1) Your manager has requested a list of orders where the shipment was delayed, so the employees responsible can reach out to affected customers. In the '**Northwind'** database, write a query to retrieve the 'order_id' from the '**orders'** table for orders shipped after the required delivery date. Include the corresponding 'customer_id' and the first and last names of the employees from the '**employees'** table who processed each order. Display the 'order_ids' in descending order.

   SELECT order_id, customer_id, first_name, last_name
   FROM orders o
   INNER JOIN employees e
   ON o.employee_id = e.employee_id
   WHERE shipped_date > required_date
   ORDER BY order_id DESC;

2) Northwind company is interested in identifying which categories have the most products with high unit prices. In the '**Northwind'** database, write a query to find the number of products in each category from the '**products'** table where the unit price exceeds 15. Show the category name from the '**categories'** table and the corresponding count of high-priced products. Sort the results by the number of products in descending order and identify the category with the most products.

   SELECT c.category_name, COUNT(*) as high_priced_order_count
   FROM products p
   INNER JOIN categories c
   ON p.category_id = c.category_id
   WHERE unit_price > 15
   GROUP BY c.category_name
   ORDER BY high_priced_order_count DESC;

   The category with most orders is CONDIMENTS.

3) In the '**Northwind'** database, list the suppliers' company name and contact name from the '**suppliers'** table that provide products in the 'Seafood' category, as defined in the '**categories'** and '**products'** tables.

```
SELECT company_name, contact_name
FROM suppliers s
INNER JOIN products p
ON s.supplier_id = p.supplier_id
INNER JOIN categories c
ON p.category_id = c.category_id
WHERE category_name = 'Seafood';
```

4) Your manager wants to know how many distinct actors have appeared in films with an 'NC-17' rating. In the '**dvdrental'** database, write a query to retrieve the number of distinct actors from the '**actor'** table who has appeared in films from the '**film'** table with an 'NC-17' rating.

```
SELECT COUNT(DISTINCT(a.actor_id)) as distinct_actor_count
FROM actor a
INNER JOIN film_actor fa
ON a.actor_id = fa.actor_id
INNER JOIN film f
ON f.film_id = fa.film_id
WHERE rating = 'NC-17';
```

5) In the '**dvdrental'** database, write a query to retrieve the 'customer_id' and email of the top 5 customers with the highest number of DVD rentals, based on the '**rental'** and '**customer'** tables.

```
SELECT c.customer_id, email
FROM customer c
INNER JOIN rental r
ON c.customer_id = r.customer_id
GROUP BY c.customer_id
ORDER BY COUNT(r.rental_id) DESC
LIMIT 5;
```

6) In the '**dvdrental'** database, write a query to retrieve the 'category name' from the '**category'** table, the average film length from the '**film'** table, and the number of films in each category. Sort the results by 'category name' in ascending order.

```
SELECT c.name as category_name, AVG(f.length) as avg_length, COUNT(f.film_id) as
number_of_films
FROM category c
INNER JOIN film_category fc
ON c.category_id = fc.category_id
INNER JOIN film f
ON f.film_id = fc.film_id
GROUP BY category_name
ORDER BY category_name;
```

7) In the '**dvdrental'** database, write a query that captures the 'actor id' and counts the number of movies each actor has made through the '**film_actor'** and '**actor'** tables. Retrieve the top 5 actors ('actor_id') with the maximum number of movies.

```
SELECT a.actor_id, COUNT(fa.film_id) as number_of_films
FROM actor a
INNER JOIN film_actor fa
ON a.actor_id = fa.actor_id
GROUP BY a.actor_id
ORDER BY number_of_films DESC
LIMIT 5;
```

8) In the '**Northwind'** database, write a query that returns the employee id and first name of employees from the '**employees'** table along with the first name of their managers. Use '**employee_id'** and '**reports_to'** columns from the employees table (Hint: Self-join). Make sure your output also includes if an employee does not have a manager. Use appropriate column aliases for the first name of the employee and manager.

```
SELECT e1.employee_id, e1.first_name as employee_name, e2.first_name as
manager_name
FROM employees e1
LEFT JOIN employees e2
ON e1.reports_to = e2.employee_id;
```

9) In the '**dvdrental'** database, write a query to find the most rented film categories by using the '**category'**, '**film_category'**, '**film'**, '**inventory'**, and '**rental'** tables.

```
SELECT c.name as category_name
FROM category c
INNER JOIN film_category fc
ON c.category_id = fc.category_id
INNER JOIN film f
ON f.film_id = fc.film_id
INNER JOIN inventory i
ON i.film_id = f.film_id
INNER JOIN rental r
ON r.inventory_id = i.inventory_id
GROUP BY category_name
ORDER BY COUNT(r.rental_id) DESC;
```

10) In the '**dvdrental'** database, write a query to find **unique** film titles from the '**film'** table that start with 'A' and have been rented, utilizing the '**inventory'** and '**rental'** tables.

```
SELECT DISTINCT(title) as rented_films_starts_with_A
FROM film f
INNER JOIN inventory i
ON f.film_id = i.inventory_id
INNER JOIN rental r
ON i.inventory_id = r.inventory_id
WHERE title ILIKE 'A%';
```

11) **Bonus question:** In the '**dvdrental'** database, write a query to find actors in the '**actor'** table who share the same first name but have different last names. Use a self-join for this query.

```
SELECT a1.first_name as first_name, a1.last_name as last_name_1, a2.last_name as
last_name_2
FROM actor a1
INNER JOIN actor a2
ON a1.first_name = a2.first_name
WHERE a1.last_name != a2.last_name;
```

This query will return first name and corresponding last names of actors having same first name.

```
SELECT a1.first_name as first_name, COUNT(a1.last_name) as number_of_last_name
FROM actor a1
INNER JOIN actor a2
ON a1.actor_id = a2.actor_id
GROUP BY a1.first_name
HAVING COUNT(a1.last_name) > 1;
```

This query will return actor first name who have more than one last name and the number of last name for a single first name.