## Exercise: Creating a Combined Address View

In this exercise, you'll be tasked with joining data from three tables: `[TABLE1]`, `[TABLE2]`, and `[TABLE3]`. Your mission is to create a view that offers a comprehensive snapshot of the provided data.

### Objective:

Construct a SQL view named `combined_address_view_[SHORTNAME]` that merges data from the `[TABLE1]`, `[TABLE2]`, and `[TABLE3]` tables.

### Tables to be used:

1. **[TABLE1]**: Contains detailed information about various records.
2. **[TABLE2]**: Holds data about different attributes.
3. **[TABLE3]**: Offers details about various classifications.

### Requirements:

1. The view should be created in the `[YOUR PROJECT].adventureworks` schema.
2. The final view should include:
    - Relevant details from the `[TABLE1]` table.
    - Information from the `[TABLE2]` table.
    - Classification from the `[TABLE3]` table.
3. Apply JOIN operations to consolidate data from the three tables, ensuring correctness based on the relevant keys.

### Steps:

1. Begin with the structured query provided below:

```
CREATE OR REPLACE VIEW
  `[YOUR PROJECT].adventureworks.combined_address_view_[SHORTNAME]` AS
WITH
  joined_data AS (
  SELECT
    t1.[FIELD1],
```

```
    t1.[FIELD2],
    -- ... Add necessary columns from the '[TABLE1]' table

    t2.[FIELD1],
    -- ... Add necessary columns from the '[TABLE2]' table

    t3.[FIELD1],
    -- ... Add necessary columns from the '[TABLE3]' table
  FROM
    `adventureworks.[TABLE1]` t1
  JOIN
    `adventureworks.[TABLE2]` t2 ON t1.[JOIN_FIELD] = t2.[JOIN_FIELD]
  JOIN
    `adventureworks.[TABLE3]` t3 ON t2.[ANOTHER_JOIN_FIELD] =
↪   t3.[ANOTHER_JOIN_FIELD]
  )
SELECT
  *
FROM
  joined_data;
```

2. Adapt the query by:
    - Inserting the appropriate table and field names.
    - Ensuring your JOIN operations are logically sound.
3. After finalizing the query, execute it to generate the view.

**Tips:**

1. Make use of table aliases to improve query clarity.
2. Be judicious in selecting the fields to include in the final view, ensuring they're relevant to the task at hand.
3. Confirm your JOIN operations are based on the right keys to maintain data integrity.

### Exercise: Identifying Top Countries by Sales Orders

In this exercise, your objective is to leverage the power of SQL to discover which countries register the highest number of sales orders. By joining the `salesorderheader` table with the `combined_address_view_[SHORTNAME]` view, you will extract insights on sales performance by country.

**Objective:**

Determine the top three countries with the most sales orders by joining the `salesorderheader` table and the `combined_address_view_[SHORTNAME]` view.

**Tables to be used:**

1. **combined_address_view_[SHORTNAME]**: A view that amalgamates data, providing a comprehensive snapshot of addresses.
2. **salesorderheader**: Contains headers for sales orders, with each entry associated with a specific address.

**Requirements:**

1. Your query should be executed in the `[YOUR PROJECT].adventureworks` schema.
2. Count the number of sales orders associated with each country.
3. The results should display:
    - Country name.
    - Count of sales orders for each country.
4. Ensure countries are ranked in descending order based on sales order count.
5. List only the top three countries.

**Steps:**

1. Use the following skeleton to build your query:

```
SELECT
  -- Count the sales orders
  -- Select the appropriate column for country name from the view
FROM
  `adventureworks.salesorderheader` AS ??  -- Choose an alias
INNER JOIN
  `adventureworks.combined_address_view_[SHORTNAME]` AS ??  -- Choose an
 ↪  alias
ON
  -- Identify the correct fields for the JOIN operation
GROUP BY
  -- Group by the appropriate column to aggregate sales orders by country
```

```
ORDER BY
  -- Order the results based on sales order count
LIMIT
  3
```

2. Fill in the gaps in the provided skeleton:
    - Select the necessary columns.
    - Choose appropriate aliases for tables.
    - Identify the correct JOIN operation based on key fields.
    - Group, aggregate, and order the results correctly.
3. After constructing the query, execute it to retrieve the top three countries by sales order count.

**Tips:**

1. The key to this exercise is understanding the relationship between the `salesorderheader` table and the `combined_address_view_[SHORTNAME]` view.
2. Ensure your aggregation and grouping mechanisms are set up correctly to count sales orders by country.
3. Use the `LIMIT` clause to restrict your output to the top three results.