Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №2

із дисципліни «**Технології розроблення програмного забезпечення**»

Основи проектування.

Аудіоредактор

Виконала
групи ІА–31
Биковська О. І.

Перевірив студентка
викладач
Мягкий М.Ю.

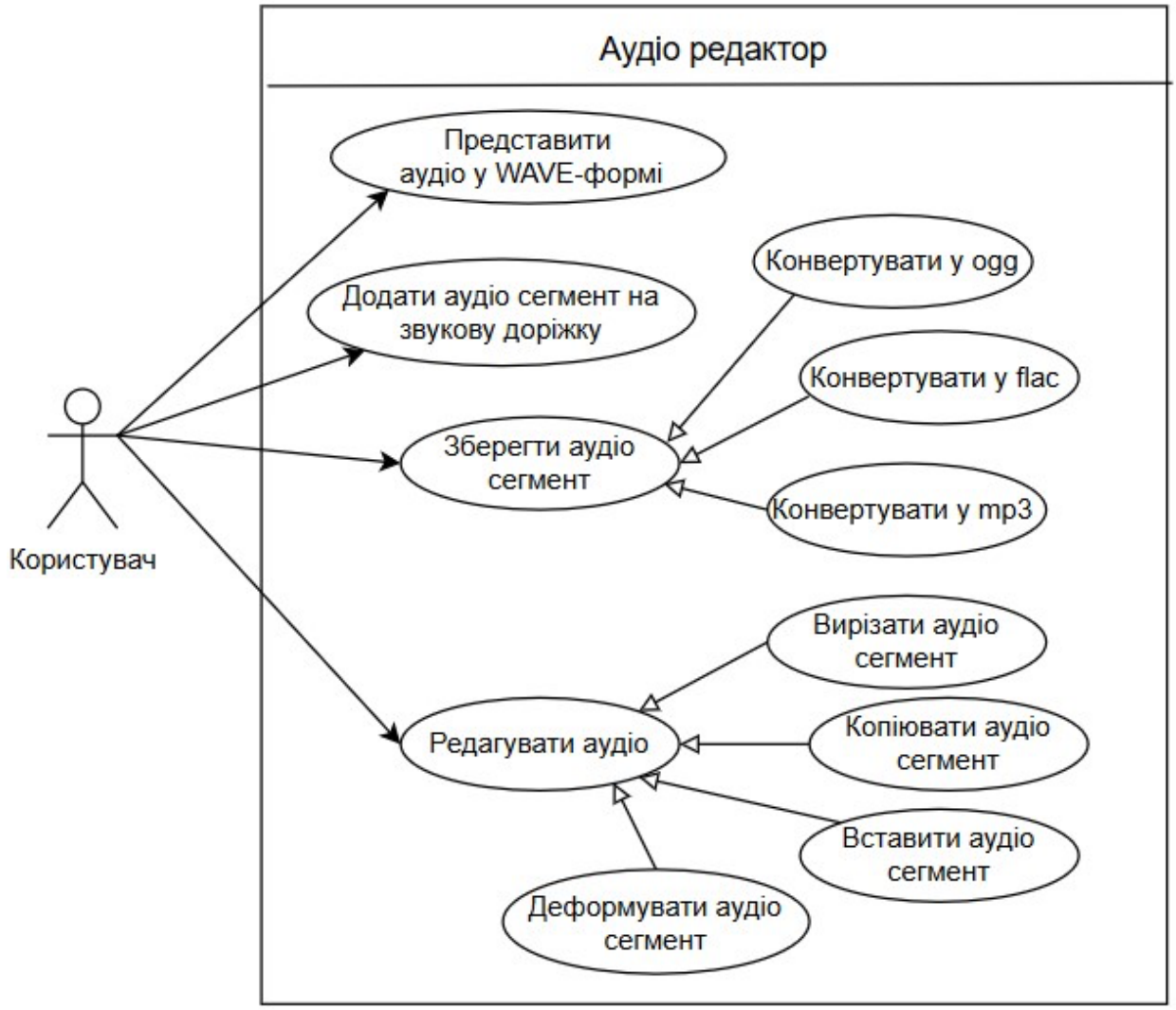
Зміст

Мета: Обрати зручну систему побудови UML-діаграм та навчитися будувати діаграми варіантів використання для системи що проєктується, розробляти сценарії варіантів використання та будувати діаграми класів предметної області.

Хід роботи

1. Ознайомитися з короткими теоретичними відомостями.
2. Проаналізувати тему та спроектувати діаграму варіантів використання відповідно до обраної теми лабораторного циклу.
3. Спроектувати діаграму класів предметної області.
4. Вибрати 3 варіанти використання та написати за ними сценарії
5. використання.
6. На основі спроектованої діаграми класів предметної області розробити основні класи та структуру бази даних системи. Класи даних повинні реалізувати шаблон Repository для взаємодії з базою даних.
7. Нарисувати діаграму класів для реалізованої частини системи.

Діаграма прецедентів



Прямокутною рамкою позначено систему.

Операції з сегментами аудіозапису.

- Копіювання
- Вставка
- Вирізання
- Деформація

Робота з декількома звуковими доріжками. Дана функція дозволяє користувачеві працювати з кількома звуковими доріжками.

Кодування в поширених форматах.

- ogg
- flac

- mp3

Можна відзначити, що є саме чотири головних сценаріїв використання: Представити у WAVE-формі, Додати аудіо сегмент на звукову доріжку, Зберегти аудіо проект, Редагувати аудіо сегмент. У свою чергу сценарій Зберегти аудіо проект, розширюють ще 3 сценарії: Зберегти аудіо проєкт у форматі .ogg, Зберегти аудіо проект у форматі .flac, Зберегти аудіо проєкт у форматі .mp3. Крім цього, ще можна побачити, що сценарій Редагувати аудіо сегмент містить в собі 4 різних сценаріїв редагування аудіо, а саме: Копіювати аудіо сегмент, Вставити аудіо сегмент, Вирізати аудіо сегмент, Деформувати аудіо сегмент.

Оберемо 3 основні прецеденти для аудіо редактора та опишемо їх:

1. Представити у WAVE-формі

Опис: Користувач отримує графічне представлення аудіофайлу для аналізу та подальшого редагування.

Передумови

- Аудіофайл завантажено у програму.
- Формат файлу підтримується системою (наприклад, .wav, .mp3, .flac).

Актори

- Основний: Користувач.
- Вторинний: Аудіообробник (внутрішній модуль, що розраховує WAVEформу).

Сценарій

1. Користувач завантажує аудіофайл через інтерфейс програми.
2. Система перевіряє файл на підтримуваний формат.
3. Система декодує аудіо та аналізує амплітуду по часовій шкалі.
4. Графічний модуль відображає WAVE-форму.
5. Користувач отримує доступ до масштабування, прокручування та виділення фрагментів.

Результат

- WAVE-форма аудіофайлу відображена на екрані.
- Користувач може почати редагування.

Виключні ситуації:

- Файл не підтримується (виводиться повідомлення про помилку).
- Помилка при завантаженні файлу (запит на повторну спробу).

2. Редагувати аудіо сегмент

Опис: Користувач виконує операції редагування, такі як копіювання, вставка, вирізання або деформація виділеного сегмента аудіо. Передумови

- WAVE-форма аудіо завантажена та відображена.
- Користувач виділив сегмент для редагування. Актори
- Основний: Користувач.
- Вторинний: Модуль редагування аудіо (внутрішній механізм). Сценарій

1. Користувач виділяє сегмент аудіо.
2. Вибирає одну з наступних дій:
    - Копіювання: Сегмент додається у буфер. ○ Вставка: Сегмент із буфера вставляється у вибране місце.
    - Вирізання: Сегмент видаляється з WAVE-форми та додається у буфер.
    - Деформація: Система виводить параметри (швидкість, тональність, реверс).
3. Система застосовує зміни до сегмента та оновлює WAVE-форму. Результат

- Виділений сегмент змінений відповідно до вибраної дії.
- Графічне представлення оновлено.

Виключні ситуації:

- Виділений сегмент виходить за межі доступного аудіо.
- Помилка при застосуванні деформації (невірно задані параметри).

3. Зберегти аудіо проект

Опис: Користувач зберігає редагований проект у вибраному форматі та, за необхідності, структуру проекту для подальшого редагування. Передумови

- Є доступний для збереження аудіопроєкт.

- Користувач вибрав місце збереження та формат. Актори
- Основний: Користувач.
- Вторинний: Модуль кодування файлів.

Сценарій

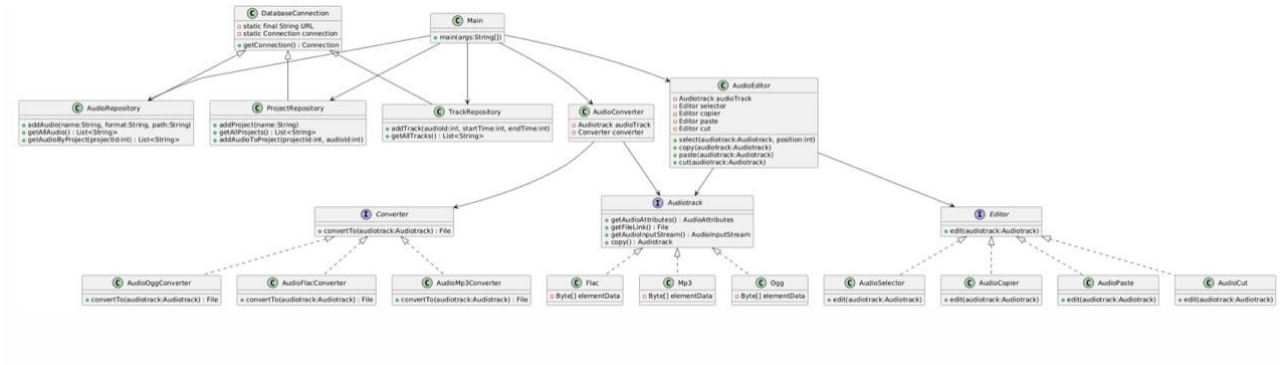
1. Користувач натискає кнопку "Зберегти проект".
2. У вікні вибору формату користувач обирає:
    - Формат файлу (.ogg, .flac, .mp3).
    - Папку для збереження.
3. Система кодує аудіопроєкт у вибраний формат.
4. Паралельно система створює файл проекту, що зберігає мета-дані для подальшого редагування.
5. Збережений проект доступний для відкриття в програмі. Результат

- Аудіопроєкт успішно збережений у вибраному форматі.
- Файл проекту (з мета-даними) створено.

Виключні ситуації:

- Помилка доступу до місця збереження (запит на зміну шляху).
- Вибраний формат не підтримується (повідомлення про помилку).
- Переривання збереження через нестачу місця на диску.

Діаграма класів



Загальна структура

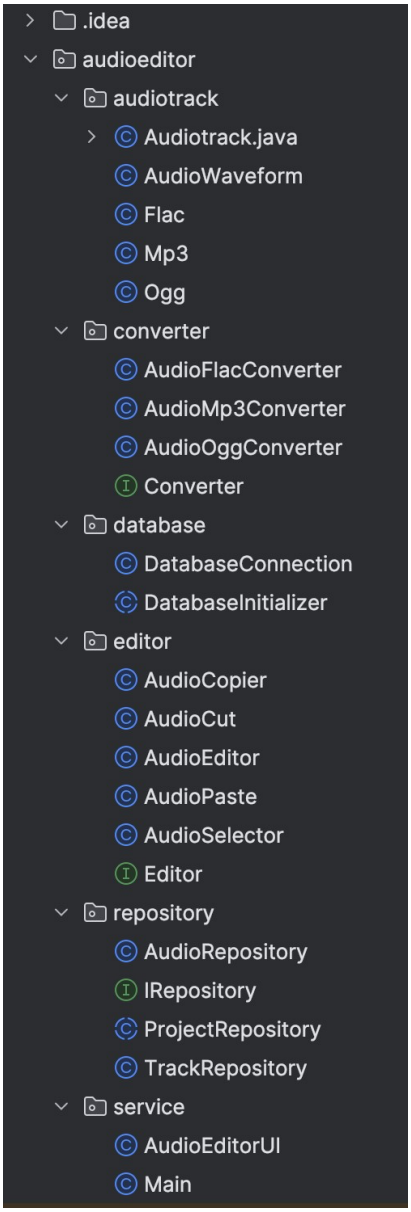
Діаграма класів демонструє модульний підхід до розробки:

1. Дані зберігаються в базі: Використовуються репозиторії.

2. Робота з аудіо: Передбачає редагування, виведення хвильової форми, та копіювання.
3. Конвертація аудіо: Легко розширюється шляхом додавання нових реалізацій Converter.

Такий дизайн полегшує підтримку та масштабування системи.

Основні класи та структура репозиторію



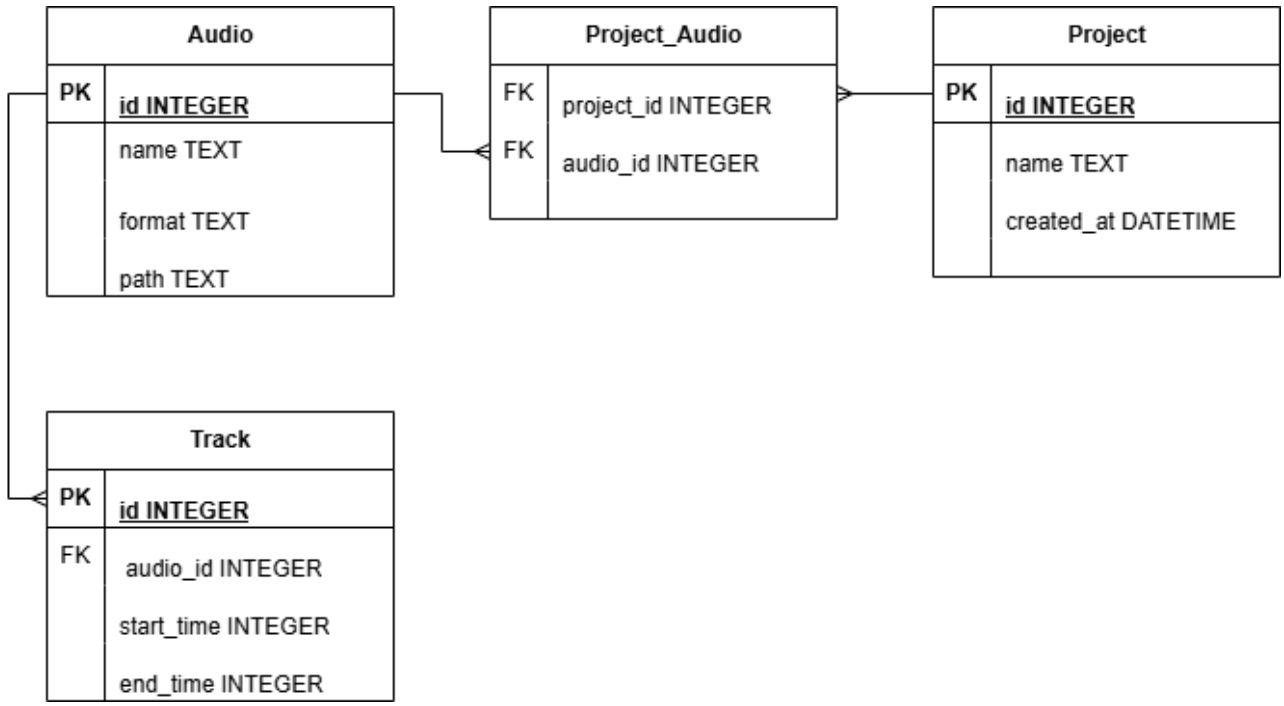
Проєкт аудіоредактора має чітко організовану модульну структуру, яка розділена на пакети відповідно до їх функціональності. Це дозволяє легко підтримувати, масштабувати та розширювати функціонал.

Основні пакети проєкту:

1. audiotrack — цей пакет представляє базовий функціонал для роботи з аудіотреками. Тут реалізовані класи для представлення аудіо у різних форматах (MP3, OGG, FLAC), а також клас для обробки аудіотреків і візуалізації їх хвильової форми.
2. converter — відповідає за конвертацію аудіофайлів між різними форматами. Завдяки інтерфейсу Converter, реалізовані класи для роботи з форматами MP3, OGG та FLAC, що дозволяє легко додавати підтримку нових форматів у майбутньому.
3. database — забезпечує роботу з базою даних. Класи в цьому пакеті відповідають за створення, ініціалізацію та управління з'єднанням з базою даних.
4. editor — реалізує функціонал для редагування аудіо. Через використання інтерфейсу Editor реалізовані різноманітні дії, такі як копіювання, вирізання, вставка та виділення частин аудіотреків.
5. repository — містить репозиторії для управління даними про аудіофайли, проєкти та треки. Цей пакет відповідає за збереження та отримання інформації з бази даних.
6. service — пакет, що відповідає за взаємодію з користувачем. Тут розташовані класи для роботи з графічним інтерфейсом користувача (GUI) та головний клас, який запускає застосунок.

Структура проєкту побудована таким чином, щоб чітко розподілити обов'язки між класами та пакетами. Це сприяє легкому внесенню змін і розширенню функціональності без порушення існуючого коду. Проєкт орієнтований на обробку аудіо, підтримку різних форматів і надання користувачеві зручних інструментів для редагування та управління аудіофайлами.

Структура бази даних



Структура бази даних побудована для зберігання даних, необхідних для роботи аудіоредактора. Вона складається з чотирьох основних таблиць: Audio, Track, Project, Project_Audio. Кожна таблиця має своє призначення та логічно взаємопов'язана з іншими. Розглянемо структуру детальніше:

1. Таблиця Audio

Зберігає інформацію про аудіофайли.

- **Колонки:**
  - id (INTEGER, PK): Унікальний ідентифікатор аудіофайлу.
  - name (TEXT): Назва аудіофайлу. ○ format (TEXT): Формат аудіофайлу (наприклад, MP3, OGG, FLAC).
  - path (TEXT): Шлях до файлу в файловій системі.

Ця таблиця містить базову інформацію про кожен аудіофайл, який доступний у проєкті.

2. Таблиця Track

Зберігає інформацію про треки, які є частинами аудіофайлів.

- **Колонки:**
    - id (INTEGER, PK): Унікальний ідентифікатор треку.
    - audio_id (INTEGER, FK): Зовнішній ключ, що посилається на аудіофайл у таблиці Audio.
    - start_time (INTEGER): Час початку треку (у секундах або мілісекундах).
    - end_time (INTEGER): Час завершення треку (у секундах або мілісекундах).

Ця таблиця дозволяє працювати з частинами аудіофайлів, виділяючи певні сегменти.

3. Таблиця Project

Зберігає інформацію про проєкти користувача.

- **Колонки:**
    - id (INTEGER, PK): Унікальний ідентифікатор проєкту.
    - name (TEXT): Назва проєкту.
    - created_at (DATETIME): Дата та час створення проєкту.

Ця таблиця представляє проєкти, в яких користувач може об'єднувати аудіофайли та працювати з ними.

4. Таблиця Project_Audio

Забезпечує зв'язок між таблицями Project та Audio.

- **Колонки:**
    - project_id (INTEGER, FK): Зовнішній ключ, що посилається на проєкт у таблиці Project.

- audio_id (INTEGER, FK): Зовнішній ключ, що посилається на аудіофайл у таблиці Audio.

Ця таблиця реалізує зв'язок "багато-до-багатьох" між проєктами та аудіофайлами. Один проєкт може містити кілька аудіофайлів, і один аудіофайл може бути частиною кількох проєктів.

Зв’язки між таблицями:

1. **Audio ↔ Track**: Один аудіофайл (Audio) може мати кілька треків (Track), але кожен трек належить тільки одному аудіофайлу.
2. **Project ↔ Project_Audio ↔ Audio**: Реалізується зв’язок "багато-добагатьох". Один проєкт може містити кілька аудіофайлів, а аудіофайл може належати до кількох проєктів.

Поточна структура бази даних забезпечує гнучке управління даними аудіоредактора. Вона дозволяє зберігати інформацію про аудіофайли, розбивати їх на сегменти (треки), організовувати файли у проєкти та підтримувати зв’язки між проєктами й аудіофайлами. Завдяки чітким зв’язкам та використанню зовнішніх ключів, структура є легкою для розширення та інтеграції з іншими модулями проєкту.

Висновок

У ході виконання першої лабораторної роботи на тему створення аудіоредактора ми ознайомилися з основними принципами побудови та проектування програмних систем, використовуючи схеми прецедентів, діаграми класів і структуру бази даних. Робота дозволила отримати теоретичне розуміння та практичний досвід у моделюванні ключових компонентів системи на основі обраного завдання.

На початку ми розглянули короткі теоретичні відомості, що дало змогу зрозуміти основні принципи побудови програмних систем, а також значення діаграм прецедентів, які описують взаємодію користувачів із системою. Після

цього було створено схему прецедентів, яка відображала основні функціональні можливості аудіоредактора, такі як завантаження аудіо, редагування та експорт у різні формати.

Наступним етапом була побудова діаграми класів для реалізованої частини системи. Діаграма класів допомогла сформувати базову структуру системи, що включає основні класи, їх атрибути та методи, а також зв’язки між ними. Це дало можливість чітко визначити відповідальність кожного компонента.

Було обрано три прецеденти, які описували базові сценарії взаємодії з аудіоредактором, а саме: завантаження аудіофайлу, редагування сегменту (копіювання, вирізання та вставка) і збереження результату в бажаному форматі. Кожен із прецедентів було детально описано, що дало змогу краще зрозуміти процеси, що відбуваються всередині системи.

Ми також розробили основні класи системи та структуру бази даних для зберігання інформації про аудіофайли та виконані операції. Це стало основою для подальшого використання шаблону Репозиторію, який забезпечує ефективну взаємодію з базою даних та ізоляцію рівня збереження даних від бізнес-логіки системи.

Таким чином, перша лабораторна робота забезпечила початкову структуризацію проєкту аудіоредактора, дала змогу закласти основу для подальшої реалізації функціоналу та дала чітке розуміння ключових етапів проектування програмної системи.

Репозиторій: https://github.com/milxxfa/trpz-BikovskaSasha-IA-31/tree/main

1. Що таке UML?
UML (Unified Modeling Language) – це стандартизована графічна мова для візуалізації, проєктування та документування програмних систем.

2. Що таке діаграма класів UML?
Діаграма класів UML – це статична структурна діаграма, що описує структуру системи, показуючи її класи, атрибути, методи та зв'язки між ними.

3. Які діаграми UML називають канонічними?
"Канонічними" (або основними) діаграмами UML зазвичай називають 5 діаграм з UML 1.x: діаграма варіантів використання, діаграма класів, діаграма послідовностей, діаграма станів та діаграма діяльності.

4. Що таке діаграма варіантів використання?
Діаграма варіантів використання – це поведінкова діаграма, яка моделює взаємодію між "акторами" (користувачами або іншими системами) та системою через "варіанти використання" (функції).

5. Що таке варіант використання?
Варіант використання – це опис послідовності дій (функції), яку система виконує у відповідь на запит актора для досягнення певної мети.

6. Які відношення можуть бути відображені на діаграмі використання?
На діаграмі використання можуть бути відображені такі відношення: асоціація (між актором та варіантом), включення (<<include>>), розширення (<<extend>>) та узагальнення (наслідування).

7. Що таке сценарій?
Сценарій – це конкретний приклад виконання варіанту використання. Один варіант використання може мати декілька сценаріїв (успішна оплата, помилка картки).

8. Що таке діаграма класів?
Діаграма класів – це статична діаграма, що показує класи системи, їх атрибути, методи та зв'язки.

9. Які зв’язки між класами ви знаєте?
Існують такі зв’язки між класами: асоціація, агрегація (ціле має частину, але частини незалежні), композиція (ціле складається з частин, частини не існують окремо), узагальнення (наслідування), реалізація (клас реалізує інтерфейс) та залежність.

10. Чим відрізняється композиція від агрегації?
Композиція відрізняється від агрегації "часом життя" компонентів. При агрегації ("має") частини можуть існувати окремо від цілого. При композиції ("складається з") частини не можуть існувати без цілого.

11. Чим відрізняється зв'язки типу агрегації від зв'язків композиції на діаграмах класів?
На діаграмах класів зв'язки агрегації та композиції відрізняються візуально: агрегація позначається незафарбованим ромбом, а композиція – зафарбованим ромбом.

12. Що являють собою нормальні форми баз даних?
Нормальні форми баз даних – це набір правил до структури реляційної бази даних, які використовуються для мінімізації надмірності даних та усунення аномалій.

13. Що таке фізична модель бази даних? Логічна?
Логічна модель бази даних – це абстрактне представлення даних (таблиці, колонки, зв'язки), що описує, що зберігається. Фізична модель – це конкретна реалізація логічної моделі в обраній СУБД, що описує, як дані зберігаються.

14. Який взаємозв'язок між таблицями БД та програмними класами?
Взаємозв'язок між таблицями БД та програмними класами найчастіше реалізується через ORM (Object-Relational Mapping): таблиця відповідає класу, колонки – полям класу, а один рядок – одному екземпляру класу.