|  |
| --- |
|  |  |
|  |  |

**What is SQL?**

**SQL** (**Structured Query Language**) is a database computer language designed for managing data in relational database management systems (RDBMS), and originally based upon Relational Algebra. Its scope includes data query and update, schema creation and modification, and data access control.

**What is database?**

A database is a logically coherent collection of data with some inherent meaning, representing some aspect of real world and which is designed, built and populated with data for a specific purpose.

**What is DBMS?**

It is a collection of programs that enables user to create and maintain a database. In other words it is general­purpose software that provides the users with the processes of defining, constructing and manipulating the database for various applications.

The database and DBMS software together is called as Database system.

Advantages of DBMS

The followings are some advantage of DBMS.

1.Redundancy is controlled.

2.Unauthorised access is restricted.

3.Providing multiple user interfaces.

4.Enforcing integrity constraints.

5.Providing backup and recovery.

**What is the use of CASCADE CONSTRAINTS?**

When this clause is used with the DROP command, a parent table can be dropped even when a child table exists.

We can’t able to delete a record from the parent table if there is some corresponding record exists in the child table.   
For this we are using cascading constraint. with the help of cascading constraint we can able to delete a record from the parent table as well as all the corresponding records from the child table, even we can set some null values.

**Why does the following command give a compilation error?  
DROP TABLE &TABLE\_NAME;**

Variable names should start with an alphabet. Here the table name starts with an '&' symbol. Any table name must start with alphabet. Table names starting with special characters are considered to be invalid hence give error.

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

Truncate is a DDL command where as delete is a DML command. The main difference is if we are using truncate command we can’t able to rollback the records what we have deleted. But in the case of delete it is possible to rollback the records. The another major difference is delete command deletes the contents of the table row by row, but truncate command will drop the entire table and reconstruct only the structure of the table and this is the reason why truncate is faster than delete.

**What is the output of the following query  
SELECT TRUNC (1234.5678,-2) FROM DUAL;?**

1200 as trunc (truncate) is used for truncation if you use –ve (-2) scale it will round 2 digit to 00 from "." if you use +ve (2) it will give 1234.56 will round up 2 digit from "."

**Which date function is used to find the difference between two dates?**

datediff   
for Eg: select datediff (dd,'2-06-2007','7-06-2007')

**How we get second highest salary in database? Please show me this coding.**

To retrieve the second max salary from database run the following query:-  
  
Taking EMPLOYEE as a database table and SALARY as a field where we have to find second max salary.  
  
SELECT MAX(SALARY) FROM EMPLOYEE WHERE SALARY NOT IN (SELECT MAX(SALARY) FROM EMPLOYEE)

*Alternatives*

* select max(salary) from employee where salary!=(select max(salary) from employee);
* select top 1 from (select top 2 from (select salary from employee order by salary) order by salary dsc)
* select min(salary) from employee where salary IN(select top 2 salary from employee order by salary desc)-----A subquery

**How to delete same id in rows?  
For example empid=5 repeated for times in rows. How to delete which sql query is used?**

sql> select distinct \* from emp;  
  
The above query will avoid the duplicate records froom the table and will dispaly the unique records. But remember it will not delete the duplicate record it will just filter the unique r display.  
  
select column1,count(column1) from table1 group by column1 having count(column1)>1   
  
The above query will just dispaly the records repeted in times.  
  
SQL> delete emp where rowid not in (select min(rowid) from emp group by name);  
  
3 rows deleted.  
  
the above is query to delete the duplicate records from the table. After executing the above command if will run the below query  
sql> select \* from emp;  
The above query will display the unique records only as there is no duplicate records exeisting in table.

**What are the advantages and disadvantages of primary key and foreign key in SQL?**

Primary key  
  
Advantages  
  
1) It is a unique key on which all the other candidate keys are functionally dependent  
  
Disadvantage  
  
1) There can be more than one keys on which all the other attributes are dependent on.  
  
Foreign Key  
  
Advantage  
  
1)It allows referencing another table using the primary key for the other table

First do remember that a table can have only 1 primary key, though we can have many columns with a single primary key, called composite primary key.  
Advantages:  
1. It prevents to enter null data.  
2. It prevents to enter duplicate data.  
3. It helps to force integrity constraints.  
  
Disadvantage  
On primary key, index will be created so during updation of table index need to be adjusted accordingly. This process makes the updation slower

**What command is used to create a table by copying the structure of another table?**

CREATE TABLE .. AS SELECT command  
Explanation:  
To copy only the structure, the WHERE clause of the SELECT command should contain a FALSE statement as in the following.  
CREATE TABLE NEWTABLE AS SELECT \* FROM EXISTINGTABLE WHERE 1=2;  
If the WHERE condition is true, then all the rows or rows satisfying the condition will be copied to the new table.

**What is the use of the DROP option in the ALTER TABLE command?**

It is used to delete The Constraints on the table or delete column in the table  
ex:  
alter table <tab-name> drop constraint <constraint name>;  
  
alter table <tab-name> drop column <column-name>

**How to programmatically find out when the SQL Server service started?**

There is one simple but awkward method is there to find out when the SQL server service is started.  
The following query will give you the time when the SQL server service is started;  
  
select crdate from [master].dbo.sysdatabases  
where name=?tempdb?  
  
This is because, the tempdb will be created whenever SQL server service is started or restarted.

**What will be the output of the following query?  
SELECT REPLACE(TRANSLATE(LTRIM(RTRIM('!! ATHEN !!','!'), '!'), 'AN', '\*\*'),'\*','TROUBLE') FROM DUAL;**

Step by step:  
1st step : rtrim('!!athen!!','!') its out put come like this "!!athen"  
2nd step: ltrim('!!athen','!') its out put come "athen"  
3rd step :trancelate('athen','an','\*\*') its output come like this "\*the\*"  
4th step :replace('\*the\*','\*','trouble')  
finally its output being look like this "trouble the trouble"

**What is a default TCP/IP socket assigned for SQL Server?**

1433 is default tcp/ip socket sql server.

**Which system table contains information on constraints on all the tables created?**

The system table user\_constraints contains all the information about constraints of all tables.

**Which system tables contain information on privileges granted and privileges obtained?**

USER\_TAB\_PRIVS\_MADE, USER\_TAB\_PRIVS\_RECD.

**What is the use of DESC in SQL?**

DESC has two purposes. It is used to describe a schema as well as to retrieve rows from table in descending order.  
Explanation:  
The query SELECT \* FROM EMP ORDER BY ENAME DESC will display the output sorted on ENAME in descending order.

**Which function is used to find the largest integer less than or equal to a specific value?**

FLOOR (Transact-SQL)  
  
Returns the largest integer less than or equal to the specified numeric expression.  
  
Syntax:  
FLOOR ( numeric\_expression )  
  
Arguments:   
numeric\_expression ->  
Is an expression of the exact numeric or approximate numeric data type category, except for the bit data type.  
  
Return Types ->  
Returns the same type as numeric\_expression.  
  
Examples ->  
The following example shows positive numeric, negative numeric, and currency values with the FLOOR function.  
  
SELECT FLOOR(123.45), FLOOR(-123.45), FLOOR($123.45)  
  
The result is the integer part of the calculated value in the same data type as numeric\_expression.  
  
123 -124 123.0000

**How to get a last inserted record or Row from a table without using any Id column?**

Below is my data which is present in a table:

insert into dupemp(name,salary,deptno) values('1sd',4000,12)  
insert into dupemp(name,salary,deptno) values('2sd',4000,12)  
insert into dupemp(name,salary,deptno) values('3sd',4000,12)  
insert into dupemp(name,salary,deptno) values('6sd',4000,12)  
insert into dupemp(name,salary,deptno) values('4sd',4000,12)

*Query:-*

**Method I:-**

select top 1 column\_list  FROM  
(select ROW\_NUMBER() over(order by (select 0)) as rownum,column\_list from table\_name) t order by rownum desc

**Method II:-**

with CTE as(select \*,ROW\_NUMBER() over(order by (select 0))as row from dupemp)  
select \* from cte where row=(select max(row) from cte)

OutPut:-

name  salary  deptno

4sd     4000    12

**How to get the second last record from a Table?**

insert into dupemp(name,salary,deptno) values('1sd',4000,12)  
  
insert into dupemp(name,salary,deptno) values('2sd',4000,12)  
  
insert into dupemp(name,salary,deptno) values('3sd',4000,12)  
  
insert into dupemp(name,salary,deptno) values('6sd',4000,12)  
  
insert into dupemp(name,salary,deptno) values('4sd',4000,12)

*Query:-*

**with CTE as(select \*,ROW\_NUMBER() over(order by (select 0))as row from dupemp )   
select \* from cte where row=(select min(row) from cte where row in(select top 2 row from cte order by row desc))**   
  
***Output:-***  **Name salary deptno**   
6sd  4000    12

**List the highest salary from each department?**

Important quries in sqlserver 

to find the highest salary value.

create table **empname**   
(   
name varchar(20),   
salary int,   
deptno int   
)   
 **name salary deptno**   
inthiyaaz 2000 10   
khaja      8000 10   
Afroz      2000 20   
Heera    1000  30   
Naseeb  1000 10   
King      1000   20   
Imroz    3000  30   
Naseer   2000 10   
Pankaj   1500  30   
--------------------------------------   
  
***Q1)* List the highest salaries from each department?**   
  
***Ans*** )with CTE as(select row\_number() over(partition by deptno order by salary desc) row,salary,deptno from empname)   
select \* from CTE where row=1   
  
**OutPut** :-   
 **row salary deptno**   
1 8000 10   
1 2000 20   
1 2000 30   
  
  
*Question)* **List the Top 2 highest salaries from each department?**   
  
with CTE as(select row\_number() over(partition by deptno order by salary desc) row,salary,deptno from empname)   
select \* from CTE where row<3   
**OutPut** :-   
 **row salary deptno**   
1 8000 10   
2 3000 10   
1 2000 20   
2 1000 20   
1 2000 30   
2 1500 30   
--------------------------------------   
  
**Q3)List the name of all Employees who are taking 3rd highest salaries?**   
Ans)***Method 1***   
select Name,salary from empname   
where salary=(select min(salary) from empname   
where salary in (select top 3 salary from empname order by salary desc))   
  
***OutPut)*   
  
name salary**   
inthiyaaz 2000   
Afroz 2000   
Imroz 2000   
  
***Method 2***  
with CTE as(select dense\_rank() over( order by salary desc) row,salary,deptno from empname)   
select \* from CTE where row=3   
***OutPut)*   
  
name salary**   
inthiyaaz 2000   
Afroz 2000   
Imroz 2000

**How can I hide a particular table name of our schema?**

You can hide the table name by creating synonyms.  
  
e.g) you can create a synonym y for table x  
  
create synonym y for x;

**TRUNCATE TABLE EMP;  
DELETE FROM EMP;**

**DROP FROM EMP;  
Will the outputs of the above two commands differ?**

**Or What is the difference between delete, drop and truncate?**

Both will result in deleting all the rows in the table EMP. A delete statement deletes all records in the table, but it does not free any space.

A Truncate statement frees the space as well. However, a truncate cannot be rolled back but delete can be rolled back.

Delete :- It is used to delete the rows from the table. The DELETE command is used to remove rows from a table. A WHERE clause can be used to only remove some rows. If no WHERE condition is specified, all rows will be removed. After performing a DELETE operation you need to COMMIT or ROLLBACK the transaction to make the change permanent or to undo it. Note that this operation will cause all DELETE triggers on the table to fire.

SQL> SELECT COUNT(\*) FROM emp;

COUNT(\*)

----------

14

SQL> DELETE FROM emp WHERE job = 'CLERK';

4 rows deleted.

SQL> COMMIT;

Commit complete.

SQL> SELECT COUNT(\*) FROM emp;

COUNT(\*)

----------

10

Truncate:- It will delete the whole table. TRUNCATE is more faster than DELETE. TRUNCATE removes all rows from a table. The operation cannot be rolled back. As such, TRUCATE is faster and doesn't use as much undo space as a DELETE.

SQL> TRUNCATE TABLE emp;

Table truncated.

SQL> SELECT COUNT(\*) FROM emp;

COUNT(\*)

----------

0

The DROP command removes a table from the database. All the tables' rows, indexes and privileges will also be removed. The operation cannot be rolled back.

SQL> DROP TABLE emp;

Table dropped.

SQL> SELECT \* FROM emp;

SELECT \* FROM emp

\*

ERROR at line 1:

DROP and TRUNCATE are DDL commands, whereas DELETE is a DML command. Therefore DELETE operations can be rolled back (undone), while DROP and TRUNCATE operations cannot be rolled back.

**What does the following query do?  
SELECT SAL + NVL(COMM,0) FROM EMP;**

This displays the total salary of all employees. The null values in the commission column will be replaced by 0 and added to salary. Wherever the commission is null, it is replaced by 0 & added to the salary.

**How to find out the database name from SQL\*PLUS command prompt?**

Select \* from global\_name;  
This will give the datbase name which u r currently connected to.

**State true or false.  
!=,   
<>,   
^=   
all denote the same operation.**

True. ^= Not same as <> or !=

**What is the advantage of specifying WITH GRANT OPTION in the GRANT command?**

The privilege receiver can further grant the privileges he/she has obtained from the owner to any other user.

**What is the value of comm and sal after executing the following query if the initial value of ?sal? is 10000  
UPDATE EMP SET SAL = SAL + 1000, COMM = SAL\*0.1;**

sal = 11000, comm = 1000.

**1. How to create a new table?  
2. How to insert another column into the table?**

Create table employee(name varchar2(10),id number(10),mobileno number(10));

Inserting a new Column into a table:  
General Syntax:  
insert into tablename(columnname datatype);  
  
Example:  
insert into employee(salary number(10));

**What are the privileges that can be granted on a table by a user to others?**

Insert, update, delete, select, references, index, execute, alter, all.

**What is the difference between varchar & varchar2?**

Varchar is used to store alphanumeric values without padding the unused memory locations whereas Varchar2 is also used to store alphanumeric values with padding the unused memory locations. So, by using varchar2 we are saving the memory locations.  
For example  
name varchar(30);  
name1 varchar2(30);  
  
In the above variable declarations compiler will allocates 30 bytes to name whereas name1 will utilities the memory location whatever the characters utilised by the variable.

**How to retrieving the data from 11th column to n th column in a table?**

select \* from emp where rowid in ( select rowid from emp where rownum <=&upto  
minus  
select rowid from emp where rownum <&startfrom)

From this, you can select between any range.

**How to display duplicate rows in a table?**

select \* from emp where EmpID in(select EmpID  
from emp  
group by EmpID having count(EmpID)>1)

**What command is used to get back the privileges offered by the GRANT command?**

Revoke is the command which revokes access privileges for database objects previously granted to other users.

REVOKE privilege\_name   
ON object\_name   
FROM {user\_name |PUBLIC |role\_name}

**When using a count(distinct) is it better to use a self-join or temp table to find redundant data, and provide an example?**

Instead of this we can use GROUP BY Clause with HAVING condition.  
For ex:  
Select count(\*),lastname from tblUsers group by lastname having count(\*)>1  
  
This query return the duplicated lastnames values in the lastname column from tblUsers table.

**I have a table with duplicate names in it. Write me a query which returns only duplicate rows with number of times they are repeated.**

SELECT COL1 FROM TAB1   
WHERE COL1 IN   
(SELECT MAX(COL1) FROM TAB1   
GROUP BY COL1   
HAVING COUNT(COL1) > 1 )

Alternatively,

Find Duplicate Row  
-------------------  
SELECT NAME,COUNT(NAME) FROM TABLE1 GROUP BY NAME  
HAVING COUNT(NAME) > 1  
  
Remove Duplicate Row (Copy & paste and run)  
----------------------------------  
declare @TBL table(EmpName varchar(10))  
  
insert into @TBL values('Manish')  
insert into @TBL values('Manish')  
insert into @TBL values('Milan')  
insert into @TBL values('Milan')  
insert into @TBL values('Mukesh')  
insert into @TBL values('Manav')  
insert into @TBL values('Madhav')  
  
select \* from @TBL -- Before Delete  
  
declare @TBL1 table(ID int identity(1,1),EmpName varchar(10))  
  
insert into @TBL1(EmpName)  
select EmpName from @TBL group by EmpName having count(EmpName) > 1  
  
declare @iLoop int,@Cnt int,@EmpName varchar(10)  
select @Cnt = count(\*) from @TBL1  
set @iLoop = 1  
while (@iLoop <= @Cnt)  
begin  
select @EmpName = EmpName from @TBL1 where ID = @iLoop  
delete top(1) from @TBL where EmpName = @EmpName  
set @iLoop = @iLoop + 1  
end  
  
select \* from @TBL -- After Delete

**What columns would you use as a clustered index on a retail store address table?**

On Primary Key column.

**What is the difference between @@identity and SCOPE\_IDENTITY()?**

This particular issue isn't so much about doing something right or wrong, it is about understanding your options so you choose the right one. Both @@IDENTITY and SCOPE\_IDENTITY() return the last identity value (primary key) that was entered by your active session, but in different scenarios they can each return different values. When I say "active session" I am referring to the current activity you are engaging in. For example, if you can a stored procedure, that is what I am referring to as your active session. Each call to t a stored procedure (or user defined function, etc) is a session, unless the a stored procedure is nested in the stored procedure you are calling. In the case of a nested stored procedure or user defined method, while they are separate methods, they are part of the current session, but not part of the current scope. Your scope is limited to the method (stored procedure or user defined function) that you explicitly invoked. This is where the difference between @@IDENTITY and SCOPE\_IDENTITY() comes in.  
  
@@IDENTITY will return the last identity value entered into a table in your current session (this is limited to your session only, so you won't get identities entered by other users). While @@IDENTITY is limited to the current session, it is not limited to the current scope. In other words, if you have a trigger on a table that causes an identity to be created in another table, you will get the identity that was created last, even if it was the trigger that created it. Now this isn't bad, as long as you ensure that things are done in the correct order. Where this can get ugly is when there is an application revision and a new trigger gets added that gets fired from your stored procedure. Your code didn't anticipate this new trigger, so you could now be getting an incorrect value back.  
  
SCOPE\_IDENTITY(), like @@IDENTITY, will return the last identity value created in the current session, but it will also limit it to your current scope as well. In other words, it will return the last identity value that you explicitly created, rather than any identity that was created by a trigger or a user defined function.

**State true or false. EXISTS, SOME, ANY are operators in SQL.**

True.

**There is a eno & gender in a table. Eno has primary key and gender has a check constraints for the values 'M' and 'F'.  
While inserting the data into the table M was misspelled as F and F as M.  
What is the update statement to replace F with M and M with F?**

UPDATE <tableName>   
SET Gender = (CASE WHEN Gender = 'F' THEN 'M' WHEN Gender = 'M' THEN 'F' END)

**What is table space?**

A database is divided into one or more logical storage units called tablespaces. Tablespaces are divided into logical units of storage called segments,   
  
Tablespace is a logical container for our Database object, Table constains differecent segment for different object, i.e., Table segment for Table, Index segment for index etc.  
There are 3 type of Tablespace in Oracle.   
1. Permanant Tablespace  
2. Temprary Tablespace  
3. Undo TableSpace  
  
First when we make a transaction, it is recorded inside undo Tablespace, but as we commit then it is recorded into Permanent Tablespace.s

**What is output of following query  
Select 2 from employee;**

It depends upon number of rows in table. This query will print 2 as many times as rows in table.

**How write a SQL statement to query the result set and display row as columns and columns as row?**

TRANSFORM Count(Roll\_no) AS CountOfRoll\_no  
SELECT Academic\_Status  
FROM tbl\_enr\_status  
GROUP BY Academic\_Status  
PIVOT Curnt\_status;

**In subqueries, which is efficient, the IN clause or EXISTS clause? Does they produce the same result?**

EXISTS is efficient bcose,  
  
1.Exists is faster than IN clause.  
  
2.IN check returns values to main query where as EXISTS returns Boolean (T or F).

**lets i have a table "bapi" having 6 field like this  
f1 f2 f3 f4 f5 f6  
... ... .. ... ... ...  
p r o m o d  
  
i want to retrive all the data by using select statement and my output will be look like this:  
xyz  
...  
p  
r  
o  
m  
o  
d**

elect xyz from  
(  
select xyz,ord from  
(  
select 1 as ord,f1 xyz from bapi  
union all  
select 2 as ord, f2 xyz from bapi  
union all  
select 3 as ord, f3 xyz from bapi  
union all  
select 4 as ord, f4 xyz from bapi  
union all  
select 5 as ord, f5 xyz from bapi  
union all  
select 6 as ord, f6 xyz from bapi  
) inqry  
--order by ord,xyz  
) outqry  
order by xyz

**How to get the first day of the week, last day of the week and last day of the month using T-SQL date functions?**

Use the following query find week days   
  
select datename(dw,0),datename(dw,6)   
Result: Monday Sunday

**How write a SQL statement to query the result set and display row as columns and columns as row?**

TRANSFORM Count(Roll\_no) AS CountOfRoll\_no  
SELECT Academic\_Status  
FROM tbl\_enr\_status  
GROUP BY Academic\_Status  
PIVOT Curnt\_status;

**How to find out the 10th highest salary in SQL query?**

The answer should be  
  
Table - Tbl\_Test\_Salary  
Column - int\_salary  
  
select min(int\_salary)  
from Tbl\_Test\_Salary  
where int\_salary in  
(select top 10 int\_Salary from Tbl\_Test\_Salary order by int\_salary desc)

**How to find the given query is optimised one or not?**

Using the power of the SQL Profiler and running a simple trace to capture the performance of your stored procedures can easily obtain these query lists that needs tuning.

**What’s the back end processes when we type "Select \* from Table"?**

First it will look into the System Global Area (SGA) weather the query is been exectued earlier.  
  
If it exist, it would retrive the same output present in memory.  
  
If not the query we typed is complied and the resulting parse tree and excution plan is been stored in SGA. Then query gets executed and output is given to the application.

**How do I write a cron which will run a SQL query and mail the results to a group?**

Use DBMS\_JOB for scheduling a cron job and DBMS\_MAIL to send the results throught email.

**Write a Query to find unique names of authors who have written books. (Using Pubs database)?**

select distinct Author\_name  
from Author\_table as At inner join books\_table as Bt   
on at.authorid = bt.authorid

**What is the usage of SAVEPOINTS?**

SAVEPOINTS are used to subdivide a transaction into smaller parts. It enables rolling back part of a transaction. Maximum of five save points are allowed.

**Can a view be updated/inserted/deleted? If Yes, then under what conditions?**

A View can be updated/deleted/inserted if it has only one base table if the view is based on columns from one or more tables then insert, update and delete is not possible.

**How to analyze the performance of a query using Explain Plan? Can any one explain me this in detail.**

From execution plan, first you have to check whether the query uses index scan or index seek. if it is index scan then you have to make it as index seek.  
and then you have to see the subtree cost of the query. always it should be 0.0003 m.seconds.

**What is the difference between Single row sub-Query and Scalar sub-Query?**

A single row sub-query can retrieve more than 1 column, but 1 row. Where-as a Scalar sub-query retrieves only 1 row 1 columns.

**How to save the output of a query/ stored procedure to a text file using T-SQL?**

Open Query Analyzer And then Press Ctrl + T   
After that run the query . It will ask for the path to save the file.

**Given an employee and manager table, write a SQL syntax that could be used to find out an employee's manager's manager, assuming all managers are in the employee table.**

It is assumed that u have created a single table for populating data of Employee and Manager (.a Manager is also an Employee so s/he will be residing in the same table) ..  
  
The sample data would be like this  
  
EmployeeID ----- EmployeeName ------- ManagerID  
  
1 Akhtar 0  
  
2 Bilal 1  
  
3 Faheem 2  
  
\*\* This query will return "Akhtar" for given EmployeeID = 3.  
  
SELECT m.EmployeeName FROM #EmpTemp m WHERE m.EmloyeeID = (SELECT e.ManagerID FROM #EmpTemp e WHERE e.EmloyeeID = (Select t.ManagerID   
FROM #EmpTemp t WHERE t.EmloyeeID = 3))  
  
\*\* This is solution is workable in case of 3-levels only not even to 2-levels of hierarchy. An n-level solution can be achieve by writing a Recursive stored procedure and that will also be given soon.

THE BEST QUERY IS TO CREATE THE DUMMY TABLE AND THEN MAKE THE RELATIONS  
  
select emp.employeeID, emp.EmployeeName,emp.ManagerID,  
empTemp.EmployeeName MANAGER  
from employees emp ,employees empTemp  
where emp.managerId=empTemp.EmployeeID

**Write a Query to pull out the maximum unit price for each in the database. (Using Northwind database).**

select empname,max(salary) from employee group by empname

**I have a table with duplicate names in it. Write me a query which returns only duplicate rows with number of times they are repeated.**

select num,count(num)as count from x  
group by (num)  
having count(num)>1

**Explain UNION, MINUS, UNION ALL and INTERSECT.**

* INTERSECT - returns all distinct rows selected by both queries.
* MINUS - returns all distinct rows selected by the first query but not by the second.
* UNION - returns all distinct rows selected by either query
* UNION ALL - returns all rows selected by either query, including all duplicates.

**If a view on a single base table is manipulated will the changes be reflected on the base table?**

If changes are made to the tables and these tables are the base tables of a view, then the changes will be reference on the view.

**What is an integrity constraint?**

Integrity constraint is a rule that restricts values to a column in a table.

**What is referential integrity constraint?**

Maintaining data integrity through a set of rules that restrict the values of one or more columns of the tables based on the values of primary key or unique key of the referenced table.

**What is the fastest way of accessing a row in a table?**

The fastest way of accessing a row in a table is using ROWID CONSTRAINTS.

**How to Select last N records from a Table?**

select top 10 \* from table\_name order by timestamp\_field desc

**What is diffrence between Co-related sub query and nested sub query?**Correlated subquery runs once for each row selected by the outer query. It contains a reference to a value from the row selected by the outer query.  
  
For example,   
  
Correlated Subquery:   
  
select e1.empname, e1.basicsal, e1.deptno from emp e1 where e1.basicsal = (select max(basicsal) from emp e2 where e2.deptno = e1.deptno)  
  
Nested subquery runs only once for the entire nesting (outer) query. It does not contain any reference to the outer query row.

Nested Subquery:   
  
select empname, basicsal, deptno from emp where (deptno, basicsal) in (select deptno, max(basicsal) from emp group by deptno)

**How many LONG columns are allowed in a table? Is it possible to use LONG columns in WHERE clause or ORDER BY?**

Only one LONG column is allowed. It is not possible to use LONG column in WHERE or ORDER BY clause.

**Why is it better to use an integrity constraint to validate data in a table than to use a stored procedure?**

Because an integrity constraint is automatically checked while data is inserted into a table. A stored procedure has to be specifically invoked.

**Explain SQL Query Order of Operations.**

* FROM clause
* WHERE clause
* GROUP BY clause
* HAVING clause
* SELECT clause
* ORDER BY clause

**Database Questions**

1. **What is the wildcard character in SQL?**Let’s say you want to query database with LIKE for all employees whose name starts with La. The wildcard character is %, the proper query with LIKE would involve ‘La%’.
2. **Explain ACID rule of thumb for transactions.**A transaction must be:  
   1.       Atomic - it is one unit of work and does not dependent on previous and following transactions.  
   2.       Consistent - data is either committed or roll back, no “in-between” case where something has been updated and something hasn’t.  
   3.       Isolated - no transaction sees the intermediate results of the current transaction).  
   4.       Durable - the values persist if the data had been committed even if the system crashes right after.
3. **What connections does Microsoft SQL Server support?**Windows Authentication (via Active Directory) and SQL Server authentication (via Microsoft SQL Server username and password).
4. **Between Windows Authentication and SQL Server Authentication, which one is trusted and which one is untrusted?**   
   Windows Authentication is trusted because the username and password are checked with the Active Directory, the SQL Server authentication is untrusted, since SQL Server is the only verifier participating in the transaction.
5. **What does the Initial Catalog parameter define in the connection string?**   
   The database name to connect to.
6. **What does the Dispose method do with the connection object?**Deletes it from the memory.  
   **To Do:** answer better.  The current answer is not entirely correct.

**What is a pre-requisite for connection pooling?**   
Multiple processes must agree that they will share the same connection, where every parameter is the same, including the security settings.  The connection string must be identical.

**Which of the following statements contains an error?**

A. SELECT \* FROM emp WHERE empid = 493945;  
B. SELECT empid FROM emp WHERE empid= 493945;  
C. SELECT empid FROM emp;  
D. SELECT empid WHERE empid = 56949 AND lastname = ‘SMITH’;

Ans: D

**Which of the following correctly describes how to specify a column alias?**

A. Place the alias at the beginning of the statement to describe the table.  
B. Place the alias after each column, separated by white space, to describe the column.  
C. Place the alias after each column, separated by a comma, to describe the column.  
D. Place the alias at the end of the statement to describe the table.

Ans: B

**The NVL function**

A. Assists in the distribution of output across multiple columns.  
B. Allows the user to specify alternate output for non-null column values.  
C. Allows the user to specify alternate output for null column values.  
D. Nullifies the value of the column output.

Ans: C

**Output from a table called PLAYS with two columns, PLAY\_NAME and AUTHOR, is shown below. Which of the following SQL statements produced it?**

PLAY\_TABLE  
————————————-  
“Midsummer Night’s Dream”, SHAKESPEARE  
“Waiting For Godot”, BECKETT  
“The Glass Menagerie”, WILLIAMS

A. SELECT play\_name || author FROM plays;  
B. SELECT play\_name, author FROM plays;  
C. SELECT play\_name||’, ‘ || author FROM plays;  
D. SELECT play\_name||’, ‘ || author PLAY\_TABLE FROM plays;

Ans: D

**Issuing the DEFINE\_EDITOR="emacs” will produce which outcome?**

A. The emacs editor will become the SQL\*Plus default text editor.  
B. The emacs editor will start running immediately.  
C. The emacs editor will no longer be used by SQL\*Plus as the default text editor.  
D. The emacs editor will be deleted from the system.

Ans: A

**The user issues the following statement. What will be displayed if the EMPID selected is 60494?**

**SELECT DECODE(empid,38475, “Terminated”,60494, “LOA”, “ACTIVE")  
FROM emp;**

A. 60494  
B. LOA  
C. Terminated  
D. ACTIVE

Ans: B

**SELECT (TO\_CHAR(NVL(SQRT(59483), “INVALID")) FROM DUAL is a valid SQL statement.**

A. TRUE  
B. FALSE

Ans: B

**The appropriate table to use when performing arithmetic calculations on values defined within the SELECT statement (not pulled from a table column) is**

A. EMP  
B. The table containing the column values  
C. DUAL  
D. An Oracle-defined table

Ans: C

**Which of the following is not a group function?**

A. avg( )  
B. sqrt( )  
C. sum( )  
D. max( )

Ans: B

**Once defined, how long will a variable remain so in SQL\*Plus?**

A. Until the database is shut down  
B. Until the instance is shut down  
C. Until the statement completes  
D. Until the session completes

Ans: D

**The default character for specifying runtime variables in SELECT statements is**

A. Ampersand  
B. Ellipses  
C. Quotation marks  
D. Asterisk

Ans: A

**A user is setting up a join operation between tables EMP and DEPT. There are some employees in the EMP table that the user wants returned by the query, but the employees are not assigned to departments yet. Which SELECT statement is most appropriate for this user?**

A. select e.empid, d.head from emp e, dept d;  
B. select e.empid, d.head from emp e, dept d where e.dept# = d.dept#;  
C. select e.empid, d.head from emp e, dept d where e.dept# = d.dept# (+);  
D. select e.empid, d.head from emp e, dept d where e.dept# (+) = d.dept#;

Ans: D

**Developer ANJU executes the following statement: CREATE TABLE animals AS SELECT \* from MASTER.ANIMALS; What is the effect of this statement?**

A. A table named ANIMALS will be created in the MASTER schema with the same data as the ANIMALS table owned by ANJU.  
B. A table named ANJU will be created in the ANIMALS schema with the same data as the ANIMALS table owned by MASTER.  
C. A table named ANIMALS will be created in the ANJU schema with the same data as the ANIMALS table owned by MASTER.  
D. A table named MASTER will be created in the ANIMALS schema with the same data as the ANJU table owned by ANIMALS.

Ans: C

**User JANKO would like to insert a row into the EMPLOYEE table, which has three columns: EMPID, LASTNAME, and SALARY. The user would like to enter data for EMPID 59694, LASTNAME Harris, but no salary. Which statement would work best?**

A. INSERT INTO employee VALUES (59694,’HARRIS’, NULL);  
B. INSERT INTO employee VALUES (59694,’HARRIS’);  
C. INSERT INTO employee (EMPID, LASTNAME, SALARY) VALUES (59694,’HARRIS’);  
D. INSERT INTO employee (SELECT 59694 FROM ‘HARRIS’);

Ans: A

**Which three of the following are valid database datatypes in Oracle? (Choose three.)**

A. CHAR  
B. VARCHAR2  
C. BOOLEAN  
D. NUMBER

Ans: A,B,D

**Omitting the WHERE clause from a DELETE statement has which of the following effects?**

A. The delete statement will fail because there are no records to delete.  
B. The delete statement will prompt the user to enter criteria for the deletion  
C. The delete statement will fail because of syntax error.  
D. The delete statement will remove all records from the table.

Ans: D

**Creating a foreign-key constraint between columns of two tables defined with two different datatypes will produce an error.**

A. TRUE  
B. FALSE

Ans: A

**Dropping a table has which of the following effects on a nonunique index created for the table?**

A. No effect.  
B. The index will be dropped.  
C. The index will be rendered invalid.  
D. The index will contain NULL values.

Ans: B

**To increase the number of nullable columns for a table,**

A. Use the alter table statement.  
B. Ensure that all column values are NULL for all rows.  
C. First increase the size of adjacent column datatypes, then add the column.  
D. Add the column, populate the column, then add the NOT NULL constraint.

Ans: A

**Which line of the following statement will produce an error?**

A. CREATE TABLE goods  
B. (good\_no NUMBER,  
C. good\_name VARCHAR2 check(good\_name in (SELECT name FROM avail\_goods)),  
D. CONSTRAINT pk\_goods\_01  
E. PRIMARY KEY (goodno));  
F. There are no errors in this statement.

Ans: C

**MAXVALUE is a valid parameter for sequence creation.**

A. TRUE  
B. FALSE

Ans: A

**Which of the following lines in the SELECT statement below contain an error?**

A. SELECT DECODE(empid, 58385, “INACTIVE”, “ACTIVE") empid  
B. FROM emp  
C. WHERE SUBSTR(lastname,1,1) > TO\_NUMBER(’S')  
D. AND empid > 02000  
E. ORDER BY empid DESC, lastname ASC;  
F. There are no errors in this statement.

Ans: F

**Which function below can best be categorized as similar in function to an IF-THEN-ELSE statement?**

A. SQRT  
B. DECODE  
C. NEW\_TIME  
D. ROWIDTOCHAR

Ans: B

**Which two of the following orders are used in ORDER BY clauses? (choose two)**

A. ABS  
B. ASC  
C. DESC  
D. DISC

Ans: B,C

**You query the database with this command**

**SELECT name  
FROM employee  
WHERE name LIKE ‘\_a%’;**

**Which names are displayed?**

A. Names starting with “a”  
B. Names starting with “a” or “A”  
C. Names containing “a” as second character  
D. Names containing “a” as any letter except the first

Ans: C

PL/SQL  
——  
**Which of the following statements is true about implicit cursors?**

A. Implicit cursors are used for SQL statements that are not named.  
B. Developers should use implicit cursors with great care.  
C. Implicit cursors are used in cursor for loops to handle data processing.  
D. Implicit cursors are no longer a feature in Oracle.

Ans: B

**Which of the following is not a feature of a cursor FOR loop?**

A. Record type declaration.  
B. Opening and parsing of SQL statements.  
C. Fetches records from cursor.  
D. Requires exit condition to be defined.

Ans: D

**A developer would like to use referential datatype declaration on a variable. The variable name is EMPLOYEE\_LASTNAME, and the corresponding table and column is EMPLOYEE, and LNAME, respectively. How would the developer define this variable using referential datatypes?**

A. Use employee.lname%type.  
B. Use employee.lname%rowtype.  
C. Look up datatype for EMPLOYEE column on LASTNAME table and use that.  
D. Declare it to be type LONG.

Ans: A

**Which three of the following are implicit cursor attributes?**

A. %found  
B. %too\_many\_rows  
C. %notfound  
D. %rowcount  
E. %rowtype

Ans: A,C,D

**If left out, which of the following would cause an infinite loop to occur in a simple loop?**

A. LOOP  
B. END LOOP  
C. IF-THEN  
D. EXIT

Ans: D

**Which line in the following statement will produce an error?**

A. cursor action\_cursor is  
B. select name, rate, action  
C. into action\_record  
D. from action\_table;  
E. There are no errors in this statement.

Ans: C

**The command used to open a CURSOR FOR loop is**

A. open  
B. fetch  
C. parse  
D. None, cursor for loops handle cursor opening implicitly.

Ans: D

**What happens when rows are found using a FETCH statement**

A. It causes the cursor to close  
B. It causes the cursor to open  
C. It loads the current row values into variables  
D. It creates the variables to hold the current row values

Ans: C

34. CREATE OR REPLACE PROCEDURE find\_cpt  
(v\_movie\_id {Argument Mode} NUMBER, v\_cost\_per\_ticket {argument mode} NUMBER)  
IS  
BEGIN  
IF v\_cost\_per\_ticket > 8.5 THEN  
SELECT cost\_per\_ticket  
INTO v\_cost\_per\_ticket  
FROM gross\_receipt  
WHERE movie\_id = v\_movie\_id;  
END IF;  
END;

**Which mode should be used for V\_COST\_PER\_TICKET?**

A. IN  
B. OUT  
C. RETURN  
D. IN OUT

Ans: D

**CREATE OR REPLACE TRIGGER update\_show\_gross  
{trigger information}  
BEGIN  
{additional code}  
END;**

**The trigger code should only execute when the column, COST\_PER\_TICKET, is greater than $3.75. Which trigger information will you add?**

A. WHEN (new.cost\_per\_ticket > 3.75)  
B. WHEN (:new.cost\_per\_ticket > 3.75  
C. WHERE (new.cost\_per\_ticket > 3.75)  
D. WHERE (:new.cost\_per\_ticket > 3.75)

Ans: B

**What is the maximum number of handlers processed before the PL/SQL block is exited when an exception occurs?**

A. Only one  
B. All that apply  
C. All referenced  
D. None

Ans: A

**For which trigger timing can you reference the NEW and OLD qualifiers?**

A. Statement and Row  
B. Statement only  
C. Row only  
D. Oracle Forms trigger

Ans: C

38.  
CREATE OR REPLACE FUNCTION get\_budget(v\_studio\_id IN NUMBER)  
RETURN number IS

v\_yearly\_budget NUMBER;

BEGIN  
SELECT yearly\_budget  
INTO v\_yearly\_budget  
FROM studio  
WHERE id = v\_studio\_id;

RETURN v\_yearly\_budget;  
END;

**Which set of statements will successfully invoke this function within SQL\*Plus?**

A. VARIABLE g\_yearly\_budget NUMBER  
EXECUTE g\_yearly\_budget := GET\_BUDGET(11);  
B. VARIABLE g\_yearly\_budget NUMBER  
EXECUTE :g\_yearly\_budget := GET\_BUDGET(11);  
C. VARIABLE :g\_yearly\_budget NUMBER  
EXECUTE :g\_yearly\_budget := GET\_BUDGET(11);  
D. VARIABLE g\_yearly\_budget NUMBER  
:g\_yearly\_budget := GET\_BUDGET(11);

Ans: A

39.  
CREATE OR REPLACE PROCEDURE update\_theater  
(v\_name IN VARCHAR2, v\_theater\_id IN NUMBER) IS  
BEGIN  
UPDATE theater  
SET name = v\_name  
WHERE id = v\_theater\_id;  
END update\_theater;

When invoking this procedure, you encounter the error:  
ORA-00001: Unique constraint(SCOTT.THEATER\_NAME\_UK) violated.

**How should you modify the function to handle this error?**

A. An user defined exception must be declared and associated with the error code and handled in the EXCEPTION section.  
B. Handle the error in EXCEPTION section by referencing the error code directly.  
C. Handle the error in the EXCEPTION section by referencing the UNIQUE\_ERROR predefined exception.  
D. Check for success by checking the value of SQL%FOUND immediately after the UPDATE statement.

Ans: A

CREATE OR REPLACE PROCEDURE calculate\_budget IS  
v\_budget studio.yearly\_budget%TYPE;  
BEGIN  
v\_budget := get\_budget(11);  
IF v\_budget

**Which system table contains information on constraints on all the tables created ?**

yes,

USER\_CONSTRAINTS,

system table contains information on constraints on all the tables created

**How to find out the database name from SQL\*PLUS command prompt?**

Select \* from global\_name;

This will give the datbase name which u r currently connected to.....

**What is the difference between SQL and SQL Server?**

SQLServer is an RDBMS just like oracle, DB2 from Microsoft whereas Structured Query Language (SQL), pronounced "sequel", is a language that provides an interface to relational database systems. It was developed by IBM in the 1970s for use in System R. SQL is a de facto standard, as well as an ISO and ANSI standard. SQL is used to perform various operations on RDBMS.

**WHAT OPERATOR PERFORMS PATTERN MATCHING?**

Pattern matching operator is LIKE and it has to used with two attributes

1. % and

2. \_ ( underscore )

% means matches zero or more characters and under score means mathing exactly one character

**How can I hide a particular table name of our schema?**

You can hide the table name by creating synonyms.

e.g) you can create a synonym y for table x

create synonym y for x;

**What is difference between DBMS and RDBMS?**

The main difference of DBMS & RDBMS is RDBMS have Normalization. Normalization means to refine the redundant and maintain the stabilization. The DBMS hasn't normalization concept.

***SQL***

##### T-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 De normalization  
  
**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 non-clustered, 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 ('?')"

**What is the difference between a HAVING CLAUSE and a WHERE CLAUSE?**

HAVING specifies a search condition for a group or an aggregate. HAVING can be used only with the SELECT statement. HAVING is typically used in a GROUP BY clause. When GROUP BY is not used, HAVING behaves like a WHERE clause. Having Clause is basically used only with the GROUP BY function in a query. WHERE Clause is applied to each row before they are part of the GROUP BY function in a query.

**What is sub-query? Explain properties of sub-query.**

Sub-queries are often referred to as sub-selects, as they allow a SELECT statement to be executed arbitrarily within the body of another SQL statement. A sub-query is executed by enclosing it in a set of parentheses. Sub-queries are generally used to return a single row as an atomic value, though they may be used to compare values against multiple rows with the IN keyword.

A subquery is a SELECT statement that is nested within another T-SQL statement. A subquery SELECT statement if executed independently of the T-SQL statement, in which it is nested, will return a result set. Meaning a subquery SELECT statement can standalone and is not depended on the statement in which it is nested. A subquery SELECT statement can return any number of values, and can be found in, the column list of a SELECT statement, a FROM, GROUP BY, HAVING, and/or ORDER BY clauses of a T-SQL statement. A Subquery can also be used as a parameter to a function call. Basically a subquery can be used anywhere an expression can be used.

**Properties of Sub-Query**

A subquery must be enclosed in the parenthesis.

A subquery must be put in the right hand of the comparison operator, and

A subquery cannot contain a ORDER-BY clause.

A query can contain more than one sub-queries.

**What are types of sub-queries?**

Single-row sub-query, where the sub-query returns only one row.

Multiple-row sub-query, where the sub-query returns multiple rows, and

Multiple column sub-query, where the sub-query returns multiple columns.

**Can we rewrite sub queries into simple select statements or with joins?**

Sub-queries can often be re-written to use a standard outer join, resulting in faster performance. As we may know, an outer join uses the plus sign (+) operator to tell the database to return all non-matching rows with NULL values. Hence we combine the outer join with a NULL test in the WHERE clause to reproduce the result set without using a sub-query.

**How many sub-queries is possible in a statement?**

A sub-query can include one or more sub-queries. You can nest up to 16 sub-queries in a statement.

**Can we nest sub-queries in update, delete, and insert statements as well as in select statements?**

Yes

**Can we use sub-queries in conditional statements?**

Yes

**What is difference between co-related sub query and nested/inner sub query?**

Correlated sub-query runs once for each row selected by the outer query. It contains a reference to a value from the row selected by the outer query.

Nested sub-query runs only once for the entire nesting (outer) query. It does not contain any reference to the outer query row.

For example,

*Correlated Sub-query:*

select e1.empname, e1.basicsal, e1.deptno from emp e1 where e1.basicsal = (select max(basicsal) from emp e2 where e2.deptno = e1.deptno)

*Nested Sub-query:*

select empname, basicsal, deptno from emp where (deptno, basicsal) in (select deptno, max(basicsal) from emp group by deptno)

**What is SQL Profiler?**

SQL Profiler is a graphical tool that allows system administrators to monitor events in an instance of Microsoft SQL Server. You can capture and save data about each event to a file or SQL Server table to analyze later. For example, you can monitor a production environment to see which stored procedures are hampering performance by executing too slowly.

Use SQL Profiler to monitor only the events in which you are interested. If traces are becoming too large, you can filter them based on the information you want, so that only a subset of the event data is collected. Monitoring too many events adds overhead to the server and the monitoring process and can cause the trace file or trace table to grow very large, especially when the monitoring process takes place over a long period of time.

***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)

***INDEXES***

**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**.

**Explain about Clustered and non clustered index? How to choose between a Clustered Index and a Non-Clustered Index?**There are clustered and non clustered 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 non clustered index is a special type of index in which the logical order of the index does not match the physical stored order of the rows on disk.* The leaf nodes of a non clustered 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.

**How to decide whether to use clustered or non clustered index on a table?**

**Disadvantage of index?**Every index increases the time it takes to perform INSERTS, UPDATES and DELETES, so the number of indexes should not be very much.

**What is a table called, if it does not have neither Cluster nor Non-cluster Index? What is it used for?**

Unindexed table or Heap. Microsoft Press Books and Book On Line (BOL) refers it as Heap.

A heap is a table that does not have a clustered index and, therefore, the pages are not linked by pointers. The IAM pages are the only structures that link the pages in a table together.

*Un-indexed tables are good for fast storing of data. Many times it is better to drop all indexes from table and then do bulk of inserts and to restore those indexes after that.*

**How to know which index a table is using?**

SELECT table\_name, index\_name FROM user\_constraints

**How to copy the tables, schema and views from one SQL server to another?**

Microsoft SQL Server 2000 Data Transformation Services (DTS) is a set of graphical tools and programmable objects that lets user extract, transform, and consolidate data from disparate sources into single or multiple destinations.

**Why you need indexing? Where that is stored and what you mean by schema object? For what purpose we are using view?**

We can’t create an Index on Index.. Index is stored in user\_index table. Every object that has been created on Schema is Schema Object like Table, View etc. If we want to share the particular data to various users we have to use the virtual table for the Base table...So that is a view.

*Indexing is used for faster search or to retrieve data faster from various table.*

Schema containing set of tables, basically schema means logical separation of the database. View is crated for faster retrieval of data. It's customized virtual table. we can create a single view of multiple tables. Only the drawback is view needs to be get refreshed for retrieving updated data.

**What are the different index configurations a table can have?**

A table can have one of the following index configurations:

* No indexes
* A clustered index
* A clustered index and many nonclustered indexes
* A nonclustered index
* Many nonclustered indexes

**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.

**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)

**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.

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

This command is basically used when a large processing of data has occurred. If a large amount of deletions any modification or Bulk Copy into the tables has occurred, it has to update the indexes to take these changes into account. UPDATE\_STATISTICS updates the indexes on these tables accordingly.

**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.  
\*\*

**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 sorts data, while inserting/updating the table.

**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

**What is fill factor? 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***

**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 |  |

**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.

**GUID datasize?**128bit

**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.

**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***

**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.

**How many types of Joins?**Joins can be categorized as:

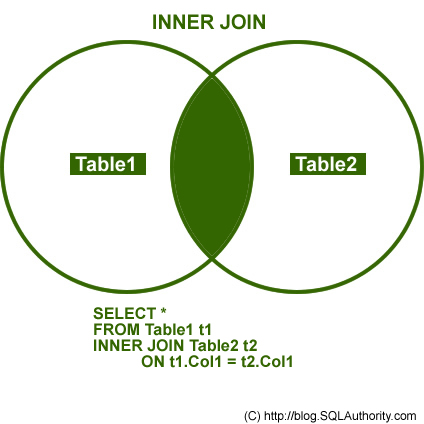
* Inner joins (the typical join operation, which uses some comparison operator like = or <>). These include equi-joins and natural joins.   
  Inner joins use a comparison operator to match rows from two tables based on the values in common columns from each table. For example, retrieving all rows where 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.)

**What is the difference between cross product & Cartesian product?**

There is no difference between Cartesian product and cross join. Although their syntax are different but they work as a same.  
  
Cartesian or cross product selects all rows from both the table.  
example  
  
tableA 6 rows  
tableB 6 rows  
  
Cartesian or cross product  
36 rows.

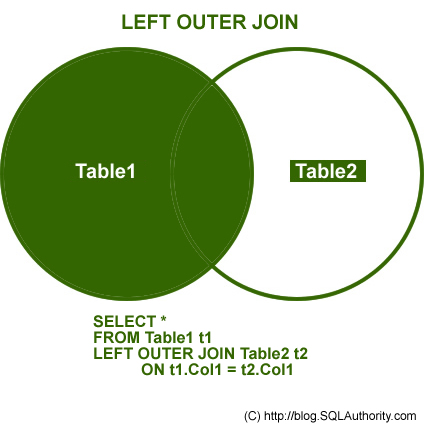
A cross join that does not have a 'where' clause gives the casterian product.  
Casterian product result set contains no of rows in first table multiplied by no of rows in second table.

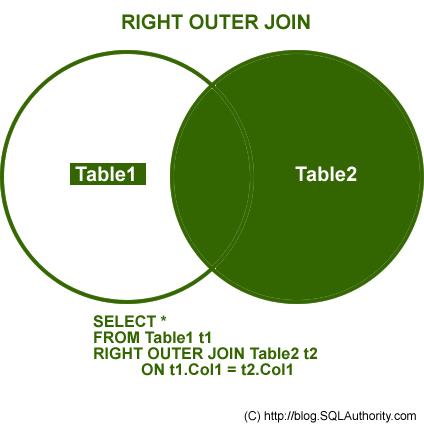
**INNER JOIN**

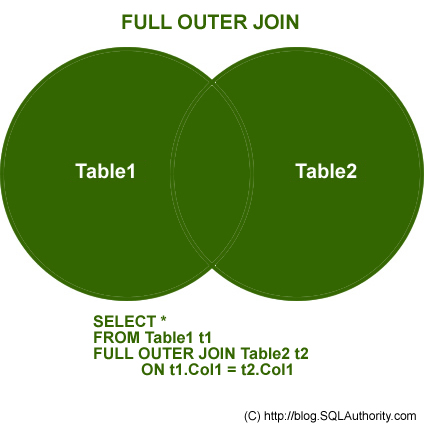
This join returns rows when there is at least one match in both the tables.  


**OUTER JOIN**

There are three different Outer Join methods.

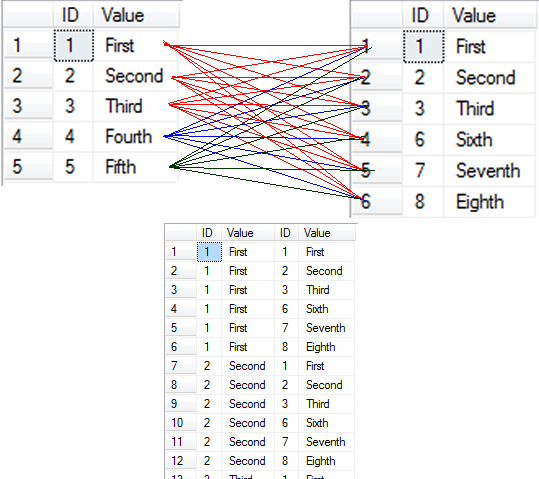
**LEFT OUTER JOIN**  
This join returns all the rows from the left table in conjunction with the matching rows from the right table. If there are no columns matching in the right table, it returns NULL values.  


**RIGHT OUTER JOIN**  
This join returns all the rows from the right table in conjunction with the matching rows from the left table. If there are no columns matching in the left table, it returns NULL values.  


**FULL OUTER JOIN**  
This join combines left outer join and right after join. It returns row from either table when the conditions are met and returns null value when there is no match.  


CROSS JOIN

This join is a Cartesian join that does not necessitate any condition to join. The resultset contains records that are multiplication of record number from both the tables.



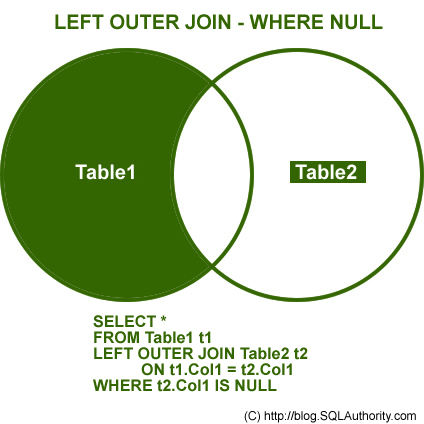
Additional Notes related to JOIN:

The following are three classic examples to display where Outer Join is useful. You will notice several instances where developers write query as given below.

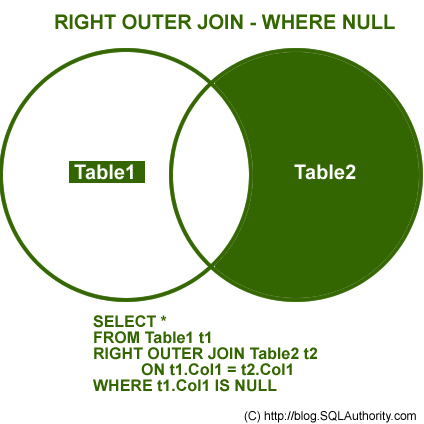
SELECT t1.\*  
FROM Table1 t1  
WHERE t1.ID NOT IN (SELECT t2.ID FROM Table2 t2)  
GO

The query demonstrated above can be easily replaced by Outer Join. Indeed, replacing it by Outer Join is the best practice. The query that gives same result as above is displayed here using Outer Join and WHERE clause in join.

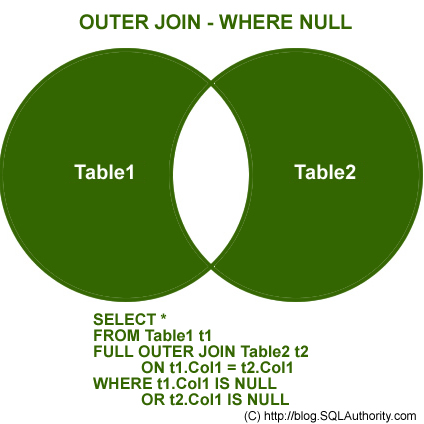
/\* LEFT JOIN - WHERE NULL \*/  
SELECT t1.\*,t2.\*  
FROM Table1 t1  
LEFT JOIN Table2 t2 ON t1.ID = t2.ID  
WHERE t2.ID IS NULL



The above example can also be created using Right Outer Join.



**NOT INNER JOIN**  
Remember, the term Not Inner Join does not exist in database terminology. However, when full Outer Join is used along with WHERE condition, as explained in the above two examples, it will give you exclusive result to Inner Join. This join will give all the results that were not present in Inner Join.



**What is Self Join? Explain it with an example.**

This is a particular case when one table joins to itself, with one or two aliases to avoid confusion. A self join can be of any type, as long as the joined tables are the same. A self join is rather unique in that it involves a relationship with only one table. The common example is when company have a hierarchal reporting structure whereby one member of staff reports to another.

Self join is just like any other join, except that two instances of the same table will be joined in the query. Here is an example: **Employees table which contains rows for normal employees as well as managers. So, to find out the managers of all the employees.**

CREATE TABLE emp

(

empid int,

mgrid int,

empname char(10)

)

INSERT emp SELECT 1,2,'Vyas'

INSERT emp SELECT 2,3,'Mohan'

INSERT emp SELECT 3,NULL,'Shobha'

INSERT emp SELECT 4,2,'Shridhar'

INSERT emp SELECT 5,2,'Sourabh'

SELECT t1.empname [Employee], t2.empname [Manager]

FROM emp t1, emp t2

WHERE t1.mgrid = t2.empid

Here's an advanced query using a LEFT OUTER JOIN that even returns the employees without managers (super bosses)

SELECT t1.empname [Employee], COALESCE(t2.empname, 'No manager') [Manager]

FROM emp t1

LEFT OUTER JOIN

emp t2

ON

t1.mgrid = t2.empid

**What is outer join? Explain with examples.**

Outer Join: Which return all the rows from one table and only those rows from secondary table where joined fields are equal (joined condition is met).  
Let's look at some data to explain how outer joins work:  
  
We have a table called suppliers with two fields (supplier\_id and name).  
It contains the following data:  
supplier\_id supplier\_name  
10000 IBM  
10001 Hewlett Packard  
10002 Microsoft  
10003 NVIDIA  
  
We have a second table called orders with three fields (order\_id, supplier\_id, and order\_date).  
It contains the following data:  
order\_id supplier\_id order\_date  
500125 10000 2003/05/12  
500126 10001 2003/05/13  
  
If we run the SQL statement below:  
  
select suppliers.supplier\_id, suppliers.supplier\_name, orders.order\_date  
from suppliers, orders  
where suppliers.supplier\_id = orders.supplier\_id(+);  
  
  
Our result set would look like this:  
supplier\_id supplier\_name order\_date  
10000 IBM 2003/05/12  
10001 Hewlett Packard 2003/05/13  
10002 Microsoft <null>  
10003 NVIDIA <null>

**What is Cross Join?**

A cross join that does not have a WHERE clause produces the Cartesian product of the tables involved in the join. The size of a Cartesian product result set is the number of rows in the first table multiplied by the number of rows in the second table. The common example is when company wants to combine each product with a pricing table to analyze each product at each price.

**What are the differences between UNION and JOINS?**A join selects columns from 2 or more tables. A union selects rows.

**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.

**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)

***LOCKS***

**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 |

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, they find 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.

## What are isolation levels? Discuss the different isolation levels.

## Isolation Levels

Isolation levels come into play when you need to isolate a resource for a transaction and protect that resource from other transactions. The protection is done by obtaining locks. What locks need to be set and how it has to be established for the transaction is determined by SQL Server referring to the Isolation Level that has been set. Lower Isolation Levels allow multiple users to access the resource simultaneously (concurrency) but they may introduce concurrency related problems such as *dirty-reads* and data inaccuracy. Higher Isolation Levels eliminate concurrency related problems and increase the data accuracy but they may introduce *blocking*.

Note that first four Isolation Levels described below are ordered from lowest to highest. The two subsequent levels are new to SQL Server 2005, and are described separately.   
  
**Read Uncommitted Isolation Level**

This is the lowest level and can be set, so that it provides higher concurrency but introduces all concurrency problems; *dirty-reads, Lost updates, Nonrepeatable reads (Inconsistent analysis)* and *phantom reads*. This Isolation Level can be simply tested.

Connection1 opens a transaction and starts updating *Employees* table.

USE Northwind

BEGIN TRAN

-- update the HireDate from 5/1/1992 to 5/2/1992

UPDATE dbo.Employees

SET HireDate = '5/2/1992'

WHERE EmployeeID = 1

Connection2 tries to read same record.

USE Northwind

SELECT HireDate

FROM dbo.Employees

WHERE EmployeeID = 1

You will see that *Connection2* cannot read data because an exclusive lock has been set for the resource by *Connection1*. The exclusive locks are not compatible with other locks. Though this reduces the concurrency, as you see, it eliminates the data inaccuracy by not allowing seeing uncommitted data for others. Now let’s set the *Isolation Level* of *Connection2* to *Read Uncommitted* and see.

USE Northwind

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

SELECT HireDate

FROM dbo.Employees

WHERE EmployeeID = 1

-- results HireDate as 5/2/1992

As you expected, *Connection2* can see the record that is being modified by *Connection1* which is an uncommitted record. This is called *dirty-*reading. You can expect higher level of concurrency by setting the *Isolation Level* to *Read Uncommitted* but you may face all concurrency related problems. Imagine the consequences when *Connection1* rolls back the transaction but *Connection2* makes a decision from the result before the roll back.

## Read Committed Isolation Level

This is the default *Isolation Level* of SQL Server. This eliminates *dirty-reads* but all other concurrency related problems. You have already seen this. Look at the sample used above. *Connection2* could not read data before the *Isolation Level* was set to *Read Uncommitted*. That is because it had been set to the default *Isolation Level* which is *Read Committed* which in turn disallowed reading uncommitted data. Though it stops *dirty-reads*, it may introduce others. Let’s take a simple example that shows *Lost Updates*.

Employee table contains data related to employee. New employee joins and record is made in the table.

USE Northwind

INSERT INTO dbo.Employees

(LastName, FirstName, Title, TitleOfCourtesy, BirthDate, HireDate)

VALUES

('Lewis', 'Jane', 'Sales Representative', 'Ms.', '03/04/1979', '06/23/2007')

This table contains a column called *Notes* that describes the employee’s education background. Data entry operators fill this column by looking at her/his file. Assume that the update code has been written as below. Note that no *Isolation Level* has been set, means default is set.

IF OBJECT\_ID(N'dbo.UpdateNotes', N'P') IS NOT NULL

BEGIN

DROP PROC dbo.UpdateNotes

END

GO

CREATE PROCEDURE dbo.UpdateNotes @EmployeeID int, @Notes ntext

AS

BEGIN

DECLARE @IsUpdated bit

BEGIN TRAN

SELECT @IsUpdated = CASE WHEN Notes IS NULL THEN 0 ELSE 1 END

FROM dbo.Employees

WHERE EmployeeID = @EmployeeID -- new record

-- The below statement added to hold the transaction for 5 seconds

-- Consider it is as a different process that do something else.

WAITFOR DELAY '00:00:5'

IF (@IsUpdated = 0)

BEGIN

UPDATE dbo.Employees

SET Notes = @Notes

WHERE EmployeeID = @EmployeeID

END

ELSE

BEGIN

ROLLBACK TRAN

RAISERROR ('Note has been alreasy updated!', 16, 1)

RETURN

END

COMMIT TRAN

END

Operator1 makes Connection1 and executes the following query.

EXEC dbo.UpdateNotes 15, 'Jane has a BA degree in English from the University of Washington.'

Within few seconds (in this case, right after Operator1 started) Operator2 makes Connection2 and executes the same with a different note, before completing the Operator1’s process.

EXEC dbo.UpdateNotes 15, 'Jane holds a BA degree in English.'

If you query the record after both processes, you will see that note that was entered by the Operator2 has been set for the record. Operator1 made the update and no error messages were returned to it, but it has *lost its update*. This could be avoided if the record was locked and held as soon as it was identified as a *not updated* record. But obtaining and holding a lock is not possible with *Read Committed* *Isolation Level*. Because of this, concurrency related problems such as *Lost Updates, Nonrepeatable reads* and *Phantom reads* can happen with this *Isolation Level*.

## Repeatable Read Isolation Level

This *Isolation Level* addresses all concurrency related problems except *Phantom reads*. Unlike *Read Committed*, it does not release the shared lock once the record is read. It obtains the shared lock for reading and keeps till the transaction is over. This stops other transactions accessing the resource, avoiding *Lost Updates* and *Nonrepeatable reads*. Change the *Isolation Level* of the stored procedure we used for *Read Committed* sample.

IF OBJECT\_ID(N'dbo.UpdateNotes', N'P') IS NOT NULL

BEGIN

DROP PROC dbo.UpdateNotes

END

GO

CREATE PROCEDURE dbo.UpdateNotes @EmployeeID int, @Notes ntext

AS

BEGIN

DECLARE @IsUpdated bit

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRAN

SELECT @IsUpdated = CASE WHEN Notes IS NULL THEN 0 ELSE 1 END

FROM dbo.Employees

WHERE EmployeeID = @EmployeeID -- new record

Now make two connections and execute below queries just as you did with *Read Committed* sample. Make sure you set the *Note* column value back to *NULL* before executing them.

With Connection1;

EXEC dbo.UpdateNotes 15, 'Jane has a BA degree in English from the University of Washington.'

With Connection2;

EXEC dbo.UpdateNotes 15, 'Jane holds a BA degree in English.'

Once you execute the code with Connection2, SQL Server will throw 1205 error and Connection2 will be a deadlock victim. This is because, Connection1 obtain and hold the lock on the resource until the transaction completes, stopping accessing the resource by others, avoiding *Lost Updates*. Note that setting *DEADLOCK\_PRIORITY* to *HIGH,* you can choose the *deadlock victim*.

Since the lock is held until the transaction completes, it avoids *Nonrepeatable Reads* too. See the code below.

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRAN

SELECT Notes

FROM dbo.Employees

WHERE EmployeeID = 10

It reads a record from the *Employees* table. The set *Isolation Level* guarantees the same result for the query anywhere in the transaction because it holds the lock without releasing, avoiding modification from others. It guarantees consistency of the information and no *Nonrepeatable reads*.

Now let’s take another simple example. In this case, we add one new table called *Allowances* and one new column to *Employees* table called *IsBirthdayAllowanceGiven*. The code for changes are as below;

USE Northwind

GO

-- table holds allowances

CREATE TABLE Allowances (EmployeeID int, MonthAndYear datetime, Allowance money)

GO

-- additional column that tells whether the birthday allowance is given or not

ALTER TABLE dbo.Employees

ADD IsBirthdayAllowanceGiven bit DEFAULT(0) NOT NULL

GO

Assume that company pays an additional allowance for employees whose birth date fall on current month. The below stored procedure inserts allowances for employees whose birth date fall on current month and update employees record. Note that *WAITFOR DELAY* has been added hold the transaction for few seconds in order to see the problem related to it. And no *Isolation Level* has been set, default applies.

IF OBJECT\_ID(N'dbo.AddBirthdayAllowance', N'P') IS NOT NULL

BEGIN

DROP PROC dbo.AddBirthdayAllowance

END

GO

CREATE PROC dbo.AddBirthdayAllowance

AS

BEGIN

BEGIN TRAN

-- inserts records to allowances table

INSERT INTO Allowances

(EmployeeID, MonthAndYear, Allowance)

SELECT EmployeeID, getdate(), 100.00

FROM dbo.Employees

WHERE IsBirthdayAllowanceGiven = 0

AND MONTH(BirthDate) = MONTH(getdate())

-- hold the transaction for 5 seconds

-- Consider this is as some other process that takes 5 seconds

WAITFOR DELAY '00:00:05'

-- update IsBirthdayAllowanceGiven column in Employees table

UPDATE dbo.Employees

SET IsBirthdayAllowanceGiven = 1

WHERE IsBirthdayAllowanceGiven = 0

AND MONTH(BirthDate) = MONTH(getdate())

COMMIT TRAN

END

Before running any queries, make sure at least one employee’s birth date falls on current month. Now open a new connection (let’s name it as Connection1) and run the stored procedure. In my Northwind database, I have one record that stratifies the criteria; EmployeeId 6: Michael Suyama.

USE Northwind

GO

EXEC dbo.AddBirthdayAllowance

Immediately, open Connection2 and insert a new employee whose birth date falls into current month.

USE Northwind

GO

INSERT INTO dbo.Employees

(LastName, FirstName, Title, TitleOfCourtesy, BirthDate, HireDate)

VALUES

('Creg', 'Alan', 'Sales Representative', 'Ms.', '07/13/1980', '07/20/2007')

Go back to Connection2. Once the transaction completed, query the *Allowances* table and see. You will see a one record that is generated for *Michael*. Then open the *Employees* table and see that how many records have been updated. It has updated two, not only *Michael* but *Alan*. Note that no record has been inserted to the *Allowances* table for *Alan*. In this case, the new record is considered as a *Phantom record* and read of the new record called as *Phantom* Read. This cannot be avoided with default *Isolation Level* that is *Read Committed*. Change the stored procedure and set the *Isolation Level* as *Repeatable Read*.

IF OBJECT\_ID(N'dbo.AddBirthdayAllowance', N'P') IS NOT NULL

BEGIN

DROP PROC dbo.AddBirthdayAllowance

END

GO

CREATE PROC dbo.AddBirthdayAllowance

AS

BEGIN

SET TRANSACTION ISOLATION LEVEL REPEATABLE READ

BEGIN TRAN

-- inserts records to allowances table

INSERT INTO Allowances

(EmployeeID, MonthAndYear, Allowance)

SELECT EmployeeID, getdate(), 100.00

FROM dbo.Employees

WHERE IsBirthdayAllowanceGiven = 0

AND MONTH(BirthDate) = MONTH(getdate())

Now bring the *Employees* table to original state.

UPDATE dbo.Employees

SET IsBirthdayAllowanceGiven = 0

DELETE dbo.Employees

WHERE FirstName = 'Alan'

DELETE dbo.Allowances

Open two connections again and try the same. Check the result. Still the *Phantom Reads* problem exists. In order to avoid this problem, you need to use highest *Isolation Level* that is *Serializable*.

## Serializable Isolation Level

This is the highest *Isolation Level* and it avoids all the concurrency related problems. The behavior of this level is just like the *Repeatable Read* with one additional feature. It obtains key range locks based on the filters that have been used. It locks not only current records that stratify the filter but new records fall into same filter. Change the stored procedure we used for above sample and set the *Isolation Level* as *Serializable*.

IF OBJECT\_ID(N'dbo.AddBirthdayAllowance', N'P') IS NOT NULL

BEGIN

DROP PROC dbo.AddBirthdayAllowance

END

GO

CREATE PROC dbo.AddBirthdayAllowance

AS

BEGIN

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

BEGIN TRAN

-- inserts records to allowances table

INSERT INTO Allowances

(EmployeeID, MonthAndYear, Allowance)

SELECT EmployeeID, getdate(), 100.00

FROM dbo.Employees

WHERE IsBirthdayAllowanceGiven = 0

AND MONTH(BirthDate) = MONTH(getdate())

Run the clean up code again to bring the *Employees* table to the original state.

Now test the stored procedure and *INSERT* statement with two connections. You will notice that *INSERT* operation is blocked until Connection1 completes the transaction, avoiding *Phantom Reads*.

Run the clean up code again and drop the new table *Allowances* and added column *IsBirthdayAllowanceGiven* in the *Employees* table.

Whenever we set the *Isolation Level* to a transaction*,* SQL Server makes sure that the transaction is not disturbed by other transactions. This is called concurrency control. All the *Isolation Levels* we discussed so far come under a control called *Pessimistic Control*. The *Pessimistic control,* SQL Server locks the resource until user performs the action she/he needs and then release for others. The other concurrency control is *Optimistic Control*. Under *Optimistic Control,* SQL Server does not hold locks but once read, check for inconsistency for next read. The two newly introduced *Isolation Levels* with SQL Server 2005 are *Snapshot* and *Read Committed Snapshot*. These two *Isolation Levels* provide *Optimistic Control* and they use *Row Versioning*.

## 

## Snapshot Isolation Level

The *Snapshot Isolation Level* works with *Row Versioning* technology. Whenever the transaction requires a modification for a record, SQL Server first stores the consistence version of the record in the tempdb. If another transaction that runs under *Snapshot Isolation Level* requires the same record, it can be taken from the version store. This *Isolation Level* prevents all concurrency related problems just like *Serializable Isolation Level*, in addition to that it allows multiple updates for same resource by different transactions concurrently.

Since there is a performance impact with *Snapshot Isolation Level* it has been turned off by default. The impact is explained below with the sample. You can enable it by altering the database.

ALTER DATABASE Northwind SET ALLOW\_SNAPSHOT\_ISOLATION ON

Let’s look at a simple sample. Make sure you have enabled *Snapshot Isolation Level* in the database before running below query. Open a new connection (Connection1) and execute query below;

USE Northwind

BEGIN TRAN

-- update the HireDate from 5/1/1992 to 5/2/1992

UPDATE dbo.Employees

SET HireDate = '5/2/1992'

WHERE EmployeeID = 1

Now open the second connection (Connection2) and try to retrieve the same record.

SELECT \*

FROM dbo.Employees

WHERE EmployeeID = 1

As you have seen with examples discussed under other levels, the record cannot be retrieved. Since we have enabled *Snapshot Isolation Level* in the database, SQL Server stores version of the record. Use below dynamic management view for retrieving versions stored in the store.

SELECT \* FROM sys.dm\_tran\_version\_store;

You will see one record in the store. Now set the *Isolation Level* of the Connection2 as *Snapshot* and try to retrieve the record.

SET TRANSACTION ISOLATION LEVEL SNAPSHOT

BEGIN TRAN

SELECT \*

FROM dbo.Employees

WHERE EmployeeID = 1

This returns record from the store that was the last consistence version of the record. Note that *HireDate* of the employee is *05/01/1992* not *05/02/1992*. Now go back to the Connection1 and commit the transaction.

COMMIT TRAN

Again open the Connection2 and execute the query. Note that even though the Connection1 has committed the change, Connection2 still gets the older record. This is because it was the consistence record in the version store when the Connection2 started the transaction and the same version is read during the transaction. SQL Server keeps this version of the record until no reference for it. If another transaction starts changing same record, another version will be stored and goes on; results longer link list in the version store. Maintaining longer link list and traversing through list will impact the performance. Committing the transaction in Connection2 will remove the reference for the first version and the first version in the store will be removed from separate clean-up process.

There is another great feature with *Snapshot Isolation Level*. It is *Conflict Detection*. One transaction reads a record from the version store and later tries to update the record. Another transaction updates the same record before previous transaction’s update. This conflict detects by the SQL Server and aborts the previous transaction.

Open a connection (Connection1) and run the below query. The update statement causes to add the current consistence version to the version store.

USE Northwind

BEGIN TRAN

-- update the HireDate from 5/1/1992 to 5/2/1992

UPDATE dbo.Employees

SET HireDate = '5/2/1992'

WHERE EmployeeID = 1

Open the second connection (Connection2) and read the same record. Note the *Isolation Level*.

USE Northwind

GO

SET TRANSACTION ISOLATION LEVEL SNAPSHOT

BEGIN TRAN

SELECT \*

FROM dbo.Employees

WHERE EmployeeID = 1

Go back to Connection1 and commit the transaction.

COMMIT TRAN

Go back to Connection2 and try to update the record. Note that the current transaction still runs. Whenever you execute the *UPDATE* statement, SQL Server detects the modification that has been done by Connection1 in between read and write, it throws an error.

UPDATE dbo.Employees

SET HireDate = '5/3/1992'

WHERE EmployeeID = 1

Snapshot isolation transaction aborted due to update conflict. You cannot use snapshot isolation to access table 'dbo.Employees' directly or indirectly in database 'Northwind' to update, delete, or insert the row that has been modified or deleted by another transaction. Retry the transaction or change the isolation level for the update/delete statement.

Once the conflict is detected, it terminates the transaction in Connection2. Though this *Isolation Level* has some great advantageous, this level is not recommended for a database that has many updates. This is suitable for database that is mainly used for read data with occasional updates.

## 

## Read Committed Snapshot Isolation Level

This is the new implementation of the *Read Committed Isolation Level*. It has to be set not at session/connection level but database level. The only different between *Read Committed* and *Read Committed Snapshot* is, *Read Committed Snapshot* is *Optimistic* whereas *Read Committed* is *Pessimistic*. The *Read Committed Snapshot* differs from *Snapshot* in two ways; Unlike *Snapshot*, it always returns latest consistence version and no conflict detection.

Let’s test this out. First, enable the *Isolation Level*.

ALTER DATABASE Northwind SET READ\_COMMITTED\_SNAPSHOT ON

Now open a new connection (Connection1) and run the below query.

USE Northwind

BEGIN TRAN

-- update the HireDate from 5/1/1992 to 5/2/1992

UPDATE dbo.Employees

SET HireDate = '5/2/1992'

WHERE EmployeeID = 1

This makes a last consistence version in the version store. Now open the second connection (Connection2) and try to retrieve the record.

USE Northwind

GO

BEGIN TRAN

SELECT \*

FROM dbo.Employees

WHERE EmployeeID = 1

You get a record from the version store. The value for the *HireDate* will be the last consistence value that is *05/01/1992*. Go back to Connection1 and commit the transaction.

COMMIT TRAN

In Connection1, execute the *SELECT* statement again. Unlike *Snapshot* the latest consistence is returned that has the *HireDate* as *05/02/1992*. Commit the Connection2 transaction too.

Since the maintaining old versions are not necessary with this level, there will be no impact for performance like *Snapshot* but all the concurrency related problems except *dirty reads* can happen.

Finally, let’s summarize. The below table depicts importance points of each level.

|  | **Dirty Reads** | **Lost Updates** | **Non-repeatable reads** | **Phantom reads** | **Concurrency model** | **Conflict Detection** |
| --- | --- | --- | --- | --- | --- | --- |
| **Read Uncommitted** | Yes | Yes | Yes | Yes | Pessimistic | No |
| **Read Committed** | No | Yes | Yes | Yes | Pessimistic | No |
| **Repeatable Read** | No | No | No | Yes | Pessimistic | No |
| **Serializable** | No | No | No | No | Pessimistic | No |
| **Snapshot** | No | No | No | No | Optimistic | Yes |
| **Read Committed Snapshot** | No | Yes | Yes | Yes | Optimistic | No |

**Wat 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.

**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. |

**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.)

**What is a NOLOCK?**

Using the NOLOCK query optimizer hint is generally considered good practice in order to improve concurrency on a busy system. When the NOLOCK hint is included in a SELECT statement, no locks are taken when data is read. The result is a Dirty Read, which means that another process could be updating the data at the exact time you are reading it. There are no guarantees that your query will retrieve the most recent data. The advantage to performance is that your reading of data will not block updates from taking place, and updates will not block your reading of data. SELECT statements take Shared (Read) locks. This means that multiple SELECT statements are allowed simultaneous access, but other processes are blocked from modifying the data. The updates will queue until all the reads have completed, and reads requested after the update will wait for the updates to complete. The result to your system is delay(blocking).

**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. |

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

**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.

###### STORED PROCEDURES

**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.

**What are the different types of Storage Procedure?**

* 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.
* 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.)
* 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.
* User stored procedure

**What is the advantage of using stored procedure over the SQL queries?**

Writing the SQL statements inside our code is usually not a good idea. In this way you expose your database schema (design) in the code which may be changed. Hence most of the time programmers use stored procedures instead of plain SQL statements. A stored procedure is a precompiled executable object that contains one or more SQL statements. Hence you can replace your complex SQL statements with a single stored procedure. Since, stored procedures are precompiled objects they execute faster at the database server. Most of the time, stored procedures contain more than one command; in this case, the time to pass the individual commands to the database server from the program is saved. The database is issued just one command (to execute the stored procedure) and the DB server executes all the commands and returns the result in the end. Hence, the overall interaction time with the DB server reduces in a great deal. This can result in a huge optimization in case where the DB server is accessed via a slow network.

**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')

**What is the difference between Function and Stored Procedure?**

UDF can be used in the SQL statements anywhere in the WHERE/HAVING/SELECT section where as Stored procedures cannot be.

UDFs that return tables can be treated as another rowset. This can be used in JOINs with other tables.

Inline UDF's can be though of as views that take parameters and can be used in JOINs and other Rowset operations.

**How can you optimize a stored procedure?**

**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.

**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:

* The stored procedure in the master database.
* The stored procedure based on any qualifiers provided (database name or owner).
* 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.

**Why do stored procedures reduce network traffic ?**

When a stored procedure is called, only the procedure call is sent to the server and not the statements that the procedure contains.

**Can a stored procedure call itself or recursive stored procedure? How many level SP nesting possible?**

Yes. Because Transact-SQL supports recursion, you can write stored procedures that call themselves. Recursion can be defined as a method of problem solving wherein the solution is arrived at by repetitively applying it to subsets of the problem. A common application of recursive logic is to perform numeric computations that lend themselves to repetitive evaluation by the same processing steps. Stored procedures are nested when one stored procedure calls another or executes managed code by referencing a CLR routine, type, or aggregate. You can nest stored procedures and managed code references up to 32 levels.

**What can cause a Stored procedure execution plan to become invalidated and/or fall out of cache?**

* Server restart
* Plan is aged out due to low use
* DBCC FREEPROCCACHE (sometime desired to force it)

**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).

**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.

**How to find out which stored procedure is recompiling? How to stop stored procedures from recompiling?**

**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.

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.

**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>

**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.

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?

**What do you do when one procedure is blocking the other?  
\*\***

**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.

**What are the differences between RAW, AUTO and Explicit modes in retrieving data from SQL Server in XML format?  
\*\***

**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

**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.

**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.

**What is the 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”

**What is User Defined Functions?**

User-Defined Functions allow to define its own T-SQL functions that can accept 0 or more parameters and return a single scalar data value or a table data type.

**What kind of User-Defined Functions can be created?**

There are three types of User-Defined functions in SQL Server 2000 and they are Scalar, Inline Table-Valued and Multi-statement Table-valued.

* *Scalar User-Defined Function*

A Scalar user-defined function returns one of the scalar data types. Text, ntext, image and timestamp data types are not supported. These are the type of user-defined functions that most developers are used to in other programming languages. You pass in 0 to many parameters and you get a return value.

* *Inline Table-Value User-Defined Function*

An Inline Table-Value user-defined function returns a table data type and is an exceptional alternative to a view as the user-defined function can pass parameters into a T-SQL select command and in essence provide us with a parameterized, non-updateable view of the underlying tables.

* *Multi-statement Table-Value User-Defined Function*

A Multi-Statement Table-Value user-defined function returns a table and is also an exceptional alternative to a view as the function can support multiple T-SQL statements to build the final result where the view is limited to a single SELECT statement. Also, the ability to pass parameters into a TSQL select command or a group of them gives us the capability to in essence create a parameterized, non-updateable view of the data in the underlying tables. Within the create function command you must define the table structure that is being returned. After creating this type of user-defined function, it can be used in the FROM clause of a T-SQL command unlike the behavior found when using a stored procedure which can also return record sets.

**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.

* Scalar Functions  
  Functions are scalar-valued if the RETURNS clause specified one of the scalar data types
* Inline Table-valued Functions  
  If the RETURNS clause specifies TABLE with no accompanying column list, the function is an inline function.
* 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.

**What are the difference between a function and a stored procedure?**

* + Functions can be used in a select statement where as procedures cannot
  + Procedure takes both input and output parameters but Functions takes only input parameters
  + Functions cannot return values of type text, ntext, image & timestamps where as procedures can
  + 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.

**How to debug a stored procedure?**

***TRIGGERS AND VIEWS***

**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.

**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

**What are the various uses of database triggers?**

Database triggers can be used to enforce business rules, to maintain derived values and perform value-based auditing.

**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.

**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.'

**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.

**Difference between Store Procedure and Trigger?**

* we can call stored procedure explicitly.

but trigger is automatically invoked when the action defined in trigger is done.

ex: create trigger after Insert on

* this trigger invoked after we insert something on that table.
* Stored procedure can't be inactive but trigger can be Inactive.
* Triggers are used to initiate a particular activity after fulfilling certain condition.It need to define and can be enable and disable according to need.

**Which virtual table does a trigger use?**

Inserted and Deleted.

**List few advantages of Stored Procedure.**

* Stored procedure can reduced network traffic and latency, boosting application performance.
* Stored procedure execution plans can be reused, staying cached in SQL Server's memory, reducing server overhead.
* Stored procedures help promote code reuse.
* Stored procedures can encapsulate logic. You can change stored procedure code without affecting clients.
* Stored procedures provide better security to your data.

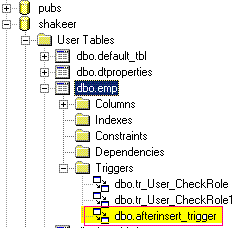
**What is the advantage to use trigger in your PL?**

Triggers are fired implicitly on the tables/views on which they are created. There are various advantages of using a trigger. Some of them are:

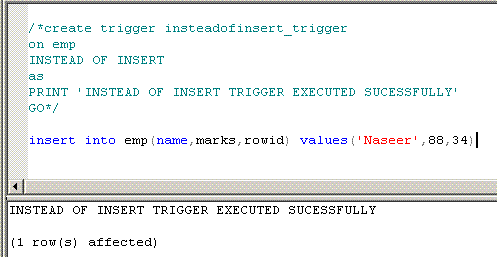
* Suppose we need to validate a DML statement(insert/Update/Delete) that modifies a table then we can write a trigger on the table that gets fired implicitly whenever DML statement is executed on that table.
* Another reason of using triggers can be for automatic updation of one or more tables whenever a DML/DDL statement is executed for the table on which the trigger is created.
* Triggers can be used to enforce constraints. For eg : Any insert/update/ Delete statements should not be allowed on a particular table after office hours. For enforcing this constraint Triggers should be used.
* Triggers can be used to publish information about database events to subscribers. Database event can be a system event like Database startup or shutdown or it can be a user even like User loggin in or user logoff.

**What are the different types of triggers?**

Triggers are of 3 types in SQL Server 2005:  
1.    DML Triggers  
.         AFTER Triggers  
.         INSTEAD OF Triggers  
2.    DDL Triggers  
3.    CLR Triggers  
  
Note:DDL and CLR Triggers cannot work in SQL Server 2000  
DML Trigger:-These Trigger is fired only when INSERT, UPDATE, and DELETE Statement occurs in table.  
  
Explanation on DML Trigger:  
Let us create a Table and insert some records in that Table.  
1) After Triggers:  
After Triggers can be created in 3 ways.  
1)    After INSERT  
2)    After UPDATE  
3)    After DELETE*1) creating After INSERT Trigger:-*Syntax:  
create trigger triggername  
on tablename  
AFTER INSERT  
As  
[SQL Statement/PRINT command]  
GO  
  
Eg:   
create trigger afterinsert\_trigger  
on emp  
AFTER INSERT  
as  
PRINT 'AFTER TRIGGER EXECUTED SUCESSFULLY'  
GO  
  
When you execute the afterinsert\_trigger it gives message as 'The Command(s) created successfully'  
You can see the is trigger is created.



**Now insert one record in a emp table. You can see the trigger will be fired automatically when the row is inserted in a table successfully.**

****

**Creating AFTER UPDATE TRIGGER:-**

**create trigger *afterupdate*\_*trigger***

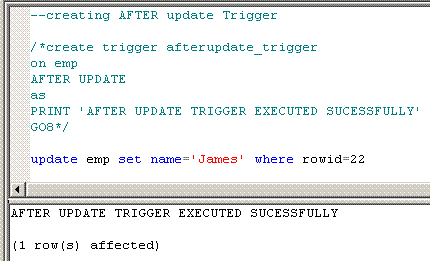
**on *emp***

**AFTER UPDATE**

**as**

**PRINT 'AFTER UPDATE TRIGGER EXECUTED SUCESSFULLY'**

**GO**

****

**Creating AFTER DELETE TRIGGER:**

**Create trigger *afterdelete*\_*trigger***

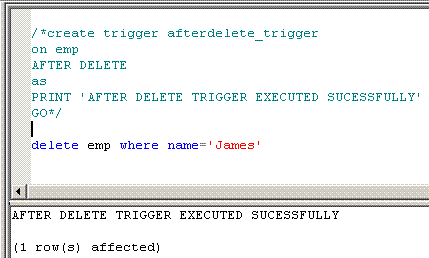
**On *emp***

**AFTER DELETE**

**as**

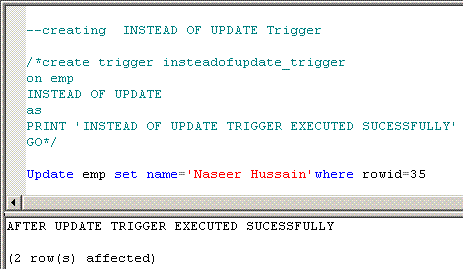
**PRINT 'AFTER DELETE TRIGGER EXECUTED SUCESSFULLY'**

**GO**

****

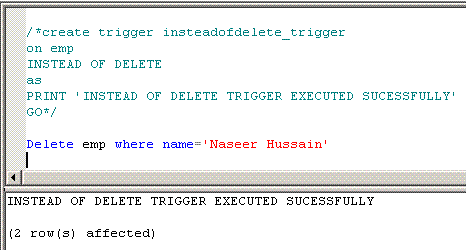
**Instead Of Update Trigger**

**Creating INSTEAD OF UPDATE TRIGGER:-  
  
create trigger *insteadofupdate*\_*trigger*  
on *emp*  
INSTEAD OF UPDATE  
as  
PRINT 'INSTEAD OF UPDATE TRIGGER EXECUTED SUCESSFULLY'  
GO**

****

**Instead of Delete Trigger**

**Creating INSTEAD OF DELETE TRIGGER:-  
  
create trigger insteadofdelete\_trigger  
on emp  
INSTEAD OF DELETE  
as  
PRINT 'INSTEAD OF DELETE TRIGGER EXECUTED SUCESSFULLY'  
GO**

****

**HOW TO Drop a Trigger?**

***Syntax: DROP TRIGGER [triggername]***

***Eg: DROP TRIGGER afterinsert\_trigger***

***VIEW***

**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

**Does the View occupy memory space?**No

**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.

**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.)

***TRANSACTIONS AND OTHERS***

**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.

**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.  
\*\*

**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?**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

**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

**There are 50 columns in a table. Write a query to get first 25 columns**Ans: Need to mention each column names.

**How to list all the tables in a particular database?**USE pubs  
GO  
sp\_help

**What is cursors?**

Cursor is a database object used by applications to manipulate data in a set on a row-by-row basis, instead of the typical SQL commands that operate on all the rows in the set at one time.

In order to work with a cursor we need to perform some steps in the following order:

* Declare cursor
* Open cursor
* Fetch row from the cursor
* Process fetched row
* Close cursor
* Deallocate cursor

**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:

* 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
* 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.

**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.

**What is DATEPART?**Returns an integer representing the specified datepart of the specified date.

**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.

**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?  
\*\***

**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

**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.

**What is DML?**Data Manipulation Language (DML), which is used to select, insert, update, and delete data in the objects defined using DDL

**What is RDBMS?**

Relational Data Base Management Systems (RDBMS) are database management systems that maintain data records and indices in tables. Relationships may be created and maintained across and among the data and tables. In a relational database, relationships between data items are expressed by means of tables. Interdependencies among these tables are expressed by data values rather than by pointers.

This allows a high degree of data independence. An RDBMS has the capability to recombine the data items from different files, providing powerful tools for data usage.

**What are the properties of the Relational tables?**

Relational tables have six properties:

* Values are atomic.
* Column values are of the same kind.
* Each row is unique.
* The sequence of columns is insignificant.
* The sequence of rows is insignificant.
* Each column must have a unique name.

**Explain the different types of keys.**

Primary key:- The attribute or combination of attributes that uniquely identifies a row or record.

Foreign Key:- an attribute or combination of attribute in a table whose value match a primary key in another table.

Composite key:- A primary key that consists of two or more attributes is known as composite key

Candidate key:- is a column in a table which has the ability to become a primary key.

Alternate Key:- Any of the candidate keys that is not part of the primary key is called an alternate key. Alternate Key or Unique Key is similar to primary key, except it accepts null Values .So that the records can still be entered submitting null values to this attribute.

**What are keys in RDBMS? What is primary key and 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. Primary keys are the unique identifiers for each row. They must contain unique values and cannot be null. Due to their importance in relational databases, Primary keys are the most fundamental of all keys and constraints. A table can have only one Primary key.

*Foreign keys* are attributes of one table that have matching values in a **primary key** in another table, allowing for relationships between tables.

Foreign keys are both a method of ensuring data integrity and a manifestation of the relationship between tables.

**What are the advantages and disadvantages of primary key and foreign key in SQL?**

*Primary key*

Advantages

1) It is a unique key on which all the other candidate keys are functionally dependent

Disadvantage

1) There can be more than one keys on which all the other attributes are dependent on.

*Foreign Key*

Advantage

1)It allows referencing another table using the primary key for the other table

Which date function is used to find the difference between two dates?

datediff

for Eg: select datediff (dd,'2-06-2007','7-06-2007')

output is 5

**What is data integrity? Explain constraints?**

Data integrity is an important feature in SQL Server. When used properly, it ensures that data is accurate, correct, and valid. It also acts as a trap for otherwise undetectable bugs within applications.

* A PRIMARY KEY constraint is a unique identifier for a row within a database table. Every table should have a primary key constraint to uniquely identify each row and only one primary key constraint can be created for each table. The primary key constraints are used to enforce entity integrity.
* A UNIQUE constraint enforces the uniqueness of the values in a set of columns, so no duplicate values are entered. The unique key constraints are used to enforce entity integrity as the primary key constraints.
* A FOREIGN KEY constraint prevents any actions that would destroy links between tables with the corresponding data values. A foreign key in one table points to a primary key in another table. Foreign keys prevent actions that would leave rows with foreign key values when there are no primary keys with that value. The foreign key constraints are used to enforce referential integrity.
* A CHECK constraint is used to limit the values that can be placed in a column. The check constraints are used to enforce domain integrity.
* A NOT NULL constraint enforces that the column will not accept null values. The not null constraints are used to enforce domain integrity, as the check constraints.

**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.

**How to implement one-to-one, one-to-many and many-to-many relationships while designing tables?**

One-to-One relationship can be implemented as a single table and rarely as two tables with primary and foreign key relationships.

One-to-Many relationships are implemented by splitting the data into two tables with primary key and foreign key relationships.

Many-to-Many relationships are implemented using a junction table with the keys from both the tables forming the composite primary key of the junction table.

**Define candidate key, alternate key, and 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.

**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.

**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.

**What is Query optimization? How is tuning a performance of query done?**

**What is the use of trace utility?  
\*\***

**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**.

**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.

**If the performance of the query suddenly decreased where you will check?**

**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'' ')

**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)

**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.

**ANY?**USE pubs  
SELECT au\_lname, au\_fname  
FROM authors  
WHERE city = ANY  
(SELECT city  
FROM publishers)

**to select date part only**SELECT CONVERT(char(10),GetDate(),101)  
--to select time part only  
SELECT right(GetDate(),7)

**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.

**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 eliminate 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.

* + **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 |

* + **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.

* + **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 |

* + **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.]
  + **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 |
| --- | --- | --- | --- | --- |

* + 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 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

* + **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.
  + **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.

**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?**

**What is de-normalization and when would you go for it?**As the name indicates, de-normalization 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.

**Explain normalization.**

Normalization means refining the redundancy and maintains stabilization. there are four types of normalization : first normal forms, second normal forms, third normal forms and fourth Normal forms.

**What is De-normalization?**

De-normalization is the process of attempting to optimize the performance of a database by adding redundant data. It is sometimes necessary because current DBMSs implement the relational model poorly. A true relational DBMS would allow for a fully normalized database at the logical level, while providing physical storage of data that is tuned for high performance. De-normalization is a technique to move from higher to lower normal forms of database modeling in order to speed up database access.

**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()

**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.

**Have you worked on Dynamic SQL? How will You handled “ (Double Quotes) in Dynamic SQL?**

**How to find dependents of a table?**Verify dependencies with **sp\_depends** before dropping an object

**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.

**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

**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.

**What is the row size in SQL Server 2000?**8060 bytes.

**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

**B-tree indexes or doubly-linked lists?**

**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().

**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

**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 **\*\***

**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.

**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.

**What is #, ##, @, @@ means?**@@ - System variables  
@ - user defined variables

**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)

**What is covered queries in SQL Server?**

**What is HASH JOIN, MERGE JOIN?**

***TOOLS AND PERMISSIONS***

**What is the use of DBCC commands?**

DBCC stands for database consistency checker. We use these commands to check the consistency of the databases, i.e., maintenance, validation task and status checks.

E.g. DBCC CHECKDB - Ensures that tables in the db and the indexes are correctly linked.

DBCC CHECKALLOC - To check that all pages in a db are correctly allocated.

DBCC CHECKFILEGROUP - Checks all tables file group for any damage.

**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. |  |

**How do you use DBCC statements to monitor various aspects of a SQL server installation?  
\*\***

**What is the output of DBCC Showcontig statement?**Displays fragmentation information for the data and indexes of the specified table.

**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

**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 |

**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.

**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.

**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"]

**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

**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.

**How will I export database?**Through DTS - Import/Export wizard  
Backup - through Complete/Differential/Transaction Log

**How to export database at a particular time, every week?**Backup - Schedule  
DTS - Schedule  
Jobs - create a new job

**How do you load large data to the SQL server database?**bcp

**How do you transfer data from text file to database (other than DTS)?**bcp

**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.

**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***

**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?  
\*\***

**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.

**Describe Application Role and explain a scenario when you will use it?  
\*\***

**After removing a table from database, what other related objects have to be dropped explicitly?**(view, SP)

**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.  
\*\***

**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.

* + Windows Authentication Mode
  + SqlServer Authentication Mode
  + Mixed Authentication Mode

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

**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)

**What are three SQL keywords used to change or set someone’s permissions?**  
Grant, Deny and Revoke

***ADMIN***

**Explain the architecture of SQL Server?  
\*\***

**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.

**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.  
\*\*

**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:

* 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.
* 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.
* 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.

**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**.

**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. 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.

**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)

**What is Collation?**

Collation refers to a set of rules that determine how data is sorted and compared. Character data is sorted using rules that define the correct character sequence, with options for specifying casesensitivity, accent marks, kana character types and character width.

**What are different type of Collation Sensitivity?**

*Case sensitivity*

A and a, B and b, etc.

*Accent sensitivity*

a and á, o and ó, etc.

*Kana Sensitivity*

When Japanese kana characters Hiragana and Katakana are treated differently, it is called Kana sensitive.

*Width sensitivity*

When a single-byte character (half-width) and the same character when represented as a double-byte character (full-width) are treated differently then it is width sensitive.

**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).

**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

**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

**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.

**Fundamentals of Data warehousing & olap?**

**What do u mean by OLAP server? What is the difference between OLAP and OLTP?**

**What is a tuple?**A **tuple** is an instance of data within a relational database.

**Services and user Accounts maintenance**

**sp\_configure commands?**Displays or changes global configuration settings for the current server.

**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.

**How to rebuild Master Database?**

Shutdown Microsoft SQL Server 2000, and then run Rebuildm.exe. This is located in the Program Files\Microsoft SQL Server\80\Tools\Binn directory.

In the Rebuild Master dialog box, click Browse.

In the Browse for Folder dialog box, select the \Data folder on the SQL Server 2000 compact disc or in the shared network directory from which SQL Server 2000 was installed, and then click OK.

Click Settings. In the Collation Settings dialog box, verify or change settings used for the master database and all other databases.

Initially, the default collation settings are shown, but these may not match the collation selected during setup. You can select the same settings used during setup or select new collation settings. When done, click OK.

In the Rebuild Master dialog box, click Rebuild to start the process.

The Rebuild Master utility reinstalls the master database.

To continue, you may need to stop a server that is running.

Source: <http://msdn2.microsoft.com/en-us/library/aa197950(SQL.80).aspx>

**What is the basic functions for master, msdb, model, tempdb databases?**

The Master database holds information for all databases located on the SQL Server instance and is the glue that holds the engine together. Because SQL Server cannot start without a functioning master database, you must administer this database with care.

The *msdb* database stores information regarding database backups, SQL Agent information, DTS packages, SQL Server jobs, and some replication information such as for log shipping.

The *tempdb* holds temporary objects such as global and local temporary tables and stored procedures. The *model* is essentially a template database used in the creation of any new user database created in the instance.

**How to get @@error and @@rowcount at the same time?**

If @@Rowcount is checked after Error checking statement then it will have 0 as the value of

@@Recordcount as it would have been reset.

And if @@Recordcount is checked before the error-checking statement then @@Error would get reset.

To get @@error and @@rowcount at the same time do both in same statement and store them in local variable. SELECT @RC = @@ROWCOUNT, @ER = @@ERROR

**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.

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

SQL Server runs on port 1433. It can be changed from the Network Utility TCP/IP properties –> Port number.both on client and the server.

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

Windows mode and mixed mode (SQL & Windows).

To change authentication mode in SQL Server click Start, Programs, Microsoft SQL Server and click SQL Enterprise Manager to run SQL Enterprise Manager from the Microsoft SQL Server program group.

Select the server then from the Tools menu select SQL Server Configuration Properties, and choose the Security page.

**Where are SQL server users names and passwords are stored in sql server?**

They get stored in master db in the sysxlogins table.

**Which command using Query Analyzer will give you the version of SQL server and operating system?**

SELECT SERVERPROPERTY('productversion'), SERVERPROPERTY ('productlevel'), SERVERPROPERTY ('edition')

**What is SQL server agent?**

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

**What is @@ERROR?**

The @@ERROR automatic variable returns the error code of the last Transact-SQL statement. If there was no error, @@ERROR returns zero. Because @@ERROR is reset after each Transact-SQL statement, it must be saved to a variable if it is needed to process it further after checking it.

**What is Raiseerror?**

Stored procedures report errors to client applications via the RAISERROR command. RAISERROR doesn't change the flow of a procedure; it merely displays an error message, sets the @@ERROR automatic variable, and optionally writes the message to the SQL Server error log and the NT application event log.

**What is log shipping?**

Log shipping is the process of automating the backup of database and transaction log files on a production SQL server, and then restoring them onto a standby server. Enterprise Editions only supports log shipping. In log shipping the transactional log file from one server is automatically updated into the backup database on the other server. If one server fails, the other server will have the same db can be used this as the Disaster Recovery plan. The key feature of log shipping is that is will automatically backup transaction logs throughout the day and automatically restore them on the standby server at defined interval.

**What is the difference between a local and a global variable?**

A local temporary table exists only for the duration of a connection or, if defined inside a compound statement, for the duration of the compound statement.

A global temporary table remains in the database permanently, but the rows exist only within a given connection. When connection are closed, the data in the global temporary table disappears. However, the table definition remains with the database for access when database is opened next time.

**What command do we use to rename a db?**

sp\_renamedb ‘oldname’ , ‘newname’

If someone is using db it will not accept sp\_renmaedb. In that case first bring db to single user using sp\_dboptions. Use sp\_renamedb to rename database. Use sp\_dboptions to bring database to multi user mode.

**What is sp\_configure commands and set commands?**

Use sp\_configure to display or change server-level settings. To change database-level settings, use ALTER DATABASE. To change settings that affect only the current user session, use the SET statement.

**What are the different types of replication? Explain.**

The SQL Server 2000-supported replication types are as follows:

* Transactional
* Snapshot
* Merge

*Snapshot replication* distributes data exactly as it appears at a specific moment in time and does not monitor for updates to the data. Snapshot replication is best used as a method for replicating data that changes infrequently or where the most up-to-date values (low latency) are not a requirement. When synchronization occurs, the entire snapshot is generated and sent to Subscribers.

*Transactional replicatio*n, an initial snapshot of data is applied at Subscribers, and then when data modifications are made at the Publisher, the individual transactions are captured and propagated to Subscribers.

*Merge replication* is the process of distributing data from Publisher to Subscribers, allowing the Publisher and Subscribers to make updates while connected or disconnected, and then merging the updates between sites when they are connected.

**What is BCP? When does it used?**

BulkCopy is a tool used to copy huge amount of data from tables and views. BCP does not copy the structures same as source to destination.

**How do you load large data to the SQL server database?**

BulkCopy is a tool used to copy huge amount of data from tables. BULK INSERT command helps to Imports a data file into a database table or view in a user-specified format.

**Can SQL Servers linked to other servers like Oracle?**

SQL Server can be lined to any server provided it has OLE-DB provider from Microsoft to allow a link.

E.g. Oracle has a OLE-DB provider for oracle that Microsoft provides to add it as linked server to SQL Server group.

**What is DataWarehousing?**

* *Subject-oriented,* meaning that the data in the database is organized so that all the data
* elements relating to the same real-world event or object are linked together;
* *Time-variant,* meaning that the changes to the data in the database are tracked and recorded so that reports can be produced showing changes over time;
* *Non-volatile,* meaning that data in the database is never over-written or deleted, once committed, the data is static, read-only, but retained for future reporting;
* *Integrated,* meaning that the database contains data from most or all of an organization's operational applications, and that this data is made consistent.

**What is OLTP(OnLine Transaction Processing)?**

In OLTP - online transaction processing systems relational database design use the discipline of data modeling and generally follow the Codd rules of data normalization in order to ensure absolute data integrity. Using these rules complex information is broken down into its most simple structures (a table) where all of the individual atomic level elements relate to each other and satisfy the normalization rules.

**How do SQL server 2000 and XML linked? Can XML be used to access data?**

FOR XML (ROW, AUTO, EXPLICIT)

You can execute SQL queries against existing relational databases to return results as XML rather than standard rowsets. These queries can be executed directly or from within stored procedures. To retrieve XML results, use the FOR XML clause of the SELECT statement and specify an XML mode of RAW, AUTO, or EXPLICIT.

*OPENXML*

OPENXML is a Transact-SQL keyword that provides a relational/rowset view over an in-memory XML document. OPENXML is a rowset provider similar to a table or a view. OPENXML provides a way to access XML data within the Transact-SQL context by transferring data from an XML document into the relational tables. Thus, OPENXML allows you to manage an XML document and its interaction with the relational environment.

**What is an execution plan? When would you use it? How would you view the execution plan?**

An execution plan is basically a road map that graphically or textually shows the data retrieval methods chosen by the SQL Server query optimizer for a stored procedure or ad-hoc query and is a very useful tool for a developer to understand the performance characteristics of a query or stored procedure since the plan is the one that SQL Server will place in its cache and use to execute the stored procedure or query. From within Query Analyzer is an option called "Show Execution Plan" (located on the Query drop-down menu). If this option is turned on it will display query execution plan in separate window when query is ran again.

**What is the difference between oracle, sql and sql server ?**

•Oracle is based on RDBMS.

•SQL is Structured Query Language.

•SQL Server is another tool for RDBMS provided by MicroSoft.

**What are the OS services that the SQL Server installation adds?**

MS SQL SERVER SERVICE, SQL AGENT SERVICE, DTC (Distribution transac co-ordinator)

**What are three SQL keywords used to change or set someone’s permissions?**

GRANT, DENY, and REVOKE.

**What does it mean to have quoted\_identifier on? What are the implications of having it off?**

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

**What is the STUFF function and how does it differ from the REPLACE function?**

STUFF function to overwrite existing characters. Using this syntax, STUFF(string\_expression, start, length, replacement\_characters), string\_expression is the string that will have characters substituted, start is the starting position, length is the number of characters in the string that are substituted, and replacement\_characters are the new characters interjected into the string.

REPLACE function to replace existing characters of all occurance. Using this syntax

REPLACE(string\_expression, search\_string, replacement\_string), where every incidence of

search\_string found in the string\_expression will be replaced with replacement\_string.

**Using query analyzer, name 3 ways to get an accurate count of the number of records in a table?**

SELECT \* FROM table1

SELECT COUNT(\*) FROM table1

SELECT rows FROM sysindexes WHERE id = OBJECT\_ID(table1) AND indid < 2

**What is Identity?**

Identity (or AutoNumber) is a column that automatically generates numeric values. A start and increment value can be set, but most DBA leave these at 1. A GUID column also generates numbers, the value of this cannot be controled. Identity/GUID columns do not need to be indexed.

**What is a Scheduled Jobs or What is a Scheduled Tasks?**

Scheduled tasks let user automate processes that run on regular or predictable cycles. User can schedule administrative tasks, such as cube processing, to run during times of slow business activity. User can also determine the order in which tasks run by creating job steps within a SQL Server Agent job. E.g. Back up database, Update Stats of Tables. Job steps give user control over flow of execution.

If one job fails, user can configure SQL Server Agent to continue to run the remaining tasks or to stop execution.

**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.

**How do we open SQL Server in single user mode?**We can accomplish this in any of the three ways given below :-

* From Command Prompt :-  
  sqlservr -m
* 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.
* 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.

**Difference between clustering and NLB (Network Load Balancing)?**\*\*

**Explain Active/Active and Active/Passive cluster configurations?  
\*\***

**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.

**What are the main steps you take care for enhancing SQL Server performance?  
\*\***

**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**

**What is a Linked Server?**

Linked Servers is a concept in SQL Server by which we can add other SQL Server to a Group and query both the SQL Server dbs using T-SQL Statements. With a linked server, you can create very clean, easy to follow, SQL statements that allow remote data to be retrieved, joined and combined with local data. Storped Procedure sp\_addlinkedserver, sp\_addlinkedsrvlogin will be used add new Linked Server.

**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 @@IDENTITY ?**Ans: Returns the last-inserted identity value.

**If a job is fail in sql server, how do find what went wrong?** **Have you used Error handling in DTS?**

| **Q:** | **What is SELECT statement?** |
| --- | --- |
| **A:** | The SELECT statement lets you select a set of values from a table in a database. The values selected from the database table would depend on the various conditions that are specified in the SQL query. |
|  | [**TOP**](http://www.allapplabs.com/interview_questions/db_interview_questions.htm#top) |

| **Q:** | **How can you compare a part of the name rather than the entire name?** |
| --- | --- |
| **A:** | SELECT \* FROM people WHERE empname LIKE '%ab%' Would return a recordset with records consisting empname the sequence '**ab**' in empname . |
|  | [**TOP**](http://www.allapplabs.com/interview_questions/db_interview_questions.htm#top) |

| **Q:** | **What is the INSERT statement?** |
| --- | --- |
| **A:** | The INSERT statement lets you insert information into a database. |
|  | [**TOP**](http://www.allapplabs.com/interview_questions/db_interview_questions.htm#top) |

|  |  |
| --- | --- |
| **Q:** | **How can I find the total number of records in a table?** |
| **A:** | You could use the COUNT keyword , example  *SELECT COUNT(\*) FROM emp WHERE age>40* |
|  | [**TOP**](http://www.allapplabs.com/interview_questions/db_interview_questions.htm#top) |

| **Q:** | **What is GROUP BY?** |
| --- | --- |
| **A:** | The GROUP BY keywords have been added to SQL because aggregate functions (like SUM) return the aggregate of all column values every time they are called. Without the GROUP BY functionality, finding the sum for each individual group of column values was not possible. |
|  | [**TOP**](http://www.allapplabs.com/interview_questions/db_interview_questions.htm#top) |

| **Q:** | **What is the difference among "dropping a table", "truncating a table" and "deleting all records" from a table**? |
| --- | --- |
| **A:** | **Dropping :**  (Table structure  + Data are deleted), Invalidates the dependent objects ,Drops the indexes  **Truncating:**  (Data alone deleted), Performs an automatic commit, Faster than delete  **Delete :** (Data alone deleted), Doesn’t perform automatic commit |
|  | [**TOP**](http://www.allapplabs.com/interview_questions/db_interview_questions.htm#top) |

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |

1. **What is a "functional dependency"? How does it relate to database table design**? - Functional dependency relates to how one object depends upon the other in the database. for example, procedure/function sp2 may be called by procedure sp1. Then we say that sp1 has functional dependency on sp2.
2. **Why can a "group by" or "order by" clause be expensive to process**? - Processing of "group by" or "order by" clause often requires creation of Temporary tables to process the results of the query. Which depending of the result set can be very expensive.
3. **What is "index covering" of a query**? - Index covering means that "Data can be found only using indexes, without touching the tables"