

## Мелехин Александр Кс-30 Вариант 9 Лабораторная работа 4

### Данные таблицы для лабораторной работы 4

Таблица salers

	saler_id [PK] integer	saler_name character varying (100)	saler_sex character varying (100)	saler_age integer
1	1	Иванов	Мужской	20
2	2	Петрова	Женский	19
3	3	Сидорова	Женский	21

Таблица brands

	brand_id [PK] integer	brand_name character varying (100)
1	1	Самсунг
2	2	Леново
3	3	Сони

Таблица sales

	sale_id [PK] integer	sale_date date	brand integer	price numeric	sale_count integer	saler integer
1	1	2005-01-03	1	12000	5	1
2	2	2005-01-15	2	8000	4	2
3	3	2005-02-02	1	25000	3	3
4	4	2005-03-02	1	10000	5	1
5	5	2005-02-14	3	11000	3	3
6	6	2005-04-04	1	19000	4	2
7	7	2005-01-07	2	16500	4	[null]
8	8	2005-01-07	[null]	12500	3	2

## Задание 1

**Задание:** придумать запрос на применение агрегирующих оконных функций

**SQL код для задания:** SELECT saler, sale\_date, price, sale\_count, SUM(price \* sale\_count) OVER (PARTITION BY saler ORDER BY sale\_date) AS "выручка продавца" FROM sales;

**Пояснение:** Запрос вычисляет общую выручку для каждого продавца, используя оконную функцию с разбиением по каждому продавцу.

## Результат

	saler integer	sale_date date	price numeric	sale_count integer	выручка продавца numeric
1	1	2005-01-03	12000	5	60000
2	1	2005-03-02	10000	5	110000
3	2	2005-01-07	12500	3	37500
4	2	2005-01-15	8000	4	69500
5	2	2005-04-04	19000	4	145500
6	3	2005-02-02	25000	3	75000
7	3	2005-02-14	11000	3	108000
8	[null]	2005-01-07	16500	4	66000

## Задание 2

**Задание:** придумать запрос на применение ранжирующих оконных функций.

**SQL код для задания:** `SELECT sale_id, brand, price, RANK() OVER (PARTITION BY brand ORDER BY price DESC) AS price_rank FROM sales;`

**Пояснение:** запрос присваивает ранг каждому товару внутри группы марок на основе цены, от самой высокой к самой низкой.

### Результат

	sale_id [PK] integer	brand integer	price numeric	price_rank bigint
1	3	1	25000	1
2	6	1	19000	2
3	1	1	12000	3
4	4	1	10000	4
5	7	2	16500	1
6	2	2	8000	2
7	5	3	11000	1
8	8	[null]	12500	1


### Задание 3

**Задание:** вывести пары атрибутов одной сущности при определенном условии.

**SQL код для задания:** `SELECT DISTINCT LEAST(s1.saler_name, s2.saler_name) AS "Продавец 1", GREATEST(s1.saler_name, s2.saler_name) AS "Продавец 2" FROM salers s1 JOIN salers s2 ON s1.saler_id < s2.saler_id WHERE s1.saler_age > 19 AND s2.saler_age > 19;`

**Пояснение:** запрос выводит пары продавцов с возрастом более 19 лет (*LEAST(s1.saler\_name, s2.saler\_name) AS "Продавец 1" и GREATEST(s1.saler\_name, s2.saler\_name) AS "Продавец 2"* нужны для того, чтобы избежать дубликатов пар, когда продавцы меняются местами или когда пара состоит из двух одинаковых продавцов).

### Результат

	Продавец 1 character varying (100) 	Продавец 2 character varying (100) 
1	Иванов	Сидорова



## Задание 4

**Задание:** тоже, что и в пункте 3 с устранением избыточности без оператора DISTINCT.

**SQL код для задания:** SELECT s1.saler\_name AS saler1, s2.saler\_name AS saler2 FROM salers s1, salers s2 WHERE s1.saler\_id < s2.saler\_id AND s1.saler\_sex <> s2.saler\_sex;

**Пояснение:** в запросе выводятся пары продавцов разного пола.

### Результат

	saler1 character varying (100) 	saler2 character varying (100) 
1	Иванов	Петрова
2	Иванов	Сидорова

## Задание 5

**Задание:** придумать однотабличный запрос, использующий подзапрос в условии отбора строк поле фразы WHERE.

**SQL код для задания:** SELECT saler\_name FROM salers WHERE saler\_age > (SELECT AVG(saler\_age) FROM salers);

**Пояснение:** запрос выбирает продавцов, чей возраст выше среднего.

### Результат

saler_name character varying (100) 🔒	
1	Сидорова

## Задание 6

**Задание:** придумать многотабличный запрос, использующий подзапрос в условии отбора строк поле фразы WHERE.

**SQL код для задания:** SELECT sale\_id, sale\_date, brand, price, sale\_count  
FROM sales WHERE saler IN (SELECT saler\_id FROM salers WHERE saler\_age  
< 21);

**Пояснение:** Подзапрос выбирает ID продавцов моложе 21 года, и эти продажи отображаются.

## Результат

	sale_id [PK] integer	sale_date date	brand integer	price numeric	sale_count integer
1	1	2005-01-03	1	12000	5
2	2	2005-01-15	2	8000	4
3	4	2005-03-02	1	10000	5
4	6	2005-04-04	1	19000	4
5	8	2005-01-07	[null]	12500	3

## Задание 7

**Задание:** придумать запрос, использующий подзапрос с агрегатной функцией в условии отбора строк поле фразы WHERE.

**SQL код для задания:** SELECT sale\_id, sale\_date, brand, price, sale\_count FROM sales WHERE price > (SELECT AVG(price) FROM sales);

**Пояснение:** Запрос выбирает продажи, у которых цена выше средней по всем продажам.

## Результат

	sale_id [PK] integer	sale_date date	brand integer	price numeric	sale_count integer
1	3	2005-02-02	1	25000	3
2	6	2005-04-04	1	19000	4
3	7	2005-01-07	2	16500	4




## Задание 8

**Задание:** придумать запрос на использование подзапросов, которые выдают много строк с помощью оператора IN.

**SQL код для задания:** SELECT saler\_name FROM salers WHERE saler\_id IN (SELECT saler FROM sales WHERE brand IN (1, 2));

**Пояснение:** Запрос ищет продавцов, которые продали товары марок "Самсунг" или "Леново".

### Результат

	saler_name character varying (100) 
1	Иванов
2	Петрова
3	Сидорова

## Задание 9

**Задание:** придумать запрос, использующий подзапрос в предложении HAVING.

**SQL код для задания:** SELECT saler, SUM(price \* sale\_count) AS total\_sales  
FROM sales GROUP BY saler HAVING SUM(price \* sale\_count) >= 110000;

**Пояснение:** запрос, выполняющий группировку по продавцу с ограничением по общей сумме продаж, превышающей 110000.

### Результат

	saler integer 	total_sales numeric 
1	2	145500
2	1	110000

## Задание 10

**Задание:** придумать запрос, использующий подзапрос в предложении FROM.

**SQL код для задания:** SELECT brand\_name, ROUND(AVG(price), 0) AS avg\_price FROM (SELECT brand, price FROM sales) AS sub JOIN brands ON brands.brand\_id = sub.brand GROUP BY brand\_name;

**Пояснение:** используется подзапрос в *FROM*, чтобы получить среднюю цену для каждой марки.

### Результат

	brand_name character varying (100) 🔒	avg_price numeric 🔒
1	Леново	12250
2	Сони	11000
3	Самсунг	16500


## Задание 11

**Задание:** придумать запрос на использование соотнесенного подзапроса, который выдает много строк с помощью оператора IN

**SQL код для задания:** `SELECT saler_name FROM salers WHERE salers.saler_id IN (SELECT saler FROM sales WHERE price > (SELECT AVG(price) FROM sales AS inner_sales WHERE inner_sales.brand = sales.brand));`

**Пояснение:** подзапрос находит продавцов, которые продали товары дороже средней по конкретной марке.

### Результат

	saler_name character varying (100) 
1	Сидорова
2	Петрова


## Задание 12

**Задание:** придумать запрос на сравнение таблицы с собой.

**SQL код для задания:** `SELECT s1.saler_name FROM salers s1 LEFT JOIN salers s2 ON s1.saler_age = s2.saler_age AND s1.saler_id <> s2.saler_id WHERE s2.saler_id IS NULL;`

**Пояснение:** сравнение таблицы *salers* с самой собой для поиска продавцов, возраст которых уникален.

### Результат

	saler_name character varying (100) 
1	Иванов
2	Петрова
3	Сидорова