**Transact-SQL Optimization Tips**

* **Use views and stored procedures instead of heavy-duty queries.**  
  This can reduce network traffic, because your client will send to server only stored procedure or view name (perhaps with some parameters) instead of large heavy-duty queries text. This can be used to facilitate permission management also, because you can restrict user access to table columns they should not see.
* **Try to use constraints instead of triggers, whenever possible.**  
  Constraints are much more efficient than triggers and can boost performance. So, you should use constraints instead of triggers, whenever possible.
* **Use table variables instead of temporary tables.**  
  Table variables require less locking and logging resources than temporary tables, so table variables should be used whenever possible. The table variables are available in SQL Server 2000 only.
* **Try to use UNION ALL statement instead of UNION, whenever possible.**  
  The UNION ALL statement is much faster than UNION, because UNION ALL statement does not look for duplicate rows, and UNION statement does look for duplicate rows, whether or not they exist.
* **Try to avoid using the DISTINCT clause, whenever possible.**  
  Because using the DISTINCT clause will result in some performance degradation, you should use this clause only when it is necessary.
* **Try to avoid using SQL Server cursors, whenever possible.**  
  SQL Server cursors can result in some performance degradation in comparison with select statements. Try to use correlated sub-query or derived tables, if you need to perform row-by-row operations.
* **Try to avoid the HAVING clause, whenever possible.**  
  The HAVING clause is used to restrict the result set returned by the GROUP BY clause. When you use GROUP BY with the HAVING clause, the GROUP BY clause divides the rows into sets of grouped rows and aggregates their values, and then the HAVING clause eliminates undesired aggregated groups. In many cases, you can write your select statement so, that it will contain only WHERE and GROUP BY clauses without HAVING clause. This can improve the performance of your query.
* **If you need to return the total table's row count, you can use alternative way instead of SELECT COUNT(\*) statement.**  
  Because SELECT COUNT(\*) statement make a full table scan to return the total table's row count, it can take very many time for the large table. There is another way to determine the total row count in a table. You can use sysindexes system table, in this case. There is ROWS column in the sysindexes table. This column contains the total row count for each table in your database. So, you can use the following select statement instead of SELECT COUNT(\*): SELECT rows FROM sysindexes WHERE id = OBJECT\_ID('table\_name') AND indid < 2 So, you can improve the speed of such queries in several times.
* **Include SET NOCOUNT ON statement into your stored procedures to stop the message indicating the number of rows affected by a T-SQL statement.**  
  This can reduce network traffic, because your client will not receive the message indicating the number of rows affected by a T-SQL statement.
* **Try to restrict the queries result set by using the WHERE clause.**  
  This can results in good performance benefits, because SQL Server will return to client only particular rows, not all rows from the table(s). This can reduce network traffic and boost the overall performance of the query.
* **Use the select statements with TOP keyword or the SET ROWCOUNT statement, if you need to return only the first n rows.**  
  This can improve performance of your queries, because the smaller result set will be returned. This can also reduce the traffic between the server and the clients.
* **Try to restrict the queries result set by returning only the particular columns from the table, not all table's columns.**  
  This can results in good performance benefits, because SQL Server will return to client only particular columns, not all table's columns. This can reduce network traffic and boost the overall performance of the query.

1.Indexes  
2.avoid more number of triggers on the table  
3.unnecessary complicated joins  
4.correct use of Group by clause with the select list  
5.in worst cases Denormalization  
  
**Index Optimization tips**

* Every index increases the time in takes to perform INSERTS, UPDATES and DELETES, so the number of indexes should not be very much. Try to use maximum 4-5 indexes on one table, not more. If you have read-only table, then the number of indexes may be increased.
* Keep your indexes as narrow as possible. This reduces the size of the index and reduces the number of reads required to read the index.
* Try to create indexes on columns that have integer values rather than character values.
* If you create a composite (multi-column) index, the order of the columns in the key are very important. Try to order the columns in the key as to enhance selectivity, with the most selective columns to the leftmost of the key.
* If you want to join several tables, try to create surrogate integer keys for this purpose and create indexes on their columns.
* Create surrogate integer primary key (identity for example) if your table will not have many insert operations.
* Clustered indexes are more preferable than nonclustered, if you need to select by a range of values or you need to sort results set with GROUP BY or ORDER BY.
* If your application will be performing the same query over and over on the same table, consider creating a covering index on the table.
* You can use the SQL Server Profiler Create Trace Wizard with "Identify Scans of Large Tables" trace to determine which tables in your database may need indexes. This trace will show which tables are being scanned by queries instead of using an index.
* You can use sp\_MSforeachtable undocumented stored procedure to rebuild all indexes in your database. Try to schedule it to execute during CPU idle time and slow production periods.  
  sp\_MSforeachtable @command1="print '?' DBCC DBREINDEX ('?')"

**T-SQL Queries**

1. 2 tables

| **Employee** | **Phone** |
| --- | --- |
| empid empname salary mgrid | empid phnumber |

1. Select all employees who doesn't have phone?  
   SELECT empname  
   FROM Employee  
   WHERE (empid NOT IN  
   (SELECT DISTINCT empid  
   FROM phone))
2. Select the employee names who is having more than one phone numbers.  
   SELECT empname  
   FROM employee  
   WHERE (empid IN  
   (SELECT empid  
   FROM phone  
   GROUP BY empid  
   HAVING COUNT(empid) > 1))
3. Select the details of 3 max salaried employees from employee table.  
   SELECT TOP 3 empid, salary  
   FROM employee  
   ORDER BY salary DESC
4. Display all managers from the table. (manager id is same as emp id)  
   SELECT empname  
   FROM employee  
   WHERE (empid IN  
   (SELECT DISTINCT mgrid  
   FROM employee))
5. Write a Select statement to list the Employee Name, Manager Name under a particular manager?  
   SELECT e1.empname AS EmpName, e2.empname AS ManagerName  
   FROM Employee e1 INNER JOIN  
   Employee e2 ON e1.mgrid = e2.empid  
   ORDER BY e2.mgrid
6. 2 tables emp and phone.  
   emp fields are - empid, name  
   Ph fields are - empid, ph (office, mobile, home). Select all employees who doesn't have any ph nos.  
   SELECT \*  
   FROM employee LEFT OUTER JOIN  
   phone ON employee.empid = phone.empid  
   WHERE (phone.office IS NULL OR phone.office = ' ')  
   AND (phone.mobile IS NULL OR phone.mobile = ' ')  
   AND (phone.home IS NULL OR phone.home = ' ')
7. Find employee who is living in more than one city.   
   Two Tables:

| **Emp** | **City** |
| --- | --- |
| Empid empName Salary | Empid City |

1. SELECT empname, fname, lname  
   FROM employee  
   WHERE (empid IN  
   (SELECT empid  
   FROM city  
   GROUP BY empid  
   HAVING COUNT(empid) > 1))
2. Find all employees who is living in the same city. (table is same as above)  
   SELECT fname  
   FROM employee  
   WHERE (empid IN  
   (SELECT empid  
   FROM city a  
   WHERE city IN  
   (SELECT city  
   FROM city b  
   GROUP BY city  
   HAVING COUNT(city) > 1)))
3. There is a table named MovieTable with three columns - moviename, person and role. Write a query which gets the movie details where Mr. Amitabh and Mr. Vinod acted and their role is actor.  
   SELECT DISTINCT m1.moviename  
   FROM MovieTable m1 INNER JOIN  
   MovieTable m2 ON m1.moviename = m2.moviename  
   WHERE (m1.person = 'amitabh' AND m2.person = 'vinod' OR  
   m2.person = 'amitabh' AND m1.person = 'vinod') AND (m1.role = 'actor') AND (m2.role = 'actor')  
   ORDER BY m1.moviename
4. There are two employee tables named emp1 and emp2. Both contains same structure (salary details). But Emp2 salary details are incorrect and emp1 salary details are correct. So, write a query which corrects salary details of the table emp2  
   update a set a.sal=b.sal from emp1 a, emp2 b where a.empid=b.empid
5. Given a Table named “Students” which contains studentid, subjectid and marks. Where there are 10 subjects and 50 students. Write a Query to find out the Maximum marks obtained in each subject.
6. In this same tables now write a SQL Query to get the studentid also to combine with previous results.
7. Three tables – student , course, marks – how do go at finding name of the students who got max marks in the diff courses.  
   SELECT student.name, course.name AS coursename, marks.sid, marks.mark  
   FROM marks INNER JOIN  
   student ON marks.sid = student.sid INNER JOIN  
   course ON marks.cid = course.cid  
   WHERE (marks.mark =  
   (SELECT MAX(Mark)  
   FROM Marks MaxMark  
   WHERE MaxMark.cID = Marks.cID))
8. There is a table day\_temp which has three columns dayid, day and temperature. How do I write a query to get the difference of temperature among each other for seven days of a week?  
   SELECT a.dayid, a.dday, a.tempe, a.tempe - b.tempe AS Difference  
   FROM day\_temp a INNER JOIN  
   day\_temp b ON a.dayid = b.dayid + 1  
   OR  
   Select a.day, a.degree-b.degree from temperature a, temperature b where a.id=b.id+1
9. There is a table which contains the names like this. a1, a2, a3, a3, a4, a1, a1, a2 and their salaries. Write a query to get grand total salary, and total salaries of individual employees in one query.  
   SELECT empid, SUM(salary) AS salary  
   FROM employee  
   GROUP BY empid WITH ROLLUP  
   ORDER BY empid
10. **How to know how many tables contains empno as a column in a database?**SELECT COUNT(\*) AS Counter  
    FROM syscolumns  
    WHERE (name = 'empno')
11. **Find duplicate rows in a table? OR I have a table with one column which has many records which are not distinct. I need to find the distinct values from that column and number of times it’s repeated.**  
    SELECT sid, mark, COUNT(\*) AS Counter  
    FROM marks  
    GROUP BY sid, mark  
    HAVING (COUNT(\*) > 1)
12. **How to delete the rows which are duplicate (don’t delete both duplicate records).**  
    SET ROWCOUNT 1  
    DELETE yourtable  
    FROM yourtable a  
    WHERE (SELECT COUNT(\*) FROM yourtable b WHERE b.name1 = a.name1 AND b.age1 = a.age1) > 1  
    WHILE @@rowcount > 0  
      DELETE yourtable  
      FROM yourtable a  
      WHERE (SELECT COUNT(\*) FROM yourtable b WHERE b.name1 = a.name1 AND b.age1 = a.age1) > 1  
    SET ROWCOUNT 0
13. **How to find 6th highest salary**SELECT TOP 1 salary  
    FROM (SELECT DISTINCT TOP 6 salary  
    FROM employee  
    ORDER BY salary DESC) a  
    ORDER BY salary
14. **Find top salary among two tables**SELECT TOP 1 sal  
    FROM (SELECT MAX(sal) AS sal  
    FROM sal1  
    UNION  
    SELECT MAX(sal) AS sal  
    FROM sal2) a  
    ORDER BY sal DESC
15. **Write a query to convert all the letters in a word to upper case**SELECT UPPER('test')
16. **Write a query to round up the values of a number. For example even if the user enters 7.1 it should be rounded up to 8.**SELECT CEILING (7.1)
17. **Write a SQL Query to find first day of month?**SELECT DATENAME(dw, DATEADD(dd, - DATEPART(dd, GETDATE()) + 1, GETDATE())) AS FirstDay

| **Datepart** | **Abbreviations** |
| --- | --- |
| year | yy, yyyy |
| quarter | qq, q |
| month | mm, m |
| dayofyear | dy, y |
| day | dd, d |
| week | wk, ww |
| weekday | dw |
| hour | hh |
| minute | mi, n |
| second | ss, s |
| millisecond | ms |

1. Table A contains column1 which is primary key and has 2 values (1, 2) and Table B contains column1 which is primary key and has 2 values (2, 3). Write a query which returns the values that are not common for the tables and the query should return one column with 2 records.  
   SELECT tbla.a  
   FROM tbla, tblb  
   WHERE tbla.a <>  
   (SELECT tblb.a  
   FROM tbla, tblb  
   WHERE tbla.a = tblb.a)  
   UNION  
   SELECT tblb.a  
   FROM tbla, tblb  
   WHERE tblb.a <>  
   (SELECT tbla.a  
   FROM tbla, tblb  
   WHERE tbla.a = tblb.a)  
     
   OR (better approach)  
     
   SELECT a  
   FROM tbla  
   WHERE a NOT IN  
   (SELECT a  
   FROM tblb)  
   UNION ALL  
   SELECT a  
   FROM tblb  
   WHERE a NOT IN  
   (SELECT a  
   FROM tbla)
2. There are 3 tables Titles, Authors and Title-Authors (check PUBS db). Write the query to get the author name and the number of books written by that author, the result should start from the author who has written the maximum number of books and end with the author who has written the minimum number of books.  
   SELECT authors.au\_lname, COUNT(\*) AS BooksCount  
   FROM authors INNER JOIN  
   titleauthor ON authors.au\_id = titleauthor.au\_id INNER JOIN  
   titles ON titles.title\_id = titleauthor.title\_id  
   GROUP BY authors.au\_lname  
   ORDER BY BooksCount DESC
3. UPDATE emp\_master  
   SET emp\_sal =   
   CASE  
   WHEN emp\_sal > 0 AND emp\_sal <= 20000 THEN (emp\_sal \* 1.01)   
   WHEN emp\_sal > 20000 THEN (emp\_sal \* 1.02)   
   END
4. List all products with total quantity ordered, if quantity ordered is null show it as 0.  
   SELECT name, CASE WHEN SUM(qty) IS NULL THEN 0 WHEN SUM(qty) > 0 THEN SUM(qty) END AS tot  
   FROM [order] RIGHT OUTER JOIN  
   product ON [order].prodid = product.prodid  
   GROUP BY name  
   Result:  
   coke 60  
   mirinda 0  
   pepsi 10
5. ANY, SOME, or ALL?  
   ALL means greater than every value--in other words, greater than the maximum value. For example, >ALL (1, 2, 3) means greater than 3.  
   ANY means greater than at least one value, that is, greater than the minimum. So >ANY (1, 2, 3) means greater than 1. SOME is an SQL-92 standard equivalent for ANY.
6. IN & = (difference in correlated sub query)  
     
   **INDEX**
7. **What is Index? It’s purpose?**Indexes in databases are similar to indexes in books. In a database, an index allows the database program to find data in a table without scanning the entire table. An index in a database is a list of values in a table with the storage locations of rows in the table that contain each value. Indexes can be created on either a single column or a combination of columns in a table and are implemented in the form of B-trees. An index contains an entry with one or more columns (the search key) from each row in a table. A B-tree is sorted on the search key, and can be searched efficiently on any leading subset of the search key. For example, an index on columns **A**, **B**, **C** can be searched efficiently on **A**, on **A**, **B**, and **A**, **B**, **C**.
8. **Explain about Clustered and non clustered index? How to choose between a Clustered Index and a Non-Clustered Index?**There are clustered and nonclustered indexes. A clustered index is a special type of index that reorders the way records in the table are physically stored. Therefore table can have only one clustered index. The leaf nodes of a clustered index contain the data pages.   
   A nonclustered 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 nodes of a nonclustered index does not consist of the data pages. Instead, the leaf nodes contain index rows.  
   Consider using a clustered index for:
   * Columns that contain a large number of distinct values.
   * Queries that return a range of values using operators such as BETWEEN, >, >=, <, and <=.
   * Columns that are accessed sequentially.
   * Queries that return large result sets.  
     Non-clustered indexes have the same B-tree structure as clustered indexes, with two significant differences:
   * The data rows are not sorted and stored in order based on their non-clustered keys.
   * The leaf layer of a non-clustered index does not consist of the data pages. Instead, the leaf nodes contain index rows. Each index row contains the non-clustered key value and one or more row locators that point to the data row (or rows if the index is not unique) having the key value.
   * Per table only 249 non clustered indexes.
9. **Disadvantage of index?**Every index increases the time in takes to perform INSERTS, UPDATES and DELETES, so the number of indexes should not be very much.
10. **Given a scenario that I have a 10 Clustered Index in a Table to all their 10 Columns. What are the advantages and disadvantages?**A: Only 1 clustered index is possible.
11. **How can I enforce to use particular index?**You can use index hint (index=<index\_name>) after the table name.  
    SELECT au\_lname FROM authors (index=aunmind)
12. **What is Index Tuning?**One of the hardest tasks facing database administrators is the selection of appropriate columns for non-clustered indexes. You should consider creating non-clustered indexes on any columns that are frequently referenced in the WHERE clauses of SQL statements. Other good candidates are columns referenced by JOIN and GROUP BY operations.  
    You may wish to also consider creating non-clustered indexes that cover all of the columns used by certain frequently issued queries. These queries are referred to as “covered queries” and experience excellent performance gains.  
    Index Tuning is the process of finding appropriate column for non-clustered indexes.  
    SQL Server provides a wonderful facility known as the Index Tuning Wizard which greatly enhances the index selection process.
13. **Difference between Index defrag and Index rebuild?**When you create an index in the database, the index information used by queries is stored in index pages. The sequential index pages are chained together by pointers from one page to the next. When changes are made to the data that affect the index, the information in the index can become scattered in the database. Rebuilding an index reorganizes the storage of the index data (and table data in the case of a clustered index) to remove fragmentation. This can improve disk performance by reducing the number of page reads required to obtain the requested data  
    DBCC INDEXDEFRAG - Defragments clustered and secondary indexes of the specified table or view.  
    \*\*
14. **What is sorting and what is the difference between sorting & clustered indexes?**The ORDER BY clause sorts query results by one or more columns up to 8,060 bytes. This will happen by the time when we retrieve data from database. Clustered indexes physically sorting data, while inserting/updating the table.
15. **What are statistics, under what circumstances they go out of date, how do you update them?**Statistics determine the selectivity of the indexes. If an indexed column has unique values then the selectivity of that index is more, as opposed to an index with non-unique values. Query optimizer uses these indexes in determining whether to choose an index or not while executing a query.  
    Some situations under which you should update statistics:  
    1) If there is significant change in the key values in the index  
    2) If a large amount of data in an indexed column has been added, changed, or removed (that is, if the distribution of key values has changed), or the table has been truncated using the TRUNCATE TABLE statement and then repopulated  
    3) Database is upgraded from a previous version
16. **What is fillfactor? What is the use of it ? What happens when we ignore it? When you should use low fill factor?**When you create a clustered index, the data in the table is stored in the data pages of the database according to the order of the values in the indexed columns. When new rows of data are inserted into the table or the values in the indexed columns are changed, Microsoft® SQL Server™ 2000 may have to reorganize the storage of the data in the table to make room for the new row and maintain the ordered storage of the data. This also applies to nonclustered indexes. When data is added or changed, SQL Server may have to reorganize the storage of the data in the nonclustered index pages. When a new row is added to a full index page, SQL Server moves approximately half the rows to a new page to make room for the new row. This reorganization is known as a page split. Page splitting can impair performance and fragment the storage of the data in a table.When creating an index, you can specify a fill factor to leave extra gaps and reserve a percentage of free space on each leaf level page of the index to accommodate future expansion in the storage of the table's data and reduce the potential for page splits. The fill factor value is a percentage from 0 to 100 that specifies how much to fill the data pages after the index is created. A value of 100 means the pages will be full and will take the least amount of storage space. This setting should be used only when there will be no changes to the data, for example, on a read-only table. A lower value leaves more empty space on the data pages, which reduces the need to split data pages as indexes grow but requires more storage space. This setting is more appropriate when there will be changes to the data in the table.  
    **DATA TYPES**
17. **What are the data types in SQL**

| bigint | Binary | bit | char | cursor |
| --- | --- | --- | --- | --- |
| datetime | Decimal | float | image | int |
| money | Nchar | ntext | nvarchar | real |
| smalldatetime | Smallint | smallmoney | text | timestamp |
| tinyint | Varbinary | Varchar | uniqueidentifier |  |

1. **Difference between char and nvarchar / char and varchar data-type?**  
   char[(n)] - Fixed-length non-Unicode character data with length of n bytes. n must be a value from 1 through 8,000. Storage size is n bytes. The SQL-92 synonym for char is character.  
   nvarchar(n) - Variable-length Unicode character data of n characters. n must be a value from 1 through 4,000. Storage size, in bytes, is two times the number of characters entered. The data entered can be 0 characters in length. The SQL-92 synonyms for nvarchar are national char varying and national character varying.   
   varchar[(n)] - Variable-length non-Unicode character data with length of n bytes. n must be a value from 1 through 8,000. Storage size is the actual length in bytes of the data entered, not n bytes. The data entered can be 0 characters in length. The SQL-92 synonyms for varchar are char varying or character varying.
2. **GUID datasize?**128bit
3. **How GUID becoming unique across machines?**To ensure uniqueness across machines, the ID of the network card is used (among others) to compute the number.
4. **What is the difference between text and image data type?**Text and image. Use text for character data if you need to store more than 255 characters in SQL Server 6.5, or more than 8000 in SQL Server 7.0. Use image for binary large objects (BLOBs) such as digital images. With text and image data types, the data is not stored in the row, so the limit of the page size does not apply.All that is stored in the row is a pointer to the database pages that contain the data.Individual text, ntext, and image values can be a maximum of 2-GB, which is too long to store in a single data row.  
     
   **JOINS**
5. **What are joins?**Sometimes we have to select data from two or more tables to make our result complete. We have to perform a join.
6. **How many types of Joins?**Joins can be categorized as:
   * Inner joins (the typical join operation, which uses some comparison operator like = or <>). These include equi-joins and natural joins.   
     Inner joins use a comparison operator to match rows from two tables based on the values in common columns from each table. For example, retrieving all rows where the student identification number is the same in both the **students** and **courses** tables.
   * Outer joins. Outer joins can be a left, a right, or full outer join.   
     Outer joins are specified with one of the following sets of keywords when they are specified in the FROM clause:
     + LEFT JOIN or LEFT OUTER JOIN -The result set of a left outer join includes all the rows from the left table specified in the LEFT OUTER clause, not just the ones in which the joined columns match. When a row in the left table has no matching rows in the right table, the associated result set row contains null values for all select list columns coming from the right table.
     + RIGHT JOIN or RIGHT OUTER JOIN - A right outer join is the reverse of a left outer join. All rows from the right table are returned. Null values are returned for the left table any time a right table row has no matching row in the left table.
     + FULL JOIN or FULL OUTER JOIN - A full outer join returns all rows in both the left and right tables. Any time a row has no match in the other table, the select list columns from the other table contain null values. When there is a match between the tables, the entire result set row contains data values from the base tables.
   * Cross joins - Cross joins return all rows from the left table, each row from the left table is combined with all rows from the right table. Cross joins are also called **Cartesian products.** (A Cartesian join will get you a Cartesian product. A Cartesian join is when you join every row of one table to every row of another table. You can also get one by joining every row of a table to every row of itself.)
7. **What is self join?**A table can be joined to itself in a self-join.
8. **What are the differences between UNION and JOINS?**A join selects columns from 2 or more tables. A union selects rows.
9. **Can I improve performance by using the ANSI-style joins instead of the old-style joins?**Code Example 1:  
   select o.name, i.name  
   from sysobjects o, sysindexes i  
   where o.id = i.id  
   Code Example 2:  
   select o.name, i.name  
   from sysobjects o inner join sysindexes i  
   on o.id = i.id  
   You will not get any performance gain by switching to the ANSI-style JOIN syntax.  
   Using the ANSI-JOIN syntax gives you an important advantage: Because the join logic is cleanly separated from the filtering criteria, you can understand the query logic more quickly.  
   The SQL Server old-style JOIN executes the filtering conditions before executing the joins, whereas the ANSI-style JOIN reverses this procedure (join logic precedes filtering).  
   Perhaps the most compelling argument for switching to the ANSI-style JOIN is that Microsoft has explicitly stated that SQL Server will not support the old-style OUTER JOIN syntax indefinitely. Another important consideration is that the ANSI-style JOIN supports query constructions that the old-style JOIN syntax does not support.
10. **What is derived table?**Derived tables are SELECT statements in the FROM clause referred to by an alias or a user-specified name. The result set of the SELECT in the FROM clause forms a table used by the outer SELECT statement. For example, this SELECT uses a derived table to find if any store carries all book titles in the **pubs** database:   
    SELECT ST.stor\_id, ST.stor\_name  
    FROM stores AS ST,  
         (SELECT stor\_id, COUNT(DISTINCT title\_id) AS title\_count  
          FROM sales  
          GROUP BY stor\_id  
         ) AS SA  
    WHERE ST.stor\_id = SA.stor\_id  
    AND SA.title\_count = (SELECT COUNT(\*) FROM titles)  
    **STORED PROCEDURE**
11. **What is Stored procedure?**A stored procedure is a set of Structured Query Language (SQL) statements that you assign a name to and store in a database in compiled form so that you can share it between a number of programs.
    * They allow modular programming.
    * They allow faster execution.
    * They can reduce network traffic.
    * They can be used as a security mechanism.
12. **What are the different types of Storage Procedure?**
    1. Temporary Stored Procedures - SQL Server supports two types of temporary procedures: local and global. A local temporary procedure is visible only to the connection that created it. A global temporary procedure is available to all connections. Local temporary procedures are automatically dropped at the end of the current session. Global temporary procedures are dropped at the end of the last session using the procedure. Usually, this is when the session that created the procedure ends. Temporary procedures named with # and ## can be created by any user.
    2. System stored procedures are created and stored in the **master** database and have the **sp\_** prefix.(or xp\_) System stored procedures can be executed from any database without having to qualify the stored procedure name fully using the database name **master**. (If any user-created stored procedure has the same name as a system stored procedure, the user-created stored procedure will never be executed.)
    3. Automatically Executing Stored Procedures - One or more stored procedures can execute automatically when SQL Server starts. The stored procedures must be created by the system administrator and executed under the **sysadmin** fixed server role as a background process. The procedure(s) cannot have any input parameters.
    4. User stored procedure
13. **How do I mark the stored procedure to automatic execution?**You can use the sp\_procoption system stored procedure to mark the stored procedure to automatic execution when the SQL Server will start. Only objects in the master database owned by dbo can have the startup setting changed and this option is restricted to objects that have no parameters.  
    USE master  
    EXEC sp\_procoption 'indRebuild', 'startup', 'true')
14. **How can you optimize a stored procedure?**
15. **How will know whether the SQL statements are executed?**When used in a stored procedure, the RETURN statement can specify an integer value to return to the calling application, batch, or procedure. If no value is specified on RETURN, a stored procedure returns the value 0.  The stored procedures return a value of 0 when no errors were encountered. Any nonzero value indicates an error occurred.
16. **Why one should not prefix user stored procedures with sp\_?**It is strongly recommended that you do not create any stored procedures using sp\_ as a prefix. SQL Server always looks for a stored procedure beginning with sp\_ in this order:
    1. The stored procedure in the master database.
    2. The stored procedure based on any qualifiers provided (database name or owner).
    3. The stored procedure using dbo as the owner, if one is not specified.

Therefore, although the user-created stored procedure prefixed with sp\_ may exist in the current database, the master database is always checked first, even if the stored procedure is qualified with the database name.

1. **What can cause a Stored procedure execution plan to become invalidated and/or fall out of cache?**
   1. Server restart
   2. Plan is aged out due to low use
   3. DBCC FREEPROCCACHE (sometime desired to force it)
2. **When do one need to recompile stored procedure?**if a new index is added from which the stored procedure might benefit, optimization does not automatically happen (until the next time the stored procedure is run after SQL Server is restarted).
3. **SQL Server provides three ways to recompile a stored procedure:** 
   * The **sp\_recompile** system stored procedure forces a recompile of a stored procedure the next time it is run.
   * Creating a stored procedure that specifies the WITH RECOMPILE option in its definition indicates that SQL Server does not cache a plan for this stored procedure; the stored procedure is recompiled each time it is executed. Use the WITH RECOMPILE option when stored procedures take parameters whose values differ widely between executions of the stored procedure, resulting in different execution plans to be created each time. Use of this option is uncommon, and causes the stored procedure to execute more slowly because the stored procedure must be recompiled each time it is executed.
   * You can force the stored procedure to be recompiled by specifying the WITH RECOMPILE option when you execute the stored procedure. Use this option only if the parameter you are supplying is atypical or if the data has significantly changed since the stored procedure was created.
4. **How to find out which stored procedure is recompiling? How to stop stored procedures from recompiling?**
5. **I have Two Stored Procedures SP1 and SP2 as given below. How the Transaction works, whether SP2 Transaction succeeds or fails?**CREATE PROCEDURE SP1 AS  
   BEGIN TRAN  
   INSERT INTO MARKS (SID,MARK,CID) VALUES (5,6,3)  
   EXEC SP2  
   ROLLBACK  
   GO  
     
   CREATE PROCEDURE SP2 AS  
   BEGIN TRAN  
   INSERT INTO MARKS (SID,MARK,CID) VALUES (100,100,103)  
   commit tran  
   GO  
   Both will get roll backed.
6. CREATE PROCEDURE SP1 AS  
   BEGIN TRAN  
       INSERT INTO MARKS (SID,MARK,CID) VALUES (5,6,3)  
       BEGIN TRAN  
           INSERT INTO STUDENT (SID,NAME1) VALUES (1,'SA')  
       commit tran  
   ROLLBACK TRAN  
   GO  
   Both will get roll backed.
7. **How will you handle Errors in Sql Stored Procedure?**INSERT NonFatal VALUES (@Column2)  
   IF @@ERROR <>0  
    BEGIN  
     PRINT 'Error Occured'  
    END   
   <http://www.sqlteam.com/item.asp?ItemID=2463>
8. **How will you raise an error in sql?**RAISERROR - Returns a user-defined error message and sets a system flag to record that an error has occurred. Using RAISERROR, the client can either retrieve an entry from the sysmessages table or build a message dynamically with user-specified severity and state information. After the message is defined it is sent back to the client as a server error message.
9. I have a stored procedure like  
   commit tran  
   create table a()  
   insert into table b  
   --  
   --  
   rollback tran  
   what will be the result? Is table created? data will be inserted in table b?
10. **What do you do when one procedure is blocking the other?  
    \*\***
11. **How you will return XML from Stored Procedure?**You use the FOR XML clause of the SELECT statement, and within the FOR XML clause you specify an XML mode: RAW, AUTO, or EXPLICIT.
12. **What are the differences between RAW, AUTO and Explicit modes in retrieving data from SQL Server in XML format?  
    \*\***
13. **Can a Stored Procedure call itself (recursive). If so then up to what level and can it be control?**Stored procedures are nested when one stored procedure calls another. You can nest stored procedures up to 32 levels. The nesting level increases by one when the called stored procedure begins execution and decreases by one when the called stored procedure completes execution. Attempting to exceed the maximum of 32 levels of nesting causes the whole calling stored procedure chain to fail. The current nesting level for the stored procedures in execution is stored in the @@NESTLEVEL function.  
    eg:  
    SET NOCOUNT ON  
    USE master  
    IF OBJECT\_ID('dbo.sp\_calcfactorial') IS NOT NULL  
    DROP PROC dbo.sp\_calcfactorial  
    GO  
    CREATE PROC dbo.sp\_calcfactorial  
    @base\_number int, @factorial int OUT  
    AS  
    DECLARE @previous\_number int  
    IF (@base\_number<2) SET @factorial=1 -- Factorial of 0 or 1=1  
    ELSE BEGIN  
    SET @previous\_number=@base\_number-1  
    EXEC dbo.sp\_calcfactorial @previous\_number, @factorial OUT -- **Recursive** call  
    IF (@factorial=-1) RETURN(-1) -- Got an error, return  
    SET @factorial=@factorial\*@base\_number  
    END  
    RETURN(0)  
    GO  
      
    calling proc.  
    DECLARE @factorial int   
    EXEC dbo.sp\_calcfactorial 4, @factorial OUT   
    SELECT @factorial
14. **Nested Triggers**Triggers are nested when a trigger performs an action that initiates another trigger, which can initiate another trigger, and so on. Triggers can be nested up to 32 levels, and you can control whether triggers can be nested through the nested triggers server configuration option.
15. **What is an extended stored procedure? Can you instantiate a COM object by using T-SQL?**An extended stored procedure is a function within a DLL (written in a programming language like C, C++ using Open Data Services (ODS) API) that can be called from T-SQL, just the way we call normal stored procedures using the EXEC statement.
16. **Difference between view and stored procedure?**Views can have only select statements (create, update, truncate, delete statements are not allowed) Views cannot have “select into”, “Group by” “Having”, ”Order by”
17. **What is a Function & what are the different user defined functions?**Function is a saved Transact-SQL routine that returns a value. User-defined functions cannot be used to perform a set of actions that modify the global database state. User-defined functions, like system functions, can be invoked from a query. They also can be executed through an EXECUTE statement like stored procedures.
    1. Scalar Functions  
       Functions are scalar-valued if the RETURNS clause specified one of the scalar data types
    2. Inline Table-valued Functions  
       If the RETURNS clause specifies TABLE with no accompanying column list, the function is an inline function.
    3. Multi-statement Table-valued Functions  
       If the RETURNS clause specifies a TABLE type with columns and their data types, the function is a multi-statement table-valued function.
18. **What are the difference between a function and a stored procedure?**
    1. Functions can be used in a select statement where as procedures cannot
    2. Procedure takes both input and output parameters but Functions takes only input parameters
    3. Functions cannot return values of type text, ntext, image & timestamps where as procedures can
    4. Functions can be used as user defined datatypes in create table but procedures cannot   
       \*\*\*Eg:-create table <tablename>(name varchar(10),salary getsal(name))  
       Here getsal is a user defined function which returns a salary type, when table is created no storage is allotted for salary type, and getsal function is also not executed, But when we are fetching some values from this table, getsal function get’s executed and the return   
       Type is returned as the result set.
19. How to debug a stored procedure?  
      
    **TRIGGER**
20. **What is Trigger? What is its use? What are the types of Triggers? What are the new kinds of triggers in sql 2000?**Triggers are a special class of stored procedure defined to execute automatically when an UPDATE, INSERT, or DELETE statement is issued against a table or view. Triggers are powerful tools that sites can use to enforce their business rules automatically when data is modified.  
    The CREATE TRIGGER statement can be defined with the FOR UPDATE, FOR INSERT, or FOR DELETE clauses to target a trigger to a specific class of data modification actions. When FOR UPDATE is specified, the IF UPDATE (column\_name) clause can be used to target a trigger to updates affecting a particular column.  
    You can use the FOR clause to specify when a trigger is executed:
    * AFTER (default) - The trigger executes after the statement that triggered it completes. If the statement fails with an error, such as a constraint violation or syntax error, the trigger is not executed. AFTER triggers cannot be specified for views.
    * INSTEAD OF -The trigger executes in place of the triggering action. INSTEAD OF triggers can be specified on both tables and views. You can define only one INSTEAD OF trigger for each triggering action (INSERT, UPDATE, and DELETE). INSTEAD OF triggers can be used to perform enhance integrity checks on the data values supplied in INSERT and UPDATE statements. INSTEAD OF triggers also let you specify actions that allow views, which would normally not support updates, to be updatable.  
      An INSTEAD OF trigger can take actions such as:
      + Ignoring parts of a batch.
      + Not processing a part of a batch and logging the problem rows.
      + Taking an alternative action if an error condition is encountered.

In SQL Server 6.5 you could define only 3 triggers per table, one for INSERT, one for UPDATE and one for DELETE. From SQL Server 7.0 onwards, this restriction is gone, and you could create multiple triggers per each action. But in 7.0 there's no way to control the order in which the triggers fire. In SQL Server 2000 you could specify which trigger fires first or fires last using sp\_settriggerorder.  
Till SQL Server 7.0, triggers fire only after the data modification operation happens. So in a way, they are called post triggers. But in SQL Server 2000 you could create pre triggers also.

1. **When should one use "instead of Trigger"? Example**CREATE TABLE BaseTable  
   (  
   PrimaryKey int IDENTITY(1,1),  
   Color nvarchar(10) NOT NULL,  
   Material nvarchar(10) NOT NULL,  
   ComputedCol AS (Color + Material)  
   )  
   GO  
     
   --Create a view that contains all columns from the base table.  
   CREATE VIEW InsteadView  
   AS SELECT PrimaryKey, Color, Material, ComputedCol  
   FROM BaseTable  
   GO  
     
   --Create an INSTEAD OF INSERT trigger on tthe view.  
   CREATE TRIGGER InsteadTrigger on InsteadView  
   INSTEAD OF INSERT  
   AS  
   BEGIN  
   --Build an INSERT statement ignoring inserrted.PrimaryKey and   
   --inserted.ComputedCol.  
   INSERT INTO BaseTable  
   SELECT Color, Material  
   FROM inserted  
   END  
   GO  
     
   -- can insert value to basetable by this insert into basetable(color,material) values ('red','abc')  
     
   -- insert into InsteadView(color,material)) values ('red','abc') can't do this.  
   -- It will give error "'PrimaryKey' iin table 'InsteadView' cannot be null."  
     
   -- can insert value through table by this<  
   insert into InsteadView values (1,'red','abc',1) --PrimaryKey, ComputedCol wont take values from here
2. **Difference between trigger and stored procedure?**Trigger will get execute automatically when an UPDATE, INSERT, or DELETE statement is issued against a table or view.  
   We have to call stored procedure manually, or it can execute automatic when the SQL Server starts (You can use the sp\_procoption system stored procedure to mark the stored procedure to automatic execution when the SQL Server will start.
3. **The following trigger generates an e-mail whenever a new title is added.**CREATE TRIGGER reminder  
   ON titles  
   FOR INSERT  
   AS  
   EXEC master..xp\_sendmail 'MaryM', 'New title, mention in the next report to distributors.'
4. **Drawback of trigger? Its alternative solution?**Triggers are generally used to implement business rules, auditing. Triggers can also be used to extend the referential integrity checks, but wherever possible, use constraints for this purpose, instead of triggers, as constraints are much faster.  
     
   **LOCK**
5. **What are locks?**Microsoft® SQL Server™ 2000 uses locking to ensure transactional integrity and database consistency. Locking prevents users from reading data being changed by other users, and prevents multiple users from changing the same data at the same time. If locking is not used, data within the database may become logically incorrect, and queries executed against that data may produce unexpected results.
6. **What are the different types of locks?**SQL Server uses these resource lock modes.

| **Lock mode** | **Description** |
| --- | --- |
| Shared (S) | Used for operations that do not change or update data (read-only operations), such as a SELECT statement. |
| Update (U) | Used on resources that can be updated. Prevents a common form of deadlock that occurs when multiple sessions are reading, locking, and potentially updating resources later. |
| Exclusive (X) | Used for data-modification operations, such as INSERT, UPDATE, or DELETE. Ensures that multiple updates cannot be made to the same resource at the same time. |
| Intent | Used to establish a lock hierarchy. The types of intent locks are: intent shared (IS), intent exclusive (IX), and shared with intent exclusive (SIX). |
| Schema | Used when an operation dependent on the schema of a table is executing. The types of schema locks are: schema modification (Sch-M) and schema stability (Sch-S). |
| Bulk Update (BU) | Used when bulk-copying data into a table and the TABLOCK hint is specified. |

1. **What is a dead lock? Give a practical sample? How you can minimize the deadlock situation? What is a deadlock and what is a live lock? How will you go about resolving deadlocks?**Deadlock is a situation when two processes, each having a lock on one piece of data, attempt to acquire a lock on the other's piece. Each process  would wait indefinitely for the other to release the lock, unless one of the user processes is terminated. SQL Server detects deadlocks and terminates one user's process.  
   A livelock is one, where a request for an exclusive lock is repeatedly denied because a series of overlapping shared locks keeps interfering. SQL Server detects the situation after four denials and refuses further shared locks. (A livelock also occurs when read transactions monopolize a table or page, forcing a write transaction to wait indefinitely.)
2. **What is isolation level?**An isolation level determines the degree of isolation of data between concurrent transactions. The default SQL Server isolation level is Read Committed. A lower isolation level increases concurrency, but at the expense of data correctness. Conversely, a higher isolation level ensures that data is correct, but can affect concurrency negatively. The isolation level required by an application determines the locking behavior SQL Server uses.  
   SQL-92 defines the following isolation levels, all of which are supported by SQL Server:
   * Read uncommitted (the lowest level where transactions are isolated only enough to ensure that physically corrupt data is not read).
   * Read committed (SQL Server default level).
   * Repeatable read.
   * Serializable (the highest level, where transactions are completely isolated from one another).

| **Isolation level** | **Dirty read** | **Nonrepeatable read** | **Phantom** |
| --- | --- | --- | --- |
| Read uncommitted | Yes | Yes | Yes |
| Read committed | No | Yes | Yes |
| Repeatable read | No | No | Yes |
| Serializable | No | No | No |

1. Uncommitted Dependency (Dirty Read) - Uncommitted dependency occurs when a second transaction selects a row that is being updated by another transaction. The second transaction is reading data that has not been committed yet and may be changed by the transaction updating the row. For example, an editor is making changes to an electronic document. During the changes, a second editor takes a copy of the document that includes all the changes made so far, and distributes the document to the intended audience.  
   Inconsistent Analysis (Nonrepeatable Read) Inconsistent analysis occurs when a second transaction accesses the same row several times and reads different data each time. Inconsistent analysis is similar to uncommitted dependency in that another transaction is changing the data that a second transaction is reading. However, in inconsistent analysis, the data read by the second transaction was committed by the transaction that made the change. Also, inconsistent analysis involves multiple reads (two or more) of the same row and each time the information is changed by another transaction; thus, the term nonrepeatable read. For example, an editor reads the same document twice, but between each reading, the writer rewrites the document. When the editor reads the document for the second time, it has changed.  
   Phantom Reads Phantom reads occur when an insert or delete action is performed against a row that belongs to a range of rows being read by a transaction. The transaction's first read of the range of rows shows a row that no longer exists in the second or succeeding read, as a result of a deletion by a different transaction. Similarly, as the result of an insert by a different transaction, the transaction's second or succeeding read shows a row that did not exist in the original read. For example, an editor makes changes to a document submitted by a writer, but when the changes are incorporated into the master copy of the document by the production department, theyfind that new unedited material has been added to the document by the author. This problem could be avoided if no one could add new material to the document until the editor and production department finish working with the original document.
2. **nolock? What is the difference between the REPEATABLE READ and SERIALIZE isolation levels?  
   Locking Hints -** A range of table-level locking hints can be specified using the SELECT, INSERT, UPDATE, and DELETE statements to direct Microsoft® SQL Server 2000 to the type of locks to be used. Table-level locking hints can be used when a finer control of the types of locks acquired on an object is required. These locking hints override the current transaction isolation level for the session.

| **Locking hint** | **Description** |
| --- | --- |
| HOLDLOCK | Hold a shared lock until completion of the transaction instead of releasing the lock as soon as the required table, row, or data page is no longer required. HOLDLOCK is equivalent to SERIALIZABLE. |
| NOLOCK | Do not issue shared locks and do not honor exclusive locks. When this option is in effect, it is possible to read an uncommitted transaction or a set of pages that are rolled back in the middle of a read. Dirty reads are possible. Only applies to the SELECT statement. |
| PAGLOCK | Use page locks where a single table lock would usually be taken. |
| READCOMMITTED | Perform a scan with the same locking semantics as a transaction running at the READ COMMITTED isolation level. By default, SQL Server 2000 operates at this isolation level. |
| READPAST | Skip locked rows. This option causes a transaction to skip rows locked by other transactions that would ordinarily appear in the result set, rather than block the transaction waiting for the other transactions to release their locks on these rows. The READPAST lock hint applies only to transactions operating at READ COMMITTED isolation and will read only past row-level locks. Applies only to the SELECT statement. |
| READUNCOMMITTED | Equivalent to NOLOCK. |
| REPEATABLEREAD | Perform a scan with the same locking semantics as a transaction running at the REPEATABLE READ isolation level. |
| ROWLOCK | Use row-level locks instead of the coarser-grained page- and table-level locks. |
| SERIALIZABLE | Perform a scan with the same locking semantics as a transaction running at the SERIALIZABLE isolation level. Equivalent to HOLDLOCK. |
| TABLOCK | Use a table lock instead of the finer-grained row- or page-level locks. SQL Server holds this lock until the end of the statement. However, if you also specify HOLDLOCK, the lock is held until the end of the transaction. |
| TABLOCKX | Use an exclusive lock on a table. This lock prevents others from reading or updating the table and is held until the end of the statement or transaction. |
| UPDLOCK | Use update locks instead of shared locks while reading a table, and hold locks until the end of the statement or transaction. UPDLOCK has the advantage of allowing you to read data (without blocking other readers) and update it later with the assurance that the data has not changed since you last read it. |
| XLOCK | Use an exclusive lock that will be held until the end of the transaction on all data processed by the statement. This lock can be specified with either PAGLOCK or TABLOCK, in which case the exclusive lock applies to the appropriate level of granularity. |

1. For example, if the transaction isolation level is set to SERIALIZABLE, and the table-level locking hint NOLOCK is used with the SELECT statement, key-range locks typically used to maintain serializable transactions are not taken.  
   USE pubs  
   GO  
   SET TRANSACTION ISOLATION LEVEL SERIALIZABLE  
   GO  
   BEGIN TRANSACTION  
   SELECT au\_lname FROM authors WITH (NOLOCK)  
   GO
2. **What is escalation of locks?**Lock escalation is the process of converting a lot of low level locks (like row locks, page locks) into higher level locks (like table locks). Every lock is a memory structure too many locks would mean, more memory being occupied by locks. To prevent this from happening, SQL Server escalates the many fine-grain locks to fewer coarse-grain locks. Lock escalation threshold was definable in SQL Server 6.5, but from SQL Server 7.0 onwards it's dynamically managed by SQL Server.  
     
   **VIEW**
3. **What is View? Use? Syntax of View?**A view is a virtual table made up of data from base tables and other views, but not stored separately.
   * Views simplify users perception of the database (can be used to present only the necessary information while hiding details in underlying relations)
   * Views improve data security preventing undesired accesses
   * Views facilite the provision of additional data independence
4. **Does the View occupy memory space?**No
5. **Can u drop a table if it has a view?**Views or tables participating in a view created with the SCHEMABINDING clause cannot be dropped. If the view is not created using SCHEMABINDING, then we can drop the table.
6. **Why doesn't SQL Server permit an ORDER BY clause in the definition of a view?**SQL Server excludes an ORDER BY clause from a view to comply with the ANSI SQL-92 standard. Because analyzing the rationale for this standard requires a discussion of the underlying structure of the structured query language (SQL) and the mathematics upon which it is based, we can't fully explain the restriction here. However, if you need to be able to specify an ORDER BY clause in a view, consider using the following workaround:  
   USE pubs  
   GO   
   CREATE VIEW AuthorsByName  
   AS  
   SELECT TOP 100 PERCENT \*  
   FROM authors   
   ORDER BY au\_lname, au\_fname  
   GO  
   The TOP construct, which Microsoft introduced in SQL Server 7.0, is most useful when you combine it with the ORDER BY clause. The only time that SQL Server supports an ORDER BY clause in a view is when it is used in conjunction with the TOP keyword. (Note that the TOP keyword is a SQL Server extension to the ANSI SQL-92 standard.)  
     
   **TRANSACTION**
7. **What is Transaction?**A transaction is a sequence of operations performed as a single logical unit of work. A logical unit of work must exhibit four properties, called the ACID (Atomicity, Consistency, Isolation, and Durability) properties, to qualify as a transaction:
   * **Atomicity -** A transaction must be an atomic unit of work; either all of its data modifications are performed or none of them is performed.
   * **Consistency -** When completed, a transaction must leave all data in a consistent state. In a relational database, all rules must be applied to the transaction's modifications to maintain all data integrity. All internal data structures, such as B-tree indexes or doubly-linked lists, must be correct at the end of the transaction.
   * **Isolation -** Modifications made by concurrent transactions must be isolated from the modifications made by any other concurrent transactions. A transaction either sees data in the state it was in before another concurrent transaction modified it, or it sees the data after the second transaction has completed, but it does not see an intermediate state. This is referred to as serializability because it results in the ability to reload the starting data and replay a series of transactions to end up with the data in the same state it was in after the original transactions were performed.
   * **Durability -** After a transaction has completed, its effects are permanently in place in the system. The modifications persist even in the event of a system failure.
8. **After one Begin Transaction a truncate statement and a RollBack statements are there. Will it be rollbacked? Since the truncate statement does not perform logged operation how does it RollBack?**It will rollback.  
   \*\*
9. Given a SQL like   
   Begin Tran  
      Select @@Rowcount  
   Begin Tran  
      Select @@Rowcount  
   Begin Tran  
      Select @@Rowcount  
   Commit Tran  
      Select @@Rowcount  
   RollBack  
      Select @@Rowcount  
   RollBack  
      Select @@Rowcount  
   What is the value of @@Rowcount at each stmt levels?  
   Ans : 0 – zero.  
   @@ROWCOUNT - Returns the number of rows affected by the last statement.  
   @@TRANCOUNT - Returns the number of active transactions for the current connection.  
   Each Begin Tran will add count, each commit will reduce count and ONE rollback will make it 0.  
     
   OTHER
10. **What are the constraints for Table Constraints define rules regarding the values allowed in columns and are the standard mechanism for enforcing integrity. SQL Server 2000 supports five classes of constraints.**NOT NULL  
    CHECK  
    UNIQUE  
    PRIMARY KEY  
    FOREIGN KEY
11. **There are 50 columns in a table. Write a query to get first 25 columns**Ans: Need to mention each column names.
12. **How to list all the tables in a particular database?**USE pubs  
    GO  
    sp\_help
13. **What are cursors? Explain different types of cursors. What are the disadvantages of cursors? How can you avoid cursors?**Cursors allow row-by-row processing of the result sets.  
    Types of cursors: Static, Dynamic, Forward-only, Keyset-driven.  
    Disadvantages of cursors: Each time you fetch a row from the cursor, it results in a network roundtrip. Cursors are also costly because they require more resources and temporary storage (results in more IO operations). Further, there are restrictions on the SELECT statements that can be used with some types of cursors.  
    How to avoid cursor:
    1. Most of the times, set based operations can be used instead of cursors. Here is an example: If you have to give a flat hike to your employees using the following criteria:  
       Salary between 30000 and 40000 -- 5000 hike  
       Salary between 40000 and 55000 -- 7000 hike  
       Salary between 55000 and 65000 -- 9000 hike  
       In this situation many developers tend to use a cursor, determine each employee's salary and update his salary according to the above formula. But the same can be achieved by multiple update statements or can be combined in a single UPDATE statement as shown below:  
       UPDATE tbl\_emp SET salary =  
       CASE WHEN salary BETWEEN 30000 AND 40000 THEN salary + 5000  
       WHEN salary BETWEEN 40000 AND 55000 THEN salary + 7000  
       WHEN salary BETWEEN 55000 AND 65000 THEN salary + 10000  
       END
    2. You need to call a stored procedure when a column in a particular row meets certain condition. You don't have to use cursors for this. This can be achieved using WHILE loop, as long as there is a unique key to identify each row. For examples of using WHILE loop for row by row processing, check out the 'My code library' section of my site or search for WHILE.
14. **What is Dynamic Cursor? Suppose, I have a dynamic cursor attached to table in a database.  I have another means by which I will modify the table.  What do you think will the values in the cursor be?**Dynamic cursors reflect all changes made to the rows in their result set when scrolling through the cursor. The data values, order, and membership of the rows in the result set can change on each fetch. All UPDATE, INSERT, and DELETE statements made by all users are visible through the cursor. Updates are visible immediately if they are made through the cursor using either an API function such as SQLSetPos or the Transact-SQL WHERE CURRENT OF clause. Updates made outside the cursor are not visible until they are committed, unless the cursor transaction isolation level is set to read uncommitted.
15. **What is DATEPART?**Returns an integer representing the specified datepart of the specified date.
16. **Difference between Delete and Truncate?**TRUNCATE TABLE is functionally identical to DELETE statement with no WHERE clause: both remove all rows in the table.  
    (1) But TRUNCATE TABLE is faster and uses fewer system and transaction log resources than DELETE. The DELETE statement removes rows one at a time and records an entry in the transaction log for each deleted row. TRUNCATE TABLE removes the data by deallocating the data pages used to store the table's data, and only the page deallocations are recorded in the transaction log.  
    (2) Because TRUNCATE TABLE is not logged, it cannot activate a trigger.  
    (3) The counter used by an identity for new rows is reset to the seed for the column. If you want to retain the identity counter, use DELETE instead.  
    Of course, TRUNCATE TABLE can be rolled back.
17. **Given a scenario where two operations, Delete Stmt and Truncate Stmt, where the Delete Statement was successful and the truncate stmt was failed. – Can u judge why?  
    \*\***
18. **What are global variables? Tell me some of them?**Transact-SQL global variables are a form of function and are now referred to as functions.  
    ABS - Returns the absolute, positive value of the given numeric expression.  
    SUM  
    AVG  
    AND
19. **What is DDL?**Data definition language (DDL) statements are SQL statements that support the definition or declaration of database objects (for example, CREATE TABLE, DROP TABLE, and ALTER TABLE).  
    You can use the ADO Command object to issue DDL statements. To differentiate DDL statements from a table or stored procedure name, set the CommandType property of the Command object to adCmdText. Because executing DDL queries with this method does not generate any recordsets, there is no need for a Recordset object.
20. **What is DML?**Data Manipulation Language (DML), which is used to select, insert, update, and delete data in the objects defined using DDL
21. **What are keys in RDBMS? What is a primary key/ foreign key?**There are two kinds of keys.  
    A primary key is a set of columns from a table that are guaranteed to have unique values for each row of that table.  
    Foreign keys are attributes of one table that have matching values in a **primary key** in another table, allowing for relationships between tables.
22. **What is the difference between Primary Key and Unique Key?**Both primary key and unique key enforce uniqueness of the column on which they are defined. But by default primary key creates a clustered index on the column, where are unique creates a nonclustered index by default. Another major difference is that, primary key doesn't allow NULLs, but unique key allows one NULL only.
23. **Define candidate key, alternate key, composite key?**  
    A candidate key is one that can identify each row of a table uniquely. Generally a candidate key becomes the primary key of the table. If the table has more than one candidate key, one of them will become the primary key, and the rest are called alternate keys.   
    A key formed by combining at least two or more columns is called composite key.
24. **What is the Referential Integrity?**Referential integrity refers to the consistency that must be maintained between primary and foreign keys, i.e. every foreign key value must have a corresponding primary key value.
25. **What are defaults? Is there a column to which a default can't be bound?**  
    A default is a value that will be used by a column, if no value is supplied to that column while inserting data. IDENTITY columns and timestamp columns can't have defaults bound to them.
26. **What is Query optimization? How is tuning a performance of query done?**
27. **What is the use of trace utility?  
    \*\***
28. **What is the use of shell commands? xp\_cmdshell**Executes a given command string as an operating-system command shell and returns any output as rows of text. Grants nonadministrative users permissions to execute **xp\_cmdshell**.
29. **What is use of shrink database?**Microsoft® SQL Server 2000 allows each file within a database to be shrunk to remove unused pages. Both data and transaction log files can be shrunk.
30. **If the performance of the query suddenly decreased where you will check?**
31. **What is a pass-through query?**Microsoft® SQL Server 2000 sends pass-through queries as un-interpreted query strings to an OLE DB data source. The query must be in a syntax the OLE DB data source will accept. A Transact-SQL statement uses the results from a pass-through query as though it is a regular table reference.  
    This example uses a pass-through query to retrieve a result set from a Microsoft Access version of the Northwind sample database.  
    SELECT \*  
    FROM OpenRowset('Microsoft.Jet.OLEDB.4.0',   
    'c:\northwind.mdb';'admin'; '',   
    'SELECT CustomerID, CompanyName  
    FROM Customers  
    WHERE Region = ''WA'' ')
32. **How do you differentiate Local and Global Temporary table?**You can create local and global temporary tables. Local temporary tables are visible only in the current session; global temporary tables are visible to all sessions. Prefix local temporary table names with single number sign (#*table\_name*), and prefix global temporary table names with a double number sign (##*table\_name*). SQL statements reference the temporary table using the value specified for *table\_name* in the CREATE TABLE statement:  
    CREATE TABLE #MyTempTable (cola INT PRIMARY KEY)  
    INSERT INTO #MyTempTable VALUES (1)
33. **How the Exists keyword works in SQL Server?**USE pubs  
    SELECT au\_lname, au\_fname  
    FROM authors  
    WHERE exists  
       (SELECT \*  
       FROM publishers  
       WHERE authors.city = publishers.city)  
    When a subquery is introduced with the keyword EXISTS, it functions as an existence test. The WHERE clause of the outer query tests for the existence of rows returned by the subquery. The subquery does not actually produce any data; it returns a value of TRUE or FALSE.
34. **ANY?**USE pubs  
    SELECT au\_lname, au\_fname  
    FROM authors  
    WHERE city = ANY  
    (SELECT city  
    FROM publishers)
35. **to select date part only**SELECT CONVERT(char(10),GetDate(),101)  
    --to select time part only  
    SELECT right(GetDate(),7)
36. **How can I send a message to user from the SQL Server?**You can use the xp\_cmdshell extended stored procedure to run net send command. This is the example to send the 'Hello' message to JOHN:  
    EXEC master..xp\_cmdshell "net send JOHN 'Hello'"  
    To get net send message on the Windows 9x machines, you should run the WinPopup utility. You can place WinPopup in the Startup group under Program Files.
37. **What is normalization? Explain different levels of normalization? Explain Third normalization form with an example?**The process of refining tables, keys, columns, and relationships to create an efficient database is called *normalization*. This should eliminates unnecessary duplication and provides a rapid search path to all necessary information.  
    Some of the benefits of normalization are:
    * Data integrity (because there is no redundant, neglected data)
    * Optimized queries (because normalized tables produce rapid, efficient joins)
    * Faster index creation and sorting (because the tables have fewer columns)
    * Faster UPDATE performance (because there are fewer indexes per table)
    * Improved concurrency resolution (because table locks will affect less data)
    * Eliminate redundancy

There are a few rules for database normalization. Each rule is called a "normal form." If the first rule is observed, the database is said to be in "first normal form." If the first three rules are observed, the database is considered to be in "third normal form." Although other levels of normalization are possible, third normal form is considered the highest level necessary for most applications.

* 1. **First Normal Form (1NF)**
     + Eliminate repeating groups in individual tables
     + Create a separate table for each set of related data.
     + Identify each set of related data with a primary key.

Do not use multiple fields in a single table to store similar data.   
Example

|  | **Subordinate1** | **Subordinate2** | **Subordinate3** | **Subordinate4** |
| --- | --- | --- | --- | --- |
| Bob | Jim | Mary | Beth |  |
| Mary | Mike | Jason | Carol | Mark |
| Jim | Alan |  |  |  |

Eliminate duplicative columns from the same table.  Clearly, the Subordinate1-Subordinate4 columns are duplicative. What happens when we need to add or remove a subordinate?

|  | **Subordinates** |
| --- | --- |
| Bob | Jim, Mary, Beth |
| Mary | Mike, Jason, Carol, Mark |
| Jim | Alan |

This solution is closer, but it also falls short of the mark. The subordinates column is still duplicative and non-atomic. What happens when we need to add or remove a subordinate? We need to read and write the entire contents of the table. That’s not a big deal in this situation, but what if one manager had one hundred employees? Also, it complicates the process of selecting data from the database in future queries.   
Solution:

|  | **Subordinate** |
| --- | --- |
| Bob | Jim |
| Bob | Mary |
| Bob | Beth |
| Mary | Mike |
| Mary | Jason |
| Mary | Carol |
| Mary | Mark |
| Jim | Alan |

* 1. **Second Normal Form (2NF)**
     + Create separate tables for sets of values that apply to multiple records.
     + Relate these tables with a foreign key.

Records should not depend on anything other than a table's primary key (a compound key, if necessary).   
For example, consider a customer's address in an accounting system. The address is needed by the Customers table, but also by the Orders, Shipping, Invoices, Accounts Receivable, and Collections tables. Instead of storing the customer's address as a separate entry in each of these tables, store it in one place, either in the Customers table or in a separate Addresses table.

* 1. **Third Normal Form (3NF)**
     + Eliminate fields that do not depend on the key.

Values in a record that are not part of that record's key do not belong in the table. In general, any time the contents of a group of fields may apply to more than a single record in the table, consider placing those fields in a separate table.  
For example, in an Employee Recruitment table, a candidate's university name and address may be included. But you need a complete list of universities for group mailings. If university information is stored in the Candidates table, there is no way to list universities with no current candidates. Create a separate Universities table and link it to the Candidates table with a university code key.  
Another Example :

| **MemberId** | **Name** | **Company** | **CompanyLoc** |
| --- | --- | --- | --- |
| 1 | John Smith | ABC | Alabama |
| 2 | Dave Jones | MCI | Florida |

The Member table satisfies first normal form - it contains no repeating groups. It satisfies second normal form - since it doesn't have a multivalued key. But the key is MemberID, and the company name and location describe only a company, not a member. To achieve third normal form, they must be moved into a separate table. Since they describe a company, CompanyCode becomes the key of the new "Company" table.

The motivation for this is the same for second normal form: we want to avoid update and delete anomalies. For example, suppose no members from the IBM were currently stored in the database. With the previous design, there would be no record of its existence, even though 20 past members were from IBM!  
Member Table

| **MemberId** | **Name** | **CID** |
| --- | --- | --- |
| 1 | John Smith | 1 |
| 2 | Dave Jones | 2 |

Company Table

| **CId** | **Name** | **Location** |
| --- | --- | --- |
| 1 | ABC | Alabama |
| 2 | MCI | Florida |

* 1. **Boyce-Codd Normal Form (BCNF)**A relation is in Boyce/Codd normal form if and only if the only determinants are candidate key. Its a different version of 3NF, indeed, was meant to replace it. [A determinant is any attribute on which some other attribute is (fully) functionally dependent.]
  2. **4th Normal Form (4NF)**A table is in 4NF if it is in BCNF and if it has no multi-valued dependencies. This applies primarily to key-only associative tables, and appears as a ternary relationship, but has incorrectly merged 2 distinct, independent relationships.  
     Eg: This could be any 2 M:M relationships from a single entity. For instance, a member could know many software tools, and a software tool may be used by many members. Also, a member could have recommended many books, and a book could be recommended by many members.

| Software |  | member |  | Book |
| --- | --- | --- | --- | --- |

* 1. The correct solution, to cause the model to be in 4th normal form, is to ensure that all M:M relationships are resolved independently if they are indeed independent.

| Software |  | membersoftware |  | member |  | memberBook |  | book |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

* 1. **5th Normal Form (5NF)(PJNF)**A table is in 5NF, also called "Projection-Join Normal Form", if it is in 4NF and if every join dependency in the table is a consequence of the candidate keys of the table.
  2. **Domain/key normal form (DKNF)**. A key uniquely identifies each row in a table. A domain is the set of permissible values for an attribute. By enforcing key and domain restrictions, the database is assured of being freed from modification anomalies. DKNF is the normalization level that most designers aim to achieve.

\*\*  
Remember, these normalization guidelines are cumulative.  For a database to be in 2NF, it must first fulfill all the criteria of a 1NF database.

1. **If a database is normalized by 3 NF then how many number of tables it should contain in minimum? How many minimum if 2NF and 1 NF?**
2. **What is denormalization and when would you go for it?**As the name indicates, denormalization is the reverse process of normalization. It's the controlled introduction of redundancy in to the database design. It helps improve the query performance as the number of joins could be reduced.
3. **How can I randomly sort query results?**To randomly order rows, or to return *x* number of randomly chosen rows, you can use the RAND function inside the SELECT statement. But the RAND function is resolved only once for the entire query, so every row will get same value. You can use an ORDER BY clause to sort the rows by the result from the NEWID function, as the following code shows:  
   SELECT \*  
   FROM Northwind..Orders   
   ORDER BY NEWID()
4. **sp\_who**Provides information about current Microsoft® SQL Server™ users and processes. The information returned can be filtered to return only those processes that are not idle.
5. **Have you worked on Dynamic SQL? How will You handled “ (Double Quotes) in Dynamic SQL?**
6. **How to find dependents of a table?**Verify dependencies with **sp\_depends** before dropping an object
7. **What is the difference between a CONSTRAINT AND RULE?**Rules are a backward-compatibility feature that perform some of the same functions as CHECK constraints. CHECK constraints are the preferred, standard way to restrict the values in a column. CHECK constraints are also more concise than rules; there can only be one rule applied to a column, but multiple CHECK constraints can be applied. CHECK constraints are specified as part of the CREATE TABLE statement, while rules are created as separate objects and then bound to the column.
8. **How to call a COM dll from SQL Server 2000?**sp\_OACreate - Creates an instance of the OLE object on an instance of Microsoft® SQL Server  
   **Syntax  
   sp\_OACreate** *progid***,** | *clsid***,**   
       *objecttoken* **OUTPUT**   
       [ **,** *context* ]

*context -* Specifies the execution context in which the newly created OLE object runs. If specified, this value must be one of the following: **1** = In-process (.dll) OLE server only  
**4** = Local (.exe) OLE server only  
**5** = Both in-process and local OLE server allowed

##### Examples

###### A. Use Prog ID - This example creates a SQL-DMO SQLServer object by using its ProgID.

DECLARE @object int

DECLARE @hr int

DECLARE @src varchar(255), @desc varchar(255)

EXEC @hr = sp\_OACreate 'SQLDMO.SQLServer', @object OUT

IF @hr <> 0

BEGIN

EXEC sp\_OAGetErrorInfo @object, @src OUT, @desc OUT

SELECT hr=convert(varbinary(4),@hr), Source=@src, Description=@desc

RETURN

END

B. Use CLSID - This example creates a SQL-DMO SQLServer object by using its CLSID.

DECLARE @object int

DECLARE @hr int

DECLARE @src varchar(255), @desc varchar(255)

EXEC @hr = sp\_OACreate '{00026BA1-0000-0000-C000-000000000046}',

@object OUT

IF @hr <> 0

BEGIN

EXEC sp\_OAGetErrorInfo @object, @src OUT, @desc OUT

SELECT hr=convert(varbinary(4),@hr), Source=@src, Description=@desc

RETURN

END

1. **Difference between sysusers and syslogins?**sysusers - Contains one row for each Microsoft® Windows user, Windows group, Microsoft SQL Server™ user, or SQL Server role in the database.  
   syslogins - Contains one row for each login account.
2. **What is the row size in SQL Server 2000?**8060 bytes.
3. **How will you find structure of table, all** tables/views **in one db, all dbs?**//structure of tablesp\_helpdb tbl\_emp  
    **//**list of all databasessp\_helpdb  
   OR  
   SELECT \* FROM master.dbo.sysdatabases  
     
   //details about database pubs. .mdf, .ldf file locations, size of database  
   sp\_helpdb pubs  
     
   //lists all tables under current database  
   sp\_tables  
   OR  
   SELECT \* FROM information\_schema.tables WHERE (table\_type = 'base table')  
   OR  
   SELECT \* FROM sysobjects WHERE type = 'U' //faster
4. **B-tree indexes or doubly-linked lists?**
5. **What is the system function to get the current user's user id?**USER\_ID(). Also check out other system functions like USER\_NAME(), SYSTEM\_USER, SESSION\_USER, CURRENT\_USER, USER, SUSER\_SID(), HOST\_NAME().
6. **What are the series of steps that happen on execution of a query in a Query Analyzer?**1) Syntax checking 2) Parsing 3) Execution plan
7. **Which event (Check constraints, Foreign Key, Rule, trigger, Primary key check) will be performed last for integrity check?**Identity Insert Check  
   Nullability constraint  
   Data type check  
   Instead of trigger  
   Primary key  
   Check constraint  
   Foreign key  
   DML Execution (update statements)  
   After Trigger **\*\***
8. **How will you show many to many relation in sql?**Create 3rd table with 2 columns which having one to many relation to these tables.
9. **When a query is sent to the database and an index is not being used, what type of execution is taking place?**A table scan.
10. **What is #, ##, @, @@ means?**@@ - System variables  
    @ - user defined variables
11. **What is the difference between a Local temporary table and a Global temporary table? How is each one denoted?**Local temporary table will be accessible to only current user session, its name will be preceded with a single hash (#mytable)  
    Global temporary table will be accessible to all users, & it will be dropped only after ending of all active connections, its name will be preceded with double hash (##mytable)
12. **What is covered queries in SQL Server?**
13. **What is HASH JOIN, MERGE JOIN?**  
    **TOOLS**
14. **Have you ever used DBCC command? Give an example for it.**The Transact-SQL programming language provides DBCC statements that act as Database Console Commands for Microsoft® SQL Serve 2000. These statements check the physical and logical consistency of a database. Many DBCC statements can fix detected problems. Database Console Command statements are grouped into these categories.

| **Statement category** | **Perform** |
| --- | --- |
| Maintenance statements | Maintenance tasks on a database, index, or filegroup. |
| Miscellaneous statements | Miscellaneous tasks such as enabling row-level locking or removing a dynamic-link library (DLL) from memory. |
| Status statements | Status checks. |
| Validation statements | Validation operations on a database, table, index, catalog, filegroup, system tables, or allocation of database pages. |
| DBCC CHECKDB, DBCC CHECKTABLE, DBCC CHECKCATALOG, DBCC CHECKALLOC, DBCC SHOWCONTIG, DBCC SHRINKDATABASE, DBCC SHRINKFILE etc. | |

1. **How do you use DBCC statements to monitor various aspects of a SQL server installation?  
   \*\***
2. **What is the output of DBCC Showcontig statement?**Displays fragmentation information for the data and indexes of the specified table.
3. **How do I reset the identity column?**You can use the DBCC CHECKIDENT statement, if you want to reset or reseed the identity column. For example, if you need to force the current identity value in the jobs table to a value of 100, you can use the following:  
   USE pubs  
   GO  
   DBCC CHECKIDENT (jobs, RESEED, 100)  
   GO
4. **About SQL Command line executables**

| **Utilities** |
| --- |
| bcp console isql sqlagent sqldiag sqlmaint sqlservr vswitch |
| dtsrun dtswiz isqlw itwiz odbccmpt osql rebuildm sqlftwiz |
| distrib logread replmerg snapshot |
| scm |
| regxmlss |

1. **What is DTC?**The Microsoft Distributed Transaction Coordinator (MS DTC) is a transaction manager that allows client applications to include several different sources of data in one transaction. MS DTC coordinates committing the distributed transaction across all the servers enlisted in the transaction.
2. **What is DTS? Any drawbacks in using DTS?**Microsoft® SQL Server™ 2000 Data Transformation Services (DTS) is a set of graphical tools and programmable objects that lets you extract, transform, and consolidate data from disparate sources into single or multiple destinations.
3. **What is BCP?**The **bcp** utility copies data between an instance of Microsoft® SQL Server™ 2000 and a data file in a user-specified format.  
   C:\Documents and Settings\sthomas>bcp  
   usage: bcp {dbtable | query} {in | out | queryout | format} datafile  
   [-m maxerrors] [-f formatfile] [-e errfile]  
   [-F firstrow] [-L lastrow] [-b batchsize]  
   [-n native type] [-c character type] [-w wide character type]  
   [-N keep non-text native] [-V file format version] [-q quoted identifier]  
   [-C code page specifier] [-t field terminator] [-r row terminator]  
   [-i inputfile] [-o outfile] [-a packetsize]  
   [-S server name] [-U username] [-P password]  
   [-T trusted connection] [-v version] [-R regional enable]  
   [-k keep null values] [-E keep identity values]  
   [-h "load hints"]
4. **How can I create a plain-text flat file from SQL Server as input to another application?**One of the purposes of Extensible Markup Language (XML) is to solve challenges like this, but until all applications become XML-enabled, consider using our faithful standby, the bulk copy program (bcp) utility. This utility can do more than just dump a table; bcp also can take its input from a view instead of from a table. After you specify a view as the input source, you can limit the output to a subset of columns or to a subset of rows by selecting appropriate filtering (WHERE and HAVING) clauses.  
   More important, by using a view, you can export data from multiple joined tables. The only thing you cannot do is specify the sequence in which the rows are written to the flat file, because a view does not let you include an ORDER BY clause in it unless you also use the TOP keyword.  
   If you want to generate the data in a particular sequence or if you cannot predict the content of the data you want to export, be aware that in addition to a view, bcp also supports using an actual query. The only "gotcha" about using a query instead of a table or view is that you must specify **queryout** in place of **out** in the bcp command line.  
   For example, you can use bcp to generate from the **pubs** database a list of authors who reside in California by writing the following code:  
   bcp "SELECT \* FROM pubs..authors WHERE state = 'CA'" queryout c:\CAauthors.txt -c -T -S
5. **What are the different ways of moving data/databases between servers and databases in SQL Server?**There are lots of options available, you have to choose your option depending upon your requirements. Some of the options you have are: BACKUP/RESTORE, detaching and attaching databases, replication, DTS, BCP, logshipping, INSERT...SELECT, SELECT...INTO, creating INSERT scripts to generate data.
6. **How will I export database?**Through DTS - Import/Export wizard  
   Backup - through Complete/Differential/Transaction Log
7. **How to export database at a particular time, every week?**Backup - Schedule  
   DTS - Schedule  
   Jobs - create a new job
8. **How do you load large data to the SQL server database?**bcp
9. **How do you transfer data from text file to database (other than DTS)?**bcp
10. **What is OSQL and ISQL utility?**The **osql** utility allows you to enter Transact-SQL statements, system procedures, and script files. This utility uses ODBC to communicate with the server.   
    The **isql** utility allows you to enter Transact-SQL statements, system procedures, and script files; and uses DB-Library to communicate with Microsoft® SQL Server™ 2000.  
    All DB-Library applications, such as isql, work as SQL Server 6.5–level clients when connected to SQL Server 2000. They do not support some SQL Server 2000 features.  
    The osql utility is based on ODBC and does support all SQL Server 2000 features. Use osql to run scripts that isql cannot run.
11. **What Tool you have used for checking Query Optimization? What is the use of profiler in sql server?  What is the first thing u look at in a 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 a production environment to see which stored procedures is hampering performance by executing too slowly.   
    Use SQL Profiler to:
    * Monitor the performance of an instance of SQL Server.
    * Debug Transact-SQL statements and stored procedures.
    * Identify slow-executing queries.
    * Test SQL statements and stored procedures in the development phase of a project by single-stepping through statements to confirm that the code works as expected.
    * Troubleshoot problems in SQL Server by capturing events on a production system and replaying them on a test system. This is useful for testing or debugging purposes and allows users to continue using the production system without interference.

Audit and review activity that occurred on an instance of SQL Server. This allows a security administrator to review any of the auditing events, including the success and failure of a login attempt and the success and failure of permissions in accessing statements and objects.  
  
**Permissions**

1. **A user is a member of Public role and Sales role. Public role has the permission to select on all the table, and Sales role, which doesn’t have a select permission on some of the tables. Will that user be able to select from all tables?  
   \*\***
2. **If a user does not have permission on a table, but he has permission to a view created on it, will he be able to view the data in table?**Yes.
3. **Describe Application Role and explain a scenario when you will use it?  
   \*\***
4. **After removing a table from database, what other related objects have to be dropped explicitly?**(view, SP)
5. **You have a SP names YourSP and have the a Select Stmt inside the SP. You also have a user named YourUser. What permissions you will give him for accessing the SP.  
   \*\***
6. **Different Authentication modes in Sql server? If a user is logged under windows authentication mode, how to find his userid?**There are Three Different authentication modes in sqlserver.
   1. Windows Authentication Mode
   2. SqlServer Authentication Mode
   3. Mixed Authentication Mode

“system\_user” system function in sqlserver to fetch the logged on user name.

1. **Give the connection strings from front-end for both type logins(windows,sqlserver)?**This are specifically for sqlserver not for any other RDBMS  
   Data Source=MySQLServer;Initial Catalog=NORTHWIND;Integrated Security=SSPI (windows)  
   Data Source=MySQLServer;Initial Catalog=NORTHWIND;Uid=” ”;Pwd=” ”(sqlserver)
2. **What are three SQL keywords used to change or set someone’s permissions?**  
   Grant, Deny and Revoke**Administration**
3. **Explain the architecture of SQL Server?  
   \*\***
4. **Different types of Backups?**
   * A full database backup is a full copy of the database.
   * A transaction log backup copies only the transaction log.
   * A differential backup copies only the database pages modified after the last full database backup.
   * A file or filegroup restore allows the recovery of just the portion of a database that was on the failed disk.
5. **What are ‘jobs’ in SQL Server? How do we create one? What is tasks?**Using SQL Server Agent jobs, you can automate administrative tasks and run them on a recurring basis.  
   \*\*
6. **What is database replication? What are the different types of replication you can set up in SQL Server? How are they used? What is snapshot replication how is it different from Transactional replication?**  
   Replication is the process of copying/moving data between databases on the same or different servers. SQL Server supports the following types of replication scenarios:
   1. Snapshot replication - It distributes data exactly as it appears at a specific moment in time and doesn’t monitor for updates. It can be used when data changes are infrequent. It is often used for browsing data such as price lists, online catalog, or data for decision support where the current data is not required and data is used as read only.
   2. Transactional replication (with immediate updating subscribers, with queued updating subscribers) - With this an initial snapshot of data is applied, and whenever data modifications are made at the publisher, the individual transactions are captured and propagated to the subscribers.
   3. Merge replication - It is the process of distributing the data between publisher and subscriber, it allows the publisher and subscriber to update the data while connected or disconnected, and then merging the updates between the sites when they are connected.
7. **How can u look at what are the process running on SQL server? How can you kill a process in SQL server?**
   * Expand a server group, and then expand a server.
   * Expand **Management**, and then expand **Current Activity**.
   * Click **Process Info**. The current server activity is displayed in the details pane.

In the details pane, right-click a Process ID, and then click **Kill Process**.

1. **What is RAID and what are different types of RAID configurations?**RAID stands for Redundant Array of Inexpensive Disks, used to provide fault tolerance to database servers. There are six RAID levels 0 through 5 offering different levels of performance, fault tolerance.
2. Some of the tools/ways that help you troubleshooting performance problems are: SET SHOWPLAN\_ALL ON, SET SHOWPLAN\_TEXT ON, SET STATISTICS IO ON, SQL Server Profiler, Windows NT /2000 Performance monitor, Graphical execution plan in Query Analyzer.
3. **How to determine the service pack currently installed on SQL Server?**The global variable @@Version stores the build number of the sqlservr.exe, which is used to determine the service pack installed.  
   eg: Microsoft SQL Server 2000 - 8.00.760 (Intel X86) Dec 17 2002 14:22:05 Copyright (c) 1988-2003 Microsoft Corporation Enterprise Edition on Windows NT 5.0 (Build 2195: Service Pack 3)
4. **What is the purpose of using COLLATE in a query?**  
   The term, collation, refers to a set of rules that determine how data is sorted and compared. In Microsoft® SQL Server 2000, it is not required to separately specify code page and sort order for character data, and the collation used for Unicode data. Instead, specify the collation name and sorting rules to use. Character data is sorted using rules that define the correct character sequence, with options for specifying case-sensitivity, accent marks, kana character types, and character width. Microsoft SQL Server 2000 collations include these groupings:
   * Windows collations - Windows collations define rules for storing character data based on the rules defined for an associated Windows locale. The base Windows collation rules specify which alphabet or language is used when dictionary sorting is applied, as well as the code page used to store non-Unicode character data. For Windows collations, the **nchar**, **nvarchar**, and **ntext** data types have the same sorting behavior as **char**, **varchar**, and **text** data types
   * SQL collations - SQL collations are provided for compatibility with sort orders in earlier versions of Microsoft SQL Server.

**Sort Order**  
Binary is the fastest sorting order, and is case-sensitive. If **Binary** is selected, the **Case-sensitive**, **Accent-sensitive**, **Kana-sensitive**, and **Width-sensitive** options are not available.

| **Sort order** | **Description** |
| --- | --- |
| **Binary** | Sorts and compares data in Microsoft® SQL Server™ tables based on the bit patterns defined for each character. Binary sort order is case-sensitive, that is lowercase precedes uppercase, and accent-sensitive. This is the fastest sorting order.  If this option is not selected, SQL Server follows sorting and comparison rules as defined in dictionaries for the associated language or alphabet. |
| **Case-sensitive** | Specifies that SQL Server distinguish between uppercase and lowercase letters.  If not selected, SQL Server considers the uppercase and lowercase versions of letters to be equal. SQL Server does not define whether lowercase letters sort lower or higher in relation to uppercase letters when Case-sensitive is not selected. |
| **Accent-sensitive** | Specifies that SQL Server distinguish between accented and unaccented characters. For example, 'a' is not equal to 'á'.  If not selected, SQL Server considers the accented and unaccented versions of letters to be equal. |
| **Kana-sensitive** | Specifies that SQL Server distinguish between the two types of Japanese kana characters: Hiragana and Katakana.  If not selected, SQL Server considers Hiragana and Katakana characters to be equal. |
| **Width-sensitive** | Specifies that SQL Server distinguish between a single-byte character (half-width) and the same character when represented as a double-byte character (full-width).  If not selected, SQL Server considers the single-byte and double-byte representation of the same character to be equal. |

Windows collation options:

* + Use **Latin1\_General** for the U.S. English character set (code page 1252).
  + Use **Modern\_Spanish** for all variations of Spanish, which also use the same character set as U.S. English (code page 1252).
  + Use **Arabic** for all variations of Arabic, which use the Arabic character set (code page 1256).
  + Use **Japanese\_Unicode** for the Unicode version of Japanese (code page 932), which has a different sort order from **Japanese**, but the same code page (932).

1. **What is the STUFF Function and how does it differ from the REPLACE function?**STUFF - Deletes a specified length of characters and inserts another set of characters at a specified starting point.   
   SELECT STUFF('abcdef', 2, 3, 'ijklmn')  
   GO  
   Here is the result set:  
   ---------   
   aijklmnef

REPLACE - Replaces all occurrences of the second given string expression in the first string expression with a third expression.  
SELECT REPLACE('abcdefghicde','cde','xxx')  
GO   
Here is the result set:   
------------  
abxxxfghixxx

1. **What does it mean to have quoted\_identifier on? What are the implications of having it off?**When SET QUOTED\_IDENTIFIER is OFF (default), literal strings in expressions can be delimited by single or double quotation marks.  
   When SET QUOTED\_IDENTIFIER is ON, all strings delimited by double quotation marks are interpreted as object identifiers. Therefore, quoted identifiers do not have to follow the Transact-SQL rules for identifiers.   
   SET QUOTED\_IDENTIFIER must be ON when creating or manipulating indexes on computed columns or indexed views. If SET QUOTED\_IDENTIFIER is OFF, CREATE, UPDATE, INSERT, and DELETE statements on tables with indexes on computed columns or indexed views will fail.   
   The SQL Server ODBC driver and Microsoft OLE DB Provider for SQL Server automatically set QUOTED\_IDENTIFIER to ON when connecting.  
   When a stored procedure is created, the SET QUOTED\_IDENTIFIER and SET ANSI\_NULLS settings are captured and used for subsequent invocations of that stored procedure. When executed inside a stored procedure, the setting of SET QUOTED\_IDENTIFIER is not changed.  
   SET QUOTED\_IDENTIFIER OFF  
   GO  
   -- Attempt to create a table with a reserved keyword as a name  
   -- should fail.  
   CREATE TABLE "select" ("identity" int IDENTITY, "order" int)  
   GO  
     
   SET QUOTED\_IDENTIFIER ON  
   GO  
   -- Will succeed.  
   CREATE TABLE "select" ("identity" int IDENTITY, "order" int)  
   GO
2. **What is the purpose of UPDATE STATISTICS?**Updates information about the distribution of key values for one or more statistics groups (collections) in the specified table or indexed view.
3. **Fundamentals of Data warehousing & olap?**
4. **What do u mean by OLAP server? What is the difference between OLAP and OLTP?**
5. **What is a tuple?**A **tuple** is an instance of data within a relational database.
6. **Services and user Accounts maintenance**
7. **sp\_configure commands?**Displays or changes global configuration settings for the current server.
8. **What is the basic functions for master, msdb, tempdb databases?**  
   Microsoft® SQL Server 2000 systems have four system databases:
   * **master** - The **master** database records all of the system level information for a SQL Server system. It records all login accounts and all system configuration settings. **master** is the database that records the existence of all other databases, including the location of the database files.
   * **tempdb** - **tempdb** holds all temporary tables and temporary stored procedures. It also fills any other temporary storage needs such as work tables generated by SQL Server. **tempdb** is re-created every time SQL Server is started so the system starts with a clean copy of the database.   
     By default, **tempdb** autogrows as needed while SQL Server is running. If the size defined for **tempdb** is small, part of your system processing load may be taken up with autogrowing **tempdb** to the size needed to support your workload each time to restart SQL Server. You can avoid this overhead by using ALTER DATABASE to increase the size of **tempdb**.
   * **model** - The **model** database is used as the template for all databases created on a system. When a CREATE DATABASE statement is issued, the first part of the database is created by copying in the contents of the **model** database, then the remainder of the new database is filled with empty pages. Because **tempdb** is created every time SQL Server is started, the **model** database must always exist on a SQL Server system.
   * **msdb** - The **msdb** database is used by SQL Server Agent for scheduling alerts and jobs, and recording operators.
9. **What are sequence diagrams? What you will get out of this sequence diagrams?**Sequence diagrams document the interactions between classes to achieve a result, such as a use case. Because UML is designed for object-oriented programming, these communications between classes are known as messages. The sequence diagram lists objects horizontally, and time vertically, and models these messages over time.
10. **What are the new features of SQL 2000 than SQL 7? What are the new datatypes in sql?**
    * XML Support - The relational database engine can return data as Extensible Markup Language (XML) documents. Additionally, XML can also be used to insert, update, and delete values in the database. (for xml raw - to retrieve output as xml type)
    * User-Defined Functions - The programmability of Transact-SQL can be extended by creating your own Transact-SQL functions. A user-defined function can return either a scalar value or a table.
    * Indexed Views - Indexed views can significantly improve the performance of an application where queries frequently perform certain joins or aggregations. An indexed view allows indexes to be created on views, where the result set of the view is stored and indexed in the database.
    * New Data Types - SQL Server 2000 introduces three new data types. **bigint** is an 8-byte integer type. **sql\_variant** is a type that allows the storage of data values of different data types. **table** is a type that allows applications to store results temporarily for later use. It is supported for variables, and as the return type for user-defined functions.
    * INSTEAD OF and AFTER Triggers - INSTEAD OF triggers are executed instead of the triggering action (for example, INSERT, UPDATE, DELETE). They can also be defined on views, in which case they greatly extend the types of updates a view can support. AFTER triggers fire after the triggering action. SQL Server 2000 introduces the ability to specify which AFTER triggers fire first and last.
    * Multiple Instances of SQL Server - SQL Server 2000 supports running multiple instances of the relational database engine on the same computer. Each computer can run one instance of the relational database engine from SQL Server version 6.5 or 7.0, along with one or more instances of the database engine from SQL Server 2000. Each instance has its own set of system and user databases.
    * Index Enhancements - You can now create indexes on computed columns. You can specify whether indexes are built in ascending or descending order, and if the database engine should use parallel scanning and sorting during index creation.
11. **How do we open SQL Server in single user mode?**We can accomplish this in any of the three ways given below :-
    1. From Command Prompt :-  
       sqlservr -m
    2. From Startup Options :-  
       Go to SQL Server Properties by right-clicking on the Server name in the Enterprise manager.   
       Under the 'General' tab, click on 'Startup Parameters'.   
       Enter a value of -m in the Parameter.
    3. From Registry :-  
       Go to HKEY\_LOCAL\_MACHINE\Software\Microsoft\MSSQLServer\MSSQLServer\Parameters.   
       Add new string value.   
       Specify the 'Name' as SQLArg(n) & 'Data' as -m.   
       Where n is the argument number in the list of arguments.
12. **Difference between clustering and NLB (Network Load Balancing)?**\*\*
13. **Explain Active/Active and Active/Passive cluster configurations?  
    \*\***
14. **What is Log Shipping?**In Microsoft® SQL Server™ 2000 Enterprise Edition, you can use log shipping to feed transaction logs from one database to another on a constant basis. Continually backing up the transaction logs from a source database and then copying and restoring the logs to a destination database keeps the destination database synchronized with the source database. This allows you to have a backup server and also provides a way to offload query processing from the main computer (the source server) to read-only destination servers.
15. **What are the main steps you take care for enhancing SQL Server performance?  
    \*\***
16. **You have to check whether any users are connected to sql server database and if any user is connected to database, you have to disconnect the user(s) and run a process in a job. How do you do the above in a job?**\*\*  
      
    **XML**
17. **How can I convert data in a Microsoft Access table into XML format?**The following applications can help you convert Access data into XML format: Access 2002, ADO 2.5, and SQLXML. Access 2002 (part of Microsoft Office XP) enables you to query or save a table in XML format. You might be able to automate this process. ADO 2.5 and later enables you to open the data into a recordset, then persist the recordset in XML format, as the following code shows:  
    rs.Save "c:\rs.xml", adPersistXML  
    You can use linked servers to add the Access database to your SQL Server 2000 database so you can run queries from within SQL Server to retrieve data. Then, through HTTP, you can use the SQLXML technology to extract the Access data in the XML format you want. **NEW**
18. **@@IDENTITY ?**Ans: Returns the last-inserted identity value.
19. **If a job is fail in sql server, how do find what went wrong?**
20. **Have you used Error handling in DTS?**