МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет: Інформаційних технологій

Кафедра: Програмного забезпечення комп'ютерних систем
(повна назва кафедри)

Методичні вказівки
щодо виконання
курсової роботи з дисципліни:

ОРГАНІЗАЦІЯ БАЗ ДАНИХ ТА ЗНАНЬ

Спеціальність 121 «Інженерія програмного забезпечення»

Дніпро
2023

Методичні вказівки що до виконання курсової роботи з дисципліни “Організація баз даних та знань” спеціальності 121 Інженерія програмного забезпечення/ Укл.: Кабак Л.В., – Дніпро Національний технічний університет «Дніпровська політехніка», 2023. – 43 с.

Укладач: Кабак Л.В. кандидат технічних наук, доцент, доцент кафедри програмного забезпечення комп’ютерних систем.

Зміст

ВСТУП

Мета курсової роботи – освоєння технології проектування і створення автоматизованих інформаційних систем (АІС), у середовищі швидкої розробки додатків (Rapid Application Development - RAD) MS Visual Studio C#, з використанням мови структурних запитів SQL.

Задачі проекту – створення концептуальної моделі конкретної предметної області, логічної моделі даних і, нарешті, фізичної моделі даних, на підставі якої створюється розподілена база даних, для роботи з якою необхідно створити додаток засобами RAD-системи MS Visual Studio C#. База даних розробляється за допомогою мови структурних запитів SQL. Тригери, процедури та функції бази даних розробляються за допомогою програмної мови PL/SQL.

Проектування бази даних – ґрунтується на концептуальній моделі, що відображає взаємозв'язки між реальними об'єктами предметної області.

Концептуальна модель відображає реальні об'єкти і їхні взаємозв'язки без указівки способів їхнього фізичного збереження і є чисто інформаційною (описовою) моделлю предметної області.

Предметною областю називається частина реальної системи, що представляє інтерес для конкретного дослідження.

Необхідно розрізняти повну предметну область (велике виробниче підприємство, склад, універмаг і т.д.) і організаційну одиницю цієї предметної області. Організаційна одиниця у свою чергу може представляти свою предметну область (наприклад, цех по виробництву кузовів автомобільного заводу чи відділ обробки даних підприємство з виробництва ЕОМ). У даному випадку цехи і відділи самі можуть відповідати визначеним предметним областям. Інформація про предметну область може включати зведення про персонал, заробітну плату, товари, накладних, рахунках, звітах по збуті, лабораторних тестах, фінансових угодах, історіях хвороб і т.д.

Процес створення концептуальної моделі починається з визначення концептуальних вимог замовника. На цьому етапі всі зусилля розроблювача повинні бути спрямовані в основному на структуризацію даних і виявлення взаємозв'язків між ними без розгляду особливостей реалізації і питань ефективності обробки. Проектування концептуальної моделі засновано на аналізі задач по обробці даних, розв'язуваних на конкретному підприємстві, і включає опису реальних об'єктів, розглянутої предметної області, що виявляються в результаті аналізу даних і їхніх взаємозв'язків.

Концептуальна модель є основою для проектування моделі даних, сумісну з тієї, котру підтримується обраної СУБД. Це може бути ієрархічна, мережна або реляційна модель. Можливо, що відображені в концептуальній моделі взаємозв'язку між об'єктами виявляться згодом нереалізованими засобами обраної СУБД. Це зажадає зміни концептуальної моделі. Версія концептуальної моделі, що може бути забезпечена конкретною СУБД, називається логічною моделлю.

Логічна модель відображає логічні зв'язки між елементами даних поза залежністю від їхнього змісту в середовищі збереження.

Логічна модель даних може бути реляційною, ієрархічною чи мережною і ніяк не зв'язана з типом фізичної пам'яті, у якій будуть зберігатися дані, і з методами доступу до цих даних.

Логічна модель потім відображається у фізичну з урахуванням типів даних, підтримуваних використовуваними для розробки СУБД (MySQL, Oracle і т.д.) або системами програмування (C# і т.д.). Після цього етапу можна переходити до проектування і створення автоматизованої інформаційної системи.

Вимоги до оформлення – пояснювальна записка (до 40 аркушів формату А4) повинна містити: титульний лист (Додаток А), завдання на курсову роботу і розділи відповідно до даної Програми.

Пояснювальна записка оформляється відповідно до загальних вимог, пропонованими до оформлення текстової документації стандартами ЕСКД (“Загальні вимоги до текстових документів” і “Текстові документи”), а також відповідно до вимог, пропонованими до оформлення програмної документації стандартами ЕСПД (ДСТУ ISO/IEC/IEEE 16326:2015 “Пояснювальна записка. Вимоги до змісту й оформлення” і “Загальні вимоги до програмних документів”). Весь необхідний графічний матеріал включається у виді рисунків або додатків (за узгодженням з керівником курсового проекту), виконаних на папері формату А4. Графічний матеріал виконуються засобами обчислювальної техніки.

Порядок консультацій – відповідно до розкладу Кафедри інформаційних систем та технологій аудиторія №25-9. На консультації студент готує результати роботи для обговорення з викладачем.

Захист – критеріями оцінки виконання курсової роботи є: відповідність створеної концептуальної моделі заданої предметної області, правильне відображення концептуальної моделі на реляційну модель даних, відповідність фізичної моделі даних обраному формату даних СУБД, реалізація всіх необхідних функцій АІС, ступінь самостійності виконання, володіння матеріалом курсової роботи на рівні формулювання й аргументування кожного з етапів роботи.

1. СТРУКТУРА КУРСОВОЇ РОБОТИ І ВИМОГИ ДО ЗМІСТУ РОЗДІЛІВ

1. Титульний лист та Лист завдання

Оформити згідно зразка (Додаток А).

2. Зміст

Відбити назви розділів і підрозділів з посиланнями на сторінки.

3. Завдання на курсову роботу

Додати, видане керівником проектування завдання на курсову роботу.

3. ПОСТАНОВКА ЗАДАЧІ І СТВОРЕННЯ КОНЦЕПТУАЛЬНОЇ МОДЕЛІ

3.1. Постановка задачі

Привести постановку задачі, спрощену схема бізнес-процесів, аналіз бізнес-процесів і організаційну структуру предметної області. При аналізі бізнесу-процесу фірми корисно відповісти на шістьох питань: що, як, де, хто, коли і чому.

3.1.1. Перше питання: "Що лежить в основі бізнесу даної фірми (або підприємства)?"

3.1.2. Друге питання: "Як це робиться?"

3.1. 3. Третє питання: "Де відбуваються дані процеси?"

3.1. 4. Четверте питання: "Хто виконує ці процеси?"

3.1. 5. П'яте питання: "Коли виконується та чи інша дія?"

3.1. 6. Шосте питання: "Чому ці дії виконуються?"

3.1.7. Загальні зауваження (міркування)

Щоб одержати адекватну концептуальну модель, необхідно зробити постановку задачі, тобто докладно описати конкретну предметну область і циркулюючі в ній інформаційні потоки, щоб зрозуміти що підлягає автоматизації.

Розберемо на конкретному прикладі, як повинні виглядати постановка задачі і створення концептуальної моделі. Як приклад виберемо діяльність фірми-дилера "Фронтон" [4] (надалі просто фірми), що займається продажем легкових автомобілів на замовлення. Процес продажу виникає в такий спосіб.

На підставі досліджень ринку потенційних покупців і пропозицій автоіндустрії відділ маркетингу розробляє каталог пропонованих до продажу автомобілів. Каталог поширюється на ринку потенційних покупців. З клієнтом, що ознайомився з каталогом і вирішив придбати автомобіль, працює служба оформлення замовлень. Фахівці, що входять у цю службу, приймають замовлення, уточнюють комплектацію обраного

автомобіля, виписують рахунок на покупку й одночасно з цим відправляють запит про придбання даного автомобіля на заводи-виготовлювачі або дистриб'юторам, стежать за оплатою рахунків, після чого по відповідному рахунку фірма підтверджує запит про придбання автомобіля і зобов'язується протягом чотирьохтижневого терміну надати покупку відповідному покупцю і, нарешті, вручають клієнту автомобіль, що надійшов. Служба внутрішньої підтримки забезпечує розподіл роботи з виконавців і вирішує виникаючі проблеми. Цей опис дозволяє скласти спрощену схему бізнесів-процесів фірми (рис. 1).

При аналізі бізнесу-процесу фірми корисно відповісти на шістьох питань: що, як, де, хто, коли і чому.

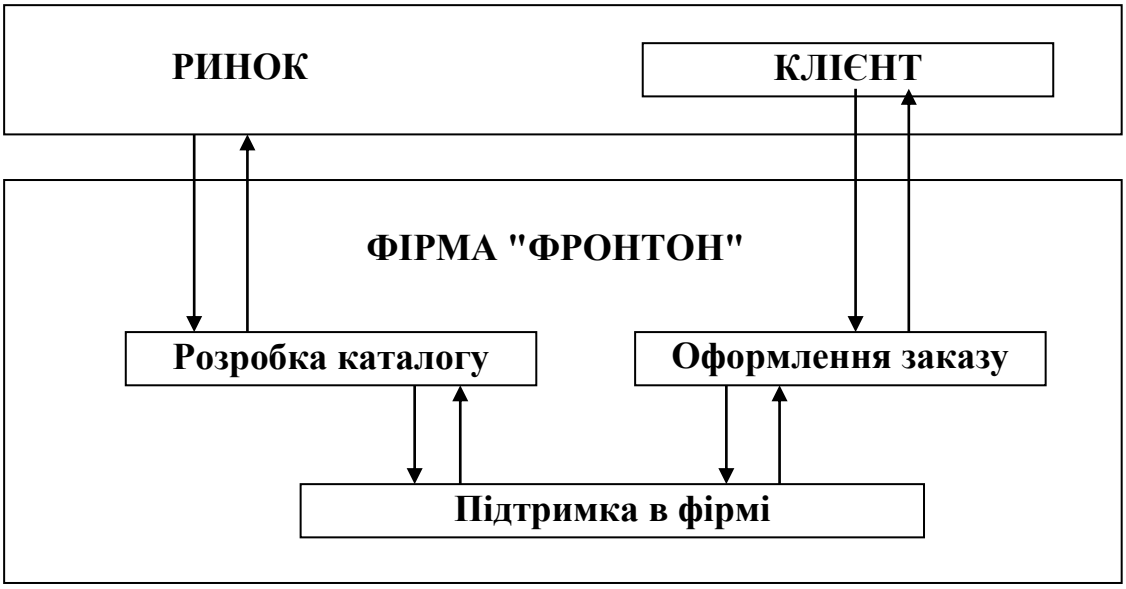

Рис. 1. Спрощена схема бізнесів-процесів фірми

3.7.1. Перше питання: "Що лежить в основі бізнесу даної фірми?" Відповідь, як правило, виявляє найбільш важливі для даного чи бізнесу виробничого процесу компоненти. У нашому випадку це будуть:

- співробітники;
- клієнти (покупці);
- постачальники;
- каталог;

• автомобілі;

• замовлення.

3.7.2. Друге питання: "Як це робиться?"

Відповідь дозволяє одержати список основних бізнесів-процесів, що відбуваються у фірмі. Для нашого приклада в такий список можна включити наступні пункти:

• складання каталогу;

• розсилання каталогу;

• аналіз ринку;

• продажу;

• оформлення рахунків і накладних;

• керування роботою персоналу;

• організація реклами;

• рішення бухгалтерських задач.

3.7.3. Третє питання: "Де відбуваються дані процеси?"

Відповідь торкається проблеми телекомунікацій і організації спільної роботи персоналу. У розглянутому прикладі допустимо, що всі операції виконуються в межах одного будинку, а організація спільного використання даних заснована на можливостях локальної мережі і сервера БД.

3.7.4. Четверте питання: "Хто виконує ці процеси?"

Відповідь дає організаційна структура фірми. Представимо графічно спрощено організаційну структуру фірми (Рис. 2).

3.1.1.5. П'яте питання: "Коли виконується та чи інша дія?"

Відповідь проясняє періодичність здійснюваних бізнесів-процесів і дозволяє правильно розставити акценти в майбутній прикладній програмі. У нашому випадку приймемо таку тимчасову послідовність виконуваних процесів:

• відновлення каталогу – раз у півроку і внесення виправлень в екстрених випадках;

• підведення підсумків продажів – щомісяця;

• річний звіт – щорічно до 20 лютого.

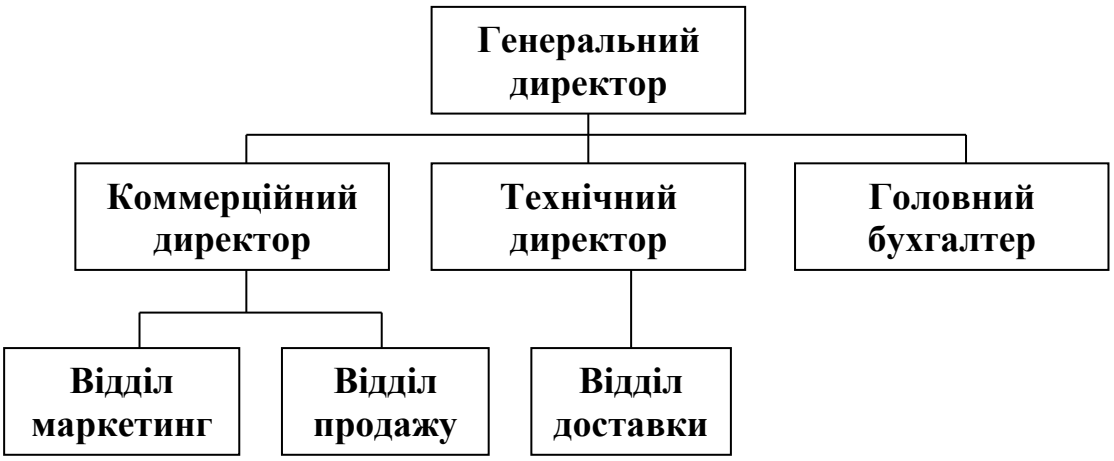

Рис.2. Організаційна структура фірми

3.1.1.6. Шосте питання: "Чому ці дії виконуються?"

Відповідь дозволяє визначити мотивацію виробничої діяльності фірми. Бізнесу-задачі фірми визначимо так:

• досягнення найкращого співвідношення "витрата-зручність" для клієнтів;

• забезпечення умов для успішної діяльності персоналу;

• одержання прийнятного прибутку;

• підвищення доходів при автоматизації обробки даних.

3.2. Створення концептуальної моделі

Привести ER-діаграму взаємозв'язку між бізнесами-компонентами і бізнесами-процесами, формалізований опис поставленої задачі.

3.2.1. Загальні зауваження (міркування)

Відповіді на шість питань розділа 4.1.1. дозволяють підійти до головного в постановці задачі – побудові концептуальної моделі у виді взаємозв'язків між бізнесами-компонентами і бізнесами-процесами, як це

показано на рис.3 .

У практиці проектування інформаційних систем такі схеми одержали назва ER-діаграм (Entity-relationship diagram (ERD) – діаграма "Сутність-зв'язок")[7,10]. ER-діаграми добре вписуються в методологію структурного аналізу і проектування інформаційних систем, забезпечуючи строгий і наочний опис проектованої системи, що починається з її загального огляду і, потім, уточнюється, даючи можливість одержати різний ступінь деталізації об'єкта з різним числом рівнів.

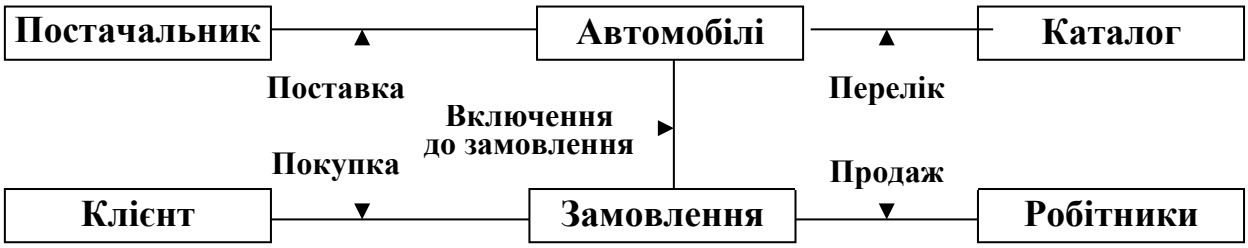

Рис.3 Концептуальна схема

Максимально формалізований опис постановки задачі в нашому випадку буде виглядати в такий спосіб.

1. Найменування задачі:

Автоматизація керування роботою фірми з продажу легкових автомобілів "AUTO_STORE".

2. Мета роботи фірми:

Продаж легкових автомобілів на замовлення по каталозі.

3. Бізнес-процеси фірми:

- Укладання договорів на постачання автомобілів.
- Ведення каталогу автомобілів, пропонованих на продаж.
- Прийом замовлень у клієнтів (покупців) на постачання автомобілів.
- Робота з клієнтами (маркетинг): реклама нових автомобілів, підготовка зведень про автомобілі, що здобуваються, аналіз продажів, ведення довідника клієнтів.
- Відправлення замовлень постачальнику автомобілів.
- Ведення розрахунків за продані автомобілі (виписка накладних).

• Облік валютного курсу.

4. Бізнес-правила:

• Зведення про клієнтів зберігаються 10 років.

• Оплата очікується 3 тижня, якщо її не відбувається, замовлення знищується.

• Підтвердження запиту про придбання автомобіля відправляється фірмі-постачальнику після приходу грошей.

• При відмовленні від поставленого автомобіля з покупця утримується 9% суми оплати по рахунку, дана величина повинна регулюватися.

• Термін постачання 4 тижні після приходу грошей.

• Прострочення доставки автомобіля клієнту оплачується самою фірмою з розрахунку 0,1% у день, дана величина повинна регулюватися.

• Якщо автомобіль не поставлений протягом 2 місяців, повертається вся сума оплати і пеня.

5. Вимоги до програми:

Програма повинна працювати під керуванням операційної системи Windows 10 та вище

Система, що буде розроблюватись у курсовій роботі, повинна проектуватись як розподілена база тобто буде складатись з центрального серверу данних та та вузлів та миати можливість проведення реплікацій, як за допомогою СУБД ORACLE так і за допомогою файлів обміну даними.

6. Перелік інформації, що вводиться:

• найменування моделі автомобіля, що продається;

• робочий обсяг двигуна, див^3;

• кількість циліндрів у двигуні;

• номінальна потужність двигуна, л. с.;

• максимальний момент, що крутить, **Н** * м;

• максимальна швидкість автомобіля, км/год;

• час розвантажування автомобіля до 100 км/год, з;

- кількість дверей;
- кількість місць;
- довжина, мм;
- ширина, мм;
- висота, мм;
- витрата палива при швидкості 90 км/год, л/100 км;
- витрата палива при швидкості 120 км/год, л/100 км;
- витрата палива при міському циклі, л/100 км;
- найменування виробника автомобіля;
- найменування країни, у якій виробляється автомобіль;
- найменування використовувуємого автомобілем палива;
- найменування шин;
- найменування типу кузова;
- дата випуску автомобіля;
- вартість автомобіля;
- найменування клієнта;
- адреса клієнта;
- телефон клієнта;
- факс клієнта;
- прізвище, ім'я та по батькові клієнта;
- ознака юридичної особи клієнта;
- примітка для запису заміток по роботі з клієнтом;
- номер рахунка;
- дата продажу;
- сума продажу;
- позначка про оплату;
- прізвище, ім'я і по батькові продавця.

7. Перелік друкованих звітів:

- номенклатура пропонованих до продажу автомобілів;
- список клієнтів;

- аналіз продажів;
- список замовлень;
- рахунок на покупку.

8. Тип логічної моделі:

- реляційна.

4. СТВОРЕННЯ ЛОГІЧНОЇ МОДЕЛІ

4.1. Визначення об'єктів (сутностей)

Привести й обґрунтувати список об'єктів (сутностей), необхідних для створення логічної моделі.

4.1.1. Загальні зауваження (міркування)

При створенні логічної моделі визначальним фактором є вибір типу моделі даних (ієрархічний, мережний, реляційний), що, у свою чергу, залежить від можливостей наявних у розпорядженні розроблювача СУБД або програмних систем. В даний час більшість настільних і промислових СУБД, а також програмних систем підтримують реляційну модель даних. У рамках реляційної моделі аналізуємо концептуальну модель, щоб з'ясувати, які об'єкти предметної області нам знадобляться.

- Об'єктом називається елемент інформаційної системи, інформацію про яке ми зберігаємо в базі даних. У реляційній теорії баз даних, об'єкт називається сутністю.

Об'єкт може бути реальним (наприклад, людина, який-небудь предмет чи населений пункт) і абстрактним (наприклад, подія, рахунок покупця чи досліджуваний студентами курс). Так, в області продажу автомобілів прикладами об'єктів можуть служити МОДЕЛЬ, КЛІЄНТ, РАХУНОК. Кожен об'єкт має визначений набір властивостей, але в базі даних інформаційної системи запам'ятовуються тільки ті властивості, що використовуються в бізнес процесах і необхідні для видачі звітів.

Виходячи з поставленої задачі, виділимо наступні сутності:

1) МОДЕЛЬ;

2) АВТОМОБІЛЬ;

3) КЛІЄНТ;

4) ПРОДАВЕЦЬ;

5) ЗАМОВЛЕННЯ;

6) ПРОДАЖ;

7) РАХУНОК.

Об'єкти є поняттями реального світу й у реляційній моделі даних [9, 22] моделюються відносинами (таблицями), у стовпцях яких і розміщаються властивості об'єктів, що називаються атрибутами.

4.2. Визначення взаємозв'язків між сутностями

Привести схему взаємозв'язків між сутностями й аргументувати її.

4.2.1. Загальні зауваження (міркування)

При визначенні взаємозв'язків між сутностями, включеними в модель, необхідно мати у виді, що взаємозв'язок виражає чи відображення зв'язок між безлічами даних сутностей. Взаємозв'язку між сутностями можуть бути трьох типів: "один до одного", "один до багатьох" і "багато хто до богатьох".

Звернемося до розглянутої нами задачі по автоматизації керування роботою фірми з продажу легкових автомобілів. Якщо клієнт робить замовлення на покупку автомобіля вперше, здійснюється первинна реєстрація його даних в об'єкті КЛІЄНТ і зведень про зроблене замовлення в об'єкті ЗАМОВЛЕННЯ. Інформація про кожного клієнта включає: найменування клієнта, адреса, телефон, факс, прізвище, ім'я, по батькові, ознаку юридичної особи і примітка. Таким чином, атрибутами об'єкта КЛІЄНТ є "Унікальний ключ клієнта", "Найменування клієнта", "Адреса клієнта" і т.д. Атрибутами об'єкта ЗАМОВЛЕННЯ є "Номер замовлення", "Ключ клієнта" і "Ключ моделі". Якщо ж клієнт робить замовлення повторно, здійснюється реєстрація тільки даного замовлення, оскільки даний клієнт уже має унікальний ідентифікаційний номер (унікальний ключ клієнта).

Наступний об'єкт, що представляє для нас інтерес - МОДЕЛЬ. Цей об'єкт має атрибути "Унікальний ключ моделі", "Найменування моделі" і

т.д. І, нарешті, четвертий розглянутий об'єкт – ПРОДАВЕЦЬ. Його атрибутами є "Унікальний ключ продавця", "Ім'я продавця", "Прізвище" і "По батькові".

4.2.1.1. Взаємозв'язок "один до одного" (між двома типами об'єктів)

Зв'язок “один до одного” являє собою найпростіший вид зв'язку даних, коли первинний ключ таблиці є в той же час зовнішнім ключем, що посилається на первинний ключ іншої таблиці. Такий зв'язок буває зручно встановлювати тоді, коли невигідно тримати різні за розміром (чи за іншими критеріями) дані в одній таблиці. Наприклад, можна виділити дані з докладним описом виробу в окрему таблицю з установленням зв'язку “один до одного” для того щоб не займати оперативну пам'ять, якщо ці дані використовуються порівняно рідко.

Допустимо, що у певний момент часу один клієнт може зробити тільки одне замовлення. У цьому випадку між об'єктами КЛІЄНТ і ЗАМОВЛЕННЯ установлюється взаємозв'язок "один до одного", що позначається одинарними стрілками, як це показано на рис. 4 а.

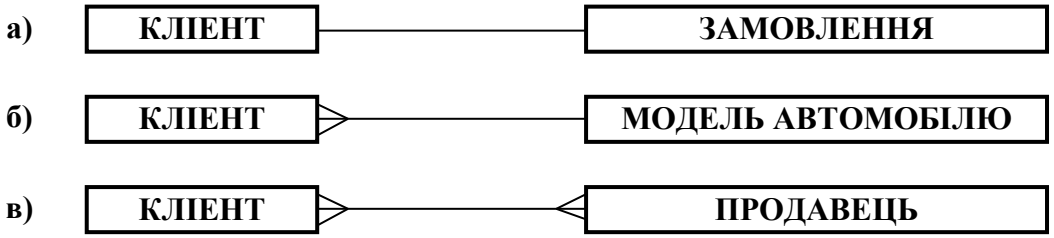

Рис. 4 Взаємозв'язки між двом об'єктами: а) "один до одного"; б) "один до багатьох"; в) "багато хто до багатьох"

Між даними, що зберігаються в об'єктах КЛІЄНТ і ЗАМОВЛЕННЯ, буде існувати взаємозв'язок, у якому кожен запис в одному об'єкті буде однозначно вказувати на запис в іншому об'єкті. На Рис. 5 приведено приклад такого взаємозв'язку між даними. Ні в першому, ні в другому об'єкті не може існувати запису, не зв'язаного з яким-небудь записом в

іншому об'єкті.

Рис. 5 Взаємозв'язок між даними при відношенні "один до одного"

4.2.1.1. Взаємозв'язок "один до багатьох" (між двома типами об'єктів)

Зв'язок “один до багатьох” у більшості випадків відображає реальний взаємозв'язок сутностей у предметній області. Вона реалізується парою “зовнішній ключ – первинний ключ”, тобто коли визначений зовнішній ключ, що посилається на первинний ключ іншої таблиці. Саме цей зв'язок описує широко розповсюджений механізм класифікаторів. Мається довідкова таблиця, що містить назви, імена і т.п. і деякі коди, причому, первинним ключем є код. У таблиці, що збирає інформацію - назвемо її інформаційною таблицею - визначається зовнішній ключ, що посилається на первинний ключ класифікатора. Після цього в неї заноситься не назва з класифікатора, а код. Така система стає стійкою від зміни назви в класифікаторах. Існують способи швидкої “підміни” у відображуваній таблиці кодів на їхні назви як на рівні сервера БД (для клієнт-серверних СУБД), так і на рівні користувальницького додатка.

У певний момент часу один клієнт може стати власником декількох моделей автомобілів, але при цьому декілька клієнтів не можуть бути власниками одного автомобіля. Взаємозв'язок "один до багатьох" можна позначити за допомогою одинарної стрілки в напрямку до "одному" і

подвійної стрілки в напрямку до "багатьох", як це показано на рис. 4,б.

У цьому випадку одного запису даних першого об'єкта (його часто називають батьківським чи основним) буде відповідати кілька записів другого об'єкта (дочірнього чи підлеглого). Взаємозв'язок "один до багатьох" дуже поширена при розробці реляційних баз даних. Як батьківський об'єкт часто виступає довідник, а в дочірньому зберігаються унікальні ключі для доступу до записів довідника.

У нашому прикладі як такий довідник можна представити об'єкт КЛІЄНТ, у якому зберігаються відомості про всіх клієнтів. При звертанні до запису для окремого клієнта нам доступний список усіх покупок, що він зробив, і відомості, про які зберігаються в об'єкті МОДЕЛЬ, як це показано на: рис.6.

Об’єкт КЛІЄНТ

УНИКАЛЬНЫЙ КЛЮЧ	НАИМЕНОВАНИЕ КЛИЕНТА
1	Петров
2	Сідоров

Об’єкт МОДЕЛЬ АВТОМОБІЛЯ

УНІКАЛЬНИЙ КЛЮЧ МОДЕЛІ	НАЙМЕНУВАННЯ МОДЕЛІ	КЛЮЧ КЛІЄНТА
7	Toyota 3.8	1
8	ZX 2.0	2
9	Stealth 3.0	1
10	348 Spider 3.4	1

Рис.6 Взаємозв'язок між даними при відношенні "один до багатьох"

У випадку, якщо в дочірньому об'єкті будуть якісь записи, для яких немає відповідних записів в об'єкті КЛІЄНТ, то ми їх не побачимо. У цьому випадку говорять, що об'єкт містить загублені (самотні) записи. Це неприпустимо, і надалі ви довідаєтеся, як уникати подібних ситуацій.

Якщо ми будемо переглядати запису об'єкта МОДЕЛЬ, то в об'єкті КЛІЄНТ ми зможемо одержати дані про клієнта, що купив даний автомобіль (див. рис. 6). Зверніть увагу, що для загублених записів

відомостей про клієнта ми не одержимо.

4.2.1.3. Взаємозв'язок "багато хто до багатьох" (між двома типами об'єктів)

Відповідно до теорії реляційних баз даних для збереження взаємозв'язку "багато хто до багатьох" вимагаються три об'єкти: по одному для кожної сутності й один для збереження зв'язків між ними (проміжний об'єкт). Проміжний об'єкт буде містити ідентифікатори зв'язаних об'єктів, як це показано на рис.7.

Зв'язок “багато хто до багатьох” у явному виді в реляційних СУБД не підтримується. Один з найбільш розповсюджених способів непрямої реалізації такого виду зв'язків полягає у введенні додаткової таблиці, рядка якої складаються з зовнішніх ключів, що посилаються на первинні ключі взаємозалежних таблиць.

У розглянутому нами прикладі кожен продавець може обслуговувати декількох клієнтів. З іншого боку, здобуваючи автомобілі в різний час, кожен клієнт цілком може бути обслуговий різними продавцями. Між об'єктами КЛІЄНТ і ПРОДАВЕЦЬ існує взаимосвязь "багато хто до багатьох". Такий взаємозв'язок позначається подвійними стрілками, як це показано на рис. 4 в. Як додатковий об'єкт між цими об'єктами може бути використаний об'єкт ЗАМОВЛЕННЯ.

Рис. 7 Відображення взаємозв'язку між даними при відношенні "багато хто до багатьох" за допомогою проміжного об'єкта

Взаємозв'язок між об'єктами є частиною концептуальної моделі і повинні відображатися в базі даних.

Отримана після цього логічна модель представлена на рис.8 (розробляти за допомогою Oracle Datamodeler).

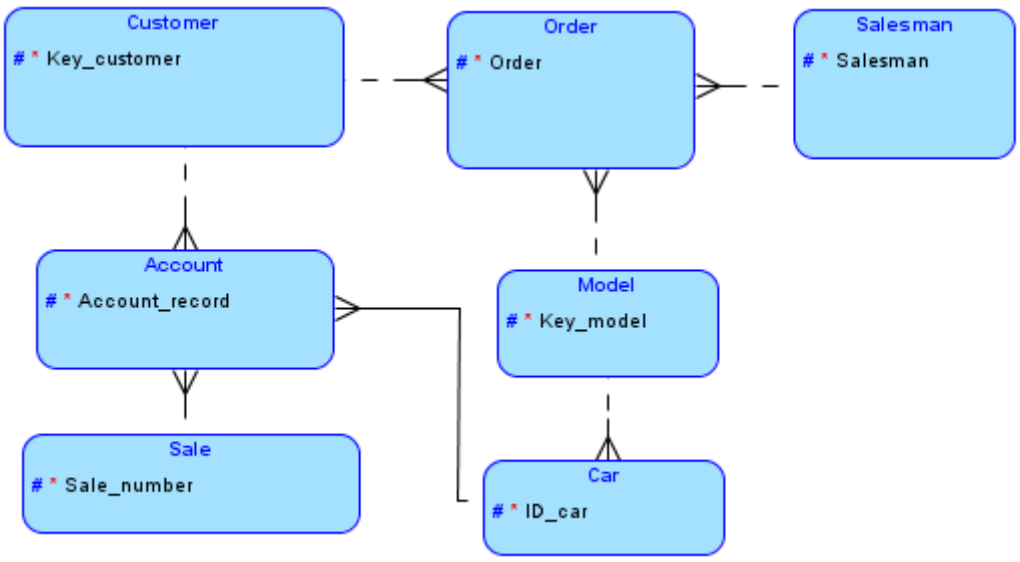

Рис. 8 Логічна модель БД

Необхідно відзначити, що на рис. 5 ,а взаємозв'язок між об'єктами КЛІЄНТ і ЗАМОВЛЕННЯ розглядається у визначений момент часу, для прикладу зв'язку "один до одному". Однак, аналізуючи даний взаємозв'язок більш широко, одержимо, що один клієнт у різний час може робити кілька замовлень. З іншого боку, одне замовлення належить тільки одному клієнту і тому на рис. 7 між сутностями КЛІЄНТ і ЗАМОВЛЕННЯ установлене взаємозв'язок «один до багатьох».

4.3. Завдання первинних і альтернативних ключів, визначення атрибутів сутностей

Визначити первинні й альтернативні ключі сутностей. Привести схему взаємозв'язків сутностей по первинних і альтернативних ключах, а також таблицю атрибутів сутностей для обраної моделі.

4.3.1. Загальні зауваження (міркування)

- Таблиця – це деяка регулярна структура, що складається з кінцевого набору полів по вертикалі й однотипних записах по горизонталі. У реляційній моделі даних таблиця називається відношенням.
- Атрибут – це інформаційне відображення властивостей об'єкта. Кожен об'єкт характеризується поруч основних атрибутів.

Наприклад, модель автомобіля характеризується типом кузова, робочим обсягом двигуна, кількістю циліндрів, потужністю, габаритами, назвою і т.д. Кожен атрибут у моделі повинний мати унікальне ім'я – ідентифікатор. Атрибут при реалізації інформаційної моделі на якому-небудь носії інформації часто називають елементом даних, полем даних чи просто полем. При цьому, у рядках таблиць розташовані записи.

Для кожної сутності необхідно визначати атрибути, що ми будемо зберігати в БД. При цьому необхідно враховувати той факт, що при переході від логічної до фізичної моделі даних може відбутися усікання числа об'єктів. Насправді, як правило, значне число даних, необхідних

користувачу, може бути досить легко підраховане в момент висновку інформації. У той же час, у зв'язку зі зміною алгоритмів чи розрахунку вихідних величин, деякі розрахункові показники приходиться записувати в БД, щоб гарантованно забезпечити фіксацію їхніх значень. Вибір показників, що обов'язково варто зберігати в БД, досить складний. Нечасто можна знайти однозначне рішення цієї проблеми, і в будь-якому випадку воно зажадає ретельного вивчення роботи підприємства й аналізу концептуальної моделі.

Інформацію про деяку предметну область можна представити за допомогою декількох об'єктів, кожний з який описується декількома елементами даних. Об'єкти зв'язуються між собою певним чином.

Деякі елементи даних володіють важливим для побудови інформаційної моделі властивістю. Якщо відоме значення, що приймає такий елемент даних об'єкта, ми можемо ідентифікувати значення, що приймають інші елементи даних цього ж об'єкта. Наприклад, знаючи унікальний номер моделі автомобіля – 7, ми можемо визначити, що це " Toyota 3.8" 11, що робочий обсяг двигуна в даної моделі "3778".

- Ключовим елементом даних у таблиці називається такий елемент, за яким можна визначити значення інших елементів даних.

Однозначно ідентифікувати об'єкт можуть два і більш елементи даних. У цьому випадку їх називають "кандидатами" у ключові елементи даних. Питання про те, який з кандидатів використовувати для доступу до об'єкта, зважується чи користувачем розроблювачем системи. Вибирати ключові елементи даних випливає ретельно, оскільки правильний вибір сприяє забезпеченню цілісності даних у базі даних.

- Первинний ключ – це атрибут (чи група атрибутів), що єдиним образом ідентифікують кожен рядок у таблиці.

Поняття первинного ключа є винятково важливим у зв'язку з поняттям цілісності баз даних. Первинний ключ не повинний мати додаткових

атрибутів. Це значить, що якщо з первинного ключа виключити довільний атрибут, що залишилися атрибутів буде недостатньо для однозначної ідентифікації окремих кортежів. Для прискорення доступу по первинному ключі у всіх системах керування базами даних (СУБД) мається механізм, називаний індексуванням. Грубо говорячи, індекс являє собою інвертований деревоподібний список, що вказує на істинне місце розташування запису для кожного первинного ключа. Природно, у різних СУБД індекси реалізовані по-різному (у локальних СУБД - як правило, у виді окремих файлів), однак, принципи їхньої організації однакові.

- Альтернативний (зовнішній) ключ – це атрибут (чи група атрибутів), незбіжний з первинним ключем і унікально ідентифікуючий екземпляр об'єкта.

Даний тип індексу застосовується для індексування відносини з використанням атрибутів, відмінних від первинного ключа з метою зменшення часу доступу при перебуванні даних у відношенні, а також для сортування. Таким чином, якщо саме відношення не упорядковане яким-небудь образом і в ньому можуть бути присутнім рядки, що залишилися після видалення деяких записів, то індекс (для локальних СУБД - індексний файл), навпроти, відсортований.

Наприклад, для об'єкта "СЛУЖБОВЕЦЬ", що має атрибути "Ідентифікатор робітника", "Прізвище", "Ім'я" і "По батькові", група атрибутів "Прізвище", " Ім'я ", " По батькові " може бути альтернативним ключем стосовно атрибута " Ідентифікатор робітника " (у припущенні, що на підприємстві не працюють повні тезки).

Для підтримки посилальної цілісності даних у багатьох СУБД мається механізм так званих зовнішніх ключів. Зміст цього механізму полягає в тому, що деякому атрибуту (чи групі атрибутів) одного відношення призначається посилання на первинний ключ іншого відношення; тим самим закріплюються зв'язки підпорядкованості між цими відносинами. При цьому відношення, на первинний ключ якого посилається зовнішній

ключ іншого відношення, називається master-відношенням, чи головним відношенням; а відношення, від якого виходить посилання, називається detail-відношенням, чи підлеглим відношенням. Після призначення такого посилання СУБД має можливість автоматично відслідковувати питання “непорушення“ зв'язків між відносинами, а саме:

* якщо Ви спробуєте вставити в підлеглу таблицю запис, для зовнішнього ключа якої не існує відповідності в головній таблиці (наприклад, там немає ще запису з таким первинним ключем), СУБД згенерує помилку;
* якщо Ви спробуєте видалити з головної таблиці запис, на первинний ключ якої мається хоча б одне посилання з підлеглої таблиці, СУБД також згенерує помилку.
* якщо Ви спробуєте змінити первинний ключ запису головної таблиці, на яку мається хоча б одне посилання з підлеглої таблиці, СУБД також згенерує помилку.

Якщо між деякими сутностями існує зв'язок, то факти з однієї сутності чи посилаються деяким чином зв'язані з фактами з іншої сутності.

Зв'язок – це функціональна залежність між сутностями.

Підтримка несуперечності функціональних залежностей між сутностями називається посилальною цілісністю.

Посилальна цілісність – це забезпечення відповідності значення зовнішнього ключа екземпляра дочірньої сутності значенням первинного ключа в батьківській сутності.

Посилальна цілісність може контролюватися при всіх операціях, що змінює дані. Для кожного зв'язку на логічному рівні можуть бути задані вимоги по обробці операцій додавання, чи відновлення видалення даних для батьківської і дочірньої сутності.

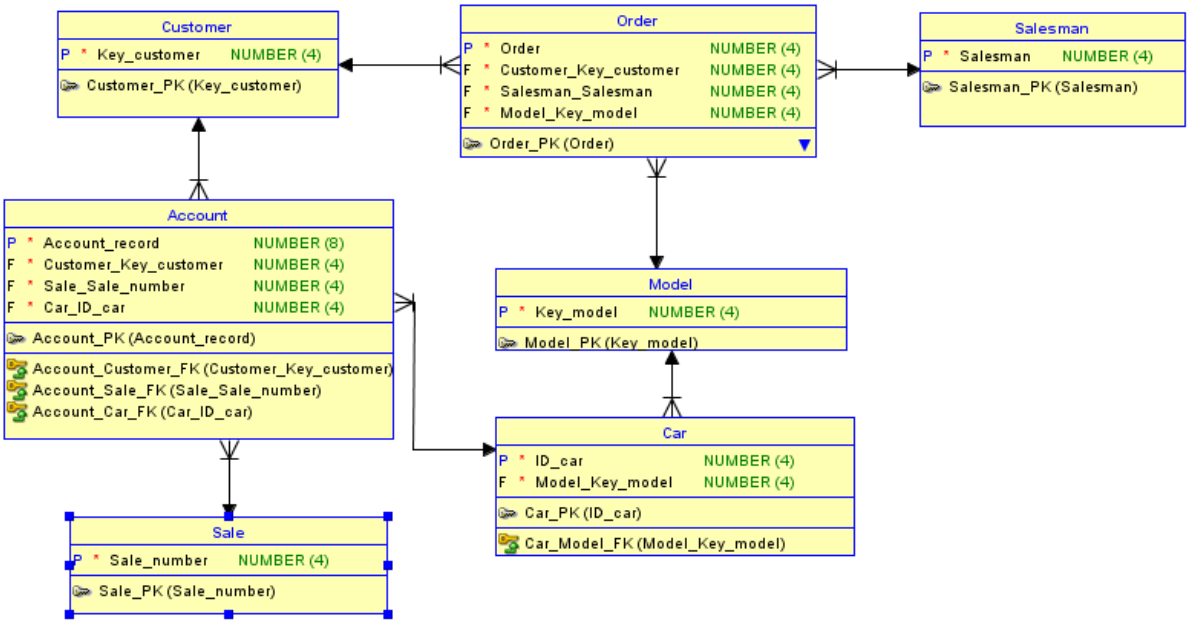

Рис. 9 ER - діаграма після третього етапу проектування.

Атрибути, що включаються до складу БД для розглянутої моделі, приведені в табл. 4.1.

Атрибути і первинні ключі сутностей інформаційної моделі **Таблиця 4.1.**

Сутність	Первинний ключ	Атрибути
МОДЕЛЬ	Унікальний ключ моделі	Унікальний ключ моделі Найменування моделі Найменування фірми Найменування країни Робочий обсяг двигуна Кількість циліндрів Потужність Момент, що крутить Найменування палива Максимальна швидкість Час розвантажування до 100 км/год Найменування шин Найменування кузова Кількість двер Кількість місць Довжина Ширина Висота Витрата палива при 90

		км/год Витрата палива при 120 км/год Витрата палива при міському циклі
АВТОМОБІЛЬ	Унікальний ключ автомобіля	Унікальний ключ автомобіля Унікальний ключ моделі Дата випуску Вартість
КЛІЄНТ	Унікальний ключ клієнта	Унікальний ключ клієнта Найменування клієнта Адреса Телефон Факс Прізвище Ім'я По батькові Ознака юридичної особи Примітка
ПРОДАЖ	Унікальний ключ продажу	Унікальний ключ продажу Унікальний номер рахунка Дата продажу Сума
РАХУНОК	Унікальний номер рахунка	Унікальний номер рахунка Унікальний ключ клієнта Унікальний ключ автомобіля Дата виписки Позначка про оплату Сума
ЗАМОВЛЕННЯ	Унікальний ключ замовлення	Унікальний ключ замовлення Унікальний ключ клієнта Унікальний ключ моделі Унікальний ключ продавця
ПРОДАВЕЦЬ	Унікальний ключ продавця	Унікальний ключ продавця Прізвище Ім'я По батькові Адреса Телефон

4.4. Приведення моделі до необхідного рівня нормальної форми;

Привести результати нормалізації моделі і схему взаємозв'язків між атрибутами сутностей після нормалізації моделі, а також таблицю атрибутів і первинних ключів, змінених чи доданих сутностей логічної моделі.

4.4.1. Загальні зауваження (міркування)

Приведення моделі до необхідного рівня нормальної форми є основою побудови реляційної БД. У процесі нормалізації елементи даних групуються в таблиці, що представляють об'єкти і їхні взаємозв'язки.

Нормалізація відносин – це процес побудови оптимальної структури таблиць і зв'язків у реляційній БД.

Теорія нормалізації заснована на тому, що визначений набір таблиць має кращі властивості при включенні модифікації і видаленні даних, чим всі інші набори таблиць, за допомогою яких можуть бути представлені ті ж дані. Введення нормалізації відносин при розробці інформаційної моделі забезпечує її мінімальний обсяг при записі на якому-небудь носії БД і її максимальна швидкодія, що прямо відбивається на якості функціонування інформаційної системи.

Отже, після визначення таблиць, полів, індексів і зв'язків між таблицями варто подивитися на проектовану базу даних у цілому і проаналізувати її, використовуючи правила нормалізації, з метою усунення логічних помилок. Важливість нормалізації полягає в тому, що вона дозволяє розбити великі відносини, як правило, що містять велику надмірність інформації, на більш дрібні логічні одиниці, що групують тільки дані, об'єднані “по природі”. Таким чином, ідея нормалізації полягає в наступному. Кожна таблиця в реляційній базі даних задовольняє умові, відповідно до якої в позиції на перетинанні кожного рядка і стовпця таблиці завжди знаходиться єдине значення, і ніколи не може бути безлічі таких значень.

Після застосування правил нормалізації логічні групи даних

розташовуються не більш ніж в одній таблиці. Це дає наступні переваги:

- ♦ дані легко обновляти чи видаляти
- ♦ виключається можливість неузгодженості копій даних
- ♦ зменшується можливість введення некоректних даних.

Нормалізація інформаційної моделі виконується в кілька етапів.

Усі вихідні дані, представлені у виді двовимірної таблиці, є першою нормальною формою реляційної моделі даних.

Перший етап нормалізації полягає в утворенні двовимірної таблиці, що містить усі необхідні атрибути інформаційної моделі, і у виділенні ключових атрибутів. Очевидно, що отримана дуже значна таблиця буде містити дуже різнорідну інформацію. У цьому випадку будуть спостерігатися аномалії включення, відновлення і видалення даних, тому що при виконанні цих дій нам доведеться приділити увага даним (вводити чи піклуватися про те, щоб вони не були стерті), що не мають до поточних дій ніякого відношення. Наприклад, може спостерігатися така парадоксальна ситуація. При включенні в каталог продажів нової моделі автомобіля нам відразу прийдеться вказати її клієнта, що купив. Тому, перший етап намагаються пропускати, представляючи логічну модель у другій нормальній формі. Приведення відносин до другої нормальної форми полягає в забезпеченні повної функціональної залежності всіх атрибутів від ключа за рахунок розбивки загальної таблиці на трохи частках, у яких усі наявні атрибути будуть мати повну функціональну залежність від ключа цієї таблиці. У процесі приведення моделі до другої нормальної форми в основному виключаються аномалії дублювання даних. Відношення задане в другій нормальній формі, якщо воно є відношенням у першій нормальній формі і кожен атрибут, що не є первинним атрибутом у цьому відношенні, цілком залежить від будь-якого можливого ключа цього відношення.

Якщо всі можливі ключі відносини містять по одному атрибуті, то це відношення задане в другій нормальній формі, тому що в цьому випадку

всі атрибути, що не є первинними, цілком залежать від можливих ключів. Якщо ключі складаються більш ніж з одного атрибута, відношення, задане в першій нормальній формі, може не бути відношенням у другій нормальній формі.

Відношення задане в третій нормальній формі, якщо воно задано в другій нормальній формі і кожен атрибут цього відношення, що не є первинним, не транзитивно залежить від кожного можливого ключа цього відношення.

Транзитивна залежність виявляє дублювання даних в одному відношенні. Якщо А, В и С – три атрибути одного відношення і З залежить від У, а У від А, то говорять, що З транзитивно залежить від А, як це схематично показано на рис . 4,а. Перетворення в третю нормальну форму відбувається за рахунок поділу вихідного відношення на два, як це показано на рис. 4,б.

Наприклад, якщо всі дані про моделі автомобілів і самих автомобілів, що надходять, зберігаються в одному відношенні, то для декількох автомобілів однієї моделі довелося б багаторазово вказувати тип кузова, кількість двер і інші технічні характеристики. У цьому випадку технічні характеристики залежать від моделі автомобіля, а при наявності декількох автомобілів однієї моделі будуть дублюватися. Дублювання зникає, якщо з одного відношення виділити відношення, у якому будуть зберігатися дані про моделі, і відношення, у якому будуть зберігатися дані про автомобілі.

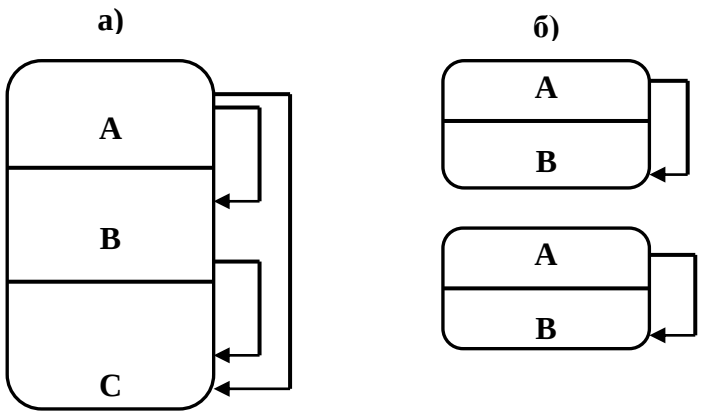

Рис. 9 Приклад транзитивної залежності:
а) відносини між об'єктами з транзитивною залежністю;
б) відносини між об'єктами без транзитивної залежності

Існують і більш високі форми нормалізації, але, як правило, у більшості випадків достатнє використання описаних вище перших трьох форм.

Сформулюємо основні правила, яким потрібно випливати при проектуванні бази даних:

- Виключайте повторювані групи – для кожного набору зв'язаних атрибутів створіть окрему таблицю і наділіть її первинним ключем. Виконання цього правила автоматично приведе до другої нормальної форми. Крім теоретичних вказівок у цьому правилі є і чисто практичний зміст. Представте, що у вашому списку замовлень ви вказуєте імена ваших клієнтів. Клієнт «Петров» досить активен і часто робить у вас замовлення. Імовірніше всього, що знайдуться лічені люди, що у десятьох згадуваннях напишуть це ім'я абсолютно однаково. Хтось де-небудь так і напише «Петренко», а для СУБД це вже інший клієнт. Тому набагато краще вести список своїх клієнтів в окремій таблиці, а в списку замовлень використовувати тільки привласнені їм унікальні ідентифікатори.
- Виключайте надлишкові дані – якщо атрибут залежить тільки від частини, складеного ключа, перемістите атрибут в окрему таблицю.

Це правило допомагає уникнути втрати одних даних при видаленні якихось інших. Скрізь, де можливе використання ідентифікаторів замість опису, виносите в окрему таблицю список ідентифікаторів з поясненнями до них.

- Виключайте стовпці, що не залежать від ключа – якщо атрибути не вносять свою лепту в опис ключа, перемістите їх в окрему таблицю.

З огляду вище викладеного в нашій моделі необхідно видозмінити список атрибутів сутності МОДЕЛЬ і додати такі нові сутності, як ПАЛИВО, ШИНИ, КУЗОВ, ФІРМА, КРАЇНА (рис. 10).

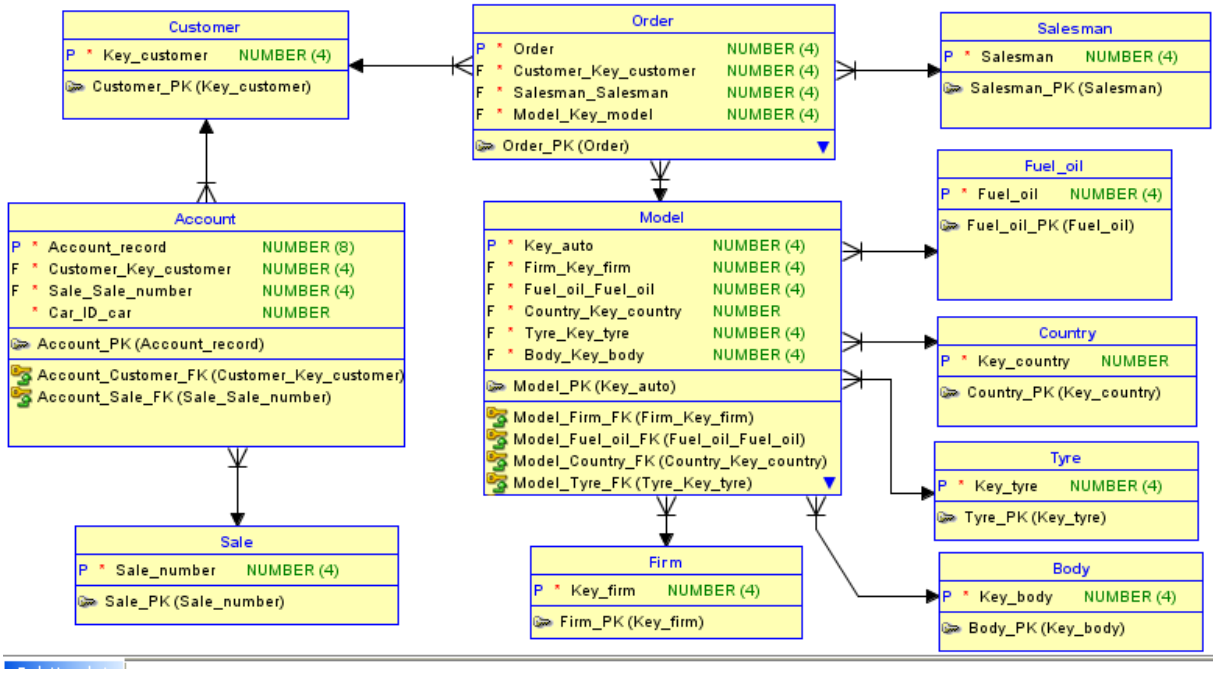

Рис. 10 ER-диаграмма взаємозв’язків між атрибутами сутностей після нормалізації моделі

В основному зміни в моделі зв'язані з введенням штучних атрибутів, що у виді кодів беруть участь у відносинах замість природних атрибутів (вид палива, марка шин і т.п.). До необхідності введення в модель штучних атрибутів ми прийшли в процесі нормалізації. У загальному випадку можна рекомендувати використовувати замість природних атрибутів коди в наступних випадках:

- У предметній області може спостерігатися синомія, тобто природний атрибут відносини не має властивості унікальність. Наприклад, серед співробітників фірми можуть бути однофамільці чи навіть повні

тезки. У цьому випадку вирішити проблему допомагає унікальний табельний номер.

- Якщо відношення бере участь у багатьох зв'язках, то для їхнього відображення створюється кілька таблиць, у кожній з який повторюється ідентифікатор відносини. Для того щоб не використовувати у всіх таблицях довгий природний атрибут об'єкта, можна застосовувати більш короткий код. Це також буде сприяти підвищенню швидкодії вашої системи.
- Якщо природний атрибут може змінюватися в часі (наприклад, прізвище), то це може викликати дуже великі складності при експлуатації системи. Представте, що ваш кращий продавець, дівчина Карина, вийшла заміж. Що буде з даними, що прив'язані до її дівочого прізвища? Використання незмінного коду (табельного номера) дозволить уникнути цих складностей.

Атрибути, що включаються в змінені чи додані в модель сутності, приведені в табл. 4.2.

Атрибути і первинні ключі змінених чи доданих сутностей логічної моделі **Таблиця 4.2.**

Сутність	Первинний ключ	Атрибути
МОДЕЛЬ	Унікальний ключ моделі	Унікальний ключ моделі Найменування моделі Унікальний ключ фірми Робочий обсяг двигуна Кількість циліндрів Потужність Момент, що крутить Унікальний ключ палива Максимальна швидкість Час розвантажування до 100 км/год Унікальний ключ шин Унікальний ключ кузова

Сутність	Первинний ключ	Атрибути
		Кількість двер Кількість місць Довжина Ширина Висота Витрата палива при 90 км/год Витрата палива при 120 км/год Витрата палива при міському циклі
ПАЛИВО	Унікальний ключ палива	Унікальний ключ палива Найменування палива
ШИНИ	Унікальний ключ шин	Унікальний ключ шин Найменування шин
КУЗОВ	Унікальний ключ кузова	Унікальний ключ кузова Найменування кузова
ФІРМА	Унікальний ключ фірми	Унікальний ключ фірми Найменування фірми Унікальний ключ країни
КРАЇНА	Унікальний ключ країни	Унікальний ключ країни Найменування країни

5. СТВОРЕННЯ ФІЗИЧНОЇ МОДЕЛІ

Привести проект таблиці для фізичної моделі з урахуванням типів даних обраної СУБД.

5.1. Загальні зауваження (міркування)

Як уже говорилася вище, ✍ Таблиця – це деяка регулярна структура, що складається з кінцевого набору полів по вертикалі й однотипних записах по горизонталі. У реляційній моделі даних таблиця називається відношенням.

✍ Атрибут – це інформаційне відображення властивостей об'єкта. Кожен об'єкт характеризується поруч основних атрибутів.

Наприклад, модель автомобіля характеризується типом кузова, робочим обсягом двигуна, кількістю циліндрів, потужністю, габаритами, назвою і т.д. Кожен атрибут у моделі повинний мати унікальне ім'я – ідентифікатор. Атрибут при реалізації інформаційної моделі на якому-небудь носії інформації часто називають елементом даних, полем даних чи просто полем. При цьому, у рядках таблиць розташовані записи.

Запис даних - це сукупність значень зв'язаних елементів даних.

Кожен запис однієї таблиці складається з кінцевого (і однакового!) числа полів, причому конкретне поле кожного запису однієї таблиці може містити дані тільки одного типу. Наприклад, елемент даних полючи "НАЙМЕНУВАННЯ МОДЕЛІ" може приймати такі значення, як " Toyota 3.8 Grand", "Continental 4.6" чи "Crown Victoria 4.6". На рис. 11 такими елементами даних є унікальний ключ і найменування моделі, робочий обсяг, кількість циліндрів і потужність двигуна. Наприклад, один із записів – "7 Toyota 3.8 Grand 3778 6 164,0". Цей рядок являє собою значення, що приймають елементи даних об'єкта МОДЕЛЬ.

✍ Тип даних характеризує кожний з елементів запису.

Поняття типу даних у фізичній моделі цілком адекватно поняттю типу даних у мовах програмування. Звичайно в сучасних СУБД допускається збереження символьних, числових даних, бітових рядків, спеціалізованих числових даних (наприклад, суми в грошових одиницях), а також даних спеціального формату (дата, час, часовий інтервал і ін.). У будь-якому випадку при виборі типу даних варто враховувати можливості тієї СУБД або системи програмування, за допомогою якої буде реалізовуватися фізична модель інформаційної системи.

Призначення імен таблиць і їхніх атрибутів відбиті в у табл. 5.1.

Проект таблиць для фізичної моделі **Таблиця 5.1.**

Model (МОДЕЛЬ)				**1**
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_model	Унікальний ключ моделі	number	4
2.	Name_model	Найменування моделі	Varchar 2	20
3.	Key_firm	Унікальний ключ фірми		
4.	Swept_volume	Робочий обсяг двигуна		
5.	Quantity_drum	Кількість циліндрів		
6.	Capacity	Потужність		
7.	Torgue	Момент, що крутить		
8.	Key_fuel_oil	Унікальний ключ палива		
9.	Top_speed	Максимальна швидкість		
10.	Starting	Час розвантажування до 100 км/год		
11.	Key_tyre	Унікальний ключ шин		
12.	Key_body	Унікальний ключ кузова		
13.	Quantity_door	Кількість двер		
14.	Quantity_sead	Кількість місць		
15.	Length	Довжина		
16.	Width	Ширина		
17.	Height	Висота		
18.	Expense_90	Витрата палива при 90 км/год		
19.	Expense_120	Витрата палива при 120 км/год		
20.	Expense_town	Витрата палива при міському циклі		

Firm (ФІРМА)				2
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_firm	Унікальний ключ фірми	number	4
2.	Name_firm	Найменування фірми	Varchar 2	20
3.	Key_country	Унікальний ключ країни	number	4

Country (КРАЇНА)				3
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_country	Унікальний ключ країни	number	4
2.	Name_country	Найменування країни	Varchar 2	20

Fuel_oil (ФІРМА)				4
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_fuel_oil	Унікальний ключ палива	number	4
2.	Name_ fuel_oil	Найменування палива	Varchar 2	20

Tyre (ШИНИ)				5
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_tyre	Унікальний ключ шин	number	4
2.	Name_ tyre	Найменування шин	Varchar 2	20

Body (КУЗОВ)				6
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_body	Унікальний ключ кузова	number	4
2.	Name_ body	Найменування кузова	Varchar 2	20

Model (Автомобіль)				7
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_auto	Унікальний ключ автомобіля	number	4
2.	Key_model	Унікальний ключ моделі	Varchar 2	20
3.	Date_issue	Дата випуску у форматі ДД.ММ.РР		
4.	Cost	Вартість, $		

Customer (КЛІЄНТ)				8
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_customer	Унікальний ключ клієнта	number	4
2.	Name_customerl	Найменування клієнта	Varchar 2	20
3.	Address	Адреса		
4.	Tel	Телефон		
5.	Fax	Факс		
6.	Last_name	Прізвище		
7.	First_name	Ім'я		
8.	Patronymic	По батькові		
9.	Juridical	Ознака юридичної особи		
10.	Comment	Примітка		

Sale (ПРОДАЖ)				9
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Sale_number	Номер продажу	number	4
2.	Date_sale	Дата продажу у форматі ДД.ММ.РР	Varchar 2	20
3.	Sum	Сума, $	number	6,2

Account (РАХУНОК)				10
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Account_record	Номер рахунка	number	4
2.	Key_customer	Унікальний ключ клієнта	number	4
3.	Key_auto	Унікальний ключ автомобіля		
4.	Date_write	Дата випуску у форматі ДД.ММ.РР		
5.	Selled	Позначка про оплату		
6.	Sum	Сума, $		

Order (ЗАМОВЛЕННЯ)				10
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_order	Унікальний ключ замовлення	number	4
2.	Key_customer	Унікальний ключ клієнта	number	4
3.	Key_model	Унікальний ключ моделі		
4.	Key_salesman	Унікальний ключ продавця		

Salesman (ПРОДАВЕЦЬ)				10
№ п/п	**Найменування стовбців**	**Примітка**	**Тип**	**Розмір**
1.	Key_salesman	Унікальний ключ продавця	number	4
2.	Last_name	Прізвище	Varchar 2	20
3.	First_name	Ім'я		
4.	Patronymic	По батькові		
5.	Address	Адреса		
6.	Tel	Телефон		

6. ОПИС ПРОГРАМИ

1. Логічна структура

Привести логічну структуру програми.

2. Екранні форми

Привести загальний вид екранних форм для введення і редагування інформації в базі даних.

3. Звіти

Привести загальний вид виведених звітів для конкретного тестового приклада.

4. Текст програми

Привести текст програми з коментарями, що виноситься в Додаток пояснювальної записки.

5. Список літератури

Список літератури приводять на окремій сторінці. Приклад бібліографічного опису літератури приведений нижче і є літературою, що рекомендується для виконання курсової роботи.

6. Додаток

Прикласти CD або DVD , що містить текст Пояснювальної записки, вихідні тексти програми і скомпільований EXE-модуль.

ЛІТЕРАТУРА

1. Анісімов А.В. Інформаційні системи та бази даних: Навчальний посібник для студентів факультету комп'ютерних наук та кібернетики. / Анісімов А.В., Кулябко П.П. − Київ. – 2017. – 110 с.
2. Харів Н. О. Бази даних та інформаційні системи: навчальний посібник / Н. О. Харів. – Рівне: НУВГП, 2018. – 127 с.
3. Мулеса О.Ю. Інформаційні системи та реляційні бази даних. Навч. посібник. – Електронне видання, 2018. – 118 с.
4. Остапченко К.Б. Бази даних. Комп'ютерний практикум : Навчальний посібник/ К.Б. Остапченко. – Київ. - НТУ КПІ ім. Ігоря Сікорського, 2022. – 251 с.
5. Charalambides S. Oracle SQL Tuning with Oracle SQLTXPLAIN: Oracle Database 12c Edition / Apress, 2017. – 408 p. – ISBN 978-1484224359
6. Пасічник В. В. / Організація баз даних та знань.// В. В. Пасічник, В. А. Резниченко /– К.: Видавнича група BHV, 2006. – 384 с.
7. Берко, А. Ю. Системи баз даних та знань: навч. посібник. Кн. 1. Організація баз даних та знань / А. Ю. Берко, О. М. Верес, В. В. Пасічник. - Л.: "Магнолія 2006", – 2008. – 456 с. – ISBN 978-966-2025-56-9
8. Pachev S. Understanding MySQL Internals // O'Reilly Media, – 2007 – 252 p. – ISBN: 9780596009571
9. James R., Paul N. Weinberg, Andy Oppel / SQL The Complete Reference, 3rd Editi // Mc Graw Hill – 2010 – 912 p.
10. Сервер Oracle 10G: Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.
11. Ms Visual Studio 2017 Режим електронного доступу – [https://www.microsoft.com/Ukraine/News/default.mspx].

ДОДАТОК А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет: Інформаційних технологій

Кафедра: Програмного забезпечення комп'ютерних систем
(повна назва кафедри)

КУРСОВА РОБОТА

з дисципліни: **ОРГАНІЗАЦІЯ БАЗ ДАНИХ ТА ЗНАНЬ**
(назва дисципліни)

на тему: **«Розробка інформаційної системи »**

Студента (ки) ____ курсу, групи_____

Спеціальності_____

_____
(прізвище та ініціали)
Керівник _доцент, к.т.н., доцент _____
___Кабак Л.В._____
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____Оцінка: ECTS _____

_____ Кабак Л.В.
(підпис) (прізвище та ініціали)

м. Дніпро – 202_ рік