Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет инфокоммуникационных технологий

**ОТЧЕТ**

**О ЛАБОРАТОРНОЙ РАБОТЕ № 4**

**по теме:** Запросы на выборку данных к БД PostgreSQL.

Представления в PostgreSQL

**по дисциплине:** Проектирование и реализация баз данных

Специальность: 09.03.03 Мобильные и сетевые технологии

Проверил: Говорова М.М. \_\_\_\_\_\_\_\_\_\_

Дата: «29» мая 2021г.   
Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполнил:   
студент группы К3240

Костылев Иван

Санкт-Петербург 2021 г

**Цель работы**

Овладение практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL

**Практическое задание**

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).
2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.
3. Изучить графическое представление запросов.
4. Просмотреть историю запросов.

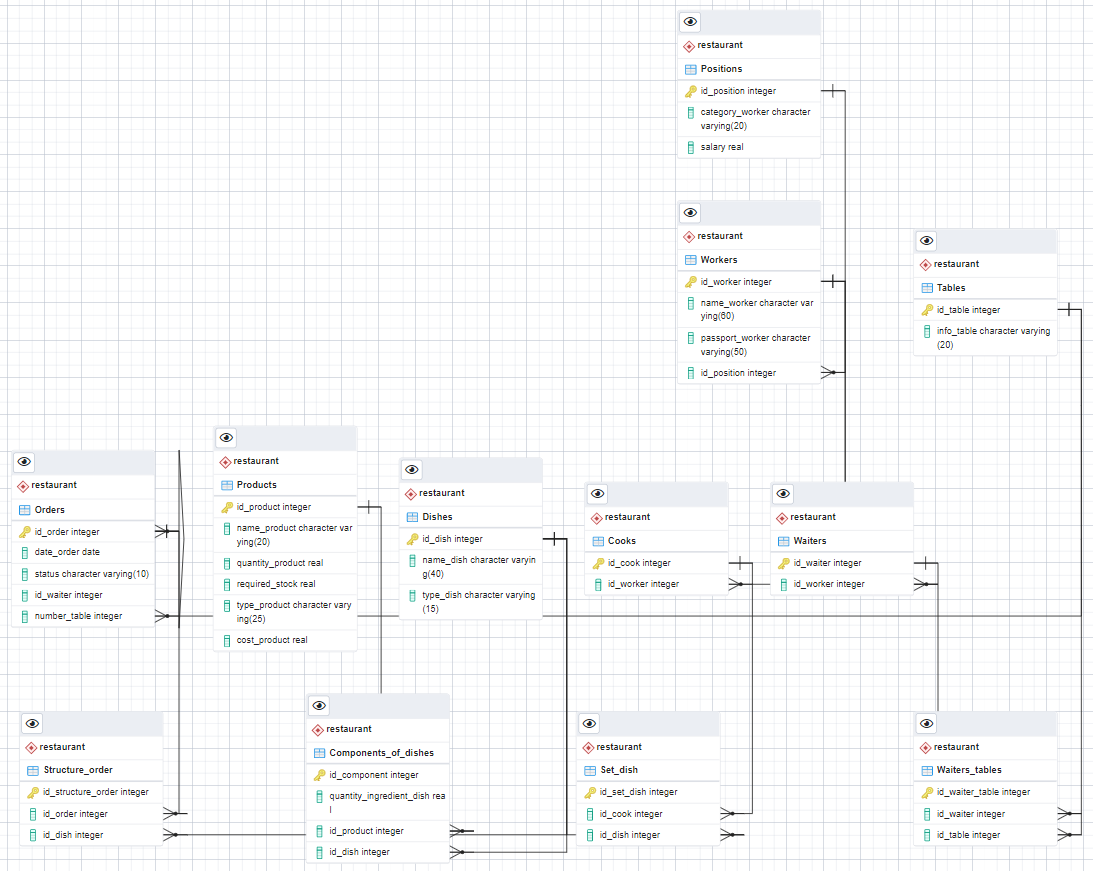
**Ход работы**

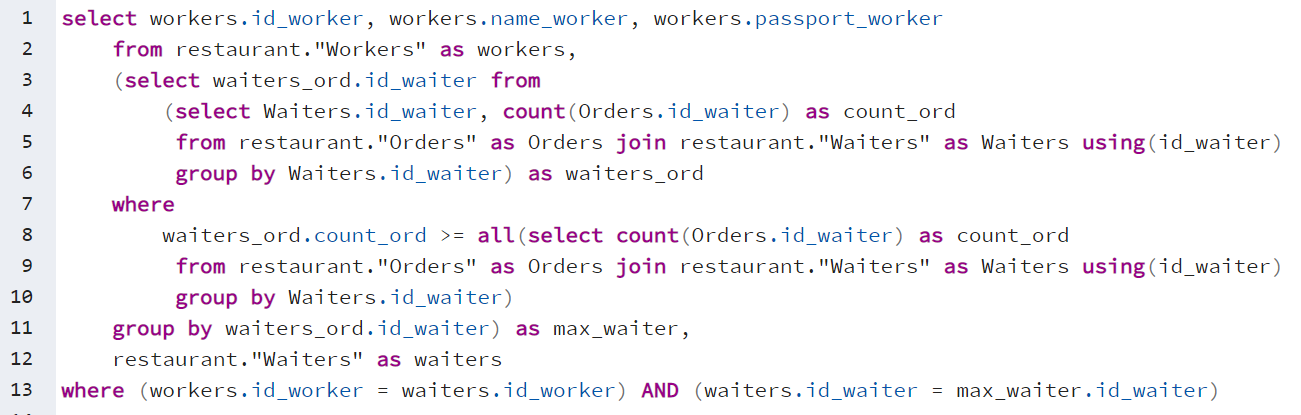
Вариант 13. БД «Ресторан»

Описание предметной области: Сотрудники ресторана – повара и официанты. За каждым официантом закреплены определенные столы. Каждый повар готовит определенный набор блюд. Запас продуктов на складе не должен быть ниже заданного значения. Цена заказа складывается из стоимости ингредиентов и наценки, которая составляет 40% стоимости ингредиентов.

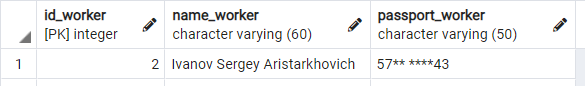
БД должна содержать следующий минимальный набор сведений: ФИО сотрудника. Паспортные данные сотрудника. Категория сотрудника. Должность сотрудника. Оклад сотрудника. Наименование ингредиента. Код ингредиента. Дата закупки. Объем закупки. Количество продукта на складе. Необходимый запас продукта. Срок годности. Цена ингредиента. Поставщик. Наименование блюда. Код блюда. Объем ингредиента. Номер стола. Дата заказа. Код заказа. Количество. Название блюда. Ингредиенты, входящие в блюдо. Тип ингредиента.

1. **Название создаваемой БД:** Restaurant
2. **Cхема инфологической модели**



1. **Скрипты запросов**
   1. Вывести данные официанта, принявшего максимальное число заказов

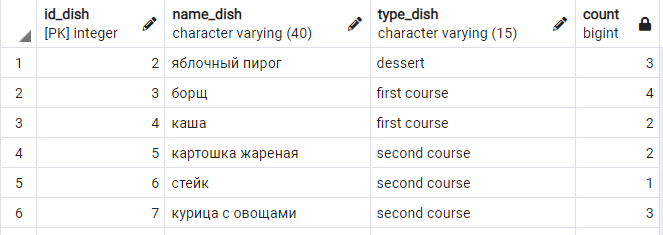
*Результат:*



* 1. Подсчитать, сколько ингредиентов содержит каждое блюдо.

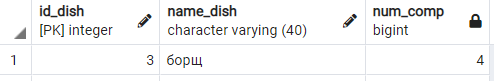
1. SELECT d.id\_dish, d.name\_dish, d.type\_dish, count(comp.id\_component)
2. FROM restaurant."Dishes" AS d,
3. restaurant."Components\_of\_dishes" AS comp
4. WHERE (d.id\_dish = comp.id\_dish)
5. **GROUP BY d.id\_dish**
6. ORDER BY id\_dish;

*Результат:*



* 1. Вывести название блюда, содержащее максимальное число ингредиентов.

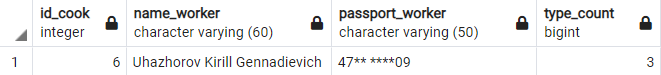
1. SELECT d1.id\_dish, d1.name\_dish, MAX(d1.num\_comp) as num\_comp FROM
2. (SELECT d.id\_dish, d.name\_dish, d.type\_dish, count(comp.id\_component) AS num\_comp
3. FROM restaurant."Dishes" AS d,
4. restaurant."Components\_of\_dishes" AS comp
5. **WHERE (d.id\_dish = comp.id\_dish)**
6. GROUP BY d.id\_dish) AS d1
7. WHERE NOT EXISTS
8. (SELECT \* FROM (
9. SELECT d.id\_dish, d.name\_dish, d.type\_dish, count(comp.id\_component) AS num\_comp
10. FROM restaurant."Dishes" AS d,
11. restaurant."Components\_of\_dishes" AS comp
12. **WHERE (d.id\_dish = comp.id\_dish)**
13. GROUP BY d.id\_dish) AS d2
14. WHERE (d1.num\_comp < d2.num\_comp ))
15. GROUP BY d1.id\_dish, d1.name\_dish;



* 1. Какой повар может приготовить максимальное число видов блюд?

1. SELECT cooks.id\_cook, cooks.name\_worker, cooks.passport\_worker, c1.type\_count FROM
2. (SELECT id\_cook, count(distinct type\_dish) AS type\_count
3. FROM restaurant."Set\_dish" JOIN restaurant."Dishes" USING (id\_dish)
4. GROUP BY id\_cook) AS c1,
5. **(SELECT cooks.id\_worker, cooks.id\_cook, wrk.name\_worker, wrk.passport\_worker**
6. FROM restaurant."Cooks" as cooks,
7. restaurant."Workers" as wrk
8. WHERE (cooks.id\_worker = wrk.id\_worker)) AS cooks
9. WHERE NOT EXISTS
10. **(SELECT \* FROM**
11. (SELECT id\_cook, count(distinct type\_dish) AS type\_count
12. FROM restaurant."Set\_dish" JOIN restaurant."Dishes" USING (id\_dish)
13. GROUP BY id\_cook) AS c2
14. WHERE c1.type\_count < c2.type\_count) AND (cooks.id\_cook = c1.id\_cook);

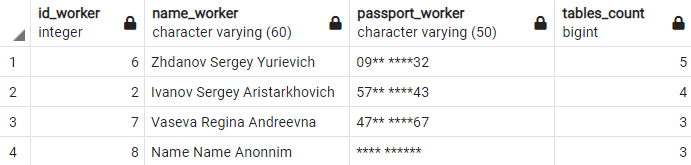
*Результат:*



* 1. Сколько закреплено столов за каждым из официантов?

1. SELECT wrk.id\_worker, wrk.name\_worker, wrk.passport\_worker, wtr\_cnt.tables\_count FROM
2. (SELECT wtr.id\_waiter, count(distinct wtr\_tbl.id\_table) AS tables\_count
3. FROM restaurant."Waiters" AS wtr,
4. restaurant."Waiters\_tables" AS wtr\_tbl
5. **WHERE (wtr.id\_waiter = wtr\_tbl.id\_waiter)**
6. GROUP BY wtr.id\_waiter) AS wtr\_cnt,
7. (restaurant."Waiters" JOIN restaurant."Workers" USING (id\_worker)) AS wrk
8. WHERE wrk.id\_waiter = wtr\_cnt.id\_waiter
9. ORDER BY tables\_count DESC;

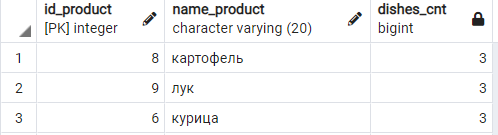
*Результат:*



* 1. Какой из ингредиентов используется в максимальном количестве блюд?

1. SELECT cnt1.id\_product, cnt1.name\_product, cnt1.dishes\_cnt
2. FROM (SELECT prod.id\_product, prod.name\_product, count(comp.id\_dish) AS dishes\_cnt
3. FROM restaurant."Components\_of\_dishes" AS comp,
4. restaurant."Products" AS prod
5. **WHERE (prod.id\_product = comp.id\_product)**
6. GROUP BY prod.id\_product) AS cnt1
7. WHERE NOT EXISTS (SELECT \*
8. FROM (SELECT prod.id\_product, prod.name\_product, count(comp.id\_dish) AS dishes\_cnt
9. FROM restaurant."Components\_of\_dishes" AS comp,
10. **restaurant."Products" AS prod**
11. WHERE (prod.id\_product = comp.id\_product)
12. GROUP BY prod.id\_product) AS cnt2
13. WHERE (cnt1.dishes\_cnt < cnt2.dishes\_cnt));

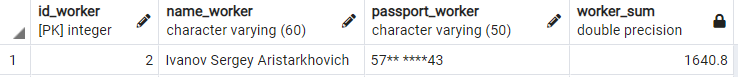
*Результат:*



* 1. Вывести данные официанта, принявшего заказы на максимальную сумму.

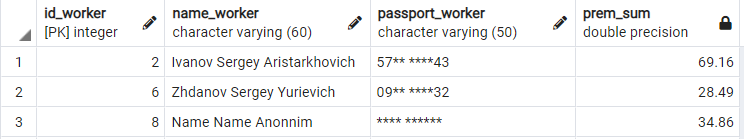
1. **SELECT** data1.id\_worker, data1.name\_worker, data1.passport\_worker, data1.worker\_sum **FROM**
2. (**SELECT** workers.id\_worker, workers.name\_worker, workers.passport\_worker, waiters\_work.sum **AS** worker\_sum **FROM**
3. (**SELECT** waiters.id\_waiter, **SUM**(costs\_ord.sum\_cost\_dish) **FROM**
4. (**SELECT** struct\_ord.id\_order, **SUM**(costs.cost\_dish) **AS** sum\_cost\_dish **FROM**
5. **(SELECT comp.id\_dish, dishes.name\_dish, SUM(comp.quantity\_ingredient\_dish\*prod.cost\_product)\*1.4 AS cost\_dish**
6. **FROM** restaurant."Components\_of\_dishes" **AS** comp,
7. restaurant."Products" **AS** prod,
8. restaurant."Dishes" **AS** dishes
9. **WHERE** comp.id\_dish = dishes.id\_dish **AND** comp.id\_product = prod.id\_product
10. **GROUP BY comp.id\_dish, dishes.name\_dish) AS costs,**
11. restaurant."Structure\_order" **AS** struct\_ord
12. **WHERE** struct\_ord.id\_dish = costs.id\_dish
13. **GROUP** **BY** struct\_ord.id\_order) **AS** costs\_ord,
14. restaurant."Waiters" **AS** waiters,
15. **restaurant."Orders" AS ord**
16. **WHERE** (costs\_ord.id\_order = ord.id\_order **AND** waiters.id\_waiter = ord.id\_waiter)
17. **GROUP** **BY** waiters.id\_waiter) **AS** waiters\_work,
18. (restaurant."Workers" **JOIN** restaurant."Waiters" USING(id\_worker)) **AS** workers
19. **WHERE** workers.id\_waiter = waiters\_work.id\_waiter) **AS** data1
20. **WHERE NOT EXISTS (SELECT \* FROM**
21. (**SELECT** workers.id\_worker, workers.name\_worker, workers.passport\_worker, waiters\_work.sum **AS** worker\_sum **FROM**
22. (**SELECT** waiters.id\_waiter, **SUM**(costs\_ord.sum\_cost\_dish) **FROM**
23. (**SELECT** struct\_ord.id\_order, **SUM**(costs.cost\_dish) **AS** sum\_cost\_dish **FROM**
24. (**SELECT** comp.id\_dish, dishes.name\_dish, **SUM**(comp.quantity\_ingredient\_dish\*prod.cost\_product)\*1.4 **AS** cost\_dish
25. **FROM restaurant."Components\_of\_dishes" AS comp,**
26. restaurant."Products" **AS** prod,
27. restaurant."Dishes" **AS** dishes
28. **WHERE** comp.id\_dish = dishes.id\_dish **AND** comp.id\_product = prod.id\_product
29. **GROUP** **BY** comp.id\_dish, dishes.name\_dish) **AS** costs,
30. **restaurant."Structure\_order" AS struct\_ord**
31. **WHERE** struct\_ord.id\_dish = costs.id\_dish
32. **GROUP** **BY** struct\_ord.id\_order) **AS** costs\_ord,
33. restaurant."Waiters" **AS** waiters,
34. restaurant."Orders" **AS** ord
35. **WHERE (costs\_ord.id\_order = ord.id\_order AND waiters.id\_waiter = ord.id\_waiter)**
36. **GROUP** **BY** waiters.id\_waiter) **AS** waiters\_work,
37. (restaurant."Workers" **JOIN** restaurant."Waiters" USING(id\_worker)) **AS** workers
38. **WHERE** workers.id\_waiter = waiters\_work.id\_waiter) **AS** data2
39. **WHERE** data2.worker\_sum > data1.worker\_sum)

*Результат:*



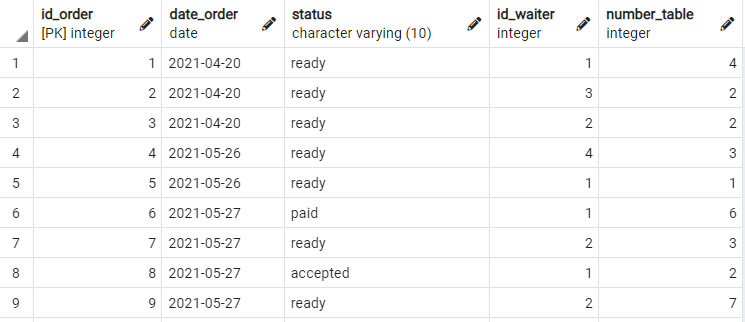
* 1. Рассчитать премию каждого официанта за последние 10 дней (5% от стоимости каждого заказа).

1. SELECT workers.id\_worker, workers.name\_worker, workers.passport\_worker, waiters\_work.sum \* 0.05 AS prem\_sum FROM
2. (SELECT waiters.id\_waiter, waiters.id\_worker, SUM(costs\_ord.sum\_cost\_dish) FROM
3. (SELECT struct\_ord.id\_order, SUM(costs.cost\_dish) AS sum\_cost\_dish FROM
4. (SELECT comp.id\_dish, dishes.name\_dish, SUM(comp.quantity\_ingredient\_dish\*prod.cost\_product)\*1.4 AS cost\_dish
5. **FROM restaurant."Components\_of\_dishes" AS comp,**
6. restaurant."Products" AS prod,
7. restaurant."Dishes" AS dishes
8. WHERE comp.id\_dish = dishes.id\_dish AND comp.id\_product = prod.id\_product
9. GROUP BY comp.id\_dish, dishes.name\_dish) AS costs,
10. **restaurant."Structure\_order" AS struct\_ord**
11. WHERE struct\_ord.id\_dish = costs.id\_dish
12. GROUP BY struct\_ord.id\_order) AS costs\_ord,
13. restaurant."Waiters" AS waiters,
14. restaurant."Orders" AS ord
15. **WHERE (costs\_ord.id\_order = ord.id\_order AND waiters.id\_waiter = ord.id\_waiter AND**
16. (date\_order + INTEGER '10' >= CURRENT\_DATE))
17. GROUP BY waiters.id\_waiter) AS waiters\_work,
18. (restaurant."Workers" JOIN restaurant."Waiters" USING(id\_worker)) AS workers
19. WHERE workers.id\_waiter = waiters\_work.id\_waiter;



1. **Запросы на модификацию данных**
   1. Добавить 3 новых заказа, взяв за основу первые 3 (дату взять сегодняшнюю, а статус задать «принят»)

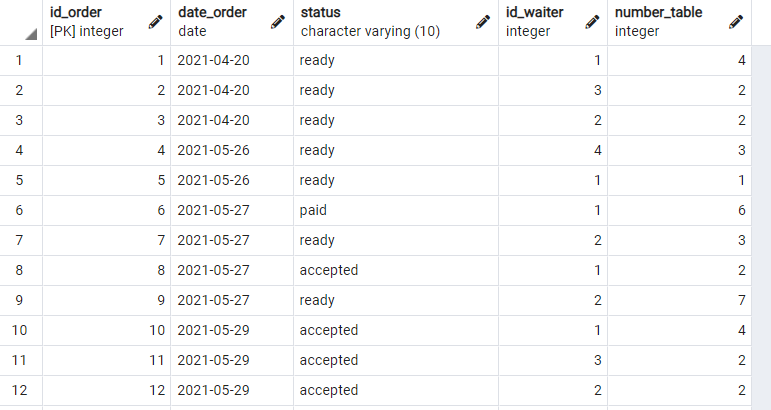
*Содержание таблицы restaurant.”Orders”:*



*Запрос вставки*

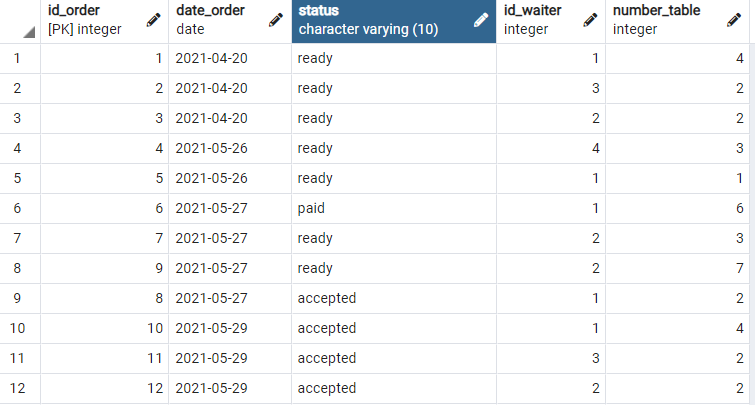
1. **INSERT** **INTO** restaurant."Orders"(
2. date\_order, **STATUS**, id\_waiter, number\_table)
3. (**SELECT** now() **AS** date\_order, 'accepted' **AS** **STATUS**, id\_waiter, number\_table
4. **FROM** restaurant."Orders" **WHERE** id\_order <= 3);

*Содержание таблицы restaurant.”Orders” после вставки:*

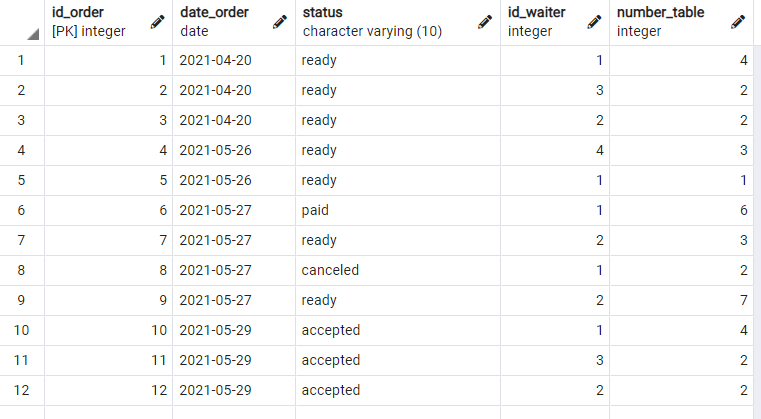


* 1. Предположим, прошло время, наступил следующий день. Тогда нам нужно изменить статус у всех заказов «accepted» на «canceled» (заказ был принят, но так и не был оплачен).

*Содержание таблицы restaurant.”Orders” перед изменением:*

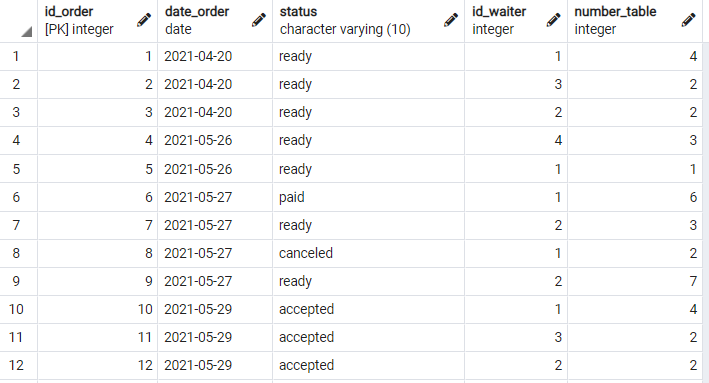


1. **UPDATE** restaurant."Orders"
2. **SET** **STATUS** = 'canceled'
3. **WHERE** **STATUS** = 'accepted' **AND** date\_order < **CURRENT\_DATE**;

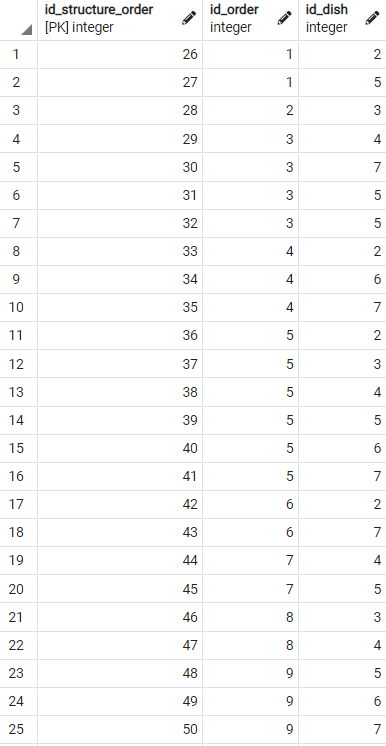


* 1. Владелец ресторана чрезмерно экономный человек, поэтому требует регулярно избавляться от старых данных в таблице заказов. Нужно удалить всю информацию о заказах, которые были ранее, чем 30 дней назад.

*Таблица Orders перед удалением*

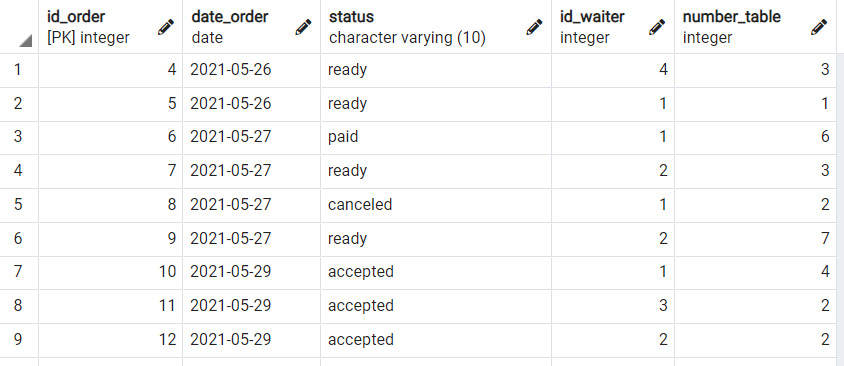
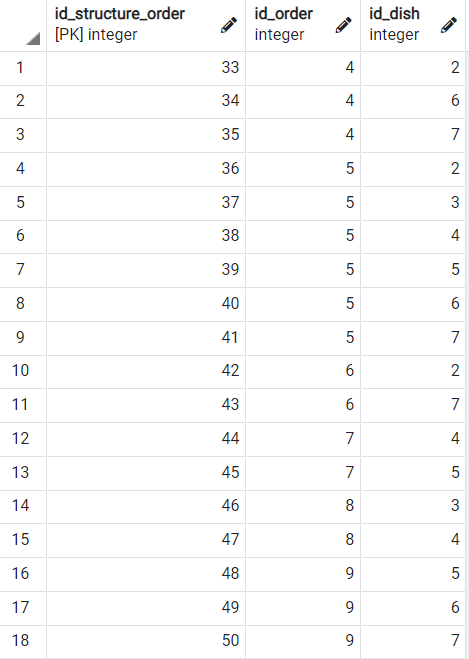


*Таблица Structure\_order перед удалением*

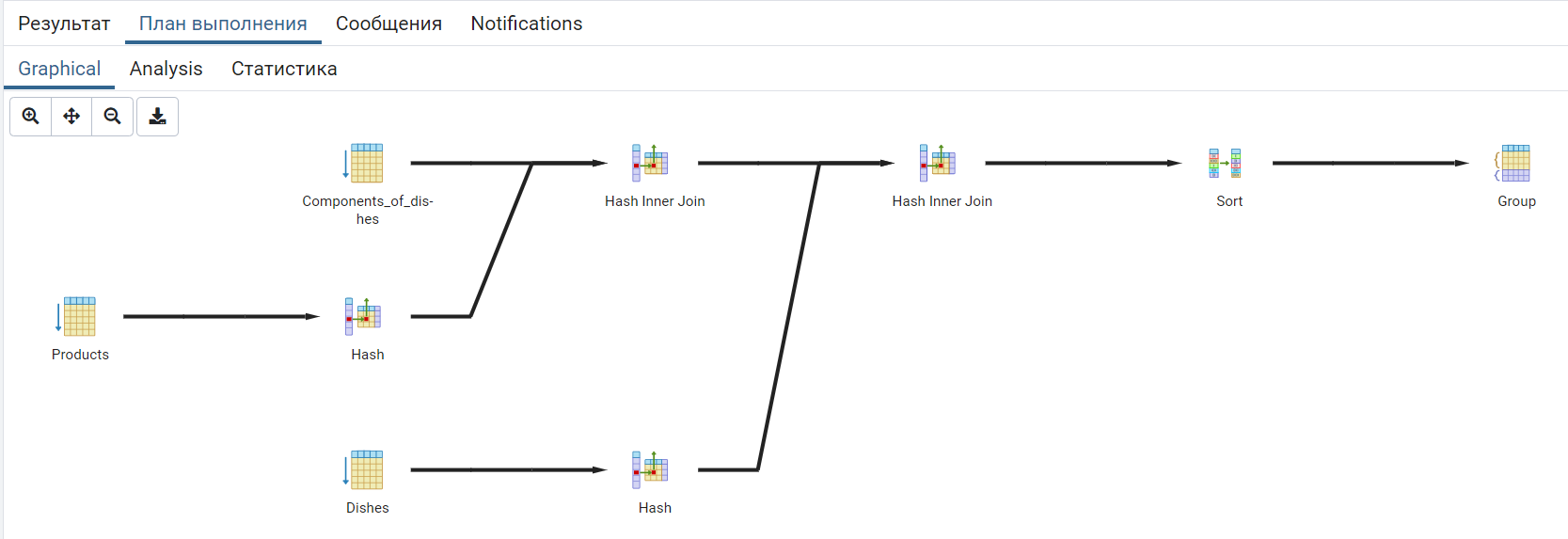


*Скрипт удаления данных:*

1. **DELETE** **FROM** restaurant."Structure\_order" **AS** struct
2. **WHERE** struct.id\_structure\_order **IN** (**SELECT** struct.id\_structure\_order
3. **FROM** restaurant."Orders" **AS** ord,
4. restaurant."Structure\_order" **AS** struct
5. **WHERE ord.date\_order < CURRENT\_DATE - INTEGER '30'**
6. **AND** struct.id\_order = ord.id\_order);
8. **DELETE** **FROM** restaurant."Orders"
9. **WHERE** id\_order **IN** (
10. **SELECT ord.id\_order FROM restaurant."Orders" AS ord**
11. **WHERE** ord.date\_order < **CURRENT\_DATE** - **INTEGER** '30');

*Таблицы Orders и Structure\_order после удаления* 

1. **Создать представления:**
   1. для расчета стоимости ингредиентов для заданного блюда;



1. **SELECT** dishes.name\_dish, prod.name\_product,
2. comp.quantity\_ingredient\_dish, prod.cost\_product, (comp.quantity\_ingredient\_dish\*prod.cost\_product) **AS** cost\_component
3. **FROM** restaurant."Components\_of\_dishes" **AS** comp,
4. restaurant."Products" **AS** prod,
5. **restaurant."Dishes" AS dishes**
6. **WHERE** comp.id\_dish = dishes.id\_dish **AND** comp.id\_product = prod.id\_product
7. **GROUP** **BY** dishes.name\_dish, prod.name\_product,
8. comp.quantity\_ingredient\_dish, prod.cost\_product,(comp.quantity\_ingredient\_dish\*prod.cost\_product)
9. **ORDER** **BY** dishes.name\_dish
   1. количество приготовленных блюд по каждому блюду за определенную дату.
10. **SELECT** **DISTINCT** ord.date\_order, d.name\_dish, **COUNT**(d.id\_dish)
11. **FROM** restaurant."Dishes" **AS** d,
12. (restaurant."Orders" **JOIN** restaurant."Structure\_order" **USING**(id\_order)) **AS** ord
13. **WHERE** ord.id\_dish = d.id\_dish
14. **GROUP BY d.id\_dish, ord.date\_order**
15. **ORDER** **BY** ord.date\_order;

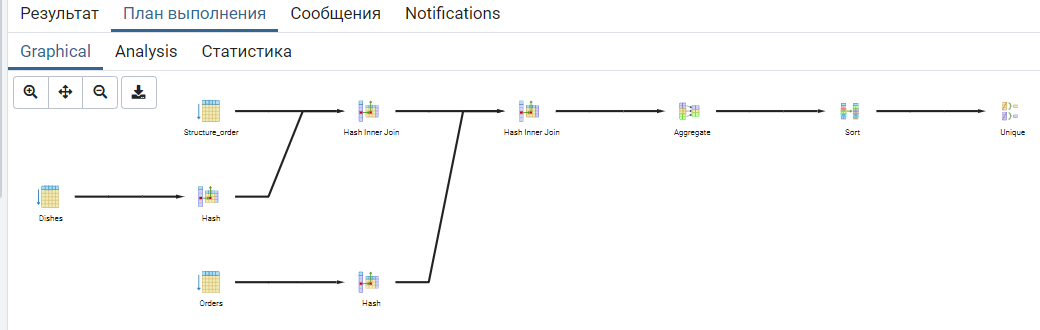


Рисунок 1 – Инфологическая модель данных