. Оброблені дзвінки впродовж 2020-01-01 – 2023-12-31

7.2.3 Процедура get_customer_feedback

Процедура створена для отримання відгуків від користувачів про кожного агента. Процедура дозволяє дізнатися середній рейтинг агента та загальну кількість відгуків. Результат виконання процедури на рис. 7.3.

DELIMITER //

CREATE PROCEDURE get_customer_feedback()

BEGIN

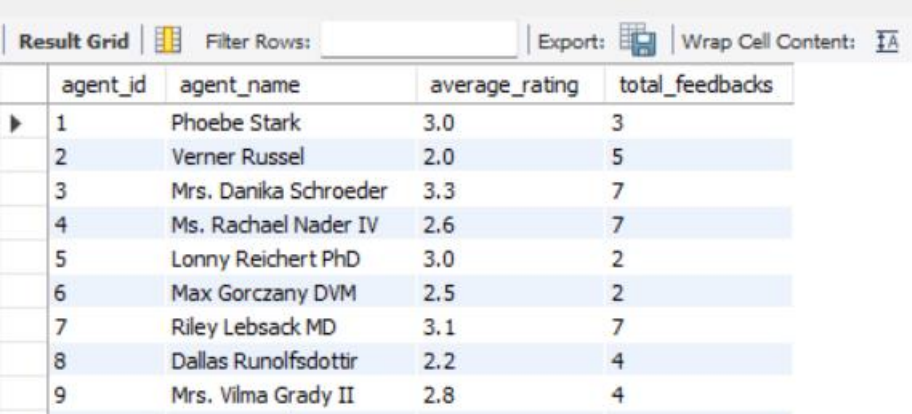
SELECT

agent.id AS agent_id,

```

contact_info.`name` AS agent_name,
FORMAT(AVG(feedback.rating), 1) AS average_rating,
COUNT(*) AS total_feedbacks
FROM
    callcenter.agent
LEFT JOIN callcenter.feedback ON feedback.agent_id = agent.id
JOIN callcenter.staff ON agent.staff_id = staff.id
JOIN callcenter.contact_info ON staff.contact_info_id = contact_info.id
GROUP BY
    agent.id;
END //
DELIMITER ;

```



	agent_id	agent_name	average_rating	total_feedbacks
▶	1	Phoebe Stark	3.0	3
	2	Verner Russel	2.0	5
	3	Mrs. Danika Schroeder	3.3	7
	4	Ms. Rachael Nader IV	2.6	7
	5	Lonny Reichert PhD	3.0	2
	6	Max Gorczany DVM	2.5	2
	7	Riley Lebsack MD	3.1	7
	8	Dallas Runolfsdottir	2.2	4
	9	Mrs. Vilma Grady II	2.8	4

Рис. 7.3 – Результат виконання процедури get_customer_feedback

7.2.4 Процедура most_popular_services

Процедура відображає кількість дзвінків у яких була замовлена та чи інша послуга. Створена для відслідковування популярності процедур серед клієнтів call-центру. Результат виконання процедури на рис. 7.4

```

DELIMITER //
CREATE PROCEDURE most_popular_services()
BEGIN
    SELECT service.id, service.`name`, COUNT(call_service.call_id) AS call_count

```

```

FROM service
JOIN call_service ON service.id = call_service.service_id
GROUP BY service.`name`
ORDER BY call_count DESC;
END //
DELIMITER ;

```

	id	name	call_count
►	4	Password Recovery	63
	13	Phone Call Troubleshooting	62
	24	Email Attachment Support	59
	17	Billing Support	58
	27	Mobile App Crash Resolution	58
	3	Software Installation	57
	18	Account Assistance	56
	12	Navigation Services	55
	19	Virus Removal	54

Рис. 7.4 – результат виконання процедури `most_populat_services`

7.2.5 Процедура `fetch_agent_related_data`

Процедруа створена для відображення всієї доступної інформації про конкретного агента. Використовується для відслідковування даних агента, для їх корегування тощо. Результат виконання процедури для користувача під номером один на рис. 7.5, рис. 7.6 та рис. 7.7.

```
DROP PROCEDURE IF EXISTS fetch_agent_related_data;
```

```
DELIMITER //
```

```
CREATE PROCEDURE fetch_agent_related_data(_agent_id INT)
```

```
BEGIN
```

```
    SELECT `call`.id AS call_id, `call`.start_at, `call`.end_at, customer_id,
    call_service.service_id, call_issue.issue_id
```

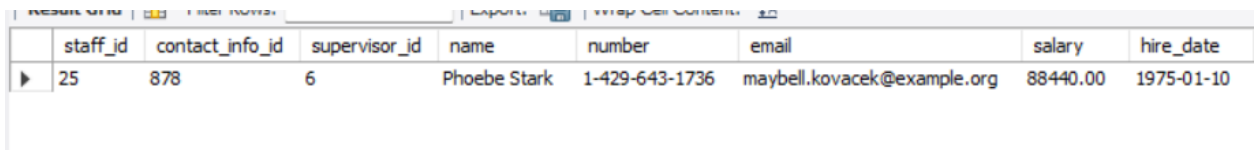
```
    FROM `call`
```

```
    LEFT JOIN call_service ON `call`.id = call_service.call_id
```

```
LEFT JOIN call_issue ON `call`.id = call_issue.call_id
WHERE `call`.agent_id = _agent_id;
```

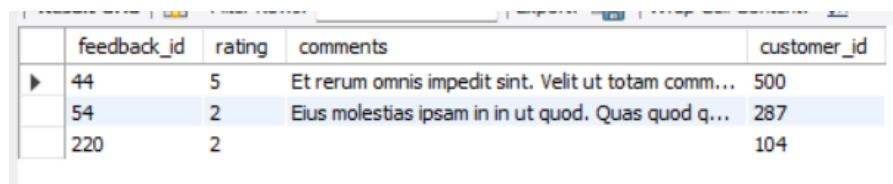
```
SELECT id AS feedback_id, rating, comments, customer_id
FROM feedback WHERE agent_id = _agent_id;
```

```
SELECT    staff.id    AS    staff_id,    contact_info_id,    supervisor_id,
`name`,`number`,`email, salary, hire_date
FROM agent
JOIN staff ON agent.staff_id = staff.id
JOIN contact_info ON staff.contact_info_id = contact_info.id
WHERE agent.id = _agent_id;
END //
DELIMITER ;
```



staff_id	contact_info_id	supervisor_id	name	number	email	salary	hire_date
25	878	6	Phoebe Stark	1-429-643-1736	maybell.kovacek@example.org	88440.00	1975-01-10

Рис. 7.5 – одна з трьох таблиць результату виконання процедури `fetch_agent_related_data`. Таблиця з особистою інформацією



feedback_id	rating	comments	customer_id
44	5	Et rerum omnis impedit sint. Velit ut totam comm...	500
54	2	Eius molestias ipsam in in ut quod. Quas quod q...	287
220	2		104

Рис. 7.6 – одна з трьох таблиць результату виконання процедури `fetch_agent_related_data`. Таблиця з відгуками про агента

	call_id	start_at	end_at	customer_id	service_id	issue_id
▶	60	2019-06-24 10:43:39	2019-06-24 11:48:15	432	16	16
	94	2016-09-24 15:43:29	2016-09-24 17:17:21	466	19	19
	94	2016-09-24 15:43:29	2016-09-24 17:17:21	466	19	29
	94	2016-09-24 15:43:29	2016-09-24 17:17:21	466	29	19
	94	2016-09-24 15:43:29	2016-09-24 17:17:21	466	29	29
	247	2021-09-21 08:16:04	2021-09-21 10:01:39	408	17	17
	247	2021-09-21 08:16:04	2021-09-21 10:01:39	408	17	27
	247	2021-09-21 08:16:04	2021-09-21 10:01:39	408	27	17
	247	2021-09-21 08:16:04	2021-09-21 10:01:39	408	27	27
	306	2014-07-27 15:41:14	2014-07-27 16:23:31	7	9	9
	306	2014-07-27 15:41:14	2014-07-27 16:23:31	7	9	12
	306	2014-07-27 15:41:14	2014-07-27 16:23:31	7	12	9
	306	2014-07-27 15:41:14	2014-07-27 16:23:31	7	12	12
	327	2018-07-05 04:53:38	2018-07-05 05:52:22	135	28	28
	377	2020-04-02 03:19:09	2020-04-02 03:52:23	327	12	12
	377	2020-04-02 03:19:09	2020-04-02 03:52:23	327	12	18

Рис. 7.7 – одна з трьох таблиць результату виконання процедури `fetch_agent_related_data`. Детальна таблиця з проведеними дзвінками

7.2.6 Процедура `increase_salary_for_agent`

Дана процедура створена для підвищення заробітної плати на конкретну суму конкретному агенту. Результат показан на рис. 7.8 та рис. 7.9

```
DELIMITER //
```

```
CREATE PROCEDURE increase_salary_for_agent(IN agent_id INT, IN
special_amount DECIMAL(7, 2))
```

```
BEGIN
```

```
    DECLARE _staff_id INT;
```

```
    DECLARE current_salary DECIMAL(7, 2);
```

```
    SELECT staff_id INTO _staff_id FROM agent WHERE id = agent_id;
```

```
    SELECT salary INTO current_salary FROM staff WHERE id = _staff_id;
```

```
    IF current_salary IS NOT NULL THEN
```

```
        UPDATE staff SET salary = current_salary + special_amount WHERE id =
        _staff_id;
```

```

SELECT CONCAT('Salary increased by ', special_amount, ' for Agent ID ',
agent_id) AS result;
ELSE
  SIGNAL SQLSTATE '45000'
  SET MESSAGE_TEXT = 'Agent not found.';
END IF;
END //
DELIMITER ;

```

staff_id	contact_info_id	supervisor_id	name	number	email	salary	hire_date
60	809	11	Verner Russel	038.624.7501x48	josiane.hegmann@example.net	52744.00	1972-08-21

Рис. 7.8 – Зарплата агента до підвищення

staff_id	contact_info_id	supervisor_id	name	number	email	salary	hire_date
60	809	11	Verner Russel	038.624.7501x48	josiane.hegmann@example.net	57744.00	1972-08-21

Рис. 7.9 – зарплата агента після підвищення.

7.2.7 Процедура get_agent_total_earnings

Процедура створена для отримання загального заробітку з продаж послуг конкретного агента. Процедура може бути корисною для оцінки працьовитості агента та здібності продавати послуги. Результат для агента номер два на рис. 7.10.

```

DELIMITER //
CREATE PROCEDURE get_agent_total_earnings(IN _agent_id INT)
BEGIN
  DECLARE total_earnings DECIMAL(10, 2);

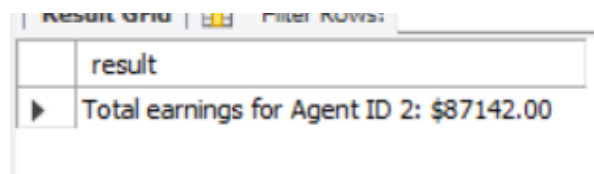
```

```

SELECT s.salary + COALESCE(SUM(serv.price), 0) INTO total_earnings
FROM staff s
LEFT JOIN agent a ON s.id = a.staff_id
LEFT JOIN `call` c ON a.id = c.agent_id
LEFT JOIN call_service cs ON c.id = cs.call_id
LEFT JOIN service serv ON cs.service_id = serv.id
WHERE s.id = _agent_id;

IF total_earnings IS NOT NULL THEN
    SELECT CONCAT('Total earnings for Agent ID ', _agent_id, ': $', total_earnings)
AS result;
ELSE
    SELECT CONCAT('No earnings available for Agent ID ', _agent_id) AS Result;
END IF;
END //
DELIMITER ;

```



result
Total earnings for Agent ID 2: \$87142.00

Рис. 7.10 – загальний заробіток агента з продаж послуг

7.2.8 Процедура update_agent_supervisor

Процедура створена для зручного оновлення наглядача для агента. Допомогає облегшити процес додавання та зміни даних. Результат оновлення процедури на рис. 7.11

```

DELIMITER //
CREATE PROCEDURE update_agent_supervisor(IN agentId INT, IN
newSupervisorId INT)

```



```

BEGIN
  IF EXISTS (SELECT 1 FROM agent WHERE id = agentId) AND EXISTS
(SELECT 1 FROM supervisor WHERE id = newSupervisorId) THEN
    UPDATE agent SET supervisor_id = newSupervisorId WHERE id = agentId;
    SELECT CONCAT('Supervisor updated for Agent ID ', agentId) AS Result;
  ELSE
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Agent or supervisor not found.';
  END IF;
END //
DELIMITER ;

```

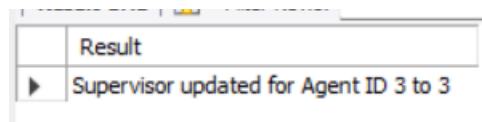


Рис. 7.11 – номер наглядача змінено для агента номер 3

7.2.9 Процедура average_call_duration

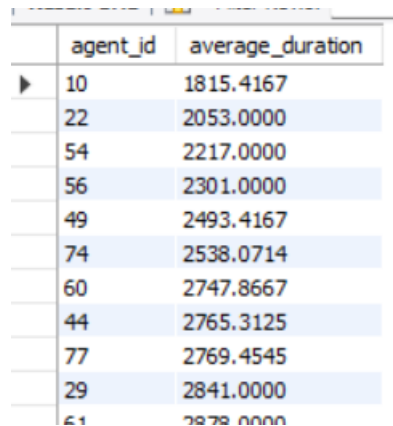
Дана процедура створена для отримання середньої тривалості розмову для кожного агента. Процедура допоможе дізнатися хто довше розмовляє з користувачами та більше їх цікавить. Результат процедури на рис. 7.12

```

DELIMITER //
CREATE PROCEDURE average_call_duration()
BEGIN
  SELECT agent.id AS agent_id, AVG(TIMESTAMPDIFF(SECOND,
`call`.start_at, `call`.end_at)) AS average_duration
  FROM agent
  JOIN `call` ON agent.id = `call`.agent_id
  GROUP BY agent.id
  ORDER BY average_duration;

```

```
END //
DELIMITER ;
```



	agent_id	average_duration
▶	10	1815.4167
	22	2053.0000
	54	2217.0000
	56	2301.0000
	49	2493.4167
	74	2538.0714
	60	2747.8667
	44	2765.3125
	77	2769.4545
	29	2841.0000
	61	2878.0000

Рис. 7.12 – середня тривалість розмови кожного агента в секундах

7.2.10 Процедура most_common_issues

Процедура створена для виведення списку найчастіших предметів дискусії. Це допоможе зрозуміти найчастіші проблеми та зменшити їхню кількість або збільшити вартість послуг, які вирішують ці проблеми. Результат процедури на рис 7.13.

```
DELIMITER //
CREATE PROCEDURE most_common_issues()
BEGIN
    SELECT `issue`.`id`, `issue`.`description`, COUNT(`call_issue`.`call_id`) AS
`call_count`
    FROM `issue`
    JOIN `call_issue` ON `issue`.`id` = `call_issue`.`issue_id`
    GROUP BY `issue`.`id`, `issue`.`description`
    ORDER BY `call_count` DESC;
END //
DELIMITER ;
```

	id	description	call_count
►	4	Lost password	63
	13	Virus or malware attack	62
	24	Email attachment issues	59
	17	Phone call dropouts	58
	27	Mobile app crashing	58
	3	Software installation issue	57
	18	Streaming quality issues	56
	12	Account login issue	55
	19	GPS not working	54
	23	Lost data during update	54
	30	GPS signal interference	54

Рис. 7.13 – найчастіші предмети дискусії в розмові

7.3 Тригери

7.3.1 Тригер delete_agent_related_data

Даний тригер створений для видалення відповідних записів з таблиць, які були пов'язані з агентом. Для виведення результату видалення даних для користувача під номером один використаємо процедуру fetch_agent_related_data на рис. 7.14, рис. 7.15 та рис. 7.16.

```
DELIMITER //
```

```
CREATE TRIGGER delete_agent_related_data
```

```
AFTER DELETE ON agent
```

```
FOR EACH ROW
```

```
BEGIN
```

```
    DECLARE _staff_id INT;
```

```
    DECLARE _contact_info_id INT;
```

```
    SELECT staff_id INTO _staff_id FROM agent WHERE id = OLD.id;
```

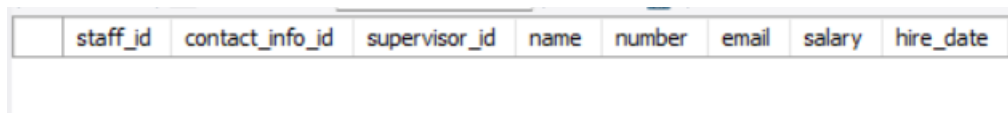
```
    SELECT contact_info_id INTO _contact_info_id FROM staff WHERE id =  
    _staff_id;
```

```
    DELETE FROM call_service WHERE call_id IN (SELECT id FROM `call`  
    WHERE agent_id = OLD.id);
```

```

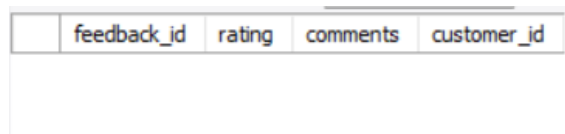
DELETE FROM call_issue WHERE call_id IN (SELECT id FROM `call`
WHERE agent_id = OLD.id);
DELETE FROM `call` WHERE agent_id = OLD.id;
DELETE FROM feedback WHERE agent_id = OLD.id;
DELETE FROM staff WHERE id = _staff_id;
DELETE FROM contact_info WHERE id = _contact_info_id;
END //
DELIMITER ;

```



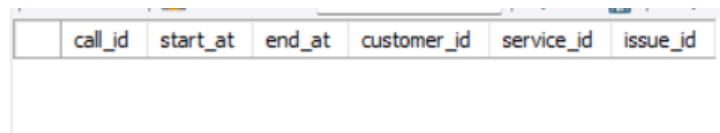
staff_id	contact_info_id	supervisor_id	name	number	email	salary	hire_date
----------	-----------------	---------------	------	--------	-------	--------	-----------

Рис. 7.14 – Уся особиста інформація про агента видалена тригером.



feedback_id	rating	comments	customer_id
-------------	--------	----------	-------------

Рис. 7.15 – Усі відгуки про агента видалені тригером.



call_id	start_at	end_at	customer_id	service_id	issue_id
---------	----------	--------	-------------	------------	----------

Рис. 7.16 – Усі дані про дзвінки агента видалено

7.3.2 Тригер delete_supervisor_related_data

Даний тригер аналогічен минулому – створений для видалення відповідних даних, які належали наглядачу. Також, він призначає агентам нового наглядача, в якого найменша кількість агентів у підпорядкуванні. Результати видалення наглядача під номером 1 на рис. 7.17 та рис. 7.18.

```

DELIMITER //
CREATE TRIGGER delete_supervisor_related_data

```

```

AFTER DELETE ON supervisor
FOR EACH ROW
BEGIN
    DECLARE _staff_id INT;
    DECLARE _contact_info_id INT;
    DECLARE _new_supervisor_id INT;
    SELECT staff_id INTO _staff_id FROM supervisor WHERE id = OLD.id;
    SELECT contact_info_id INTO _contact_info_id FROM staff WHERE id =
_staff_id;

    -- Пошук наглядча з найменшою кількістю агентів
    SELECT id INTO _new_supervisor_id FROM supervisor
    WHERE id != OLD.id
    ORDER BY (SELECT COUNT(*) FROM agent WHERE supervisor_id =
supervisor.id) ASC
    LIMIT 1;

    -- Перерозподіл агентів до нового наглядча
    UPDATE agent SET supervisor_id = _new_supervisor_id WHERE supervisor_id
= OLD.id;

    DELETE FROM staff WHERE id = _staff_id;
    DELETE FROM contact_info WHERE id = _contact_info_id;
END //
DELIMITER ;

```

	supervisor_id	contact_info_id	staff_id	agent_id
	6	797	6	83
	7	855	7	7
	7	855	7	9
	7	855	7	12
	7	855	7	19
	7	855	7	53
	7	855	7	60
	7	855	7	70
	8	710	8	52
	8	710	8	61
	9	501	9	74

Рис. 7.17 – Дані наглядачів та їхніх агентів до видалення

	supervisor_id	contact_info_id	staff_id	agent_id
	6	797	6	83
	8	710	8	7
	8	710	8	9
	8	710	8	12
	8	710	8	19
	8	710	8	52
	8	710	8	53
	8	710	8	60
	8	710	8	61
	8	710	8	70
	9	501	9	74

Рис. 7.18 – Дані наглядачів після видалення наглядача під номером 7. Усі підпорядковані йому агенти перейшли до наглядача 8

7.3.3 Тригер delete_customer_related_record

Даний тригер схожий на два минулих – він створений для видалення відповідних записів з таблиці contact_info, які належали користувачу. Результати видалення всіх даних користувача номер 21 на прикладі його дзвінків рис. 7.19 та рис. 7.20.

```
DELIMITER //
```

```
CREATE TRIGGER delete_customer_related_data
```

```
AFTER DELETE ON customer
```

```
FOR EACH ROW
```

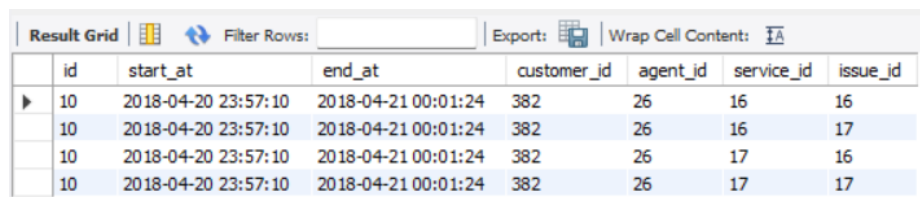
```
BEGIN
```



```

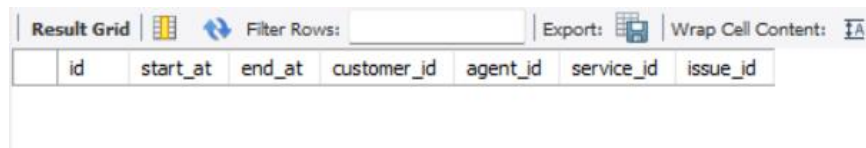
AFTER DELETE ON `call`
FOR EACH ROW
BEGIN
    DELETE FROM call_service WHERE call_id = OLD.id;
    DELETE FROM call_issue WHERE call_id = OLD.id;
END //
DELIMITER ;

```



	id	start_at	end_at	customer_id	agent_id	service_id	issue_id
▶	10	2018-04-20 23:57:10	2018-04-21 00:01:24	382	26	16	16
	10	2018-04-20 23:57:10	2018-04-21 00:01:24	382	26	16	17
	10	2018-04-20 23:57:10	2018-04-21 00:01:24	382	26	17	16
	10	2018-04-20 23:57:10	2018-04-21 00:01:24	382	26	17	17

Рис. 7.21 – Дані дзвінків номер 10 до видалення



	id	start_at	end_at	customer_id	agent_id	service_id	issue_id
	10	2018-04-20 23:57:10	2018-04-21 00:01:24	382	26	17	17

Рис. 7.22 – Дані дзвінків номер 10 після видалення

7.3.5 Тригер auto_assign_supervisor

Даний тригер створений для автоматичного назначення наглядча за агентом, в залежності від кількості агентів у підпорядкуванні наглядчу. Результат роботи на новому працівнику під номером 101 на рис. 7.23.

```

DELIMITER //
CREATE TRIGGER auto_assign_supervisor
BEFORE INSERT ON agent
FOR EACH ROW
BEGIN
    DECLARE _new_supervisor_id INT;

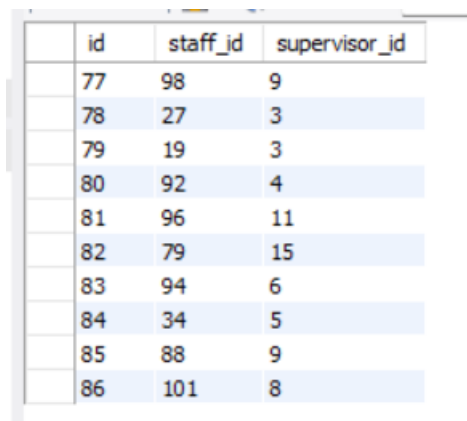
    SELECT id INTO _new_supervisor_id FROM supervisor

```



```
ORDER BY (SELECT COUNT(*) FROM agent WHERE supervisor_id =
supervisor.id) ASC
LIMIT 1;
```

```
SET NEW.supervisor_id = _new_supervisor_id;
END //
DELIMITER ;
```



	id	staff_id	supervisor_id
	77	98	9
	78	27	3
	79	19	3
	80	92	4
	81	96	11
	82	79	15
	83	94	6
	84	34	5
	85	88	9
	86	101	8

Рис. 7.23 – Агент під номером 86 має наглядча номер 8

7.4 Представлення

7.4.1 Представлення StaffView

Представлення StaffView містить повну інформацію про всіх співробітників call-центру. Вигляд представлення на рис. 7.24.

```
CREATE VIEW StaffView AS
```

```
SELECT
```

```
s.id AS staff_id,
```

```
s.salary,
```

```
s.hire_date,
```

```
c.name AS staff_name,
```

```
c.number AS staff_contact_number,
```

```
sv.id AS supervisor_id
```

FROM staff s

JOIN contact_info c ON s.contact_info_id = c.id

LEFT JOIN supervisor sv ON s.id = sv.staff_id;

	staff_id	salary	hire_date	staff_name	staff_contact_number	supervisor_id
▶	1	15594.00	1979-12-14	Charlie Rolfson	318-117-4003x63	1
	2	87142.00	2007-09-04	Vena Senger	752-064-5600x94	2
	3	460.00	2019-04-26	Eloy Block	128.139.6320x12	3
	4	44471.00	1979-10-16	Leon Nolan Jr.	978-508-9532x22	4
	5	42923.00	1985-10-23	Vivien Botsford IV	845-171-1008	5
	6	15485.00	2018-02-26	Vella Mueller I	982.630.7353	6
	7	7600.00	2005-01-05	Brett Gerlach	(036)156-6381	7
	8	70248.00	1988-03-19	Jairo Hills	948.313.2919x47	8
	9	6086.00	2006-06-24	Dr. Dereck Becker	430-782-1244x29	9
	10	92827.00	1987-10-25	Jacinthe Barton	774-392-4365x58	10
	11	92111.00	2008-08-23	Wilhelmine Glover I	014-092-8258x45	11

Рис. 7.24 – Вигляд представлення StaffView

7.4.2 Представлення CallDetailsView

Представлення CallDetailsView містить інформацію повну інформацію про дзвінки, назву послуги, предмет дискусії. Вигляд представлення на рис. 7.25.

CREATE VIEW CallDetailsView AS

SELECT

ca.id AS call_id,

ca.start_at,

ca.end_at,

cu.id AS customer_id,

a.id AS agent_id,

s.name AS service_name,

iss.description AS issue_description

FROM `call` ca

JOIN customer cu ON ca.customer_id = cu.id

JOIN agent a ON ca.agent_id = a.id

LEFT JOIN call_service cs ON ca.id = cs.call_id

```

LEFT JOIN service s ON cs.service_id = s.id
LEFT JOIN call_issue ci ON ca.id = ci.call_id
LEFT JOIN issue iss ON ci.issue_id = iss.id;

```

call_id	start_at	end_at	customer_id	agent_id	service_name	issue_description
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Device Setup	Account login issue
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Device Setup	Mobile app crashing
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Navigation Services	Website not accessible
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Navigation Services	Account login issue
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Navigation Services	Mobile app crashing
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Mobile App Crash Resolution	Website not accessible
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Mobile App Crash Resolution	Account login issue
89	2020-01-18 07:45:29	2020-01-18 08:48:36	182	52	Mobile App Crash Resolution	Mobile app crashing
216	2015-06-27 16:11:26	2015-06-27 17:24:07	64	52	Data Recovery	Print job errors
216	2015-06-27 16:11:26	2015-06-27 17:24:07	64	52	Data Recovery	Account login issue
216	2015-06-27 16:11:26	2015-06-27 17:24:07	64	52	Data Recovery	Mobile app crashing

Рис. 7.25 – Вигляд представлення CallDetailsView

7.4.3 Представлення CustomerFeedbackView

Представлення CustomerFeedbackView створено для відображення інформації про користувача та відгук, що він залишив. Представлення необхідно для оцінки загальної картини серед роботи агентів. Вигляд представлення на рис. 7.26.

```

CREATE VIEW CustomerFeedbackView AS
SELECT
    c.id AS customer_id,
    ci.`name` AS customer_name,
    ci.`number` AS customer_contact_number,
    a.id AS agent_id,
    f.id AS feedback_id,
    f.rating,
    f.comments
FROM
    feedback f
JOIN customer c ON f.customer_id = c.id

```

JOIN agent a ON f.agent_id = a.id

JOIN contact_info ci ON c.contact_info_id = ci.id;

	customer_id	customer_name	customer_contact_number	agent_id	feedback_id	rating	comments
▶	383	Freddie Heathcote	1-026-085-0969x	69	1	2	Doloremque nihil deserunt enim eos tempore a a...
	387	Ena Schiller Jr.	014-370-6880x46	35	2	3	
	371	Ms. Pauline Schumm	466.649.9383x04	83	3	5	
	408	Miss Frieda Boyer	+15(8)285603232	52	4	1	
	78	Anissa Schoen	+03(5)319249554	62	5	3	Culpa culpa voluptatum id enim pariatur aspern...
	149	Kendra Jacobson DVM	1-747-642-5994	81	6	4	
	483	Carrie Gleason	1-261-173-9300	27	7	5	
	54	Brando Kohler MD	(835)217-6531	38	8	3	
	500	Miss Delores Nienow	(392)638-6362x2	23	9	1	Dolorem perspiciatis optio et pariatur eum recus...
	115	Garnett Schowalter	(806)769-6214	65	10	5	
	489	Mr. Rocio Braun	(408)133-8676x8	51	11	4	

Рис. 7.26 – Вигляд представлення CustomerFeedbackView

7.5 SQL-запити

1) Запит на отримання кількості клієнтів, що використовують кожен сервіс(результат на рис. 7.27):

```
SELECT s.id AS service_id, s.name, COUNT(cs.call_id) AS usage_count
FROM service s
LEFT JOIN call_service cs ON s.id = cs.service_id
GROUP BY s.id;
```

	service_id	name	usage_count
▶	1	Technical Support	51
	2	Internet Package	43
	3	Software Installation	57
	4	Password Recovery	63
	5	Hardware Repair	43
	6	Network Troubleshooting	51
	7	Email Services	49
	8	Website Maintenance	44
	9	Anti-virus Protection	53
	10	Device Setup	52
	11	Data Recovery	43

Рис. 7.27 – Результат виконання запиту 1

2) Запит на отримання п'яти агентів з найбільшою кількістю вігдухів (результат на рис. 7.28):

```

SELECT a.id AS agent_id, COUNT(f.id) AS feedback_count
FROM agent a
LEFT JOIN feedback f ON a.id = f.agent_id
GROUP BY a.id
ORDER BY feedback_count DESC
LIMIT 5;

```

	agent_id	feedback_count
►	80	12
	58	8
	47	7
	3	7
	7	7

Рис. 7.28 – Результат виконання запиту 2

3) Запит на отримання п'яти послуг, з найкращим середнім рейтингом(результат на рис. 7.29):

```

SELECT s.id AS service_id, s.name, AVG(s.price) AS avg_price
FROM service s
GROUP BY s.id
ORDER BY avg_price DESC
LIMIT 5;

```

	service_id	name	avg_price
►	1	Technical Support	2999.990000
	5	Hardware Repair	79.990000
	25	Website Security Consultation	59.990000
	2	Internet Package	49.990000
	11	Data Recovery	49.990000

Рис. 7.29 – Результат виконання запиту 3

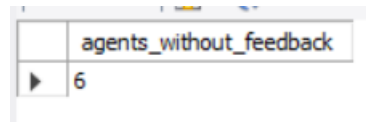
4) Запит для отримання кількості агентів, які не отримали відгук (результат на рис. 7.30):

```

SELECT COUNT(*) AS agents_without_feedback
FROM agent a

```

```
LEFT JOIN feedback f ON a.id = f.agent_id
WHERE f.id IS NULL;
```

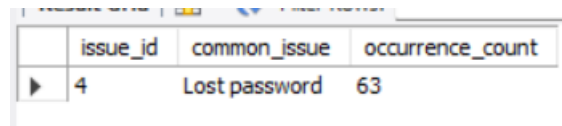


	agents_without_feedback
▶	6

Рис. 7.30 – Результат виконання запиту 4

5) Запит для отримання проблеми, яка найчастіше виникає в звітах про виклики (результат на рис. 7.31):

```
SELECT i.id AS issue_id, i.description AS common_issue, COUNT(ci.issue_id)
AS occurrence_count
FROM issue i
LEFT JOIN call_issue ci ON i.id = ci.issue_id
GROUP BY i.id
ORDER BY occurrence_count DESC
LIMIT 1;
```



	issue_id	common_issue	occurrence_count
▶	4	Lost password	63

Рис. 7.31 – Результат виконання запиту 5

6) Запит для обчислення рейтингу для кожної послуги на основі розмов користувачів та відгуків, що вони залишили (результат на рис. 7.32):

```
SELECT s.id AS service_id, s.name AS service_name, AVG(f.rating) AS
average_rating
FROM service s
JOIN call_service cs ON s.id = cs.service_id
JOIN `call` c ON cs.call_id = c.id
JOIN feedback f ON c.customer_id = f.customer_id
GROUP BY s.id;
```

	service_id	service_name	average_rating
►	8	Website Maintenance	3.0740740740...
	11	Data Recovery	3.0333333333...
	1	Technical Support	3
	4	Password Recovery	2.9032258064...
	20	Device Configuration	3
	17	Billing Support	2.9
	27	Mobile App Crash Reso...	3
	5	Hardware Repair	2.8571428571...
	9	Anti-virus Protection	2.96875
	18	Account Assistance	2.9285714285...

Рис. 7.32 – Результат виконання запиту 6

7) Запит розраховує середню кількість питань на один виклик (результат на рис. 7.33):

```
SELECT COUNT(ci.issue_id) / COUNT(DISTINCT ca.id) AS avg_issues_per_call
FROM `call` ca
LEFT JOIN call_issue ci ON ca.id = ci.call_id;
```

	avg_issues_per_call
►	1.5000

Рис. 7.33 – Результат виконання запиту 7

8) Запит показує кількість викликів для кожної години у добі (результат на рис. 7.34):

```
SELECT HOUR(start_at) AS hour_of_day, COUNT(id) AS call_count
FROM `call`
GROUP BY hour_of_day;
```

	hour_of_day	call_count
▶	15	41
	3	48
	7	51
	9	57
	0	31
	10	49
	11	44
	23	46
	20	44
	1	34
	22	37

Рис. 7.34 – Результат виконання запиту 8

9) запит визначає трьох агентів з найбільшою кількістю викликів, в яких були предмети дискусії. (результат на рис. 7.35):

```
SELECT a.id AS agent_id, COUNT(ci.call_id) AS issue_call_count
FROM agent a
LEFT JOIN call_issue ci ON a.id = ci.issue_id
GROUP BY a.id
ORDER BY issue_call_count DESC
LIMIT 3;
```

	agent_id	issue_call_count
▶	4	63
	13	62
	24	59

Рис. 7.35 – Результат виконання запиту 9

10) Запит рахує кількість викликів для кожної послуги кожного місяця (результат на рис. 7.36):

)

```
SELECT s.name AS service_name, MONTH(ca.start_at) AS month,
COUNT(cs.call_id) AS call_count
FROM service s
LEFT JOIN call_service cs ON s.id = cs.service_id
```



```
LEFT JOIN `call` ca ON cs.call_id = ca.id
GROUP BY s.name, month;
```

service_name	month	call_count
Account Recovery Assi...	1	6
Account Recovery Assi...	5	4
Account Recovery Assi...	3	4
Account Recovery Assi...	12	2
Account Recovery Assi...	4	6
Account Recovery Assi...	8	2
Anti-virus Protection	3	9
Anti-virus Protection	9	4
Anti-virus Protection	5	6
Anti-virus Protection	4	3
Anti-virus Protection	7	4

Рис. 7.36 – Результат виконання запиту 10

11) Запит рахує кількість агентів, які працювали у вихідні дні (результат на рис. 7.37):

```
SELECT COUNT(DISTINCT a.id) AS weekend_agent_count
FROM agent a
JOIN `call` ca ON a.id = ca.agent_id
WHERE DAYOFWEEK(ca.start_at) IN (1, 7);
```

weekend_agent_count
83

Рис. 7.37 – Результат виконання запиту 11

12) Запит визначає кількість агентів, які не отримали жодного відгуку для відповідної наданої послуги (результат на рис. 7.38):

```
SELECT s.id AS service_id, COUNT(DISTINCT a.id) AS
agents_without_feedback_count
FROM service s
LEFT JOIN call_service cs ON s.id = cs.service_id
LEFT JOIN `call` ca ON cs.call_id = ca.id
```

```

LEFT JOIN agent a ON ca.agent_id = a.id
LEFT JOIN feedback f ON a.id = f.agent_id
WHERE f.id IS NULL
GROUP BY s.id;

```

	service_id	agents_without_feedback_count
▶	1	2
	2	4
	3	1
	5	2
	6	2
	7	2
	8	2
	9	2
	10	2
	11	2
	12	2

Рис. 7.38 – Результат виконання запиту 12

13) Запит визначає загальну тривалість розмов для кожного агента (результат на рис. 7.39):

```

SELECT
  a.id AS agent_id,
  ci.name AS agent_name,
  SUM(TIMESTAMPDIFF(SECOND, c.start_at, c.end_at)) AS
total_duration_seconds
FROM agent a
JOIN staff s ON a.staff_id = s.id
JOIN contact_info ci ON s.contact_info_id = ci.id
JOIN `call` c ON a.id = c.agent_id
GROUP BY a.id, ci.name;;

```

	agent_id	agent_name	total_duration_seconds
▶	1	Phoebe Stark	54622
	2	Verner Russel	45662
	3	Mrs. Danika Schroeder	51471
	4	Ms. Rachael Nader IV	58988
	5	Lonny Reichert PhD	29245
	6	Max Gorczany DVM	37518
	7	Riley Lebsack MD	48328
	8	Dallas Runolfsdottir	46642
	9	Mrs. Vilma Grady II	51586
	10	Jerry Kassulke MD	21785
	11	Howard Ferry IV	60409

Рис. 7.39 – Результат виконання запиту 13

14) Запит для отримання списку клієнтів, що не залишили відгука (результат на рис. 7.40):

```
SELECT cu.id AS customer_id, ci.`name` AS customer_name
FROM customer cu
JOIN contact_info ci ON cu.contact_info_id = ci.id
WHERE cu.id NOT IN (SELECT DISTINCT customer_id FROM feedback);
```

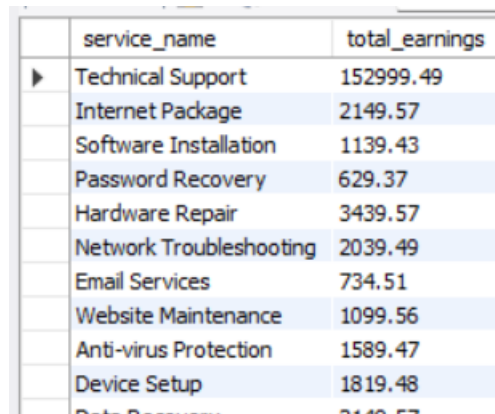
	customer_id	customer_name
▶	358	Bret Boyer
	1	Yvette Reilly
	160	Zula Hansen Sr.
	18	Chad Friesen
	96	Mr. Makenna Metz
	427	Miss Kari Runte DDS
	10	Bryon McCullough III
	73	Miss Dariana D'Amore
	136	Jamir Kihn
	392	Abigayle Kuphal
	335	Miss Zaria Ahnott Sr.

Рис. 7.40 – Результат виконання запиту 14

15) Запит для кількості заробітку кожної послуги (результат на рис. 7.41):

```
SELECT s.name AS service_name, SUM(s.price) AS total_earnings
FROM call_service cs
JOIN service s ON cs.service_id = s.id
```

GROUP BY s.name;



service_name	total_earnings
Technical Support	152999.49
Internet Package	2149.57
Software Installation	1139.43
Password Recovery	629.37
Hardware Repair	3439.57
Network Troubleshooting	2039.49
Email Services	734.51
Website Maintenance	1099.56
Anti-virus Protection	1589.47
Device Setup	1819.48
Data Recovery	2149.57

Рис. 7.41 – Результат виконання запиту 15

16) Запит для отримання списку користувачів, які дали найвищу оцінку (результат на рис. 7.42):

```
SELECT ci.name AS customer_name, f.rating AS highest_rating
FROM customer cu
JOIN contact_info ci ON cu.contact_info_id = ci.id
JOIN feedback f ON cu.id = f.customer_id
WHERE f.rating = (SELECT MAX(rating) FROM feedback);
```



customer_name	highest_rating
Ms. Pauline Schumm	5
Carrie Gleason	5
Garnett Schowalter	5
Mr. Norwood Hauck DDS	5
Kyla Smith	5
Allie Wintheiser IV	5
Malvina Walsh Sr.	5
Leola Klocko	5
Montserrat Zemlak	5
Miss Delores Nienow	5
Eduardo Gusikowski	5

Рис. 7.42 – Результат виконання запиту 16

17) Запит для отримання номерів агента, який мав найбільшу кількість дзвінків у травні (результат на рис. 7.43):

```

SELECT
  a.id AS agent_id,
  a.staff_id,
  COUNT(c.id) AS call_count
FROM agent a
JOIN `call` c ON a.id = c.agent_id
WHERE MONTH(c.start_at) = 5 -- Specify the desired month
GROUP BY a.id, a.staff_id
ORDER BY call_count DESC
LIMIT 1;

```

	agent_id	staff_id	call_count
►	12	89	5

Рис. 7.43 – Результат виконання запиту 17

18) Запит для отримання списку розмов з ціною послуг більше ніж середня (результат на рис. 7.44):

```

SELECT
  ca.id AS call_id,
  ca.start_at,
  ca.end_at,
  s.name AS service_name,
  s.price
FROM `call` ca
JOIN call_service cs ON ca.id = cs.call_id
JOIN service s ON cs.service_id = s.id
WHERE s.price > (SELECT AVG(price) FROM service);

```

	call_id	start_at	end_at	service_name	price
►	5	2023-01-04 00:24:24	2023-01-04 01:36:16	Technical Support	2999.99
	24	2019-01-09 07:09:44	2019-01-09 07:09:54	Technical Support	2999.99
	37	2019-11-02 12:49:35	2019-11-02 13:39:51	Technical Support	2999.99
	82	2015-06-16 07:05:26	2015-06-16 08:00:38	Technical Support	2999.99
	83	2019-01-31 17:01:55	2019-01-31 18:17:19	Technical Support	2999.99
	118	2014-04-30 13:21:46	2014-04-30 15:06:53	Technical Support	2999.99
	221	2015-11-02 10:42:06	2015-11-02 12:28:28	Technical Support	2999.99
	228	2020-01-12 05:58:02	2020-01-12 06:09:35	Technical Support	2999.99
	233	2016-10-03 04:45:23	2016-10-03 06:41:49	Technical Support	2999.99
	260	2016-12-18 00:47:06	2016-12-18 01:25:13	Technical Support	2999.99
	264	2017-05-21 09:10:33	2017-05-21 10:50:32	Technical Support	2999.99

Рис. 7.44 – Результат виконання запиту 18

19) Запит для визначення агента з найвищим середнім рейтингом враховуючи тільки агентів у яких хоча б 2 відгуки (результат на рис. 7.45):

```

SELECT
  a.id AS agent_id,
  a.staff_id,
  AVG(CAST(f.rating AS DECIMAL(3, 2))) AS average_rating
FROM agent a
LEFT JOIN feedback f ON a.id = f.agent_id
WHERE a.id IN (
  SELECT agent_id
  FROM feedback
  GROUP BY agent_id
  HAVING COUNT(*) >= 2
)
GROUP BY a.id, a.staff_id
ORDER BY average_rating DESC
LIMIT 1;

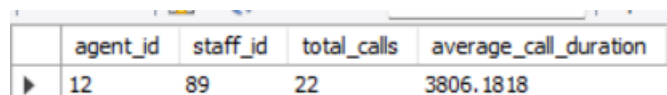
```

	agent_id	staff_id	average_rating
►	14	18	4.333333

Рис. 7.45 – Результат виконання запиту 19

20) Запит для отримання агента з найбільшою кількістю дзвінків. Відображає інформацію про агента та середню тривалість розмови (результат на рис. 7.46):

```
SELECT
  a.id AS agent_id,
  a.staff_id,
  COUNT(c.id) AS total_calls,
  AVG(TIMESTAMPDIFF(SECOND, c.start_at, c.end_at)) AS
average_call_duration
FROM agent a
JOIN `call` c ON a.id = c.agent_id
WHERE a.id = (
  SELECT agent_id
  FROM `call`
  GROUP BY agent_id
  ORDER BY COUNT(id) DESC
  LIMIT 1
)
GROUP BY a.id, a.staff_id;
```



	agent_id	staff_id	total_calls	average_call_duration
▶	12	89	22	3806.1818

Рис. 7.46 – Результат виконання запиту 20

7.6 Індeksi та результати оптимізації

Для оптимізації роботи бази даних були створені індeksi для даних в таблицях для швидкого пошуку.

Запит 1: Середній рейтинг для кожної послуги

```
SELECT s.id AS service_id, s.name AS service_name, AVG(f.rating) AS
average_rating
FROM service s
JOIN call_service cs ON s.id = cs.service_id
JOIN `call` c ON cs.call_id = c.id
JOIN feedback f ON c.customer_id = f.customer_id
GROUP BY s.id;
```

Створення індексів для оптимізації:

```
CREATE INDEX idx_service_id ON call_service (service_id);
CREATE INDEX idx_call_id ON call_service (call_id);
CREATE INDEX idx_customer_id ON `call` (customer_id);
```

✓	88	05:57:09	SELECT s.id AS service_id, s.name AS service_...	30 row(s) returned	0.016 sec / 0.000 sec
⚠	89	05:58:18	CREATE INDEX idx_service_id ON call_service ...	0 row(s) affected, 1 warning(s): 1831 Duplicate index 'idx_service_id' de...	0.047 sec
✓	90	05:58:18	CREATE INDEX idx_call_id ON call_service (cal...	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
⚠	91	05:58:18	CREATE INDEX idx_customer_id ON `call` (cust...	0 row(s) affected, 1 warning(s): 1831 Duplicate index 'idx_customer_id' ...	0.047 sec
✓	92	05:59:28	SELECT s.id AS service_id, s.name AS service_...	30 row(s) returned	0.000 sec / 0.000 sec

Рис. 7.47 – Час виконання запиту 1 до (0.016 секунд), та після оптимізації (0.000 секунд)

Запит 2: Сумарна тривалість дзвінків для кожного агента

```
SELECT
a.id AS agent_id,
ci.name AS agent_name,
SUM(TIMESTAMPDIFF(SECOND, c.start_at, c.end_at)) AS
total_duration_seconds
FROM agent a
JOIN staff s ON a.staff_id = s.id
JOIN contact_info ci ON s.contact_info_id = ci.id
JOIN `call` c ON a.id = c.agent_id
```


GROUP BY a.id, ci.name;

Створення індексів для оптимізації:

CREATE INDEX idx_agent_id ON `call` (agent_id);

✓	136	06:05:42	SELECT a.id AS agent_id, ci.name AS agent_...	85 row(s) returned	0.015 sec / 0.000 sec
⚠	137	06:06:33	CREATE INDEX idx_agent_id ON `call` (agent_id)	0 row(s) affected, 1 warning(s): 1831 Duplicate index 'idx_agent_id' defi...	0.032 sec
✓	138	06:06:38	SELECT a.id AS agent_id, ci.name AS agent_...	85 row(s) returned	0.000 sec / 0.000 sec

Рис. 7.48 – Час виконання запиту 2 до (0.015 секунд), та після оптимізації (0.000 секунд)

Запит 3: Найпопулярніша проблема серед відгуків

```
SELECT i.id AS issue_id, i.description AS popular_issue, COUNT(ci.issue_id) AS
occurrence_count
FROM issue i
LEFT JOIN call_issue ci ON i.id = ci.issue_id
GROUP BY i.id
ORDER BY occurrence_count DESC
LIMIT 1;
```

Створення індексів для оптимізації:

CREATE INDEX idx_issue_id ON call_issue (issue_id);

✓	139	06:09:22	SELECT i.id AS issue_id, i.description AS popula...	1 row(s) returned	0.016 sec / 0.000 sec
⚠	140	06:09:30	CREATE INDEX idx_issue_id ON call_issue (iss...	0 row(s) affected, 1 warning(s): 1831 Duplicate index 'idx_issue_id' defi...	0.016 sec
✓	141	06:09:35	SELECT i.id AS issue_id, i.description AS popula...	1 row(s) returned	0.000 sec / 0.000 sec

Рис. 7.49 – Час виконання запиту 3 до (0.016 секунд), та після оптимізації (0.000 секунд)

Можна впевнено зазначити, що вдалося оптимізувати роботи бази даних за допомогою створення індексів для полей у таблицях. Час виконання запитів зменшився в середньому на 0.015 секунд.

Під час роботи над цим розділом було створено процедури, представлення та виконано велику кількість SQL запитів до бази даних call-центру.

ВИСНОВКИ

Під час виконання курсової роботи я провів аналіз предметного середовища та встановив чіткі вимоги для створення бази даних для підтримки діяльності call-центру. Протягом аналізу матеріалу, за допомогою існуючих програмних продуктів, встановив бізнес правила та розробив ER-модель бази даних.

Розроблена ER-модель відображає всі наявні сутності та зв'язки між ними. Сама база даних включає створення таблиць, імпортування даних, створення багатокористувацького інтерфейсу, а також розробку генераторів, збережених процедур, тригерів, представлень та SQL-запитів.

У підсумку була створена ефективна та найдіна база даних для кол-центру, що забезпечить автоматизацію поширених процесів, а також надасть доступну звітність про успішність роботи агентів.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. СУБД MySQL. URL: <https://www.mysql.com/> (дата звернення: 20.01.2024).
2. Хмарні рішення для кол-центрів Five9. URL: <https://www.five9.com/products/capabilities/global-voice> (дата звернення: 20.01.2024).
3. Програмне забезпечення для кол-центрів VoIPTime Contact Center. URL: <https://www.voiptime.net/uk/> (дата звернення: 20.01.2024).
4. The Open source definition. URL: <https://opensource.org/osd/> (last accessed: 20.01.2024).
5. MySQL Workbench. URL: <https://www.mysql.com/products/workbench/> (last accessed: 20.01.2024).
6. Сайт для генерації даних. URL: <https://cobbl.io/> (дата звернення: 21.01.2024).
7. Сайт для генерації даних. URL: <https://filldb.info/> (дата звернення: 21.01.2024).