**1. What is DBMS?**

A Database Management System (DBMS) is a program that controls creation, maintenance and use of a database. DBMS can be termed as File Manager that manages data in a database rather than saving it in file systems.

**2. What is RDBMS?**

RDBMS stands for Relational Database Management System. RDBMS store the data into the collection of tables, which is related by common fields between the columns of the table. It also provides relational operators to manipulate the data stored into the tables.

**Example: SQL Server.**

**3. What is SQL?**

SQL stands for Structured Query Language , and it is used to communicate with the Database. This is a standard language used to perform tasks such as retrieval, updation, insertion and deletion of data from a database.

**4. What is a Database?**

Database is nothing but an organized form of data for easy access, storing, retrieval and managing of data. This is also known as structured form of data which can be accessed in many ways.

Example: School Management Database, Bank Management Database.

**5. What are tables and Fields?**

A table is a set of data that are organized in a model with Columns and Rows. Columns can be categorized as vertical, and Rows are horizontal. A table has specified number of column called fields but can have any number of rows which is called record.

Example:.

Table: Employee.

Field: Emp ID, Emp Name, Date of Birth.

Data: 201456, David, 11/15/1960.

**What is a primary key?**

A primary key is a combination of fields which uniquely specify a row. This is a special kind of unique key, and it has implicit NOT NULL constraint. It means, Primary key values cannot be NULL.

**7. What is a unique key?**

A Unique key constraint uniquely identified each record in the database. This provides uniqueness for the column or set of columns.

A Primary key constraint has automatic unique constraint defined on it. But not, in the case of Unique Key.

There can be many unique constraint defined per table, but only one Primary key constraint defined per table.

**8. What is a foreign key?**

A foreign key is one table which can be related to the primary key of another table. Relationship needs to be created between two tables by referencing foreign key with the primary key of another table.

**9. What is a join?**

This is a keyword used to query data from more tables based on the relationship between the fields of the tables. Keys play a major role when JOINs are used.

**10. What are the types of join and explain each?**

There are various types of join which can be used to retrieve data and it depends on the relationship between tables.

**Inner join.**

Inner join return rows when there is at least one match of rows between the tables.

**Right Join.**

Right join return rows which are common between the tables and all rows of Right hand side table. Simply, it returns all the rows from the right hand side table even though there are no matches in the left hand side table.

**Left Join.**

Left join return rows which are common between the tables and all rows of Left hand side table. Simply, it returns all the rows from Left hand side table even though there are no matches in the Right hand side table.

**Full Join.**

Full join return rows when there are matching rows in any one of the tables. This means, it returns all the rows from the left hand side table and all the rows from the right hand side table.

**11. What is normalization?**

Normalization is the process of minimizing redundancy and dependency by organizing fields and table of a database. The main aim of Normalization is to add, delete or modify field that can be made in a single table.

**12. What is Denormalization.** DeNormalization is a technique used to access the data from higher to lower normal forms of database. It is also process of introducing redundancy into a table by incorporating data from the related tables.

**13. What are all the different normalizations?**

The normal forms can be divided into 5 forms, and they are explained below -.

**First Normal Form (1NF):.**

This should remove all the duplicate columns from the table. Creation of tables for the related data and identification of unique columns.

**Second Normal Form (2NF):.**

Meeting all requirements of the first normal form. Placing the subsets of data in separate tables and Creation of relationships between the tables using primary keys.

**Third Normal Form (3NF):.**

This should meet all requirements of 2NF. Removing the columns which are not dependent on primary key constraints.

**Fourth Normal Form (3NF):.**

Meeting all the requirements of third normal form and it should not have multi- valued dependencies.

**14. What is a View?**

A view is a virtual table which consists of a subset of data contained in a table. Views are not virtually present, and it takes less space to store. View can have data of one or more tables combined, and it is depending on the relationship.

**15. What is an Index?**

An index is performance tuning method of allowing faster retrieval of records from the table. An index creates an entry for each value and it will be faster to retrieve data.

**16. What are all the different types of indexes?**

There are three types of indexes -.

**Unique Index.**

This indexing does not allow the field to have duplicate values if the column is unique indexed. Unique index can be applied automatically when primary key is defined.

**Clustered Index.**

This type of index reorders the physical order of the table and search based on the key values. Each table can have only one clustered index.

**NonClustered Index.**

NonClustered Index does not alter the physical order of the table and maintains logical order of data. Each table can have 999 nonclustered indexes.

**17. What is a Cursor?**

A database Cursor is a control which enables traversal over the rows or records in the table. This can be viewed as a pointer to one row in a set of rows. Cursor is very much useful for traversing such as retrieval, addition and removal of database records.

**18. What is a relationship and what are they?**

Database Relationship is defined as the connection between the tables in a database. There are various data basing relationships, and they are as follows:.

* One to One Relationship.
* One to Many Relationship.
* Many to One Relationship.
* Self-Referencing Relationship.

**19. What is a query?**

A DB query is a code written in order to get the information back from the database. Query can be designed in such a way that it matched with our expectation of the result set. Simply, a question to the Database.

**20. What is subquery?**

A subquery is a query within another query. The outer query is called as main query, and inner query is called subquery. SubQuery is always executed first, and the result of subquery is passed on to the main query.

**21. What are the types of subquery?**

There are two types of subquery – Correlated and Non-Correlated.

A correlated subquery cannot be considered as independent query, but it can refer the column in a table listed in the FROM the list of the main query.

A Non-Correlated sub query can be considered as independent query and the output of subquery are substituted in the main query.

**22. What is a stored procedure?**

Stored Procedure is a function consists of many SQL statement to access the database system. Several SQL statements are consolidated into a stored procedure and execute them whenever and wherever required.

**What is a trigger?**

A DB trigger is a code or programs that automatically execute with response to some event on a table or view in a database. Mainly, trigger helps to maintain the integrity of the database.

Example: When a new student is added to the student database, new records should be created in the related tables like Exam, Score and Attendance tables.

**24. What is the difference between DELETE and TRUNCATE commands?**

DELETE command is used to remove rows from the table, and WHERE clause can be used for conditional set of parameters. Commit and Rollback can be performed after delete statement.

TRUNCATE removes all rows from the table. Truncate operation cannot be rolled back.

**25. What are local and global variables and their differences?**

Local variables are the variables which can be used or exist inside the function. They are not known to the other functions and those variables cannot be referred or used. Variables can be created whenever that function is called.

Global variables are the variables which can be used or exist throughout the program. Same variable declared in global cannot be used in functions. Global variables cannot be created whenever that function is called.

**26. What is a constraint?**

Constraint can be used to specify the limit on the data type of table. Constraint can be specified while creating or altering the table statement. Sample of constraint are.

* NOT NULL.
* CHECK.
* DEFAULT.
* UNIQUE.
* PRIMARY KEY.
* FOREIGN KEY.

**27. What is data Integrity?**

Data Integrity defines the accuracy and consistency of data stored in a database. It can also define integrity constraints to enforce business rules on the data when it is entered into the application or database.

**28. What is Auto Increment?**

Auto increment keyword allows the user to create a unique number to be generated when a new record is inserted into the table. AUTO INCREMENT keyword can be used in Oracle and IDENTITY keyword can be used in SQL SERVER.

Mostly this keyword can be used whenever PRIMARY KEY is used.

**29. What is the difference between Cluster and Non-Cluster Index?**

Clustered index is used for easy retrieval of data from the database by altering the way that the records are stored. Database sorts out rows by the column which is set to be clustered index.

A nonclustered index does not alter the way it was stored but creates a complete separate object within the table. It point back to the original table rows after searching.

**30. What is Datawarehouse?**

Datawarehouse is a central repository of data from multiple sources of information. Those data are consolidated, transformed and made available for the mining and online processing. Warehouse data have a subset of data called Data Marts.

**31. What is Self-Join?**

Self-join is set to be query used to compare to itself. This is used to compare values in a column with other values in the same column in the same table. ALIAS ES can be used for the same table comparison.

**32. What is Cross-Join?**

Cross join defines as Cartesian product where number of rows in the first table multiplied by number of rows in the second table. If suppose, WHERE clause is used in cross join then the query will work like an INNER JOIN.

**33. What is user defined functions?**

User defined functions are the functions written to use that logic whenever required. It is not necessary to write the same logic several times. Instead, function can be called or executed whenever needed.

**34. What are all types of user defined functions?**

Three types of user defined functions are.

* Scalar Functions.
* Inline Table valued functions.
* Multi statement valued functions.

Scalar returns unit, variant defined the return clause. Other two types return table as a return.

**35. What is collation?**

Collation is defined as set of rules that determine how character data can be sorted and compared. This can be used to compare A and, other language characters and also depends on the width of the characters.

ASCII value can be used to compare these character data.

**36. What are all different types of collation sensitivity?**

Following are different types of collation sensitivity -.

* Case Sensitivity – A and a and B and b.
* Accent Sensitivity.
* Kana Sensitivity – Japanese Kana characters.
* Width Sensitivity – Single byte character and double byte character.

**37. Advantages and Disadvantages of Stored Procedure?**

Stored procedure can be used as a modular programming – means create once, store and call for several times whenever required. This supports faster execution instead of executing multiple queries. This reduces network traffic and provides better security to the data.

Disadvantage is that it can be executed only in the Database and utilizes more memory in the database server.

**38. What is Online Transaction Processing (OLTP)?**

Online Transaction Processing or OLTP manages transaction based applications which can be used for data entry and easy retrieval processing of data. This processing makes like easier on simplicity and efficiency. It is faster, more accurate results and expenses with respect to OTLP.

Example – Bank Transactions on a daily basis.

**39. What is CLAUSE?**

SQL clause is defined to limit the result set by providing condition to the query. This usually filters some rows from the whole set of records.

Example – Query that has WHERE condition

Query that has HAVING condition.

**40. What is recursive stored procedure?**

A stored procedure which calls by itself until it reaches some boundary condition. This recursive function or procedure helps programmers to use the same set of code any number of times.

**41. What is Union, minus and Interact commands?**

UNION operator is used to combine the results of two tables, and it eliminates duplicate rows from the tables.

MINUS operator is used to return rows from the first query but not from the second query. Matching records of first and second query and other rows from the first query will be displayed as a result set.

INTERSECT operator is used to return rows returned by both the queries.

**42. What is an ALIAS command?**

ALIAS name can be given to a table or column. This alias name can be referred in WHERE clause to identify the table or column.

Example-.



|  |  |
| --- | --- |
| 1 | Select st.StudentID, Ex.Result from student st, Exam as Ex where st.studentID = Ex. StudentID |

Here, st refers to alias name for student table and Ex refers to alias name for exam table.

**43. What is the difference between TRUNCATE and DROP statements?**

TRUNCATE removes all the rows from the table, and it cannot be rolled back. DROP command removes a table from the database and operation cannot be rolled back.

**44. What are aggregate and scalar functions?**

Aggregate functions are used to evaluate mathematical calculation and return single values. This can be calculated from the columns in a table. Scalar functions return a single value based on the input value.

Example -.

Aggregate – max(), count – Calculated with respect to numeric.

Scalar – UCASE(), NOW() – Calculated with respect to strings.

**45. How can you create an empty table from an existing table?**

Example will be -.



|  |  |
| --- | --- |
| 1 | Select \* into studentcopy from student where 1=2 |

Here, we are copying student table to another table with the same structure with no rows copied.

**46. How to fetch common records from two tables?**

Common records result set can be achieved by -.



|  |  |
| --- | --- |
| 1 | Select studentID from student. <strong>INTERSECT </strong> Select StudentID from Exam |

**47. How to fetch alternate records from a table?**

Records can be fetched for both Odd and Even row numbers -.

To display even numbers-.



|  |  |
| --- | --- |
| 1 | Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=0 |

To display odd numbers-.



|  |  |
| --- | --- |
| 1 | Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=1 |

from (Select rowno, studentId from student) where mod(rowno,2)=1.[/sql]

**48. How to select unique records from a table?**

Select unique records from a table by using DISTINCT keyword.



|  |  |
| --- | --- |
| 1 | Select DISTINCT StudentID, StudentName from Student. |

**49. What is the command used to fetch first 5 characters of the string?**

There are many ways to fetch first 5 characters of the string -.



|  |  |
| --- | --- |
| 1 | Select SUBSTRING(StudentName,1,5) as studentname from student |



|  |  |
| --- | --- |
| 1 | Select RIGHT(Studentname,5) as studentname from student |

**50. Which operator is used in query for pattern matching?**

LIKE operator is used for pattern matching, and it can be used as -.

1. % – Matches zero or more characters.
2. \_(Underscore) – Matching exactly one character.

**Which TCP/ IP port does the SQL Server run on? How can it be changed?**

SQL Server runs on port 1433. It can be changed from the SQL Server Configuration Manager -> SQL Server Network Configurations -> Protocols for SQL Server -> TCP/ IP properties –> IP Addresses -> TCP Port number, both on the client and the server.

**What are the differences between clustered index and non-clustered index ?**

A clustered index is a special type of index that reorders the way records in the table are physically stored. Therefore, any given table can have only one clustered index. The leaf nodes of a clustered index contain the actual data.

A non-clustered index is a special type of index in which the logical order of the index does not match the physical stored order of the rows on disk. The leaf node of a non-clustered index does not consist of the data pages. Instead, the leaf nodes contain index rows and a pointer to the data (clustered index key or RID).

**When is the use of the UPDATE\_STATISTICS command appropriate?**

This command is basically used when a large amount of data is processed. If a large amount of deletions, modifications or bulk copies into the tables has occurred, it has to update the indices to take these changes into account. UPDATE\_STATISTICS updates the indices on these tables accordingly.

**What is SQL Profiler?**

SQL Profiler is a graphical tool that allows system administrators to monitor events in an instance of Microsoft SQL Server. You can capture and save data about each event to a file or SQL Server table to analyze later. For example, you can monitor SQL Server in a production environment to see which stored procedures are executing very slowly and hampering performance. Use SQL Profiler to monitor only the events in which you are interested. If traces are becoming too large, you can filter them based on the information you want, so that only a subset of the event data is collected. Monitoring too many events adds overhead to the server and the monitoring process. This can cause the trace file or trace table to grow very large, especially when the monitoring process takes place over a long period of time. In a production environment, enable server side trace to reduce the impact of collecting trace data.

**What is SQL Server agent?**

The SQL Server agent plays an important role in the day-to-day tasks of a database administrator (DBA). It is often overlooked as one of the main tools for SQL Server management. Its purpose is to ease the implementation of tasks for the DBA, with its full-function scheduling engine, which allows you to schedule your own jobs and scripts.

**What is BCP and when is it used?**

BCP or “Bulk Copy Program” is a tool used to copy huge amounts of data from tables and views. BCP does not copy the complete structures from source to destination. The BULK INSERT command helps to import a data file into a database table or view in a user-specified format.

**What are the authentication modes in SQL Server? How can it be changed?**

There are two authentication modes in SQL Server.

Windows mode • Mixed mode – SQL and Windows

To change authentication mode in SQL Server, go to Start -> Programs- > Microsoft SQL Server and click SQL Server Management Studio and under Object Explorer, right-click the server, and then click Properties. On the Security page, under Server authentication, select the new server authentication mode, and then click OK.

**Can you configure SQL Server without the SA account?**

The SA account is a well-known guessable SQL Server account and is often targeted by malicious users. Do not enable the SA account unless your application requires it. With SQL Server 2008 R2 onwards, you can also rename the SA account. If Windows Authentication mode is selected during installation, the SA login is disabled and a password is assigned by setup. If you later change authentication mode to SQL Server and Windows Authentication mode, the SA login remains disabled.

**What is log shipping?**

Log shipping is the process of automating the backup of database and transaction log files on a production SQL server with the goal of then restoring them onto a standby server. In log shipping, the log file from one server is automatically updated to the backup database on the backup server. If one server fails, the other server will have the same transactions run and therefore the other db and can be used as a disaster recovery plan. The key feature of log shipping is that it will automatically backup transaction logs throughout the day and automatically restore them on the standby server at defined intervals.

**Name a few ways to get an accurate count of the number of records in a table?**

Using tables and catalog views:

**What does it mean to have the QUOTED\_IDENTIFIER ON? What are the implications of having it OFF?**

When SET QUOTED\_IDENTIFIER is ON, identifiers can be delimited by double quotation marks, and literals must be delimited by single quotation marks. When SET QUOTED\_IDENTIFIER is OFF, identifiers cannot be quoted and must follow all T-SQL rules for identifiers.

**How do you rebuild the master database?**

The master database is a system database that contains information about the server’s running configuration. When SQL Server 2005 is installed, it usually creates master, model, msdb, tempdb, resourcedb, and the distribution system database by default. Without the master database, SQL Server cannot be started. This is the reason why it is extremely important to back up the master database. To rebuild the master database, run setup.exe, verify, and repair the SQL Server instance, and rebuild the system databases. This procedure is most often used to rebuild the master database for a corrupted installation of SQL Server.

**What are standby servers? Explain the types of standby servers**

. A standby server is a type of server that can be brought online in a situation when the primary server goes offline. This is useful for when an application needs continuous (high) availability from the server. There is always a need to set up a mechanism where data and objects from the primary server are moved to a secondary (standby) server. This mechanism usually involves the process of moving a backup from the primary server to the secondary server using T-SQL scripts. Often, database wizards are used to set up this process. The different types of standby servers are given as follows:

1. **Hot standby**:

A hot standby can be achieved in SQL Server using SQL Server 2005 Enterprise Edition or later versions of SQL Server Enterprise. SQL Server 2005 introduced database mirroring to be able to configure automatic failover in a disaster recovery scenario. When using of synchronous mirroring, the database is replicated to both servers simultaneously. This is a little more expensive but provides the highestavailability. In this case, both primary and standby servers have identical data all times.

1. **Warm standby**

In warm standby, automatic failover is not configured. This is usually set up using log shipping or asynchronous mirroring. Sometimes a warm standby will lag by a few minutes or seconds, which results in the loss of a few of the latest updates when the primary server fails and the secondary server needs to come online. Sometimes a warm standby server that is lagging by a few transactions is brought back to the current state by applying the most recent transaction log.

1. **Cold standby**:

Cold standby servers need to be switched manually, and sometimes all the backups as well as the required OS needs to be applied. A cold standby just physically replaces the previous server.

**What is the difference between GRANT and WITH GRANT while giving permissions to the user?**

When using GRANT, the principal cannot grant the same permission to other users. On the other hand, when using WITH GRANT, the principal will be able to give the same permission to other users.

A table-level DENY does not take precedence over a column-level GRANT.

**How do you copy tables, schemas, and views from one SQL Server to another?**

There are multiple ways to do this: 1) “Detach database” from one server and “attach database” to another server. 2) Manually script all of the objects using SSMS and run the script on a new server. 3) Use the Move Database Wizard of SSMS.

**Where are SQL Server usernames and passwords stored in SQL Server?**

The system catalog views of sys.server\_principals and sys.sql\_logins can show the various users in the system. These are stored in the master database.

**What is SQLCMD?**

SQLCMD is an enhanced version of isql and osql, and it provides more functionality than the other two options. In other words, SQLCMD is a better replacement of isql (which will be deprecated eventually) and osql (not included in SQL Server 2005 RTM). For a lesson on SQLCMD see Chapter 6 of Beginning SQL Joes 2 Pros Volume 1 ISBN: 1-4392-5317-X.

SQLCMD can work in two modes - i) BATCH and ii) interactive modes. (Read more here http:// bit.ly/ sqlinterview108)

**What is Utility Control Point (UCP)?**

The SQL Server utility models an organization’s SQL Server-related entities in a unified view. Utility Explorer and SQL Server Utility viewpoints in (SSMS) provide administrators a holistic view of resource health through an instance of SQL Server that serves as a Utility Control Point (UCP). The UCP collects configuration and performance information from managed instances of SQL Server every 15 minutes by default. To remove an instance of SQL Server from Utility Control Point, make sure that the SQL Server and SQL Server Agent services are running on the instance to remove.

**What can be monitored via UCP?**

Entities that can be viewed in SQL Server UCP include: • Instances of SQL Server. • Data-tier applications. • Database files. • Storage volumes. Resource utilization dimensions that can be viewed in SQL Server UCP include: • CPU utilization. • Storage space utilization. Some of the current restrictions include: • The Instance has to be a SQL Server relational engine. • It must be a SQL 2008 R2 or a SQL 2008 SP2 instance. • UCP cannot be done for Express editions. • It must operate within a single Windows domain, or across domains with two-way trust relationships.

**How is SQLCMD different from OSQL?**

The SQLCMD utility is a command prompt utility to run adhoc T-SQL statements and scripts. You can also run SQLCMD interactively. SQLCMD is the new choice for scripting with SQLServer. There are interesting extended options which make SQLCMD worth using, like using the –A for DAC Dedicated Administrative Connections. SSMS uses the Microsoft .NET Framework SqlClient for execution in regular and SQLCMD mode in Query Editor. When SQLCMD is run from the command line, SQLCMD uses the OLE DB provider.

**What is the data collector?**

SQL Server 2008 introduced the concept of the data collector. You can obtain and save data that is gathered from several sources about the health of your SQL Server. The data collector provides data collection containers that you can use to determine the scope and frequency of data collection on a SQL Server system. The data collector provides predefined collector types that you can use for data collection. The out-of-box collector types are: • Generic T-SQL query collector type • Generic SQL trace collector type • Performance counters collector type • Query activity collector type

**What system data collection sets are predefined inside SQL Server?**

During the installation, there are 3 system data collections that are made available to DBAs. These are to be later configured to monitor SQL Server. These cannot be deleted: • Disk usage: Collects data about disk and log usage for all the databases installed on the system. • Server activity: Collects resource usage statistics and performance data from the server and SQL Server. • Query statistics: Collects query statistics, individual query text, query plans, and specific queries. The data collector uses the role-based security model implemented by SQL Server Agent.

**When will you use the SQLDiag tool?**

The SQLdiag utility is a general purpose diagnostics collection utility that can be run as a console application or as a service and can be very useful for performance tuning exercises. SQLdiag is fully configurable through the SQLdiag.xml configuration file and can collect a variety of diagnostic information like Windows performance logs, Windows event logs, SQL Server

column to a trace. • sp\_trace\_setfilter - Applies a filter to a trace. • sp\_trace\_setstatus - Starts, stops, and close traces. • sp\_trace\_generateevent - Creates a user-defined event.

**What are the events of a default trace?**

The default SQL Server trace from SQL Server 2005 onwards is a background trace that runs continuously and records event information that can be useful in troubleshooting problems. Though the list is long, the following are captured by the default trace: • Data file auto grow • Data file auto shrink • Database mirroring status change • Log file auto grow • Log file auto shrink • Error log • Missing column statistics • Missing join predicate • Object altered • Object created • Object deleted • Server memory change Apart from these there are many other SQL Server security auditing events also captured like: Add DB user event, DBCC event, login failed, backup/ restore event, server starts and stops, and many more.

**What is central management inside SQL Server?**

SQL Server 2008 introduced a new method of administering multiple servers by enabling you to designate a central management server. An instance of SQL Server that is designated as a central management server maintains a list of registered servers. A typical scenario for a DBA is to write a single query across multiple servers to determine the version of all the servers in a single result set using a single query. Transact-SQL statements and policy-based management policies can be executed at the same time against server groups.

**What tools are available for extended events?**

For the SQL Server engine, XEvent is configured using a series of T-SQL statements. There is no graphical tool support for XEvent in the current version of SQL Server 2008 R2. There are no tools provided allowing you to view the results of XEvent targets except

those provided by Windows to view ETW data (ETW = Event Tracing for Windows). Note: Event tracing for Windows is the standard way to trace all features used by Windows. **How do you disable an index?**

ALTERINDEX [IndexName] ON TableName DISABLE GO

**How do you enable an index**?

ALTERINDEX [IndexName] ON TableName REBUILD GO (Read more here http:// bit.ly/ sqlinterview109)

**What is data compression?**

In SQL SERVER 2008, data compression comes in three major flavors: • Row compression • Page compression • Dictionary compression

**Row compression**

Row compression changes the format of the physical storage of data. It minimizes the metadata (column information, length, offsets etc.) associated with each record. Numeric data types and fixed-length strings are stored in variable-length storage format, just like Varchar. (Read more here http:// bit.ly/ sqlinterview110) NVarchar( max) data is never compressed even if it is stored in the row.

**Page Compression**

Page compression allows common data to be shared between rows for a given page. It uses the following techniques to compress data: • Row compression. • Prefix compression. For every column in a page, duplicate prefixes are identified. These prefixes are saved in compression information headers which reside in the page header. A reference number is assigned to these prefixes and that reference number is replaced wherever those prefixes are being used.

**Dictionary compression**

Dictionary compression searches for duplicate values throughout the page and stores them in a Clustered Index. The main difference between prefix and dictionary compression is that the

former is only restricted to one column while the latter is applicable to the complete page. SQL Server 2008 R2 brings support for compression on Unicode data.

**What are wait types?**

There are three types of Wait Types, namely:

**Resource waits**. Resource waits occur when a worker requests access to a resource that is not available because that resource is either currently used by another worker thread or it’s not yet available.

**Queue waits** - Queue waits occur when a worker thread is idle, waiting for work to be assigned. **External waits** - External waits occur when a SQL Server worker thread is waiting for an external event. (Read more here http:// bit.ly/ sqlinterview111). For a lesson on Advanced Wait Stats Concepts see book Joes 2 Pros: SQL Performance Tuning Techniques Using Wait Statistics, Types & Queues ISBN: 1-4662-3477-6.

**What is FILLFACTOR**?

The FILLFACTOR setting was introduced in SQL Server 2000 and helps prevent the need for pages to split. We can tell SQL Server not to fill up every section of every page on the first sweep of data. We can instruct it to leave some empty space for later inserts, so we don’t have to move around the other existing pieces of data. For a lesson on FILLFACTOR see Chapter 8 of SQL Architecture Basics Joes 2 Pros Volume 3 ISBN: 1451579462.

A FILLFACTOR is one of the important arguments that can be used while creating an index. According to MSDN, FILLFACTOR specifies a percentage that indicates how much the database engine should fill each index page during index creation or rebuild. The FILLFACTOR is always an integer valued from 1 to 100. The FILLFACTOR option is designed for improving index performance and data storage. By setting the fillfactor value, you specify the percentage of space on each page to be filled with data, reserving free space on each page for future table growth.

Specifying a fillfactor value of 70 would imply that 30 percent of each page will be left empty, providing space for index expansion as data is added to the underlying table. The empty space is reserved between the indexed rows rather than at the end of the index. The FILLFACTOR setting applies only when the index is created or rebuilt. (Read more here http:// bit.ly/ sqlinterview112

**) What are points to remember while using the FILLFACTOR argument?** If the FILLFACTOR is set to 100 or 0, the database engine fills pages to their capacity while creating indexes. The server-wide default FILLFACTOR is set to 0. To modify the server-wide default value, use the sp\_configure system stored procedure. To view the FILLFACTOR value of one or more indexes, use sys.indexes. To modify or set the FILLFACTOR value for individual indexes, use CREATE INDEX or ALTER INDEX statements. Creating a clustered index with a FILLFACTOR < 100 may significantly increase the amount of space the data occupies because the database engine physically reallocates the data while building the clustered index. (Read more here http:// bit.ly/ sqlinterview113)

**What is the comparison of SQL Server ’100’ and ‘0’ FILLFACTOR values?**

FILLFACTOR settings of 0 and 100 are equal! (Read more here http:// bit.ly/ sqlinterview114) **What is PAD\_INDEX?**

PAD\_INDEX is the percentage of free space applied to the intermediate-level pages of the index as specified by the FILLFACTOR. The PAD\_INDEX option is useful only when the FILLFACTOR is specified.

**What is the difference between a view and a materialized view?**

A view takes the output of a query and makes it appear like a virtual table. A view can be used in place of tables. A materialized view provides indirect access to table data by storing the results of a query in a separate schema object. To create a materialized view you need to put an index on the view.

**What is the concept ‘optimize for ad hoc workloads’ option?**

In SQL Server 2008, the “optimize for ad hoc workloads” option is a new server configuration option used to improve the efficiency of the plan cache for workloads that contain many single use ad hoc batches. This option is greatly useful for third party applications that the DBA might not have control over – ERP, CRM systems are typical here. When this option is set to 1, the database engine stores a small compiled plan stub instead of the full compiled plan. This plan stub is created when a batch is compiled for the first time. This helps to relieve memory pressure by not allowing the plan cache to become filled with compiled plans that are not reused. Setting optimize for ad hoc workloads to 1 affects only new plans; plans that are already in the plan cache are unaffected.

**What is policy management?**

Policy management in SQL SERVER 2008 allows you to define and enforce policies for configuring and managing SQL Server across the enterprise. Policy-based management is configured in SQL Server Management Studio (SSMS). Navigate to the Object Explorer and expand the management node and under that is the policy management node. In policy management you will see the policies, conditions, and facets nodes. (Read more here http:// bit.ly/ sqlinterview115)

**What are the basics of policy management?**

SQL server 2008 has introduced a policy management framework, which is the latest technique for the SQL server database engine. The SQL policy administrator uses SQL Server Management Studio to create policies that can handle entities on the server side like SQL Server objects and the instance of SQL Server databases. The policy management framework consists of three components: policy administrators (who create policies), policy management, and explicit administration. Policy-based management in SQL Server assists the database administrators in defining and enforcing policies that tie to database objects and instances. These policies allow the administrator to configure and manage SQL Server across the enterprise. (Read more here http:// bit.ly/ sqlinterview116)

**What are the policy management terms?**

To have a better grip on the concept of policy-based management, there are some key terms you need to understand: Target – A type of entity that is appropriately managed by policy-basedmanagement. A target can be a table, database or index, to name a few.

**Facet** - A property that can be managed in policy-based management. A clear example of a facet is the name of a trigger or the auto shrink property of database.

**Conditions** – Criteria that specify the state of a facet to true or false. For example, you can adjust the state of a facet that gives you clear specifications of all stored procedures in the schema named ‘Banking’.

**Policy** – A set of rules specified for the server objects or the properties of database. (Read more here http:// bit.ly/ sqlinterview117)

**What are the advantages of policy management?**

The following advantages can be achieved by appropriate administration of policy management system: • It interacts with various policies for successful system configuration. • It handles the changes in the systems that are the result of configurations against authoring policies. • It reduces the cost of ownership with a simple elaboration of administration tasks. • It detects various compliance issues in SQL Server Management Studio. (Read more here http:// bit.ly/ sqlinterview118)

**What is transparent data encryption?**

Transparent data encryption (TDE) introduces a new database option that encrypts the database files automatically, without needing to alter any applications. This prevents unauthorized users from accessing a database, even if they obtain the database files or database backup files. Transparent data encryption (TDE) performs real-time I/ O encryption and decryption of the data and log files. The encryption uses a database encryption key (DEK), which is stored in the database boot record for availability during recovery. When enabling TDE, you should immediately back up the certificate and the private key associated with the certificate.

**What is “Extensible Key Management” in SQL Server?**

The Extensible Key Management (EKM) feature allows third-party enterprise key management and Hardware Security Module (HSM) vendors to register their devices in SQL Server. Once registered, SQL Server users can use the encryption keys stored on these modules, as well as leveraging the advanced encryption features that these modules support. Examples include bulk encryption/ decryption and many key management functions such as key aging and key rotation. Data can be encrypted and decrypted using TSQL cryptographic statements, and SQL Server uses the external EKM device as the key store. By default, Extensible Key Management is off. To enable this feature, use the sp\_configure command.

**What are signed modules?**

SQL Server 2005 introduced the capability to sign modules within the database, such as stored procedures, functions, triggers and assemblies. The need to encrypt the definition or the logic inside these procedures and functions has been there for a long time in enterprises. Signed modules are an efficient and powerful way to do the same for SQL Server. By signing a module with a certificate, the certificate is then granted the relevant permission and goes beyond what can be achieved with the Execute As feature, especially from an auditing perspective. Data Definition Language (DDL) triggers cannot be signed.

**How do we use DBCC commands?** The Transact-SQL programming language provides DBCC statements that act as Database Consistency Commands for SQL Server. DBCC commands are used to perform the following tasks: • Maintenance tasks on a database, index, or filegroup. • Tasks that gather and display various types of information. • Validation operations on a database, table, index, catalog, filegroup, or allocation of database pages. • Miscellaneous tasks such as enabling trace flags or removing a DLL from memory. (Read more here http:// bit.ly/ sqlinterview119)

**What is the difference between ROLLBACK IMMEDIATE and WITH NO\_WAIT during ALTER DATABASE?**

ROLLBACK AFTER integer [SECONDS] | ROLLBACK IMMEDIATE:

Specifies whether to roll back after a specified number of seconds or immediately if the transaction is not complete.

**NO\_WAIT**: Specifies that if the requested database state or option change cannot complete immediately without waiting for transactions to commit or roll back on their own, then the database state or option change request will fail.

**What is database mirroring?**

Database mirroring involves two copies of a single database that typically reside on different computers. At any given time, only one copy of the database is currently available to clients, which is known as the principal database. Updates made by the clients to the principal database are applied to the other copy of the database, known as the mirror database. Mirroring involves applying the transaction log from every insertion, update, or delete made on the principal database to the mirror database.

**What are the Database Mirroring (DBM) Enhancements done with SQL Server 2008 R2?** There were a number of enhancements done with DBM, some to call out are: • Write-ahead on the incoming log stream on the mirror server. • Improved use of “Log Send” buffers. • Compression of the stream of transaction log records. • Automatic Recovery from Corrupted Pages.

**What is peer-to-peer replication?**

Peer-to-peer replication is a special type of transactional replication extension which provides a scale-out and high-availability solution by maintaining copies of data across multiple server instances. Peer-to-peer replication propagates transactionally consistent changes in near real-time. To avoid potential data inconsistency, make sure that you avoid conflicts in a peer-to-peer topology, even with conflict detection enabled. Conflict detection on peer-to-peer replication was introduced with SQL Server 2008 R2.

**What is bidirectional transactional replication?**

Bidirectional transactional replication is a specific form of transactional replication that allows both the publisher and the subscriber to send data to each other. This is not often used since a better option here would be to use peer-to-peer replication.

**What is failover clustering?**

With failover clustering, the nodes share disks, but only a single node has access to the database at a time. It is possible to install additional SQL Server failover cluster instances across the nodes; however, this configuration cannot be used to redirect workloads for a single database (for example, separating reads from writes). (Read more here http:// bit.ly/ sqlinterview121)

**What are the questions and considerations you will make for HA/ DR design?**

• Understand prioritized HA (High Availability) / DR (Disaster Recovery) requirements for the application. What are the SLA’s set by the customer. • Are customers comfortable or budgeted for a shared storage solution? • What is the Recovery Point Objective (RPO)? This decides the combination of configurations like failover clustering. Failover clustering is often deployed alongside database mirroring, with clustering used for local HA, and database mirroring used for DR. • Consider a geocluster (or stretch cluster) as a combined HA/ DR solution. This solution requires software to enable the cluster and storage-level replication from the storage vendor. • What is the Recovery Time Objective (RTO)? This is how fast the system has to get back online after a site failure. Though these are some of the high level questions, they do help narrow down a solution quickly or at least lead to more solution options.

**What is the concept of Piecemeal Restore on SQL Server?**

Online Piecemeal Restore is available from SQL Server 2005 Enterprise Edition onward. This allows administrators of databases that employ multiple filegroups to restore missing filegroups in stages while the database is online. Piecemeal Restore works with simple, bulk-logged, and full recovery models.

**What are OFFLINE datafiles In SQL Server?** The OFFLINE directive is a new feature of the ALTER DATABASE command. This allows databases that employ multiple filegroups to be online serving queries, while some of the database data may be unavailable because one or more filegroup( s) are marked as offline.

**Why can’t I run TRUNCATE TABLE on a published table?**

TRUNCATE TABLE is a minimally-logged operation and it does not fire any triggers. It is not possible to use them on replicated databases because replication cannot track the changes caused by the operation. Transactional replication tracks changes through the transaction log; Merge replication tracks changes through triggers on published tables.

**If a stored procedure is encrypted, then can we see its definition in Activity Monitor?**

No, we can’t see the definition of encrypted stored procedures in Activity Monitor. (Read more here http:// bit.ly/ sqlinterview122) SQL Server 2005 introduces “Signed Modules” which is highly recommended for encrypting SPs and functions.

**What are the different states a database can get into?** The standard states defined in sys.databases are:

0 = ONLINE

1 = RESTORING

2 = RECOVERING

3 = RECOVERY\_PENDING

4 = SUSPECT

5 = EMERGENCY

6 = OFFLINE

**How do you stop a log file from growing too big?**

If your transaction log file was growing too big and you need to manage its size, then instead of truncating the transaction log file you should choose one of the options mentioned below:

**Convert the recovery model to simple recovery**: If you change your recovery model to the simple recovery model, then you will not encounter the extraordinary growth of your log file.

**2) Start taking transaction log backup:** In the full recovery model, your transaction log will grow until you take a backup of it. You need to take the T-log backup at regular intervals. This way, your log would not grow beyond the size of your interval activity. (Read more here http:// bit.ly/ sqlinterview123)

**What is the Resource Governor in SQL Server?** The Resource Governor is a feature given by SQL Server 2008 to control and allocate CPU and memory resources depending on the priority of applications. The Resource Governor will control the allocation of CPU and memory for the SQL Server Relational Engine Instance.

**How would you define the Resource Governor?** The three core fundamental concepts required to define a Resource Governor are: •

**Resource pools** – These define the various groups of resources that can be used within the server. You can define groups for CPU and memory buckets into what workloads are defined. • **Workload groups** - A workload group serves as a container for session requests so that there can be a predictable performance guaranteed for the group. Typical classifications can be the CxO group, the reporting group, the developer group, and the admin group etc. •

**Classification function** – this is based on a set of user-defined criteria contained in a user defined function. The results of the function logic enable the Resource Governor to classify sessions into existing workload groups. The Resource Governor does not impose any controls on a Dedicated Administrator Connection (DAC).

**How do you restart SQL Server in single user mode?**

There are a couple of ways to start SQL Server in single user mode: • You can start an instance of SQL Server in single-user mode by using the startup option –m • You can use -m option with sqlcmd or Management Studio • You can also start sqlservr.exe with –m option

The CHECKPOINT process is not executed when you are using single user mode.

**What are the different backup options within SQL Server?**

At a high-level the 3 most important backups to understand are: •

**Full backup** – These backups contain ALL the data in a specific database. •

**Differential backup** - A differential backup contains only the data that has changed since the last full database backup. At restore time, the full backup is restored first, followed by the most recent differential backup. •

**Transactional log backups** - The transaction log is a serial record of all the transactions that have been performed against the database since the transaction log was last backed up. With transaction log backups, you can always recover the database to a specific point in time or to the point of failure in a FULL recovery model. You can also recover to a point in time using the bulk logged recovery model if the point in time you want to restore to was not during a bulk insert.

Before you can create the first log backup, you must create a full backup, such as a database backup. There are other special types of backups which we didn’t cover including: • Partial Backup

• File Backup

• Differential Partial Backup

• Differential File Backup

• Copy-Only Backups

**What are the different recovery models inside SQL Server?** There are 3 different recovery models inside SQL Server: •

**Simple recovery model** – This uses minimum administrative overhead for the transaction log, the simple recovery model risks significant work-loss exposure if the database is damaged. Data is recoverable only to the most recent backup. In simple recovery all transactions are truncating the log on each checkpoint.

**• Bulk-logged recovery model –** Used for logging transactions while not filling up the log during bulk operations. This is used for large-scale operations such as bulk import or index creation. Switching temporarily to the bulk-logged recovery model increases performance and reduces log space consumption. Log backups are still required if you want to be able to restore up to the point of failure.

**• Full recovery model** - The full recovery model guarantees the least risk of losing your work if a data file gets damaged. In this model, SQL Server fully logs all operations. In this recovery model, you can recover to any Point-in-time and it is the most recommended model for financial systems.

**What is the difference between DB mirroring and log shipping?**

Log shipping is one of the oldest forms of high-availability strategies inside SQL Server. The concept here is the primary database on the server is backed up and restored on one or more secondary servers. After this step, transaction logs are restored from the primary database to the secondary database over a periodic interval defined.

Database mirroring provides a redundant copy of a single database that is automatically configured to update the changes. Database mirroring works by sending transaction log records from the main principal database to the mirror server. The transaction log records are then replayed on the mirror database continuously. Some of the differences include: • A log shipping secondary can also be set to allow read-only access to the database in between transaction log restore operations. • The log shipping process is controlled through SQL Server agent jobs that perform the backups, copies, restores, and monitoring. • Database mirroring can detect failures automatically. Even automatic page repairs are possible. • With database mirroring failovers can also be automated.

**What are plan guides?**

Starting in SQL Server 2005, there is a new feature called Plan Guides that can help out in cases where you discover poorly performing queries that you don’t have direct control over (like ones made by third party applications). Plan guides influence optimization of queries by attaching query hints to them. When the query executes, SQL Server matches the query to the plan guide and attaches the OPTION clause to the query at run time. Plan guides that misuse query hints can cause complications, poor execution, and performance problems.

**How can you validate a backup copy of your database?**

The best option to validate a backup copy is the RESTORE VERIFYONLY command. This option checks to see that the backup set is complete and the entire backup is readable. However, RESTORE VERIFYONLY does not attempt to verify the structure of the data contained in the backup volumes. You can also restore a backup to check the validity of a backup copy.

**What is the ONLINE rebuilding of index?**

An online operation means that when online operations are occurring in the database, then the database is in a normal operational condition. The processes which are participating in online operations do not require exclusive access to the database. In the case of online indexing operations, when index operations (create, rebuild, dropping) occur they do not require exclusive access to the database and do not lock any database tables. This is a major important upgrade in SQL Server from previous versions. (Read more here http:// bit.ly/ sqlinterview124) The Online Indexing feature is available only in the Enterprise Edition of SQL Server

**What are the basics of table partitioning with SQL Server?**

The concept of partitioning is not new to SQL Server. In fact, some form of partitioning has been possible in every release of the product. However, partitioning has traditionally been cumbersome and thus underutilized by DBAs as a strategy. Because of the significant performance gains inherent in the concept, SQL Server 7.0 began improving the feature by enabling forms of partitioning through partitioned views (but not tables). For a lesson on Partitioned Tables see Chapters 8-11 of SQL Architecture Basics Joes 2 Pros Volume 3 ISBN: 1451579462. While the improvements in SQL Server 7.0 and SQL Server 2000 significantly enhanced performance when using partitioned views, they did not simplify the administration, design, or development of a partitioned dataset. When using partitioned views, all of the base tables (on which the view is defined) must be created and managed individually.

With SQL Server 2005 came a much more accessible and workable process, and SQL Server 2008 now offers the most advanced method for partitioning large datasets through partitioned table. However, the nature of partitioning a table is slightly different from slicing an apple into sections which are then permanently separated. After you partition a table, it remains a single, unified object with its underlying data contained in separate filegroups. This allows the table to function logically as a single object, while its data is physically stored in separate locations.

You can partition a logical set of data into multiple physical storage locations for manageability and performance. Partitions were introduced in SQL Server 2005. Both SQL Server 2005 and SQL Server 2008 allow up to 1000 partitions. Partition functions define boundaries for your tables. One partition function boundary value means two table partitions. The filegroups are the physical locations for these partitions. The main reason for partitioning a table is to get improved performance when executing DML on a large table.

A partition scheme maps the partitions to the filegroups. The partition scheme is used to create a partitioned table. The partition function sets datatype and range values. The partition scheme maps the partitions to the filegroups.

To create partition tables do your steps in this order:

1. Create the files and filegroups.

2. Create the partition function.

3. Create the partition scheme.

4. Create the table.

5. Populate the table.

**What are the steps to create a table partition?**

If you already have your files and filegroups set up, what are the 3 steps for creating a table partition: • Partition function – This defines how you want to partition the data. • Partition scheme – This defines where each of the partitions defined by the function will reside. • Attaching the partition scheme to a table – You map the partition scheme to the table based on a column’s data using the ON clause in table definition. Below is a code sample of how to create a partition function, use that function in a partition scheme and then base a partitioned table on the partition scheme.

**Log shipping**

SQL Server Log shipping allows you to automatically send transaction log backups from a *primary database* on a *primary server* instance to one or more *secondary databases* on separate *secondary server* instances. The transaction log backups are applied to each of the secondary databases individually. An optional third server instance, known as the *monitor server*, records the history and status of backup and restore operations and, optionally, raises alerts if these operations fail to occur as scheduled.

**Benefits**

* Provides a disaster-recovery solution for a single primary database and one or more secondary databases, each on a separate instance of SQL Server.
* Supports limited read-only access to secondary databases (during the interval between restore jobs).
* Allows a user-specified delay between when the primary server backs up the log of the primary database and when the secondary servers must restore (apply) the log backup. A longer delay can be useful, for example, if data is accidentally changed on the primary database. If the accidental change is noticed quickly, a delay can let you retrieve still unchanged data from a secondary database before the change is reflected there.

primary server

The instance of SQL Server that is your production server.

primary database

The database on the primary server that you want to back up to another server. All administration of the log shipping configuration through SQL Server Management Studio is performed from the primary database.

secondary server

The instance of SQL Server where you want to keep a warm standby copy of your primary database.

secondary database

The warm standby copy of the primary database. The secondary database may be in either the RECOVERING state or the STANDBY state, which leaves the database available for limited read-only access.

monitor server

An optional instance of SQL Server that tracks all of the details of log shipping, including:

* When the transaction log on the primary database was last backed up.
* When the secondary servers last copied and restored the backup files.
* Information about any backup failure alerts.

|  |
| --- |
| **Important note Important** |
| Once the monitor server has been configured, it cannot be changed without removing log shipping first. |

backup job

A SQL Server Agent job that performs the backup operation, logs history to the local server and the monitor server, and deletes old backup files and history information. When log shipping is enabled, the job category "Log Shipping Backup" is created on the primary server instance.

copy job

A SQL Server Agent job that copies the backup files from the primary server to a configurable destination on the secondary server and logs history on the secondary server and the monitor server. When log shipping is enabled on a database, the job category "Log Shipping Copy" is created on each secondary server in a log shipping configuration.

restore job

A SQL Server Agent job that restores the copied backup files to the secondary databases. It logs history on the local server and the monitor server, and deletes old files and old history information. When log shipping is enabled on a database, the job category "Log Shipping Restore" is created on the secondary server instance.

alert job

A SQL Server Agent job that raises alerts for primary and secondary databases when a backup or restore operation does not complete successfully within a specified threshold. When log shipping is enabled on a database, job category "Log Shipping Alert" is created on the monitor server instance.

|  |
| --- |
| **Tip Tip** |
| For each alert, you need to specify an alert number. Also, be sure to configure the alert to notify an operator when an alert is raised. |

Arrow icon used with Back to Top linkLog shipping consists of three operations:

1. Back up the transaction log at the primary server instance.
2. Copy the transaction log file to the secondary server instance.
3. Restore the log backup on the secondary server instance.

The log can be shipped to multiple secondary server instances. In such cases, operations 2 and 3 are duplicated for each secondary server instance.

A log shipping configuration does not automatically fail over from the primary server to the secondary server. If the primary database becomes unavailable, any of the secondary databases can be brought online manually.

You can use a secondary database for reporting purposes.

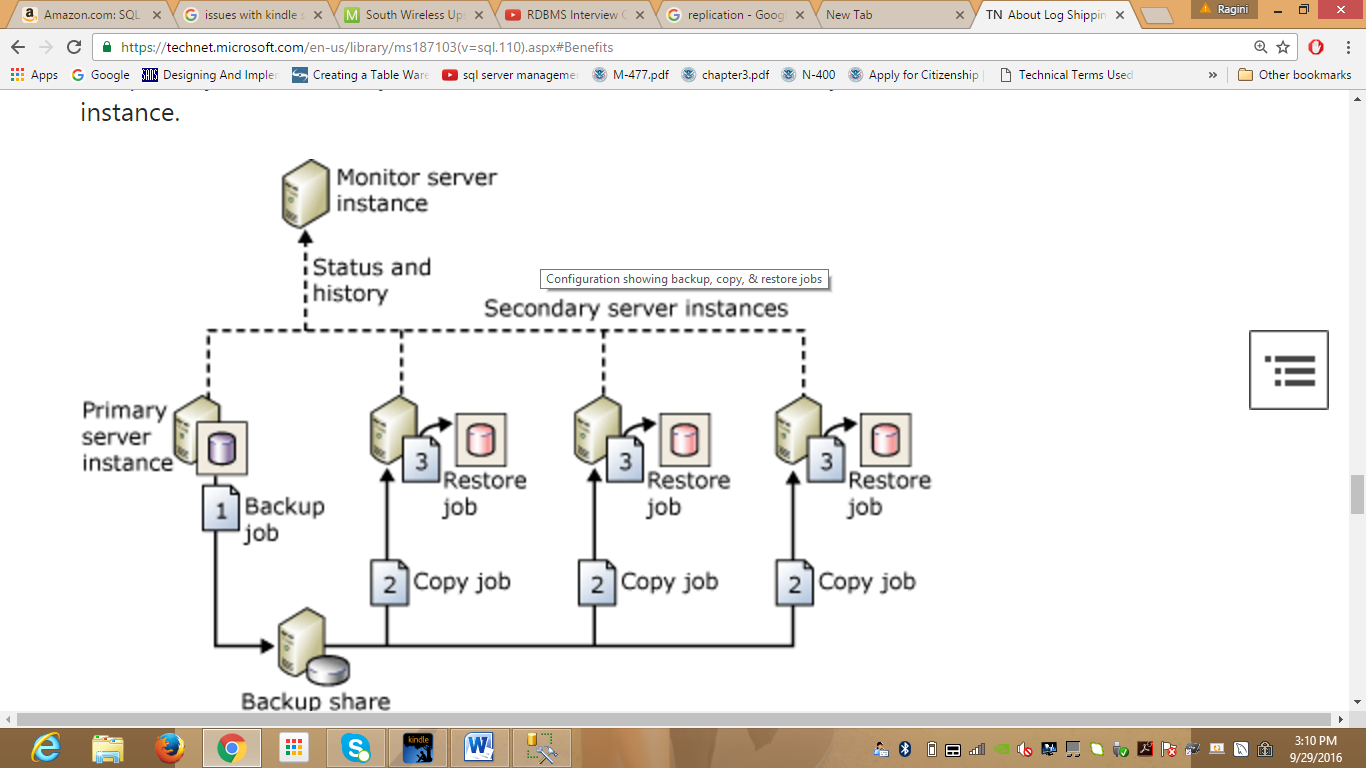
In addition, you can configure alerts for your log shipping configuration.

A Typical Log Shipping Configuration

The following figure shows a log shipping configuration with the primary server instance, three secondary server instances, and a monitor server instance. The figure illustrates the steps performed by backup, copy, and restorejobs, as follows:

1. The primary server instance runs the backup job to back up the transaction log on the primary database. This server instance then places the log backup into a primary log-backup file, which it sends to the backup folder. In this figure, the backup folder is on a shared directory—the *backup share*.
2. Each of the three secondary server instances runs its own copy job to copy the primary log-backup file to its own local destination folder.
3. Each secondary server instance runs its own restore job to restore the log backup from the local destination folder onto the local secondary database.

The primary and secondary server instances send their own history and status to the monitor server instance.



|  |
| --- |
| **Note Note** |
| AlwaysOn Availability Groups and database mirroring are mutually exclusive. A database that is configured for one of these features cannot be configured for the other. |

It is possible to preserve log shipping configurations when upgrading from SQL Server 2005, SQL Server 2008, or SQL Server 2008 R2 to SQL Server 2012. This topic describes alternative scenarios and best practices for upgrading a log shipping configuration.

|  |
| --- |
| **Note Note** |
| [Backup compression](https://technet.microsoft.com/en-us/library/bb964719(v=sql.110).aspx) was introduced in SQL Server 2008 Enterprise. An upgraded log shipping configuration uses the **backup compression default** server-level configuration option to control whether backup compression is used for the transaction log backup files. The backup compression behavior of log backups can be specified for each log shipping configuration. For more information, see [Configure Log Shipping (SQL Server)](https://technet.microsoft.com/en-us/library/ms190640(v=sql.110).aspx). |

**In This Topic:**

* [Protect Your Data Before the Upgrade](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#ProtectData)
* [Upgrading the Monitor Server Instance](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#UpgradeMonitor)
* [Upgrading Log Shipping Configurations with a Single Secondary Server](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#UpgradeSingleSecondary)
* [Upgrading Multiple Secondary Server Instances](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#MultipleSecondaries)
* [Redeploying Log Shipping](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#Redeploying)

## [Protect Your Data Before the Upgrade](javascript:void(0))

As a best practice, we recommend that you protect your data before a log shipping upgrade.

**To protect your data**

1. Perform a full database backup on every primary database.

For more information, see [Create a Full Database Backup (SQL Server)](https://technet.microsoft.com/en-us/library/ms187510(v=sql.110).aspx).

1. Run the [DBCC CHECKDB](https://technet.microsoft.com/en-us/library/ms176064(v=sql.110).aspx) command on every primary database.

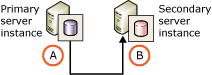
## [Upgrading the Monitor Server Instance](javascript:void(0))

The monitor server instance, if any, can be upgraded at any time.

While the monitor server is being upgraded, the log shipping configuration continues to work, but its status is not recorded in the tables on the monitor. Any alerts that have been configured will not be triggered while the monitor server is being upgraded. After the upgrade, you can update the information in the monitor tables by executing the[sp\_refresh\_log\_shipping\_monitor](https://technet.microsoft.com/en-us/library/ms189754(v=sql.110).aspx) system stored procedure.

## [Upgrading Log Shipping Configurations with a Single Secondary Server](javascript:void(0))

The upgrade process described in this section assumes a configuration consisting of the primary server and only one secondary server. This configuration is represented in the following illustration, which shows a primary server instance, A, and a single secondary server instance, B.



For information about upgrading multiple secondary servers, see [Upgrading Multiple Secondary Server Instances](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#MultipleSecondaries), later in this topic.

**In This Section:**

* [Upgrading the Secondary Server Instance](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#UpgradeSecondary)
* [Upgrading the Primary Server Instance](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#UpgradePrimary)

### Upgrading the Secondary Server Instance

The upgrade process involves upgrading the secondary server instances of a SQL Server 2005, SQL Server 2008, or SQL Server 2008 R2 log shipping configuration to SQL Server 2012 before upgrading the primary server instance. Always upgrade the secondary server instance first. If the primary server were upgraded before a secondary server, log shipping would fail because a backup created on a newer version of SQL Server cannot be restored on an older version of SQL Server.

Log shipping continues throughout the upgrade process because the upgraded secondary servers continue to restore the log backups from the SQL Server 2005, SQL Server 2008, or SQL Server 2008 R2 primary server. The process for upgrading the secondary server instances depends partly on whether the log shipping configuration possesses multiple secondary servers. For more information, see [Upgrading Multiple Secondary Server Instances](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#MultipleSecondaries), later in this topic.

While the secondary server instance is being upgraded, the log shipping copy and restore jobs do not run, so unrestored transaction log backups will accumulate. The amount of accumulation depends on the frequency of scheduled backup on the primary server. Also, if a separate monitor server has been configured, alerts might be raised indicating restores have not been performed for longer than the configured interval.

Once the secondary server has been upgraded, the log shipping agents jobs resume and continue to copy and restore log backups from the primary server instance, server A. The amount of time required for the secondary server to bring the secondary database up to date varies, depending on the time taken to upgrade the secondary server and the frequency of the backups on the primary server.

|  |
| --- |
| **Note Note** |
| During the server upgrade, the secondary database is not upgraded to a SQL Server 2012 database. It will get upgraded only if it is brought online. |
| **Important note Important** |
| The RESTORE WITH STANDBY option is not supported for a database that requires upgrading. If an upgraded secondary database has been configured by using RESTORE WITH STANDBY, transaction logs can no longer be restored after upgrade. To resume log shipping on that secondary database, you will need to set up log shipping again on that standby server. For more information about the STANDBY option, see [RESTORE Arguments (Transact-SQL)](https://technet.microsoft.com/en-us/library/ms178615(v=sql.110).aspx). |

### Upgrading the Primary Server Instance

When planning an upgrade, a significant consideration is the amount of time that your database will be unavailable. The simplest upgrade scenario involves the database being unavailable while you upgrade the primary server (scenario 1, below).

At the cost of a more complicated upgrade process, you can maximize your database availability by failing over the SQL Server 2005, SQL Server 2008, or SQL Server 2008 R2 primary server to a SQL Server 2012 secondary server before upgrading the original primary server (scenario 2, below). There are two variants of the failover scenario. You can switch back to the original primary server and keep the original log shipping configuration. Alternatively, you can remove the original log shipping configuration before upgrading the original primary server and later create a new configuration using the new primary server. This section describes both these scenarios.

|  |
| --- |
| **Important note Important** |
| Be sure to upgrade the secondary server instance before upgrading the primary server instance. For more information, see[Upgrading the Secondary Server Instance](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#UpgradeSecondary), earlier in this topic. |

**In This Section:**

* [Scenario 1: Upgrade Primary Server Instance Without Failover](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#Scenario1)
* [Scenario 2: Upgrade Primary Server Instance with Failover](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#Scenario2)

#### Scenario 1: Upgrade Primary Server Instance Without Failover

This is the simpler scenario, but it causes more downtime than using failover. The primary server instance is simply upgraded and the database is unavailable during this upgrade.

Once the server is upgraded, the database is automatically brought back online, which causes it to be upgraded. After the database is upgraded, the log shipping jobs resume.

#### Scenario 2: Upgrade Primary Server Instance with Failover

This scenario maximizes availability and minimizes downtime. It utilizes a controlled failover to the secondary server instance, which keeps the database available while the original primary server instance is upgraded. Downtime is limited to the relatively short time required to fail over, rather than the time required to upgrade the primary server instance.

Upgrading the primary server instance with failover involves three general procedures: performing a controlled failover to the secondary server, upgrading the original primary server instance to SQL Server 2012, and setting up log shipping on a SQL Server 2012 primary server instance. These procedures are described in this section.

|  |
| --- |
| **Important note Important** |
| If you plan to have the secondary server instance as the new primary server instance, you need to remove the log shipping configuration. Log shipping will need to be reconfigured from the new primary to the new secondary, after the original primary server instance has been upgraded. For more information, see [Remove Log Shipping (SQL Server)](https://technet.microsoft.com/en-us/library/ms189071(v=sql.110).aspx). |

**In This Section:**

* [Procedure 1: Perform a Controlled Failover to the Secondary Server](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#Procedure1)
* [Procedure 2:. Upgrade the Original Primary Server Instance to SQL Server 2012](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#Procedure2)
* [Procedure 3: Set Up Log Shipping on SQL Server 2012](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#Procedure3)

#### Procedure 1: Perform a Controlled Failover to the Secondary Server

Controlled failover to the secondary server:

1. Manually perform a [tail-log backup](https://technet.microsoft.com/en-us/library/ms179314(v=sql.110).aspx) of the transaction log on the primary database specifying WITH NORECOVERY. This log backup captures any log records that have not been backed up yet and takes the database offline. Note that while the database is offline, the log shipping backup job will fail.

The following example creates a tail log backup of the AdventureWorks database on the primary server. The backup file is named Failover\_AW\_20080315.trn:

BACKUP LOG AdventureWorks

TO DISK = N'\\FileServer\LogShipping\AdventureWorks\Failover\_AW\_20080315.trn'

WITH NORECOVERY;

GO

We recommend that you use a distinct file naming convention to differentiate the manually-created backup file from the backup files created by the log shipping backup job.

1. On the secondary server:
   1. Ensure that all backups taken automatically by the log shipping backup jobs have been applied. To check which backup jobs have been applied, use the [sp\_help\_log\_shipping\_monitor](https://technet.microsoft.com/en-us/library/ms187820(v=sql.110).aspx) system stored procedure on the monitor server or on the primary and secondary servers. The same file should be listed in the **last\_backup\_file**,**last\_copied\_file**, and **last\_restored\_file** columns. If any of the backup files have not been copied and restored, manually invoke the agent copy and restore jobs for the log shipping configuration.

For information about starting a job, see [Start a Job](https://technet.microsoft.com/en-us/library/ms190774(v=sql.110).aspx).

* 1. Copy your the final log backup file that you created in step 1 from the file share to the local location that is used by log shipping on the secondary server.
  2. Restore the final log backup specifying WITH RECOVERY to bring the database online. As part of being brought online, the database will upgraded to SQL Server 2012.

The following example restores the tail log backup of the AdventureWorks database on the secondary database. The example uses the WITH RECOVERY option, which brings the database online:

RESTORE LOG AdventureWorks

FROM DISK = N'c:\logshipping\Failover\_AW\_20080315.trn'

WITH RECOVERY;

GO

|  |
| --- |
| **Note Note** |
| For a configuration that contains more than one secondary server, there are additional considerations. For more information, see [Upgrading Multiple Secondary Server Instances](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#MultipleSecondaries), later in this topic. |

* 1. Fail over the database by redirecting clients from the original primary server (server A) to the online secondary server (server B).
  2. Take care that the transaction log of the secondary database does not fill while the database is online. To prevent the transaction log from filling, you might need to back it up. If so, we recommend that you back it up to a shared location, a *backup share*, to make the backups available for restoring on the other server instance.

#### Procedure 2: Upgrade the Original Primary Server Instance to SQL Server 2012

After you upgrade the original primary server instance to SQL Server 2012, the database will still be offline and in the format.

#### Procedure 3: Set Up Log Shipping on SQL Server 2012

The rest of the upgrade process depends on whether log shipping is still configured, as follows:

* If you have kept the SQL Server 2005, SQL Server 2008, or SQL Server 2008 R2 log shipping configuration, switch back to the original primary server instance. For more information, see [To Switch Back to the Original Primary Server Instance](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#SwitchToOrigPrimary), later in this section.
* If you removed the log shipping configuration before failing over, create a new log shipping configuration in which the original secondary server instance is the new primary server instance. For more information, see [To Keep the Old Secondary Server Instance As the New Primary Server Instance](https://technet.microsoft.com/en-us/library/cc645954(v=sql.110).aspx#KeepOldSecondaryAsNewPrimary), later in this section.

#### To Switch Back to the Original Primary Server Instance

1. On the interim primary server (server B), back up the tail of the log using WITH NORECOVERY to create a tail-log backup and take the database offline. The tail log backup is named Switchback\_AW\_20080315.trn.For example:
2. BACKUP LOG AdventureWorks
3. TO DISK = N'\\FileServer\LogShipping\AdventureWorks\Switchback\_AW\_20080315.trn'
4. WITH NORECOVERY;
5. GO
6. If any transaction log backups were taken on the interim primary database, other than the tail backup that you created in step 1, restore those log backups using WITH NORECOVERY to the offline database on the original primary server (server A). The database is upgraded to SQL Server 2012 format when the first log backup is restored.
7. Restore the tail-log backup, Switchback\_AW\_20080315.trn, on the original primary database (on server A) using WITH RECOVERY to bring the database online.
8. Fail over back to the original primary database (on server A) by redirecting clients to the online secondary server from the original primary server.

After the database comes online, the original log shipping configuration will resume.

#### To Keep the Old Secondary Server Instance As the New Primary Server Instance

Establish a new log shipping configuration using the old secondary server instance, B, as the primary server and the old primary server instance, A, as the new secondary server, as follows:

|  |
| --- |
| **Important note Important** |
| The old log shipping configuration should have been removed from the original primary server at the start of the process before taking the manual transaction log backup that took the database offline. |

1. To avoid performing a complete backup and restore of the database on the new secondary server (server A), apply the log backups from the new primary database to the new secondary database. In the example configuration, this involves restoring the log backups taken on server B to the database on server A.
2. Back up the log from the new primary database (on server B).
3. Restore the log backups to the new secondary server instance (server A) using WITH NORECOVERY. The first restore operation updates the database to SQL Server 2012.
4. Configure log shipping with the former secondary server (server B) as the primary server instance.

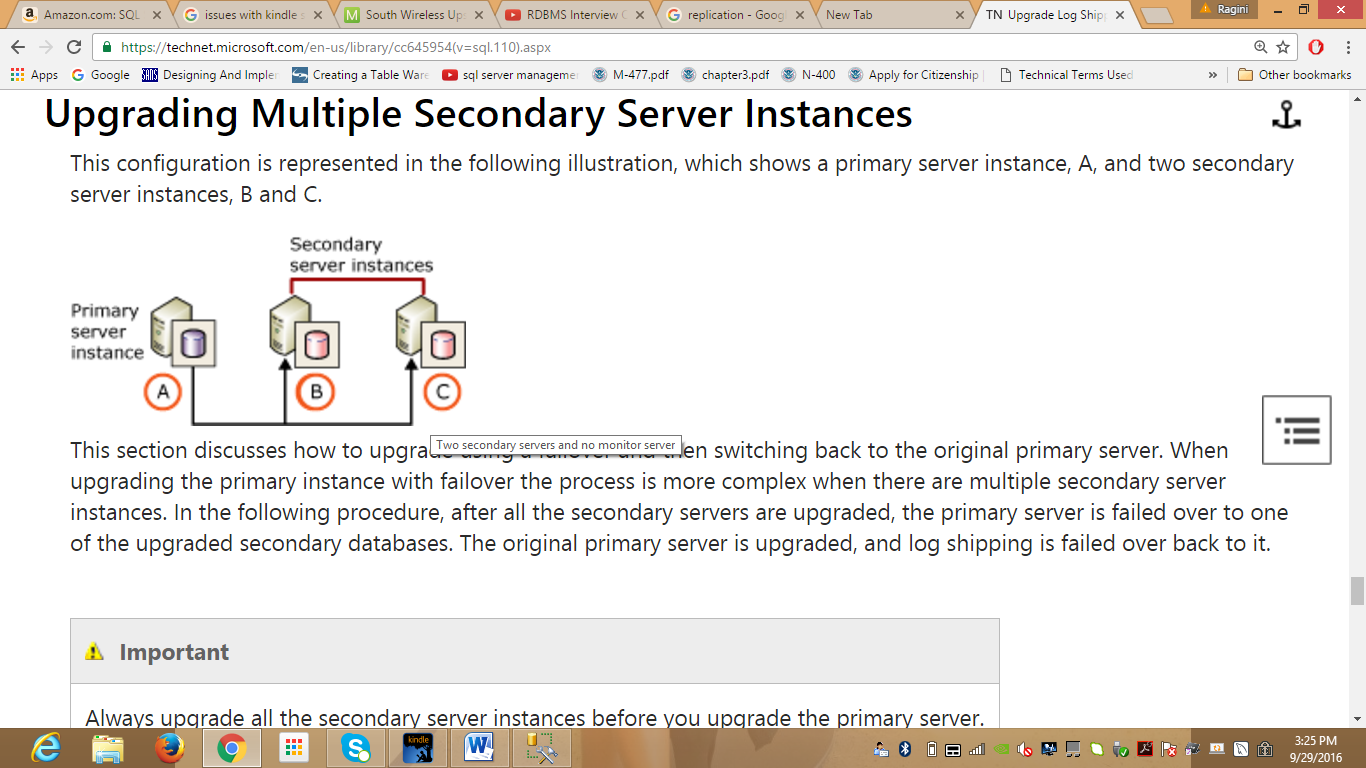
|  |
| --- |
| **Important note Important** |
| If you use SQL Server Management Studio, specify that the secondary database is already initialized. |

1. For more information, see [Configure Log Shipping (SQL Server)](https://technet.microsoft.com/en-us/library/ms190640(v=sql.110).aspx).
2. Fail over the database by redirecting clients from the original primary server (server A) to the online secondary server (server B).

|  |
| --- |
| **Important note Important** |
| When you failover to a new primary database, you should ensure that its metadata is consistent with the metadata of the original primary database. For more information, see [Manage Metadata When Making a Database Available on Another Server Instance (SQL Server)](https://technet.microsoft.com/en-us/library/ms187580(v=sql.110).aspx). |

## [Upgrading Multiple Secondary Server Instances](javascript:void(0))

This configuration is represented in the following illustration, which shows a primary server instance, A, and two secondary server instances, B and C.



|  |
| --- |
| **Important note Important** |
| Always upgrade all the secondary server instances before you upgrade the primary server. |

**To upgrade using a failover and then switching back to original primary server**

1. Upgrade all the secondary server instances (server B and server C).
2. Obtain the tail of the transaction log of the primary database (on server A), and take the database offline, by backing up the transaction log using WITH NORECOVERY.
3. On the secondary server to which you plan to fail over (server B), bring the secondary database online, by restoring the log backup using WITH RECOVERY.
4. On every other secondary server (server C), leave the secondary database offline by restoring the log backup using WITH NORECOVERY.

|  |
| --- |
| **Note Note** |
| The log shipping copy and restore jobs will run on the secondary servers, but the jobs will do nothing because new log-backup files will not be placed on the backup share. |

1. Fail over the database by redirecting clients from the original primary server (server A) to the online secondary server (server B). The online database becomes an interim primary server, keeping the database available while the original primary server is offline (server A).
2. Upgrade the original primary server (server A).
3. On the database to which you failed over—the interim primary database (on server B), manually back up the transaction log using WITH NORECOVERY. This takes the database offline.
4. Restore all transaction log backups that you created on the interim primary database (on server B) to every other secondary database (on server C) using WITH NORECOVERY. This allows log shipping to continue from the original primary database after its upgrade, without requiring a full database restore on each secondary database.
5. Restore the transaction log from the interim primary server (server B) to the original primary database (on server A) using WITH RECOVERY.

## [Redeploying Log Shipping](javascript:void(0))

If you do not want to migrate your log shipping configuration using one of the procedures shown above, you can redeploy log shipping from scratch by reinitializing your secondary database with a full backup and restore of the primary database. This may be a desirable option if you have a small database or if high availability is not crucial during the upgrade procedure.