НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС
«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»
НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

КУРСОВА РОБОТА

з дисципліни

Організація баз даних
на тему *«БД автопідприємства міста»*

Студента *III* курсу групи *КА-85*
Спеціальність *124 Системний аналіз*
Діхтяра Артема Вікторовича

Керівник: Афанасьєва І.В.

Національна оцінка: _____
Кількість балів: _____ Оцінка ECTS: _____

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

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

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

Київ – 2021 рік

АНОТАЦІЯ

Даний документ є пояснювальною запискою до курсової роботи з дисципліни «Організація баз даних» на тему «БД Автопідприємства міста»

Курсова робота присвячена створенню та демонстрації БД автопідприємства міста. Робота була створена у середовищі SQL Workbench.

ABSTRACT

This document is an explanatory note for the final project assignment of the “Organization of Databases” discipline on topic “ Auto-enterprises of the city Database”.

Project is devoted to create and demonstrate work of a copy of a database of Auto-enterprises of the city Database. The project was created in SQL Workbench.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – База Даних

ПЗ – Програмне забезпечення

РСУБД - Система Управління Реляційними Базами Даних

СУБД - Система Управління Базами Даних

ID – Identifier

PK – Primary Key

SQL - Structured Query Language

ВСТУП

На сьогоднішній день проектування баз даних набуло важливого значення для багатьох організацій, які для підвищення продуктивності своєї роботи використовують комп'ютерні технології. Бази даних стали основою інформаційних систем, а їх використання становиться невіддільною частиною функціонування будь-яких підприємств.

Метою створення БД є вдосконалення процесу управління діяльністю підприємства шляхом побудови єдиного програмно-технічного комплексу на основі електронного документообігу. У базу даних програми внесені дані про службовців різних спеціалізацій, про марки машин, що утримуються на підприємстві і дані про маршрути перевезення. Програма дає можливість додавати, зберігати й видаляти дані про службовців, наявні одиниці автотранспорту, пробіг та ремонти.

Завданням курсової роботи є проектування бази даних та підготовка усіх необхідних запитів для роботи з нею. Практичним завданням є відточення навичок SQL-програмування, проектування баз даних, дослідження принципів розробки реляційних баз даних на прикладі проектування та створення «БД Автопідприємства», набуття навичок створення запитів у СУБД MySQL Workbench.

В ході виконання роботи було проведено аналізу предметної області, згенеровано таблиці «БД Автопідприємства» та побудовані запити в даній БД.

При виконанні роботи було використано таке програмне забезпечення: операційна система Windows 8.1, СУБД MySQL Workbench 6.3 CE, веб-браузер Google Chrome для пошуку інформації на веб-сайтах, текстовий редактор Microsoft Word 2016 – для оформлення пояснювальної записки.

Пояснювальна записка складається зі вступу, трьох розділів, висновків, додатків та списку використаних джерел.

При написанні роботи використані підручники, навчальні посібники, роботи українських та закордонних науковців, фахівців-програмістів, чинне законодавство України, державні та закордонні стандарти в галузі інформаційних технологій.

РОЗДІЛ 1
ПОСТАНОВКА ЗАДАЧІ

1.1. Умова задачі

Автопідприємство міста займається організацією пасажирських і вантажних перевезень у місті. У віданні підприємства знаходиться автотранспорт різного призначення: автобуси, таксі, маршрутні таксі та інший легковий транспорт. Вантажний транспорт, транспорт допоміжного характеру представлені різними марками. Кожна з перерахованих категорій транспорту має характеристики, властиві тільки цій категорії: наприклад, до характеристик тільки вантажного транспорту відноситься вантажопідйомність, пасажирський транспорт характеризується місткістю тощо. З плином часу, з одного боку, транспорт старіє і списується (можливо, продається), а з іншого, - підприємство поповнюється новим автотранспортом.

Підприємство має штат водіїв, закріплених за автомобілями (за одним автомобілем може бути закріплено більш одного водія). Обслуговуючий персонал (техніки, зварники, слюсарі, складальники тощо) займається технічним обслуговуванням автомобільної техніки, при цьому різні перераховані вище категорії також можуть мати унікальні для даної категорії атрибути. Обслуговуючий персонал і водії об'єднується в бригади, якими керують бригадири, далі йдуть майстри, потім начальники ділянок і цехів. У віданні підприємства знаходяться об'єкти гаражного господарства (цеху, гаражі, бокси тощо), де міститься і ремонтується автомобільна техніка.

Пасажирський автотранспорт (автобуси, маршрутні таксі) перевозить пасажирів за визначеними маршрутами, за кожним з них закріплені окремі одиниці автотранспорту. Ведеться облік числа перевезених пасажирів, на підставі чого проводиться перерозподіл транспорту з одного маршруту на інший. Враховується також пробіг, число ремонтів і витрати на ремонт по всьому автотранспорту, обсяг вантажних перевезень для вантажного транспорту,

інтенсивність використання транспорту допоміжного призначення. Враховується інтенсивність роботи бригад по ремонту (число ремонтів, обсяг виконаних робіт), число замінених і відремонтованих вузлів і агрегатів (двигунів, КП, мости, шасі тощо) по кожній автомашині, і сумарно по ділянці, цеху та підприємству.

Види запитів в інформаційній системі:

1. Отримати дані про автопарк підприємства.

2. Отримати перелік і загальне число водіїв по підприємству, або для зазначеної автомашині.

3. Отримати розподіл водіїв по автомобілях.

4. Отримати дані про розподіл пасажирського автотранспорту по маршрутах.

5. Отримати відомості про пробіг автотранспорту певної категорії або конкретної автомашини за вказаний день, місяць та рік.

6. Отримати дані про число ремонтів та їх вартості для автотранспорту певної категорії, окремої марки автотранспорту або вказаної автомашини за вказаний період.

7. Отримати дані про підпорядкованість персоналу.

8. Отримати відомості про наявність гаражного господарства в цілому і по кожній категорії транспорту.

9. Отримати дані про розподіл автотранспорту на підприємстві.

10. Отримати відомості про вантажоперевезення, виконаних зазначеною автомашиною за встановлений період.

11. Отримати дані про число використаних для ремонту вказаних вузлів і агрегатів для транспорту певної категорії, окремої марки автотранспорту або конкретної автомашини за вказаний період.

12. Отримати відомості про отримання та списання автотехніки за вказаний період.

13. Отримати склад підлеглих зазначеного бригадира, майстра тощо.

14. Отримати дані про роботи, виконані зазначеним фахівцем (зварником, слюсарем тощо) за означений період в цілому і по конкретній автомашині.

1.2. Обґрунтування вибору методів та інструментів реалізації

Першим кроком у реалізації будь-якого програмного продукту є вибір мови та середовища розробки.

Звичайно, оскільки важливість баз даних стрімко зростає, РСУБД набирають популярність. Серед них найпопулярнішими є MySQL та SQL Server, які виконують однакову функцію, хоча й мають різні області використання. Тому перш ніж почати розробку програмного продукту необхідно визначитися з СУБД, у якій буде реалізовано проект.

Як зазначає авторка статті «Реляційні СУБД – порівняння MySQL і SQL сервера», існує декілька ключових відмінностей між цими двома системами які в даному програмному продукті грають важливу роль:

- Синтаксис. За допомогою інформаційного ресурсу «Порівняння різних реалізацій SQL» можна порівняти наявні функції в обох СУБД та побачити яким чином вони реалізуються. Набувши навичок роботи з MySQL протягом семестру цей набір таблиць допоможе освоїтися у роботі з SQL Server.
- Скасування запитів. Не всім відомо що MySQL не дозволяє скасовувати запити посеред їх виконання. Це означає що як тільки команда була запущена на виконання, залишається тільки сподіватися що можливі збитки є оборотними. В даній ситуації SQL Server без лишніх труднощів дозволив би зупинити запит посеред його виконання. В моєму випадку зважаючи на апаратне забезпечення яке я використовую, ця різниця є досить важливою.

- Вартість. MySQL має перевагу в тому що ця система розповсюджується безкоштовно. Хоча ціна для юридичних осіб на MS SQL Server виявляється неприємною для більшої частини організацій, Microsoft пропонує безкоштовну версію продукту.

Володимир Драч у своїй статті також перераховує декілька недоліків обох СУБД. Так наприклад, MySQL «пропонує велику кількість функцій», «прекрасно документована» та «може працювати з іншими БД, включаючи DB2 та Oracle», проте в ній «відсутня вбудована підтримка XML або OLAP», «для безкоштовної версії доступна тільки платна підтримка» і «доведеться витратити багато часу і зусиль, зоб змусити MySQL виконувати нескладні задачі, хоча інші системи роблять це автоматично». З іншої сторони Драч зазначає, що MS SQL Server «простий у використанні», його «поточна версія працює швидко і стабільно», «движок надає можливість регулювати й відстежувати рівні продуктивності, які допомагають знизити використання ресурсів».

Таким чином, дана БД реалізована в СУБД MySQL за допомогою середовища розробки MySQL Workbench 6.3, яке в свою чергу надає засоби для налаштування, спостереження й адміністрування баз даних і за допомогою якого можна розгортати, відстежувати та оновлювати компоненти рівня даних, що використовуються вашими додатками, а також створювати запити і скрипти.

РОЗДІЛ 2
АРХІТЕКТУРА ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ БД

2.1. Аналіз функціонування та організаційні засади підприємства

Для реалізації даного програмного продукту необхідно спроектувати БД автопідприємства міста. Для цього проаналізуємо умову та вимоги завдання і на їх основі створимо таблиці.

До основних функціональних завдань підприємства відносяться: пасажирські та вантажні перевезення, облік персоналу(водіїв, слюсарів, зварників, тощо), контроль маршрутів, ремонт автотранспорту.

З умови видно, що для даної БД необхідно створити таблиці для Автомобілів, Працівників, Вантажоперевезень, Пробігу, Маршрутів, Гаражних господарств.

2.2. Проектування структури бази даних

В результаті аналізу функціонування та організаційні засади підприємства, можна зробити висновок про таблиці, що нам знадобляться.

Діаграма БД зображена на рисунку 2.1.

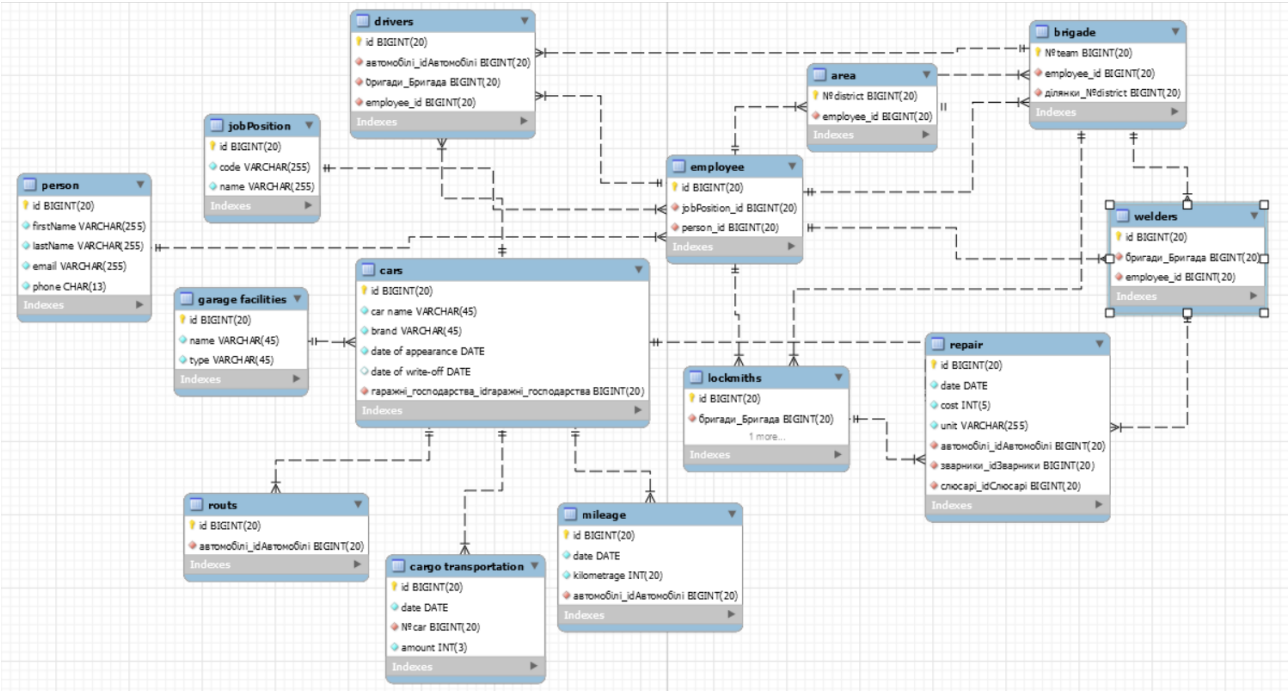

Рис. 2.1 ERD БД Автопідприємства

2.3. Життєвий цикл БД

1. Попереднє планування

На даному етапі була сформована модель структури бази даних, визначені сутності, проаналізовано зв’язки між сутностями на основі вимог до даної БД.

2. Перевірка здійсненності

Для розробки даної системи необхідно мати персональний комп’ютер, на якому інстальовано інструменти для розробки бази даних мовою програмування MySQL. Розробник має бути знайомий з даною мовою програмування. Наведені вимоги виконано, отже завдання є технічно здійсненним.

3. Визначення вимог

Вимоги до даної БД були сформовані замовником, зокрема вказана структура організації, види інформаційних запитів, які повинні бути передбачені у даній системі.

4. Проектування

Для реалізації технічного завдання обрано реляційну модель бази даних. В якості середовища розробки обрано MySQL Workbench. Проаналізовано сутності, визначені атрибути, ключові поля, типи зв’язків, побудовано EER-діаграму. Протестовано роботу системи на тестових даних.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ПРОГРАМНОЇ ВЗАЄМОДІЇ З БД

3.1. Інструкція користувача

Для того щоб користувач мав можливість редагувати та додавати дані в БД йому потрібно мати будь-який редактор MySQL. В даній інструкції вважаємо, що цим редактором є MySQL Workbench 6.3.

Перш за все необхідно створити сервер або під єднатися до існуючого. Далі треба створити саму БД на цьому сервері. Для цього створюємо новий запит і копіюємо код з Додатку А під назвою “CREATE_BD”. Натискаємо F5, або кнопку “Execute”, що запустити виконання програми. Для заповнення цієї БД далі аналогічно копіюємо частину коду з додатку під назвою “INSERT_BD”. Запустимо виконання. Для виконання запитів необхідно аналогічно спопіювати та запустити відповідну частину коду під назвою “REQUESTS_BD”. Вже працюючи в MySQL, для виконати окремий шматок коду досить виділити його та запустити. Фільтри та сортування можна редагувати, додаючи або прибираючи відповідні умови та замінюючи відповідні значення.

Детальніше розглянемо кожний запит окремо у підрозділах 3.2.1 – 3.2.13.

3.2. Реалізація механізмів SQL. SQL-запити

В результаті роботи було отримано 11 таблиць, які можна розділити на декілька підтипів за функціоналом. Основними таблицями вважатимемо таблиці *Cars*, одна з таблиць працівників, наприклад *Drivers*, оскільки з ними зв’язана більшість таблиць. Опис кожної з цих таблиць представлено у таблицях 3.1 та 3.2

Таблиця 3.1 Опис таблиці *Cars*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	унікальний ідентифікатор
	car name	varchar(45)	назва машини
	brand	varchar(45)	марка машини
	date of appearance	date	дата появи машини у підприємсті
	date of write-off	date	дата списання автомобіля(null, якщо авомобіль не списано)
	type	BIGINT(20)	Тип машини від його функцій(вантажівка, автобус, тощо).

Таблиця 3.2 Опис таблиці *Drivers*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	унікальний ідентифікатор
	Cars	BIGINT(20)	пов’язана машина
	brigade	BIGINT(20)	пов’язана бригада
	employee_id	BIGINT(20)	Ідентифікатор працівника

Таким чином у таблиці *Cars* зберігається основна інформація про кожну індивідуальну машину, а в таблиці *Drivers* – про кожного водія.

Опишемо інші таблиці, що необхідно створити для виконання завдань, споміжні таблиці, додамо до них опис.

Бригади – таблиця для обліку бригад, у які об’єднуються працівники.

Опис таблиці наведено у таблиці 3.3

Таблиця 3.3 Опис таблиці *Brigade*

	Назва поля	**Тип**	**Опис**
PK	№team	BIGINT(20)	унікальний номер бригади
	employee_id	BIGINT(20)	Ідентифікатор головного по бригаді – бригадира
	district	BIGINT(20)	ділянка, до якої належить бригада

Ділянка - це найвища управлінська одиниця в структурі підприємста. В ділянку входять кілька бригад.

Опис Ділянок у таблиці 3.4

Таблиця 3.4 Опис таблиці *Area*

	Назва поля	**Тип**	**Опис**
PK	№district	BIGINT(20)	унікальний номер ділянки
	employee_id	BIGINT(20)	Ідентифікатор головного по ділянці

Ведеться облік вантажоперевезень, що виконуються вантажівками для кожного дня та окремої машини.

Опис таблиці Вантажоперевезення знаходиться у таблиці 3.5

Таблиця 3.5 Опис таблиці *Cargo transportation*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор вантажоперевезення
	date	date	дата, коли було здійснено вантажоперевезення
	№car	BIGINT(20)	номер машини, якою було виконано вантажоперевезення
	amount	INT(3)	число вантажу у тонах, що перевезла машина у цю дату

Для машин ведеться облік пробігу. Пробіг записується окремо для кожної машини в кожен окремий день

Інформація про Пробіг у таблиці 3.6

Таблиця 3.6 Опис таблиці *Mileage*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор пробігу
	date	date	дата запису пробігу автомобіля
	kilometrage	INT(20)	кілометраж
	car_id	BIGINT(20)	машина, що здійснила поїздку

Окрім водіїв у штаті містяться робітники, що займаються ремонтними роботами. Це слюсарі та зварники. Опис цих таблиць знаходиться у таблицях 3.7 та 3.8.

Таблиця 3.7 Опис таблиці *Lockmiths*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор слюсаря
	employee_id	BIGINT(20)	ідентифікатор працівника
	Brigade_id	BIGINT(20)	бригада, в яку входить конкретний працівник

Таблиця 3.8 Опис таблиці *Welders*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор зварника
	employee_id	BIGINT(20)	ідентифікатор працівника
	Brigade_id	BIGINT(20)	бригада, в яку входить конкретний працівник

Зварники та слюсарі займаються ремонтом автомобілей, заміною чи відновленням певних вузлів чи агрегатів. Також ведеться облік марок авто, що ремонтуються, адже часто однакові деталі однакових марок машин можна без проблем встановити різним автомобілям. Ведеться облік вартості ремонту та запчастин, що ремонтуються.

Опис таблиці *Repair* знаходиться у таблиці 3.9.

Таблиця 3.9 Опис таблиці *Repair*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор ремонту
	car_id	BIGINT(20)	ідентифікатор машини, що ремонтується
	date	date	дата ремонту
	cost	INT(5)	вартість ремонту
	unit	VARCHAR(255)	назва запчастини, що ремонтується
	lockmiths_id	BIGINT(20)	ідентифікатор зварника, що проводив ремонт
	welders_id	BIGINT(20)	ідентифікатор слюсаря, що проводив ремонт

Вся техніка міститься в гаражних господарствах. Інформація про гаражні господарства знаходиться в таблиці 3.10

Таблиця 3.10 Опис таблиці *Garage facilities*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор гаражного господарства
	name	varchar(45)	назва гаражного господарства
	type	varchar(45)	Тип автомобілей, що знаходяться в цьому гаражному господарстві

Пасажирський автотранспорт (автобуси, маршрутні таксі) перевозить пасажирів за визначеними маршрутами, за кожним з них закріплені окремі одиниці автотранспорту.

Опис таблиці *Routs* міститься в таблиці 3.11.

Таблиця 3.11 Опис таблиці *Routs*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор
	car_id	BIGINT(20)	ідентифікатор машини, що їздить за певним маршрутом

Пасажирський автотранспорт (автобуси, маршрутні таксі) перевозить пасажирів за визначеними маршрутами, за кожним з них закріплені окремі одиниці автотранспорту.

Вся наявна інформація про персонал, наявний на фірмі, можливі варіанти працевлаштування та зіставлення цих таблиць у робітників подано в описі наступних таблиць 3.12 – 3.14.

Таблиця 3.12 Опис таблиці *Person*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор
	firstName	VARCHAR(255)	ім’я працівника
	lastName	VARCHAR(255)	прізвище працівника
	email	VARCHAR(255)	Email працівника
	phone	CHAR(13)	Номер телефону працівника

Таблиця 3.13 Опис таблиці *JobPosition*

	Назва поля	**Тип**	**Опис**
PK	ID	BIGINT(20)	ідентифікатор робочого місця
	code	varchar(45)	Код робочого місця
	name	varchar(45)	Назва робочого місця

Повний код запитів можна побачити у Додатку А. Тому розглянемо кожний запит конкретніше.

3.2.1. Запит 1

Завдання: 1. Отримати дані про автопарк підприємства.

Щоб отримати дані про автопарк, запустимо процедуру task1

```
delimiter $$
-- drop procedure if exists task1;
create procedure task1()
begin
     select `car name` from cars;
end $$
delimiter $$
```

3.2.2. Запит 2

Завдання: Отримати перелік і загальне число водіїв по підприємству, або для зазначеної автомашині.

Щоб отримати перелік і загальне число водіїв по підприємству, запустимо процедуру task2_1

```
delimiter $$
-- drop procedure if exists task2_1;
create procedure task2_1()
begin
     select person.firstName, person.lastName, jobPosition.name
     from employee
     join person on employee.person_id = person.id
     join jobPosition on employee.jobPosition_id = jobPosition.id
     where jobPosition.name='водій';
    select count(*)
     from employee
     join person on employee.person_id = person.id
```

```
     join jobPosition on employee.jobPosition_id = jobPosition.id
     where jobPosition.name='водій';
end$$
DELIMITER $$
```

Щоб отримати перелік і загальне число водіїв, відповідальних за вказану машину, запустимо процедуру task2_2

```
delimiter $$
-- drop procedure if exists task2_2;
create procedure task2_2(IN name_car VARCHAR(45))
begin
    select person.firstName, person.lastName, jobPosition.name
     from employee
     join person on employee.person_id = person.id
     join jobPosition on employee.jobPosition_id = jobPosition.id
    join drivers on drivers.employee_id=employee.person_id
    join cars on drivers.автомобілі_idАвтомобілі=cars.id
    where jobPosition.name='водій' and cars.`car name`=name_car;
    select count(*)
     from employee
     join person on employee.person_id = person.id
     join jobPosition on employee.jobPosition_id = jobPosition.id
    join drivers on drivers.employee_id=employee.person_id
    join cars on drivers.автомобілі_idАвтомобілі=cars.id
    where jobPosition.name='водій' and cars.`car name`=name_car;
end$$
DELIMITER $$
```

3.2.3. Запит 3

Завдання: Отримати розподіл водіїв по автомобілях.

Запустимо процедуру task3

```
delimiter $$
-- drop procedure if exists task3;
create procedure task3()
begin
     select person.firstName, person.lastName, jobPosition.name,
cars.`car name`
     from employee
     join person on employee.person_id = person.id
     join jobPosition on employee.jobPosition_id = jobPosition.id
    join drivers on drivers.employee_id=employee.person_id
    join cars on drivers.автомобілі_idАвтомобілі=cars.id
     where jobPosition.name='водій';
end$$
```

```
DELIMITER $$
```

3.2.4. Запит 4

Завдання: Отримати дані про розподіл пасажирського автотранспорту по маршрутах.

Запустимо процедуру task4

```
delimiter $$
-- drop procedure if exists task4;
create procedure task4()
begin
     select cars.`car name`, routs.id as 'маршрути' from routs
    join cars on routs.автомобілі_idАвтомобілі=cars.id;
end$$
DELIMITER $$
```

3.2.5. Запит 5

Завдання: Отримати відомості про пробіг автотранспорту певної категорії або конкретної автомашини за вказаний день, місяць та рік.

Для відомостей про пробіг автівок певної категорії запустимо процедуру task5_1

```
delimiter $$
-- drop procedure if exists task5_1;
create procedure task5_1(IN type_car VARCHAR(45))
begin
     select mileage.date, mileage.kilometrage, cars.`car name`, `garage
facilities`.type
    from mileage
    join cars on cars.id = mileage.автомобілі_idАвтомобілі
    join `garage facilities` on
cars.гаражні_господарства_idгаражні_господарства = `garage facilities`.id
    where `garage facilities`.type = type_car;
end$$
DELIMITER $$
```

Для відомостей про пробіг конкретної машини за вказаний період запустимо процедуру task5_2

```
delimiter $$
-- drop procedure if exists task5_2;
```

```
create procedure task5_2(IN car VARCHAR(45), IN date1 DATE, IN date2
DATE)
begin
     select mileage.date, mileage.kilometrage, mileage.date, cars.`car
name`
    from mileage
    join cars on cars.id = mileage.автомобілі_idАвтомобілі
    where cars.`car name` = car and mileage.date between date1 and date2;
end$$
DELIMITER $$
```

3.2.6. Запит 6

Завдання: Отримати дані про число ремонтів та їх вартості для автотранспорту певної категорії, окремої марки автотранспорту або вказаної автомашини за вказаний період.

Щоб отримати дані про число ремонтів та їх вартості для авто певної категорії запустимо task6_1

```
delimiter $$
-- drop procedure if exists task6_1;
create procedure task6_1(IN type_car VARCHAR(45))
begin
     select count(cost), sum(cost)
    from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    join `garage facilities` on
cars.гаражні_господарства_idгаражні_господарства = `garage facilities`.id
    where `garage facilities`.type = type_car;
end$$
DELIMITER $$
```

Щоб отримати дані про число ремонтів та їх вартості для авто певної марки запустимо task6_2

```
delimiter $$
-- drop procedure if exists task6_2;
create procedure task6_2(IN marka_car VARCHAR(45))
begin
     select count(cost), sum(cost)
    from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where cars.brand = marka_car;
end$$
DELIMITER $$
```

Щоб отримати дані про число ремонтів та їх вартості для певного авто за вказаний період запустимо task6_3

```
delimiter $$
-- drop procedure if exists task6_3;
create procedure task6_3(IN car VARCHAR(45), IN date1 DATE, IN date2
DATE)
begin
     select count(cost), sum(cost)
    from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where cars.`car name` = car and `repair`.date between date1 and
date2;
end$$
DELIMITER $$
```

3.2.7. Запит 7

Завдання: Отримати дані про підпорядкованість персоналу.

Запустимо task7

```
delimiter $$
-- drop procedure if exists task7;
create procedure task7(IN firstName_person varchar(45),in lastName_person
varchar(45))
begin
     if (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'водій')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join brigade on employee.id = brigade.employee_id
        where jobPosition.name = 'Бригадир' and brigade.`№team` = 1
           union
        select "Водій", null;
        end;

    elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
```

```
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'зварник')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join brigade on employee.id = brigade.employee_id
        where jobPosition.name = 'Бригадир' and brigade.`№team` = 2
           union
        select "зварник", null;
        end;

    elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'слюсар')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join brigade on employee.id = brigade.employee_id
        where jobPosition.name = 'Бригадир' and brigade.`№team` = 2
           union
        select "слюсар", null;
        end;

     elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Бригадир')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join area on employee.id = area.employee_id
        where jobPosition.name = 'Начальник ділянки'
           union
        select "Бригадир", null;
        end;

     elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
```

```
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Начальник ділянки' )) then begin

           select firstName_person, lastName_person
           union
        select "Начальник ділянки", null;
        end;

     else select "some problems";
     end if;
end$$
DELIMITER $$
```

3.2.8. Запит 8

Завдання: Отримати відомості про наявність гаражного господарства в цілому і по кожній категорії транспорту.

Щоб дізнатись про наявність гаражного господарства в цілому запустимо task8_1

```
delimiter $$
-- drop procedure if exists task8_1;
create procedure task8_1()
begin
     select if((select count(*) from `garage facilities`)!=0, 'гаражні
господарства є', 'гаражних господарств немає');
end$$
DELIMITER $$
```

Щоб дізнатись про наявність гаражного господарства для конкретної категорії запустимо task8_2

```
delimiter $$
-- drop procedure if exists task8_2;
create procedure task8_2(IN type_car varchar(45))
begin
     select if((select count(*) from `garage facilities` where type =
type_car)!=0, 'гаражне господарство такого типу є', 'гаражного
господарства такого типу немає');
end$$
DELIMITER $$
```

3.2.9. Запит 9

Завдання: Отримати дані про розподіл автотранспорту на підприємстві.

Запустимо task9

```
delimiter $$
-- drop procedure if exists task9;
create procedure task9()
begin
     select cars.`car name`, `garage facilities`.type, `garage
facilities`.name as `Гаражне господарство` from cars
     join `garage facilities` on `garage facilities`.id =
cars.гаражні_господарства_idгаражні_господарства;
end$$
DELIMITER $$
```

3.2.10. Запит 10

Завдання: Отримати відомості про вантажоперевезення, виконаних зазначеною автомашиною за встановлений період.

Щоб отримати перелік і загальне число тварин, яким необхідний певний тип кормів з відповідними фільтрами запустимо

```
delimiter $$
-- drop procedure if exists task10;
create procedure task10(IN car VARCHAR(45), IN date1 DATE, IN date2 DATE)
begin
     select `cargo transportation`.`id`,`date`,amount  from `cargo
transportation`
    join cars on cars.id = `cargo transportation`.`№car`
    where `date` between date1 and date2 AND cars.`car name` = car;
end$$
DELIMITER $$
```

3.2.11. Запит 11

Завдання: Отримати дані про число використаних для ремонту вказаних вузлів і агрегатів для транспорту певної категорії, окремої марки автотранспорту або конкретної автомашини за вказаний період.

Щоб отримати дані про число використаних для ремонту вказаних вузлів і агрегатів для транспорту певної категорії запустимо task11_1

```
delimiter $$
-- drop procedure if exists task11_1;
create procedure task11_1(IN element_car VARCHAR(45), IN type_car
varchar(45))
```

```
begin
    select count(cost) from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    join `garage facilities` on `garage facilities`.id =
cars.гаражні_господарства_idгаражні_господарства
    where `repair`.unit = element_car AND `garage facilities`.type =
type_car;
end$$
DELIMITER $$
```

Щоб отримати дані про число використаних для ремонту вказаних вузлів і агрегатів для транспорту окремої марки запустимо task11_2

```
delimiter $$
-- drop procedure if exists task11_2;
create procedure task11_2(IN element_car VARCHAR(45), IN marka_car
varchar(45))
begin
     select count(cost) from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where `repair`.unit = element_car AND cars.brand = marka_car;
end$$
DELIMITER $$
```

Щоб отримати дані про число використаних для ремонту вказаних вузлів і агрегатів для конкретної машини за вказаний період запустимо task11_3

```
delimiter $$
-- drop procedure if exists task11_3;
create procedure task11_3(IN element_car varchar(45), IN car VARCHAR(45),
IN date1 DATE, IN date2 DATE)
begin
     select count(cost) from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where `repair`.unit = element_car AND (`repair`.date between date1
and date2) and cars.`car name` = car;
end$$
DELIMITER $$
```

3.2.12. Запит 12

Завдання: Отримати відомості про отримання та списання автотехніки за вказаний період.

Запустимо task12

```
delimiter $$
-- drop procedure if exists task12;
create procedure task12(IN date1 DATE, IN date2 DATE)
begin
     select `car name`, `date of appearance`, `date of write-off` from
cars
    where `date of appearance` between date1 and date2 OR `date of write-
off` between date1 and date2;
end$$
DELIMITER $$
```

3.2.13. Запит 13

Завдання: Отримати склад підлеглих зазначеного бригадира, майстра тощо.

Запустимо task13

```
delimiter $$
-- drop procedure if exists task13;
create procedure task13(IN firstName_person varchar(45), IN
lastName_person varchar(45))
begin
     if (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    join brigade on employee.id = brigade.employee_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Бригадир' and brigade.`№team` = 1 )) then begin
           select "Бригадир", null
        union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        where jobPosition.name = 'водій';
        end;

     elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    join brigade on employee.id = brigade.employee_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Бригадир' and brigade.`№team` = 2 )) then begin
           select "Бригадир", null
        union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        where jobPosition.name = 'зварник'
```

```
        union
        select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        where jobPosition.name = 'слюсар';
        end;

     elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    join area on employee.id = area.employee_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Начальник ділянки')) then begin
           select "Начальник ділянки", null
        union
        select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join area on employee.id = area.employee_id
        where jobPosition.name = 'Начальник ділянки';
        end;

     else select "some problems";
    end if;
end$$
DELIMITER $$
```

3.2.14. Запит 14

Завдання: Отримати дані про роботи, виконані зазначеним фахівцем (зварником, слюсарем тощо) за означений період в цілому і по конкретній автомашині.

Щоб отримати дані про роботи, виконані зазначеним фахівцем (зварником, слюсарем тощо) за означений період в цілому запустимо task14_1

```
delimiter $$
-- drop procedure if exists task14_1;
create procedure task14_1(IN firstName_person varchar(45), IN
lastName_person varchar(45), IN car varchar(45))
begin
     select `repair`.date, cars.`car name`, unit from `repair`
    join cars on `repair`.автомобілі_idАвтомобілі = cars.id
    join welders on welders.id = `repair`.зварники_idЗварники
    join lockmiths on lockmiths.id = `repair`.слюсарі_idСлюсарі
    join employee on (lockmiths.employee_id = employee.id or
welders.employee_id = employee.id)
    join person on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
```

```
    where (firstName = firstName_person and lastName = lastName_person
and jobPosition.name = 'зварник' and cars.`car name` = car)
    or (firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'слюсар' and cars.`car name` = car);
end$$
DELIMITER $$
```

Щоб отримати дані про роботи, виконані зазначеним фахівцем (зварником, слюсарем тощо) по конкретній автомашині запустимо task14_2

```
delimiter $$
-- drop procedure if exists task14_2;
create procedure task14_2(IN firstName_person varchar(45), IN
lastName_person varchar(45), IN date1 DATE, IN date2 DATE)
begin
     select `repair`.date, cars.`car name`, unit from `repair`
    join cars on `repair`.автомобілі_idАвтомобілі = cars.id
    join welders on welders.id = `repair`.зварники_idЗварники
    join lockmiths on lockmiths.id = `repair`.слюсарі_idСлюсарі
    join employee on (lockmiths.employee_id = employee.id or
welders.employee_id = employee.id)
    join person on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where (firstName = firstName_person and lastName = lastName_person
and jobPosition.name = 'зварник' and `repair`.date between date1 and
date2)
    or (firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'слюсар' and `repair`.date between date1 and date2);
end$$
DELIMITER $$
```

3.3. Вимоги до апаратних і програмних засобів

Даний програмний продукт був протестований на комп’ютері з процесором Intel Core i7-4500U CPU @ 1.80GHz за допомогою ПЗ MySQL Workbench 6.3. Мінімальні вимоги для апаратного забезпечення: 512 MB пам’яті, x64 процесор: 1.4GHz типу AMD Opteron, AMD Athlon 64, Intel Xeon з підтримкою Intel EM64T або Intel Pentium IV з підтримкою EM64T.

3.4. Випробування розроблених програм

В даному підрозділі продемонстровано роботу запитів на конкретних випадках.

3.4.1 Запит 1

Отримати дані про автопарк підприємства.

call task1();

car name
ЛАЗ-697
Рута 22
Рута 23
Стрий Авто А075
Стрий Авто А0756
Богдан-3355
ГАЗ-51
ГАЗ-68
ЗІЛ-4331
Богдан А091

Рисунок 3.1 Таблиця task1

3.4.2 Запит 2

Отримати перелік і загальне число водіїв по підприємству, або для зазначеної автомашині.

call task2_1()

firstName	lastName	name
Альберт	Вескер	водій
Люцій	Блок	водій
Самсон	Вебер	водій
Тайлер	Дерден	водій
Закир	Ганди	водій
Нільс	Нільсон	водій
Осип	Дефо	водій

Рисунок 3.2 Таблиця task2_1

count(*)
7

Рисунок 3.3 Таблиця task2_1

call task2_2('Рута 23')

firstName	lastName	name
Альберт	Вескер	водій

Рисунок 3.4 Таблиця task2_2

count(*)
1

Рисунок 3.5 Таблиця task2_2

3.4.3. Запит 3

Отримати розподіл водіїв по автомобілях.

call task3();

firstName	lastName	name	car name
Альберт	Вескер	водій	Рута 23
Люцій	Блок	водій	Стрий Авто А075
Самсон	Вебер	водій	Стрий Авто А0756
Тайлер	Дерден	водій	ГАЗ-51
Закир	Ганди	водій	ГАЗ-68
Нільс	Нільсон	водій	ЗІЛ-4331
Осип	Дефо	водій	Богдан А091

Рисунок 3.6 Таблиця task3

3.4.4. Запит 4

Отримати дані про розподіл пасажирського автотранспорту по маршрутах.

```
call task4();
```

car name	маршрути
ЛАЗ-697	1
Рута 22	2
Рута 23	3
Стрий Авто А075	4

Рисунок 3.7 Таблиця task4

3.4.5. Запит 5

Отримати відомості про пробіг автотранспорту певної категорії або конкретної автомашини за вказаний день, місяць та рік.

```
call task5_1('Автобус');
```

date	kilometrage	car name	type
2017-01-01	100	Рута 23	автобус
2017-08-23	120	Стрий Авто А075	автобус
2018-08-23	150	Стрий Авто А0756	автобус

Рисунок 3.8 Таблиця task5_1

```
call task5_2('Стрий Авто А0756', '2016-01-01', '2019-01-01');
```

date	kilometrage	date	car name
2018-08-23	150	2018-08-23	Стрий Авто А0756

Рисунок 3.9 Таблиця task5_2

3.4.6. Запит 6

Отримати дані про число ремонтів та їх вартості для автотранспорту певної категорії, окремої марки автотранспорту або вказаної автомашини за вказаний період.

call task6_1('Автобус');

count(cost)	sum(cost)
2	18000

Рисунок 3.10 Таблиця task6_1

call task6_2('Рута');

count(cost)	sum(cost)
1	10000

Рисунок 3.11 Таблиця task6_2

call task6_3('Стрий Авто А075', '2016-07-06', '2020-08-06');

count(cost)	sum(cost)
1	8000

Рисунок 3.12 Таблиця task6_3

3.4.7. Запит 7

Отримати дані про підпорядкованість персоналу: робітники-бригадири - майстри - начальники ділянок і цехів.

call task7('Альберт','Вескер');

firstName_person	lastName_person
Альберт	Вескер
Микита	Венарт
Водій	NULL

Рисунок 3.13 Таблиця task7

3.4.8. Запит 8

Отримати відомості про наявність гаражного господарства в цілому і по кожній категорії транспорту.

call task8_1();

if((select count(*) from `garage facilities`)!=0,
гаражні господарства є

Рисунок 3.14 Таблиця task8_1

call task8_2('Таксі');

if((select count(*) from `garage facilities` where
гаражне господарство такого типу є

Рисунок 3.15 Таблиця task8_2

3.4.9. Запит 9

Отримати дані про розподіл автотранспорту на підприємстві.

```
call task9();
```

car name	type	Гаражне господарство
ЛАЗ-697	автобус	Г-1
Рута 22	автобус	Г-1
Рута 23	автобус	Г-1
Стрий Авто А075	автобус	Г-1
Стрий Авто А0756	автобус	Г-1
Богдан-3355	вантажівка	Г-2
ГАЗ-51	вантажівка	Г-2
ГАЗ-68	вантажівка	Г-2
ЗІЛ-4331	вантажівка	Г-2
Богдан А091	вантажівка	Г-2

Рисунок 3.16 Таблиця task9

3.4.10. Запит 10

Отримати відомості про вантажоперевезення, виконаних зазначеною автомашиною за встановлений період.

```
call task10('ГАЗ-51', '2015-07-06', '2020-08-06');
```

id	date	amount
1	2016-08-23	20

Рисунок 3.17 Таблиця task10

3.4.11. Запит 11

Отримати дані про число використаних для ремонту вказаних вузлів і агрегатів для транспорту певної категорії, окремої марки автотранспорту або конкретної автомашини за вказаний період.

```
call task11_1('двигун', 'Автобус');
```

count(cost)
1

Рисунок 3.18 Таблиця task11_1

```
call task11_2('двигун', 'Рута');
```

count(cost)
1

Рисунок 3.19 Таблиця task11_2

```
call task11_3('двигун', 'Богдан-3355', '2016-07-06', '2020-08-06');
```

count(cost)
1

Рисунок 3.20 Таблиця task11_3

3.2.12. Запит 12

Отримати відомості про отримання та списання автотехніки за вказаний період.

```
call task12('2015-07-23', '2020-08-23');
```

car name	date of appearance	date of write-off
ЛАЗ-697	2015-07-23	2018-08-23
Рута 22	2015-07-23	2018-08-23
Рута 23	2015-08-23	NULL
Стрий Авто А075	2016-07-23	NULL
Стрий Авто А0756	2016-07-23	NULL
Богдан-3355	2015-07-23	2018-07-23
ГАЗ-51	2015-07-23	NULL
ГАЗ-68	2015-07-23	NULL
ЗІЛ-4331	2016-07-23	NULL
Богдан А091	2016-07-23	NULL

Рисунок 3.21 Таблиця task12

3.2.13. Запит 13

Отримати склад підлеглих зазначеного бригадира, майстра тощо.

```
call task13('Олексій', 'Чижов');
```

Начальник ділянки	NULL
Начальник ділянки	NULL
Олексій	Чижов

Рисунок 3.22 Таблиця task13

3.2.14. Запит 14

Отримати дані про роботи, виконані зазначеним фахівцем (зварником, слюсарем тощо) за означений період в цілому і по конкретній автомашині.

call task14_1('Франц','Зайцев','Рута 22');

date	car name	unit
2018-08-20	Рута 22	двигун

Рисунок 3.23 Таблиця task14_1

call task14_2('Ясон','Масляков','2015-01-01','2021-01-01');

date	car name	unit
2018-02-23	Богдан-3355	двигун
2020-08-20	Богдан A091	шасі

Рисунок 3.24 Таблиця task14_

3.5. Опис тестової бази даних

На наступних рисунках наведено таблиці із бази даних

car name	brand	date of appearance	date of write-off	garage_id
ЛАЗ-697	ЛАЗ	2015-07-23	2018-08-23	1
Рута 22	Рута	2015-07-23	2018-08-23	1
Рута 23	Рута	2015-08-23	NULL	1
Стрий Авто А075	Стрий Авто	2016-07-23	NULL	1
Стрий Авто А0756	Стрий Авто	2016-07-23	NULL	1
Богдан-3355	Богдан	2015-07-23	2018-07-23	2
ГАЗ-51	ГАЗ	2015-07-23	NULL	2
ГАЗ-68	ГАЗ	2015-07-23	NULL	2
ЗІЛ-4331	ЗІЛ	2016-07-23	NULL	2
Богдан А091	Богдан	2016-07-23	NULL	2

Рисунок 3.1 Таблиця *Cars*

№team	employee_id	district_id
1	2	1
2	3	1
NULL	NULL	NULL

Рисунок 3.2 Таблиця *Brigade*

id	date	№car	amount
1	2016-08-23	7	20
2	2016-12-24	8	50
NULL	NULL	NULL	NULL

Рисунок 3.3 Таблиця *Cargo transportation*

id	car_id	brigade_id	employee_id
1	3	1	4
2	4	1	5
3	5	1	6
4	7	1	7
5	8	1	8
6	9	1	9
7	10	1	10
NULL	NULL	NULL	NULL

Рисунок 3.4 Таблиця *Drivers*

id	name	type
1	Г-1	автобус
2	Г-2	вантажівка
3	Г-3	таксі
NULL	NULL	NULL

Рисунок 3.5 Таблиця *Garage facilities*

№district	employee_id
1	1
NULL	NULL

Рисунок 3.6 Таблиця *Area*

id	brigade_id	employee_id
1	2	11
2	2	12
3	2	13
4	2	4
NULL	NULL	NULL

Рисунок 3.7 Таблиця *Welders*

id	car_id
1	1
2	2
3	3
4	4
NULL	NULL

Рисунок 3.8 Таблиця *Routs*

id	date	kilometrage	car_id
1	2017-01-01	100	3
2	2017-08-23	120	4
3	2018-08-23	150	5
4	2016-08-23	600	7
5	2016-12-24	650	8
NULL	NULL	NULL	NULL

Рисунок 3.9 Таблиця *Mileage*

id	date	cost	unit	car_id	welders_id	lockmiths_id
1	2018-08-20	10000	двигун	2	1	1
2	2017-07-23	8000	КП	4	2	2
3	2018-02-23	10000	двигун	6	3	3
4	2020-08-20	12000	шасі	10	1	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Рисунок 3.10 Таблиця *Repair*

id	brigade_id	employee_id
1	2	14
2	2	15
3	2	16
NULL	NULL	NULL

Рисунок 3.11 Таблиця *Lockmiths*

id	firstName	lastName	email	phone
1	Олексій	Чижов	aderf@net	+380111111111
2	Микита	Венарт	gjdo@net	+380222222222
3	Альберт	Розмареллі	NNN@net	+380994765937
4	Альберт	Вескер	alb@net	+380994768473
5	Люцій	Блок	goj@net	+380969475937
6	Самсон	Вебер	out@net	+380994000007
7	Тайлер	Дерден	into@net	+380111115937
8	Закир	Ганди	up@net	+380992222937
9	Нільс	Нільсон	down@net	+380994733337
10	Осип	Дефо	hot@net	+380944445937
11	Франц	Зайцев	intro@net	+380994765555
12	Потап	Капиця	uotro@net	+380966665937
13	Эдмунд	Клінтон	AAA@net	+380994767777
14	Рамон	Лайтман	BBB@net	+380994765888
15	Клод	Шенон	CCC@net	+380999995937
16	Ясон	Масляков	DDD@net	+380994760037
NULL	NULL	NULL	NULL	NULL

Рисунок 3.12 Таблиця *Person*

id	code	name
1	captain	Начальник ділянки
2	brigadier	Бригадир
3	driver	водій
4	welders	зварник
5	lockmiths	слюсар
NULL	NULL	NULL

Рисунок 3.13 Таблиця *JobPosition*

id	jobPosition_id	person_id
1	1	1
2	2	2
3	2	3
4	3	4
5	3	5
6	3	6
7	3	7
8	3	8
9	3	9
10	3	10
11	4	11
12	4	12
13	4	13
14	5	14
15	5	15
16	5	16
NULL	NULL	NULL

Рисунок 3.14 Таблиця *Employee*

ВИСНОВКИ

В результаті виконання курсової роботи було розроблено Базу Даних Автопідприємства. В ході роботи над програмним продуктом «БД Автопідприємства» було закріплено знання отримані під час вивчення курсу «Організація баз даних та знань», а також набуто навичок роботи з СУБД MySQL та середовищем розробки MySQL Workbench 6.3.

Всі пункти технічного завдання виконані. Розроблений програмний продукт задовольняє всі пункти, які поставив замовник, хоча і потребує деяких уточнень через некоректно поставлені умови в деяких пунктах.

В першому розділі було описана умова задачі, а також розглядались використані інструменти розробки.

В другому розділі було описано та створено спроектовано ієрархію БД на основі досліджень праць українських та закордонних науковців й віхівців – програмістів, а також власних знань, здобутих протягом курсу.

В третьому розділі було надано опис використаних таблиць, складено керівництво користувача даного продукту, а також проведено тестування на імітаційних даних, в процесі якого було виявлено декілька недоліків.

Надалі розробка може бути удосконалена шляхом додавання тригерів та процедур, які значно полегшить навігацію та роботу з БД. Також значним вдосконаленням може бути імплементація зручного інтерфейсу, який спростить роботу з «БД Автопідприємства» для некваліфікованих користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Elena B.. Реляционные СУБД – сравнение MySQL и SQL сервер. – Hostinger, 2018 [Електронний ресурс] - https://www.hostinger.com.ua/rukovodstva/reljacionnye-subd-sravnenie-mysql-i-sql-server/#MySQL-SQL

2. Troels Arvin. Comparison of different SQL implementations [Електронний ресурс] - http://troels.arvin.dk/db/rdbms/

3. Драч. В. Сравнение современных СУБД – 2017 [Електронний ресурс] - http://drach.pro/blog/hi-tech/item/145-db-comparison

4. Гусев А.В., Дмитриев А.Г.. Microsoft SQL Server против MySQL в медицинских информационных системах – 2004 [Електронний ресурс] - http://citforum.ru/database/articles/msmysql/

5. Microsoft. Довідник по Transact-SQL (компонент Database Engine) [Електронний ресурс] - https://docs.microsoft.com/en-us/sql/t-sql/language-reference?view=sql-server-ver15

6. Docs.Data Довідник по SQL. [Електронний ресурс] - https://docs.data.world/documentation/sql/concepts/basic/intro.html

7. Гарсиа-Молина Г., Ульман Д., Уидом Д. Системы баз данных. Полный курс.: Пер. с англ. - М.: Издательский дом "Вильямс", 2004. - 1088 с.

8. Карпова Т.С. Базы данных: модели, разработка, реализация. – СПб.: Питер, 2001. – 304 с.

9. Голицина О.Л., Максимов Н.В., Попов И.И. Базы данных: навчальний посібник. – М.: ФОРУМ: ИНФРА-М, 2005. 352с: ил. (Профессиональное образование).

10. Види програм і програмних документів: ДСТ 19.101-77 ЕСПД

11. Документація. Звіти у сфері науки і техніки. Структура і правила оформлення: ДСТУ 3008-95 (ГОСТ 7.32-91). – Чинний від 2006-01-01. К.: Держспоживстандарт України, 2006. – 231 с. – (Національний стандарт України).

12. Схеми алгоритмів і програм. Правила виконання. ДСТ 19.002-80 ЕСПД.

13. Посилання на GitHub репозиторій. [Електронний ресурс] - https://github.com/AfJhAD/KR.git

ДОДАТОК А

Лістинг ПЗ

“REQUESTS_BD"

```
use mydb;

-- 1
delimiter $$
-- drop procedure if exists task1;
create procedure task1()
begin
     select `car name` from cars;
end $$
delimiter $$
call task1()$$

-- 2_1
delimiter $$
-- drop procedure if exists task2_1;
create procedure task2_1()
begin
     select person.firstName, person.lastName, jobPosition.name
     from employee
     join person on employee.person_id = person.id
     join jobPosition on employee.jobPosition_id = jobPosition.id
     where jobPosition.name='водій';
    select count(*)
     from employee
     join person on employee.person_id = person.id
     join jobPosition on employee.jobPosition_id = jobPosition.id
     where jobPosition.name='водій';
end$$
DELIMITER $$
call task2_1()$$

-- 2_2
delimiter $$
-- drop procedure if exists task2_2;
create procedure task2_2(IN name_car VARCHAR(45))
begin
    select person.firstName, person.lastName, jobPosition.name
     from employee
     join person on employee.person_id = person.id
     join jobPosition on employee.jobPosition_id = jobPosition.id
    join drivers on drivers.employee_id=employee.person_id
    join cars on drivers.автомобілі_idАвтомобілі=cars.id
    where jobPosition.name='водій' and cars.`car name`=name_car;
    select count(*)
```

```
      from employee
      join person on employee.person_id = person.id
      join jobPosition on employee.jobPosition_id = jobPosition.id
    join drivers on drivers.employee_id=employee.person_id
    join cars on drivers.автомобілі_idАвтомобілі=cars.id
    where jobPosition.name='водій' and cars.`car name`=name_car;
end$$
DELIMITER $$
call task2_2('Рута 23')$$

-- 3
delimiter $$
-- drop procedure if exists task3;
create procedure task3()
begin
      select person.firstName, person.lastName, jobPosition.name,
cars.`car name`
      from employee
      join person on employee.person_id = person.id
      join jobPosition on employee.jobPosition_id = jobPosition.id
    join drivers on drivers.employee_id=employee.person_id
    join cars on drivers.автомобілі_idАвтомобілі=cars.id
      where jobPosition.name='водій';
end$$
DELIMITER $$
call task3$$

-- 4
delimiter $$
-- drop procedure if exists task4;
create procedure task4()
begin
      select cars.`car name`, routs.id as 'маршрути' from routs
    join cars on routs.автомобілі_idАвтомобілі=cars.id;
end$$
DELIMITER $$
call task4$$

-- 5_1
delimiter $$
-- drop procedure if exists task5_1;
create procedure task5_1(IN type_car VARCHAR(45))
begin
      select mileage.date, mileage.kilometrage, cars.`car name`, `garage
facilities`.type
    from mileage
    join cars on cars.id = mileage.автомобілі_idАвтомобілі
    join `garage facilities` on
cars.гаражні_господарства_idгаражні_господарства = `garage facilities`.id
    where `garage facilities`.type = type_car;
```

```
end$$
DELIMITER $$
call task5_1('Автобус')$$

-- 5_2
delimiter $$
-- drop procedure if exists task5_2;
create procedure task5_2(IN car VARCHAR(45), IN date1 DATE, IN date2
DATE)
begin
     select mileage.date, mileage.kilometrage, mileage.date, cars.`car
name`
    from mileage
    join cars on cars.id = mileage.автомобілі_idАвтомобілі
    where cars.`car name` = car and mileage.date between date1 and date2;
end$$
DELIMITER $$
call task5_2('Стрий Авто А0756', '2016-01-01', '2019-01-01')$$

-- 6_1
delimiter $$
-- drop procedure if exists task6_1;
create procedure task6_1(IN type_car VARCHAR(45))
begin
     select count(cost), sum(cost)
    from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    join `garage facilities` on
cars.гаражні_господарства_idгаражні_господарства = `garage facilities`.id
    where `garage facilities`.type = type_car;
end$$
DELIMITER $$
call task6_1('Автобус')$$

-- 6_2
delimiter $$
-- drop procedure if exists task6_2;
create procedure task6_2(IN marka_car VARCHAR(45))
begin
     select count(cost), sum(cost)
    from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where cars.brand = marka_car;
end$$
DELIMITER $$
call task6_2('Рута')$$

-- 6_3
delimiter $$
-- drop procedure if exists task6_3;
```

```
create procedure task6_3(IN car VARCHAR(45), IN date1 DATE, IN date2
DATE)
begin
      select count(cost), sum(cost)
    from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where cars.`car name` = car and `repair`.date between date1 and
date2;
end$$
DELIMITER $$
call task6_3('Стрий Авто А075', '2016-07-06', '2020-08-06')$$

-- 7
delimiter $$
-- drop procedure if exists task7;
create procedure task7(IN firstName_person varchar(45),in lastName_person
varchar(45))
begin
      if (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'водій')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join brigade on employee.id = brigade.employee_id
        where jobPosition.name = 'Бригадир' and brigade.`№team` = 1
           union
        select "Водій", null;
        end;

    elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'зварник')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join brigade on employee.id = brigade.employee_id
        where jobPosition.name = 'Бригадир' and brigade.`№team` = 2
           union
        select "зварник", null;
```

```
        end;

    elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'слюсар')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join brigade on employee.id = brigade.employee_id
        where jobPosition.name = 'Бригадир' and brigade.`№team` = 2
           union
        select "слюсар", null;
        end;

     elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Бригадир')) then begin

           select firstName_person, lastName_person
           union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join area on employee.id = area.employee_id
        where jobPosition.name = 'Начальник ділянки'
           union
        select "Бригадир", null;
        end;

     elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Начальник ділянки' )) then begin

           select firstName_person, lastName_person
           union
        select "Начальник ділянки", null;
        end;

     else select "some problems";
     end if;
end$$
```

```
DELIMITER $$
call task7('Альберт','Вескер')$$
call task7('Олексій', 'Чижов')$$
call task7('Микита', 'Венарт')$$
call task7('Рамон', 'Лайтман')$$
call task7('Потап', 'Капиця')$$

-- 8_1
delimiter $$
-- drop procedure if exists task8_1;
create procedure task8_1()
begin
	select if((select count(*) from `garage facilities`)!=0, 'гаражні
господарства є', 'гаражних господарств немає');
end$$
DELIMITER $$
call task8_1()$$

-- 8_2
delimiter $$
-- drop procedure if exists task8_2;
create procedure task8_2(IN type_car varchar(45))
begin
	select if((select count(*) from `garage facilities` where type =
type_car)!=0, 'гаражне господарство такого типу є', 'гаражного
господарства такого типу немає');
end$$
DELIMITER $$
call task8_2('Таксі')$$
call task8_2('Автобус')$$

-- 9
delimiter $$
-- drop procedure if exists task9;
create procedure task9()
begin
	select cars.`car name`, `garage facilities`.type, `garage
facilities`.name as `Гаражне господарство` from cars
	join `garage facilities` on `garage facilities`.id =
cars.гаражні_господарства_idгаражні_господарства;
end$$
DELIMITER $$
call task9()$$

-- 10
delimiter $$
-- drop procedure if exists task10;
create procedure task10(IN car VARCHAR(45), IN date1 DATE, IN date2 DATE)
begin
```

```
     select `cargo transportation`.`id`,`date`,amount  from `cargo
transportation`
    join cars on cars.id = `cargo transportation`.`№car`
    where `date` between date1 and date2 AND cars.`car name` = car;
end$$
DELIMITER $$
call task10('ГАЗ-51', '2015-07-06', '2020-08-06')$$

-- 11_1
delimiter $$
-- drop procedure if exists task11_1;
create procedure task11_1(IN element_car VARCHAR(45), IN type_car
varchar(45))
begin
    select count(cost) from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    join `garage facilities` on `garage facilities`.id =
cars.гаражні_господарства_idгаражні_господарства
    where `repair`.unit = element_car AND `garage facilities`.type =
type_car;
end$$
DELIMITER $$
call task11_1('двигун', 'Автобус')$$

-- 11_2
delimiter $$
-- drop procedure if exists task11_2;
create procedure task11_2(IN element_car VARCHAR(45), IN marka_car
varchar(45))
begin
     select count(cost) from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where `repair`.unit = element_car AND cars.brand = marka_car;
end$$
DELIMITER $$
call task11_2('двигун', 'Рута')$$

-- 11_3
delimiter $$
-- drop procedure if exists task11_3;
create procedure task11_3(IN element_car varchar(45), IN car VARCHAR(45),
IN date1 DATE, IN date2 DATE)
begin
     select count(cost) from `repair`
    join cars on cars.id = `repair`.автомобілі_idАвтомобілі
    where `repair`.unit = element_car AND (`repair`.date between date1
and date2) and cars.`car name` = car;
end$$
DELIMITER $$
call task11_3('двигун', 'Богдан-3355', '2016-07-06', '2020-08-06')$$
```

```
-- 12
delimiter $$
-- drop procedure if exists task12;
create procedure task12(IN date1 DATE, IN date2 DATE)
begin
     select `car name`, `date of appearance`, `date of write-off` from
cars
    where `date of appearance` between date1 and date2 OR `date of write-
off` between date1 and date2;
end$$
DELIMITER $$
call task12('2015-07-23', '2020-08-23')$$

-- 13
delimiter $$
-- drop procedure if exists task13;
create procedure task13(IN firstName_person varchar(45), IN
lastName_person varchar(45))
begin
     if (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    join brigade on employee.id = brigade.employee_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Бригадир' and brigade.`№team` = 1 )) then begin
           select "Бригадир", null
        union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        where jobPosition.name = 'водій';
        end;

     elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    join brigade on employee.id = brigade.employee_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Бригадир' and brigade.`№team` = 2 )) then begin
           select "Бригадир", null
        union
           select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        where jobPosition.name = 'зварник'
        union
        select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
```

```
        where jobPosition.name = 'слюсар';
        end;

      elseif (exists(select firstName from person
    join employee on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    join area on employee.id = area.employee_id
    where firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'Начальник ділянки')) then begin
           select "Начальник ділянки", null
        union
        select firstName, lastName from person
        join employee on person.id = employee.person_id
        join jobPosition on jobPosition.id = employee.jobPosition_id
        join area on employee.id = area.employee_id
        where jobPosition.name = 'Начальник ділянки';
        end;

      else select "some problems";
    end if;
end$$
DELIMITER $$
call task13('Китаєв', 'Олег')$$
call task13('Олексій', 'Чижов')$$

-- 14_1
delimiter $$
-- drop procedure if exists task14_1;
create procedure task14_1(IN firstName_person varchar(45), IN
lastName_person varchar(45), IN car varchar(45))
begin
      select `repair`.date, cars.`car name`, unit from `repair`
    join cars on `repair`.автомобілі_idАвтомобілі = cars.id
    join welders on welders.id = `repair`.зварники_idЗварники
    join lockmiths on lockmiths.id = `repair`.слюсарі_idСлюсарі
    join employee on (lockmiths.employee_id = employee.id or
welders.employee_id = employee.id)
    join person on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where (firstName = firstName_person and lastName = lastName_person
and jobPosition.name = 'зварник' and cars.`car name` = car)
    or (firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'слюсар' and cars.`car name` = car);
end$$
DELIMITER $$
call task14_1('Рамон','Лайтман','Рута 22')$$
call task14_1('Франц','Зайцев','Рута 22')$$

-- 14_2
delimiter $$
```

```
-- drop procedure if exists task14_2;
create procedure task14_2(IN firstName_person varchar(45), IN
lastName_person varchar(45), IN date1 DATE, IN date2 DATE)
begin
     select `repair`.date, cars.`car name`, unit from `repair`
    join cars on `repair`.автомобілі_idАвтомобілі = cars.id
    join welders on welders.id = `repair`.зварники_idЗварники
    join lockmiths on lockmiths.id = `repair`.слюсарі_idСлюсарі
    join employee on (lockmiths.employee_id = employee.id or
welders.employee_id = employee.id)
    join person on person.id = employee.person_id
    join jobPosition on jobPosition.id = employee.jobPosition_id
    where (firstName = firstName_person and lastName = lastName_person
and jobPosition.name = 'зварник' and `repair`.date between date1 and
date2)
    or (firstName = firstName_person and lastName = lastName_person and
jobPosition.name = 'слюсар' and `repair`.date between date1 and date2);
end$$
DELIMITER $$
call task14_2('Франц','Зайцев','2015-01-01','2021-01-01')$$
call task14_2('Ясон','Масляков','2015-01-01','2021-01-01')$$

-- smth
delimiter $$
-- drop procedure if exists welders();
create procedure welders(in ide int, in idbrig int)
begin
     declare t int;
     set t = (select count(*) from drivers where employee_id=ide);
    case
    when t=0
     then  INSERT INTO `welders`(`бригади_Бригада`,`employee_id`)
VALUES(idbrig, ide);
    else
    select 'Duplbicate' as 'Error message';
    end case;
end$$
DELIMITER $$
call welders(5, 2)$$
```

“CREATE_BD"

```
-- 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='TRADITIONAL,ALLOW_INVALID_DATES';

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

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`garage facilities`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`garage facilities` ;

CREATE TABLE IF NOT EXISTS `mydb`.`garage facilities` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `type` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
```

```
DEFAULT CHARACTER SET = utf8;

CREATE UNIQUE INDEX `Тип_UNIQUE` ON `mydb`.`garage facilities` (`type` ASC);


-- -----------------------------------------------------
-- Table `mydb`.`cars`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`cars` ;

CREATE TABLE IF NOT EXISTS `mydb`.`cars` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `car name` VARCHAR(45) NOT NULL,
  `brand` VARCHAR(45) NOT NULL,
  `date of appearance` DATE NOT NULL,
  `date of write-off` DATE NULL DEFAULT NULL,
  `гаражні_господарства_idгаражні_господарства` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_автомобілі_гаражні_господарств1`
    FOREIGN KEY (`гаражні_господарства_idгаражні_господарства`)
    REFERENCES `mydb`.`garage facilities` (`id`)
    ON DELETE RESTRICT
    ON UPDATE RESTRICT)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;

CREATE INDEX `fk_автомобілі_гаражні_господарст_idx` ON `mydb`.`cars` (`гаражні_господарства_idгаражні_господарства` ASC);
```

```
-- -----------------------------------------------------
-- Table `mydb`.`jobPosition`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`jobPosition` ;

CREATE TABLE IF NOT EXISTS `mydb`.`jobPosition` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `code` VARCHAR(255) NOT NULL,
  `name` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;

CREATE UNIQUE INDEX `code_UNIQUE` ON `mydb`.`jobPosition` (`code` ASC);

CREATE UNIQUE INDEX `name_UNIQUE` ON `mydb`.`jobPosition` (`name` ASC);


-- -----------------------------------------------------
-- Table `mydb`.`person`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`person` ;

CREATE TABLE IF NOT EXISTS `mydb`.`person` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `firstName` VARCHAR(255) NOT NULL,
```

```
  `lastName` VARCHAR(255) NOT NULL,
  `email` VARCHAR(255) NOT NULL,
  `phone` CHAR(13) NOT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;

CREATE UNIQUE INDEX `email_UNIQUE` ON `mydb`.`person` (`email` ASC);

CREATE UNIQUE INDEX `phone_UNIQUE` ON `mydb`.`person` (`phone` ASC);


-- -----------------------------------------------------
-- Table `mydb`.`employee`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`employee` ;

CREATE TABLE IF NOT EXISTS `mydb`.`employee` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `jobPosition_id` BIGINT(20) NOT NULL,
  `person_id` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_employee_jobPosition1`
    FOREIGN KEY (`jobPosition_id`)
    REFERENCES `mydb`.`jobPosition` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_employee_person1`
    FOREIGN KEY (`person_id`)
```

```
  REFERENCES `mydb`.`person` (`id`)
  ON DELETE CASCADE
  ON UPDATE CASCADE)
ENGINE = InnoDB;


CREATE INDEX `fk_employee_jobPosition1_idx` ON `mydb`.`employee`
(`jobPosition_id` ASC);


CREATE INDEX `fk_employee_person1_idx` ON `mydb`.`employee` (`person_id`
ASC);


CREATE UNIQUE INDEX `person_id_UNIQUE` ON `mydb`.`employee`
(`person_id` ASC);



-- -----------------------------------------------------
-- Table `mydb`.`area`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`area` ;


CREATE TABLE IF NOT EXISTS `mydb`.`area` (
 `№district` BIGINT(20) NOT NULL AUTO_INCREMENT,
 `employee_id` BIGINT(20) NOT NULL,
 PRIMARY KEY (`№district`),
 CONSTRAINT `fk_ділянки_employee1`
  FOREIGN KEY (`employee_id`)
  REFERENCES `mydb`.`employee` (`id`)
  ON DELETE CASCADE
  ON UPDATE CASCADE)
```

```
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


CREATE INDEX `fk_ділянки_employee1_idx` ON `mydb`.`area` (`employee_id`
ASC);


CREATE UNIQUE INDEX `employee_id_UNIQUE` ON `mydb`.`area`
(`employee_id` ASC);



-- -----------------------------------------------------
-- Table `mydb`.`brigade`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`brigade` ;


CREATE TABLE IF NOT EXISTS `mydb`.`brigade` (
  `№team` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `employee_id` BIGINT(20) NOT NULL,
  `ділянки_№district` BIGINT(20) NOT NULL,
  PRIMARY KEY (`№team`),
  CONSTRAINT `fk_бригади_employee1`
    FOREIGN KEY (`employee_id`)
    REFERENCES `mydb`.`employee` (`id`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_бригади_ділянки1`
    FOREIGN KEY (`ділянки_№district`)
    REFERENCES `mydb`.`area` (`№district`)
    ON DELETE RESTRICT
```

```
  ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


CREATE INDEX `fk_бригади_employee1_idx` ON `mydb`.`brigade`
(`employee_id` ASC);


CREATE UNIQUE INDEX `employee_id_UNIQUE` ON `mydb`.`brigade`
(`employee_id` ASC);


CREATE INDEX `fk_бригади_ділянки1_idx` ON `mydb`.`brigade`
(`ділянки_№district` ASC);



-- -----------------------------------------------------
-- Table `mydb`.`cargo transportation`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`cargo transportation` ;


CREATE TABLE IF NOT EXISTS `mydb`.`cargo transportation` (
 `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
 `date` DATE NOT NULL,
 `№car` BIGINT(20) NOT NULL,
 `amount` INT(3) UNSIGNED NOT NULL,
 PRIMARY KEY (`id`),
 CONSTRAINT `№car`
  FOREIGN KEY (`№car`)
  REFERENCES `mydb`.`cars` (`id`)
  ON DELETE CASCADE
```

```
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;

CREATE INDEX `zsrfgxvxfh_idx` ON `mydb`.`cargo transportation` (`№car` ASC);


-- -----------------------------------------------------
-- Table `mydb`.`drivers`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`drivers` ;

CREATE TABLE IF NOT EXISTS `mydb`.`drivers` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `автомобілі_idАвтомобілі` BIGINT(20) NOT NULL,
  `бригади_Бригада` BIGINT(20) NOT NULL,
  `employee_id` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_водії_автомобілі1`
    FOREIGN KEY (`автомобілі_idАвтомобілі`)
    REFERENCES `mydb`.`cars` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `fk_водії_бригади1`
    FOREIGN KEY (`бригади_Бригада`)
    REFERENCES `mydb`.`brigade` (`№team`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
```

```
  CONSTRAINT `fk_водії_employee1`
    FOREIGN KEY (`employee_id`)
    REFERENCES `mydb`.`employee` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


CREATE INDEX `fk_водії_автомобілі1_idx` ON `mydb`.`drivers`
(`автомобілі_idАвтомобілі` ASC);


CREATE INDEX `fk_водії_бригади1_idx` ON `mydb`.`drivers`
(`бригади_Бригада` ASC);


CREATE INDEX `fk_водії_employee1_idx` ON `mydb`.`drivers` (`employee_id`
ASC);


CREATE UNIQUE INDEX `автомобілі_idАвтомобілі_UNIQUE` ON
`mydb`.`drivers` (`автомобілі_idАвтомобілі` ASC);



-- -----------------------------------------------------
-- Table `mydb`.`welders`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`welders` ;


CREATE TABLE IF NOT EXISTS `mydb`.`welders` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `бригади_Бригада` BIGINT(20) NOT NULL,
```

```
  `employee_id` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_зварники_бригади1`
    FOREIGN KEY (`бригади_Бригада`)
    REFERENCES `mydb`.`brigade` (`№team`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_зварники_employee1`
    FOREIGN KEY (`employee_id`)
    REFERENCES `mydb`.`employee` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


CREATE INDEX `fk_зварники_бригади1_idx` ON `mydb`.`welders`
(`бригади_Бригада` ASC);


CREATE INDEX `fk_зварники_employee1_idx` ON `mydb`.`welders`
(`employee_id` ASC);


CREATE UNIQUE INDEX `employee_id_UNIQUE` ON `mydb`.`welders`
(`employee_id` ASC);



-- -----------------------------------------------------
-- Table `mydb`.`routs`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`routs` ;
```

```
CREATE TABLE IF NOT EXISTS `mydb`.`routs` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `автомобілі_idАвтомобілі` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_маршрути_автомобілі1`
    FOREIGN KEY (`автомобілі_idАвтомобілі`)
    REFERENCES `mydb`.`cars` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;

CREATE INDEX `fk_маршрути_автомобілі1_idx` ON `mydb`.`routs`
(`автомобілі_idАвтомобілі` ASC);



-- -----------------------------------------------------
-- Table `mydb`.`mileage`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`mileage` ;

CREATE TABLE IF NOT EXISTS `mydb`.`mileage` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `date` DATE NOT NULL,
  `kilometrage` INT(20) UNSIGNED NOT NULL,
  `автомобілі_idАвтомобілі` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
```

```
  CONSTRAINT `fk_пробіг_автомобілі1`
    FOREIGN KEY (`автомобілі_idАвтомобілі`)
    REFERENCES `mydb`.`cars` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


CREATE INDEX `fk_пробіг_автомобілі1_idx` ON `mydb`.`mileage`
(`автомобілі_idАвтомобілі` ASC);



-- -----------------------------------------------------
-- Table `mydb`.`lockmiths`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`lockmiths` ;


CREATE TABLE IF NOT EXISTS `mydb`.`lockmiths` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `бригади_Бригада` BIGINT(20) NOT NULL,
  `employee_id` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_слюсарі_бригади1`
    FOREIGN KEY (`бригади_Бригада`)
    REFERENCES `mydb`.`brigade` (`№team`)
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_слюсарі_employee1`
```

```
    FOREIGN KEY (`employee_id`)
    REFERENCES `mydb`.`employee` (`id`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


CREATE INDEX `fk_слюсарі_бригади1_idx` ON `mydb`.`lockmiths`
(`бригади_Бригада` ASC);


CREATE INDEX `fk_слюсарі_employee1_idx` ON `mydb`.`lockmiths`
(`employee_id` ASC);


CREATE UNIQUE INDEX `employee_id_UNIQUE` ON `mydb`.`lockmiths`
(`employee_id` ASC);




-- -----------------------------------------------------
-- Table `mydb`.`repair`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`repair` ;


CREATE TABLE IF NOT EXISTS `mydb`.`repair` (
  `id` BIGINT(20) NOT NULL AUTO_INCREMENT,
  `date` DATE NOT NULL,
  `cost` INT(5) UNSIGNED NOT NULL,
  `unit` VARCHAR(255) NOT NULL,
  `автомобілі_idАвтомобілі` BIGINT(20) NOT NULL,
  `зварники_idЗварники` BIGINT(20) NOT NULL,
```

```
  `слюсарі_idСлюсарі` BIGINT(20) NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `fk_ремонт_автомобілі1`
    FOREIGN KEY (`автомобілі_idАвтомобілі`)
    REFERENCES `mydb`.`cars` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ремонт_зварники1`
    FOREIGN KEY (`зварники_idЗварники`)
    REFERENCES `mydb`.`welders` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ремонт_слюсарі1`
    FOREIGN KEY (`слюсарі_idСлюсарі`)
    REFERENCES `mydb`.`lockmiths` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;


CREATE INDEX `fk_ремонт_автомобілі1_idx` ON `mydb`.`repair`
(`автомобілі_idАвтомобілі` ASC);


CREATE INDEX `fk_ремонт_зварники1_idx` ON `mydb`.`repair`
(`зварники_idЗварники` ASC);


CREATE INDEX `fk_ремонт_слюсарі1_idx` ON `mydb`.`repair`
(`слюсарі_idСлюсарі` ASC);
```

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

“INSERT_BD"

```
INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Олексій', 'Чижов', 'aderf@net', '+380111111111');
INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Микита', 'Венарт', 'gjdo@net', '+380222222222');
INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Альберт', 'Розмареллі', 'NNN@net', '+380994765937');
INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Альберт', 'Вескер', 'alb@net', '+380994768473');
INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Люцій', 'Блок', 'goj@net', '+380969475937');
INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Самсон', 'Вебер', 'out@net', '+380994000007');
```

```
INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Тайлер', 'Дерден', 'into@net', '+380111115937');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Закир', 'Ганди', 'up@net', '+380992222937');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Нільс', 'Нільсон', 'down@net', '+380994733337');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Осип', 'Дефо', 'hot@net', '+380944445937');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Франц', 'Зайцев', 'intro@net', '+380994765555');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Потап', 'Капиця', 'uotro@net', '+380966665937');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Эдмунд', 'Клінтон', 'AAA@net', '+380994767777');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Рамон', 'Лайтман', 'BBB@net', '+380994765888');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Клод', 'Шенон', 'CCC@net', '+380999995937');

INSERT INTO person (`firstName`, `lastName`, `email`, `phone`) VALUES ('Ясон', 'Масляков', 'DDD@net', '+380994760037');


INSERT INTO `jobPosition`(`code`,`name`)VALUES('captain','Начальник ділянки');

INSERT INTO `jobPosition`(`code`,`name`)VALUES('brigadier','Бригадир');

INSERT INTO `jobPosition`(`code`,`name`)VALUES('driver','водій');

INSERT INTO `jobPosition`(`code`,`name`)VALUES('welders','зварник');

INSERT INTO `jobPosition`(`code`,`name`)VALUES('lockmiths','слюсар');


INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(1,1);

INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(2,2);

INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(2,3);
```

```
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(3,4);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(3,5);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(3,6);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(3,7);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(3,8);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(3,9);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(3,10);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(4,11);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(4,12);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(4,13);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(5,14);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(5,15);
INSERT INTO `employee`(`jobPosition_id`,`person_id`) VALUES(5,16);

INSERT INTO `garage facilities`(`name`,`type`) VALUES('Г-1','автобус');
INSERT INTO `garage facilities`(`name`,`type`) VALUES('Г-2','вантажівка');
INSERT INTO `garage facilities`(`name`,`type`) VALUES('Г-3','таксі');

INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-
off`,`гаражні_господарства_idгаражні_господарства`)
VALUES('ЛАЗ-697', 'ЛАЗ', '2015-07-23','2018-08-23', 1);
INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-
off`,`гаражні_господарства_idгаражні_господарства`)
VALUES('Рута 22', 'Рута', '2015-07-23','2018-08-23', 1);
INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-
off`,`гаражні_господарства_idгаражні_господарства`)
VALUES('Рута 23', 'Рута', '2015-08-23', NULL, 1);
INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-
off`,`гаражні_господарства_idгаражні_господарства`)
```

```
VALUES('Стрий Авто А075', 'Стрий Авто','2016-07-23', NULL, 1);

INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-off`,`гаражні_господарства_idгаражні_господарства`)

VALUES('Стрий Авто А0756', 'Стрий Авто', '2016-07-23', NULL, 1);

INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-off`,`гаражні_господарства_idгаражні_господарства`)

VALUES('Богдан-3355', 'Богдан', '2015-07-23', '2018-07-23', 2);

INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-off`,`гаражні_господарства_idгаражні_господарства`)

VALUES('ГАЗ-51', 'ГАЗ', '2015-07-23', NULL, 2);

INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-off`,`гаражні_господарства_idгаражні_господарства`)

VALUES('ГАЗ-68', 'ГАЗ', '2015-07-23', NULL, 2);

INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-off`,`гаражні_господарства_idгаражні_господарства`)

VALUES('ЗІЛ-4331', 'ЗІЛ', '2016-07-23', NULL, 2);

INSERT INTO `cars`(`car name`,`brand`,`date of appearance`,`date of write-off`,`гаражні_господарства_idгаражні_господарства`)

VALUES('Богдан А091', 'Богдан', '2016-07-23', NULL, 2);


INSERT INTO `area`(`employee_id`) VALUES(1);


INSERT INTO `routs`(`автомобілі_idАвтомобілі`) VALUES(1);

INSERT INTO `routs`(`автомобілі_idАвтомобілі`) VALUES(2);

INSERT INTO `routs`(`автомобілі_idАвтомобілі`) VALUES(3);

INSERT INTO `routs`(`автомобілі_idАвтомобілі`) VALUES(4);


INSERT INTO mileage(`date`,`kilometrage`,`автомобілі_idАвтомобілі`)
VALUES('2017-01-01',100, 3);
```

```
INSERT INTO mileage(`date`,`kilometrage`,`автомобілі_idАвтомобілі`)
VALUES('2017-08-23',120, 4);

INSERT INTO mileage(`date`,`kilometrage`,`автомобілі_idАвтомобілі`)
VALUES('2018-08-23',150, 5);

INSERT INTO mileage(`date`,`kilometrage`,`автомобілі_idАвтомобілі`)
VALUES('2016-08-23',600, 7);

INSERT INTO mileage(`date`,`kilometrage`,`автомобілі_idАвтомобілі`)
VALUES('2016-12-24',650, 8);


INSERT INTO `cargo transportation`(`date`,`№car`,`amount`) VALUES('2016-08-
23',7, 20);

INSERT INTO `cargo transportation`(`date`,`№car`,`amount`) VALUES('2016-12-
24',8, 50);


INSERT INTO `brigade`(`employee_id`,`ділянки_№district`) VALUES(2, 1);

INSERT INTO `brigade`(`employee_id`,`ділянки_№district`) VALUES(3, 1);


INSERT INTO `welders`(`бригади_Бригада`,`employee_id`) VALUES(2, 11);

INSERT INTO `welders`(`бригади_Бригада`,`employee_id`) VALUES(2, 12);

INSERT INTO `welders`(`бригади_Бригада`,`employee_id`) VALUES(2, 13);


INSERT INTO `lockmiths`(`бригади_Бригада`,`employee_id`) VALUES(2, 14);

INSERT INTO `lockmiths`(`бригади_Бригада`,`employee_id`) VALUES(2, 15);

INSERT INTO `lockmiths`(`бригади_Бригада`,`employee_id`) VALUES(2, 16);


INSERT INTO
`drivers`(`автомобілі_idАвтомобілі`,`бригади_Бригада`,`employee_id`)
VALUES(3, 1, 4);

INSERT INTO
`drivers`(`автомобілі_idАвтомобілі`,`бригади_Бригада`,`employee_id`)
VALUES(4, 1, 5);
```

```
INSERT INTO
`drivers`(`автомобілі_idАвтомобілі`,`бригади_Бригада`,`employee_id`)
VALUES(5,  1, 6);

INSERT INTO
`drivers`(`автомобілі_idАвтомобілі`,`бригади_Бригада`,`employee_id`)
VALUES(7,  1, 7);

INSERT INTO
`drivers`(`автомобілі_idАвтомобілі`,`бригади_Бригада`,`employee_id`)
VALUES(8,  1, 8);

INSERT INTO
`drivers`(`автомобілі_idАвтомобілі`,`бригади_Бригада`,`employee_id`)
VALUES(9,  1, 9);

INSERT INTO
`drivers`(`автомобілі_idАвтомобілі`,`бригади_Бригада`,`employee_id`)
VALUES(10,  1, 10);


INSERT INTO
`repair`(`date`,`cost`,`unit`,`автомобілі_idАвтомобілі`,`зварники_idЗварники`,`сл
юсарі_idСлюсарі`)

VALUES('2018-08-20',10000,  'двигун',2,1,1);

INSERT INTO
`repair`(`date`,`cost`,`unit`,`автомобілі_idАвтомобілі`,`зварники_idЗварники`,`сл
юсарі_idСлюсарі`)

VALUES('2017-07-23',8000,  'КП',4,2,2);

INSERT INTO
`repair`(`date`,`cost`,`unit`,`автомобілі_idАвтомобілі`,`зварники_idЗварники`,`сл
юсарі_idСлюсарі`)

VALUES('2018-02-23',10000,  'двигун',6,3,3);

INSERT INTO
`repair`(`date`,`cost`,`unit`,`автомобілі_idАвтомобілі`,`зварники_idЗварники`,`сл
юсарі_idСлюсарі`)

VALUES('2020-08-20',12000,  'шасі',10,1,3);
```