Продажи магазина велосипедов

-- BEGINNER QUESTIONS --

1) Список всех продуктов: напишите запрос для получения всех названий продуктов и соответствующих им названий брендов

```
SELECT product_name, brand_name
FROM products p
INNER JOIN brands b
    ON p.brand_id = b.brand_id
ORDER BY brand_name;
```

product_name	brand_name
Electra Townie Original 21D - 2016	Electra
Electra Cruiser 1 (24-Inch) - 2016	Electra
Electra Girl's Hawaii 1 (16-inch) - 2015/2016	Electra
Electra Moto 1 - 2016	Electra
Electra Townie Original 7D EQ - 2016	Electra
Electra Townie Original 7D EQ - Women's - 2016	Electra
Electra Cruiser 1 (24-Inch) - 2016	Electra

2) Поиск активного персонала: напишите запрос для поиска всех активных сотрудников и названий их магазинов

```
SELECT first_name, last_name, store_name
FROM staffs sf
INNER JOIN stores st
    ON sf.store_id = st.store_id
WHERE active = TRUE;
```

first_name	last_name	store_name
Fabiola	Jackson	Santa Cruz Bikes
Mireya	Copeland	Santa Cruz Bikes
Genna	Serrano	Santa Cruz Bikes
Virgie	Wiggins	Santa Cruz Bikes
Jannette	David	Baldwin Bikes
Marcelene	Boyer	Baldwin Bikes
Venita	Daniel	Baldwin Bikes
Kali	Vargas	Rowlett Bikes
Layla	Terrell	Rowlett Bikes
Bernardine	Houston	Rowlett Bikes

3) Сведения о клиенте: напишите запрос для списка всех клиентов с их полными именами, адресами электронной почты и номерами телефонов

```
SELECT first_name, last_name, email, phone
FROM customers;
```

first_name	last_name	email	phone
Debra	Burks	debra.burks@yahoo.com	NULL
Kasha	Todd	kasha.todd@yahoo.com	NULL
Tameka	Fisher	tameka.fisher@aol.com	NULL
Daryl	Spence	daryl.spence@aol.com	NULL
Charolette	Rice	charolette.rice@msn.com	(916) 381-6003
Lyndsey	Bean	lyndsey.bean@hotmail.com	NULL
Latasha	Hays	latasha.hays@hotmail.com	(716) 986-3359
Jacquline	Duncan	jacquline.duncan@yahoo.com	NULL

4) Категории продуктов: напишите запрос для подсчета количества продуктов в каждой категории

```
SELECT category_name,
        COUNT(pr.category_id) AS quantity
FROM products pr
INNER JOIN categories ct
    ON pr.category_id = ct.category_id
GROUP BY pr.category_id
ORDER BY quantity DESC;
```

category_name	quantity
Cruisers Bicycles	78
Mountain Bikes	60
Road Bikes	60
Children Bicycles	59
Comfort Bicycles	30
Electric Bikes	24
Cyclocross Bicycles	10

5) Заказы по клиенту: напишите запрос для подсчета общего количества заказов, размещенных каждым клиентом

```
SELECT last_name, first_name, COUNT(*) AS quantity
FROM orders o
INNER JOIN customers cs
    ON o.customer_id = cs.customer_id
GROUP BY o.customer_id
ORDER BY last_name, first_name;
```

	last_name	first_name	quantity
▶	Acevedo	Ester	1
	Acevedo	Jamika	1
	Acevedo	Penny	1
	Acosta	Bettyann	1
	Acosta	Shery	1
	Adams	Corinna	1
	Adkins	Phylis	1
	Aguilar	Elinore	3
	Aguirre	Janetta	2

-- INTERMEDIATE QUESTIONS --

1) Общий объем продаж по продукту: напишите запрос для расчета общей суммы продаж для каждого продукта (с учетом количества, цены по прейскуранту и скидки)

```
SELECT product_name,
        ROUND(SUM(quantity * (oi.list_price * (1 - discount))), 2) AS
sales_revenue
FROM products pr
INNER JOIN order_items oi
    ON pr.product_id = oi.product_id
GROUP BY pr.product_id
ORDER BY sales_revenue DESC;
```

	product_name	sales_revenue
▶	Trek Slash 8 27.5 - 2016	555558.61
	Trek Conduit+ - 2016	389248.70
	Trek Fuel EX 8 29 - 2016	368472.73
	Surly Straggler 650b - 2016	226765.55
	Trek Domane SLR 6 Disc - 2017	211584.62
	Surly Straggler - 2016	203507.62
	Trek Remedy 29 Carbon Frameset - 2016	203380.87
	Trek Powerfly 8 FS Plus - 2017	188249.62

2) Заказы по статусу: напишите запрос для подсчета количества заказов для каждого статуса заказа

```
SELECT order_status,
        COUNT(order_id) AS num_of_order
FROM orders
GROUP BY order_status
ORDER BY order_status;
```

	order_status	num_of_order
▶	1	62
	2	63
	3	45
	4	1445

3) Заказы клиентов: напишите запрос для списка всех клиентов, которые разместили хотя бы один заказ, включая их полное имя и общее количество заказов

```
SELECT CONCAT(last_name, ' ', first_name) AS full_name,
        COUNT(order_id) AS num_of_orders
FROM orders o
INNER JOIN customers cs
    ON o.customer_id = cs.customer_id
GROUP BY o.customer_id
ORDER BY num_of_orders DESC, full_name;
```

	full_name	num_of_orders
▶	Aguilar Elinore	3
	Albert Jamaal	3
	Baldwin Genoveva	3
	Bean Lyndsey	3
	Becker Lorrie	3
	Berg Monika	3
	Booth Cleotilde	3
	Branch Linnie	3

4) Наличие на складе: напишите запрос для поиска общего количества каждого продукта, доступного во всех магазинах

```
SELECT product_name,
        SUM(quantity) AS total_quantity
FROM products pr
INNER JOIN stocks st
    ON pr.product_id = st.product_id
GROUP BY st.product_id
ORDER BY total_quantity DESC;
```

product_name	total_quantity
Trek XM700+ Lowstep - 2018	86
Electra Townie Original 7D - 2017	82
Sun Bicycles Cruz 7 - Women's - 2017	79
Trek Verve+ - 2018	79
Trek Powerfly 8 FS Plus - 2017	78
Trek Domane AL 2 Women's - 2018	77
Trek Domane SL 5 Disc - 2018	77
Trek Fuel EX 8 29 XT - 2018	75

5) Доход по магазину: напишите запрос для расчета общего дохода, полученного каждым магазином

```
SELECT store_name,
        ROUND(SUM(quantity * list_price * (1 - discount)), 2) AS
total_store_revenue
FROM order_items oi
INNER JOIN orders o
    ON oi.order_id = o.order_id
INNER JOIN stores st
    ON st.store_id = o.store_id
GROUP BY o.store_id
ORDER BY total_store_revenue DESC;
```

store_name	total_store_revenue
Baldwin Bikes	5215751.28
Santa Cruz Bikes	1605823.04
Rowlett Bikes	867542.24

-- ADVANCED QUESTIONS --

1) Анализ ежемесячных продаж: напишите запрос для расчета общей суммы продаж за каждый месяц

```
SELECT YEAR(order_date) AS year,
        MONTH(order_date) AS month,
        ROUND(SUM(quantity * list_price * (1 - discount)), 2) AS total_of_month
FROM order_items oi
INNER JOIN orders o
    ON oi.order_id = o.order_id
GROUP BY YEAR(order_date), MONTH(order_date);
```

	year	month	total_of_month
▶	2016	1	215146.42
	2016	2	156112.32
	2016	3	180600.33
	2016	4	167144.05
	2016	5	205270.01
	2016	6	210562.12
	2016	7	199556.81

2) Лучшие клиенты: напишите запрос для поиска 5 лучших клиентов, которые потратили больше всего денег

```
SELECT CONCAT(first_name, ' ', last_name) AS full_name,
        ROUND(SUM(quantity * list_price * (1 - discount)), 2) AS spent
FROM order_items oi
INNER JOIN orders o
    ON oi.order_id = o.order_id
INNER JOIN customers cs
    ON o.customer_id = cs.customer_id
GROUP BY cs.customer_id
ORDER BY spent DESC
LIMIT 5;
```

	full_name	spent
▶	Sharyn Hopkins	34807.94
	Pamelia Newman	33634.26
	Abby Gamble	32803.01
	Lyndsey Bean	32675.07
	Emmitt Sanchez	31925.89

3) Иерархия сотрудников: напишите запрос для перечисления всех сотрудников вместе с именами их менеджеров

```
SELECT e.staff_id,
        CONCAT(e.first_name, ' ', e.last_name) AS emploee,
        CONCAT(m.first_name, ' ', m.last_name) AS manager
FROM staffs e
LEFT JOIN staffs m
    ON m.staff_id = e.manager_id;
```

staff_id	emploee	manager
1	Fabiola Jackson	NULL
2	Mireya Copeland	Fabiola Jackson
3	Genna Serrano	Mireya Copeland
4	Virgie Wiggins	Mireya Copeland
5	Jannette David	Fabiola Jackson
6	Marcelene Boyer	Jannette David
7	Venita Daniel	Jannette David
8	Kali Vargas	Fabiola Jackson
9	Layla Terrell	Venita Daniel
10	Bernardine Houston	Venita Daniel

4) Эффективность продукта: напишите запрос для определения того, какие продукты имеют самый высокий объем продаж в 2018 году

```
SELECT products.product_id,
        product_name,
        ROUND(SUM(quantity * oi.list_price * (1 - discount)), 2) AS unit_sold
FROM products pr
INNER JOIN order_items oi
    ON pr.product_id = oi.product_id
INNER JOIN orders o
    ON oi.order_id = o.order_id
WHERE YEAR(order_date) = '2018'
GROUP BY product_id
ORDER BY unit_sold DESC
LIMIT 10;
```

product_id	product_name	unit_sold
155	Trek Domane SLR 9 Disc - 2018	54359.95
205	Trek Super Commuter+ 8S - 2018	37299.93
203	Trek Powerfly 7 FS - 2018	36499.93
43	Trek Fuel EX 9.8 27.5 Plus - 2017	33866.94
61	Trek Powerfly 8 FS Plus - 2017	32949.93
153	Trek Domane SL 7 Women's - 2018	31399.94
252	Electra Townie Commute Go! Ladies' - 2018	29849.90
142	Trek Fuel EX 8 29 XT - 2018	29503.91
207	Trek Boone 7 Disc - 2018	28239.93
58	Trek Madone 9.2 - 2017	27599.94

5) Анализ местоположения клиентов: напишите запрос для подсчета количества клиентов, общего объема продаж, средней стоимости заказа и максимальной стоимости заказа для клиентов в каждом городе

```
SELECT city,
        COUNT(cs.customer_id) AS customer_count,
```

```
        ROUND(SUM(quantity * oi.list_price * (1 - discount)), 2) AS sales_volume,
        ROUND(AVG(quantity * oi.list_price * (1 - discount)), 2) AS avg_order,
        ROUND(MAX(quantity * oi.list_price * (1 - discount)), 2) AS max_order
FROM customers cs
INNER JOIN orders o
    ON cs.customer_id = o.customer_id
INNER JOIN order_items oi
    ON o.order_id = oi.order_id
GROUP BY city
ORDER BY sales_volume DESC;
```

city	customer_count	sales_volume	avg_order	max_order
Mount Vernon	60	105563.33	1759.39	9499.98
Ballston Spa	52	98619.75	1896.53	11159.98
San Angelo	53	98429.26	1857.16	10449.98
Baldwinsville	37	96375.67	2604.75	12089.98
Howard Beach	36	95328.99	2648.03	21599.98
Orchard Park	48	90920.47	1894.18	10229.98
Canyon Country	44	86520.53	1966.38	7599.98
Monroe	44	84076.36	1910.83	21599.98
Houston	44	81021.73	1841.40	9499.98
Astoria	32	79823.88	2494.50	12349.98

---------- ОКОННЫЕ ФУНКЦИИ ----------

1) Ранг товаров: напишите запрос для распределения самых дорогих товаров к самым дешевым

```
SELECT product_name,
        list_price,
        DENSE_RANK() OVER (ORDER BY list_price DESC) AS product_dense_rank
FROM products;
```

product_name	list_price	product_dense_rank
Trek Domane SLR 9 Disc - 2018	11999.99	1
Trek Domane SLR 8 Disc - 2018	7499.99	2
Trek Silque SLR 8 Women's - 2017	6499.99	3
Trek Domane SL Frameset - 2018	6499.99	3
Trek Domane SL Frameset Women's - 2018	6499.99	3
Trek Emonda SLR 8 - 2018	6499.99	3
Trek Silque SLR 7 Women's - 2017	5999.99	4

2) Количество заказов в день: напишите запрос для определения общего числа заказов по дням без учета отмененных заказов, сформируйте накопительную сумму заказов

```
WITH subq AS (
    SELECT order_date,
            COUNT(order_id) AS orders_count
    FROM orders
    GROUP BY order_date
)
SELECT order_date,
        orders_count,
        SUM(orders_count) OVER (ORDER BY order_date) AS orders_cum_count
FROM subq;
```

order_date	orders_count	orders_cum_count
2016-01-01	2	2
2016-01-02	1	3
2016-01-03	2	5
2016-01-04	3	8
2016-01-05	3	11
2016-01-06	1	12
2016-01-08	1	13
2016-01-09	2	15
2016-01-12	2	17
2016-01-14	3	20

3) Нумерование заказов пользователей: напишите запрос для определения порядкового номера каждого заказа для каждого покупателя

```
SELECT customer_id,
        order_id,
        order_date,
        ROW_NUMBER() OVER (PARTITION BY customer_id ORDER BY order_date) AS order_number
FROM orders
WHERE order_id NOT IN (
    SELECT order_id
    FROM orders
    WHERE order_status = 3
);
```

customer_id	order_id	order_date	order_number
1	599	2016-12-09	1
1	1555	2018-04-18	2
2	1084	2017-08-21	1
2	1509	2018-04-09	2
3	1468	2018-03-27	1
3	1496	2018-04-06	2
4	700	2017-02-07	1
4	1556	2018-04-18	2
5	571	2016-11-24	1

4) Время между заказами: напишите запрос для определения прошедшего времени с момента предыдущего заказа для каждого пользователя без учета отмеенных заказов

```
WITH subq AS (
    SELECT customer_id,
            order_id,
            order_date,
            LAG(order_date, 1) OVER (PARTITION BY customer_id ORDER BY order_date) AS time_lag
     FROM orders
     WHERE order_id NOT IN (
          SELECT order_id
          FROM orders
          WHERE order_status = 3
    )
)

SELECT customer_id,
          order_id,
          order_date,
          time_lag,
          DATEDIFF(order_date, time_lag) AS time_diff_days
FROM subq;
```

customer_id	order_id	order_date	time_lag	time_diff_days
1	599	2016-12-09	NULL	NULL
1	1555	2018-04-18	2016-12-09	495
2	1084	2017-08-21	NULL	NULL
2	1509	2018-04-09	2017-08-21	231
3	1468	2018-03-27	NULL	NULL
3	1496	2018-04-06	2018-03-27	10
4	700	2017-02-07	NULL	NULL
4	1556	2018-04-18	2017-02-07	435
5	571	2016-11-24	NULL	NULL
5	1544	2018-04-17	2016-11-24	509

5) Время между заказами: рассчитайте среднее время между заказами для каждого пользователя, у которого более одного заказа

```
WITH subq1 AS (
    SELECT customer_id,
            order_id,
            order_date,
            LAG(order_date, 1) OVER (PARTITION BY customer_id ORDER BY order_date) AS time_lag
    FROM orders
    WHERE order_id NOT IN (
        SELECT order_id
        FROM orders
        WHERE order_status = 3
    )
),
subq2 AS (
    SELECT customer_id,
            order_id,
            order_date,
            time_lag,
            DATEDIFF(order_date, time_lag) AS time_diff_days
    FROM subq1
)

SELECT customer_id,
            AVG(time_diff_days) AS days_between_orders
```

```
FROM subq2
GROUP BY customer_id
HAVING COUNT(order_id) > 1;
```

customer_id	days_between_orders
1	495.00
2	231.00
3	10.00
4	435.00
5	509.00
6	256.00
7	771.00
8	540.00

6) Продуктивные продавцы: напишите запрос для определения продавцов, которые осуществили больше заказов, чем среднее значение продаж

```
WITH subq1 AS (
    SELECT DISTINCT staff_id,
            COUNT(order_id) OVER (PARTITION BY staff_id) AS staff_orders
    FROM orders
),
subq2 AS (
      SELECT staff_id,
              staff_orders,
              ROUND(AVG(staff_orders) OVER (), 2) AS avg_orders_staff
      FROM subq1
)

SELECT staff_id,
          staff_orders,
          avg_orders_staff,
    CASE
          WHEN staff_orders > avg_orders_staff
          THEN 'YES'
          ELSE 'NO'
    END AS is_above_avg
FROM subq2;
```

	staff_id	staff_orders	avg_orders_staff	is_above_avg
▶	2	164	269.17	NO
	3	184	269.17	NO
	6	553	269.17	YES
	7	540	269.17	YES
	8	88	269.17	NO
	9	86	269.17	NO

7) Первые и повторные заказы: напишите запрос для определения числа первых и повторных для каждого дня

```
WITH subq  AS(
    SELECT customer_id,
                order_date,
                order_id,
                CASE
                    WHEN order_date = MIN(order_date) OVER (PARTITION BY
customer_id)
                    THEN 'Первый'
                    ELSE 'Повторный'
                END AS order_type
    FROM orders
    WHERE order_id NOT IN (
            SELECT order_id
            FROM orders
            WHERE order_status = 3
    )
)

SELECT order_date,
            order_type,
            COUNT(order_id) AS orders_count
FROM subq
GROUP BY order_date, order_type
ORDER BY order_date, order_type;
```

order_date	order_type	orders_count
2016-01-01	Первый	2
2016-01-02	Первый	1
2016-01-03	Первый	2
2016-01-04	Первый	3
2016-01-05	Первый	3
2016-01-06	Первый	1
2016-01-08	Первый	1
2016-01-09	Первый	2
2016-01-12	Первый	2
2016-01-14	Первый	3