МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

факультет програмної інженерії та бізнесу

кафедра інженерії програмного забезпечення

**Лабораторна робота №4**

з дисципліни « Реляційні бази даних »

*назва дисципліни*

на тему: «Функції, Процедури та Курсори Баз Даних»

Виконав: студент 2 курсу групи № 622п

освітньої програми

121 інженерія програмного забезпечення

(шифр і назва ОП)

Зайченко Я.І.

(прізвище й ініціали студента)

Прийняв: канд.тех.наук, каф. 603

Манжос Ю.С.

(посада, науковий ступінь, прізвище й ініціали)

Кількість балів:

Харків – 2024

ПОСТАНОВКА ЗАДАЧІ

1. Визначити склад процедур та функцій.
2. Для кожної таблиці з автоінкрементним первинним ключем, написати функцію, яка приймає значення полів запису, крім первинного ключа. Ця функція має отримувати новий первинний ключ, вставляти новий запис в таблицю з цим ключем і з зазначеними у вигляді параметрів значеннями полів. Функція має повертати первинний ключ вставленої записи або NULL , якщо запис НЕ був вставлений.
3. Створити всі збережені процедури та функції (всі процедури повинні передбачати обробку аномальних ситуацій).
4. Перевірити коректність курсорів.
5. Оформити звіт.

**Варіант завдань – 10**

Розробити:

1. процедуру, яка по номеру дисконту перевіряє кількість покупок бензину, і, якщо це число більше 10 і не призначена знижка, то встановити знижку у 3%;
2. функцію, яка розраховує вартість купленого бензину з урахуванням поточної вартості бензину і знижки до дисконту;
3. процедуру, що видаляє записи про продажах, яким більше одного року.

Виконання роботи

Визначення складу процедур та функцій БД з обробкою виняткових ситуацій

Процедура, яка по номеру дисконту перевіряє кількість покупок бензину, і якщо це число більше 10 та не призначена знижка, то встановити знижку 3%;

DELIMITER $$

CREATE PROCEDURE `CheckAndSetDiscount`(IN `discountNumber` INT)

BEGIN

DECLARE purchaseCount INT;

DECLARE Log\_mess VARCHAR(255);

-- Перевіряємо існування запису в таблиці клієнт

IF NOT EXISTS (SELECT 1 FROM клієнт WHERE idAccount = discountNumber) THEN

-- Якщо немає, звітуємо про це

SET Log\_mess = 'Акаунт клієнта з таким номером не існує.';

ELSE

-- Отримуємо кількість покупок бензину для вказаного номеру дисконту

SELECT COUNT(\*) INTO purchaseCount

FROM продаж

WHERE НомерДисконту = discountNumber;

-- Перевіряємо кількість покупок

IF purchaseCount > 10 THEN

-- Перевіряємо існування знижки

IF NOT EXISTS (SELECT 1 FROM клієнт WHERE idAccount = discountNumber AND Percent = '3') THEN

-- Встановлюємо знижку в 3%, якщо її немає

UPDATE `клієнт` SET Percent = '3' WHERE idAccount = discountNumber;

SET Log\_mess = 'Знижка 3% успішно встановлена.';

ELSE

-- Перевіряємо наявність знижки меньше 3

IF (SELECT Percent FROM клієнт WHERE idAccount = discountNumber) < '3' THEN

-- Якщо знижка вже є, але меньше 3% - оновлюємо значення

UPDATE клієнт SET Percent = '3' WHERE idAccount = discountNumber;

SET Log\_mess = 'Знижка 3% успішно оновлена.';

ELSE

SET Log\_mess = 'Знижка більше 3% вже призначена.';

END IF;

END IF;

ELSE

SET Log\_mess = 'Кількість покупок становить менше 10.';

END IF;

END IF;

SELECT Log\_mess AS Message;

END$$

DELIMITER ;

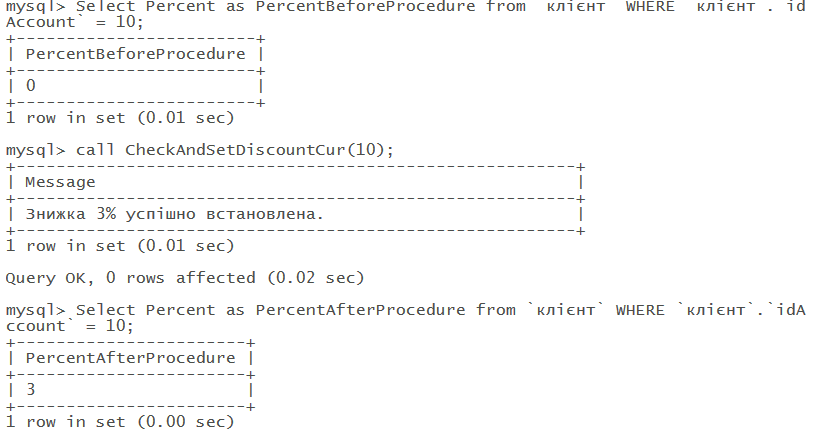


Рисунок 1 – призначення знижки

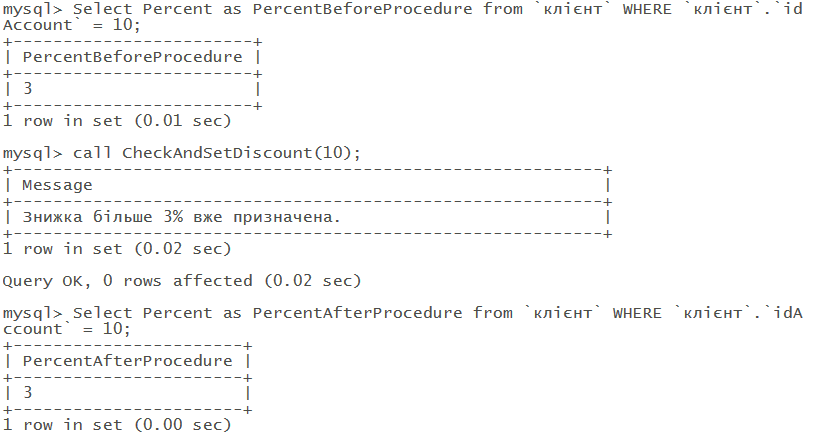


Рисунок 1 – знижка вже призначена, або більше 3

Функція, яка розраховує вартість купленого бензину з урахуванням поточної вартості бензину і знижки до дисконту;

DELIMITER $$

CREATE FUNCTION CalculateCostDiscounted(discountID INT, fuelType ENUM('А-76','А-92','А-95','А-98'), fuelAmount DECIMAL(10,2))

RETURNS DECIMAL(10,2)

BEGIN

DECLARE currentPrice DECIMAL(10,2); -- Поточна ціна пального

DECLARE discountIndex INT; -- Індекс знижки

DECLARE totalCost DECIMAL(10,2); -- Загальна вартість

DECLARE errorMessage VARCHAR(255); -- Повідомлення про помилку

-- Перевіряємо наявність запису про клієнта з заданим idAccount

IF EXISTS (SELECT 1 FROM клієнт WHERE idAccount = discountID) THEN

-- Отримуємо поточну ціну бензину обраного типу

SELECT ЦінаПродажу\_грн INTO currentPrice

FROM `вартість бензину`

WHERE МаркаБенизну = fuelType

ORDER BY ДатаВстановленняВартості DESC

LIMIT 1;

-- Отримуємо індекс знижки клієнта з таблиці клієнт і віднімаємо 1

SELECT Percent INTO discountIndex

FROM клієнт

WHERE idAccount = discountID;

-- Перевіряємо, щоб індекс знижки був не NULL

IF discountIndex IS NULL THEN

SET discountIndex = 0; -- Якщо індекс знижки NULL, встановлюємо його в 0

ELSE

SET discountIndex = discountIndex - 1; -- Віднімаємо 1 з індекса знижки

END IF;

-- Розраховуємо загальну вартість з урахуванням знижки

SET totalCost = fuelAmount \* currentPrice \* (1 - discountIndex / 100);

RETURN totalCost;

ELSE

-- Якщо запис про клієнта не існує, встановлюємо повідомлення про помилку

SET errorMessage = 'Акаунт клієнта з таким номером не існує.';

RETURN errorMessage;

END IF;

END$$

DELIMITER ;

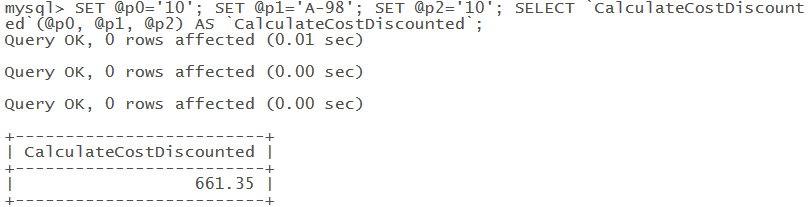


Рисунок 2 – отримання результату ціни зі знижкою

Процедура, що видаляє записи про продажах, яким більше одного року.

DELIMITER $$

CREATE PROCEDURE DeleteOldSales()

BEGIN

DECLARE cutoffDate DATE; -- Дата обрізки

DECLARE rowsAffected INT; -- Кількість змінених рядків

-- Обчислюємо дату, яка знаходиться на рік назад від поточної дати

SET cutoffDate = DATE\_SUB(CURRENT\_DATE(), INTERVAL 1 YEAR);

-- Видаляємо записи про продажі, зроблені більше року тому

DELETE FROM продаж

WHERE ДатаПродажу < cutoffDate; -- Отримуємо кількість видалених записів

SET rowsAffected = ROW\_COUNT(); -- Перевіряємо, чи були видалені записи

IF rowsAffected > 0 THEN -- Якщо видалені записи, виводимо повідомлення про успішне видалення

SELECT CONCAT('Успішно видалено ', rowsAffected, ' записів') AS Message;

ELSE

-- Якщо записи не були видалені, виводимо повідомлення про відсутність записів для видалення

SELECT 'Записів для видалення не знайдено' AS Message;

END IF;

END$$

DELIMITER ;

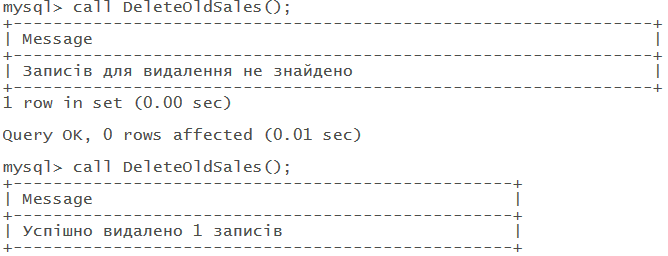


Рисунок 3 – результат виконання процедури з видалення старих записів продажів

Для кожної таблиці з автоінкрементним первинним ключем, написати функцію, яка приймає значення полів запису, крім первинного ключа. Ця функція має отримувати новий первинний ключ, вставляти новий запис в таблицю з цим ключем і з зазначеними у вигляді параметрів значеннями полів. Функція має повертати первинний ключ вставленого запису або NULL , якщо запис НЕ був вставлений.

**для таблиці клієнт:**

DELIMITER $$

CREATE FUNCTION `InsertClient`(p\_DateGiven DATETIME, p\_LastName VARCHAR(50), p\_FirstName VARCHAR(20), p\_Father VARCHAR(50), p\_Percent ENUM('0','1','2','3','4','5'))

RETURNS INT DETERMINISTIC

BEGIN

DECLARE v\_idAccount INT;

-- Встановлюємо p\_DateGiven в поточний час, якщо воно NULL або '0000-00-00 00:00:00'

IF p\_DateGiven IS NULL OR p\_DateGiven = '0000-00-00 00:00:00' THEN

SET p\_DateGiven = CURRENT\_TIMESTAMP;

END IF;

INSERT INTO клієнт (DateGiven, LastName, FirstName, Father, Percent)

VALUES (p\_DateGiven, p\_LastName, p\_FirstName, p\_Father, p\_Percent);

-- Отримуємо ID вставленого запису

SET v\_idAccount = LAST\_INSERT\_ID();

-- Повертаємо ID вставленого запису або NULL, якщо запис не був вставлений

IF ROW\_COUNT() = 0 THEN

RETURN NULL;

ELSE

RETURN v\_idAccount;

END IF;

END$$

DELIMITER ;

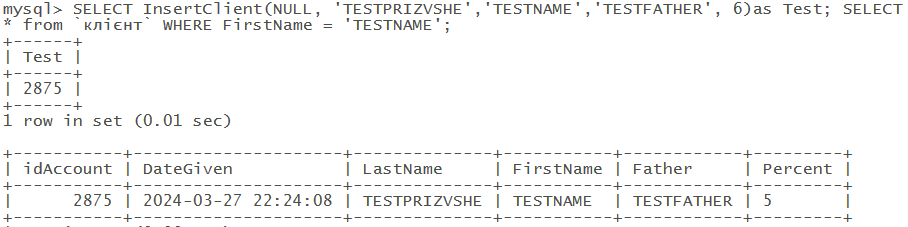
****

Рисунок 1 – створення нового клієнта без введення Id

**для продажу:**

 DELIMITER $$

CREATE FUNCTION InsertSale(

p\_DateGiven DATETIME,

p\_DiscountNumber INT,

p\_GasolineBrand VARCHAR(10),

p\_LitersAmount INT,

p\_CostCode INT

)

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE v\_SaleId INT;

-- Встановлюємо p\_DateGiven в поточний час, якщо воно NULL або '0000-00-00 00:00:00'

IF p\_DateGiven IS NULL OR p\_DateGiven = '0000-00-00 00:00:00' THEN

SET p\_DateGiven = CURRENT\_TIMESTAMP;

END IF;

-- Додаємо запис про продаж

INSERT INTO продаж (ДатаПродажу, НомерДисконту, МаркаБензину, КількістьЛітрів\_л, КодВартості)

VALUES (p\_DateGiven, p\_DiscountNumber, p\_GasolineBrand, p\_LitersAmount, p\_CostCode);

-- Отримуємо ID вставленого запису

SET v\_SaleId = LAST\_INSERT\_ID();

-- Повертаємо ID вставленого запису або NULL, якщо запис не був вставлений

RETURN IF(v\_SaleId IS NULL, NULL, v\_SaleId);

END$$

DELIMITER ;

PREPARE t3 FROM 'SELECT DATE(ДатаПродажу) AS Дата\_продажу, SUM(КількістьЛітрів\_л) AS Сумарна\_кількість\_проданих\_літрів FROM продаж

WHERE МаркаБензину = ? AND НомерДисконту IN (SELECT idAccount FROM клієнт WHERE Percent = ?) GROUP BY DATE(ДатаПродажу)'; SET @N = 'А-95'; SET @M = '3'; EXECUTE t3 USING @N, @M;

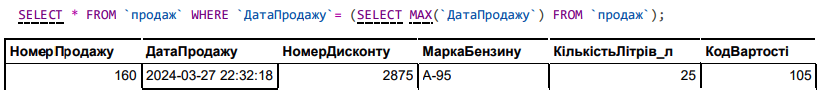
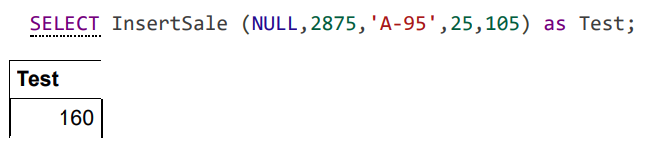


Рисунок 5 – створення нового продажу без введення НомеруПродажу

ВИСНОВОК

У ході виконання лабораторної роботи було створено та використано процедури, функції, курсори до створеної на першій лабораторній роботі бази даних.

Навчилися використовувати та створювати функції, процедури та курсори в базах даних для виконання повторюваних дій з певними умовами, що дозволяє більш ефективно використовувати бази даних.