НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ
(повна назва інституту/факультету)

КАФЕДРА інформатики та програмної інженерії
(повна назва кафедри)

# КУРСОВА РОБОТА

з дисципліни «Бази даних»
(назва дисципліни)

на тему: База даних з результатами зовнішнього незалежного оцінювання

Студента (ки) __2__ курсу _ІП-13_____ групи
спеціальності 121 «Інженерія програмного забезпечення»
_Шевцової Анастасії Андріївни_____
(прізвище та ініціали)

Керівник _Ліщук Олександр Васильович_____
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____ Оцінка ECTS _____

Члени комісії

_____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

_____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

_____ _____
(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

Київ – 2022 рік

**Національний технічний університет України**
**«Київський політехнічний інститут імені Ігоря Сікорського»**

Факультет Інформатики та обчислювальної техніки
(повна назва)

Кафедра Інформатики та програмної інженерії
(повна назва)

Дисципліна Бази даних

Курс 2 Група ІП-13 Семестр 3

# З А В Д А Н Н Я
## НА КУРСОВУ РОБОТУ СТУДЕНТУ

Шевцовій Анастасії Андріївні
(прізвище, ім'я, по батькові)

1. Тема роботи: База даних з результатами зовнішнього незалежного оцінювання

керівник роботи Ліщук Олександр Васильович
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

2. Строк подання студентом роботи 10.01.2023

3. Вихідні дані до роботи завдання на розробку бази даних для зберігання результатів зовнішнього незалежного оцінювання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1) Аналіз предметного середовища
2) Побудова ER-моделі
3) Побудова реляційної схеми з ER-моделі
4) Створення бази даних, у форматі обраної системи управління базою даних
5) Створення користувачів бази даних
6) Імпорт даних з використанням засобів СУБД в створену базу даних
7) Створення мовою SQL запитів
8) Оптимізація роботи запитів

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

6. Дата видачі завдання 08.11.2022

# КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів виконання курсового проекту | Строк виконання етапів проекту | Примітка |
|---|---|---|---|
| 1 | Аналіз предметного середовища | 12.11.2022 | |
| 2 | Побудова ER-моделі | 10.01.2023 | |
| 3 | Побудова реляційної схеми з ER-моделі | 10.01.2023 | |
| 4 | Створення бази даних, у форматі обраної системи управління базою даних | 10.01.2023 | |
| 5 | Створення користувачів бази даних | 10.01.2023 | |
| 6 | Імпорт даних з використанням засобів СУБД в створену базу даних | 10.01.2023 | |
| 7 | Створення мовою SQL запитів | 10.01.2023 | |
| 8 | Оптимізація роботи запитів | 10.01.2023 | |
| 9 | Оформлення пояснювальної записки | 10.01.2023 | |
| 10 | Захист курсової роботи | 10.01.2022 | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

**Студент** _____ __Шевцова А. А._____
(підпис ) (прізвище та ініціали)

**Керівник роботи** _____ ___Ліщук О.В._____
(підпис ) (прізвище та ініціали)

# ЗМІСТ

# ВСТУП

Для оптимальної організації даних використовують бази даних. Замість величезних архівів з купою паперів зараз використовують цифрові бази даних, які зберігають дані організовано в таблицях, що пришвидшує пошук потрібної інформації та її обробку.

Дана тема є актуальною, оскільки система зовнішнього незалежного оцінювання уніфіковує результати вступних випробувань, що створює рівні умови доступу до освіти для всіх, незалежно від матеріальних можливостей. А також запобігає корупції на локальному рівні та наближує українську систему освіти до європейських стандартів.

Метою курсової роботи є створення бази даних з результатами зовнішнього незалежного оцінювання, що спростить доступ до даних ЗНО, полегшить процес створення звітів та проведення аналізу.

# 1. ОПИС ПРЕДМЕТНОГО СЕРЕДОВИЩА

Зовнішнє оцінювання проводиться з метою забезпечення прав осіб на рівний доступ до вищої освіти та оцінювання відповідності результатів навчання, здобутих на основі повної загальної середньої освіти, державним вимогам[1].

Кожен учасник ЗНО має інформаційну картку з результатами, яка має унікальний номер, PIN-код, рік отримання та номер Сертифіката зовнішнього незалежного оцінювання.

Дані в інформаційній картці завантажуються з відомості результатів зовнішнього незалежного оцінювання, яка включає номер облікового запису, предмет, кількість тестових балів, результат ЗНО за шкалою 100-200 балів та результат ДПА за шкалою 1-12 балів.

Кожен предмет має назву, максимальну кількість балів, яку можливо набрати, і пов’язаний з таблицею переведення тестових балів ЗНО.

В інформації про учасника ЗНО вказано його прізвище та ініціали, дані про те чи брав участь в тестуванні, місце проходження тестування та вказана сесія ЗНО(основна, додаткова, спеціальна). Місце проходження тестування включає назву області, району та населеного пункту.

За результатами проходження ЗНО формується регіональна статистика. Вона відображає кількість осіб, які зареєструвалися для проходження тестування та ти, які взяли участь у тестуванні. А також показує середній бал ЗНО в даному регіоні.

# 2. ПОСТАНОВКА ЗАВДАННЯ

Метою даної роботи є розробка бази даних з результатами зовнішнього незалежного оцінювання. Тобто організація даних таким чином, щоб доступ до результатів ЗНО та оперування даними було швидким ти ефективним. Отже основні задачі:

Учасник ЗНО повинен мати змогу:

- Зареєструватися до участі у ЗНО
- Переглядати результати ЗНО

Для отримання результатів перевіряючий повинен мати змогу:

- Виставляти бали
- Переглядати дані учасників

Результати іспитів мають автоматично переводитися у шкалу ЗНО 100-200 балів та шкалу ДПА 1-12 балів.

За результатами проведення ЗНО має формуватися загальнодоступна статистика за регіонами.

# 3. ПРОЕКТУВАННЯ БАЗИ ДАНИХ

## 3.1.Побудова ER-моделі

Після аналізу було виділено такі сутності та зв’язки між ними:

### 3.1.1. Бізнес-правила

- Бали ЗНО виставляються по шкалі 100-200
- Бали ДПА виставляються по шкалі 1-12

### 3.1.2. Вибір сутностей

- Учасник ЗНО
- Предмет
- Відомість результатів ЗНО
- Інформаційна картка
- Шкала переведення балів ЗНО та ДПА
- Локація екзамену
- Сесія
- Сертифікат
- Статистика результатів

### 3.1.3. Набори атрибутів сутностей

Таблиця 3.1 – Сутності та їхні атрибути

| Сутність | Атрибут |
|---|---|
| Student | student_id<br>full_name<br>info_card_id<br>participated<br>location_id |
| InformationCard | info_card_id<br>pin_code<br>year_of_receipt<br>certificate_id |
| ResultInfo | res_info_id<br>subject_id<br>test_score<br>result<br>sfe<br>session_name<br>info_card_id |
| Certificate | certificate_id<br>subjects |
| Subject | subject_id<br>subject_name<br>max_test_score |
| ScoreScale | score_scale_id<br>subject_id<br>test_score<br>rating_score<br>sfe |
| Session | session_name |
| TestingLocation | location_id<br>region<br>district<br>settlement |

Продовження таблиці 3.1

| | |
|---|---|
| Stats | stats_id<br>location_id<br>registered_num<br>participated_num<br>average_score |

Сутність Student пов’язана з сутністю InformationCard зв’язком один до одного, бо в кожного учасника є одна інформаційна картка.

Сутність Student пов’язана з сутністю TestingLocation зв’язком багато до одного, бо в одному місці можуть складати ЗНО декілька учасників.

Сутність InformationCard пов’язана з сутністю Certificate зв’язком один до одного, бо до кожної інформаційної картки прикріплений сертифікат з відповідним номером.

Сутність ResultInfo пов’язана з сутністю Subject зв’язком один до багатьох, бо по кожному предмету може бути багато результатів.

Сутність ResultInfo пов’язана з сутністю Session зв’язком один до багатьох, бо по кожній сесії може бути багато результатів.

Сутність ResultInfo пов’язана з сутністю InformationCard зв’язком один до багатьох, бо в кожній інформаційній картці може бути багато результатів.

Сутність ScoreScale пов’язана з сутністю Subject зв’язком багато до одного, бо до кожного предмета є відповідна шкала.

Сутність Stats пов’язана з сутністю TestingLocation зв’язком один до одного, бо статистика формується по кожній локації.

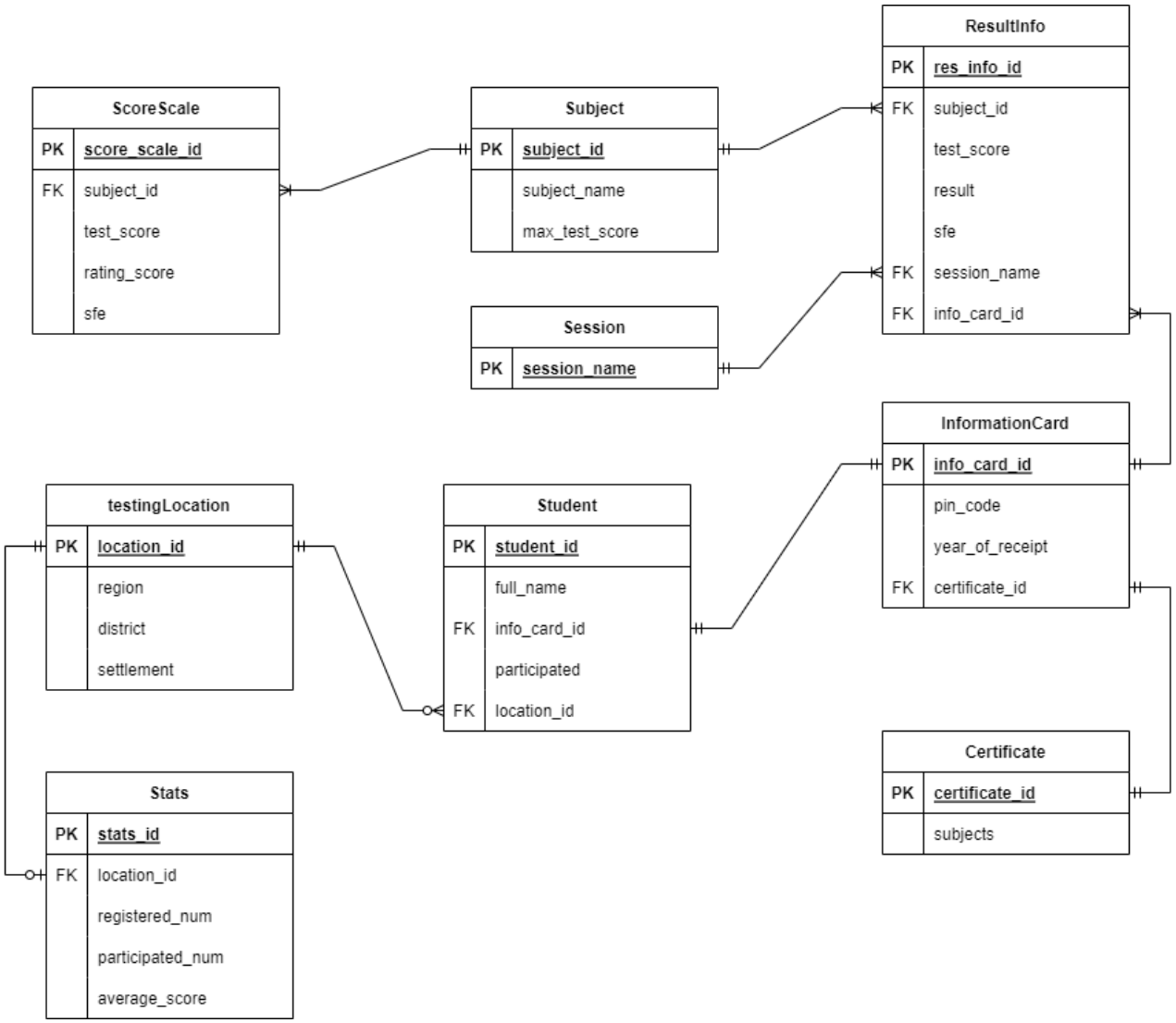


Рисунок 3.1 – ER-діаграма

## 3.2.Даталогічна модель бази даних

### 3.2.1. Побудова необхідних відношень та визначення первинних і зовнішніх ключів

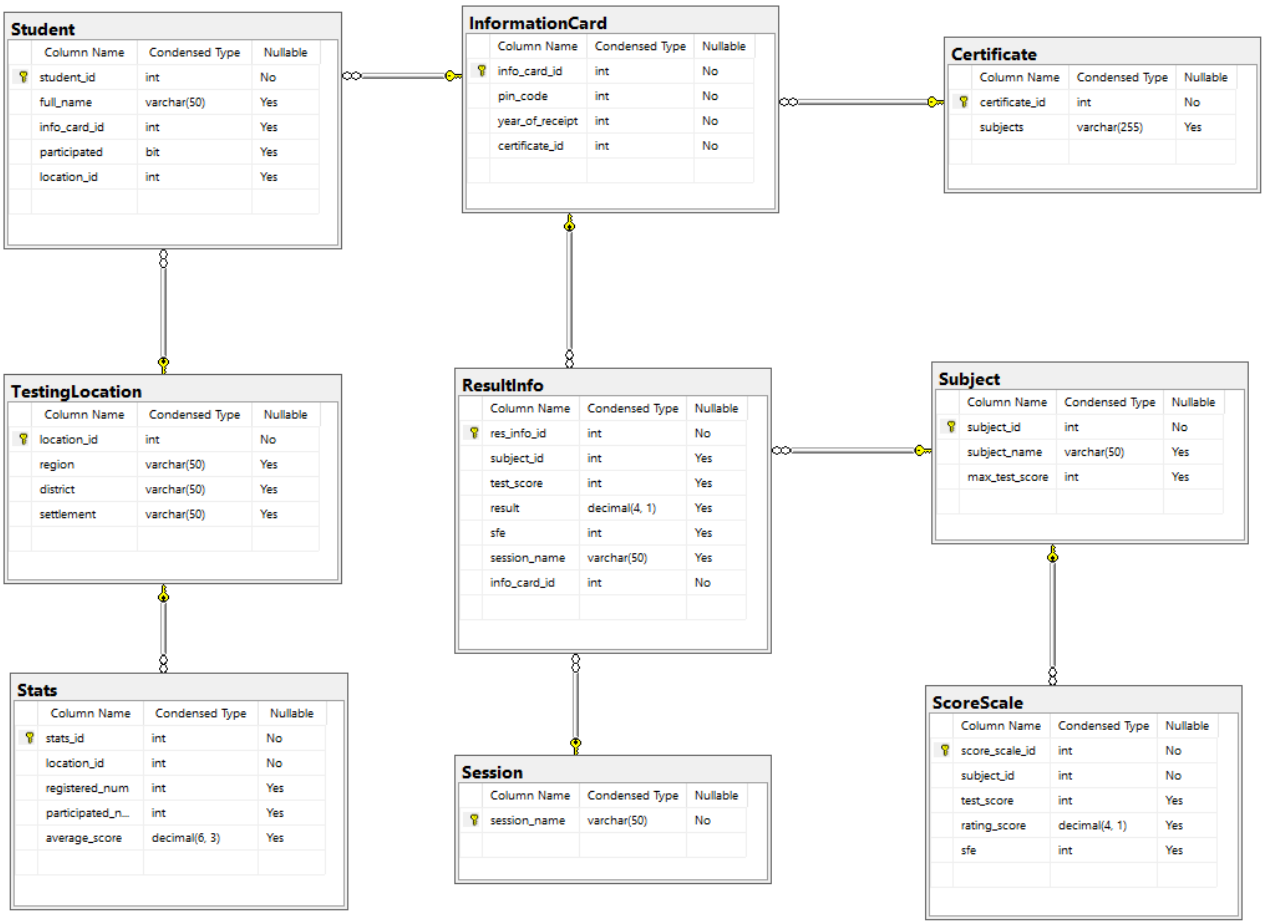


Рисунок 3.2 – Даталогічна модель бази даних

На даній схемі видно, що база даних знаходиться у третій нормальній формі, адже всі поля таблиць декомпозовані та всі атрибути таблиць функціонально повно залежать від первинного ключа, кожен неключовий атрибут не є транзитивно залежним від первинного ключа.

- Визначення обмежень цілісності для спроектованих відношень. Обмеження цілісності: Рядок батьківської таблиці може бути видалений лише у тому випадку, якщо немає зовнішніх ключів, що посилаються на значення преференційного ключа цього рядка. Реалізовано відсутністю параметра ON DELETE {CASCADE|SET NULL} при створенні таблиці, що за умовчанням значитиме ON DELETE RESTRICT;

- Обов’язкові атрибути таблиць мають обмеження NOT NULL, для запобіганню помилок при роботі з даними.

# 4. РЕАЛІЗАЦІЯ БАЗИ ДАНИХ

## 4.1.Створення бази даних у форматі СУБД MS SQL

### 4.1.1. Створення таблиць

```
IF OBJECT_ID ('dbo.Student', 'U') IS NOT NULL
   DROP TABLE Student;
GO
CREATE TABLE Student
(
       student_id INT IDENTITY(1,1) NOT NULL,
       full_name VARCHAR(50),
       info_card_id INT,
       participated BIT,
       location_id INT,
       CONSTRAINT PK_Student_student_id PRIMARY KEY (student_id)
);


IF OBJECT_ID ('dbo.InformationCard', 'U') IS NOT NULL
   DROP TABLE InformationCard;
GO
CREATE TABLE InformationCard
(
       info_card_id INT IDENTITY(1,1) NOT NULL,
       pin_code INT NOT NULL,
       year_of_receipt INT NOT NULL,
       certificate_id INT NOT NULL,
       CONSTRAINT PK_InformationCard_info_card_id PRIMARY KEY (info_card_id)
);


IF OBJECT_ID ('dbo.ResultInfo', 'U') IS NOT NULL
   DROP TABLE ResultInfo;
GO
CREATE TABLE ResultInfo
(
       res_info_id INT IDENTITY(1,1) NOT NULL,
       subject_id INT,
       test_score INT,
       result DECIMAL(4,1),
       sfe INT,
       session_name VARCHAR(50),
       info_card_id INT NOT NULL,
       CONSTRAINT PK_ResultInfo_res_info_id PRIMARY KEY (res_info_id)
);


IF OBJECT_ID ('dbo.Certificate', 'U') IS NOT NULL
   DROP TABLE "Certificate";
GO
CREATE TABLE "Certificate"
(
       certificate_id INT IDENTITY(1,1) NOT NULL,
       subjects VARCHAR(255),
       CONSTRAINT PK_Certificate_certificate_id PRIMARY KEY (certificate_id)
);


IF OBJECT_ID ('dbo.Subject', 'U') IS NOT NULL
```

```
   DROP TABLE "Subject";
GO
CREATE TABLE "Subject"
(
	subject_id INT IDENTITY(1,1) NOT NULL,
	subject_name VARCHAR(50),
	max_test_score INT,
	CONSTRAINT PK_Subject_subject_id PRIMARY KEY (subject_id)
);


IF OBJECT_ID ('dbo.ScoreScale', 'U') IS NOT NULL
   DROP TABLE ScoreScale;
GO
CREATE TABLE ScoreScale
(
	score_scale_id INT IDENTITY(1,1) NOT NULL,
	subject_id INT NOT NULL,
	test_score INT,
	rating_score DECIMAL(4,1),
	sfe INT,
	CONSTRAINT PK_ScoreScale_score_scale_id PRIMARY KEY (score_scale_id)
);


IF OBJECT_ID ('dbo.Session', 'U') IS NOT NULL
   DROP TABLE "Session";
GO
CREATE TABLE "Session"
(
	session_name VARCHAR(50) NOT NULL,
	CONSTRAINT PK_Session_session_name PRIMARY KEY (session_name)
);


IF OBJECT_ID ('dbo.TestingLocation', 'U') IS NOT NULL
   DROP TABLE TestingLocation;
GO
CREATE TABLE TestingLocation
(
	location_id INT IDENTITY(1,1) NOT NULL,
	region VARCHAR(50),
	district VARCHAR(50),
	settlement VARCHAR(50),
	CONSTRAINT PK_TestingLocation_location_id PRIMARY KEY (location_id)
);


IF OBJECT_ID ('dbo.Stats', 'U') IS NOT NULL
   DROP TABLE "Stats";
GO
CREATE TABLE "Stats"
(
	stats_id INT IDENTITY(1,1) NOT NULL,
	location_id INT NOT NULL,
	registered_num INT,
	participated_num INT,
	average_score DECIMAL(6,3),
	CONSTRAINT PK_Stats_stats_id PRIMARY KEY (stats_id)
);
```

### 4.1.2. Створення зав’язків

```
ALTER TABLE Student
```

```
	ADD CONSTRAINT FK_Student_info_card_id FOREIGN KEY (info_card_id)
		REFERENCES InformationCard (info_card_id)
		ON DELETE CASCADE
		ON UPDATE CASCADE,
		CONSTRAINT FK_Student_location_id FOREIGN KEY (location_id)
		REFERENCES TestingLocation (location_id)
		ON DELETE SET NULL
		ON UPDATE CASCADE;


ALTER TABLE InformationCard
	ADD CONSTRAINT FK_InformationCard_certificate_id FOREIGN KEY (certificate_id)
		REFERENCES "Certificate" (certificate_id)
		ON DELETE CASCADE
		ON UPDATE CASCADE;


ALTER TABLE ResultInfo
	ADD CONSTRAINT FK_ResultInfo_subject_id FOREIGN KEY (subject_id)
		REFERENCES "Subject" (subject_id)
		ON DELETE SET NULL
		ON UPDATE CASCADE,
		CONSTRAINT FK_ResultInfo_info_card_id FOREIGN KEY (info_card_id)
		REFERENCES InformationCard (info_card_id)
		ON DELETE CASCADE
		ON UPDATE CASCADE,
		CONSTRAINT FK_ResultInfo_session_name FOREIGN KEY (session_name)
		REFERENCES "Session" (session_name)
		ON DELETE CASCADE
		ON UPDATE CASCADE,
		CHECK (result >= 100 AND result <= 200),
		CHECK (sfe >= 1 AND sfe <= 12);


ALTER TABLE ScoreScale
	ADD CONSTRAINT FK_ScoreScale_subject_id FOREIGN KEY (subject_id)
		REFERENCES "Subject" (subject_id)
		ON DELETE CASCADE
		ON UPDATE CASCADE,
		CHECK (rating_score >= 100 AND rating_score <= 200),
		CHECK (sfe >= 1 AND sfe <= 12);


ALTER TABLE "Stats"
	ADD CONSTRAINT FK_Stats_location_id FOREIGN KEY (location_id)
		REFERENCES TestingLocation (location_id)
		ON DELETE CASCADE
		ON UPDATE CASCADE;
```

### 4.1.3. Створення обмежень

Обмеження для таблиці ResultInfo:

```
CHECK (result >= 100 AND result <= 200),
CHECK (sfe >= 1 AND sfe <= 12);
```

Обмеження для таблиці ScoreScale:

```
CHECK (rating_score >= 100 AND rating_score <= 200),
CHECK (sfe >= 1 AND sfe <= 12);
```

## 4.2.Імпортування даних у таблиці

Імпортувати дані у вибрану мною СУБД можна двома способами:

1. Імпортувати дані з файлу csv формату;
2. Імпортувати дані із файлу-скрипта sql формату.

Для першого методу створено .csv файли для кожної таблиці.

Для другого методу створено файл DownloadData.sql. Його вміст:

```
INSERT INTO [dbo].[Stats] (location_id, [registered_num])
SELECT [dbo].[TestingLocation].[location_id], COUNT([dbo].[Student].[student_id])
FROM [dbo].[TestingLocation] JOIN [dbo].[Student]
ON [dbo].[TestingLocation].location_id = [dbo].[Student].location_id
GROUP BY [dbo].[TestingLocation].[location_id];

UPDATE [dbo].[Stats]
SET [participated_num] = (SELECT COUNT([student_id]) FROM [dbo].[Student]
WHERE ([dbo].[Stats].location_id = [dbo].[Student].location_id) AND
[dbo].[Student].participated = 1);

UPDATE [dbo].[Stats]
SET [average_score] = (SELECT AVG([result]) FROM
(SELECT [dbo].[Student].location_id, [dbo].[ResultInfo].result
FROM [dbo].[Student] JOIN [dbo].[ResultInfo]
ON [dbo].[Student].info_card_id = [dbo].[ResultInfo].info_card_id)
AS ResByLocation
	WHERE ResByLocation.location_id = [dbo].[Stats].location_id);
```

# 5. РОБОТА З БАЗОЮ ДАНИХ

## 5.1.Створення користувачів

### 5.1.1. Учасник ЗНО

```
CREATE LOGIN student WITH PASSWORD = '<password123>';
GO

CREATE ROLE student;

GRANT SELECT ON [dbo].[Certificate] TO student;
GRANT SELECT ON [dbo].[InformationCard] TO student;
GRANT SELECT ON [dbo].[ResultInfo] TO student;
GRANT SELECT ON [dbo].[Stats] TO student;
GRANT SELECT, INSERT ON [dbo].[Student] TO student;
GRANT INSERT ON [dbo].[TestingLocation] TO student;

CREATE USER new_student FOR LOGIN student;
ALTER ROLE student ADD MEMBER new_student;
```

### 5.1.2. Перевіряючий

```
CREATE LOGIN examiner WITH PASSWORD = '<password123>';
GO

CREATE ROLE examiner;

GRANT SELECT ON [dbo].[Certificate] TO examiner;
GRANT SELECT ON [dbo].[InformationCard] TO examiner;
GRANT SELECT, INSERT, UPDATE, DELETE ON [dbo].[ResultInfo] TO examiner;
GRANT SELECT ON [dbo].[ScoreScale] TO examiner;
GRANT SELECT ON [dbo].[Session] TO examiner;
GRANT SELECT ON [dbo].[Stats] TO examiner;
GRANT SELECT, INSERT, UPDATE, DELETE ON [dbo].[Student] TO examiner;
GRANT SELECT ON [dbo].[Subject] TO examiner;
GRANT SELECT ON [dbo].[TestingLocation] TO examiner;

CREATE USER new_examiner FOR LOGIN examiner;
ALTER ROLE examiner ADD MEMBER new_examiner;
```

## 5.2.SQL запити

### 5.2.1. Генератори

Під час створення таблиць були створені генератори типу IDENTITY(1,1) для primary key. Тому при додаванні нових значень у таблиці нам не потрібно вручну додавати значення.

Наприклад, для заповнення таблиці Stats використовується запит INSERT:

```
INSERT INTO [dbo].[Stats] (location_id, [registered_num])
SELECT [dbo].[TestingLocation].[location_id], COUNT([dbo].[Student].[student_id])
FROM [dbo].[TestingLocation] JOIN [dbo].[Student]
ON [dbo].[TestingLocation].location_id = [dbo].[Student].location_id
```

```
GROUP BY [dbo].[TestingLocation].[location_id];
```

При виконанні даного запиту поле stats_id заповнюється автоматично. В результаті маємо:

| | stats_id | location_id | registered_num | participated_num | average_score |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 200.000 |
| 2 | 2 | 2 | 2 | 2 | 184.500 |
| 3 | 3 | 3 | 2 | 0 | NULL |
| 4 | 4 | 5 | 1 | 1 | 146.000 |
| 5 | 5 | 6 | 2 | 2 | 177.571 |
| 6 | 6 | 7 | 1 | 1 | 183.333 |
| 7 | 7 | 9 | 1 | 1 | 192.000 |
| 8 | 8 | 10 | 2 | 2 | 183.200 |
| 9 | 9 | 11 | 1 | 0 | NULL |
| 10 | 10 | 12 | 3 | 3 | 178.700 |

Рисунок 5.1 – Приклад роботи генераторів

### 5.2.2. Збережені процедури та функції

1) Процедура для автоматичного розрахунку балів ЗНО та ДПА

```
CREATE PROC CountScore
AS
UPDATE [dbo].[ResultInfo]
SET
[result] = [dbo].[ScoreScale].rating_score,
[sfe] = [dbo].[ScoreScale].[sfe]
FROM [dbo].[ScoreScale]
   WHERE [dbo].[ResultInfo].test_score = [dbo].[ScoreScale].test_score AND
   [dbo].[ResultInfo].subject_id = [dbo].[ScoreScale].subject_id;
```

| | res_info_id | subject_id | test_score | result | sfe | session_name | info_card_id |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 11 | 82 | 200.0 | 12 | основна | 11 |
| 2 | 2 | 2 | 28 | 101.0 | 3 | основна | 1 |
| 3 | 3 | 1 | 65 | 190.0 | 11 | основна | 13 |
| 4 | 4 | 13 | 38 | 152.0 | 7 | додаткова | 10 |
| 5 | 5 | 4 | 8 | NULL | 3 | основна | 2 |
| 6 | 6 | 7 | 72 | 200.0 | 12 | основна | 26 |
| 7 | 7 | 2 | 102 | 190.0 | 11 | основна | 20 |
| 8 | 8 | 1 | 15 | NULL | 3 | основна | 17 |
| 9 | 9 | 3 | 19 | 129.0 | 4 | основна | 10 |
| 10 | 10 | 6 | 42 | 146.0 | 7 | основна | 12 |

Рисунок 5.2 – Приклад роботи збереженої процедури

2) Функція для отримання середнього балу ЗНО за id учасника

```
CREATE FUNCTION dbo.AvgScoreById(@id INT)
RETURNS DECIMAL(6,3)
AS
BEGIN
```

```
        DECLARE @avg_score int;
        SET @avg_score = (SELECT AVG(result)
        FROM [dbo].[ResultInfo] JOIN [dbo].[Student]
        ON [dbo].[ResultInfo].info_card_id = [dbo].[Student].info_card_id
        WHERE [dbo].[Student].student_id = @id);
        RETURN @avg_score;
END;
```

За допомогою SQL-скрипту: `SELECT dbo.AvgScoreById(3) AS average_score` перевіримо роботу функції

| | average_score |
|---|---|
| 1 | 187.000 |

Рисунок 5.3 – Приклад роботи функції

### 5.2.3. Тригери

1) Тригер на додавання нового учасника. При цьому в таблиці зі статистикою кількість зареєстрованих збільшується на 1

```
CREATE TRIGGER Student_insert
ON [dbo].[Student]
AFTER INSERT
AS
BEGIN
        UPDATE [dbo].[Stats]
        SET [dbo].[Stats].registered_num = [dbo].[Stats].registered_num + 1
        FROM [dbo].[Stats] JOIN [dbo].[Student]
        ON [dbo].[Student].location_id = [dbo].[Stats].location_id
END;
```

| | stats_id | location_id | registered_num | participated_num | average_score |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 200.000 |

Рис 5.4 – Поле таблиці Stats до додавання учасника

| | stats_id | location_id | registered_num | participated_num | average_score |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 2 | 1 | 200.000 |

Рис 5.5 – Поле таблиці Stats після додавання учасника

2) Тригер на видалення учасника. При цьому в таблиці зі статистикою кількість зареєстрованих зменшується на 1

```
CREATE TRIGGER Student_delete
ON [dbo].[Student]
AFTER DELETE
AS
BEGIN
        UPDATE [dbo].[Stats]
        SET [dbo].[Stats].registered_num = [dbo].[Stats].registered_num - 1
        FROM [dbo].[Stats] JOIN [dbo].[Student]
        ON [dbo].[Student].location_id = [dbo].[Stats].location_id
END;
```

| | stats_id | location_id | registered_num | participated_num | average_score |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 2 | 1 | 200.000 |

Рис 5.6 – Поле таблиці Stats до видалення учасника

| | stats_id | location_id | registered_num | participated_num | average_score |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 200.000 |

Рис 5.7 – Поле таблиці Stats після видалення учасника

### 5.2.4. Представлення

Представлення для id учасника, номеру його інформаційної картки, назви предмету, кількості балів, максимальної кількості балів з даного предмету, результату в шкалі 100-200 балів та шкалі 1-12 балів

```
CREATE VIEW results AS
SELECT
[dbo].[Student].student_id, [dbo].[InformationCard].info_card_id,
[dbo].[Subject].subject_name, [dbo].[ResultInfo].res_info_id,
[dbo].[ResultInfo].test_score, [dbo].[Subject].max_test_score, [dbo].[ResultInfo].result,
[dbo].[ResultInfo].sfe
FROM
[dbo].[Student] JOIN [dbo].[InformationCard] ON [dbo].[Student].info_card_id =
[dbo].[InformationCard].info_card_id
JOIN [dbo].[ResultInfo] ON [dbo].[ResultInfo].info_card_id =
[dbo].[InformationCard].info_card_id
JOIN [dbo].[Subject] ON [dbo].[Subject].subject_id = [dbo].[ResultInfo].subject_id
```

За допомогою SQL-скрипту: `SELECT * FROM results ORDER BY student_id` отримаємо

| | student_id | info_card_id | subject_name | res_info_id | test_score | max_test_score | result | sfe |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 19 | математика (завдання рівня стандарту) | 28 | 38 | 50 | NULL | 10 |
| 2 | 1 | 19 | хімія | 49 | 74 | 74 | 200.0 | 12 |
| 3 | 1 | 19 | географія | 67 | 90 | 90 | 200.0 | 12 |
| 4 | 1 | 19 | французька мова | 82 | 58 | 72 | 176.0 | 10 |
| 5 | 2 | 1 | українська мова і література | 2 | 28 | 116 | 101.0 | 3 |
| 6 | 2 | 1 | фізика | 18 | 56 | 64 | 191.0 | 11 |
| 7 | 2 | 1 | математика (завдання рівня стандарту) | 53 | 50 | 50 | NULL | 12 |
| 8 | 3 | 23 | хімія | 33 | 74 | 74 | 200.0 | 12 |
| 9 | 3 | 23 | хімія | 48 | 74 | 74 | 200.0 | 12 |
| 10 | 3 | 23 | математика | 97 | 37 | 67 | 163.0 | 7 |

Рисунок 5.8 – Приклад роботи представлення

### 5.2.5. SQL-запити

1) Запит з використанням JOIN для відображення балів ЗНО по всіх предметах для кожного учасника. А також сортування за прізвищем

```
SELECT
[dbo].[Student].full_name, [dbo].[Subject].subject_name, [dbo].[ResultInfo].result
FROM
[dbo].[Student] JOIN [dbo].[ResultInfo] ON [dbo].[Student].info_card_id =
[dbo].[ResultInfo].info_card_id
JOIN [dbo].[Subject] ON [dbo].[Subject].subject_id = [dbo].[ResultInfo].subject_id
ORDER BY full_name
```

| | full_name | subject_name | result |
|---|---|---|---|
| 1 | Бадига Б. В. | українська мова і література | 190.0 |
| 2 | Бадига Б. В. | іспанська мова | 200.0 |
| 3 | Бадига Б. В. | українська мова | 187.0 |
| 4 | Бадига Б. В. | хімія | 196.0 |
| 5 | Басараб С. А. | іспанська мова | 139.0 |
| 6 | Басараб С. А. | історія України | 143.0 |
| 7 | Басараб С. А. | математика | 192.0 |
| 8 | Бойко Б. Д. | хімія | 200.0 |
| 9 | Бойко Б. Д. | хімія | 200.0 |
| 10 | Бойко Б. Д. | математика | 163.0 |
| 11 | Волокита М. А. | хімія | 200.0 |
| 12 | Волокита М. А. | англійська мова | 200.0 |

Рисунок 5.9 – Результат запиту

2) Запит з використанням JOIN для відображення даних інформаційної картки та сертифікати для кожного учасника

```
SELECT
[dbo].[Student].full_name, [dbo].[Student].info_card_id, [dbo].[InformationCard].pin_code,
[dbo].[InformationCard].year_of_receipt, [dbo].[Certificate].subjects
FROM
[dbo].[Student] JOIN [dbo].[InformationCard] ON [dbo].[Student].info_card_id =
[dbo].[InformationCard].info_card_id
JOIN [dbo].[Certificate] ON [dbo].[InformationCard].certificate_id =
[dbo].[Certificate].certificate_id
```

| | full_name | info_card_id | pin_code | year_of_receipt | subjects |
|---|---|---|---|---|---|
| 1 | Потаскалов О. В. | 19 | 4316 | 2017 | українська мова; математика; хімія; фізика |
| 2 | Горошко М. І. | 1 | 8468 | 2021 | українська мова; математика; хімія; англійська мова |
| 3 | Бойко Б. Д. | 23 | 9612 | 2017 | українська мова; математика; фізика |
| 4 | Басараб С. А. | 29 | 7540 | 2021 | українська мова і література; математика; історія У... |
| 5 | Ліщук А. О. | 26 | 6186 | 2021 | українська мова і література; математика (завданн... |
| 6 | Радіченко Н. О. | 27 | 9388 | 2021 | українська мова; математика; історія України |
| 7 | Бадига Б. В. | 20 | 6064 | 2020 | українська мова і література; математика; історія У... |
| 8 | Жадан В. В. | 8 | 5281 | 2020 | українська мова; математика; англійська мова; фр... |
| 9 | Саварина В. С. | 3 | 3429 | 2022 | українська мова і література; математика (завданн... |
| 10 | Тютюнник М. Р. | 2 | 3418 | 2021 | українська мова; математика; фізика |

Рисунок 5.10 – Результат запиту

3) Запит з використанням JOIN, функції SUM() та GROUP BY для відображення кількості зареєстрованих та тих, хто взяли участь у ЗНО по областях

```
SELECT
[dbo].[TestingLocation].region, SUM([dbo].[Stats].registered_num) AS registered,
SUM([dbo].[Stats].participated_num) AS participated
FROM
[dbo].[TestingLocation] JOIN [dbo].[Stats] ON [dbo].[TestingLocation].location_id =
[dbo].[Stats].location_id
GROUP BY region
```

| | region | registered | participated |
|---|---|---|---|
| 1 | Волинська область | 2 | 2 |
| 2 | Дніпропетровська область | 3 | 3 |
| 3 | Донецька область | 1 | 1 |
| 4 | Житомирська область | 3 | 3 |
| 5 | Івано-Франківська область | 2 | 0 |
| 6 | Київська область | 2 | 2 |
| 7 | Кіровоградська область | 2 | 2 |
| 8 | Львівська область | 1 | 1 |
| 9 | м.Київ | 3 | 3 |
| 10 | Миколаївська область | 1 | 1 |
| 11 | Одеська область | 1 | 1 |
| 12 | Полтавська область | 1 | 1 |
| 13 | Тернопільська область | 2 | 1 |
| 14 | Харківська область | 1 | 1 |
| 15 | Херсонська область | 1 | 0 |
| 16 | Хмельницька область | 3 | 3 |
| 17 | Чернігівська область | 1 | 1 |

Рисунок 5.11 – Результат запиту

4) Запит з використанням JOIN, функції AVG() та GROUP BY для відображення середнього балу ЗНО по областях

```
SELECT
[dbo].[TestingLocation].region, AVG([dbo].[Stats].average_score) AS average_score
FROM
[dbo].[TestingLocation] JOIN [dbo].[Stats] ON [dbo].[TestingLocation].location_id =
[dbo].[Stats].location_id
GROUP BY region
```

| | region | average_score |
|---|---|---|
| 1 | Волинська область | 177.571000 |
| 2 | Дніпропетровська область | 184.888666 |
| 3 | Донецька область | 198.000000 |
| 4 | Житомирська область | 151.300000 |
| 5 | Івано-Франківська область | NULL |
| 6 | Київська область | 183.200000 |
| 7 | Кіровоградська область | 184.500000 |
| 8 | Львівська область | 183.333000 |
| 9 | м.Київ | 157.055666 |
| 10 | Миколаївська область | 169.250000 |
| 11 | Одеська область | NULL |
| 12 | Полтавська область | 165.750000 |
| 13 | Тернопільська область | 193.250000 |
| 14 | Харківська область | 152.667000 |
| 15 | Херсонська область | NULL |
| 16 | Хмельницька область | 178.700000 |
| 17 | Чернігівська область | 192.000000 |

Рисунок 5.12 – Результат запиту

5) Запит з використанням JOIN, функції AVG() та GROUP BY для відображення середнього балу ЗНО по предметах

```
SELECT
[dbo].[Subject].subject_name, AVG([dbo].[ResultInfo].result) AS average_score
FROM
[dbo].[Subject] JOIN [dbo].[ResultInfo] ON [dbo].[Subject].subject_id =
[dbo].[ResultInfo].subject_id
GROUP BY subject_name
```

| | subject_name | average_score |
|---|---|---|
| 1 | англійська мова | 181.000000 |
| 2 | біологія | 174.333333 |
| 3 | географія | 176.428571 |
| 4 | іспанська мова | 171.888888 |
| 5 | історія України | 182.200000 |
| 6 | математика | 170.250000 |
| 7 | математика (завдання рівня стандарту) | NULL |
| 8 | німецька мова | 152.750000 |
| 9 | українська мова | 186.777777 |
| 10 | українська мова і література | 139.571428 |
| 11 | фізика | 198.166666 |
| 12 | французька мова | 166.500000 |
| 13 | хімія | 188.666666 |

Рисунок 5.13 – Результат запиту

6) Запит з використанням JOIN, функції COUNT() та GROUP BY для відображення кількості людей, які отримали максимальний бал ЗНО по предметах

```
SELECT
[dbo].[Subject].subject_name, COUNT([dbo].[ResultInfo].result) AS max_num
FROM
[dbo].[Subject] JOIN [dbo].[ResultInfo] ON [dbo].[Subject].subject_id =
[dbo].[ResultInfo].subject_id
WHERE [dbo].[ResultInfo].result = 200
GROUP BY subject_name
```

| | subject_name | max_num |
|---|---|---|
| 1 | англійська мова | 3 |
| 2 | біологія | 3 |
| 3 | географія | 2 |
| 4 | іспанська мова | 3 |
| 5 | математика | 1 |
| 6 | німецька мова | 1 |
| 7 | українська мова | 3 |
| 8 | фізика | 4 |
| 9 | французька мова | 2 |
| 10 | хімія | 5 |

Рисунок 5.14 – Результат запиту

7) Запит з використанням JOIN, функції COUNT() та GROUP BY для відображення кількості людей в кожній сесії

```
SELECT
[dbo].[ResultInfo].session_name, COUNT([dbo].[Student].student_id) AS stud_num
FROM [dbo].[ResultInfo] JOIN [dbo].[Student] ON [dbo].[ResultInfo].info_card_id =
[dbo].[Student].info_card_id
GROUP BY session_name
```

| | session_name | stud_num |
|---|---|---|
| 1 | додаткова | 5 |
| 2 | основна | 82 |
| 3 | спеціальна | 1 |

Рисунок 5.15 – Результат запиту

8) Запит з використанням JOIN, функції COUNT() та GROUP BY для відображення кількості людей, які писали екзамен по предметах

```
SELECT
[dbo].[Subject].subject_name, COUNT([dbo].[ResultInfo].subject_id) AS stud_num
FROM [dbo].[Subject] JOIN [dbo].[ResultInfo] ON [dbo].[Subject].subject_id =
[dbo].[ResultInfo].subject_id
GROUP BY subject_name
```

| | subject_name | stud_num |
|---|---|---|
| 1 | англійська мова | 7 |
| 2 | біологія | 6 |
| 3 | географія | 7 |
| 4 | іспанська мова | 9 |
| 5 | історія України | 6 |
| 6 | математика | 8 |
| 7 | математика (завдання рівня стандарту) | 7 |
| 8 | німецька мова | 6 |
| 9 | українська мова | 11 |
| 10 | українська мова і література | 7 |
| 11 | фізика | 7 |
| 12 | французька мова | 10 |
| 13 | хімія | 9 |

Рисунок 5.16 – Результат запиту

9) Запит з використанням JOIN для отримання шкали балів для предмета по його індексу

```
SELECT [dbo].[Subject].subject_name, [dbo].[ScoreScale].test_score,
[dbo].[ScoreScale].rating_score, [dbo].[ScoreScale].sfe
FROM [dbo].[Subject] JOIN [dbo].[ScoreScale] ON [dbo].[Subject].subject_id =
[dbo].[ScoreScale].subject_id
WHERE [dbo].[Subject].subject_id = 2
```

| | subject_name | test_score | rating_score | sfe |
|---|---|---|---|---|
| 25 | українська мова і література | 25 | NULL | 3 |
| 26 | українська мова і література | 26 | NULL | 3 |
| 27 | українська мова і література | 27 | 100.0 | 3 |
| 28 | українська мова і література | 28 | 101.0 | 3 |
| 29 | українська мова і література | 29 | 102.0 | 4 |
| 30 | українська мова і література | 30 | 104.0 | 4 |
| 31 | українська мова і література | 31 | 105.0 | 4 |
| 32 | українська мова і література | 32 | 106.0 | 4 |
| 33 | українська мова і література | 33 | 108.0 | 4 |
| 34 | українська мова і література | 34 | 109.0 | 4 |
| 35 | українська мова і література | 35 | 110.0 | 4 |
| 36 | українська мова і література | 36 | 112.0 | 4 |
| 37 | українська мова і література | 37 | 113.0 | 4 |

Рисунок 5.17 – Результат запиту

10) Запит з використанням JOIN та функції COUNT() для отримання кількості учасників, які отримали 200 балів

```
SELECT COUNT(DISTINCT [student_id]) AS stud_num
FROM [dbo].[Student]
JOIN  [dbo].[ResultInfo] ON [dbo].[ResultInfo].info_card_id = [dbo].[Student].info_card_id
WHERE [dbo].[ResultInfo].result = 200
```

| | stud_num |
|---|---|
| 1 | 16 |

Рисунок 5.18 – Результат запиту

11) Запит з використанням JOIN для отримання імен учасників, які складали ЗНО у 2022 році

```
SELECT full_name
FROM [dbo].[Student]
JOIN [dbo].[InformationCard] ON [dbo].[Student].info_card_id =
[dbo].[InformationCard].info_card_id
WHERE [dbo].[InformationCard].year_of_receipt = 2022
```

| | full_name |
|---|---|
| 1 | Саварина В. С. |
| 2 | Мельник Г. Ю. |
| 3 | Черкасов А. С. |

Рисунок 5.19 – Результат запиту

12) Запит з використанням JOIN та функції COUNT() для отримання кількості учасників, які складали ЗНО по роках

```
SELECT
[dbo].[InformationCard].year_of_receipt, COUNT([dbo].[Student].student_id) AS stud_num
FROM
[dbo].[InformationCard] JOIN [dbo].[Student] ON [dbo].[Student].info_card_id =
[dbo].[InformationCard].info_card_id
GROUP BY year_of_receipt
```

| | year_of_receipt | stud_num |
|---|---|---|
| 1 | 2016 | 5 |
| 2 | 2017 | 4 |
| 3 | 2018 | 1 |
| 4 | 2019 | 4 |
| 5 | 2020 | 2 |
| 6 | 2021 | 7 |
| 7 | 2022 | 3 |

Рисунок 5.20 – Результат запиту

13) Запит з використанням JOIN для отримання імен учасників, які зареєструвалися, але не взяли участь у ЗНО із зазначенням регіону

```
SELECT [dbo].[Student].full_name,
([dbo].[TestingLocation].region + ', ' + [dbo].[TestingLocation].district + ', ' +
[dbo].[TestingLocation].settlement) AS region
FROM [dbo].[Student] JOIN [dbo].[TestingLocation] ON [dbo].[Student].location_id =
[dbo].[TestingLocation].location_id
```

```
WHERE [dbo].[Student].participated = 0
```

| | full_name | region |
|---|---|---|
| 1 | Хвиць М. С. | Івано-Франківська область, Долинський район, м.Долина |
| 2 | Морква Д. Л. | Тернопільська область, Тернопільська область, м.Тернопіль |
| 3 | Гронтковський Б. О. | Івано-Франківська область, Долинський район, м.Долина |
| 4 | Поліщук А. В. | Херсонська область, Чаплинський район, смт Чаплинка |

Рисунок 5.21 – Результат запиту

14) Запит із використанням JOIN, функції SUM(), GROUP BY та ORDER BY для сортування учасників за сумою балів ЗНО

```
SELECT [dbo].[Student].full_name, SUM([dbo].[ResultInfo].result) AS sum_result
FROM [dbo].[Student] JOIN [dbo].[ResultInfo] ON [dbo].[Student].info_card_id =
[dbo].[ResultInfo].info_card_id
GROUP BY full_name
ORDER BY sum_result DESC
```

| | full_name | sum_result |
|---|---|---|
| 1 | Радіченко Н. О. | 781.0 |
| 2 | Бадига Б. В. | 773.0 |
| 3 | Мельник Г. Ю. | 697.0 |
| 4 | Малий О. С. | 694.0 |
| 5 | Черкасов А. С. | 677.0 |
| 6 | Гричина Н. А. | 663.0 |
| 7 | Потаскалов О. В. | 576.0 |
| 8 | Бойко Б. Д. | 563.0 |
| 9 | Саварина В. С. | 550.0 |
| 10 | Ганчар О. М. | 538.0 |

Рисунок 5.22 – Результат запиту

15) Запит із використанням JOIN, функції AVG(), GROUP BY та ORDER BY для сортування учасників за середнім балом ЗНО

```
SELECT [dbo].[Student].full_name, AVG([dbo].[ResultInfo].result) AS avg_result
FROM [dbo].[Student] JOIN [dbo].[ResultInfo] ON [dbo].[Student].info_card_id =
[dbo].[ResultInfo].info_card_id
GROUP BY full_name
ORDER BY avg_result DESC
```

| | full_name | avg_result |
|---|---|---|
| 1 | Волокита М. А. | 200.000000 |
| 2 | Ліщук А. О. | 200.000000 |
| 3 | Негря А. В. | 200.000000 |
| 4 | Парцхаладзе Д. Л. | 198.000000 |
| 5 | Радіченко Н. О. | 195.250000 |
| 6 | Бадига Б. В. | 193.250000 |
| 7 | Потаскалов О. В. | 192.000000 |
| 8 | Бойко Б. Д. | 187.666666 |
| 9 | Саварина В. С. | 183.333333 |
| 10 | Поляк П. Ю. | 179.333333 |

Рисунок 5.23 – Результат запиту

16) Запит із використанням JOIN, функції COUNT(), GROUP BY та HAVING для знаходження кількості учасників, які отримали 200 балів з двох і більше предметів

```
SELECT COUNT(*) AS stud_num
FROM (
SELECT [student_id]
FROM [dbo].[Student]
JOIN  [dbo].[ResultInfo] ON [dbo].[ResultInfo].info_card_id = [dbo].[Student].info_card_id
WHERE [dbo].[ResultInfo].result = 200
GROUP BY [student_id]
HAVING COUNT(*) > 1) AS students
```

| | stud_num |
|---|---|
| 1 | 6 |

Рисунок 5.24 – Результат запиту

17) Запит з використанням JOIN, функції COUNT() та GROUP BY для відображення кількості людей, які писали екзамен по предметах та подолали поріг склав/не склав

```
SELECT
[dbo].[Subject].subject_name, COUNT([dbo].[ResultInfo].subject_id) AS stud_num
FROM [dbo].[Subject] JOIN [dbo].[ResultInfo] ON [dbo].[Subject].subject_id =
[dbo].[ResultInfo].subject_id
WHERE [dbo].[ResultInfo].result IS NOT NULL
GROUP BY subject_name
```

| | subject_name | stud_num |
|---|---|---|
| 1 | англійська мова | 7 |
| 2 | біологія | 6 |
| 3 | географія | 7 |
| 4 | іспанська мова | 9 |
| 5 | історія України | 5 |
| 6 | математика | 8 |
| 7 | німецька мова | 4 |
| 8 | українська мова | 9 |
| 9 | українська мова і література | 7 |
| 10 | фізика | 6 |
| 11 | французька мова | 8 |
| 12 | хімія | 9 |

Рисунок 5.25 – Результат запиту

18) Запит з використанням JOIN, функції COUNT() та GROUP BY для відображення кількості людей, які складали ЗНО в основну сесію та отримали більше 150 балів по предметах

```
SELECT
[dbo].[Subject].subject_name, COUNT([dbo].[ResultInfo].subject_id) AS stud_num
FROM [dbo].[Subject] JOIN [dbo].[ResultInfo] ON [dbo].[Subject].subject_id =
[dbo].[ResultInfo].subject_id
WHERE [dbo].[ResultInfo].result >= 150 AND [dbo].[ResultInfo].session_name = 'основна'
GROUP BY subject_name
```

| | subject_name | stud_num |
|---|---|---|
| 1 | англійська мова | 6 |
| 2 | біологія | 4 |
| 3 | географія | 5 |
| 4 | іспанська мова | 5 |
| 5 | історія України | 3 |
| 6 | математика | 5 |
| 7 | німецька мова | 1 |
| 8 | українська мова | 9 |
| 9 | українська мова і література | 2 |
| 10 | фізика | 6 |
| 11 | французька мова | 6 |
| 12 | хімія | 8 |

Рисунок 5.26 – Результат запиту

19) Запит з використанням JOIN, функції COUNT(), ROW_NUMBER(), GROUP BY та ORDER BY для відображення рейтингу популярності серед предметів

```
SELECT
ROW_NUMBER() OVER(ORDER BY COUNT([dbo].[ResultInfo].subject_id) DESC) subject_num,
[dbo].[Subject].subject_name
FROM [dbo].[Subject] JOIN [dbo].[ResultInfo] ON [dbo].[Subject].subject_id =
[dbo].[ResultInfo].subject_id
GROUP BY subject_name
```

| | subject_num | subject_name |
|---|---|---|
| 1 | 1 | українська мова |
| 2 | 2 | французька мова |
| 3 | 3 | хімія |
| 4 | 4 | іспанська мова |
| 5 | 5 | математика |
| 6 | 6 | математика (завдання рівня стандарту) |
| 7 | 7 | українська мова і література |
| 8 | 8 | фізика |
| 9 | 9 | англійська мова |
| 10 | 10 | географія |
| 11 | 11 | біологія |
| 12 | 12 | історія України |
| 13 | 13 | німецька мова |

Рисунок 5.27 – Результат запиту

20) Запит з використанням JOIN, функції MIN() та GROUP BY для відображення мінімального балу з предметів (не враховуючи тих, хто не подолав поріг склав/не склав)

```
SELECT
[dbo].[Subject].subject_name, MIN([dbo].[ResultInfo].result) AS average_score
FROM
[dbo].[Subject] JOIN [dbo].[ResultInfo] ON [dbo].[Subject].subject_id =
[dbo].[ResultInfo].subject_id
GROUP BY subject_name
```

| | subject_name | average_score |
|---|---|---|
| 1 | англійська мова | 146.0 |
| 2 | біологія | 107.0 |
| 3 | географія | 106.0 |
| 4 | іспанська мова | 116.0 |
| 5 | історія України | 143.0 |
| 6 | математика | 129.0 |
| 7 | математика (завдання рівня стандарту) | NULL |
| 8 | німецька мова | 126.0 |
| 9 | українська мова | 153.0 |
| 10 | українська мова і література | 101.0 |
| 11 | фізика | 191.0 |
| 12 | французька мова | 118.0 |
| 13 | хімія | 152.0 |

Рисунок 5.28 – Результат запиту

5.2.6. Індекси та їхній приклад роботи

Для пришвидшення пошуку в таблицях з великою кількістю даних використовують індекси.

Створимо індекс на колонку subject_id для таблиці ScoreScale:

```
CREATE INDEX idx_subject_id
	ON [dbo].[ScoreScale] ([subject_id])
```

Перевіримо роботу індексу на простому запиті SELECT * FROM [dbo].[ScoreScale];

Час роботи запиту без використання індексу: 31 мс

Час роботи запиту з використанням індексу: 24 мс

Як бачимо використання індексу пришвидшило роботу запиту.

# ВИСНОВКИ

В ході даної лабораторної роботи було створено базу даних з результатами з результатами зовнішнього незалежного оцінювання.

Першим кроком був аналіз предметного середовища. На основі якого були визначені сутності, їхні атрибути та зв’язки між об’єктами.

На наступному етапі бела побудована ER-діаграма заданого предметного середовища. На її основі створено реляційну схему бази даних. Виділено первинні та зовнішні ключі, додано обмеження для підтримки цілісності бази даних.

Написано SQL скрипти для побудови спроєктованої бази даних. А саме: створення таблиць, обмежень, користувачів; заповнення таблиць відповідними даними; створення збережених процедур, функцій, представлень, тригерів та індексів.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. https://testportal.gov.ua/zno-dpa-2/
2. https://www.w3schools.com/sql/default.asp
3. https://learn.microsoft.com/en-us/sql/t-sql/language-reference?view=sql-server-ver16
4. https://learn.microsoft.com/en-us/sql/sql-server/tutorials-for-sql-server-2016?view=sql-server-ver16