70-433 Q&As

TS: Microsoft SQL Server 2008, Database Development

Pass Microsoft 70-433 Exam with 100% Guarantee

Free Download Real Questions & Answers **PDF** and **VCE** file from:

https://www.geekcert.com/70-433.html

100% Passing Guarantee
100% Money Back Assurance

Following Questions and Answers are all new published by Microsoft Official Exam Center

QUESTION 1

You administer a Microsoft SQL Server 2008 database that contains a stored procedure named dbo.SalesOrderDetails. The stored procedure has following definition:

```
CREATE PROCEDURE dbo.SalesOrderDetails
  @CustomerID int,
  @OrderDate datetime,
  @SalesOrderID int
AS
SELECT
  h.SalesOrderID,
  h.OrderDate,
  d.OrderQty,
  d.ProductID
FROM
  Sales.SalesOrderHeader h
    INNER JOIN
  Sales.SalesOrderDetail d
    ON d.SalesOrderID = h.SalesOrderID
WHERE
  h.CustomerID = @CustomerID
  or h.OrderDate > @OrderDate
  or h.SalesOrderID > @SalesOrderID
GO
```


Parameter values passed to the stored procedure largely vary. You discover that the stored procedure executes quickly for some parameters but slowly for other parameters. You need to ensure that the query plan generated is optimized to provide the most consistent execution times for any set of parameters passed to the stored procedure. Which query hint should you use?

A. OPTION (FAST 25)

B. OPTION (ROBUST PLAN)

C. OPTION (KEEP PLAN)

D. OPTION (OPTIMIZE FOR UNKNOWN)

Correct Answer: C

QUESTION 2

You need to round the value 1.75 to the nearest whole number.

Which code segment should you use?

A. Select ROUND(1.75,0)

B. Select ROUND(1.75,2)

C. Select ROUND(1.75,1.0)

D. Select ROUND(1.75,2.0)

Correct Answer: A

QUESTION 3

You are a developer for a Microsoft SQL Server 2008 R2 database instance used to support a customer service application. You create tables named complaint, customer, and product as follows:

```
CREATE TABLE [dbo].[complaint]
 ([ComplaintID] [int],
  [ProductID] [int],
  [CustomerID] [int],
  [ComplaintDate] [datetime]);

CREATE TABLE [dbo].[customer]
 ([CustomerID] [int],
  [CustomerName] [varchar](100),
  [Address] [varchar](200),
  [City] [varchar](100),
  [State] [varchar](50),
  [ZipCode] [varchar](5));

CREATE TABLE [dbo].[product]
 ([ProductID] [int],
  [ProductName] [varchar](100),
  [SalePrice] [money],
  [ManufacturerName] [varchar](100
```


You need to write a query to return all customer names and total number of complaints for customers who have made more than 10 complaints. Which SQL query should you use?

A.
```
SELECT
  c.CustomerName,
  p.ProductName,
  SUM(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID INNER JOIN
  customer c ON com.CustomerID = c.CustomerID
GROUP BY GROUPING SETS ((c.CustomerName, p.ProductName), ());
```

B.
```
SELECT
  c.CustomerName,
  p.ProductName,
  SUM(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID INNER JOIN
  customer c ON com.CustomerID = c.CustomerID
GROUP BY GROUPING SETS ((c.CustomerName), (p.ProductName), ());
```

C.
```
SELECT
  c.CustomerName,
  COUNT(com.ComplaintID) AS Complaints
FROM
  customer c INNER JOIN
  complaint com ON c.CustomerID = com.CustomerID
WHERE
  COUNT(com.ComplaintID) > 10
GROUP BY
  c.CustomerName;
```

D.
```
SELECT
  c.CustomerName
  COUNT(com.ComplaintID) AS complaints
FROM
  customer c INNER JOIN
  complaint com ON c.CustomerID = com.CustomerID
GROUP BY
  c.CustomerName
HAVING
  COUNT(com.ComplaintID) > 10;
```

E.
```
SELECT
  c.CustomerName,
  AVG(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID INNER JOIN
  customer c ON com.CustomerID = c.CustomerID
WHERE
  com.ComplaintDate > '09/01/2011'
GROUP BY
  c.CustomerName
HAVING
  AVG(p.SalePrice) >= 500
```

F.
```
SELECT
  c.CustomerName,
  AVG(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID INNER JOIN
  customer c ON com.CustomerID = c.CustomerID
WHERE
  com.ComplaintDate > '09/01/2011' AND
  AVG(p.SalePrice) >= 500
```

G.
```
SELECT
  p.ProductName,
  DATEPART(mm, com.ComplaintDate) ComplaintMonth,
  SUM(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID
GROUP BY CUBE(p.ProductName, DATEPART(mm, com.ComplaintDate));
```

H.
```
SELECT
  p.ProductName,
  DATEPART(mm, com.ComplaintDate) ComplaintMonth,
  SUM(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID
GROUP BY CUBE;
```

A. B. C. D. E. F. G. H.

I.
```
SELECT
  p.ProductName,
  DATEPART(mm, com.ComplaintDate) ComplaintMonth,
  SUM(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID
GROUP BY p.ProductName, ComplaintMonth;
```

J.
```
SELECT
  p.ProductName,
  DATEPART(mm, com.ComplaintDate) ComplaintMonth,
  SUM(p.SalePrice) AS Sales
FROM
  product p INNER JOIN
  complaint com ON p.ProductID = com.ProductID
GROUP BY p.ProductName, DATEPART(mm, com.ComplaintDa
  AVG(p.SalePrice) >= 500
```

I. J.

Correct Answer: D

QUESTION 4

You administer a Microsoft SQL Server 2008 database that contains a table named dbo.SalesOrders.

The table has the following definition:

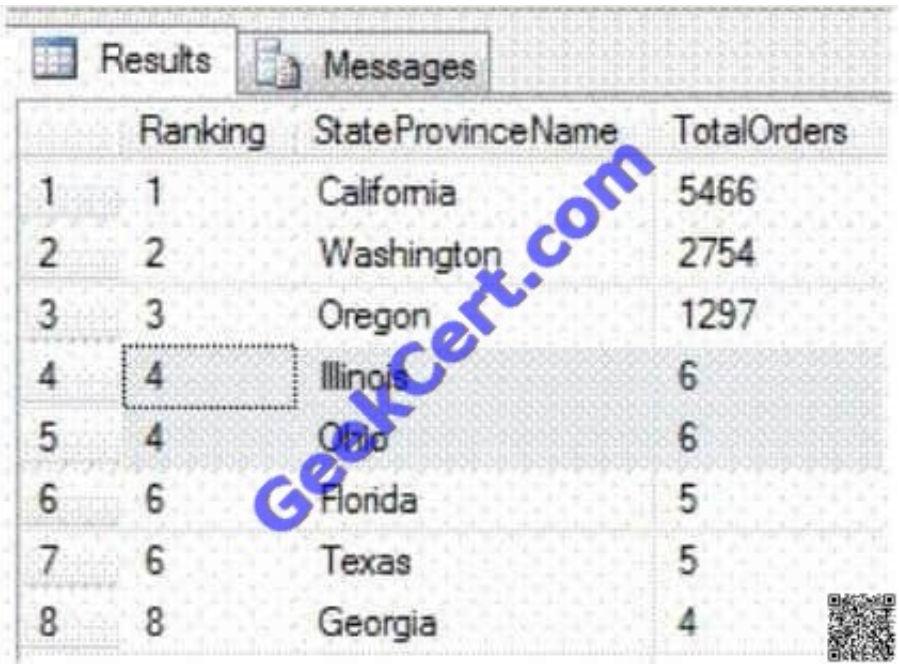
```
CREATE TABLE [dbo].[SalesOrder](
  [SalesOrderNumber] NVARCHAR(20) NOT NULL,
  [FullDateAlternateKey] DATETIME NOT NULL,
  [CustomerName] NVARCHAR(100) NOT NULL,
  [AddressLine] NVARCHAR(120) NOT NULL,
  [City] NVARCHAR(30) NOT NULL,
  [StateProvinceName] NVARCHAR(50) NOT NULL,
  [CountryName] NVARCHAR(50) NOT NULL,
  [SalesAmount] MONEY NOT NULL,
  CONSTRAINT [PK_SalesOrderNumber] PRIMARY KEY CLUSTERED
(
  [SalesOrderNumber] ASC
) ON [PRIMARY]
) ON [PRIMARY]
GO
```

The SalesOrder table contains one million rows.

You want to create a report that meets the following requirements:

Only the states of the Unites States are ranked against each other based on the total number of orders received from each state. When two states have the same rank, the rank of the subsequent state is one plus the number of ranks that

come before that row, as shown in the exhibit. (Click the Exhibit button.)



	Ranking	StateProvinceName	TotalOrders
1	1	California	5466
2	2	Washington	2754
3	3	Oregon	1297
4	4	Illinois	6
5	4	Ohio	6
6	6	Florida	5
7	6	Texas	5
8	8	Georgia	4

You need to execute a Transact-SQL query to generate the report. Which Transact-SQL query should you use?

A.
```
SELECT
  DENSE_RANK() OVER (PARTITION BY CountryName ORDER BY TotalOrders DESC) AS Ranking,
  StateProvinceName,
  TotalOrders
FROM
  (
  SELECT
    CountryName,
    StateProvinceName,
    count(*) AS TotalOrders
  FROM
    dbo.SalesOrder
  GROUP BY
    CountryName,
    StateProvinceName
  ) AS
```

B.
```
SELECT
  RANK() OVER (ORDER BY StateProvinceName DESC) AS Ranking,
  StateProvinceName,
  TotalOrders
FROM
  (
  SELECT
    StateProvinceName,
    count(*) AS TotalOrders
  FROM
    dbo.SalesOrder
  WHERE
    CountryName='United States'
  GROUP BY
    StateProvinceName
  ) AS
```

C.
```
SELECT
  RANK() OVER (PARTITION BY CountryName ORDER BY TotalOrders DESC) AS Ranking,
  StateProvinceName,
  TotalOrders
FROM
  (
  SELECT
    CountryName,
    StateProvinceName,
    count(*) AS TotalOrders
  FROM
    dbo.SalesOrder
  GROUP BY
    CountryName,
    StateProvinceName
  ) AS
```

D.
```
SELECT
  RANK() OVER (ORDER BY TotalOrders DESC) AS Ranking,
  StateProvinceName,
  TotalOrders
FROM
  (
  SELECT
    StateProvinceName,
    count(*) AS TotalOrders
  FROM
    dbo.SalesOrder
  WHERE
    CountryName='United States'
  GROUP BY
    StateProvinceName
  ) AS
```

A. B. C. D.

Correct Answer: D

QUESTION 5

You have two tables named Customers and Orders.

for customers that have placed at least one order, you need to produce a list of customer names and the number of orders for each customer.

Which query should you use?

A. SELECT c.CustomerName, SUM(o.OrderID) AS [OrderCount] FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID GROUP BY c.CustomerName

B. SELECT COUNT(o.OrderId) AS [OrderCount] FROM CUSTOMERS c JOIN ORDERS o ON c.CUSTOMERID = o.CUSTOMERID

C. SELECT c.CustomerName, COUNT(o.OrderID) AS [OrderCount] FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID GROUP BY c.CustomerName HAVING COUNT(o.OrderID) > 1

D. SELECT c.CustomerName, COUNT(o.OrderId) AS [OrderCount] FROM Customers c JOIN Orders o ON c.CustomerId = o.CustomerId GROUP BY c.CustomerName

Correct Answer: D