



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

SQL-запити

```
CREATE TABLE IF NOT EXISTS contragents
(
    IPN          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,
    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
(
    num          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
(
    id          serial PRIMARY KEY,
    height      int NOT NULL,
    width       int NOT NULL,
    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__.py

```

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:
            try:
                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')
    try:
        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

```

Терминал

ФайлПравкаВидПоискТерминалСправка

db1-lab2

SUCCESSFUL

Item was successfully created. id: 15
Goods [id=15, height=2345, width=200, depth=156, weight=2500, description=box, invoice_num=1]

Press 'q' to exit | Press any key to go back

Терминал

ФайлПравкаВидПоискТерминалСправка

db1-lab2

GOODS LIST

| #id | height,mm | width,mm | depth,mm | weight, g | description | inv#num |
|-----|-----------|----------|----------|-----------|-------------------|---------|
| 1 | 6324 | 9346 | 3940 | 2211 | presenting news b | 1 |
| 2 | 1650 | 3482 | 1392 | 899980 | carpet guess 497 | 1 |
| 3 | 4860 | 7140 | 5364 | 138736 | erratic parliamen | 1 |
| 5 | 2112 | 6820 | 4920 | 607954 | gale russian inst | 2 |
| 8 | 6592 | 9678 | 8028 | 475531 | inflated threefol | 2 |
| 9 | 8284 | 5612 | 7574 | 274102 | suspects stockhol | 3 |
| 11 | 9421 | 3213 | 8573 | 654054 | 197 necklace supe | 4 |
| 12 | 7573 | 2498 | 8907 | 219995 | frighten doubly s | 4 |
| 15 | 2345 | 200 | 156 | 2500 | box | 1 |

Listed 9 of 9. Page 1 of 1

Press 'q' to exit | Press Enter to choose option | Press Backspace to go back

Вилучення даних

```
Терминал
Файл  Правка  Вид  Поиск  Терминал  Справка
db1-lab2
DELETING GOODS ITEM

Enter unique identifier of goods:
15

Input specified data and press Enter

Терминал
Файл  Правка  Вид  Поиск  Терминал  Справка
db1-lab2
SUCCESSFUL

An item Goods [id=15, height=2345, width=200, depth=156, weight=2500, description=box, invoice_num=1] was successfully deleted

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

| Терминал | | | | | | | |
|--|-----------|----------|----------|-----------|-------------------|--|---------|
| Файл Правка Вид Поиск Терминал Справка | | | | | | | |
| db1-lab2 | | | | | | | |
| GOODS LIST | | | | | | | |
| #id | height,mm | width,mm | depth,mm | weight, g | description | | inv#num |
| 1 | 6324 | 9346 | 3940 | 2211 | <empty> | | 1 |
| 2 | 1650 | 3482 | 1392 | 899980 | carpet guess 497 | | 1 |
| 3 | 4860 | 7140 | 5364 | 138736 | erratic parliamen | | 1 |
| 5 | 2112 | 6820 | 4920 | 607954 | gale russian inst | | 2 |
| 8 | 6592 | 9678 | 8028 | 475531 | inflated threefol | | 2 |
| 9 | 8284 | 5612 | 7574 | 274102 | suspects stockhol | | 3 |
| 11 | 9421 | 3213 | 8573 | 654054 | 197 necklace supe | | 4 |
| 12 | 7573 | 2498 | 8907 | 219995 | frighten doubly s | | 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.py

```
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 c1.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):
                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.py

```

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

```
Терминал
Файл Правка Вид Поиск Терминал Справка
db1-lab2

INVOICES LIST

#num | dep date | arr date | cost, ₴ | send#ipn| recp#ipn| w d#num| w a#num|
-----|-----|-----|-----|-----|-----|-----|-----|
1| 10.07.2019| 13.12.2019| 422.58| 1055007| 9241175| 2| 3|
2| 19.03.2019| 05.11.2019| 425.07| 7526602| 9798846| 1| 2|
3| 08.05.2019| NULL| 463.60| 7526602| 9798846| 2| 1|
4| 08.07.2019| 17.11.2019| 823.17| 7526602| 9241175| 3| 1|
-----|-----|-----|-----|-----|-----|-----|-----|

Listed 4 of 4. Page 1 of 1
Press 'q' to exit | Press Enter to choose option | Press Backspace to go back

Терминал
Файл Правка Вид Поиск Терминал Справка
db1-lab2

GOODS LIST

#id | height,mm| width,mm| depth,mm| weight, g| description | inv#num|
-----|-----|-----|-----|-----|-----|-----|
1| 6324| 9346| 3940| 2211| <empty> | 1|
2| 1650| 3482| 1392| 899980| carpet guess 497 | 1|
3| 4860| 7140| 5364| 138736| erratic parliamen | 1|
5| 2112| 6820| 4920| 607954| gale russian inst | 2|
8| 6592| 9678| 8028| 475531| inflated threefol | 2|
9| 8284| 5612| 7574| 274102| suspects stockhol | 3|
-----|-----|-----|-----|-----|-----|-----|

Listed 6 of 6. Page 1 of 1
Press 'q' to exit | Press Enter to choose option | Press Backspace to go back
```

```
Терминал
Файл Правка Вид Поиск Терминал Справка
db1-lab2
DELETING INVOICES ITEM
Enter unique identifier of invoices:
4
Input specified data and press Enter
```

```
Терминал
Файл Правка Вид Поиск Терминал Справка
db1-lab2
SUCCESSFUL
An item Invoice [num=4, date_departure=2019-07-08, date_arrival=2019-11-17, shipping_cost=823.17, sender_ipn=7526602, recipient_ipn=9241175, warehouse_dep_num=3, warehouse_arr_num=1] was successfully deleted
Press 'q' to exit | Press any key to go back
```

```
Терминал
Файл Правка Вид Поиск Терминал Справка
db1-lab2
DELETING INVOICES ITEM
Enter unique identifier of invoices:
3
Input specified data and press Enter
```

```
Терминал
Файл Правка Вид Поиск Терминал Справка
db1-lab2
ERROR
ОШИБКА: UPDATE или DELETE в таблице "invoices" нарушает ограничение внешнего ключа "invoice_num" таблицы "goods"
DETAIL: На ключ (num)=(3) всё ещё есть ссылки в таблице "goods".
Press 'q' to exit | Press any key to go back
```

```
Терминал
Файл Правка Вид Поиск Терминал Справка
db1-lab2

INVOICES LIST

#num | dep date | arr date | cost, € | send#ipn| recp#ipn| w d#num| w a#num|
-----
1| 10.07.2019| 13.12.2019| 422.58| 1055007| 9241175| 2| 3|
2| 19.03.2019| 05.11.2019| 425.07| 7526602| 9798846| 1| 2|
3| 08.05.2019| NULL | 463.60| 7526602| 9798846| 2| 1|
-----

Listed 3 of 3. Page 1 of 1
Press 'q' to exit | Press Enter to choose option | Press Backspace to go back
```