

Бази даних та інформаційні системи

ЛАБОРАТОРНА РОБОТА №4

Обмеження цілісності даних та індекси в SQL

Виконав:

Ст. Яцуляк Андрій

Група ПМІ-21

Оцінка

Перевірила:

доц. Малець Р.Б.

Тема: Обмеження цілісності даних та індекси в SQL.

Мета роботи: Ознайомлення з поняттями обмеження цілісності даних та індексами в SQL, їх створенням і використанням.

Розв'язання

```
CREATE TABLE IF NOT EXISTS public.publisher
(
    publisher_id integer NOT NULL DEFAULT 'nextval('publisher_publisher_id_seq'::regclass)',
    publisher_name base_name COLLATE pg_catalog."default",
    publisher_address base_address COLLATE pg_catalog."default",
    publisher_phone_num base_phone_num COLLATE pg_catalog."default",
    CONSTRAINT publisher_pkey PRIMARY KEY (publisher_id)
)

CREATE TABLE IF NOT EXISTS public.form
(
    form_id integer NOT NULL DEFAULT 'nextval('"Form_form_id_seq"::regclass)',
    user_name base_name COLLATE pg_catalog."default",
    user_surname base_surname COLLATE pg_catalog."default",
    adress base_address COLLATE pg_catalog."default",
    phone_num base_phone_num COLLATE pg_catalog."default",
    date base_date,
    CONSTRAINT "Form_pkey" PRIMARY KEY (form_id)
)

CREATE TABLE IF NOT EXISTS public.catalogue
(
    catalogue_id integer NOT NULL DEFAULT 'nextval('catalogue_catalogue_id_seq'::regclass)',
    genre base_genre COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT catalogue_pkey PRIMARY KEY (catalogue_id)
)

CREATE TABLE IF NOT EXISTS public.borrowing
(
    borrowing_id integer NOT NULL DEFAULT 'nextval('borrowing_borrowing_id_seq'::regclass)',
    form_id integer,
    book_id integer,
    date base_date NOT NULL,
    CONSTRAINT borrowing_pkey PRIMARY KEY (borrowing_id),
    CONSTRAINT borrowing_book_id_fkey FOREIGN KEY (book_id)
        REFERENCES public.book (book_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT borrowing_form_id_fkey FOREIGN KEY (form_id)
        REFERENCES public.form (form_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)
```

```

CREATE TABLE IF NOT EXISTS public.book
(
    book_id integer NOT NULL DEFAULT 'nextval('book_book_id_seq'::regclass)',
    book_name base_name COLLATE pg_catalog."default",
    publisher_id integer,
    catalogue_id integer,
    status base_status COLLATE pg_catalog."default" NOT NULL,
    pub_year smallint NOT NULL,
    CONSTRAINT book_pkey PRIMARY KEY (book_id),
    CONSTRAINT book_catalogue_id_fkey FOREIGN KEY (catalogue_id)
        REFERENCES public.catalogue (catalogue_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT book_publisher_id_fkey FOREIGN KEY (publisher_id)
        REFERENCES public.publisher (publisher_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT pub_year_check CHECK (pub_year >= 1800 AND pub_year <= 2999)
)
CREATE TABLE IF NOT EXISTS public.autor
(
    autor_id integer NOT NULL DEFAULT 'nextval('autor_autor_id_seq'::regclass)',
    autor_name base_name COLLATE pg_catalog."default",
    autor_surname base_surname COLLATE pg_catalog."default",
    CONSTRAINT autor_pkey PRIMARY KEY (autor_id)
)
CREATE TABLE IF NOT EXISTS public.book_autor
(
    book_id integer NOT NULL,
    autor_id integer NOT NULL,
    CONSTRAINT book_autor_pkey PRIMARY KEY (book_id, autor_id),
    CONSTRAINT book_autor_autor_id_fkey FOREIGN KEY (autor_id)
        REFERENCES public.autor (autor_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT book_autor_book_id_fkey FOREIGN KEY (book_id)
        REFERENCES public.book (book_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

```

Індекси

Для таблиці «Form» я створив індекс для «user_surname» та для «user_name». Але в даному випадку ці індекси не працюють. Індекс використовується для підвищення швидкодії запитів до бази даних, шляхом створення дерева пошуку для визначення розташування даних в таблиці. Однак, якщо таблиця має малу кількість даних, то індекс може не мати достатньої користі, щоб виправдати його використання, оскільки звичайний пошук по таблиці може бути достатньо швидким.

Також варто враховувати, що якщо в таблиці дуже мало записів, то оптимізатор запитів може вирішити просто прочитати всі записи в таблиці і відсортувати їх в пам'яті, замість використання індексу. Це може бути більш ефективним варіантом, ніж використання індексу.

Query

Query History

1

`explain`

2

`select * from form`









3


`where user_surname = 'Yatsuliak'`

Data Output

Messages

Notifications



	QUERY PLAN	
	text	
1	Seq Scan on form (cost=0.00..1.13 rows=1 w...	
2	Filter: ((user_surname)::text = 'Yatsuliak'::text)	