

SQL SERVER Interview Questions & Answers - SET 1 (50 Questions)

<http://msbiskills.com/>

Question 1. What are statistics? Where they are used and how to check statistics.

Answer.

SQL server optimizer uses the statistics to choose the best query plan. If the statistics are incorrect (means outdated), then there are chances that SQL server engine might choose an incorrect query plan.

You can check statistics by using below command -

```
DBCC SHOW_STATISTICS('TestRIDInxs', 'Ix_Index')
```

Table Name - TestRIDInxs

Index Name - 'Ix_Index'

Output of the above query is given below-

Name	Updated	Rows	Rows Sampled	Steps	Density	Average key length	String Index	Filter Expression	Unfiltered Rows
Ix_Index	Apr 28 2015 2:02PM	1000001	1000001	2	1	8	NO	NULL	1000001

All density	Average Length	Columns
9.99999E-07	8	Eld

RANGE_HI_KEY	RANGE_ROWS	EQ_ROWS	DISTINCT_RANGE_ROWS	AVG_RANGE_ROWS
1	0	1	0	1
1000001	999999	1	999999	1

Question 2. What are the types of Fragmentations

Answer -

Fragmentation means the data is **NOT stored contiguously** on disk. There are two types of Fragmentation.

There are two kinds of fragmentation-

- 1. Logical fragmentation** – Here the next logical page as determined by the index order is not the next physical page in the data file.
- 2. Physical (or internal) fragmentation** – Here the space is being wasted on index pages. The rows inside the page are

not contiguous.

These can both affect query performance, as well as the expense of having to do the page split in the first place.

If you want to understand this in detail please visit - <http://www.sqlskills.com/blogs/paul/category/fragmentation/>

Question 3. Can we use GUID as Primary key in a table?

Answer – We can use GUID as primary key in a table But **we should NOT**. It will create fragmentation issue.

For details please visit - <http://msbiskills.com/2015/04/09/guid-causes-fragmentation-in-clustered-indexes/>

Question 4. What is the difference between Unique Index and Unique Constraint?

Answer –

A unique index is just an index, whereas a unique constraint is a unique index that's listed as a constraint object in the database. In the *sysobjects* table, the unique constraint will have an *xtype* value of "UQ".

Unique key basically creates a unique index internally to maintain uniqueness.

Note - Unique index and a unique constraint have same effect on a table. Performance is also same for both. Command to create unique index and unique constraint are given below.

--Command to add Index

```
CREATE UNIQUE INDEX Ix_Indexer ON TestRIDInxs (EId)
```

--Command to add constraint

```
ALTER TABLE TestRIDInxs  
ADD CONSTRAINT UNQ_Constraint UNIQUE (EId)
```

Now if you check the table definition (shortcut - ALT+F1) it will give you below information-

	index_name	index_description	index_keys				
1	ix_Index	nonclustered located on PRIMARY	EId				
2	ix_Indexer	nonclustered, unique located on PRIMARY	EId				
3	UNQ_Constraint	nonclustered, unique, unique key located on PRIMARY	EId				
	constraint_type	constraint_name	delete_action	update_action	status_enabled	status_for_replication	constraint_keys
1	UNIQUE (non-clustered)	UNQ_Constraint	(n/a)	(n/a)	(n/a)	(n/a)	EId

Question 5. What are RED Flags in SQL Server and what is there usage?

Answer –

There are some flags in execution plan which normally reduces the performance of the query. Some of them are given below.

1. High Percentage Operations
2. Table Scans, Index Scans, Clustered Index Scans
3. Spools
4. Parallelism operations
5. Warnings
6. Thick Arrows
7. Hash Joins
8. Bookmark Lookups
9. Sorting

For details please visit below-

- <http://www.sqlservercentral.com/articles/Performance+Tuning/sevenshowplanredflags/1425/>
- <http://www.slideshare.net/kkline84/ten-query-tuning-techniques-every-sql-server-programmer-should-know>

Question 6. What are the types of physical joins in SQL Server? What are the different join operators in SQL Server?

Answer –

There are three types of physical joins given below-

Nested Loop

- Used usually when one table is significantly small
- The larger table has an index which allows seeking it using the join key

Merge Join

- Both inputs are sorted on the join key
- An equality operator is used
- Excellent for very large tables

Hash Match

- If the SQL Server can't use any of the above mentioned joins then it will use Hash Match join.
- Uses a hash table and a dynamic hash match function to match rows

For details please visit below links-

Nested Loop Joins: <http://blogs.msdn.com/b/craigfr/archive/2006/07/19/671712.aspx>

Merge Joins: <http://blogs.msdn.com/b/craigfr/archive/2006/08/03/merge-join.aspx>

Hash Joins: <http://blogs.msdn.com/b/craigfr/archive/2006/08/10/687630.aspx>

Question 7. What is a Latch in SQL Server?

Answer-

Latches perform the task of thread synchronization. For example, if a thread is reading a page from disk and creating a memory structure to contain it, it will create one or more Latches to prevent corruption of these structures. Once the operation is complete, the Latches will be released and other threads will be able to access that page and memory structure again. For the most part, latches are transient, taking a few milliseconds to complete.

A latch can be defined as an object that ensures data integrity on other objects in SQL Server memory, particularly pages. They are a logical construct that ensures controlled access to a resource and isolationism when required for pages in use. In contrast to locks, latches are an internal SQL Server mechanism that isn't exposed outside the SQLOS. They come in many different types but can be split into roughly two classes - buffer latches, and non-buffer latches.

A latch is a lightweight synchronization object used by the Storage Engine to protect memory structures used internally by SQL Server. A latch is nothing more than a so-called Critical Section in multi-threaded programming – with some differences.

Question 8. Difference between Latch and Lock.

Answer-

Latches are internal to the SQL Server engine. They are used to provide memory consistency. Locks are used by SQL Server to provide logical transactional consistency.

For details please visit –

<http://www.sqlskills.com/blogs/paul/category/latches/>
<http://www.sqlskills.com/blogs/paul/category/locking/>

Question 9. Could you please provide SQL 2014 New Features in DB Engine?

Answer-

New features in SQL 2014 Db engine are given below -

1. In-Memory OLTP (In-Memory Optimization)
2. SQL Server Data Files in Windows Azure
3. Host a SQL Server Database in a Windows Azure Virtual Machine
4. Backup and Restore Enhancements
 - ☑SQL Server Backup to URL
 - ☑SQL Server Managed Backup to Windows Azure
 - ☑Encryption for Backups
5. New Design for Cardinality Estimation
6. Delayed Durability
7. Updateable Column Store Indexes
8. Incremental Statistics & Partition Enhancement
9. Buffer Pool Extension to Solid State Drives (SSDs).
10. Managing Locks in Online Index

11. Always On Improvements

12. Resource Governor Enhancements

For details please refer - <https://pawankmr.wordpress.com/2015/04/08/sql-server-2014-new-features/>

Question 10. In which scenarios we should not use CTE's.

Answer-

We should not use CTE's for large tables.

A CTE can be used:

- For recursion
- Substitute for a view when the general use of a view is not required; that is, you do not have to store the definition in metadata.
- Reference the resulting non large table multiple times in the same statement.

Question 11. What do you mean by Cardinality in SQL Server?

Answer-

Cardinality refers to the uniqueness of data values contained in a particular column (attribute) of a database table. The lower the cardinality, the more duplicated elements in a column

There are 3 types of cardinality:

High-cardinality, Normal-cardinality, and Low-cardinality

For details please refer - <http://msbiskills.com/2015/04/08/cardinality-in-sql/>

Question 12. What are Histogram and Density Vector?

Answer-

Density is the ratio of unique values with in the given column or a set of columns. Density measure the uniqueness of column or selectivity of column. Density can have value between 0 and 1. If the column has density value 1, it means all the records have same value in that column and less selectivity. Higher the density lowers the selectivity. If the column has density value 0.003, that means there are $1/0.003=333$ distinct values in that column.

Density = 1 / (Number of distinct values for a column or set of column)

```
DBCC SHOW_STATISTICS('TestRIDInxs', 'Ix_Index') WITH DENSITY_VECTOR
```

```
DBCC SHOW_STATISTICS('TestRIDInxs', 'Ix_Index') WITH DENSITY_VECTOR
```

	All density	Average Length	Columns
1	9.99999E-07	8	EId

DBCC SHOW_STATISTICS('TestRIDInxs', 'Ix_Index') WITH HISTOGRAM

Below information is self-explanatory.

```
DBCC SHOW_STATISTICS('TestRIDInxs', 'Ix_Index') WITH HISTOGRAM
```

	RANGE_HI_KEY	RANGE_ROWS	EQ_ROWS	DISTINCT_RANGE_ROWS	AVG_RANGE_ROWS
1	1	0	1	0	1
2	1000001	999999	1	999999	1

Question 13. Why we can't put Order by inside the view?

Answer –

There is no need of using ORDER BY in the View. Order by in a View is pointless.

If you try to include ORDER BY in View, it will throw the following error

Msg 1033, Level 15, State 1, Procedure vw_ViewLimit1, Line 5

The ORDER BY clause is invalid in views, inline functions, derived tables, subqueries, and common table expressions, unless TOP or FOR XML is also specified.

You can do below like-

```
SELECT * FROM [View_Name] ORDER By [Column_Name]
```

Question 14. What is a cross apply and how to use this?

Answer –

The APPLY operator comes in two variants, CROSS APPLY and OUTER APPLY. It is useful for joining two SQL tables or XML expressions. CROSS APPLY is equivalent to an INNER JOIN expression and OUTER APPLY is equivalent to a LEFT OUTER

JOIN expression. E.g. below-

```
SELECT f.*,  
ROW_NUMBER() OVER(ORDER BY file_id, page_id, slot_id) AS Row  
FROM flag f  
CROSS APPLY sys.fn_PhysLocCracker(%%physloc%%)  
ORDER BY Row
```

Question 15. Can we use more than one CTE in a single select query?

Answer – Yes we can. Please check out the example below.

```
;  
WITH CTE1 AS  
(  
    SELECT 10 Col1  
)  
CTE2 AS  
(  
    SELECT 20 Col2  
)  
SELECT a.Col1,b.Col2  
FROM CTE1 a  
FULL OUTER JOIN CTE2 b ON a.Col1 = b.Col2
```

Question 16. How do you know the total transaction count?

Answer – Please check out the query below-

```
SELECT * FROM sys.dm_os_performance_counters  
WHERE  
1=1  
AND counter_name = 'Transactions/sec'  
AND instance_name = '_Total';
```

Please use WAITFOR DELAY if you want to know how many transactions are happening on server during certain interval.

```

SELECT * FROM sys.dm_os_performance_counters
WHERE
1=1
AND counter_name = 'Transactions/sec'
AND instance_name = '_Total';

```

	object_name	counter_name	instance_name	cntr_value	cntr_type
1	SQLServer:Databases	Transactions/sec	_Total	4038749	272696576

Question 17. I have created a table variable can we use it in a nested stored procedure? If not what is the scope of a table variable?

Answer-

Scope of the Table variable is the Batch or Stored Procedure in which it is declared. And they can't be dropped explicitly, they are dropped automatically when batch execution completes or the Stored Procedure execution completes.

```

DECLARE @temp TABLE
(
    col1 INT
)

```

```

INSERT @temp VALUES(1)
SELECT * FROM @TEMP

```

GO

```

SELECT * FROM @temp

```

After executing above script you will get below error.

(1 row(s) affected)

(1 row(s) affected)

Msg 1087, Level 15, State 2, Line 11
Must declare the table variable "@temp".

Question 18. Top performance tuning tools

Answer-

Query Analyzer, SQL Profiler, Index Wizard, Performance Monitor are some Microsoft tools. Some other tools are given below-

- System Stored Procedures (Transact-SQL)

- Red-Gate SQL Monitor
- Trace Flags (Transact-SQL)
- Database Engine Tuning Advisor
- SQL Sentry Performance Advisor

Question 19. What is a bitmap index?

A bitmap index is a special type of structure used by most high-end database management systems to optimize search and retrieval for low-variability data.

Question 20. Can we fire a trigger manually?

Answer-

No you can't. If you want a procedure that can be executed manually, then you should just create a stored procedure.

Question 21. What are the magic tables? Do we have "Updated" magic table?

Answer-

No we don't have UPDATED magic table.

The 'magic tables' are the **INSERTED** and **DELETED** tables, as well as the **update()** and **columns_updated()** functions, and are used to determine the changes resulting from DML statements.

- For an INSERT statement, the INSERTED table will contain the inserted rows.
- For an UPDATE statement, the INSERTED table will contain the rows after an update, and the DELETED table will contain the rows before an update.
- For a DELETE statement, the DELETED table will contain the rows to be deleted.

Question 22. What is a filtered index?

Answer-

Consider filter index as --> INDEX with a Where Clause

It uses a **filter** predicate to **index** a portion of rows in the table. A **filtered index** can improve query performance as well as reduce **index** maintenance and storage costs compared with full-table **indexes**.

Important points about filtered indexes are -

- They can be created only as NonClustered Index
- They can be used on Views only if they are persisted views.
- They cannot be created on full-text Indexes.

Please check out the sample example below -

```
CREATE NONCLUSTERED INDEX Ix_NCI ON TestRIDInxs(ID)
WHERE Title= 20
```

Question 23. Basic difference between stored procedure and user defined function?

Answer-

There are many differences between functions and stored procedures. Some of them are given below-

Functions will allow only Select statement, it will not allow us to use insert, update and delete statement. Procedures can have select statements as well as DML statements such as insert, update, delete.

Transactions are not allowed within functions. Can use transactions within Stored procedures.

For details please visit –

<http://msbiskills.com/2010/08/12/differences-between-stored-procedure-and-function-in-sql-server/>

Question 24. See we have a simple query that's calling a static function, like "Select * from employee where joiningdate < getstaticdate()"? Does it call function for every time or only for matched rows? How you tune this query?

Answer-

We can do something like below-

```
DECLARE @t AS DATETIME = GETDATE()
SELECT * FROM TestRIDInxs WHERE joiningDate <= @t
```

Question 25. Why should we use CTE?

Answer-

A CTE can be used:

- For recursion
- Substitute for a view when the general use of a view is not required; that is, you do not have to store the definition in metadata.
- Reference the resulting non large table multiple times in the same statement.

Question 26. What is the difference between sub query and correlated query

Answer-

Subquery: - The inner query is executed only once. The inner query will get executed first and the output of the inner query used by the outer query. The inner query is not dependent on outer query.

Please check out the example below-

```
SELECT * FROM Visits WHERE CustomerId IN (SELECT CustomerId FROM Visits WHERE VisitDate = '2013-09-03 00:00:00.000')
```

Correlated subquery: - The outer query will get executed first and for every row of outer query, inner query will get executed. So the inner query will get executed as many times as number of rows in result of the outer query. The outer query output can use the inner query output for comparison. This means inner query and outer query dependent on each other

Please check out the famous example below- (2nd Highest Salary)

```
SELECT * FROM NthHighest N WHERE 1 = (SELECT DISTINCT(COUNT(*)) FROM NthHighest m WHERE n.Salary < m.Salary )
```

Question 27 - How do you retrieve random 5 rows from a table

Answer – `SELECT TOP 5 * FROM [Table_Name] ORDER BY NEWID()`

Question 28 - What exactly you check in the query execution plan window?

Answer –

Query Execution Plans describe the steps and the order used to access data in the database. Normally we check the execution plan for missing indexes, if any. We also check the execution plans if the query or stored procedure is working slowly. If working slowly we need to check for Red flags and see how we can remove them.

You can use execution plan to check out following things –

- Missing Indexes
- Missing Statistics
- Red Flags
- How data is being retrieved (Which path , what all indexes are used)

Question 29 - What is the output of `SELECT Len(1234.56)`

Answer – 7 (In this case it will consider all the characters including.)

Question 30 - Where in MS SQL Server is '100' equal to '0'?

Ans – Fill Factor is the answer. Fill-factor settings of 0 and 100 are equal. If fill-factor is set to 100 or 0, the SQL Server Engine fills pages to their capacity while creating indexes.
Server default Value = 0.

Question 31- I have a table with millions of rows. I want to retain only last 5% of the rows? How does u do it?

Answer –

1. Transfer last 5 % rows in a temp table
2. Delete all the rows from the table.
3. Move back data from temp table to the main table.

Question 32 - Explain transaction log in detail

Answer –

A transaction log stores every transaction made to a SQL Server database, except some which are minimally logged like BULK IMPORT or SELECT INTO.

Internally it is split into the smaller parts called Virtual Log Files (VLFs). When one VLF becomes full logging continue to write into the next available in the transaction log.

The transaction log file can be represented as a circular file. When the logging reaches the end of the file it starts again from the beginning, but only if all the requirements has been met and the inactive parts has been truncated. The truncation process is necessary to mark all inactive parts so they can be used again and overwritten.

A Log Sequence Number (LSN) identifies every transaction in the transaction log.

Question 33. Can we call a procedure from a function?

Answer – No.

Question 34. Can we write DML inside a function?

Answer – You can only use SELECT inside a function. You cannot use Insert, Update and Delete inside a function.

Question 35. How to avoid bookmark lookup in execution plan?

Answer-

Key lookup operation occurs when index seek is done on a non-clustered index to locate one or more rows, but the non-clustered index does not contain all the columns necessary for the query. The clustered index key (which is always included in all non-clustered indexes) is then used to locate the row in the clustered index, to retrieve the remaining data.

We can include all the necessary columns in the NonClustered key.

Question 36. What is XACT_ABORT ON?

When SET XACT_ABORT is ON, if a Transact-SQL statement raises a run-time error, the entire transaction is terminated and rolled back.

When SET XACT_ABORT is OFF, in some cases only the Transact-SQL statement that raised the error is rolled back and the transaction continues processing. Depending upon the severity of the error, the entire transaction may be rolled back even when SET XACT_ABORT is OFF. OFF is the default setting.

Compile errors, such as syntax errors, are not affected by SET XACT_ABORT.

XACT_ABORT must be set ON for data modification statements in an implicit or explicit transaction against most OLE DB providers, including SQL Server. The only case where this option is not required is if the provider supports nested transactions.

Sample e.g.

```
SET XACT_ABORT OFF;  
SET XACT_ABORT ON;
```

Question 37. Does temp tables and table variables both stored in tempDB?

Answer-

Temp tables and table variables are both objects created INSIDE TEMPDB. They are not created in memory. Their presence in memory is a result of their usage. Meaning: their pages are transferred and stored inside the buffer pool because this is the common way SQL Server works with all data pages – when a data page is requested for reads or writes it is read from disk and saved in the Buffer pool. So unless they are too big or server memory is under pressure they will be cached in memory anyway.

For details please visit - <http://magineumova.com/temp-tables-or-temp-variables/>

Question 38 - Explain Column Store Index

Answer-

It's a new feature introduced in SQL 2012.

The column store index stores data in columnar fashion and uses compression aggressively to reduce the disk I/O needed to serve the query request.

A ColumnStore index stores each column in a separate set of disk pages, rather than storing multiple rows per page as data traditionally has been stored.

We can only create non clustered ColumnStore index. Column store index has been designed to substantially accelerate common data warehouse queries, which require scanning, aggregation and filtering of large amounts of data or joining multiple tables like a star schema. With column store index, you can get interactive response time for queries against billions of rows on an economical SMP server with enough RAM to hold your frequently accessed data.

Creating a ColumnStore index is a parallel operation, subject to the limitations on the number of CPUs available and any restrictions set on MAXDOP setting.

Question 39. What happens when a rollback happens in inside a nested stored procedure?

Answer-

If the rollback is present in the outermost transaction then everything will be rolled back irrespective of what is written in the inner transactions.

If the rollback is present in the inner transaction then we will not have any transaction in the outer most transaction.

Also note that the output depends on the statements written.

For details please visit –

<http://msbiskills.com/2015/05/12/sql-server-nested-transactions-rollback-or-commit-confusions/>

Question 40. Why cursors are so costly?

Answer-

SQL Server like any good relational database management system (RDBMS) is optimized for set-based operations and NOT for row by row operation. Cursors will fetch data on a row by row basis, and it takes heck of a lot longer to do so. This is because the set-based logic for which RDBMS systems like SQL Server are optimized is completely broken and the entire query process has to be repeated for each row.

For details please visit –

<http://www.sqlshack.com/sql-server-cursor-performance-problems/>

Question 41. What is the best value for MAXDOP value?

Answer-

MAXDOP is Max degree of parallelism. This option controls the number of processors that are used for the execution of a query in a parallel plan.

One can use the max degree of parallelism option to limit the number of processors to use for parallel plan execution and to prevent run-away queries from impacting SQL Server performance by using all available CPUs.

The answer is: It depends. It depends on the hardware, the environment (OLTP vs. OLAP), and the load and so on.

The default value for MAXDOP is 0 (zero) and can be set or viewed using (**sp_configure**). A value of 0 means that SQL Server will use all processors if a query runs in parallel.

For details please visit –

<http://www.mssqltips.com/sqlservertip/2650/what-maxdop-setting-should-be-used-for-sql-server/>

Question 42. Which is better “Left Outer” Or “NOT EXIST”?

Answer-

In SQL Server, **NOT EXISTS** and **NOT IN** predicates are the best ways to search for missing values, as long as both columns in question are NOT NULL. They produce the safe efficient plans with some kind of an Anti-Join.

LEFT JOIN is less efficient, since it makes no attempt to skip the already matched values in the right table, returning all results and filtering them out instead.

<http://explainextended.com/2009/09/15/not-in-vs-not-exists-vs-left-join-is-null-sql-server/>

Question 43. How to find the statistics are outdated?

Answer –

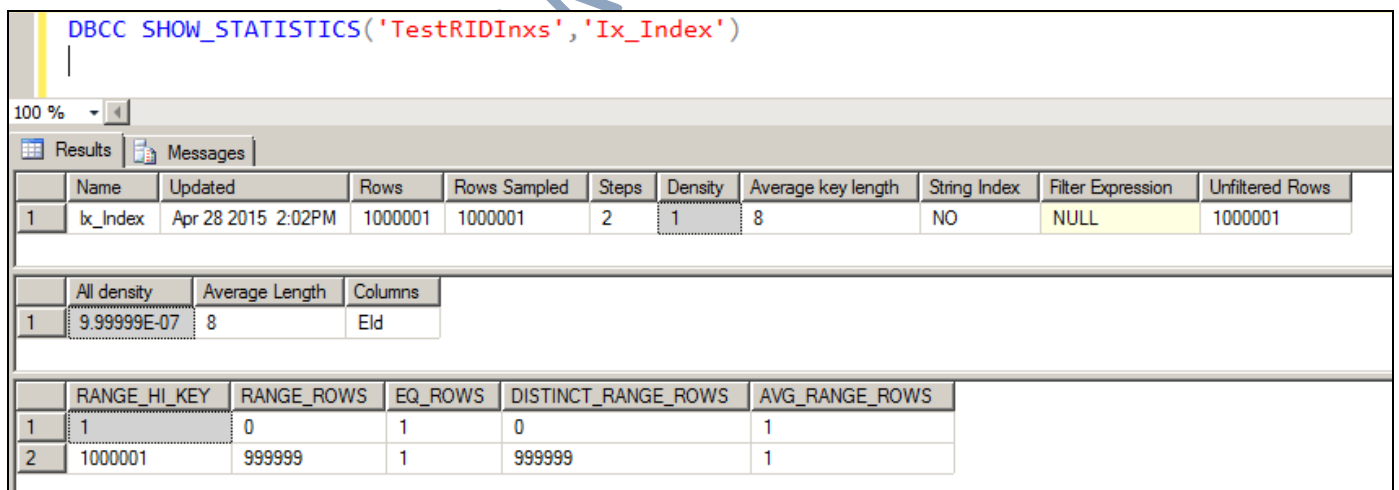
You can check statistics by using below command -

```
DBCC SHOW_STATISTICS('TestRIDInxs', 'Ix_Index')
```

Table Name - TestRIDInxs

Index Name - 'Ix_Index'

Output of the above query is given below-



The screenshot shows the SQL Server Enterprise Manager interface. At the top, the command `DBCC SHOW_STATISTICS('TestRIDInxs', 'Ix_Index')` is entered. Below the command, the 'Results' pane displays three tables of statistics data.

Name	Updated	Rows	Rows Sampled	Steps	Density	Average key length	String Index	Filter Expression	Unfiltered Rows
Ix_Index	Apr 28 2015 2:02PM	1000001	1000001	2	1	8	NO	NULL	1000001

All density	Average Length	Columns
9.99999E-07	8	Eld

RANGE_HI_KEY	RANGE_ROWS	EQ_ROWS	DISTINCT_RANGE_ROWS	AVG_RANGE_ROWS
1	0	1	0	1
1000001	999999	1	999999	1

Question 44. How to find the query running on a given SPID?

Answer-

You can use below query to find query running on an active SPID. Number (52) is the SPID showing in the below

example.

```
DECLARE @sqltext VARBINARY(128)
SELECT @sqltext = sql_handle
FROM sys.dm_exec_requests
WHERE session_id = 52
SELECT TEXT
FROM sys.dm_exec_sql_text(@sqltext);
```

Question 45. Is “TRUNCATE” DDL or DML command?

Answer- DDL

Question 46. I have written a recursive CTE, it's been running infinitely and failed with an error. The requirement is that the recursion required for 1000 times. How could you be able to handle the situation?

Answer-

Well if you write a proper recursive CTE then it will not run indefinitely.

MAXRECURSION can be used to break the recursive CTE. MAXRECURSION can be used to prevent a poorly formed recursive CTE from entering into an infinite loop.

The following example intentionally creates an infinite loop and uses the MAXRECURSION hint to limit the number of recursion levels to ten.

```
WITH CTE AS
(
    SELECT ParentID FROM Hierarchies WHERE ParentID = 1
    UNION ALL
    SELECT C.ParentID FROM Hierarchies H INNER JOIN CTE C
    ON C.ParentID = H.ParentID
)
SELECT * FROM CTE
OPTION (MAXRECURSION 10);
GO
```

You will also get this kind of message-

```
Msg 530, Level 16, State 1, Line 1
The statement terminated. The maximum recursion 10 has been exhausted before statement completion.
```

Question 47. Which is better a CTE or a subquery? Why?

Answer-

Both CTEs and Sub Queries have pretty much the same performance and function.

CTE's have an advantage over using a subquery in that you can use recursion in a CTE.

The biggest advantage of using CTE is readability. CTEs can be reference multiple times in the same statement where as sub query cannot.

Question 48. Any alternative to triggers?

Answer -

There are some options available depending exclusively on the requirement. They are

Change Data Capture

Change Tracking

Using output clause (Write the data out to a specific Audit table)

Computed column

Question 49. Why resourceDB introduced?

Answer -

It is a read-only database that contains system objects that are included with SQL Server. SQL Server system objects, such as sys.objects, are physically persisted in the Resource database, but they logically appear in the sys schema of every database. The Resource database does not contain user data or user metadata.

Also Resource database makes upgrading to a new version of SQL Server an easier and faster procedure. In earlier versions of SQL Server, upgrading required dropping and creating system objects. Because the Resource database file contains all system objects, an upgrade is now accomplished simply by copying the single Resource database file to the local server.

For details please visit –

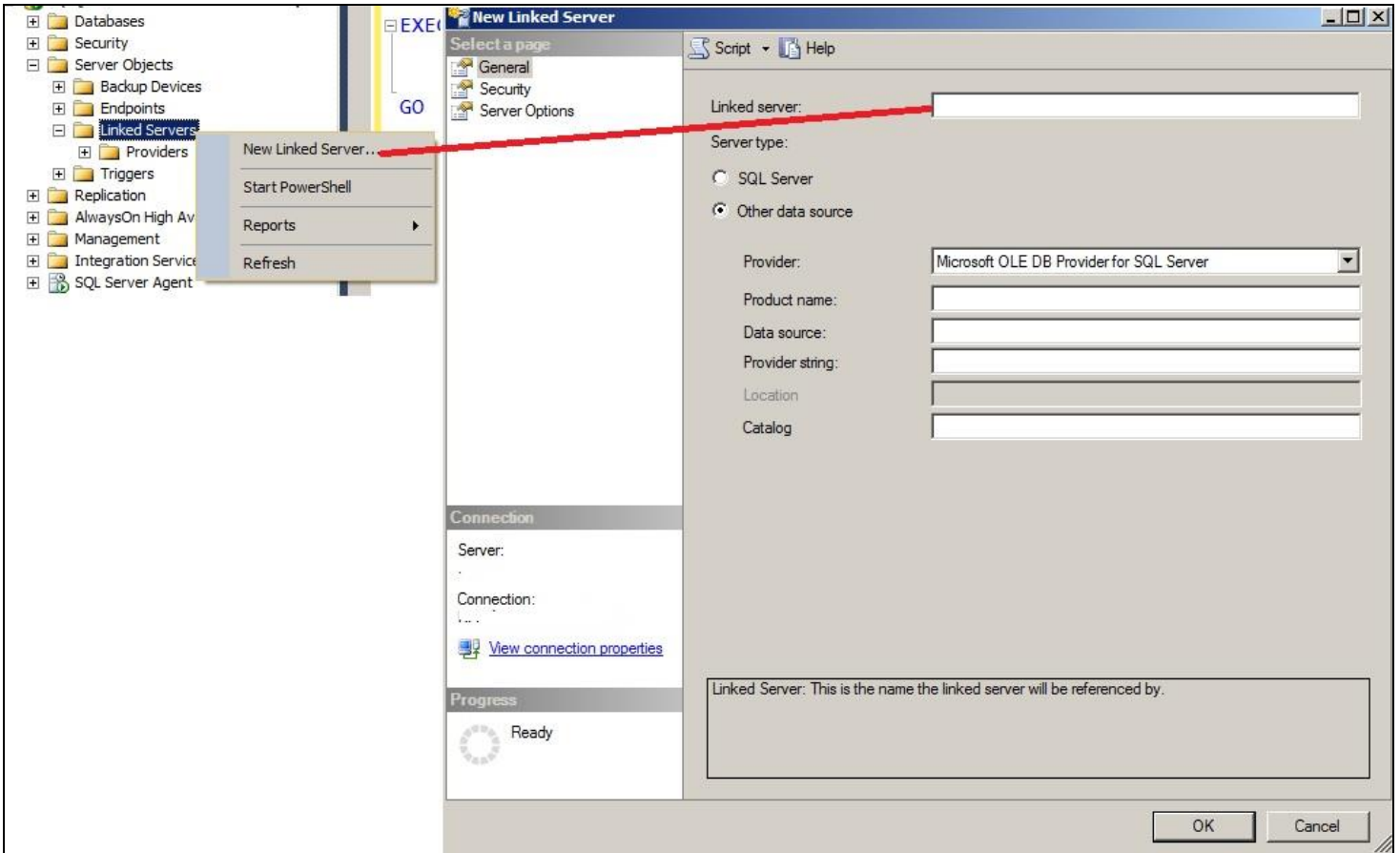
<http://blogs.msdn.com/b/vsanil/archive/2012/11/02/resource-database-common-questions.aspx>

Question 50. How to create linked server.

Answer -

Please use below T-SQL script or follow the screen shots

```
USE [master]
GO
EXEC MASTER.dbo.sp_addlinkedserver
    @server = N'SRVR002\ACCTG',
    @srvproduct=N'SQL Server' ;
GO
```



Pawan Kumar Khosla