# Вступ

За останні століття накопичилася величезна кількість знань, а за останні десятиліття спостерігається істотне збільшення їх обсягів, що приводить в свою чергу до збільшення складності їх викладу в навчальних закладах. При цьому в останніх спостерігається недолік висококваліфікованих викладацьких кадрів. Великі труднощі часто виникають при оперативній підготовці, виготовленні та розповсюдженні навчальних посібників різних видів. Зазначені фактори негативно позначаються на якості підготовки учнів. У зв'язку з цим велика увага приділяється застосуванню прогресивних методів навчання з використанням революційних інформаційних технологій. Для сучасного етапу розвитку освіти характерний перехід від традиційного навчання до нових педагогічних технологій. Особлива увага приділяється тим технологіям, які створюють найбільш сприятливе середовище для розвитку обдарованих дітей та міцну основу для побудови дидактичної системи розвитку потенціалу, наявного у кожної дитини.

Використання сучасних інформаційних технологій у навчальному процесі дозволяє підвищити якість навчального матеріалу і підсилити освітні ефекти, оскільки дає викладачам додаткові можливості для побудови індивідуальних освітніх траєкторій учнів. Застосування інформаційних технологій дозволяє реалізувати диференційований підхід до учнів з різним рівнем готовності до навчання.

Програми, призначені для передачі кого навчають знань і для оцінки поточного його рівня знань щодо рівня знань якогось еталона (за еталон можуть бути взяті знання викладача), отримали назву Автоматизовані навчальні системи (АНС).

Місце АНС вже давно показано в сучасній системі освіти. Також показано і те, що їх впровадження не спрямоване на підміну людини-педагога або традиційних навчальних посібників, а, навпаки, покликане заповнювати і розширювати їх можливості Таким чином, пасивно, без бажання ознайомитися з досліджуваними матеріалами, ефекту виявлення здібностей не буде. І бездушний комп'ютер з АНС в цьому плані не помічник, а от людина емоційна розуміє природу подібних собі може залучити до предмета щиру зацікавленість, якщо він звичайно людина розуміє суть проблеми навчання, а не тільки викладається матеріалу.

Процес створення сценарію майбутньої АНС викладачем процес творчий і тому має бути звільнений від усіляких рутинних операцій для максимальної зосередженості на елементах і відносинах предметної області. Цей процес являє собою не що інше, як придбання знань у експерта-викладача, що є двічі експертом: в професійній області і в галузі педагогіки.

Таким чином, виділена наступна проблема: існує безліч методів і розроблених на їх основі засобів навчання, проте вони мають недоліки з точки зору педагогіки.

# ПОСТАНОВКА ЗАдачі

Метою даної роботи є розробка віртуального лабораторного комплексу, в якій навчальний матеріал структурований згідно розробленої технологічної карти, є в наявності контрольні питання, тренаж і підсумкове тестування.

Об'єктом дослідження є адаптивні навчальні системи, а адаптивна навчальна система по темі СКБД буде предметом дослідження.

У ході науково-дослідної роботи студента, ми ставимо перед собою мету: підвищити якість засвоєння навчального матеріалу по темі «Бази даних». СКБД студентами на лекціях за рахунок використання в освітньому процесі адаптивної навчальної системи.

Гіпотеза: підвищення якості засвоєння навчального матеріалу студентами по розділу Бази даних. СКБД може бути досягнуто, якщо використовувати в освітньому процесі (як на лекціях, так і вдома) автоматизовану навчальну систему, в якій навчальний матеріал структурований згідно розробленої технологічної карти, є в наявності контрольні питання, тренаж і підсумкове тестування.

З поставленої нами мети випливають такі завдання:

1. Аналіз застосування в освітньому процесі навчальних систем.
2. Розробити модель автоматизованого навчання.
3. Провести відбір змісту і структурування навчального матеріалу з теми Бази даних. СКБД.
4. Розробити методичні рекомендації щодо використання.
5. Провести апробацію створеного програмного продукту.

Для вирішення поставлених завдань були використані такі методи дослідження: вивчення та аналіз психолого-педагогічної, наукової та методичної літератури з проблематики дослідження; педагогічний експеримент; спостереження; опитування навчаються.

Для створення системи необхідно розглянути декілька технологій. По-перше, це технологія об'єктно-орієнтованого проектування. Дана технологія є базовою для всіх проектів, що розробляються з застосуванням мови Java і відповідної платформи. По-друге, це технологія розробки проекту, яка керується тестуванням. Застосування цієї технології дозволить збільшити покриття проекту тестами, що важливо для його подальшої підтримки.

Виходячи з цього, розробляється система повинна переслідувати такі цілі для кокистувача, якими є отримання:

* знань з конкретної предметної області;
* умінь застосовувати різні методи і алгоритми;
* навичок вирішення завдань;
* оцінки набутих знань, умінь і навичок.

Крім цього необхідно отримати знання, що стосуються застосування багаторівневої архітектури та шаблонів проектування. Ці підходи до проектування програмного забезпечення дозволяють використовувати загальновідомі і перевірені рішення для загальних завдань.

Також для створення цієї системи необхідно розглянути і застосувати ряд технічних засобів. Ці технічні засоби повинні забезпечити можливість управління процесом розробки програмного забезпечення.

Як мінімум необхідно використовувати три засоби – для управління збором проекту, управління вихідними кодами і тестування. Цих коштів достатньо для створення технологічного процесу розробки програмного забезпечення.

Виходячи з цього, навчальна програма повинна включати в себе 3 основні частини:

* теоретичну;
* тренуючу;
* контролюючу.

Крім позначених функціональних вимог до розроблюваної системі висувається і ряд нефункціональних. Ці вимоги обумовлені необхідністю підтримувати і розширювати цю систему надалі.

Нефункціональними вимогами є:

1. система не повинна залежати від конкретного середовища розробки;
2. система повинна підтримувати можливості для автоматичного складання і тестування;
3. система повинна бути модульною;
4. при розробці системи повинні використовуватися сучасні та загальновідомі інструменти.

Відповідність нефункціональним вимогам полегшить розробникам, які будуть розширювати і підтримувати проект у майбутньому.

# Поняття демонстраційної системи

Одним з ефективних методів навчання, на мою думку, є використання тестів. Використання тестів є перспективним засобом навчання у вузі, оскільки застосування персональних комп'ютерів дозволяє автоматизувати процес обробки результатів і скоротити часові витрати на перевірку рішень. У системі підготовки фахівців тести виконують такі функції: інформаційні, діяльні, контролюючі, актуализирующие, що діагностують, розвиваючі, навчально-творчі та навчально-тренувальні. Безпосереднім результатом практичних занять за розробленою методикою тестування є формування умінь: з'ясовувати точний зміст теми; застосовувати матеріал до конкретних відносин; правильно визначати характер і структуру теми.

До таких методів належить проблемне навчання, що передбачає формування навичок для вирішення проблемних завдань, які не мають однозначної відповіді, самостійної роботи над матеріалом і вироблення умінь застосовувати набуті знання на практиці.

Застосовувані на сучасному етапі інноваційні методи навчання у вузі передбачають і метод, пріоритетом якого є моральні цінності. Він сприяє формуванню індивідуальних моральних установок, заснованих на професійній етиці, виробленні критичного мислення, вміння представляти і відстоювати власну думку.

У висновку хотілося б відзначити, що інноваційні методи дозволили змінити і роль викладача, який є не тільки носієм знання, але і наставником, який ініціює творчі пошуки студентів.

## 2.1 Автоматизовані адаптивні навчальні системи

Все більш помітна тенденція до постійного підвищення швидкості оновлення знань. Для сучасно студента ключовим фактором є знання. І чим швидше студент зможе вивчитися, тим швидше він почне розуміти необхідний матеріал з якої-небудь дисципліни. Для цього необхідні кошти, щоб забезпечити мобільність знань, що спрощують і підвищують оперативність доступу до них.

Одним з напрямків підвищення ефективності навчання, засвоєння інформації і скорочення витрат на сам процес навчання є розробка і використання автоматизованих навчальних систем.

Автоматизована навчальна система - це комплекс програмно-технічних та навчально-методичних засобів, що забезпечують надання учнем досліджуваного матеріалу, перевірку знань учнів, інтерактивна взаємодія учнів і викладачів в процесі навчання, а також надання учнем можливості самостійної роботи з освоєння досліджуваного матеріалу.

В даний час застосовується безліч термінів, що позначають автоматизовану навчальну систему, які, по суті, є аналогічними. Найбільш поширеними з них є:

* система дистанційного навчання (СДН);
* автоматизована система дистанційного навчання (АСДН);
* автоматизована система навчання (АСН);
* віртуальний лабораторний комплекс;
* навчальна програма;
* навчальна система і ряд інших, менш поширених понять.

## 2.2 Цілі автоматизованих адаптивних навчаючих систем

Можливо найважливішими цілями, які переслідують організації, впроваджують автоматизовані навчальні системи (системи дистанційного навчання) або ж стоять перед необхідністю їх впровадження, є:

1. надання однакового якісного навчального матеріалу великій кількості користувачів і, одночасно з цим;
2. скорочення витрат на навчання.

Крім цього застосування АНС дозволяє:

* вирішити проблеми з одночасним навчанням значної кількості студентів;
* вирішити проблеми з навчанням географічно розподілених фахівців;
* забезпечити навчання студентів у зручний для них час;
* підвищити конкурентоспроможність, за рахунок надання клієнтам можливості дистанційного навчання правилам використання продуктів і послуг;
* істотно скоротити витрати на оновлення навчального матеріалу (немає необхідності актуалізувати інформацію у кожного користувача, коли можна один раз оновити інформацію навчальних модулів (курсів), розміщених на сервері);
* представити інформацію в максимально наочній формі за рахунок можливості використання в складі навчальних курсів різного медіа-контенту.

## 2.3 Функціональна структура автоматизованих адаптивних навчаючих систем

Автоматизована навчальна система (система дистанційного навчання) являє собою програмно-апаратну середовище, що складається з наступних функціональних блоків(див. рис 2.1):

1. Підсистема доступу учнів до навчального матеріалу, статистичними даними, а також різних сервісів (електронна пошта, чати, новинні стрічки тощо) АНС.
2. Підсистема адміністрування навчального процесу, яка забезпечує управління доступом учнів до інформації АНС (модулі, курси, тестові питання), планування навчання, обробку статистичної інформації за результатами діяльності учня в системі.
3. Підсистема зберігання інформації АНС. Під цим мається на увазі, як правило, база даних, яка містить всю інформацію, необхідну для реалізації навчального процесу - це курси (модулі, заняття, тестові питання), додатковий матеріал (інструкції, рекомендації, довідкова інформація) і статистичні дані процесу навчання (час знаходження користувача в системі, активність його на семінарах і засудженнях, оцінки, отримані користувачем при проходженні тестових питань і ряд інших показників).
4. Останній функціональний блок призначений для створення електронного навчального контенту (курсів, тестових питань додаткових матеріалів), що розміщується в АНС.



Рисунок 2.1 – Функціональна структура автоманизованих адаптивних навчаючих систем

# Актуальність роботи і Аналіз існуючих аналогів

3.1 Актуальність роботи

Одна з найважливіших проблем дидактики – проблема методів навчання – залишається актуальною як в теоретичному, так і безпосередньо в практичному плані. Залежно від її рішення перебувають сам навчальний процес, діяльність викладача і студентів, а, отже, і результат навчання у вищій школі в цілому.

На сучасному етапі розвитку нашого суспільства як ніколи зросла соціальна потреба в нестандартно мислячих творчих особистостей. Потреба у творчій активності спеціаліста та розвиненому технічному мисленні, в умінні конструювати, оцінювати, раціоналізувати техніку і технологію швидко зростає. Вирішення цих проблем багато в чому залежить від змісту і технології навчання майбутніх фахівців. У педагогічному процесі інноваційні методи навчання передбачають введення нововведень в цілі, методи, зміст і форми навчання і виховання, у спільну діяльність викладача і учня. Ці інновації можуть бути спеціально спроектованими, вже розробленими або знову з'явилися завдяки педагогічної ініціативи.

Головним завданням вищого навчального закладу на сучасному етапі є підготовка фахівців, здатних нестандартно, гнучко і своєчасно реагувати на зміни, які відбуваються у світі. Тому для підготовки студентів до професійної діяльності в майбутньому і використовуються інноваційні методи навчання у вузі. Однією з центральних проблем у сучасній методиці викладання є проблема підвищення ефективності навчання.

Інтенсифікація навчання передбачає використання сучасних форм і засобів навчання, застосування у викладанні нових методів пізнання: використання комп'ютерів, аудіо, відео та електронно-обчислювальної техніки.

Створення, та вдосконалення комп'ютерів призвело до створення нових технологій у різних сферах наукової та практичної діяльності. Однією з таких сфер стало утворення процес передачі систематизованих знань, навичок і вмінь.

Завдяки своїм конструктивним і функціональним особливостям сучасний комп'ютер є унікальною за своїми можливостями навчальною машиною. Він знаходить застосування в навчанні найрізноманітнішим дисциплін і служить базою для створення великої кількості нових інформаційних технологій.

Використання комп'ютера як навчального засобу дозволяє:

* активізувати навчальний процес;
* індивідуалізувати навчання;
* підвищити наочність у пред'явлення матеріалу;
* змістити акценти від теоретичних знань до практичних;
* підвищити інтерес учнів до навчання.

Можливості комп'ютера роблять його дуже перспективним для використання в навчальному процесі навчальним засобом.

В даний час існує величезна безліч навчальних програм, орієнтованих на самі різні категорії учнів.

3.2 Аналіз існуючих аналогів

Суспільство зі скепсисом ставиться до самої ідеї заміни вчителя-людину вчителем-комп'ютером. У більшості своїй панує ідея про те, що комп'ютер ніколи не зможе замінити вчителя. При цьому вказується на неможливість комп'ютера зацікавити учня, на неможливість заміни комп'ютером реального спілкування, на неможливість замінити автоматизованим діалогом емоційно-особистісних компонентів взаємодії і співпраці, на інші соціальні та психологічні причини незамінності людини-викладача.

Автоматизація педагогічного процесу розглядається з позицій концепції автоматизації основних напрямків розумової праці, що передбачала перекласти всі рутинні функції з особи на комп'ютер.

Вже при створенні АНС першого покоління були вперше сформульовані принципи адаптації автоматизованого навчання під рівень підготовленості учня.

З часом з'являються перші АНС створюють модель учня і предметної області; їх називають експертними або інтелектуальними системами. Такі системи адаптують зміст освіти до цілей учня, а також рівню його підготовки, створюючи модель учня.

АНС все більше поступаються місцем дистанційним систем навчання, що використовують ресурси мережі Інтернет. Такі системи, що базуються на основі розподілених прав доступу до сайту, дозволяють здійснювати віддалене навчання в асинхронному режимі, коли вчитель і учень заходять в систему у зручний для них час; деякі підтримують організацію синхронних сеансів навчання.

Були створені вітчизняні програмні оболонки для навчання через Інтернет: «Прометей», eLEARNING SERVER 3000, ОРОКС, «Доцент», WEBTUTOR, COMPETENTUM, СДТ REDCLASS, СДО AcademicNT, а також зарубіжні: WebCT, Oracle iLearning, IBM Workplace Collaborative Learning, Microsoft ClassServer , Learn eXact, Moodle.

## 3.2.1 АНС «Контакт»

АНС «Контакт» була створена колективом Ризького політехнічного інституту під керівництвом Л.В. Ніцецкого в кінці 60-х на базі комп'ютерів «Мінськ-32»; в якості терміналів використовувалися електричні друкарські машинки, телетайпи і дисплеї.

Програма здійснювала контроль знань з комплексу питань попередньо занесених до її базу даних; програма підтримувала розгалужену логіку видачі обучающемуся питань: на підставі правильності відповідей на перші питання програма формувала уявлення про підготовленість учня і видавала ті чи інші питання виходячи з рівня його знань. Модифікація АНС «Контакт/НС» 1982 включала 40 обучающе-контролюючих і 29 контролюючих комплексів питань з мов програмування, операційних систем, філософії, інженерної графіки та інших дисциплін.

3.2.2 АНС ЕОНС та АТНС

АНС ЕОНС (Експериментальна Обчислювальна Навчальна Система) та АТНС (Автоматизована Телевізійна Навчальна Система) були створені в Білоруському Державному Університеті в результаті наукових досліджень, санкціонованих Мінвузом і ДКНТ. Евос використовувала спеціальні пульти викладача та учнів, з'єднані з єдиним пристроєм управління, в ролі якого виступав комп'ютер.

За таким же принципом був побудований АТНС, що використав в якості терміналу телевізор «Юність» і спеціально розроблену клавіатуру.

## 3.2.3 АНС САДКН

АНС САДКН (Система Автоматизованого Діалогу та Колективного Навчання) була розроблена у ВЦ Мінвузу РРФСР і МТІПП; базувалася на спільному застосуванні комп'ютера та навчального посібника на паперовому носії. Учень проходив певний навчальний матеріал за посібником та, виконавши завдання в робочому зошиті, вводив відповідь за допомогою спеціального пульта в АНС у вигляді формул, чисел, букв або тексту. АНС перевіряла відповідь і залежно від його правильності видавала інформацію про рішення, а також про номер сторінки посібники, на яку учень повинен був перейти для продовження навчання. Інтерфейс програми можна побачити на рисунку 3.1.

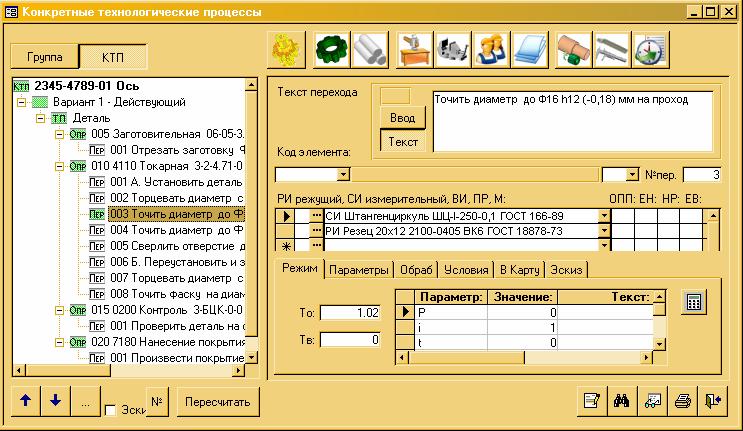


Рисунок 3.1 – Інтерфейс програми АНС САДКН

Програма була розрахована на самостійне індивідуальне навчання, час виконання завдань не обмежувалося. Проведені дослідження показали можливість одночасного формування при вирішенні навчальних завдань як способів пізнавальних дій, так і рефлексивних дій (таких, як самоконтроль, самокорекція).

## 3.2.4 АНС КАДИС

АНС КАДИС (Система Комплексів Автоматизованої Дидактичних Коштів) була створена в Центрі нових інформаційних технологій Самарського державного аерокосмічного університету в 90-х роках. Перші варіанти КАДИС були створені для IBM-сумісних комп'ютерів під управлінням операційної системи MS-DOS 3.1/3.11; в 1997 році була створена версія для Windows. КАДИС грунтувалася на концепції поділу навчального матеріалу на артікуліруемих та неартікуліруемих частину. До першої відносилися знання, які можна передати декларативним шляхом: інформацію у вигляді текстів, ілюстрацій, анімації, аудіо-та відеокліпів. До другої відносили досвід та інтуїцію, які могли бути освоєні тільки в результаті власної навчальної діяльності. Інтерфейс програми можна побачити на рисунку 3.2.

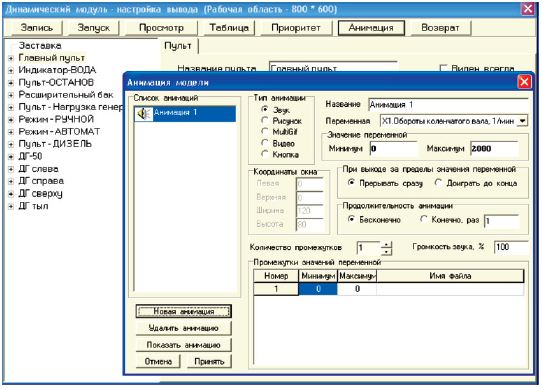


Рисунок 3.2 – Інтерфейс програми АНС КАДИС

Для навчання по другому типу в КАДИС були присутні тренажери і навчальні пакети прикладних програм. Технічно КАДИС складалася з чотирьох модулів: програвач УМК (забезпечував видачу навчального матеріалу з готові навчально-методичних комплексів), формування УМК (дозволяв формувати навчально-методичні комплекси з різних компонентів, що знаходяться на комп'ютері в електронному вигляді), тестова підсистема (забезпечувала створення тестових завдань , а також проведення тестів по них), онлайновий конвертор (готував матеріали навчально-методичних комплексів для розміщення в мережі Інтернет).

## 3.2.5 АНС ДЕЛЬФІН

АНС ДЕЛЬФІН розроблена в Центрі нових інформаційних технологій Московського енергетичного інституту (ЦНІТ МЕІ) і являє собою середовище створення навчальних курсів і програвач навчальних курсів. Інтерфейс програми можна побачити на рисунку 3.3.

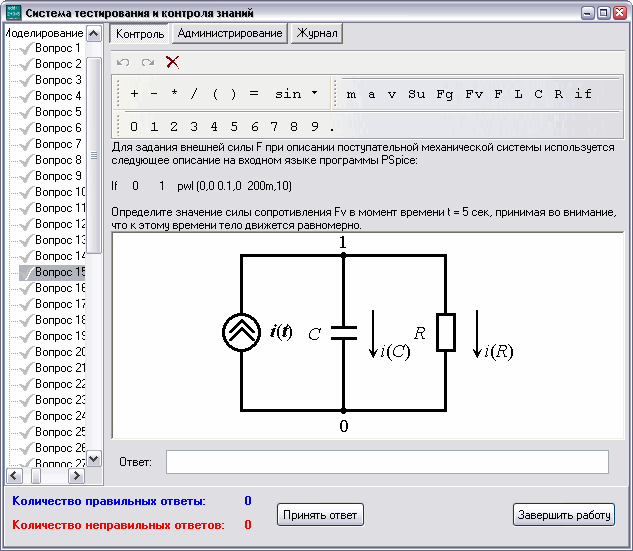


Рисунок 3.3 – Інтерфейс програми АНС ДЕЛЬФІН

АОС підтримує створення навчальних, контролюючих, тренувальних, довідково-консультаційних та інформаційних, що включають в себе мультимедійні технології. Програвач навчальних курсів підтримував вільний вибір навчальних курсів, перехід між ними, а також збирав і видавав статистику проходить навчання.

## 3.2.6 АНС Moodle

Найбільшу популярність придбала система управління навчанням Moodle (Modular Object-Oriented Dynamic Learning Environment). Вона призначена для створення ряду навчальних матеріалів (індивідуальних завдань і проектів для групових занять) та забезпечення спільного доступу до них. Вона була створена на основі систем управління вмістом сайту (Content Management System - CMS) - систем на мові PHP, які забезпечують управління великими сайтами Інтернету. Moodle підтримує ієрархію користувачів: творців курсів, викладачів та учнів, які мають різними правами в системі. Інтерфейс програми можна побачити на рисунку 3.4.

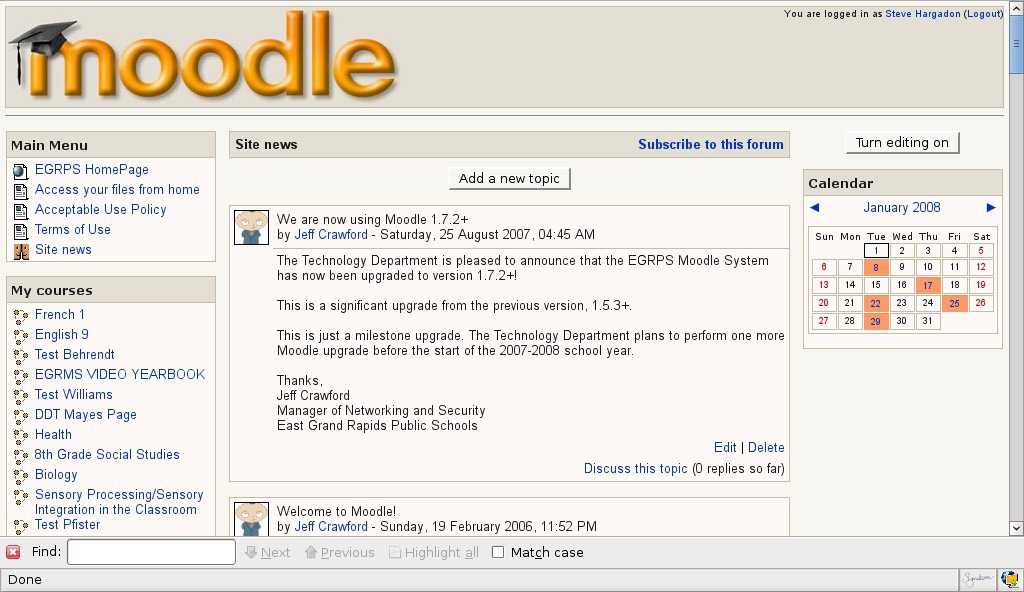


Рисунок 3.4 – Інтерфейс програми АНС Moodle

Навчальні курси включають не тільки текстову інформацію, але й зображення, презентації, тестові матеріали та файли різних форматів. Контрольні роботи мають обмеження за часом виконання і висилаються викладачам курсів, які перевіряють їх, виставляють оцінку і дають коментарі. Також підтримуються контрольні роботи-тести, які перевіряються автоматично.

## 3.3 Аналіз компонентів розроблюваної системи

У результаті аналізу можливостей великої кількості програм виявилося, що більшість їх призначене тільки для складання тестів і проведення тестування.

Можливості таких систем коливаються від найпростіших, що дозволяють тільки проводити тестування і зберігати його результати, а потім надавати їх викладачеві, до досить складних, забезпечених різними засобами аналізу результатів навчання (статистична обробка результатів), оформлення звітів за різними критеріями, установки параметрів питань (наприклад, коефіцієнт складності), параметрів тестів (наприклад, обмеження часу тестування), розмежування прав доступу і т.д.

Розрізняються системи та за типами питань: найчастіше мається можливість створювати тільки Одноваріантні питання (дається кілька варіантів відповідей, з яких тільки один є правильним). Більш складні системи дозволяють використовувати кілька типів питань: багатоваріантні (кілька варіантів відповідей, з яких правильних може бути кілька або всі), відкриті (учень повинен ввести відповідь сам), на знаходження відповідності (дається два списки, потрібна для кожного елемента першого списку встановити відповідний до нього елемент другого списку).

У даній роботі буде зроблена спроба створити засіб розробки навчальних систем, що об'єднує в собі всі три основні частини навчальних систем (теорія, контроль і тренаж), що має засоби обробки результатів тестування і вбудований вирішувач завдань, що має невеликий обсяг і що дозволяє вирішувати завдання з різних областей.

# Загальні відомості про реляційну алгебру

Бази даних завжди були найважливішою темою при вивченні інформаційних систем. Багато студентів вважають цей предмет приємним і нтересно, навіть незважаючи на його складність. Метою дисципліни є навчання студентів фундаментальним поняттям баз даних та систем управління базами даних, принципам проектування баз даних: будовою, методам і способам проектування, створення і роботи з базою даних і розробці прикладних програм для реляційних баз даних.

Концепція баз даних, рівні організації даних, життєвий цикл бази даних, реляційна модель даних, реляційна алгебра − все це входить в дисципліну Бази даних. У рамках дипломного проекту буде розглядатися маленький фрагмент цієї дисципліни − це реляційна алгебра.

Реляційна алгебра - замкнута система операцій над відносинами в реляційної моделі даних. Операції реляційної алгебри також називають реляційними операціями.

Основна ідея реляційної алгебри полягає в тому, що коли незабаром відносини є множинами, то кошти маніпулювання відносинами можуть базуватися на традиційних теоретико-множинних операціях, доповнених деякими спеціальними операціями, специфічними для баз даних.

Існує багато підходів до визначення реляційної алгебри, які розрізняються набором операцій і способами їх інтерпретації, але в принципі, більш-менш рівносильні. У цій роботі буде описано трохи розширений початковий варіант алгебри, який був запропонований Коддом. У цьому варіанті набір основних алгебраїчних операцій складається з восьми операцій, які діляться на два класи – теоретико-множинні операції та спеціальні реляційні операції.

До складу теоретико-множинних операцій входять операції:

* об'єднання відносин;
* перетинаняя відносин;
* взяття різниці відносин;
* прямого добутку відносин.

Спеціальні реляційні операції включають:

* обмеження відносини;
* проекцію відносини;
* підключення відносин;
* поділ відносин.

Крім того, до складу алгебри включається операція присвоювання, що дозволяє зберегти в базі даних результати обчислення алгебраїчних виразів, і операція перейменування атрибутів, що дає можливість коректно сформувати заголовок (схему) результуючого відносини.

## 4.1 Загальна інтерпретація реляційних операцій

Якщо не вдаватися в деякі тонкощі, які ми розглянемо в наступних підрозділах, то майже всі операції запропонованого вище набору володіють очевидною і простою інтерпретацією.

1. При виконанні операції об'єднання двох відносин здійснюється ставлення, що включає всі кортежі, що входять хоча б в одне з відносин-операндів.
2. Операція перетину двох відносин виробляє ставлення, що включає всі кортежі, що входять в обидва відносини-операнда.
3. Ставлення, що є різницею двох відносин включає всі кортежі, що входять у відношення - перший операнд, такі, що жоден з них не входить до відношення, яке є другим операндом.
4. При виконанні прямого твори двох відносин здійснюється ставлення, кортежі якого є конкатенації (зчепленням) кортежів першого і другого операндів.
5. Результатом обмеження відносини по деякому умові є ставлення, що включає кортежі відносини-операнда, яке задовольняє цій умові.
6. При виконанні проекції відносини на заданий набір його атрибутів виробляється ставлення, кортежі якого виробляються шляхом знаходження відповідних значень з кортежів відносини-операнда.
7. При з'єднанні двох відносин по деякому умові утворюється результуюче ставлення, кортежі якого є конкатенації кортежів першого і другого відносин і задовольняють цій умові.
8. У операції реляційного ділення два операнди - бінарне і Унарне відносини. Результуюче відношення складається з одноатрібутних кортежів, що включають значення першого атрибута кортежів першого операнда таких, що безліч значень другого атрибута (при фіксованому значенні першого атрибута) збігається з безліччю значень другого операнда.
9. Операція перейменування виробляє ставлення, тіло якого збігається з тілом операнда, але імена атрибутів змінені.
10. Операція присвоювання дозволяє зберегти результат обчислення реляційного вираження в існуючому відношенні БД.
11. Оскільки результатом будь-якої реляційної операції (крім операції привласнення) є деяке відношення, можна утворювати реляційні вирази, в яких замість відносини-операнда деякої реляційної операції знаходиться вкладене реляційне вираз.

## 4.2 Замкнутість реляційної алгебри і операція перейменування

Кожне відношення характеризується схемою (або заголовком) і набором кортежів (або тілом). Тому, якщо дійсно бажати мати алгебру, операції якої замкнуті щодо поняття відносини, то кожна операція повинна виробляти ставлення в повному розумінні, тобто воно повинно мати і тілом, і заголовком. Тільки в цьому випадку буде дійсно можливо будувати вкладені вирази.

Реляційна алгебра являє собою набір операторів, які використовують відносини в якості аргументів, і повертають відносини в якості результату. Таким чином, реляційний оператор виглядає як функція з відносинами в якості аргументів:

http://citforum.ru/database/dblearn/image135.gif.

Реляційна алгебра є замкнутою, тому що в якості аргументів на реляційні оператори можна підставляти інші реляційні оператори, які підходять по типу:

http://citforum.ru/database/dblearn/image136.gif.

Таким чином, в реляційних виразах можна використовувати вкладені вирази як завгодно складної структури.

Кожне відношення зобов'язане мати унікальне ім'я в межах бази даних. Ім'я відносини, отриманого в результаті виконання реляційної операції, визначається в лівій частині рівності. Однак можна не вимагати наявності імен від відносин, отриманих в результаті реляційних виразів, якщо ці відносини підставляються в якості аргументів на інші реляційні вирази. Такі відносини називаються неіменованого відносинами. Неіменовані відносини реально не існують в базі даних, а тільки обчислюються в момент обчислення значення реляційного оператора.

## 4.3 Операцій реляційної алгебри

Хоча в основі теоретико-множинної частини реляційної алгебри лежить класична теорія множин, відповідні операції реляційної алгебри володіють деякими особливостями.

Почнемо з операції об'єднання (все, що буде говоритися з приводу об'єднання, переноситься на операції перетину і взяття різниці). Сенс операції об'єднання в реляційної алгебри в цілому залишається теоретико-множинним. Але якщо в теорії множин операція об'єднання осмислена для будь-яких двох множин-операндів, то у випадку реляційної алгебри результатом операції об'єднання повинно бути ставлення. Якщо допустити в реляційної алгебри можливість теоретико-множинного об'єднання довільних двох відносин (з різними схемами), то, звичайно, результатом операції буде безліч, але безліч різнотипних кортежів, тобто не ставлення. Якщо виходити з вимоги замкнутості реляційної алгебри щодо поняття відносини, то така операція об'єднання є безглуздою.

Всі ці міркування приводять до появи поняття сумісності відносин з об'єднання: два відносини сумісні з об'єднання в тому і тільки в тому випадку, коли володіють однаковими заголовками. Більш точно, це означає, що в заголовках обох відносин міститься один і той же набір імен атрибутів, і однойменні атрибути визначені на одному і тому ж домені.

Якщо два відносини сумісні з об'єднання, то при звичайному виконанні над ними операцій об'єднання, перетину і взяття різниці результатом операції є ставлення з корректно певним заголовком, що збігається із заголовком кожного з відносин-операндів. Нагадаємо, що якщо два відносини "майже" сумісні з об'єднання, тобто сумісні у всьому, окрім імен атрибутів, то до виконання операції типу з'єднання ці відносини можна зробити повністю сумісними з об'єднання шляхом застосування операції перейменування.

Зауважимо, що включення до складу операцій реляційної алгебри трьох операцій об'єднання, перетину і взяття різниці є очевидно надлишковим, оскільки відомо, що будь-яка з цих операцій виражається через дві інших. Тим не менш, Кодд свого часу вирішив включити всі три операції, виходячи з інтуїтивних потреб потенційного користувача системи реляційних БД, далекого від математики.

Інші проблеми пов'язані з операцією взяття прямого твори двох відносин. У теорії множин пряме твір може бути отримано для будь-яких двох множин, та елементами результуючого безлічі є пари, складені з елементів першого і другого множин. Оскільки відносини є множинами, то і для будь-яких двох відносин можливе отримання прямого твори. Але результат не буде відношенням. Елементами результату будуть не кортежі, а пари кортежів.

Тому в реляційної алгебри використовується спеціалізована форма операції взяття прямого добутку – розширене пряме твір відносин. При взяття розширеного прямого твори двох відносин елементом результуючого відносини є кортеж, що є конкатенації (або злиттям) одного кортежу першого відношення і одного кортежу другого відношення.

Але тепер важливо отримати коректно сформований заголовок відносини-результату. Очевидно, що проблемою може бути іменування атрибутів результуючого відносини, якщо відносини-операнди володіють однойменними атрибутами.

Ці міркування приводять до появи поняття сумісності з узяття розширеного прямого твори. Два відносини сумісні з узяття прямого твори в тому і тільки в тому випадку, якщо безлічі імен атрибутів цих відносин не перетинаються. Будь-які два відносини можуть бути зроблені сумісними з узяття прямого твори шляхом застосування операції перейменування до одного з цих відносин.

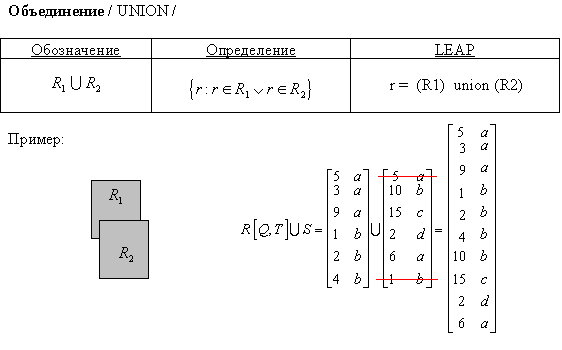
Слід зауважити, що операція взяття прямого твори не є занадто осмисленої на практиці. По-перше, потужність її результату дуже велика навіть при допустимих потужностях операндів, а по-друге, результат операції не більше інформативний, ніж взяті в сукупності операнди. Як ми побачимо трохи нижче, основний сенс включення операції розширеного прямого твори до складу реляційної алгебри полягає в тому, що на її основі визначається справді корисна операція з'єднання.

З приводу теоретико-множинних операцій реляційної алгебри слід ще зауважити, що всі чотири операції є асоціативними. Т. е., якщо позначити через OP будь-яку з чотирьох операцій, то (A OP B) OP C = A (B OP C), і отже, без введення двозначності можна писати A OP B OP C (A, B і C - відносини, що мають властивості, необхідними для коректного виконання відповідної операції). Всі операції, крім взяття різниці, є комутативними, тобто A OP B = B OP A.

## 4.3.1 Об'єднання

Об'єднанням двох сумісних по типу відносин R1 і R2 називається відношення з тим же заголовком, що і у відносин R1 і R2, і тілом, що складається з кортежів, що належать або R1, або R2, або обом відносинам(див. рис. 4.1).

Синтаксис операції об'єднання: R1 UNION R2.



Об'єднання

Приклад:

Позначення

Визначення

Рисунок 4.1 – Об'єднання відносін

Об'єднання, як і будь-яке відношення, не може містити однакових кортежів. Тому, якщо деякий кортеж входить і у відношення R1, і ставлення R2, то в об'єднання він входить один раз.

Як видно з наведеного прикладу, потенційні ключі, які могли бути у відносинах R1 і R2 не успадкують об'єднанням цих відносин. Тому, вони можуть містити дублікати значень. Якби це було не так, і ключі успадковувалися б, то це суперечило б поняттю об'єднання як "об'єднання множин". Звичайно, об'єднання відносин R1 і R2 має, як і будь-яке відношення, потенційний ключ, наприклад, що складається з усіх атрибутів.

## 4.3.2 Перетинання

Перетином двох сумісних по типу відносин R1 і R2 називається відношення з тим же заголовком, що і у відносин R1 і R2, та тілом, що складається з кортежів, що належать одночасно обом відносинам R1 і R2 (див. рис. 4.2).

Синтаксис операції перетину: R1 INTERSECT R2.

Перетинання

Позначення

Визначення

Приклад:

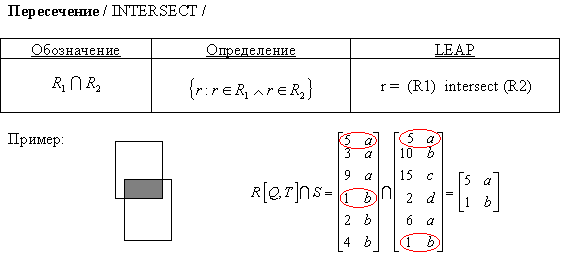


Рисунок 4.2 – Перетинання відносін

Здавалося б, що на відміну від операції об'єднання, потенційні ключі могли б успадковуватися перетинанням відносин. Однак це не так. Взагалі, ніякі реляційні оператори не передають результатірующій відношенню ніяких даних про потенційних ключах. В якості причини цього можна було б навести тривіальне міркування, що так виходить більш просто і симетрично - всі оператори влаштовані однаково. Насправді причина більш глибока, і полягає в тому, що потенційний ключ - семантичне поняття, що відображає различимость об'єктів предметної області. Наявність потенційних ключів не виводиться зі структури відносини, а явно задається для кожного відносини, виходячи з його змісту. Реляційні ж оператори є формальними операціями над відносинами і виконуються однаково, незалежно від змісту даних, що містяться у відносинах.

## 4.3.3 Різниця

Відніманням двох сумісних по типу відносин R1 і R2 називається відношення з тим же заголовком, що і у відносин R1 і R2, і тілом, що складається з кортежів, що належать відношенню R1 і не належать відношенню R2(див. рис. 4.3).

Синтаксис операції віднімання: R1 MINUS R2.

Різниця

Позначення

Визначення

Приклад:

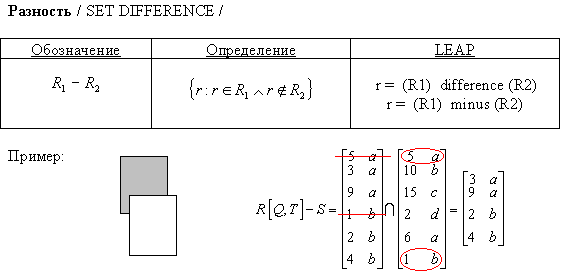


Рисунок 4.3 – Різниця відносін

Операція в чому схожа на INTERSECT, за винятком того, що в результуючому відношенні містяться кортежі, присутні в першому і відсутні в другому вихідних відносинах.

## 4.3.4 Декартовий добуток

Декартовим твором двох відносин R1(R11, R12,…, R1n) и R2(R21, R22,…, R2m) називається відношення, заголовок якого є зчепленням заголовків відносин R1 и R2:

(R11, R12,… ,R1n, R21, R22,…, R2m),

а тіло складається з кортежів, які є зчепленням кортежів відносин R1и R2

(r11, r12,…,r1n, r21, r22,…,r2m),

таких, що (r11, r12,…,r1n) ∈ R1, (r21, r22,…,r2m) ∈ R2(див. рис. 4.4).

Синтаксис операції декартового твори: R1 TIMES R2

Потужність добутку R1 TIMES R2 дорівнює добутку потужностей відносин R1 і R2, тому що кожен кортеж відношення R1 з'єднується з кожним кортежем відносини R2.

Якщо у відносини R1 і R2 є атрибути з однаковими найменуваннями, то перед виконанням операції декартового твори такі атрибути необхідно перейменувати.

Перемножать можна будь-які два відношення, сумісність за типом при цьому не вимагається.

Сама по собі операція декартового твори не дуже важлива, тому що вона не дає ніякої нової інформації, у порівнянні з вихідними відносинами. Для реальних запитів ця операція майже ніколи не використовується. Проте операція декартового твори важлива для виконання спеціальних реляційних операцій, про які йтиметься нижче.

Декартовий добуток

Позначення

Визначення

Приклад:

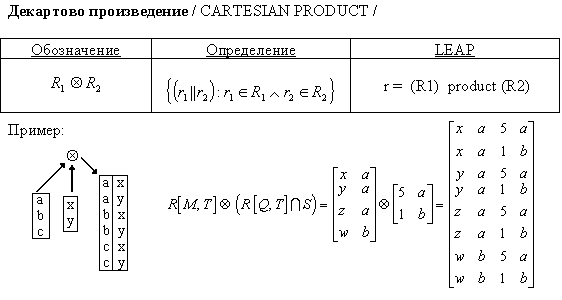


Рисунок 4.4 – Декартовий добуток відносин

Операція обмеження вимагає наявності двох операндів: обмежувати відносини і простого умови обмеження.

В результаті виконання операції обмеження виробляється ставлення, заголовок якого співпадає з заголовком відносини-операнда, а в тіло входять ті кортежі відносини-операнда, для яких значенням умови обмеження є true.

З використанням цих визначень можна використовувати операції обмеження, в яких умовою обмеження є довільне Булевського вираз, складене з простих умов з використанням логічних зв'язок AND, OR, NOT і дужок.

На інтуїтивному рівні операцію обмеження найкраще представляти як взяття деякої "горизонтальної" вирізки з відносини-операнда.

## 4.4.5 Вибірка

Вибіркою (обмеженням, селекцією) на відношенні R з умовою v називається відношення з тим же заголовком, що і у відносини R, і тілом, що складається з кортежів, значення атрибутів яких при підстановці в умову v дають значення TRUE. v являє собою логічне вираження, до якого можуть входити атрибути відношення R і (або) скалярні вирази.

У найпростішому випадку умова v має вигляд A1 θ A2, где θ – один з операторів порівняння (=, ≠, <, ≤,>, ≥ і т.д.), а A1 і A2 – атрибути відносини R або скалярні значення(див. рис. 4.5).

Синтаксис операції вибірки: R WHERE v або R WHERE A1 θ A2.

Вибірка

Позначення

Визначення

Приклад:

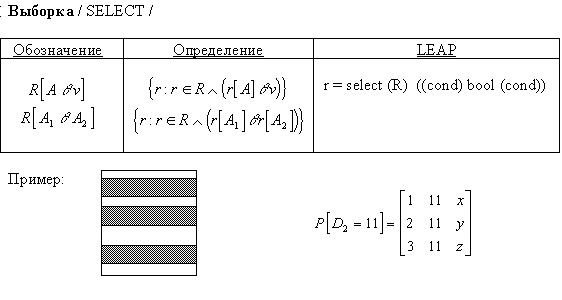


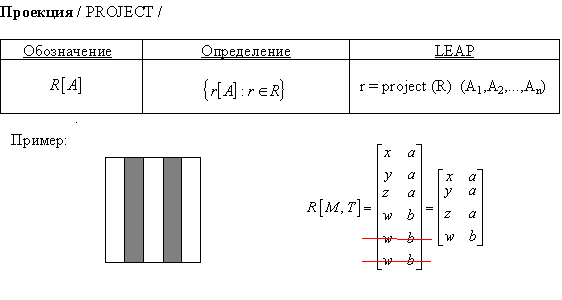
Рисунок 4.5 – Вибірка відносин

Сенс операції вибірки очевидний - вибрати кортежі відносини, що задовольняють деякій умові. Таким чином, операція вибірки дає "горизонтальний зріз" відносини по деякому умові.

## 4.4.6 Проекція

Проекцією відношення R за атрибутами A, де кожен з атрибутів належить відношенню R, називається відношення із заголовком A і тілом, що містить безліч кортежів виду a, таких, для яких у відношенні R знайдуться кортежі зі значенням атрибута A рівним a(див. рис. 4.6).

Синтаксис операції проекції: R [A].



Проекція

Позначення

Визначення

Приклад:

Рисунок 4.6 – Проекція відносин

Операція проекції дає "вертикальний зріз" відносини, в якому видалені всі виниклі при такому зрізі дублікати кортежів.

## 4.4.7 З'єднання

Нехай дано відношення R1(R11, R12,…, R1n, А1, А2,..., Аk) і R2(R21, R22,…, R2m, А1, А2,..., Аk) і, що мають однакові атрибути А1, А2,..., Аk (тобто атрибути з однаковими іменами і визначені на однакових доменах).

Тоді природним з'єднанням відносин R1 і R2 називається відношення із заголовком (R11, R12,…, R1n, А1, А2,..., Аk, R21, R22,…, R2m) і тілом, що містить безліч кортежів (r11, r12,…, r1n, a1, a2,..., ak, r21, r22,…, r2m), таких, що (r11, r12,…, r1n, a1, a2,..., ak) ∈ R1 и (a1, a2,..., ak, r21, r22,…, r2m) ∈ R2(див. рис. 4.7).

Тут додано умова задовольняють опредленних співвідношенню між атрибутами з'єднання (А1, A2) відповідних відносин.

Природне з'єднання настільки важливо, що для нього використовують спеціальний синтаксис: R1 JOIN R2.

З'єднання

Позначення

Визначення

Приклад:

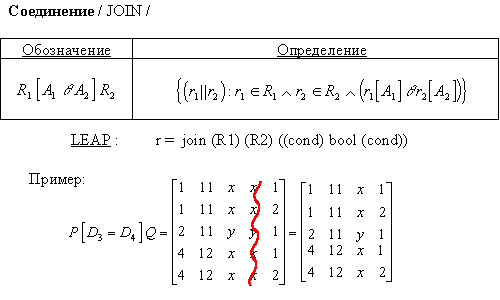


Рисунок 4.7 – З'єднання відносин

У синтаксисі природного з'єднання не вказуються, за яким атрибутам проводиться з'єднання. Природне з'єднання проводиться по всіх однаковим атрибутам.

Природне з'єднання еквівалентно такій послідовності реляційних операцій:

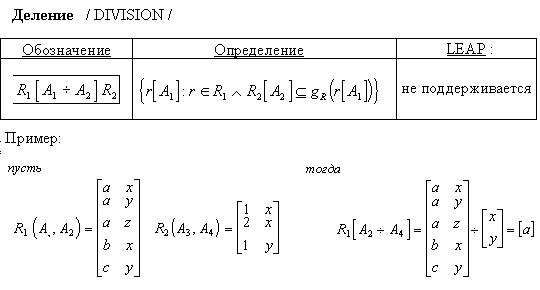
1. Перейменувати однакові атрибути у відносинах.
2. Виконати декартовій твір відносин.
3. Виконати вибірку за співпадаючими значеннями атрибутів, що мали однакові імена.
4. Виконати проекцію, видаливши повторювані атрибути.
5. Перейменувати атрибути, повернувши їм початкові імена.

## 4.4.8 Ділення

Нехай дано відношення R1(R11, R12,…, R1n, А1, А2,..., Аk) и R2(А1, А2,..., Аk), причому атрибути А1, А2,..., Аk − загальні для двох відносин.

Розподілом відносин R1 на R2 називається відношення із заголовком (R11, R12,…, R1n) і тілом, що містить безліч кортежів (r11, r12,…, r1n), таких, що для всіх кортежів (a1, a2,..., ak) ∈ R2 щодо R1 знайдеться кортеж (r11, r12,…, r1n, a1, a2,..., ak) (див. рис. 4.8).

Синтаксис операції ділення: R1 DEVIDBY R2.



Ділення

Позначення

Визначення

Приклад:

*нехай*

*тоді*

не підтримується

Рисунок 4.8 – Ділення відносин

Ставлення R1 виступає в ролі діленого, ставлення R2 виступає в ролі дільника. Ділення відносин аналогічно діленню чисел із залишком.

Типові запити, реалізовані за допомогою операції ділення, зазвичай у своїй формулюванні мають слово "все".

# Вибір технологій та способів реалізації

Описаний в цій роботі проект був розроблений з використанням об'єктно-орієнтованої технології. Ця технологія широко користується попитом на даний момент.

В якості мови програмування в даній роботі використовується Java. Дана мова широко поширена і дозволяє створити об'єктно-орієнтованих систем.

Організації процесу розробки програмного забезпечення вимагає використання ряду засобів (див. рис. 5.1).

Тести

Віртуальний лабораторний комплекс

Реляційні операції

Java, Spring

HTML, CSS, JavaScript, jQuery, Twitter Bootstrap

Рисунок 5.1 − Структура використовуваних засобів розробки проекту

Віртуальний лабораторний комплекс "Побудова запитів до реляційної БД" мультиплатформений застосування, який може працювати на будь-яких операційних системах з браузером.

Для створення тестів лабораторного комплексу був обраний об'єктно-орієнтована мова програмування Java і Spring - бібліотека для створення графічного інтерфейсу для програм на Java. Для створення головної частини була вибрана мова гіпертекстової розмітки документа HTML. Для оформлення головного меню використовувалися каскадні таблиці стилів - CSS. Мова програмування JavaScript застосований для його анімації. Twitter Bootstrap для оформлення для веб-форм, кнопок, міток, блоків навігації і інших компонентів веб-інтерфейсів. А так само бібліотека jQuery що фокусується на взаємодії JavaScript і HTML.

Описаний набір засобів мінімальний, але його використання дозволяє зробити розробку програмного забезпечення наближеною до реального технологічного процесу. Використання цих засобів дозволяє отримати досвід, який згодом може використовуватися при розробці складних проектів.

У цій главі буде розглянуто застосування всіх перерахованих вище засобів. При цьому розгляд буде обмежено тими функціями, які безпосередньо необхідні для реалізації проекту.

## 5.1 Використання Java і бібліотеки Spring для написання тестів лабораторного комплексу

## 5.1.1 Java

Теcти для проекту реалізовані за допомогою мови Java 1.6.-об'єктно-орієнтована мова програмування, розроблений компанією Sun Microsystems. Програми Java зазвичай компілюються в спеціальний байт-код, тому вони можуть працювати на будь віртуальній Java-машині (JVM) незалежно від комп'ютерної архітектури. Гідність подібного способу виконання систем - в повній незалежності байт-коду від операційної системи та обладнання, що дозволяє виконувати Java-додатки на будь-якому пристрої, для якого існує відповідна віртуальна машина. Іншою важливою особливістю технології Java є гнучка система безпеки завдяки тому, що виконання системи повністю контролюється віртуальною машиною. Будь-які операції, які перевищують встановлені повноваження програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером) викликають негайне переривання.

Крім багатоплатформеності, Java має ще одне безперечне достоїнство: орієнтація на об'єктно-орієнтований підхід програмування. Це дозволяє найбільш швидко, ефективно і якісно здійснити розробку системи, а також її подальшу підтримку.

Тому Java була обрана в якості мови для реалізації автоматизованої системи.

## 5.1.2 Spring

В якості фреймворку для програми був обраний Spring Framework - один з кращих фреймворків для побудови enterprise-додатків на Java з хорошою документацією і великим співтовариством розробників.

Spring Framework - універсальний фреймворк з відкритим вихідним кодом для Java-платформи.

Фреймворк був вперше випущений під ліцензією Apache 2.0 license в червні 2003 року. Поточна версія - 3.2.2.

Spring Framework надає велику свободу Java-розробникам в проектуванні, крім того, він надає добре документовані і легкі у використанні засоби вирішення проблем, що виникають при створенні додатків корпоративного масштабу.

Тим часом, особливості ядра Spring Framework застосовні в будь-якому Java-додатку, і існує безліч розширень і удосконалень для побудови веб-додатків на Java Enterprise платформі. З цих причин Spring придбав велику популярність.

Spring Framework забезпечує вирішення багатьох завдань, з якими стикаються Java розробники та організації, які хочуть створити інформаційну систему, засновану на платформі Java. Через широкої функціональності важко визначити найбільш значущі структурні елементи, з яких він складається. Spring Framework НЕ повністю пов'язані з платформою Java Enterprise, незважаючи на його масштабну інтеграцію з нею, що є важливою причиною його популярності.

Цей фреймворк пропонує послідовну модель і робить її застосовної до більшості типів додатків, які вже створені на основі платформи Java. Вважається, що Spring Framework реалізує модель розробки, засновану на кращих стандартах індустрії, і робить її доступною в багатьох областях Java.

З причини великої кількості реалізованих фіч і динаміки свого розвитку на мій погляд Spring є фактично ідеальним варіантом enterprise фреймворку.

## 5.2 Використання HTML, CSS, JavaScript та бібліотеки jQuery для написання тестів

## HTML

Для створення головної частини була вибрана мова гіпертекстової розмітки документа HTML

HTML − стандартна мова розмітки документів у Всесвітній павутині. Більшість веб-сторінок створюються за допомогою мови HTML (або XHTML). Мова HTML інтерпретується браузерами і відображається у вигляді документа в зручній для людини формі.

Спочатку мова HTML був задуманий і створений як засіб структурування та форматування документів без їх прив'язки до засобів відтворення (відображення). В ідеалі, текст з розміткою HTML повинен був без стилістичних та структурних спотворень відтворюватися на обладнанні з різною технічною оснащеністю (кольоровий екран сучасного комп'ютера, монохромний екран органайзера, обмежений за розмірами екран мобільного телефону або пристрою та програми голосового відтворення текстів). Проте сучасне застосування HTML дуже далеко від його початкової задачі. Наприклад, тег <TABLE>, кілька разів використаний для форматування сторінки, яку ви на даний момент читаєте, призначений для створення в документах самих звичайних таблиць, але, як можна переконатися, тут немає жодної таблиці. З плином часу основна ідея платформонезалежних мови HTML була принесена в жертву сучасним потребам у мультимедійному і графічному оформленні.

## 5.2.2 CSS

Для оформлення головного меню використовувалися каскадні таблиці стилів CSS.

CSS − формальна мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки.

Переважно використовується як засіб опису, оформлення зовнішнього вигляду веб-сторінок, написаних за допомогою мов розмітки HTML і XHTML, але може також застосовуватися до будь-яких XML-документами, наприклад, до SVGілі XUL.

CSS використовується творцями веб-сторінок для завдання кольорів, шрифтів, розташування окремих блоків та інших аспектів представлення зовнішнього вигляду цих веб-сторінок. Основною метою розробки CSS було розділення опису логічної структури веб-сторінки (яке проводиться за допомогою HTML або інших мов розмітки) від опису зовнішнього вигляду цієї веб-сторінки (яке тепер проводиться за допомогою формальної мови CSS). Такий поділ може збільшити доступність документа, надати велику гнучкість і можливість управління його поданням, а також зменшити складність і повторюваність в структурному вмісті. Крім того, CSS дозволяє представляти один і той же документ в різних стилях або методах виведення, таких як екранне уявлення, друковане подання, читання голосом (спеціальним голосовим браузером або програмою читання з екрану), або при виведенні пристроями, що використовують шрифт Брайля.

Правила CSS пишуться на формальному мові CSS і розташовуються в таблицях стилів, тобто таблиці стилів містять в собі правила CSS. Ці таблиці стилів можуть розташовуватися як у самому веб-документі, зовнішній вигляд якого вони описують, так і в окремих файлах, які мають формат CSS. (По суті, формат CSS - це звичайний текстовий файл. У файлі. Css не міститься нічого, крім переліку правил CSS та коментарів до них.) Тобто, ці таблиці стилів можуть бути підключені, впроваджені в описуваний ними веб-документ чотирма різними способами :

1. коли таблиця стилів знаходиться в окремому файлі, вона може бути підключена до веб-документу за допомогою тега <link>, розташованого в цьому документі між тегами <head> і </head>. (Тег <link> буде мати атрибут href, що має значенням адресу цієї таблиці стилів). Всі правила цієї таблиці діють протягом усього документа;
2. коли таблица стилей находится в отдельном файле, она может быть подключена к веб-документу посредством директивы @import, располагающейся в этом документе между тегами <style> и </style> (которые, в свою очередь, располагаются в этом документе между тегами <head> и </head>) сразу после тега <style>, которая также указывает (в своих скобках, после слова url) на адрес этой таблицы стилей. Все правила этой таблицы действуют на протяжении всего документа;
3. коли таблиця стилів описана в самому документі, вона може розташовуватися в ньому між тегами <style> і </style> (які, у свою чергу, розташовуються в цьому документі між тегами <head> і </head>). Всі правила цієї таблиці діють протягом усього документа;
4. коли таблиця стилів описана в самому документі, вона може розташовуватися в ньому в тілі якогось окремого тега (за допомогою його атрибуту style) цього документа. Всі правила цієї таблиці діють тільки на вміст цього тега.

У перших двох випадках говорять, що до документа застосовані зовнішні таблиці стилів, а в других двох випадках - внутрішні таблиці стилів.

Всі ці випадки були використані для написанні тестів для віртуального лабораторного комплексу.

## 5.2.3 JavaScript

Мова програмування JavaScript застосованиа для анімації.

JavaScript − прототипна-орієнтований сценарний мову програмування. Є діалектом мови ECMAScript.

JavaScript зазвичай використовується як вбудовувана мова для програмного доступу до об'єктів додатків. Найбільш широке застосування знаходить в браузерах як мова сценаріїв для додання інтерактивності веб-сторінок.

Основні архітектурні риси: динамічна типізація, слабка типізація, автоматичне керування пам'яттю, прототипна програмування, функції як об'єкти першого класу.

На JavaScript вплинули багато мов, при розробці була мета зробити мову схожим на Java, але при цьому легким для використання непрограмістів. Мовою JavaScript не володіє будь-яка компанія або організація, що відрізняє його від ряду мов програмування, використовуваних у веб-розробці.

Для додавання JavaScript-коду на сторінку, можна використовувати теги <script> </script>, які рекомендується, але не обов'язково, поміщати всередині контейнера <head>. Контейнерів <script> в одному документі може бути скільки завгодно. Атрибут «type = 'text/javascript'» вказувати необов'язково, оскільки за умовчанням стоїть javascript.

JavaScript використовується в клієнтської частини веб-додатків: клієнт-серверних програм, в якому клієнтом виступає браузер, а сервером - веб-сервер, що мають розподілену між сервером і клієнтом логіку. Обмін інформацією у веб-додатках відбувається по мережі. Однією з переваг такого підходу є той факт, що клієнти не залежать від конкретної операційної системи користувача, тому веб-додатки є кроссплатформенную сервісами.

## 5.2.4 jQuery

jQuery − бібліотека JavaScript, що фокусується на взаємодії JavaScript і HTML. Бібліотека jQuery допомагає легко отримувати доступ до будь-якого елементу DOM, звертатися до атрибутів і вмісту елементів DOM, маніпулювати ними. Також бібліотека jQuery надає зручний API по роботі з AJAX.

Точно так само, як CSS відокремлює візуалізацію від структури HTML, JQuery відокремлює поведінку від структури HTML. Наприклад, замість прямої вказівки на обробник події натискання кнопки управління передається JQuery, яка ідентифікує кнопки і потім перетворює його в обробник події кліка. Такий поділ поведінки і структури також називається принципом ненав'язливого JavaScript.

Бібліотека jQuery містить функціональність, корисну для максимально широкого кола завдань. Тим не менш, розробниками бібліотеки не ставилося завдання суміщення в jQuery функцій, які підійшли б усюди, оскільки це призвело б до великого кодом, більша частина якого не затребувана. Тому була реалізована архітектура компактного універсального ядра бібліотеки і плагінів. Це дозволяє зібрати для ресурсу саме ту JavaScript-функціональність, яка на ньому була б затребувана.

Вся робота з jQuery ведеться за допомогою функції $. Якщо на сайті застосовуються інші JavaScript бібліотеки, де $ може використовуватися для своїх потреб, то можна використовувати її синонім − jQuery.

Роботу з jQuery можна розділити на 2 типи:

1) Отримання jQuery-об'єкта за допомогою функції $(). Наприклад, передавши в неї CSS-селектор, можна отримати jQuery-об'єкт всіх елементів HTML, що потрапляють під критерій і далі працювати з ними за допомогою різних методів jQuery-об'єкта. У разі, якщо метод не повинен повертати якого-небудь значення, він повертає посилання на jQuery об'єкт, що дозволяє вести ланцюжок викликів методів згідно концепції текучого інтерфейсу.

2) Виклик глобальних методів у об'єкту $, наприклад, зручних ітераторів по масиву.

Методи, що починаються з $., Зручно застосовувати для обробки глобальних об'єктів

## 5.2.5 Twitter Bootstrap

Twitter Bootstrap - вільний набір інструментів для створення сайтів і веб-додатків. Включає в себе HTML і CSS шаблони оформлення для типографіки, веб-форм, кнопок, міток, блоків навігації і інших компонентів веб-інтерфейсів, включаючи JavaScript розширення.

# Опис програмної реалізації

## 6.1 Інтерфейс програми

Перед початком розробки програмного забезпечення була поставлена ціль спроектувати дуже простий, та у той же час практичний інтерфейс користувача. Головне вікно програми розділено на блоки, елементи в яких розміщені за функціональними цілями. Схема розміщення блоків елементів головного вікна показана на рисунку 6.1.



Рисунок 6.1 – Схема головного вікна програми

Одним з найбільш популярних варіантів верстки веб-сторінок виступає двоколонковому макет, де в лівій колонці розташовуються посилання, а в правій колонці - основний контент. Така схема досить традиційна і звична для користувачів, тому користується визнанням серед розробників.

Створення двох колонок відбувається за допомогою стильової властивості float із значенням left, яке застосовується для лівої колонки. При цьому одночасно слід встановити відступ зліва для правої колонки через margin-left, значення якого дорівнює ширині лівої колонки плюс відстань між ними.

Щоб макет розташовувався по центру веб-сторінки, був доданий шар з ім'ям container і колонки розташуємо всередині нього. Для самого container необхідно встановити ширину (width) і вирівнювання по центру (margin: 0 auto).

У лівій колонці можна вибрати реляційну операцію з якою треба попрацювати(див. рис 6.2). Для вибору доступні наступні реляційні операції:

* об'єднання;
* перетинання;
* різниця;
* добуток;
* вибірка;
* проекція;
* з'єднання;
* поділ.

Також можна вибрати інші пункти:

* про програму;
* тести.

У правій колонці розташовується опис реляційної алгебри по вкладці «Про програму», але коли здійснюється перехід по реляційним операціям, то права колонка поділяється на два блока(див. рис. 6.3):

* опис;
* практикум.

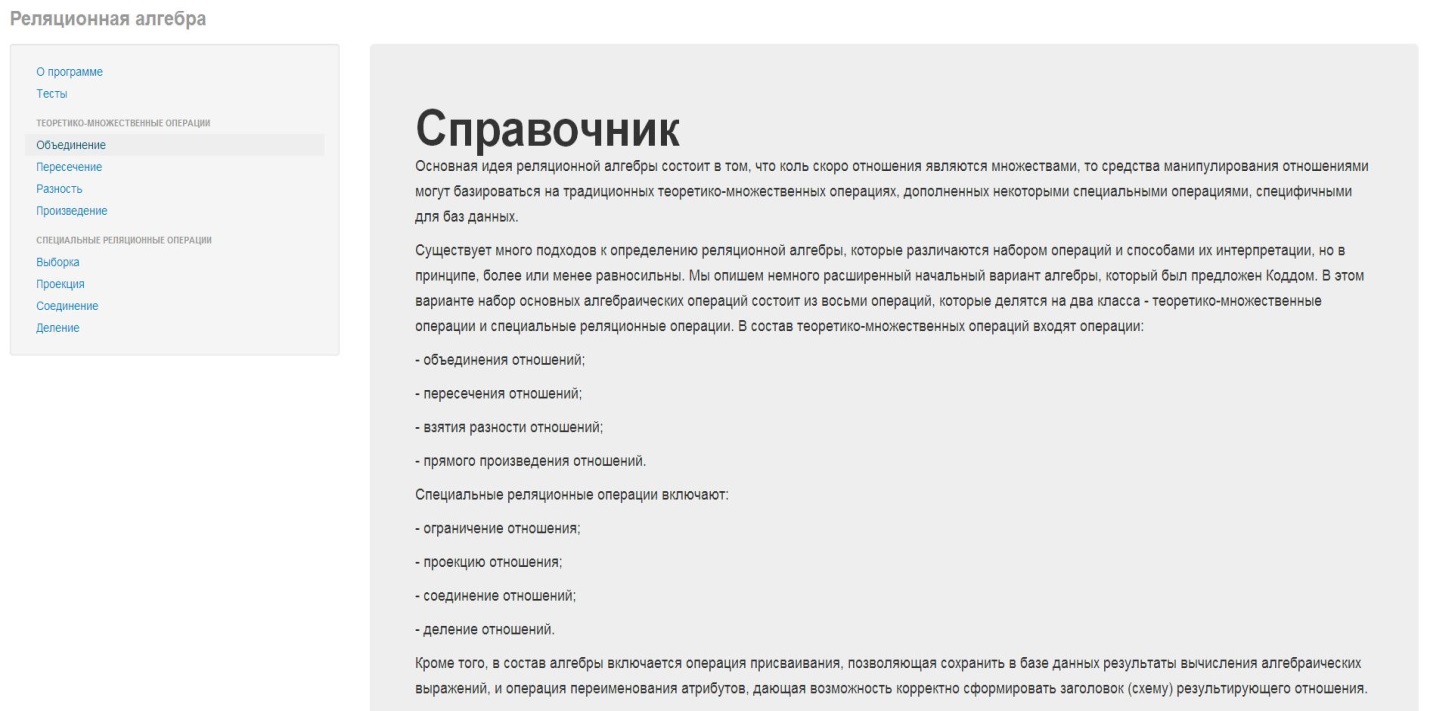


Рисунок 6.2 – Головне вікно програми

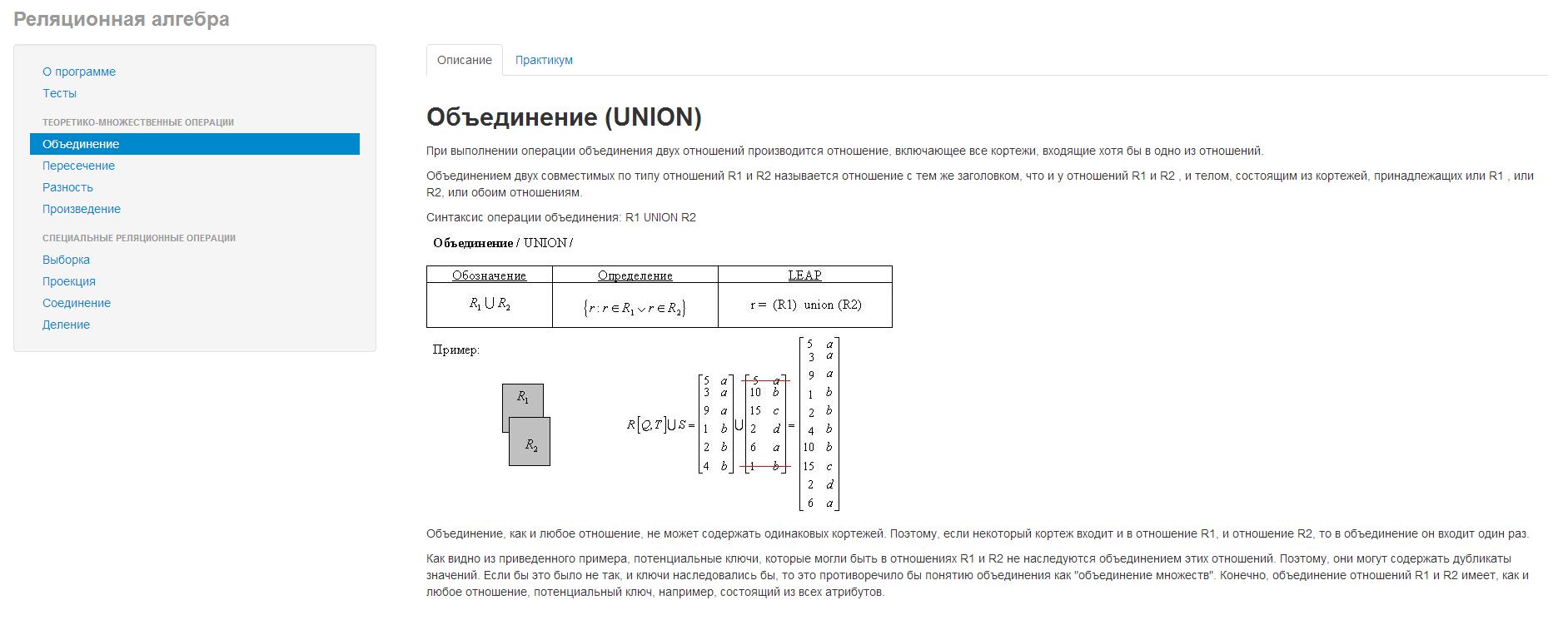


Рисунок 6.3 – Вид вкладки «Опис» у програмі

По вкладці «Опис» можна прочитати принцип роботи обраної реляційної операції, потім перейти на вкладку «Практикум» і спробувати цю реляційну операцію в дії на справжніх таблицях(див. рис 6.4).

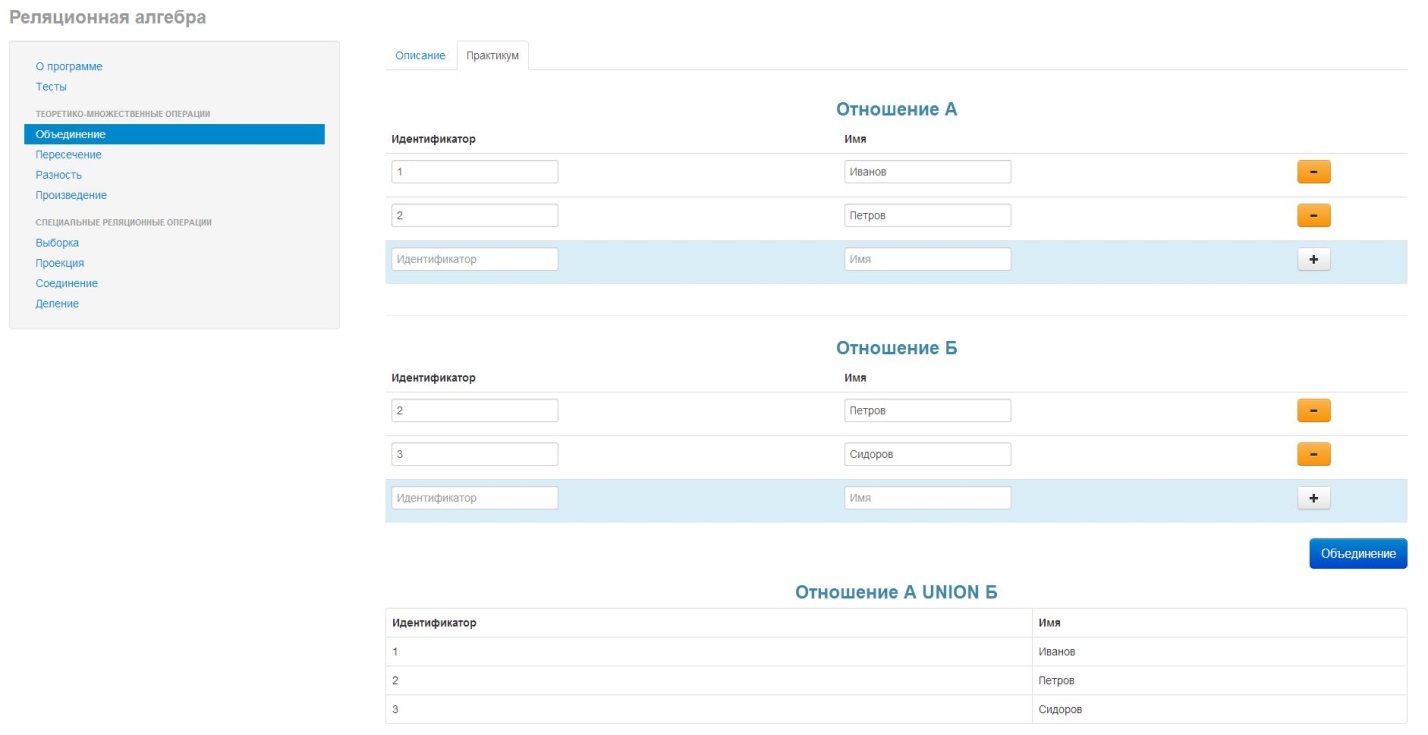


Рисунок 6.4 – Вид вкладки «Практикум» у програмі

Після того як зрозумілий принцип роботи оператора реляційної алгебри, можна перейти на вкладку «Тести» та пройти тест з даної теми.

Після переходу по заданой вкладці можна побачити вікно авторизації (див.рис. 6.5).



Рисунок 6.5 – Вікно авторизації

Якщо користувач зайшов під обліковими даними студента, то йому доступні кілька варіантів тестів. У даному випадку це два тести:

* Реляційна алгебра. Базовий тест;
* Реляційна алгебра. Розширений тест.

Інтерфейс програми можна побачити на рисунку 6.6.

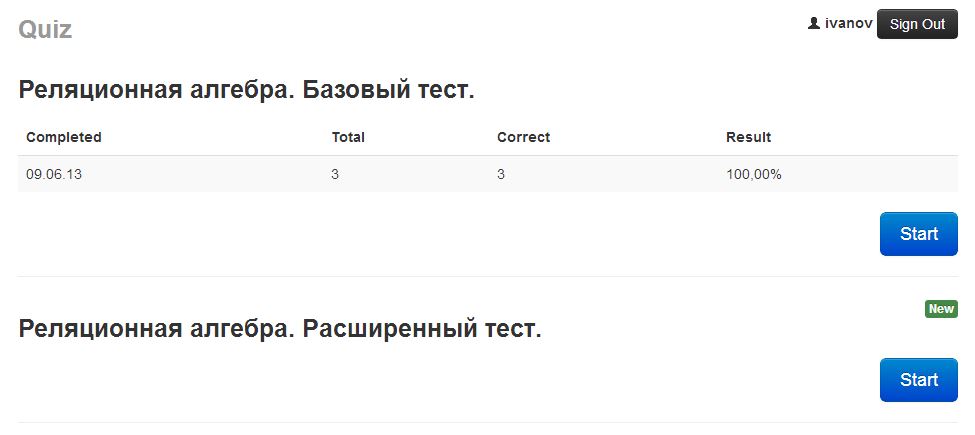


Рисунок 6.6 – Вікно з тестами під обліковими даними студента

Тут можна побачити результати з пройденого тесту, а так само пройти тест ще раз. Якщо користувач один і той же тест проходить кілька разів, то зберігається кращий результат. Поміткою «New» біля тесту, позначає що тест новий і не проходився ще жодного разу(див. рис. 6.7).

Якщо користувач зайшов під обліковими даними викладача, то йому доступні списки студентів, які пройшли тести.

Ці списки містять:

* ПІБ студента;
* групу;
* дату проходження тесту;
* кількість питань;
* кількість правильних відповідей;
* результат у відсотковому співвідношенні.

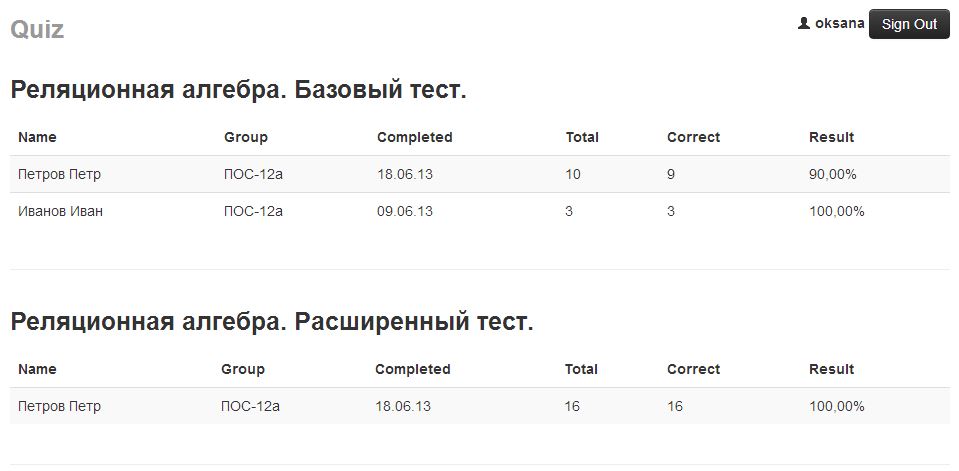


Рисунок 6.7 – Вікно з тестами під обліковими даними викладача

## 6.2 Опис структури системи, класів та функцій

Застосування для тестування побудовано на основі багаторівневої архіетктури з використанням каркаса Spring.

Ця архітектура припускає, що кожен модуль розташований на певному рівні і може використовувати модулі, які знаходяться на нижчих рівнях.

Кількість і призначення рівнів залежить від вирбанного каркаса.

Наприклад, при використанні Spring можна виділити чотири рівні(див. рис. 6.8).

Першим є рівень представлення, який безпосередньо представлений і на стороні сервера і на стороні клієнта.

На стороні сервера це .jsp сторінки, а на сторінці клієнта − результат їх перетворення: HTML, CSS І JavaScript.



Рисунок 6.8 – Архітектура програми

Суттєво .jsp сторінка уявляє собою розширений опис HTML сторінки.

Для створення рівня представлення використовуються наступні технології мови і бібліотеки: JSP, JSTL, HTML, CSS, JavaScript, JQuery і Bootstrap.

Мова HTML використовується для розмітки сторінок.

Мова CSS дозволяє офофрмлять ці сторінки, а бібліотека Bootstrap надає широкий набір предопісанних стилів.

Мова JavaScript дозволяє додавати поведінку до сторінки, наприклад анімацію.

Для ефективного використання JavaScript розроблені бібліотеки JQuery і Bootstrap.

Наступним є рівень додатки, який відповідає за зв'язування Java коду і HTTP запитів.

Ця анотація вказує Spring, що даний клас задає зв'язки між HTTP запитами і Java кодом.

Ці зв'язки також задаються анотаціями, які накладаються на клас і на окремі його методи.

З урахуванням базового шляху, повний шлях буде виглядати наступним чином:

https: \ \ localhost: 8080 \ quiz \ login.

Таблиця 6.1 – Зв'язування Java коду і HTTP запитів

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Шлях | Запит | Параметри | Контролер | Описи |
| \login | GET | − | LoginController | Вхід у систему |
| \login | GET | retry | LoginController | Вхід у систему після помилки |
| \quiz | GET | quizId | QuizController | Отримання тесту |
| \quiz | POST | complete | QuizController | Завершення тесту |
| \quizzes | GET | − | QuizzesController | Отримання списку тестів |
| \summary | GET | − | QuizzesController | Отримання даний про результати проходження тестів |

У додатку використовується тільки один запит POST, тобто запит на зміну стану даних web-додатки (див. рис 6.9).

Це запит формований коли користувач натискає кнопку завершення тесту.

Параметри можуть передаватися різними способами.

Для GET-запитів вони передаються як частина адреси, наприклад:

* localhost: 8080\quiz\login?retry;
* localhost: 8080\quiz\quiz?quizId = 1.

Для POST-запитів вони передаються як дані форми.

Кожен запит веде на певну JSP сторінку, на основі якої генерується HTML сторінка, яка пересилається користувачеві(див. таб. 6.2).

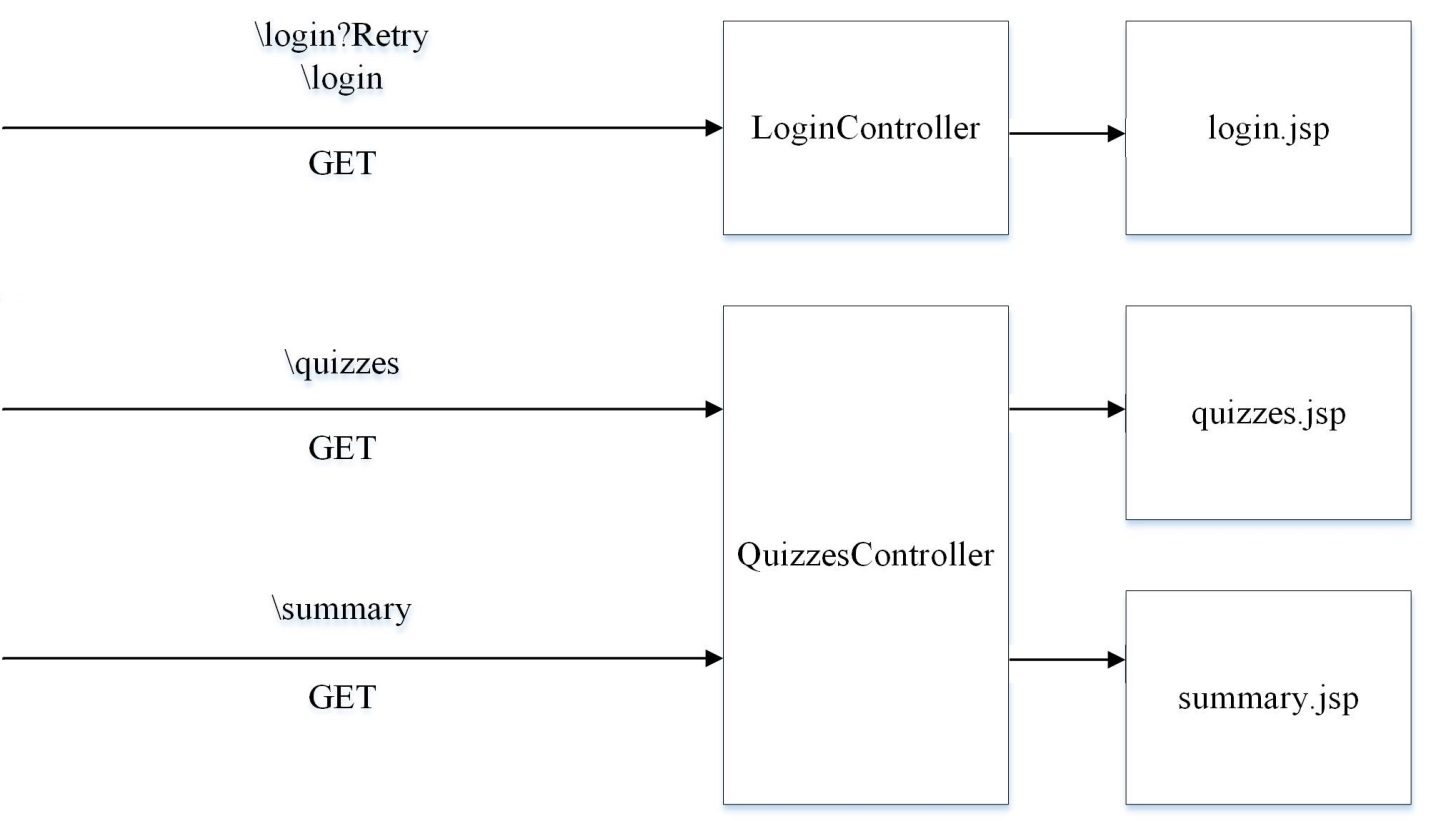


Рисунок 6.9 – Запити на зміну даних на сторінці

Таблиця 6.2 – Відповідність JSP запитів HTML сторінкам

|  |  |  |  |
| --- | --- | --- | --- |
| Шлях | Запит | Параметри | Сторінка |
| \login | GET | − | login.jsp |
| \login | GET | retry | login.jsp |
| \quiz | GET | quizId | quiz.jsp |
| \quiz | POST | complete | results.jsp |
| \quizzes | GET | − | quizzes.jsp |
| \summary | GET | − | summary.jsp |

Тобто при отриманні HTTP запиту викликається метод Java, який може його обробити.

Цей метод повертає ім'я подання, яка має бути відображена в результаті обробки запиту.

Для обробки запитів рівень додатки використовує модулі двох нижчих рівнів: рівня предметної обрласті та рівня даних.

Рівень предметної області включає її модель і сервіси.

Сервіси представляють собою реалізацію алгоритмів, які розміщуються за межами сутностей предметної області.

Сервіси позначаються анотацією @Service.

Наприклад, сервіс UserService отримує інформацію про користувачів з бази даних або створює нового користувача. якщо той увійшов в систему в перший раз.

Модель містить структуру, яка описуює предметну область(див. рис. 6.10).



Рисунок 6.10 – Структура бази даних

У даному випадку це:

* Quiz − тест;
* Question − питання тесту;
* Answer − відповідь на питання тесту;
* Result − результати проходження тесту користувачем;
* User − користувач системи.

У таблиці Answer можна побачити відповіді на всі запитання. Відповідей на питання може буди скільки завгодно. В основному відповідей на питання становить від 3 до 5. Таблиця має чотири поля(див. рис 6.11):

* id – первинний ключ відповіді;
* content – варіанти відповідей;
* correct – яка відповідь вірна;
* question\_id – зовнішній ключ вопроса.

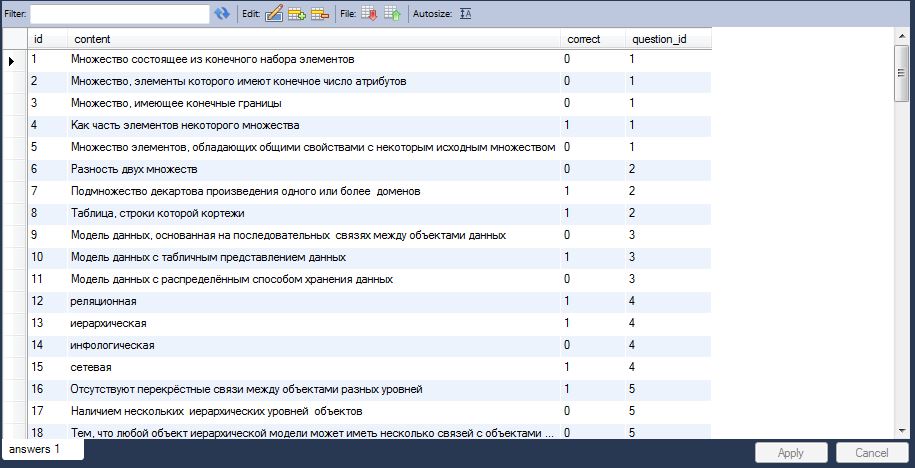


Рисунок 6.11 – Таблиця Answer

Таблиця Answer зв’язана з таблицею Question вдношенням \*:1 відповідно. У тестах може бути скільки завгодно питань. Маблиця має всього 26 питань на обидва тести. У таблиці Question можна побачити усі запитяння для тестів. Якщо користувач невірно обрав відповідь на питання з декікьками варіантами відповідей, то відповідь на питання вважається невірною. Таблиця має три поля(див. рис. 6.12):

* id – первинний ключ запитання;
* content – запитання;
* quiz\_id – зовнішній ключ теста.

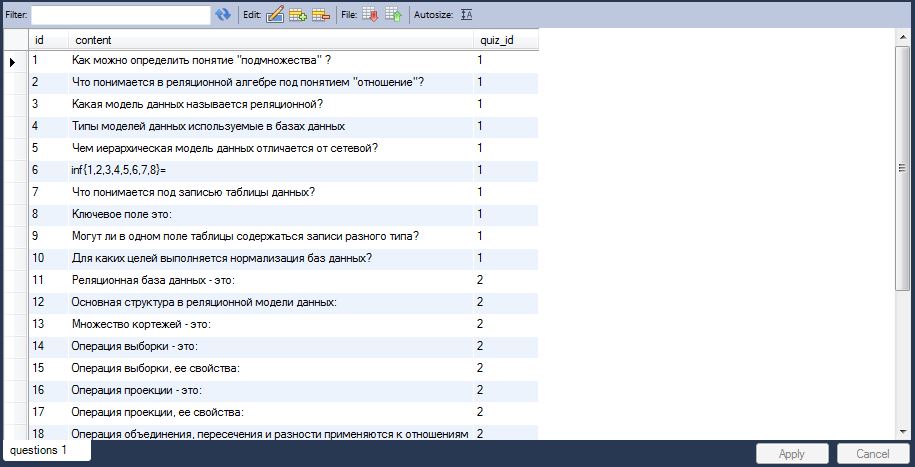


Рисунок 6.12 – Таблиця Question

Таблиця Question зв’язана з таблицею Quiz вдношенням \*:1 відповідно. Тестів може бути скільки завгодно, для прикладу зроблено два тести. У таблиці Quiz можна побачити усі тести.

Таблиця має два поля(див. рис. 6.13):

* id – первинний ключ тесту;
* title – назва тесту.

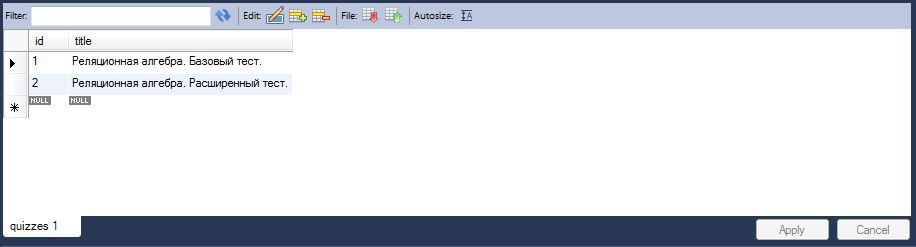


Рисунок 6.13 – Таблиця Quiz

Таблиця Quiz зв’язана з таблицею Result вдношенням 1:\* відповідно. У таблиці Result можна побачити результати студентів за тестовими завданнями. Студен може проходити один і той саме тест кілька разів. У таблиці можна побачити скільки разів студент проходив тест. Таблиця має сім полів(див. рис. 6.14):

* id – первинний ключ результату;
* completed – коли був пройдений тест;
* correct – кількість вірних відповідей;
* total – кількість питань;
* value – результат у відсотках;
* quiz\_id − зовнішній ключ теста;
* userId – зовнішній ключ студента.

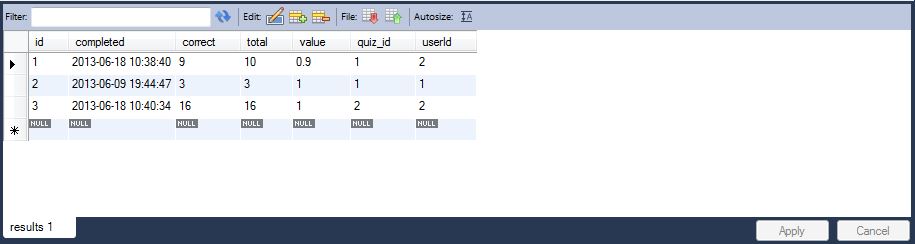


Рисунок 6.14 – Таблиця Result

Таблиця Result зв’язана з таблицею User вдношенням \*:1 відповідно. У таблиці User можна побачити всіх користувачів. У цій таблиці і робиться поділ користувачів на студентів та викладачів. Це можна побачити в полі role. Таблиця має п’ять полів(див. рис. 6.15):

* id – первинний ключ користувача;
* fullname – П.І.Б. користувача;
* name – логін;
* party – група користувача;
* role – види користувачів.

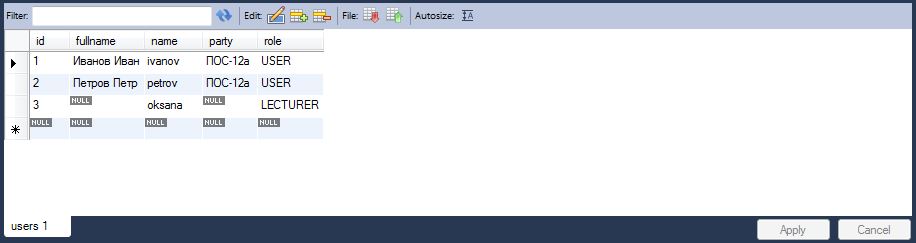


Рисунок 6.15 – Таблиця User

Повну схему даних можна побачити на рисунку 6.16.

База даних MySQL зв’язана зі службою дирекрорій Open DJ. Якщо користувач увійшов під обліковими даними студента, то йому доступні тести для проходження. Якщо користувач увійшов під обліковими даними викладача, то йому доступний список усіх студентів які пройшли тести та їх результати. Усих користувавів можно побачити на рисунку 6.17.

Найголовніше те, щоб поле name, яке знаходиться у таблиці User, співпадало з полем User ID у службі директорій Open DJ. Це є логін до свого акаунту. Якщо у службі директорій не буде знайдено користувача с таким логіном, то програма видасть помилку авторизації.



Рисунок 6.16 – Схема даних

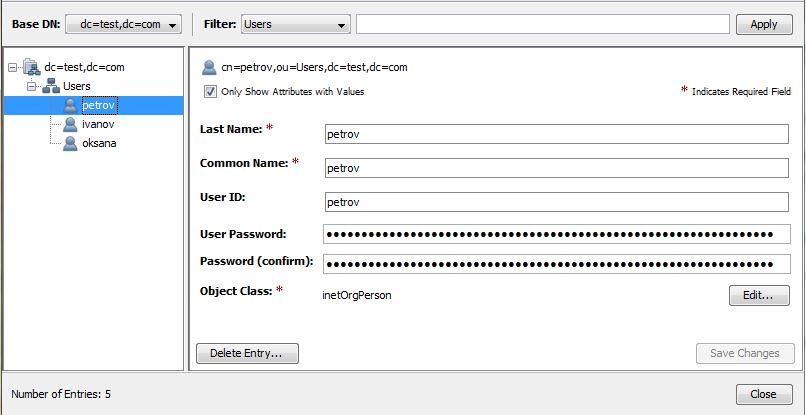


Рисунок 6.17 – Користувачі у службі дирекрорій Open DJ

Зв'язки між сутностями, показані на діаграмі безпосередньо представлені в коді за допомогою анотацій.

Наприклад, зв'язок один-до-багатьох представляється за допомогою анотації @OneToMany.

Прості властивості сутностей анотується як @Basic або @Temporal (для дат).

Анотації @Column і @JoinColumn є більш низькорівневими і дозволяють зв'язати властивість об'єкта зі стовпцем таблиці в базі даних.

Ці класи анотуванні анотаціями з пакету javax.persistence, що дозволяє автоматизувати CRUD операції з базою даних.

Безпосередньо взаємодія з базою даних здійснюється за допомогою інтерфейсів репозитаріїв:

* QuizRepository − репозиторій тестів;
* UserRepository − репозиторій користувачів;
* ResultRepository − Репозиторій результатів.

Кожен репозиторій надає повний набір CRUD операцій для одного класу з моделі предметної області:

* QuizRepository – Quiz;
* UserRepository – User;
* ResultRepository – Result.

Ці інтерфейси будуть реалізовані динамічно бібліотекою Hibernate.

Як результат у додатку досить коректно описати і проаннотіровать класи, що описують предметну область і створити інтерфейси репозиторіїв.

А їх реалізація буде створена автоматично, що значно спрощує створення програми.

Тобто на кожному рівні розташовані класи певного виду(див. рис 6.18):

* Застосування − @Controller;
* Предметна область − @Service, @Entity;
* Дані – JpaRepository.

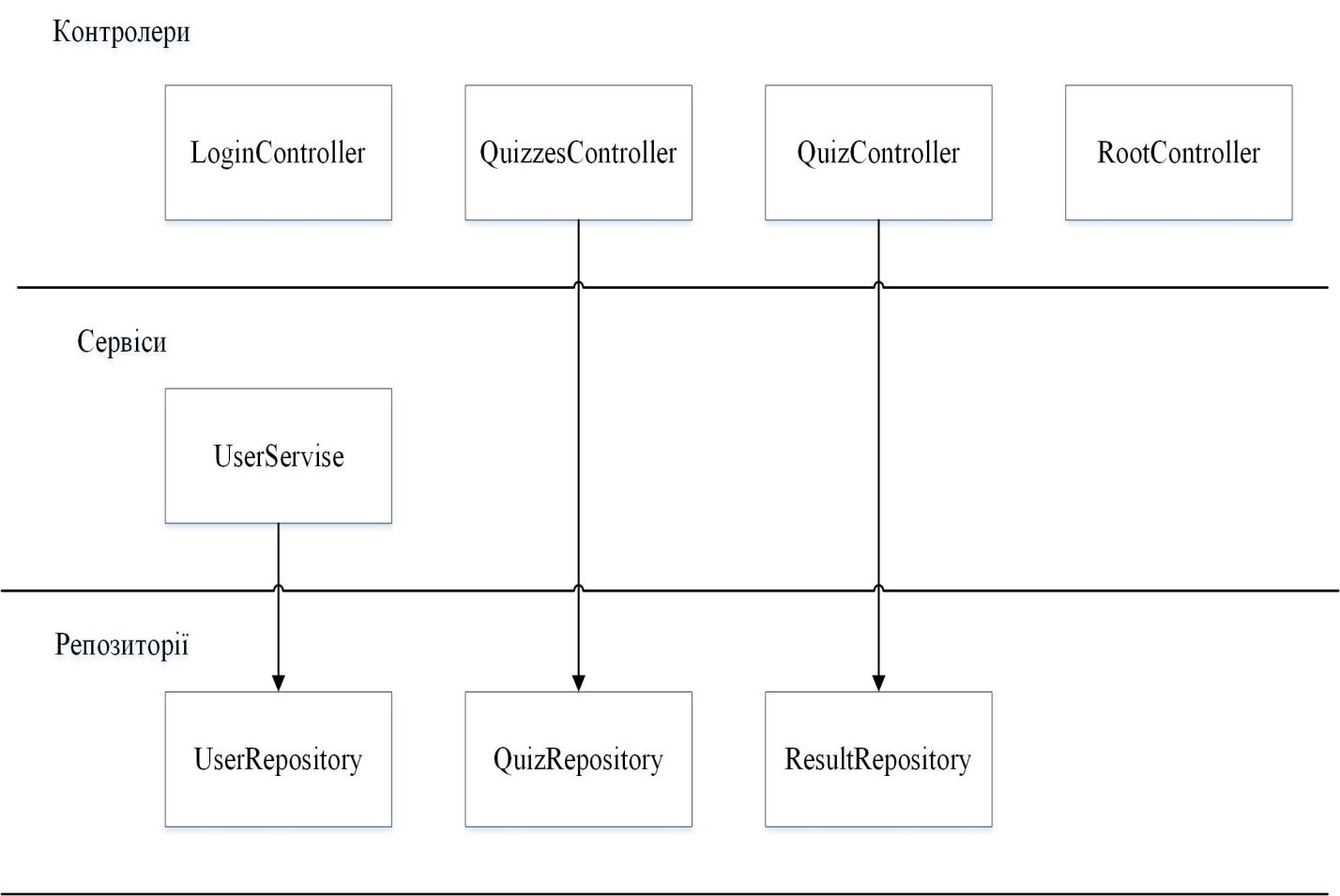


Рисунок 6.18 – Архітектурні рівні програми

Це були розглянуті внутришності програми. Сам опис теорії знаходиться в HTML сторінках. Таких сторінок усьго дев’ять: одна головна сторінка та вісім сторінок з реляційними операціями.

Для кожної операції є своя HTML сторінка. Вони пов’язані лінками з головною сторінкою.

Інтерфейс головної сторінки можна побачити у додатку Г.

На початку сторінки знаходяться лінки на сторінки з реляційними операціями. Ще вони називаються якорями. В них вказує id самої операції та лінка href на jQuery функцію goToOperation('to\_union', 'html/union.html'), яка робить перехід на сторінку з вибраною реляційною операцією. Далі іде блок який містить теорію, яка ділиться на абзаци тегами <p></p>. В кінці сторінки підключаються файли зі скриптами.