**SQL Server**

**What is a database?**

A database is described as an organized way of collection of DATA. It is the collection of schemes, tables, queries, reports, views and other objects.

Syntax: CREATEDATABASEDatabaseName

Example: CREATEDATABASE Student

or you can Create Database through Design/ Wizard form by right clicking on DATABASE option-New Database.

**What is DBMS?**

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

**What is SQL?**

Structured Query Language, also known as SQL, is a programming language designed for managing Relational Database Management Systems (RDBMSs). SQL is an International Organization for Standardization (ISO) standard. In RDBMS all the data is stored in tables with each table consisting of rows and columns.

Example of Sql Server 2014 SQL format:

Example of Oracle SQL format below:

Create database:

Output: Here we can see our database is created.

**What is PL/SQL?**

PL/SQL Control Statements in Oracle.

Control Statements,

• Control statements are very important in PL/SQL.

• Control Statements are elements in a program that control the flow of program execution.

• The syntax of control statements are similar to regular English and are very similar to choices that we make every day.

• Branching statements are as follows:

o If statement

o If - THEN - ELSE o Nested IF

o Branching with logical connectivity

o While

o For Loop

**What is the difference between SQL and PL/SQL?**

SQL: It is referred as Structured Query Language.

• Only simple IF / Else statements.

• Through SQL you can interact with database through ADO.NET

• In SQL you can execute a line of code

• It can run only on windows

PL/SQL: It is referred as Procedure Language / Structure Query Language:

• In PL/SQL you can execute a block of code not a single line of code.

• Deep control statements

• It can run in UNIX also.

• PL/SQL language includes object oriented programming techniques such as encapsulation, function overloading, and information hiding (all but inheritance).

**What is RDBMS?**

RDBMS: It is referred as Relation Database Management Systems (RDBMS). RDBMS possesses a set of the below given characteristics:

• Write-intensive operations: The RDBMS is frequently written to and is often used in transaction-oriented applications.

• Data in flux or historical data: The RDBMS is designed to handle frequently changing data. Alternatively, RDBMS can also store vast amounts of historical data, which can later be analyzed or "mined".

• Application-specific schema: The RDBMS is configured on a per-application basis and a unique schema exists to support each application.

• Complex data models. The relational nature of the RDBMS makes it suitable for handling sophisticated, complex data models that require many tables, foreign key values, complex join operations, and so on.

• Data integrity: The RDBMS features many components designed to ensure data integrity. This includes rollback operations, referential integrity, and transaction-oriented operations.

**What is a database table?**

Database table: Table contains records in the form of rows and columns. A permanent table is created in the database you specify and remains in the database permanently, until you delete it.

Syntax:

1. Create table TableName (ID INT, NAME VARCHAR(30) )
2. Drop syntax: drop table TableName
3. Select Syntax: Select \* from TableName

**What is a query?**

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

**What is subquery?**

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

**What are the types of subquery?**

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

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

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

**How to create a table in SQL?**

SQL provides an organized way for table creation.

Syntax: Create table TableName (columnName1 datatype, columnName2 datatype )

The following is an example of creating a simple table-

create table Info (Name varchar(20), BirthDate date,Phone nvarchar(12), City varchar(20))

**What are tables and Fields?**

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

**How to delete a table in SQL Server?**

Delete Data Record from Database Table and deleting an existing table by the following method:

Syntax: To delete all table records of a table:

Delete TableName DELETE info

**How to update a database table using SQL?**

To update an existing Table we use SQL Command UPDATE: It will update the records as per user defined query/need.

Syntax:

Update TableName SET ColumnName = NewData where Condition

Update info Set City = 'Baroda' where id = 2

**What is a database relationship?**

Relationships are created by linking the column in one table with the column in another table. There are four different types of relationship that can be created.

The relationships are listed below:

1. One to One Relationship
2. Many to One relationship
3. Many to Many relationship
4. One to One relationship

One to Many & Many to One Relationship:

For a One to many relationships, a single column value in one table has one or more dependent column values in another table. Look at the following diagram:

Many to Many Relationship:

The third table acts as a bridge between the tables that want to establish a Many to Many relationship. The bridge table stores the common information between Many to Many relationship tables. Have a look at the following diagram:

**What is a primary key of a database?**

Primary key:-

A table column with this constraint is called the key column for the table. This constraint helps the table to make sure that the value is not repeated and also that there are no null entries.

Now this column does not allow null values and duplicate values. You can try inserting values to violate these conditions and see what happens. A table can have only one Primary key. Multiple columns can participate on the primary key.

Example:

**What is a unique key?**

A Unique key constraint uniquely identified each record in the database. This provides uniqueness for the column or set of columns. A Primary key constraint has automatic unique constraint defined on it. But not, in the case of Unique Key. There can be many unique constraint defined per table, but only one Primary key constraint defined per table.

**What is a foreign key of a database?**

To define the relationship between two tables (one is called parent and the other one is the child table) connected by columns, a foreign key constraint is used. In this constraint the values of the child table must appear in the parent table, which means that for a foreign key, one table should point to a Primary Key in another table. A table can have multiple foreign keys and each foreign key can have a different referenced table.

Example: To understand the foreign key clearly let's assume the following two tables:

CUSTOMER {Cust\_ID, Cust\_Name, Age, ContactNo, Gender, Address} VENDOR {Vend\_ID, Vend\_Name, Cust\_ID}

Example: Foreign Key Constraint while using CREATE TABLE statement.

Syntax: CREATE TABLE table\_name(Col1 datatype NOT NULL, Col2 datatype NOT NULL, Col3 datatype NOT NULL, CONSTRAINT FK\_Column FOREIGN KEY(Col1, Col2, Col3) REFERENCES parent \_table(Col1, Col2, Col3) );

AT SINGLE COLUMN LEVEL

**What is database normalization?**

Database normalization is the process of organizing the fields and tables of a relational database to minimize redundancy and dependency. Normalization usually involves dividing large tables into smaller (and less redundant) tables and defining relationships among them. Normalization is a bottom-up technique for database design.

The evolution of Normalization theories is illustrated below:

• First Normal Form (1NF)

• Second Normal Form (2NF)

• Third Normal Form (3NF)

• Boyce-Codd Normal Form (BCNF)

• 4th Normal Form

• 5th Normal Form

• 6th Normal Form

**What are database normalization forms?**

Normalization is the process of organizing data into a related table. it also eliminates redundancy and increases the integrity which improves performance of the query. To normalize a database, we divide the database into tables and establish relationships between the tables.

• First Normal Form (1st NF)

• Second Normal Form (2nd NF)

• Third Normal Form (3rd NF)

• Boyce-Codd Normal Form (BCNF)

• Fourth Normal Form (4th NF)

• Fifth Normal Form (5th NF)

First Normal Form (1NF):

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

Second Normal Form (2NF):

Meeting all requirements of the first normal form. Placing the subsets of data in separate tables and Creation of relationships between the tables using primary keys. Third Normal Form (3NF):

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

Fourth Normal Form (3NF):

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

**What is Denormalization.**

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

**What is a stored procedure?**

A Stored Procedure is a collection or a group of T-SQL statements. Stored Procedures are a precompiled set of one or more statements that are stored together in the database. They reduce the network load because of the precompilation. We can create a Stored Procedure using the "Create proc" statement.

**Why we use Stored Procedure**

There are several reasons to use a Stored Procedure. They are a network load reducer and decrease execution time because they are precompiled. The most important use of a Stored Procedure is for security purposes. They can restrict SQL Injection. We can avoid SQL injection by use of a Stored Procedure.

**How to create a Stored Procedure**

CREATE PROCEDURE spEmployee AS

BEGIN

SELECT EmployeeId, Name, Gender, DepartmentName FROM tblEmployees INNER JOIN tblDepartments ON tblEmployees.EmployeeDepartmentId = tblDepartments.DepartmentId

END

Advantages of using a Stored Procedure in SQL Server

• It is very easy to maintain a Stored Procedure and they are re-usable.

• The Stored Procedure retains the state of the execution plans.

• Stored Procedures can be encrypted and that also prevents SQL Injection Attacks

**What is a function in SQL Server?**

A function is a sequence of statements that accepts input, processes them to perform a specific task and provides the output. Functions must have a name but the function name can never start with a special character such as @, $, #, and so on.

Types of function

• Pre-Defined Function

• User-Defined Function

User-defined Function:

In a user-defined function we write our logic according to our needs. The main advantage of a user-defined function is that we are not just limited to pre-defined functions. We can write our own functions for our specific needs or to simplify complex SQL code. The return type of a SQL function is either a scalar value or a table.

Creation of a function

Create function ss(@id int)

returns table as return select \* from item where itemId = @id

Execution of a Function

select \* from ss(1)

Output:

**What are the different types of functions in SQL Server?**

A function must return a result. So that is also called a function that returns a result or a value. When we create it a function must specify a value type that will return a value.

• Functions only work with select statements.

• Functions can be used anywhere in SQL, such as AVG, COUNT, SUM, MIN, DATE and so on with select statements.

• Functions compile every time.

• Functions must return a value or result.

• Functions only work with input parameters.

• Try and catch statements are not used in functions.

Function Types:

The following is the function list in SQL Server databases.

SQL Server contains the following aggregates functions:

**What is a trigger in SQL Server?**

"A Trigger is a Database object just like a stored procedure or we can say it is a special kind of Stored Procedure which fires when an event occurs in a database.".

It is a database object that is bound to a table and is executed automatically. We cannot explicitly call any trigger. Triggers provide data integrity and used to access and check data before and after modification using DDL or DML query.

There are two types of Triggers:

1. DDL Trigger
2. DML trigger

DDL Triggers: They fire in response to DDL (Data Definition Language) command events that start with Create, Alter and Drop like Create\_table, Create\_view, drop\_table, Drop\_view and Alter\_table.

Code of DDL Triggers:

create trigger saftey on database for

create\_table, alter\_table, drop\_table as print 'you can not create ,drop and alter table in this database' rollback;

Output:

DML Triggers: They fire in response to DML (Data Manipulation Language) command events that start with Insert, Update and Delete like insert\_table, Update\_view and Delete\_table.

Code of DML Trigger:

create trigger deep on emp for insert, update, delete as print 'you can notinsert,update and delete this table I' rollback;

Output:

When we insert, update or delete in a table in a database then the following message appears:

**Why do we need triggers?**

**Why and when to use a trigger:**

We use a trigger when we want some event to happen automatically on certain desirable scenarios. You have a table that changes frequently, now you want to know how many times and when these changes take place. In that case you can create a trigger that will insert the desired data into another table whenever any change in the main table occurs.

In SQL Server we can create the following 3 types of triggers:

• Data Definition Language (DDL) triggers

• Data Manipulation Language (DML) triggers

• Logon triggers

Example:

CREATE TRIGGER trgAfterInsert ON[dbo].[Employee\_Test]

FOR INSERT

AS

declare@ empid int; declare@ empname varchar(100);

declare@ empsal decimal(10, 2);

declare@ audit\_action varchar(100);

select@ empid = i.Emp\_ID from inserted i;

select@ empname = i.Emp\_Name from inserted i;

select@ empsal = i.Emp\_Sal from inserted i;

set@ audit\_action = 'Inserted Record -- After Insert Trigger.';

insert into Employee\_Test\_Audit (Emp\_ID, Emp\_Name, Emp\_Sal, Audit\_Action, Audit\_Timestamp) values(@empid, @empname, @empsal, @audit\_action, getdate());

PRINT 'AFTER INSERT trigger fired.'

GO

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

Triggers are a special type of stored procedure which is executed automatically based on the occurrence of a database event. These events can be categorized as:

• Data Manipulation Language (DML) and

• Data Definition Language (DDL) events.

The benefit derived from triggers is based in their events driven nature. Once created, the trigger automatically fires without user intervention based on an event in the database.

A. Using DML Triggers: DML triggers are invoked when any DML command such as INSERT, DELETE, and UPDATE occurs on the data of a table and/or view.

• DML triggers are powerful objects for maintaining database integrity and consistency.

• DML triggers evaluate data before it has been committed to the database. o During this evaluation the following actions are performed.

We cannot use the following commands in DML trigger:

o ALTER DATABASE

o CREATE DATABASE

o DISK DATABASE

o LOAD DATABASE

o RESTORE DATABASE

B. Using DDL Triggers:

• These triggers focus on changes to the definition of database objects as opposed to changes to the actual data.

• This type of trigger is useful for controlling development and production database environments.

Let us create DDL trigger now-

The following is the syntax.

CREATE TRIGGER trigger\_name ON{ALL SERVER | DATABASE }

[WITH < ddl\_trigger\_option > [, ...n]]{ FOR | AFTER } {

event\_type | event\_group }[, ...n] AS

{sql\_statement[;][...n] | EXTERNAL NAME < method specifier > [;] }

CREATE TRIGGER tr\_TableAudit ON DATABASE FOR CREATE\_TABLE, ALTER\_TABLE, DROP\_TABLE AS PRINT 'You must disable the TableAudit trigger in order to change any table in this database ' ROLLBACK GO

**What is a view in the database?**

A View is nothing but a select query with a name given to it or we can simply say a view is a Named Query. Ok! Why do we need a view? There can be many answers for this. Some of the important stuff I feel is:

• A view can combine data from multiple tables using adequate joins and while bringing it may require complex filters and calculated data to form the required result set. From a user's point of view, all these complexities are hidden data queried from a single table.

• Sometimes for security purposes, access to the table, table structures and table relationships are not given to the database user. All they have is access to a view not knowing what tables actually exist in the database.

• Using the view, you can restrict the user update only portions of the records.

The following are the key points to be noted about views:

1. Multiple views can be created on one table.
2. Views can be defined as read-only or updatable.
3. Views can be indexed for better performance.
4. Insert, update, delete can be done on an updatable view.

**Why do I need views in a database?**

There are a number of scenarios where we have to look for a view as a solution.

• To hide the complexity of the underlying database schema, or customize the data and schema for a set of users.

• To control access to rows and columns of data.

• To aggregate data for performance.

Views are used for security purposes because they provide encapsulation of the name of the table. Data is in the virtual table, not stored permanently. Views display only selected data.

Syntax of a View:

CREATE VIEW view\_name AS

SELECT column\_name(s) FROM table\_name WHERE condition

There are two types of views.

• Simple View

• Complex View

**What is the difference between Primary key and unique key?**

Primary key does not allow the null values but unique key allows one null value.

Primary key will create clustered index on column but unique key will create non-clustered index by default.

**How can you increase SQL performance?**

Following are tips which will increase your SQl performance:-

Every index increases the time takes to perform INSERTS, UPDATES, and DELETES, so the number of indexes should not be too 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 orders of the columns in the key are very important. Try to order the columns in the key as to enhance selectivity, with the most selective columns to the leftmost of the key.

If you want to join several tables, try to create surrogate integer keys for this purpose and create indexes on their columns. Create surrogate integer primary key (identity for example) if your table will not have many insert operations.

Clustered indexes are more preferable than nonclustered, if you need to select by a range of values or you need to sort results set with GROUP BY or ORDER BY. If your application will be performing the same query over and over on the same table, consider creating a covering index on the table.

You can use the SQL Server Profiler Create Trace Wizard with "Identify Scans of Large Tables" trace to determine which tables in your database may need indexes. This trace will show which tables are being scanned by queries instead of using an index.

**What is the use of OLAP?**

OLAP is useful because it provides fast and interactive access to aggregated data and the ability to drill down to detail.

**What is a measure in OLAP?**

Measures are the key performance indicator that you want to evaluate. To determine which of the numbers in the data might be measures. A rule of thumb is if a number makes sense when it is aggregated, then it is a measure.

**What are dimensions in OLAP?**

Dimensions are the categories of data analysis. For example, in a revenue report by month by sales region, the two dimensions needed are time and sales region. Typical dimensions include product, time, and region.

**What are levels in dimensions?**

Dimensions are arranged in hierarchical levels, with unique positions within each level. For example, a time dimension may have four levels, such as Year, Quarter, Month, and Day. Or the dimension might have only three levels, for example, Year, Week, and Day. The values within the levels are called members. For example, the years 2002 and 2003 are members of the level Year in the Time dimension.

**What are fact tables and dimension tables in OLAP?**

Twist: - Can you explain the star schema for OLAP?

The dimensions and measures are physically represented by a star schema. Dimension tables revolve around fact table. A fact table contains a column for each measure as well as a column for each dimension. Each dimension column has a foreign -key relationship to the related dimension table, and the dimension columns taken together are the key to the fact table.

**What is DTS?**

DTS is used to import data and while importing it helps us to transform and modify data. The name itself is self explanatory DTS ( Data transformation Services).

**What is fill factor ? or When does page split occurs?**

The 'fill factor' option specifies how full SQL Server will make each index page. When there is no free space to insert new row on the index page, SQL Server will create new index page and transfer some rows from the previous page to the new one. This operation is called page splits. You can reduce the number of page splits by setting the appropriate fill factor option to reserve free space on each index page. The fill factor is a value from 1 through 100 that specifies the percentage of the index page to be left empty. The default value for fill factor is 0. It is treated similarly to a fill factor value of 100, the difference in that SQL Server leaves some space within the upper level of the index tree for FILLFACTOR = 0. The fill factor percentage is used only at the time when the index is created. If the table contains read-only data (or data that very rarely changed), you can set the 'fill factor' option to 100. When the table's data is modified very often, you can decrease the fill factor to 70% or whatever you think is best.

**What is RAID and how does it work?**

Redundant Array of Independent Disks (RAID) is a term used to describe the technique of improving data availability through the use of arrays of disks and various data-striping methodologies. Disk arrays are groups of disk drives that work together to achieve higher data-transfer and I/O rates than those provided by single large drives. An array is a set of multiple disk drives plus a specialized controller (an array controller) that keeps track of how data is distributed across the drives. Data for a particular file is written in segments to the different drives in the array rather than being written to a single drive.

For speed and reliability, it is better to have more disks. When these disks are arranged in certain patterns and are use a specific controller, they are called a Redundant Array of Inexpensive Disks (RAID) set. There are several numbers associated with RAID, but the most common are 1, 5 and 10.

RAID 1 works by duplicating the same writes on two hard drives. Let us assume you have two 20-Gigabyte drives. In RAID 1, data is written at the same time to both the drives. RAID1 is optimized for fast writes.

RAID 5 works by writing parts of data across all drives in the set (it requires at least three drives). If a drive failed, the entire set would be worthless. To combat this problem, one of the drives stores a "parity" bit. Think of a math problem, such as 3 + 7 = 10. You can think of the drives as storing one of the numbers, and the 10 is the parity part. By removing any one of the numbers, you can get it back by referring to the other two, like this: 3 + X = 10. Of course, losing more than one could be evil. RAID 5 is optimized for reads.

RAID 10 is a bit of a combination of both types. It does not store a parity bit, so it is faster, but it duplicates the data on two drives to be safe. You need at least four drives for RAID 10. This type of RAID is probably the best compromise for a database server.

Note :- It's difficult to cover complete aspect of RAID in this book. It's better to take some decent SQL SERVER book for in detail knowledge, but yes from interview aspect you can probably escape with this answer.

**SQL Server Difference between @@IDENTITY, SCOPE\_IDENTITY () and IDENT\_CURRENT**

@@IDENTITY

It will return last or newly inserted record id of any table in current session but it’s not limited to current scope. In current session if any trigger or functions inserted record in any table that it will return that latest inserted record id regardless of table. We need to use this property whenever we don’t have any other functions or triggers that run automatically.

Syntax: SELECT @@IDENTITY

SCOPE\_IDENTITY()

This property will return last or newly inserted record id of table in current session or connection and it’s limited to current scope that means it will return id of newly inserted record in current session / connection stored procedure or query executed by you in current scope even we have any other functions or triggers that run automatically. Its better we can go with property whenever we need to get last or newly inserted record id in table.

Syntax: SELECT SCOPE\_IDENTITY()

IDENT\_CURRENT

This property will return last or newly inserted record id of specified table. It’s not limited to any session or scope it’s limited to mentioned table so it will return last inserted record id of specified table.

Syntax: SELECT IDENT\_CURRENT(table\_name)

Finally we can say SCOPE\_IDENTITY properties is best to get newly inserted record id from executed stored procedure or query when compared with other properties

Example

CREATE TABLE SAMPLE1 (Id INT IDENTITY)

CREATE TABLE SAMPLE2 (Id INT IDENTITY(100,1))

-- Trigger to execute while inserting data into SAMPLE1 table

GO

CREATE TRIGGER TRGINSERT ON SAMPLE1 FOR INSERT

AS

BEGIN

INSERT SAMPLE2 DEFAULT VALUES

END

GO SELECT \* FROM SAMPLE1 -- It will return empty value

SELECT \* FROM SAMPLE2 -- It will return empty value

When we execute above statements we will get output like as shown below

Now we will insert default values in “SAMPLE1” table by executing following query and check values of @@identity, scope\_identity() and ident\_current(‘tablenae’)

INSERT SAMPLE1 DEFAULT VALUES

SELECT @@IDENTITY -- It returns value 100 this was inserted by trigger

SELECT SCOPE\_IDENTITY() -- It returns value 1 this was inserted by insert query in SAMPLE1

SELECT IDENT\_CURRENT('SAMPLE2') -- It returns value inserted in SAMPLE2 table

**Difference between char varchar and nvarchar in sql server**

Char DataType

Char datatype which is used to store fixed length of characters. Suppose if we declared char(50) it will allocates memory for 50 characters. Once we declare char(50) and insert only 10 characters of word then only 10 characters of memory will be used and other 40 characters of memory will be wasted.

varchar DataType

Varchar means variable characters and it is used to store non-unicode characters. It will allocate the memory based on number characters inserted. Suppose if we declared varchar(50) it will allocates memory of 0 characters at the time of declaration. Once we declare varchar(50) and insert only 10 characters of word it will allocate memory for only 10 characters.

nvarchar DataType

nvarchar datatype same as varchar datatype but only difference nvarchar is used to store Unicode characters and it allows you to store multiple languages in database. nvarchar datatype will take twice as much space to store extended set of characters as required by other languages.

So if we are not using other languages then it’s better to use varchar datatype instead of nvarchar

Difference between bit tinyint smallint int and bigint datatypes in SQL Server

Bit DataType

This datatype represents a single bit that can be 0 or 1.

tinyint DataType

This datatype represents a single byte which is used to store values from 0 to 255 (MinVal: 0, MaxVal: 255). Its storage size is 1 byte.

smallint DataType

This datatype represents a signed 16-bit integer which is used to store values from -2^15 (-32,768) through 2^15 - 1 (32,767) and its storage size is 2 bytes.

int DataType

This datatype represents a signed 32-bit integer which is used to store values from -2^31(-2,147,483,648) to 2 ^31-1(2,147,483,647). Its storage size is 4 bytes.

Bigint DataType

This datatype represents a signed 64-bit integer which is used to store values from -2^63 (-9223372036854775808) through 2^63-1 (9223372036854775807). Its storage size is 8 bytes.

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

DELETE TABLE syntax logs the deletes thus make the delete operation slow. TRUNCATE table does not log any information but it logs information about deallocation of data page of the table so TRUNCATE table is faster as compared to delete table.

DELETE table can have criteria while TRUNCATE cannot.

TRUNCATE table does not invoke trigger.

I had mentioned that TRUNCATE table can not be rolled back while delete can be.

**If locking is not implemented, what issues can occur?**

Following are the problems that occur if you do not implement locking properly in SQLSERVER.

Lost Updates

Lost updates occur if you let two transactions modify the same data at the same time, and the transaction that completes first is lost. You need to watch out for lost updates with the READ UNCOMMITTED isolation level. This isolation level disregards any type of locks, so two simultaneous data modifications are not aware of each other. Suppose that a customer has due of 2000$ to be paid. He pays 1000$ and again buys a product of 500$ . Lets say that these two transactions are now been entered from two different counters of the company.

Now both the counter user starts making entry at the same time 10:00 AM. Actually speaking at 10:01 AM the customer should have 2000$-1000$+500 = 1500$ pending to be paid. But as said in lost updates the first transaction is not considered and the second transaction overrides it. So the final pending is 2000$+500$ = 2500$.....I hope the company does not loose the customer.

Non-Repeatable Read

Non-repeatable reads occur if a transaction is able to read the same row multiple times and gets a different value each time. Again, this problem is most likely to occur with the READ UNCOMMITTED isolation level. Because you let two transactions modify data at the same time, you can get some unexpected results. For instance, a customer wants to book flight, so the travel agent checks for the flights availability. Travel agent finds a seat and goes ahead to book the seat. While the travel agent is booking the seat, some other travel agent books the seat. When this travel agent goes to update the record, he gets error saying that “Seat is already booked”. In short, the travel agent gets different status at different times for the seat.

Dirty Reads

Dirty reads are a special case of non-repeatable read. This happens if you run a report while transactions are modifying the data that you are reporting on. For example, there is a customer invoice report, which runs on 1:00 AM in afternoon and after that all invoices are sent to the respective customer for payments. Let us say one of the customer has 1000$ to be paid. Customer pays 1000$ at 1:00 AM and at the same time report is run. Actually, customer has no money pending but is still issued an invoice.

Phantom Reads

Phantom reads occur due to a transaction being able to read a row on the first read, but not being able to modify the same row due to another transaction deleting rows from the same table. Lets say you edit a record in the mean time somebody comes and deletes the record, you then go for updating the record which does not exist...Panicked.

Interestingly, the phantom reads can occur even with the default isolation level supported by SQL Server: READ COMMITTED. The only isolation level that does not allow phantoms is SERIALIZABLE, which ensures that each transaction is completely isolated from others. In other words, no one can acquire any type of locks on the affected row while it is being modified.

**What are different transaction levels in SQL SERVER?**

Twist: - What are different types of locks in SQL SERVER?

Transaction Isolation level decides how is one process isolated from other process. Using transaction levels, you can implement locking in SQL SERVER.

There are four transaction levels in SQL SERVER:-

READ COMMITTED

The shared lock is held for the duration of the transaction, meaning that no other transactions can change the data at the same time. Other transactions can insert and modify data in the same table, however, as long as it is not locked by the first transaction.

READ UNCOMMITTED

No shared locks and no exclusive locks are honored. This is the least restrictive isolation level resulting in the best concurrency but the least data integrity.

REPEATABLE READ

This setting disallows dirty and non-repeatable reads. However, even though the locks are held on read data, new rows can still be inserted in the table, and will subsequently be interpreted by the transaction.

SERIALIZABLE

This is the most restrictive setting holding shared locks on the range of data. This setting does not allow the insertion of new rows in the range that is locked; therefore, no phantoms are allowed.

Following is the syntax for setting transaction level in SQL SERVER.

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE

**What are the different locks in SQL SERVER?**

Depending on the transaction level, six types of lock can be acquired on data:-

Intent

The intent lock shows the future intention of SQL Server's lock manager to acquire locks on a specific unit of data for a particular transaction. SQL Server uses intent locks to queue exclusive locks, thereby ensuring that these locks will be placed on the data elements in the order the transactions were initiated. Intent locks come in three flavors: intent shared (IS), intent exclusive (IX), and shared with intent exclusive (SIX).

IS locks indicate that the transaction will read some (but not all) resources in the table or page by placing shared locks.

IX locks indicate that the transaction will modify some (but not all) resources in the table or page by placing exclusive locks.

SIX locks indicates that the transaction will read all resources, and modify some (but not all) of them. This will be accomplished by placing the shared locks on the resources read and exclusive locks on the rows modified. Only one SIX lock is allowed per resource at one time; therefore, SIX locks prevent other connections from modifying any data in the resource (page or table), although they do allow reading the data in the same resource.

Shared

Shared locks (S) allow transactions to read data with SELECT statements. Other connections are allowed to read the data at the same time; however, no transactions are allowed to modify data until the shared locks are released.

Update

Update locks (U) are acquired just prior to modifying the data. If a transaction modifies a row, then the update lock is escalated to an exclusive lock; otherwise, it is converted to a shared lock. Only one transaction can acquire update locks to a resource at one time. Using update locks prevents multiple connections from having a shared lock that want to eventually modify a resource using an exclusive lock. Shared locks are compatible with other shared locks, but are not compatible with Update locks.

Exclusive

Exclusive locks (X) completely lock the resource from any type of access including reads. They are issued when data is being modified through INSERT, UPDATE and DELETE statements.

Schema

Schema modification locks (Sch -M) are acquired when data definition language statements, such as CREATE TABLE, CREATE INDEX, ALTER TABLE, and so on are being executed. Schema stability locks (Sch-S) are acquired when store procedures are being compiled.

Bulk Update

Bulk update locks (BU) are used when performing a bulk-copy of data into a table with TABLOCK hint. These locks improve performance while bulk copying data into a table; however, they reduce concurrency by effectively disabling any other connections to read or modify data in the table.

**Can we suggest locking hints to SQL SERVER?**

We can give locking hints that helps you over ride default decision made by SQL Server. For instance, you can specify the ROWLOCK hint with your UPDATE statement to convince SQL Server to lock each row affected by that data modification. Whether it is prudent to do so is another story; what will happen if your UPDATE affects 95% of rows in the affected table? If the table contains 1000 rows, then SQL Server will have to acquire 950 individual locks, which is likely to cost a lot more in terms of memory than acquiring a single table lock. So think twice before you bombard your code with ROWLOCKS.

**What is LOCK escalation?**

Lock escalation is the process of converting of low-level locks (like rowlocks, 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 SQL Server dynamically manages it.

**What are the different ways of moving data between 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.

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

You can use Having Clause with the GROUP BY function in a query and WHERE Clause is applied to each row before, they are part of the GROUP BY function in a query.

**What is the difference between UNION and UNION ALL SQL syntax?**

UNION SQL syntax is used to select information from two tables. But it selects only distinct records from both the table, while UNION ALL selects all records from both the tables.

Note :- Selected records should have same datatype or else the syntax will not work.

**What are the different types of triggers in SQl SERVER?**

There are two types of triggers:-

INSTEAD OF triggers

INSTEAD OF triggers fire in place of the triggering action. For example, if an INSTEAD OF UPDATE trigger exists on the Sales table and an UPDATE statement is executed against the Salestable, the UPDATE statement will not change a row in the sales table. Instead, the UPDATE statement causes the INSTEAD OF UPDATE trigger to be executed.

AFTER triggers

AFTER triggers execute following the SQL action, such as an insert, update, or delete. This is the traditional trigger which existed in SQL SERVER.

INSTEAD OF triggers are executed automatically before the Primary Key and the Foreign Key constraints are checked, whereas the traditional AFTER triggers is executed after these constraints are checked. Unlike AFTER triggers, INSTEAD OF triggers can be created on views.

**if we have multiple AFTER Triggers on table how can we define the sequence od the triggers?**

If a table has multiple AFTER triggers, then you can specify which trigger should be executed first and which trigger should be executed last using the stored procedure sp\_settriggerorder.

**What is SQL injection?**

It is a Form of attack on a database-driven Web site in which the attacker executes unauthorized SQL commands by taking advantage of insecure code on a system connected to the Internet, bypassing the firewall. SQL injection attacks are used to steal information from a database from which the data would normally not be available and/or to gain access to an organization’s host computers through the computer that is hosting the database.

SQL injection attacks typically are easy to avoid by ensuring that a system has strong input validation.

As name suggest we inject SQL which can be relatively dangerous for the database. Example this is a simple SQL

SELECT email, passwd, login\_id, full\_name FROM members WHERE email = 'x'

Now somebody does not put “x” as the input but puts “x ; DROP TABLE members;”. So the actual SQL which will execute is:-

SELECT email, passwd, login\_id, full\_name FROM members WHERE email = ‘x’; DROP TABLE members;

Think what will happen to your database.

**What is the difference between Stored Procedure (SP) and User Defined Function (UDF)?**

Following are some major differences between a stored procedure and user defined functions:-

You can not change any data using UDF while you can do everything with a stored procedure.

UDF cannot be used in XML FOR clause but SP’s can be used.

UDF does not return output parameters while SP’s return output parameters.

If there is an error in UDF its stops executing. But in SP’s it just ignores the error and moves to the next statement.

UDF cannot make permanent changes to server environments while SP’s can change some of the server environment.

**How can you raise custom errors from stored procedure?**

The RAISERROR statement is used to produce an ad hoc error message or to retrieve a custom message that is stored in the sysmessages table. You can use this statement with the error handling code presented in the previous section to implement custom error messages in your applications. The syntax of the statement is shown here.

RAISERROR ({msg\_id |msg\_str }{,severity ,state }

[ ,argument [ ,,...n ] ] ))

[ WITH option [ ,,...n ] ]

A description of the components of the statement follows.

msg\_id: The ID for an error message, which is stored in the error column in sysmessages.

msg\_str: A custom message that is not contained in sysmessages.

Severity: The severity level associated with the error. The valid values are 0–25. Severity levels 0–18 can be used by any user, but 19–25 are only available to members of the fixed-server role sysadmin. When levels 19–25 are used, the WITH LOG option is required.

State A value that indicates the invocation state of the error. The valid values are 0–127. This value is not used by SQL Server.

Argument . . .

One or more variables that are used to customize the message. For example, you could pass the current process ID (@@SPID) so it could be displayed in the message.

WITH option, . . .

The three values that can be used with this optional argument are described here.

LOG: Forces the error to log in the SQL Server error log and the NT application log.

NOWAIT: Sends the message immediately to the client.

SETERROR: Sets @@ERROR to the unique ID for the message or 50,000.

The number of options available for the statement makes it seem complicated, but it is actually easy to use. The following shows how to create an ad hoc message with a severity of 10 and a state of 1.

RAISERROR ('An error occurred updating the Nonfatal table',10,1)

--Results--

An error occurred updating the Nonfatal table

The statement does not have to be used in conjunction with any other code, but for our purposes, it will be used with the error handling code presented earlier. The following alters the ps\_NonFatal\_INSERT procedure to use RAISERROR.

USE tempdb

go

ALTER PROCEDURE ps\_NonFatal\_INSERT

@Column2 int =NULL

AS

DECLARE @ErrorMsgID int

INSERT NonFatal VALUES (@Column2) SET @ErrorMsgID =@@ERROR

IF @ErrorMsgID <>0

BEGIN

RAISERROR ('An error occured updating the NonFatal table',10,1)

END

When an error-producing call is made to the procedure, the custom message is passed to the client.

**What is DBCC?**

DBCC (Database Consistency Checker Commands) is used to check logical and physical consistency of database structure.DBCC statements can fix and detect problems. These statements are grouped in to four categories:-

Maintenance commands like DBCC DBREINDEX, DBCC DBREPAR etc, they are mainly used for maintenance tasks in SQL SERVER.

Miscellaneous commands like DBCC ROWLOCK, DBCC TRACEO etc, they are mainly used for enabling row-level locking or removing DLL from memory.

Status Commands like DBCC OPENTRAN, DBCC SHOWCONTIG etc , they are mainly used for checking status of the database.

Validation Commands like DBCC CHECKALLOC, DBCCCHECKCATALOG etc , they perform validation operations on database.

Note :- Check MSDN for list of all DBCC commands, it is very much possible specially during DBA interviews they can ask in depth individual commands.

Below is a sample screen in which DBCC SHOWCONTIG command is run. DBCC SHOWCONTIG is used to display fragmentation information for the data and indexes of the specified table.In the sample screen “Customer” table is checked for fragmentation

Fragmentation information. If “Scan density” is 100 then everything is contigious.The above image has scan density of 95.36% which is decent percentage. So such type of useful information can be collected by DBCC command and database performance and maintenance can be improved.

**What is the purpose of Replication?**

Replication is way of keeping data synchronized in multiple databases. SQL server replication has two important aspects publisher and subscriber.

Publisher

Database server that makes data available for replication is known as Publisher.

Subscriber

Database Servers that get data from the publishers is called as Subscribers.

**What are the different types of replication supported by SQL SERVER?**

There are three types of replication supported by SQL SERVER:

Snapshot Replication.

Snapshot Replication takes snapshot of one database and moves it to the other database. After initial load data can be refreshed periodically. The only disadvantage of this type of replication is that all data has to be copied each time the table is refreshed.

Transactional Replication

In transactional replication, data is copied first time as in snapshot replication, but later only the transactions are synchronized rather than replicating the whole database. You can either specify to run continuously or on periodic basis.

Merge Replication.

Merge replication combines data from multiple sources into a single central database. Again as usual, the initial load is like snapshot but later it allows change of data both on subscriber and publisher, later when they come on-line it detects and combines them and updates accordingly.

**What is BCP utility in SQL SERVER?**

BCP (Bulk Copy Program) is a command line utility by which you can import and export large amounts of data in and out of SQL SERVER database.

**What is a Cursor?**

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

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

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

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

**What is an index?**

An Index is one of the most powerful techniques to work with this enormous information. Database tables are not enough for getting the data efficiently in case of a huge amount of data. In order to get the data quickly we need to index the column in a table.

An index is a database object that is created and maintained by the DBMS. Indexed columns are ordered or sorted so that data searching is extremely fast. An index can be applied on a column or a view. A table can have more than one index.

Types of Index: Microsoft SQL Server has two types of indexes. These are:

Clustered Index: A Clustered Index sorts and stores the data in the table based on keys. A Clustered Index can be defined only once per table in the SQL Server Database, because the data rows can be sorted in only one order. Text, nText and Image data are not allowed as a Clustered index.

SET STATISTICS IO ON

SELECT \* FROM Employee WHERE EmpID = 20001

EmpID EmpName Cell Dept

20001 Black Smith 12345678901 1

Non-Clustered Index: Non Clustered Indexes or simply indexes are created outside of the table. SQL Server supports 999 Non-Clustered per table and each Non-Clustered can have up to 1023 columns. A Non-Clustered Index does not support the Text, nText and Image data types.

CREATE NONCLUSTERED INDEX NCL\_ID ON Employee(DeptID)

SET STATISTICS IO ON

SELECT \* FROM Employee WHERE DeptID = 20001

EmpID EmpName Cell Dept

40001 Black Smith 12345678901 20001

**Why do I need an index in a database?**

An Index is a database object that can be created on one or more columns (16 max column combinations). When creating the index it will read the column(s) and forms a relevant data structure to minimize the number of data comparisons. The index will improve the performance of data retrieval and add some overhead to the data modification such as create, delete and modify. So it depends on how much data retrieval can be done on table versus how much of DML (Insert, Delete and Update) operations.

Need of Index in Database: An index is basically used for fast data retrieval from the database.

Syntax: CREATE INDEX index\_name ON table\_name;

or

DROP INDEX index\_name;

Type of Index:

In a SQL Server database there are mainly the following two types of indexes:

1. Clustered index and
2. Non Clustered index

**What is a query in a database?**

SQL is a complete data manipulation language that is used not only for database queries, but also to modify and update data in the database. Compared to the complexity of the SELECT statement, which supports SQL queries, the SQL statements that modify and create database contents are somewhat simple.

However, database updates pose some challenges for a DBMS beyond those presented by database queries. The DBMS must protect the integrity of the stored data during changes, ensuring that only valid data is introduced into the database. The DBMS must also coordinate simultaneous updates by multiple users, ensuring that the users and their changes do not interfere with one another.

INSERT statement adds new rows of data to a table.

Syntax:

Insert into table\_Name(column names) values(Values for column).

INSERT INTO employee(ID, SURNAME, FIRSTNAME, EMAIL, COUNTRY, PHONE) VALUES(111, 'vithal', 'wadje', '[vithal.wadje@yahoo.com](mailto:vithal.wadje@yahoo.com)', 'India', ' +914545455454')

**What are query types in a database?**

Types of Commands in SQL ServerThese commands are categorized into:

• DDL

• DCL

• DML

• TCL

Let's see these categories one-by-one.

DDL - Data Definition Language (DDL) commands are the category responsible for dealing with the structure of objects. I mean that with these commands we can modify our object/entity structure. For example if there's one table and you want to modify the structure of that table, you can use DDL commands.

The following are the commands in this category:

Command

Description

Create

Used to create objects.

Alter

Used to modify created object.

Drop

Used to delete object.

Using these commands you can create any objects like tables, views, databases, triggers, and so on.

For example:

CREATE DATABASE DB2

GO

CREATE TABLE tblDemo (Id int primary key, Name char(20))

GO

DROP DATABASE DB2

DML - Data Manipulation Language (DML) commands manipulate data stored in objects like tables, views and so on. With the help these commands you can easily modify, insert and delete your data.

The following are the commands in this category:

Command

Description

Insert

Insert data into table.

Delete

Delete data from table.

Update

Update data into table.

Insert Into

Insert bulk data into table.

Using these commands you can manipulate any kind of data stored in entities.

For example: INSERT INTO tblDemo VALUES(1, 'Abhishek')

GO

DELETE FROM tblDemo WHERE Id = 4

GO

UPDATE tblDemo

SET Name = 'Sunny' WHERE Id = 6

GO

DCL - Data Control Language (DCL) commands are for security purposes. These commands are used to provide roles, permissions, access and so on.

The following are the commands in this category:

Command

Description

Grant

Provide user access to Database or any other object.

Revoke

Take back the access from user.

For example: we have the following data:

Database: kansiris.org

Table: kansiris1

User: Kansiris

currently we didn't provide any permission to this user.

Now we'll create a table in the KansirisDB database:

CREATE table tblArticles(ArticleId int primary key identity, ArticleName varchar(10),Category varchar(10) )

If we execute this command, we'll get an error message.

Msg 262, Level 14, State 1, Line 1

CREATE TABLE permission denied in database 'Kansiris'.

This is because this user doesn't have permission to create anything right now. We'll learn how to grant or revoke permission on objects in our next article.

TCL - Transaction Control Language (TCL) commands are for managing transactions in SQL Server. The following are the commands in this category:

Command Description

Commit

Used to save any transaction permanently.

Rollback

This command Is used to restore database to its last committed state.

Save Tran

This command is used to save the transaction so that we can rollback that transaction to the point whenever necessary.

For example, we have a table named "tblStudent" with 3 records as shown below:

Now we'll begin our transaction and add another record and commit that transaction.

1. Begin Tran
2. Insert INTO tblStudents VALUES('Sumit')
3. COMMIT

Now we have 4 Records.

Now, we'll add three records, one by one with save point, but we don't commit our transaction.

Begin Tran

Insert INTO tblStudents VALUES('Kajal')

SAVE Tran A;

Insert INTO tblStudents VALUES('Rahul')

SAVE Tran B;

Insert INTO tblStudents VALUES('Ram')

SAVE Tran C;

SELECT \* from tblStudents

Now we have the following records in the table, from which the last three records are not yet committed.

Now we have 3 savepoints, in other words A, B and C. Since our transaction is not yet committed, we can roll it back to any savepoint. We'll roll it back to point B, in other words at "Rahul".

ROLLBACK TRAN B

COMMIT now when you fire the select query, you'll get records up to ID 6.

In this, we have seen the types of commands in SQL Server and done some overview of that. We have also seen how to commit transactions and how to roll back any transaction to any saved point.

**What is a join in SQL Server?**

If you want to retrieve data from multiple tables then you need to use joins in SQL Server. Joins are used to get data from two or more tables based on the relationships among some of the columns in the tables.

Syntax: The Inner join syntax is as follows:

SELECT < column list >FROM < left joined table > [INNER] JOIN < right joined table > ON < join condition >

The example is developed in SQL Server 2012 using the SQL Server Management Studio.

Creating Table in SQL Server:

Now to create 3 tables in the Master database named Table1, Table2 and Table3.

Table1-

CREATE TABLE Table1 ( ID INT, Name VARCHAR(20) )

Table2-

CREATE TABLE Table2 ( ID INT, Name VARCHAR(30) )

Table3-

CREATE TABLE Table3 ( ID INT, Name VARCHAR(40) )

**What are different types of joins in SQL Server?**

Joins are useful for bringing data together from different tables based on their database relations. First we will see how the join operates between tables. Then we will explore the Order of Execution when both a join and a where condition exist. Finally we will move our exploration to the importance of the Join order.

A Join condition defines a way two tables are related in a query by:

• Specifying the column to be used for the Join from each table. In joining foreign keys in a table and its associated key in the other table.

• To use the logical operator in comparing values from the columns.

There are three type of joins available based on the way we join columns of two different tables.

1. Full Join
2. Inner Join
3. Left outer Join
4. Right outer Join

Full Join - A full join is somewhat different from the Cartesian product. A Cartesian product will get all the possible row combinations of the two joining tables. A Full Join takes the matching columns plus all table rows from the left table that does not match the right and all table rows in the right that does not match the left. It applies null for unmatched rows on the other end when doing so. The following example shows the full join between Table\_A and Table\_C

**What is Self-Join?**

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

**What is Cross-Join?**

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

**What is user defined functions?**

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

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

Three types of user defined functions are.

l Scalar Functions.

l Inline Table valued functions.

l Multi statement valued functions.

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

**What is collation?**

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

ASCII value can be used to compare these character data.

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

Following are different types of collation sensitivity -.

l Case Sensitivity – A and a and B and b.

l Accent Sensitivity.

l Kana Sensitivity – Japanese Kana characters.

l Width Sensitivity – Single byte character and double byte character.

**Advantages and Disadvantages of Stored Procedure?**

Stored procedure can be used as a modular programming – means create once, store and call for several times whenever required. This supports faster execution instead of executing multiple queries. This reduces network traffic and provides better security to the data. Disadvantage is that it can be executed only in the Database and utilizes more memory in the database server.

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

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

Example – Bank Transactions on a daily basis.

**What is CLAUSE?**

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

Example – Query that has WHERE condition

Query that has HAVING condition.

**What is recursive stored procedure?**

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

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

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

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

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

**What is an ALIAS command?**

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

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

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

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

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

**What are aggregate and scalar functions?**

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

Example -.

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

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

**What is an inner join in SQL?**

Inner or Self Join - This Join returns a row when there is at least one match in both tables.

Let’s see an example:

Select \* From Table1 Inner Join Table2 on table1.ID = table2.ID

The following query displays the Employee Name and the corresponding Manager Name within the employee table.

SELECT e1.Employee\_Name EmployeeName, e2.Employee\_Name ManagerName FROM employee e1(nolock), employee e2(nolock) WHERE e1.EmployeeID = e2.ManagerID

Output:

An inner join (sometimes called a "simple join") is a join of two or more tables that returns only those rows that satisfy the join condition.

**What is an outer join in SQL?**

There are three different types of outer joins; let's see 1 by 1.

• Left Outer Join

• Right Outer Join

• Full Outer Join

Left Outer Join - A LEFT OUTER JOIN is one of the JOIN operations that allows you to specify a join clause. It preserves the unmatched rows from the first (left) table, joining them with a NULL row in the shape of the second (right) table.

Select \* From Table1 Left Outer Join on table1.ID = table2.ID

Right Outer Join - A RIGHT OUTER JOIN is one of the JOIN operations that allows you to specify a JOIN clause. It preserves the unmatched rows from the Table2 (right) table, joining them with a NULL in the shape of the Table1 (left) table. A LEFT OUTER JOIN B is equivalent to B RIGHT OUTER JOIN A, with the columns in a different order.

Select \* From Table1 Right Outer Join on table1.ID = table2.ID

**What is full join in SQL?**

A Full Outer Join fetches all records of both tables; where the record does not match, it returns Null. select e.empId, e.empName, e1.empAdd from emp e full outer join emp\_add e1 on e.empI d = e1.empId

Output:

Or

Full Outer Join - FULL OUTER JOIN: This JOIN is a combination of both. All records from both Left\_Table and Right\_Table are in the result set and matched when they can be on the Join\_Condition; when no record is found in the opposite table, NULL values are used for the columns.

Select \* From Table1 Full Outer Join on table1.ID = table2.ID

**What is left join in SQL Server?**

Left Join: A LEFT OUTER JOIN is one of the JOIN operations that allow you to specify a join clause. It preserves the unmatched rows from the first (left) table, joining them with a NULL row in the shape of the second (right) table.

Select \* From Table1 Left Outer Join on table1.ID=table2.ID

A left outer join displays all the rows from the first table and the matched rows from the second table.

Example: The following query retrieves the employee name and the corresponding department he belongs to, whereas all the departments are displayed even if the employee is not assigned to any department.

SELECT e.EmployeeID, e.Employee\_Name, d.Department\_Name FROM employee e(nolock) LEFT JOIN department d(nolock) ON e.DepartmentID = d.DepartmentID

Output:

**What is a right join in SQL Server?**

Right JOIN - A RIGHT OUTER JOIN is one of the JOIN operations that allows you to specify a JOIN clause. It preserves the unmatched rows from the Table2 (right) table, joining them with a NULL in the shape of the Table1 (left) table. A LEFT OUTER JOIN B is equivalent to B RIGHT OUTER JOIN A, with the columns in a different order.

Select \* From Table1 Right Outer Join on table1.ID = table2.ID

The right outer join displays all the rows from the second table and matched rows from the first table.

Example:

SELECT e.EmployeeID, e.Employee\_Name, d.Department\_Name FROM employee e(nolock) RIGHT JOIN department d(nolock) ON e.DepartmentID = d.DepartmentID

Output:

**What is database engine in SQL Server?**

The SQL Server Database Engine, SQL Server Agent, and several other SQL Server components run as services. These services typically are started when the operating system starts. This depends on what is specified during setup; some services are not started by default.

A service is a type of application (executable) that runs in the system background. Services usually provide core operating system features, such as Web serving, event logging, or file serving. Services can run without showing a user interface on the computer desktop. The SQL Server Database Engine, SQL Server Agent, and several other SQL Server components run as services. These services typically are started when the operating system starts. This depends on what is specified during setup; some services are not started by default.

This describes the management of the various SQL Server services on your machine. Before you log in to an instance of SQL Server, you need to know how to start, stop, pause, resume, and restart an instance of SQL Server. After you are logged in, you can perform tasks such as administering the server or querying a database.

Let's start now, select start/All Programs/Microsoft SQL Server2005/Configuration Tools/SQL Server Configuration Manager.

**What are the Analysis Services in SQL Server?**

The purpose of analysis services is to turn data into information and to provide quick and easy access to that information for decision makers. SSAS provides OLAP by letting you design, create and manage multidimensional structures that contain data aggregated from other data sources, such as relational databases and provides many data mining algorithms for mining data from data sources. So for delivering OLAP and data mining it uses client and server technologies.

The main idea of SSAS is to provide fast results from data sources when we apply a query because in order to make a decision we need data of various dimensions.

Components of the Architecture in detail:

1. Server Architecture: This runs as a Windows service. The Msmdsrv.exe application is a server component. This application consists of security, XMLA listener, query processor and other components that perform the following tasks:
2. Client Architecture: SSAS has a thin client Component Architecture. All queries and calculations are resolved by the server only. So for each request a server to client connection is required. There are several providers with SSAS to support various programming languages. These providers communicate using SOAP packets. You can better understand this by the following diagram:

**What are the integration services in SQL Server?**

Integration Services is a platform for building high performance data integration and workflow solutions, including extraction, transformation and loading (ETL) operations for data warehousing.

This includes graphical tools and wizards for building and debugging packages.

Uses of Integration Services:

One use of Integration Services is to merge data from multiple data stores and update the data to data warehouses and/or data marts. Create the Data Transformation process logic and automate the data loading process.

Architecture of Integration Services:

Some important components to using Integration Services:

• SSIS Designer

• Runtime engine

• Tasks and other executables

• Data Flow engine and Data Flow components

• API or object model

• Integration Services Service

• SQL Server Import and Export Wizard

• Other tools, wizards and command prompt utilities

**What are the data quality services in SQL Server?**

SQL Server Data Quality Services - SQL Server 2012 Data Quality Services (DQS) is the data quality product from Microsoft SQL Server 2012. DQS enables you to perform a variety of critical data quality tasks, including correction, enrichment, standardization and de-duplication of your data.

DQS provides the following features to resolve data quality issues:

• Data Cleansing

• Matching

• Reference Data Services

• Profiling and Monitoring

• Knowledge Base

DQS enables you to perform data cleansing using cloud-based reference data services provided by reference data providers. DQS also provides profiling that is integrated into its data-quality tasks, enabling to analyze the integrity of the data.

Data Quality Server - Data Quality Server is implemented as three SQL Server catalogs DQS\_MAIN, DQS\_PROJECTS, and DQS\_STAGING\_DATA. DQS\_MAIN includes DQS Stored Procedures, DQS engine, and published Knowledge Bases.

DQS\_PROJECTS includes data that is required for Knowledge Base management and DQS project activities.

DQS\_STAGING\_DATA provides an intermediate staging database where you can copy source data to perform DQS operations, and then export your processed data.

**What are the reporting services in SQL Server?**

SQL Server Reporting Services is a comprehensive reporting platform that includes processing components. Processing components are the basis for the multilayered architecture of SQL Server Reporting Services. Processing components interact with each other to retrieve data and deliver a report. S QL Server Reporting Services has the following two basic components:

• Processors

• Extensions

Tools and Components of SQL Server Reporting Services architecture: This architecture consists mainly of the following types of components and tools:

• Report Builder

• Report Designer

• Report Manager

• Report Server

• Report server database

• Data sources

**What are the master data services in SQL Server?**

The goal of MDS is to address the challenges of both operational and analytical master data management by providing a master data hub to centrally organize, maintain, and manage your master data. This master data hub supports these capabilities with a scalable and extensible infrastructure built on SQL Server and the Windows Communication Foundation (WCF) APIs.

Master Data Services Components: The wizard installs Master Data Services Configuration Manager, installs the files necessary to run the Master Data Services Web service, and registers assemblies. After installation, you use the Master Data Services Configuration Manager to create and configure a Master Data Services database in a SQL Server instance that you specify, create the Master Data Services Web application, and enable the Web service.

Data Stewardship: Master Data Manager is the data stewardship portal in which authorized business users can perform all activities related to master data management. At minimum, a user can use this Web application to review the data in a master data model. Users with higher permissions can make changes to the master data and its structure, define business rules, review changes to master data, and reverse changes.

Model Objects: Most activities in MDS revolve around models and the objects they contain. A model is a container for all objects that define the structure of the master data. A model contains at least one entity, which is analogous to a table in a relational database. An entity contains members, which are like the rows in a table, as shown in Figure 7-1. Members (also known as leaf members) are the master data that you are managing in MDS. Each leaf member of the entity has multiple attributes, which correspond to table columns in the analogy.

Master Data Maintenance: Master Data Manager is more than a place to define model objects. It also allows you to create, edit, and update leaf members and consolidated members. When you add a leaf member, you initially provide values for only the Name and Code attributes, as shown in Figure 7-4. You can also use a search button to locate and select the parent consolidated member in each hierarchy.

**What is replication in SQL Server?**

Replication is a process or method to synchronize the data across multiple servers. Replication is done by a replica set. A replication maintains the same data set. Replica sets provide redundancy and high availability with multiple copies of data on different database servers.

Replication removes dependencies from a single server so replication protects a database from the loss of a single server. Replication provides a mechanism to recover from hardware failure and service interruptions.

Replication is also used to increase the read capacity. Replication provides choices for the client to select a different server for read and write operations. Replication maintains copies in different data centers to increase the locality and availability of data for distributed applications.

Example: Snapshot Replication

Step 1: Open the replication node in your database and choose the option Local Publications.

Step 2: Right-click on Local Publications and click on New publication.

Step 3: After clicking on the new publication tab the following window will appear and click on the “Next” button.

**How to I select data from an SQL Server table?**

How to select specific rows or all columns, selecting distinct rows, filtering with where clause, sorting rows using orderby and so on. We will be using the AdventureWorks2012 database for this demo.

1. To select all the rows and columns from a table, we use the following query:

SELECT \* FROM HumanResources.Employee

Execute the query by pressing F5 or via the execute button.

Output:

There is another way to select all the columns from a table. Instead of using \* we can specify the column names.

1. SELECT BusinessEntityID, NationalIDNumber, LoginID, OrganizationNode, Organizatio nLevel, JobTitle, BirthDate, MaritalStatus, Gender, HireDate, SalariedFlag, VacationHour s, SickLeaveHours, CurrentFlag, rowguid, ModifiedDate FROM HumanResources.Employee The output will be the same.

If you feel lazy in writing this long query given above then what you can do is go to the Object Explorer window, then expand adventureWorks2012 then select HumanResourcesEmployee table and right-click on it. After that "select script table as" then select "To", then you will see a New query editor window.

SQL Server will generate the SELECT query for us.

**What is a check in SQL?**

A Check Constraint is a rule that identifies valid values Constraint helps to enforce Domain Integrity. If the condition satisfied then it prevents the value from entering into the database.

for columns of data. A Check in a Check Constraint is not

Syntax:

Create table tableName(Column1 dataType Check(expression), Column2, columnN)

Example:

create table emp(empId int check(empId >10),empName varchar(15))

Output:

insert into emp values(8,'d')

Output:

Dropping the Check Constraint:

Firstly, we can determine the name of the constraint using the following command:

exec sp\_help emp

Output:

**What is a default in SQL?**

Constraints are rules that decide what kind of data can enter into the database tables. SQL server has six types of constraints and we will explore all these constraints here with suitable examples. The constraints that we are going to explore are listed below:

1. Primary Key Constraint
2. Foreign Key Constraint
3. Not Null Constraint
4. Unique constraint
5. Default Constraint
6. Check Constraint

Default Constraint:

Default constraint allows you set a default value for the column. That means when a row is created for the first time, and there is no entry specified for the column that has a default constraint on it, then the default value is stored in the column.

Note that this not a Not Null constraint and do not confuse the default value constraint with disallowing the Null entries. Default value for the column is set only when the row is created for the first time and column value is ignored on the Insert. Modification to the column with NULL value or even the Insert operation specifying the Null value for the column is allowed.

Let us set the Default value of 1 for the Class. Here are the steps:

• Bring up the table designer

• Select the Class Row as you already did.

• At the bottom of the layout, you will see Column properties.

**How to create a database using SQL?**

A database is described as an organized way of collection of DATA. It is the collection of schemes, tables, queries, reports, views and other objects.

Syntax: CREATEDATABASEDatabaseName

Example: CREATEDATABASE Student

Or you can create Database through Design/ Wizard form by right clicking on DATABASE option - New Database

**What is a constraint in SQL?**

Constraints are the rules that decide what kind of data can enter into the database tables. SQL server has six types of constraints and we will explore all these constraints here with suitable examples. The constraints that we are going to explore are listed below:

1. Primary Key Constraint
2. Foreign Key Constraint
3. Not Null Constraint
4. Unique constraint
5. Default Constraint
6. Check Constraint

To explain these constraints we need two tables. Firstly, let us create these tables. Run the scripts shown below to create the tables. Copy and paste the code into the new Query Editor window, then execute it.

CREATE TABLE Student(StudId smallint, StudName varchar(50), Class tinyint);

CREATE TABLE TotalMarks(StudentId smallint, TotalMarks smallint);

Go

Note that there are no constraints at present on these tables. We will add the constraints one by one.

Primary Key Constraint - A table column with this constraint is called the key column for the table. This constraint helps the table to make sure that the value is not repeated and also that there are no null entries. We will mark the StudId column of the Student table as the primary key. Follow these steps:

• Right click the student table and click on the modify button

• From the displayed layout select the StudId row by clicking the Small Square like button on the left side of the row.

• Click on the Set Primary Key toolbar button to set the StudId column as primary key column.

Now this column does not allow null values and duplicate values. You can try inserting values to violate these conditions and see what happens. A table can have only one Primary key. Multiple columns can participate on the primary key column. Then the uniqueness is considered among all the participant columns by combining their values.

Not Null Constraint - This constraint is useful to stop storing the null entries in the specified columns. We will mark student name column as not null column. This allows us to always having some entries in the student name column of the student table without having NULL. Follow the steps below:

As you did previously, bring up the table design view by clicking the modify context menu for the table. 393 Remove the check mark as shown in the picture below. This action will enable the Not Null constraint for the StudName column.

Default Constraint - Default constraint allows you set a default value for the column. That means when a row is created for the first time, and there is no entry specified for the column that has a default constraint on it, then the default value is stored in the column. Note that this not a Not Null constraint and do not confuse the default value constraint with disallowing the Null entries.

Default value for the column is set only when the row is created for the first time and column value is ignored on the Insert. Modification to the column with NULL value or even the Insert operation specifying the Null value for the column is allowed. Let us set the Default value of 1 for the Class. Here are the steps:

• Bring up the table designer

• Select the Class Row as you already did.

• At the bottom of the layout, you will see a Column properties as shown in the below picture. Set the default as shown below:

**What is data Integrity?**

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

**What is Auto Increment?**

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

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

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

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

**What is Datawarehouse?**

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

**How do I define constraints in SQL?**

Constraints are rules and restrictions applied on a column or a table such that unwanted data can't be inserted into tables. This ensures the accuracy and reliability of the data in the database. We can create constraints on single or multiple columns of any table.

Constraints maintain the data integrity and accuracy in the table.

Constraints can be classified into the following two types:

Column Types Constraints - Definitions of these types of constraints is given when the table is created.

Create Table My\_Constraint (IID int NOT NULL, Salary int CHECK(Salary > 5000) )

Table Types Constraints - Definitions of these types of constraints is given after the creation of the table using the Alter Command.

Alter Table My\_Cosntraint Add constraint Check\_Constraint Check(Age>50)

SQL Server contains the following six types of constraints:

• Not Null Constraint

• Check Constraint

• Default Constraint

• Unique Constraint

• Primary Constraint

• Foreign Constraint

Not Null Constraint - A Not Null constraint restrict the insertion of null values into a column. If we are using a Not Null Constraint for a column then we cannot ignore the value of this column during an insertion of data into the table.

Column Level-

Syntax:

CREATE TABLE Table\_Name(Column\_Name Datatype CONSTRAINT Constraint\_Name NOT NULL, );

Example:

Create Table My\_Constraint(IID int NOT NULL, Name nvarchar(50) CONSTRAINT Cons\_NotNull not null, Age int Not Null, )

Table Level-

Syntax:

ALTER TABLE Table\_Name

ALTER COLUMN Column\_Name Datatype NOT NULL

Example:

Alter Table My\_Constraint

Alter Column IId int Not Null

Without SQL Command - We can also create a Not Null constraint in Microsoft SQL Server without execution of a SQL query.

First right-click on the table and select and click on the design option. Now check all the columns in the “Allow Nulls” option that should have a Null Value.

Check Constraint - A Check constraint checks for a specific condition before inserting data into a table. If the data passes all the Check constraints then the data will be inserted into the table otherwise the data for insertion will be discarded. The CHECK constraint ensures that all values in a column satisfies certain conditions.

**What is the meaning of Not Null in SQL?**

Constraints are rules that decide what kind of data can enter into the database tables. SQL server has six types of constraints and we will explore all these constraints here with suitable examples. The constraints that we are going to explore are listed below:

• Primary Key Constraint

• Foreign Key Constraint

• Not Null Constraint

• Unique constraint

• Default Constraint

• Check Constraint

This constraint is useful to stop storing the null entries in the specified columns. We will mark student name column as not null column. This allows us to always have some entries in the student name column of the student table without having NULL. Here are the steps:

1. As you did previously, bring up the table design view by clicking the modify context menu for the table.
2. Remove the check mark as shown in the picture below. This action will enable the Not Null constraint for the StudName column.

Example:

**How to alter a table schema in SQL Server?**

Altering Tables: It is used to modify an existing table.

CREATE TABLE Stock ( ID SMALLINT ); mysql > ALTER TABLE Stock -

ADD COLUMN Quantity SMALLINT UNSIGNED NOT NULL, -

MODIFY ID SMALLINT UNSIGNED NOT NULL, - > ADD PRIMARY KEY(ID);

mysql > Describe Stock;

mysql > ALTER TABLE Stock;

Example in Sql:

**How to create index in SQL Server?**

Indexes are data structures that are used to improve the searching speed in a table. The user can not see the index directly. An index increases the performance of select statements and where clausees and slows down insert and update statements. So we create indexes only for those columns that are not frequently updated.

Creation of index: Example: create index i\_select on emp(empName)

Creation of composite index: It is created on more than one column of the table using.

Example: create index i\_select on emp(empId,empName)

Creation of Unique index: Used for Data Integrity. A Unique index does not allow duplicate values to be inserted into the table.

Example: create index i\_unique on emp(empId) any

**How to get unique records in SQL?**

Unique Constraint: It ensures that each row for a column must have a unique value. It is like a Primary key but it can accept only one null value. In a table one or more column can contain a Unique Constraint.

Column Level:

Syntax:Create Table Table\_Name (Column\_Name Datatype Constraint Constraint\_Name Unique)

Example:Create Table MY\_Tab(IId int constraint Unique\_Cons Unique ,Name nvarchar(50))

**How to create a date column in SQL Server?**

Datetime data type can store dates from January 1, 1753 to December 31, 9999 with a precision up to 0.003 fraction of a second. The smalldate data type can store dates from January 1, 1900 to June 6, 2079 with a precision of a second.

create table tbDate ( col datetime); go insert into tbDate values('8:00 AM'); go insert into tbDate values('March 24,2008'); go

There are styles to format the input and output when converting from datetime into characters. We must use the convert function and the following list of styles.

Style ID

Style Type

0 or 100

mon dd yyyy hh:miAM (or PM)

101

mm/dd/yy

102

yy.mm.dd

103

dd/mm/yy

104

dd.mm.yy

105

dd-mm-yy

106

dd mon yy

107

Mon dd, yy

108

hh:mm:ss

9 or 109

mon dd yyyy hh:mi:ss:mmmAM (or PM)

110

mm-dd-yy

111

yy/mm/dd

112

Yymmdd

13 or 113

dd mon yyyy hh:mm:ss:mmm(24h)

114

hh:mi:ss:mmm(24h)

20 or 120

yyyy-mm-dd hh:mi:ss(24h)

21 or 121

yyyy-mm-dd hh:mi:ss.mmm(24h)

126

yyyy-mm-dd Thh:mm:ss.mmm(no spaces)

130

dd mon yyyy hh:mi:ss:mmmAM

131

dd/mm/yy hh:mi:ss:mmmAM

**What is ACID fundamental? What are transactions in SQL SERVER?**

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.

Isolation

Modifications made by concurrent transactions must be isolated from the modifications made by any other concurrent transactions. A transaction either see data in the state it was 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.

**What is a candidate key?**

A table may have more than one combination of columns that could uniquely identify the rows in a table; each combination is a candidate key. During database design you can pick up one of the candidate keys to be the primary key. For example, in the supplier table supplierid and suppliername can be candidate key but you will only pick up supplierid as the primary key.

**How do GROUP and ORDER BY Differ?**

ORDER BY alters the order in which items are returned.

GROUP BY will aggregate records by the specified columns which allows you to perform aggregation functions on non-grouped columns (such as SUM, COUNT, AVG, etc).

The ORDER BY clause’s purpose is to sort the query result by specific columns.

The GROUP BY clause’s purpose is summarize unique combinations of columns values.

Before we get into their differences consider the general setup of the SELECT statement:

SELECT columnlist From table GROUP BY columnA, columnB ORDER BY columnlist

Notice that the ORDER BY clause appears at the end. You can use this as a clue to understand that the ORDER BY statement is used to sort the final result of the query. In fact, it can be used to sort results from a GROUP BY clause. Confused? I was at first! Ok, let’s break it down.

ORDER BY

The ORDER BY statement is used to sort values. You probably already knew that! So

SELECT SalesOrderID, ProductID, OrderQty\* UnitPrice As ExtendedPrice FROM Sales.SalesOrderDetail ORDER BY SalesOrderID will sort the value, according to SalesOrderID.

Every row in the table is included in the result. The values are sorted in ascending order according to the SalesOrderID.

GROUP BY

Contrast this to the GROUP BY clause, which is used to group like column values into a single row.

This is useful as it allows you to summarize information. For instance you can use aggregate functions such as SUM and AVERAGE to calculate values.

In this example

SELECT SalesOrderID, SUM(OrderQty\* UnitPrice) As TotalPrice FROM Sales.SalesOrderDetail GROUP BY SalesOrderID

We are grouping by SalesOrderID and summing each order’s product prices to return the total. This is the magic of the GROUP BY clause: it allows you to perform summary calculations on multiple rows.

With the GROUP BY clause not every row is include in the result. Instead, only unique combinations of SalesOrderID along with the sum are included.

Now the ORDER BY and GROUP BY can be used together. You may ask what is the point, if the results are already grouped by SalesOrderID, but what about ordering by the total price? You can do this as

SELECT SalesOrderID, SUM(OrderQty\* UnitPrice) As TotalPrice FROM Sales.SalesOrderDetail

GROUP BY SalesOrderID ORDER BY TotalPrice

To summarize, the key difference between order by and group by is:

l ORDER BY is used to sort a result by a list of columns or expressions.

l GROUP BY is used to create unique combinations of a list of columns that can be used to form summaries. A byproduct of this operation is that the grouping tend to be sorted; however, this isn’t a guarantee.

**Compare SQL & PL/SQL**

Criteria SQL PL/SQL

**What it is Single query or command execution Full programming language**

What it comprises Data source for reports, web pages Application language to build, format and display report, web pages Characteristic Declarative in nature Procedural in nature Used for Manipulating data Creating applications

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

It is a tool used to duplicate enormous quantity of information from tables and views. It does not facsimile the structures same as foundation to target. BULK INSERT command helps to bring in a data folder into a record, table or view in a user-specific arrangement.

**When is the UPDATE\_STATISTICS command used?**

This command is used, ones the processing of large data is done. When we delete a large number of files, alteration or reproduction takes place in the tables, to be concerned of these changes we need to restructure the indexes This is done UPDATE\_STATISTICS.

**Explain the steps needed to Create the scheduled job?**

Steps to create a Scheduled Job :

1. Connect to the database of SQL server in SQL Server Management Studio. On the SQL Server Agent, we will find a Jobs folder.
2. Right click on jobs and choose Add New.
3. A New Job window will come into view. Give an associated name for the same.
4. Click next on the “Steps” in the left list of options. An SQL job can have multiple steps either in the form of SQL declaration or a stored practice call.
5. Click on the “Schedules” in the left list of options. An SQL job can comprise of one or supplementary schedules. It is basically the instance at which SQL job will jog itself. We can spell out returning schedules also.

**When are we going to use truncate and delete?**

1. TRUNCATE is a DDL command, whereas DELETE is a DML command.
2. We can’t execute a trigger in case of TRUNCATE whilst with DELETE, we can accomplish a trigger.
3. TRUNCATE is quicker than DELETE, for the reason that when we use DELETE to delete the data, at that time it store the whole statistics in the rollback gap on or after where we can get the data back after removal. In case of TRUNCATE, it will not store data in rollback gap and will unswervingly rub it out. TRUNCATE do not recover the deleted data.
4. We can use any condition in WHERE clause using DELETE but it is not possible with TRUNCATE.5.If a table is referenced by any foreign key constraints, then TRUNCATE won’t work.

**Explain correlated query work?**

It’s most important to be attentive of the arrange of operations in an interrelated subquery.

First, a row is processed in the outer doubt.

Then, for that exacting row, the subquery is executed – as a result for each row processed by the outer query, the subquery will also be processed. In correlated subquery, each time a line is worked for Emp1, the subquery will also make a decision on the exacting row’s value for Emp1.Salary and run. And the outer query will move on to the next row, and the subquery will execute for that row’s value of Emp1.Salary.

It will persist in anticipation of the “WHERE (1) = (… )” state is pleased.

**When is the Explicit Cursor Used ?**

If the developer needs to perform the row by row operations for the result set containing more than one row, then he unambiguously declares a pointer with a name. They are managed by OPEN, FETCH and CLOSE.%FOUND, %NOFOUND, %ROWCOUNT and %ISOPEN characteristics are used in all types of pointers.

**Find What is Wrong in this Query?**

SELECT subject\_code, AVG (marks) FROM students WHERE AVG(marks) > 75 GROUP BY subject\_code;

The WHERE clause cannot be used to restrict groups. Instead, the HAVING clause should be used.

SELECT subject\_code, AVG (marks) FROM students HAVING AVG(marks) > 75 GROUP BY subject\_code;

**Write the Syntax for STUFF function in an SQL server?**

STUFF (String1, Position, Length, String2)

String1 - String to be overwritten

Position - Starting location for overwriting

Length - Length of substitute string

String2- String to overwrite.

1. Name some commands that can be used to manipulate text in T-SQL code. For example, a command that obtains only a portion of the text or replace a text string, etc.

l CHARINDEX( findTextData, textData, [startingPosition] ) – Returns the starting position of the specified expression in a character string. The starting position is optional.

l LEFT( character\_expression , integer\_expression ) – Returns the left part of a character string with the specified number of characters.

l LEN( textData ) – Returns integer value of the length of the string, excluding trailing blanks.

l LOWER ( character\_expression ) – Returns a character expression after converting uppercase character data to lowercase.

l LTRIM( textData) – Removes leading blanks. PATINDEX( findTextData, textData ) – Returns integer value of the starting position of text found in the string.

l REPLACE( textData, findTextData, replaceWithTextData ) – Replaces occurrences of text found in the string with a new value.

l REPLICATE( character\_expression , integer\_expression ) – Repeats a character expression for a specified number of times.

l REVERSE( character\_expression ) – Returns the reverse of a character expression.

l RTRIM( textData) – Removes trailing blanks. SPACE( numberOfSpaces ) – Repeats space value specified number of times.

l STUFF( textData, start , length , insertTextData ) – Deletes a specified length of characters and inserts another set of characters at a specified starting point.

l SUBSTRING( textData, startPosition, length ) – Returns portion of the string.

l UPPER( character\_expression ) – Returns a character expression with lowercase character data converted to uppercase.

**What are the three ways that Dynamic SQL can be executed?**

l Writing a query with parameters.

l Using EXEC.

l Using sp\_executesql.

**In what version of SQL Server were synonyms released? How do synonyms work and explain its use cases? Synonyms were released with SQL Server 2005.**

l Synonyms enable the reference of another object (View, Table, Stored Procedure or Function) potentially on a different server, database or schema in your environment. In simple words, the original object that is referenced in the whole code is using a completely different underlying object, but no coding changes are necessary. Think of this as an alias as a means to simplify migrations and application testing without the need to make any dependent coding changes.

l Synonyms can offer a great deal of value when converting underlying database objects without breaking front end or middle tier code. This could be useful during a re-architecture or upgrade project.

**If you are a SQL Developer, how can you delete duplicate records in a table with no primary key?**

Use the SET ROWCOUNT command. For instance,

if you have 2 duplicate rows, you would SET ROWCOUNT 1, execute DELETE command and then SET ROWCOUNT 0.

**Is it possible to import data directly from T-SQL commands without using SQL Server Integration Services? If so, what are the commands?**

Yes, six commands are available to import data directly in the T-SQL language. These commands include :

l BCP : The bulk copy (bcp) command of Microsoft SQL Server provides you with the ability to insert large numbers of records directly from the command line. In addition to being a great tool for command-line aficionados, bcp is a powerful tool for those seeking to insert data into a SQL Server database from within a batch file or other programmatic method.

l Bulk Insert : The BULK INSERT statement was introduced in SQL Server 7 and allows you to interact with bcp (bulk copy program) via a script.

l OpenRowSet : The OPENROWSET function can be referenced in the FROM clause of a query as if it were a table name. The OPENROWSET function can also be referenced as the target table of an INSERT, UPDATE, or DELETE statement, subject to the capabilities of the OLE DB provider. Although the query might return multiple result sets, OPENROWSET returns only the first one.

l OPENDATASOURCE : Provides ad hoc connection information as part of a four-part object name without using a linked server name.

l OPENQUERY : Executes the specified pass-through query on the specified linked server. This server is an OLE DB data source. OPENQUERY can be referenced in the FROM clause of a query as if it were a table name.

l Linked Servers : Configure a linked server to enable the SQL Server Database Engine to execute commands against OLE DB data sources outside of the instance of SQL Server. Typically linked servers are configured to enable the Database Engine to execute a Transact-SQL statement that includes tables in another instance of SQL Server, or another database product such as Oracle.

**What is the native system stored procedure to execute a command against all databases?**

l The sp\_MSforeachdb system stored procedure accepts the @Command parameter which can be exetecuted against all databases. The ‘?’ is used as a placeholder for the database name to execute the same command.

l The alternative is to use a cursor to process specific commands against each database.

**How can a SQL Developer prevent T-SQL code from running on a production SQL Server?**

Use IF logic with the @@SERVERNAME function compared against a string with a RETURN command before any other logic.

**How do you maintain database integrity where deletions from one table will automatically cause deletions in another table?**

You can create a trigger that will automatically delete elements in the second table when elements from the first table are removed.

**What port does SQL server run on?**

1433 is the standard port for SQL server.

**What is the SQL CASE statement used for? Explain with an example?**

It allows you to embed an if-else like clause in the SELECT clause.

SELECT Employee\_Name, CASE Location WHEN 'alex' THEN Bonus \* 2 WHEN 'robin' THEN Bonus \*, 5 ELSE Bonus END "New Bonus" FROM Intellipaat\_employee;

**What are the risks of storing a hibernate-managed object in cache? How do you overcome the problems?**

The primary problem here is that the object will outlive the session it came from. Lazily loaded properties won’t get loaded if needed later. To overcome the problem, perform cache on the object’s id and class and then retrieve the object in the current session context.

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

Updating statistics ensures that queries compile with up-to-date statistics. However, updating statistics causes queries to recompile. We recommend not updating statistics too often because there is a performance tradeoff between improving query plans and the time it takes to recompile queries. The specific tradeoffs depend on your application. UPDATE STATISTICS can use tempdb to sort the sample of rows for building statistics.

Syntax: UPDATE STATISTICS table\_or\_indexed\_view\_name

[ { { index\_or\_statistics\_\_name } | ( { index\_or\_statistics\_name } [ ,...n ] ) } ]

[ WITH [ FULLSCAN | SAMPLE number { PERCENT | ROWS } | RESAMPLE

[ ON PARTITIONS ( { | } [, …n] ) ] | [ ,...n ] ]

[ [ , ] [ ALL | COLUMNS | INDEX ] [ [ , ] NORECOMPUTE ] [ [ , ] INCREMENTAL = { ON | OFF } ] ] ;

::=

[ STATS\_STREAM = stats\_stream ]

[ ROWCOUNT = numeric\_constant ]

[ PAGECOUNT = numeric\_contant ]

**What is SQL Profiler?**

Microsoft SQL Server Profiler is a graphical user interface to SQL Trace for monitoring an instance of the Database Engine or Analysis Services. You can capture and save data about each event to a file or table to analyze later.

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.

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

SELECT SERVERPROPERTY (‘productversion’), SERVERPROPERTY (‘productlevel’), SERVERPROPERTY (‘edition’).

**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 in SQL?**

Stuff function : This function is used to replace string from the given start position, passed as 2nd argument with string passed as last argument. In Stuff function, 3rd argument defines the number of characters which are going to be replaced.

Syntax : STUFF ( character\_expression , start , length , replaceWith\_expression )

For example : Select Stuff ('Intellipaat', 3, 3, 'abc')

This query will return the string "Iabcllipaat". In this example, Stuff function replaces the string "Intellipaat" onwards the 3rd position('nte') with 'abc'.

Replace Function : Replace function is used to replace all occurrence of a specified with the string passed as last argument.

Syntax :- REPLACE ( string\_expression , string\_pattern , string\_replacement )

For example : Select Replace ('Abcabcabc', 'bc', 'xy')

This query will return the string Axyaxyaxy. In this example, Replace function replaces the occurrence of each 'bc' string with 'xy'.

**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 is de-normalization in SQL database administration? Give examples**

De-normalization is used to optimize the readability and performance of the database by adding redundant data. It covers the inefficiencies in the relational database software.

De-normalization logical data design tend to improve the query responses by creating rules in the database which are called as constraints.

Examples include the following :

l Materialized views for implementation purpose such as :

l Storing the count of “many” objects in one-to-many relationship.

l Linking attribute of one relation with other relations.

l To improve the performance and scalability of web applications.

**Can you explain about buffer cash and log Cache in SQL Server?**

l Buffer Cache : Buffer cache is a memory pool in which data pages are read. The ideal performance of the buffer cache is indicated as: 95% indicates that pages that were found in the memory are 95% of time. Another 5% is need physical disk access.

If the value falls below 90%, it is the indication of more physical memory requirement on the server.

l Log Caches : Log cache is a memory pool used to read and write the log pages. A set of cache pages are available in each log cache. The synchronization is reduced between log and data buffers by managing log cache separately from the buffer cache.

**Describe how to use Linked Server.**

MS SQL Server supports the connection to different OLE DB on an ad hoc basis. This persistent connection is referred as Linked Server.

Following are the steps to use Linked Server for any OLE DB. You can refer this to use an MS-Excel workbook.

1. Open SQL Server Management Studio in SQL Server.
2. Expand Server Objects in Object Explorer.
3. Right-click on Linked Servers. Click on New Linked Server.
4. Select General page in the left pane and

o Type any name for the linked server in the first text box.

o Select the Other Data Source option.

o Click on Microsoft Jet 4.0 OLE DB Provider from the Provider list.

o Type the Excel as the name of the OLE DB data source.

o Type the full path and file name of the Excel file in Data Source box.

o Type the Excel version no. (7.0, 8.0 etc) in the Provider String. Use Excel 8.0 for Excel 2000, Excel 2002 or Excel 97.

o To create a linked server click on OK.

**Explain how to send email from SQL database.**

SQL Server has a feature for sending mails. Stored procedures can also be used for sending mail on demand. With SQL Server 2005, MAPI client is not needed for sending mails.

The following is the process for sending emails from database.

l Make sure that the SQL Server Mail account is configured correctly and enable Database Mail.

l Write a script to send an e-mail. The following is the script.

l USE [YourDB]

l EXEC msdb.dbo.sp\_send\_dbmail

l @recipients = '[xyz@intellipaat.com](mailto:xyz@intellipaat.com); [abc@intellipaat.com](mailto:abc@intellipaat.com);[pqr@intellipaat.com](mailto:pqr@intellipaat.com)’

l @body = ' A warm wish for your future endeavor',

l @subject = 'This mail was sent using Database Mail' ;

GO

**How to make remote connection in database?**

The following is the process to make a remote connection in database :

Use SQL Server Surface Area Configuration Tool for enabling the remote connection in database.

Click on Surface Area Configuration for Services and Connections.

Click on SQLEXPRESS/Database Engine/RemoteConnections.

Select the radio button: Local and Remote Connections and select ‘Using TCP/IP only’ under Local and Remote Connections.

Click on OK button / Apply button

**What is the purpose of OPENXML clause SQL server stored procedure?**

OPENXML parses the XML data in SQL Server in an efficient manner. It’s primary ability is to insert XML data to the RDB. It is also possible to query the data by using OpenXML. The path of the XML element needs to be specified by using ‘xpath’.

The following is a procedure for retrieving xml data:

DECLARE @index int

DECLARE @xmlString varchar(8000)

SET @xmlString =' abc 9343463943/PhoneNo> xyz 9342673212 '

EXEC sp\_xml\_preparedocument @index OUTPUT, @xmlString

SELECT \* FROM OPENXML (@index, 'Persons/Person') WITH (id varchar(10), Name varchar(100) 'Name' , PhoneNo varchar(50) 'PhoneNo')

EXEC sp\_xml\_removedocument @index

The above code snippet results the following:

15201 abc 9343463943

15202 xyz 9342673212

**How to store pdf file in SQL Server?**

Create a column as type ‘blob’ in a table. Read the content of the file and save in ‘blob’ type column in a table.

Or

Store them in a folder and establish the pointer to link them in the database.

**Explain the use of keyword WITH ENCRYPTION. Create a Store Procedure with Encryption.**

It is a way to convert the original text of the stored procedure into encrypted form. The stored procedure gets obfuscated and the output of this is not visible to

CREATE PROCEDURE Abc WITH ENCRYPTION AS

<< SELECT statement>>

GO

WITH ENCRYPTION indicates that SQL Server will convert the original text of CREATE PROCEDURE statement to an encrypted format. Users that do not have no access to system tables or database files cannot retrieve the encrypted text. However, the text will be available to privileged users.

Example:

CREATE PROCEDURE salary\_sum WITH ENCRYTION

AS

SELECT sum(salary) FROM employee WHERE emp\_dept LIKE Develop

**What is lock escalation?**

Lock escalation is used to convert row locks and page locks into table locks thereby “escalating” the smaller or finer locks. This increases the system performance as each lock is nothing but a memory structure. Too many locks would mean more consumption of memory. Hence, escalation is used.

Lock escalation from SQL Server 7.0 onwards is dynamically managed by SQL Server. It is the process of converting a lot of low level locks into higher level locks.

**What is Failover clustering overview?**

Failover clustering is mainly used for data availability. Typically, in a failover cluster, there are two machines.

l One machine provides the basic services and the second is available to run the service when the primary system fails.

l The primary system is monitored periodically to check if it works. This monitoring may be performed by the failover computer or an independent system also called as cluster controller. In an event of failure of primary computer, the failover system takes control.

**What is Builtin/Administrator?**

The Builtin/Administrator account is basically used during some setup to join some machine in the domain. It should be disabled immediately thereafter. For any disaster recovery, the account will be automatically enabled. It should not be used for normal operations.

**What XML support does the SQL server extend?**

SQL Server (server-side) supports 3 major elements :

Creation of XML fragments: This is done from the relational data using FOR XML to the select query.

Ability to shred xml data to be stored in the database.

Finally, storing the xml data.

Client-side XML support in SQL Server is in the form of SQLXML. It can be described in terms of :

l XML Views : providing bidirectional mapping between XML schemas and relational tables.

l Creation of XML Templates : allows creation of dynamic sections in XML.

SQL server can return XML document using FOR XML clause. XML documents can be added to SQL Server database and you can use the OPENXML clause to display the data from the document as a relational result set. SQL Server 2000 supports XPath queries.Get to know more about SQL Techniques that can help you grow in your career.

**Difference between Primary Key and Foreign Key**

Primary key uniquely identify a record in the table. Foreign key is a field in the table that is primary key in another table. Primary Key can't accept null values. Foreign key can accept multiple null value. By default, Primary key is clustered index and data in the database table is physically organized in the sequence of clustered index. Foreign key do not automatically create an index, clustered or non-clustered. You can manually create an index on foreign key. We can have only one Primary key in a table. We can have more than one foreign key in a table.

Primary Key:

l There an only be one primary key in a table l In some DBMS it cannot be NULL - e.g. MySQL adds NOT NULL l Primary Key is a unique key identifier of the record l Primary key cannot have a NULL value. l Each table can have only one primary key. l By default, Primary key is clustered index and data in the database table is physically organized in the sequence of clustered index. l Primary key can be related with another table's as a Foreign Key. l We can generated ID automatically with the help of Auto Increment field. Primary key supports Auto Increment value.

Unique Key: l Can be more than one unique key in one table l Unique key can have null values l It can be a candidate key l Unique key can be null and may not be unique l Unique Constraint may have a NULL value. l Each table can have more than one Unique Constraint. l By default, Unique key is a unique non-clustered index. l Unique Constraint can not be related with another table's as a Foreign Key. l Unique Constraint doesn't supports Auto Increment value.

A PRIMARY Key and UNIQUE Key constraints both are similar and it provide unique enforce uniqueness of the column on which they are defined.

Foreign Key l Foreign key is a field in the table that is primary key in another table. l Foreign key can accept multiple null value. l Foreign key do not automatically create an index, clustered or non-clustered. You can manually create an index on foreign key. l We can have more than one foreign key in a table. l There are actual advantages to having a foreign key be supported with a clustered index, but you get only one per table. What's the advantage? If you are selecting the parent plus all child records, you want the child records next to each other. This is easy to accomplish using a clustered index.

l Having a null foreign key is usually a bad idea. In the example below, the record in [dbo].[child] is what would be referred to as an "orphan record". Think long and hard before doing this.

**SQL Queries**

**SQL Query to find second highest salary of Employee**

There are many ways to find second highest salary of Employee in SQL, you can either use SQL Join or Subquery to solve this problem. Here is SQL query using Subquery:

Method 1:

1. select \* from employees emp1 where 1 = (select count(DISTINCT(emp2.salary)) from employees emp2 where emp2.salary > emp1.salary)

Method 2:

select top 2 salary from employees emp order by sal desc

Method 3:

select MAX(Salary) from Employee WHERE Salary NOT IN (select MAX(Salary) from Employee );

**SQL Query to find Max Salary from each department.**

You can find the maximum salary for each department by grouping all records by DeptId and then using MAX() function to calculate maximum salary in each group or each department.

SELECT DeptID, MAX(Salary) FROM Employee GROUP BY DeptID.

These questions become more interesting if Interviewer will ask you to print department name instead of department id, in that case, you need to join Employee table with Department using foreign key DeptID, make sure you do LEFT or RIGHT OUTER JOIN to include departments without any employee as well. Here is the query

SELECT DeptName, MAX(Salary) FROM Employee e RIGHT JOIN Department d ON e.DeptId = d.DeptID GROUP BY DeptName;

In this query, we have used RIGHT OUTER JOIN because we need the name of the department from Department table which is on the right side of JOIN clause, even if there is no reference of dept\_id on Employee table.

**Write SQL Query to display the current date.**

SQL has built-in function called GetDate() which returns the current timestamp. This will work in Microsoft SQL Server, other vendors like Oracle and MySQL also has equivalent functions. SELECT GetDate();

**Write an SQL Query to check whether date passed to Query is the date of given format or not.**

SQL has IsDate() function which is used to check passed value is a date or not of specified format, it returns 1(true) or 0(false) accordingly. Remember ISDATE()is an MSSQL function and it may not work on Oracle, MySQL or any other database but there would be something similar.

SELECT ISDATE('1/08/13') AS "MM/DD/YY";

It will return 0 because passed date is not in correct format.

**Write an SQL Query to print the name of the distinct employee whose DOB is between 01/01/1960 to 31/12/1975.**

This SQL query is tricky, but you can use BETWEEN clause to get all records whose date fall between two dates.

SELECT DISTINCT EmpName FROM Employees WHERE DOB BETWEEN ‘01/01/1960’ AND ‘31/12/1975’;

**Write an SQL Query find number of employees according to gender whose DOB is between 01/01/1960 to 31/12/1975.**

SELECT COUNT(\*), sex from Employees WHERE DOB BETWEEN '01/01/1960' AND '31/12/1975' GROUP BY sex;

**Write an SQL Query to find an employee whose Salary is equal or greater than 10000.**

SELECT EmpName FROM Employees WHERE Salary>=10000;

**Write an SQL Query to find name of employee whose name Start with ‘M’**

SELECT \* FROM Employees WHERE EmpName like 'M%';

**find all Employee records containing the word "Joe", regardless of whether it was stored as JOE, Joe, or joe.**

SELECT \* from Employees WHERE UPPER(EmpName) like '%JOE%';

**Write an SQL Query to find the year from date.**

Here is how you can find Year from a Date in SQL Server 2008

SELECT YEAR(GETDATE()) as "Year";

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

Example will be -

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

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

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

Common records result set can be achieved by -.

1 Select studentID from student. **INTERSECT**Select StudentID from Exam

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

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

To display even numbers-.

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

To display odd numbers-.

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

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

Select unique records from a table by using DISTINCT keyword.

1 Select DISTINCT StudentID, StudentName from Student.

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

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

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

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

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

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

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

Example -.

1 Select \* from Student where studentname like ‘a%’

1 Select \* from Student where studentname like ‘ami\_’

**Write SQL Query to find duplicate rows in a database? and then write SQL query to delete them?**

You can use the following query to select distinct records:

SELECT \* FROM emp a WHERE rowid = (SELECT MAX(rowid) FROM EMP b WHERE a.empno=b.empno)

to Delete:

DELETE FROM emp a WHERE rowid != (SELECT MAX(rowid) FROM emp b WHERE a.empno=b.empno);

**There is a table which contains two column Student and Marks, you need to find all the students, whose marks are greater than average marks i.e. list of above average students.**

This query can be written using subquery as shown below:

SELECT student, marks from table where marks > SELECT AVG(marks) from table)

**How do you find all employees which are also manager? .**

You have given a standard employee table with an additional column mgr\_id, which contains employee id of the manager.

You need to know about self-join to solve this problem. In Self Join, you can join two instances of the same table to find out additional details as shown below

SELECT e.name, m.name FROM Employee e, Employee m WHERE e.mgr\_id = m.emp\_id;

this will show employee name and manager name in two column e.g.

name manager\_name

John David

One follow-up is to modify this query to include employees which don't have a manager. To solve that, instead of using the inner join, just use left outer join, this will also include employees without managers.

**You have a composite index of three columns, and you only provide the value of two columns in WHERE clause of a select query? Will Index be used for this operation?**

For example if Index is on EmpId, EmpFirstName, and EmpSecondNameand you write query like

SELECT \* FROM Employee WHERE EmpId=2 and EmpFirstName='Radhe'

If the given two columns are secondary index column then the index will not invoke, but if the given 2 columns contain the primary index(first column while creating index) then the index will invoke. In this case, Index will be used because EmpId and EmpFirstNameare primary columns.

**What is the default join in SQL? Give an example query?**

The default join is INNER JOIN.

Example

SELECT column\_name(s)  FROM table1  INNER JOIN table2  ON table1.column\_name=table2.column\_name;

**Describe all the joins with examples in SQL?**

• SQL LEFT JOIN

The LEFT JOIN keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match.

1. SQL LEFT JOIN Syntax

SELECT column\_name(s)  FROM table1 LEFT JOIN table2  ON table1.column\_name=table2.column\_name;

• SQL RIGHT JOIN

The right join returns all the rows in the right table i.e. table 2 with the matching ones on the left table (table1).

SELECT column\_name(s)  FROM table1  RIGHT JOIN table2 ON table1.column\_name=table2.column\_name;

• SQL FULL OUTER

The full join returns all rows from the left table (table1) and from the right table (table2).

SELECT column\_name(s)  FROM table1  FULL OUTER JOIN table2  ON table1.column\_name=table2.column\_name;

**What is Union and Union All ? Explain the differences?**

• SQL UNION

The UNION operator is used to combine the result-set of two or more SELECT statements.

Notice that each SELECT statement within the UNION must have the same number of columns. The columns must also have similar data types. Also, the columns in each SELECT statement must be in the same order.

Note: The UNION operator selects only distinct values by default.

SELECT column\_name(s) FROM table1  UNION  SELECT column\_name(s) FROM table2;

• SQL UNION ALL

1. SQL UNION ALL Syntax

SELECT column\_name(s) FROM table1   UNION ALL  SELECT column\_name(s) FROM table2;

Allows duplicate values.