**ДНІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ІМЕНІ ОЛЕСЯ ГОНЧАРА**

Кафедра електронних обчислювальних машин

**КУРСОВА РОБОТА**

з дисципліни «Організація баз даних»

на тему «Розробка та створення реляційної моделі БД в НФБК»

Студента 3 курсу групи КІ-21-2

спеціальність «Комп’ютерна інженерія»

Шейко Р.О.

(прізвище та ініціали)

Керiвник: Пономарьов І. В.

Кількість балів

Національна шкала

Члени комісії:

Ахметшина Л.Г.

(підпис)

Герасимов В. В.

(підпис)

Матвєєва Н.О.

(підпис)

м. Дніпро

2023р.

**РЕФЕРАТ**

Курсова робота: с. 48, рис. 41, джерел 6.

Об’єктом дослідження є бази даних (БД), система управління базами даних (СУБД) Microsoft – SQL Server.

Предметом дослідження – розробка та нормалізація реляційної БД, створення запитів, подань та тригерів на мові SQL.

Мета роботи – розробити БД з предметної області.

Одержані висновки: розроблено і створено реляційну модель БД в НФБК.

Перелік ключових слів: ЗАПИТ, ПОДАННЯ, РБД, ТРИГГЕР, ПЕРВІСНИЙ КЛЮЧ, ЗОВНІШНІЙ КЛЮЧ, БД.

Зміст

[**Вступ** 4](#_Toc153479065)

[**Завдання на курсову роботу** 6](#_Toc153479066)

[**1.** **ТЕОРЕТИЧНА ЧАСТИНА** 8](#_Toc153479067)

[**1.1.** **Архітектура бази даних SQL Server** 8](#_Toc153479068)

[**1.2.** **Поняття про систему безпеки SQL Server** 11](#_Toc153479069)

[**1.3.** **Таблиці бази даних** 12](#_Toc153479070)

[**1.4.** **Тригери** 14](#_Toc153479071)

[**1.5.** **Вимоги до додатку** 15](#_Toc153479072)

[**1.6.** **Функціональна специфікація** 17](#_Toc153479073)

[**1.7.** **Словник даних для додатку БД Будівельної компанії** 17](#_Toc153479074)

[**1.8.** **Логічна модель. ER-діаграми** 20](#_Toc153479075)

[**1.9.** **Об`єкти і атрибути БД з розширеними кодовими полями** 21](#_Toc153479076)

[**1.10.** **Нормалізація БД** 23](#_Toc153479077)

[**2. ПРАКТИЧНА ЧАСТИНА** 25](#_Toc153479078)

[**2.1.Установка Microsoft SQL Server 2019** 25](#_Toc153479079)

[**2.2. Створення БД** 27](#_Toc153479080)

[**2.3. Створення таблиць** 29](#_Toc153479081)

[**2.4. Обмеження** 30](#_Toc153479082)

[**2.5. Створення звя’зків** 32](#_Toc153479083)

[**2.6. Заповнення таблиць даними** 33](#_Toc153479084)

[**2.7. Створення запитів та подань** 38](#_Toc153479085)

[**2.8. Створення тригерів** 43](#_Toc153479086)

[**2.9. Функції** 46](#_Toc153479087)

[**2.10. Процедури** 46](#_Toc153479088)

[**ВИСНОВКИ** 48](#_Toc153479089)

[**СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ** 50](#_Toc153479090)

# **Вступ**

Ефективне функціонування будь-якої компанії неможливе без налагодженого механізму контролю над інформацією. В умовах стрімкого розвитку інформаційних технологій та поширення мереж Internet/intranet, роль баз даних (БД) в сучасному бізнес-середовищі претерпує суттєві зміни. Збільшується обсяг інформації, кількість одночасних користувачів, а структура даних стає складнішою, роблячи необхідним вдосконалення механізмів контролю.

Змінюється погляд на побудову та управління додатками, які взаємодіють з БД. Вони тепер містять не лише технічні аспекти, але і бізнес-правила, а також логіку у вигляді сценарних мов. Ці додатки відіграють ключову роль у створенні активних та діагностичних Web-сторінок, надаючи компаніям можливість ефективного управління та аналізу інформації.

Системи керування базами даних (СКБД) стають необхідною складовою для зберігання, обробки та аналізу інформації. Сучасні СКБД представляють собою потужні інтегровані середовища, що вимагають значних апаратних ресурсів та кваліфікованих спеціалістів для ефективної роботи.

На сучасному етапі існує розмаїття систем керування базами даних, кожна з яких відрізняється своїми унікальними характеристиками. Незважаючи на різноманітність, ці платформи відповідають загальним критеріям, що становлять важливість для сучасного бізнесу:

• безперебійна робота з даними;

• підтримка різних мов програмування для створення додатків, що відповідають унікальним потребам підприємства;

• сумісність з різними платформами;

• можливість інтеграції з даними інших систем керування базами даних;

• швидке відновлення даних, надійна система безпеки та забезпечення цілісності інформації та інші важливі аспекти.

У зв'язку з постійно зростаючими вимогами та завданнями, системи керування базами даних (СКБД) постійно еволюють та покращуються. MS SQL Server визнана широким застосуванням в середніх та дрібних компаніях як завершене рішення для ефективного управління та аналізу даних. Ця платформа дозволяє оперативно розгортати веб-додатки нового покоління, інтерактивні бізнес-рішення та сховища даних, забезпечуючи необхідну масштабованість для підтримки динамічних інформаційних середовищ.

Метою цього курсового проекту є проведення інсталяції та створення реляційної бази даних за допомогою MS SQL Server, а також отримання практичних навичок роботи з цією системою. Це є важливою складовою автоматизації обробки інформації, і дозволяє студентам отримати поглиблені знання та навички у сфері сучасних систем керування базами даних.

# **Завдання на курсову роботу**

**Варіант завдання: 19.**

1. Установити Microsoft SQL Server 2005 Express Edition і Microsoft SQL Server Management Studio Express.
2. Настроїти змішану (Windows + SQL Server) аутентифікацію.
3. Створити користувача.
4. Розробити і створити реляційну модель БД в НФБК. Модель повинна бути розроблена з урахуванням ключів та обмежень.
5. При розробці обґрунтувати причину створення первинних, зовнішніх ключів, обмежень і індексів.
6. Побудувати діаграму, що відображає зв'язки й первинні ключі отриманих таблиць.
7. Заповнити отримані таблиці даними.
8. Привести тестову перевірку працездатності.
9. Написати DDL тригери.

**БД Будівельна компанія.**

Таблиці:

1. Люди (Код людини, прізвище, ім'я, по батькові, дата народження, стать, місто, адреса, телефон).
2. Посади (Код посади, найменування посади, обов'язки, оклад, вимоги).
3. Види робіт (Код роботи, найменування, опис, вартість).
4. Матеріали (Код матеріалу, найменування, упаковка, опис, вартість).
5. Бригади(Код бригади, дата створення, дата розформування).
6. Бригадні роботи(Код запису, код бригади, код виконуваної роботи)
7. Замовлення (Код замовлення, опис, вартість, дата початку, дата закінчення, код людини, код робітника).
8. Замовлення потрібні роботи (Код запису, код замовлення, код роботи, код бригади)
9. Оплата замовлень (Код оплати, код замовлення, статус оплати, опис, дата сума)
10. Паспорт (Код, Код людини, громадянство, номер).
11. Позиції робітників (Код, код людини, код посади, дата прийняття на роботу, дата звільнення, ставка, опис).
12. Список робітників(Код списку, код робітника, код бригади)
13. Матеріали для роботи(Код, код роботи, код матеріалу).

Запити і подання:

1. Подання усіх працівників компанії.
2. Подання про роботи з замовленнями.
3. Подання використаних матеріалів по замовленням.
4. Подання загальний звіт про оплату замовлення.
5. Запит кількість працівників на кожній посаді.
6. Запит незакінчених робіт.
7. Запит використаних матеріалів на певній роботі.
8. Запит список робітників, що входить до певної бригади.
9. Запит сумарна вартість робіт для кожного замовлення.

Тригери:

1. DDL тригер, що при спробі видалення кожної з таблиць робить відкат транзакції.
2. DDL тригер, що при спробі видалення даних з таблиць робить відкат транзакції.
3. DDL Тригер, що при спробі видалення ключів робить відкат транзакції.

Функції:

1. Функція, яка буде рахувати кількість вже виконаних замовлення.

Збережені процедури:

1. Процедура, для обчислення статусу оплати для існуючих та нових замовлень.

# **ТЕОРЕТИЧНА ЧАСТИНА**

Сучасні бази даних мають великі обсяги інформації, тому ручна обробка, тобто перегляд і редагування даних в таблицях по черзі, стає досить важливим завданням. З метою оптимізації використання баз даних використовуються запити, які дозволяють здійснювати паралельну обробку даних. Це означає, що можна одночасно внести, змінити або видалити значну кількість записів, а також витягти інформацію з таблиць.

### **Архітектура бази даних SQL Server**

SQL Server - це серверна реляційна СУБД, яка надає єдиний метод доступу клієнтів до бази даних і розподіляє "обов'язки" між клієнтом і сервером. Основними складовими SQL Server є реляційна база даних, мова Transact-SQL та графічні інтерфейси користувача (GUT). Основу бази даних SQL Server становить сервер, або ядро бази даних. Ядро бази даних відповідає за обробку запитів, що надходять від клієнтів, і передачу відповідних результатів клієнтським компонентам. Під час встановлення SQL Server автоматично створює кілька баз даних (чотири системні - master, model, msdb, tempdb і дві користувацькі - pubs і north-wind), кожна з яких зберігається в одному або декількох файлах. У базі даних master знаходиться вся інформація щодо конфігурації та функціонування SQL Server.

Вона включає інформацію про всі облікові записи користувачів, інші бази даних та параметри сервера. База даних "model" є шаблоном для створення інших баз даних і завжди повинна бути присутня в системі. При створенні нової користувацької бази даних SQL Server автоматично створює копію бази даних "model". В базі даних "msdb" зберігається інформація про планування завдань (jobs) і подій (alerts), а також про організацію роботи операторів, які отримують повідомлення. База даних "tempdb", яку створює SQL Server, призначена для збереження тимчасових таблиць. "Tempdb" є глобальним ресурсом, доступним автоматично всім користувачам і створюється кожного разу під час запуску SQL Server. Усі тимчасові таблиці "tempdb" автоматично видаляються при відключенні користувача від сервера. База даних "pubs" входить до складу постачання SQL Server і є освітньою базою даних. База даних "northwind" також входить до комплекту постачання SQL Server і є такою ж освітньою базою даних, як і "pubs", але імітує базу даних Microsoft Access.

Кожна база даних SQL Server складається з різних об'єктів, які використовуються для зберігання, організації та обробки інформації. Серед них можна виділити такі об'єкти бази даних SQL Server, як таблиці, індекси, представлення, обмеження, правила, значення за замовчуванням, тригери, процедури і типи даних.

Основним елементом бази даних є таблиці, що містять всі записи, що належать до бази даних (файли таблиць мають розширення .mdf). У SQL Server існують два типи таблиць - системні та користувацькі. Системні таблиці містять інформацію про сам SQL Server і його об'єкти, тоді як користувацькі таблиці зберігають інформацію з первинних документів. Назви всіх системних таблиць починаються з префікса sys. Кожна таблиця включає файл трансакцій (Mf), який формується автоматично при створенні бази даних і використовується для забезпечення цілісності та відновлення бази даних у випадку помилок. У журналі трансакцій зберігаються нові та попередні значення рядків таблиці при внесенні змін до бази даних.

Індекси, які мають розширення idx, використовуються для оптимізації часу виконання операцій пошуку та вибірки даних із таблиць (перелік індексних файлів визначає користувач). Ці індексні файли формуються на основі порядкових номерів записів, розташованих у послідовності, яку вони матимуть, якщо б вони були відсортовані за конкретними полями.

Представлення (Views) - це SQL-інструкція Select, яка використовується для вибору даних з однієї або кількох таблиць та формування вихідної таблиці. Зазвичай представлення використовуються для відбору даних з певного набору рядків або стовпців з таблиць, об'єднання стовпців із різних таблиць та обчислення сумарних значень на основі даних одного чи кількох стовпців.

Обмеження призначені для забезпечення цілісності даних у таблицях і, як правило, додаються користувачами після створення таблиці. Вони можуть бути визначені на рівні стовпців або на рівні таблиці. SQL Server підтримує п'ять типів обмежень цілісності.

1. Primary Key - Первинний ключ (обмеження за первинним ключем) - забезпечує унікальний ідентифікатор для кожного рядка в таблиці, що не може дорівнювати NULL. Використання обмеження первинного ключа також створює унікальний індекс для таблиці.

2. Foreign Key - Зовнішній ключ (обмеження за зовнішнім ключем) - встановлює зв'язок між одним або декількома стовпцями таблиці та первинним ключем, гарантуючи відповідність вказаних відносин між двома таблицями.

3. Unique - Унікальний (обмеження унікальності) - заважає повторенню значень в будь-якому стовпці.

4. Check - Перевірка (обмеження за значенням) - забезпечує контроль за допустимими значеннями, які можуть бути введені в стовпець таблиці. Наприклад, можна встановити обмеження для перевірки введення значень в межах від 1 до 100.

5. Not Null (обмеження на невизначене значення) - гарантує, що у стовпці не буде допустимих значень NULL.

Правила - це, по суті, ті ж обмеження, але більш детальні. Вони формуються на основі переліку значень і логічних виразів (у практиці їх використовують рідко).

Значення за замовчуванням (за замовчуванням) - встановлює значення, яке автоматично записується в комірку стовпця таблиці, якщо в це поле не було введено жодне значення.

Тригер - це процедура, яка виконується автоматично під час зміни таблиці з використанням інструкції Update, Insert, Delete.

Процедура - це кілька послідовних інструкцій Transact-SQL, які під час її створення компілюються у спеціальний формат і подальше використання для реалізації різноманітних функцій адміністрування бази даних, обробки даних і т.д.

Типи даних - визначають тип інформації, яку можна зберігати в певному стовпці таблиці. Типи даних поділяються на числові цілі, числові з плаваючою крапкою (дробові), текстові, дати і часу, спеціальні.

SQL Server використовує діалект мови структурованих запитів SQL - Transact-SQL, за допомогою якого описують дані БД і зв'язки між ними, виконують операції над даними (додавання таблиць, внесення змін у таблиці, формування запитів для вибірки даних з бази даних і т.д.).

Для доступу і роботи з базою даних SQL Server існує шість графічних інтерфейсів користувача (GUI). Перший - SQL Server Manager, який надає можливість адміністратору системи запускати, зупиняти або тимчасово призупиняти процеси SQL Server. SQL Server Enterprise Manager використовується для виконання всіх адміністративних операцій і створення об'єктів і завдань бази даних, другий - Query Analyzer, який дозволяє використовувати мову Transact-SQL в SQL Server і формувати запити до бази даних, третій - SQL Server Client Configuration Utility, який забезпечує інтерфейс між клієнтом і сервером, четвертий - SQL Server Books Online, що містить 15 книг документації по SQL Server і використовується при необхідності отримати будь-яку інформацію. І, нарешті, SQL Performance Monitor застосовується для налагодження системи оптимальним чином.

### **Поняття про систему безпеки SQL Server**

Дані представляють собою цінний ресурс, і необхідно строго контролювати та регламентувати доступ до неї.

Всі користувачі SQL Server об'єднуються в робочі групи, де члени однієї групи мають однакові права на використання ресурсів. Адміністратор системи формує робочі групи та визначає їхні права на ресурси бази даних. Кожному учаснику групи надається обліковий запис і користувацький запис.

Обліковий запис (ім'я користувача в системі та його пароль) призначений для підключення користувача до сервера SQL Server. Користувацький запис надає право доступу до певних ресурсів сервера (групи). Права доступу у SQL Server реалізовані у вигляді ролей, які визначають дозвіл на виконання конкретних дій над таблицями та іншими об'єктами у базі даних. Перелік ролей зберігається в ядрі системи. Таким чином, кожен користувач перед початком роботи повинен зареєструватись і визначити сервер та базу даних, з якими він може працювати.

Створювати базу даних може адміністратор системи або користувач, якому надано права (роль) адміністратора.

### **Таблиці бази даних**

Однією з ключових аспектів у процесі створення таблиці є визначення типів даних для її полів. Тип даних поля таблиці визначає, якого роду інформацію буде зберігати це поле. Поняття типу даних в SQL Server повністю аналогічне концепції типу даних у сучасних мовах програмування. SQL-сервер підтримує велику кількість різноманітних типів даних: текстові, числові, бінарні.

Давайте здійснимо короткий огляд типів даних у SQL Server.

Для зберігання символьної інформації використовуються символьні типи даних, такі як CHAR (фіксована довжина), VARCHAR (змінна довжина), NCHAR (фіксована довжина для Unicode), NVARCHAR (змінна довжина для Unicode). Останні два призначені для зберігання символів Unicode. Максимальна довжина обмежена 8000 символами (4000 для символів Unicode).

Якщо потрібно зберігати великі об'єми символьних даних (до 2 ГБ), використовуються текстові типи даних TEXT і NTEXT.

До цілих типів даних відносяться INT (INTEGER), SMALLINT, TINYINT, BIGINT. Для зберігання цілих чисел використовуються, відповідно, 4 байти (діапазон від -231 до 231-1), 2 байти (діапазон від -215 до 215-1), 1 байт (діапазон від 0 до 255) або 8 байтів (діапазон від -263 до 263-1). Об'єкти та вирази цільового типу можуть бути використані в будь-яких математичних операціях.

Числа, які мають десяткову точку, називаються нецілочисельними. Нецілочисельні дані поділяються на два види - десяткові та приблизні.

До десяткових типів даних належать DECIMAL [(точність[, масштаб])] і NUMERIC [(точність[, масштаб])]. Типи даних DECIMAL і NUMERIC дають можливість самостійно визначити формат точності числа з плаваючою комою. Параметр точності вказує на максимальну кількість цифр для цього типу даних (до і після десяткової точки усього), а параметр масштабу - максимальну кількість цифр після десяткової точки. За звичайних умов сервер дозволяє вводити не більше 28 цифр для типів DECIMAL і NUMERIC (від 2 до 17 байт).

До приблизних типів даних відносяться FLOAT (точність до 15 цифр, 8 байт) і REAL (точність до 7 цифр, 4 байти). Ці типи представляють дані у форматі з плаваючою комою, тобто для представлення чисел використовуються мантиса і порядок, що забезпечує однакову точність обчислень незалежно від того, наскільки мале або велике значення.

Для зберігання інформації про дату і час призначені такі типи даних, як DATETIME і SMALLDATETIME, які використовують для представлення дати і часу 8 і 4 байти відповідно. Типи даних MONEY і SMALLMONEY забезпечують зберігання інформації про гроші; вони мають точність значень до 4 знаків після коми і використовують 8 і 4 байти відповідно.

Тип даних BIT дозволяє зберігати один біт, який приймає значення 0 або 1. Тип даних TABLE, аналогічно тимчасовим таблицям, забезпечує зберігання набору рядків, призначених для подальшої обробки. Тип даних TABLE може використовуватися тільки для визначення локальних змінних і значень, повертаних користувачеві функціями. Приклад використання типу даних TABLE наведено в лекції, присвяченій функціям користувача.

Тип даних CURSOR необхідний для роботи з об'єктами, такими як курсори, і може бути потрібний тільки для змінних і параметрів процедур, що зберігаються. Курсори SQL Server є механізмом обміну даними між сервером і клієнтом. Курсор дозволяє клієнтським застосункам працювати не з повним набором даних, а лише з однією або декількома рядками. Приклади використання типу даних CURSOR розглядаються в лекціях, присвячених курсорам і зберігаються процедурам..

### **Тригери**

Тригер - це збережена процедура, яка починає виконувати свою роботу у випадку виконання дії, на яку тригер налаштований.

Тригери використовуються для забезпечення цілісності даних та реалізації складної бізнес-логіки. Вони активуються сервером автоматично при спробі зміни даних у таблиці, з якою вони пов'язані. Усі внесені ними зміни вважаються виконаними в межах транзакції, в якій відбулася дія, що викликала спрацьовування тригера. Відповідно, у разі виявлення помилки або порушення цілісності даних може відбутися відкат цієї транзакції.

Момент запуску тригера визначається за допомогою ключових слів BEFORE (тригер активується перед виконанням пов'язаної з ним події; наприклад, перед додаванням запису) або AFTER (після події). У випадку, якщо тригер активується перед подією, він може внести зміни до модифікованого подією запису (зазвичай, за умови, що подія - не видалення запису). Деякі системи управління базами даних обмежують оператори, які можуть бути використані в тригері (наприклад, може бути заборонено вносити зміни в таблицю, на якій "висить" тригер і т.д.).

Крім того, тригери можуть бути пов'язані не з таблицею, а з представленням (VIEW). У цьому випадку вони реалізують механізм "оновлюваного виду". При цьому ключові слова BEFORE і AFTER впливають лише на послідовність виклику тригерів, оскільки сама подія (видалення, вставка або оновлення) не відбувається.

Тригери застосовуються для вирішення завдань підтримки цілісності (коректності) даних, коли з певних причин неможливо використовувати обмеження FOREIGN KEY (або обмеження на значення стовпців), а також для забезпечення безпеки, наприклад, в разі неприпустимих змін в даних. Тригери можуть бути різних типів: DML, DDL і LOGON. DML тригери можуть спрацьовувати при виконанні (після виконання або замість виконання) команд INSERT, UPDATE і DELETE для таблиць або представлень. DDL тригери можуть спрацьовувати при виконанні, після виконання команд CREATE, ALTER, DROP, GRANT, DENY, REVOKE, UPDATE STATISTICS і деяких системних процедур. LOGON тригери активуються після встановлення з'єднання з MS SQL Server..

* 1. **Вимоги до додатку**

Поспілкувавшись з потенційним користувачем виділяємо основні бізнес-вимоги до нашої БД:

Таблиця Люди – повинна мати окремо Прізвище, Ім'я, По-батькові для того, щоб можна було робити смс-сповіщення типу «[Ім'я] [ По-батькові], будь ласка заберіть своє замовлення!». Дата народження, стать потрібна для загальної потреби. Місто повинно бути окремо, для можливості доставки у різні міста(наприклад, усі закази їхали на одне місто). Адреса для доставки. Телефон – для зв’язку у телефонному режимі або месенджерах.

Таблиця Посади – повинна мати Назву посади. Оклад – це мінімальний розмір заробітної плати, що отримує співробітник працюючи на цій посаді. Обов’язки – це список того, що повинен робити співробітник на даній посаді. Вимоги – це те, що потрібно для вступу на цю посаду.

Таблиця Види робіт – повинна мати Назву роботи. Також потрібно мати стовпець Опис, де буде описано саму роботу й мати Вартість – з вартістю робіт.

Таблиця Матеріали - повинна мати Назву матеріалу. Далі обов’язково потрібно вказати, в яку упаковку запаковано матеріал. Не менш важливо описати матеріал, для цього потрібно створити колонку Опис. Надалі також вводиться вартість матеріалу.

Таблиця Бригади – повинна мати дві дати: дату створення й розформування, бо всі бригади – вони створюються й існують певний період часу, мають обмежений перелік робіт, що може виконати лише та бригада.

Таблиця Бригадні роботи – існує для зв’язку бригад та робіт, що може виконати бригада.

Таблиця Замовлення – повинна мати Опис замовлення. Також кожне замовлення має свою вартість. Також, в кожного замовлення є дата початку виконання замовлення й дата завершення. За кожним замовленням закріплюється відповідальний, що приймає й доводить до кінця всі роботи. Закріплюється також замовник.

Таблиця Замовлення потрібні роботи – повинна мати одне замовлення. Замовлення зв’язується з бригадою й роботою, що виконає дана бригада.

Таблиця Оплата замовлень – повинна мати зв’язок з замовленням через Код замовлення. Повинен бути присутній статус оплати, також опис. Також окремим стовпчиком дату проведення замовлення й суму, яка була сплачена. Доречі, оплат за одне замовлення може бути багато.

Таблиця Паспорт – повинна мати зв’язок з людиною через її код. Також важливо, щоб було вказано громадянство, тому що по можливості ми повинні виключити працю нелегалів на будівництві. Вказується обов’язково номер паспорту.

Таблиця Позиції робітників – повинна мати посилання на людину й посаду. Також обов’язково мати дату прийняття на роботу й дату звільнення. Якщо робітник не звільнений, то дата звільнення не вказується. Ставка – може бути як і 1.0, так і 1.5 взалежності від того, скільки роботи виконує певна людина. Також потрібен опис позиції. Також робітник може займати декілька позицій.

Таблиця Список робітників – повинна мати зв’язок із робітником й бригадою.

Таблиця Матеріали для роботи мають зв’язок з роботою й матеріалом.

* 1. **Функціональна специфікація**

Додатком будуть користуватись керівник будівельної фірми, 2 менеджери по продажу, бухгалтер, два касира, і два офісних співробітника будівельної фірми – всього 8 користувачів. Передбачається, що додатком не будуть користуватись більше, ніж 3 користувача. Бухгалтерії потрібно знати лише про оплату, а менеджерам не потрібно знати нічого про оплати. Касири можуть лише додавати значення.

Всі користувачі, можуть додавати значення. При зміні БД повинно бути зафіксованний час заходу, час зміни й хто зайшов.

Один з офісних співробітників буде призначений системний адміністратором, що буде слідкувати за роботою бази даних. Він буде вести облікові записи всіх користувачів.

* 1. **Словник даних для додатку БД Будівельної компанії**

Тепер, після аналізу бізнес-вимог ми створимо за словник даних:

|  |  |
| --- | --- |
| Елемент даних  Таблиці Люди | Опис |
| Прізвище | Прізвище людини |
| Ім'я | Ім'я людини |
| По-батькові | По-батькові людини |
| Дата народження | Дата народження людини |
| Стать | Стать людини |
| Місто | Місто проживання людини |
| Адреса | Адреса людини |
| Телефон | Контактний телефон людини |
| Елемент даних  Таблиці  Посади | Опис |
| Найменування | Назва посади |
| Оклад | Заробітна плата співробітника |
| Обов’язки | Що повинен робити співробітник |
| Вимоги | Що повинен вміти та знати співробітник |
| Елемент даних  Таблиці  Види робіт | Опис |
| Найменування | Назва роботи |
| Опис | Опис роботи |
| Вартість | Вартість роботи |
| Елемент даних  Таблиці  Матеріали | Опис |
| Найменування | Назва матеріалу |
| Упаковка | Упаковка матеріалу |
| Опис матеріалу | Опис матеріалу |
| Вартість | Ціна за матеріал |
| Елемент даних  Таблиці  Бригади | Опис |
| Дата створення | Дата створення бригади |
| Дата розформування | Дата розформування бригади |
| Елемент даних  Таблиці  Бригадні роботи | Опис |
| Код бригади | Код бригади |
| Код виконувальної роботи | Код виконувальної роботи |
| Елемент даних  Таблиці  Замовлення | Опис |
| Опис | Опис замовлення |
| Вартість | Вартість замовлення |
| Дата початку | Дата початку виконання замовлення |
| Дата завершення | Дата завершення виконання замовлення |
| Код людини | Код людини |
| Код робітника | Код робітника |
| Елемент даних  Таблиці  Замовлення потрібні роботи | Опис |
| Код замовлення | Код замовлення |
| Код бригади | Код бригади |
| Код роботи | Код роботи |
| Елемент даних  Таблиці  Оплата замовлень | Опис |
| Код замовлення | Код замовлення |
| Статус оплати | Статус оплати |
| Опис | Опис оплати |
| Дата | Дата оплати |
| Сума | Сума оплати |
| Елемент даних  Таблиці  Паспорт | Опис |
| Код людини | Код людини |
| Громадянство | Громадянство паспорта |
| Номер | Номер паспорту |
| Елемент даних  Таблиці  Позиції робітників | Опис |
| Код людини | Код людини |
| Код посади | Код посади |
| Дата прийняття | Дата прийняття на позицію |
| Дата звільнення | Дата звільнення з позиції |
| Ставка | Ставка позиції |
| Опис | Опис позиції |
| Елемент даних  Таблиці  Список робітників | Опис |
| Код списку | Код списку |
| Код робітника | Код робітника |
| Код бригади | Код бригади |
| Елемент даних  Таблиці  Матеріали для роботи | Опис |
| Код роботи | Код роботи |
| Код матеріалу | Код матеріалу |

* 1. **Логічна модель. ER-діаграми**

Тепер смоделюємо базу даних та створимо ER-діаграму.

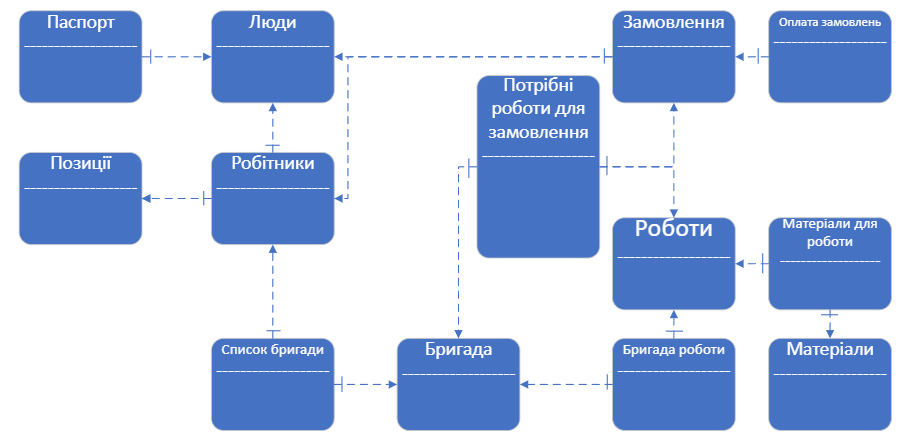


Рис 1. ER-діаграма

* 1. **Об`єкти і атрибути БД з розширеними кодовими полями**

|  |  |
| --- | --- |
| Об`єкти | Атрибути |
| Таблиця Люди | Код |
| Прізвище |
| Ім'я |
| По-батькові |
| Дата народження |
| Стать |
| Місто |
| Адреса |
| Телефон |
| Таблиця  Посади | Код |
| Найменування |
| Оклад |
| Обов’язки |
| Вимоги |
| Таблиця  Види робіт | Код |
| Найменування |
| Опис |
| Вартість |
| Таблиця  Матеріали | Код |
| Найменування |
| Упаковка |
| Опис матеріалу |
| Вартість |
| Таблиця  Бригади | Код |
| Дата створення |
| Дата розформування |
| Таблиця  Бригадні роботи | Код |
| Код бригади |
| Код виконувальної роботи |
| Таблиця  Замовлення | Код |
| Опис |
| Вартість |
| Дата початку |
| Дата завершення |
| Код людини |
| Код робітника |
| Таблиця  Замовлення потрібні роботи | Код |
| Код замовлення |
| Код бригади |
| Код роботи |
| Таблиця  Оплата замовлень | Код |
| Код замовлення |
| Статус оплати |
| Опис |
| Дата |
| Сума |
| Таблиця  Паспорт | Код |
| Код людини |
| Громадянство |
| Номер |
| Таблиця  Позиції робітників | Код |
| Код людини |
| Код посади |
| Дата прийняття на роботу |
| Дата звільнення |
| Ставка |
| Опис |
| Таблиця  Список робітників | Код |
| Код робітника |
| Код бригади |
| Таблиця  Матеріали для роботи | Код |
| Код роботи |
| Код матеріалу |

* 1. **Нормалізація БД**

Нормалізація бази даних — це процес розкладання однієї таблиці (зазвичай, називається "несправною" або "ненормалізованою") на дві чи більше таблиць (нормалізовані), щоб уникнути аномалій та забезпечити ефективність та стабільність бази даних.

Таблиці нормалізуються до певних ступенів, відомих як нормальні форми (1NF, 2NF, 3NF, BCNF і т.д.). Третя нормальна форма (3NF) є однією з таких ступенів, і вона включає в себе дві основні вимоги:

1. Всі неключові атрибути функціонально залежать від первинного ключа.

- Це означає, що відомості в таблиці повинні зберігатися без дублювання, і будь-який неключовий атрибут має функціонально залежати від всього первинного ключа, а не тільки від частини його.

2. Всі неключові атрибути нефункціонально залежать від інших неключових атрибутів.

- Це означає, що немає атрибутів, які функціонально залежать від інших неключових атрибутів.

Третя нормальна форма допомагає підтримувати консистентність та уникати аномалій оновлення, видалення та вставки в базу даних. У даному випадку, структура таблиць, відповідає цим вимогам 3NF, оскільки вона дотримується цих правил нормалізації.

На зараз в нас мається БД в третій нормальній формі, тому подальша нормалізація тут не потребується.

# **2. ПРАКТИЧНА ЧАСТИНА**

### **2.1.Установка Microsoft SQL Server 2019**

1. Запускаємо установку Microsoft SQL Server.

|  |  |
| --- | --- |
|  |  |
| Рис 1. – Вибір установки | Рис 2. – Шлях установки |

1. Обираємо тип Custom.
2. Обираємо шлях.
3. Обираємо stand-alone installation.

|  |  |
| --- | --- |
|  |  |
| Рис 3. – Інсталяція | Рис 4. – Ліцензія |

1. Приймаємо ліцензію.
2. Натискаємо Next.

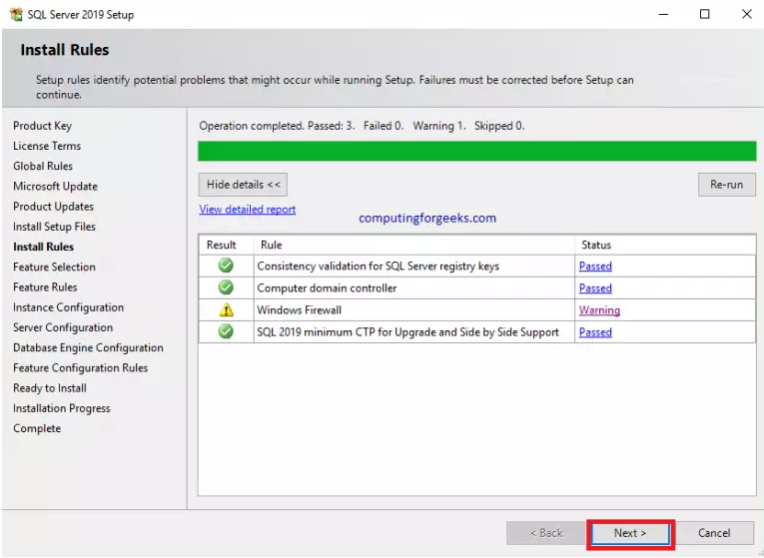


Рис 5. – Ліцензія

1. Усі настройки обираємо за замовчуванням. Переходимо до обрання режиму. Нам потрібен змішаний.

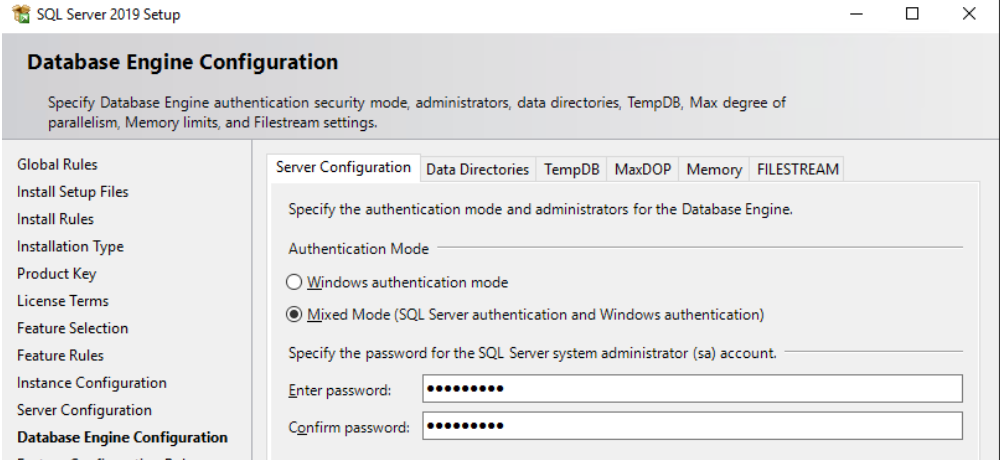


Рис 6. – Змішаний тип

1. Після завершення установки, переходимо до служб. У параметрах робимо вхід через обліковий запис.

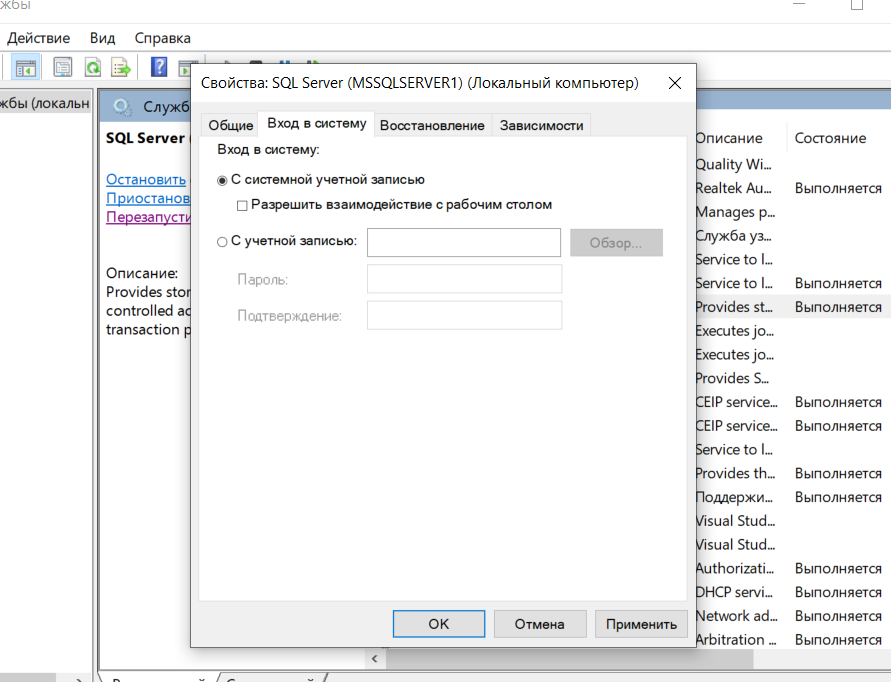


Рис 7. – Обліковий запис

### **2.2. Створення БД**

1. Натискаємо лівою кнопкою миші по База данных- Создать новую БД.
2. Вводимо назву БД.

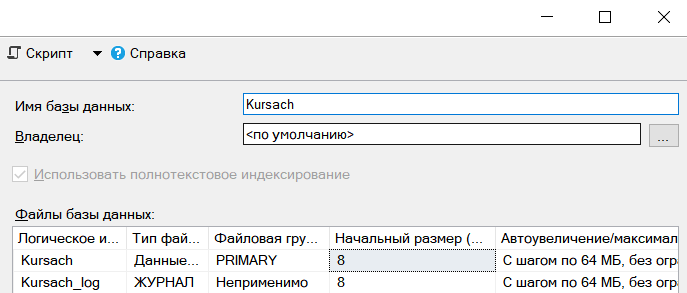


Рис 8. – Назва БД

1. Переходимо до розділу Безопасность- Имена входа- Создать имя входа.

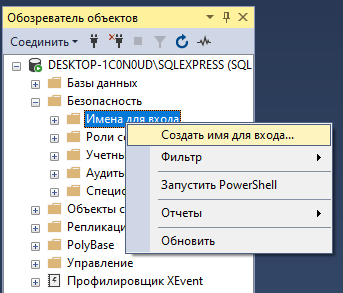


Рис 9. – Назва БД

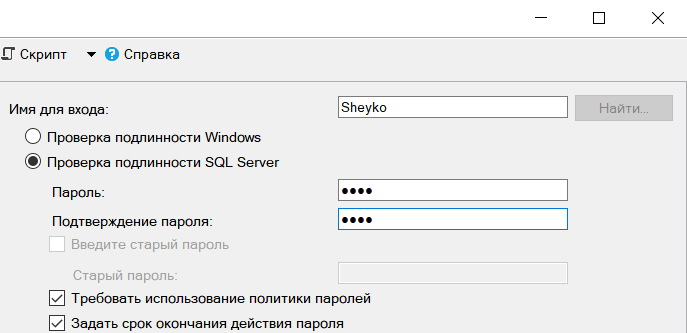


Рис 10. – Ім`я входу

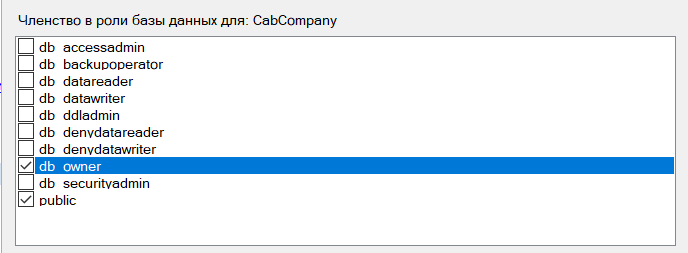


Рис 11. – Вибір ролей для входу

### **2.3. Створення таблиць**

1. По вкладці Таблицы- ЛКМ Создать-Таблица.

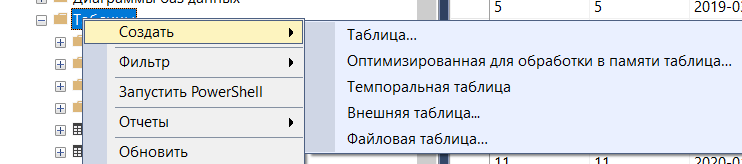


Рис 12. – Створення таблиці

|  |  |
| --- | --- |
|  |  |
| Рис 13. – Таблиця BrigadeEmployee | Рис 14. – Таблиця Brigades |

|  |  |
| --- | --- |
|  |  |
| Рис 15. – Таблиця BrigadeWorks | Рис 16. – Таблиця Employee |

|  |  |
| --- | --- |
|  |  |
| Рис 17. – Таблиця Materials | Рис 18. – Таблиця MaterialsForWork |

|  |  |
| --- | --- |
|  |  |
| Рис 19. – Таблиця OrderPayments | Рис 20. – Таблиця Orders |

|  |  |
| --- | --- |
|  |  |
| Рис 21. – Таблиця Passport | Рис 22. – Таблиця Peoples |

|  |  |
| --- | --- |
|  |  |
| Рис 23. – Таблиця Positions | Рис 24. – Таблиця TheRequiredWorksForOrders |

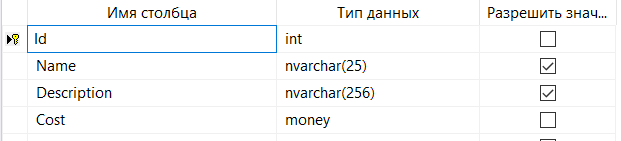


Рис 25. – Таблиця Works

Для кожного поля id зробимо автоінкремент

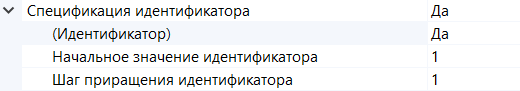


Рис 26. – Cтворення автоінкременту

### **2.4. Обмеження**

Створимо обмеження, яке буде перевіряти щоб вік співробітника не менше 18.

Запит:

ALTER TABLE Kursach.dbo.Peoples

ADD CONSTRAINT EmployeeAge

CHECK (year(getdate()) - year(Date) >= 18);

Перевіримо:

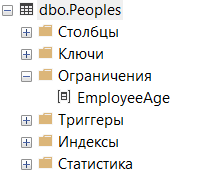


Рис 27. – Обмеження на дату народження

Створимо ще одне обмеження на стать. Можливі лише m-для чоловічої та w-для жіночої. Спробуємо це зробити через вікно обмежень:

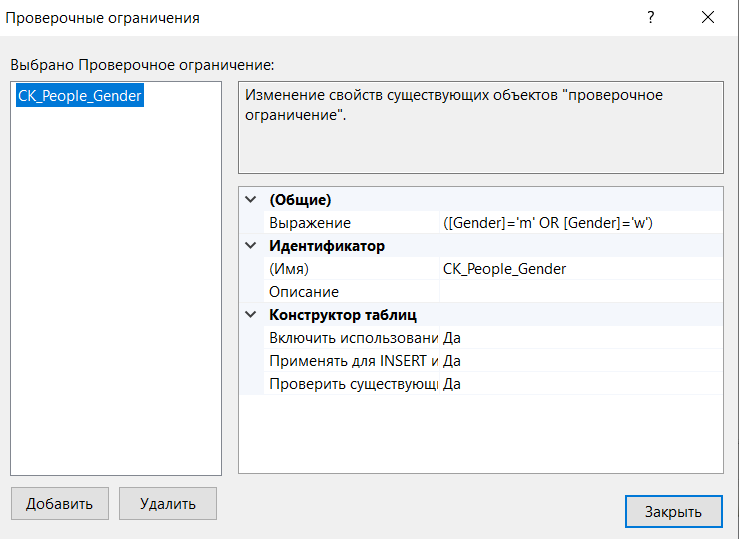


Рис 28. – Обмеження на дату народження

### **2.5. Створення звя’зків**

Для зв'язування таблиць бази даних потрібно викликати контекстне меню на вузлі «Диаграммы базы данных» і обрати пункт «Создать диаграмму базы данных».

Поля id з таблиць passport, peoples, positions, employee, brigadeEmployee, works, materialsForWork, materials, theRequiredWorksForOrders, orders, ordersPayments, brigades та brigadeWorks є первинними ключами.

В таблиці passport поле id\_people - зовнішній ключ для peoples. В таблиці employee поля id\_ people, id\_position - зовнішній ключі для таблиць people та position відповідно. В таблиці brigadeEmployee поля id\_employee, id\_brigade - зовнішній ключі для таблиць employee та brigades відповідно. В таблиці brigadeWorks поля id\_work, id\_brigade - зовнішній ключі для таблиць works та brigades відповідно. В таблиці materialsForWork поля id\_work, id\_material - зовнішній ключі для таблиць works та materials відповідно. В таблиці theRequiredWorksForOrders поля id\_orders, id\_work, id\_brigade - зовнішній ключі для таблиць orders, work та brigade відповідно. В таблиці orders поля id\_people, id\_employee - зовнішній ключі для таблиць peoples та materials відповідно. В таблиці ordersPayments поле id\_order - зовнішній ключ для orders.

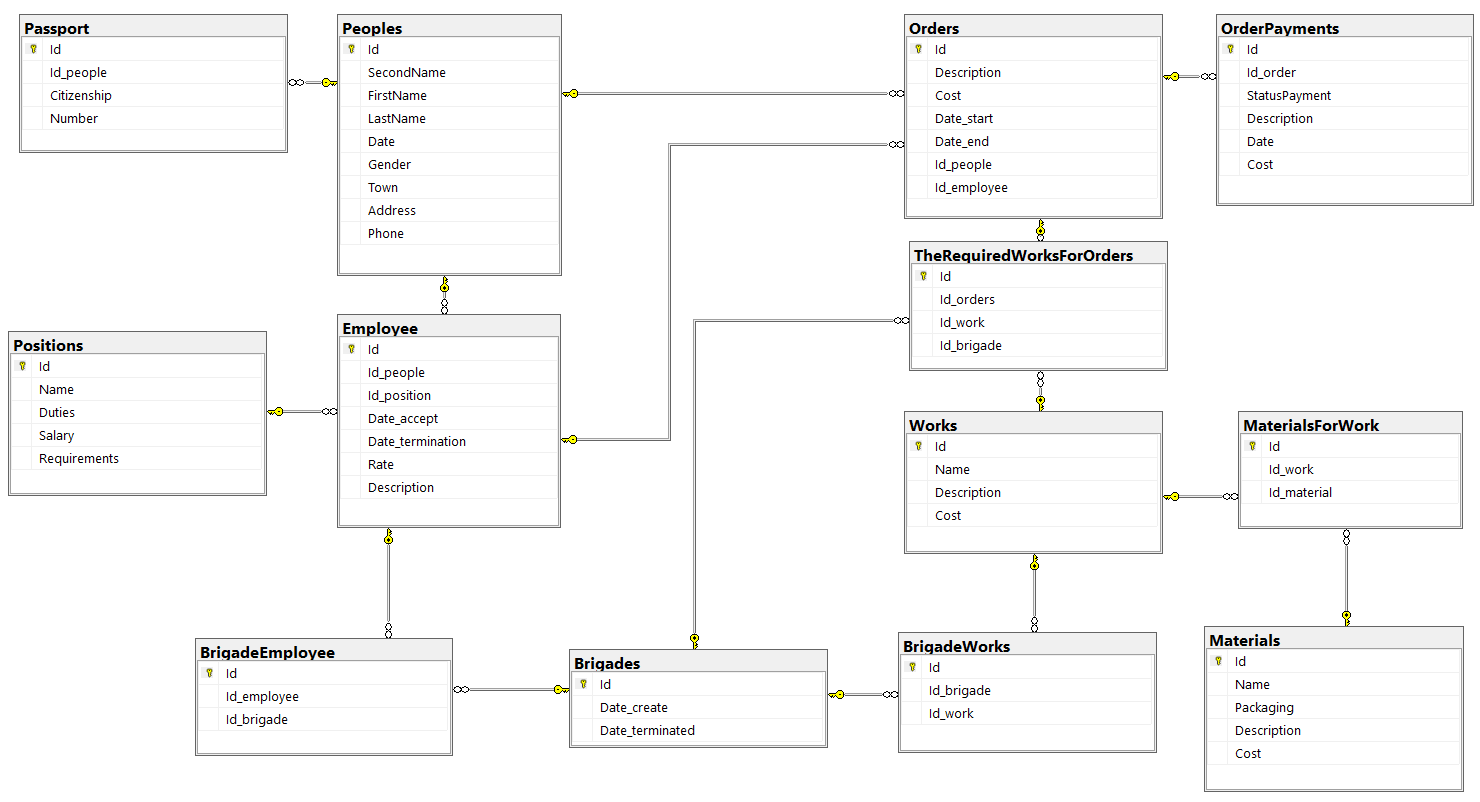


Рис 29. – Схема БД

Маємо:

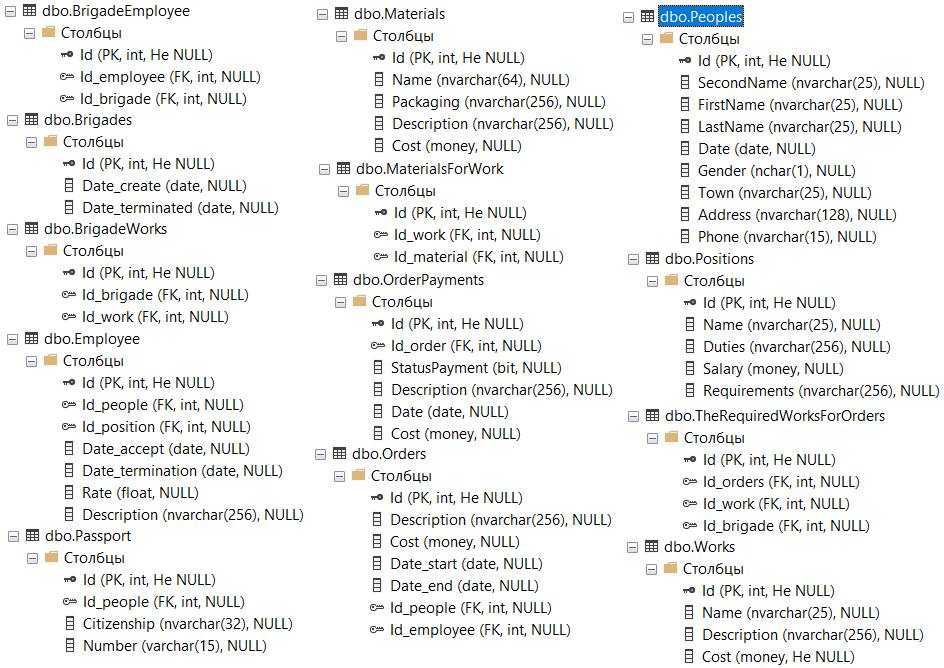


Рис 30. – Зв`язки таблиць

### **2.6. Заповнення таблиць даними**

Створимо запит для нашої бд та заповними таблиці даними:

Таблиця бригади:

INSERT INTO Brigades (Date\_create, Date\_terminated) VALUES

('2023-01-01', '2023-01-10'),

('2023-02-01', '2023-02-10'),

('2023-03-01', '2023-03-10'),

('2023-04-01', '2023-04-10'),

('2023-05-01', '2023-05-10'),

('2023-06-01', '2023-06-10'),

('2023-07-01', '2023-07-10'),

('2023-08-01', '2023-08-10'),

('2023-09-01', '2023-09-10'),

('2023-10-01', '2023-10-10');

Таблиця матеріали:

INSERT INTO Materials (Name, Packaging, Description, Cost) VALUES

('Сталеві листи сплаву', 'Підсилена коробка', 'Довговічні сталеві листи сплаву, придатні для важких застосувань.', 10.99),

('Рулони шовку', 'Бархатна сумка', 'Розкішні рулони шовку, ідеальні для високоякісних модних та текстильних проектів.', 15.99),

('Набір електронних компонентів', 'Герметична упаковка', 'Комплексний набір передових електронних компонентів для проектів із передовими технологіями.', 12.49),

('Старовинні дерев`яні дошки', 'Індивідуальна скриня', 'Аутентичні старовинні дерев`яні дошки, врятовані з історичних будівель, ідеальні для створення рустикального меблів.', 18.75),

('Артізанська колекція паперу', 'Мистецький конверт', 'Ручна артізанська колекція паперу, кожен лист унікальний і ідеально підходить для творчих проектів.', 8.99),

('Труби промислового класу', 'Міцна труба', 'Високоякісні труби промислового класу для сантехнічних та будівельних цілей.', 20.50),

('Асортимент сировини оптом', 'Важка піддона', 'Асортимент сировини оптом для виробництва, надійно упакований на важкому піддоні.', 30.25),

('Ексклюзивні будівельні матеріали', 'Елегантна упаковка', 'Ексклюзивні будівельні матеріали зі стильним дизайном, ідеальні для використання в ремонтних та будівельних проектах.', 5.99),

('Колекція алюмінієвих профілів', 'Піддається вторинному використанню профіль', 'Екологічно чисті алюмінієві профілі, підходять для використання в різних будівельних роботах.', 14.25),

('Модульні системи для зберігання', 'Стійка система контейнерів', 'Універсальні модульні системи для організації та зберігання різноманітних будівельних матеріалів ефективно.', 25.99);

Таблиця люди:

INSERT INTO Peoples (SecondName, FirstName, LastName, Date, Gender, Town, Address, Phone)

VALUES

('Іванов', 'Іван', 'Іванович', '1990-01-01', 'm', 'Київ', 'вул. Шевченка, 10', '+380501234567'),

('Петров', 'Петро', 'Петрович', '1991-02-01', 'm', 'Львів', 'вул. Франка, 22', '+380631234567'),

('Сидоров', 'Сидір', 'Сидорович', '1992-03-01', 'm', 'Одеса', 'вул. Дерібасівська, 5', '+380971234567'),

('Smith', 'John', 'Doe', '1985-05-15', 'm', 'New York', 'Broadway, 123', '+12125551234'),

('Johnson', 'Alice', 'Johnson', '1988-08-20', 'w', 'Los Angeles', 'Hollywood Blvd, 456', '+13235559876'),

('Kim', 'Sung', 'Lee', '1993-04-10', 'm', 'Seoul', 'Gangnam Street, 789', '+821012345678'),

('Garcia', 'Maria', 'Rodriguez', '1980-12-05', 'w', 'Barcelona', 'Rambla, 33', '+34932123456'),

('Müller', 'Hans', 'Schmidt', '1987-06-25', 'm', 'Berlin', 'Alexanderplatz, 15', '+4915112345678'),

('Chen', 'Wei', 'Liu', '1995-09-30', 'm', 'Shanghai', 'Nanjing Road, 88', '+8613812345678'),

('Abe', 'Yuki', 'Sato', '1983-11-17', 'w', 'Tokyo', 'Shibuya Crossing, 2', '+819012345678');

Таблиця позиції:

INSERT INTO Positions (Name, Duties, Salary, Requirements)

VALUES

('Менеджер', 'Управління будівельними проектами', 5000.00, 'Досвід управління будівельними проектами'),

('Бригадир', 'Керівництво робочою бригадою', 7000.00, 'Досвід у будівельних роботах'),

('Робітник', 'Виконання будівельних робіт', 6000.00, 'Досвід у фізичних роботах'),

('Аналітик', 'Аналітика в будівництві', 5500.00, 'Сильні аналітичні навички'),

('Маркетолог', 'Маркетинг у будівництві', 6500.00, 'Досвід у маркетингу в галузі будівництва'),

('Клієнтський менеджер', 'Підтримка клієнтів у будівництві', 4500.00, 'Відмінні навички комунікації'),

('Інженер-будівельник', 'Інженерні роботи в будівництві', 7500.00, 'Технічна освіта, досвід у будівництві'),

('Продажник', 'Продажі будівельних матеріалів', 6000.00, 'Досвід у продажах в будівельній сфері'),

('Кадровик', 'Управління персоналом в будівництві', 5500.00, 'Досвід в управлінні персоналом в будівництві'),

('Дослідник', 'Дослідження в будівництві', 6000.00, 'Досвід в проведенні досліджень в галузі будівництва');

Таблиця роботи:

INSERT INTO Works (Name, Description, Cost)

VALUES

('Укладання плитки', 'Професійне укладання плитки на підлозі та стінах для створення естетичного та міцного покриття.', 5000.00),

('Прокладка труб', 'Систематична прокладка труб та створення водопостачання для забезпечення надійного водопостачання.', 7000.00),

('Реконструкція', 'Проведення всебічних реконструкційних робіт для поліпшення та модернізації будівель.', 10000.00),

('Дизайн екстер`єру', 'Створення індивідуального та привабливого дизайну зовнішнього вигляду будівлі.', 1200.00),

('Встановлення вікон', 'Професійне встановлення вікон для покращення енергоефективності та естетичного вигляду.', 3000.00),

('Утеплення стін та підлоги', 'Утеплення будівельних конструкцій для забезпечення комфорту та ефективності енергоспоживання.', 4500.00),

('Встановлення потолків', 'Створення унікальних та креативних рішень для декорування інтер`єру за допомогою потолків із рисунками.', 8000.00),

('Установка вхідних дверей', 'Професійна установка вхідних дверей для забезпечення безпеки та стильного вигляду.', 2000.00),

('Строительство дому', 'Комплексне будівництво житлового об`єкту, від проектування до завершення будівельних робіт.', 3500.00),

('Вертикальне озеленення', 'Створення зеленого оазису на вертикальних поверхнях будівлі для покращення екології та естетики.', 6000.00);

Таблиця паспорт:

INSERT INTO Passport (Id, Id\_people, Citizenship, Number)

VALUES

(1, 3, 'Ukrainian', 'AB123456'),

(2, 4, 'Ukrainian', 'CD789012'),

(3, 5, 'Ukrainian', 'EF345678'),

(4, 6, 'American', 'GH901234'),

(5, 7, 'French', 'IJ567890'),

(6,8, 'German', 'KL123456'),

(7,9, 'Japanese', 'MN789012'),

(8,10, 'Russian', 'OP345678'),

(9,11, 'Chinese', 'QR901234'),

(10,12, 'Brazilian', 'ST567890');

Таблиця матеріали для роботи:

INSERT INTO MaterialsForWork (Id\_work, Id\_material)

VALUES

(1, 1), (1, 4), (1, 7), (2, 2), (2, 5), (2, 8), (3, 3), (3, 6), (3, 9), (3, 10),

(4, 1), (4, 4), (4, 7), (4, 10), (5, 2), (5, 5), (5, 8), (5, 9), (6, 3), (6, 6),

(6, 8), (7, 2), (7, 4), (7, 7), (7, 9), (8, 1), (8, 5), (8, 6),(8, 10), (9, 3),

(9, 5),(9, 7),(9, 8),(10, 1),(10, 2),(10, 6),(10, 9);

Таблиця бригадні роботи:

INSERT INTO BrigadeWorks (Id\_brigade, Id\_work)

VALUES

(1, 1), (1, 4), (1, 7), (2, 2), (2, 5), (2, 8), (3, 3), (3, 6),

(3, 9), (4, 1), (4, 4), (4, 7), (5, 2), (5, 5), (5, 8), (6, 3),

(6, 6), (6, 9), (7, 1), (7, 4), (7, 7), (8, 2), (8, 5), (8, 8),

(9, 3), (9, 6), (9, 9), (10, 1), (10, 4),(10, 7);

Таблиця робітники:

INSERT INTO Employee (Id\_people, Id\_position, Date\_accept, Date\_termination, Rate, Description)

VALUES

(3, 7, '2023-01-01', NULL, 20.00, ''),

(4, 8, '2023-02-01', NULL, 25.00, ''),

(5, 9, '2023-03-01', NULL, 22.00, ''),

(6, 10, '2023-04-01', NULL, 18.50, ''),

(7, 11, '2023-05-01', NULL, 21.00, ''),

(8, 12, '2023-06-01', NULL, 27.00, ''),

(9, 13, '2023-07-01', NULL, 24.00, ''),

(10, 14, '2023-08-01', NULL, 19.50, ''),

(11, 15, '2023-09-01', NULL, 23.00, ''),

(12, 16, '2023-10-01', NULL, 26.00, '');

Таблиця бригадні робітники:

INSERT INTO BrigadeEmployee (Id\_employee, Id\_brigade)

VALUES

(1, 1),

(2, 2),

(3, 3),

(4, 1),

(5, 2),

(6, 3),

(7, 1),

(8, 2),

(9, 3),

(10, 1);

Таблиця закази:

INSERT INTO Orders (Description, Cost, Date\_start, Date\_end, Id\_people, Id\_employee)

VALUES

('Будівництво житлового будинку', 50000.00, '2023-01-01', '2023-02-28', 3, 1),

('Офісний ремонт', 12000.00, '2023-03-15', '2023-04-30', 4, 2),

('Будівництво апартаментів', 80000.00, '2023-05-01', '2023-09-30', 5, 3),

('Розширення складу', 35000.00, '2023-06-10', '2023-08-20', 6, 4),

('Будівництво роздрібного магазину', 25000.00, '2023-07-01', '2023-09-15', 7, 5),

('Будівництво школи', 60000.00, '2023-09-01', '2024-06-30', 8, 6),

('Ремонт готелю', 45000.00, '2023-10-15', '2024-01-15', 9, 7),

('Будівництво медичного центру', 90000.00, '2024-02-01', '2024-07-31', 10, 8),

('Центр громади', 30000.00, '2024-04-01', '2024-08-31', 11, 9),

('Розкішні квартири', 120000.00, '2024-06-01', '2024-12-31', 12, 10);

Таблиця потрібні роботи для заказу:

INSERT INTO TheRequiredWorksForOrders (Id\_orders, Id\_work, Id\_brigade)

VALUES

(1, 1, 1),

(2, 2, 2),

(3, 3, 3),

(4, 4, 4),

(5, 5, 5),

(6, 6, 6),

(7, 7, 7),

(8, 8, 8),

(9, 9, 9),

(10, 10, 10);

Таблиця оплата заказів:

INSERT INTO OrderPayments (Id\_order, StatusPayment, Description, Date, Cost)

VALUES

(1, 1, 'Payment received', '2023-01-15', 500.00),

(2, 0, 'Waiting for payment', NULL, 800.00),

(3, 1, 'Payment received', '2023-10-15', 1200.00),

(4, 0, 'Waiting for payment', NULL, 950.00),

(5, 1, 'Payment received', '2023-02-28', 670.00),

(6, 0, 'Waiting for payment', NULL, 1100.00),

(7, 1, 'Payment received', '2023-11-05', 890.00),

(8, 0, 'Waiting for payment', NULL, 750.00),

(9, 1, 'Payment received', '2023-03-20', 1120.00),

(10, 0, 'Waiting for payment', NULL, 660.00);

### **2.7. Створення запитів та подань**

1. Створимо подання усіх працівників компанії:

CREATE OR ALTER VIEW AllEmployees AS

SELECT

Employee.Id AS EmployeeId,

Peoples.Id AS PeopleId,

Peoples.SecondName,

Peoples.FirstName,

Peoples.LastName,

Peoples.Date AS PeopleDate,

Peoples.Gender,

Peoples.Town,

Peoples.Address,

Peoples.Phone,

Positions.Id AS PositionId,

Positions.Name AS PositionName,

Positions.Duties,

Positions.Salary,

Positions.Requirements

FROM Employee

INNER JOIN Peoples ON Employee.Id\_people = Peoples.Id

INNER JOIN Positions ON Employee.Id\_position = Positions.Id;

Для того, щоб перевірити дані в БД нам потрібно написати запит:

SELECT \* FROM [Kursach].[dbo].[AllEmployees]

Звернемось до БД й отримаємо дані:

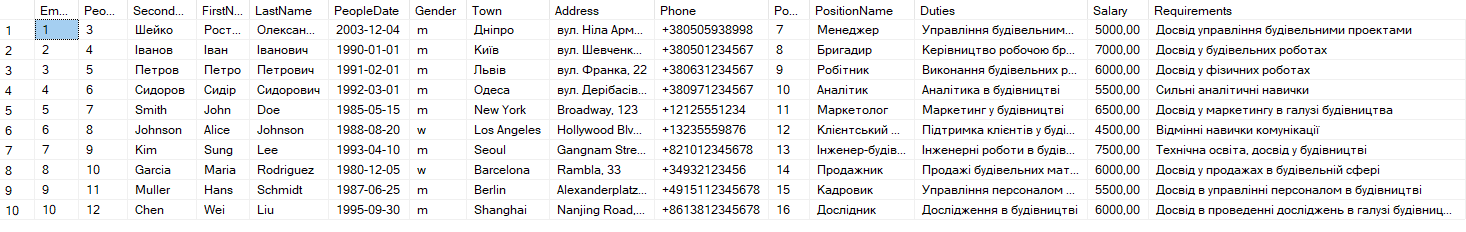


Рис 31. – Дані в подані

1. Створимо подання про роботи з замовленнями.

CREATE OR ALTER VIEW OrderDetails AS

SELECT o.Id AS OrderId, o.Description AS OrderDescription, w.Name AS WorkName, w.Description AS WorkDescription

FROM Orders o

JOIN TheRequiredWorksForOrders rwo ON o.Id = rwo.Id\_orders

JOIN Works w ON rwo.Id\_work = w.Id

Для того, щоб перевірити дані в БД нам потрібно написати запит:

SELECT \* FROM [Kursach].[dbo].[OrderDetails]

Звернемось до БД й отримаємо дані:

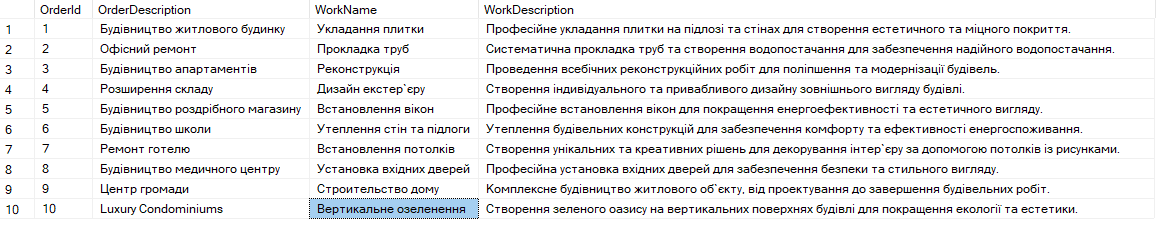


Рис 32. – Дані в подані

1. Створимо подання використаних матеріалів по замовленням.

CREATE OR ALTER VIEW UsedMaterialsInOrders AS

SELECT o.Id AS OrderId, o.Description AS OrderName, m.Name AS MaterialName, m.Description AS MaterialDescription

FROM Orders o

JOIN MaterialsForWork mfw ON o.Id = mfw.Id\_work

JOIN Materials m ON mfw.Id\_material = m.Id;

Для того, щоб перевірити дані в БД нам потрібно написати запит:

SELECT \* FROM [Kursach].[dbo].[UsedMaterialsInOrders]

Звернемось до БД й отримаємо дані:



Рис 33. – Дані в подані

1. Подання загальний звіт про оплату замовлення.

CREATE VIEW OrderPaymentsReport AS

SELECT o.Id AS OrderId, o.Description AS OrderDescription, op.StatusPayment, op.Description AS PaymentDescription, op.Date, op.Cost

FROM Orders o

JOIN OrderPayments op ON o.Id = op.Id\_order;

Для того, щоб перевірити дані в БД нам потрібно написати запит:

SELECT \* FROM [Kursach].[dbo].[OrderPaymentsReport]

Звернемось до БД й отримаємо дані:

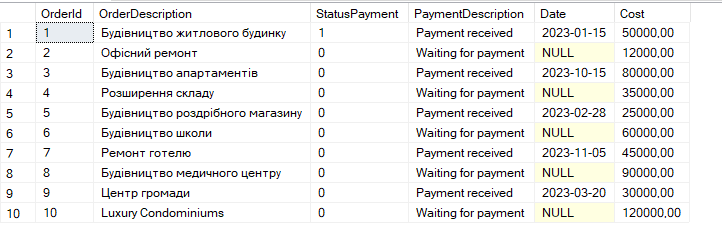


Рис 34. – Дані в подані

1. Запит кількість працівників на кожній посаді.

SELECT Positions.Name as PositionName, COUNT(Employee.Id) as EmployeeCount

FROM Positions

LEFT JOIN Employee ON Positions.Id = Employee.Id\_position

GROUP BY Positions.Name;

Звернемось до БД й отримаємо дані:



Рис 35. – Дані в запиті

1. Запит незакінчених робіт.

Спочатку вставимо в БД незакінчені роботи:

INSERT INTO Works (Name, Description, Cost)

VALUES

('Загадкова скульптура', 'Художній твір, що вразить уяву', 15000.00),

('Футуристичний ландшафт', 'Створення неперевершеного пейзажу', 20000.00),

('Магічний мурал', 'Живописна стінна композиція', 18000.00);

Далі створимо запит:

SELECT \*

FROM Works

WHERE Works.Id NOT IN (SELECT Id\_work FROM TheRequiredWorksForOrders);

Звернемось до БД й отримаємо дані:

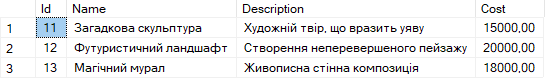


Рис 36. – Дані в запиті

1. Запит використаних матеріалів на певній роботі.

SELECT o.Id AS OrderId, o.Description AS OrderName, m.Name AS MaterialName, m.Description AS MaterialDescription

FROM Orders o

JOIN MaterialsForWork mfw ON o.Id = mfw.Id\_work

JOIN Materials m ON mfw.Id\_material = m.Id

WHERE o.Id = 1;

Звернемось до БД й отримаємо дані:

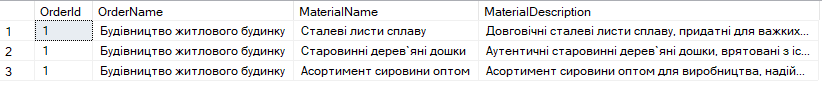


Рис 37. – Дані в запиті

1. Запит список робітників, що входить до певної бригади.

SELECT Peoples.FirstName, Peoples.LastName, Peoples.Address, Peoples.Phone

FROM Peoples

INNER JOIN Employee ON Peoples.Id = Employee.Id\_people

INNER JOIN BrigadeEmployee ON Employee.Id = BrigadeEmployee.Id\_employee

WHERE BrigadeEmployee.Id\_brigade = 1;

Звернемось до БД й отримаємо дані:

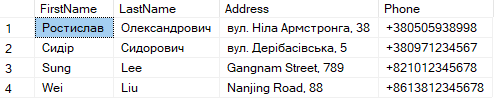


Рис 38. – Дані в запиті

1. Запит сумарна вартість робіт для кожного замовлення.

SELECT Orders.Id AS OrderId, Orders.Description AS OrderDescription, SUM(Works.Cost) AS TotalCost

FROM Orders

INNER JOIN TheRequiredWorksForOrders ON Orders.Id = TheRequiredWorksForOrders.Id\_orders

INNER JOIN Works ON TheRequiredWorksForOrders.Id\_work = Works.Id

GROUP BY Orders.Id, Orders.Description;

Звернемось до БД й отримаємо дані:

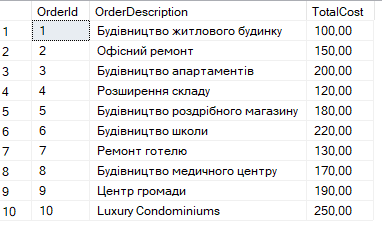


Рис 39. – Дані в запиті

### **2.8. Створення тригерів**

1. Створимо тригер, який заборонятиме видалення таблиці з БД.

Для цього необхідно скористатися SQL-запитом:

CREATE TRIGGER TRG\_RollbackOnDelete

ON DATABASE

FOR DROP\_TABLE

AS

BEGIN

ROLLBACK;

END;

Перевіримо його працездатність, спробувавши видалити таблицю. Але краще створимо нову таблицю TRG\_RollbackOnDelete і спробуємо її видалити:

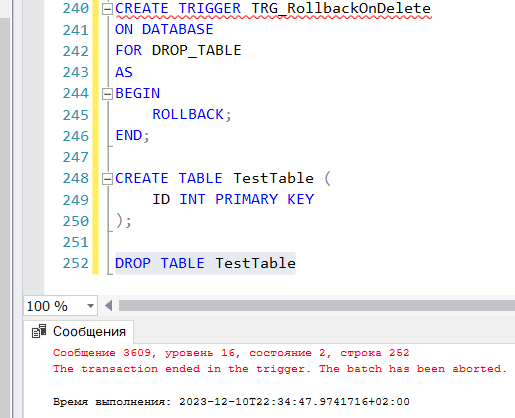


Рис 40. – Тестування

1. Створимо тригер, який заборонятиме видалення будь-яких даних з таблиці для прикладу:

CREATE TRIGGER TRG\_RollbackOnDeleteMaintaining

ON TestTable

AFTER DELETE

AS

BEGIN

ROLLBACK;

END;

Перевіримо:

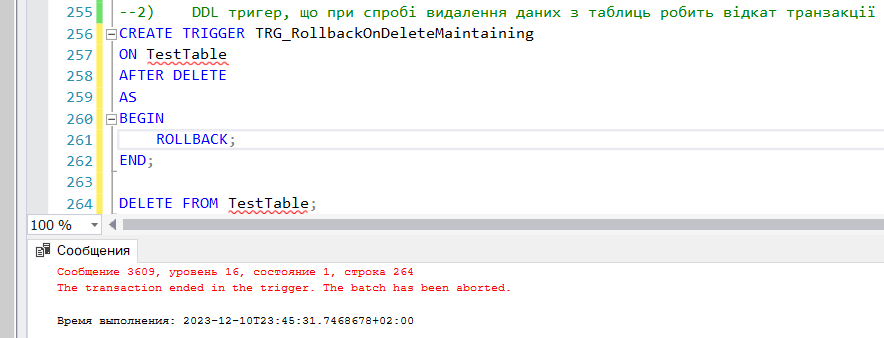


Рис 41. – Тестування

1. Створимо тригер, який буде забороняти видалення ключів:

CREATE TRIGGER TRG\_RollbackOnDeleteKeys

ON DATABASE

FOR DROP\_INDEX, DROP\_PRIMARY\_KEY, DROP\_FOREIGN\_KEY

AS

BEGIN

ROLLBACK;

END;

Перевіримо:

DROP INDEX IX\_TempTestTable\_ID ON TestTable;

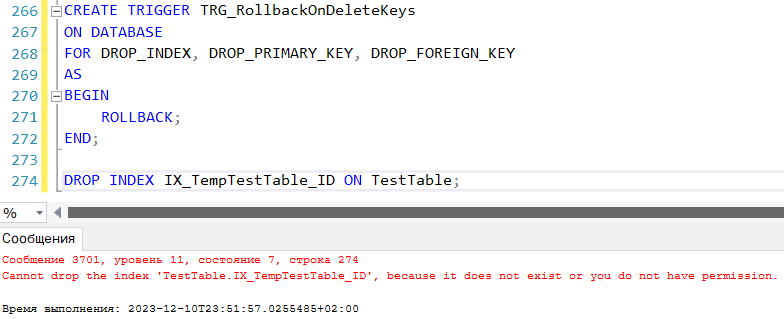


Рис 42. – Тестування

Щоб видалити створені тригери необхідно виконати наступний запит:

DROP TRIGGER name\_trigger ON DATABASE;

### **2.9. Функції**

1. Створимо функцію, яка буду рахувати кількість вже виконаних викликів:

CREATE OR ALTER FUNCTION CalculateAverageCostForBrigade(@brigadeId INT)

RETURNS DECIMAL(18, 2)

AS

BEGIN

DECLARE @averageCost DECIMAL(18, 2)

SELECT @averageCost = AVG(w.Cost)

FROM Works w

INNER JOIN BrigadeWorks bw ON w.Id = bw.Id\_work

WHERE bw.Id\_brigade = @brigadeId;

RETURN @averageCost;

END;

1. Перевіримо:

DECLARE @brigadeId INT = 1; -- фактичний ідентифікатор бригади

-- Виклик функції та отримання результату

DECLARE @averageCost DECIMAL(18, 2);

SET @averageCost = dbo.CalculateAverageCostForBrigade(@brigadeId);

-- Вивід результату

SELECT 'Середня вартість робіт для бригади ' + CAST(@brigadeId AS NVARCHAR) + ': ' + CAST(@averageCost AS NVARCHAR) AS Result;



Рис 43. – Тестування

### **2.10. Процедури**

1. Зробимо процедуру, яка виконує обчислення статусу оплати для існуючих та нових замовлень, а також розраховує вартість оплати відповідно до вартості робіт. Ви можете викликати цю процедуру з програмного коду чи з інтерфейсу бази даних для оновлення інформації замовлень та оплат.

CREATE OR ALTER PROCEDURE GetOrderDetails

AS

BEGIN

-- Вибірка деталей замовлень та оплат за замовлення

SELECT

O.Id AS OrderId,

O.Description AS OrderDescription,

O.Cost AS OrderCost,

O.Date\_start AS OrderStartDate,

O.Date\_end AS OrderEndDate,

OP.StatusPayment,

OP.Description AS PaymentDescription,

OP.Date AS PaymentDate,

OP.Cost AS PaymentCost

FROM Orders O

LEFT JOIN OrderPayments OP ON O.Id = OP.Id\_order;

END;

Перевіримо

EXEC GetOrderDetails;

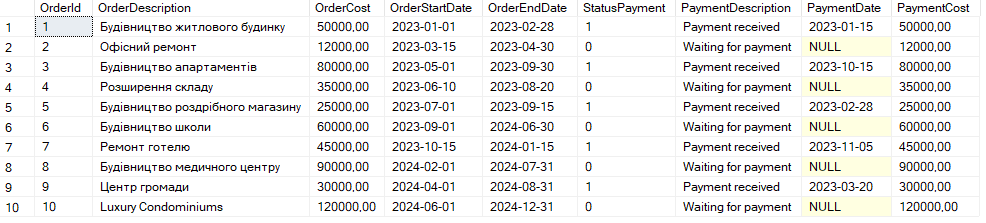


Рис 43. – Тестування процедури

# **ВИСНОВКИ**

В даній курсовій роботі було розроблено реляційну модель БД BuildITCorp та реалізовано її в середовищі MicrosoftSQL Server Management Studio.

В ході роботи було створено 13 таблиць:

* Люди (Код людини, прізвище, ім'я, по батькові, дата народження, стать, місто, адреса, телефон).
* Посади (Код посади, найменування посади, обов'язки, оклад, вимоги).
* Види робіт (Код роботи, найменування, опис, вартість).
* Матеріали (Код матеріалу, найменування, упаковка, опис, вартість).
* Бригади(Код бригади, дата створення, дата розформування).
* Бригадні роботи(Код запису, код бригади, код виконуваної роботи)
* Замовлення (Код замовлення, опис, вартість, дата початку, дата закінчення, код людини, код робітника).
* Замовлення потрібні роботи (Код запису, код замовлення, код роботи, код бригади)
* Оплата замовлень (Код оплати, код замовлення, статус оплати, опис, дата сума)
* Паспорт (Код, Код людини, громадянство, номер).
* Позиції робітників (Код, код людини, код посади, дата прийняття на роботу, дата звільнення, ставка, опис).
* Список робітників(Код списку, код робітника, код бригади)
* Матеріали для роботи(Код, код роботи, код матеріалу).

Для таблиць було створено: діаграма зв’язків, 5 запитів, 4 подань та 3 тригери, 1 функція і 1 процедура.

В результаті було отримано навички роботи з MicrosoftSQL ServerManagement, зокрема у створенні SQL-запитів, а також проектуванні структури БД.

# **СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ**

1. SQL. Быстрый старт. К. Фиайли [Текст]: Курс лекций / С. Д. Кузнецов– М.: Интернет-университет информационных технологий, 2005. – 455 с.

2. Кузин, А.В.Базы данных[Текст] / А.В. Кузин – И.:Академия,  2012. – 69 с.

3. Вшивцев,В. Знакомство с MS SQL Server[Текст] /В. Вшивцев. — И.: Русская редакция, 2009. – 125 с.

4. УстановкаMS SQL Server 2019 [Електронний ресурс] – https://winitpro.ru/index.php/2020/01/15/guide-ustanovka-ms-sql-server-best-practices/

5.[Интерактивный учебник по SQL](http://www.sql-tutorial.ru/ru/content.html)[Електронний ресурс] – http://www.sql-tutorial.ru/

6. MySQL Pocket Reference: SQL Functions and Utilities. 2 ed G. Reese

[Текст]: MySQLPocket Reference, 2007. – 126 с.