# 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.”

Варіант 7

Виконав студент ІП-11 Головня О.Р.

(шифр, прізвище, ім'я, по батькові)

Перевірив(-ла) Ліщук Катерина Ігорівна

(прізвище, ім'я, по батькові)

Київ 2022

**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

**Постановка задачі:**

При виконанні лабораторної роботи необхідно виконати наступні дії:

1) Створити наступні запити:

a. запит з використанням функції COUNT;

b. запит з використанням функції SUM;

c. запит з використанням функцій UPPER, LOWER;

d. запит з використанням функцій для роботи з датами;

e. запит з використанням групування по декільком стовпцям;

f. запит з використанням умови відбору груп HAVING;

g. запит з використанням HAVING без GROUP BY;

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

i. 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).

1. з використанням функції COUNT;

SELECT COUNT(\*) AS driver\_count FROM driver;

(Рахуємо кількість водіїв)

1. з використанням функції SUM;

SELECT SUM(passenger\_num) AS passengers\_sum, COUNT(\*) trolley\_count FROM brand, trolley WHERE trolley.brand\_name = brand.brand\_name;

(Скрипт виведе кількість тролейбусів та кількість пасажирів для цих тролейбусів)

1. з використанням функцій UPPER, LOWER

SELECT LOWER(stop\_name) AS edited, stop\_name AS actual FROM stop;

(Все в нижньому реєстрі)

SELECT UPPER(full\_name) AS first\_name, full\_name FROM driver;

(Все у верхньому реєстрі)

1. з використанням функцій для роботи з датами;

SELECT tech\_inspection\_date FROM trolley WHERE tech\_inspection\_date < now()- INTERVAL 5 year;

(Виведе дату тех перевірки давнішу ніж на 5 років)

1. з використанням групування по декільком стовпцям;

SELECT brand\_name, parking\_id, COUNT(brand\_name) FROM trolley GROUP BY brand\_name, parking\_id ORDER BY COUNT(brand\_name) DESC;

(Беремо тролейбуси, у них можуть повторятись назва марки та на якому вони паркінгу, ми по них групуємо і рахуємо скільки їх, сортуємо від більшого до меншого)

1. з використанням умови відбору груп HAVING;

SELECT route\_id, COUNT(route\_id) AS driver\_count FROM route GROUP BY(route\_id) HAVING COUNT(route\_id) > 0 ORDER BY COUNT(route\_id) DESC;

(Рахуємо кількість водіїв, що їздять по маршрутам, якщо кількість цих маршрутів для водія більше ніж 0 і також сортуємо від більшого до меншого)

1. з використанням HAVING без GROUP BY;

SELECT SUM(parking\_spaces\_num) AS total\_parking FROM parking HAVING COUNT(\*) > 3;

(Вивести суму парків для тролейбусів тільки тоді, коли всього таких парків більше 3

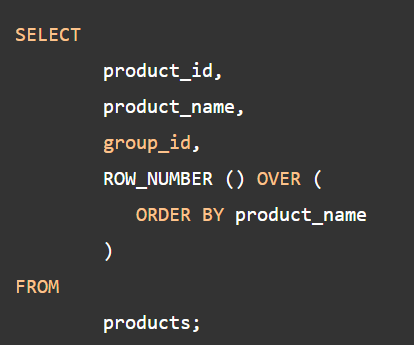
SELECT COUNT(\*) FROM parking;

(Дивимось скільки всього парків)

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

SELECT full\_name, roll\_number, ROW\_NUMBER () OVER(PARTITION BY roll\_number) FROM driver

(row\_number() дає кожному запису свій номер, тож ми «рахуємо» водіїв але коли табельний номер змінюється, ми обнуляємо лічильник)



1. з використанням сортування по декільком стовпцям.

SELECT \* FROM brand ORDER BY passenger\_num, brand\_name;

(Сортуємо марки по кількості пасажирів(якщо пасажирів однакова к-сть, то відсортує по алфавіту)

2. Робота з представленнями (view):

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

CREATE OR REPLACE VIEW route\_stops\_adr (route\_id, start\_adr, end\_adr) AS

SELECT route\_id,

(SELECT stop\_adr FROM stops WHERE stop\_name = start\_stop\_name),

(SELECT stop\_adr FROM stops WHERE stop\_name = end\_stop\_name)

FROM route;

Select \* from route\_stops\_adr

(view складається з id маршруту, початку та кінця адреси. В підзапитах ми шукаємо де назви початкової та кінцевої адреси співпадають)

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

CREATE OR REPLACE VIEW master\_route (start\_adr, end\_adr, route) AS

SELECT start\_adr, end\_adr, brand\_name FROM route\_stops\_adr, trolley, route WHERE route\_stops\_adr.route\_id = route.route\_id AND route.route\_id = trolley.trolley\_id;

SELECT \* FROM master\_route;

(Ми до минулої view хочемо добавити марки, що їздять по цих маршрутах та адресах відповідго )

SELECT \* FROM master\_route;

c. модифікувати представлення (MYSQL!!!!!!!!!!!!!!!!!!)

RENAME master\_route TO my\_route;

(Просто переіменуємо та надамо овнера)

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

Sp\_help = SHOW CREATE TABLE yourTableName;

Sp\_helptext = SHOW CREATE PROCEDURE <name>

**Висновок:** виконуючи лабораторну роботу, я вивчив оператори, які використовуються в реляційних СУБД, для вибірки даних з таблиць, групування та сортування даних, навчився використовувати вбудовані функції в запитах, дослідив призначення представлень (view) баз даних, синтаксис та семантику команд SQL для їх створення, зміни та видалення, системних збережених процедур для отримання інформації про представлення