**Міністерство освіти і науки України**

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

**Факультет інформатики та обчислювальної техніки**

**Кафедра інформатики та програмної інженерії**

**Звіт**

з лабораторної роботи № 4 з дисципліни

«Бази даних»

„ **Створення запитів на групування, сортування, використання вбудованих функцій. Створення та керування представленнями**”

**Виконав(ла)**

(шифр, прізвище, ім'я, по батькові)

*ІП-15 Мєшков Андрій Ігорович*

**Перевірив**

(прізвище, ім'я, по батькові)

*Ліщук Олександр*

Київ 2023

**Лабораторна робота № 4**

**Створення запитів на групування, сортування, використання вбудованих функцій. Створення та керування представленнями**

**Мета**:

−  Вивчити оператор, котрий використовується в реляційних СУБД, для вибірки даних з таблиць, групування та сортування даних

−  Навчитись використовувати вбудовані функції в запитах

−  Вивчити призначення представлень (view) баз даних, синтаксису та семантики команд SQL для їх створення, зміни та видалення, системних збережених процедур для отримання інформації про представлення

**Теоретичні основи**

Викладені в лекційному матеріалі

**Постановка задачі лабораторної роботи №4**

При виконанні лабораторної роботи необхідно виконати наступні дії:

При виконанні лабораторної роботи необхідно виконати наступні дії:

1) Створити наступні запити:

* 1. запит з використанням функції COUNT;
  2. запит з використанням функції SUM;
  3. запит з використанням функцій UPPER, LOWER;
  4. запит з використанням функцій для роботи з датами;
  5. запит з використанням групування по декільком стовпцям;
  6. запит з використанням умови відбору груп HAVING;
  7. запит з використанням HAVING без GROUP BY;
  8. запит з використанням функцій row\_number() over ....;
  9. запит з використанням сортування по декільком стовпцям.

2) Робота з представленнями (view):

a. створити представлення, котре містить дані з декількох таблиць;  
b. створити представлення, котре містить дані з декількох таблиць та

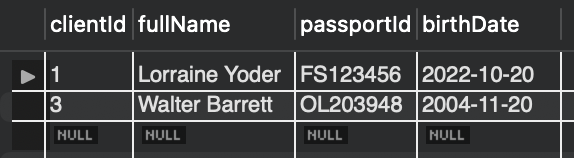
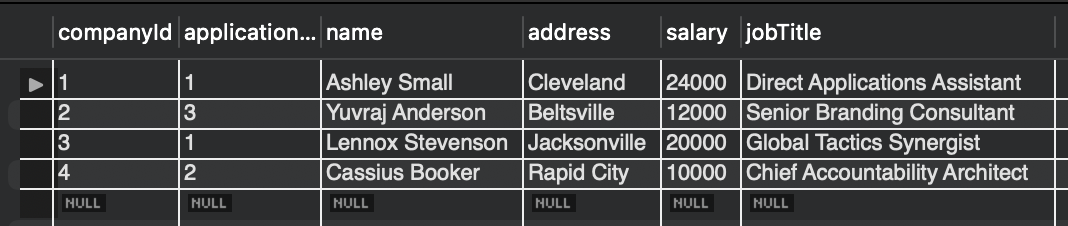
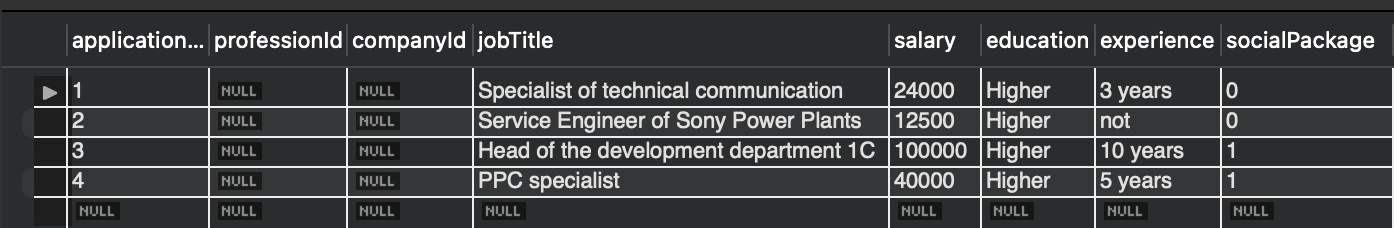
використовує представлення, котре створене в п.a;  
c. модифікувати представлення з використанням команди ALTER VIEW;  
d. отримати довідникову інформацію про ці представлення з використанням вбудованих процедур (наприклад в MsSQL sp\_help, sp\_helptext та sp\_depends).

Варіант 18

**Кадрова агенція**. Кадрова агенція займається підбором персоналу для підприємств і пошуком вакансій для людей. При здійсненні підбору персоналу підприємство звертається в кадрову агенцію і оформляє заявку на пошук працівника. У заявці вказується назва вакансії, професійні вимоги до претендента, пропонований рівень заробітної плати, наявність соціального пакету та інше. Працівники агенції розміщують заяву на сайті агенції, подають оголошення у відповідні друковані видання, вказуючи для контактів свої робочий телефон і e-mail. Претенденти на вакансію звертаються в кадрову агенцію, де проходять початкову співбесіду. Якщо працівник кадрової агенції вважає, що претендент задовольняє виставленим вимогам, то він реєструє його дані (ПІБ, контактний телефон або e-mail) в базі даних і повідомляє їх підприємству. При здійсненні підбору вакансій клієнт заповнює бланк, в якому вказує свої професійні можливості з переліку професійних вимог працедавців, паспортні дані, бажаний рівень заробітної плати. Працівник агенції виконує пошук вакансій по вказаних параметрах, при знаходженні вакансії він повідомляє клієнту контактні дані працедавця. Серед іншого адміністрація кадрової агенції може видавати статистику про середній рівень пропонованої заробітної плати по професіях, по конкретному підприємству і так далі.

**SQL-скрипти**

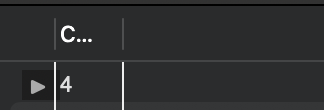
Будуть використані дані з таблиць Company, Application, Client:



1) Створюємо наступні запити:

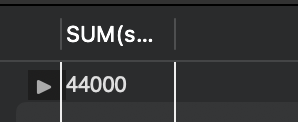
1. запит з використанням функції COUNT

SELECT COUNT(\*) FROM Aplication WHERE education = 'Higher';



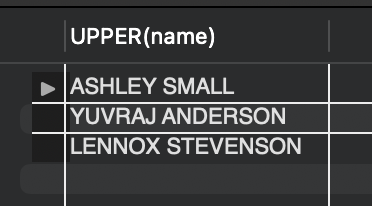
1. запит з використанням функції SUM

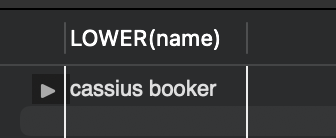
SELECT SUM(salary) FROM Company WHERE applicationId = 1;



1. запит з використанням функцій UPPER, LOWER

SELECT UPPER(name) FROM Company WHERE salary > 11000;



SELECT LOWER(name) FROM Company WHERE salary < 11000; 

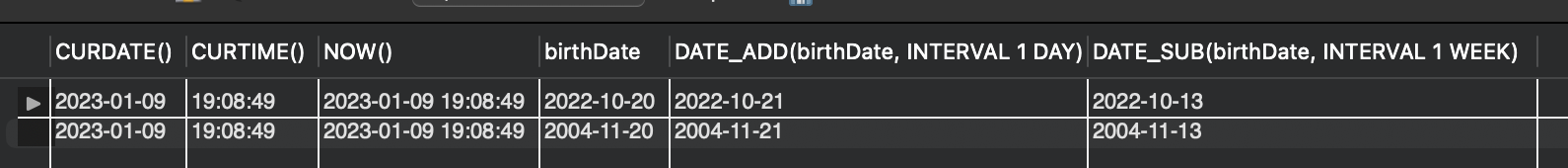
1. запит з використанням функцій для роботи з датами

SELECT CURDATE(), CURTIME(), NOW(),

birthDate,

DATE\_ADD(birthDate, INTERVAL 1 DAY), DATE\_SUB(birthDate, INTERVAL 1 WEEK)

FROM Client;

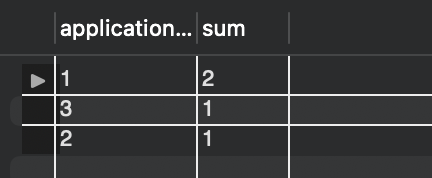


1. запит з використанням групування по декільком стовпцям

SELECT applicationId, COUNT(\*) as sum

FROM Company

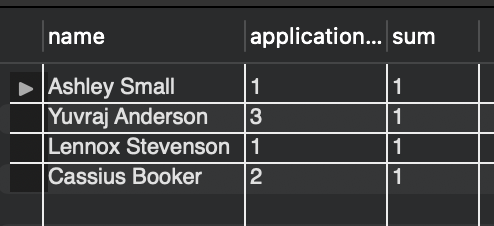
GROUP BY applicationId;



SELECT name, applicationId, COUNT(\*) as sum

FROM Company

GROUP BY name, applicationId;



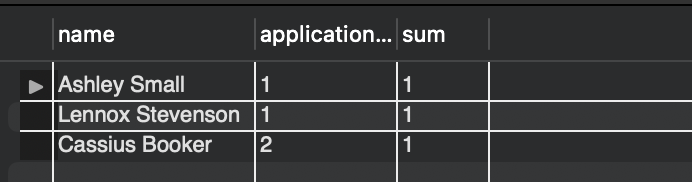
1. запит з використанням умови відбору груп HAVING

SELECT name, applicationId, COUNT(\*) as sum

FROM Company

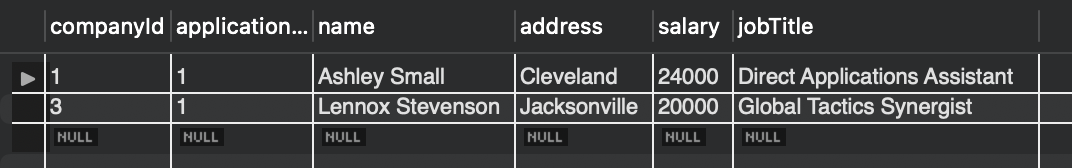
GROUP BY name, applicationId

HAVING applicationId < 3;



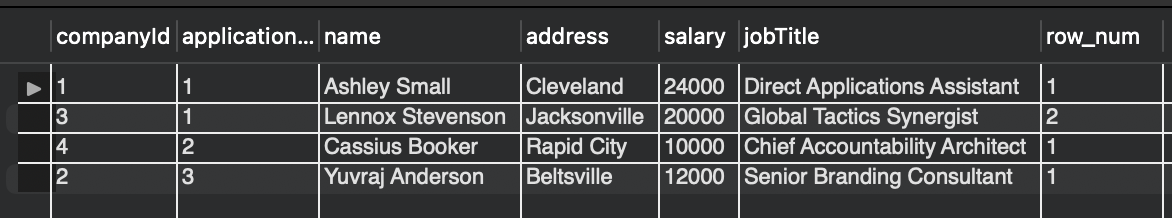
1. запит з використанням HAVING без GROUP BY

SELECT \*FROM Company HAVING salary > 13000;

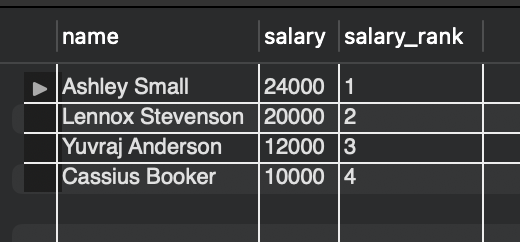


1. запит з використанням функцій row\_number() over ....;

SELECT \*, row\_number() OVER (PARTITION BY applicationId ORDER BY salary DESC) as row\_num FROM Company;



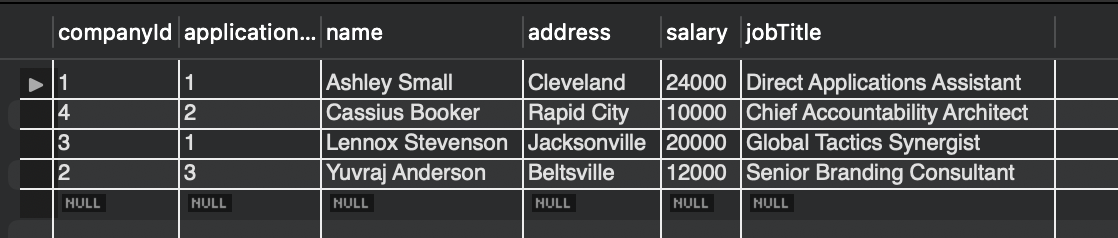
SELECT name, salary, DENSE\_RANK() OVER (ORDER BY salary DESC) as salary\_rank FROM Company;



1. запит з використанням сортування по декільком стовпцям

SELECT \* FROM Company

ORDER BY name, jobTitle;

****

2) Робота з представленнями (view):

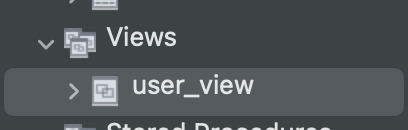
a. створити представлення, котре містить дані з декількох таблиць;

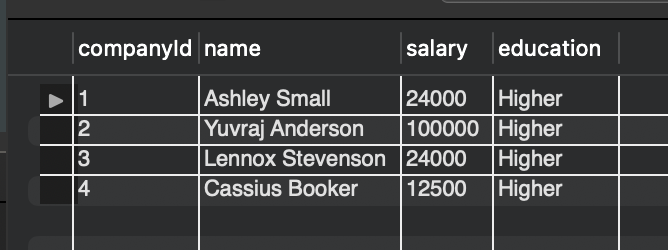
CREATE VIEW user\_view AS

SELECT c.companyId, c.name, a.salary, a.education

FROM Company c

JOIN Aplication a ON c.applicationId = a.applicationId;





b. створити представлення, котре містить дані з декількох таблиць та використовує представлення, котре створене в п.a;

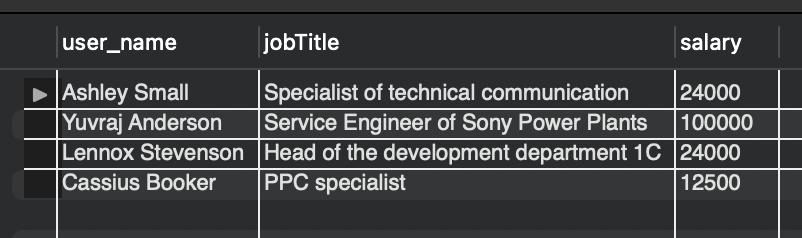
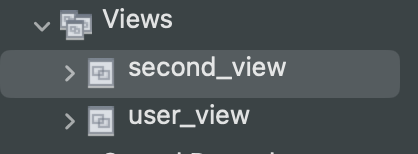
CREATE VIEW second\_view AS

SELECT c.name as user\_name, a.jobTitle, uv.salary

FROM Company c

JOIN Aplication a ON c.companyId = a.applicationId

JOIN user\_view uv ON a.applicationId = uv.companyId;



c. модифікувати представлення з використанням команди ALTER VIEW;

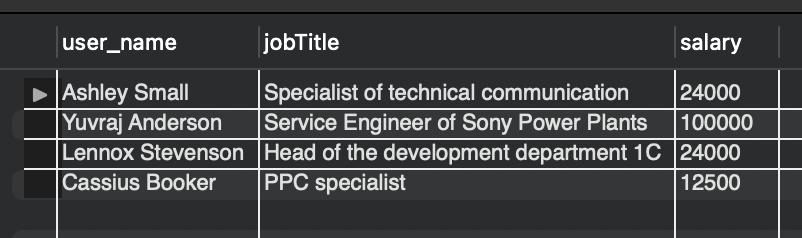
ALTER VIEW second\_view AS

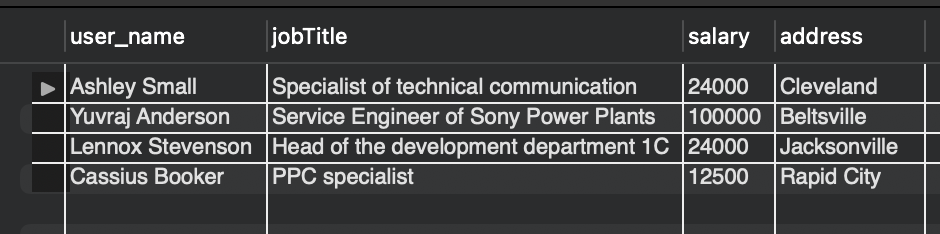
SELECT c.name as user\_name, a.jobTitle, uv.salary, c.address

FROM Company c

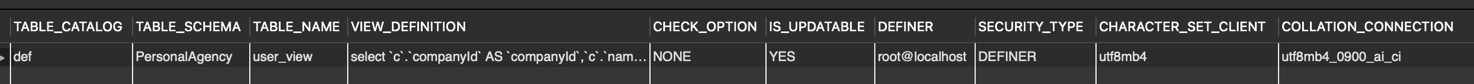
JOIN Aplication a ON c.companyId = a.applicationId

JOIN user\_view uv ON a.applicationId = uv.companyId;

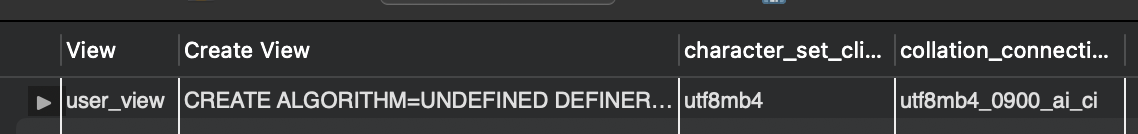


  
d. отримати довідникову інформацію про ці представлення з використанням вбудованих процедур (наприклад в MsSQL sp\_help, sp\_helptext та sp\_depends).

SELECT \* FROM INFORMATION\_SCHEMA.VIEWS WHERE TABLE\_NAME = 'user\_view';



SHOW CREATE VIEW user\_view;

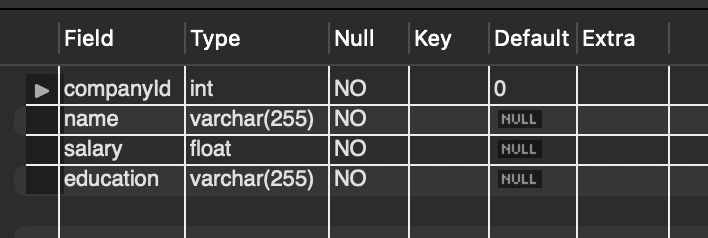


SHOW TABLE STATUS LIKE 'user\_view';

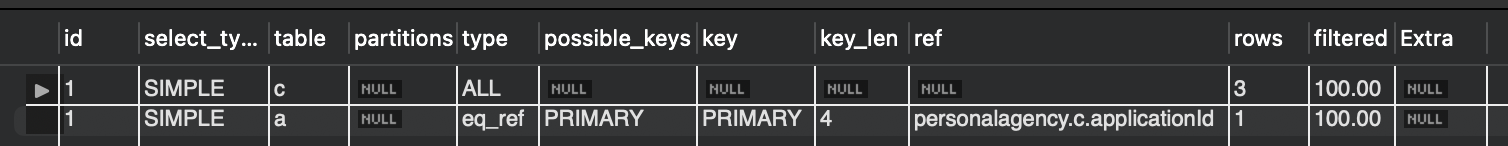


SHOW COLUMNS FROM user\_view;

DESCRIBE user\_view;



EXPLAIN SELECT \* FROM user\_view;

****

**Висновок**: Під час виконання лабораторної работи я використав оператор, котрий використовується в реляційних СУБД, для вибірки даних з таблиць, групування та сортування даних. Навчився використовувати вбудовані функції в запитах. Вивчив призначення представлень (view) баз даних, синтаксису та семантики команд SQL для їх створення, зміни та видалення, системних збережених процедур для отримання інформації про представлення.