SQL Assignment Solutions

1. Find all employees whose first names start with a vowel and whose last names end with a consonant:

```
SELECT first_name, last_name
FROM employees
WHERE LEFT(first_name, 1) IN ('A', 'E', 'I', 'O', 'U')
AND RIGHT(last_name, 1) NOT IN ('A', 'E', 'I', 'O', 'U');
```

2. For each department, display the total salary expenditure, the average salary, and the highest salary:

```
SELECT department_id, salary,
    SUM(salary) OVER (PARTITION BY department_id) AS total_salary,
    AVG(salary) OVER (PARTITION BY department_id) AS avg_salary,
    MAX(salary) OVER (PARTITION BY department_id) AS max_salary
FROM salaries;
```

3. Fetch employees, their department, manager's name (self-join), and their salary:

```
SELECT e1.first_name AS employee_name, d.department_name,
    e2.first_name AS manager_name, s.salary
FROM employees e1
LEFT JOIN departments d ON e1.department_id = d.department_id
LEFT JOIN employees e2 ON e1.manager_id = e2.employee_id
```

```
JOIN salaries s ON e1.employee_id = s.employee_id;
```

4. Recursive CTE for reporting chain:

```
WITH RECURSIVE employee_hierarchy AS (
    SELECT employee_id, first_name, manager_id
    FROM employees
    WHERE manager_id IS NULL  -- top-level managers
    UNION ALL
    SELECT e.employee_id, e.first_name, e.manager_id
    FROM employees e
    JOIN employee_hierarchy eh ON e.manager_id = eh.employee_id
)
SELECT * FROM employee_hierarchy;
```

5. Fetch employees earning above a certain salary threshold and suggest improvements:

```
SELECT *
FROM employees e
JOIN salaries s ON e.employee_id = s.employee_id
WHERE s.salary > 60000;

-- Performance Improvement Suggestions:
-- 1. Create an index on the 'salary' column to speed up query performance.
-- Example: CREATE INDEX idx_salary ON salaries(salary);
```

6. Create a temporary table for sales report:

```
CREATE TEMPORARY TABLE product_sales (
    product_id INT,
    total_sales DECIMAL(10, 2),
    avg_sales_per_customer DECIMAL(10, 2),
    top_salesperson VARCHAR(50)
);

INSERT INTO product_sales (product_id, total_sales, avg_sales_per_customer, top_salesperson)
SELECT product_id,
       SUM(sales_amount) AS total_sales,
       AVG(sales_amount / customer_count) AS avg_sales_per_customer,
       MAX(salesperson_name) AS top_salesperson
FROM sales
GROUP BY product_id;
```