

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського» ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування та спеціалізованих комп'ютерних систем

Лабораторна робота № 2

з дисципліни «Бази даних та засоби управління»

Тема: «Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL»

Виконав: студент III курсу

ФПМ групи КВ-84

Бендецький Є.О.

Перевірив: Петрашенко А.В.

Київ

2020

Завдання

Відношення	Атрибут	Тип даних		
Відношення "Invoices" Вміщує інформацію про транспортні накладні	пит — унікальний номер накладної date_departure — дата оформлення та відправлення посилки date_arrival — дата прибуття посилки shipping_cost — вартість доставки sender_ipn — ІПН відправника recipient_ipn — ІПН отримувача warehouse_dep_num — номер складу, з якого відправлено посилку warehouse_arr_num — номер складу до якого прямує посилка	Числовий Дата Дата Грошовий Числовий Числовий Числовий Числовий		
Відношення "Wares" Вміщує інформацію про товари та вантажі, що перевозяться	id — унікальний ідентифікатор товару height — висота посилки width — ширина посилки depth — глибина посилки weight — вага посилки description — опис вантажу invoice_num — номер накладної, до якої належить ця посилка	Числовий Числовий Числовий Числовий Числовий Текстовий Числовий		
Відношення "Contragents" Вміщує інформацію про осіб, які є відправниками або отримувачами	ipn — ідентифікаційний податковий номер особи (ІПН) name — ПІБ особи phone_number — мобільний номер телефона особи	Числовий Текстовий Текстовий		
Відношення "Warehouses" Вміщує інформацію про склади, між якими транспортуються вантажі	num — унікальний номер складу address — адреса, за якою знаходиться склад phone_number — номер телефону складу	Числовий Текстовий Текстовий		
Відношення "Cities" Вміщує інформацію про міста, в яких знаходяться відділення	id — унікальний номер міста name — наза міста	Числовий Текстовий		

SQL-запити

```
CREATE TABLE IF NOT EXISTS contragents
                 int PRIMARY KEY,
    name
                  varchar(255) NOT NULL,
    phone number char (15) NOT NULL
);
CREATE TABLE IF NOT EXISTS cities
    id serial PRIMARY KEY,
   name varchar(255) NOT NULL
);
CREATE TABLE IF NOT EXISTS warehouses
    num serial PRIMARY KEY,
address text NOT NULL,
   num
    phone number char (15) NOT NULL,
    city id serial NOT NULL,
    CONSTRAINT city id FOREIGN KEY (city id)
        REFERENCES cities (id)
        ON UPDATE RESTRICT
        ON DELETE RESTRICT
);
CREATE TABLE IF NOT EXISTS invoices
(
                      serial PRIMARY KEY,
    date_departure date NOT NULL,
    date_arrival date,
shipping_cost money NOT NULL,
sender_ipn int NOT NULL,
recipient_ipn int NOT NULL,
    warehouse dep num serial NOT NULL,
    warehouse arr num serial NOT NULL,
    CONSTRAINT recipient ipn FOREIGN KEY (recipient ipn)
        REFERENCES contragents (IPN)
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT sender ipn FOREIGN KEY (sender ipn)
        REFERENCES contragents (IPN)
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT warehouse arr num FOREIGN KEY (warehouse arr num)
        REFERENCES warehouses (num)
        ON UPDATE RESTRICT
        ON DELETE RESTRICT,
    CONSTRAINT warehouse dep num FOREIGN KEY (warehouse dep num)
        REFERENCES warehouses (num)
        ON UPDATE RESTRICT
        ON DELETE RESTRICT
);
CREATE TABLE IF NOT EXISTS goods
               serial PRIMARY KEY,
int NOT NULL,
int NOT NULL,
    height
    width
    depth int NOT NULL, weight int NOT NULL,
    description text,
```

```
invoice num serial NOT NULL,
    CONSTRAINT invoice num FOREIGN KEY (invoice num)
        REFERENCES invoices (num)
        ON UPDATE RESTRICT
        ON DELETE RESTRICT
);
CREATE INDEX IF NOT EXISTS goods descriptions ON goods USING
gin(to tsvector('english', description));
controller/ init .pv
class BaseController(ABC):
    def __init__(self, model: BaseModel, view: BaseView):
        self.__model = model
        self.__view = view
        self. cb show prev state = None
    def show(self, pk: int = None):
        pk_not_specified = pk is None
        if pk not specified:
            pk = self. view.get item pk('Reading')
        try:
            if isinstance(pk, str):
               pk = int(pk)
            item = self.__model.read(pk)
            self.__view.show_item(item)
        except (Exception, psycopg2.Error) as e:
            exception_handler(e, self.__model.rollback)
            self. view.show error(str(e))
        finally:
            if pk not specified:
                self.choose operation()
            else:
                self.show all()
    def insert(self):
        input items =
self. get input items form(self. prompt values for input())
        command = self. view.show input item form(input items, 'Create')
        if command == ConsoleCommands.GO BACK:
            return self.choose operation()
        if command == ConsoleCommands.CONFIRM:
                pk_name = self.__model.primary_key_name
                item =
self. model.create(self. create obj from input(input items))
                self. view.show created item(item, pk name)
            except (Exception, psycopg2.Error) as e:
                exception_handler(e, self._ model.rollback)
                self. view.show error(str(e))
            finally:
                self.choose_operation()
    def update(self):
        pk = self. view.get item pk('Updating')
        try:
            if isinstance(pk, str):
                pk = int(pk)
            item = self.__model.read(pk)
            input items =
self.__get_input_items_form(self._prompt_values_for_input(item, True))
            command = self.__view.show_input_item_form(input_items, 'Update')
            if command == ConsoleCommands.GO BACK:
```

```
return self.choose operation()
            if command == ConsoleCommands.CONFIRM:
                new_item = self._create_obj_from_input(input_items)
                pk_name = self.__model.primary_key_name
setattr(new_item, pk_name, getattr(item, pk_name))
                self.__model.update(new_item)
                self.__view.show_updated_item(item, new_item)
        except (Exception, psycopg2.Error) as e:
            exception_handler(e, self.__model.rollback)
            self. view.show error(str(e))
        finally:
            self.choose operation()
    def delete(self):
        pk = self.__view.get_item pk('Deleting')
            if isinstance(pk, str):
               pk = int(pk)
            item = self. model.read(pk)
            confirm = self. view.confirm deleting form(item)
            if confirm.strip().lower() != "yes":
                return self.choose operation()
            self.__model.delete(pk)
            self. view.show success(f"An item {item} was successfully
deleted")
        except (Exception, psycopg2.Error) as e:
            exception handler(e, self. model.rollback)
            self.__view.show_error(str(e))
        finally:
            self.choose operation()
```

Додавання даних

```
Терминал

Файл Правка Вид Поиск Терминал Справка

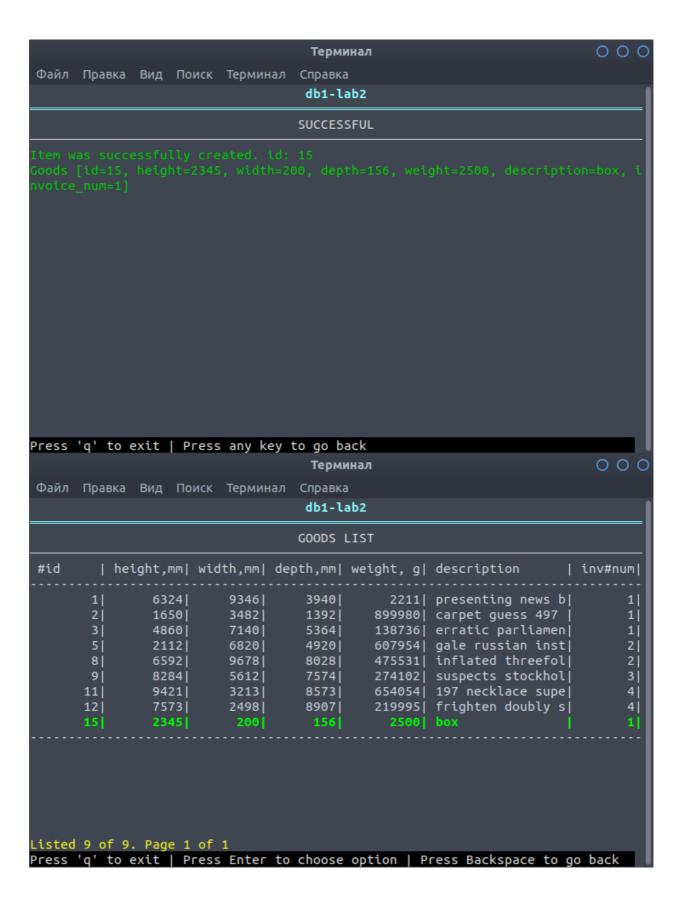
db1-lab2

CREATE GOODS

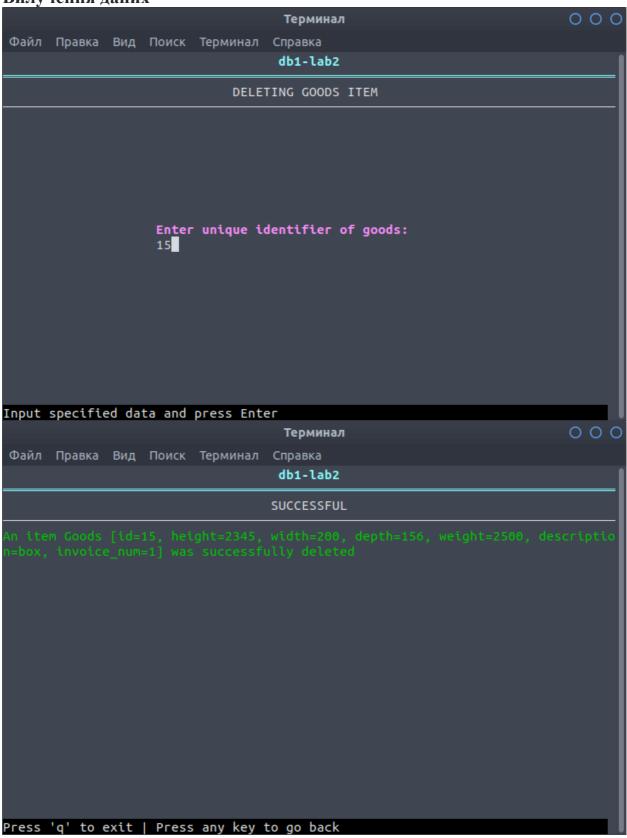
Height (mm): 2345
Width (mm): 200
Depth (mm): 156
Weight (g): 2500
Description (could be empty): box
Invoice number: 1

➤ Confirm entered data

Press 'q' to exit | Press any key to go back | Press Enter to proceed
```



Вилучення даних



```
Терминал
                                                                                    000
Файл Правка Вид Поиск Терминал Справка
                                        db1-lab2
                                       GOODS LIST
 #id
          | height,mm| width,mm| depth,mm| weight, g| description
                                                                               | inv#num|
                                        39401
                                                     2211 <empty>
                                                 899980| carpet guess 497
         2|
                 1650
                             3482
                                       1392
                                                                                        1|
                             7140|
                                                  138736| erratic parliamen|
         3|
                 4860
                                       5364
                                                                                        1|
                                                 607954| gale russian inst|
475531| inflated threefol|
         5|
                            6820
                                       4920
                                                                                        2|
                 2112
                                       8028
                             9678
                                                                                        2|
        8|
                 6592
                                                 274102| suspects stockhol|
654054| 197 necklace supe|
219995| frighten doubly s|
                             5612
        9|
                  8284
                                                                                        3|
                                       8573
        11|
                  9421
                             3213
                                                                                        4
                                       8907|
        12|
                  7573
                             2498
                                                                                        4|
Listed 8 of 8. Page 1 of 1
Press 'q' to exit | Press Enter to choose option | Press Backspace to go back
```

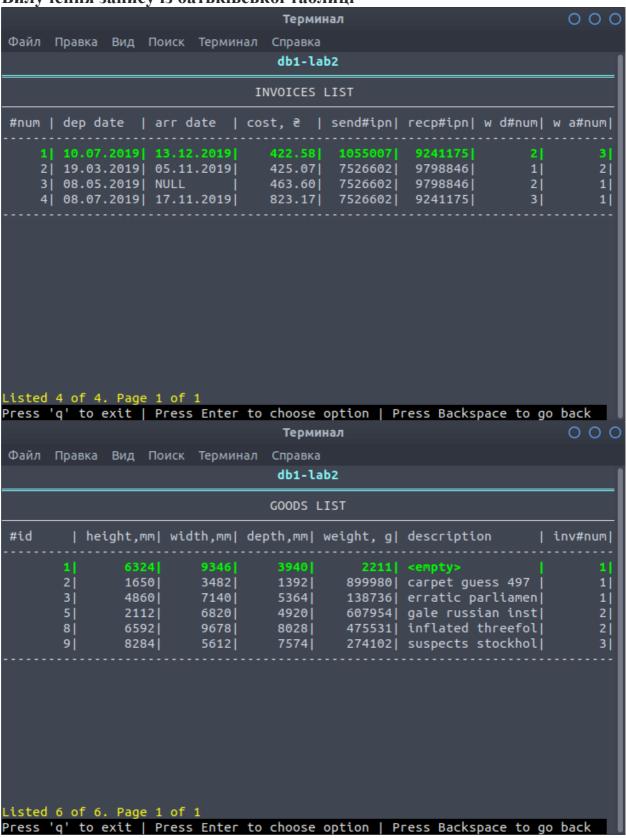
model/common.pv

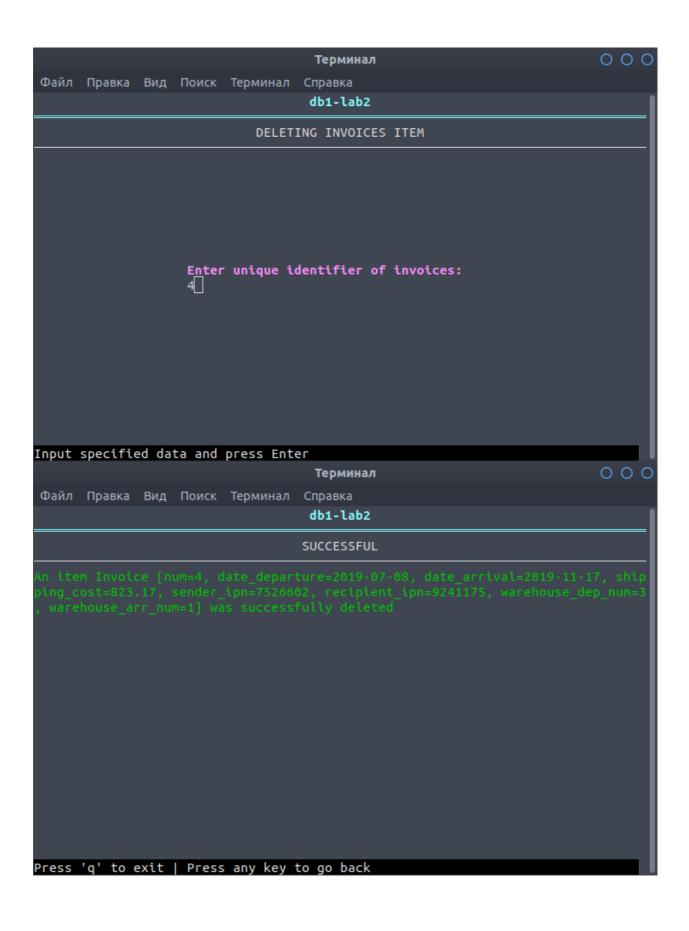
```
def filter_items(self, cost_from: int, cost_to: int, sender_name: str = None,
recipient name: str = None):
        query = "SELECT num, date departure, date arrival, shipping cost,
c1.name, " \
                "c1.phone number, c2.name, c2.phone number from invoices i "
                "INNER JOIN contragents c1 on i.sender ipn = c1.ipn " \
                "INNER JOIN contragents c2 on i.recipient ipn = c2.ipn " \
                "WHERE " \
                "shipping cost::numeric BETWEEN (%(min)s) AND (%(max)s)"
        if isinstance(sender name, str):
            query += " AND cl.name = (%(sender)s)"
        if isinstance(recipient_name, str):
            query += " AND c2.name = (%(recipient)s)"
        self. cursor.execute(query, {'min': cost from, 'max': cost to,
                                      'sender': sender name, 'recipient':
recipient name})
        rows = self. cursor.fetchall()
        if isinstance (rows, list):
           return rows
        else:
            raise Exception("There are no items")
    def fulltext search(self, query: str, including: bool):
        if not including:
            words = query.split()
            if len(words) > 0:
               words[0] = "!" + words[0]
            counter = 1
            while counter < len(words):</pre>
                words[counter] = "& !" + words[counter]
            query = ' '.join(words)
        query excluding = "SELECT ts headline (description, q) " \
```

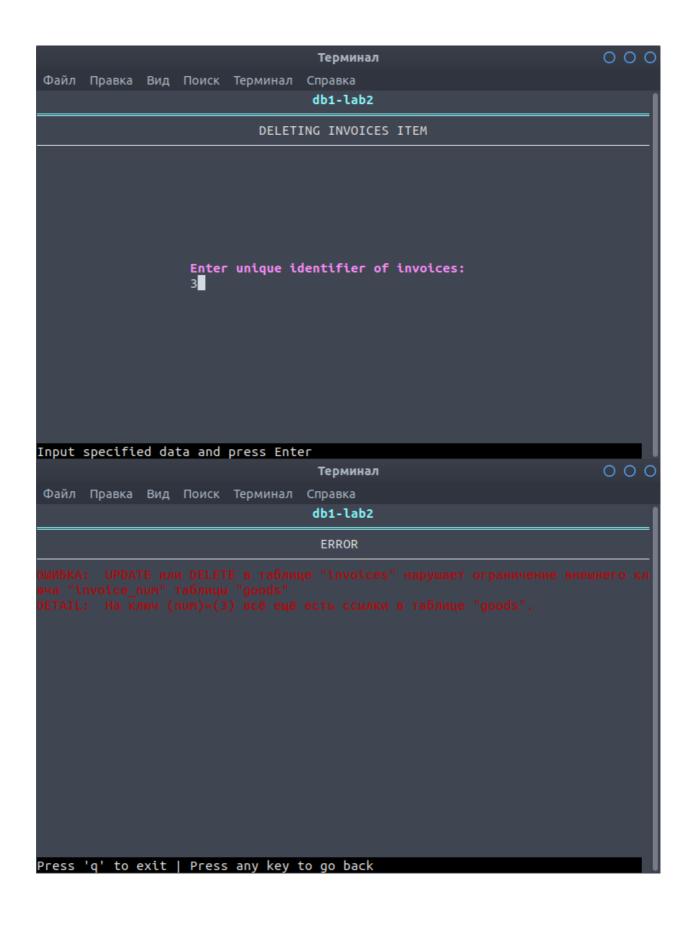
```
"FROM goods, to tsquery('english', %(query)s) AS q
" \
                          "WHERE to tsvector('english', description) @@ q "
        query including = "SELECT ts headline (description, q) " \
                          "FROM goods, plainto tsquery('english', %(query)s)
AS q " \
                          "WHERE to tsvector('english', description) @@ q "
        self.__cursor.execute(query_including if including else
query excluding, {'query': query})
        rows = self.__cursor.fetchall()
        if isinstance(rows, list):
            return rows
        else:
            raise Exception ("There are no items")
model/__init__.py
class BaseModel(ABC):
    def __init__(self, connection, insert_query, select_query, update_query,
                delete_query, select_all_query, count_query,
primary_key_name):
        self.\_connection = connection
        self. cursor = connection.cursor(cursor factory=DictCursor)
       self.__insert_query = insert query
       self.__select_query = select_query
       self.__update_query = update_query
       self.__delete_query = delete_query
       self.__select_all_query = select all query
        self.__count_query = count_query
        self. primary key name = primary key name
    def create(self, item: object):
        should return id = "returning" in self. insert query.lower()
        if not self. is valid item dict(item. dict , not should return id):
            raise Exception("Item is not valid")
        self. cursor.execute(self. insert query, item. dict )
        self. connection.commit()
        if should return id:
            row = self. cursor.fetchone()
            if row is not None and isinstance (row[self. primary key name],
int):
                self. insert pk in item(item, row[self. primary key name])
                return item
            else:
                raise Exception("No rows received from DB")
    def read(self, pk: int):
        if not isinstance(pk, int):
            raise Exception("Primary key should be an integer")
        self. cursor.execute(self. select query, [pk])
        row = self. cursor.fetchone()
        if row is not None and self. is valid item dict(row):
            return self. get item from row(row)
        else:
            raise Exception(f"No item with such primary key {pk} was found")
    def read all(self, offset: int = 0, limit: int = None):
        self. cursor.execute(self. select all query, {'limit': limit,
'offset': offset})
        rows = self. cursor.fetchall()
        if isinstance(rows, list) and all(self. is valid item dict(row) for
row in rows):
            return [self. get item from row(row) for row in rows]
        else:
```

```
raise Exception ("There are no items")
    def update(self, item: object):
        if not self. is valid item dict(item. dict ):
            raise Exception ("Item is not valid")
        self._cursor.execute(self.__update_query, item.__dict__)
        self. connection.commit()
    def delete(self, pk: int):
        if not isinstance(pk, int):
            raise Exception ("Primary key should be an integer")
        self. cursor.execute(self. delete query, [pk])
        self. connection.commit()
model/goods.pv
class Goods:
    def init (self, height: int, width: int, depth: int, weight: int,
                invoice num: int, description: str = None, g id: int =
None):
       self.id = g_id
       self.height = height
       self.width = width
       self.depth = depth
       self.weight = weight
       self.description = description
       self.invoice num = invoice num
    def str (self):
        return f"Goods [id={self.id}, height={self.height},
width={self.width}, depth={self.depth}, " \
               f"weight={self.weight}, description={self.description},
invoice num={self.invoice num}]"
class GoodsModel(BaseModel):
    def init (self, connection):
        insert query = "INSERT INTO goods (height, width, depth, weight,
description, invoice num) " \
                       "VALUES (% (height)s, % (width)s, % (depth)s, % (weight)s,
%(description)s, %(invoice num)s)" \
                       "RETURNING id"
        select query = "SELECT * FROM goods WHERE id = %s"
        update query = "UPDATE goods SET height = % (height)s, width =
%(width)s, depth = %(depth)s, " \
                       "weight = %(weight)s, description = %(description)s,
invoice_num = %(invoice_num)s " \
                       "WHERE id = %(id)s"
        delete query = "DELETE FROM goods WHERE id = %s"
        select all query = "SELECT * FROM goods ORDER BY id OFFSET %(offset)s
LIMIT %(limit)s"
        count query = "SELECT COUNT(*) FROM goods"
        primary key name = "id"
        super(). init (connection, insert query, select query,
update_query,
                         delete query, select all query, count query,
primary key name)
```

Вилучення запису із батьківської таблиці







						Терми	нал						000
Файл	Правка	Вид	Поиск	Термин	ал	Справка							
						db1-la	b2						
INVOICES LIST													
#num	dep da	ate	arr d	ate	cos	st, 2	send#ip	n	recp#ipn	W (d#num	W	a#num
	10.07						105500				2		3 2
	19.03 08.05			.2019		425.07 463.60	752660 752660				1 2		2 1
		. 2019											
Listed	3 of 3	. Page	e 1 of	1									
Press	'q' to	exit	Press	Enter	to	choose	option	РΓ	ess Backs	pace	e to go	b	ack