**Лабораторна робота 4.**

**Прості запити на вибірку даних.**

**Мета роботи:** закріпити та вдосконалити навички створення запитів на вибірку даних із таблиць БД.

**Задачі:**

* Розгортання учбової БД
* Основні речення запиту до однієї таблиці.
* Об’єднання таблиць.
* Виконання індивідуального завдання згідно варіанту

1. Підключаємось до сервера баз даних SQL Server відкриваємо БД «StudyBase». Відкриємо діаграму БД та переглянемо таблиці БД та іх зв’язки (рис.1)

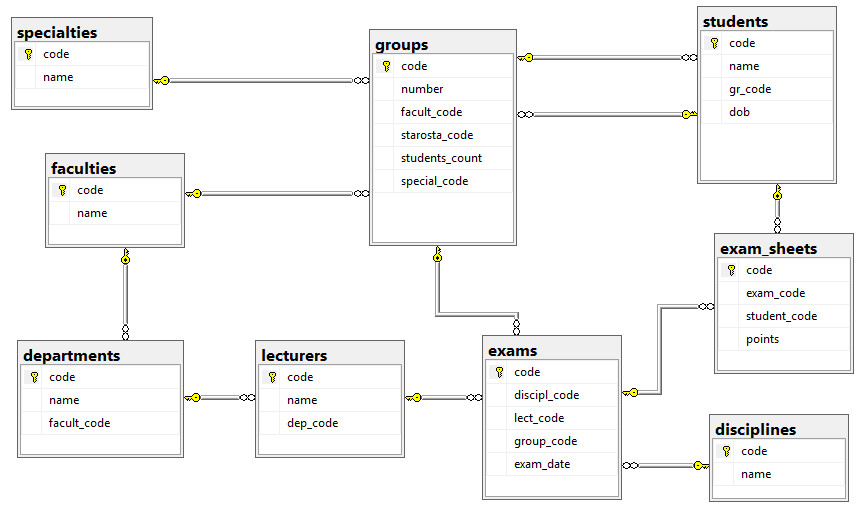


Рис.1. Структура БД «StudyBase».

1. Наведемо стислий опис структури нашої БД. База даних охоплює облік іспитово-залікових відомостей відомостей. Облік ведеться у розрізі викладачів, дисциплін та студентів із прив’язкою до кафедр, факультетів, студентських груп та спеціальностей.

Окрім таблиць, створених у попередній роботі (спеціальності, групи та факультети) додано іще 6 таблиць. Розглянемо кожну із них окремо.

Кафедри (departments) – перелік кафедр, до яких належать викладачі, що приймають заліки-іспити. Поля: code: сурогатний первинний ключ (автоінкриментне), name – назва кафедри, facult\_code – код факультету, до якого належить кафедра (зовнішній ключ).

Викладачі (lecturers) – список викладачів, що приймають заліки-іспити. Поля: code: сурогатний первинний ключ (автоінкриментне), name – Прізвище, Ім’я, По батькові викладача, dep\_code – код кафедри, до якої належить викладач (зовнішній ключ).

Студенти (students) – список студентів, що складають заліки-іспити. Поля: code: сурогатний первинний ключ (автоінкриментне), name – Прізвище, Ім’я, По батькові студента, gr\_code – код групи, до якої належить студент (зовнішній ключ), dob – дата народження.

Дисципліни (disciplines) – список дисциплін, що складаються студентами. Поля: code: сурогатний первинний ключ (автоінкриментне), name – назва дисципліни.

Відомості (exams) – список іспитових відомостей. Поля: code: сурогатний первинний ключ (автоінкриментне), disciple\_code – код дисципліни, що складається (зовнішній ключ), lect\_code – код викладача, що приймає іспит (зовнішній ключ), group\_code – код студентської групи, що складає іспит (зовнішній ключ), exam\_date – дата складання іспиту.

Вміст відомостей (exam\_sheets) – список студентів із балами, що складали іспит. Поля: code: сурогатний первинний ключ (автоінкриментне), exam\_code – код іспиту, що складається (зовнішній ключ), student\_code – код студенту, що складає іспит (зовнішній ключ), points – кількість балів, отримана на іспиті.

1. Тепер, розібравши структуру БД та окремих її таблиць, приступимо до написання запитів. Почнемо із із найпростішого – виберемо всі поля та рядки із таблиці студентів (рис.1):

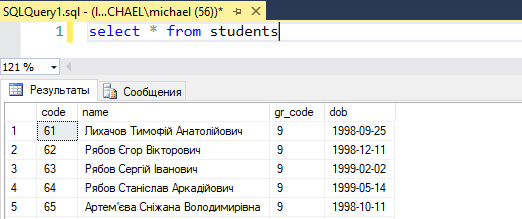


Рис. 1. Вибірка всіх даних таблиці студентів

Виконаємо фільтрацію полів таблиці, виберемо прізвище та дату народження (рис.2)

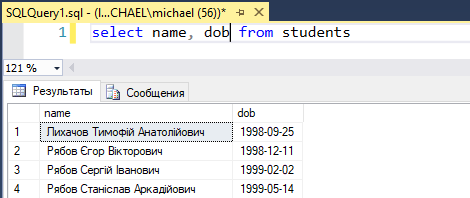


Рис.2. Вибірка окремих полів таблиці студентів

Слід зауважити, що конструкцію «select \* » не рекомендовано використовувати у робочих проектах, а тільки на етапах відлагодження запиту. Тому навіть у випадку вибірки всіх полів, запит (рис.1) краже переписати в наступному вигляді (рис.3)

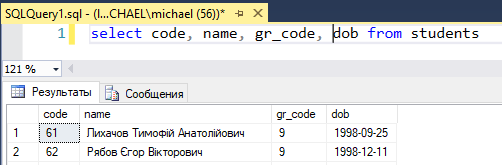


Рис.3. Альтернативна форма запиту рис.1.

Також порядок слідування полів у реченні select може не співпадати із порядком полів у таблиці (рис.4)

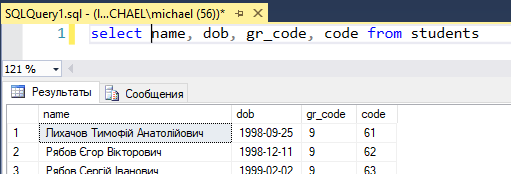


Рис.4. Зміна порядку полів у запиті.

1. Виконаємо фільтрацію рядків таблиці. Виберемо студентів групи із кодом 10 (рис.5)

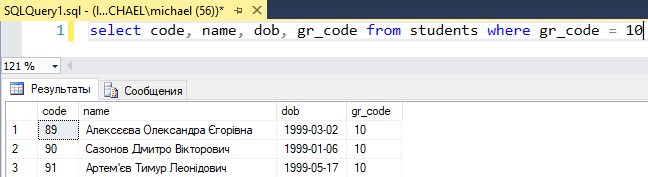


Рис.5. Студенти групи із кодом 10

Також виконаємо фільтрацію на рівність із типом даних дата та час. Виберемо студентів із датою народження 26 листопада 1998 року (рис.6). При роботі із рядками значення також має бути взято в одинарні лапки.

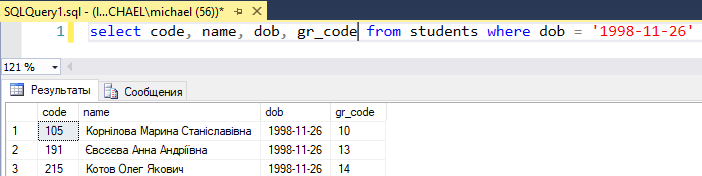


Рис.6. Фільтрація на рівність із типом «дата»

При фільтрації також можуть бути використані інші оператори порівняння, такі як >, <, >=, <=, <> (або !=). Наприклад, виберемо студентів із довжиною прізвища, імені та по батокові більше 30 символів (рис.7.)

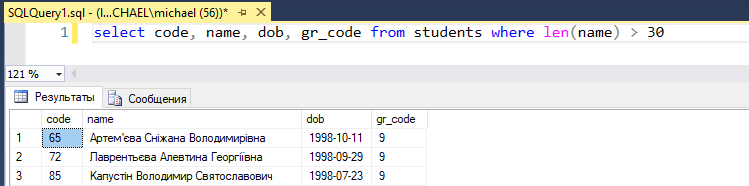


Рис.7. Використання інших операторів порівняння, окрім =.

У реченні where оператори можуть об’єднуватись за допомогою булевих операторів «and», «or» та «not». Виберемо студентів із довжиною прізвища, імені та по батькові більше 30 символів, що народились не пізніше 01.01.1998 року (рис.8).

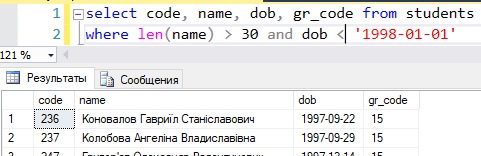


Рис.8. Об’єднання умов. Оператор and

Виберемо студентів груп із кодами 10 та 11. Слід звернути увагу, ща в даному випадку, незважаючи та слово «та» використання оператора «and» було б недоречно, та такий запит не поверне жодного рядку даних (рис.9), оскільки поле gr\_code окремого рядку таблиці «студенти» не може одночасно дорівнювати 10 та 11.

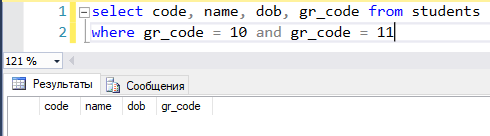


Рис.9. Приклад невірної вибірки студентів 2-х груп

Вірним у даному випадку буде використати оператор «or», тобто «вибрати студентів із кодом групи 10 або 11» (рис.10).

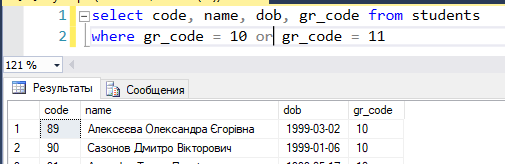


Рис.10. Правильна вибірка студентів 2-х груп

1. Далі розглянемо використання у реченні where операторів in, between та like. in дозволяє виконати перевірку на входження значення в якусь підмножину. Реалізуємо попередній приклад (рис.10) із використанням in (рис.11).

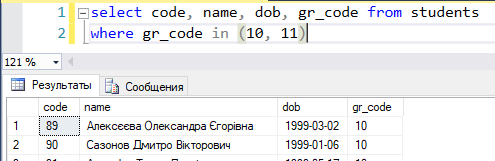


Рис.11. Оператор in

Оператор between дозволяє виконати перевірку на входження у діапазон значень (включно із границями діапазону). Виберемо студентів із датою народження у січні 1998 року (рис.12)

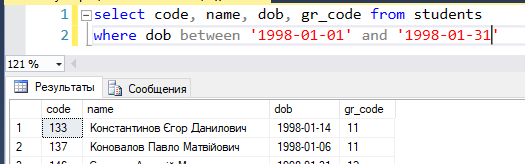


Рис.12. Приклад роботи оператора between

Оператор like дозволяє порівняти рядок на відповідність шаблону. 2 найбільш вживані мета символи: «%» - будь яка кількість символів та «\_» - будь-який символ. Виберемо студенті, прізвище яких починається із літери «К» та має третю літеру «а» (рис.13)

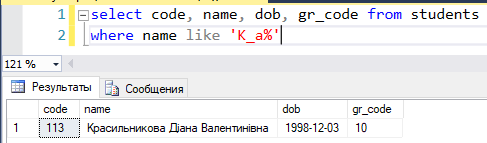


Рис.13. Приклад використання оператору like

1. Наступний розділ – групування та функції агрегації даних. Пригадаємо основні функції агрегації – це min, max, avg, sum та count. Далі розглянемо вибірку мінімальної та максимальної дати народження із списку студентів (рис.14)

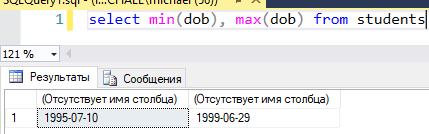


Рис.14. Вибірка мінімального та максимального значення

Також розглянемо приклад отримання кількості студентів (рис.15). Хочеться звернути увагу, що count повертає кількість рядків, для яких передане у якості аргумента поле має значення (не null). Тому для отримання загальної кількості рядків у якості аргумента бажано передавати первиний ключ.

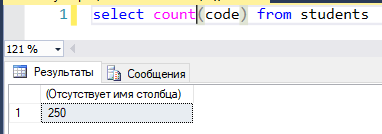


Рис.15. Отримання кількості рядків таблиці «студенти»

Функція «count» також може бути використана разом із інструкцією «distinct», що дозволяє підрахувати кількість різних значень у полі таблички. Наприклад, можна визначити кількість студентських груп (рис.15-а)

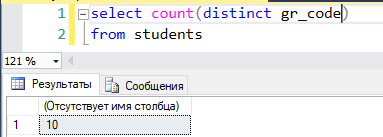


Рис.15-а. Кількість студентських груп

У наступних прикладах почнемо використовувати конструкцію «group by». Визначимо, у скількох студентів дата народження припадає на кожен місяць року (рис.16)

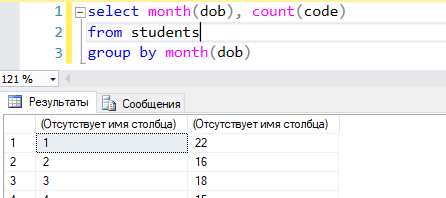


Рис. 16. Використання group by

При використанні «group by» існує обмеження. Щоб звернутись до поля напряму у реченні «select», воно обов’язково має входити у перелік полів, за якими групуються дані. Інакше до нього обов’язково має бути застосована функція агрегації. Проілюструємо це – додамо до попереднього прикладу номер групи (рис.17)

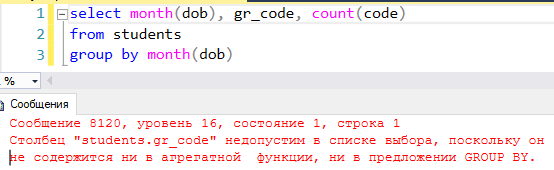


Рис.17. Помилковий запит із використанням group by

Виправити помилку можна, додавши «gr\_code» до «group by» (рис.18) або застосувавши до нього функцію агрегації (рис.19)

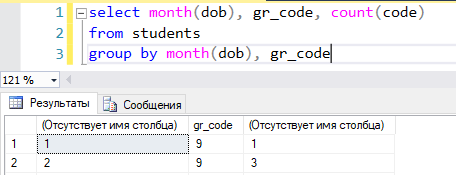


Рис.18. Виправлення помилки рис.17. Шлях 1.

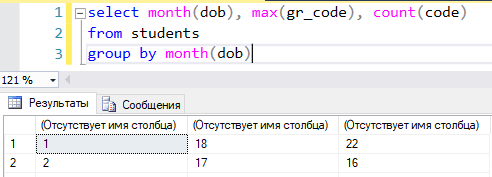


Рис.19. Виправлення помилки рис.17. Шлях 2.

1. Окремої уваги заслуговує фільтрування даних при виконанні групування за допомогою інструкцій «where» та «having». В деяких випадках можна скористатись тільки першим, в деяких обома, а іноді і тим і іншим. Щоб зрозуміти закономірність, слід для початку зазначити що інструкція «where» виконується до «group by», а «having» - після. Розглянемо декілька прикладів. Виберемо скільки у кожній групі навчається студентів, що старших 21 року (рис.20)

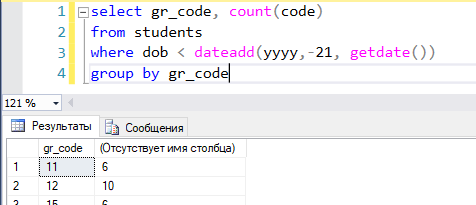


Рис.20. Використання where та group by

Даний підхід є вірним, оскільки поле dob існує у таблиці «студенти», але його немає після виконання групування даних. Переконаємося у цьому, перенісши умову до розділу having (рис.21)

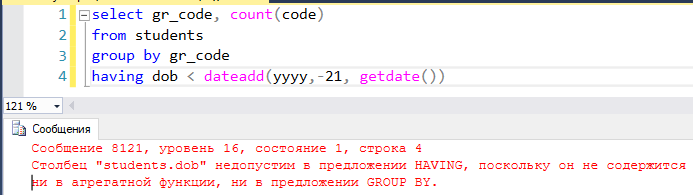


Рис.21. Помилкове використання having та group by

Розглянемо протилежну ситуацію. Виберемо групи, у яких навчається більше 27 студентів. Спробуємо зробити це через «where» (рис.22)

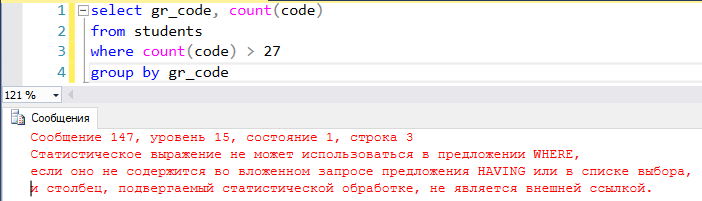


Рис.22. Помилкове використання where та group by

Дана помилка виникає унаслідок того, що на етапі фільтру «where» інструкція «group by» ще не відпрацювала і count(code) ще не обчислено. Але даний приклад може бути цілком успішно реалізовано із використанням інструкції «having» (рис.23)

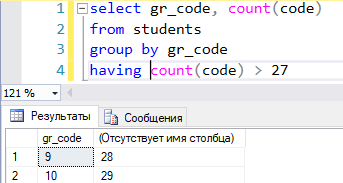


Рис.23. Використання having та group by

1. Виконаємо упорядкування виводу за допомогою речення order by. Виведемо студентів, упорядкованих за прізвищем за зростанням (рис.24)

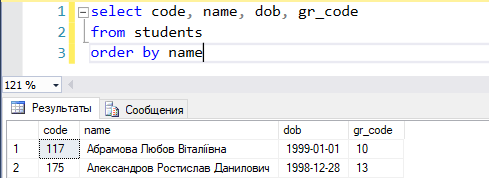


Рис.24. Список студентів, упорядкованих за прізвищем

Для упорядкування за спаданням треба використати опцію desc (рис.25)

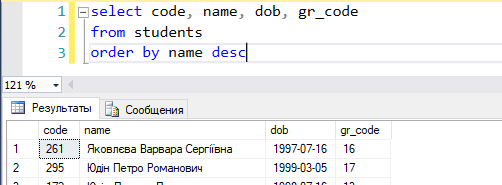


Рис.25. Впорядкування за спаданням

Також упорядкування може бути виконано за двома полями, із використанням різного порядку, наприклад впорядкуємо за кодом групи за спаданням, а у межах однакової групи – за прізвищем за зростанням (рис.26).

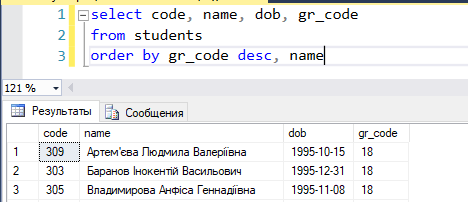


Рис.26. Впорядкування за декількома полями

Також впорядкування може бути виконане за порядковим номером колонки. Наприклад, виведемо кількість студентів по групах, від найбільшої до найменшої (рис.27)

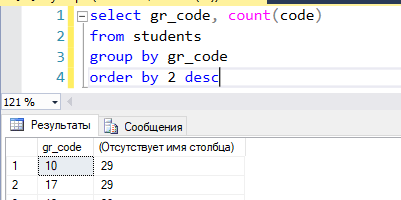


Рис.27. Впорядкування за номером колонки

1. Наведемо також специфічні для T-SQL конструкції «top» та «offset-fetch». За допомогою «top» можна обмежити кількість рядків, що повертається запитом. Так, наприклад, виберемо 3 групи, до складу яких входить найбільше студентів (рис28)

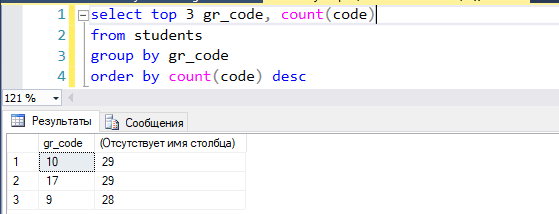


Рис.28. Виведення перших 3-х рядків результату запиту

У деяких випадках наступний рядок за останнім, що потрапив у вибірку буде мати ідентичне значення поля, за яким було виконано сортування, але незважаючи на це не потрапляє до результату. Якщо дану ситуацію треба виправити, слід вказати додаткову конструкцію «with ties» (рис.29)

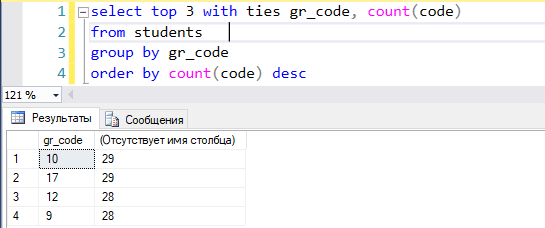


Рис.29. Вибірка екстра-даних, що мають значення, ідентичне останньому рядку.

Також конструкція «top» може бути використана разом із ключовим словом «percent» для вказання кількості рядків у відсотках від загальної. Виберемо перші 10% рядків для попереднього прикладу (рис.30)

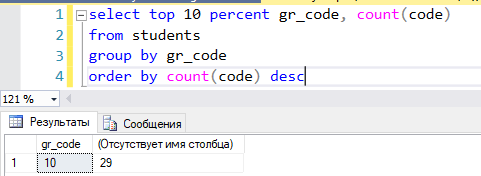


Рис.30. Вибірка перших 10% рядків

Основний недолік фільтра «top» у тому, що він не підтримує пропуск окремих результатів. В T-SQL offset-fetch є частиною речення «order by», де результат може бути додатково відфільтровано. Так, наприклад, для виводу четвертої та п’ятої за кількістю студентів груп, маємо виконати наступний запит (рис.31)

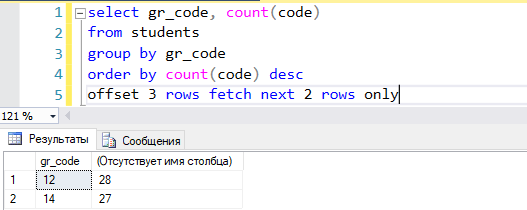


Рис.31. Фільтрація даних за допомогою offset-fetch

1. Розглянемо звернення в одному запиті більш, ніж до однієї таблиці. Почнемо із 2-х таблиць із використанням конструкції join. У якості прикладу виберемо список номерів студентських груп із вказанням назви спеціальності групи (рис.32)

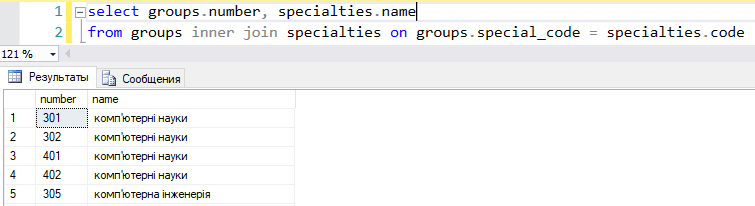


Рис.32. Групи та назви спеціальностей

При поєднанні у запиті 2-х та більше таблиць за допомогою join важливим є правильне вказання типу зв’язку. Тип inner забезпечує вибірку даних, що є як в правій, так і в лівій таблицях. При вказанні типу left або right є можливість виведення всіх записів з лівої або правої таблиці, незалежно від того, чи є вони в іншій таблиці. Наприклад, у попередньому прикладі для виведення всіх спеціальностей, незалежно від того, чи є чи немає на них студентські групи, виконаємо наступний запит із типом з’єднання «right» (рис.33)

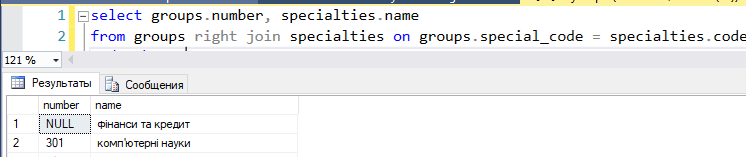


Рис.33. Виведення всіх спеціальностей та їх груп

Також можемо виконати додаткову фільтрацію для отримання спеціальносте, на яких немає жодної групи (рис.34)

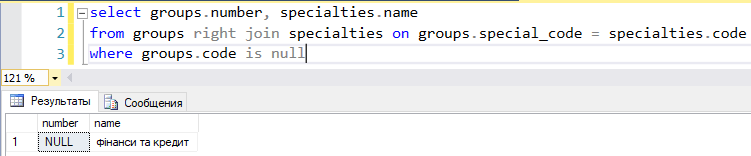


Рис.34. Спеціальності, на яких немає груп

При збільшенні кількості таблиць принцип їх поєднання не змінюється. Так, наприклад, ми хочемо додати до прикладу, наведеного на рис.32, назву факультету (рис.35)

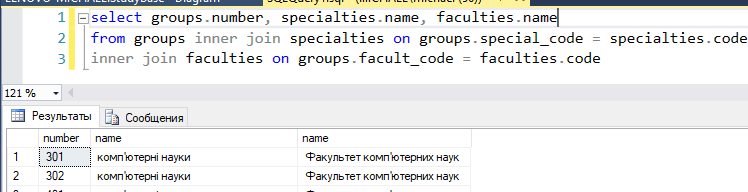


Рис.35. Список груп із назвами факультету та спеціальності

1. Кінець роботи

**Завдання для індивідуального виконання.**

1. Виконати обов’язкові запити до власної БД згідно варіанту
2. Доповнити власну БД мінімум трьома таблицями із первинними та зовнішніми ключами (перед додаванням структуру бажано узгодити із викладачем)
3. Виконати додаткові запити до власної БД (придумати самостійно), використовуючи наступні засоби (при здачі має бути наведено текст запиту, результат та словесний опис, які саме дані вибирає запит):

* Конструкція «offset-fetch»
* Поєднання більш, ніж 2-х таблиць

! УВАГА. Запит не обов’язково має повертати дані. Оцінюється правильність тексту запиту, а не наповненість БД даними (хоча мінімальні вимоги щодо наповненості БД із попередньої роботи все ж мають бути виконані)

**Варіанти для виконання індивідуальних завдань.**

1. БД Автостоянка. Запити:

* Загальна кількість пасажирських місць по автостоянках (назва автостоянки; сумарна кількість пасажирських місць)
* Кількість вільних місць для транспорту по автостоянках (назва автостоянки; кількість вільних місць)
* Список власників транспортних засобів, що мають більше 10 пасажирських місць (Прізвища без повторень)

1. БД Облік кадрів. Запити:

* Середній досвід роботи співробітників по відділах (назва відділу, середній досвід)
* Перелік посад із кількістю співробітників, що їх займають у відділах, де немає вакансій (назва посади, кількість співробітників)
* Відділи та їх співробітники, де кількість вакансій перевищує 10 (назва відділу та прізвище співробітника).

1. БД Облік книжок. Запити:

* Книги, видані у місті Києві (назва книги)
* Загальна кількість книжок за типами книг, що мають від 100 до 200 сторінок (тип книги, кількість книг)
* Кількість типів книг по видавництвах (Назва видавництва, кількість типів книг)

1. БД Лікарські рецепти. Запити:

* Лікарі, що виписували рецепти більше року назад (прізвища без повторень)
* Лікарі, що виписали більше, ніж 100 рецептів за останній місяць (прізвище та кількість рецептів за останній місяць)
* Список лікарських засобів із кількістю рецептів із строком дії більше 100 днів (назва засобу та кількість рецептів)

1. БД Нерухомість. Запити:

* Власники-фізичні особи, у володінні яких є більше однієї квартири в одному домі (прізвища без повторень)
* Загальна кількість квартир у володінні кожного власника, що знаходяться у 5 та 9 поверхових домах (прізвище власнику та кількість квартир)
* Максимальні номери квартир у будинках (адреса будинку та максимальний номер квартири)

1. БД Облік комунального транспорту. Запити:

* Тип транспорту із найбільшою кількістю транспортних засобів (назва типу – один рядок)
* Маршрути із кількістю транспортних засобів, кількість пасажирських місць у яких від 10 до 20 (номер маршруту та кількість транспортних засобів)
* Список типів транспорту із їх місткістю на маршрутах (назва типу, номер маршруту, кількість пас.місць без повторень)

1. БД Кінопрокат. Запити:

* Список сьогоднішніх сеансів у кінотеатрі «Мультилекс Мик.» із роком виходу кінофільму 2018 (назва фільму та дата і час сеансу)
* Список кінотеатрів із кількістю «бюджетних» (вартість квитку < 50 грн) сеансів за останній місяць (назва кінотеатру та кількість сеансів)
* Кількість кінофільмів (не сеансів), що транслювалась у першій половині дня – 12:00 (кількість – один рядок)

1. БД Ремонт телефонів. Запити:

* Модель із максимальною кількістю пристрої на обліку (назва моделі)
* Кількість моделей по брендах, що мають діагональ від 5 до 6 (назва бренду та кількість)
* Бренди, що не мають пристроїв із діагоналлю менше 7 (назви без повторень)

1. БД Облік тварин ветеринарної клініки. Запити:

* Список власників кішок-довгожителів (>20 років) (Прізвища без повторень)
* Кількість тварин на обліку із віком від 3 до 5 за видами тварин (назва виду та кількість)
* Власники, що мають більше однієї тварини (прізвища без повторень)

1. БД Розклад занять. Запити:

* Студентські групи із кількістю студентів <15 та кількістю занять на тиждень < 10 (номери груп без повторень)
* Дисципліни, що викладаються лише по понеділках, або в інші дні на 1 або 2 парах (назви без повторень)
* Скільки разів кожного дня тижня викладається «Основи програмування» у групах із кількістю студентів > 28 (день тижня та кількість разів)

1. БД Облік приміщень та обладнання університету. Запити:

* загальна кількість приміщень на поверсі у корпусах із кількістю поверхів більше 5 (номер поверху та кількість приміщень)
* Назва корпусу, в якому розташоване приміщення із максимальною площею (назва)
* Кількість приміщень за типом приміщення у корпусах із площею від 50 до 100 (назва типу та кількість)

1. БД Туристичної компанії. Запити:

* Перелік довгих туристичних турів (> 20 днів), що замовлялися не менше 20 разів (назви без повторень)
* Клієнти, що замовили турів із сумарною кількістю днів > 100 (Прізвища без повторень)
* Замовлення Іванова П.С. за останні пів-року на одноденні тури (назва туру та дата замовлення)

1. БД Регістратура поліклініки. Запити:

* Перелік пацієнтів, що відвідували «не своїх» (з іншого участку) лікарів (Прізвища пацієнтів без повторень)
* Кількість пацієнтів по лікарях (прізвище лікаря та кількість пацієнтів)
* Пацієнти, що відвідали лікаря більше 10 разів за останній рік (прізвища та кількість відвідувань)

1. БД Оренда транспортних засобів. Запити:

* Транспортні засоби, що мають побіг до 50 000 та не були в оренді (прізвища без повторень)
* Орендарі, що мали або мають в оренді більше 10 транспортних засобів (прізвища із кількістю транспортних засобів)
* Кількість транспортних засобів, що на даних момент орендуються (врахувати поля «Дата» та «Строк оренди днів») юридичними особами (кількість)

1. БД Оренди приміщень. Запити:

* Кількість приміщень за орендарями із площею > 1000 та строком оренди > 200 (Прізвище орендаря та кількість)
* Орендарі, що не орендували нічого вже 3 роки (Прізвища без повторень)
* Приміщення, що орендувалися за останні 3 місяці більше 5 разів (Адреси без повторень)