Національний технічний університет України

“Київський політехнічний інститут”

БАЗИ ДАНИХ ТА ІНФОРМАЦІЙНІ СИСТЕМИ

Розрахункова робота на тему:

“Створення бази даних товарів на складі”

Виконав:

Студент групи ФБ-74

Варіант 7

Новіков Олексій

Перевірив:

Коломицев М.В.

Київ 2019

Мета роботи: Освоєння методів проектування баз даних і роботи з базами даних у середовищі СУБД MS SQL Server.

Варіант 7

Предметна область – Складское підприємство.
Задачі, що вирішує інформаційна система – наявність товарів на складі.
Завдання на обробку даних:

1. Створити впорядковані списки:
   - Товарів за категоріями (по складам);
   - Товарів, які є на вибраному складі, але немає на інших складах.
2. обчислення:
   - Перевірити, чи достатньо на складі товарів для виконання конкретного замовлення.
3. корекція:
   - Зміна залишку товару на складі.
4. Звіти виду:
   - "Наявність товарів": Товар - місце зберігання - залишок (з урахуванням одиниці виміру товару).
   - "Замовлені товари" Товар - місце зберігання - замовлену кількість.
   - "Вага замовлення": номер замовлення - загальна кількість замовлених товарів.
   - "Картка товару": номер складу - номер лінії - номер стелажа - одиниця виміру - залишок товару.

1. Завдання Створення діаграми потоків даних (DFD-моделі)

Після аналізу завдання, розробляємо можливу діаграму потоків даних. Для створення DFD-діаграми будемо користуватися сайтом draw.io

Для компактнішого і структурного проектування визначемо декілька підсистем.

Результат роботи видно на рисунках 1,2,3.

Рисунок 1. Підсистема “Визначення типу звіту”

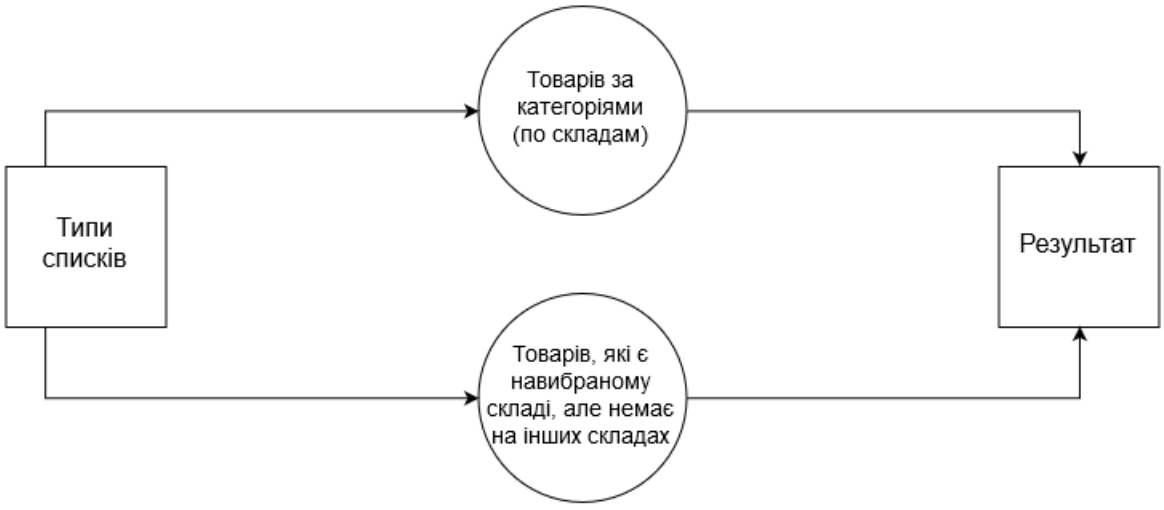


Рисунок 2. Підсистема “Визначення типу списків”

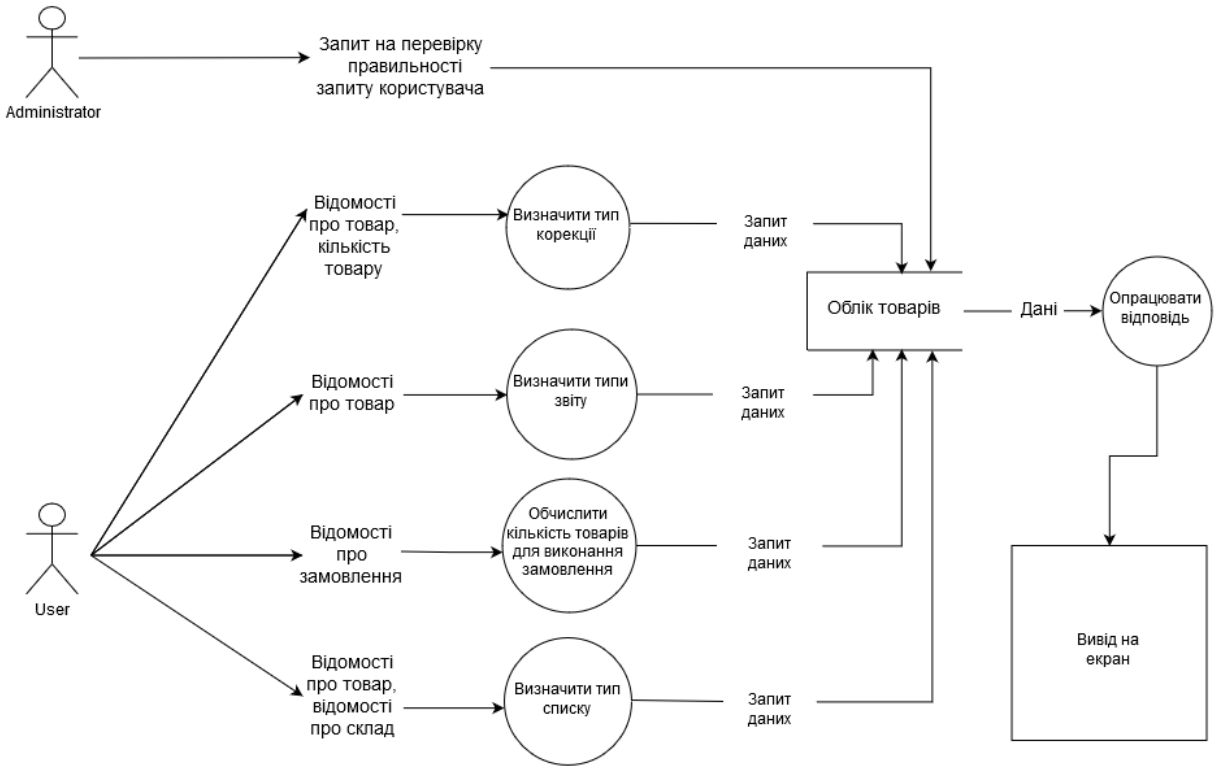


Рисунок 3. Головна діаграма потоків даних

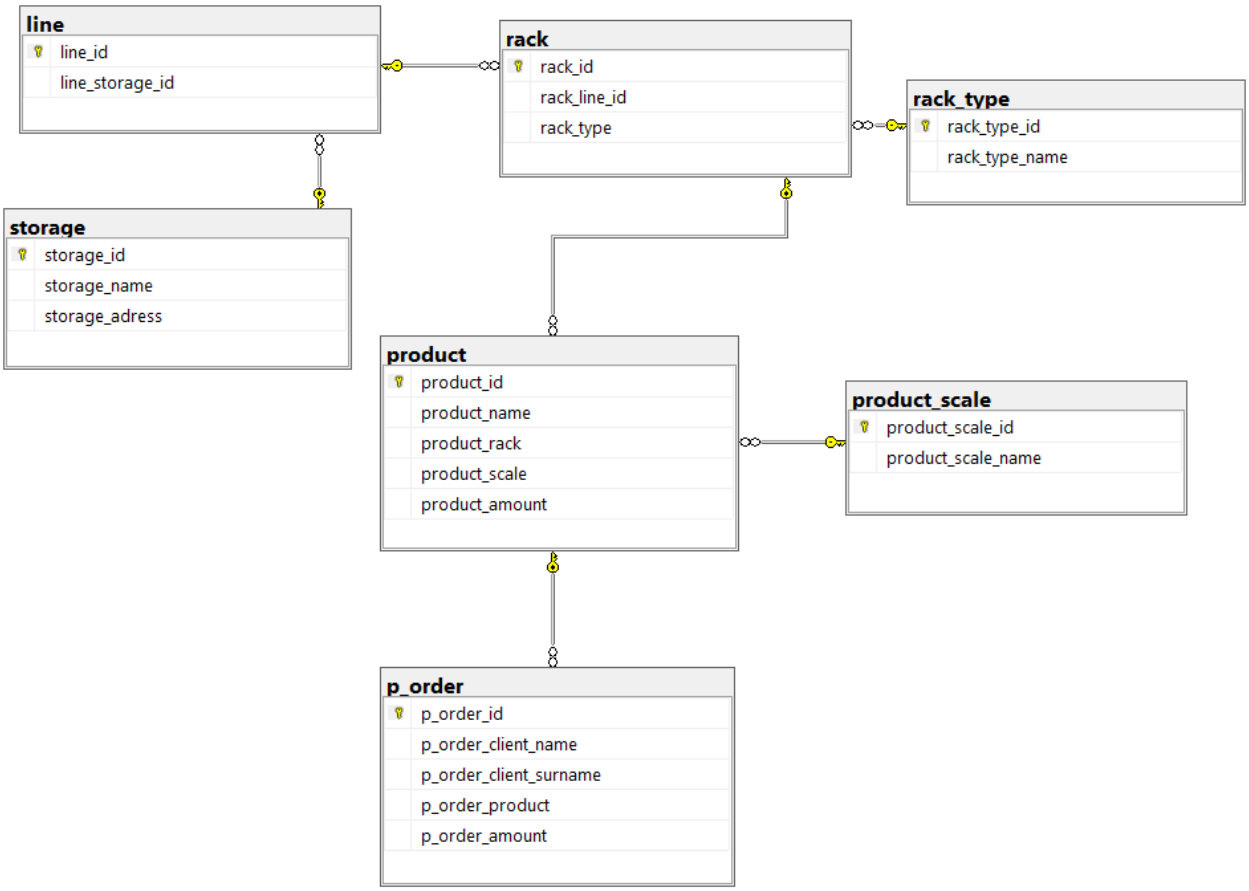
2. Проектування бази данних в MSSQL

```
use rr_var7
go

drop table p_order, product, product_scale, rack, rack_type, line, storage

create table storage (
        storage_id int primary key identity not null,
        storage_name varchar(30) not null,
        storage_adress varchar(40) not null
);

create table line (
        line_id int primary key identity not null,
--      line_type int foreign key references line_type(line_type_id),
        line_storage_id int foreign key references storage(storage_id)
);

create table rack_type (
        rack_type_id int primary key identity not null,
        rack_type_name varchar(40) not null
);

create table rack (
        rack_id int primary key identity not null,
        rack_line_id int foreign key references line(line_id),
        rack_type int foreign key references rack_type(rack_type_id)
);

create table product_scale (
        product_scale_id int primary key identity not null,
        product_scale_name varchar(40) not null,
```

```
--      product_scale_cost int not null
);

create table product (
        product_id int primary key identity not null,
        product_name varchar(40) not null,
        product_rack int foreign key references rack(rack_id),
        product_scale int foreign key references product_scale(product_scale_id),
        product_amount int not null
);

create table p_order (
        p_order_id int primary key identity not null,
        p_order_client_name varchar(40) not null,
        p_order_client_surname varchar(40) not null,
        p_order_product int foreign key references product(product_id),
        p_order_amount int not null
);

use rr_var7
go

------------------------Storage info
insert into     storage (storage_name, storage_adress) values ('first-auto-parts', 'St. Lawrence str.')
insert into     storage (storage_name, storage_adress) values ('second-electricity-components', '12th str.')
------------------------Lines
insert into line (line_storage_id) values (1);
insert into line (line_storage_id) values (1);
insert into line (line_storage_id) values (2);
insert into line (line_storage_id) values (2);
-----------------------Rack-type
insert into rack_type(rack_type_name) values ('car-light-components');
insert into rack_type(rack_type_name) values ('car-engine-components');
insert into rack_type(rack_type_name) values ('electricity-light-components');
insert into rack_type(rack_type_name) values ('electricity-wiring-components');
------------------------Rack
insert into rack(rack_line_id, rack_type) values (1, 1);
insert into rack(rack_line_id, rack_type) values (2, 2);
insert into rack(rack_line_id, rack_type) values (1, 3);
insert into rack(rack_line_id, rack_type) values (3, 1);
insert into rack(rack_line_id, rack_type) values (3, 3);
insert into rack(rack_line_id, rack_type) values (4, 4);
------------------------Product-scale
insert into product_scale(product_scale_name) values ('mililiter');
insert into product_scale(product_scale_name) values ('piece');
insert into product_scale(product_scale_name) values ('meter');
------------------------Product-scale
--- first storage - lines - 1 and 2 - racks: 1 - 3
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A3-2016-R-F-
light', 1, 2, 10);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A3-2016-L-F-
light', 1, 2, 15);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A3-2018-engine-
oil', 2, 1, 100000);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A4-2018-engine-
piston-head', 2, 2, 100);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A4-2018-engine-
spark-plug', 2, 2, 80);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A4-2018-L-turn-
signal', 1, 2, 15);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A4-2018-R-turn-
signal', 1, 2, 20);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Lamps-E16', 3, 2,
100);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Lamps-E12', 3, 2,
150);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A3-2016-LED-
Light', 1, 3, 1050);
--- second storage - lines - 3 and 4 - racks: 4 - 6
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A3-2016-R-F-
light', 4, 2, 20);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A3-2016-L-F-
light', 4, 2, 5);
```

```
insert into product(product_name, product_rack, product_scale, product_amount) values ('Lamps-E16', 5, 2,
200);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Lamps-E12', 5, 2,
350);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Lamps-G20', 5, 2, 50);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Lamps-K1', 5, 2, 150);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Rubber-for-wiring', 6,
1, 100500);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Twisted-pair-Cat45',
6, 3, 10000);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Twisted-pair-Cat72',
6, 3, 1500);
insert into product(product_name, product_rack, product_scale, product_amount) values ('Audi-A3-2016-LED-
Light', 4, 3, 150);
------------------------ IPT over
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Alexey', 'Petrov', 1, 1);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Alexey', 'Petrov', 2, 1);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Max', 'Sundy', 8, 10);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Max', 'Sundy', 14, 4);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Max', 'Sundy', 15, 10);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Sergi', 'Naikha', 3, 1000);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Sergi', 'Naikha', 4, 6);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Sergi', 'Naikha', 5, 6);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Sergi', 'Naikha', 6, 1);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Sergi', 'Naikha', 7, 1);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Sergi', 'Naikha', 20, 5);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Anton', 'Haida', 17, 1000);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Anton', 'Haida', 18, 100);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Anton', 'Haida', 19, 120);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Anton', 'Haida', 3, 100);
insert into p_order(p_order_client_name, p_order_client_surname, p_order_product, p_order_amount) values
('Anton', 'Haida', 3, 1000000);
```

3. ER-діаграма



4. Створення необхідного функціоналу

4.1. Створення впорядкованих списків

А) Товарів за категоріями (по складам);

```
create view list_first
as
	select
		s.storage_name, rt.rack_type_name, p.product_name
	from product p
	inner join
		rack r
		on p.product_rack = r.rack_id
	inner join
		rack_type rt
		on rt.rack_type_id = r.rack_type
	inner join
		line l
		on l.line_id = r.rack_line_id
	inner join
		storage s
		on s.storage_id = l.line_storage_id
	group by
```

```
            s.storage_name, rt.rack_type_name, p.product_name
with check option
go

select * from list_first
go
```

В) Товарів, які є на вибраному складі, але немає на інших складах.

```
create view list_second
as
	select
		p.product_name
	from product p
	inner join
		rack r
		on p.product_rack = r.rack_id
	inner join
		rack_type rt
		on rt.rack_type_id = r.rack_type
	inner join
		line l
		on l.line_id = r.rack_line_id
	inner join
		storage s
		on s.storage_id = l.line_storage_id
	group by
		s.storage_id, p.product_name
	having
		s.storage_id <> 1          -- shows only products that kept on second storage
with check option
go

select * from list_second
go
```

4.2 Обчислення

А) Перевірити, чи достатньо на складі товарів для виконання конкретного замовлення.

```
create proc calculation
(
	@p_order_id int,
	@storage_id int = 1
)
as
	if exists(
		select
			p.product_amount
		from
			p_order po
			inner join
				product p
				on po.p_order_product = p.product_id
			inner join
				rack r
				on r.rack_id = p.product_rack
			inner join
				line l
				on l.line_id = r.rack_line_id
			inner join
				storage s
```

```
                        on s.storage_id = l.line_storage_id
                group by
                        po.p_order_id, p.product_amount, s.storage_id, l.line_id,
r.rack_id
                having
                        po.p_order_id = @p_order_id
                        and s.storage_id = @storage_id
                        and p.product_amount >= (
                        select
                                po.p_order_amount
                        from
                                p_order po
                        where
                                po.p_order_id = @p_order_id
                        )
        )
        begin
                print('On the storage sufficient number of products');
        end
        else
                begin
                        print('On the storage not sufficient number of products for this
order');
                end
go

exec calculation 1, 1
go
```

4.3 Корекція

А) Зміна залишку товару на складі.

```
create proc correction
(
        @product_id int,
        @new_amount int
)
as
        begin;
        update
                product
        set
                product_amount = @new_amount
        where
                product_id = @product_id
        end;
go

exec correction 1, 100
go
```

4.4 Створення звіту

А) "Наявність товарів": Товар - місце зберігання - залишок (з урахуванням одиниці виміру товару).

```
select
        s.storage_name, p.product_name, p.product_amount, ps.product_scale_name
from
        product p
        inner join
```

```
                product_scale ps
                on ps.product_scale_id = p.product_scale
        inner join
                rack r
                on r.rack_id = p.product_rack
        inner join
                line l
                on l.line_id = r.rack_line_id
        inner join
                storage s
                on s.storage_id = l.line_storage_id
group by
        s.storage_name, p.product_name, p.product_amount, ps.product_scale_name
```

B) "Замовлені товари" Товар - місце зберігання - замовлену кількість.

```
select
        s.storage_name, p.product_name, po.p_order_amount, ps.product_scale_name
from
        p_order po
        inner join
                product p
                on p.product_id = po.p_order_product
        inner join
                product_scale ps
                on ps.product_scale_id = p.product_scale
        inner join
                rack r
                on r.rack_id = p.product_rack
        inner join
                line l
                on l.line_id = r.rack_line_id
        inner join
                storage s
                on s.storage_id = l.line_storage_id
group by
        s.storage_name, p.product_name, po.p_order_amount, ps.product_scale_name
```

C) "Вага замовлення": номер замовлення - загальна кількість замовлених товарів.

```
select
        s.storage_name, l.line_id, r.rack_id, rt.rack_type_name, p.product_name,
p.product_amount
from
                product p
                inner join
                        product_scale ps
                        on ps.product_scale_id = p.product_scale
                inner join
                        rack r
                        on r.rack_id = p.product_rack
                inner join
                        rack_type rt
                        on rt.rack_type_id = r.rack_type
                inner join
                        line l
                        on l.line_id = r.rack_line_id
                inner join
```

```
                    storage s
                    on s.storage_id = l.line_storage_id
group by
        s.storage_name,     l.line_id,     r.rack_id,     rt.rack_type_name,     p.product_name,
p.product_amount
```

5. Перевірка нормалізації розробленої моделі

Для цього необхідно перевірити чи належить дана модель до третьої нормальної форми. Для цього модель повинна належати до першої і другої форми, а також не повинна мати транзитивних відношень, тобто залежностей не ключевого атрибуту від іншого не ключового.

Для початку перевіримо належність до першої нормальної форми. Для цього усі атрибути повинні буди атомарні і повинні бути відсутні групи, які повторюються. Виконується.

Далі перевіримо належність до другої нормальної групи. Це буде виконуватись, якщо модель належить до першої нормальної групи (це вже перевірено), а також будуть відсутні неповні функціональні залежності не ключових атрибутів первинного ключа. Виконується.

Залишилось перевірити відсутність транзитивних відношень, тобто ні одне з ключових полів не має ідентифікуватися за допомогою іншого ключового поля. Значить розроблена модель належить до третьої нормальної форми. Виконується.

Висновок

У ході виконання розрахунково-графічної роботи, я дослідив методи проектування баз даних, навчився проектувати власну базу даних на сайті

draw.io, а також створювати функціонал для потреб користувача в MS SQL SERVER.