Complex Queries:

Complex queries have 3 or more tables joined, a custom scalar function and use built-in SQL functions and group by summarization.

Premise 01: How many vacation days have been taken by each department?
Detailed explanation of the problem: Return all Departments and the calculated amount of vacation days each has taken.

Query:

```
CREATE FUNCTION dbo.Ufngetvacationdays(@VacationHours AS INT)
returns INT
AS
  BEGIN
      RETURN @VacationHours / 24
  END;

go

USE adventureworks2014;

SELECT humanresources.department.NAME,
       Sum(dbo.Ufngetvacationdays(Employee_1.vacationhours)) AS VacationDays
FROM   humanresources.employee AS Employee_1
       INNER JOIN humanresources.employeedepartmenthistory
               ON Employee_1.businessentityid =
                  humanresources.employeedepartmenthistory.businessentityid
       INNER JOIN humanresources.department
               ON humanresources.employeedepartmenthistory.departmentid =
                  humanresources.department.departmentid
GROUP  BY humanresources.department.NAME
ORDER  BY vacationdays;
```

Sample Output:
Premise 02: Provide the Email Addresses for those who have taken the most vacation days.
Detailed explanation of the problem: Return the first name, last name, and e-mail address of the top 10 employees with the most vacation days.

Query:

```
SELECT TOP(10) person.person.firstname,
               person.person.lastname,
               person.emailaddress.emailaddress,
             dbo.Ufngetvacationdays(Employee_1.vacationhours) AS VacationDays
FROM   humanresources.employee AS Employee_1
       INNER JOIN person.person
              ON Employee_1.businessentityid = person.person.businessentityid
       INNER JOIN person.emailaddress
               ON person.person.businessentityid =
                  person.emailaddress.businessentityid
ORDER  BY vacationdays DESC;
```

Sample Output:

Premise 03: Which states/regions have the most orders shipping within 2 weeks?

Detailed explanation of the problem: Return all the States and Countries, and the amount of Orders that are shipping in rates less than 2 weeks.

Query:

```
CREATE FUNCTION dbo.Getshippingdays (@shipdate AS DATE,
                                     @duedate  AS DATE)
returns INT
AS
  BEGIN
      RETURN Datediff(year, @shipdate, @duedate)
  END;

go

SELECT person.stateprovince.stateprovincecode,
       person.stateprovince.countryregioncode,
       Count(salesorderid) AS OrderCount
FROM   person.stateprovince
       INNER JOIN person.address
               ON person.stateprovince.stateprovinceid =
                  person.address.stateprovinceid
       INNER JOIN sales.salesorderheader
               ON person.address.addressid =
                  sales.salesorderheader.billtoaddressid
                  AND person.address.addressid =
                      sales.salesorderheader.shiptoaddressid
WHERE  dbo.Ufngetshippingdays(salesorderheader.shipdate,
       salesorderheader.duedate) < 14
GROUP  BY person.stateprovince.stateprovincecode,
          person.stateprovince.countryregioncode
ORDER  BY ordercount DESC;
```

Sample Output:

Premise 04: Which sales country has taken the most vacation days?

Detailed explanation of the problem: Return the top country with the most vacation days taken.

Query:

```
SELECT TOP(1) sales.salesterritory.countryregioncode,
              Count(dbo.Ufngetvacationdays(Employee_1.vacationhours)) AS
              VacationDays
FROM   humanresources.employee AS Employee_1
       INNER JOIN sales.salesperson
               ON Employee_1.businessentityid =
                  sales.salesperson.businessentityid
       INNER JOIN sales.salesterritory
               ON sales.salesperson.territoryid =
                  sales.salesterritory.territoryid
GROUP  BY salesterritory.countryregioncode
ORDER  BY vacationdays DESC;
```

Sample Output:

Premise 05: What is the average amount of sales per month for each store?

Detailed explanation of the problem: Return all store names, and the calculated average of their sales per month.

Query:

```
CREATE FUNCTION dbo.Avgsalespermonth(@SalesLastYear AS INT)
returns INT
AS
  BEGIN
      RETURN @SalesLastYear / 12
  END

go

SELECT sales.store.NAME,
    dbo.Ufnavgsalespermonth(salesterritory.saleslastyear) AS AvgSalesPerMonth
FROM   sales.store
       INNER JOIN sales.customer
               ON sales.store.businessentityid = sales.customer.storeid
       INNER JOIN sales.salesterritory
             ON sales.customer.territoryid = sales.salesterritory.territoryid
ORDER  BY sales.store.NAME;
```

Sample Output:

Premise 06: How many employees were hired in January for each department?

Detailed explanation of the problem: Return all Departments and the count of employees hired in January.

Query:

```
CREATE FUNCTION dbo.Gethiremonth (@HireDate AS DATE)
returns INT
AS
  BEGIN
      RETURN Month(@HireDate)
  END;

go

SELECT humanresources.department.NAME,
       Count(dbo.Gethiremonth(Employee_1.hiredate)) AS HiredinMonth
FROM   humanresources.employeedepartmenthistory
       INNER JOIN humanresources.employee AS Employee_1
               ON humanresources.employeedepartmenthistory.businessentityid =
                  Employee_1.businessentityid
       INNER JOIN humanresources.department
               ON humanresources.employeedepartmenthistory.departmentid =
                  humanresources.department.departmentid
WHERE  dbo.Gethiremonth(Employee_1.hiredate) = 1
GROUP  BY humanresources.department.NAME
ORDER  BY hiredinmonth DESC;
```

Sample Output:

Premise 07: Provide the E-Mail Addresses for employees hired in June**.**

Detailed explanation of the problem: Return all Email Addresses for Employees hired in June.

Query:

```
SELECT person.person.firstname,
       person.person.lastname,
       person.emailaddress.emailaddress
FROM   humanresources.employee AS Employee_1
       INNER JOIN person.person
              ON Employee_1.businessentityid = person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
       INNER JOIN person.emailaddress
               ON person.person.businessentityid =
                  person.emailaddress.businessentityid
WHERE  dbo.Gethiremonth(Employee_1.hiredate) = 5
GROUP  BY person.person.firstname,
          person.person.lastname,
          person.emailaddress.emailaddress
ORDER  BY person.firstname;
```

Premise 08: Provide the same information as above for employees hired on the same month as their birthday.

Detailed explanation of the problem: Return all Email Addresses for employees hired on their birth month.

Query:

```
SELECT person.person.firstname,
       person.person.lastname,
       person.emailaddress.emailaddress
FROM   humanresources.employee AS Employee_1
       INNER JOIN person.person
              ON Employee_1.businessentityid = person.person.businessentityid
                 AND Employee_1.businessentityid =
                     person.person.businessentityid
                 AND Employee_1.businessentityid =
                     person.person.businessentityid
                 AND Employee_1.businessentityid =
                     person.person.businessentityid
       INNER JOIN person.emailaddress
               ON person.person.businessentityid =
                  person.emailaddress.businessentityid
WHERE  dbo.Gethiremonth(Employee_1.hiredate) = Month(Employee_1.birthdate)
GROUP  BY person.person.firstname,
          person.person.lastname,
          person.emailaddress.emailaddress
```

Sample Output:

Premise 09: Who was hired in 2009, and what are their phone numbers?

Detailed explanation of the problem: Return the Last Name and Phone Numbers associated with the employees hired in 2009.

Query:

```
CREATE FUNCTION dbo.Gethireyear (@HireDate AS DATE)
returns INT
AS
  BEGIN
      RETURN Year(@HireDate)
  END;

go

SELECT person.person.lastname,
       person.personphone.phonenumber
FROM   person.personphone
       INNER JOIN humanresources.employee AS Employee_1
               ON person.personphone.businessentityid =
                  Employee_1.businessentityid
       INNER JOIN person.person
               ON person.personphone.businessentityid =
                  person.person.businessentityid
                  AND person.personphone.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
                  AND Employee_1.businessentityid =
                      person.person.businessentityid
WHERE  ( dbo.Gethireyear(Employee_1.hiredate) ) = 2009
ORDER  BY lastname;
```

Sample Output:

Premise 10: Provide the average pay rates of employees hired in 2010 for each department?

Detailed explanation of the problem: Return all Departments and the average pay rate for each one. Make sure it is for 2010 employees only.

Query:

```
SELECT humanresources.department.NAME,
       Avg(humanresources.employeepayhistory.rate) AS AvgPayRate
FROM   humanresources.employee AS Employee_1
       INNER JOIN humanresources.employeedepartmenthistory
               ON Employee_1.businessentityid =
                  humanresources.employeedepartmenthistory.businessentityid
       INNER JOIN humanresources.department
               ON humanresources.employeedepartmenthistory.departmentid =
                  humanresources.department.departmentid
       INNER JOIN humanresources.employeepayhistory
               ON Employee_1.businessentityid =
                  humanresources.employeepayhistory.businessentityid
WHERE  dbo.Gethireyear(Employee_1.hiredate) = 2010
GROUP  BY NAME
ORDER  BY avgpayrate DESC;
```