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

- **Instant Download** After Purchase
- **100% Money Back** Guarantee
- **365 Days** Free Update
- **800,000+** Satisfied Customers

QUESTION 1

You have a table named Person that contains a nvarchar column named Surname. The Person table currently has a clustered index on PersonID.

The Surname column contains Russian and Japanese characters.

The following code segment will be used to search by Surname.

IF @lang =\\'Russian\\' SELECT PersonID, Surname FROM Person WHERE Surname = @SearchName COLLATE Cyrillic_General_CI_AS if @lang = \\'Japanese\\' SELECT PersonID, Surname FROM Person WHERE Surname = @SearchName COLLATE Japanese_CI_AS_KS

You need to enable SQL Server to perform an index seek for these queries.

What should you do?

A. Create an index on the Surname column.

B. Create a computed column for each collation that needs to be searched. Create an index on the Surname column.

C. Create a computed column for each collation that needs to be searched. Create an index on each computed column.

D. Create a new column for each collation that needs to be searched and copy the data from the Surname column. Create an index on each new column.

Correct Answer: C

-- Add computed columns with different collations.

ALTER TABLE Person

ADD Surname_RU AS Surname COLLATE Cyrillic_General_CI_AS, Surname_JP AS Surname COLLATE Japanese_CI_AS_KS; -- Create an index on the computed columns.

CREATE NONCLUSTERED INDEX IX_Person_Surname_RU ON Person (Surname_RU); CREATE NONCLUSTERED INDEX IX_Person_Surname_JP ON Person (Surname_JP); GO

QUESTION 2

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 3

You have a SQL Server database. The database contains two schemas named Marketing and Sales. The Marketing schema is owned by a user named MarketingManager. The Sales schema is owned by a user named SalesManager.

A user named John must be able to access the Sales.Orders table by using a stored procedure named Marketing.GetSalesSummary. John is not granted a SELECT permission on the Sales.Orders table. A user named SalesUser does have

SELECT permission on the Sales.Orders table. You need to implement appropriate permissions for John and the stored procedure Marketing.GetSalesSummary.

What should you do?

A. Marketing.GetSalesSummary should be created by using the EXECUTE AS \\'SalesUser\\' clause. John should be granted EXECUTE permission on Marketing.GetSalesSummary.

B. Marketing.GetSalesSummary should be created by using the EXECUTE AS OWNER clause. John should be granted EXECUTE WITH GRANT OPTION on Marketing.GetSalesSummary.

C. Marketing.GetSalesSummary should be created by using the EXECUTE AS CALLER clause. John should be granted IMPERSONATE permission for the user named SalesUser.

D. Marketing.GetSalesSummary should be created without an EXECUTE AS clause. John should be granted SELECT permission on the Sales.Orders table.

Correct Answer: A

1.

When the module is executed, the Database Engine first verifies that the user executing the module has EXECUTE permission on the module. So John should be granted EXECUTE permission on Marketing. GetSalesSummary stored procedure.

2.

Additional permissions checks on objects that are accessed by the module are performed against the user account specified in the EXECUTE AS clause. The user executing the module is, in effect, impersonating the specified user. Because John is not granted a SELECT permission on the Sales.Orders table which is referenced by the stored procedure, EXECUTE AS CALLER is not suitable. (CALLER specifies the statements inside the module are executed in the context of the caller of the module. The user executing the module must have appropriate permissions not only on the module itself, but also on any database objects that are referenced by the module.) Because the user named SalesUser DOES have SELECT permission on the Sales.Orders table, he can be specified in EXECUTE AS clause. It means that Marketing. GetSalesSummary stored procedure should be created by using the EXECUTE AS \\'SalesUser\\' clause.

QUESTION 4

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 5

Your database contains Products and Orders tables. You need to write a query which return ProductID of the products which have not been placed in any order.

Which operator can you use?

A. Union

B. Union ALL

C. Intersect

D. Except

Correct Answer: D

EXCEPT returns any distinct values from the left query that are not also found on the right query.

* Example: The following query returns any distinct values from the query to the left of the EXCEPT operand that are not also found on the right query. USE AdventureWorks2012; GO SELECT ProductID FROM Production.Product EXCEPT SELECT ProductID FROM Production.WorkOrder ; --Result: Products without work orders

To Read the Whole Q&As, please purchase the Complete Version from Our website.

Try our product !

100% Guaranteed Success
100% Money Back Guarantee
365 Days Free Update
Instant Download After Purchase
24x7 Customer Support
Average 99.9% Success Rate
More than 800,000 Satisfied Customers Worldwide
Multi-Platform capabilities - Windows, Mac, Android, iPhone, iPod, iPad, Kindle

We provide exam PDF and VCE of Cisco, Microsoft, IBM, CompTIA, Oracle and other IT Certifications.
You can view Vendor list of All Certification Exams offered:

https://www.geekcert.com/allproducts

Need Help

Please provide as much detail as possible so we can best assist you.
To update a previously submitted ticket: