WsCube Tech

JENSON USA

PRESENTED BY - BHUVAN JARI

ABOUT JENSON COMPANY

JENSON USA IS A LEADING ONLINE RETAILER OF BICYCLES, BIKE PARTS, APPAREL, AND ACCESSORIES. IT HAS BEEN SERVING THE CYCLING COMMUNITY SINCE 1994. THE COMPANY'S MISSION IS TO INSPIRE PEOPLE TO "RIDE, EXPERIENCE, AND EXPLORE." IT SELLS PRODUCTS FOR ROAD BIKES, MOUNTAIN BIKES, TRIATHLONS, BMX, GRAVEL, AND COMMUTER BIKES. IT HAS RETAIL LOCATIONS IN CORONA AND RIVERSIDE, CALIFORNIA.
JENSON USA IS INVOLVED IN SEVERAL COMMUNITY INITIATIVES, INCLUDING:
COMMUTE FOR KIDS: PROVIDES BIKES TO ELEMENTARY SCHOOL KIDS IN UNDERSERVED AREAS.
WORLD BICYCLE RELIEF: DONATES BICYCLES WITH EMPLOYEES AND CUSTOMERS ON GIVING TUESDAY.
IMBA: PROVIDES INPUT INTO MOUNTAIN BIKE INITIATIVES.
LOCAL MONTHLY RIDES: HOSTS MONTHLY RIDES FROM THEIR RIVERSIDE AND CORONA STORES.

WE'VE ALL EXPERIENCED THE BEAUTY OF THE TRAIL OR THE FREEDOM OF THE OPEN ROAD. FOR MANY, CYCLING PRESENTS A WORLD OF POSSIBILITIES AND OPPORTUNITIES AT EVERY TURN AT JENSON USA, OUR PASSION FOR CYCLING SPROUTED BACK IN 1994 AND HAS SINCE TAKEN ROOT AND GROWN INTO A COMMUNITY OF PEOPLE DEVOTED TO LIVING LIFE TO ITS FULLEST WE'RE BUILDING A CULTURE OF PEOPLE WHO RESPECT EACH OTHER, SET THE PACE, AND LEAD THROUGH SERVICE. WE STRIVE DAILY TO BETTER SERVE OUR CUSTOMERS, OUR FELLOW EMPLOYEES, AND OUR COMMUNITY. OVER THE YEARS WE'VE FOUND THAT HAPPY PEOPLE ARE HARDWORKING PEOPLE, AND THAT THE TIME WE SPEND ON TWO WHEELS TOGETHER BUILDS THE GREATEST PROFIT OF ALL

WE ALL APPROACH LIFE WITH DIFFERENT STORIES AND SPECIAL EXPERIENCES THAT HAVE SHAPED US INTO WHO WE ARE AS INDIVIDUALS. DEEP DOWN, IT'S A LOVE OF ADVENTURE THAT BRINGS US TOGETHER, INSPIRES US TO PUSH THE LIMITS OF WHAT WE ARE CAPABLE OF, AND DAILY REDISCOVER THE FREEDOM FOUND WHEN WE FIRST STARTED PEDALING.

QUESTIONS???

1. FIND THE TOTAL NUMBER OF PRODUCTS SOLD BY EACH STORE ALONG WITH THE STORE NAME.
2. CALCULATE THE CUMULATIVE SUM OF QUANTITIES SOLD FOR EACH PRODUCT OVER TIME.
3. FIND THE PRODUCT WITH THE HIGHEST TOTAL SALES (QUANTITY * PRICE) FOR EACH CATEGORY.
4. FIND THE CUSTOMER WHO SPENT THE MOST MONEY ON ORDERS.
5. FIND THE HIGHEST-PRICED PRODUCT FOR EACH CATEGORY NAME.
6. FIND THE TOTAL NUMBER OF ORDERS PLACED BY EACH CUSTOMER PER STORE.
7. FIND THE NAMES OF STAFF MEMBERS WHO HAVE NOT MADE ANY SALES.
8. FIND THE TOP 3 MOST SOLD PRODUCTS IN TERMS OF QUANTITY.
9. FIND THE MEDIAN VALUE OF THE PRICE LIST.
10. LIST ALL PRODUCTS THAT HAVE NEVER BEEN ORDERED.(USE EXISTS)
11. LIST THE NAMES OF STAFF MEMBERS WHO HAVE MADE MORE SALES THAN THE AVERAGE NUMBER OF SALES BY ALL STAFF MEMBERS.
12. IDENTIFY THE CUSTOMERS WHO HAVE ORDERED ALL TYPES OF PRODUCTS (I.E., FROM EVERY CATEGORY).

Find the total number of products sold by each store along with the store name.

```sql
SELECT
    stores.store_name, SUM(order_items.quantity)
FROM
    stores
        JOIN
    orders ON stores.store_id = orders.store_id
        JOIN
    order_items ON order_items.order_id = orders.order_id
GROUP BY stores.store_name;
```

BHUVAN JARI

Calculate the cumulative sum of quantities sold for each product over time.

```sql
SELECT
    products.product_name,
    orders.order_date,
    order_items.quantity,
    SUM(order_items.quantity)
        OVER (PARTITION BY products.product_name
              ORDER BY orders.order_date) AS cumulative_sum
FROM
    products
JOIN
    order_items ON products.product_id = order_items.product_id
JOIN
    orders ON orders.order_id = order_items.order_id;
```

BHUVAN JARI

Find the product with the highest total sales (quantity * price) for each category.

```sql
WITH category_products_by_sales AS (
SELECT
    products.category_id,
    categories.category_name,
    products.product_id,
    products.product_name,
    SUM(order_items.quantity * order_items.list_price) AS total_sales,
    RANK() OVER(PARTITION BY products.category_id ORDER BY SUM(order_items.quantity * order_items.list_price) desc) AS product_sales_rank
FROM
    order_items
        JOIN
    products USING (product_id)
        JOIN
    categories ON products.category_id = categories.category_id
GROUP BY products.category_id, products.product_id)
SELECT category_id, category_name, product_id, product_name, total_sales FROM category_products_by_sales WHERE product_sales_rank = 1;
```

BHUVAN JARI

Find the customer who spent the most money on orders.

```sql
SELECT
    customers.customer_id,
    customers.first_name,
    customers.last_name,
    SUM(order_items.list_price * order_items.quantity) total_sales
FROM
    order_items
        JOIN
    orders USING (order_id)
        JOIN
    customers ON orders.customer_id = customers.customer_id
GROUP BY customers.customer_id , customers.first_name , customers.last_name
ORDER BY total_sales DESC
LIMIT 1;
```

BHUVAN JARI

Find the highest-priced product for each category name.

```sql
WITH product_rank_cte AS (
    SELECT
        c.category_id,
        c.category_name,
        p.product_id,
        p.product_name,
        p.list_price,
        RANK() OVER (PARTITION BY c.category_id ORDER BY p.list_price DESC) AS prod_rank
    FROM
        products p
    JOIN
        categories c USING (category_id))
SELECT
    category_id,
    category_name,
    product_id,
    product_name,
    list_price
FROM
    product_rank_cte
WHERE
    prod_rank = 1;
```

BHUVAN JARI

Find the total number of orders placed by each customer per store.

```sql
SELECT
    customers.customer_id,
    customers.first_name,
    customers.last_name,
    stores.store_id,
    stores.store_name,
    COUNT(*) AS num_orders
FROM
    orders
        JOIN
    customers USING (customer_id)
        JOIN
    stores ON orders.store_id = stores.store_id
GROUP BY customers.customer_id , customers.first_name , customers.last_name , stores.store_id , stores.store_name
ORDER BY customers.customer_id;
```

BHUVAN JARI

Find the names of staff members who have not made any sales.

```sql
SELECT
    staff_id, first_name, last_name
FROM
    staffs
WHERE
    staff_id NOT IN (SELECT DISTINCT
            staff_id
        FROM
            orders);
```

BHUVAN JARI

Find the top 3 most sold products in terms of quantity.

```sql
SELECT
    p.product_id, product_name, SUM(quantity) total_quantity
FROM
    order_items o
        JOIN
    products p USING (product_id)
GROUP BY product_id
ORDER BY total_quantity DESC
LIMIT 3;
```

BHUVAN JARI

Find the median value of the price list.

```sql
WITH list_price_cte AS (
    SELECT
        list_price,
        ROW_NUMBER() OVER (ORDER BY list_price) AS price_rank,
        COUNT(*) OVER () AS total_count
    FROM products)
SELECT
    CASE
        WHEN total_count % 2 = 0 THEN
            (
                SELECT AVG(list_price)
                FROM list_price_cte
                WHERE price_rank IN (total_count / 2, total_count / 2 + 1))
        ELSE
            (
                SELECT list_price
                FROM list_price_cte
                WHERE price_rank = (total_count + 1) / 2)
    END AS median
FROM list_price_cte
LIMIT 1;
```

BHUVAN JARI

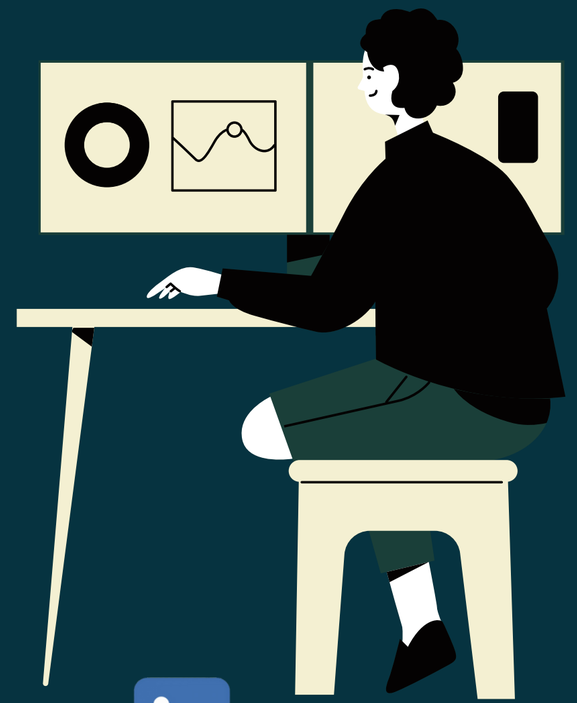
List all products that have never been ordered.(use Exists)

```sql
SELECT
    p.product_id, product_name
FROM
    products p
WHERE
    NOT EXISTS( SELECT
            1
        FROM
            order_items ot
        WHERE
            ot.product_id = p.product_id);
```

BHUVAN JARI

LIST THE NAMES OF STAFF MEMBERS WHO HAVE MADE MORE SALES THAN THE AVERAGE NUMBER OF SALES BY ALL STAFF MEMBERS.

```sql
WITH staff_sales AS (
    SELECT
        staff_id,
        SUM(quantity * list_price) AS sales
    FROM
        orders o
    JOIN
        order_items ot USING (order_id)
    GROUP BY
        staff_id
    ORDER BY
        sales DESC)
SELECT
    staffs.staff_id,
    first_name,
    last_name,
    sales
FROM
    staff_sales
JOIN
    staffs USING (staff_id)
WHERE
    sales > (
        SELECT AVG(sales)
        FROM staff_sales);
```



BHUVAN JARI

IDENTIFY THE CUSTOMERS WHO HAVE ORDERED ALL TYPES OF PRODUCTS (I.E., FROM EVERY CATEGORY)

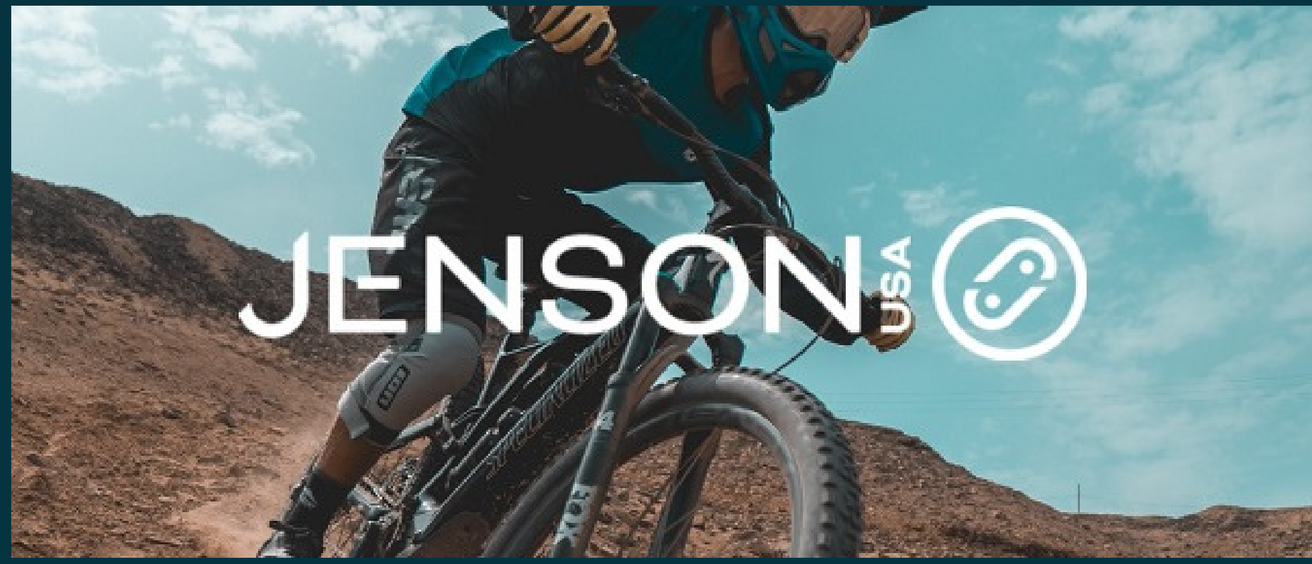
```sql
SELECT
    orders.customer_id,
    customers.first_name,
    customers.last_name
FROM
    order_items
        JOIN
    orders USING (order_id)
        JOIN
    products ON order_items.product_id = products.product_id
        JOIN
    customers ON orders.customer_id = customers.customer_id
GROUP BY orders.customer_id , customers.first_name , customers.last_name
HAVING COUNT(DISTINCT products.category_id) = (SELECT
        COUNT(category_id)
    FROM
        categories)
ORDER BY orders.customer_id;
```

BHUVAN JARI

THANK YOU
FOR YOUR ATTENTION

WsCube Tech

IF YOU FIND THIS HELPFUL, PLEASE LIKE AND SHARE IT WITH YOUR FRIENDS



BHUVANJAR0001@GMAIL.COM

PRESENTED BY - BHUVAN JARI