

Get Free PDF Microsoft 70-433 Braindumps From Braindump2go - Guarantee 100% Pass 70-433 Exam (161-170)

Braindump2go Guarantees Your Microsoft 70-433 Exam 100% Success with Our Unique Official 70-433 Exam Questions Resources! Braindump2go's 70-433 Braindumps are Developed by Experiences IT Certifications Professionals Working in Today's Prospering Companies and Data Centers! Braindump2go 70-433 Exam Dumps are Checked by Our Experts Team every day to ensure you have the Latest Updated Exam Dumps! Exam Code: 70-433Exam Name: TS: Microsoft SQL Server 2008, Database DevelopmentCertification Provider: MicrosoftKeywords: 70-433 Exam Dumps,70-433 Practice Tests,70-433 Practice Exams,70-433 Exam Questions,70-433 PDF,70-433 VCE Free,70-433 Book,70-433 E-Book,70-433 Study Guide,70-433 Braindump,70-433 Prep Guide

Compared Before Buying Microsoft 70-433 PDF & VCE!		
Pass4sure	Braindump2go	Test King
202 Q&As - Practice	210 Q&As – Real Questions	202 Q&As - Practice
\$125.99	\$99.99	\$124.99
No Discount	Coupon Code: BDNT2014	No Discount

QUESTION 161 You are a database developer responsible for maintaining an application. The application has a table named Programs that has the following definition:

```
CREATE TABLE [dbo].[Customers]
(
    [ID] int NOT NULL IDENTITY(1,1),
    [Name] varchar(100) NOT NULL,
    [Address] varchar(255) NOT NULL,
    [City] varchar(100) NOT NULL,
    [State] char(3) NOT NULL,
    [Zip] int NOT NULL,
    [Active] bit NOT NULL
)
GO
ALTER TABLE [dbo].[Customers] ADD CONSTRAINT PK_Customers PRIMARY KEY NONCLUSTERED ([Name])
GO
```

You need to modify the Customers table to meet the following requirements: Which Transact-SQL statement or statements should you use?

- A. ALTER TABLE [dbo].[Customers] DROP CONSTRAINT PK_Customers
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT PK_Customers PRIMARY KEY ([ID])
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_DataChecks CHECK ([Priority] LIKE 'abc') AND ISNUMERIC ([Postal]) = 1 AND LEN([Postal]) = 9 AND [Active] IS (0,1))
- B. ALTER TABLE [dbo].[Customers] DROP CONSTRAINT PK_Customers
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT PK_Customers PRIMARY KEY NONCLUSTERED ([ID])
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_Priority CHECK ([Priority] IN ('a', 'b', 'c'))
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_Postal CHECK ([Postal] LIKE '[0-9][0-9][0-9][0-9][0-9]')
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT DF_Active DEFAULT (1) FOR [Active]
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_Active CHECK ([Active] IN (0,1))
- C. ALTER TABLE [dbo].[Customers] DROP CONSTRAINT PK_Customers
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT PK_Customers PRIMARY KEY ([ID])
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_DataChecks CHECK ([Priority] IN 'abc') AND ISNUMERIC ([Postal]) = 1 AND [Active] IS (0,1))
- D. ALTER TABLE [dbo].[Customers] ADD CONSTRAINT PK_Customers PRIMARY KEY CLUSTERED ([ID])
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_Priority VALIDATE ([Priority] IN ('a', 'b', 'c'))
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_Postal VALIDATE ([Postal] LIKE '[0-9][0-9][0-9][0-9][0-9]')
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT DF_Active DEFAULT (1) FOR [Active]
 ALTER TABLE [dbo].[Customers] ADD CONSTRAINT CK_Active VALIDATE ([Active] IS (0,1))

A. Option AB. Option BC. Option CD. Option D

QUESTION 162 Drag and Drop Question You create a database named AdventureWorks. You want to create a new table to store customer reviews for all products within the database. The table must meet the following requirements: Which three Transact-SQL statements should you use? (To answer, move the appropriate statements from the list of statements to the answer area and arrange them in the correct order.)

```
USE AdventureWorks
GO

ALTER TABLE ProductReview
ALTER COLUMN Review INT NULL ADD SPARSE;
GO

ALTER TABLE ProductReview
ALTER COLUMN Review ADD SPARSE ;
GO

CREATE TABLE ProductReviews
(
    CustomerID nchar(4) NOT NULL,
    ProductID nchar(4) NOT NULL,
    Rating nfloat NOT NULL,
    Review varchar(500) NOT NULL,
    CONSTRAINT PK_ProductReviews PRIMARY KEY
    CLUSTERED
    (
        CustomerID ASC,
        ProductID ASC
    )
)
GO

CREATE TABLE ProductReviews
(
    CustomerID nchar(4) NOT NULL,
    ProductID nchar(4) NOT NULL,
    Rating nfloat NOT NULL,
    Review varchar(500) NULL,
    CONSTRAINT PK_ProductReviews PRIMARY KEY
    CLUSTERED
    (
        CustomerID ASC,
        ProductID ASC
    )
)
GO
```

Answer:

```
USE AdventureWorks
GO

ALTER TABLE ProductReviews
ALTER COLUMN Review NOT NULL ADD SPARSE;
GO

ALTER TABLE ProductReviews
ALTER COLUMN Review ADD SPARSE ;
GO

CREATE TABLE ProductReviews
(
    CustomerID nchar(4) NOT NULL,
    ProductID nchar(4) NOT NULL,
    Rating nfloat NOT NULL,
    Review varchar(500) NOT NULL,
    CONSTRAINT PK_ProductReviews PRIMARY KEY
    CLUSTERED
    (
        CustomerID ASC,
        ProductID ASC
    )
)
GO

CREATE TABLE ProductReviews
(
    CustomerID nchar(4) NOT NULL,
    ProductID nchar(4) NOT NULL,
    Rating nfloat NOT NULL,
    Review varchar(500) NULL,
    CONSTRAINT PK_ProductReviews PRIMARY KEY
    CLUSTERED
    (
        CustomerID ASC,
        ProductID ASC
    )
)
GO

CREATE TABLE ProductReviews
(
    CustomerID nchar(4) NOT NULL,
    ProductID nchar(4) NOT NULL,
    Rating nfloat NOT NULL,
    Review varchar(500) SPARSE NOT NULL,
    CONSTRAINT PK_ProductReviews PRIMARY KEY
    CLUSTERED
    (
        CustomerID ASC,
        ProductID ASC
    )
)
GO
```

QUESTION 163 You administer a Microsoft SQL Server 2008 database that contains a table named Sales.SalesOrderDetail and a view named Sales.ProductOrders. The view has the following definition:

```
CREATE VIEW Sales.ProductOrders AS
SELECT
    ProductID,
    COUNT(*) as NbrOfOrders,
    SUM(OrderQty) as TotalOrderQty
FROM
    Sales.SalesOrderDetail
GROUP BY
    ProductID
GO
```

The Sales.SalesOrderDetail table contains 5 million rows. Report queries that join to this view consume excessive disk I/O. You need to create an index on the view. Which Transact-SQL statement or statements should you use?

- A. IF EXISTS (SELECT * FROM sys.views WHERE object_id = OBJECT_ID(N'[Sales].[ProductOrders]')) DROP VIEW [Sales].[ProductOrders] GO CREATE VIEW Sales.ProductOrders AS SELECT ProductID, COUNT(*) as NbrOfOrders, SUM(OrderQty) as TotalOrderQty FROM Sales.SalesOrderDetail GROUP BY ProductID GO CREATE UNIQUE CLUSTERED INDEX IX_V_ProductOrders ON Sales.ProductOrders (ProductID)
- B. IF EXISTS (SELECT * FROM sys.views WHERE object_id = OBJECT_ID(N'[Sales].[ProductOrders]')) DROP VIEW [Sales].[ProductOrders] GO CREATE VIEW Sales.ProductOrders WITH SCHEMABINDING AS SELECT ProductID, COUNT(*) as NbrOfOrders, SUM(OrderQty) as TotalOrderQty FROM Sales.SalesOrderDetail GROUP BY ProductID GO CREATE UNIQUE CLUSTERED INDEX IX_V_ProductOrders ON Sales.ProductOrders (ProductID)
- C. ALTER VIEW Sales.ProductOrders WITH SCHEMABINDING AS SELECT ProductID, COUNT(*) as NbrOfOrders, SUM(OrderQty) as TotalOrderQty FROM Sales.SalesOrderDetail GROUP BY ProductID GO CREATE UNIQUE CLUSTERED INDEX IX_V_ProductOrders ON Sales.ProductOrders (ProductID)
- D. IF EXISTS (SELECT * FROM sys.views WHERE object_id = OBJECT_ID(N'[Sales].[ProductOrders]')) DROP VIEW [Sales].[ProductOrders] GO CREATE VIEW Sales.ProductOrders WITH SCHEMABINDING AS SELECT ProductID, COUNT(*) as NbrOfOrders, SUM(OrderQty) as TotalOrderQty FROM Sales.SalesOrderDetail GROUP BY ProductID GO CREATE UNIQUE CLUSTERED INDEX IX_V_ProductOrders ON Sales.ProductOrders (ProductID)

A. Option AB. Option BC. Option CD. Option D Answer: C QUESTION 164 You administer a Microsoft SQL Server 2008 database for an inventory management system. The application contains a product table that has the following definition:

```
CREATE TABLE [Production].[Product]
(
    [ProductID] [int] IDENTITY(1,1) NOT NULL,
    [Name] [nvarchar](50) NOT NULL,
    [ProductNumber] [nvarchar](25) NOT NULL,
    [Color] [nvarchar](15) NOT NULL,
    [Class] [nchar](2) NOT NULL,
    [Style] [nchar](10) NOT NULL,
    [Modifiable] [bit] NOT NULL,
    CONSTRAINT [PK_Product] PRIMARY KEY
    (
        [ProductID] ASC
    ) ON [PRIMARY]
) ON [PRIMARY]
```

You want to add a new field to the Product table to meet the following requirements: You need to add a field named User_Data_1 to support integer values ranging from -10 through 10. Which SQL statement should you use?

A. ALTER TABLE [Production].[Product] ADD [User_Data_1] TINYINTB.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] SMALLINTC.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] INTD.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] BIGINTE.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] BITF.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NUMERIC(11,6)G.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NUMERIC(6,11)H.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NUMERIC(5,6)I.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] SMALLMONEYJ.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] MONEYK.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] CHAR(100)L.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] VARCHAR(100)M.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NCHAR(100)N.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NVARCHAR(100)O.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] SMALLDATETIMEP.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] DATETIMEQ.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] DATETIME2R.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] DATE

Answer: B QUESTION 165 You administer a Microsoft SQL Server 2008 database for an inventory management system. The application contains a product table that has the following definition:

```
CREATE TABLE [Product]
(
  [ProductID] [int]
  [Name] [nvarchar]
  [ProductNumber] [int]
  [Color] [nvarchar]
  [Class] [nchar] (2)
  [Style] [nchar] (2)
  [AcquisitionDate] [datetime]
  [ModifiedDate] [datetime]
)
CONSTRAINT [PK_Product]
(
  [ProductID] ASC
) ON [PRIMARY]
) ON [PRIMARY]
GO
```

You want to add a new field to the Product table to meet the following requirements: You need to add a field named User_Data_1 to support only values that are 1 or 0. Which SQL statement should you use?

A. ALTER TABLE [Production].[Product] ADD [User_Data_1] TINYINTB.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] SMALLINTC.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] INTD.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] BIGINTE.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] BITF.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NUMERIC(11, 6)G.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NUMERIC(6, 11)H.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NUMERIC(5,6)I.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] SMALLMONEYJ.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] MONEYK.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] CHAR(100)L.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] VARCHAR(100)M.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NCHAR(100)N.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] NVARCHAR(100)O.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] SMALLDATETIMEP.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] DATETIMEQ.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] DATETIME2R.
 ALTER TABLE [Production].[Product] ADD [User_Data_1] DATE

Answer: E QUESTION 166 You administer a Microsoft SQL Server 2008 database that contains a table named dbo.SalesOrders. The table has the following definition:

```
CREATE TABLE [dbo].[SalesOrders]
(
  [SalesOrderNumber] [int]
  [FullDateAlternateKey] [datetime]
  [CustomerName] [nvarchar]
  [AddressLine] [nvarchar]
  [City] [nvarchar]
  [StateProvinceName] [nvarchar]
  [SalesPerson] [nvarchar]
  [SalesAmount] [money]
)
CONSTRAINT [PK_SalesOrders]
(
  [SalesOrderNumber]
) ON [PRIMARY]
) ON [PRIMARY]
GO
```

The SalesOrder table contains one million rows. You want to create a report that meets the following requirements:

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

A. Option AB. Option BC. Option CD. Option D Answer: D QUESTION 168 You are a developer for a Microsoft SQL Server 2008 R2 database instance. You create tables named order, customer, and product as follows:

```

CREATE TABLE [dbo].[order]
([OrderID] [int],
[ProductID] [int],
[CustomerID] [int],
[OrderDate] [datetime]);

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

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

You need to write a query to identify all customers who have ordered for an average amount of more than 500 or more from September 01, 2011. Which SQL query should you use?

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

C. SELECT
    c.CustomerName,
    COUNT(o.OrderID) AS Orders
FROM
    customer c INNER JOIN
    [order] o ON c.CustomerID = o.CustomerID
GROUP BY
    c.CustomerName;

D. SELECT
    c.CustomerName,
    COUNT(o.OrderID) AS Orders
FROM
    customer c INNER JOIN
    [order] o ON c.CustomerID = o.CustomerID
GROUP BY
    c.CustomerName
HAVING
    COUNT(o.OrderID) > 10;

E. SELECT
    c.CustomerName,
    AVG(p.SalePrice) AS Sales
FROM
    product p INNER JOIN
    [order] o ON p.ProductID = o.ProductID INNER JOIN
    customer c ON o.CustomerID = c.CustomerID
WHERE
    o.OrderDate > '09/01/2011'
GROUP BY
    c.CustomerName
HAVING
    AVG(p.SalePrice) >= 500;
```

Braindump2go.com

```
F. SELECT
    c.CustomerName,
    AVG(p.SalePrice) AS Sales
FROM
    product p INNER JOIN
    [order] o ON p.ProductID = o.ProductID INNER JOIN
    customer c ON o.CustomerID = c.CustomerID
WHERE
    o.OrderDate > '09/01/2011' AND
    AVG(p.SalePrice) >= 500;

G. SELECT
    p.ProductName,
    DATEPART(mm, o.OrderDate) OrderMonth,
    SUM(p.SalePrice) AS Sales
FROM
    product p INNER JOIN
    [order] o ON p.ProductID = o.ProductID
GROUP BY CUBE(p.ProductName, DATEPART(mm, o.OrderDate));

H. SELECT
    p.ProductName,
    DATEPART(mm, o.OrderDate) OrderMonth,
    SUM(p.SalePrice) AS Sales
FROM
    product p INNER JOIN
    [order] o ON p.ProductID = o.ProductID
GROUP BY CUBE;

I. SELECT
    p.ProductName,
    DATEPART(mm, o.OrderDate) OrderMonth,
    SUM(p.SalePrice) AS Sales
FROM
    product p INNER JOIN
    [order] o ON p.ProductID = o.ProductID
GROUP BY p.ProductName, OrderMonth;

J. SELECT
    p.ProductName,
    DATEPART(mm, o.OrderDate) OrderMonth,
    SUM(p.SalePrice) AS Sales
FROM
    product p INNER JOIN
    [order] o ON p.ProductID = o.ProductID
GROUP BY p.ProductName, DATEPART(mm, o.OrderDate);
```

Braindump2go.com

A. Option AB. Option BC. Option CD. Option DE. Option EF. Option FG. Option GH. Option HI. Option IJ. Option J
Answer: E
QUESTION 169 You are a developer for a Microsoft SQL Server 2008 R2 database instance. You create tables named order, customer, and product as follows:


```
CREATE TABLE [dbo].[order]
([OrderID] [int],
[ProductID] [int],
[CustomerID] [int],
[OrderDate] [datetime]);

CREATE TABLE [dbo].[customer]
([CustomerID] [int],
[CustomerName] [varchar](100),
[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 sum the sales of all orders by the following entries: Which SQL query should you use?

- A. SELECT
c.CustomerName,
p.ProductName,
SUM(p.SalePrice) AS Sales
FROM
product p INNER JOIN
[order] o ON p.ProductID =
customer c ON o.CustomerID
GROUP BY GROUPING SETS ((c.C
- B. SELECT
c.CustomerName,
p.ProductName,
SUM(p.SalePrice) AS Sales
FROM
product p INNER JOIN
[order] o ON p.ProductID =
customer c ON o.CustomerID
GROUP BY GROUPING SETS ((c.C
- C. SELECT
c.CustomerName,
COUNT(o.OrderID) AS Orders
FROM
customer c INNER JOIN
[order] o ON c.CustomerID =
o.CustomerID
GROUP BY
c.CustomerName;
- D. SELECT
c.CustomerName,
COUNT(o.OrderID) AS Orders
FROM
customer c INNER JOIN
[order] o ON c.CustomerID =
o.CustomerID
GROUP BY
c.CustomerName
HAVING
COUNT(o.OrderID) > 10;
- E. SELECT
c.CustomerName,
AVG(p.SalePrice) AS Sales
FROM
product p INNER JOIN
[order] o ON p.ProductID =
customer c ON o.CustomerID
WHERE
o.OrderDate > '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
[order] o ON p.ProductID = o.ProductID INNER JOIN
customer c ON o.CustomerID = c.CustomerID
WHERE
o.OrderDate > '09/01/2011' AND
AVG(p.SalePrice) >= 500
- G. SELECT
p.ProductName,
DATEPART(mm, o.OrderDate) OrderMonth,
SUM(p.SalePrice) AS Sales
FROM
product p INNER JOIN
[order] o ON p.ProductID = o.ProductID
GROUP BY CUBE(p.ProductName, DATEPART(mm, o.OrderDate));
- H. SELECT
p.ProductName,
DATEPART(mm, o.OrderDate) OrderMonth,
SUM(p.SalePrice) AS Sales
FROM
product p INNER JOIN
[order] o ON p.ProductID = o.ProductID
GROUP BY CUBE;
- I. SELECT
p.ProductName,
DATEPART(mm, o.OrderDate) OrderMonth,
SUM(p.SalePrice) AS Sales
FROM
product p INNER JOIN
[order] o ON p.ProductID = o.ProductID
GROUP BY p.ProductName, OrderMonth;
- J. SELECT
p.ProductName,
DATEPART(mm, o.OrderDate) OrderMonth,
SUM(p.SalePrice) AS Sales
FROM
product p INNER JOIN
[order] o ON p.ProductID = o.ProductID
GROUP BY p.ProductName, DATEPART(mm, o.OrderDate);

A. Option AB. Option BC. Option CD. Option DE. Option EF. Option FG. Option GH. Option HI. Option IJ.
Option J Answer: G QUESTION 170 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](100),
[City] [varchar](100),
[State] [varchar](50),
[ZipCode] [varchar](5));

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

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
p.ProductID = com.ProductID
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

```

C 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

C 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));

C 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;

C 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;

C 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.ComplaintDate);
    AVG(p.SalePrice) >= 500
    
```

A. Option AB. Option BC. Option CD. Option DE. Option EF. Option FG. Option GH. Option HI. Option IJ. Option J
 Answer: D For those who feel the overwhelming anxiety before their 70-433 exam, Braindump2go Latest updated 70-433 Exam Dumps will help you Pass 100% in a short time preparation! 70-433 Exam Dumps PDF & VCE Full Version Instant Download!

Compared Before Buying Microsoft 70-433 PDF & VCE!		
Pass4sure	Braindump2go 100% Pass OR Money Back	Test King
202 Q&As - Practice	210 Q&As - Real Questions	202 Q&As - Practice
\$125.99	\$99.99	\$124.99
No Discount	Coupon Code: BDNT2014	No Discount

<http://www.braindump2go.com/70-433.html>