МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

Харківський національний університет радіоелектроніки

Кафедра системотехніки

ЗВІТ

з виконання завдань практичного заняття № 1

дисципліни «Проектування високонавантажених систем зберігання даних»

на тему: «ПРОЕКТУВАННЯ ВИСОКОНАВАНТАЖЕНИХ БАЗ ДАНИХ НА ПЛАТФОРМІ СУБД MySQL З ВИКОРИСТАННЯМ ТАБЛИЦЬ ТИПУ MyISAM ТА InnoDB »

Виконав
студент WILDAU - KHARKIV
Якунін Ігор

Перевірив
професор кафедри СТ
Колесник Л.В.

Харків, 2024

1.1 Мета заняття

Набуття практичних навичок з розробки баз даних на платформі MySQL з таблицями типу MyIsam і InnoDB, з урахуванням особливостей роботи високонавантаженої інформаційної системи зберігання даних.

Завдання на самостійну роботу

Обрати предметну область для виконання індивідуальних завдань. Обрана така область – **«Сайт сервісного центру по ремонту техніки»**

Завдання 1.1. Провести аналіз предметної області високонавантаженої інформаційної системи, що складається із серверної (бази даних) і клієнтської (інтерфейсу доступу до бази даних) частин.

Характеристика діяльності підприємства:

1.Сервіс приймає в ремонт техніку, також можлива продажа запчастин зі складу чи під замовлення.

Стан прийнятої техніки кожен клієнт може дізнатися через сайт.

Крім того на інтернет-сайті представлені (виставлені на продаж) деякі типові послуги, а також запчастини та їх ціни. Кожен з них має певну назву, ціну придбання, ціну продажу та одиницю виміру (штуки, кілограми, літри).

2.Для проведення досліджень та оптимізації роботи Сервісу ви намагаєтесь збирати дані з клієнтів. При цьому для вас визначальне значення мають їх стандартні анкетні дані, телефон і адреса електронної пошти для зв'язку.

3. За кожним фактом послуги/продажу ви автоматично фіксуєте клієнта, послугу, запчастини продані, або витрачені в ремонті, кількість, ціну, дату отримання, дату готовності, дату возврата з ремонту, дату та номер накладної служби доставки.

4. Запчастини придбаються у Постачальників(контрагентів, клієнтів), треба фіксувати факт придбання, Постачальника у якого придбали , дата надходження, загальна сума, назва запчастинин, їх кількість, ціну придбання, опис.

5. Треба фіксувати факт оплати як от клієнта (в случає продажу), так і клієнту(постачальнику, в случає купівлі запчастин) – назва клієнта, дата оплати, сума оплати, форма оплати(готівка, оплата з р/р), рахунок по якому здійснена оплата.

6. У разі відправки запчастин (або відремонтовоної техніки) клієнту кур'єрськими службами передбачити можливість вказівки кур'єра, дату відправки клієнту товару та номер накладної кур'єра.

7. Облік співробітників – хто виписав рахунок, хто прийняв техніку в ремонт, кто и що відремонтував, яку отримав заробітну плату.

Також всі дані співробітників, ПІБ, місто, адреса, дата народження, Ід.код, телефон, email.

Перелік програмного забезпечення, необхідного для реалізації трирівневої (триланкової) архітектури «клієнт–сервер» високонавантаженої інформаційної системи

Трирівнева архітектура "клієнт-сервер" є поширеною моделлю для розробки програмного забезпечення, особливо веб-застосунків. Вона передбачає розподіл функціональності на три основні рівні:

1. Клієнтський рівень: Інтерфейс користувача, з яким взаємодіє користувач.
2. Серверний рівень додатків: Логіка бізнес-процесів, взаємодія з базою даних.
3. Сервер бази даних: Зберігання даних (в нашому випадку MySQL).

Переваги трирівневої архітектури:

- Розділення відповідальності: Кожен рівень відповідає за свою частину функціональності, що полегшує розробку, тестування та підтримку.
- Масштабованість: Можливість масштабувати окремі компоненти системи.
- Гнучкість: Легше вносити зміни та розширювати функціональність.
- Повторне використання: Можливість використовувати компоненти в різних проектах.

Типовим прикладом програмних застосунків, які використовують цю архітектуру з MySQL є приклад трирівневої архітектури для веб-магазину:

- Клієнт: Веб-браузер користувача, який відображає каталог товарів, кошик, форму оформлення замовлення.
- Сервер додатків: Веб-сервер на PHP, який обробляє запити від клієнта, взаємодіє з базою даних MySQL для отримання інформації про товари, користувачів та замовлення, генерує HTML-сторінки для клієнта.
- База даних MySQL: Зберігає інформацію про товари, користувачів, замовлення, ціни та інші дані.

Опис основних бізнес-процесів сервісного центру.

1. Клієнти реєструються на сайті, щоб була можливість узнавати статус ремонту техніки, купувати запчастини онлайн, перевіряти раніш створені/оплачені рахунки (історія клієнта). Не зареєстрований відвідувач сайта може проглядати каталог товарів.

2. Підприємство працює с клієнтами (контрагентами, постачальниками, покупцями). Враховуючи те, що вимоги до полів постачальників/покупців однакові, доцільно зробити одну таблицю для всіх контрагентів.

3. Всі дані співробітників, зберігаються в окремої таблиці з їх даними (Прізвище, ім’я, по-батькові, день народження, телефон, email, податковий номер, адрес, посада, статус та пароль для входу на сайт). Робітникам щомісяця нараховується зарплата, сплачуються податки (ще одна таблиця).

4. Техніка поступає в ремонт, її приймають та заповнюють акт приймання, який містить: назву та модель вироба, дату приймання, серійний номер, опис комплектації вироба та опис несправності. Автоматично

формується Замовлення, поки що тільки с номером та датой. Фіксується ім’я співробітника, який прийняв вироб в ремонт.

5. Далі несправний вироб надходить власне в сервіс, де і ремонтується. Тут можливі кілька несправностей, які виправляють декілька співробітників. Після закінчення кожної фази ремонту треба зберігати ім’я співробітника, що проводив ремонт, дату закінчення, суму за ремонт, опис, що було зроблено.

6. При ремонтуванні вироба можливо будут потрібні матеріали, запчастини. Треба організувати склад запчастин, потрібно вести облік таких параметрів, як ім’я запчастини, номер по каталогу (part number), група запчастин, ціна вхідна та вихідна (на продаж), залишок товарів. Також можлива продажа матеріалів. Тут важливо відмітити нюанс придбання однакової запчастини з плином часу, або різних Постачальників за різними цінами.

7. На склад запчастини надходять від Постачальників. Треба здійснювати надходження запчастин, фіксувати від кого прийшли, хто прийняв, дату надходження, кількість та ціну кожної запчастини, загальну суму прибуткової накладної.

8. Треба вести взаєморозрахунки з клієнтами та постачальниками по дате і суме оплат, а також номер приходу або замовлення, по якому були витрачені гроші, або прийшла оптата.

9. Через певний проміжок часу передбачити переведення даних у режим тільки читання, а ще пізніше (не меньш як три роки) видалення застарілих даних.

Опис основних бізнес-функцій сервісного центру.

1. Додавання співробітників.
2. Редагування, блокування (деактівація) співробітників.
3. Додавання Замовників, Постачальників.

4. Редагування, блокування (деактівація) Замовників, Постачальників.
5. Каталог запчастин.
6. Картка запчастини, створення.
7. Картка запчастини, редагування, блокування.
8. Оформлення замовлення на ремонт техніки (приймання в ремонт, передача в сервіс, списання запчастин зі складу).
9. Редагування замовлень.
10. Оформлення приходу запчастин за прибутковою накладною.
11. Редагування приходів.
12. Відомість замовлень за заданий проміжок часу.
13. Знайти замовлення та оплати конкретного покупця.
14. Знайти приходи та оплати конкретного постачальника.
15. Відомість приходов за день/місяц/квартал/рік.
16. Знайти замовлення, де брав участь конкретний співробітник.
17. Відомость по виплаті заробітной плати робітникам.
18. Знайти списання матеріалів по кожному замовленню.
19. Відомість по залишкам запчастин з урахуванням зарезервованих.
20. Рух (прихід, витрата) за кожним товаром у заданий проміжок часу.
21. Рух коштів по банку.
22. Рух коштів (готівка) по касі.
23. Блокування замовлень, приходів, оплат пакетом за певний проміжок часу.
24. Не оплачені замовлення.
25. Замовники – боржники.
26. Видалення застарілих даних (більш як 3 роки, крім робітників).

Бізнес-функції інтерфейсу клієнтської частини високонавантаженої інформаційної системи для повнотекстового пошуку з різними параметрами, обумовленими предметною областю.

1. Повнотекстовий пошук в замовленнях (приймання, ремонт техніки).
2. Повнотекстовий пошук в оплатах.
3. Повнотекстовий пошук в замовниках, постачальниках.
4. Повнотекстовий пошук в описі запчастин.

5. Повнотекстовий пошук в списанні запчастин.

Таблиця 1.1 – Перелік статусів можливих типів користувачів

№	Користувач	Статус	Доступ до даних
1	Адміністратор	admin	Має доступ до запису й читання для всіх таблиць БД, а також каскадного видалення деяких даних
2	Бухгалтер	buh	Має доступ до запису й читання для всіх таблиць БД
3	Менеджер	manager	Має доступ до запису й читання таблиць customers, orders, orders_goods, orders_received, orders_repair
4	Зареєстрований користувач	client	Має доступ до запису й читання таблиць customers, orders, orders_goods, читання – orders_received, orders_repair
5	Незареєстрований користувач	немає	Має доступ до читання таблиць goods, goods_dscr

Таблиця 1.2 – Перелік елементів інтерфейсу й бізнес-функцій (приклад декількох функцій)

№	Елементи інтерфейсу	Бізнес-функція високонавантаженої інформаційної системи
1	Текстове поле «phone» або «email» Текстове поле «Key» Кнопка «Вхід» Відображення імені користувача	Використовуються для аутентіфікації, авторізації. На телефон або пошту надходить ключ, або посилання для входу, щоб увійти в систему й змінити свій статус «незареєстрований клієнт» на «зареєстрований клієнт». Після входу в систему ім’я користувача відображається, наприклад, в правому верхньому куті сайту

2	Поля: customer, goods, quantity	Оформлення замовлення на продаж запчастин, доступна для зареєстрованих користувачів on-line
3	Поля: customer, brand, model_name, sn, equipment, fault_description	Оформлення замовлення на ремонт техніки, доступна менеджеру, клієнту для чтения його замовлення
4	Поля: order_id, customer, employee, date_ready, price_repear, description	Оформлення ремонту техніки, доступна менеджеру, частково клієнту для чтения його замовлення
5	order_id, goods, quantity, price_out, description	Списання запчастин в ремонт, доступна менеджеру склада
6	customer, date_arrival, total, goods, quantity, description	Приход запчастин в ремонт, доступна бухгалтеру, менеджеру склада
7	customer, date_pay, pay_sum, arrival_id, order_id pay_form, description	Оплата, доступна бухгалтеру

Таблиця 1.3 – Порівняльний аналіз таблиць типу MyIsam і InnoDB

№	Опис	MyISAM	InnoDB
1	Підтримка транзакцій – Немає /Так	Немає	Так
2	Підтримка зв’язків за зовнішніми ключами – Немає /Є	Немає	Є

3	Підтримки посилальної цілісності зв’язків для інструкцій UPDATE, DELETE, INSERT	Немає	UPDATE DELETE
4	Блокування – на рівні таблиць / на рівні записів / немає блокування	на рівні таблиць	на рівні записів
5	Одночасні запити до різних записів однієї таблиці – повільніше / швидше.	повільніше	швидше
6	Під час змішаного навантаження за запитами SELECT, UPDATE, DELETE, INSERT – повільніше / швидше	повільніше	швидше
7	Однотипні операції INSERT – повільніше / швидше	швидше	повільніше
8	Однотипні операції SELECT – повільніше / швидше	швидше	повільніше
9	Запит Count(*) – повільніше / швидше	швидше	повільніше
10	Взаємне блокування операцій (Deadlock) – можлива / неможлива	неможлива	можлива
11	Підтримка повнотекстового пошуку – Немає /Є	Є	Є
12	Підтримка індексування полів у запитах – Немає /Є	Є	Є
13	Можливі типи індексів	***B-tree, Full-text, Geospatial***	***B-tree, Full-text, Geospatial,Clustered indexes, Adaptive Hash Index***
14	Можливість бінарного копіювання таблиць – Немає /Є	Є	немає
15	Розмір таблиці БД – укажіть максимальний розмір даних. Storage limits	256Tb	64Tb
16	Можливість відновлення у випадку збою	Немає*	Є
17	СУБД створює за замовчуванням для таблиці MyIsam / InnoDB (укажіть кількість файлів, їх розширення, папки для зберігання та їх призначення)**	Кожна `MyISAM` таблиця зберігається на диску у трьох файлах***	File-Per-Table, наприклад, status.ibd

* Статичний формат `MyISAM` таблиць є найпростішим і найбезпечнішим, якщо ваш комп'ютер виходить з ладу під час запису сервера MySQL, **myisamchk** може легко визначити, де починається і закінчується кожен рядок, тому він зазвичай може відновити всі рядки, крім частково записаного.

** "c:\ProgramData\MySQL\MySQL Server 8.0\Data\your_name_db" – шлях до файлів БД

***Кожна `MyISAM` таблиця зберігається на диску у трьох файлах. Файли мають імена, які починаються з імені таблиці та мають розширення, що вказує на тип файлу. Файл даних має розширення `.MYD`(`MYData`). Індексний файл має розширення `.MYI` (`MYIndex`). Визначення таблиці зберігається в словнику даних MySQL.Також створюються файли метаданих типа JSON (наприклад, status_818.sdi - Serialized Dictionary Information)

Завдання 1.2. Розробити серверну частину високонавантаженої інформаційної системи з таблицями СУБД MySQL типу MyIsam.

– створення логічної моделі бази даних у вигляді ER-діаграми згідно з нотацією IDEF1X;

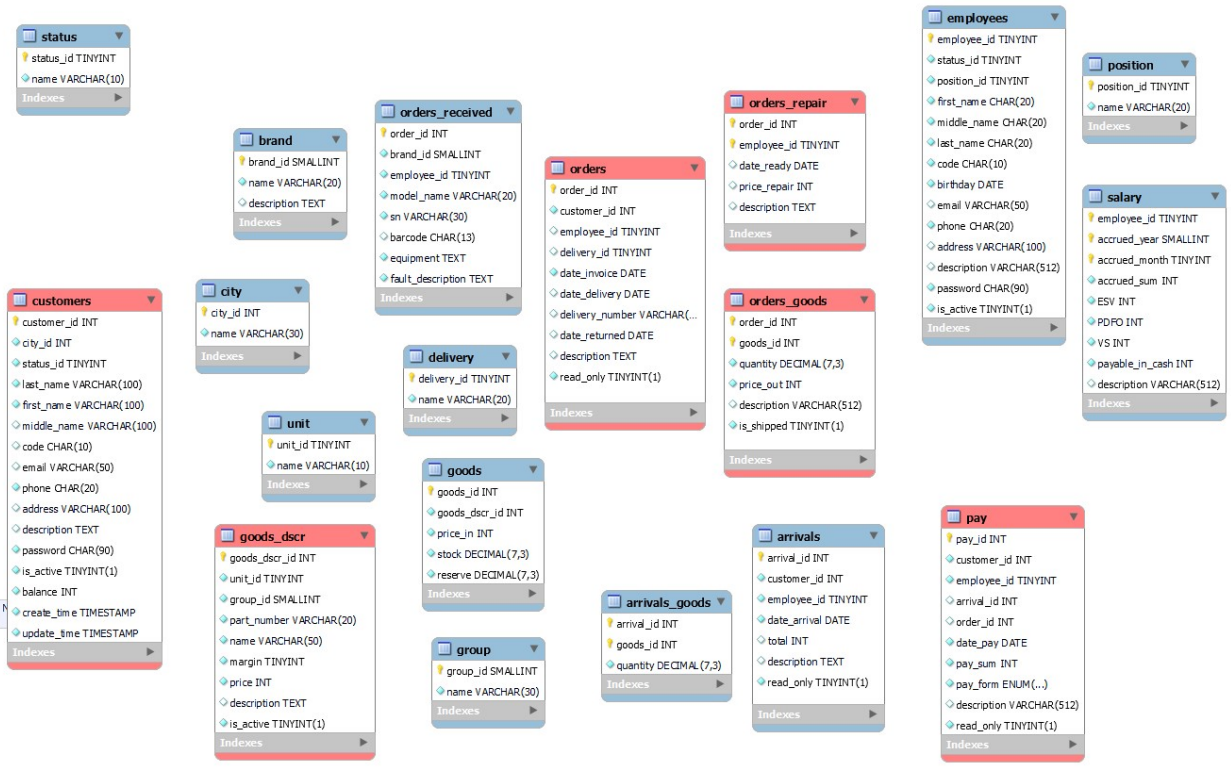

Рис. 1.1 Схема БД з таблицями типу MyIsam

– створення бази даних з таблицями типу MyIsam, лістинг коду

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERRO
R_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema ss_myisam
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `ss_myisam` ;

-- -----------------------------------------------------
-- Schema ss_myisam
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `ss_myisam` DEFAULT CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci ;
USE `ss_myisam` ;

-- -----------------------------------------------------
-- Table `ss_myisam`.`customers`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`customers` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`customers` (
  `customer_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `city_id` INT UNSIGNED NOT NULL,
  `status_id` TINYINT UNSIGNED NOT NULL,
  `last_name` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `first_name` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `middle_name` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `code` CHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'Ід.Код -
ціфри',
  `email` VARCHAR(50) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `phone` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'номер уникален (по ньому вхід) +380 (067) 123 45 67',
  `address` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
```

```
  `password` CHAR(90) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'Хеш пароля (для входа online)',
  `is_active` TINYINT(1) UNSIGNED NOT NULL DEFAULT 1 COMMENT 'активен -1, ні - 0',
  `balance` INT NOT NULL DEFAULT 0 COMMENT 'тут зберігається поточний баланс контрагента',
  `create_time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `update_time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`customer_id`),
  UNIQUE INDEX `phone_UNIQUE` (`phone` ASC) VISIBLE)
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`employees`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`employees` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`employees` (
  `employee_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `status_id` TINYINT UNSIGNED NOT NULL,
  `position_id` TINYINT UNSIGNED NOT NULL,
  `first_name` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `middle_name` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `last_name` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `code` CHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'налог. номер',
  `birthday` DATE NOT NULL,
  `email` VARCHAR(50) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `phone` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'формат +380 (067) 123 45 67',
  `address` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `description` VARCHAR(512) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL
COMMENT 'якийсь опис співробітника',
  `password` CHAR(90) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'Хеш пароля',
  `is_active` TINYINT(1) UNSIGNED NOT NULL DEFAULT 1 COMMENT 'активен - 1, ні - 0',
  PRIMARY KEY (`employee_id`),
  UNIQUE INDEX `phone_UNIQUE` (`phone` ASC) VISIBLE)
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
```

```
COLLATE = utf8mb4_unicode_ci
COMMENT = '98BFDA';


-- -----------------------------------------------------
-- Table `ss_myisam`.`orders`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`orders` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`orders` (
  `order_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `customer_id` INT UNSIGNED NOT NULL,
  `employee_id` TINYINT UNSIGNED NULL COMMENT 'Поле заповнюється при видачіі. При створенні замовлення поле is null',
  `delivery_id` TINYINT UNSIGNED NULL,
  `date_invoice` DATE NOT NULL COMMENT 'дата створення рахунка по итогу прийняття техніки в ремонт',
  `date_delivery` DATE NULL COMMENT 'відвантажено, отримано кур\'єром',
  `delivery_number` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `date_returned` DATE NULL COMMENT 'повернуто клієнту',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `read_only` TINYINT(1) NOT NULL DEFAULT 0 COMMENT '0-доступен, 1-только чтение',
  PRIMARY KEY (`order_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`pay`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`pay` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`pay` (
  `pay_id` INT NOT NULL AUTO_INCREMENT,
  `customer_id` INT UNSIGNED NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `arrival_id` INT UNSIGNED NULL,
  `order_id` INT UNSIGNED NULL,
  `date_pay` DATE NOT NULL COMMENT 'дата оплати',
  `pay_sum` INT NOT NULL DEFAULT 0 COMMENT 'сума оплати У КОПІЙКАХ(!) (для прихода < 0)',
```

```
  `pay_form` ENUM('готівка', 'оплата на р.р.') CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci'
NOT NULL DEFAULT 'оплата на р.р.' COMMENT '\'готівка\', \'оплата на р.р.\'',
  `description` VARCHAR(512) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `read_only` TINYINT(1) NOT NULL DEFAULT 0 COMMENT '0-доступно, 1-только читання',
  PRIMARY KEY (`pay_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`orders_received`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`orders_received` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`orders_received` (
  `order_id` INT UNSIGNED NOT NULL COMMENT '1:1 order_id',
  `brand_id` SMALLINT UNSIGNED NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `model_name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `sn` VARCHAR(30) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `barcode` CHAR(13) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'EAN-
13',
  `equipment` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'У
який комплектації прибув прибор, зовнішні пошкодження',
  `fault_description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL
COMMENT 'опис несправності',
  PRIMARY KEY (`order_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`brand`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`brand` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`brand` (
  `brand_id` SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
```

```
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'опис бренду',
  PRIMARY KEY (`brand_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`orders_repair`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`orders_repair` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`orders_repair` (
  `order_id` INT UNSIGNED NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `date_ready` DATE NULL COMMENT 'Дата готовності після ремонту',
  `price_repair` INT NULL DEFAULT 0 COMMENT 'вартість викононой роботи У КОПІЙКАХ(!)',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'що зроблено',
  PRIMARY KEY (`employee_id`, `order_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`goods`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`goods` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`goods` (
  `goods_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `goods_dscr_id` INT UNSIGNED NOT NULL,
  `price_in` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'вхідна ціна У КОПІЙКАХ(!)',
  `stock` DECIMAL(7,3) UNSIGNED NOT NULL DEFAULT 0 COMMENT 'залишок запчастин',
  PRIMARY KEY (`goods_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;
```

```
-- -----------------------------------------------------
-- Table `ss_myisam`.`group`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`group` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`group` (
  `group_id` SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(30) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  PRIMARY KEY (`group_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`unit`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`unit` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`unit` (
  `unit_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'одиниця виміру',
  PRIMARY KEY (`unit_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`delivery`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`delivery` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`delivery` (
  `delivery_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT COMMENT 'Назва кур\'єра доставки',
  `name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  PRIMARY KEY (`delivery_id`),
  UNIQUE INDEX `name_UNIQUE` (`name` ASC) VISIBLE)
ENGINE = MyISAM
```

```
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`orders_goods`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`orders_goods` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`orders_goods` (
  `order_id` INT UNSIGNED NOT NULL,
  `goods_id` INT UNSIGNED NOT NULL,
  `quantity` DECIMAL(7,3) UNSIGNED NOT NULL DEFAULT 1,
  `price_out` INT UNSIGNED NOT NULL DEFAULT 0,
  `description` VARCHAR(512) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  PRIMARY KEY (`order_id`, `goods_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`status`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`status` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`status` (
  `status_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'admin, buh, manager, client - права доступу',
  PRIMARY KEY (`status_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`arrivals`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`arrivals` ;
```

```
CREATE TABLE IF NOT EXISTS `ss_myisam`.`arrivals` (
  `arrival_id` INT NOT NULL AUTO_INCREMENT,
  `customer_id` INT UNSIGNED NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `date_arrival` DATE NOT NULL,
  `total` INT UNSIGNED NULL DEFAULT 0 COMMENT 'Сумма приходной накладной У КОПІЙКАХ(!)',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `read_only` TINYINT(1) NOT NULL DEFAULT 0 COMMENT '0-доступен, 1-только чтение',
  PRIMARY KEY (`arrival_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`arrivals_goods`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`arrivals_goods` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`arrivals_goods` (
  `arrival_id` INT UNSIGNED NOT NULL,
  `goods_id` INT UNSIGNED NOT NULL,
  `quantity` DECIMAL(7,3) UNSIGNED NOT NULL DEFAULT 1,
  PRIMARY KEY (`arrival_id`, `goods_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`goods_dscr`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`goods_dscr` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`goods_dscr` (
  `goods_dscr_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `unit_id` TINYINT UNSIGNED NOT NULL,
  `group_id` SMALLINT UNSIGNED NOT NULL,
  `part_number` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `name` VARCHAR(50) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `margin` TINYINT UNSIGNED NOT NULL DEFAULT 30 COMMENT 'наценка, у процентах',
```

```
  `price` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'ціна продажу У КОПІЙКАХ(!)',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1 COMMENT '1-активен, 0-блок',
  UNIQUE INDEX `part_number_UNIQUE` (`part_number` ASC) VISIBLE,
  PRIMARY KEY (`goods_dscr_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`position`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`position` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`position` (
  `position_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'посада',
  PRIMARY KEY (`position_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`city`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`city` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`city` (
  `city_id` INT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(30) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  PRIMARY KEY (`city_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_myisam`.`salary`
```

```
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_myisam`.`salary` ;

CREATE TABLE IF NOT EXISTS `ss_myisam`.`salary` (
  `employee_id` TINYINT UNSIGNED NOT NULL COMMENT 'рік, за який нарахована ЗП',
  `accrued_year` SMALLINT UNSIGNED NOT NULL DEFAULT 2024 COMMENT 'рік нарахування ЗП > 2023',
  `accrued_month` TINYINT UNSIGNED NOT NULL COMMENT 'місяц, за який нарахована ЗП, between 1 and 12',
  `accrued_sum` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'нараховано сума ЗП в копійках',
  `ESV` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'єдиний соціальний внесок',
  `PDFO` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'податок на доход, ПДФО',
  `VS` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'військовий збір',
  `payable_in_cash` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'до виплати на руки',
  `description` VARCHAR(512) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  PRIMARY KEY (`accrued_year`, `accrued_month`, `employee_id`))
ENGINE = MyISAM
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

– заповнення основних таблиць даними.

Для заповнення таблиц фейковими даними я використал програму на мові Python (задіяв бібліотеки pymysql, faker, random).

Приклад програми, що генерує 20 записів у таблиці employees:

```python
from faker import Faker
import pymysql.cursors
from config24 import host, user, password, db_name2
import random
from passlib.hash import pbkdf2_sha256

try:
    connection = pymysql.connect(
        host=host,
        port=3306,
        user=user,
        password=password,
        database=db_name2,
        charset='utf8mb4',
        cursorclass=pymysql.cursors.DictCursor
    )
```

```
    print(f"{db_name2} - successfully connected...")
    print("=" * 30)
except Exception as ex:
    print(f"{db_name2} - connection refused...")
    print(ex)

if __name__ == '__main__':
    fake = Faker("uk_UA")  # створюємо екземпляр з певною локалізацією
    fake_ru = Faker("ru_RU")

    with connection.cursor() as cursor:
      for _ in range(20):
        status_id = random.randint(1, 6)
        position_id = random.randint(1, 6)
        full_name = fake.full_name()  # 'Бабенко Оксенія Геннадіївна'
        last_name = full_name.split()[0]
        first_name = full_name.split()[1]
        middle_name = full_name.split()[2]
        code = fake.ssn()
        birthday = fake_ru.date_of_birth(minimum_age=20, maximum_age=60)
        email = fake.free email()
        phone = fake.phone_number()
        address = fake.address()
        dscr = fake.sentence(nb_words=45, variable_nb_words=False)
        if len(dscr) > 299:
            dscr = dscr[:299]

        password = pbkdf2_sha256.hash('password')
        print(status_id, position_id, last_name, first_name, middle_name,
code, birthday, email, phone, address, password, dscr)

        sql = "INSERT INTO `employees` (`status_id`, `position_id`,
`first_name`, `last_name`, `middle_name`, `code`, `birthday`, `email`,
`phone`, `address`, `description`, `password`) VALUES (%s, %s, %s, %s, %s,
%s, %s, %s, %s, %s, %s, %s)"

        cursor.execute(sql, (status_id, position_id, first_name, last_name,
middle_name, code, birthday, email, phone, address, dscr, password))
```

Скріншоти, які демонструють кількість записів, введених у таблиці MyISAM приведени на рис.1.2 – 1.11

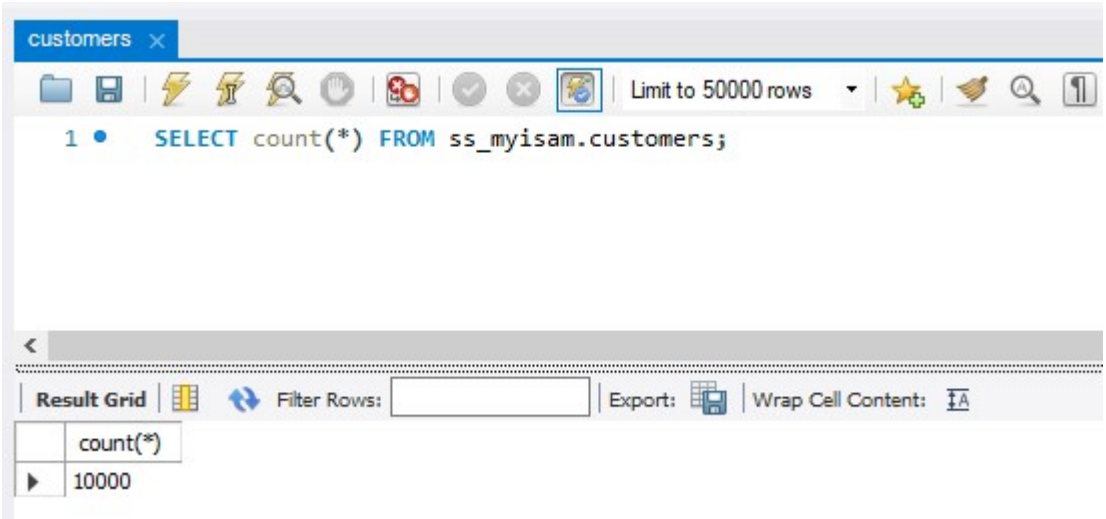

Рис. 1.2 Скріншот введення даних у таблицю customers

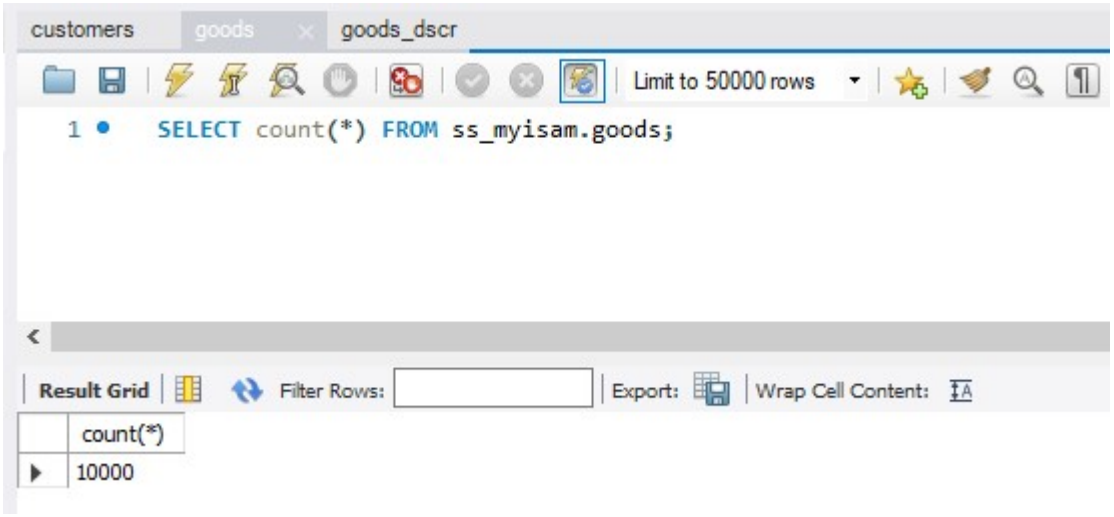

Рис. 1.3 Скріншот введення даних у таблицю goods

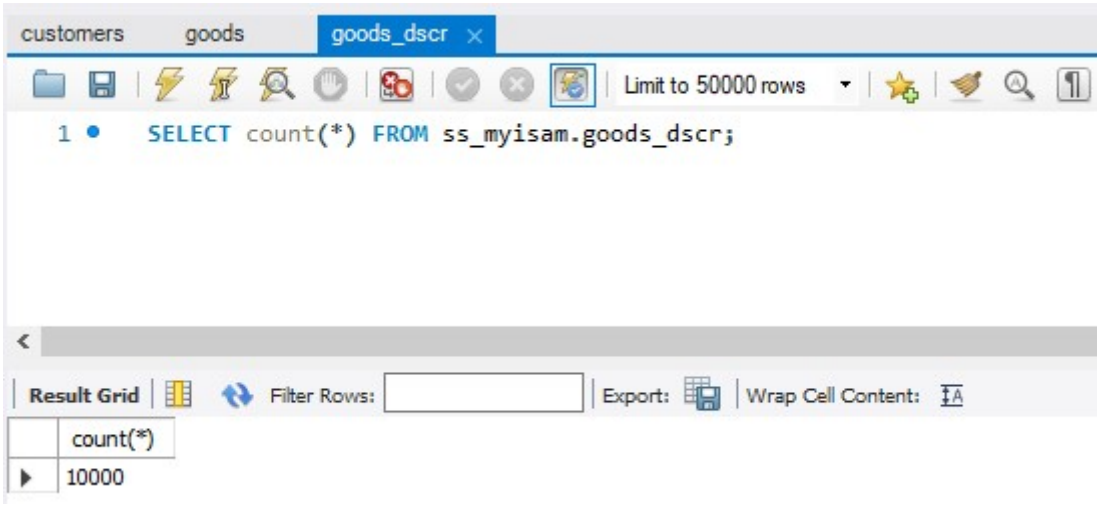

Рис. 1.4 Скріншот введення даних у таблицю goods_dscr

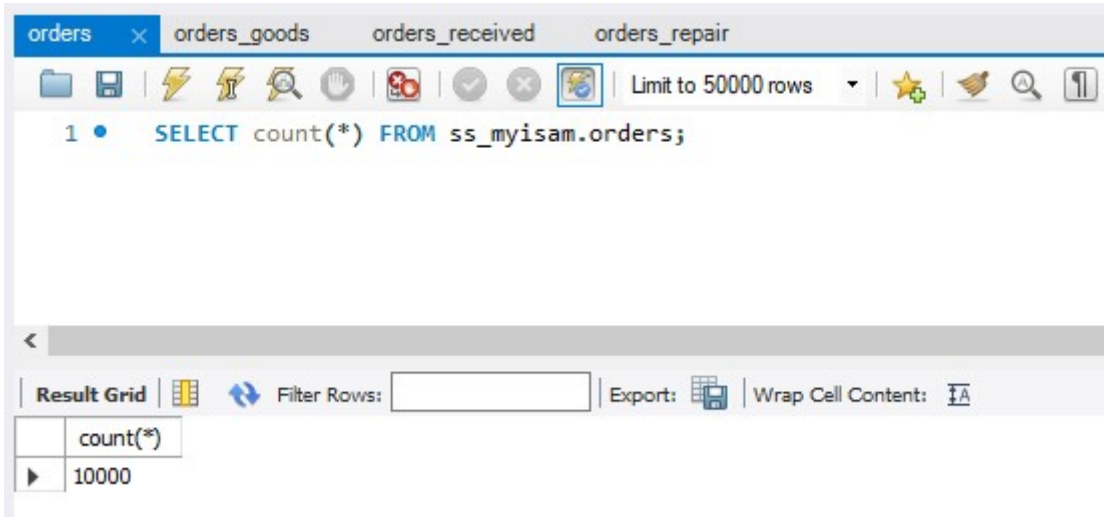

Рис. 1.5 Скріншот введення даних у таблицю orders

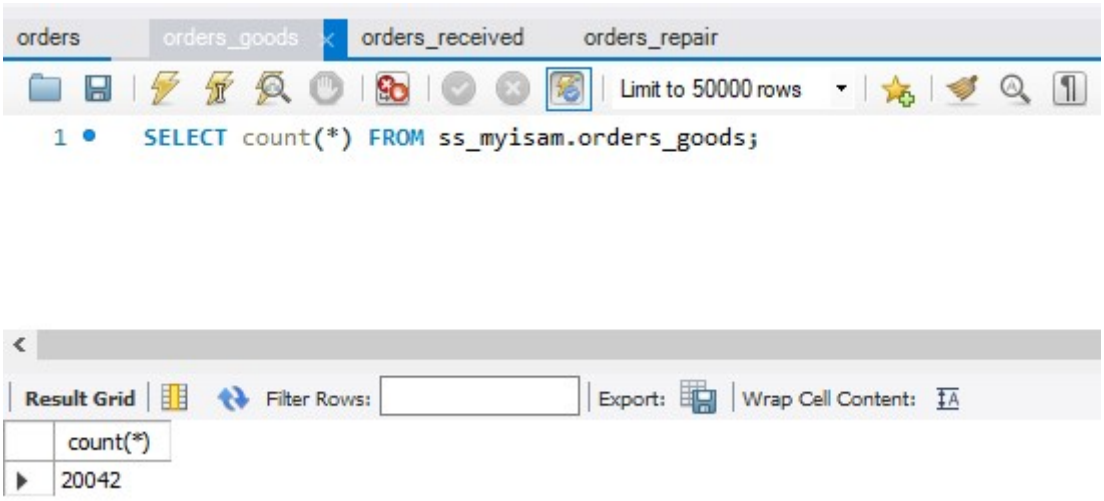

Рис. 1.6 Скріншот введення даних у таблицю orders_goods

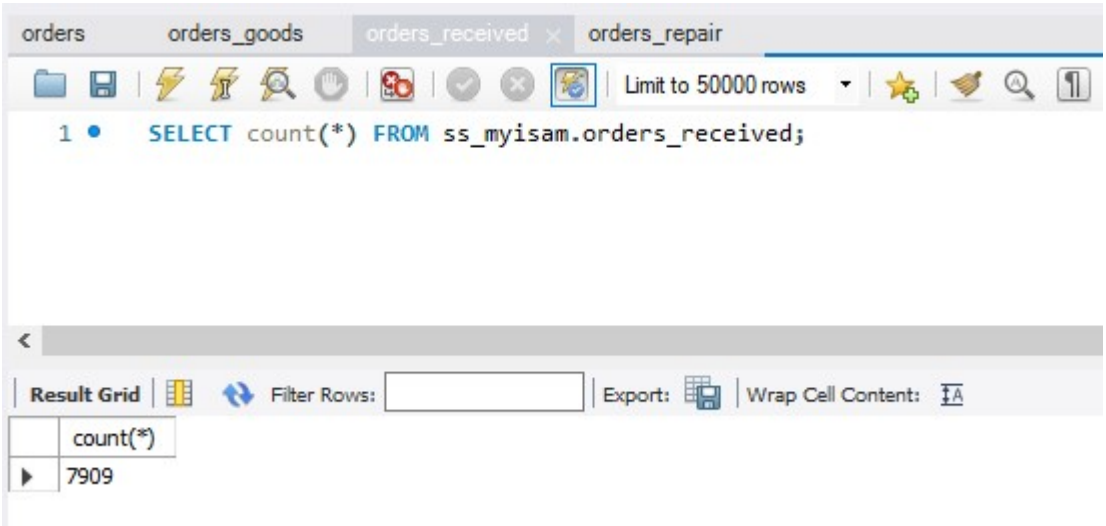

Рис. 1.7 Скріншот введення даних у таблицю orders_received

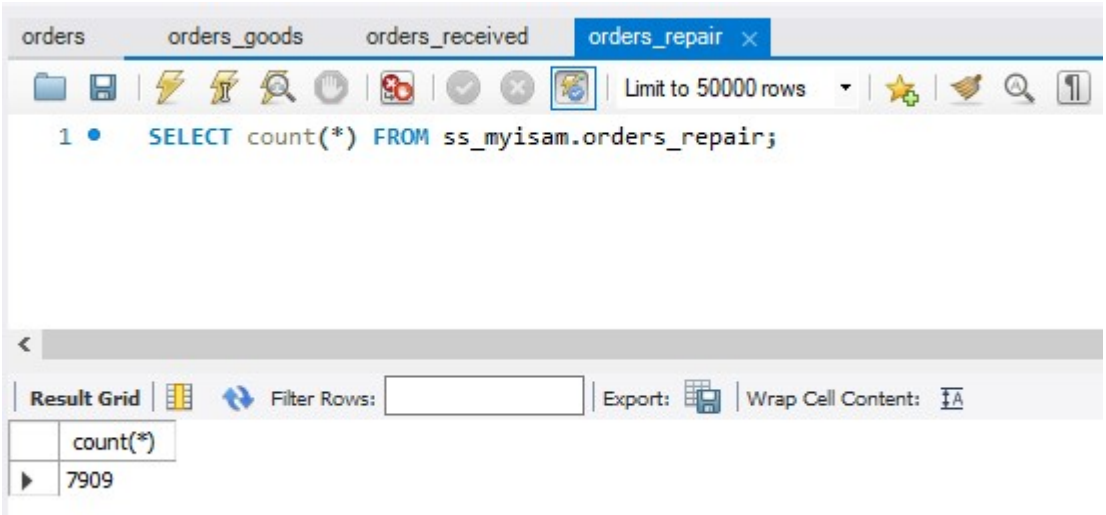

Рис. 1.8 Скріншот введення даних у таблицю orders_repair

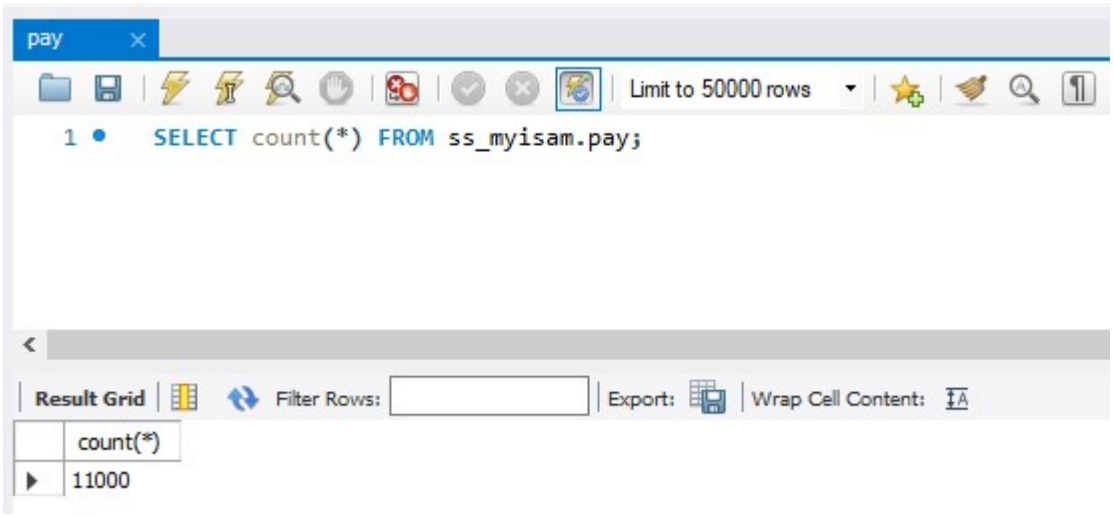

Рис. 1.9 Скріншот введення даних у таблицю pay

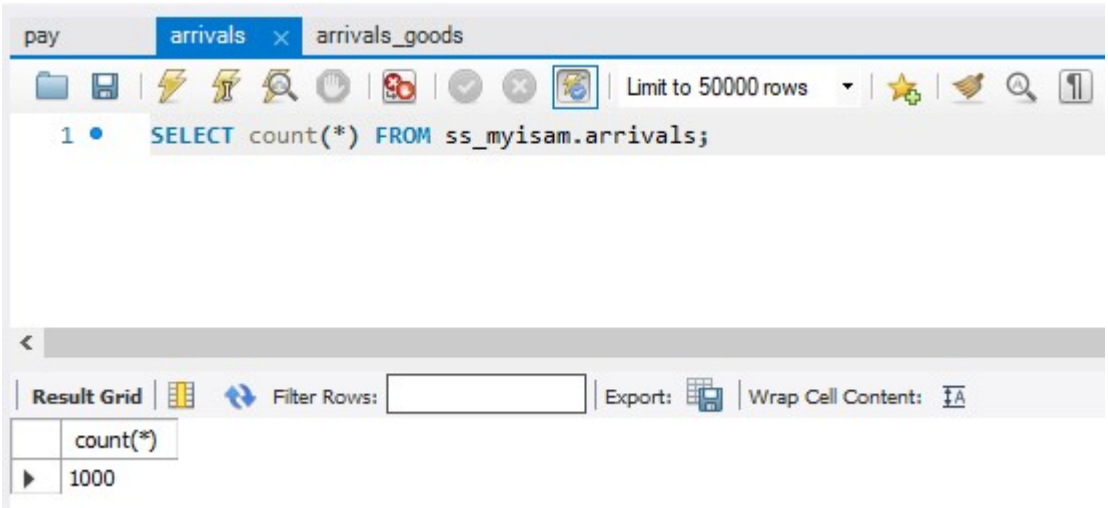

Рис. 1.10 Скріншот введення даних у таблицю arrivals

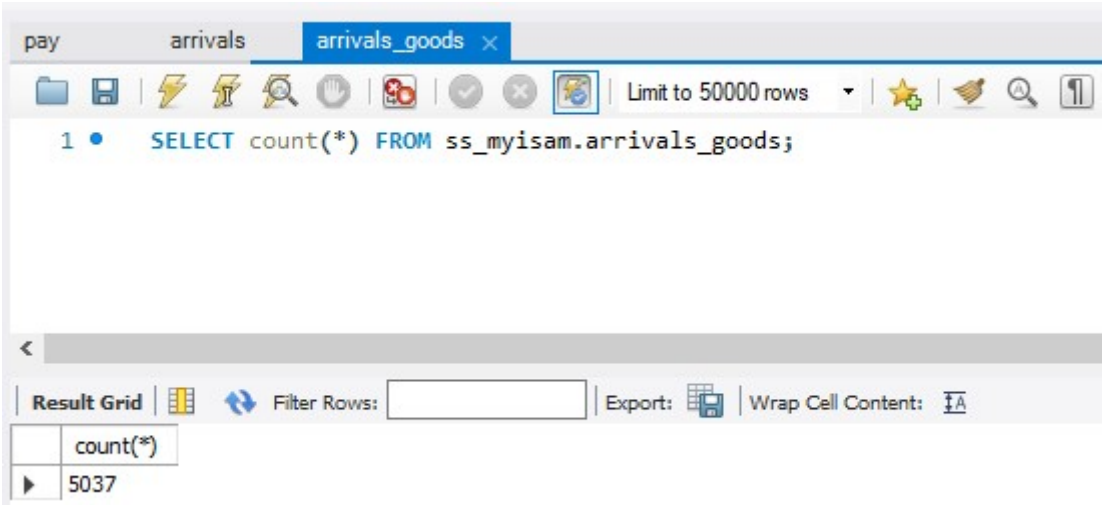

Рис. 1.11 Скріншот введення даних у таблицю arrivals_goods

Завдання 1.3. На основі логічної моделі бази даних завдання 1.2 розробити серверну частину високонавантаженої інформаційної системи з таблицями СУБД MySQL типу InnoDB.

Таблиця 1.4 – Перелік типів посилальної цілісності

№	Ім’я таблиці 1, зовнішній ключ	Ім’я таблиці 2, первинний ключ	SQL -інструкція для таблиці 1	Тип посилальної цілісності
1	customers, status_id	status, status_id	UPDATE	CASCADE
2	customers, status_id	status, status_id	DELETE	RESTRICT
3	customers, city_id	city, city_id	UPDATE	CASCADE
4	customers, city_id	city, city_id	DELETE	RESTRICT
5	goods_dscr, unit_id	unit, unit_id	UPDATE	CASCADE
6	goods_dscr, unit_id	unit, unit_id	DELETE	RESTRICT
7	goods_dscr, group_id	group, group_id	UPDATE	CASCADE
8	goods_dscr, group_id	group, group_id	DELETE	RESTRICT
9	goods, goods_dscr_id	goods_dscr, goods_dscr_id	UPDATE, DELETE	CASCADE
11	orders_goods, goods_id	goods, goods_id	UPDATE, DELETE	CASCADE
12	orders_goods, order_id	orders, order_id	UPDATE, DELETE	CASCADE

13	orders_repair, order_id	orders, order_id	UPDATE, DELETE	CASCADE
14	orders_repair, employee_id	employees, employee_id	UPDATE	CASCADE
15	orders_repair, employee_id	employees, employee_id	DELETE	RESTRICT
16	orders_received, order_id	orders, order_id	UPDATE, DELETE	CASCADE
17	orders_received, brand _id	brand, brand _id	UPDATE	CASCADE
18	orders_received, brand _id	brand, brand _id	DELETE	RESTRICT
19	orders_received, employee_id	employees, employee_id	UPDATE	CASCADE
20	orders_received, employee_id	employees, employee_id	DELETE	RESTRICT
21	orders, employee_id	employees, employee_id	UPDATE	CASCADE
22	orders, employee_id	employees, employee_id	DELETE	RESTRICT
23	orders, delivery_id	delivery, delivery_id	UPDATE	CASCADE
24	orders, delivery_id	delivery, delivery_id	DELETE	RESTRICT
25	orders, customer_id	customers, customer_id	UPDATE, DELETE	CASCADE
26	salary, employee_id	employees, employee_id	UPDATE	CASCADE
27	salary, employee_id	employees, employee_id	DELETE	RESTRICT
28	pay, employee_id	employees, employee_id	UPDATE	CASCADE

29	pay, employee_id	employees, employee_id	DELETE	RESTRICT
30	pay, customer_id	customers, customer_id	UPDATE, DELETE	CASCADE
31	pay, arrival_id	arrivals, arrival_id	UPDATE, DELETE	CASCADE
32	pay, order_id	orders, order_id	UPDATE, DELETE	CASCADE
33	arrivals, employee_id	employees, employee_id	UPDATE	CASCADE
34	arrivals, employee_id	employees, employee_id	DELETE	RESTRICT
35	arrivals, customer_id	customers, customer_id	UPDATE, DELETE	CASCADE
36	arrival_goods, arrival_id	arrivals, arrival_id	UPDATE, DELETE	CASCADE
37	arrival_goods, good_id	goods, good_id	UPDATE, DELETE	CASCADE
38	employees, status_id	status, status_id	UPDATE	CASCADE
39	employees, status_id	status, status_id	DELETE	RESTRICT
40	employees, position_id	position, position_id	UPDATE	CASCADE
41	employees, position_id	position, position_id	DELETE	RESTRICT

Посилальна цілісність прописана щоб виконати бізнес-функцію №26 «Видалення застарілих даних (більш як 3 роки, крім робітників)», для всіх зв’язків, крім робітників, CASCADE/CASCADE. При цьому треба прописати тригери, щоб позбутися випадкового видалення даних раніше, ніж треба.

Для робітників зв’язки встановлені як CASCADE/RESTRICT, тому що по законодавству України дані по робітникам видаляти не можна, треба зберігати 75 років.

– створення бази даних з таблицями типу InnoDB, лістинг коду.

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';

-- -----------------------------------------------------
-- Schema ss_inno
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `ss_inno` ;

-- -----------------------------------------------------
-- Schema ss_inno
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `ss_inno` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci ;
USE `ss_inno` ;

-- -----------------------------------------------------
-- Table `ss_inno`.`city`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`city` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`city` (
  `city_id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(30) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  PRIMARY KEY (`city_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`status`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`status` ;
```

```
CREATE TABLE IF NOT EXISTS `ss_inno`.`status` (
  `status_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'admin, buh, manager, client - права доступу',
  PRIMARY KEY (`status_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`customers`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`customers` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`customers` (
  `customer_id` INT NOT NULL AUTO_INCREMENT,
  `status_id` TINYINT UNSIGNED NOT NULL,
  `city_id` INT NOT NULL,
  `last_name` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `first_name` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `middle_name` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `code` CHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'Ід.Код - цифри',
  `email` VARCHAR(50) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `phone` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'номер уникален (по ньому вхід) +380 (067) 123 45 67',
  `address` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `password` CHAR(90) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'Хеш пароля (для входа online)',
  `is_active` TINYINT(1) UNSIGNED NOT NULL DEFAULT 1 COMMENT 'активен -1, ні - 0',
  `balance` INT NOT NULL DEFAULT 0 COMMENT 'тут зберігається поточний баланс контрагента',
  `create_time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `update_time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`customer_id`),
  UNIQUE INDEX `phone_UNIQUE` (`phone` ASC) VISIBLE,
  INDEX `fk_customer_city_idx` (`city_id` ASC) VISIBLE,
  INDEX `fk_customers_status1_idx` (`status_id` ASC) VISIBLE,
  CONSTRAINT `fk_customer_city`
    FOREIGN KEY (`city_id`)
```

```
    REFERENCES `ss_inno`.`city` (`city_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_customers_status1`
    FOREIGN KEY (`status_id`)
    REFERENCES `ss_inno`.`status` (`status_id`)
    ON DELETE RESTRICT
    ON UPDATE RESTRICT)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`position`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`position` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`position` (
  `position_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'посада',
  PRIMARY KEY (`position_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`employees`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`employees` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`employees` (
  `employee_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `status_id` TINYINT UNSIGNED NOT NULL,
  `position_id` TINYINT UNSIGNED NOT NULL,
  `first_name` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `middle_name` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `last_name` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
```

```
  `code` CHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'налог. номер',
  `birthday` DATE NOT NULL,
  `email` VARCHAR(50) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `phone` CHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'формат +380 (067) 123 45 67',
  `address` VARCHAR(100) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `description` VARCHAR(512) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'якийсь опис співробітника',
  `password` CHAR(90) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'Хеш пароля',
  `is_active` TINYINT(1) UNSIGNED NOT NULL DEFAULT 1 COMMENT 'активен - 1, ні - 0',
  PRIMARY KEY (`employee_id`),
  UNIQUE INDEX `phone_UNIQUE` (`phone` ASC) VISIBLE,
  INDEX `fk_employees_position1_idx` (`position_id` ASC) VISIBLE,
  INDEX `fk_employees_status1_idx` (`status_id` ASC) VISIBLE,
  CONSTRAINT `fk_employees_position`
    FOREIGN KEY (`position_id`)
    REFERENCES `ss_inno`.`position` (`position_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_employees_status`
    FOREIGN KEY (`status_id`)
    REFERENCES `ss_inno`.`status` (`status_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci
COMMENT = '98BFDA';


-- -----------------------------------------------------
-- Table `ss_inno`.`delivery`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`delivery` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`delivery` (
  `delivery_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT COMMENT 'Назва кур\'єра доставки',
  `name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  PRIMARY KEY (`delivery_id`),
```

```
  UNIQUE INDEX `name_UNIQUE` (`name` ASC) VISIBLE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`orders`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`orders` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`orders` (
  `order_id` INT NOT NULL AUTO_INCREMENT,
  `customer_id` INT NOT NULL,
  `employee_id` TINYINT UNSIGNED NULL COMMENT 'Поле заповнюється при видачіі. При створенні замовлення поле is null',
  `delivery_id` TINYINT UNSIGNED NULL,
  `date_invoice` DATE NOT NULL COMMENT 'дата створення рахунка по итогу прийняття техніки в ремонт',
  `date_delivery` DATE NULL COMMENT 'відвантажено, отримано кур\'єром',
  `delivery_number` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `date_returned` DATE NULL COMMENT 'повернуто клієнту',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `read_only` TINYINT(1) NOT NULL DEFAULT 0 COMMENT '0-доступен, 1-только чтение',
  PRIMARY KEY (`order_id`),
  INDEX `fk_orders_customers_idx` (`customer_id` ASC) VISIBLE,
  INDEX `fk_orders_delivery1_idx` (`delivery_id` ASC) VISIBLE,
  INDEX `fk_orders_employees1_idx` (`employee_id` ASC) VISIBLE,
  CONSTRAINT `fk_orders_customers`
    FOREIGN KEY (`customer_id`)
    REFERENCES `ss_inno`.`customers` (`customer_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_orders_delivery`
    FOREIGN KEY (`delivery_id`)
    REFERENCES `ss_inno`.`delivery` (`delivery_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_orders_employees`
    FOREIGN KEY (`employee_id`)
    REFERENCES `ss_inno`.`employees` (`employee_id`)
```

```
    ON DELETE RESTRICT
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`arrivals`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`arrivals` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`arrivals` (
  `arrival_id` INT NOT NULL AUTO_INCREMENT,
  `customer_id` INT NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `date_arrival` DATE NOT NULL,
  `total` INT UNSIGNED NULL DEFAULT 0 COMMENT 'Сумма приходной накладной У КОПІЙКАХ(!)',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `read_only` TINYINT(1) NOT NULL DEFAULT 0 COMMENT '0-доступен, 1-только чтение',
  PRIMARY KEY (`arrival_id`),
  INDEX `fk_arrivals_customers1_idx` (`customer_id` ASC) VISIBLE,
  INDEX `fk_arrivals_employees1_idx` (`employee_id` ASC) VISIBLE,
  CONSTRAINT `fk_arrivals_customers`
    FOREIGN KEY (`customer_id`)
    REFERENCES `ss_inno`.`customers` (`customer_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_arrivals_employees`
    FOREIGN KEY (`employee_id`)
    REFERENCES `ss_inno`.`employees` (`employee_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`pay`
-- -----------------------------------------------------
```

```
DROP TABLE IF EXISTS `ss_inno`.`pay` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`pay` (
  `pay_id` INT NOT NULL AUTO_INCREMENT,
  `customer_id` INT NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `arrival_id` INT NULL,
  `order_id` INT NULL,
  `date_pay` DATE NOT NULL COMMENT 'дата оплати',
  `pay_sum` INT NOT NULL DEFAULT 0 COMMENT 'сума оплати У КОПІЙКАХ(!) (для прихода < 0)',
  `pay_form` ENUM('готівка', 'оплата на р.р.') CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci'
NOT NULL DEFAULT 'оплата на р.р.' COMMENT '\'готівка\', \'оплата на р.р.\'',
  `description` VARCHAR(512) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `read_only` TINYINT(1) NOT NULL DEFAULT 0 COMMENT '0-доступно, 1-только читання',
  PRIMARY KEY (`pay_id`),
  INDEX `fk_pay_customers1_idx` (`customer_id` ASC) VISIBLE,
  INDEX `fk_pay_employees1_idx` (`employee_id` ASC) VISIBLE,
  INDEX `fk_pay_arrivals1_idx` (`arrival_id` ASC) VISIBLE,
  INDEX `fk_pay_orders1_idx` (`order_id` ASC) VISIBLE,
  CONSTRAINT `fk_pay_customers`
    FOREIGN KEY (`customer_id`)
    REFERENCES `ss_inno`.`customers` (`customer_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_pay_employees`
    FOREIGN KEY (`employee_id`)
    REFERENCES `ss_inno`.`employees` (`employee_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_pay_arrivals`
    FOREIGN KEY (`arrival_id`)
    REFERENCES `ss_inno`.`arrivals` (`arrival_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_pay_orders`
    FOREIGN KEY (`order_id`)
    REFERENCES `ss_inno`.`orders` (`order_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
```

```
COLLATE = utf8mb4_unicode_ci
COMMENT = ' ';


-- -----------------------------------------------------
-- Table `ss_inno`.`brand`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`brand` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`brand` (
  `brand_id` SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'опис бренду',
  PRIMARY KEY (`brand_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`orders_received`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`orders_received` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`orders_received` (
  `order_id` INT NOT NULL,
  `brand_id` SMALLINT UNSIGNED NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `model_name` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `sn` VARCHAR(30) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `barcode` CHAR(13) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'EAN-13',
  `equipment` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'Який комплектації прибув прибор, зовнішні пошкодження',
  `fault_description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT 'опис несправності',
  PRIMARY KEY (`order_id`),
  INDEX `fk_orders_received_brand1_idx` (`brand_id` ASC) VISIBLE,
  INDEX `fk_orders_received_employees1_idx` (`employee_id` ASC) VISIBLE,
  CONSTRAINT `fk_orders_description_orders`
```

```
    FOREIGN KEY (`order_id`)
    REFERENCES `ss_inno`.`orders` (`order_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_orders_received_brand`
    FOREIGN KEY (`brand_id`)
    REFERENCES `ss_inno`.`brand` (`brand_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_orders_received_employees`
    FOREIGN KEY (`employee_id`)
    REFERENCES `ss_inno`.`employees` (`employee_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`orders_repair`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`orders_repair` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`orders_repair` (
  `order_id` INT NOT NULL,
  `employee_id` TINYINT UNSIGNED NOT NULL,
  `date_ready` DATE NULL COMMENT 'Дата готовності після ремонту',
  `price_repair` INT NULL DEFAULT 0 COMMENT 'вартість викононой роботи У КОПІЙКАХ(!)',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL COMMENT 'що зроблено',
  PRIMARY KEY (`order_id`, `employee_id`),
  INDEX `fk_orders_repair_orders1_idx` (`order_id` ASC) VISIBLE,
  INDEX `fk_orders_repair_employees1_idx` (`employee_id` ASC) VISIBLE,
  CONSTRAINT `fk_orders_repair_orders`
    FOREIGN KEY (`order_id`)
    REFERENCES `ss_inno`.`orders` (`order_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_orders_repair_employees`
    FOREIGN KEY (`employee_id`)
```

```
   REFERENCES `ss_inno`.`employees` (`employee_id`)
   ON DELETE CASCADE
   ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`unit`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`unit` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`unit` (
  `unit_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(10) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL COMMENT
'одиниця виміру',
  PRIMARY KEY (`unit_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`group`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`group` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`group` (
  `group_id` SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(30) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  PRIMARY KEY (`group_id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`goods_dscr`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`goods_dscr` ;
```

```
CREATE TABLE IF NOT EXISTS `ss_inno`.`goods_dscr` (
  `goods_dscr_id` INT NOT NULL AUTO_INCREMENT,
  `unit_id` TINYINT UNSIGNED NOT NULL,
  `group_id` SMALLINT UNSIGNED NOT NULL,
  `part_number` VARCHAR(20) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `name` VARCHAR(50) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NOT NULL,
  `margin` TINYINT UNSIGNED NOT NULL DEFAULT 30 COMMENT 'наценка, у процентах',
  `price` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'ціна продажу У КОПІЙКАХ(!)',
  `description` TEXT CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  `is_active` TINYINT(1) NOT NULL DEFAULT 1 COMMENT '1-активен, 0-блок',
  UNIQUE INDEX `part_number_UNIQUE` (`part_number` ASC) VISIBLE,
  INDEX `fk_goods_unit1_idx` (`unit_id` ASC) VISIBLE,
  PRIMARY KEY (`goods_dscr_id`),
  INDEX `fk_goods_dscr_group1_idx` (`group_id` ASC) VISIBLE,
  CONSTRAINT `fk_goods_unit`
    FOREIGN KEY (`unit_id`)
    REFERENCES `ss_inno`.`unit` (`unit_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_goods_dscr_group`
    FOREIGN KEY (`group_id`)
    REFERENCES `ss_inno`.`group` (`group_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`goods`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`goods` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`goods` (
  `goods_id` INT NOT NULL AUTO_INCREMENT,
  `goods_dscr_id` INT NOT NULL,
  `price_in` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'вхідна ціна У КОПІЙКАХ(!)',
  `stock` DECIMAL(7,3) UNSIGNED NOT NULL DEFAULT 0 COMMENT 'залишок запчастин',
  PRIMARY KEY (`goods_id`),
```

```
  INDEX `fk_goods_goods_dscr1_idx` (`goods_dscr_id` ASC) VISIBLE,
  CONSTRAINT `fk_goods_goods_dscr`
    FOREIGN KEY (`goods_dscr_id`)
    REFERENCES `ss_inno`.`goods_dscr` (`goods_dscr_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`orders_goods`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`orders_goods` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`orders_goods` (
  `order_id` INT NOT NULL,
  `goods_id` INT NOT NULL,
  `quantity` DECIMAL(7,3) UNSIGNED NOT NULL DEFAULT 1,
  `price_out` INT UNSIGNED NOT NULL DEFAULT 0,
  `description` VARCHAR(512) CHARACTER SET 'utf8mb4' COLLATE 'utf8mb4_unicode_ci' NULL,
  PRIMARY KEY (`order_id`, `goods_id`),
  INDEX `fk_goods_repair_goods1_idx` (`goods_id` ASC) VISIBLE,
  CONSTRAINT `fk_goods_orders`
    FOREIGN KEY (`order_id`)
    REFERENCES `ss_inno`.`orders` (`order_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_goods_goods`
    FOREIGN KEY (`goods_id`)
    REFERENCES `ss_inno`.`goods` (`goods_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`arrivals_goods`
```

```
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`arrivals_goods` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`arrivals_goods` (
  `arrival_id` INT NOT NULL,
  `goods_id` INT NOT NULL,
  `quantity` DECIMAL(7,3) UNSIGNED NOT NULL DEFAULT 1,
  PRIMARY KEY (`arrival_id`, `goods_id`),
  INDEX `fk_arrival_goods_arrivals1_idx` (`arrival_id` ASC) VISIBLE,
  CONSTRAINT `fk_arrival_goods_goods`
    FOREIGN KEY (`goods_id`)
    REFERENCES `ss_inno`.`goods` (`goods_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_arrival_goods_arrivals`
    FOREIGN KEY (`arrival_id`)
    REFERENCES `ss_inno`.`arrivals` (`arrival_id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


-- -----------------------------------------------------
-- Table `ss_inno`.`salary`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `ss_inno`.`salary` ;

CREATE TABLE IF NOT EXISTS `ss_inno`.`salary` (
  `employee_id` TINYINT UNSIGNED NOT NULL COMMENT 'рік, за який нарахована ЗП',
  `accrued_year` SMALLINT UNSIGNED NOT NULL DEFAULT 2024 COMMENT 'рік нарахування ЗП > 2023',
  `accrued_month` TINYINT UNSIGNED NOT NULL COMMENT 'місяц, за який нарахована ЗП, between 1 and 12',
  `accrued_sum` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'нараховано сума ЗП в копійках',
  `ESV` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'єдиний соціальний внесок',
  `PDFO` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'податок на доход, ПДФО',
  `VS` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'військовий збір',
  `payable_in_cash` INT UNSIGNED NOT NULL DEFAULT 0 COMMENT 'до виплати на руки',
  `description` VARCHAR(512) NULL,
```

```
  INDEX `fk_salary_employees1_idx` (`employee_id` ASC) VISIBLE,
  PRIMARY KEY (`employee_id`, `accrued_year`, `accrued_month`),
  CONSTRAINT `fk_salary_employees`
    FOREIGN KEY (`employee_id`)
    REFERENCES `ss_inno`.`employees` (`employee_id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_unicode_ci;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

– створення фізичної моделі бази даних з таблицями типу InnoDB у вигляді ER-діаграм у нотації IDEF1X

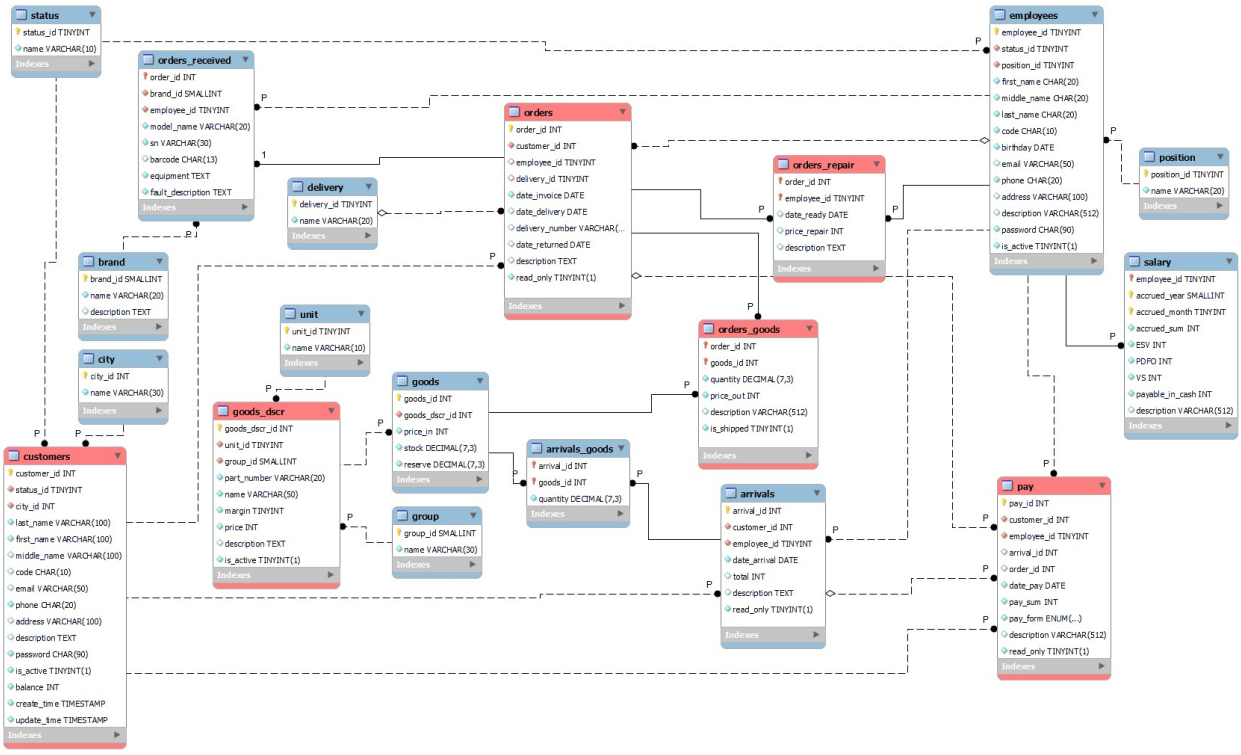

Рис. 1.12 Схема БД з таблицями типу InnoDB

– заповнення основних таблиць даними.

Для заповнення таблиц фейковими даними я використал програму на мові Python (задіяв бібліотеки pymysql, faker, random).

Приклад програми, що генерує 10 тисяч записів у таблиці customers:

```
from faker import Faker
import pymysql.cursors
from config24 import host, user, password, db_name
import random
from passlib.hash import pbkdf2_sha256

try:
    connection = pymysql.connect(
        host=host,
        port=3306,
        user=user,
        password=password,
        database=db_name,
        charset='utf8mb4',
        cursorclass=pymysql.cursors.DictCursor
    )
    print(f"{db_name} - successfully connected...")
    print("=" * 30)
except Exception as ex:
    print(f"{db_name} - connection refused...")
    print(ex)

if __name__ == '__main__':
    fake = Faker("uk_UA")
    num_rows = 10000

    with connection:
        with connection.cursor() as cursor:
            for _ in range(num_rows):
                status_id = 6  # client
                city_id = random.randint(1, 10)
                full_name = fake.full_name()  # 'Бабенко Оксенія Геннадіївна'
                last_name = full_name.split()[0]
                first_name = full_name.split()[1]
                middle_name = full_name.split()[2]
                code = fake.ssn()
                email = fake.free_email()
                phone = fake.phone_number()
                address = fake.street_address()
                dscr = fake.sentence(nb_words=45, variable_nb_words=False)
                if len(dscr) > 299:
                    dscr = dscr[:299]

                password = pbkdf2_sha256.hash('password')

                sql = "INSERT INTO `customers` (`city_id`, `status_id`,
`last_name`, `first_name`, `middle_name`, `code`, `email`, `phone`,
`address`, `description`,`password`) VALUES (%s, %s, %s, %s, %s, %s, %s, %s,
%s, %s, %s)"
                cursor.execute(sql, (city_id, status_id, last_name,
first_name, middle_name, code, email, phone, address, dscr, password))

            connection.commit()

    print(f"Вставлено {num_rows} рядків у таблицю customers")
```

Скриншоти вставки даних у таблиці БД ss_innodb показани на рис.1.13-1.22

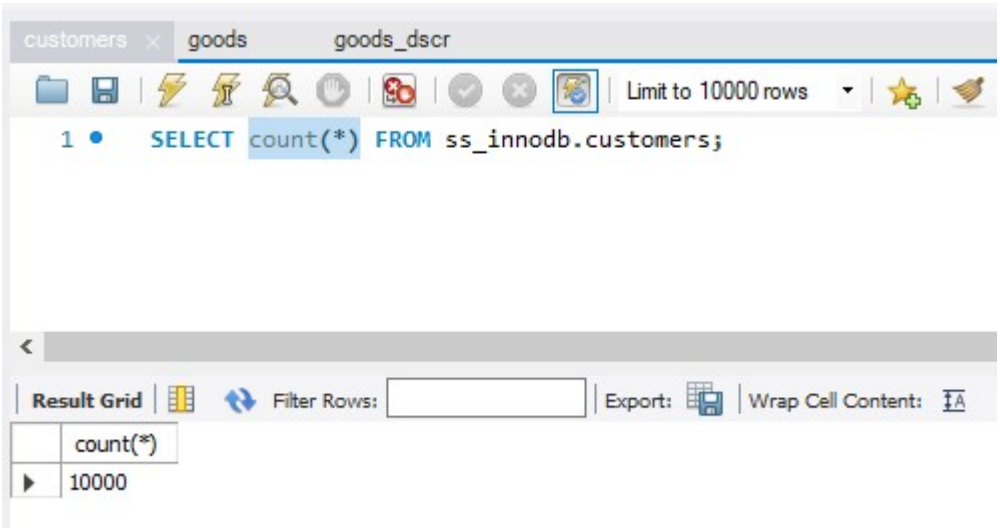

Рис. 1.13 Скріншот введення даних у таблицю customers

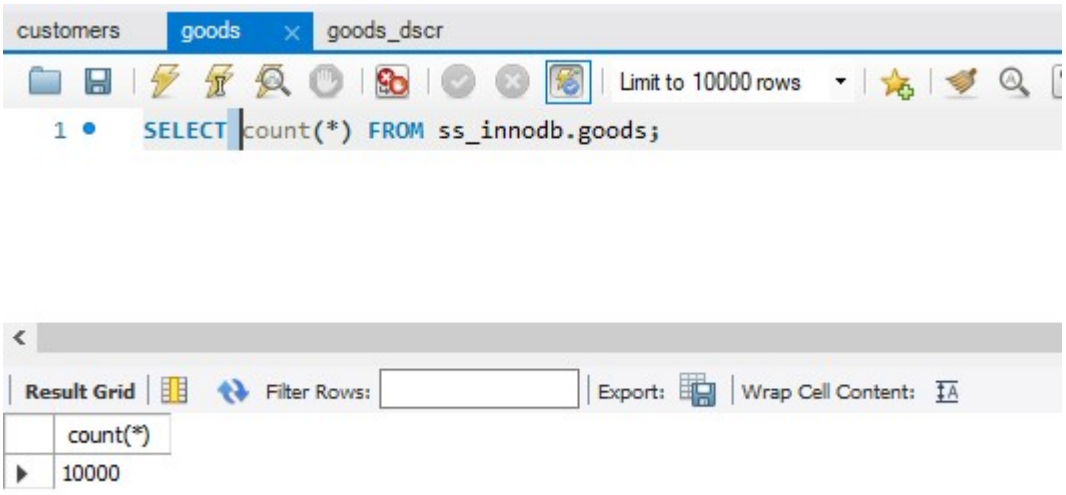

Рис. 1.14 Скріншот введення даних у таблицю goods

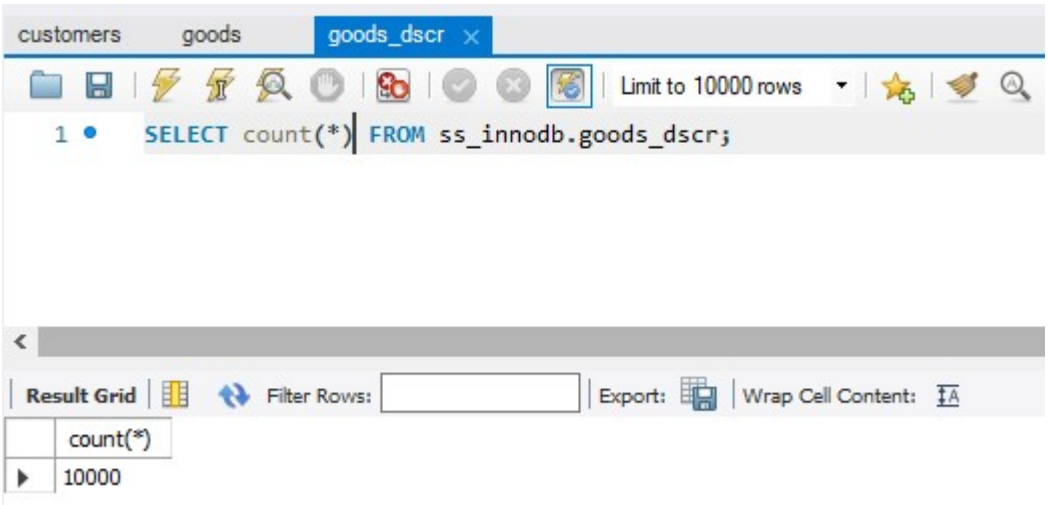

Рис. 1.15 Скріншот введення даних у таблицю goods_dscr

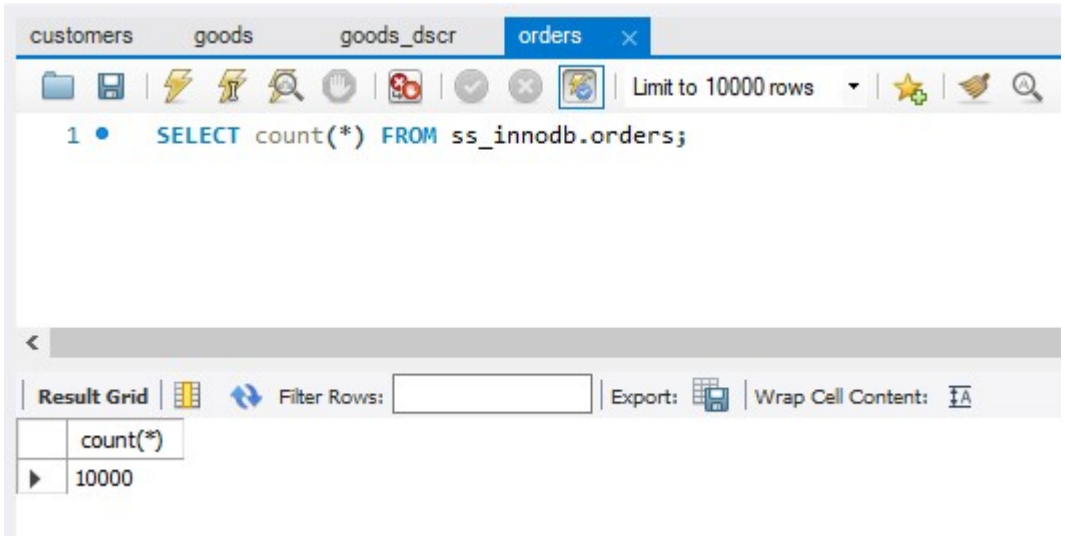

Рис. 1.16 Скріншот введення даних у таблицю orders

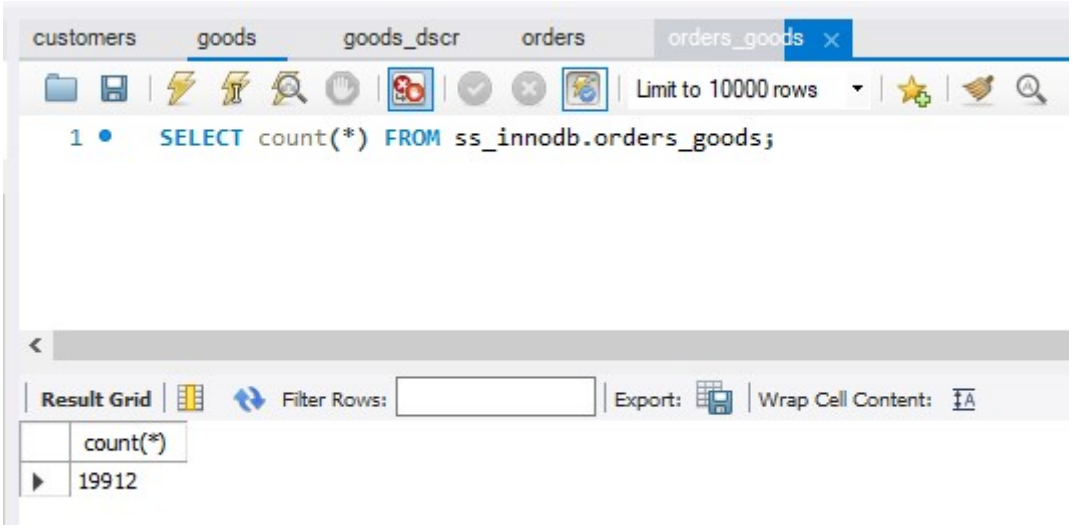

Рис. 1.17 Скріншот введення даних у таблицю orders_goods

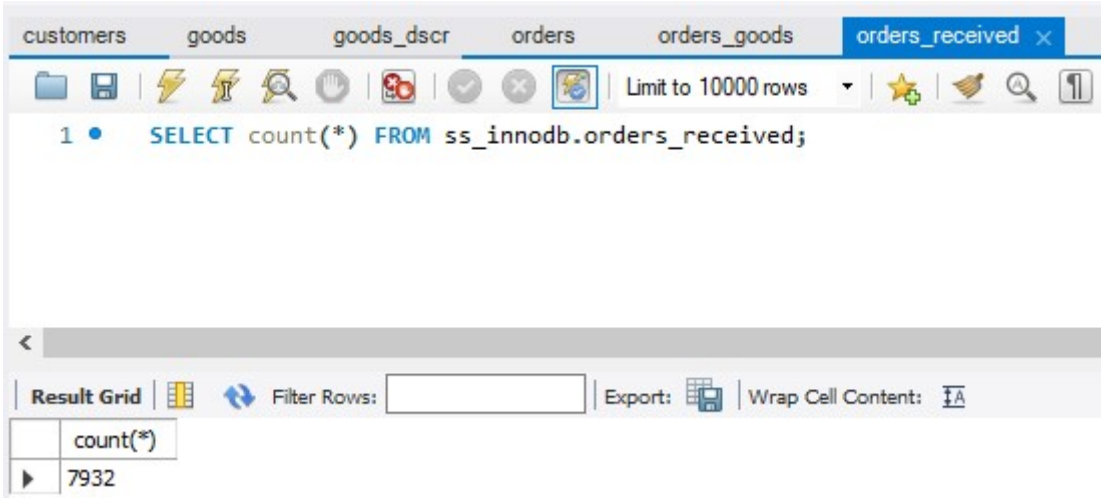

Рис. 1.18 Скріншот введення даних у таблицю orders_received

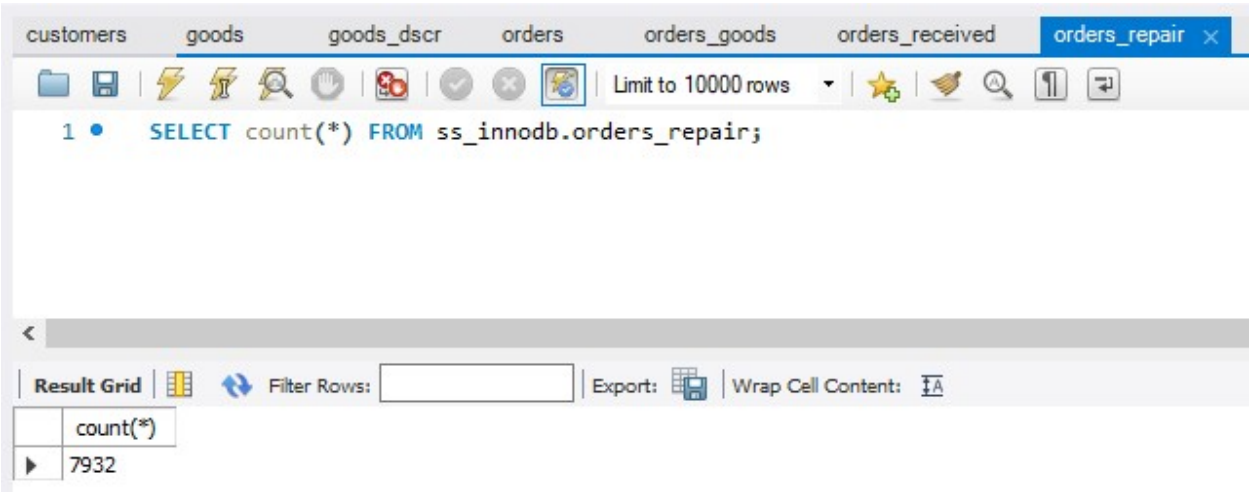

Рис. 1.19 Скріншот введення даних у таблицю orders_repair

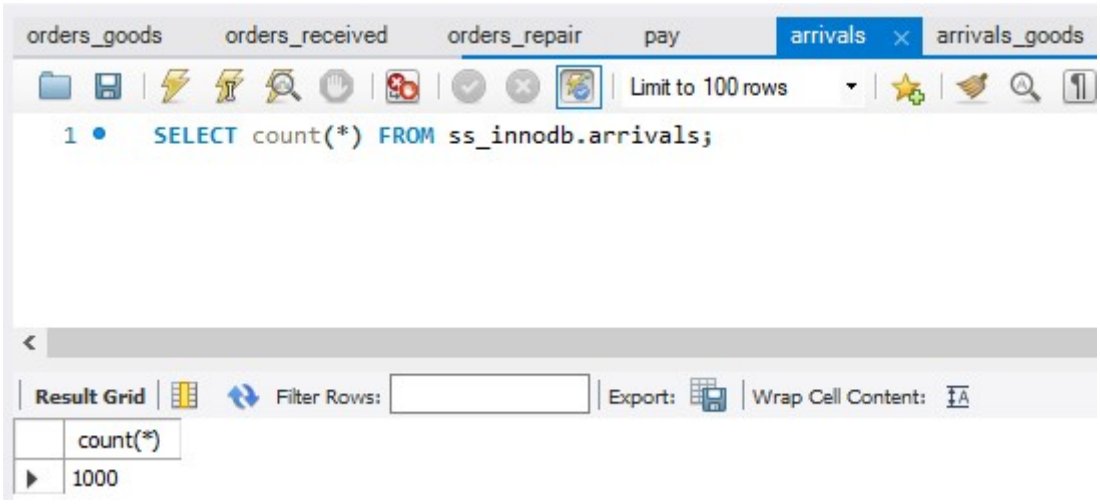

Рис. 1.20 Скріншот введення даних у таблицю arrivals

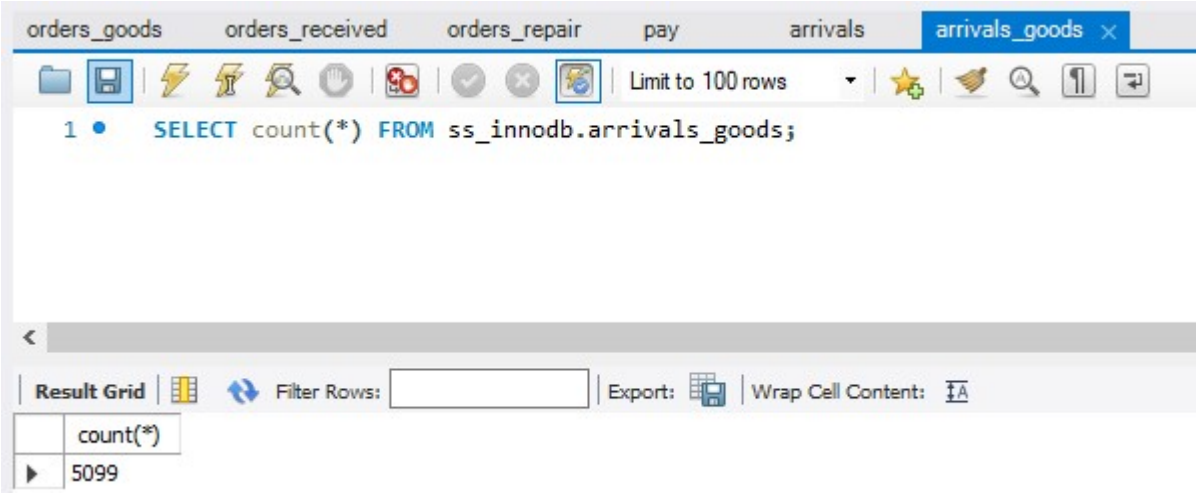

Рис. 1.21 Скріншот введення даних у таблицю arrivals_goods

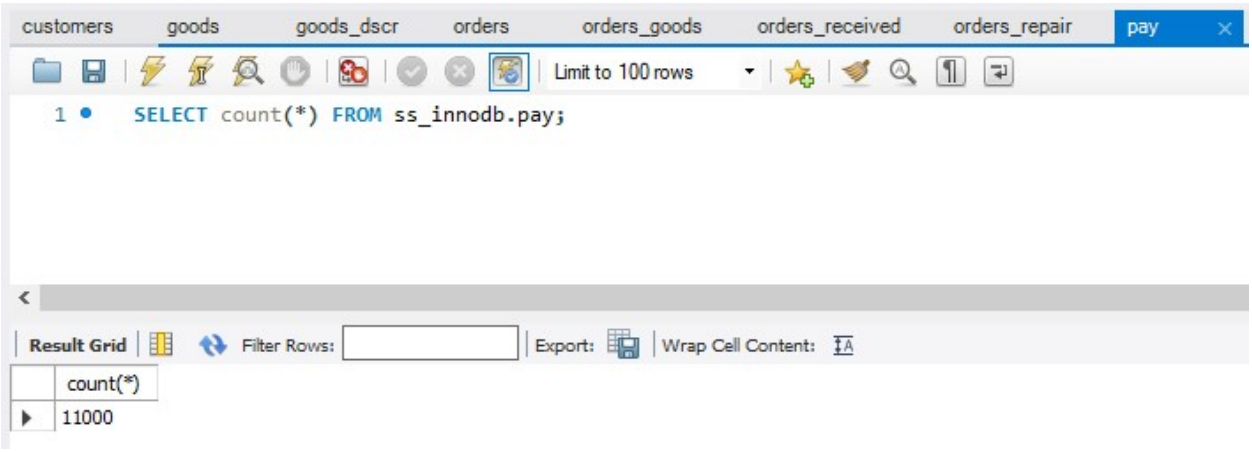

Рис. 1.22 Скріншот введення даних у таблицю pay

Як і очікувалося,БД, створена на таблицях MyISAM займає менше міста (майже в 3 рази) на диску, ніж БД на таблицях InnoDB, рис. 1.23.

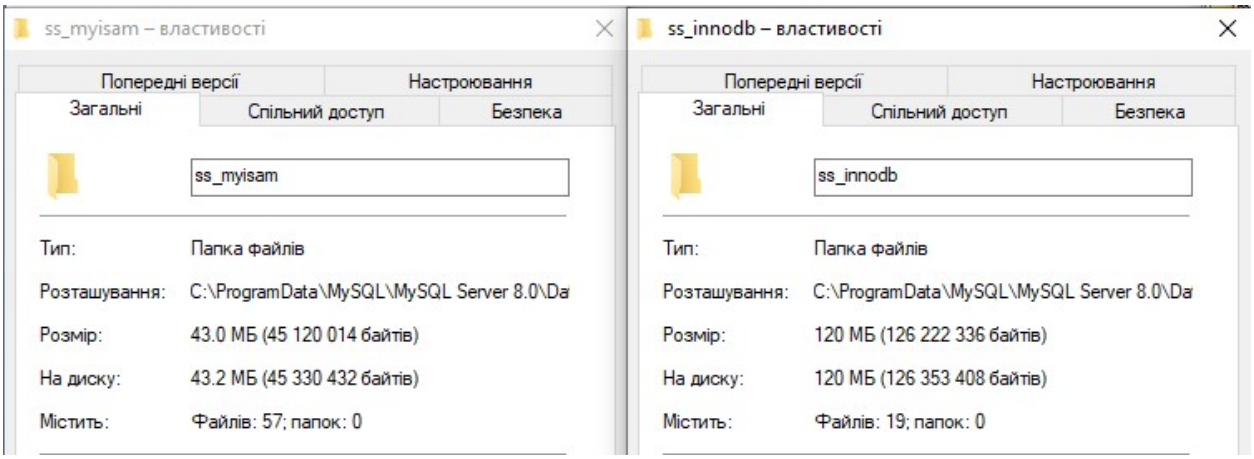

Рис. 1.23 Скріншот властивостей каталогів с БД

– висновки з роботи з обов'язковим зазначенням переваг і недоліків розроблених варіантів реалізації баз даних на платформі СУБД MySQL.

Вибір між движками зберігання даних MyISAM та InnoDB в MySQL є важливим рішенням при проектуванні бази даних. Кожен з них має свої особливості, переваги та недоліки.

Переваги MyISAM:

Швидкість читання: MyISAM, як правило, забезпечує швидший доступ до даних при виконанні операцій читання (SELECT).

Менше дискового простору: Таблиці MyISAM займають менше місця на диску, ніж InnoDB.

Недоліки:

Відсутність транзакцій: Не підтримує транзакцій, що може призвести до втрати даних у разі збоїв.

Відсутність зовнішніх ключів: Не підтримує зовнішні ключі, що ускладнює забезпечення цілісності даних.

Блокування таблиць: При модифікації даних блокується вся таблиця, що може вплинути на продуктивність при високому навантаженні.

Коли використовувати MyISAM:

Системи з високим навантаженням на читання: Якщо система в основному виконує операції читання, MyISAM може бути хорошим вибором завдяки своїй швидкості.

Системи, де не потрібні транзакції: Якщо цілісність даних може бути забезпечена іншими засобами, MyISAM також може бути варіантом.

Системи, де важлива економія дискового простору: Якщо дисковий простір обмежений, MyISAM може бути кращим вибором.

Переваги InnoDB:

Транзакції: Підтримує транзакції ACID, що забезпечує цілісність даних.

Зовнішні ключі: Підтримує зовнішні ключі, що дозволяє встановлювати зв'язки між таблицями.

Блокування на рівні рядків: Блокує лише необхідні рядки, що підвищує продуктивність при одночасних модифікаціях.

Підтримка багатоверсійного конкуретного контролю (MVCC): Дозволяє одночасне читання та запис даних без блокувань.

Недоліки:

Більше дискового простору: Займає більше місця на диску, ніж MyISAM.

Менша швидкість читання: Як правило, трохи повільніший при виконанні операцій читання, ніж MyISAM.

Коли використовувати InnoDB:

Системи, де потрібна висока цілісність даних: Якщо важлива цілісність даних і підтримка транзакцій, InnoDB є кращим вибором.

Системи з високим навантаженням на записування: Якщо система виконує багато операцій запису, InnoDB забезпечить більш високу продуктивність.

Системи з складними запитами: InnoDB краще справляється зі складними запитами, що включають з'єднання, агрегаційні функції та підзапити.

Коли вибрати MyISAM, а коли InnoDB?

- Перед вибором движка треба ретельно проаналізувати, які операції будуть виконуватися з даними.

- Перед запуском у виробництво треба обов'язково протестувати продуктивність системи з різними движками.

Внесення даних у таблиці InnoDB і MyISAM показали несподівані для мене результати по часу виконнання запису даних у таблиці.

Дані в таблиці InnoDB записувались однією транзакцією і в кожному разі час запису не перевищував десятка секунд. Тому я навіть не перейнявся заміром часу запису.

MyISAM не підтримує транзакції, тому дані записувались одразу після запиту. Загальний час виконання запису виявився значно вище, ніж у таблицях InnoDB.

Заміри часу виконання (скриншоти) внесення даних у таблиці MyISAM наводжу на рис. 1.24

```
ss_myisam - successfully connected...
==============================
Вставка 10000 рядків у таблиці goods_dscr і goods зайняла 22.280686140060425 секунд

Process finished with exit code 0
```

```
ss_myisam - successfully connected...
==============================
Вставка 10000 рядків у таблицю orders, а також в таблиці orders_goods, orders_received, orders_repair зайняла 57.353368043899536 секунд

Process finished with exit code 0
```

```
ss_myisam - successfully connected...
==============================
Вставка 10000 рядків у таблицю pay зайняла 13.821887254714966 секунд

Process finished with exit code 0
```

```
ss_myisam - successfully connected...
==============================
Вставка 1000 рядків у таблици arrivals, pay, а також arrivals_goods зайняла 8.224932670593262 секунд

Process finished with exit code 0
```

Рис. 1.24 Скріншот введення даних у таблиці MyISAM

Ця ситуація мене дуже зацікавила і я вирішив створити ще одну, точно таку саму БД InnoDB і перевірити результати внесення даних на ній.

1. БД InnoDB, при включеному `autocommit`, імітування запису даних як у таблиці MyISAM:

Заміри часу виконання (скриншоти) внесення даних у таблиці **InnoDB** з використанням **`autocommit`**, рис. 1.25

```
ss_inno - successfully connected...
==============================
Вставка 10000 рядків у таблиці goods_dscr і goods зайняла 24.902625560760498 секунд

Process finished with exit code 0
```

```
ss_inno - successfully connected...
==============================
Вставка 10000 рядків у таблицю orders, а також в таблиці orders_goods, orders_received, orders_repair зайняла 67.71184587478638 секунд

Process finished with exit code 0
```

```
ss_inno - successfully connected...
==============================
Вставка 10000 рядків у таблицю pay зайняла 14.78528642654419 секунд

Process finished with exit code 0
```

Рис. 1.25 Скріншот введення даних у таблиці **InnoDB(autocommit)**

2. Запис даних у таблиці InnoDB однією транзакцією:

Заміри часу виконання (скриншоти) внесення даних у таблиці InnoDB з використанням однієї транзакції, рис. 1.26

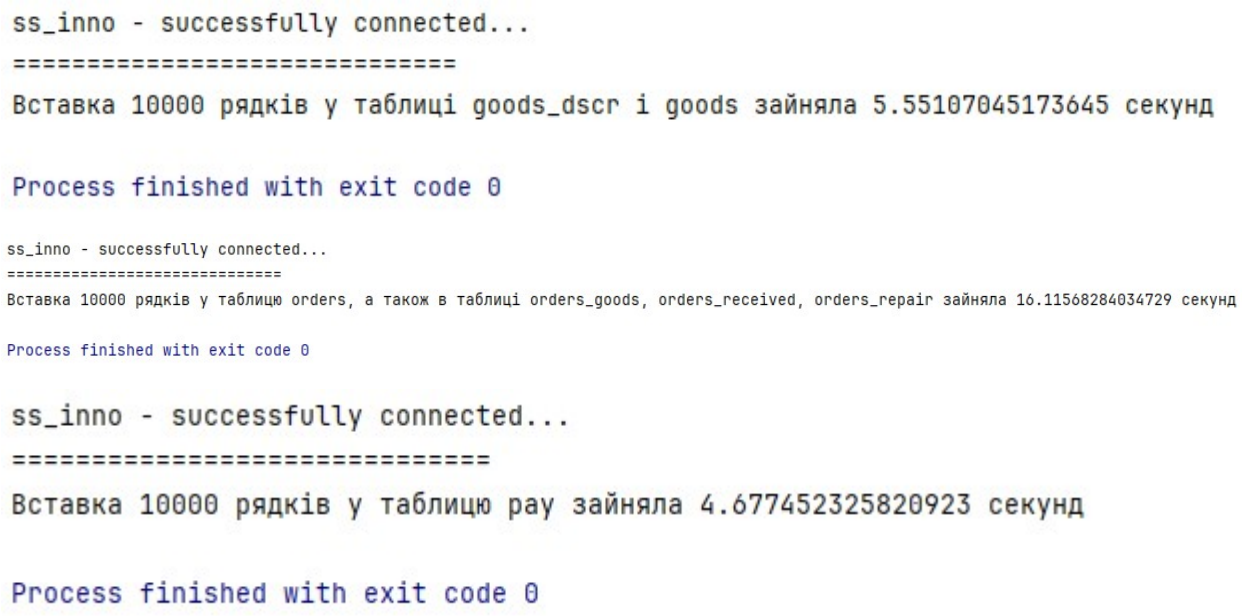

Рис. 1.26 Скріншот введення даних у таблиці InnoDB(однією транзакцією)

Результати експерименту звів у таблицю 1.5

Engine	**Час виконання вставки даних у таблиці:**		
	Goods(2таблиці),с	orders (4таблиці),с	Pay,с
MyISAM	**22.3**	**57.4**	**13.8**
InnoDB(auto)	**24.9**	**67.7**	**14.8**
InnoDB(1 транзакція)	**5.6**	**16.1**	**4.7**

Висновок.

В равних умовах engine InnoDB записує дані трошки повільніше, ніж engine MyISAM (від 6,7% до 15,2% в залежності від типу даних.

Однак використання механізму транзакцій разом із кешуванням даних змінюю картину радикально, тепер InnoDB випереджає MyISAM в 3-4 рази!

Перші результати отримали, далі буде…