**Лабораторна робота №8** Підзапити та використання у SQL-запитах на вибірку даних

**Мета:** Набуття практичних навичок формування SQL-підзапитів до БД

Контрольні запитання

1. Охарактеризуйте переваги мови SQL.
2. Коротко охарактеризуйте структуру мови SQL.
3. Наведіть приклади використання операторів мови DDL.
4. Опишіть структуру оператора SELECT. Наведіть приклад простого запиту.
5. Які види умов пошуку ви знаєте? Наведіть приклади використання конструкції WHERE.
6. Які особливості використання конструкції ORDER BY?
7. Які вбудовані функції містить мова  SQL? Наведіть приклади.
8. Наведіть приклад використання конструкції GROUP BY.
9. Що таке підзапит? Які види підзапитів в знаєте?

**Хід роботи**

**Постановка задачі**. Продовжуємо працювати з постановкою задачі з минулих лабораторних та практичних робіт по створенню БД гіпотетичного навчального закладу.

**Завдання до лабораторної роботи**

*При вивченні теми слід звернути увагу на створення та збереження структури бази даних(БД), редагування структури, визначення типів полів, зв’язування таблиць у БД.*

1. Повторити лекції №7, 8, теоретичну частину ЛР № 4,5,6,7 та ПР№6. Ознайомитися з теоретичною частиною цієї ЛР.
2. За результатами виконання ЛР № 4,5,6 та ПР№6 повинна бути сформована та заповнена реляційна БД з таблицями Викладач, Студент, Групи, Предмети, Розклад, Аудиторії, Оцінки. Для цих таблиць виконати завдання:
3. Створити запит для демонстрації використання запиту з підзапитом, що повертає єдине значення.
4. Продемонструвати роботу запиту, який виконує вибірку даних з результату виконання підзапиту (вкладеного підзапиту).
5. Продемонструвати роботу вкладених SQL-підзапитів
6. Продемонструвати роботу пов’язаного підзапиту.
7. Продемонструвати виконання підзапитів з ALL та SOME.
8. Продемонструвати роботу використання підзапитів з вбудованими функціями.
9. Продемонструвати роботу HAVING з використанням підзапиту.

Таблиці та функції обрати самостійно.

1. Результати оформити звітом, в якому надаються сформовані SQL-запити, опис дій, що вони виконуть та скріншот виконання на вашій БД.
2. Оформити результати відповідно до стандарту подання лабораторних/ практичних робіт.
3. Результати надсилати на електронну адресу викладача [t.i.lumpova@gmail.com](mailto:t.i.lumpova@gmail.com)

Файл повинен мати назву в такому форматі:

**DB<Номер групи><Номер лекції / практичної / лабораторної [літера позначення типу роботи L – лекція, P – практична, R – лабораторна]<Прізвище англійською>**. Наприклад, **DB3101R**buts.doc.

Не копіюйте фрагментів з різних інформаційних джерел, подумайте і викладіть свою точку зору. При наявності робіт -"близнюків" відповідь буде зараховуватися першому за часом надсилання.

**Строк виконання цієї роботи ІПЗ-31,32 - 16.11.2023**

**Теоретична частина**

**Підзапити**

Підзапит – це SQL-вираз, що починається з оператора SELECT, який міститься в умові оператора WHERE АБО HAVING для іншого запиту. Таким чином, підзапит – це запит на вибірку даних, вкладений в інший запит. Зовнішній запит, що містить підзапит, якщо тільки він сам не є підзапитом, не обов’язково повинен починатися з оператора SELECT. У свою чергу, підзапит може містити інший підзапит і т. д. При цьому спочатку виконується підзапит, що має найглибший рівень вкладення, і т. д. Часто, але не завжди, зовнішній запит звертається до однієї таблиці, а підзапит – до іншої. На практиці саме цей випадок найбільш цікавий.

Стандарт SQL допускає розміщення в операторі SELECT вкладених запитів. Зовнішній оператор SELECTвикористовує результат виконання внутрішнього оператора для визначення змісту остаточного результату всієї операції. Внутрішні запити можуть знаходитися в конструкціях WHERE і HAVING, а також у списку вибірки зовнішнього оператора SELECT. В цьому випадку вони отримують назву підзапитів, або вкладених запитів. Також, внутрішні оператори SELECT можуть використовуватися в операторах INSERT, UPDATE і DELETE.

**Прості підзапити**. Прості підзапити характеризуються тим, що вони формально ніяк не пов’язані з їх зовнішніми запитами. Це дозволяє спочатку виконати підзапит, результат якого потім використовується для виконання зовнішнього запиту.

Розглядаючи прості підзапити, слід виділити три часткових випадки:

- підзапити, що повертають єдине значення;

- підзапити, що повертають список значень з одного стовпця таблиці;

- підзапити, що повертають набір записів.

Розглянемо ці окремі випадки більш детально.

**Робота з єдиним значенням**. Припустимо, з таблиці продажем BookSales потрібно вибрати дані про книги для яких кількість проданих більше середнього значення. Це можна зробити за допомогою наступного запиту:

SELECT \* FROM BookSales WHERE Units> (SELECT АVG (Units) FROM BookSales);

У даному запиті спочатку виконується підзапит (SELECT АVG (Units) FROM BookSales). Він повертає єдине значення (а не набір записів) – середнє значення стовпця кількість проданих Units. Якщо сказати точніше, то даний підзапит повертає єдиний запис, що містить єдине поле. Далі виконується зовнішній запит, який виводить всі стовпці таблиці клієнти і записи, в яких значення стовпця Units більше значення, отриманого за допомогою підзапиту. Таким чином, спочатку виконується підзапит, а потім зовнішній запит, що використовує результат підзапиту.

Спроба виконати наступний простий запит призведе до помилки:

SELECT \* FROM BookSales WHERE Units> АVG (Units);

Вираз підзапиту обов’язково має бути укладений в круглі дужки.

**Робота зі списком значень з одного стовпця**. Підзапит, взагалі кажучи, може повертати декілька записів. Наприклад, нам потрібно вивести назви книг, які вже продавалися, тобто є відомості про продажі даних книг в таблиці BookSales.

SELECT Title FROM Titles WHERE ISBN IN (SELECT ISBN FROM BookSales);

Щоб в умові зовнішнього оператора WHERE можна було використовувати оператори порівняння, що вимагають одного значення, використовуються квантори, такі як ALL (наприклад …> ALL (підзапит) означає, що наше значення повинне бути більше всіх значень вибраних підзапитом ) і SOME (наприклад …> SOME (підзапит) означає, що наше значення повинне бути більше хоча б одного значення з значень вибраних підзапитом).

Вкладені підзапити можуть містити умови, зумовлені оператором WHERE. Зрозуміло, це один з можливих варіантів запитів, який повертає необхідні дані. Робота з набором записів У повнофункціональних базах даних підзапити можна використовувати не тільки в операторах WHERE і HAVING, але і в операторі FROM:

SELECT T.стовпець1, T.стовпець2, ... , T.стовпець n FROM (SELECT ...) T WHERE ... ;

Тут таблиці, яка повертається підзапитом в операторі FROM, присвоюється псевдонім Т, а зовнішній запит виділяє стовпці цієї таблиці і, можливо, записи відповідно до деякої умови, яка зазначена в операторі WHERE.

**Пов’язані підзапити**

Пов’язані (корельовані) підзапити з практичної точки зору найбільш цікаві. Однак для їх застосування важливо розуміти, що саме буде робити СКБД для надання відповіді. Для початку достатньо усвідомити, що при виконанні запитів, що містять пов’язані підзапити, немає такого чіткого поділу в часі виконання між підзапитом і запитом, як у випадку простих підзапитів. У випадку простих підзапитів спочатку виконується підзапит, а потім запит що його містить. У випадку пов’язаних підзапитів порядок виконання запиту зовсім інший, і його бажано розуміти, щоб уникнути непорозумінь. Основна ознака пов’язаного підзапиту полягає в тому, що він не може бути виконаний самостійно, поза всяким зв’язком з основним запитом. Формально ця ознака виявляється в виразі складного запиту наступним чином: підзапит посилається на таблицю, яка згадується в основному запиті. Отже, підзапит повинен бути виконаний в якомусь контексті з поточним станом виконання основного запиту.

Розглянемо деякий абстрактний і, в той же час, типовий запит, який містить пов’язаний підзапит:

SELECT A FROM T1

WHERE B=

(SELECT B FROM T2 WHERE C=T1.C)

Даний запит на вибірку даних (оскільки він починається з оператора SELECT) містить підзапит, сформульований в виразі, розміщеному в основному запиті після ключового слова WHERE.

Даний запит використовує дві таблиці: Т1 і Т2, в яких є стовпці з однаковими іменами В і С і однаковими типами. Підзапит, розташований у виразі після ключового слова WHERE основного запиту (SELECT B FROM T2 WHERE C=T1.C), звертається до цих же таблиць. Оскільки одна з таблиць (Т1) зустрічається як в підзапиті, так і в зовнішньому запиті, то підзапит не можна виконати самостійно, без зв’язку із зовнішнім запитом. Тому виконання запиту в цілому відбувається наступним чином:

1. Спочатку виділяється перший запис з таблиці Т1, зазначеної в операторі FROM зовнішнього запиту (весь запис таблиці Т1, а не тільки значення стовпця А). Цей запис називається поточним. Значення стовпців для цього запису доступні і можуть бути використані в підзапиті.

2. Потім виконується підзапит, який повертає список значень стовпця В таблиці Т2 для яких значення стовпця С дорівнює значенню стовпця С таблиці Т1.

3. Запит, сформульований в розглянутому прикладі, передбачає, що його підзапит повертає єдине значення (список з єдиним елементом), оскільки в операторі WHERE використовується оператор порівняння (=). Ми будемо вважати, що в цьому прикладі підзапит повертає єдине значення. Тепер виконується оператор WHERE основного запиту. Якщо значення стовпця В поточному (виділеному) записі таблиці Т1 дорівнює значенню, отриманому підзапитом, то цей запис виділяється зовнішнім запитом.

4. Оператор SELECT зовнішнього запиту виконує перевірку умови свого оператора WHERE. А саме він перевіряє, чи рівне поточне значення стовпця в таблиці Т1 значенню, одержаному підзапитом. Якщо так, то значення стовпця А поточного запису таблиці Т1 поміщається в результатну таблицю, в іншому випадку запис ігнорується. Потім відбувається перехід до наступного запису таблиці Т1. Тепер для неї виконується підзапит. Аналогічним чином все описане відбувається для кожного запису таблиці Т1.

**Приклади вкладених запитів (підзапитів)**

Розглянемо три типи підзапитів:

1. *Скалярний підзапит* повертає значення, що отримується з перетину одного стовпця з одним рядком, тобто єдине значення. Варіанти використання скалярних підзапитів наведені в прикладі:

*Визначити, у скількох квартир, що здаються в оренду, орендна плата перевищує 4000 грн*.

SELECT COUNT(\*) AS count

   FROM FlatForRent

   WHERE rent > 4000;

***Використання скалярного підзапиту в конструкції WHERE*.**

*Скласти список співробітників, які працюють у відділенні компанії, яке розташоване по вулиці ‘Хмельницьке шосе, 90’.*

SELECT staffNo, fName, IName, position

   FROM Staff

   WHERE branchNo = (SELECT branchNo

                                  FROM Branch

                                  WHERE street = ‘*Хмельницьке шосе, 90’*);

Внутрішній оператор SELECT призначений для визначення номера відділення компанії, розташованого по вулиці ‘Хмельницьке шосе,90’. Після отримання номера необхідного відділення виконується зовнішній підзапит, призначений для вибірки відомостей про працівників цього відділення.

В результаті зовнішній оператор SELECT набуває вигляду:

SELECT staffNo, fName, IName, position

   FROM Staff

   WHERE branchNo = ‘Vin03’;

Підзапит можна вказувати безпосередньо після операторів порівняння (тобто операторів =, <,>, <=,> =, <>) в конструкції WHERE або HAVING. Текст під запиту повинен бути розміщений в круглих дужках.

2. *Рядковий підзапит* повертає значення декількох стовпців таблиці, але у вигляді єдиного рядка. Варіант рядкового підзапиту наведено в прикладах:

*Визначити загальну кількість директорів компанії та загальну суму їх місячної заробітної плати.*

 SELECT COUNT(staffNo) AS count, SUM(salary) AS sum

     FROM Staff

     WHERE position = 'Директор';

*Обчислити значення максимальної, мінімальної та середньої заробітної плати співробітників компанії.*

SELECT MAX(salary) AS max, MIN(salary) AS min, AVG(salary) AS avg

    FROM Staff;

В даному прикладі необхідно опрацювати відомості про всіх співробітників компанії, тому використовувати конструкцію WHERE не потрібно.

3. *Табличний підзапит* повертає значення одного або декількох стовпців таблиці, що розміщені в більш ніж одному рядку.

**Приклад використання підзапитів з вбудованими функціями.**

*Скласти список всіх співробітників, які мають заробітну плату вище середньої, зазначивши, на скільки їх заробітна плата перевищує середню.*

SELECT staffNo, fName, IName, position, salary - (SELECT AVG(salary)

                                                                       FROM Staff) AS salDiff

    FROM Staff

    WHERE salary > (SELECT AVG(salary)

                               FROM Staff);

Не можна безпосередньо включити в запит вираз WHERE salary > AVG (salary), оскільки застосовувати вбудовані функції в конструкції WHERE заборонено. Для досягнення бажаного результату потрібно створити підзапит, що обчислює середнє значення заробітної плати, а потім використовувати його в зовнішньому операторі SELECT, призначеному для вибірки відомостей про тих працівниках компанії, чия зарплата перевищує це середнє значення.

До підзапитів застосовуються такі *правила і обмеження*.

1. У підзапитах не повинна використовуватися конструкція ORDER BY, хоча вона може бути присутньою в зовнішньому операторі SELECT.

2.  За замовчуванням імена стовпців в підзапиті відносяться до таблиці, ім'я якої зазначено в конструкції FROM підзапиту. Однак дозволяється посилатися і на стовпці таблиці, зазначеної в конструкції FROM зовнішнього запиту, для чого використовуються уточнені імена стовпців (як описано нижче).

3. Якщо підзапит є одним з двох операндів, що беруть участь в операції порівняння, то підзапит повинен вказуватися в правій частині цієї операції.

З підзапитами, які повертають один стовпець чисел, можуть використовуватися ключові слова ANY і ALL. Якщо підзапиту передуватиме ключове слово ALL, умова порівняння вважається виконаною тільки в тому випадку, якщо вона виконується для всіх значень в результуючому стовпці підзапиту. Якщо підзапиту передує ключове слово ANY, то умова порівняння буде вважатися виконаною, якщо вона задовольняється хоча б для будь-якого (одного або декількох) значень в результуючому стовпці підзапиту. Якщо в результаті виконання підзапиту буде отримано порожнє значення, то для ключового слова ALL умова порівняння буде вважатися виконаною, а для ключового слова ANY – невиконаною. Відповідно до стандарту ISO додатково можна використовувати ключове слово SOME, що є синонімом ключового слова ANY.

**Приклад використання ключового слова ANY/ SOME.**

*Знайти всіх співробітників, чия заробітна плата перевищує заробітну плату хоча б одного співробітника відділення компанії з номером ‘Vin03’.*

SELECT staffNo, fName, IName, position, salary

    FROM Staff

    WHERE salary > SOME (SELECT salary

                                       FROM Staff

                                       WHERE branchNo = 'Vin03');

Цей запит міг бути записаний з використанням підзапиту, що визначає мінімальну зарплату персоналу відділення під номером 'Vin03', після чого зовнішній підзапит зможе вибрати відомості про весь персонал компанії, чия зарплата перевищує це значення (див. приклад використання підзапитів з вбудованими функціями).

**Приклад використання ключового слова ALL.**

*Знайти всіх співробітників, чия заробітна плата перевищує заробітну плату будь-якого співробітника відділення компанії з номером ‘Vin03’.*

SELECT staffNo, fName, INarae, position, salary

    FROM Staff

    WHERE salary > ALL(SELECT salary

                                   FROM Staff

                                   WHERE branchNo = 'Vin03');

В даному випадку можна було б використовувати підзапит, який визначає максимальне значення зарплати персоналу відділення під номером 'Vin03', після чого за допомогою зовнішнього запиту вибрати відомості про всіх працівників компанії, зарплата яких перевищує це значення.

Розбираємо запити

select max (grade) FROM grades

***Використання скалярного підзапиту в конструкції WHERE***

SELECT count (\*) from grades G where G.grade < (select max (grade) FROM grades )

SELECT \* from students S, grades G where G.grade = (select max (grade) FROM grades )

and G.CodeStud =S.ID\_stud

SELECT S.\*,G.grade from students S, grades G where G.grade = (select max (grade) FROM grades )

and G.CodeStud =S.ID\_stud

SELECT COUNT(distinct CodeStud) AS count\_stud

FROM grades WHERE Sub\_code < 10;

***Рядковий підзапит***

SELECT COUNT(distinct Grades.CodeStud) AS count\_stud,Subject .Sub\_name

FROM grades, Subject WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code; (???)

SELECT COUNT(distinct Grades.CodeStud) AS count\_stud,Subject .Sub\_name

FROM grades, Subject WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code

GROUP BY Grades.Sub\_code

SELECT COUNT(distinct Grades.CodeStud) AS count\_stud,Subject .Sub\_name

FROM grades, Subject WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code

GROUP BY Grades.CodeStud (??? )

SELECT COUNT(Grades.CodeStud) AS count\_stud,Subject .Sub\_name,Students.Name

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

GROUP BY Grades.CodeStud (???)

SELECT Grades.CodeStud,Subject .Sub\_name,Students.Name

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and Students.ID\_stud=Grades.CodeStud

GROUP BY Grades.CodeStud (check)

SELECT Grades.CodeStud,Subject.Sub\_name,Students.Name ,Grades.grade

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

GROUP BY Grades.CodeStud (???)

SELECT Grades.CodeStud,Subject .Sub\_name,Students.Name ,Grades.grade

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

--GROUP BY Grades.CodeStud

SELECT Grades.CodeStud,Subject .Sub\_name,Students.Name ,avg(Grades.grade)

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

GROUP BY Grades.CodeStud

SELECT Grades.CodeStud, Subject.Sub\_name, Students.Name, count(Grades.CodeStud)

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

GROUP BY Students.ID\_stud --Grades.CodeStud (Check)

SELECT Grades.CodeStud,Subject .Sub\_name,Students.Name ,Grades.grade - avg(Grades.grade)

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

-- GROUP BY Students.ID\_stud --Grades.CodeStud

SELECT Grades.CodeStud,Subject .Sub\_name,Students.Name ,Grades.grade - avg(Grades.grade)

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

-- GROUP BY Students.ID\_stud --Grades.CodeStud

SELECT Grades.CodeStud,Subject .Sub\_name,Students.Name ,round(Grades.grade - avg(Grades.grade),1) delta

FROM grades, Subject,Students

WHERE Grades.Sub\_code < 10 and Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

GROUP BY Students.ID\_stud --Grades.CodeStud

SELECT Subject .Sub\_name,Students .Name ,Grades.grade

FROM Subject,Students,Grades

WHERE Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

and grades.grade < ALL (select grade from Grades where Sub\_code <10)

--GROUP BY Students.ID\_stud --Grades.CodeStud

SELECT Subject .Sub\_name,Students .Name ,Grades.grade

FROM Subject,Students,Grades

WHERE Subject .Sub\_code=Grades.Sub\_code and

Students.ID\_stud=Grades.CodeStud

and grades.grade < ANY (select DISTINCT Grades.grade from Grades where Grades.Sub\_code=1)