КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра теорії та технології програмування

Кваліфікаційна робота

на здобуття ступеня бакалавра

за спеціальністю 122 Комп'ютерні науки

на тему:

РОЗРОБКА ІНТЕРАКТИВНОЇ НАВЧАЛЬНОЇ СИСТЕМИ У СФЕРІ ПРОЄКТУВАННЯ БАЗ ДАНИХ

Виконав студент 4-го курсу
Савицький Максим Віталійович

(підпис)

Науковий керівник:
доцент, кандидат технічних наук
Ткаченко Олексій Миколайович

(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент

(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«01» червня 2022 р.,
протокол № 10
Завідувач кафедри
М. С. Нікітченко

(підпис)

Київ – 2022

РЕФЕРАТ

Обсяг роботи 59 сторінок, 17 ілюстрацій, 14 джерел посилань.

WEB-ПРОЄКТ, ДИСТАНЦІЙНЕ НАВЧАННЯ, МОВА SQL, БАЗИ ДАНИХ, SQL-ЗАПИТ, НАВЧАЛЬНА СИСТЕМА, СУБД.

Об’єктом роботи є процес навчального характеру у сфері проєктування баз даних. Предметом роботи є інформаційні системи підтримки навчального процесу.

Метою роботи є розробка інтерактивної навчальної системи в браузері в якій можна отримати навички у сфері проєктування баз даних.

Методи розроблення: система з інтерактивними курсами, розробка програмного продукту на основі ітераційної моделі.

Інструменти розроблення: редактор вихідного коду Visual Studio Code, мова програмування JavaScript версії ES6, мова розмітки сторінок HTML5/CSS3, об'єктна модель документа DOM.

Результати роботи: розроблено інтерактивну навчальну систему, структуровану по курсам та темами до них.

Результат роботи можуть використовувати усі користувачі, які бажають покращити свої навички у сфері проєктування баз даних або отримати інформацію про розуміння принципів складання SQL запитів на зміну та вибірку полів у базах даних. Використання навчальної системи дозволяє користувачеві отримати структуровану за темами інформацію про проєктування баз даних та виконати практичні завдання для закріплення теоретичного матеріалу у вигляді тестів, питань для самоперевірки та спробувати скласти SQL запити різних типів.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

МВОП - масові відкриті освітні платформи;

SQL - абревіатура від Structured Query Language, структурована мова запитів;

SCORM - абревіатура від Sharable Content Object Reference Model, модель посилань на спільно використовувані об'єкти вмісту;

БД - база даних;

DOM - абревіатура від Document Object Model, об'єктна модель документу;

СУБД - система управління базами даних;

CSHARP - мова програмування C#;

ISO - абревіатура від International Organization for Standardization, міжнародна організація зі стандартизації;

DDL - абревіатура від Data Definition Language, мова визначення даних;

DML - абревіатура від Data Manipulation Language, мова маніпулювання даними.

ВСТУП

Оцінка сучасного стану об’єкта розробки. Більшість сучасних сервісів та додатків, навчальних закладів, банків, магазинів, заводів та взагалі будь-які підприємства та державні установи мають системи зберігання даних. У багатьох з них використовуються реляційні системи управління баз даних, для яких SQL – це базова мова для роботи з даними.

Користувачі використовують системи зберігання даних для переведення даних в електронний вигляд та їх об'єднання, а також оперативного доступу до таких даних. Це суттєво дозволяє економити час та кошти на витрати. Робота з базами даних є найважливішими навичками в роботі з комп'ютером, а фахівці цієї галузі стають все більш затребуваними.

Актуальність роботи та підстави для її виконання. Впровадження онлайн-платформ, що дозволяють навчатися дистанційно, допомагає вирішити низку проблем, пов'язаних безпосередньо як з здобуттям основної освіти, так і з реальною професійною діяльністю. Онлайн-курси не забирають у людини зайвого часу. Студент може займатися освоєнням нової спеціальності паралельно з навчанням у класичному університеті. Фахівець, у рамках підвищення кваліфікації, може проходити онлайн-курс після роботи — вдома, у затишній та спокійній обстановці.

Аналіз масових відкритих освітніх платформ (МВОП), що надають курси для освоєння технологій баз даних, показав, що більшість з них дає лише загальне уявлення про тему, але не дозволяє повноцінно сформувати навички, які фахівець зможе потім застосовувати у своїй професійній діяльності. Аудиторію таких ресурсів насамперед складають студенти вищих навчальних закладів. Курси в МВОП зазвичай не мають тренажерів для тренування практичних навичок написання SQL-запитів з достатньою кількістю завдань.

У той самий час очевидно, що запит на ресурси з освоєння SQL є, зокрема у сфері підвищення кваліфікації програмістів. [4]

Мета й завдання роботи. Метою роботи є розробка інтерактивної навчальної системи, в якій буде розділений на курси і структурований за темами матеріал для отримання навичок у сфері проєктування баз даних. Для досягнення цієї мети було поставлені наступні завдання:

- Дослідити наявні навчальні системи та курси, що використовуються у сфері проєктування баз даних.
- Дослідити та зібрати матеріали, необхідні для створення інтерактивних курсів.
- Структурувати зібрану інформацію, продумати логіку та послідовність теоретичного та практичного матеріалу в курсах.
- Розробити веб-застосунок з простим та зрозумілим для користувача інтерфейсом.
- Інтегрувати матеріал в навчальну систему за продуманою послідовністю та структурою.

Об’єкт, методи й засоби розроблення. Об’єктом роботи є процес учбового характеру у сфері проєктування баз даних. Предметом роботи є інтерактивна навчальна система, яка є ефективною для отримання знань та корисних навичок у даній сфері. Перед початком виконання роботи було проведено збір та структурування матеріалу необхідного для інтегрування в навчальну систему.

Під час розробки системи з інтерактивними навчальними курсами було використано ітераційну модель. Особливість ітераційної моделі полягає в тому, що увесь життєвий цикл продукту розбитий на ряд окремих міні-циклів, замість однієї тривалої послідовності процесів. Базові стадії моделі життєвого циклу є складовою частиною кожного із таких циклів. Ці міні-цикли називаються ітераціями, тому і сама модель називається ітераційною.

Розробка окремого компонента системи відбувається у кожній з ітерацій, після чого цей компонент додається до раніше розробленого функціоналу.

Для початку робіт над продуктом ітераційна модель не передбачає повного обсягу вимог. Розробка програми може починатися з вимог до частини функціоналу, які можуть згодом доповнюватись та змінюватись. Щоб забезпечити створення нової версії продукту кожного циклу процес повторюється. [1]

Для створення інтерактивної навчальної системи було використано мову програмування JavaScript версії ES6 з використанням Visual Studio Code, мови розмітки та стилів HTML5, CSS3, а також була розроблена логіка дерева об'єктної моделі документа DOM.

JavaScript – популярна мова програмування, яку використовують переважно веб-браузери. Більшість функцій і додатків, які роблять інтернет незамінним для сучасного життя, закодовані в тій чи іншій формі JavaScript. [3]

Можливі сфери застосування. Розроблена інтерактивна навчальна система може використовуватись усіма користувачами, які бажають покращити свої навички у сфері проєктування баз даних або отримати інформацію про розуміння принципів складання SQL запитів на зміну та вибірку полів у базах даних. Використання навчальної системи дозволить користувачам отримати структуровану за темами інформацію про проєктування баз даних та можливість виконати практичні завдання для закріплення теоретичного матеріалу у вигляді тестів, питань для самоперевірки та спробувати скласти SQL запити різних типів.

РОЗДІЛ 1 ЗАГАЛЬНІ ВІДОМОСТІ ПРЕДМЕТНОЇ ОБЛАСТІ

У великих організаціях баз даних містять критично важливі елементи записів, які мають складні логічні зв'язки з величезною кількістю інших наборів даних, кількість яких зростає разом з кількістю користувачів. В результаті організаціям необхідно активно контролювати, налаштовувати та покращувати свої бази даних для забезпечення високого рівня продуктивності.

На продуктивність баз даних впливають кілька ключових факторів, включаючи системні ресурси, робоче навантаження, пропускну здатність та оптимізацію. Без інструмента моніторингу бази даних важко точно оцінити, як кожен із цих факторів впливає на продуктивність баз даних та має довгостроковий вплив на продуктивність додатків та бізнесу.

1.1 Мова структурованих запитів

Мова структурованих запитів (SQL) - це стандартизована мова програмування, яка використовується для керування реляційними базами даних та виконання операцій над даними в них.

Створена ще в 1970-х роках, SQL використовується не тільки адміністраторами баз даних, а й розробниками, які пишуть сценарії інтеграції даних, а також аналітиками даних, які бажають створювати та виконувати аналітичні запити.

Мова SQL використовується для:

- зміни структури таблиць та індексів баз даних;
- додавання, оновлення та видалення рядків даних;
- Вилучення підмножин інформації з реляційних систем управління базами даних (РСУБД) - ця інформація може використовуватися для обробки транзакцій, аналітичних додатків та інших програм, що вимагають взаємодії з реляційною базою даних.

SQL-запити та інші операції приймають форму команд, записаних як тверджень, і об'єднуються у програми, які дозволяють користувачам додавати, змінювати чи отримувати дані з таблиць бази даних.

Основною одиницею баз даних являють собою таблиці, вони складаються з рядків і стовпців даних. В одній таблиці зберігаються записи, і кожен запис зберігається у рядку таблиці. Таблиці - це найбільш використовуваний тип об'єктів бази даних, або структур, які зберігають або посилаються на дані реляційної бази даних. До інших типів об'єктів бази даних належать індекси, які є таблицями пошуку і допомагають прискорити виконання функцій пошуку у базі даних. Кожен стовпець у таблиці відповідає категорії даних - наприклад, ім'я клієнта або адреса - а кожен рядок містить значення даних для стовпця, що перетинається.

Стандарт SQL та локальні розширення. Офіційний стандарт SQL був прийнятий Американським національним інститутом стандартів (ANSI) у 1986 році, а міжнародна організація зі стандартизації (ISO) прийняла стандарт у 1987 році. Нові версії стандарту SQL публікуються кожні кілька років. ISO/IEC 9075 - це стандарт ISO SQL, розроблений спільно ISO та міжнародною електротехнічною комісією.

Деякі версії SQL включають розширення стандартної мови для процедурного програмування та інших функцій. Наприклад, Microsoft пропонує набір розширень під назвою Transact-SQL, а розширена версія стандарту від Oracle - Procedural Language for SQL. Комерційні постачальники пропонують власні розширення для диференціації своїх продуктів, надаючи клієнтам додаткові можливості та функції. В результаті різні варіанти розширеного SQL, запропоновані постачальниками, не сумісні в повному обсязі один з одним.

Синтаксис SQL. Синтаксис SQL - набір правил написання та форматування операторів SQL, аналогічний іншим мовам програмування.

SQL, по суті, є мовою програмування, призначеною для доступу, зміни та вилучення інформації з реляційних баз даних. Як мова програмування, SQL має команди та синтаксис для введення цих команд. Команди SQL поділяються на кілька різних типів.

Команди Data Definition Language (DDL) називають командами визначення даних, оскільки вони використовуються для визначення таблиць даних.

Команди Data Manipulation Language (DML) використовуються для маніпулювання даними в існуючих таблицях шляхом додавання, зміни або видалення даних. На відміну від команд DDL, які визначають, як зберігаються дані, команди DML працюють у таблицях. команди DML працюють у таблицях, визначених командами DDL.

Мова запитів до даних складається з однієї команди SELECT, що використовується для отримання конкретних даних з таблиць. Цю команду іноді об'єднують із командами DML.

Команди мови керування даними використовуються для надання або скасування привілеїв доступу користувачів. Команди мови керування транзакціями використовуються для зміни стану деяких даних - наприклад, COMMIT транзакційних змін або ROLLBACK транзакційних змін. На рисунку 1 зображено приклад всіх типів команд.

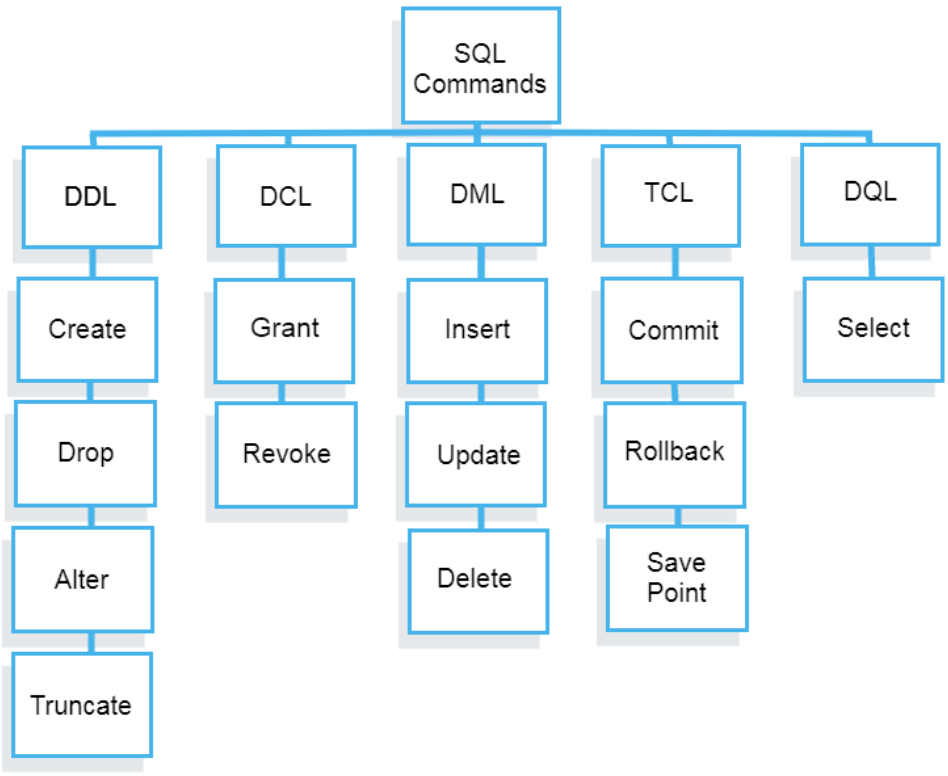


Рисунок 1 – Структура мови SQL

Більшість реалізацій SQL підтримують введення операторів у командному рядку, через графічний інтерфейс користувача, за допомогою програм SQL або через прикладні програмні інтерфейси для доступу до баз даних SQL за допомогою інших мов програмування. Більшість команд SQL використовуються разом із операторами зміни або зменшення обсягу даних, із якими працює оператор.

Інструменти SQL-on-Hadoop. Системи запитів SQL-on-Hadoop - це нове відгалуження SQL, яке дозволяє організаціям з архітектурою великих даних, побудованої на базі сховищ даних Hadoop, використовувати SQL як мову запитів і дає можливість спеціалістам з баз даних використовувати знайому мову запитів замість необхідності використовувати більш складних та менш знайомих мов - зокрема, середовище програмування MapReduce для розробки додатків пакетної обробки даних.

Понад десять інструментів SQL-on-Hadoop доступні у постачальників дистрибутивів Hadoop та інших виробників; багато хто з них мають відкритий вихідний код або комерційні версії. Крім того, механізм обробки даних Apache Spark, який часто використовується разом із Hadoop, включає модуль Spark SQL, який також підтримує програмування на основі SQL.

Не всі інструменти SQL-on-Hadoop підтримують усі функціональні можливості, які пропонуються в реляційних реалізаціях SQL. Однак інструменти SQL-on-Hadoop є регулярним компонентом розгортання Hadoop, оскільки компанії прагнуть залучити розробників та аналітиків даних із навичками SQL до програмування програм для роботи з великими даними.

Система безпеки SQL. SQL-сервери схильні до більшості тих же вразливостей, що й будь-які інші корпоративні програми, включаючи слабку автентифікацію, небезпечний дизайн, неправильну конфігурацію та інші проблеми безпеки застосунків. Однак SQL-ін'єкції, про які вперше було повідомлено у 1998 році, продовжують домінувати у питаннях безпеки SQL-систем. Атаки SQL-ін'єкцій зазвичай використовують слабкі місця в системах, де дані не скануються для видалення потенційно шкідливого коду, вбудованого або впровадженого в дані. [11]

Переваги та недоліки. До плюсів мови SQL відносять:

Стандартність. Міжнародні організації стандартизували SQL. Тобто мова SQL застосовується майже у всіх національних браузерах світу. Програмісти, які володіють цією мовою, можуть знайти роботу у всіх країнах.

Незалежність. Мова SQL не спрямовано на конкретну базу даних. Відповідно, його можна застосовувати з усіма існуючими СУБД. Переносити інформацію з одного сховища в інше можна з мінімальними коригуваннями.

Реляційна основа. SQL - це мова реляційних баз даних. Це і спричинило її поширення.

Програмний доступ до баз даних. Продукт застосовується як допоміжний інструмент при проектуванні додатків, яким потрібний доступ до баз даних. Користувачі можуть використовувати оператори SQL для інтерактивного та програмного доступу. Завдяки цьому при розробці програми розробник може протестувати його в різних режимах. Це допомагає уникнути помилок при подальшій роботі.

Динамічне розширення та зміна структури. Вбудовані інструменти дозволяють користувачам здійснювати маніпуляції зі структурою баз даних. Це забезпечує гнучкість мови в плані вимог предметної області, що змінюються.

Підтримка різноманітної архітектури. Продукт підтримує клієнт-сервіс та вважається найкращим інструментом для розробки додатків на цій платформі. SQL тут виступає посередник між клієнтською системою, яка взаємодіє з користувачем і безпосередньо серверною частиною, яка управляє базами даних.

Мова застосовується майже у всіх галузях діяльності, де потрібна обробка запитів користувачів. Це єдиний засіб програмування, стандартизований у всіх країнах. IT-фахівці застосовують його повсюдно. Розробники, що працюють із системою управління базами даних, створюють свою продукцію, застосовуючи SQL або SQL-інтерфейс.

До недоліків відносять:

Складність. Спочатку мова була задумана як робота кінцевого користувача. Проте на початку XXI століття складність мови SQL зросла. В 2022 році вона вважається професійним інструментом програмістів.

Можливість відступу від правил. Засіб підпадає під дію міжнародного стандарту ANSI SQL-92. Проте організації, які розробляють програмне забезпечення систем управління базами даних, дозволяють собі вносити зміну продукції, вироблену за допомогою SQL. До них входить і Microsoft. Цим вони відступають від загальноприйнятих стандартів. Через це кожної СУБД, розробленої цими компаніями, з'являються раніше невідомі діалекти мови. Вони можуть викликати складнощі у програмістів, які раніше не стикалися з ними.

Розробники як негативну сторону мови виділяють також складність роботи з ієрархічними структурами. Але це швидше питання досвіду програміста, ніж недолік мови SQL. [11]

1.2 Відмінності відомих СУБД

Існує безліч популярних систем управління баз даних SQL, включаючи SQLite, MySQL, Postgres, Oracle та Microsoft SQL Server. Усі вони підтримують загальний стандарт мови SQL, але кожна реалізація може відрізнятися додатковими можливостями та типами зберігання даних, які він підтримує.

Система управління базами даних - це комплекс програмних засобів, що дозволяють створити бази даних та керувати даними. Являє собою набір програм, що дозволяє організовувати, контролювати та адмініструвати бази даних. Більшість сайтів не можуть працювати без баз даних, тому Система управління базами даних використовуються практично в усіх бізнес процесах.

Основні функції систем управління баз даних:

- керування даними в оперативній пам'яті з використанням дискового кешу;
- керування даними у зовнішній пам'яті (на дисках);
- підтримка мов БД (мова визначення даних, мова маніпулювання даними).

- журналізація змін (збереження історії), резервне копіювання та відновлення бази даних після збоїв;

Однією з ознак класифікації СУБД є тим, що вона ґрунтується на будь-якій моделі даних. За моделлю даних СУБД бувають:

1. Ієрархічні. У цій моделі даних використовується уявлення БД як деревоподібної структури, що складається з даних різних рівнів.
2. Мережеві. Ця модель є розширенням ієрархічного підходу. Ієрархічна модель передбачає, що запис-нащадок може мати строго один кількість предків, тоді як у мережевий структурі нащадок може мати будь-яку кількість предків.
3. Реляційні. СУБД, орієнтовані на організацію даних як набір пов'язаних записів та атрибутів у двовимірній таблиці.
4. Об'єктно-орієнтовані. Для управління БД, що базуються на об'єктній моделі даних. Як правило ґрунтуються на об'єктно-орієнтованих мовах програмування.
5. Об'єктно-реляційні. Поєднує у собі концепції реляційної моделі з додатковими об'єктно-орієнтованими можливостями.

Сьогодні, як і раніше, найбільш популярними при створенні веб-застосунків та сервісів залишаються реляційні бази даних. Відмінності реляційної і нереляційної бази даних зображено на рисунку 2. Для керування реляційними базами даних використовується мова SQL. Спочатку SQL був інструментом роботи користувача з базою даних, проте згодом мова ускладнилася і стала швидше інструментом розробника, ніж кінцевого користувача. [9]

	Relational database	Non-relational database
QUERY LANGUAGE	SQL	SQL plus others
DATA TYPE	Structured	Unstructured
DATABASE NORMALIZATION	Yes	No
FORMAT	Tabular	Hierarchical
STORES RELATIONSHIPS OF VALUES	Yes	No
EXAMPLE OF DATA TYPE	Online transaction processing data	Molecular structure data
EXAMPLE OF PLATFORM	MySQL	MongoDB

Рисунок 2 – Характеристики реляційних та нереляційних баз даних [11]

Різні рейтинги найпопулярніших СУБД очолюють Oracle, MySQL, Microsoft SQL Server, PostgreSQL, SQLite, MongoDB.

MySQL. Вважається однією з найпоширеніших СУБД. MySQL - реляційна СУБД з відкритим вихідним кодом, головними плюсами якої є швидкість та гнучкість, яка забезпечена підтримкою великої кількості різних типів таблиць. Крім того, це надійна безкоштовна система із простим інтерфейсом та можливістю синхронізації з іншими базами даних. У сукупності ці фактори дозволяють використовувати MySQL як великим корпораціям, і невеликим компаніям.

Плюси MySQL:

- простий інтерфейс;
- підтримка різних типів таблиць (MyISAM, InnoDB, EXAMPLE та ін.);
- економне споживання ресурсів;
- синхронізація з іншими базами даних (Oracle, DB2 та інших.).

Мінуси MySQL:

- фрагментарне використання SQL;

- платна технічна підтримка (навіть для безкоштовних версій).

Microsoft SQL Server. Оптимальна для використання в операційних системах сімейства Windows, проте може працювати з Linux. Як випливає із назви, фірмова СУБД, розроблена Microsoft. В цілому, зберігає свою популярність, чимало через те, що продукти корпорації Microsoft використовуються багатьма компаніями. Система дозволяє синхронізуватися з іншими програмними продуктами компанії Microsoft, а також забезпечує надійний захист даних та простий інтерфейс, проте відрізняється високою вартістю ліцензії та підвищеним споживанням ресурсів.

Плюси Microsoft SQL Server:

- простий інтерфейс;
- синхронізація з іншими програмними продуктами Microsoft;
- гарний захист даних (шифрування, динамічне маскування та ін.);
- відмінна масштабованість (без втрати продуктивності обробляє кілька мільярдів записів).

Мінуси Microsoft SQL Server:

- висока ціна (стандартна ліцензія на один сервер коштуватиме 865 доларів);
- підвищене споживання ресурсів;
- обмежений функціонал для роботи з веб-програмами.

PostgreSQL. Популярна та безкоштовна система управління базами даних. Найбільше застосування даної системи - це керування БД веб-сайтів та різноманітних сервісів. PostgreSQL - об'єктно-реляційна СУБД, що дає їй деякі переваги над іншими безкоштовними СУБД, які є реляційними. PostgreSQL підійде для роботи з більшістю популярних платформ.

Плюси PostgreSQL:

- висока масштабованість;

- підтримка json (текстового формату обміну даних JavaScript);
- відповідність ACID (вимогам до системи, що забезпечують максимальну передбачуваність її роботи);
- можливість налаштування власного інтерфейсу;
- універсальність (підходить для використання більшості популярних платформ).

Мінуси PostgreSQL:

- підвищена витрата ресурсів;
- слабка технічна підтримка;
- проблеми із хостингом.

Oracle. Перша версія об'єктно-реляційної СУБД Oracle з'явилася наприкінці 70-х, і з того часу зарекомендувала себе як надійна, функціональна та практична. СУБД Oracle постійно розвивається і допрацьовується, спрощуючи установку та початкове налаштування та розширюючи функціонал. Однак істотним мінусом цієї СУБД є висока вартість ліцензії, тому вона використовується в основному великими компаніями та корпораціями, що працюють із величезними обсягами даних. [9]

Плюси Oracle:

- функціонал (Oracle містить grid framework та масу фішок, які в інших СУБД потрібно встановлювати додатково);
- відмінна масштабованість;
- надійність;
- можливість використання Oracle APEX для веб-додатків.

Мінуси Oracle:

- висока ціна (однокористувацька ліцензія коштує 350 доларів, процесорна – 17,5 тисяч);

- високе споживання системних ресурсів (часто перед установкою доводиться оновлювати обладнання);
- складні зміни (не кожен користувач впорається з використанням та обслуговуванням Oracle).

SQLite. Швидка та легка однофайлова СУБД, яка не має сервера та дозволяє зберігати всю базу локально на одному пристрої. Для роботи SQLite не потрібні сторонні бібліотеки або служби. Двигун SQLite - це не окремий процес, з яким взаємодіє програма, а бібліотека. Програма компонується з нею, і двигун служить складовою програми. Виклики функцій бібліотеки SQLite використовуються як протокол обміну. SQLite влаштована таким чином, що немає сервера. Це означає, що всі дані програмне забезпечення зберігає на одному пристрої. СУБД вбудовується у додаток та працює як його складова частина. Якщо встановити на комп'ютер програму, що використовує SQLite, то база даних теж зберігатиметься на ньому. Формат бази - один текстовий файл, який можна прочитати на будь-якій платформі. Такий підхід підвищує продуктивність та швидкість роботи.

Плюси SQLite:

- висока швидкість (компоненти СУБД вбудовані в застосунок і викликаються у тому же процесі);
- зберігання даних в одному файлі;
- надійність;
- доступність;
- кросплатформеність;
- автономність (система незалежна від стороннього програмного забезпечення, бібліотек або фреймворків, додаткові компоненти не потрібні).

Мінуси SQLite:

- обмежена підтримка типів даних;

- обмеження у застосуванні;
- платна технічна підтримка.

MongoDB. MongoDB - СУБД, яка працює з документоорієнтованою моделлю даних. MongoDB не потребує таблиці, схеми або окремої мови запитів, на відміну від реляційних СУБД. MongoDB не використовує схеми, як це роблять реляційні бази даних, що підвищує продуктивність усієї системи. Інформація зберігається у вигляді документів чи колекцій.

Плюси MongoDB:

- легкість масштабування;
- доступність (безкоштовна і може працювати на Linux);
- можливість вибирати, який рівень узгодженості потрібен (залежно від цінності даних).

Мінуси MongoDB:

- більший розмір даних (у кожному документі зберігаються імена полів);
- менша гнучкість при складанні запитів (наприклад, немає JOINів);
- підтримуються тільки певні атомарні операції на рівні одного документа (немає підтримки транзакцій).

1.3 Проєктування баз даних

Процес створення схеми бази даних та визначення необхідних обмежень цілісності називають проєктуванням баз даних. Забезпечення зберігання у БД всієї необхідної інформації, забезпечення можливості отримання даних за всіма необхідними запитами, скорочення надмірності та дублювання даних та забезпечення цілісності бази даних є основними завданнями проєктування баз даних.

Проектування баз даних складається із трьох основних фаз: концептуального, логічного і фізичного моделювання.

При концептуальному моделюванні відбувається побудова інформаційної моделі найвищого рівня абстракції, тобто семантичної моделі предметної області. Найчастіше концептуальна модель бази даних включає в себе опис інформаційних об'єктів або понять предметної галузі та зв'язків між ними, а також опис обмежень цілісності, тобто вимог до допустимих значень даних та зв'язків між ними. Конкретний вид та зміст концептуальної моделі бази даних визначається обраним для цього формальним апаратом. Зазвичай використовуються графічні нотації, подібні до ER-діаграм.

Під час логічного моделювання уточнюються вимоги і розробляється модель бази даних. Перетворення концептуальної моделі на логічну модель, як правило, здійснюється за формальними правилами. Цей етап може бути значною мірою автоматизований. На етапі логічного проектування враховується специфіка конкретної моделі даних, але може враховуватися специфіка конкретної СУБД.

Під час фізичного моделювання створюється модель, оптимізована для конкретної програми та СУБД. Саме ця модель реалізується практично.

Процес проектування бази даних складається з наступних етапів:

1. Збір інформації;
2. Визначення сутностей;
3. Визначення атрибутів кожної сутності;
4. Визначення зв'язків між сутностями;
5. Нормалізація;
6. Перетворення до фізичної моделі;
7. Створення бази даних.

На етапі уточнення вимог необхідно точно визначити, як буде використовуватися база даних і яка інформація буде в ній зберігатися. Грубо кажучи, збирається якомога більше відомостей про те, що система повинна робити і чого не повинна. [10]

На етапі визначення сутностей необхідно визначити сутності, з яких буде складатися база даних.

Сутність - це об'єкт у базі даних, у якому зберігаються дані. У фізичній моделі сутність називається таблицею. Сутності складаються з атрибутів (стовпців таблиці) та записів (рядків у таблиці).

Зазвичай бази даних складаються з кількох основних сутностей, пов'язаних із великою кількістю підлеглих сутностей. Основні сутності називаються незалежними: вони залежать ні від будь-якої іншої сутності. Підлеглі сутності називаються залежними: щоб існувала одна з них, повинна існувати пов'язана з нею основна таблиця. Будь-яка таблиця має такі характеристики:

- в таблиці немає однакових рядків;
- всі стовпці (атрибути) у таблиці повинні мати різні імена;
- елементи в межах однієї колонки мають однаковий тип (рядок, число, дата);
- порядок прямування рядків у таблиці може бути довільним.

На цьому етапі необхідно виявити всі категорії інформації (сутності), які зберігатимуться у базі даних.

Визначення атрибутів. Атрибут є властивістю, що описує сутність. Атрибути часто бувають числом, датою чи текстом. Всі дані, що зберігаються в атрибуті, повинні мати однаковий тип і мати однакові властивості. У фізичній моделі атрибути називають стовпчиками. Після визначення сутності необхідно визначити всі атрибути цих сутностей.

Для кожного атрибуту визначається тип даних, їх розмір, допустимі значення та будь-які інші правила. До них відносяться правила обов'язковості заповнення, змінності та унікальності. Правило обов'язковості заповнення визначає, чи є атрибут обов'язковою частиною сутності. Якщо атрибут є необов'язковою частиною сутності, він може приймати NULL-значення,

інакше - ні. Також потрібно визначити, чи буде атрибут змінюватися і його унікальність. Значення деяких атрибутів не можуть бути змінені після створення запису. Якщо атрибут унікальний то його значення не може повторюватися.

Визначення ключа в базах даних. Ключем називається набір атрибутів, що однозначно визначає запис. Ключі діляться на два класи: прості та складові.

Простий ключ складається лише з одного атрибута. Наприклад, у базі даних «Паспорти громадян країни» номер паспорта буде простим ключем: адже немає двох паспортів з однаковим номером. Складовий ключ складається з кількох атрибутів. У тій самій базі даних «Паспорти громадян країни» може бути складовий ключ із такими атрибутами: прізвище, ім'я, по батькові, дата народження.

Можливий ключ являє собою будь-який набір атрибутів, що однозначно ідентифікують запис у таблиці. Можливий ключ може бути простим чи складним. Кожна сутність повинна мати принаймні один можливий ключ, хоча таких ключів може бути кілька. Жоден з атрибутів первинного ключа не може набувати невизначеного (NULL) значення. Можливий ключ називається також сурогатним.

Первинним ключем називається сукупність атрибутів, що однозначно ідентифікують запис у таблиці (сутності). Один із можливих ключів стає первинним ключем. На діаграмах первинні ключі часто зображуються вище за основний список атрибутів або виділяються спеціальними символами.

Будь-який можливий ключ, який не є первинним, називається альтернативним ключем. Сутність може мати кілька альтернативних ключів.

Зовнішнім ключем називається сукупність атрибутів, що посилаються на первинний чи альтернативний ключ іншої сутності. Якщо зовнішній ключ пов'язані з первинної сутністю, він може містити лише невизначені значення.

Якщо при цьому ключ є складовим, всі атрибути зовнішнього ключа повинні бути невизначеними. На діаграмах атрибути, що поєднуються у зовнішні ключі, позначаються спеціальними символами.

Ключі є логічними конструкціями, а чи не фізичними об'єктами. У реляційних базах даних передбачено механізми, що забезпечують збереження ключів.

Визначення зв'язків між сутностями. Реляційні бази даних дозволяють поєднувати інформацію, що належить різним сутностям.

Відношення сутностей - це ситуація, коли одна сутність посилається на первинний ключ другої сутності. Відносини визначаються у процесі проєктування баз даних. Для цього слід проаналізувати сутності та виявити логічні зв'язки, що існують між ними. Тип відношення визначає кількість записів сутності, пов'язаних із записом іншої сутності. Відносини поділяються на три основні типи:

- один-до-одного (1 : 1);
- один-до-багатьох (1 : ∞);
- багато-до-багатьох (∞ : ∞).

Один-до-одного. Кожному запису першої сутності відповідає лише один запис із другої сутності. А кожному запису другої сутності відповідає лише один запис із першої сутності.

Один-до-багатьох. Кожен запис першої сутності може відповідати кілька записів з другої сутності. Однак кожному запису другої сутності відповідає лише один запис із першої сутності.

Багато-до-багатьох. Кожен запис першої сутності може відповідати кілька записів з другої сутності. Однак і кожний запис другої сутності може відповідати кілька записів з першої сутності.

Приклад застосування усіх типів сутностей зображено на рисунку 3.

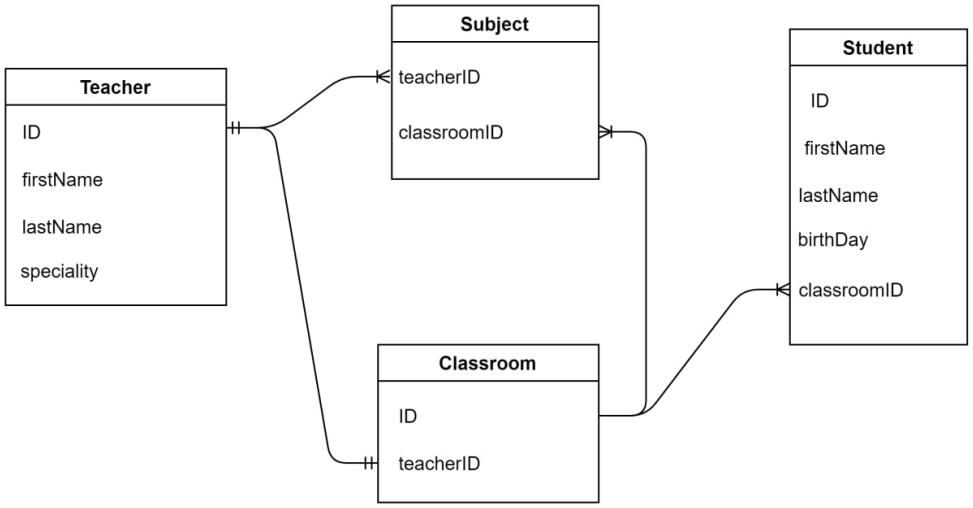


Рисунок 3 – Приклад діаграми усіх типів сутностей

За критерієм обов'язковості відносини поділяються на обов'язкові та необов'язкові. Обов'язкове ставлення означає, що з кожної записи з першої сутності обов'язково повинні бути присутні пов'язані записи на другий сутності. Необов'язкове відношення означає, що з записи з першої сутності може й не існувати записи на другий сутності.

Нормалізація. Нормалізацією називається процес видалення надлишкових даних із бази даних. Кожен елемент даних повинен зберігатися в основі в одному і тільки в одному екземплярі. Існує п'ять найпоширеніших форм нормалізації. Як правило, база даних наводиться до третьої нормальної форми.

У процесі нормалізації виконуються певні дії щодо видалення надлишкових даних. Нормалізація підвищує швидкодію, прискорює сортування та побудову індексу, зменшує кількість індексів на сутність, прискорює операції вставки та оновлення.

Нормалізована база даних зазвичай відрізняється більшою гнучкістю. При модифікації запитів або даних, що зберігаються, в нормалізовану базу

зазвичай доводиться вносити менше змін, а внесення змін має менше наслідків.

Перша нормальна форма. Щоб перетворити сутність на першу нормальну форму, слід виключити повторювані групи значень і домогтися, щоб кожен атрибут містив лише одне значення, списки значень не допускаються. Іншими словами, кожен атрибут, по суті, повинен зберігатися тільки в одному екземплярі.

Друга нормальна форма. Таблиця у другій нормальній формі містить ті дані, які до неї ставляться. Значення не ключових атрибутів сутності залежить від первинного ключа. Якщо точніше, то атрибути залежать від первинного ключа, від всього первинного ключа і тільки від первинного ключа. Для відповідності другий нормальній формі сутності повинні бути у першій нормальній формі.

Третя нормальна форма. У третій нормальній формі виключаються атрибути, які не залежать від всього ключа. Будь-яка сутність, що знаходиться в третій нормальній формі, знаходиться також і в другій. Це найпоширеніша форма бази даних.

Обмеження та збережені процедури. Обмеження - правила, дотримання за якими стежить система управління бази даних. Обмеження визначають множину значень, які можна вводити в стовпець або стовпчики.

Процедури, що зберігаються - це попередньо відкомпільовані процедури, що зберігаються в базі даних. Збережені процедури можна використовувати для визначення правил, з допомогою яких можна здійснювати складніші обчислення, ніж з допомогою лише обмежень.

Збережені процедури можуть містити логіку виконання програми, а також запити до бази даних. Вони можуть приймати параметри та повертати результати у вигляді таблиць або одиночних значень. Збережені процедури схожі на звичайні процедури або функції будь-якої програми.

Процедури, що зберігаються, знаходяться в базі даних і виконуються на сервері бази даних. Зазвичай, вони швидше операторів SQL, оскільки зберігаються у компільованому вигляді.

Цілісність даних. Організувавши дані в таблиці та визначивши зв'язки між ними, можна вважати, що була створена модель, яка правильним чином відображає бізнес-середовище. Тепер необхідно забезпечити, щоб дані, що вводяться до бази, давали правильне уявлення про стан. Іншими словами, необхідно забезпечити виконання правил та підтримку цілісності бази даних.

Коректність даних у реляційних базах забезпечується набором правил. Правила цілісності даних поділяються на чотири категорії.

Цілісність сутностей - кожен запис сутності повинен мати унікальний ідентифікатор і містити дані. Адже треба розрізняти всі записи в базі даних.

Цілісність атрибутів - кожен атрибут набуває лише допустимих значень. Наприклад, сума покупки, безумовно, не може бути меншою за нуль.

Посилальна цілісність - набір правил, що забезпечують логічну узгодженість первинних та зовнішніх ключів при вставці, оновленні та видаленні записів. Цілісність посилань забезпечує, щоб для кожного зовнішнього ключа існував відповідний первинний ключ.

Користувальницькі правила цілісності - будь-які правила цілісності, що не належать до жодної з перерахованих категорій.

Тригери. Тригер - це аналог процедури, що зберігається, який викликається автоматично при зміні даних у таблиці. Тригери є потужним механізмом підтримки цілісності бази даних. Тригери викликаються до або після зміни даних у таблиці. За допомогою тригерів э можливість не тільки відмінити ці зміни, але й змінити дані у будь-якій іншій таблиці.

Фізична модель. Наступним кроком, після створення логічної моделі, є побудова фізичної моделі. Фізична модель – це практична реалізація бази

даних. Фізична модель визначає всі об'єкти, які вам належить реалізувати. При переході від логічної моделі до фізичної сутності перетворюються на таблиці, а атрибути на стовпці. Відносини між сутностями можна перетворити на таблиці або залишити як зовнішні ключі. Первинні ключі перетворюються на обмеження первинних ключів. Можливі ключі - обмеження унікальності.

Денормалізація. Денормалізація - це навмисне зміна структури бази даних, що порушує правила нормальних форм. Зазвичай це робиться з метою покращення продуктивності бази даних. Надмірна нормалізація бази даних може призвести до того, що при кожному видаленні даних доведеться звертатися до кількох таблиць. Зазвичай, у запиті повинні брати участь чотири таблиці або менше. Стандартними прийомами денормалізації є об'єднання кількох таблиць в одну, збереження однакових атрибутів у кількох таблицях, і навіть зберігання у таблиці зведених чи обчислюваних даних. [10]

РОЗДІЛ 2 АНАЛІЗ НАЯВНИХ НАВЧАЛЬНИХ СИСТЕМ ТА КУРСІВ

2.1 Платформа Ulearn

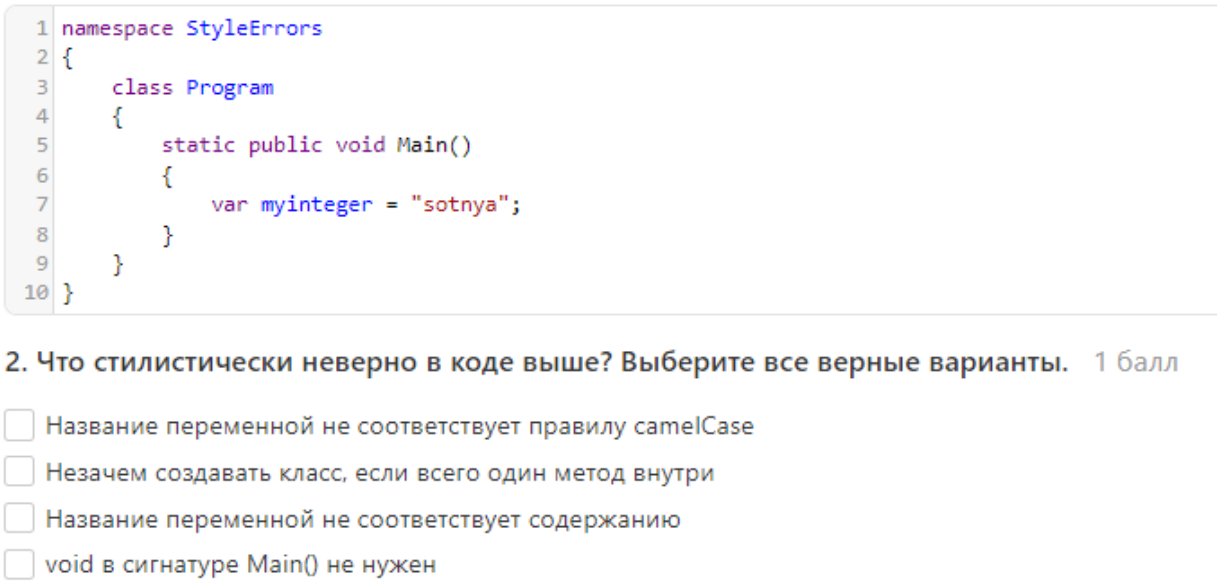
Дана навчальна система хоч і не присвячена сфері проєктування баз даних, але є одним з найкращих безкоштовних інтерактивних онлайн-курсів по вивченню мови програмування CSharp з нуля. Платформа Ulearn являється гарним прикладом того, як повинні виглядати і працювати інтерактивні навчальні системи. Нерідко автори багатьох онлайн-курсів ведуть простий переказ специфікації мови програмування, а також не розкривають те, що виходить за рамки документації даної сфери. Порівнюючи з іншими безкоштовними онлайн курсами, автори інтерактивної навчальної системи Ulearn добре розуміють те, як повинна виглядати ефективна структура навчальної системи. Основи програмування подані максимально просто, без зайвих технічних на ранніх етапах деталей, намагаються застосовувати правильні аналогії.

Серед недоліків можна зазначити тільки відсутність української мови та відсутність матеріалу про бази даних.

В курсі наявні такі інтерактивні матеріли та методи як:

- тести;
- фрагменти коду;
- вбудований в систему компілятор мови CSharp;
- завдання для самоперевірки;
- завдання з картками.

Приклад практичного завдання у форматі тесту навчальної платформи Ulearn зображено на рисунку 4. На ньому можна побачити фрагмент коду мови програмування CSharp, в звичному для розробника форматі та одного з тестових завдань до нього.



```
namespace StyleErrors
{
    class Program
    {
        static public void Main()
        {
            var myinteger = "sotnya";
        }
    }
}
```

2. Что стилистически неверно в коде выше? Выберите все верные варианты. 1 балл

- ☐ Название переменной не соответствует правилу camelCase
- ☐ Незачем создавать класс, если всего один метод внутри
- ☐ Название переменной не соответствует содержанию
- ☐ void в сигнатуре Main() не нужен

Рисунок 4 – Тестове завдання на платформі Ulearn [7]

Приклади інших практичних завдань з платформи Ulearn можна переглянути в додатку А.

Якщо перейти до навчальної платформи Ulearn та відкрити один з наявних курсів то можна побачити, що зліва буде знаходитись перелік списку усіх тем з курсу, розбитий за тематичними модулями - тижнями. Особливими значками у списку відзначені слайди із завданнями та тестами. Під відео з викладеним теоретичним матеріалом зазвичай будуть короткі нотатки, за якими можливо швидко зрозуміти зміст відео. Часто після відео пропонуються одне або кілька завдань чи тестів для закріплення матеріалу. Лекційний матеріал краще засвоюється після застосування на практиці. Виконувати ці завдання можна прямо в браузері, а спеціальна система перевірки та вбудований в систему компілятор одразу ж перевірить правильність виконання завдання. Після успішного виконання деяких завдань користувач зможе переглянути рішення інших учнів. Іноді вони бувають досить корисними. Велика частина завдань на платформі мають вбудовану

автоматичну перевірку та рекомендації від викладачів, які автоматично виводиться на екран користувача.

2.2 Платформа Sqlbolt

Інтерактивна навчальна система з вивчення SQL. Рекомендується для новачків та тих, хто бажає освіжити свої знання. Тут немає типового поділу на практику та теорію, інформація подається у форматі уроків. Уроки складаються з необхідної теорії з прикладами, а наприкінці пропонується кілька завдань із щойно прочитаного матеріалу, приклад одного з таких завдань зображено на рисунку 5. Вступних уроків 18, розбираються такі теми, як обмеження, join'и, вирази, агрегати та дії з таблицями та рядками. Наприкінці кожного теоретичного матеріалу наводиться тренажер для відпрацювання знань, отриманих вище.

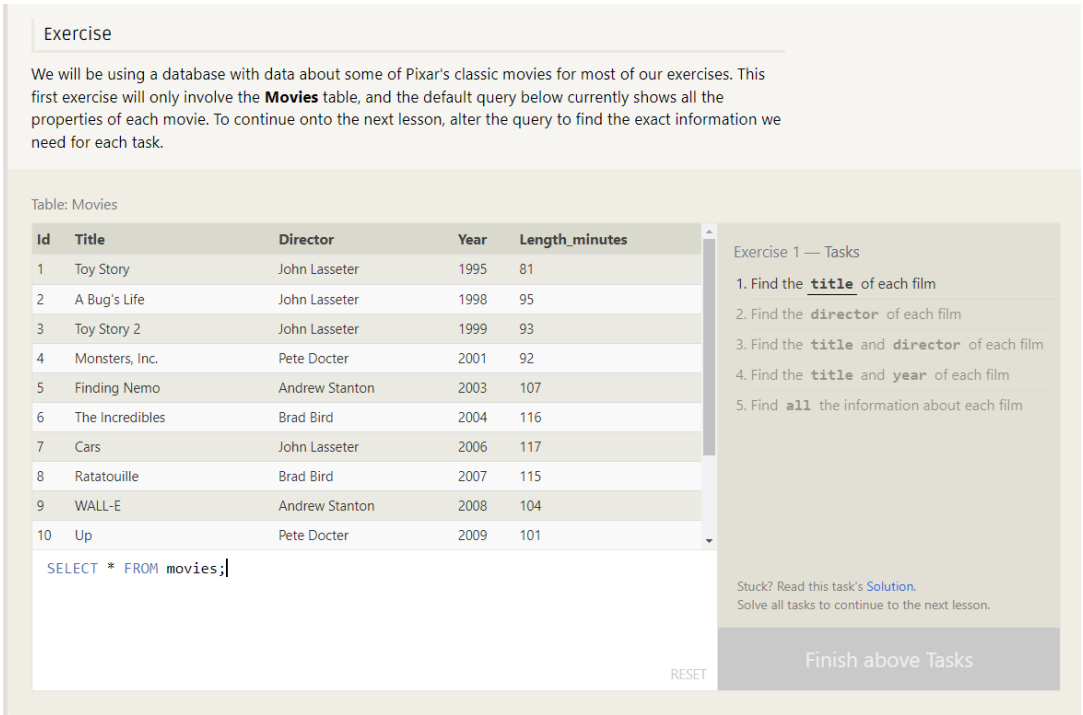


Рисунок 5 – Практичне завдання на плаформі SQLBolt [8]

SQLBolt поєднує в собі легко здійсненні інструкції, простий інтерфейс та інтерактивні вправи для навчання базових навичок SQL.

У той час як багато підручників засновані виключно на кодуванні, SQLBolt поєднує письмові пояснення з пробним кодуванням, щоб дати студентам більш чітке та міцне уявлення про SQL. SQLBolt являє собою щось середнє між підручниками, в яких тільки код, і надто технічними курсами. Ресурс рекомендується студентам, які віддають перевагу текстовим поясненням, підкріпленим стандартними вправами.

2.3 SQL-курс на платформі HackerRank

HackerRank - це соціальна платформа, яка пропонує завдання різної складності програмування. Запущено платформу в 2012 році. Зараз аудиторія проекту налічує близько півтора мільйона користувачів. Статистику за результатами рішень завдань своїми користувачами ресурс регулярно публікує рейтинг по 50 країнам. На рисунку 6 зображено один з таких рейтингів HackerRank.

Rank	Domain	Percent of Challenges Solved
1	Algorithms	39.5%
2	Java	9.3%
3	Data Structures	9.1%
4	C++	6.6%
5	Tutorials	6.5%
6	Mathematics	6.1%
7	Python	5.3%
8	SQL	4.6%
9	Shell	3.1%
10	Artificial Intelligence	2.9%
11	Functional Programming	2.5%
12	Databases	1.5%
13	Ruby	1.0%
14	Distributed Systems	1.0%
15	Security	0.9%
	Total	100.0%

Рисунок 6 - Найбільш популярні теми серед користувачів платформи [12]

Рейтинг HackerRank враховує завдання з 15 найпопулярніших дициплін. Найпопулярніші завдання стосуються різних алгоритмів з динамічним програмуванням, аналізом великих даних та іншими темами. Для вирішення таких завдань користувач може працювати з будь-якою програмною мовою. Крім алгоритмів у топ-3 популярних завдань входять завдання щодо структури даних та програмування на Java. Такі завдання виконуються 9,1% та 9,3% від усіх користувачів відповідно. Всі бали фахівців з різних країн аналізуються, після чого виводиться загальний бал для кожної країни з рейтингу. Максимальна кількість балів, яку може отримати країна, - 100.

Рівень складності завдань на HackerRank трохи вище, трапляються завдання, що виходять за межі базового програмування. Можна розвиватися у кількох областях, включаючи алгоритми, математику, SQL, функціональне програмування, AI та багато іншого. Також платформа надає функціонал корпоративного програмування.

В рамках платформи є секція для тренування навичок із SQL. Є підтримка 4 СУБД, збереження попередніх відповідей та система рейтингу. На рисунку 7 зображений тренажер для тренування навичок із SQL. HackerRank зручна платформа, де можна тренувати володіння мовою завдяки наявності редактора коду. Також є можливість не лише навчитися, а й підготуватись до інтерв'ю з майбутнім роботодавцем. Діє рейтингова система мотивації користувачів. [12]

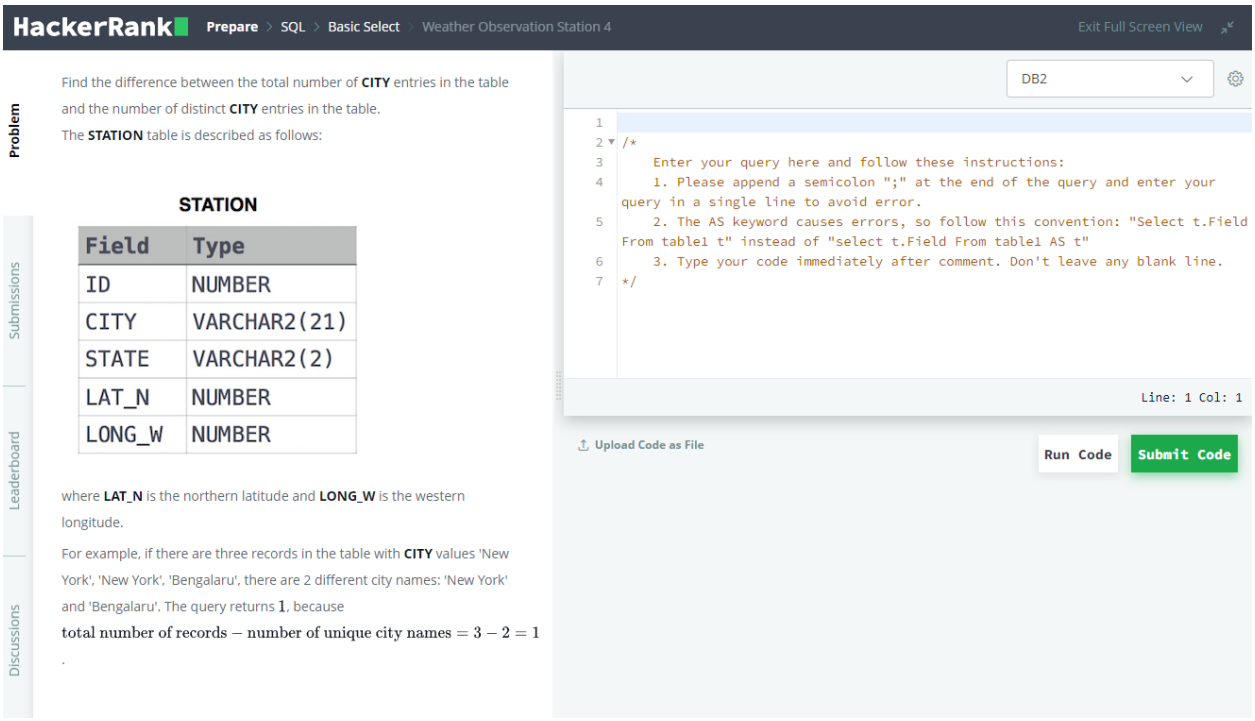


Рисунок 7 – Приклад завдання на платформі HackerRank [12]

SQL-завдання дещо відрізняються від своїх звичайних алгоритмів/структур даних. Всього на платформі 58 SQL-завдань. Вони вимагають від вас більш організованого мислення, структурування коду та перемикання між різними рівнями абстракції. Секція SQL-курсу на платформі HackerRank допомагає опанувати ці "повільні" навички.

2.4 Платформа SQLZoo

SQLZoo - це онлайн-платформа, що добре зарекомендувала себе (з 1999 року) для написання і виконання SQL-запитів до баз даних. Це означає, що ви можете побачити фактичний результат вашого запиту без необхідності скрупульозно перевіряти відповідність вашого запиту рішенню - важливий саме результат. Це важливо, тому що часто існує безліч підходів до складних питань, причому один із них не обов'язково є найкращим.

Система SQLZoo, розроблена компанією Andrew Cumming, призначена для реалізації таких функцій, як MySQL, SQL та SQLite. SQLZOO - це

безкоштовна програма із закритим вихідним кодом, що працює на багатьох платформах, в тому числі Web.

SQLZoo включає навчальні посібники та посилання на підтримку людей, що вивчають SQL. Система включає в себе:

- Інтерактивний доступ до кількох SQL-моделей;
- Приклади баз даних;
- Практичні завдання;
- Миттєвий зворотній зв'язок про успіхи спроб користувача.

На рисунку 8 можна побачити інтерфейс праткичного завдання на платформі SQLZoo.

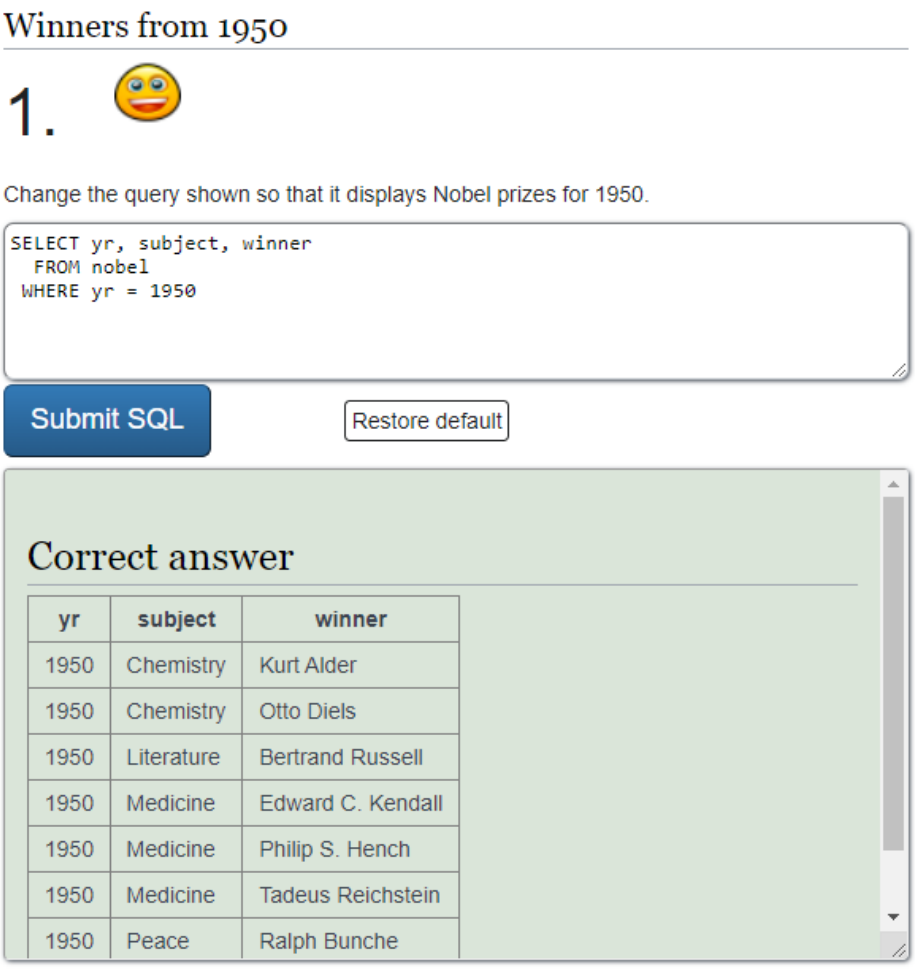


Рисунок 8 – Практичне завдання на платформі SQLZoo [13]

Система SQLZoo має освітній розділ "Assessments". Він містить складніші приклади, що дозволяють глибоко зануритися в базу даних на різних рівнях складності. На платформі є гарні завдання - Help Desk та Guest House, в яких є докладні діаграми, що пояснюють базу даних, а також кілька складніших завдань.

Написання якісних SQL-запитів - не така вже й проста справа, оскільки необхідно враховувати читабельність, швидкість, ефективність, надійність - все це має значення для бізнесу. Під час проходження курсу на платформі SQLZoo користувач зустріне такі питання як: що можна було б написати більш лаконічно? Як можна було б підвищити ефективність? Що б сталося, якщо деякі стовпці містили значення NULL?

Варто зазначити, що SQLZoo побудований за допомогою сервера MariaDB, що підтримує MySQL. Це означає, що деякі з технік, які зазвичай застосовують користувачі в інших СУБД не будуть працювати. Це не дуже зручно, але це хороший шанс попрактикуватися в інших прийомах, про які ви можете не думати при використанні того варіанта SQL, з яким регулярно працюєте.

РОЗДІЛ 3 СТРУКТУРА ТА ПОСЛІДОВНІСТЬ НАВЧАЛЬНОГО ПРОЦЕСУ

Перш ніж вивчати синтаксис SQL, важливо мати уявлення, що таке реляційна база даних. Реляційна база даних являє собою набір пов'язаних (двовимірних) таблиць. Кожна таблиця схожа на електронну таблицю Excel, з фіксованою кількістю іменованих стовпців (атрибути або властивості таблиці) і будь-якою кількістю рядків даних.

Наприклад, якби департамент автотранспорту мав базу даних, ви могли б знайти таблицю, яка містить усі відомі автомобілі, якими керують жителі вашого регіону. У таких таблицях можливо зберігати, наприклад, назву моделі, тип, кількість коліс та дверей кожного автомобілю.

У таких базах даних ви можете знайти додаткові зв'язані таблиці, що містять таку інформацію, як список всіх зареєстрованих водіїв у регіоні, типи водійських прав, які можуть бути видані, або навіть порушення правил керування для кожного водія.

Мета вивчення SQL - навчитися відповідати на конкретні питання про ці дані, наприклад, "Які типи автомобілів на дорогах мають менше чотирьох коліс?" або "Скільки моделей автомобілів виробляє компанія Tesla?", щоб допомогти нам у подальшому приймати більш правильні рішення. [8]

3.1 Онлайн курси та дистанційне навчання

Про онлайн-навчання, масові відкриті онлайн-курси та дистанційне навчання вже давно говорять як про нову освітню систему у соцмережах та головних світових медіа.

Вперше про відкриті онлайн-курси заговорили майже десять років тому, коли канадським педагогам з університету Манітоби вдалося запустити курс Connectivism and Connective Knowledge, який зібрав понад дві тисячі передплатників. Зі зростанням кількості та популярності перших онлайн-

курсів, стало зрозуміло, що це не новий модний інтернет-тренд, а сучасна альтернатива традиційної освітньої системи.

І якщо ще десять років тому записатися на курси Массачусетського технологічного інституту або вивчати основи програмування під керівництвом викладачів Гарвардського університету могла тільки невелика частина обраних студентів, яким таке навчання обходилося в десятки тисяч доларів, з появою університетських курсів, безкоштовний доступ до кращих джерел знань на планеті став відкритий усім, хто має комп'ютер і підключення до інтернету.

Через кілька років після того, як університетські лекції онлайн стали набирати популярності у світі, все більше вищих навчальних закладів почали говорити про перспективність так званої змішаної освіти (blended education), суть якої полягає в тому, що курси кращих викладачів стають основою для навчання в інших вузах, а успіхи студентів в онлайні переносяться у дипломи. Такий підхід покликаний об'єднати найкраще від офлайн та онлайн в академічній освіті. За словами експертів, таким чином можна вивести освіту на якісно новий рівень - адже з лекторами світових університетів зможуть конкурувати лише найкращі локальні викладачі та вищі навчальні заклади.

У той же час стало зрозуміло, що потенціал онлайн-освіти набагато більший, ніж можливість просто слухати лекції в мережі. Інтернет не тільки зробив навчання доступним для широких мас людей (як це сталося у разі поширення університетських курсів), але й повністю змінив сам підхід до процесу навчання.

Один із головних аргументів противників онлайн-освіти - низький відсоток слухачів, які проходять курси онлайн до кінця. Тобто дуже часто, зареєструвавшись на курс та прослухавши кілька перших лекцій, користувачі кидають навчання. Причиною цього стає не лише недостатність мотивації, без якої будь-яке самонавчання просто неможливе. Виявилося, що головна проблема лежить глибше - більшість онлайн-курсів стали спробою просто перенести університетську довгу освіту у формат онлайн. Збереглися теми та структура лекцій, часові рамки, а також форми контролю. Єдине, що змінилося – це канал трансляції: тепер з'явилася можливість слухати лекції та проходити тестування не в університетській аудиторії, а вдома перед екраном комп'ютера ноутбука чи навіть смартфону.

Перенесення курсів із вузів в інтернет зробило знання більш доступними, але все ж таки не вирішило проблем нестачі часу та мотивації. В умовах динамічної зміни ринку праці, необхідність підвищити рівень своєї кваліфікації або навіть кардинально змінити спеціальність, не відриваючись від роботи, стає актуальною як ніколи.

Чим більше онлайн-навчання є структурованим та гнучким, тим ефективнішим воно виявляється – за мінімальних часових витрат можна значно розширити список компетенцій.

Це, у свою чергу, дає значну перевагу під час пошуку нової роботи. Саме тому за платформами, що надають доступ до онлайн-курсами університетів, все активніше стали розвиватися сервіси іншого формату. Вони обрали шлях структурування навчального контенту, зміну форматів подачі матеріалу та орієнтацію на практичні професійні теми. Результатом стала можливість вивчити невеликий обсяг інформації, яку можна негайно застосувати на практиці. Найбільш популярні серед таких сервісів – Udacity, Udemy, Lynda та багато інших.

Крім можливості, швидко отримати нову кваліфікацію або оновити існуючу, короткі онлайн-формати зробили більш доступним отримання знань у будь-якій галузі. Різноманітні майстер-класи, туторіали та відеоролики здатні допомогти навчитися тому, чого давно хотілося, але часу, як завжди, катастрофічно «не вистачало».

Ще одне поле, на якому онлайн-освіта є актуальною, - це динаміка змін знань, що транслюються. Сьогодні, коли наука та технології розвиваються як ніколи стрімко, виникає питання про те, наскільки раціонально створювати нові підручники та посібники бо з кожним роком вони старіють дедалі швидше. Те саме може стосуватися і університетських курсів, перенесених до онлайн-формату. Водночас коротші курси або туторіали від професіоналів-практиків можуть оновлюватися набагато частіше, а значить, інформація залишатиметься актуальною набагато довше.

Але, за всіх очевидних плюсів онлайн, його головними мінусами досі є складність підтримки мотивації студента на високому рівні та відсутність безпосереднього контакту між учнем та вчителем. Якщо в навчанні формату офлайн завжди присутні зовнішні фактори - чіткий розклад занять, контроль прогулів або тимчасові рамки то той, хто навчається онлайн, залишається наодинці зі своєю здатністю до самоорганізації. Часто люди стикаються з тим, що не можуть дійти до кінця навчальної програми в онлайн-курсах, адже жодних санкцій у випадку, якщо ви не пройдете курс або не опануєте конкретної навички, не буде. Але проблема тут швидше за відсутності навичок самоорганізації, а не у формі навчання.

Онлайн-освіта дає можливість навчатися у тих, хто знаходиться від тебе територіально далеко і будь-коли. Завдяки їй можна знайти курс відповідного рівня і відповідного автора - вибір величезний. Онлайн-освіта підходить для мотивованих людей - тут ніхто не стоятиме над студентом і змушуватиме.

Таким чином, головне правило для того, щоб навчатися онлайн якісно та ефективно полягає в тому, щоб визначитися з метою навчання, знаннями або навичками, які хоче отримати користувач та розібратися , яка з освітніх платформ підійде найкраще.

Поява онлайн-освіти якщо не спровокувала, то значною мірою значно сприяла розвитку концепції безперервного навчання, яка з кожним роком знаходить дедалі більше нових шанувальників. Завдяки онлайну, вам вже не потрібно витрачати години, місяці або роки свого життя, щоб отримати ті чи інші навички чи знання. Онлайн-формати дозволили значно розширити спектр предметів, доступних для вивчення, зміст курсів став більш ємним та структурованим, а процес навчання - комфортнішим. Поки що далеко не всі українці наважуються вчитися у мережі. Проте онлайн-освіта в Україні стала настільки популярною, що ігнорувати її вже просто неможливо. Можливо, саме завдяки їй нарешті вдасться кардинально змінити ситуацію з освітою в країні, зробивши її такою, що відповідає запитам нового часу та доступна тим, хто реально хоче вчитися. [14]

3.2 Принципи інтерактивності

Інтерактивний навчальний контент, такий як в електронному навчанні SCORM, відео, онлайн-вікторини, інфографіка та обговорення на форумах, може стати відмінним способом поглибити навчання онлайн та надати студентам метод підходу до самостійної практики. У цій області проводяться численні дослідження. Вчені вважають, що використання інтерактивного навчального вмісту та методів, як частини процесу електронного навчання може допомогти зробити його більш ефективним загалом. [5]

Інтерактивне онлайн-навчання передбачає вихід за межі пасивного одностороннього процесу читання, прослуховування та перегляду статичного матеріалу. Воно включає в себе врахування саме потрібного вмісту та маніпулювання ним правильним чином та певним структуруванням. Розглянемо чотири основні концепції інтерактивного онлайн-навчання.

Інтерактивний тип вмісту включає попередньо встановлені запрограмовані об’єкти, такі як гіперпосилання на зображення та запитання з кількома варіантами відповіді. Навігація повинна включати в себе умовні гілки, які повертаються до своїх початкових точок на лінійному шляху користувача. Користувачі повинні мати певний контроль над процесом навчання, але мати можливість робити лише те, на що спрямований контент системи.

Якщо слайди не враховують потреб користувачів і не відповідають їм, вміст вважається «неінтерактивним». Натискання кнопки «Далі», щоб перейти до наступного слайду, не являє собою інтерактивним типом змісту. Це нічим не відрізняється від натискання пульта дистанційного керування телевізора або перегортання сторінок книги.

Інтерактивний тип вмісту повинен відповідати як на явні запити користувачів, так і на їх неявні потреби. Прикладами такого вмісту є відеоігри, високоякісні моделювання, навчальні посібники тощо. Суть інтерактивного онлайн-навчання є систематичний, спланований процес розвитку. Навчальна система повинна залучити учня з самого початку, взявши сильний старт і підготувавши основу для навчання. [6]

3.3 Курс SQL на розробленій платформі

Навчальний процес на розробленій навчальній системі у сфері проєктування баз даних структурно розбитий по розділам там підрозділам до них, які продемонстровано на рисунку 9. Розроблену навчальну систему з інтерактивними онлайн-курсами можуть використовувати всі користувачі, які бажають покращити свої навички у сфері проєктування баз даних або отримати інформацію про розуміння принципів складання SQL запитів на зміну та вибірку полів у базах даних.

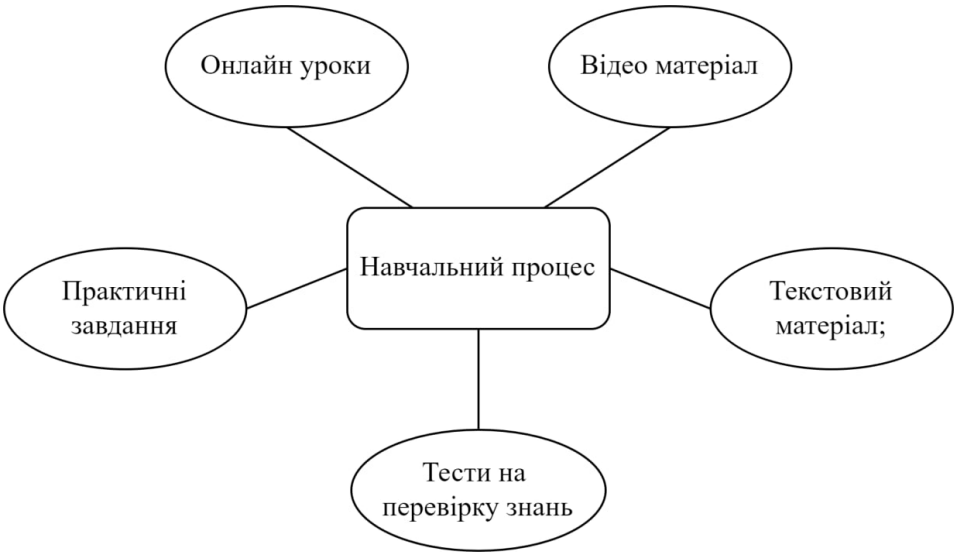


Рисунок 9 – Схема навчального процесу на розробленій платформі

Всього на платформі можна знайти близько 20 тем з навчальними відео, а також майже після кожної теми є можливість пройти одне або декілька практичних завдань для закріплення матеріалу. Сам матеріал навчального курсу у сфері проєктування баз даних структурований таким чином, де на початку користувач зможе ознайомитись з базовими поняттями і відповісти на питання, що таке бази даних? Що призвело до створення баз даних і яку технічну проблему вони вирішують? Як бази даних використовуються в сучасних IT проектах? Як працює вебсайт, який використовує бази даних? Далі користувачу пропонується пройти серію розділів для ознайомлення з тим, як працювати з базами даних за допомогою мови SQL. Відповідно вивчення синтаксису мови взаємодії з базами даних.

Увесь матеріал, присвячений мові SQL, вивченню запитів та команд, інформації про типи даних, зв’язкам між таблицями супроводжується практичними завданнями.

РОЗДІЛ 4 ТЕХНОЛОГІЯ РОЗРОБКИ ІНТЕРАКТИВНОЇ НАВЧАЛЬНОЇ СИСТЕМИ

4.1 Методологія процесу розробки

В даний час застосування каскадної моделі процесу розробки та традиційний підхід до управління створенням програмного забезпечення скорочується. Сучасні підходи вимагають швидкого створення початкової версії системи на ранніх етапах процесу розробки, в якій приділялася б особлива увага високим ризикам, стабілізації базової архітектури та уточненню основних вимог.

Вирішення таких завдань здійснюється при ітераційному методу розробки, який зарекомендував себе якнайкраще при створенні безлічі систем.

Ітераційна модель передбачає розбиття проекту на певні ітерації та проходження етапів життєвого циклу на кожному з них, на рисунку 10 зображено приклад життєвого циклу при використанні даного методу. Сукупність етапів формує кінцевий результат, а кожен етап є окремим.

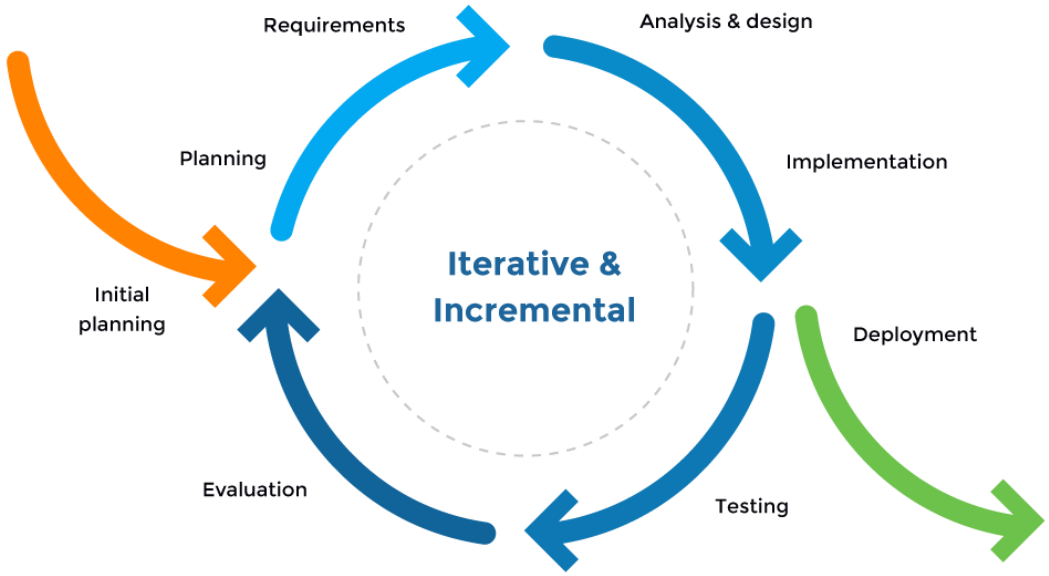


Рисунок 10 – Ітераційна модель

З кожним етапом розробка наближається до кінцевого бажаного результату і відповідно у будь-який момент поточний етап може бути останнім або лише черговим етапом до кінцевого виду рішення.

Ітераційна модель застосовується, коли:

- визначено ґрунтовані вимоги;
- команда застосовує чи вивчає нові технології;
- присутні високі ризики;
- мінливе ринкове середовище.

Переваги:

- швидке постачання робочого функціоналу та відповідно швидкий фідбек від ринку/користувачів;
- періодичний та ранній результат роботи;
- паралельна розробка;
- економія коштів;
- швидке реагування на можливі ризики та зміни;
- підтримка зміни вимог.

Недоліки:

- іноді потребує більше ресурсів;
- вимагає підвищеної уваги з боку проєкт менеджера;
- можуть бути недоліки в системній архітектурі або дизайні через спочатку непродумані вимоги;
- не завжди підходить для маленьких проектів;
- складність управління;
- прогрес проєкту залежить від управління ризиками.

Використання ітераційної моделі дає можливість завершення розробки в кінці будь-якої ітерації. Вона також знижує ризики та витрати усього бюджету, який був виділений на певний проєкт. Слід розуміти, що не завжди

та модель, яка сподобалася з опису, буде найкращою для реалізації саме для конкретного проєкту. [1]

4.2 Архітектура системи

Архітектура розробленої системи поділяється на два модулі: клієнтський та адміністративний. Детально схему модулів та всіх можливих функцій до них зображено на рисунку 11.

Адміністративний модуль включає в себе функцію роботи з навчальним матеріалом, який передбачає додавання нових курсів та редагування існуючих курсів. До функції додавання нових курсів входить можливість створення нового курсу з будь-якою кількістю тем, лекцій та практичних завдань. Їх кількість та зміст залежить безпосередньо від навчального процесу у сфері проєктування баз даних та матеріалу до нього. Аналогічно до функції редагування входять можливість редагування існуючих курсів, будь то зміна або видалення існуючих розділів, практичних та лекційних завдань до них.

Клієнтських модуль передбачає функції реєстрації, введення профілю та проходження навчання. Завдяки функції реєстрації користувач повинен зареєструватися в системі, після чого він отримує особистий акаунт для проходження матеріалу на навчальній платформі. До функції ведення профілю входить можливість переглядати та редагувати особистий кабінет. Функція проходження навчання включає в себе ознайомлення з теоретичним матеріалом, а також можливість виконати практичні завдання для закріплення пройденого матеріалу.

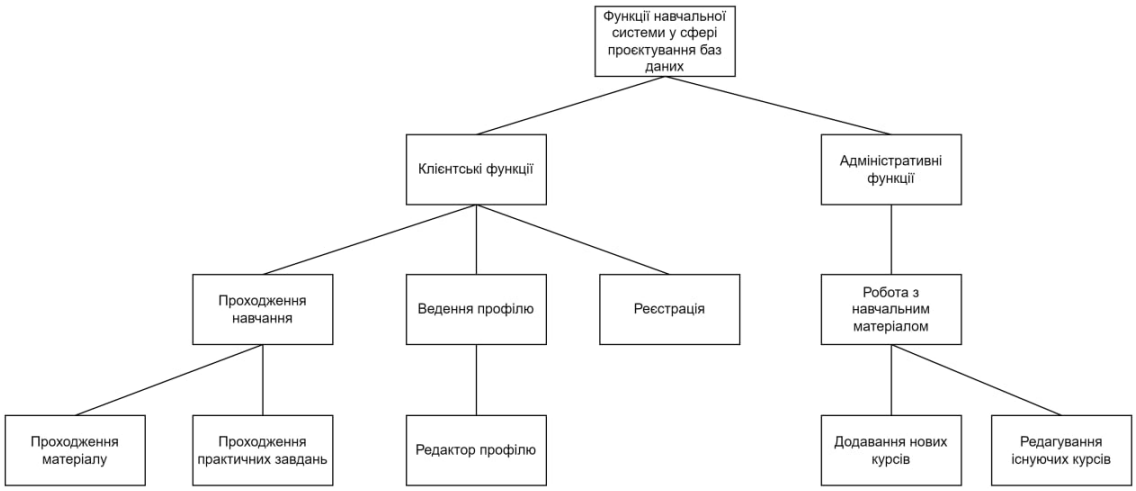


Рисунок 11 – Схема архітектури розробленої системи

Функціонування системи починається після логіну або реєстрації користувачем на початковій сторінці авторизації. Після успішної авторизації або реєстрації користувач переходить до головної сторінки усіх курсів та можливих тем до них у сфері проєктування баз даних. Користувач має можливість не починати з самого початку та вибрати той розділ, який його цікавить або який ще не проходив. Перейшовши до відповідного розділу система завантажує з бази даних інформацію цього курсу необхідний матеріал та практичні завдання, які користувач може відкривати та проходити в порядку за власним бажанням. Діаграма даної бази даних можна детально розглянути на рисунку 12. Після проходження практичного завдання користувач має можливість перевірити правильність його відповідей та пройти практичне завдання знову за необхідності. При повторній спробі пройти практичне завдання, сторінка оновить усі заповненні поля. Детальна схема функціонування розробленої системи можна переглянути в додатку В.

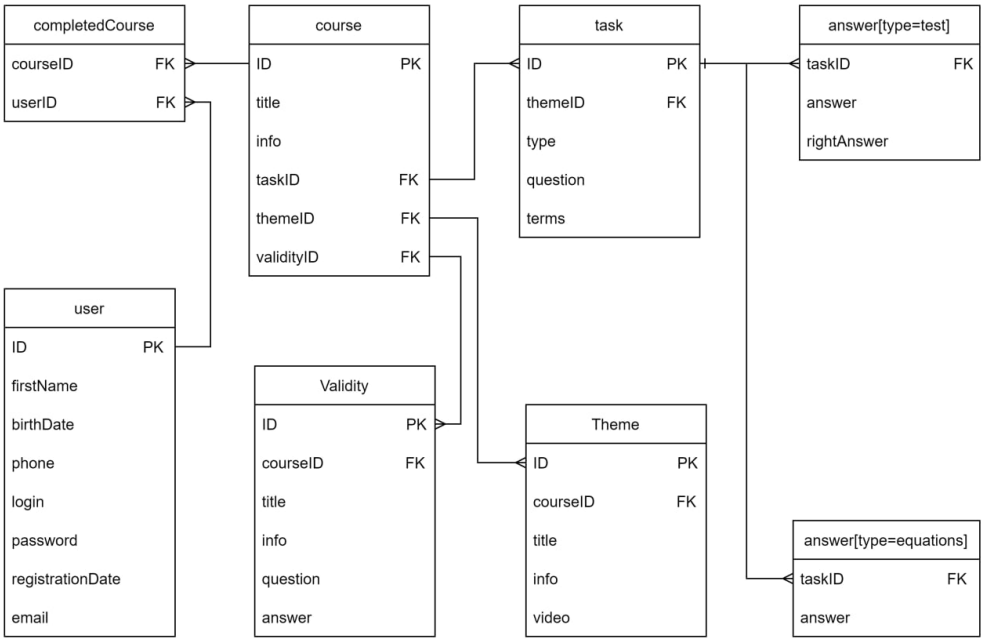


Рисунок 12 – Діаграма бази даних

Система розроблена таким чином, що будь-які оновлення та розширення фунціоналу застосунку можуть бути швидко та ефективно інтегровані.

4.3 Структура лекційного інтерфейсу

Будь-який навчальний процес неможливий без вивчення теоретичного матеріалу. Весь курс у сфері проєктування баз даних на розробленій навчальній системі супроводжується темами з теоретичним матеріалом. Він викладений в спеціальному лекційному інтерфейсі. Приклад даного інтерфейсу зображений на окремо створеному для демонстрації можливостей роботи лекційного інтерфейсу на рисунку 13.

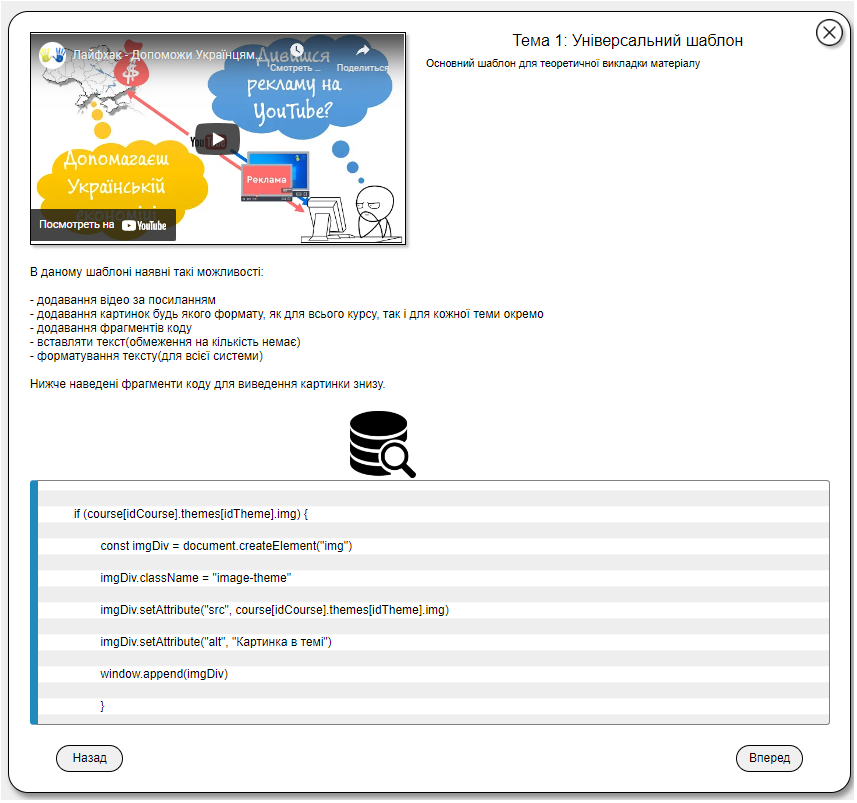


Рисунок 13 – Демонстрація лекційного інтерфейсу

Структура відображення лекційного інтерфейсу:

- навчальне відео за посиланням;
- будь який текст під відео для пояснень або уточнень(обмеження на кількість немає);
- картинки та таблички будь-якого формату, як для всього курсу, так і для кожної теми окремо;
- фрагменти коду.

В системі присутня також навігація по темам, лекціям та практичним завданням, яка зображена на рисунку 14. Вона відображається з правої частини екрану і дає можливість зручно пересуватись по темам курсу.

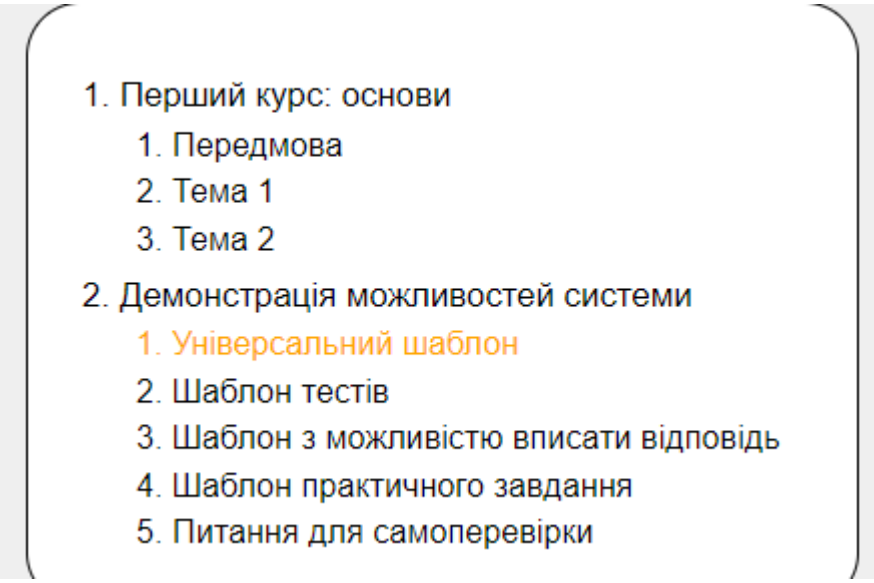


Рисунок 14 – Навігація по темам та завданням курсу

4.4 Структура інтерфейсу практичних завдань

Щоб краще засвоїти теоретичний матеріал, в розробленій навчальній системі є можливість зробити практичні завдання після кожної лекції. Приклад одного з таких практичних завдань, де потрібно вписати правильну відповідь зображено на рисунку 15.

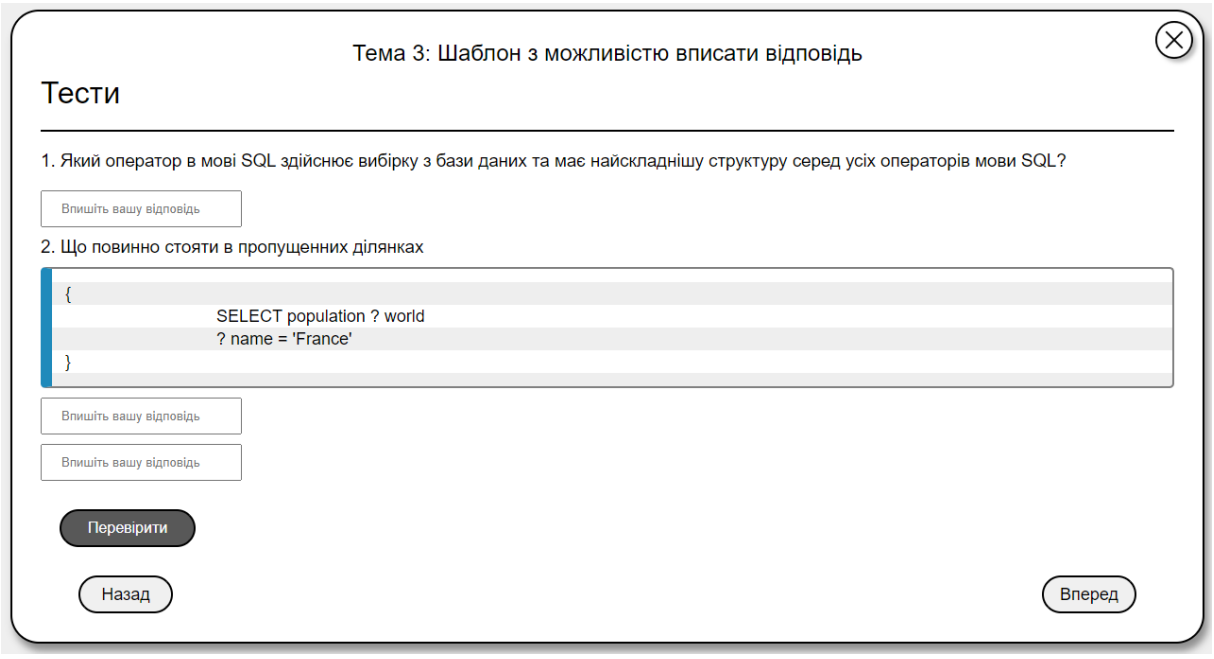


Рисунок 15 – Приклад практичного завдання

Кількість та тип практичних завдань для кожної теми можуть відрізнятись. В кожному окремому практичному завданні кількість питань та їх тип також може бути різним, що продемонстровано на рисунку 17. В багатьох тестових завданнях є можливість перевірити правильність відповідей та виконати завдання ще раз. Коли користувач натисне кнопку «Перевірити», сторінка оновиться і підкреслить зеленим та красним кольорами правильні та неправильні відповіді відповідно. Приклад демонстрації роботи інтерфейсу практичного завдання після його перевірки зображено на рисунку 16.

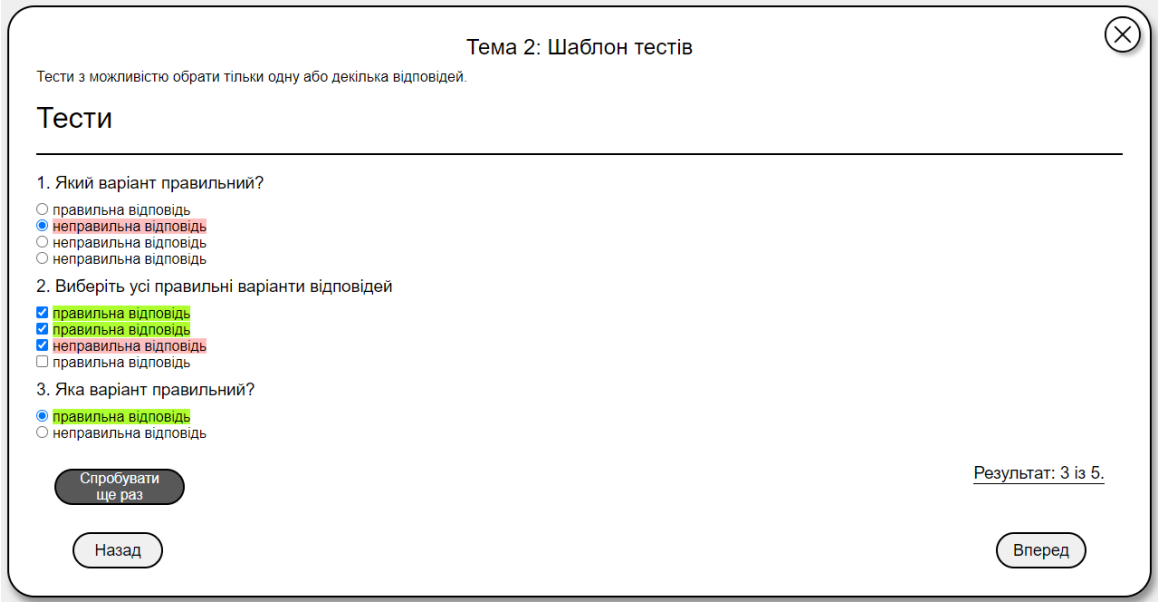


Рисунок 16 – Перевірка тестових завдань

Перевірка тестових завданнь, де можливо обрати декілька відповідей, не відобразить в системі красним кольором ті правильні відповіді, які не були обрані користувачем і не будуть записуватись у суму поля «Результат».

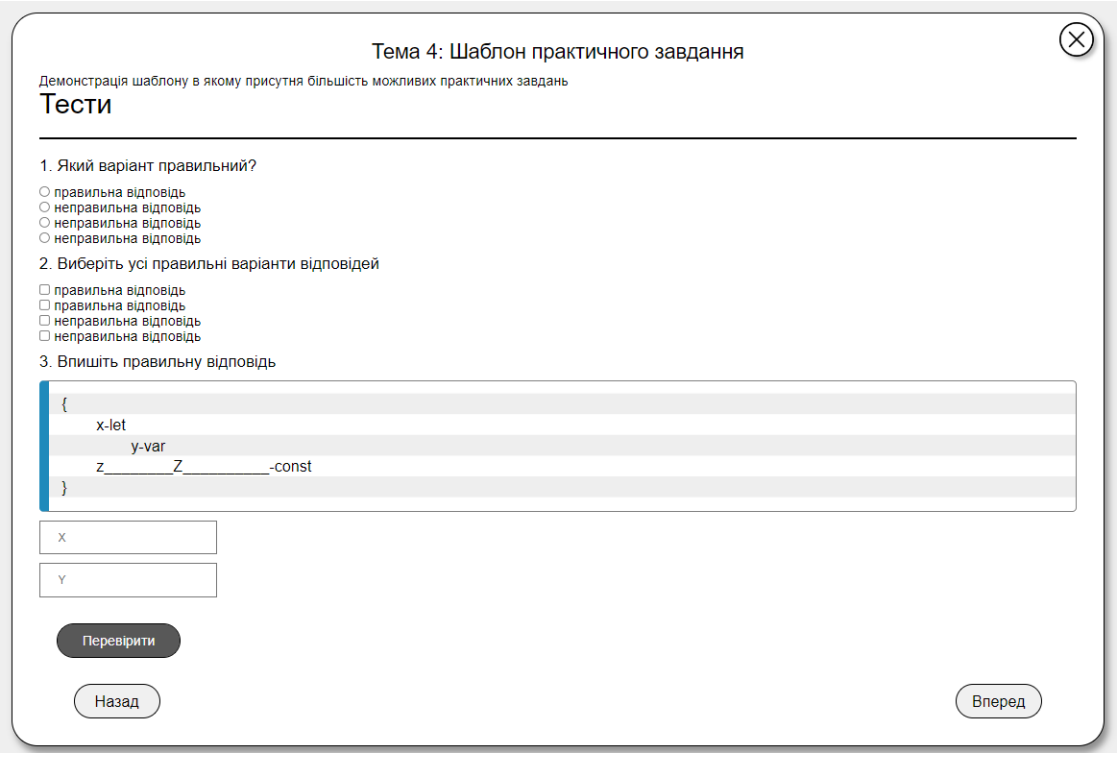


Рисунок 17 – Інтерфейс зі змішаним типом завдань

Код основної частини системи та реалізація інтерфейсів можна переглянути в додатку Б.

ВИСНОВКИ

Проведено аналіз інформаційних систем підтримки навчального процесу.

Досліджено наявні навчальні системи та курси, що використовуються у сферах проєктування баз даних та різних мов програмування.

Проведено збір матеріалів та інформації, необхідної для створення інтерактивних курсів. Викладення матеріалу структурно розбито по курсам і темам та інтегровано в інтерактивну навчальну систему.

Результатом роботи є інтерактивна навчальна система у сфері проєктування баз даних, структурована по курсам та темами до них.

Розроблену навчальну систему з інтерактивними онлайн-курсами можуть використовувати всі користувачі, які бажають покращити свої навички у сфері проєктування баз даних або отримати інформацію про розуміння принципів складання SQL запитів на зміну та вибірку полів у базах даних.

Використання навчальної системи дозволяє користувачеві отримати структуровану за темами інформацію про проєктування баз даних та виконати практичні завдання для закріплення теоретичного матеріалу у вигляді тестів, питань для самоперевірки та спробувати скласти SQL-запити різних типів.

Система розроблена таким чином, що будь-які оновлення та розширення фунціоналу застосунку можуть бути швидко та ефективно інтегровані.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Соммервілл И. Інженерія програмного забезпечення, 6-е видання: Пер. з англ. - М.: Видавничий дім "Вільямс", 2002. - 624 с.

2. Virtual Document Object Model [електронний ресурс] – Режим доступу до ресурсу: https://habr.com/en/post/256965/ (31.05.2022)

3. What is JavaScript, and why is it important [електронний ресурс] – Режим доступу до ресурсу: https://www.bigcommerce.com/ecommerceanswers/what-javascript-and-why-it-important/ (31.05.2022)

4. Тематична платформа для вивчення SQL [електронний ресурс] – Режим доступу до ресурсу: https://habr.com/en/company/spbifmo/blog/595825/ (31.05.2022)

5. Interactive Learning Content [електронний ресурс] – Режим доступу до ресурсу: https://www.cae.net/interactive-learning-content-elearning/ (31.05.2022)

6. Interactive Online Learning: Meaning, Principles, Examples [електронний ресурс] – Режим доступу до ресурсу: https://raccoongang.com/blog/interactive-online-learning-meaning-principles-examples/ (31.05.2022)

7. Навчальна платформа Ulearn [електронний ресурс] – Режим доступу до ресурсу: https://ulearn.me/ (31.05.2022)

8. Introduction to SQL [електронний ресурс] – Режим доступу до ресурсу: https://sqlbolt.com/lesson/introduction (31.05.2022)

9. What is Database Management Systems (DBMS)? [електронний ресурс] – Режим доступу до ресурсу: https://www.appdynamics.com/topics/database-management-systems#~2-examples-of-dbms (03.06.22)

10. Date C. J. Introduction to Database Systems, 8-е видання: Пер. з англ. - М.: Видавничий дім "Вільямс", 2006. - 1328 с.

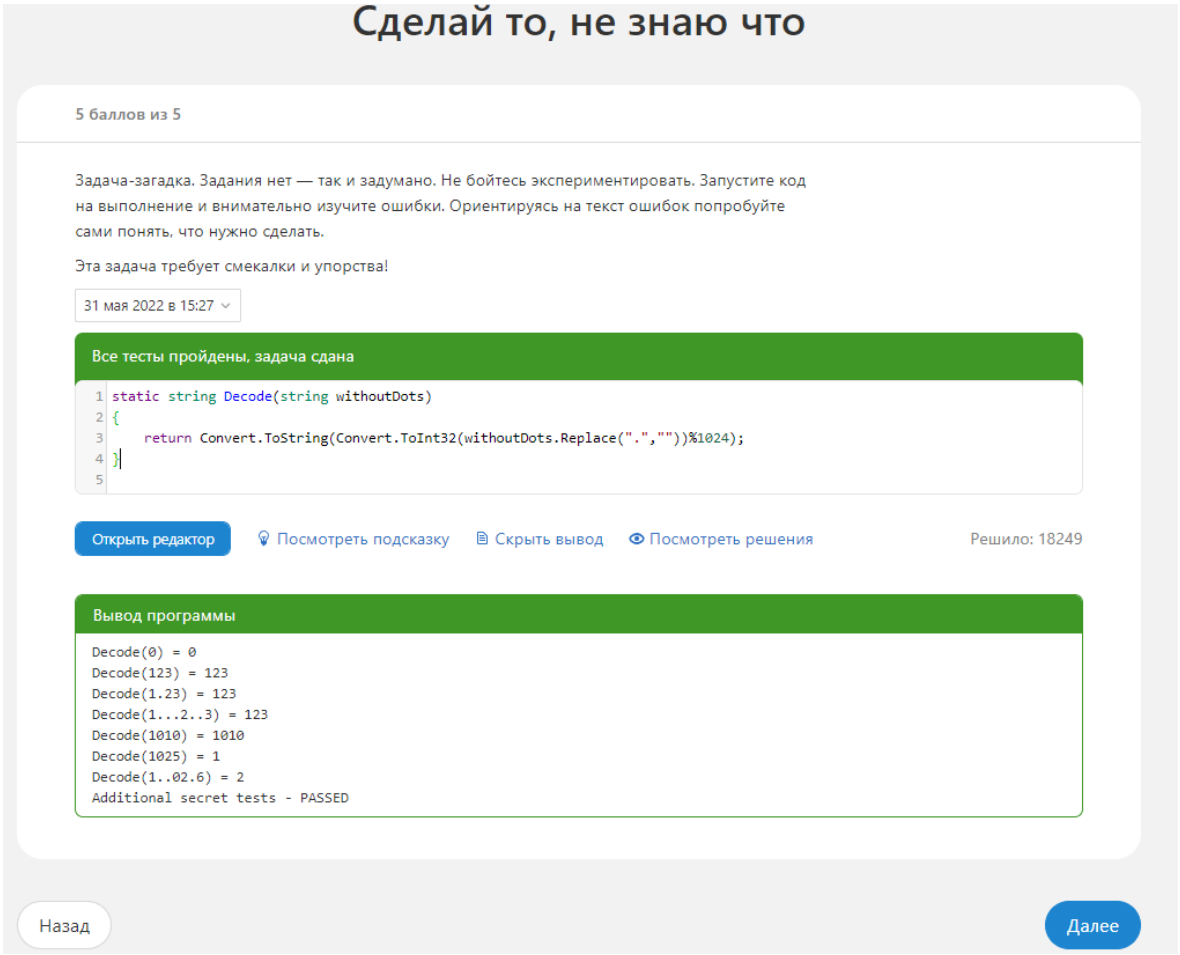
11. Structured Query Language [електронний ресурс] – Режим доступу до ресурсу: https://www.techtarget.com/searchdatamanagement/definition/SQL (04.06.2022)

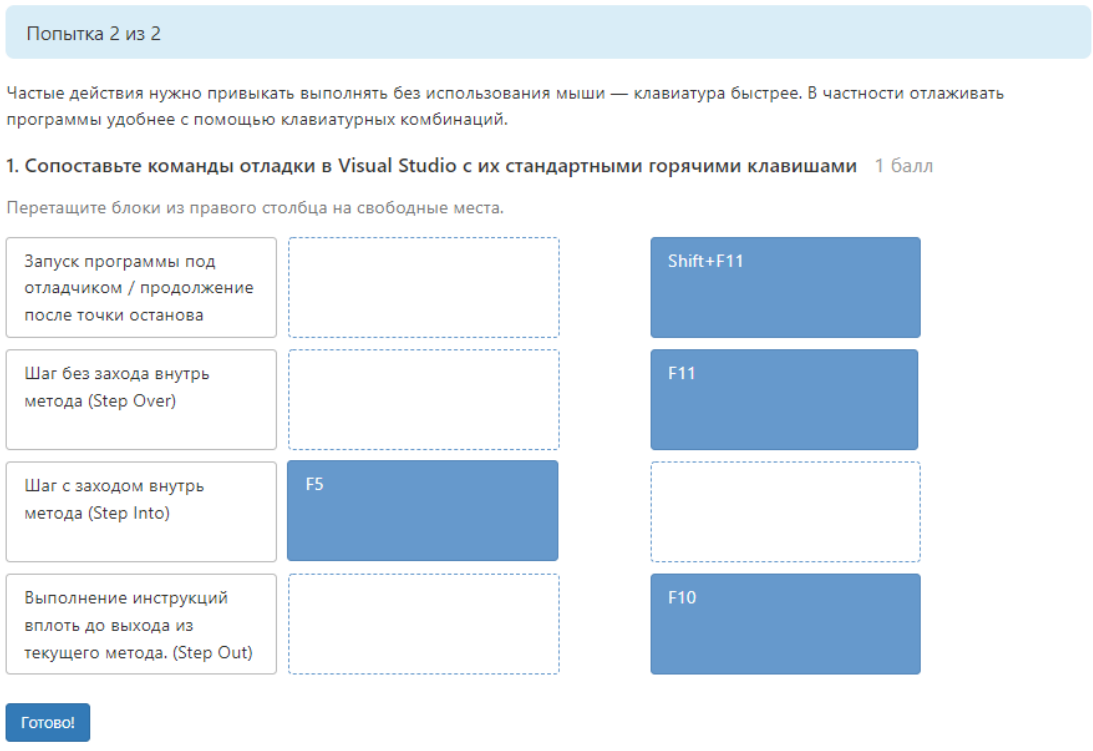
12. Платформа HackerRank [електронний ресурс] – Режим доступу до ресурсу: https://www.hackerrank.com/dashboard (04.06.22)

13. Платформа SQLZOO [електронний ресурс] – Режим доступу до ресурсу: https://sqlzoo.net/wiki/SQLZOO:About (04.06.22)

14. Чому онлайн-освіта перемагає у світі та в Україні [електронний ресурс] – Режим доступу до ресурсу: https://platfor.ma/magazine/text-sq/projects/traditional-vs-online-education/ (04.06.22)

ДОДАТОК А ПРИКЛАД ПРАКТИЧНИХ ЗАВДАНЬ НА ПЛАТФОРМІ ULEARN





ДОДАТОК Б ПРОГРАМНИЙ КОД ОСНОВНОЇ ЧАСТИНИ СИСТЕМИ ТА ВІДОБРАЖЕННЯ ІНТЕРФЕЙСІВ

```
const courseOutput = (course) => {

    const container = document.querySelector(".container")

    const divContainer = document.createElement("div")
    divContainer.className = "container__right-container"

    container.append(divContainer)

    for (let i = 0; i < course.length; i++) {
        const courseThemes = document.createElement("div")
        courseThemes.className = "right-container__bottom-themes"

        const circle = document.createElement("div")
        circle.className = "circle-progress"

        const circleInCircle = document.createElement("img")
        circleInCircle.className = "circle-progress-center"
        circleInCircle.setAttribute("src", course[i].img)

        divContainer.append(courseThemes)
        courseThemes.append(circle)
        circle.append(circleInCircle)

        const centerInfo = document.createElement("div")
        centerInfo.className = "center-info"

        const title = document.createElement("h2")
        title.innerHTML = course[i].title

        const info = document.createElement("p")
        info.innerHTML = course[i].info

        const paragraph = document.createElement("p")

        courseThemes.append(centerInfo)
        centerInfo.append(title)
        centerInfo.append(info)
        centerInfo.append(paragraph)

        const divLessonsThemes = document.createElement("div")
        divLessonsThemes.className = "lessons-themes"
        centerInfo.append(divLessonsThemes)

        const olList = document.createElement("details")
```

```
        divLessonsThemes.append(olList)

        for (let j = 0; j < course[i].themes.length; j++) {

            if (j < 1) {
                const titleListItem = document.createElement("summary")
                titleListItem.innerHTML = "Показати список тем"
                olList.append(titleListItem)
            }
            const listItem = document.createElement("p")
            olList.append(listItem)

            const listItemLink = document.createElement("a")
            listItemLink.setAttribute("onclick", `createLesson(${i},${j})`)
            listItemLink.innerHTML = `${course[i].themes[j].title}`
            listItem.append(listItemLink)
        }
    }
}


const createLesson = (idCourse, idTheme) => {
    courseClose()

    crossClose()

    const menuRight = document.querySelector(".rightMenu")
    if (menuRight) menuRight.remove
    rightMenu(course, idCourse, idTheme)

    const body = document.querySelector("body")
    const window = document.createElement("div")
    window.className = "window-leson"

    body.append(window)
    const videoTitle = document.createElement("div")
    videoTitle.className = "video__title"
    window.append(videoTitle)

    if (course[idCourse].themes[idTheme].video) {
        const iframe = document.createElement("iframe")
        iframe.classList = "video"
        iframe.setAttribute("src", course[idCourse].themes[idTheme].video)
        iframe.setAttribute("title", "YouTube video player")
        iframe.setAttribute("frameborder", 0)
        iframe.setAttribute("allow", "accelerometer; autoplay; clipboard-write;
encrypted-media; gyroscope; picture-in-picture")
        iframe.setAttribute("allowfullscreen", '')
        videoTitle.append(iframe)
    }
```

ДОДАТОК В СХЕМА ФУНКЦІОНУВАННЯ РОЗРОБЛЕНОЇ НАВЧАЛЬНОЇ СИСТЕМИ