Views and Partition Tables

Easy

1. Create a view on tables employee an department table

Employee:

EmployeeID

FirstName

LastName

DeptID

Address

City

State

Country

Department:

DeptID

DeptName

DeptDesc

DeptAddedBy

PostedDate

1. List Types of views with example for each

Medium

1. View was created by using the below script now build a query to Modify the view by

adding a WHERE clause to limit the rows returned

Hint: use HireDate and give a specific date to limit rows

CREATE VIEW HumanResources.EmployeeHireDate

AS

SELECT p.FirstName, p.LastName, e.HireDate

FROM HumanResources.Employee AS e

JOIN Person.Person AS p

ON e.BusinessEntityID = p.BusinessEntityID;

1. How many columns can a view have
2. 1023 b)1042 c)1012 d)1024
3. Write a query for creating an indexed view which will only pull the first and last name of the user from the users table, and all of the columns from the Contacts table and automatically pick up any new columns that are added to the Contacts table, with no further modifications. (Schemabinding)
4. Create a clustered index on the above view which has been created using a UserId will only be in the contacts table once per Organization, so we can make a unique clustered index spanning the OrgId and UserId columns(Revisit)
5. Create partition view on employee table and department table, drop if it already exists and also print the message when view is created or dropped. Columns of employee table and department table (Revisit)

Employee:

EmployeeID

FirstName

LastName

DeptID

Address

City

State

Country

Department:

DeptID

DeptName

DeptDesc

DeptAddedBy

PostedDate

1. Create a partition function with range specified, first 10 records are stored in one filegroup and rest are stored in different filegroup
2. Create a trigger for inserting rows into view manger\_info where view looks as below

CREATE VIEW manager\_info AS

SELECT e.ename

,e.empno

,d.dept\_type

,d.deptno

,p.prj\_level

,p.projno

FROM Emp\_tab e, Dept\_tab d, Project\_tab p

WHERE e.empno = d.mgr\_no

AND d.deptno = p.resp\_dept;

1. Create partition view across the three server tables, partitiontable1 in server1, partitiontable2 in server2 and partitiontable3 in server3

11. Create a view “alphabetical\_list\_of\_products” which uses the Production.ProductsubCategory and Production.Product tables in adventureworks database to display the products name in the Production.Product table in alphabetical order, for all the products with productsubcategoryid column values common in both the tables

CREATE VIEW "Alphabetical\_list\_of\_products" AS

SELECT p.name productname , ps.Name Category\_Name

FROM production.productsubcategory ps INNER JOIN Production.Product p

ON ps.ProductSubcategoryID = p.ProductSubcategoryID

select \* from Alphabetical\_list\_of\_products

12.Create a table student with columns student\_id, student\_name, student\_gender.

Create a view student\_view on this table , on all the columns.

Write a query which will display all the column names defined in the view. ( 5 marks)

* SYSCOLUMNS — stores the names of the columns defined in the view.

13. Create a view Customers\_without\_orders on customerid and accountnumber columns in the table Sales.Customer in adventureworks database; **to get the list of customers who do not have any orders**; based on the **condition** where customerid column values present in Sales.Customer are not present in Sales.SalesOrderHeader.

CREATE VIEW vwCustomersWithoutOrders

AS

SELECT CustomerID, AccountNumber

FROM Sales.Customer

WHERE NOT EXISTS

(SELECT CustomerID FROM Sales.SalesOrderHeader WHERE

Sales.Customer.CustomerID = Sales.SalesOrderHeader.CustomerID)

--This example selects all customers who do not have orders. You call the view like this:

SELECT \* FROM vwCustomersWithoutOrders

14.Create following four tables with the data as below:

TableA   
ID Name     
1   John      
2   Paul      
   
TableB   
ID Name

1   Ringo   
2   George   
   
TableC   
ID Name     
1   Bob      
   
TableD   
ID Name     
1   Kate

Create a single view to display all the columns from all the tables created.

Complex

1. Build a query to Attach Partition Scheme to filegroups, then partition should be created on primary and secondary filegroup
2. Create a table with Partition Key and Partition Scheme where ID column should be the Partition Key(Provide schema)
3. How to verify whether rows inserted in Partitions or not
4. sys.partitions b) sys.columns c) syspartitions d) sys.functions
5. A partition column cannot be

a)computed b)identity c)timestamp d)all of the above

Database console commands

Medium

1. create a database and specifying a collation name and TRUSTYWORTHY and DB\_CHAINING options
2. write a query for checking the consistency of disk space allocation structures for a ‘adventureworks’ database
3. write a query which checks for catalog consistency within the ‘adventureworks’ database
4. Build a query which helps to check the integrity of a specified constraint or all constraints on a employee table in the ‘adventureworks’ database
5. How can we check the logical and physical integrity of all the objects in the ‘adventureworks’ database
6. Which console command displays current query optimization statistics for a table or indexed view
7. Find out the database console commands for miscellaneous statements

a) DBCC HELP b) DBCC PINTABLE c) DBCC CHECKFILEGROUP d) A&B alone

1. Find out the for console commands for validation statements

a) DBCC CHECKFILEGROUP b) DBCC CHECKIDENT

c) DBCC CHECKCONSTRAINTS d) all of the above

1. Find out the for database console commands for status statements

a) DBCC USEROPTIONS b) DBCC SHOW\_STATISTICS

c) DBCC SQLPERF d) all of the above

1. Find out the for database console commands for maintenance statements

a) DBCC UPDATEUSAGE b) DBCC CLEANTABLE

c) A&B alone d) none of these

Creating database

Medium

1. creates the database Sales that has the following filegroups:

The primary filegroup with the files Spri1\_dat and Spri2\_dat. The FILEGROWTH increments for these files are specified as 15%.

A filegroup named SalesGroup1 with the files SGrp1Fi1 and SGrp1Fi2.

A filegroup named SalesGroup2 with the files SGrp2Fi1 and SGrp2Fi2

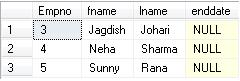
1. Create a database Sales that specifies the data and transaction log files. Because the keyword PRIMARY is not used, the first file (Sales\_dat) becomes the primary file. Because neither MB nor KB is specified in the SIZE parameter for the Sales\_dat file, it uses MB and is allocated in megabytes. The Sales\_log file is allocated in megabytes because the MB suffix is explicitly stated in the SIZE parameter.
2. Create a database Archive by specifying multiple data and transaction log files that has three 100-MB data files and two 100-MB transaction log files. This places the database on the D: drive instead of with the master database.
3. Build a query to verify the database files and sizes where database name is ‘archive’

Database files and filegroups

Complex

1. Write a query to find out where your data and transaction log files are for a given database
2. Write a query to find database file location
3. Suppose we have two tables in which Import\_Employee is a source table and Employee is a target table.

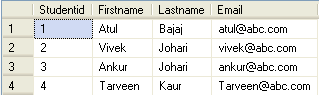
[](http://2.bp.blogspot.com/-IcfxQsRTMU0/T4ulNj55WaI/AAAAAAAAA6k/vbhS27-4CVw/s1600/Merged_1.jpg)----------------------🡪 Import\_Employee (source table)

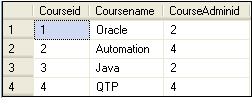
[](http://4.bp.blogspot.com/-Lrdfxx_e8Po/T4ulWku1reI/AAAAAAAAA6s/tgMUdEU0bS8/s1600/Merged_2.jpg)----------------🡪 Employee (target table)

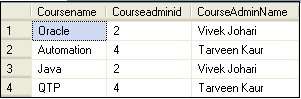
Suppose we have the following requirements:-  
1) If a empno is exists in both the table then the corresponding values of the column Fname and Lname for the empno in the table Employee is updated by the corresponding values of the column  firstname and  lastname of the Empno from the source table Import\_Employee  .  
2) If there is no corresponding empno in the Target table Employee for a empno of the table Import\_Employee, then the data from the Source table is inserted into the target table for that empno.  
3) If there exists a records for a empno in the Target table whose corresponding records are not in the Source table, these these record should be deleted from the target table.(7 Marks)

Ans: MERGE employee AS TARGET  
  
USING Import\_Employee AS SOURCE  
  
ON TARGET.empno=SOURCE.empno   
  
WHEN MATCHED   
  
THEN update set TARGET.fname=SOURCE.firstname,  
  
TARGET.lname = SOURCE.lastname  
  
WHEN NOT MATCHED BY TARGET THEN   
  
insert(empno,fname,lname)  
  
values(SOURCE.empno,SOURCE.firstname,SOURCE.lastname)  
  
WHEN NOT MATCHED BY SOURCE THEN   
  
Delete;  
  
Select \* from  Employee

1. we want to get the details of the Courses including the name of their course admin from the Course table (3Marks)

[](http://4.bp.blogspot.com/_8iWtCcGgcH8/SsI2s5I4_PI/AAAAAAAAAYw/BoZVJXgSlOk/s1600-h/Studenttable.PNG)----------🡪student table

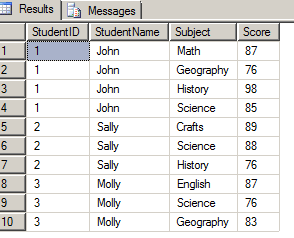
[](http://2.bp.blogspot.com/_8iWtCcGgcH8/SsI3jMZNAsI/AAAAAAAAAY4/eds8tiS3qXQ/s1600-h/CourseTable.PNG)--------------------🡪course table

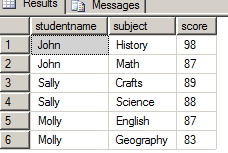
[](http://3.bp.blogspot.com/_8iWtCcGgcH8/SsI6yM2mTzI/AAAAAAAAAZQ/JL9C1oXewdY/s1600-h/CorrelatedSubquery_1.PNG)-----------🡪expected output

Ans: select Coursename ,Courseadminid,(select Firstname+' '+Lastname  from student where studentid=Course.courseadminid)as CourseAdminName from course

1. Build a query for the below grades table to display highest top two scores of each student. (3Marks)

Hint: use row number with over by clause

-----------------🡪grades table

-----------------------------🡪excepted result set

Ans: select studentname, subject, score

from

(

select studentname, subject, score,

row\_number() over (partition by studentid order by score desc) as rownum

from grades

) dt

where rownum<=2

1. Delete records from Both Student details and Library. Database will only commit, if both delete statement execute successfully, If fails it will Rollback. Intentionally I have passed a wrong roll ( Which causes) the exception and transaction will rollback

BEGIN TRY

BEGIN TRANSACTION

DELETE FROM StudentDetails

DELETE FROM Library WHERE Roll = 'a'

COMMIT

Insert into LogDetails(ID,Details) values ('1','Transaction Successful');

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0

ROLLBACK

END CATCH