Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра Програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни «Бази даних»

ІНФОРМАЦІЙНА СИСТЕМА «ШКОЛА»

Керівник , доцент Мазурова О.О.

Студент гр. ПІ-15-1 Євченко Б.О.

Комісія: Доц. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Мазурова О.О.

Ст. викл. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Черепанова Ю.Ю.

Ст. викл. \_\_\_\_\_\_\_\_\_\_\_\_\_ Широкопетлєва М.С.

Харків 2016

\_\_\_\_\_\_\_\_\_\_\_\_\_Харківський національний університет радіоелектроніки\_\_\_\_\_\_\_\_\_\_

**Кафедра** \_\_\_Програмної інженерії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Дисципліна** \_\_\_Бази даних\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Спеціальність** \_Програмна інженерія\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Курс** \_\_\_2\_\_\_\_\_\_\_\_\_\_\_\_\_ **Група** \_\_\_ПІ-15-1\_\_\_\_\_\_\_\_\_ **Семестр**\_\_\_\_ 3\_\_\_\_\_\_\_\_\_\_\_

**ЗАВДАННЯ**

**на курсову роботу студента**

\_\_\_\_\_\_\_\_\_\_\_\_*Євченка Богдана Олеговича*\_\_\_\_\_\_\_\_\_\_

**1. Тема роботи:** \_\_\_\_Інформаційна система «Школа»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**2. Строк здачі закінченої роботи** \_\_30.12.2016\_\_\_\_\_\_\_\_

**3. Вихідні дані для роботи:** методичні вказівки до виконання курсової роботи, вимоги до інформаційної системи, предметна область, що пов’язана з\_діяльністю школи.

**4. Зміст розрахунково - пояснювальної записки:** вступ, аналіз предметної області; постановка задачі; проектування бази даних; опис програми; висновки; перелік посилань.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**5. Перелік графічного матеріалу:** загальна схема концептуальної моделі, ER-діаграма, структура 1НФ, 2НФ, 3НФ, схема БД в 3НФ, UML-діаграми, копії екранів (“скріншоти”) прикладної програми, приклади звітів прикладної програми\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**6. Дата видачі завдання** \_\_\_01.10.2016 р.\_\_\_**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| Номер | Назва етапів курсової роботи | Строк виконання етапів роботи | Примітки |
| 1 | Аналіз предметної області | 01.10.16 – 24.09.16 | Виконано |
| 2 | Постановка задачі | 20.09.16 – 30.09.16 | Виконано |
| 3 | Побудова ER-діаграми бази даних | 27.09.16 – 15.10.16 | Виконано |
| 4 | Оформлення розділів 1, 2 та 3.1, 3.2 пояснювальної записки | 15.10.16 - 27.10.16 | Виконано |
| 5 | Перша контрольна точка з курсового проекту | 24.10.16 – 28.10.16 | Виконано |
| 6 | Нормалізація бази даних | 20.10.16 - 10.11.16 | Виконано |
| 7 | Створення демо-версії програми | 20.10.16 – 20.11.16 | Виконано |
| 8 | Тестування програми, наповнення бази даних | 15.11.16 - 25.11.16 | Виконано |
| 9 | Друга контрольна точка з курсового проекту | 21.12.16– 02.12.16 | Виконано |
| 10 | Реалізація остаточної версії програми | 1.12.16-15.12.16 | Виконано |
| 11 | Оформлення інших розділів пояснювальної записки | 1.11.16 – 15.12.16 | Виконано |
| 12 | Захист курсового проекту (третя контрольна точка) | 12.12.16- 30.12.16 | Виконано |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *доц. Мазурова О.О.*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2016 р.

РЕФЕРАТ

Пояснювальна записка до курсової роботи: 49 с., 41 рис., 6 джерел.

Метою роботи є розробка програми «Школа» на засадах об'єктно-орієнтованого програмування з проектуванням реляційної бази даних.

Програма розроблялася за допомогою технології Microsoft ASP.NET, мови програмування C#, мови запитів SQL, реляційної бази даних, розроблена в середі програмування Visual Studio 2015 з СУБД MS SQL.

Посилання на github: https://github.com/olderor/school

В результаті отримана програма під назвою «Школа», яка дозволяє працювати з оцінками, успішністю, відвідуванням та багатьма іншими можливостями програми. Є можливість додавати, видаляти та редагувати будь-яку інформацію. Також можна отримати різні статистики, декілька звітів. Також програма має автоматизовані функції надсилання повідомлень батькам.

ШКОЛА, ВЧИТЕЛЬ, УЧЕНЬ, ОЦІНКА, ПРОГРАМА, СУТНІСТЬ, БАЗА ДАНИХ, SQL, ASP.NET, C#, ПРЕДМЕТНА ОБЛАСТЬ, АВТОМАТИЗАЦІЯ, ЗАПИТ, ЗВІТ, ТАБЛИЦЯ, ER-ДІАГРАМА, СТАТИСТИКА, КОНЦЕПТУАЛЬНА МОДЕЛЬ.

ЗМІСТ

Вступ…………………………………………………………………………..5

1. Аналіз та концептуальне моделювання предметної області……..……..6

* 1. Аналіз предметної області………………………...……………………..6
  2. Опис функціональної структури………………………………………...7
  3. Опис інформаційних потреб користувачів……………………………..8
  4. Опис об’єктів та зв’язки між ними………………………………….…..10
  5. Опис існуючого документообігу……………………………………..…11
  6. Опис обмежень цілісності……………………………………………….12
  7. Опис алгоритмічних залежностей………………………………………13
  8. Лінгвістичні відносини…………………………………………………..14
  9. Вимоги до програми……………………………………………………..15

2. Постановка задачі………………………………………………………….16

3. Проектування бази даних…………………………………………………19

3.1 UML – моделювання……………………………………………………..19

3.2 ER – моделювання………………………………………………………..21

3.3 Побудова схеми реляційної бази даних у третій нормальній формі….22

4. Опис програми……………………………………………………………..28

4.1. Загальні відомості………………………………………………………..28

4.2. Виклик та завантаження………………………………………………...28

4.3 Призначення та логічна структура програми…………………………..29

4.4 Опис фізичної моделі бази даних……………………………………….30

4.5 Опис програмної реалізації……………………………………………...36

4.6 Опис задачі автоматизації……………………………………………….43

Висновки……………………………………………………………………...47

Перелік посилань…………….………………………………………………48

ВСТУП

Бази даних на сьогоднішній день використовуються в різноманітних областях, такі як великі та малі підприємства, муніципальні освітні установи та ін.

Незважаючи на те, що муніципальні освітні установи бувають різного обсягу, бази даних дуже спрощують роботу з інформацією, яким би не була ця освітня установа.

Система бази даних «Школа» необхідна перш за все вчителям та іншим працівникам школи, наприклад, завуч чи директор, а також була б корисливою для батьків та учнів. В даній області існує багато проблем, які ця система може виправити та допомогти вчителям вести облік більш зручним та швидким способом.

Програмна система «Школа» була створена завдяки мові програмування C#. В основу покладений об’єктно-орієнтований підхід, завдяки якому програму дуже легко супроводжувати під час її використання.

У якості інструменту для збереження даних обрано реляційну модель бази даних, яка дозволяє чітко та лаконічно маніпулювати даними. Для керування базою даних використовується СУБД MS SQL Server, яка є однією з найпопулярніших та найефективнішою на теперішній момент.

Під час написання програми були використані такі технології: об’єктно-орієнтоване програмування, реляційні бази даних і нормалізація відношень в них, технологія доступу до баз даних, мова запитів SQL та деякі особливості СУБД MS SQL Server.

1. АНАЛІЗ ТА КОНЦЕПТУАЛЬНЕ МОДЕЛЮВАННЯ ПРЕДМЕТНОЇ ОБЛАСТІ
   1. Аналіз предметної області

Перш за все, найбільшою проблемою є сучасна система обліку – шкільний журнал. Більша частина інформації про учнів зберігається в шкільному журналі, який є єдиний на клас, тобто одночасно мати доступ до цієї інформації може мати лише одна людина. Це не є зручно, бо при формуванні звіту чи перевірки завучем журнал вилучається на деякий час, вчитель не може мати доступ до нього та зробити відповідні записи в цей час. Також проблемою є «шукання» шкільного журналу по школі. Староста може випадково забути забрати журнал з попереднього уроку чи інший вчитель може забрати журнал на деякий час для виставлення оцінок з його предмету. Весь цей час, поки журнал шукають по всій школі, витрачається зайвий час, вчитель не може своєчасно зробити записи. Рішенням цієї проблеми є ведення електронного журналу, який дасть змогу формувати звіти, отримувати необхідні данні та робити записи одночасно декільком людям.

Іншою проблемою є слідкування за успішністю учня, наприклад, виставлення оцінок у щоденник. Вчителю необхідно робити відповідні записи та ставити оцінки у щоденник учня. Учень може випадково забути чи пошкодити щоденник, таким чином запис не буде зроблений. При ведені електронного обліку, можливо буде продивитися відповідні записи та слідкувати за успішністю та відвідуванням учнем уроків як вчителям, так і родичам цього учня.

Іноді вчителям потрібно робити деякі обчислення з журналом, наприклад, виставлення семестрових чи річних оцінок, рахувати скільки уроків було пропущено учнем та інше. Все це необхідно рахувати, підкреслюю, для кожного учня, що є не тільки важкою роботою для вчителя, але і займає багато часу. Такі процеси можна автоматизувати, що надасть змогу вчителям заощадливо та з користю проводити час з дітьми, ніж займатися бюрократичною працею.

Звичайний шкільний журнал має лише одну клітинку в певній даті, для певного предмета для певного учня. Електронний журнал може дати можливість ставити декілька оцінок чи інших відміток для учня на певному предметі, а також можливість робити деякі коментарі за необхідністю.

* 1. Опис функціональної структури

Розглянемо функціональну структуру системи. Проаналізувавши інформацію про користувачів системи (див. рис. 1.1), можна зробити висновок, що користувачів системи можна умовно поділити на три частини (див. рис. 1.2).

Перша частина – це адміністратори бази даних – працівники школи, які мають доступ до редагування та введення нової інформації про користувачів системи, наприклад, зареєструвати нового вчителя, тобто люди, які керують школою.

Друга частина – це працівники школи, які мають доступ до редагування та введення нової шкільної інформації (ставити оцінки, додавати нові теми чи предмети та інше). Також для працівників школи необхідно зробити спеціалізовані запити для формування звітів. Зауважимо, що працівники школи, які відносяться до адміністраторів, також відносяться до цієї групи.

ІС «Школа»

Завуч

Вчитель

Батьки

Учень

ьь

Директор

Класний

керівник

Рисунок 1.1 – Функціональна структура ІС «Школа»

Третя частина – це учні та родичі, наприклад, батьки, які не можуть вносити свої корективи до бази, а лише спостерігати за інформацію, наприклад, за успішністю відповідного учня. Також вони потребують інші функції системи, наприклад, які пропущені теми необхідно розглянути учню, як покращити успішність та отримати необхідну оцінку у семестрі.

* 1. Опис інформаційних потреб користувачів

Звичайні користувачі системи (учні чи родичі) потребують наступну інформацію:

1. отримати та переглянути інформацію:
   1. оцінки;
   2. пропуски;
   3. коментарі та зауваження від вчителя;
2. робити передбачення, які оцінки необхідно отримати, щоб мати оцінку X за семестр\рік;
3. отримати статистку по успішності учня:
   1. з певного предмета (предмет, оцінка, дата);
   2. загалом (предмет, оцінка).

Завуч

Вчитель

Директор

Класний

керівник

адміністратор

ІС «Школа»

працівник

Учні

Батьки

користувач

Рисунок 1.2 – Функціональна структура ІС «Школа»

Працівники школи (вчителя):

1. отримати та переглянути інформацію:
   1. оцінки певного учня з певного предмету;
   2. відповідні теми з предмету;
2. мати можливість ставити відмітки та редагувати інформацію;
3. знайти учня, в якого найменше оцінок з предмету, для виклику до дошки;
4. формувати звіти:
   1. успішність учнів з певного предмету;
   2. пропуски учнів.

Адміністратор:

1. отримати інформацію про певного користувача в системі;
2. додати нових користувачів чи змінити їх дані.

Слід зауважити, що необхідно автоматично повідомляти родичів учня, якщо його успішність знижається з часом.

* 1. Опис об’єктів та зв’язки між ними

Взаємодія об’єктів відбувається наступним чином (див. рис. 1.3). Необхідно зауважити, що об’єкт «відмітка» може бути як оцінкою, так і іншим будь-яким коментарем.

Відмітка

Учень

Клас

ставиться учню

Вчитель

ставиться вчителем

навчається в класі

Предмет

відмітка з предмету

викладає предмет

Тема

тема з предмету

Кабінет

викладається в кабінеті

Рисунок 1.3 – Зв’язки між об’єктами

* 1. Опис існуючого документообігу

Документообіг в предметній області складається з:

1. успішність учнів з певного предмету (предмет, учень, оцінка);
2. пропуски учнів (дата, предмет, вчитель, додаткова інформація);
3. табель учня (предмет, оцінка).
   1. Опис обмежень цілісності

Опис обмежень цілісності відносно зв’язків об’єктів:

1. в учня може бути багато родичів, у родича може бути багато учнів;
2. в учня може бути багато відміток, одна відмітка може бути тільки в одного учня;
3. відмітка може бути тільки з одного закріпленого предмету, за закріплений предмет може бути багато оцінок;
4. вчитель може викладати багато закріплених предметів, закріплений предмет може бути викладений тільки одним вчителем;
5. закріплений предмет може мати лише один опис предмету, предмет може мати багато закріплених предметів.

Опис обмежень цілісності відносно ідентифікації об’єктів:

1. кожен вчитель ідентифікується кодом;
2. кожен учень ідентифікується кодом;
3. кожен родич ідентифікується кодом;
4. кожна оцінка ідентифікується кодом;
5. кожен предмет ідентифікується кодом;
6. кожен закріплений предмет ідентифікується кодом;
7. кожна тема ідентифікується кодом;
8. кожен закріплена тема ідентифікується кодом;
9. кожен класний керівник ідентифікується кодом вчителя;
10. кожен кабінет ідентифікується номером кабінету.
    1. Опис алгоритмічних залежностей

Для учня є характеристика «відмінник». Значення цієї характеристики можна визначити наступним чином: якщо учень має всі семестрові та річні оцінки не менші ніж 10, то цього учня можна вважати відмінником.

Для визначення середньої оцінки (наприклад, за семестр) необхідно знайти суму всіх оцінок та поділити на їх кількість. Нехай учень набрав оцінок по кожна, де – індекс відповідної оцінки. Тоді формула середньої оцінки:

Для визначення мінімальної кількості оцінок, які необхідні учню для отримання оцінки в семестрі, необхідно знайти такі оцінки, щоб середня оцінка була не менше ніж . Для того, щоб кількість була мінімальною, учню необхідно набрати оцінки з максимальним значенням, тобто по 12 балів. Щоб знайти кількість таких оцінок використаємо формулу для знаходження середньої оцінки трохи змінивши її. Нехай учню необхідно набрати оцінок по 12 балів та впродовж семестру він набрав оцінок по кожна. Маємо наступну формулу для визначення середньої оцінки:

Домножимо обидві частини на та згрупуємо:

Маємо, – мінімальна кількість оцінок по 12, яка необхідна учню, щоб закінчити семестр з оцінкою не менше ніж . Необхідно окремо виділити випадок, коли учень вже набрав необхідну кількість балів та вже не потребує додаткових оцінок. В такому випадку .

Отже, фінальна формула така:

* 1. Лінгвістичні відносини

В цій роботі використовувалися наступні терміни:

1. Клас – набір учнів;
2. Класний керівник – вчитель, що відповідає за певний клас;
3. Юзер – користувач системи;
4. Нік – віртуальне ім’я користувача;
5. Закріплений предмет – предмет, що закріплений за класом;
6. Закріплена тема – тема, що закріплена за класом;
7. Id – унікальний код;
8. ІПН – ідентифікаційний податковий номер фізичної особи;
9. Адміністратор – людина, що керує базою даних;
10. Семестр – позначення півріччя в шкільних закладах;
11. Семестрова оцінка – підсумкова оцінка за півріччя, середнє арифметичне всіх оцінок за півріччя;
12. Завуч – завідуючий навчальною частиною, заступник директора.
    1. Вимоги до програми

Для вирішення поставлених задач буде створена інформаційна система «Школа» за допомогою мови програмування C#. Програма буде розроблена в середовищі Microsoft Visual Studio 2015 Community Version та використанням СУБД Microsoft SQL Server.

1. ПОСТАНОВКА ЗАДАЧІ

Необхідно спроектувати базу даних, побудувати програму, що забезпечує взаємодію з нею в режимі діалогу. Необхідно зареєструвати адміністраторів, які будуть мати доступ до занесення нових користувачів системи.

В кожній школі є учні. Необхідно мати таку інформацію про них в базі:

1. прізвище, ім’я та по батькові;
2. клас, в якому навчається;
3. дата народження;
4. адреса, за якою проживає;
5. номер свідоцтва про народження;
6. контактний телефон;
7. рік, в якому почав вчитися;
8. родичі;
9. дата початку навчання;
10. дата кінця навчання;
11. додаткова інформація.

Маємо наступну інформацію про вчителя:

1. прізвище, ім’я та по батькові;
2. адреса, за якою проживає;
3. ставка;
4. стаж;
5. контактний телефон;
6. профіль.

Та інформація для предмета:

1. назва предмету;
2. профіль предмета.

Оскільки ми маємо лише загальну інформацію про предмет, нам необхідно мати інформацію про те, який вчитель веде цей предмет.

Відмітка в журналі (може бути, наприклад, оцінка, пропуск заняття, домашнє завдання та інше):

1. значення;
2. дата, коли була поставлена;
3. закріплений предмет;
4. учень;
5. тип;
6. коментар.

Також користувачами системи можуть бути родичі, інформацію про яких теж необхідно мати в системі:

1. ПІБ;
2. телефон;
3. адреса;
4. додаткова інформація;
5. тип рідні (мати, батько, бабуся та інше).

Підведемо підсумок – користувач системи:

1. нік;
2. електрона пошта;
3. пароль;
4. тип (учень\вчитель\працівник школи);

Програма повинна підтримувати перелік функцій:

1. отримати та фільтрувати інформацію:
   1. пропуски (за датою, за предметом, за вчителем);
   2. коментарі та зауваження від вчителя (за типом оцінки, за додатковою інформацією);
   3. оцінки певного учня з певного предмету (за ПІБ учня, за оцінкою);
   4. інформацію про певного користувача в системі (за ніком, за поштою, за типом аккаунта);
2. знаходити по датам пропусків теми, які були пропущені (пошук за датою);
3. повідомити родичів, якщо успішність їхнього учня зменшилася в порівнянні з попередніми оцінками (задача автоматизації);
4. отримати статистку по успішності учня (запити):
   1. з певного предмета (предмет, оцінка, дата);
   2. загалом (предмет, оцінка);
5. знайти учня, в якого найменше оцінок з предмету, для виклику до дошки (запит);
6. формувати звіти:
   1. успішність учнів з певного предмету;
   2. пропуски учнів;
   3. табель учня.
7. ПРОЕКТУВАННЯ БАЗИ ДАНИХ

3.1 UML – моделювання

На основі концептуального моделювання, проведеного у першому розділі можна виділити таких основних акторів системи: учень, вчитель, адміністратор (керівник школи). Учень отримує інформацію про поточний стан навчання. Вчитель ставить оцінки та інші відмітки учням, може переглянути інформацію, отримати статистику та сформувати звіти. Адміністратор має повний доступ до бази, може створювати нових користувачів, формує нові предмети та класи. Слід зазначити, що вчитель також може виконувати дії учня, адміністратор може виконувати дії учня та вчителя (для спрощення це не було вказано в діаграмі). На рисунку 3.1 приведена діаграма USE CASE, яка досить гарно ілюструє функції і варіанти взаємодії акторів із системою.

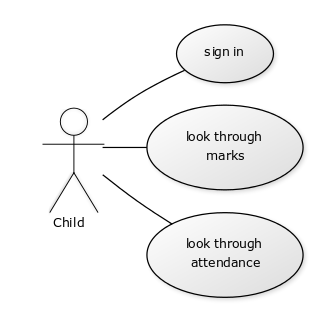


Рисунок 3.1 – Діаграма варіантів use case системи «Школа». Актор «Учень»

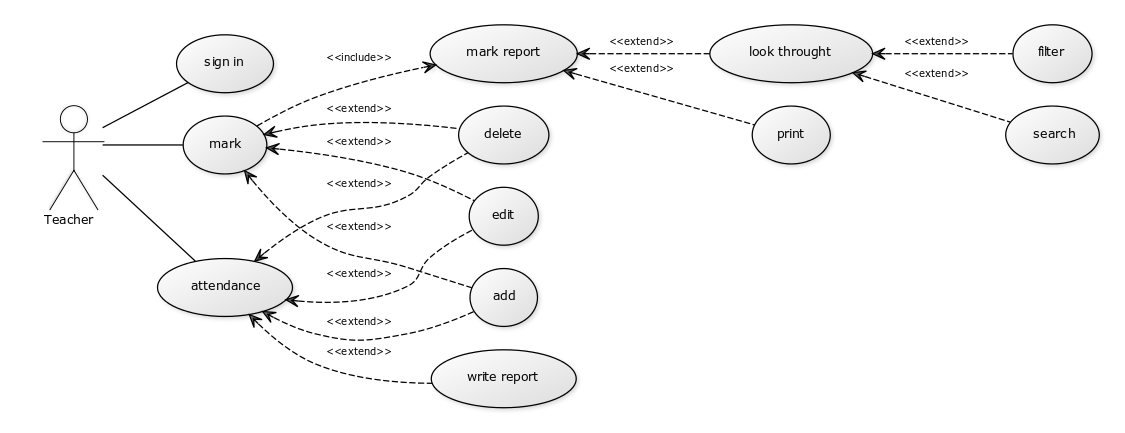


Рисунок 3.2 – Діаграма варіантів use case системи «Школа». Актор «Вчитель»

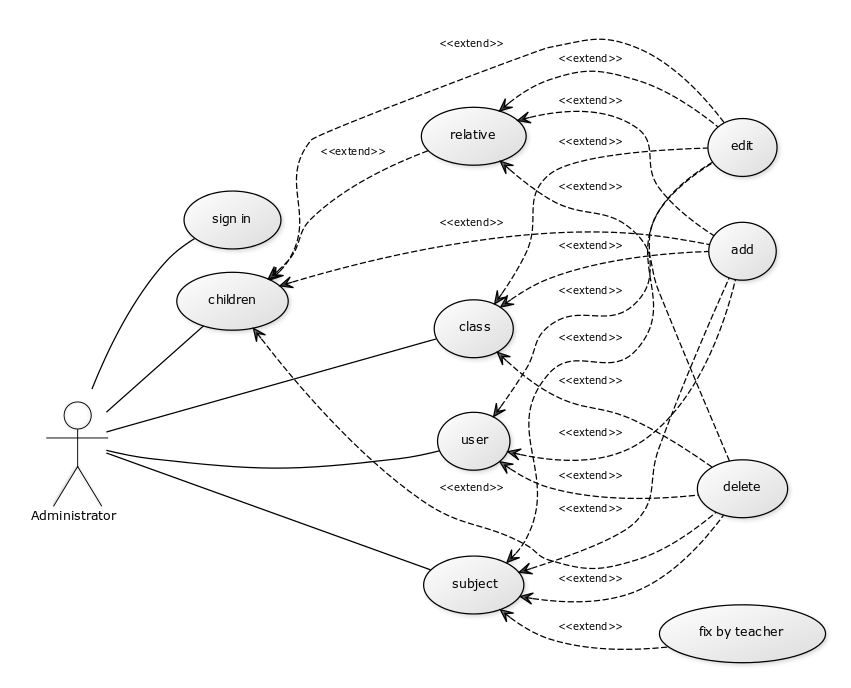


Рисунок 3.3 – Діаграма варіантів use case системи «Школа». Актор «Вчитель»

3.2 ER – моделювання

Опис ER-діаграми буде наведений нижче, розбитий на декілька частин (див. рис. 3.4 та 3.5).

Оскільки родич може мати декілька учнів, що навчаються в школі, та учень може мати декілька родичів, між таблицею родич та учень виникає зв’язок багато к багатьом. Для цього, необхідно створити додаткову таблицю «Відомості про родичів», таким чином родич буде мати відомісті про учнів, та учень буде мати відомісті про родичів.

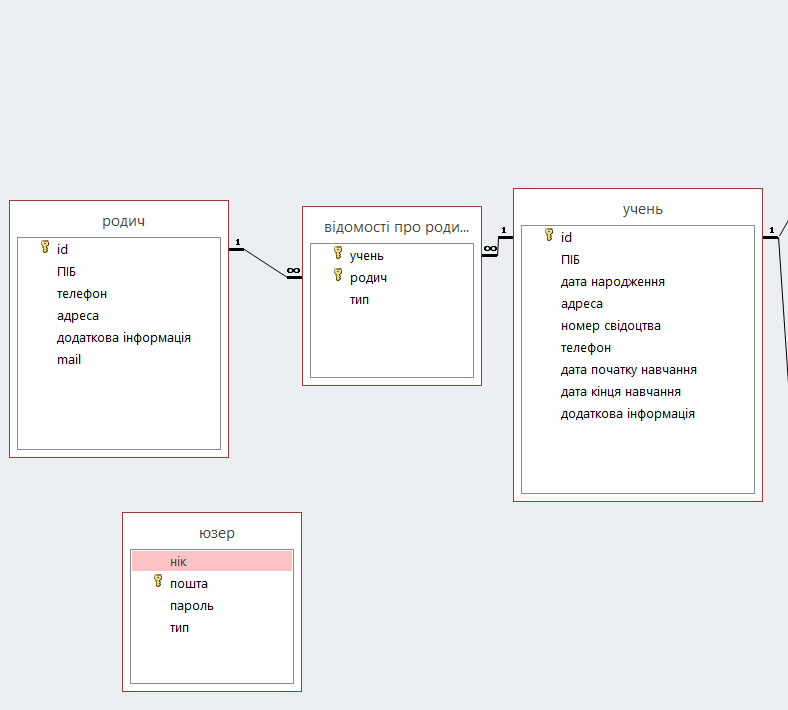


Рисунок 3.4 – Перша частина ER діаграми

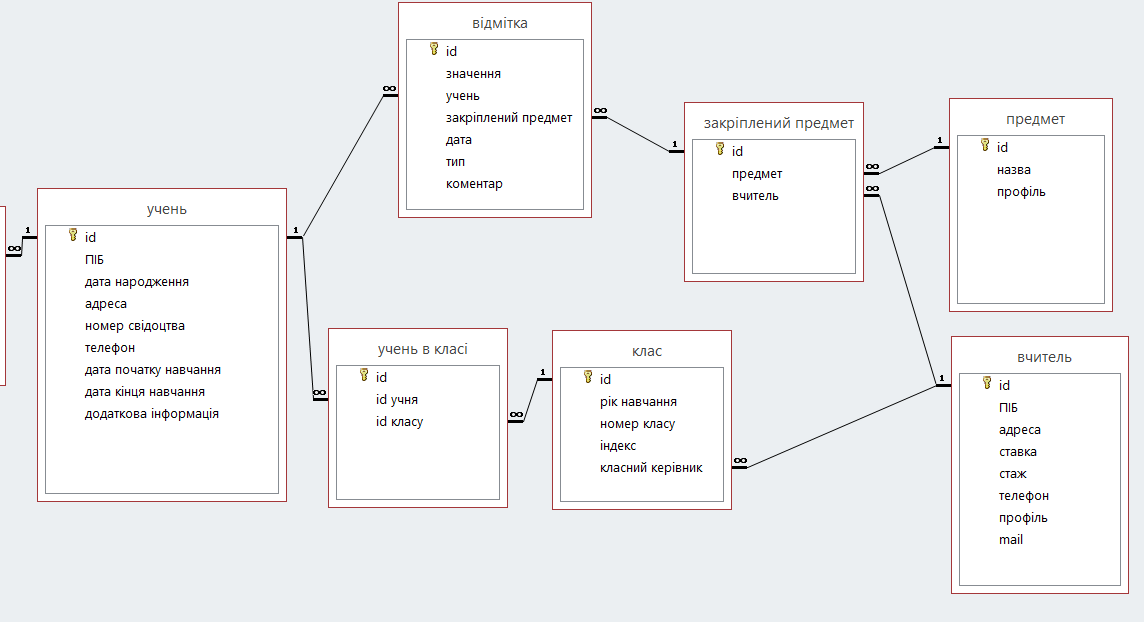


Рисунок 3.5 – Друга частина ER діаграми

3.3 Побудова схеми реляційної бази даних у третій нормальній формі

Нормалізація – це розділення таблиці на дві або більше, таких, що мають ліпші властивості під час включення, зміни та видалення даних. Кінцева мета нормалізації – отримання такого проекту бази даних, в якому кожен факт з’являється лише в одному місті, тобто виключена надмірність інформації.

Спочатку побудуємо схему бази даних «Відмітка» у першій нормальній формі, визначимо первинні ключі та покажемо залежності між атрибутами (див. рис. 3.6). Деякі атрибути було опущено задля спрощення відображення.

Таблиця «Відмітка» знаходиться у першій нормальній формі (1НФ), тому що виконується правило атомарності, також знаходиться у другій нормальній формі (2НФ), бо не кожен неключовий атрибут повністю функціонально залежить від первинного ключа відношення.

id відмітки

значення

тип оцінки

id учня

ПІБ учня

номер свідоцтва

id родича

ПІБ родича

адреса

пошта родича

тип рідства

id закріпленого предмета

id предмета

назва

профіль

id вчителя

ПІБ вчителя

телефон вчителя

стаж роботи

ставка

профіль вчителя

пошта вчителя

Рисунок 3.6 – Загальна таблиця «Відмітка» у 1НФ та 2НФ

В таблиці «Відмітка» існують наступні залежності.

Атрибут «ПІБ учня» залежить від ключа «id відмітки» та «id учня + id родича». Створимо нову таблицю «Рідство» (див. рис. 3.7), однак вона не є в 3НФ, бо існують наступні залежності. Атрибут «ПІБ учня» залежить від ключа «id учня» та «id учня + id родича». Створимо таблицю «Учень», яка буде в 2НФ та 3НФ (див. рис 3.8). Атрибут «ПІБ родича» залежить від ключа «id родича» та «id учня + id родича». Створимо таблицю «Родич», яка буде в 2НФ та 3НФ (див. рис 3.9).

Після цього таблиця «Рідство» буде в 2НФ та 3НФ (див. рис 3.10).

Атрибут «назва» залежить від ключа «id відмітки» та «id закріпленого предмета». Створимо нову таблицю «Закріплений предмет» (див. рис. 3.11), однак вона не є в 3НФ, бо існують наступні залежності. Атрибут «назва» залежить від ключа «id предмета» та «id закріпленого предмета». Створимо таблицю «Предмет», яка буде в 2НФ та 3НФ (див. рис 3.12). Атрибут «ПІБ вчителя» залежить від ключа «id вчителя» та «id закріпленого предмета». Створимо таблицю «Вчитель», яка буде в 2НФ та 3НФ (див. рис 3.13).

Після цього таблиця «Закріплений предмет» буде в 2НФ та 3НФ (див. рис 3.14).

Отже, таблиця «Відмітка» буде в 2НФ та 3НФ (див. рис 3.15).

Після нормалізації нові таблиці будуть в 3НФ, бо нема транзитивних залежностей між неключовими атрибутами.

Підсумовуючи, в результаті нормалізації була отримана наступна схема бази даних (див. рис. 3.16).

id учня

ПІБ учня

номер свідоцтва

id родича

ПІБ родича

адреса

пошта родича

тип рідства

Рисунок 3.7 – «Рідство» у 2НФ

id учня

ПІБ учня

номер свідоцтва

Рисунок 3.8 – «Учень» у 3НФ

id родича

ПІБ родича

адреса

пошта родича

Рисунок 3.9 – «Родич» у 3НФ

id учня

id родича

тип рідства

Рисунок 3.10 – «Рідство» у 3НФ

id закріпленого предмета

id предмета

назва

профіль

id вчителя

ПІБ вчителя

телефон вчителя

стаж роботи

ставка

профіль вчителя

пошта вчителя

Рисунок 3.11 – «Закріплений предмет» у 2НФ

id предмета

назва

профіль

Рисунок 3.12 – «Предмет» у 3НФ

id вчителя

ПІБ вчителя

телефон вчителя

стаж роботи

ставка

профіль вчителя

пошта вчителя

Рисунок 3.13 – «Вчитель» у 3НФ

id закріпленого предмета

id предмета

id вчителя

Рисунок 3.14 – «Закріплений предмет» у 3НФ

id відмітки

значення

тип оцінки

id учня

id закріпленого предмета

Рисунок 3.15 – Таблиця «Відмітка» у 3НФ

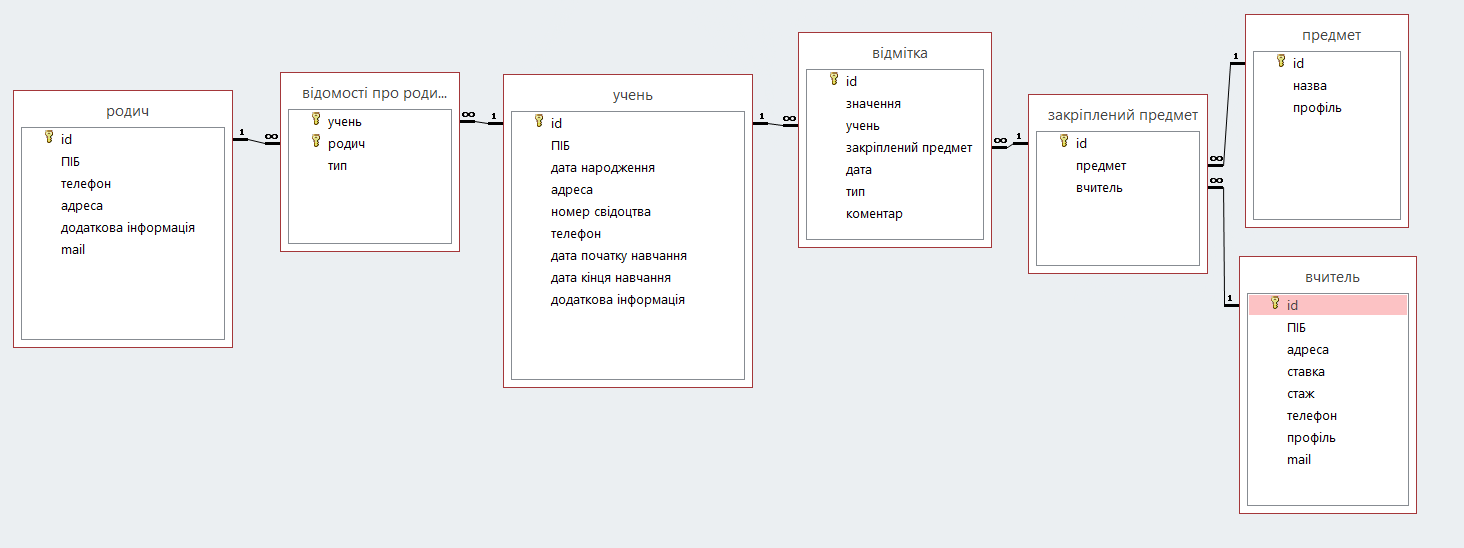


Рисунок 3.16 – Результат нормалізації

4 ОПИС ПРОГРАМИ

4.1 Загальні відомості

Програмна система «Школа» була створена завдяки мові програмування C#. В основу покладений об’єктно-орієнтований підхід, завдяки якому програму дуже легко супроводжувати під час її використання.

У якості інструменту для збереження даних обрано реляційну модель бази даних, яка дозволяє чітко та лаконічно маніпулювати даними. Для керування базою даних використовується СУБД MS SQL Server, яка є однією з найпопулярніших та найефективнішою на теперішній момент.

Згідно з потребами користувачів програма містить всі необхідні розділи: робота з оцінками, робота з предметами, робота з користувачами (вчителями, родичами, учнями). Задля безпеки даних програмна реалізація вищезазначеної системи дозволяє керувати доступом до бази даних. Для цього адміністратор має можливість створити аккаунт та вказати рівень доступа до інформації. Інформаційною системою можуть користуватися: керівництво школи, вчителя, учні та їх родичі. Завдяки ергономічному та мінімалістичному інтерфейсу, користувачі мають можливість з легкістю та комфортністю працювати з даними. Для роботи програми необхідна наявність на комп’ютері СУБД MS SQL Server.

Розмір програми 27.6 МБ, включаючи програмний код - 61.8 МБ.

4.2 Виклик та завантаження

Під час запуску програми надається можливість виконати вхід до системи. Якщо були введені недостовірні дані, програма повідомить користувача про це. Адміністратор бази даних повинен сам створити нового користувача системи за відповідним запросом користувача, таким чином до інформації не будуть мати доступ третім особам.

4.3 Призначення та логічна структура програми

Призначення програми – це поліпшення праці робітникам школи.

Програму можна розділити на три основні модулі:

1. модуль учня/родича (перегляд успішності, відвідування);
2. модуль вчителя (виставлення оцінок, відвідування; перегляд статистичної інформації; друк документів);
3. модуль адміністратора (налаштування бази даних, додання нових користувачів, перегляд усієї інформації).

Для того, щоб відкрити певний модуль потрібно виконати вхід до системи з відповідного акаунту (рівень доступу задається адміністратором під час реєстрації).

Програмою можуть користуватися багато користувачів, а отже потрібно було створити систему для їх авторизації, що і було зроблено. Перш за все це було необхідно для безпеки та цілісності даних. Також, це надає змогу захистити інформацію від третіх осіб.

Програмна система виконує наступні функції:

1. додавання, зберігання, редагування, видалення, перегляд інформації про кожний об’єкт в базі даних;
2. пошук, сортування, фільтрація інформації в базі даних;
3. формування статистики;
4. завантаження звітів;
5. виконання задачі автоматизації.

Пошук в програмі за замовчуванням здійснюється по всім атрибутам об’єкта, але користувач має можливість обрати те поле, за яким бажає здійснити пошук. Пошук відрізняється від фільтрації тим, що потрібна інформація підсвічуються, а під час фільтрації залишається лише та інформація, що задовольняє фільтру.

Сортування можна виконувати за певним одним атрибутом сутності. Також користувач має можливість обрати вид сортування: за зростанням чи за спаданням.

Статистика в програмі представлена у вигляді таблиць, що задаються за допомогою декількох атрибутів з різних таблиць бази даних.

При завантаженні звітів користувач має можливість обрати формат, вид документа і т. д. Деякі звіти потребують введення вхідних даних користувачем.

Користувач може спрогнозувати свою подальшу успішність, вказавши необхідну йому оцінку в табелі, може дізнатися чи є учень відмінником. Також програма дозволяє автоматично повідомляти батьків про зниження успішності їх учня.

4.4 Опис фізичної моделі бази даних

В якості в бази даних для програми використовується MS SQL Server. Цей продукт є одним з лідерів серед найпопулярніших баз даних. MS SQL Server справно використовують вже більш 15 років для найрізноманітніших програм.

До переваг MS SQL Server відноситься її продуктивність. Програма має прозоре стиснення та шифрування даних. В MS SQL Server є більш ефективний контроль доступу та засобів управління дозвіл і забезпечує більш високу продуктивність при зборі даних. Також програма має високий рівень захищеності даних, тому її часто використовують в програмах, в яких присутня аутентифікація та авторизація користувачів, хоча вона підходить для надійного збереження будь-якої важливої інформації. MS SQL Server підтримує зберігання даних в дуже великих об’ємах.

Із урахуванням усіх вимог до функціональної складової інформаційної системи «Школа», була створена база даних, яка складається з 10 таблиць (див. рис. 4.1-4.10).

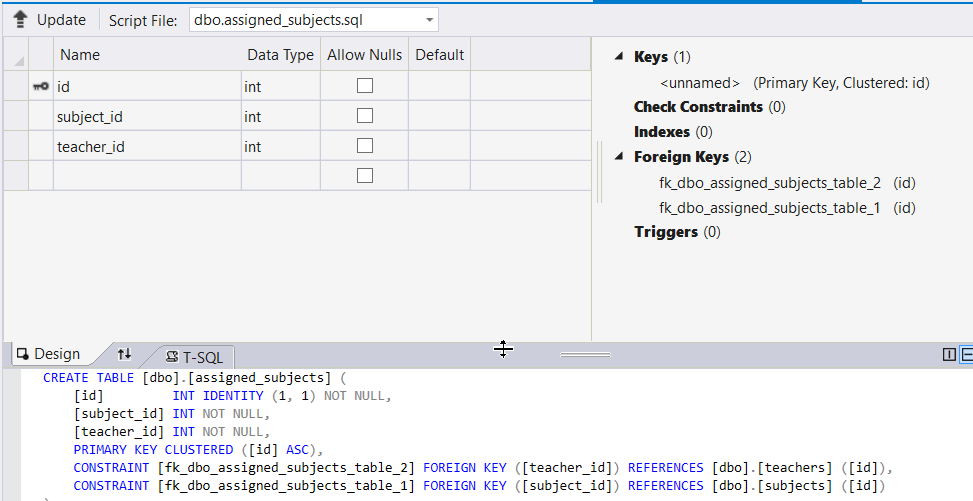


Рисунок 4.1 - Структура таблиці «Зафіксований предмет»

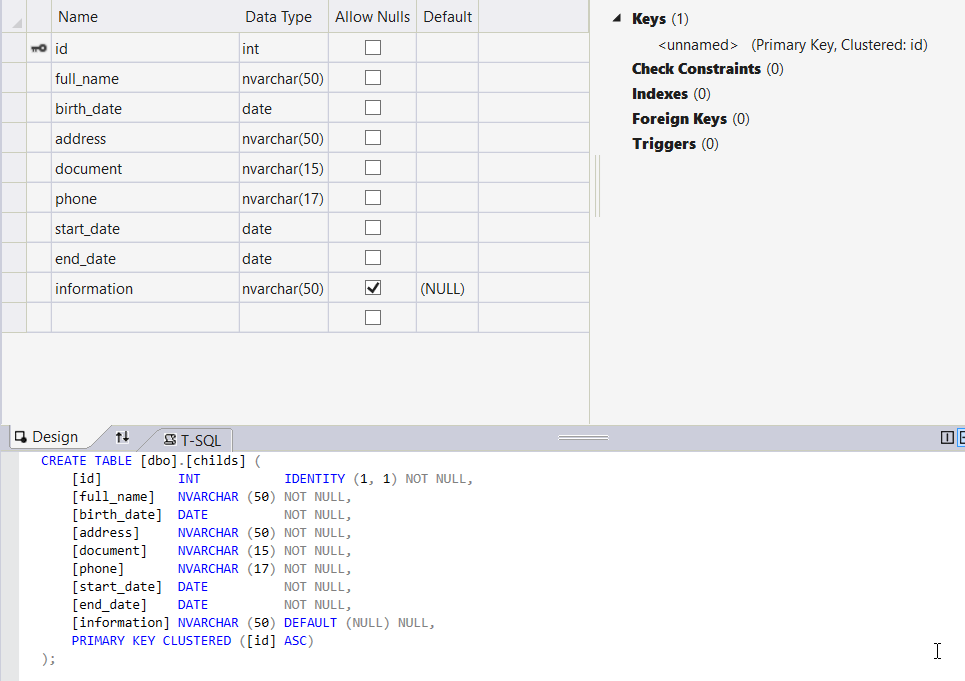


Рисунок 4.2 - Структура таблиці «Учні»

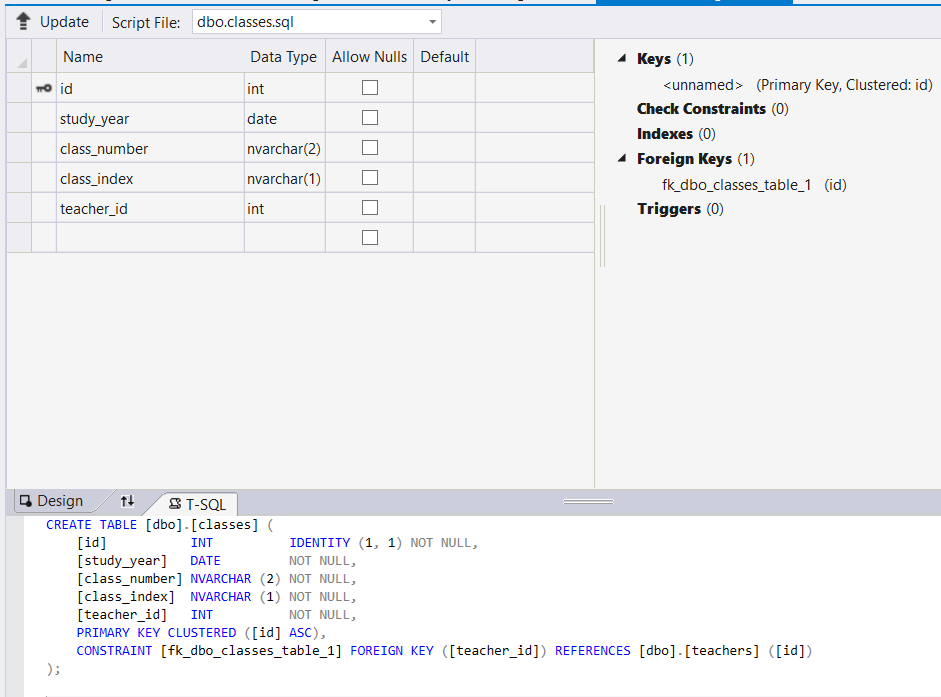


Рисунок 4.3 - Структура таблиці «Класи»

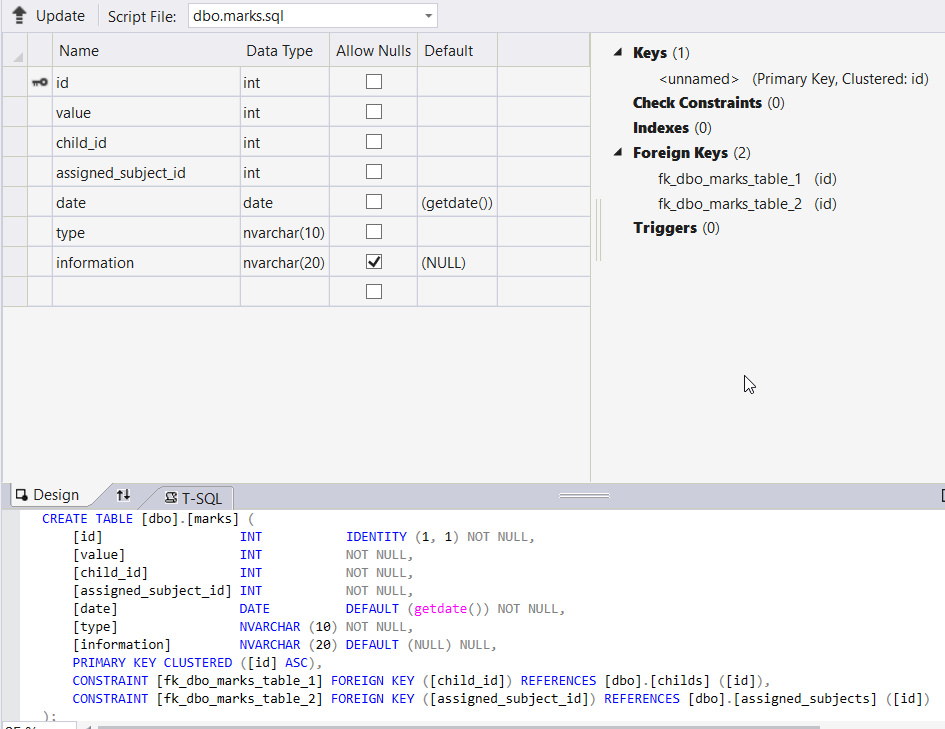


Рисунок 4.4 - Структура таблиці «Відмітки»

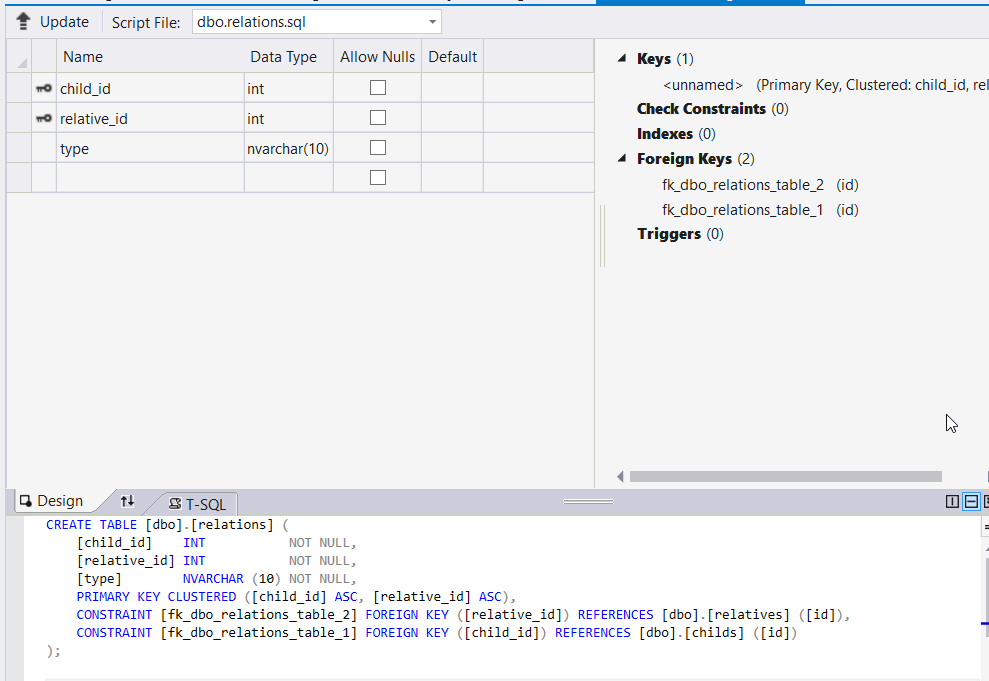


Рисунок 4.5 - Структура таблиці «Зв’язок учня з родичем»

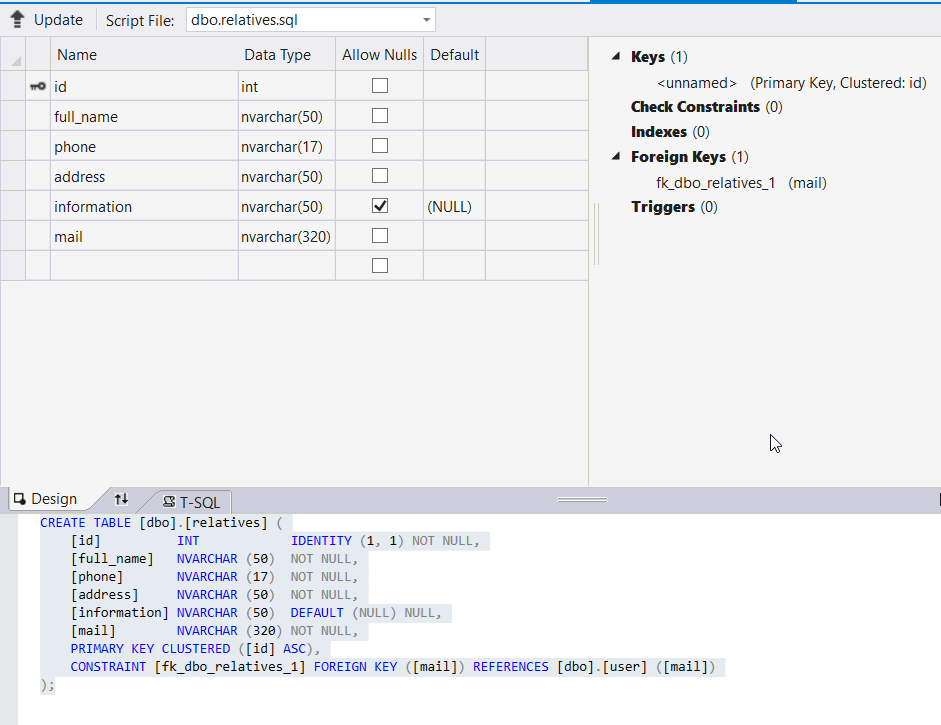


Рисунок 4.6 - Структура таблиці «Родичі»

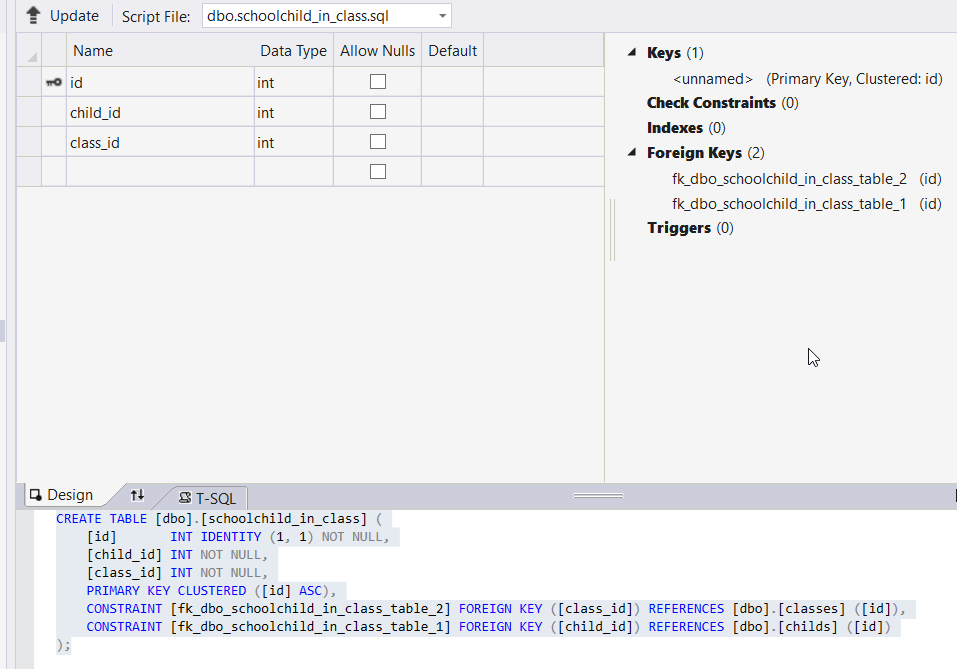


Рисунок 4.7 - Структура таблиці «Учень в класі»

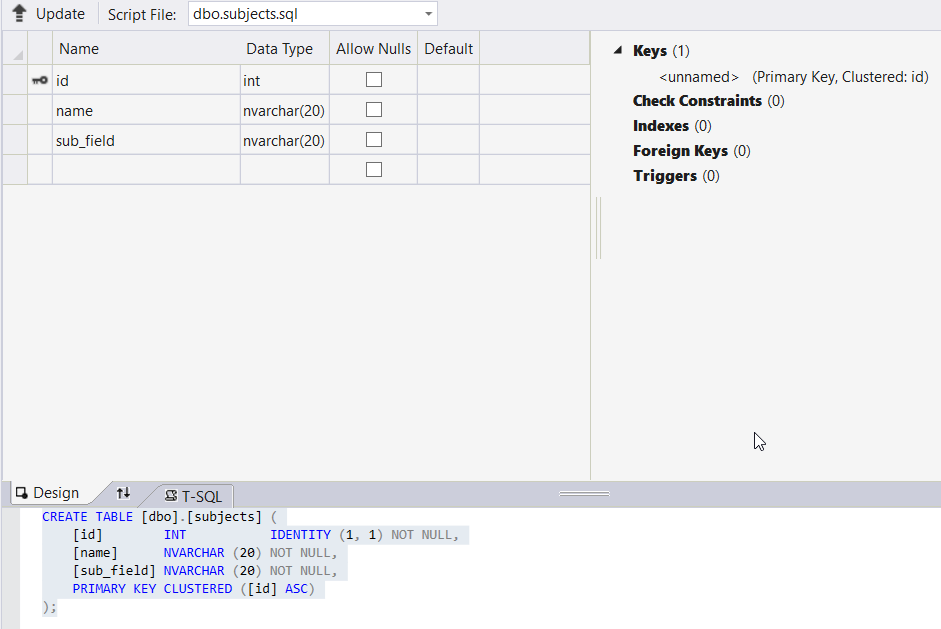


Рисунок 4.8 - Структура таблиці «Предмети»

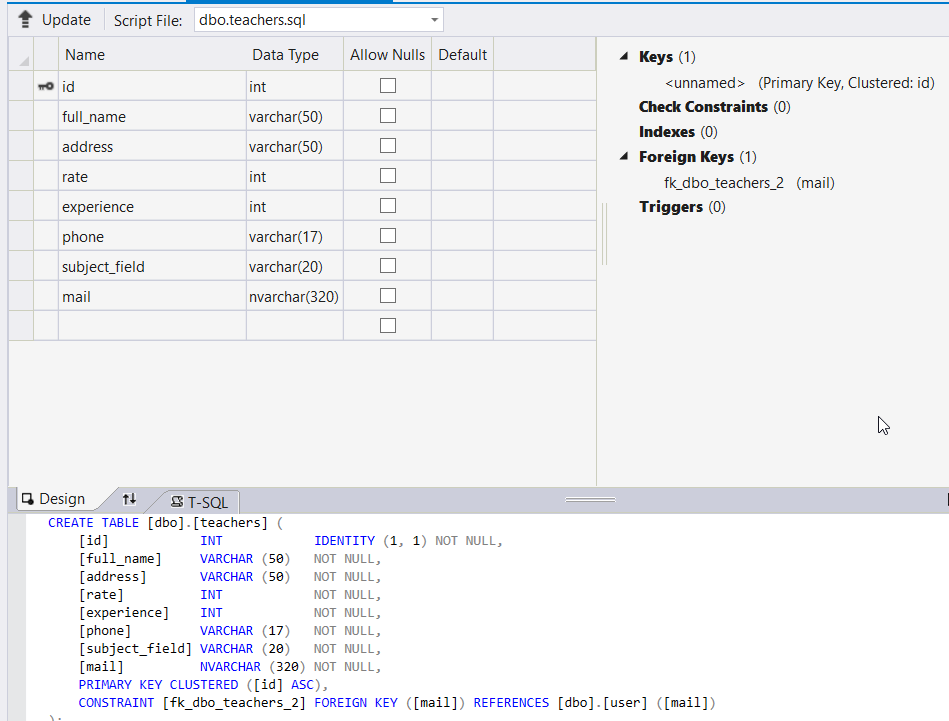


Рисунок 4.9 - Структура таблиці «Вчителя»

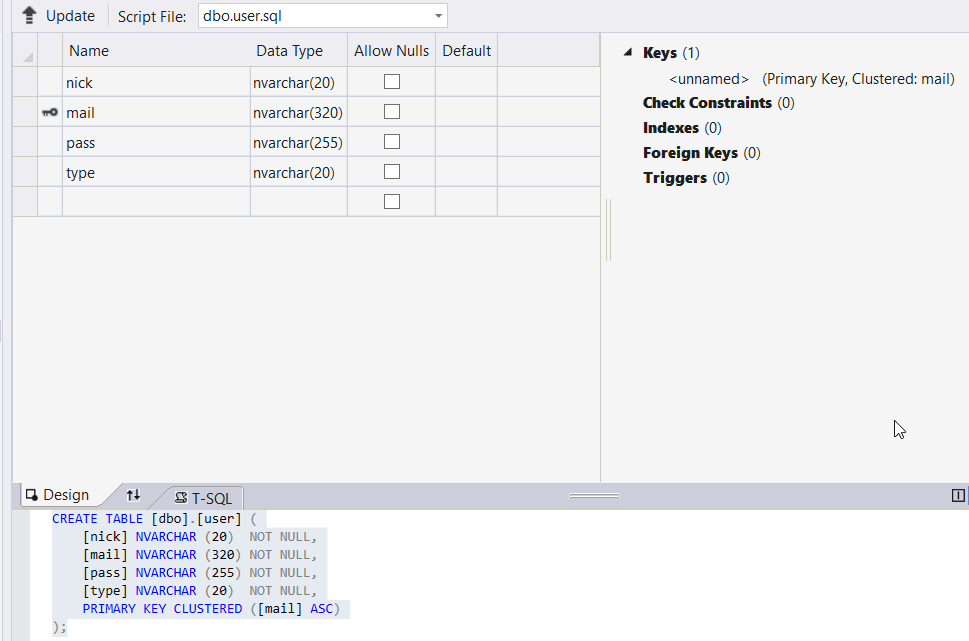


Рисунок 4.10 - Структура таблиці «Користувачі»

4.5 Опис програмної реалізації

Відображення інформації відбувається за допомогою таблиць.

Навігація в програмі виконується за допомогою пункту Файл – Перейти (див. рис. 4.11). В кожному головному вікні є можливість перейти до кожного іншого головного вікна. Слід зауважити, що у дочірніх вікнах (наприклад, вікно редагування чи вікно пошуку) такої можливості нема.

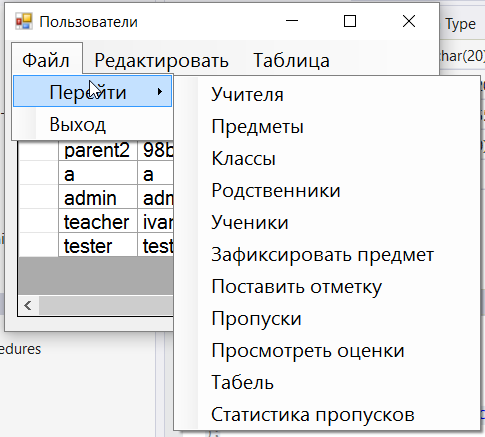


Рисунок 4.11 – Приклад навігації

Пошук, фільтрація та сортування інформації для кожної сутності знаходяться за домогою пункту меню Таблица (див. рис. 4.12). Для того, щоб відсортувати таблицю, необхідно вказати поле, за яким потрібно відсортувати, під час повторного вибору поля відбувається сортування в зворотному порядку.

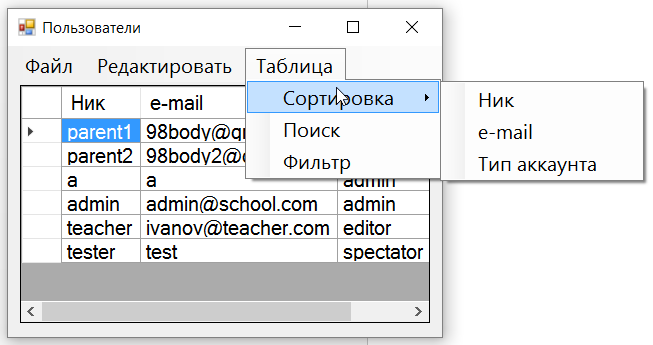


Рисунок 4.12 – Приклад сортування, пошуку, фільтру

Функції додання, редагування та видалення доступні в пункті меню Редактировать (див. рис. 4.13). Для кожної сутності існує своє окреме вікно редагування зі своїми специфічними властивостями. Наприклад, для таблиці користувачів це Рисунок 4.14, а для таблиці учнів – Рисунок 4.15. Слід зауважити, що тут можна редагувати зв’язок між учнем та родичем.

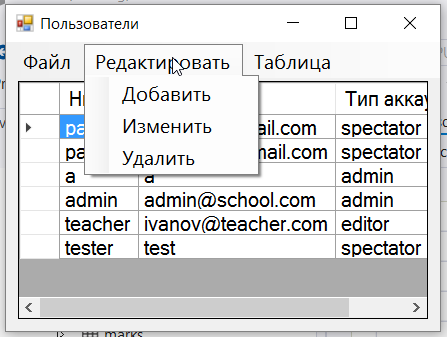


Рисунок 4.13 – Приклад редагування

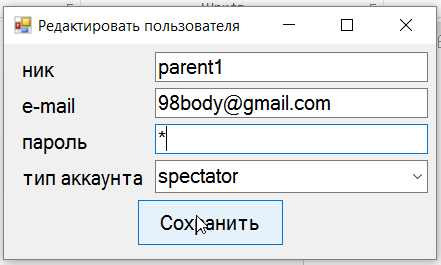


Рисунок 4.14 – Приклад редагування користувача

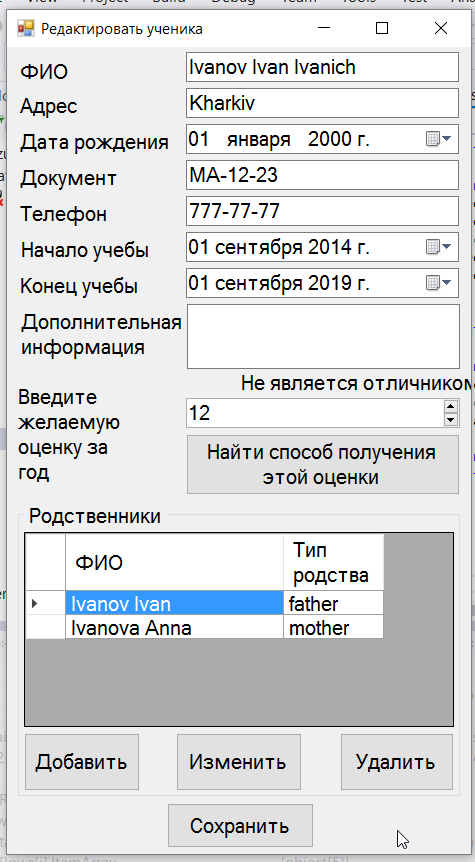


Рисунок 4.15 – Приклад редагування учня

Для кожного учня програма вичислює певні характеристики. Наприклад, характеристика «Відмінник» (див. рис. 4.15). Також там можна зробити передбачення та прорахувати необхідні оцінки, для отримання шуканої середньої.

int value = Convert.ToInt32(numericUpDown1.Value);

schoolDataSetTableAdapters.markTableAdapter ta = new schoolDataSetTableAdapters.markTableAdapter();

schoolDataSet.markDataTable dt = new schoolDataSet.markDataTable();

ta.Fill(dt, DateTime.Now.Year, id, "оценка");

int result = 1;

if (dt.Rows.Count != 0)

{

double val2 = value - 0.5;

int sum = 0;

for (int i = 0; i < dt.Rows.Count; ++i)

{

sum += Convert.ToInt32(dt.Rows[i].ItemArray[0]);

}

double answer = val2 \* dt.Rows.Count - sum;

answer = answer / (double)(12 - val2);

result = Convert.ToInt32(Math.Ceiling(answer));

result = Math.Max(0, result);

}

MessageBox.Show("Ученику необходимо получить " + result + " оценок по 12 баллов, чтобы получить оценку " + value + " в семестре.", "Решение", MessageBoxButtons.OK);

Лістинг 4.1 – Фрагмент коду, що робить передбачення

За допомогою програми можна дізнатися інформацію про пропуски учнем уроків (див. рис. 4.16). Для цього необхідно вибрати учня (див. рис. 4.17), для якого необхідно знайти цю інформацію.

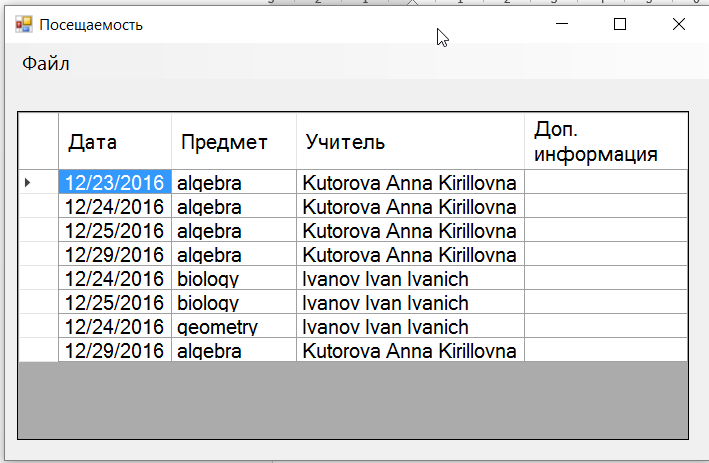


Рисунок 4.16 – Відобразити пропуски уроків для окремого учня

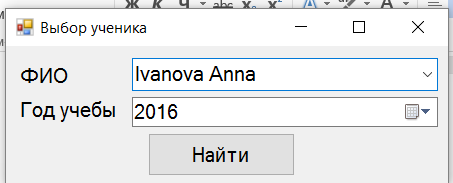


Рисунок 4.17 – Вибір учня

Також програма дозволяє дізнатися статистику пропусків аналогічним чином (див. рис. 4.18). За необхідністю, вчитель може сформувати доповідну записку на цього учня, повідомляючи про занадто велику кількість пропущених занять (див. рис. 4.19) та роздрукувати її.

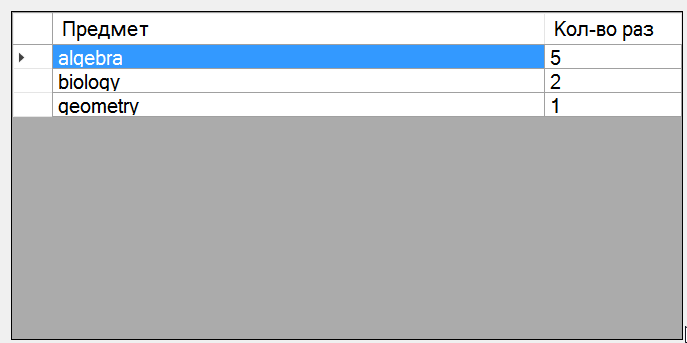


Рисунок 4.18 – Відобразити статистику пропусків

Для кожного учня можна переглянути статистику успішності з предметів, що він вивчає (див. рис. 4.20). Під кінець року вчитель може сформувати та роздрукувати табель (див. рис. 4.21).

SELECT COUNT(marks.value) as times, subjects.name FROM marks, subjects, assigned\_subjects WHERE YEAR(marks.date)=@year AND marks.child\_id=@child\_id AND marks.type=@type AND marks.assigned\_subject\_id=assigned\_subjects.id AND assigned\_subjects.subject\_id=subjects.id GROUP BY subjects.name

Лістинг 4.2 – SQL-запит для отримання статистики пропусків



Рисунок 4.19 – Доповідна записка на учня.

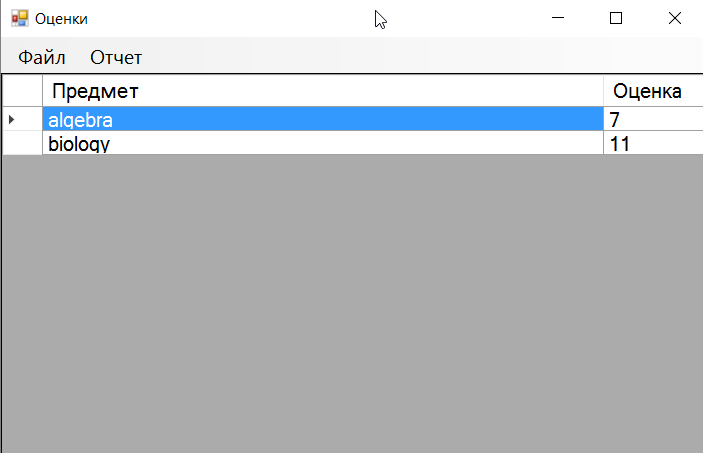


Рисунок 4.20 – Статистика успішності

SELECT AVG(marks.value) as mark FROM marks WHERE YEAR(marks.date)=@year AND marks.child\_id=@child\_id AND marks.type=@type AND marks.assigned\_subject\_id=@assigned\_subject\_id

Лістинг 4.3 – SQL-запит для отримання статистики успішності

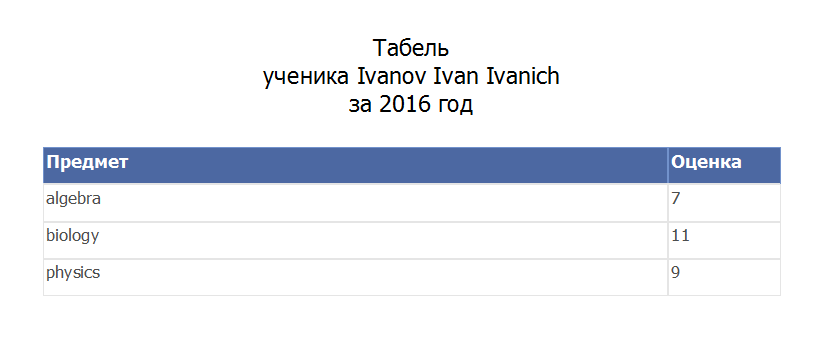


Рисунок 4.21 – Приклад табеля учня

4.6 Опис задачі автоматизації

В шкільнії практиці існує багато прикладів, коли успішність учня з часом знижується. Вчителю необхідно повідомити родичів учня про це, аби вони звернули на це увагу та вжили необхідні на їх думку заходи. Зазвичай для цього вчителю необхідно звернути увагу на попередні оцінки учня, зробити деякі обчислення та прикинути на око чи є поточна оцінка задовільною для учня. Потім вчитель робить відповідний запис про це у щоденник. Нажаль, не всі батьки перевіряють щоденник учня, також відомі випадки, коли учень хоче приховати погану оцінку від батьків. Вирішенням цієї проблеми є задача автоматизації, яка сама робить обчислення, та при необхідності автоматично відправляє повідомлення про зниження успішності усім родичям учня на електронну пошту (див. рис. 4.22).

Алгоритм задачі автоматизації наступний:

1. програма автоматично починає виконувати роботу, як тільки вчитель поставив нову оцінку учню;
2. далі система збирає усі минулі оцінки з цього предмету та знаходить середнє значення оцінки за предмет;
3. якщо різниця між середньою оцінкою та поточною оцінкою значуща, необхідно перейти до наступного кроку відправки писем, інакше нічого відправляти не треба;
4. система шукає в базі всіх родичів учня та формує список з їх електронних пошт;
5. система відправляє лист з вказанням ПІБ учня та проханням звернути увагу на зниження успішності кожному родичю з заздалегіть зареєстрованої електронної пошти.

Таким чином, зникає проблема обчислень для вчителя та повідомлення родичів.

Слід зауважити, що всі обчислення виконуються на фоні програми, тобто користувач може виконувати інші дії та працювати з програмою навіть не підозрюючи, що програма робить розсилку в цей час.

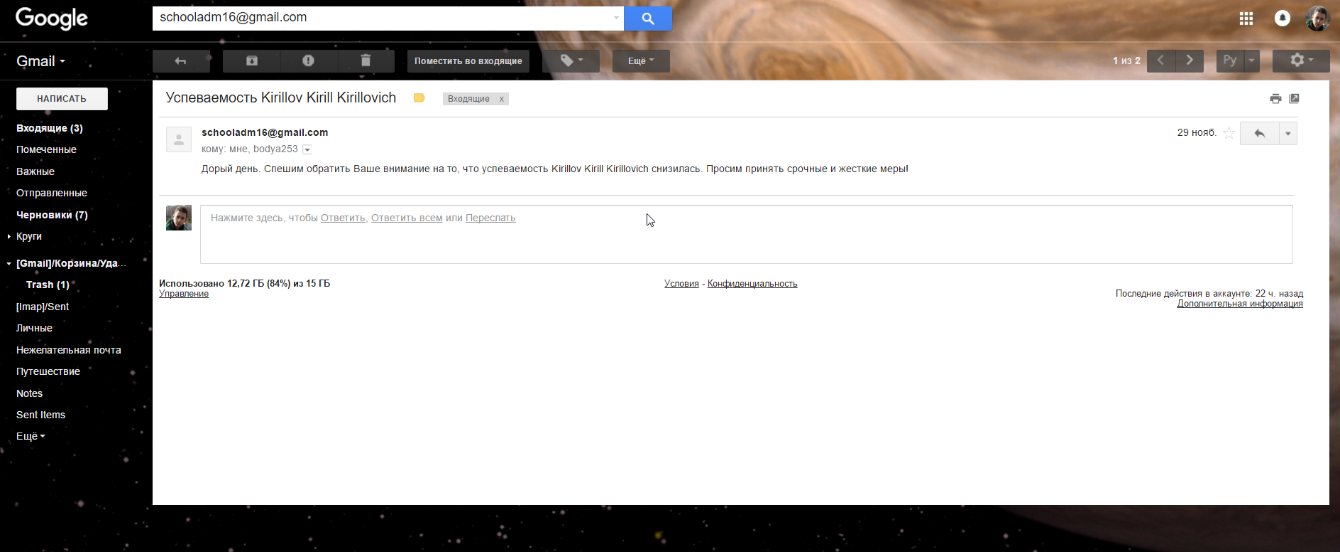


Рисунок 4.22 – Результат виконання задачі автоматизації

Лістинг задачі автоматизації

// Class NotificationManager is used for making notifications.

class NotificationManager

{

// Function for checking if relatives are needed to be notified.

// If they need, it will notify them.

public static void NotifyIfNeeded(int childId, int newMark, int assignedSubjectId)

{

schoolDataSetTableAdapters.marksReportTableAdapter ta = new schoolDataSetTableAdapters.marksReportTableAdapter();

schoolDataSet.marksReportDataTable dt = new schoolDataSet.marksReportDataTable();

int? averageValue = null;

try

{

averageValue = Convert.ToInt32(ta.GetBySubject(DateTime.Now.Year, childId, "оценка", assignedSubjectId));

}

catch { }

if (averageValue == null)

{

return;

}

Console.WriteLine("Compare " + averageValue + " ? " + newMark);

if (averageValue - newMark > 2)

{

NotifyParents(childId);

}

}

// Function for force notification without any checking.

// Finds all mails of parents and send them message.

static void NotifyParents(int childId)

{

schoolDataSetTableAdapters.childsTableAdapter ta = new schoolDataSetTableAdapters.childsTableAdapter();

schoolDataSet.childsDataTable dt = new schoolDataSet.childsDataTable();

ta.FillById(dt, childId);

string childName = Convert.ToString(dt.Rows[0].ItemArray[1]);

schoolDataSetTableAdapters.relatives\_notificationTableAdapter ta2 = new schoolDataSetTableAdapters.relatives\_notificationTableAdapter();

schoolDataSet.relatives\_notificationDataTable dt2 = new schoolDataSet.relatives\_notificationDataTable();

ta2.FillByChildId(dt2, childId);

List<string> mails = new List<string>();

for (int i = 0; i < dt2.Rows.Count; ++i)

{

mails.Add(Convert.ToString(dt2[i].ItemArray[0]));

}

sendMessage(mails, childName);

}

// Function for sending message to the list of parents.

// Forms message for every parent in the list and send it.

static async void sendMessage(List<string> mailTo, string childName)

{

SmtpClient Smtp = new SmtpClient("smtp.gmail.com", 587);

Smtp.Credentials = new NetworkCredential("schooladm16@gmail.com", "school123456");

Smtp.EnableSsl = true;

MailMessage Message = new MailMessage();

Message.From = new MailAddress("schooladm16@gmail.com");

mailTo.ForEach(c => Message.To.Add(new MailAddress(c)));

Message.Subject = "Успеваемость " + childName;

Message.Body = "Дорый день. Спешим обратить Ваше внимание на то, что успеваемость " + childName + " снизилась. Просим принять срочные и жесткие меры!";

try

{

await Smtp.SendMailAsync(Message);

}

catch (SmtpException e)

{

Console.WriteLine("Messages not sent: " + e.Message);

MessageBox.Show(e.Message, "Ошибка при отправке письма.", MessageBoxButtons.OK);

}

Console.WriteLine("Messages sent");

}

}

ВИСНОВКИ

В результаті виконаної роботи була створена інформаційна система «Школа». Для реалізації потрібного функціоналу була проаналізована предметна область, описані взаємозв’язки між об’єктами інформаційної системи. Для зберігання інформації була спроектована та нормалізована база даних, яка відображає всі зв’язки між сутностями даної предметної області. Дана система повністю відповідає потребам користувачів, оскілька вона є швидким та ефективним засобом роботи, що значно зменшую часові затрати при роботі з програмою.

Заздалегідь продуманий зрозумілий інтерфейс дозволяє ефективно працювати з програмою, швидко знаходити всі її функції. Саме тому програма може одразу використовуватися людьми без спецальних знань.

Отримана система задовольняє заявленим до неї вимогам, в ній реалізовані пошук, фільтрація, сортування, задача автоматизації, а також можливість перегляду та формування статистики, завантаження звітів.

Під час написання програми були повторені та засвоєні такі технології: об’єктно-орієнтоване програмування, реляційні бази даних і нормалізація відношень в них, технологія доступу до баз даних, мова запитів SQL та деякі особливості СУБД MS SQL Server.

Система була протестована вчителями. Програма отримала від них відмінні відгуки, що вказує на необхідність повної реалізації цього продукту та подальшої праці над ним.

ПЕРЕЛІК ПОСИЛАНЬ

1. Методичні вказівки до курсової роботи з дисципліни «Бази даних » для студентів усіх форм навчання спеціальності 6.050103 – “Програмна інженерія” [Текст] / Упоряд.: О.О.Мазурова, М.С.Широкопетлєва, Ю.Ю.Черепанова.– Харків: ХНУРЕ, 2015. – 42 с.
2. Мартин Грабер. SQL. [Текст] - К.: Ид-во “ЛОРИ”, 2003. - 644 с.
3. Дейт, К. Дж. Введення у системи баз даних [Текст] / К. Дж. Дейт.- 7-е вид. – М. : Вид. дім «Вільямс», 2001. – 846 с.
4. Джеймс Грофф, Пол Вайнберг, Эндрю Оппель. SQL: полное руководство 3-е издание. [Текст] - М.: Издательский дом «Вильямс», 2015 г. - 959 с.
5. Троелсен: «Мова програмування С# 2005 та платформа .NET 2.0» [Текст]: учеб. пособ, 3е видання. : Пер. з англ. – Москва. : ООО «И.Д. Вільямс», 2007. – 1168с.
6. Теория и практика построения баз данных. [Текст] 8-е изд. / Д.Кренке. – СПб.: Питер, 2003. – 800 с.: ил.