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.lead4pass.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 R2 database that has a table named Customer. The table has the following definition:

```
CREATE TABLE Customer
   (CustomerID int NOT NULL PRIMARY KEY,
    FirstName varchar(255) NOT NULL,
    LastName varchar(255) NOT NULL,
    CustomerAddress varchar(1024))
```

The database also has a table named PreferredCustomerList. Data will be added to the PreferredCustomerList table regularly. The PreferredCustomerList table has the following definition:

```
CREATE TABLE PreferredCustomerList
   (FirstName varchar(255) NOT NULL,
    LastName varchar(255) NOT NULL)
```

You need to create a view that returns all records and columns of the Customer table that are also present in the PreferredCustomerList table. Which Transact-SQL statement should you use?

```
CREATE VIEW vw_ValidCustomer
AS
SELECT c.CustomerID,
   c.FirstName,
   c.LastName,
   c.CustomerAddress
FROM Customer c
LEFT OUTER JOIN PreferredCustomerList cel
   ON c.Firstname = cel.FirstName
   AND c.LastName = cel.LastName
WHERE cel.LastName IS NULL
```

A.

B.
```
CREATE VIEW vw_ValidCustomer
AS
SELECT c.CustomerID,
  c.FirstName,
  c.LastName,
  c.CustomerAddress
FROM Customer c
INTERSECT
SELECT c.CustomerID,
  c.FirstName,
  c.LastName,
  c.CustomerAddress
FROM Customer c
INNER JOIN PreferredCustomerList cel
  ON c.Firstname = cel.FirstName
  AND c.LastName = cel.LastName
```

C.
```
CREATE VIEW vw_ValidCustomer
AS
SELECT c.CustomerID,
  c.FirstName,
  c.LastName,
  c.CustomerAddress
FROM Customer c
LEFT OUTER JOIN PreferredCustomerList cel
  ON c.Firstname = cel.FirstName
  AND c.LastName = cel.LastName
WHERE cel.FirstName IS NULL
```

D.
```
CREATE VIEW vw_ValidCustomer
AS
SELECT FirstName,
  LastName
FROM Customer c
EXCEPT
SELECT FirstName,
  LastName
FROM PreferredCustomerList
```

B. C. D.

Correct Answer: B

QUESTION 2

You have a table named Product.

You need to increase product prices for only the vendor named Coho Winery by 10 percent and then return a list of the products and updated prices.

Which code segment should you use?

A. UPDATE Product SET Price = Price * 1.10, ProductName = ProductName WHERE Product.VendorName = \\'Coho Winery\\'

B. UPDATE Product SET Price = Price * 1.10 OUTPUT inserted.ProductName, deleted.Price WHERE Product.VendorName = \\'Coho Winery\\'

C. UPDATE Product SET Price = Price * 1.10 OUTPUT inserted.ProductName, inserted.Price WHERE Product.VendorName = \\'Coho Winery\\'

D. UPDATE Product SET Price = Price * 1.10, VendorName = \\'Coho Winery\\' OUTPUT inserted.ProductName, inserted.Price

Correct Answer: C

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 are a database developer for your organization. You have an application hosted on Microsoft SQL Server 2008 R2. One of the tables in the application database has the following definition:

```
CREATE TABLE XMLData
(XMLPage xml NOT NULL)
```

No indexes, keys, or constraints are defined along with the table.

The table currently contains more than one million entries. An example of the XML data is shown below:

```
Company
  <Employee Name = "Mark" = "Department Finance"/>
  <Employee Name = "Peter" = "Department Sales"/>
  <Employee Name = "Susan" = "Department Facilities"/>
Company
```

The stored procedure has the following definition:

```
CREATE PROCEDURE p_GetFinanceUser
(@Name varchar(20))
AS

SELECT Name = Company.Employee.value('@Name','varchar(20)')
  ,Department = Company.Employee.value('@Department','varchar(20)')
FROM XMLData c
CROSS APPLY XMLPage.nodes('/Company/Employee') Department(Employee)
WHERE Company.Employee.exist('@Department[.="Finance"]') = 1
```


Users report that the response time of the stored procedure has decreased.

You need to ensure that the response time of the stored procedure is improved. 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.)

Select and Place:

```
CREATE INDEX idx_XMLPage ON XMLData
(XMLPage)
```

```
CREATE PRIMARY XML INDEX idx_XMLPage ON
XMLData (XMLPage)
```

```
CREATE XML INDEX idx_XMLPage_PATH ON XMLData
(XMLPage)
USING XML INDEX idx_XMLPage
FOR PATH
```

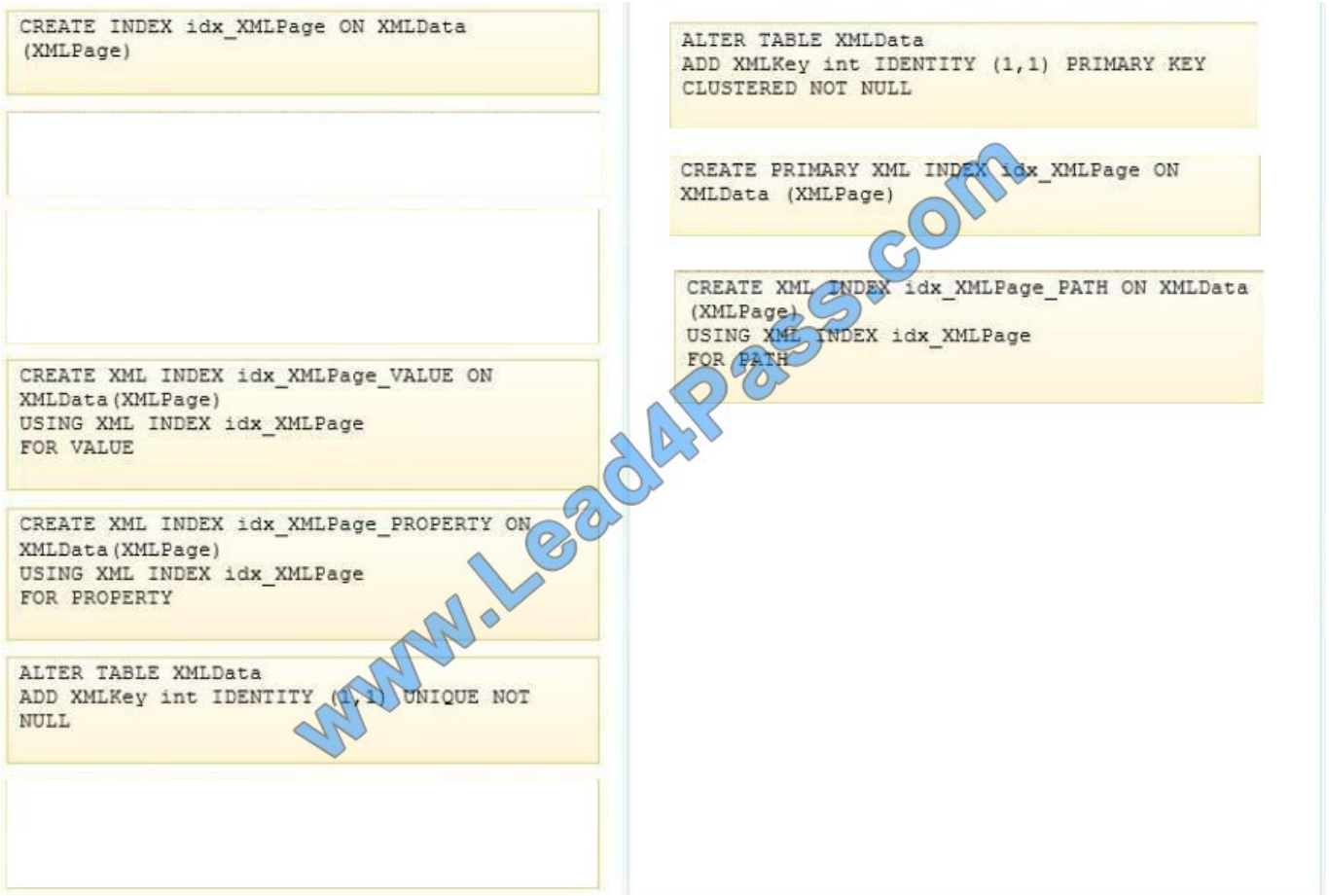
```
CREATE XML INDEX idx_XMLPage_VALUE ON
XMLData(XMLPage)
USING XML INDEX idx_XMLPage
FOR VALUE
```

```
CREATE XML INDEX idx_XMLPage_PROPERTY ON
XMLData(XMLPage)
USING XML INDEX idx_XMLPage
FOR PROPERTY
```

```
ALTER TABLE XMLData
ADD XMLKey int IDENTITY (1,1) UNIQUE NOT
NULL
```

```
ALTER TABLE XMLData
ADD XMLKey int IDENTITY (1,1) PRIMARY KEY
CLUSTERED NOT NULL
```


Correct Answer:



QUESTION 5

You work for an international charity organization. You are writing a query to list the highest 100 different amounts that were donated. You have written the following code segment (Line numbers are included for reference only):

01 SELECT * 02 FROM (SELECT Customer.CustomerID, SUM(TotalDue) AS TotalGiven, 04 FROM Customer 05 JOIN SalesOrder 06 ON Customer.CustomerID = SalesOrder.CustomerID 07 GROUP BY Customer.CustomerID) AS DonationsToFilter 08 WHERE FilterCriteria