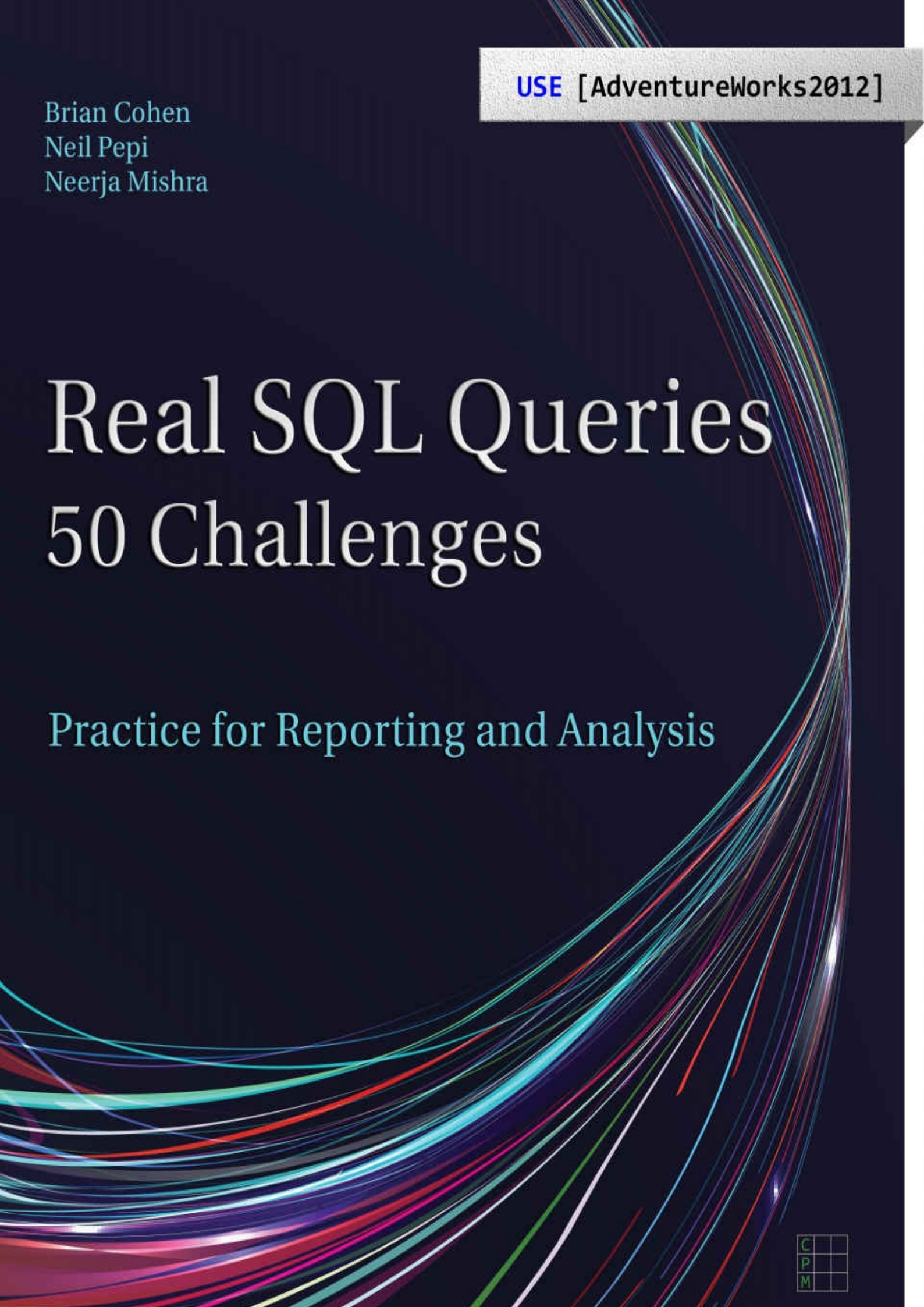

Brian Cohen
Neil Pepi
Neerja Mishra

Real SQL Queries
50 Challenges

Practice for Reporting and Analysis

REAL SQL QUERIES
50 CHALLENGES

Brian Cohen

Neil Pepi

Neerja Mishra

Real SQL Queries: 50 Challenges

By Brian Cohen, Neil Pepi, and Neerja Mishra

© 2015 Brian Cohen, Neil Pepi, and Neerja Mishra.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without prior written permission of the authors at the address below.

Brian Cohen

31748 Blythewood Way

Wesley Chapel, FL 33543

Microsoft, Excel, and SQL Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are the property of their respective owners. When trademark designations appear in this book, and the authors were aware of a trademark claim, the designations have been printed in initial capital letters.

While every precaution has been taken in the preparation of this book, the authors assume no responsibility for errors or omissions, or for damages resulting from the use or information contained herein.

Copy Editor: Haley DeLeon

Additional content available online at http://realsqlqueries.com

ABOUT THE AUTHORS

BRIAN COHEN

Brian Cohen is Senior Database Marketing Analyst at Bisk Education where he analyzes the effectiveness of marketing initiatives. Brian earned his Bachelor of Arts in Economics from University of South Florida in 1997. He holds a Master Certificate in Internet Marketing from University of San Francisco and is currently enrolled with Villanova University, seeking a Master Certificate in Business Intelligence. More about Brian Cohen at http://bricohen.com

Many thanks to Brian's wife René for her encouragement throughout the process of creating the book. Brian also thanks his colleagues at Bisk Education. Brian is fortunate to work alongside so many experts who share knowledge generously.

NEIL PEPI

Neil learned a lot of what he knows today at Bisk Education, where he is currently a Database Reporting Analyst. Bisk created an environment that holds practical, on-the-job learning as a priority. He would be remiss if he didn’t thank the owners of Bisk Education and all of the talented people that work there.

Neil earned his Bachelor of Science in Accounting from the University of South Florida. He can be contacted at neilpepi1@gmail.com.

NEERJA MISHRA

Neerja Mishra is a SQL Developer currently working with Citibank in Tampa Florida. She graduated from University of Delhi, India with a Bachelor in Science and she earned her Master's in Engineering from South Dakota State University. Besides her work, she enjoys music, travelling, watching documentaries and reading. Neerja Mishra can be contacted at nes1711@hotmail.com.

TABLE OF CONTENTS

SOLUTIONS TO CHALLENGE QUESTIONS

INTRODUCTION

We, the authors, are practical developers. While we are interested in conceptual learning, we think real problem solving is king. In textbooks, we skip to the practice queries. At work, we scour the internet for coding techniques to answer questions about our business. We're most excited by SQL queries that get the job done.

With 50 challenge questions, this book enables the practical developer. To make the material more relatable, we've designed the challenges with report writers and analysts in mind. And we've tailored all challenges to AdventureWorks2012, Microsoft's universally-accessible sample database, so that you can begin querying immediately.

THE CHALLENGES

The difficulty of the challenges throughout the book is mixed, so the material accommodates many backgrounds. Some questions are simple, while others call for strategic, multi-step solutions. We aimed to include a broad group of SQL users within our audience.

THE SOLUTIONS

We believe independent reasoning is essential. Accordingly, solutions are provided but not necessarily dissected. We empower you to think through our code without interruption or noise.

Our solutions favor human logic and readability over performance. We worked through the problems in ways that were rational to us while carrying reasonable efficiency. Performance tuning is certainly worthy, but its craft falls outside the scope of this book.

There are many possible approaches to any given solution, so you're likely to discover alternate routes that feel more appropriate. Likewise, numerous coding styles exist and our preferences may not line up with yours. We encourage you to rewrite the SQL as you see fit.

WHAT'S NEEDED

The book is best supported by SQL Server 2012 with AdventureWorks2012. The Express version of SQL Server 2012 is free, so you should install the software if needed. AdventureWorks2012 is free as well.

The following links should assist you with downloads and installations:

SQL Server Express 2012 Download

AventureWorks2012 Download

ORGANIZATION

The book is divided into two parts. In Part I, we begin with a discussion about navigating relationships among unfamiliar tables. We've outlined ideas to help you prepare the right road map for the right problem-solving journey. Part I continues with lots of varied challenge questions for you try.

In Part II, you'll find hints, strategies, and solutions. We placed this material far away from the challenge questions by design. You won't accidentally bump into unwanted answers while you're working through questions. As mentioned earlier, we enable you to think without interruption.

We've prepared each solution as a distinct .sql file for download. Feel free to retrieve whichever files you like at http://realsqlqueries.com/

EXPLORING A NEW DATABASE

It is not uncommon for a report writer or analyst to be thrown into a new database with little to no guidance, let alone a data dictionary. For exactly these reasons, it can be beneficial to be able to investigate a database yourself before asking your coworkers unnecessary questions. Luckily, SQL Server has a lot of robust tools built-in for conducting such research.

For example, let's say you want to find a column somewhere in the database, but you're not sure what table it exists on. The code below can do exactly that; all you have to do is replace the database name and the column name you're searching for.

```
USE DatabaseName
GO

—DROP TABLE #data

SELECT
   TableView = ISNULL (N2.Name, N3.Name)
   ,ObjectView =
      CASE
         WHEN N2.Name IS NOT NULL
            THEN 'Table'
         ELSE 'View'
      END
   ,ColName = N1.Name
INTO #data
FROM sys.columns N1
LEFT JOIN sys.tables N2
   ON N1.object_id = N2.object_id
LEFT JOIN sys.views N3
   ON N1.object_id = N3.object_id
WHERE N2.object_id IS NOT NULL
   OR N3.object_id IS NOT NULL
ORDER BY TableView

SELECT *
FROM #data
WHERE ColName LIKE '%ColumnName%'
```

Now, let's say you've found the table you want to use, but you're not entirely sure how it's structured. For example, how do you find out if a column is unique if there isn't a primary key constraint? The code below can show you any value for the

field selected that has more than 1 row. If this query returns no results, the field you've chosen is unique.

```
SELECT ColumnName
FROM TableName
GROUP BY ColumnName
HAVING COUNT (*) > 1
```

If we augment that code slightly, we can now check for relationships. The following query is checking if any values exist for ColumnName1 that have multiple values in ColumnName2.

```
SELECT ColumnName1
FROM TableName
GROUP BY ColumnName1
HAVING COUNT (DISTINCT ColumnName2) > 1
```

If we then run the opposite, we can determine the relationship between these two columns.

```
SELECT ColumnName2
FROM TableName
GROUP BY ColumnName2
HAVING COUNT (DISTINCT ColumnName1) > 1
```

Based on the results, the table below can show you the relationship between ColumnName1 and ColumnName2.

		Second Query	
		Returns results	**Does not return results**
First Query	**Returns results**	many : many	1 : many
	Does not return results	many : 1	1 : 1

CHALLENGE QUESTIONS

CHALLENGE QUESTION 1: YEAR OVER YEAR COMPARISONS

Difficulty: Intermediate

An executive requests data concerning fiscal quarter sales by salesperson. She'd like to see comparisons from the fiscal quarters of 2008 to the same fiscal quarters of 2007.

For example, suppose sales for salesperson X totaled $1,000 during Fiscal Year 2008, Fiscal Quarter 2. If sales for salesperson X totaled $900 in Fiscal Year 2007, Fiscal Quarter 2, this reflects about 11.1% growth between the two periods for salesperson X.

Notes:

- For Adventure Works, the fiscal year spans July through June
- Tax and freight will not be considered with revenue
- Dates are based on OrderDate
- Disregard online orders

Your output should include the following columns, corresponding to all sales people:

- LastName
- SalesPersonID
- Fiscal year
- Fiscal quarter
- Fiscal quarter sales
- Sales during the same fiscal quarter of the previous fiscal year
- Change in revenue between the two periods
- Percent change in revenue between the two periods

Following the same example, your output for one row would appear as follows:

LastName	SalesPersonID	FY	FQ	FQSales	SalesSameFQLast	Change	%Change
X	1	2008	2	1000	900	100	11.1

CHALLENGE QUESTION 2: THE 2/22 PROMOTION

Difficulty: Intermediate

A marketing manager devised the “2/22” promotion, in which orders subtotaling at least $2,000 ship for $0.22. The strategy assumes that freight losses will be offset by gains from higher value orders. According to the marketing manager, orders between $1,700 and $2,000 will likely boost to $2,000 as customers feel compelled to take advantage of bargain freight pricing.

You are asked to test the 2/22 promotion for hypothetical profitability based on the marketing manager’s assumption about customer behavior. Examine orders shipped to California during fiscal year 2008 for net gains or losses under the promotion.

Part I

Create a table that includes the following columns:

- SalesOrderID
- Ship to state (California)
- OrderDate
- Historical order subtotal (prior to any changes as a result of the promotion)
- Historical freight (prior to any changes as a result of the promotion)
- Potential promotional effect. Indicate one of three hypothetical scenarios related to the order:
    - Increase order to $2,000 and pay $0.22 freight
    - No order change and pay $0.22 freight
    - No order change and pay historical freight
- Potential order gain
- Potential freight loss
- Potential promotional net gain/loss

Notes:

- For Adventure Works, the fiscal year spans July through June
- Tax should not be considered

Part II

Aggregate data from Part I by PotentialPromotionalEffect. Include the following:

- PotentialPromotionalEffect
- Potential order gains
- Potential freight losses
- Overall net gain/loss

CHALLENGE QUESTION 3: TEN MILLION DOLLAR BENCHMARK

Difficulty: Intermediate

Ten million dollars of revenue is a common benchmark for Adventure Works. For each fiscal year (2007 and 2008), find the first dates when the cumulative running revenue total hit $10 million.

Notes:

- For Adventure Works, the fiscal year spans July through June
- Do not consider tax and freight with revenue

Your output should include the following columns:

- Fiscal year (2007 or 2008)
- Order date in which $10 million was reached or exceeded
- Order number within the fiscal year in which $10 million was reached or exceeded. Note, this is a count of orders. For example, if the $10 million goal was reached on the 50th order, then the appropriate value to report is 50.
- Running total revenue in which $10 million was reached or exceeded

Example output:

FiscalYear	OrderDate	FYOrder#	RunningTotal
2007	7/10/2007	6	10,000,008
2008	7/29/2007	5	10,000,012

CHALLENGE QUESTION 4: UPSELL TUESDAYS

Difficulty: Beginner

Tuesday's are "upsell" days for sales people at Adventure Works. Management wants to compare sales from Tuesday to other days of the week to see if the initiative is working. Help monitor the upsell initiative by creating a query to calculate average revenue per order by day of week in 2008.

Include the following columns with your output:

- Day of week
- Revenue
- Orders
- Revenue per order

Notes:

- Dates based on OrderDate
- Tax and freight should not be considered
- Exclude online orders

CHALLENGE QUESTION 5: EXPIRED CREDIT CARDS

Difficulty: Intermediate

The Accounting department found instances where expired credit cards were used with sales orders. You are asked examine all credit cards and report the extent of such activity.

PART I

Based on each CreditCardID, find the following:

- CreditCardType
- ExpirationDate
- Last order date
- Number of sales orders with order dates earlier than or equal to the card's expiration date
- Number of sales orders with order dates later than the card's expiration date

Note:

Adventure Works stores information about a credit card's expiration year and expiration month. Expiration dates pertain to the last day of a card's expiration

month. For example, if the expiration year is 2007 and the expiration month is “4”, the card’s expiration date will be April 30, 2007.

Part II

Based on CreditCardType, summarize data returned from Part I. Your output should include the following columns:

- CreditCardType
- Number of sales orders with order dates earlier than or equal to the card’s expiration date
- Number of sales orders with order dates later than the card’s expiration date

CHALLENGE QUESTION 6: PRINT CATALOG

Difficulty: Beginner

Adventure Works will feature one product for the cover of its print catalog. Help select a list of products for consideration.

Your list should contain products which meet all of the following conditions:

- Finished goods (not products utilized to make other products)
- List price at least $1,500
- At least 150 in inventory
- Currently available for sale

Your output should contain the following columns:

- ProductID
- ProductName
- Color
- ListPrice
- Inventory quantity

CHALLENGE QUESTION 7: SPECIAL TEAM

Difficulty: Intermediate

An Adventure Works executive asks for a list of all salespeople representing the Northwest, Southwest, and Canadian sales territories, along with their 2008 revenue. You ask about the purpose of the information. “I’m assembling a ‘special team,’ the executive responds. “I’ll tell you more soon.”

Create a list of sales people with revenue as requested. Do not consider tax and freight.

CHALLENGE QUESTION 8: KNAPSACK PROBLEM

Difficulty: Advanced

The following day, after completing the previous challenge, the executive approaches you again.

“I’m starting a new company,” he says. “That’s why I asked for the information yesterday. I need you to recommend candidates for recruitment.

I want one salesperson from the Northwest territory, one from the Southwest territory, and one from the Canadian territory. Which three-person combination of sales people results in the highest revenue? Base your calculations on the 2008 revenue you previously found.

I want the best team, but we cannot exceed the salary constraints. My salary budget for the combined roster must be less than $210,000.”

Below are the salaries you can offer to each sales person, as provided by the executive:

- Pak: $79,500
- Vargas: $60,000
- Campbell: $59,500
- Mensa-Annan: $56,000
- Ito: $68,000
- Michel: $80,000

Based on 2008 revenue, find the most valuable combination of sales people within the salary constraint.

Your output should include a single row with the following columns:

- Sum of salaries of your three person team
- Sum of revenue from your three person team
- Territory of the first sales person
- Last name of first sales person
- Revenue of first sales person
- Salary of first sales person
- Territory of the second sales person
- Last name of second sales person
- Revenue of second sales person
- Salary of second sales person
- Territory of the third sales person
- Last name of third sales person
- Revenue of third sales person
- Salary of third sales person

Example output:

Aggregate Salary	Aggregate Rev	1st Territory	1st Sales Person	1st Sales Person Rev	1st Salary	2nd Territory	2nd Sales Person	2nd Sales Person Rev	2nd Salary	3rd Territory	3rd Sales Person	3rd Sales Person Rev	3rd Salary
209000	425000	Canada	Smith	1800000	70000	Northwest	Jones	800000	50000	Southwest	White	900000	65000

CHALLENGE QUESTION 9: PRODUCT COMBINATIONS

Difficulty: Advanced

The executive management team wants to analyze the buying behavior of customers.

PART I

Provide the following calculations:

- Percentage of sales orders containing at least one bike and at least one accessory item
- Percentage of sales orders containing at least one bike and at least two different clothing products

PART II

Count sales orders by product type. For example, if 500 sales orders included the product types Bikes and Clothing, with no accessories and components purchased, that output row would appear as follows:

Bikes	Accessories	Clothing	Components	Orders
1	0	1	0	500

PART III

Count customers by product line. For example, if 100 customers purchased products from product lines M and T, but not S and R, that output row would appear as follows:

M	S	T	R	Customers
1	0	1	0	100

CHALLENGE QUESTION 10: MEDIAN REVENUE

Difficulty: Intermediate

An analyst notified the Vice President of Sales that averages can be skewed by outliers. In response, he asks to see median revenue in addition to average revenue. He also asks you to add minimum and maximum revenue to your report.

Write a query of sales by year that includes the following columns:

- Order year
- Minimum sale
- Maximum sale
- Average sale
- Median sale

Notes:

- Years based on OrderDate
- Tax and freight should not be considered with revenue

CHALLENGE QUESTION 11: NEEDY ACCOUNTANT

Difficulty: Beginner

An accountant needs to add assumptions about sales tax rates to his Excel worksheet. He asks you to provide one sales tax rate, conservatively, for each of the countries in which tax rates are known.

To fulfill his request, report the maximum sales tax rate of each country.

CHALLENGE QUESTION 12: PRODUCT INVENTORY UPDATES

Difficulty: Advanced

You are asked to provide frequent updates about the Adventure Works product inventory. Create a view that includes the following:

- Number of distinct products by LocationID
- Quantity of products by LocationID
- A rollup with total number of distinct products throughout all LocationIDs
- A rollup with total quantity of products throughout all LocationIDs

CHALLENGE QUESTION 13: VACATION HOURS

Difficulty: Intermediate

Human Resources is reevaluating a policy about maximum allowable vacation rollover hours. You are asked to help by identifying the employee or group of employees with the greatest number of vacation hours. Since many Human Resources files are indexed by NationalIDNumber, please include the last four digits with your output.

In all, your output should contain the following information:

- Last four digits of NationalIDNumber
- FirstName
- LastName
- JobTitle
- Number of vacation hours

CHALLENGE QUESTION 14: PURCHASING

Difficulty: Intermediate

For each product ordered by the Purchasing Department in 2007, indicate the quantity ordered by order date. In all, include the following columns with your output:

- ProductID
- Product name
- OrderDate
- Quantity ordered

Arrange data in descending order by quantity ordered.

CHALLENGE QUESTION 15: INTERPRETATION NEEDED

Difficulty: Intermediate

The Adventure Works Marketing department utilizes contractors to review foreign language product descriptions. To help the contractors, you are asked to prepare a list of all product descriptions written in languages other than English.

Your output should contain the following columns:

- ProductModelID
- Name of product model
- Product description
- Language

CHALLENGE QUESTION 16: ONLINE/OFFLINE

Difficulty: Beginner

Create a summary table that shows, by territory, the percentage of orders placed online in comparison to orders not placed online.

Your output should include the following columns:

- TerritoryID
- Total orders
- Percentage of orders placed online
- Percentage of orders not placed online

To make the table easier to read, display percentages with a percent sign without decimals. For example, ninety five percent will be displayed as 95%.

CHALLENGE QUESTION 17: LONG TIME NO SALE

Difficulty: Intermediate

The sales department will visit stores without recent sales orders. Suppose today's date is October 7, 2008. Create a report that identifies stores in which the last order date was at least 12 months ago.

Your output should include the following columns:

- BusinessEntityID
- CustomerID
- StoreID
- StoreName
- Last order date
- Number of months since last order

CHALLENGE QUESTION 18: COSTS VARY

Difficulty: Intermediate

A team was formed with the goal of reducing product costs. Help the team kick off their first meeting by compiling baseline data about historical product cost variability.

Query the data by ProductID. Your output should include the following:

- ProductID
- ProductName
- SubCategory
- Minimum historical cost
- Maximum historical cost
- Historical cost variability (maximum historical cost minus minimum historical cost)
- Ranking of all historical cost variabilities (rank of “1” reflects the product ID exhibiting the greatest historical cost variability)

CHALLENGE QUESTION 19: THERMOFORM TEMPERATURE

Difficulty: Intermediate

You are asked to report the most common reasons why products were scrapped through the manufacturing process. Create a query by ProductID that includes the following:

- ProductID
- ProductName
- Number of work orders affected
- Most common scrap reason

For example, suppose “Thermoform temperature too high” was the most common reason why ProductID 398 was scrapped. If the product was scrapped 100 times as a result of this reason, your row about the product would appear as follows:

ProductID	ProductName	WorkOrderCount	ScrapReason
398	Handlebar Tube	100	Thermoform temperature too high

CHALLENGE QUESTION 20: TORONTO

Difficulty: Intermediate

Provide address data about stores with main offices located in Toronto.

Your output should include the following columns:

- Store name
- AddressLine1
- AddressLine2
- City
- StateProvince
- PostalCode

CHALLENGE QUESTION 21: MARKETING EMPLOYEES

Difficulty: Intermediate

An administrator from Human Resources asks you for a list of employees who are currently in the Marketing department and were hired prior to 2002 or later than 2004.

Your output should include the following columns:

- FirstName
- LastName
- JobTitle
- BirthDate
- MaritalStatus
- HireDate

CHALLENGE QUESTION 22: WHO LEFT THAT REVIEW?

Difficulty: Intermediate

The executives want to know if it's possible to link people who have made product reviews with customer data. The end goal is to link sales information to product reviews.

As a first step, try looking up BusinessEntityID of reviewers based on e-mail addresses. If known, BusinessEntityIDs can point to sales orders through CustomerIDs.

Your output should include the following columns:

- ProductReviewID
- Product ID
- Product name
- ReviewerName
- Rating
- Reviewer's email address
- Reviewer's BusinessEntityID (if known)

CHALLENGE QUESTION 23: LABEL MIX-UP

Difficulty: Intermediate

Some clothing items were mislabeled, and management will inform customers. You are asked to help by compiling a list of affected customers with phone numbers. The issue pertains to all orders for shorts placed online after July 7, 2008.

Your list should contain the following columns:

- SalesOrderID
- OrderDate
- ProductName
- Customer’s first name
- Customer’s last name
- Customer’s phone number

CHALLENGE QUESTION 24: CLEARANCE SALE

Difficulty: Intermediate

The Marketing department will prepare a mass e-mail to notify individual retail customers about a clearance sale. You are asked to report the depth of e-mail addresses within the company's databases.

According the requestor, e-mail address counts should be based on e-mail preferences. E-mail preferences are recorded in the Person.Person table within the column EmailPromotion.

Utilize the following e-mail preference conversions as part of your output:

- The value "0" indicates "Contact does not wish to receive e-mail promotions"
- The value "1" indicates "Contact does wish to receive e-mail promotions from AdventureWorks"
- The value "2" indicates "Contact does wish to receive e-mail promotions from AdventureWorks and selected partners"

Example output:

Email Preference	Count
Contact does not wish to receive e-mail promotions	500
Contact does wish to receive e-mail promotions from AdventureWorks	400
Contact does wish to receive e-mail promotions from AdventureWorks and selected partners	300

CHALLENGE QUESTION 25: TOP TERRITORIES

Difficulty: Intermediate

In terms of revenue, which two sales territories were top performers during fiscal years 2006 and 2007?

Notes:

- For AdventureWorks, the fiscal year spans July through June
- Tax and freight will not be considered with revenue

Your output should include the following columns:

- Fiscal year
- Territory name
- Revenue
- Territory rank

CHALLENGE QUESTION 26: COMMISSION PERCENTAGES

Difficulty: Beginner

Rank commission percentages by sales person.

Notes:

- A rank of “1” should relate to the sales person with the greatest commission percentage
- If commission percentages are equal among sales people, rank by Bonus in descending order

Your solution should include the following columns:

- BusinessEntityID
- Commission percent
- Bonus
- Rank

CHALLENGE QUESTION 27: WORK ORDERS

Difficulty: Beginner

PART I

The Production department asks you to report the number of work orders by ProductID. Order your results from the greatest number of work orders to the least.

PART II

Report the number of work orders by product name. Order your results from the greatest number of work orders to the least.

CHALLENGE QUESTION 28: REVENUE TRENDED

Difficulty: Intermediate

PART I

Suppose today's date is May 24, 2008. Using only revenue information from May 1 through May 23, estimate revenue for the whole month of May.

Notes:

- Dates are based on OrderDate
- Tax and freight will not be considered with revenue

Your output should include the following columns:

- Number of days in month so far
- Total revenue in month so far
- Revenue per day for the month so far
- Monthly revenue trended for all of May

PART II

For the sake of comparison, pull the actual revenue information.

Your output should include the following columns:

- Actual revenue per day
- Actual revenue

CHALLENGE QUESTION 29: SEPARATION

Difficulty: Intermediate

The HumanResources.Employee table includes the column LoginID in which user names and domains are combined.

In your query, separate the names from the domains. Your output should include the following:

- BusinessEntityID
- LoginID (for example, adventure-works\ken0)
- Domain (for example, adventure-works)
- Username (for example, ken0)

CHALLENGE QUESTION 30: SHIFT COVERAGE

Difficulty: Intermediate

Management will review the current distribution of labor by shift within the Production department.

Create a report that includes the following:

- Department name (Production)
- Shift name
- Number of employees

CHALLENGE QUESTION 31: LABELS

Difficulty: Beginner

Labels representing product sizes will be applied to the boxes and packages of some products. The variety of labels include “S” (size “Small”), “M” (size “Medium”), “L” (size “Large”), and “XL” (size “Extra Large”).

Part I

Write a query to determine if the variety of labels is sufficient to cover all alpha-sized products. For example, since “2XL” labels do not exist, no label could be applied to a “2XL” product. If a “2XL” product existed, the variety of labels would be insufficient.

Part II

Suppose 1,000 labels are available in each size. Calculate the number of additional labels needed to cover all the relevant products in the inventory.

By size, create a table that includes the following columns:

- Size
- Current quantity
- Additional labels needed

CHALLENGE QUESTION 32: EMPLOYMENT SURVEY

Difficulty: Intermediate

Adventure Works will participate in a third-party employment survey among bicycle manufacturers. The Human Resources department asks you to help prepare data to submit.

PART I

Provide the following:

- Total number of employees throughout the company
- Percentage of employees who are Male
- Percentage of employees who are Female
- Average number of months of employment. Pretend today is January 1, 2008.

Sample output:

Employees	%Male	%Female	AvgMonthsEmployed
100	50.00	50.00	10

PART II

Divide employee data into quartiles based on average number of months of employment. By quartile, provide the following:

- Total number of employees throughout the company
- Percentage of employees who are Male
- Percentage of employees who are Female
- Average number of months employed. Pretend today is January 1, 2008.

Sample output:

Quartile	Employees	%Male	%Female	AvgMonthsEmployed
1	25	50.00	50.00	10
2	25	50.00	50.00	10

CHALLENGE QUESTION 33: AGE GROUPS

Difficulty: Intermediate

In the previous question, you provided data for a third-party employment survey among bicycle manufacturers. More information is needed.

Create a query summarizing pay rates and age groups by job title. Assume today is January 1, 2008. Your output should be structured with the following columns:

- JobTitle
- Age group, in years, categorized as follows:
  - < 18
  - 18 – 35
  - 36 – 50
  - 51 – 60
  - 61 +
- Pay rate
- Number of employees

As an example, if 10 Production Assistants were 39 years old on January 1, 2008, and their pay rate was 23.65, the corresponding output row would appear as follows:

JobTitle	AgeGroup	Rate	Employees
Production Assistant	36 – 50	23.65	10

CHALLENGE QUESTION 34: REVENUE BY STATE

Difficulty: Beginner

Report revenue by state in 2006. Order the data from states with the greatest revenue to states with the least revenue.

Notes:

- Dates based on OrderDate
- Revenue includes tax and freight
- States based on shipping address

CHALLENGE QUESTION 35: TWO FREE BIKES

Difficulty: Intermediate

Two employees are given free bicycles at the start of each quarterly meeting. The employees are chosen at random, with eligibility limited to the least senior positions.

Create a view to generate employee names. The view should include the following columns:

- FirstName
- LastName
- JobTitle

CHALLENGE QUESTION 36: VOLUME DISCOUNTS

Difficulty: Intermediate

You are asked to report data about volume discounts.

PART I

Create a query about sales orders that utilized volume discounts.

Your output should include the following columns:

- SalesOrderID
- OrderDate
- Total volume discount (the sum of volume discounts applied to the order)

PART II

Summarize data from Part I by order year.

Include the following:

- Order year
- Total volume discount

CHALLENGE QUESTION 37: OVERPAYING

Difficulty: Intermediate

Some products are purchased from multiple vendors. Concerned about overpaying for products, the executive team asks to see price comparisons among vendors.

Using the Purchasing.ProductVendor table exclusively, determine if products are purchased at significantly lower prices from one vendor to another. By ProductID, create a query to return the following information:

- ProductID
- Most expensive price
- Second most expensive price
- Percent difference from most expensive price to second most expensive (expressed as a two-digit decimal. For example, a 93% price difference will be displayed as 0.93).

CHALLENGE QUESTION 38: MARGINS

Difficulty: Intermediate

Create a query calculating the profit margins of bike models.

Notes:

- Profit margin is based on the percent difference between ListPrice and StandardCost
- Only consider bike models currently sold

Your output should contain the following columns, with models exhibiting the greatest profit margins listed first.

- ProductModelID
- Product name
- Profit margin (expressed as a two-digit decimal. For example, a 93% profit margin will be displayed as 0.93)

CHALLENGE QUESTION 39: PERCENT TO QUOTA

Difficulty: Intermediate

Each sales person is subject to a quarterly quota stated within the Sales.SalesPersonQuotaHistory table. The QuotaDate column represents the first date of the quota quarter.

PART I

Build a table to show the quota, actual sales, and percent to quota for each quarter and sales person. Store your data in a temporary table to be utilized in Part II.

Note: Do not include tax and freight with revenue.

Your results should contain the following columns:

- BusinessEntityID
- Quota date
- Sales quota
- Actual sales
- Percent to quota

Sort data by BusinessEntityID followed by quota date.

PART II

Summarize results from Part I by sales person, by year.

Include the following:

- Business Entity ID
- Quota year
- Total quota
- Total sales
- Total percent to quota
- Average quarterly percent to quota

Sort output by Business Entity ID followed by Quota year.

CHALLENGE QUESTION 40: REVENUE RANGES

Difficulty: Beginner

Based on sales data from 2005, calculate the number of sales orders within each of the following revenue ranges:

1. $0 - $100
2. $100 - $500
3. $500 - $1,000
4. $1,000 - $2,500
5. $2,500 - $5,000
6. $5,000 - $10,000
7. $10,000 - $50,000
8. $50,000 - $100,000
9. > $100,000

Notes:

- Revenue includes tax and freight
- Dates based on OrderDate

Create a SortID to sort your results in the sequence presented above. Example output:

SortID	SalesAmountCategory	Orders
1	$0 - $100	10
2	$100 - $500	10
3	$500 - $1,000	10
4	$1,000 - $2,500	10
5	$2,500 - $5,000	10
6	$5,000 - $10,000	10
7	$10,000 - $50,000	10
8	$50,000 - $100,000	10
9	> $100,000	10

CHALLENGE QUESTION 41: E-MAIL MYSTERY

Difficulty: Intermediate

A sales person was unable to locate a returning customer's account by e-mail address. Frustrated, he pulled up the account by the customer's last name. With the account information on his screen, he realized why his customer's e-mail address was not found. The customer's e-mail address appeared as an "adventure-works.com" address, rather than "gmail.com."

Examine the prevalence of adventure-works.com e-mail addresses throughout the company's database. Create a query by PersonType, with the following output:

- PersonType
- Number of e-mail addresses containing the adventure-works.com domain
- Number of e-mail addresses not containing the adventure-works.com domain
- Total number of e-mail addresses

Display your results by the greatest number of e-mail addresses to the fewest number of e-mail addresses.

CHALLENGE QUESTION 42: THE MENTORS

Difficulty: Intermediate

The Vice President of Sales wants the five most successful sales people to mentor the five least successful sales people. Create a list of sales people to match with one another.

Notes:

- Success is measured by 2008 revenue.
- Dates are based on OrderDate.
- Do not consider tax and freight with revenue.
- Ignore orders with no SalesPersonID.

Your output should contain the following columns:

- SalesPersonID of the successful sales person
- Revenue of the successful sales person
- SalesPersonID of the unsuccessful sales person
- Revenue of the successful sales person

For example, suppose the sales person with SalesPersonID 10 was the most successful sales person and her revenue was $1,000. If the sales person with SalesPersonID 20 was the least successful sales person and her revenue was $200, then the first row of your five-row output would appear as follows:

SuccessSalesPersonID	SuccessRevenue	UnsuccessSalesPersonID	UnsuccessRevenue
10	$1,000	20	$200

CHALLENGE QUESTION 43: CALENDAR OF WORK DAYS

Difficulty: Advanced

Calculate the number of work days in each year, without consideration for holidays, to help forecast energy costs. Use this exercise as an opportunity to create a comprehensive calendar of work days present in each year from January 1, 1990 to January 1, 2015 to be used for future purposes.

Save your calendar of work days as a table. Your table should contain the following columns:

- DateID (unique identifier)
- Date (for example, 1990-01-20 00:00:00.000)
- TextMonth (for example, January 1990)
- DateMonth (for example, 1990-01-01 00:00:00.000)
- DayOfWeek (for example, Monday)
- IsBusinessDay (0 or 1)

Based on the table you created, summarize the number of working days per year with the following columns:

- Year
- BusinessDays

CHALLENGE QUESTION 44: ANNUAL SALARY BY EMPLOYEE

Difficulty: Advanced

Create a query to show annual salary by employee from 2005 to 2008.

Assumptions:

- Today's date is January 1, 2009
- The Rate column within HumanResources.EmployeePayHistory represents hourly pay rates
- Each employee works eight hours per day, Monday through Friday

Notes:

- Holidays should not be considered
- You may utilize the calendar of work days created from Challenge Question 43.
- It would be helpful to utilize temporary tables; the next challenge question will incorporate information from this exercise

CHALLENGE QUESTION 45: ANNUAL SALARIES BY DEPARTMENT

Difficulty: Advanced

The Vice President of Human Resources reviewed your report about annual salaries from Challenge Question 44. He liked your work and he asked to see data from an additional perspective.

Write a query about annual salaries by department from 2008. Your output should include the following columns:

- DepartmentID
- Minimum salary
- Average salary
- Maximum salary

CHALLENGE QUESTION 46: HOLIDAY BONUS

Difficulty: Beginner

Human Resources will issue holiday bonuses to salaried employees. The bonus amount will equal current pay rate multiplied by 50. For example, an employee earning $10 per hour will receive a holiday bonus of $500.

Calculate holiday bonuses. Your output should include the following columns:

- BusinessEntityID
- FirstName
- LastName
- JobTitle
- Bonus

CHALLENGE QUESTION 47: COMPANY PICNIC

Difficulty: Intermediate

Name tags will be printed for a company picnic. You are asked to help prepare a list of employee names.

Create a query in which first names, last names, and suffixes are consolidated into one value, with a comma and a space separating the last name from the suffix. For example, if FirstName = David, LastName = Baez, and Suffix = Jr, the name would be consolidated as David Baez, Jr. If Suffix is NULL, the name appears as David Baez.

In all, your output should contain the following columns:

- BusinessEntityID
- Full name
- Department

Display the list alphabetically by department followed by full name.

CHALLENGE QUESTION 48: SALES QUOTA CHANGES

Difficulty: Intermediate

Management will review sales quota changes from 2006 through 2007. Create a report, by sales person, that includes the following information:

- BusinessEntityID
- LastName
- Sales quota from the start of 2006 (first quarter)
- Sales quota from the end of 2007 (last quarter)
- Percent change in sales quotas

For sound comparisons, do not include information about sales people who were not assigned sales quotas during the start of 2007 or the end of 2007.

CHALLENGE QUESTION 49: SCRAP RATE

Difficulty: Intermediate

The Production department is concerned about work orders in which scrap rates exceed 3%. Scrap rate equals scrapped quantity divided by order quantity.

Create a view that displays, by most recent due dates, the top 10% of work orders in which the scrap rate was greater than 3%, ordered by most recent due date.

Your view should contain the following:

- WorkOrderID
- DueDate
- ProductName
- Scrap reason
- Scrapped quantity
- Order quantity
- Percent scrapped

Example output:

WorkOrderID	DueDate	ProdName	ScrappedQty	OrderQty	PercScrapped
10000	2008-01-01	Blade	10	20	50.00

CHALLENGE QUESTION 50: REASONS

Difficulty: Intermediate

Adventure Works collects data on some customer's reasons for purchasing (seen on Sales.SalesOrderHeaderSalesReason). Sometimes, customers cite one reason, like "Price," for ordering a product. Other times, customers cite multiple reasons, like "Price" and "Quality."

Create a query about sales order reasons. When a sales order has only one reason, categorize as "Exclusive Reason." When a sales order has more than one reason, categorize as "Contributing Reason." Then, create a summary count of sales orders by reason name and your newly created ReasonInfluence column (Exclusive Reason or Contributing Reason).

Based on the directions stated above, your output will contain the following columns:

- ReasonName
- ReasonInfluence (Exclusive Reason or Contributing Reason)
- SalesOrderCount

CHALLENGE QUESTION 51: EXCESS INVENTORY

Difficulty: Intermediate

Occasionally, Adventure Works has excess inventory on some of its products. To sell these overstocked products quickly, the company creates special discounts.

PART I

To help choose the discount percentage to be applied, create a query about historical excess inventory discounts. Pull a list of the previous excess inventory discounts Adventure Works has created.

Your output should contain the following columns:

- SpecialOfferID
- Discount type (Excess Inventory)
- Discount description
- Discount category (Customer or Reseller)
- Discount start date
- Discount end date
- Discount percentage

PART II

Add an additional column to the output from Part I. List the number of sales orders in which the discount was utilized.

CHALLENGE QUESTION 52: PAY RATE CHANGES

Difficulty: Intermediate

Human Resources will review pay increases. For each employee, report the latest pay rate and the pay rate prior to the latest rate.

Your output should include the following columns:

- BusinessEntityID
- Previous rate (Pay rate prior to the latest rate)
- Latest pay rate
- Percent change from previous rate to latest pay rate. Express the percent increase with two digits followed by a percent sign. For example, 10.01%.

HINTS FOR CHALLENGE QUESTIONS

HINTS FOR CHALLENGE QUESTION 1: YEAR OVER YEAR COMPARISONS

Key Table: Sales.SalesOrderHeader

Key Table: Person.Person

Strategy Step 1:

- Organize sales by salesperson, fiscal year, and fiscal quarter
- Exclude online orders
- Store results in temp table

Strategy Step 2:

- Find sales of the same quarter of the previous fiscal year by joining to the temp table

HINTS FOR CHALLENGE QUESTION 2: THE 2/22 PROMOTION

Key Table: Sales.SalesOrderHeader

Key Table: Person.BusinessEntityAddress

Key Table: Person.Address

Key Table: Person.StateProvince

HINTS FOR CHALLENGE QUESTION 3: TEN MILLION DOLLAR BENCHMARK

Key Table: Sales.SalesOrderHeader

Key Column: OrderDate

Key Column: SubTotal

HINTS FOR CHALLENGE QUESTION 4: UPSELL TUESDAYS

Key Table: Sales.SalesOrderHeader

Key Column: OnlineOrderFlag

Key Function: DATENAME

HINTS FOR CHALLENGE QUESTION 5: EXPIRED CREDIT CARDS

Key Table: Sales.CreditCard

Key Function: EOMONTH

Key Function: DATEFROMPARTS

Key Table: Sales.SalesOrderHeader

HINTS FOR CHALLENGE QUESTION 6: PRINT CATALOG

Key Table: Production.Product

Key Column: SellEndDate

Key Column: FinishedGoodsFlag

Key Table: Production.ProductInventory

HINTS FOR CHALLENGE QUESTION 7: SPECIAL TEAM

Key Table: Sales.SalesOrderHeader

Key Table: Sales.SalesTerritoryHistory

Key Column: EndDate

Key Table: Person.Person

HINTS FOR CHALLENGE QUESTION 8: KNAPSACK PROBLEM

Strategy:

- Add salary data to results from Challenge Question 4
- Create three tables, one related to each territory
- CROSS JOIN tables
- Select best combination from eight possibilities

HINTS FOR CHALLENGE QUESTION 9: PRODUCT COMBINATIONS

Strategy: Build a temporary table with information about all sales orders. The table will be utilized throughout all parts of the solution. It should include at least the following columns:

- CustomerID
- SalesOrderID
- ProductType
- ProductLine
- ProductID

Key Table: Sales.SalesOrderHeader

Key Table: Sales.SalesOrderDetail

Key Table: Production.Product

Key Table: Production.ProductSubcategory

Key Table: Production.ProductCategory

Key Function: PIVOT

HINTS FOR CHALLENGE QUESTION 10: MEDIAN REVENUE

Key Table: Sales.SalesOrderHeader

Strategy:

- Create a row number for every order within each year
- Determine the maximum number of orders for each year
- Find the order with row number equal to half of the maximum number of orders

Key Function: PERCENTILE_DISC

HINTS FOR CHALLENGE QUESTION 11: NEEDY ACCOUNTANT

Key Table: Sales.SalesTaxRate

Key Table: Person.StateProvince

Key Table: Person.CountryRegion

HINTS FOR CHALLENGE QUESTION 12: PRODUCT INVENTORY UPDATES

Key Table: Production.ProductInventory

Key Function: GROUPING

Key Operator: ROLLUP

HINTS FOR CHALLENGE QUESTION 13: VACATION HOURS

Key Table: HumanResources.Employee

Key Function: RIGHT

HINTS FOR CHALLENGE QUESTION 14: PURCHASING

Key Table: Purchasing.PurchaseOrderDetail

Key Table: Production.Product

Key Table: Purchasing.PurchaseOrderHeader

HINTS FOR CHALLENGE QUESTION 15: INTERPRETATION NEEDED

Key Table: Production.ProductDescription

Key Table: Production.ProductModelProductDescriptionCulture

Key Table: Production.ProductModel

Key Table: Production.Culture

HINTS FOR CHALLENGE QUESTION 16: ONLINE/OFFLINE

Key Table: Sales.SalesOrderHeader

Key Column: OnlineOrderFlag

Key Function: CONVERT

Key Function: ROUND

HINTS FOR CHALLENGE QUESTION 17: LONG TIME NO SALE

Key Table: Sales.SalesOrderHeader

Key Table: Sales.Customer

Key Table: Sales.Store

Key Function: DATEDIFF

HINTS FOR CHALLENGE QUESTION 18: COSTS VARY

Key Table: Production.ProductCostHistory

Key Table: Production.Product

Key Table: Production.ProductSubcategory

Key Function: DENSE_RANK

HINTS FOR CHALLENGE QUESTION 19: THERMOFORM TEMPERATURE

Key Table: Production.WorkOrder

Key Table: Production.Product

Key Table: Production.ScrapReason

HINTS FOR CHALLENGE QUESTION 20: TORONTO

Key Table: Person.BusinessEntityAddress

Key Table: Person.Address

Key Table: Person.AddressType

Key Table: Sales.Store

HINTS FOR CHALLENGE QUESTION 21: MARKETING EMPLOYEES

Key Table: HumanResources.EmployeeDepartmentHistory

Key Column: EndDate (NULL indicates the current department)

Key Table: HumanResources.Department

Key Table: Person.Person

Key Table: HumanResources.Employee

HINTS FOR CHALLENGE QUESTION 22: WHO LEFT THAT REVIEW?

Key Table: Production.ProductReview

Key Table: Production.Product

Key Table: Person.EmailAddress

HINTS FOR CHALLENGE QUESTION 23: LABEL MIX-UP

Key Table: Sales.SalesOrderDetail

Key Table: Production.Product

Key Table: Sales.SalesOrderHeader

Key Table: Sales.Customer

Key Table: Person.Person

Key Table: Person.PersonPhone

HINTS FOR CHALLENGE QUESTION 24: CLEARANCE SALE

Key Table: Person.EmailAddress

Key Table: Person.Person

Key Column: PersonType

Key Value: IN (to locate individual retail customers)

HINTS FOR CHALLENGE QUESTION 25: TOP TERRITORIES

Key Table: Sales.SalesOrderHeader

Key Table: Sales.SalesTerritory

Key Function: DENSE_RANK

HINTS FOR CHALLENGE QUESTION 26: COMMISSION PERCENTAGES

Key Table: Sales.SalesPerson

Key Function: DENSE_RANK

HINTS FOR CHALLENGE QUESTION 27: WORK ORDERS

Key Table: Production.WorkOrder

HINTS FOR CHALLENGE QUESTION 28: REVENUE TRENDED

Key Table: Sales.SalesOrderHeader

Key Column: Subtotal

Key Function: EOMONTH

HINTS FOR CHALLENGE QUESTION 29: SEPARATION

Key Table: HumanResources.Employee

Key Function: LEFT

Key Function: CHARINDEX

Key Function: RIGHT

Key Function: LEN

HINTS FOR CHALLENGE QUESTION 30: SHIFT COVERAGE

Key Table: HumanResources.EmployeeDepartmentHistory

Key Column: EndDate

Key Table: HumanResources.Department

Key Table: HumanResources.Shift

HINTS FOR CHALLENGE QUESTION 31: LABELS

PART I

Key Table: Production.Product

Key Function: ISNUMERIC

PART II

Key Table: Production.Product

Key Table: Production.ProductInventory

HINTS FOR CHALLENGE QUESTION 32: EMPLOYMENT SURVEY

Key Table: HumanResources.Employee

Key Function: DATEDIFF

Key Function: NTILE

HINTS FOR CHALLENGE QUESTION 33: AGE GROUPS

Key Table: HumanResources.Employee

Key Function: DATEDIFF

Key Table: HumanResources.EmployeePayHistory

HINTS FOR CHALLENGE QUESTION 34: REVENUE BY STATE

Key Table: Sales.SalesOrderHeader

Key Table: Person.Address

Key Table: Person.StateProvince

HINTS FOR CHALLENGE QUESTION 35: TWO FREE BIKES

Key Table: HumanResources.Employee.

Key Column: OrganizationalLevel. Interpret the values of OrganizationalLevel appropriately by examining the values of various senior-level job titles.

Key Table: Person.Person

HINTS FOR CHALLENGE QUESTION 36: VOLUME DISCOUNTS

Key Table: Sales.SalesOrderDetail

Key Table: Sales.SpecialOffer

Key Column: Type

Key Table: Sales.SalesOrderHeader

HINTS FOR CHALLENGE QUESTION 37: OVERPAYING

Key Table: Purchasing.ProductVendor

Key Column: LastReceiptCost

HINTS FOR CHALLENGE QUESTION 38: MARGINS

Key Table: Production.Product

Key Column: SellEndDate

Key Table: Production.ProductSubcategory

Key Table: Production.ProductCategory

Key Table: Production.ProductModel

HINTS FOR CHALLENGE QUESTION 39: PERCENT TO QUOTA

Key Table: Sales.SalesPersonQuotaHistory

Key Column: QuotaDate

Key Table: Sales.SalesOrderHeader

Key Consideration: Search for sales throughout the entire quarter. Since Quota Date represents the start of each quarter, the function DATEADD can be used to extend the date range.

HINTS FOR CHALLENGE QUESTION 40: REVENUE RANGES

Key Table: Sales.SalesOrderHeader

Key Column: TotalDue

Key Expression: CASE

HINTS FOR CHALLENGE QUESTION 41: E-MAIL MYSTERY

Key Table: Person.EmailAddress

Key Table: Person.Person

HINTS FOR CHALLENGE QUESTION 42: THE MENTORS

Key Table: Sales.SalesOrderHeader

Key Column: SalesPersonID

Key Column: Subtotal

Key Function: ROW_NUMBER

HINTS FOR CHALLENGE QUESTION 43: CALENDAR OF WORK DAYS

Strategy to create the table about work days per year:

- Create a shell table with the necessary columns
- Create a loop that runs for the same amount of times as rows needed in the table
- Update table with details about each day

Key Functions:

- DATENAME
- DATEADD

HINTS FOR CHALLENGE QUESTION 44: ANNUAL SALARY BY EMPLOYEE

As shown below, some BusinessEntityID's within the HumanResources.EmployeePayHistory table appear on multiple rows:

```
SELECT BusinessEntityID
FROM HumanResources.EmployeePayHistory
GROUP BY BusinessEntityID
HAVING COUNT (*) > 1
```

Each row may represent a pay rate change for employees with multiple pay rates over time.

Strategy:

- Identify the start date, end date, and the number of business days corresponding to each employee's range of dates for each pay rate.
- To count business days, utilize the calendar table you created through the previous challenge.

HINTS FOR CHALLENGE QUESTION 45: ANNUAL SALARIES BY DEPARTMENT

- You must account for the possibility that an employee can switch departments
- Key Table: HumanResources.EmployeeDepartmentHistory
- Utilize the calendar of work days from previous challenge to find data about employee pay

HINTS FOR CHALLENGE QUESTION 46: HOLIDAY BONUS

Key Table: HumanResources.Employee

Key Column: SalariedFlag

Key Table: HumanResources.EmployeePayHistory

Key Table: Person.Person

HINTS FOR CHALLENGE QUESTION 47: COMPANY PICNIC

Key Table: Person.Person

Key Column: PersonType

Key Values: SP (Sales person), EM (non-sales employee)

Key Table: HumanResources.EmployeeDepartmentHistory

Key Table: HumanResources.Department

HINTS FOR CHALLENGE QUESTION 48: SALES QUOTA CHANGES

Key Table: Sales.SalesQuotaPersonHistory

Key Table: Person.Person

HINTS FOR CHALLENGE QUESTION 49: SCRAP RATE

Key Table: Production.WorkOrder

Key Table: Production.ScrapReason

Key Table: Production.Product

HINTS FOR CHALLENGE QUESTION 50: REASONS

Key Table: Sales.SalesOrderHeaderSalesReason

Key Table: Sales.SalesReason

HINTS FOR CHALLENGE QUESTION 51: EXCESS INVENTORY

Key Table: Sales.SpecialOffer

Key Table: Sales.SalesOrderDetail

HINTS FOR CHALLENGE QUESTION 52: PAY RATE CHANGES

Key Table: HumanResources.EmployeePayHistory

Key Column: RateChangeDate

SOLUTIONS TO CHALLENGE QUESTIONS

Solutions to challenge questions available for download at http://realsqlqueries.com.

SOLUTION TO CHALLENGE QUESTION 1: YEAR OVER YEAR COMPARISONS

```
—DROP TABLE #data

SELECT
  SalesPersonID
  ,FY =
     DATEPART (YEAR,
       DATEADD (MONTH, 6, OrderDate))
  ,FQ = DATEPART (QUARTER,
       DATEADD (MONTH, 6, OrderDate))
  ,FQSales = SUM (Subtotal)
INTO #data
FROM Sales.SalesOrderHeader
WHERE OnlineOrderFlag = 0
GROUP BY
  SalesPersonID
  ,DATEPART (YEAR,
    DATEADD (MONTH, 6, OrderDate))
  ,DATEPART (QUARTER,
    DATEADD (MONTH, 6, OrderDate))

SELECT
  N3.LastName
  ,N1.*
  ,SalesSameFQLastYr =   N2.FQSales
  ,Change = N1.FQSales - N2.FQSales
  ,[%Change] =
    ((N1.FQSales - N2.FQSales)
    / N2.FQSales) * 100
FROM #data N1
LEFT JOIN #Data N2
  ON N1.SalesPersonID = N2.SalesPersonID
    AND N1.FQ = N2.FQ
    AND N1.FY -1 = N2.FY
INNER JOIN Person.Person N3
  ON N1.SalesPersonID = N3.BusinessEntityID
WHERE N1.FY = 2008
ORDER BY
  SalesPersonID, FY DESC, FQ DESC
```

SOLUTION TO CHALLENGE QUESTION 2: THE 2/22 PROMOTION

```
—   Part I

—DROP TABLE #data

SELECT
  N1.SalesOrderID
  ,ShipToState = N4.Name
  ,OrderDate = N1.OrderDate
  ,[HistoricalOrder$] = N1.SubTotal
  ,HistoricalFreight = N1.Freight
  ,PotentialPromoEffect =
    CASE
      WHEN N1.SubTotal > = 1700
        AND N1.SubTotal < 2000
        THEN 'INCREASE ORDER TO $2,000 & PAY 22 CENTS FREIGHT'
      WHEN N1.Subtotal > = 2000
        THEN 'NO ORDER CHANGE AND PAY 22 CENTS FREIGHT'
      ELSE 'NO ORDER CHANGE & PAY HISTORICAL FREIGHT'
    END
  ,PotentialOrderGain =
    CASE
      WHEN N1.SubTotal > = 1700
        AND N1.SubTotal < 2000
        THEN 2000 - N1.SubTotal
      ELSE 0
    END
  ,PotentialFreightLoss =
    CASE
      WHEN N1.SubTotal > =  1700
        THEN 0.22
        ELSE N1.Freight
      END
    - N1.Freight
  ,[PromoNetGain/Loss] =
    CASE
      WHEN N1.SubTotal > = 1700
        AND N1.SubTotal < 2000
        THEN 2000 - N1.SubTotal
      ELSE 0
    END
    +   CASE
        WHEN N1.SubTotal > =  1700
```

```
            THEN 0.22
          ELSE N1.Freight
        END
  - N1.Freight
INTO #data
FROM Sales.SalesOrderHeader N1
INNER JOIN Person.BusinessEntityAddress N2
  ON N1.ShipToAddressID = N2.AddressID
INNER JOIN Person.[Address] N3
  ON N2.AddressID = N3.AddressID
INNER JOIN Person.StateProvince N4
  ON N3.StateProvinceID = N4.StateProvinceID
WHERE N4.Name = 'California'
  AND DATEPART (YEAR,
    DATEADD (MONTH,
      6, N1.OrderDate)) = 2008

SELECT *
FROM #data

—   Part II
SELECT
  PotentialPromoEffect
  ,PotentialOrderGains = SUM (PotentialOrderGain)
  ,PotentialFreightLosses = SUM (PotentialFreightLoss)
  ,OverallNet = SUM ([PromoNetGain/Loss])
FROM #data
GROUP BY PotentialPromoEffect
```

SOLUTION 1 TO CHALLENGE QUESTION 3: TEN MILLION DOLLAR BENCHMARK

```
—DROP TABLE #Sales

SELECT
  FiscalYear =
    YEAR (DATEADD (MONTH, 6, OrderDate))
  ,OrderDate =
    CAST (OrderDate AS DATE)
  ,OrderNumber =
    ROW_NUMBER () OVER
      (PARTITION BY YEAR (
        DATEADD (MONTH, 6, OrderDate))
      ORDER BY OrderDate)
  ,SubTotal
  ,RunningTotal =   CONVERT (FLOAT, NULL)
INTO #Sales
FROM Sales.SalesOrderHeader

UPDATE N1 SET
RunningTotal =
  (SELECT SUM (SubTotal)
  FROM #Sales X1
  WHERE N1.FiscalYear = X1.FiscalYear
  AND X1.OrderNumber < = N1.OrderNumber)
FROM #Sales N1

—DROP TABLE #FindOrder

SELECT
  FiscalYear
  ,OrderNumberOver10M =
    (SELECT TOP 1 X1.OrderNumber
    FROM #Sales X1
    WHERE N1.FiscalYear = X1.FiscalYear
      AND X1.RunningTotal > = 10000000
    ORDER BY X1.RunningTotal)
INTO #FindOrder
FROM #Sales N1
GROUP BY N1.FiscalYear

SELECT
```

```
   N2.FiscalYear
   ,N2.OrderDate
   ,N2.OrderNumber
   ,N2.RunningTotal
FROM #FindOrder N1
INNER JOIN #Sales N2
   ON N1.FiscalYear = N2.FiscalYear
      AND N1.OrderNumberOver10M =
         N2.OrderNumber
WHERE N1.FiscalYear IN (2007, 2008)
```

SOLUTION 2 TO CHALLENGE QUESTION 3: TEN MILLION DOLLAR BENCHMARK

```
WITH FY2007 AS
  (SELECT
    FY = 2007
    ,OrderDate = CAST (OrderDate AS DATE)
    ,[OrderNumber] = ROW_NUMBER ()
      OVER (ORDER BY SalesOrderID)
    ,RunningTotal =   SUM (SubTotal)
      OVER (ORDER BY Orderdate
        ROWS BETWEEN UNBOUNDED
          PRECEDING AND CURRENT ROW)
    FROM Sales.SalesOrderHeader
    WHERE DATEPART (YEAR,
      DATEADD (MONTH, 6, OrderDate)) = 2007)

  ,FY2008 AS
    (SELECT
      FY = 2008
      ,OrderDate = CAST (OrderDate AS DATE)
      ,[OrderNumber] = ROW_NUMBER ()
        OVER (ORDER BY SalesOrderID)
      ,RunningTotal =   SUM (SubTotal)
        OVER (ORDER BY Orderdate
          ROWS BETWEEN UNBOUNDED
            PRECEDING AND CURRENT ROW)
      FROM Sales.SalesOrderHeader
      WHERE DATEPART (YEAR,
        DATEADD (MONTH, 6, OrderDate)) = 2008)

SELECT TOP 1 *
FROM FY2007
WHERE RunningTotal > = 10000000

UNION

SELECT TOP 1 *
FROM FY2008
WHERE RunningTotal > = 10000000
```

SOLUTION TO CHALLENGE QUESTION 4: UPSELL TUESDAYS

```
SELECT
   DayCategory =
      DATENAME (WEEKDAY, OrderDate)
   ,Revenue = SUM (Subtotal)
   ,Orders =   COUNT (*)
   ,RevenuePerOrder =
      SUM (Subtotal) / COUNT (*)
FROM Sales.SalesOrderHeader
WHERE YEAR (OrderDate) = 2008
   AND OnlineOrderFlag = 0
GROUP BY DATENAME (WEEKDAY, OrderDate)
ORDER BY RevenuePerOrder DESC
```

SOLUTION TO CHALLENGE QUESTION 5: EXPIRED CREDIT CARDS

— Part I

—DROP TABLE #data

```
SELECT
   N1.CreditCardID
   ,N1.CardType
   ,ExpDate =
      EOMONTH (
         DATEFROMPARTS (
            N1.ExpYear, N1.ExpMonth, 1))
   ,LastOrderDate =
      CAST (N2.LastOrderDate AS DATE)
   ,[Orders<=Exp] =
      COUNT (DISTINCT N3.SalesOrderID)
   ,[Orders>Exp] =
      COUNT (DISTINCT N4.SalesOrderID)
INTO #data
FROM Sales.CreditCard N1
LEFT JOIN
   (SELECT
      X1.CreditCardID
      ,LastOrderDate = MAX (X1.OrderDate)
   FROM Sales.SalesOrderHeader X1
   GROUP BY X1.CreditCardID) N2
   ON N1.CreditCardID = N2.CreditCardID
LEFT JOIN Sales.SalesOrderHeader N3
   ON N1.CreditCardID = N3.CreditCardID
      AND N3.OrderDate < = EOMONTH (
         DATEFROMPARTS (
            N1.ExpYear, N1.ExpMonth, 1))
LEFT JOIN Sales.SalesOrderHeader N4
   ON N1.CreditCardID = N4.CreditCardID
      AND N4.OrderDate >  EOMONTH (
         DATEFROMPARTS (
            N1.ExpYear, N1.ExpMonth, 1))
GROUP BY
   N1.CreditCardID
   ,N1.CardType
   ,N2.LastOrderDate
```

```
   ,N1.ExpYear
   ,N1.ExpMonth

SELECT *
FROM #data
ORDER BY [Orders>Exp] DESC

—Part II
SELECT
   CardType
   ,[Orders<=Exp] = SUM ([Orders<=Exp])
   ,[Orders>Exp] =   SUM ([Orders>Exp])
FROM #data
GROUP BY CardType
```

SOLUTION TO CHALLENGE QUESTION 6: PRINT CATALOG

```
SELECT
   N1.ProductID
   ,ProductName = N1.Name
   ,N1.Color
   ,N1.ListPrice
   ,N2.TotalQty
FROM Production.Product N1
INNER JOIN
   (SELECT
      ProductID
      ,TotalQty = SUM (Quantity)
   FROM Production.ProductInventory
   GROUP BY ProductID) N2 ON N1.ProductID = N2.ProductID
WHERE N1.SellEndDate IS NULL
   AND N2.TotalQty > = 150
   AND N1.ListPrice > = 1500
   AND N1.FinishedGoodsFlag = 1
```

SOLUTION TO CHALLENGE QUESTIONS 7: SPECIAL TEAM

```
—DROP TABLE #2008RevbySalesPerson

SELECT
   SalesPersonID
   ,[2008Sales] = SUM (Subtotal)
INTO #2008RevbySalesPerson
FROM Sales.SalesOrderHeader N1
WHERE YEAR (N1.OrderDate) = 2008
GROUP BY SalesPersonID

—DROP TABLE #TerritoriesAndSales

SELECT
   N1.BusinessEntityID
   ,N3.LastName
   ,Territory = N2.Name
   ,N4.[2008Sales]
INTO #TerritoriesAndSales
FROM Sales.SalesTerritoryHistory N1
INNER JOIN Sales.SalesTerritory N2
   ON N1.TerritoryID = N2.TerritoryID
INNER JOIN Person.Person N3
   ON N1.BusinessEntityID = N3.BusinessEntityID
INNER JOIN #2008RevbySalesPerson N4
   ON N1.BusinessEntityID = N4.SalesPersonID
WHERE N1.EndDate IS NULL
   AND N2.Name IN
      ('Northwest', 'Southwest', 'Canada')

SELECT *
FROM #TerritoriesAndSales
ORDER BY Territory, LastName
```

SOLUTION TO CHALLENGE QUESTION 8: KNAPSACK PROBLEM

The solution to challenge question 8 begins with the solution to challenge question 7.

```
—DROP TABLE #2008RevbySalesPerson
SELECT
   SalesPersonID
   ,[2008Sales] = SUM (Subtotal)
INTO #2008RevbySalesPerson
FROM Sales.SalesOrderHeader N1
WHERE YEAR (N1.OrderDate) = 2008
GROUP BY SalesPersonID

—DROP TABLE #TerritoriesAndSales

SELECT
   N1.BusinessEntityID
   ,N3.LastName
   ,Territory = N2.Name
   ,N4.[2008Sales]
INTO #TerritoriesAndSales
FROM Sales.SalesTerritoryHistory N1
INNER JOIN Sales.SalesTerritory N2
   ON N1.TerritoryID = N2.TerritoryID
INNER JOIN Person.Person N3
   ON N1.BusinessEntityID = N3.BusinessEntityID
INNER JOIN #2008RevbySalesPerson N4
   ON N1.BusinessEntityID = N4.SalesPersonID
WHERE N1.EndDate IS NULL
   AND N2.Name IN
      (‘Northwest’, ‘Southwest’, ‘Canada’)

SELECT *
FROM #TerritoriesAndSales
ORDER BY Territory, LastName

ALTER TABLE #TerritoriesAndSales
ADD Salary MONEY

UPDATE N1
SET Salary =
   CASE
```

```
      WHEN LastName = 'Pak'
        THEN 79500
      WHEN LastName = 'Vargas'
        THEN  60000
      WHEN LastName = 'Campbell'
        THEN 59500
      WHEN LastName = 'Mensa-Annan'
        THEN 56000
      WHEN LastName = 'Ito'
        THEN 68000
      WHEN LastName = 'Mitchell'
        THEN 80000 END
FROM #TerritoriesAndSales N1

—DROP TABLE #Canada
—DROP TABLE #Northwest
—DROP TABLE #Southwest

SELECT *
INTO #Canada
FROM #TerritoriesAndSales
WHERE Territory = 'Canada'

SELECT *
INTO #Northwest
FROM #TerritoriesAndSales
WHERE Territory = 'Northwest'

SELECT *
INTO #Southwest
FROM #TerritoriesAndSales
WHERE Territory = 'Southwest'

—DROP TABLE #final
SELECT
  AggregateSalary =
    N1.Salary + N2.Salary + N3.Salary
  ,AggregateSales =
    N1.[2008Sales] + N2.[2008Sales] + N3.[2008Sales]
  ,[1stTerritory] =    N1.Territory
  ,[1stSalesPerson] =   N1.LastName
  ,[1stSalesPersonSales] = N1.[2008Sales]
  ,[1stSalary] = N1.Salary
  ,[2ndTerritory] = N2.Territory
  ,[2ndSalesPerson] = N2.LastName
```

```
   ,[2ndSalesPersonSales] =   N2.[2008Sales]
   ,[2ndSalary] = N2.Salary
   ,[3rdTerritory] =    N3.Territory
   ,[3rdSalesPerson] =    N3.LastName
   ,[3rdSalesPersonSales] = N3.[2008Sales]
   ,[3rdSalary] = N3.Salary
INTO #final
FROM #Canada N1
CROSS JOIN #Northwest N2
CROSS JOIN #Southwest N3

SELECT TOP 1 *
FROM #final
WHERE AggregateSalary < 210000
ORDER BY AggregateSales DESC
```

SOLUTION TO CHALLENGE QUESTION 9: PRODUCT COMBINATIONS

```
—Temp table to be utilized throughout all parts of the solution

—DROP TABLE #ProductSales

SELECT
   N1.CustomerID
   ,N1.SalesOrderID
   ,ProductType = N5.Name
   ,N3.ProductLine
   ,N3.ProductID
INTO #ProductSales
FROM Sales.SalesOrderHeader N1
INNER JOIN Sales.SalesOrderDetail N2
   ON N1.SalesOrderID = N2.SalesOrderID
INNER JOIN Production.Product N3
   ON N2.ProductID = N3.ProductID
INNER JOIN Production.ProductSubcategory N4
   ON N3.ProductSubcategoryID = N4.ProductSubcategoryID
INNER JOIN Production.ProductCategory N5
   ON N4.ProductCategoryID = N5.ProductCategoryID

— Part I

DECLARE @TotalOrders FLOAT =
   (SELECT
      COUNT (DISTINCT SalesOrderID)
   FROM #ProductSales)

DECLARE @BikeAccessoryOrders FLOAT =
   (SELECT
      COUNT (DISTINCT N1.SalesOrderID)
   FROM #ProductSales N1
   INNER JOIN #ProductSales N2
      ON N1.SalesOrderID = N2.SalesOrderID
   WHERE N1.ProductType = 'Bikes'
      AND N2.ProductType = 'Accessories')

SELECT BikeAndAccessory =
   CONVERT (VARCHAR(10),
      CONVERT (DECIMAL (5,2),
```

```
        (@BikeAccessoryOrders / @TotalOrders)
          * 100)) + ' %'

DECLARE @BikeClothingOrders FLOAT =
  (SELECT COUNT (*)
  FROM
    (SELECT SalesOrderID
    FROM #ProductSales
    GROUP BY SalesOrderID
    HAVING
      SUM (CASE WHEN ProductType = 'Bikes'
                THEN 1 ELSE 0 END) > = 1
      AND SUM
          (CASE WHEN ProductType = 'Clothing'
                  THEN 1
                ELSE 0 END) > = 2)
  X1)

SELECT
  BikeAndClothing =
    CONVERT (VARCHAR(10),
      CONVERT (DECIMAL (5,2),
        (@BikeClothingOrders / @TotalOrders)
          * 100)) + ' %'

— Part II

—DROP TABLE #Pivot

SELECT *
INTO #Pivot
FROM
  (SELECT DISTINCT
    SalesOrderID
    ,ProductType
    ,Cnt = 1
  FROM #ProductSales) N1

PIVOT
  (COUNT (Cnt)
  FOR ProductType IN
    ([Bikes], [Accessories]
    ,[Clothing], [Components])
  ) X1
```

```
SELECT
   Bikes
   ,Accessories
   ,Clothing
   ,Components
   ,Orders = COUNT(*)
FROM #Pivot
GROUP BY Bikes, Accessories, Clothing, Components
ORDER BY Bikes, Accessories, Clothing, Components

— Part III

—DROP TABLE #Pivot2

SELECT *
INTO #Pivot2
FROM
   (SELECT DISTINCT
      CustomerID
      ,ProductLine, Cnt = 1
   FROM #ProductSales) N1
PIVOT
   (COUNT (Cnt)
   FOR ProductLine IN ([M],[S],[T],[R])
   ) X1

SELECT M, S, T, R, Customers = COUNT (*)
FROM #Pivot2
GROUP BY M, S, T, R
ORDER BY M, S, T, R
```

SOLUTION 1 TO CHALLENGE QUESTION 10: MEDIAN REVENUE

```
—DROP TABLE #Sales

SELECT
  OrderYear = YEAR (OrderDate)
  ,SubTotal
  ,RowNumbForMedian =
    ROW_NUMBER () OVER
      (PARTITION BY YEAR (OrderDate)
      ORDER BY SubTotal)
INTO #Sales
FROM Sales.SalesOrderHeader

—DROP TABLE #SalesGrouped

SELECT
  OrderYear
  ,NumbOrders = COUNT (*)
  ,NumbOrdersEven   =
    CASE
      WHEN COUNT (*) % 2 = 0
        THEN 1
      ELSE 0
    END
  ,FindMedian   = (COUNT (*) / 2) + 1
  ,Median   = CONVERT (FLOAT, NULL)
INTO #SalesGrouped
FROM #Sales
GROUP BY OrderYear

UPDATE N1 SET
Median = N2.SubTotal
FROM #SalesGrouped N1
INNER JOIN #Sales N2
  ON N1.OrderYear = N2.OrderYear
    AND N1.FindMedian = N2.RowNumbForMedian
WHERE NumbOrdersEven = 0

UPDATE N1 SET
  Median =
    (SELECT AVG (SubTotal)
    FROM #Sales X1
    WHERE N1.OrderYear = X1.OrderYear
```

```
        AND X1.RowNumbForMedian IN
          (N1.FindMedian, (N1.FindMedian - 1)))
  FROM #SalesGrouped N1
  WHERE NumbOrdersEven = 1

SELECT
  N1.OrderYear
  ,MinSale = MIN (SubTotal)
  ,MaxSale = MAX (SubTotal)
  ,AvgSale = AVG (SubTotal)
  ,MedianSale = N2.Median
FROM #Sales N1
INNER JOIN #SalesGrouped N2
  ON N1.OrderYear = N2.OrderYear
GROUP BY N1.OrderYear, N2.Median
ORDER BY N1.OrderYear
```

SOLUTION 2 TO CHALLENGE QUESTION 10: MEDIAN REVENUE

```
WITH MedianSales AS
  (SELECT DISTINCT
    OrderYear = YEAR (OrderDate)
    ,MedianSale =
      PERCENTILE_DISC (0.5)
        WITHIN GROUP (
          ORDER BY Subtotal)
          OVER (PARTITION BY YEAR (
            OrderDate))
  FROM Sales.SalesOrderHeader)

SELECT
  OrderYear = YEAR (N1.OrderDate)
  ,MinSale = MIN (N1.SubTotal)
  ,MaxSale = MAX (N1.SubTotal)
  ,AvgSale = AVG (N1.SubTotal)
  ,N2.MedianSale
FROM Sales.SalesOrderHeader N1
INNER JOIN MedianSales N2
  ON YEAR (N1.OrderDate) = N2.OrderYear
GROUP BY
  YEAR (N1.OrderDate)
  ,N2.MedianSale
ORDER BY YEAR (N1.OrderDate)
```

SOLUTION TO CHALLENGE QUESTION 11: NEEDY ACCOUNTANT

```
SELECT
   Country = N3.Name
   ,MaxTaxRate = MAX (N1.TaxRate)
FROM Sales.SalesTaxRate N1
INNER JOIN Person.StateProvince N2
   ON N1.StateProvinceID = N2.StateProvinceID
INNER JOIN Person.CountryRegion N3
   ON N2.CountryRegionCode = N3.CountryRegionCode
GROUP BY N3.Name
```

SOLUTION TO CHALLENGE QUESTION 12: PRODUCT INVENTORY UPDATES

```
—DROP VIEW Production.Vw_Product_Inventory

CREATE VIEW Production.Vw_Product_Inventory

AS

SELECT
  LocationID =
    CASE
      WHEN GROUPING (LocationID) = 1
        THEN CONVERT (VARCHAR (5), 'Total')
      ELSE CONVERT (VARCHAR (5), LocationID)
    END
  ,DistinctProducts =
    COUNT (DISTINCT ProductID)
  ,Quantity = SUM (Quantity)
FROM Production.ProductInventory
GROUP BY LocationID WITH ROLLUP
```

SOLUTION TO CHALLENGE QUESTION 13: VACATION HOURS

```
WITH MaxVacHrs AS
  (SELECT
    MaxVacHrs =
      MAX (VacationHours)
    FROM HumanResources.Employee)

SELECT
  NationalID =
    RIGHT (N1.NationalIDNumber, 4)
  ,N2.FirstName
  ,N2.LastName
  ,N1.JobTitle
  ,N1.VacationHours
FROM HumanResources.Employee N1
INNER JOIN Person.Person N2
  ON N1.BusinessEntityID = N2.BusinessEntityID
INNER JOIN MaxVacHrs N3
  ON N1.VacationHours = N3.MaxVacHrs
```

SOLUTION TO CHALLENGE QUESTION 14: PURCHASING

```
SELECT
   N1.ProductID
   ,ProductName = N2.Name
   ,N3.OrderDate
   ,QuantityOrdered = SUM (N1.OrderQty)
FROM Purchasing.PurchaseOrderDetail N1
INNER JOIN Production.Product N2
   ON N1.ProductID = N2.ProductID
INNER JOIN Purchasing.PurchaseOrderHeader N3
   ON N1.PurchaseOrderID = N3.PurchaseOrderID
WHERE YEAR (N3.OrderDate) = 2007
GROUP BY
   N1.Productid
   ,N2.Name
   ,N3.OrderDate
ORDER BY SUM (N1.OrderQty) DESC
```

SOLUTION TO CHALLENGE QUESTION 15: INTERPRETATION NEEDED

```
SELECT
   N2.ProductModelID
   ,ProductModel = N3.Name
   ,N1.[Description]
   ,[Language] = N4.Name
FROM Production.ProductDescription N1
INNER JOIN Production.ProductModelProductDescriptionCulture N2
   ON N1.ProductDescriptionID = N2.ProductDescriptionID
INNER JOIN Production.ProductModel N3
   ON N2.ProductModelID = N3.ProductModelID
INNER JOIN Production.Culture N4
   ON N2.CultureID = N4.CultureID
WHERE N4.Name <> 'English'
```

SOLUTION TO CHALLENGE QUESTION 16: ONLINE/OFFLINE

```
SELECT
  TerritoryID
  ,TotalOrders    = COUNT (*)
  ,PercOnline =
    CONVERT (VARCHAR (50),
      ROUND (
        (CONVERT (FLOAT,
          SUM (CASE
                WHEN OnlineOrderFlag = 1
                  THEN 1
                ELSE 0
              END))
          / COUNT (*))
          * 100
        ,0)
    ) + '%'
  ,PercOffline =
    CONVERT (VARCHAR (50),
      ROUND (
        (CONVERT (FLOAT,
          SUM (CASE
                WHEN OnlineOrderFlag = 0
                  THEN 1
                ELSE 0 END))
          / COUNT (*))
          * 100
        ,0)
    ) + '%'

FROM Sales.SalesOrderHeader
GROUP BY TerritoryID
ORDER BY TerritoryID
```

SOLUTION TO CHALLENGE QUESTION 17: LONG TIME NO SALE

```
WITH Stores AS
  (SELECT
    N3.BusinessEntityID
    ,N1.CustomerID
    ,N2.StoreID
    ,StoreName = N3.Name
    ,LastOrderDate = MAX (N1.OrderDate)
    ,MonthsSinceLastOrder =
      DATEDIFF (MONTH
        ,MAX (N1.OrderDate), '2008-10-07')
FROM Sales.SalesOrderHeader N1
INNER JOIN Sales.Customer N2
  ON N1.CustomerID = N2.CustomerID
INNER JOIN Sales.Store N3
  ON N2.StoreID = N3.BusinessEntityID
GROUP BY
  N3.BusinessEntityID
  ,N2.StoreID
  ,N1.CustomerID
  ,N3.Name)

SELECT *
FROM Stores
WHERE MonthsSinceLastOrder > = 12
ORDER BY MonthsSinceLastOrder DESC
```

SOLUTION TO CHALLENGE QUESTION 18: COSTS VARY

```
SELECT
  N1.ProductID
  ,ProductName = N2.Name
  ,SubCategory = N3.Name
  ,MinCost = MIN (N1.StandardCost)
  ,MaxCost = MAX (N1.StandardCost)
  ,CostVar =
    MAX (N1.StandardCost)
    - MIN (N1.StandardCost)
  ,CostVarRank =
    CASE
      WHEN MAX (N1.StandardCost)
        - MIN (N1.StandardCost) = 0
        THEN 0
      ELSE DENSE_RANK ()
        OVER (ORDER BY
          MAX (N1.StandardCost)
          - MIN (N1.StandardCost) DESC)
    END
FROM Production.ProductCostHistory N1
INNER JOIN Production.Product N2
  ON N1.ProductID = N2.ProductID
INNER JOIN Production.ProductSubcategory N3
  ON N2.ProductSubcategoryID = N3.ProductSubcategoryID
GROUP BY
  N1.ProductID
  ,N2.Name
  ,N3.Name
ORDER BY
  MAX (N1.StandardCost)
  - MIN (N1.StandardCost) DESC
```

SOLUTION TO CHALLENGE QUESTION 19: THERMOFORM TEMPERATURE

```
WITH Temp AS
  (SELECT
    RNK =
      ROW_NUMBER () OVER
        (PARTITION BY
          N1.ProductID
        ORDER BY
          COUNT (N2.Name) DESC)
    ,N1.ProductID
    ,ProductName = N2.Name
    ,WorkOrderCount = COUNT (N2.Name)
    ,ScrapReason =   N3.Name
  FROM Production.WorkOrder N1
  INNER JOIN Production.Product N2
    ON N1.ProductID = N2.ProductID
  INNER JOIN Production.ScrapReason N3
    ON N1.ScrapReasonID = N3.ScrapReasonID
  GROUP BY N1.ProductID, N2.Name, N3.Name)

SELECT
  ProductID
  ,ProductName
  ,WorkOrderCount
  ,ScrapReason
FROM Temp
WHERE RNK = 1
ORDER BY WorkOrderCount DESC
```

SOLUTION TO CHALLENGE QUESTION 20: TOTONTO

```
SELECT
   AddressType    = N3.Name
   ,StoreName = N4.Name
   ,N2.AddressLine1
   ,N2.AddressLine2
   ,N2.City
   ,StateProvince = N5.Name
   ,N2.PostalCode
FROM Person.BusinessEntityAddress N1
INNER JOIN Person.[Address] N2
   ON N1.AddressID = N2.AddressID
INNER JOIN Person.AddressType N3
   ON N1.AddressTypeID = N3.AddressTypeID
INNER JOIN Sales.Store N4
   ON N1.BusinessEntityID = N4.BusinessEntityID
INNER JOIN Person.StateProvince N5
   ON N2.StateProvinceID = N5.StateProvinceID
WHERE N3.Name = ‘Main office’ and N2.City = ‘Toronto’
```

SOLUTION TO CHALLENGE QUESTION 21: MARKETING EMPLOYEES

```
SELECT
   N3.FirstName
   ,N3.LastName
   ,N4.JobTitle
   ,N4.BirthDate
   ,N4.MaritalStatus
   ,N4.HireDate
FROM HumanResources.EmployeeDepartmentHistory N1
INNER JOIN HumanResources.Department N2
   ON N1.DepartmentID = N2.DepartmentID
INNER JOIN Person.Person N3
   ON N1.BusinessEntityID = N3.BusinessEntityID
INNER JOIN HumanResources.Employee N4
   ON N1.BusinessEntityID = N4.BusinessEntityID
WHERE N2.Name = 'Marketing'
   AND ((YEAR (N4.HireDate) < 2002)
      OR YEAR (N4.HireDate) > 2004)
   AND N1.EndDate IS NULL
```

SOLUTION TO CHALLENGE QUESTION 22: WHO LEFT THAT REVIEW?

```
SELECT
   N1.ProductReviewID
   ,N1.ProductID
   ,ProductName = N2.Name
   ,N1.ReviewerName
   ,N1.Rating
   ,ReviewerEmail = N1.EmailAddress
   ,N3.BusinessEntityID
FROM Production.ProductReview N1
INNER JOIN Production.Product N2
   ON N1.ProductID = N2.ProductID
LEFT JOIN Person.EmailAddress N3
   ON N1.EmailAddress = N3.EmailAddress
```

—Execution returns each BusinessEntityID as NULL. It will not be possible to locate sales orders related to product reviews because there are no matches.

SOLUTION TO CHALLENGE QUESTION 23: LABEL MIX-UP

```
SELECT
   N1.SalesOrderID
   ,N3.OrderDate
   ,ProductName = N2.Name
   ,N5.FirstName
   ,N5.LastName
   ,N6.PhoneNumber
FROM Sales.SalesOrderDetail N1
INNER JOIN Production.Product N2
   ON N1.ProductID = N2.ProductID
INNER JOIN Sales.SalesOrderHeader N3
   ON N1.SalesOrderID = N3.SalesOrderID
INNER JOIN Sales.Customer N4
   ON N3.CustomerID = N4.CustomerID
INNER JOIN Person.Person N5
   ON N4.PersonID = N5.BusinessEntityID
INNER JOIN Person.PersonPhone N6
   ON N5.BusinessEntityID = N6.BusinessEntityID
WHERE N2.Name like ‘%shorts%’
   AND N3.OrderDate > ‘2008-07-07’
   AND N3.OnlineOrderFlag = 1
ORDER BY N1.SalesOrderID
```

SOLUTION TO CHALLENGE QUESTION 24: CLEARANCE SALE

```
WITH Email AS
(SELECT
   N1.BusinessEntityID
   ,N1.EmailAddress
   ,EmailPref =
      CASE WHEN N2.EmailPromotion = 0 THEN
         'Contact does not wish to receive e-mail promotions'
           WHEN N2.EmailPromotion = 1 THEN
         'Contact does wish to receive e-mail promotions from AdventureWorks'
           WHEN N2.EmailPromotion = 2 THEN
         'Contact does wish to receive e-mail promotions from AdventureWorks and
selected partners'
      END
   FROM Person.EmailAddress N1
   LEFT JOIN Person.Person N2
      ON N1.BusinessEntityID = N2.BusinessEntityID
   WHERE N2.PersonType = 'IN')

SELECT
   EmailPref
   ,[Count] = COUNT (*)
FROM Email
GROUP BY EmailPref
ORDER BY COUNT (*) DESC
```

SOLUTION TO CHALLENGE QUESTION 25: TOP TERRITORIES

```
WITH TerritoryRank AS
  (SELECT
    FY = YEAR (DATEADD (
      MONTH, 6, N1.OrderDate))
    ,Territory = N2.Name
    ,Revenue = SUM (N1.SubTotal)
    ,[Territory$Rank] =
      DENSE_RANK () OVER
        (PARTITION BY YEAR (
          DATEADD (
            MONTH, 6, N1.OrderDate))
          ORDER BY
            SUM (N1.Subtotal) DESC)
    FROM Sales.SalesOrderHeader N1
    INNER JOIN Sales.SalesTerritory N2
      ON N1.TerritoryID = N2.TerritoryID
    GROUP BY
      YEAR (
        DATEADD (
          MONTH, 6, N1.OrderDate))
      ,N2.Name)

SELECT *
FROM TerritoryRank
WHERE FY IN (2006, 2007)
  AND Territory$Rank IN (1, 2)
ORDER BY FY, Territory$Rank
```

SOLUTION TO CHALLENGE QUESTION 26: COMMISSION PERCENTAGES

```
SELECT
   BusinessEntityID
   ,CommissionPct
   ,Bonus
   ,[Rank] =
      DENSE_RANK () OVER
         (ORDER BY CommissionPct DESC
            ,Bonus DESC)
FROM Sales.SalesPerson
ORDER BY CommissionPct DESC
```

SOLUTION TO CHALLENGE QUESTION 27: WORK ORDERS

— Part I

```
SELECT
   ProductID
   ,WorkOrders = COUNT (*)
FROM Production.WorkOrder
GROUP BY ProductID
ORDER BY COUNT (*) DESC
```

— Part II

```
SELECT
   ProductName = N2.Name
   ,WorkOrders = COUNT (*)
FROM Production.WorkOrder N1
INNER JOIN Production.Product N2
   ON N1.ProductID = N2.ProductID
GROUP BY N2.Name
ORDER BY COUNT (*) DESC
```

SOLUTION TO CHALLENGE QUESTION 28: REVENUE TRENDED

```
DECLARE @StartDate DATE = '2008-05-01'
DECLARE @EndDate DATE = '2008-05-23'

—Part I:
SELECT
  DaysInMonthSoFar =
    DATEDIFF (day, @StartDate, @EndDate) + 1
  ,RevenueInMonthSoFar =
    SUM (SubTotal)
  ,RevPerDayforMonthSoFar =
    (SUM (SubTotal)
      / (DATEDIFF
        (day, @StartDate, @EndDate) + 1))
  ,DaysInMonth =
    DAY (EOMONTH (@StartDate))
  ,MonthlyRevTrended =
    SUM (SubTotal)
      / (DATEDIFF
        (day, @StartDate, @EndDate) + 1)
    * DAY (EOMONTH (@StartDate))
FROM Sales.SalesOrderHeader
WHERE OrderDate BETWEEN
  @StartDate AND @EndDate

— Part II:
SELECT
  ActualPerDay =
    SUM (SubTotal)
      / DAY (EOMONTH (@StartDate))
  ,ActualRev =   SUM (Subtotal)
FROM Sales.SalesOrderHeader
WHERE OrderDate BETWEEN
  @StartDate AND EOMONTH (@EndDate)
```

SOLUTION TO CHALLENGE QUESTION 29: SEPARATION

```
SELECT
  BusinessEntityID
  ,LoginID
  ,Domain =
    LEFT (LoginID,
      CHARINDEX ('', LoginID, 1) - 1)
  ,UserName =
    RIGHT (LoginID,
      LEN (LoginID)
        - CHARINDEX ('', LoginID, 1))
FROM HumanResources.Employee
ORDER BY BusinessEntityID
```

SOLUTION TO CHALLENGE QUESTION 30: SHIFT COVERAGE

```
SELECT
   DepartmentName = N2.Name
   ,ShiftName = N3.Name
   ,Employees = COUNT (*)
FROM HumanResources.EmployeeDepartmentHistory N1
INNER JOIN HumanResources.Department N2
   ON N1.DepartmentID = N2.DepartmentID
INNER JOIN HumanResources.[Shift] N3
   ON N1.ShiftID = N3.ShiftID
WHERE N2.Name = 'Production'
   AND N1.EndDate IS NULL
GROUP BY N2.Name, N3.Name
ORDER BY N2.Name, N3.Name
```

SOLUTION TO CHALLENGE QUESTION 31: LABELS

```
— Part I
  SELECT DISTINCT Size
FROM Production.Product
WHERE ISNUMERIC (Size) = 0
  AND Size IS NOT NULL
```

— The variety of stickers is appropriate for assignment to the company's products.

```
— Part II
SELECT
  N1.Size
  ,CurrentQty = SUM (N2.Quantity)
  ,AdditLabelsNeeded =
    CASE
      WHEN SUM (N2.Quantity) - 1000 < 0
        THEN 0
      ELSE SUM (N2.Quantity) - 1000
    END
FROM Production.Product N1
INNER JOIN Production.ProductInventory N2
  ON N1.ProductID = N2.ProductID
WHERE ISNUMERIC (N1.Size) = 0
  AND N1.Size IS NOT NULL
GROUP BY N1.Size
```

SOLUTION TO CHALLENGE QUESTION 32: EMPLOYMENT SURVEY

— Part I

```
SELECT
  Employees = COUNT (*)
  ,[%Male] =
    ROUND (SUM (
      CASE
        WHEN Gender = 'M'
          THEN 1
        ELSE 0
      END)
        / CONVERT (
          FLOAT, COUNT (*))
            * 100, 2)
  ,[%Female] =
    ROUND (SUM (
      CASE
        WHEN Gender = 'F'
          THEN 1
        ELSE 0
      END)
        / CONVERT (
          FLOAT, COUNT (*))
            * 100, 2)
,AvgMonthsEmp =
  AVG (DATEDIFF (
    MONTH, HireDate, '2008-01-01'))
FROM HumanResources.Employee
```

— Part II

```
SELECT
  X1.Quartile
  ,Employees = COUNT (*)
  ,[%Male] =
    ROUND (SUM (
      CASE
        WHEN X1.Gender = 'M'
          THEN 1
        ELSE 0
      END)
```

```
        / CONVERT (
          FLOAT, COUNT (*))
            * 100, 2)
  ,[%Female] =
    ROUND (SUM (
      CASE
        WHEN X1.Gender = 'F'
          THEN 1
        ELSE 0
      END)
        / CONVERT (
          FLOAT, COUNT (*))
            * 100, 2)
  ,AvgMonthsEmp =   AVG (X1.MonthsEmployed)
FROM
(SELECT
  BusinessEntityID
  ,Quartile = NTILE (4) OVER
    (ORDER BY
      DATEDIFF (
        MONTH, HireDate, '2008-01-01'))
  ,HireDate
  ,MonthsEmployed  =
    DATEDIFF (
        MONTH, HireDate, '2008-01-01')
  ,Gender
FROM HumanResources.Employee) X1
GROUP BY X1.Quartile
```

SOLUTION TO CHALLENGE QUESTION 33: AGE GROUPS

```
SELECT
  N1.JobTitle
  ,AgeGroup =
    CASE
      WHEN DATEDIFF (
        YY, N1.BirthDate, '2008-01-01') < 18
        THEN '< 18'
      WHEN DATEDIFF (
        YY, N1.BirthDate, '2008-01-01') < 35
        THEN '18 - 35'
      WHEN DATEDIFF (
        YY, N1.BirthDate, '2008-01-01') < 50
        THEN '36 - 50'
      WHEN DATEDIFF (
        YY, N1.BirthDate, '2008-01-01') < 60
        THEN '51 - 60'
      ELSE '61 +'
    END
  ,N2.Rate
  ,Employees = COUNT (N1.BusinessEntityID)
FROM HumanResources.Employee N1
INNER JOIN HumanResources.EmployeePayHistory N2
  ON N1.BusinessEntityID = N2.BusinessEntityID
INNER JOIN
  (SELECT
    BusinessEntityID
    ,RatechangeDate = MAX (RateChangeDate)
  FROM HumanResources.EmployeePayHistory
  GROUP BY BusinessEntityID) N3
  ON N3.BusinessEntityID = N2.BusinessEntityID
    AND N3.RatechangeDate = N2.RateChangeDate
GROUP BY
  JobTitle
  ,Rate
  ,CASE
    WHEN DATEDIFF (
      YY, N1.BirthDate, '2008-01-01') < 18
      THEN '< 18'
    WHEN DATEDIFF (
      YY, N1.BirthDate, '2008-01-01') < 35
      THEN '18 - 35'
    WHEN DATEDIFF (
```

```
      YY, N1.BirthDate, '2008-01-01') < 50
      THEN '36 - 50'
    WHEN DATEDIFF (
      YY, N1.BirthDate, '2008-01-01') < 60
      THEN '51 - 60'
    ELSE '61 +'
END
```

SOLUTION TO CHALLENGE QUESTION 34: REVENUE BY STATE

```
SELECT
   [State] = N3.Name
   ,TotalSales =   SUM (N1.TotalDue)
FROM Sales.SalesOrderHeader N1
INNER JOIN Person.[Address] N2
   ON N1.ShipToAddressID = N2.AddressID
INNER JOIN Person.StateProvince N3
   ON N2.StateProvinceID = N3.StateProvinceID
WHERE YEAR (N1.OrderDate) = 2006
GROUP BY N3.Name
ORDER BY SUM (N1.TotalDue) DESC
```

SOLUTION TO CHALLENGE QUESTION 35: TWO FREE BIKES

```
—DROP VIEW
  —HumanResources.Vw_Employee_Bicycle_Giveaway

CREATE VIEW
  HumanResources.Vw_Employee_Bicycle_Giveaway

AS

SELECT TOP 2
  N2.FirstName
  ,N2.LastName
  ,N1.JobTitle
FROM HumanResources.Employee N1
INNER JOIN Person.Person N2
  ON N1.BusinessEntityID = N2.BusinessEntityID
WHERE N1.OrganizationLevel =
  (SELECT MAX (OrganizationLevel)
  FROM HumanResources.Employee)
ORDER BY NEWID ()
```

SOLUTION TO CHALLENGE QUESTION 36: VOLUME DISCOUNTS

```
—DROP TABLE #data

SELECT
   N1.SalesOrderID
   ,N3.OrderDate
   ,TotalVolumeDiscount =
      SUM (N1.UnitPriceDiscount
        * N1.UnitPrice * N1.OrderQty)
INTO #data
FROM Sales.SalesOrderDetail N1
INNER JOIN Sales.SpecialOffer N2
   ON N1.SpecialOfferID = N2.SpecialOfferID
INNER JOIN Sales.SalesOrderHeader N3
   ON N1.SalesOrderID = N3.SalesOrderID
WHERE N2.[Type] = 'Volume Discount'
GROUP BY
   N1.SalesOrderID
   ,N2.[Type], N3.OrderDate
HAVING SUM (
   N1.UnitPriceDiscount * N1.UnitPrice
   * N1.OrderQty) > 0

— Part I

SELECT *
FROM #data
ORDER BY SalesOrderID


— Part II

SELECT
   OrderYear = YEAR (OrderDate)
   ,TotalVolumeDiscount =
      SUM (TotalVolumeDiscount)
FROM #data
GROUP BY YEAR (OrderDate)
```

SOLUTION TO CHALLENGE QUESTION 37: OVERPAYING

```
—DROP TABLE #ProductVendor

SELECT
   ProductID
   ,MostExpensivePrice =
      MAX (LastReceiptCost)
   ,SecondMostExpensivePrice =
      CONVERT (FLOAT,NULL)
   ,PercOverSecondPrice =
      CONVERT (FLOAT,NULL)
INTO #ProductVendor
FROM Purchasing.ProductVendor
GROUP BY ProductID
HAVING
   COUNT (DISTINCT LastReceiptCost) > 1

UPDATE N1
   SET SecondMostExpensivePrice =
      (SELECT TOP 1 X1.LastReceiptCost
      FROM Purchasing.ProductVendor X1
      WHERE N1.ProductID = X1.ProductID
         AND N1.MostExpensivePrice <> X1.LastReceiptCost
      ORDER BY X1.LastReceiptCost DESC)
FROM #ProductVendor N1

UPDATE N1
   SET PercOverSecondPrice =
      CONVERT (DECIMAL (10,2),
         CONVERT (FLOAT,
            (MostExpensivePrice - SecondMostExpensivePrice))
            / SecondMostExpensivePrice)
FROM #ProductVendor N1

SELECT *
FROM #ProductVendor
ORDER BY PercOverSecondPrice DESC
```

SOLUTION TO CHALLENGE QUESTION 38: MARGINS

```
SELECT
   N1.ProductModelID
   ,ProductName = N4.Name
   ,ProfitMargin =
      CONVERT (DECIMAL(10,2),
         CONVERT (FLOAT,
            (N1.ListPrice - N1.StandardCost))
            / N1.StandardCost)
FROM Production.Product N1
INNER JOIN Production.ProductSubcategory N2
   ON N1.ProductSubcategoryID = N2.ProductSubcategoryID
INNER JOIN Production.ProductCategory N3
   ON N2.ProductCategoryID = N3.ProductCategoryID
INNER JOIN Production.ProductModel N4
   ON N1.ProductModelID = N4.ProductModelID
WHERE N3.Name = 'Bikes' AND N1.SellEndDate IS NULL
GROUP BY
   N1.ProductModelID
   ,N4.Name
   ,CONVERT (DECIMAL (10,2)
      ,CONVERT (FLOAT,
         (N1.ListPrice - N1.StandardCost))
         / N1.StandardCost)
ORDER BY ProfitMargin DESC
```

SOLUTION TO CHALLENGE QUESTION 39: PERCENT TO QUOTA

```
—   Part I

—DROP TABLE #SalesQuotaSummary

SELECT
   N1.BusinessEntityID
   ,N1.QuotaDate
   ,N1.SalesQuota
   ,ActualSales =
      CONVERT (DECIMAL (10,2),
         SUM (N2.SubTotal))
   ,PercToQuota =
      CONVERT (DECIMAL (10,2),
         CONVERT (FLOAT,
            SUM (N2.SubTotal))
            / N1.SalesQuota)
INTO #SalesQuotaSummary
FROM Sales.SalesPersonQuotaHistory N1
LEFT JOIN Sales.SalesOrderHeader N2
   ON N1.BusinessEntityID = N2.SalesPersonID
      AND N2.OrderDate > = N1.QuotaDate
      AND N2.OrderDate < DATEADD (
         MONTH, 3, N1.QuotaDate)
GROUP BY
   N1.BusinessEntityID
   ,N1.QuotaDate
   ,N1.SalesQuota

SELECT *
FROM #SalesQuotaSummary
ORDER BY BusinessEntityID, QuotaDate

—   Part II

SELECT
   BusinessEntityID
   ,QuotaYear = YEAR (QuotaDate)
   ,TotalQuota = SUM (SalesQuota)
   ,TotalSales = SUM (ActualSales)
   ,TotalPercToQuota =
      CONVERT (DECIMAL (10,2),
         CONVERT (FLOAT,
```

```
            SUM (ActualSales))
              / SUM (SalesQuota))
   ,AvgQrtlyPercToQuota =
      CONVERT (DECIMAL (10,2),
        AVG (PercToQuota))
FROM #SalesQuotaSummary
GROUP BY
   BusinessEntityID
   ,YEAR (QuotaDate)
ORDER BY
   BusinessEntityID
   ,YEAR (QuotaDate)
```

SOLUTION TO CHALLENGE QUESTION 40: REVENUE RANGES

```
SELECT
  SortID =
    CASE
      WHEN TotalDue < 100
        THEN 1
      WHEN TotalDue < 500
        THEN 2
      WHEN TotalDue < 1000
        THEN 3
      WHEN TotalDue < 2500
        THEN 4
      WHEN TotalDue < 5000
        THEN 5
      WHEN TotalDue < 10000
        THEN 6
      WHEN TotalDue < 50000
        THEN 7
      WHEN TotalDue < 100000
        THEN 8
      ELSE 9
    END
  ,SalesAmountCategory =
    CASE
      WHEN TotalDue < 100
        THEN '0 - 100'
      WHEN TotalDue < 500
        THEN '100 - 500'
      WHEN TotalDue < 1000
        THEN '500 - 1,000'
      WHEN TotalDue < 2500
        THEN '1,000 - 2,500'
      WHEN TotalDue < 5000
        THEN '2,500 - 5,000'
      WHEN TotalDue < 10000
        THEN '5,000 - 10,000'
      WHEN TotalDue < 50000
        THEN '10,000 - 50,000'
      WHEN TotalDue < 100000
        THEN '50,000 - 100,000'
      ELSE '> 100,000'
    END
  ,Orders = COUNT (*)
```

```
FROM Sales.SalesOrderHeader
WHERE YEAR (OrderDate) = 2005
GROUP BY
  CASE
    WHEN TotalDue < 100
      THEN 1
    WHEN TotalDue < 500
      THEN 2
    WHEN TotalDue < 1000
      THEN 3
    WHEN TotalDue < 2500
      THEN 4
    WHEN TotalDue < 5000
      THEN 5
    WHEN TotalDue < 10000
      THEN 6
    WHEN TotalDue < 50000
      THEN 7
    WHEN TotalDue < 100000
      THEN 8
    ELSE 9
  END
  ,CASE
    WHEN TotalDue < 100
      THEN '0 - 100'
    WHEN TotalDue < 500
      THEN '100 - 500'
    WHEN TotalDue < 1000
      THEN '500 - 1,000'
    WHEN TotalDue < 2500
      THEN '1,000 - 2,500'
    WHEN TotalDue < 5000
      THEN '2,500 - 5,000'
    WHEN TotalDue < 10000
      THEN '5,000 - 10,000'
    WHEN TotalDue < 50000
      THEN '10,000 - 50,000'
    WHEN TotalDue < 100000
      THEN '50,000 - 100,000'
    ELSE '> 100,000'
  END
ORDER BY SortID
```

SOLUTION TO CHALLENGE QUESTION 41: E-MAIL MYSTERY

```
SELECT
  N2.PersonType
  ,AWEmail =
    SUM (
      CASE
        WHEN N1.EmailAddress LIKE '%adventure-works%'
          THEN 1
        ELSE 0
      END)
  ,NotAWEmail =
    SUM (
      CASE
        WHEN N1.EmailAddress NOT LIKE '%adventure-works%'
          THEN 1
        ELSE 0
      END)
  ,Total = COUNT (*)
FROM Person.EmailAddress N1
INNER JOIN Person.Person N2
  ON N1.BusinessEntityID = N2.BusinessEntityID
GROUP BY N2.PersonType
ORDER BY Total DESC
```

SOLUTION TO CHALLENGE QUESTION 42: THE MENTORS

```
WITH SalesGrouping AS
  (SELECT
    SalesPersonID
    ,SalesTotal =   SUM (SubTotal)
    ,SalesRankSubTotalDESC =
      ROW_NUMBER () OVER (
        ORDER BY
          SUM (Subtotal) DESC)
    ,SalesRankSubTotalASC =
      ROW_NUMBER () OVER (
        ORDER BY
          SUM (Subtotal))
FROM Sales.SalesOrderHeader
WHERE YEAR (OrderDate) = 2008
  AND SalesPersonID IS NOT NULL
GROUP BY SalesPersonID)

SELECT TOP 5
  SuccessSalesPersonID = N1.SalesPersonID
  ,SuccessRevenue =   N1.SalesTotal
  ,UnsuccessSalesPersonID = N2.SalesPersonID
  ,UnsuccessRevenue = N2.SalesTotal
FROM SalesGrouping N1
INNER JOIN SalesGrouping N2
  ON N1.SalesRankSubTotalDESC = N2.SalesRankSubTotalASC
ORDER BY N1.SalesRankSubTotalDESC
```

SOLUTION TO CHALLENGE QUESTION 43: CALENDAR OF WORK DAYS

```
—DROP TABLE HumanResources.Calendar

CREATE TABLE HumanResources.Calendar
  (DateID INT
  ,[Date] DATETIME
  ,[Year] INT
  ,TextMonth VARCHAR (50)
  ,DateMonth DATETIME
  ,[DayOfWeek] VARCHAR (50)
  ,IsBusinessDay TINYINT)

DECLARE @StartDate DATETIME = '1990-01-01'
DECLARE @EndDate DATETIME = '2015-01-01'

DECLARE @TotalDays INT =
  DATEDIFF (
    DAY, @StartDate, @EndDate) + 1

DECLARE @Index INT = 1

WHILE @Index <= @TotalDays
BEGIN
  INSERT INTO HumanResources.Calendar (DateID)
  SELECT @Index
  SET @Index = @Index + 1
END

UPDATE N1
SET [Date] =
  DATEADD (DAY, DateID - 1, @StartDate)
FROM HumanResources.Calendar N1

UPDATE N1
  SET
    [Year] = YEAR ([Date])
    ,TextMonth   =
      DATENAME (
        MONTH, [Date])
          + ' ' + DATENAME (
            YEAR, [Date])
```

```
      ,DateMonth   =
        DATEADD (
          MONTH, DATEDIFF (
            MONTH, 0, [Date]), 0)
,[DayOfWeek] =   DATENAME (DW, [Date])
,IsBusinessDay =
  CASE
    WHEN DATENAME (
      DW, [Date]) IN ('Saturday', 'Sunday')
      THEN 0
    ELSE 1
  END
FROM HumanResources.Calendar N1

SELECT
  [Year]
  ,BusinessDays = SUM (IsBusinessDay)
FROM HumanResources.Calendar
GROUP BY [Year]
ORDER BY [Year]
```

SOLUTION TO CHALLENGE QUESTION 44: ANNUAL SALARIES BY EMPLOYEE

```
—DROP TABLE #EmployeePayHistory

SELECT *
  ,BusinessEntityOrder =
    ROW_NUMBER () OVER (
      PARTITION BY BusinessEntityID
      ORDER BY RateChangeDate)
  ,RateEndDate =   CONVERT (DATETIME, NULL)
  ,BusinessDays =   CONVERT (INT, NULL)
INTO #EmployeePayHistory
FROM HumanResources.EmployeePayHistory

UPDATE N1
SET RateEndDate =
  (SELECT TOP 1 RateChangeDate
  FROM #EmployeePayHistory X1
  WHERE N1.BusinessEntityID = X1.BusinessEntityID
    AND X1.BusinessEntityOrder = N1.BusinessEntityOrder + 1
  ORDER BY X1.RateChangeDate)
FROM #EmployeePayHistory N1

UPDATE N1
SET BusinessDays =
  (SELECT COUNT (*)
  FROM HumanResources.Calendar X1
  WHERE X1.[Date] BETWEEN N1.RateChangeDate
    AND ISNULL (N1.RateEndDate, '2009-01-01')
    AND X1.IsBusinessDay = 1)
  FROM #EmployeePayHistory N1

—DROP TABLE #EmployeePayCalendar

SELECT *
  ,DailyPay = Rate * 8
INTO #EmployeePayCalendar
FROM #EmployeePayHistory N1
INNER JOIN HumanResources.Calendar N2
  ON N2.[Date] > = N1.RateChangeDate
    AND N2.[Date] < ISNULL (
      N1.RateEndDate, '2009-01-01')
```

```
      AND N2.IsBusinessDay = 1

SELECT
   BusinessEntityID
   ,WorkingYear =   [Year]
   ,TotalPay = SUM (DailyPay)
FROM #EmployeePayCalendar
WHERE [Year] BETWEEN 2005 AND 2008
GROUP BY BusinessEntityID, [Year]
ORDER BY BusinessEntityID, [Year]
```

SOLUTION TO CHALLENGE QUESTION 45: ANNUAL SALARIES BY DEPARTMENT

```
—DROP TABLE #EmployeePayDepartmentCalendar

SELECT
   N1.DepartmentID
   ,N1.StartDate
   ,N1.EndDate
   ,N2.*
INTO #EmployeePayDepartmentCalendar
FROM HumanResources.EmployeeDepartmentHistory N1
LEFT JOIN #EmployeePayCalendar N2
   ON N1.BusinessEntityID = N2.BusinessEntityID
      AND N2.[Date] BETWEEN
         N1.StartDate AND
         ISNULL (N1.EndDate, '2009-01-01')

—DROP TABLE #DepartmentEmployeeSummary

SELECT
   DepartmentID
   ,BusinessEntityID
   ,TotalPay = SUM (DailyPay)
INTO #DepartmentEmployeeSummary
FROM #EmployeePayDepartmentCalendar
WHERE [Year] = 2008
GROUP BY
   DepartmentID
   ,BusinessEntityID

SELECT
   DepartmentID
   ,MinSalary = MIN (TotalPay)
   ,AvgSalary = AVG (TotalPay)
   ,MaxSalary = MAX (TotalPay)
FROM #DepartmentEmployeeSummary
GROUP BY DepartmentID
```

SOLUTION TO CHALLENGE QUESTION 46: HOLIDAY BONUS

```
SELECT
   N1.BusinessEntityID
   ,N2.FirstName
   ,N2.LastName
   ,N1.JobTitle
   ,Bonus  =
      (SELECT TOP 1 Rate
      FROM HumanResources.EmployeePayHistory X1
      WHERE N1.BusinessEntityID = X1.BusinessEntityID
      ORDER BY RateChangeDate DESC) * 50
FROM HumanResources.Employee N1
INNER JOIN Person.Person N2
   ON N1.BusinessEntityID = N2.BusinessEntityID
WHERE N1.SalariedFlag = 1
ORDER BY N1.BusinessEntityID
```

SOLUTION TO CHALLENGE QUESTION 47: COMPANY PICNIC

```
SELECT
  N1.BusinessEntityID
  ,FullName =
    CONVERT (VARCHAR (50),
       FirstName) + ' '
    + CONVERT (VARCHAR (50),
      LastName) + ISNULL (', ' + Suffix, '')
  ,Dept = N4.Name
FROM Person.Person N1
INNER JOIN
  (SELECT
    BusinessEntityID
    ,MaxStart = MAX (StartDate)
    FROM HumanResources.EmployeeDepartmentHistory
    GROUP BY BusinessEntityID) N2
  ON N1.BusinessEntityID = N2.BusinessEntityID
INNER JOIN HumanResources.EmployeeDepartmentHistory N3
  ON N2.MaxStart = N3.StartDate
    AND N2.BusinessEntityID = N3.BusinessEntityID
INNER JOIN HumanResources.Department N4
  ON N3.DepartmentID = N4.DepartmentID
WHERE N1.PersonType IN ('SP', 'EM')
ORDER BY Dept, FullName
```

SOLUTION TO CHALLENGE QUESTION 48: SALES QUOTA CHANGES

```
SELECT DISTINCT
   N1.BusinessEntityID
   ,SalesRepLastName = N4.LastName
   ,Yr2006StartQuota = N2.SalesQuota
   ,Yr2007EndQuota = N3.SalesQuota
   ,[%ChangeQuota] =
       (N3.SalesQuota - N2.SalesQuota)
      / N2.SalesQuota * 100
FROM Sales.SalesPersonQuotaHistory N1
INNER JOIN Sales.SalesPersonQuotaHistory N2
   ON N1.BusinessEntityID = N2.BusinessEntityID
      AND N2.QuotaDate =
         (SELECT MIN (QuotaDate)
         FROM Sales.SalesPersonQuotaHistory
         WHERE YEAR (QuotaDate) = 2006)
INNER JOIN Sales.SalesPersonQuotaHistory N3
   ON N1.BusinessEntityID = N3.BusinessEntityID
      AND N3.QuotaDate =
         (SELECT MAX (QuotaDate)
         FROM Sales.SalesPersonQuotaHistory
         WHERE YEAR (QuotaDate) = 2007)
INNER JOIN Person.Person AS N4
   ON N1.BusinessEntityID = N4.BusinessEntityID
```

SOLUTION TO CHALLENGE QUESTION 49: SCRAP RATE

```
—DROP VIEW Production.Vw_ScrapRates

CREATE VIEW Production.Vw_ScrapRates

AS

SELECT TOP 10 PERCENT
   N1.WorkOrderID
   ,DueDate = CAST (N1.DueDate AS DATE)
   ,ProdName = N3.Name
   ,ScrapReason =    N2.Name
   ,N1.ScrappedQty
   ,N1.OrderQty
   ,[PercScrapped] =
      ROUND (
         N1.ScrappedQty /
         CONVERT (FLOAT,
            N1.OrderQty)* 100, 2)
FROM Production.WorkOrder N1
INNER JOIN Production.ScrapReason N2
   ON N1.ScrapReasonID = N2.ScrapReasonID
INNER JOIN Production.Product N3
   ON N1.ProductID = N3.ProductID
WHERE N1.ScrappedQty / CONVERT (FLOAT,
   N1.OrderQty) >  0.03
ORDER BY N1.DueDate DESC
```

SOLUTION TO CHALLENGE QUESTION 50: REASONS

```
WITH Reasons AS
  (SELECT
    N1.SalesOrderID
    ,ReasonName =   N2.Name
    ,ReasonInfluence =
      CASE
        WHEN COUNT (N3.SalesOrderID) >  1
          THEN 'Contributing Reason'
        WHEN COUNT (N3.SalesOrderID) = 1
          THEN 'Exclusive Reason' END
    FROM Sales.SalesOrderHeaderSalesReason N1
    INNER JOIN Sales.SalesReason N2
      ON N1.SalesReasonID = N2.SalesReasonID
    INNER JOIN Sales.SalesOrderHeaderSalesReason N3
      ON N1.SalesOrderID = N3.SalesOrderID
    GROUP BY N1.SalesOrderID, N2.Name)

SELECT
  ReasonName
  ,ReasonInfluence
  ,SalesOrderCount = COUNT (*)
FROM Reasons
GROUP BY
  ReasonName
  ,ReasonInfluence
ORDER BY ReasonName, SalesOrderCount DESC
```

SOLUTION TO CHALLENGE QUESTION 51: EXCESS INVENTORY

— Part I

```
SELECT
   SpecialOfferID
   ,DiscountType =    [Type]
   ,DiscountDescr   = [Description]
   ,Category
   ,StartDate
   ,EndDate
   ,DiscountPct
FROM Sales.SpecialOffer
WHERE [Type] = 'Excess Inventory'
```

— Part II

```
SELECT
   N1.SpecialOfferID
   ,DiscountType = [Type]
   ,DiscountDescr   = [Description]
   ,N1.Category
   ,N1.StartDate
   ,N1.EndDate
   ,N1.DiscountPct
   ,SalesOrders =
      (SELECT
         COUNT (DISTINCT X1.SalesOrderID)
      FROM Sales.SalesOrderDetail X1
      WHERE N1.SpecialOfferID = X1.SpecialOfferID)
FROM Sales.SpecialOffer N1
WHERE N1.[Type] = 'Excess Inventory'
```

SOLUTION TO CHALLENGE QUESTION 52: PAY RATE CHANGES

```
WITH Data AS
  (SELECT
    BusinessEntityID
    ,PayRateNumber =
      ROW_NUMBER () OVER (
        PARTITION BY BusinessEntityID
        ORDER BY RateChangeDate DESC)
    ,RateChangeDate
    ,Rate
  FROM HumanResources.EmployeePayHistory)

SELECT
  N1.BusinessEntityID
  ,RatePrior = N2.Rate
  ,LatestRate = N1.Rate
  ,PercentChange =
    CONVERT (VARCHAR (10),
      (N1.Rate - N2.Rate)
      / N2.Rate * 100) + '%'
FROM Data N1
LEFT JOIN Data N2
  ON N1.BusinessEntityID = N2.BusinessEntityID
    AND N2.PayRateNumber = 2
WHERE N1.PayRateNumber = 1
```