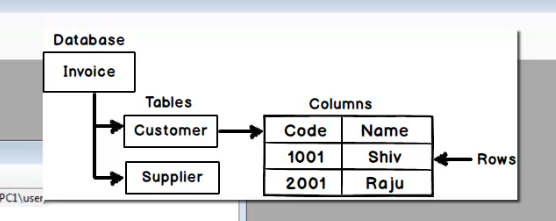
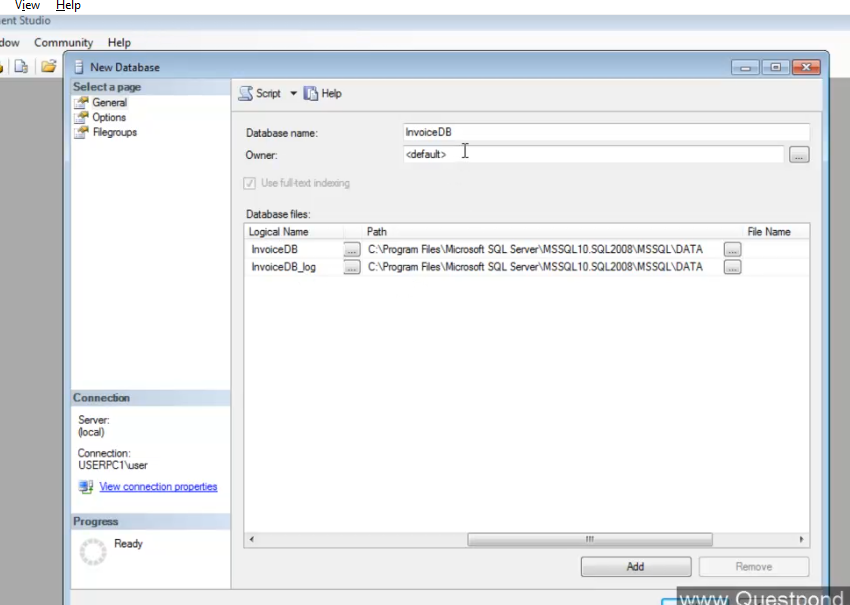


Heart of the Db handling core things run at the background

When we connect to DB engine basically running this service from the background

Data is organized as 

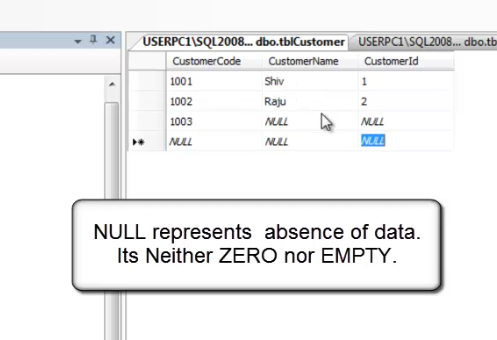
Every data base you create two files created on for data and another for the log



**NULLS:**

Data is not available

In Db Null means absence of value its neither zero or nor empty

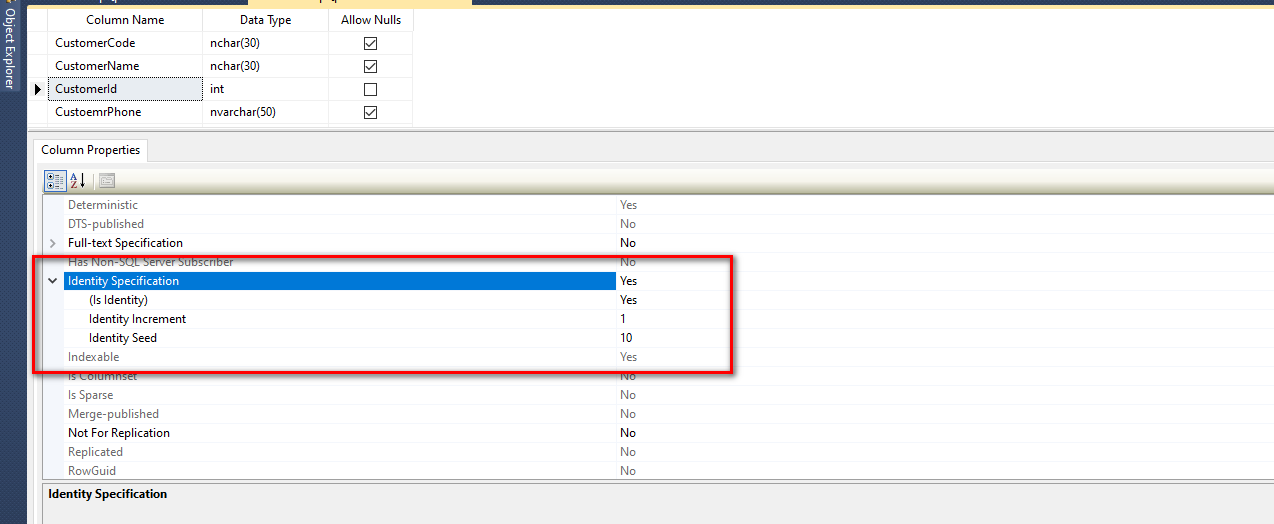


CTRL +0 inserts null to the column selected.

**Identity column:**

When we say a column as identity the value of the column will be inserted automatically with incremented value and it will also keep the log of value has been generated so no value is repeated.

Always numeric type cannot have null



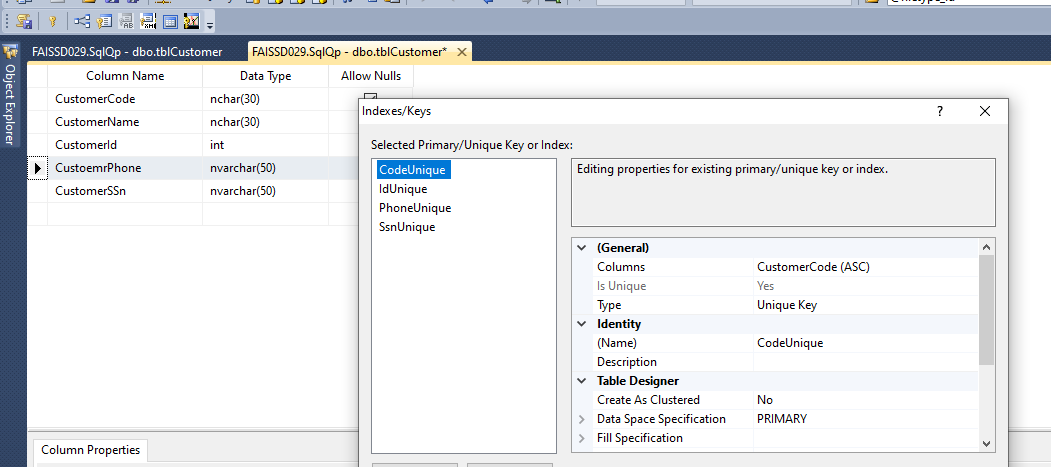
**Unique column/ unique key**

This indicates that the value entered for this field should be unique and we cannot repeat the value

Right click on the column and say index/keys

Can have null value

Say I have made column as unique and nullable then we can keep one record with null as value and one record with empty as value.

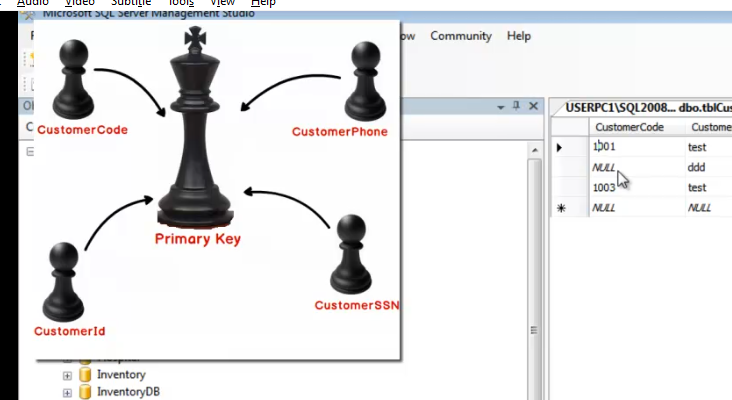


**Primary key:**

Used to uniquely locate the record in the db

1. Null is not allowed
2. Should be unique
3. It cannot be modified

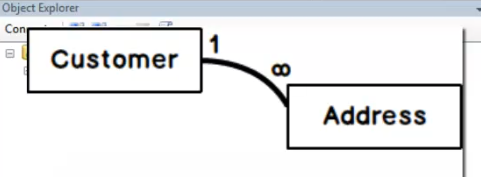
Because based on this we make indexing, references



**Candidate Key:**

List of unique key in the in the table that can compete to become the primary key.

All unique key including primary key is a member of primary key.



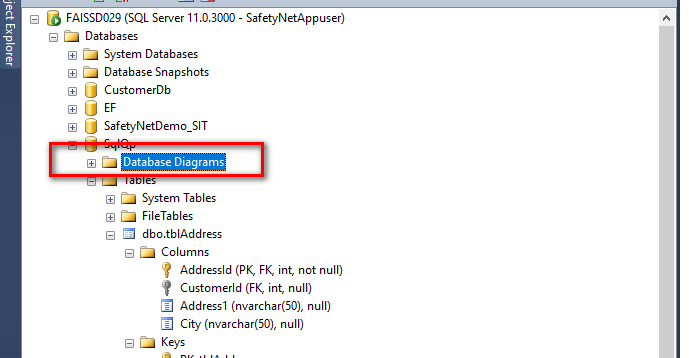
**Foreign key (referential integrity)**

Referential integrity constrains makes sure that if two tables are referencing to each other then it should refer to the correct value in the parent table.

Or else no reference record will be created

Key used for this scenarios is called reference key or foreign key

**Database diagram:**



Graphical representation of the tables in the data base

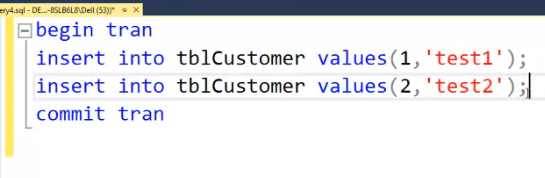
Composite key:

Suppose if the table has combination of more that one column as primary key such key are called as composite key.

Transaction locks Isolation level:

Set of task grouped under single execution unit

Either all the taks will be commited r failed if any operation fails



Nested transaction:

When and inner trnsaction cimmits no physical commit happens at the Db end only the transaction count will be decremented and marks for the physical commit

Finaly physical commit happens only when the commit triggered from the out most transaction.

If the roll back staement issued from any inner transaction then everything will be rolled back

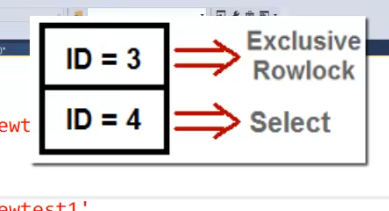
Check points in transaction:



Concurrency:

Multiple user trying to access same information.

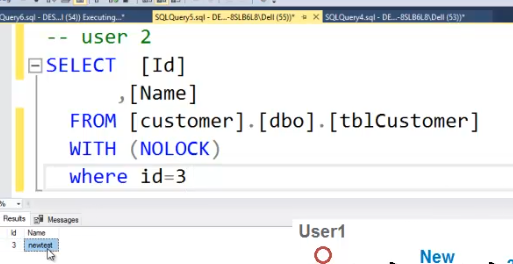
When transaction runs it puts a default lock on the record which involved in the operation. Because by default read is read committed mode



Since the read operation is read committed. There is a possiblity that read operation moved to waiting state when other peration performing any operation.

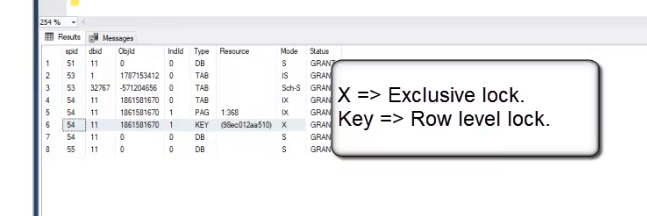
Spo to avoid read commite and to read the volatile data we can Use nolock statement in my Sql operation.

SO this is called read uncommited operation this will view the volatile data.



SP\_Lock: Gets you all the information aboutcurrent locks in the

SP\_Who: gives me the information about whos is actively involved in that lock



Deadlock:

Two transactions are blocking the progress of each other is called a deadlock

This is a scenario when in one transaction is waiting for the other transaction to release the record o which it ha put the exclusive access.

Stop deadlock:

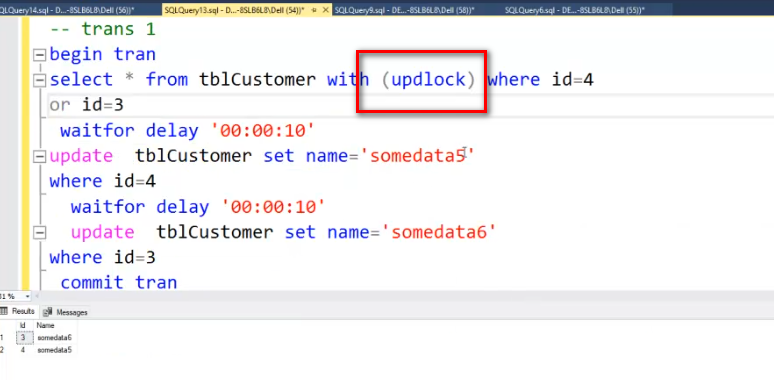
1. Try to reduce the time of execution
2. Keep the operation in same chronological order.
3. Use update lock

So in this approach basically we fetch the record from the Db an lock it for update using (updlock)

So that nobody can perform update operation on this record unless it is released by the transaction

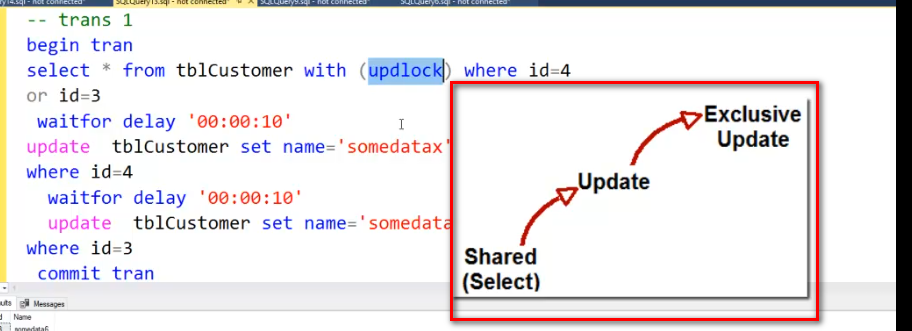
But mean time other can fetch this record using select for read operation. Until update lock is changed to exclusive lock

This way update operation will be blocked from other transaction so no deadlock will happen



Shared lock similar to update lock only difference is the scope of the lock is till you rin the select operation.

While running this select operation no transaction can perform update delete select operation



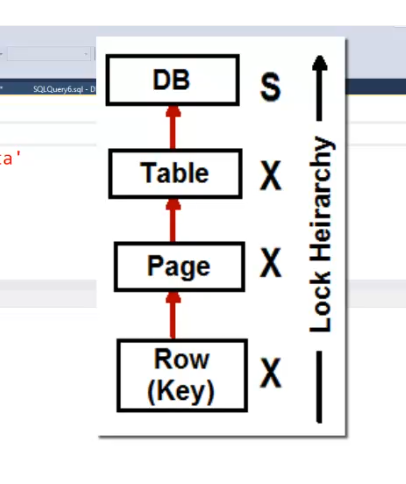
So when we perform this update lock

1.Initially shared lock will be applied when we perform select operation

2. while performing other operation updated lock will be applied

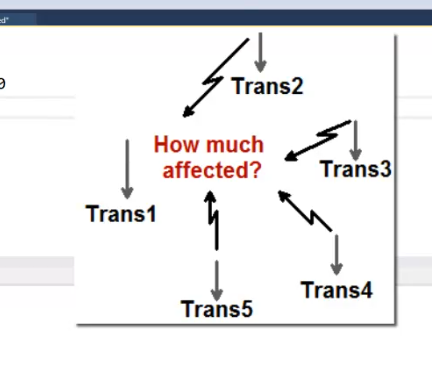
3. finally in run of update operation exclusive lock will be applied on the record.

Lock hierarchy:



Isolation level:

Basically what kind of change you want to see from the other transaction



3 things happens when you run multiple transaction

1. Repeatable read

Inside transaction im reading same data multiple time and it returns different data modified by the other transaction

1. Dirty read

One transaction perform select operation. Where in other never commits its data

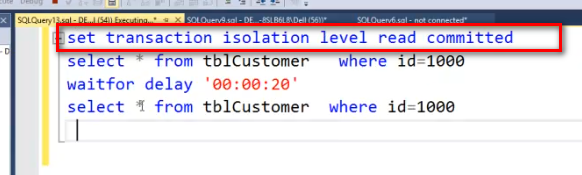
Basically transaction 1 is fetching the uncommitted data

Eg: transaction 1 performs data update and rolls back it after 20 sec

Mean time transaction 2 fetched the data without any lock this read is dirty read because it reads the uncommitted data.

1. Phantom reads:

When query executed records in not available after some time when you run the query record appears suddenly.



Different isolation level



ACID properties:

Atomicity: either commit everything or Rolle back everything (Begin tran……..commit tran)

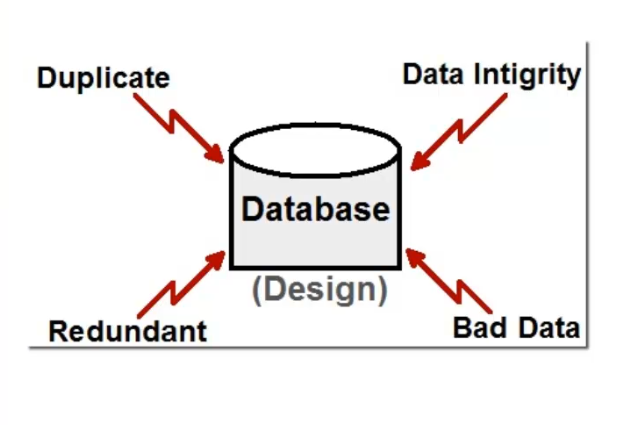
Consistent: create valid state of the data or else leave db as it is before the transaction is run.

Isolation: there should be clear isolation level between 2 transactions running parallel.

Durability: committed data must be saved to the db and in case of failure we need to make sure its available to the user.

Mirror, backup, replication.

**Data Base normalization:**



Design issues:

1. Do not put proper primary key and unique and candidate key or else this will lead to a duplicate value (eg Aadhar number is unique for every one so two person cannot have same aadhar. SO make this column unique preventing the value being duplicated)
2. Column should have atomic value don’t put multiple values in a single column
3. There shouldn’t be any redundant value this will lead more memory being consumed by the Db (eg: person profile with city and country as column if it is a text column on person we may end up inserting same text multiple times.)

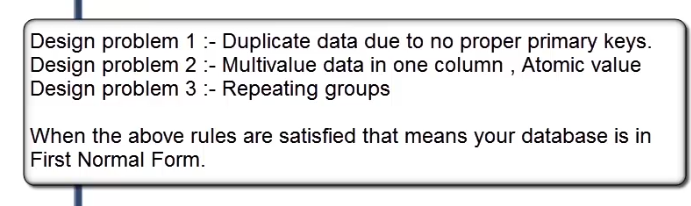
**Duplicate data Vs redundant data:**



Redundant column leads to redundant data

To solve this we move such columns to new table this process is called normalization.

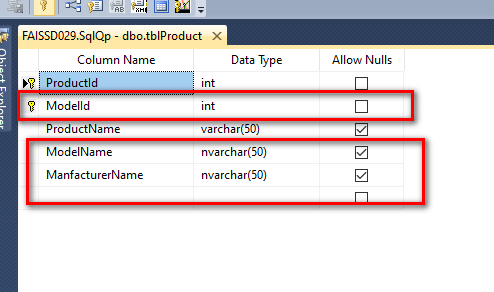
If the table design follow all the below condition then we call it as the tables are in first normal form



Prime attribute is the one used to uniquely identify or locate the record in the Db

1. Second normal form says that every nonprime attribute is fully dependent on primary key. There shouldn’t be any partial dependency.

If not, then there are two entities present in the table save in same table which lead to problem in future changing one table lead to changing the other.

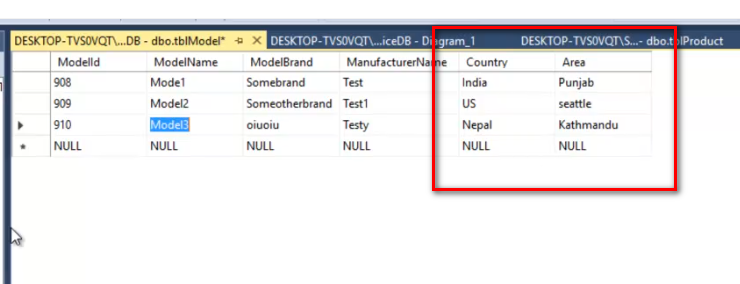


So in one table we can save only one entity there shouldn’t be more than two entity.

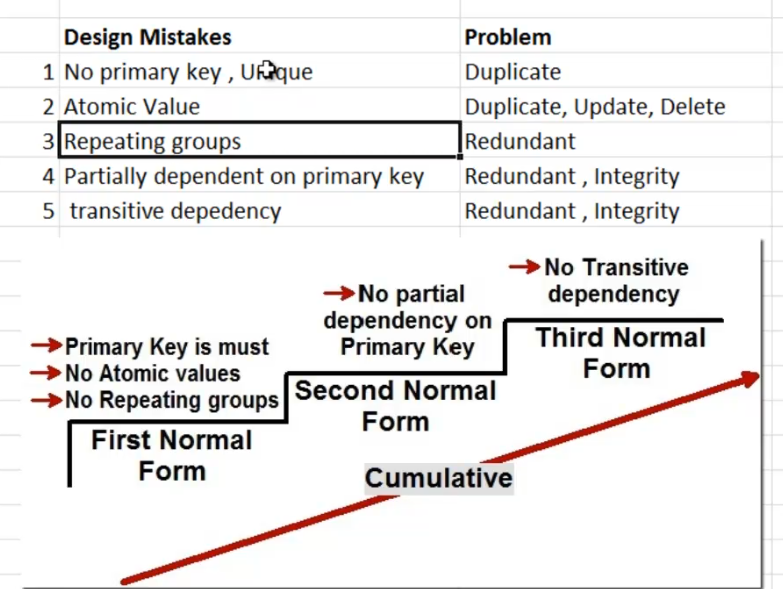
1. Transitive dependency:

Where a non prime attribute is dependent on another nonprime attribute is called as transitive dependency

In below example city is dependent on country and both are non prime attributes of the table soe we need to move this entity out of this table.



Summery:



Break tab into number of smaller functional units so that each table represent on entity.

**Normalization**. **Normalization** is the process of organizing the data in the database. **Normalization** is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.

Sql operation:

Distinct used to get unique set of values, If I put this on multiple selected column then its applied to all combination of selected value from the list.

Patter matching wild card

% ‘%s’ ends with s

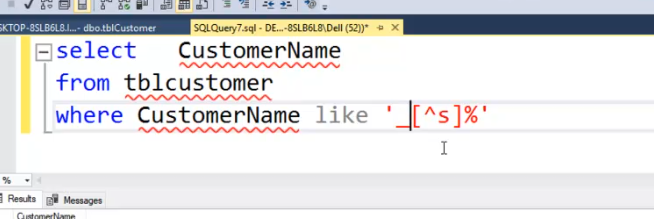
\_ ‘S\_m’ word stats with s ends with m and in between there can be any character present.

We use like key word for the patter matching

Where name like ‘s%’

Suppose you want to negate the like operation

Where name like ‘[^n]%’ this will get me al names that doesn’t start with S

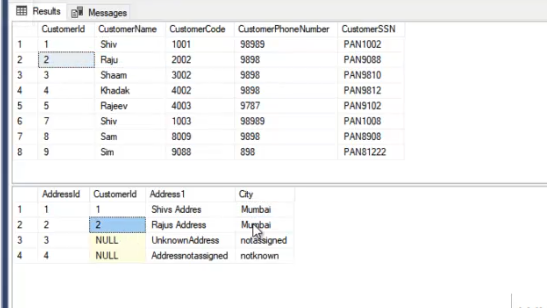


Union & union all

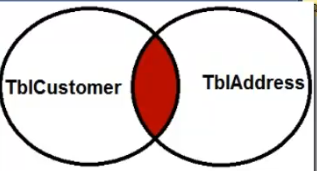
Use to club the output of the two Sql query and return is as single data table. But while using this our select statement should have union compatibility

I.e. Number of columns selected from both query should be same and their domain should match with each other.

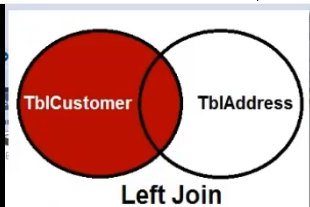
Joins



Inner join normal join : This join will return only matching records from both table



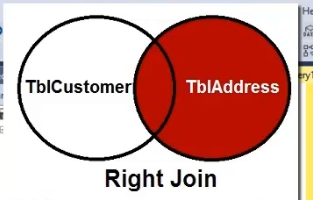
Left join and right join:



Select all record from left table but only matching from right table

For the right table that doesn’t have matching value such table columns mapped with null as value

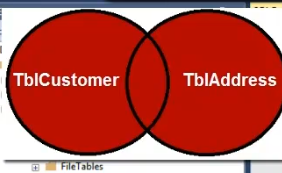
Right join



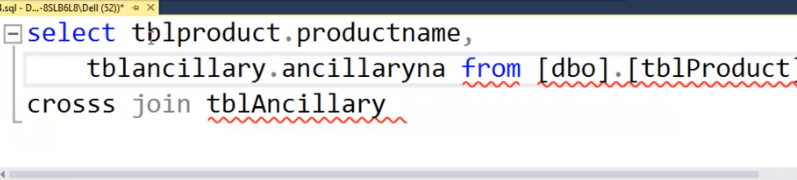
Full outer join:

Returns all matching record and un matching record from the table

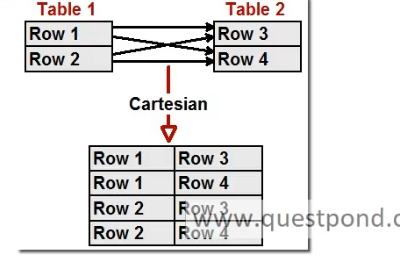
If there is a match returns proper record for that.



Cross join or Cartesian product



Makes all X\*X combination and return all combination of the rows from the join operation.

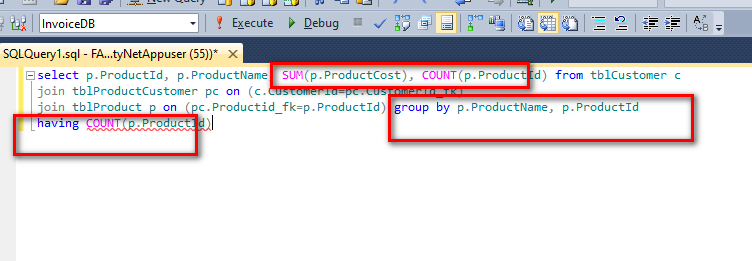


While using group by put only those columns in selected list which are part of group by clause. We can have aggregate function in select list part from column listed in group by clause.

Filter used for group by clause is

Having

Aggregate functions Max, min, sum, count , avg



Self join:

Parent child relation

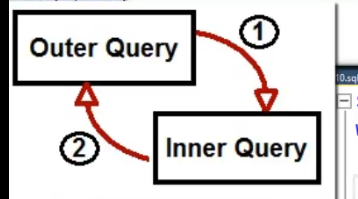
Sub query: Here inner query is evaluated first and the result of inner query is fed to the outer query.

Co-related query:

In co-related query outer query is executed first and the result is passed to the inner query

Example is finding the 1st highest and second highest price of the product even for employee salary.

select \* from tblProduct p1 where 3=(select COUNT(\*) from tblProduct p2 where p2.ProductCost>= p1.ProductCost)



DM L: data manipulation language

Insert

Update

Delete

Truncate

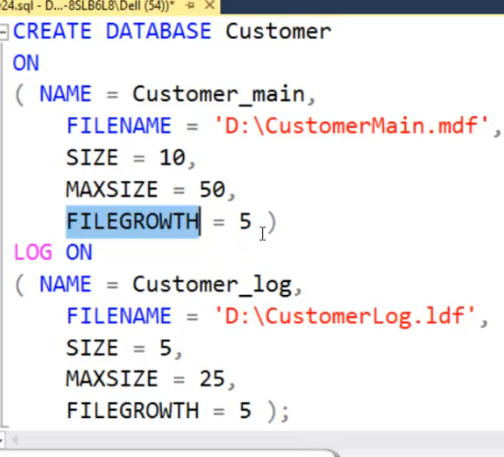
DDL : data defination lanaguage

Used to create modify, alter the structure of data bale

Create

Alter

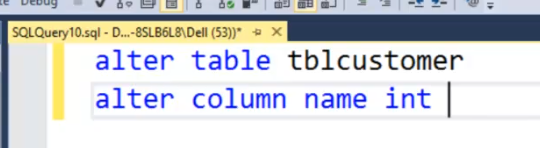
Drop

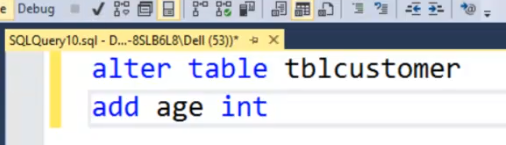


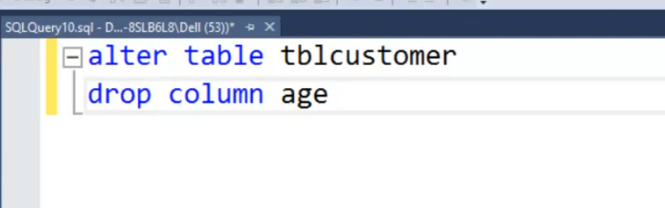
Data base backup command

backup database InvoiceDB to disk='F:\id.bak'

Alter table:

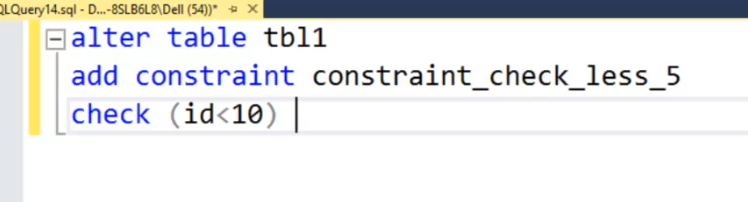




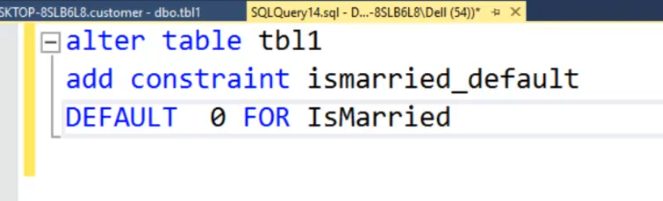


Check constraints

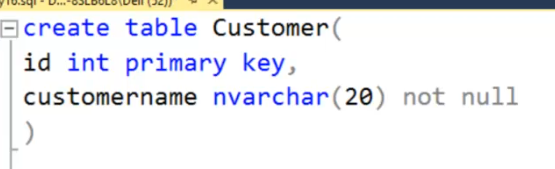
Limit the value to the column in data base

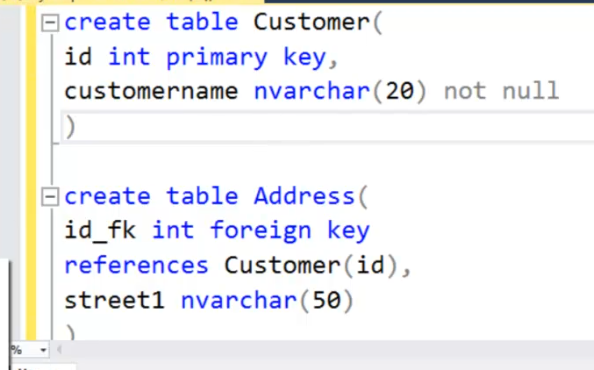


Default constraint:



Primary and foreign key constaint





Isnull vs Coalesce

Is null can take only two column name as parameter and it retunes second parameter passed a result if the first parameter value is null

Coalesce: function returns first none null column from the list of column mentioned as parameter to the function

**Row\_Number():**

Unique number for the row is generated when the select query runs in the order in which the records are returned from the query.

There has to the over clause when we use row number where in we specify the ordering of the column

select ROW\_NUMBER() over (order by name asc) as rowNum,name, color,created\_by from ask\_filetype

**Partition:**

Makes a partition on top of generated row number. Based on the column selected for the partition.

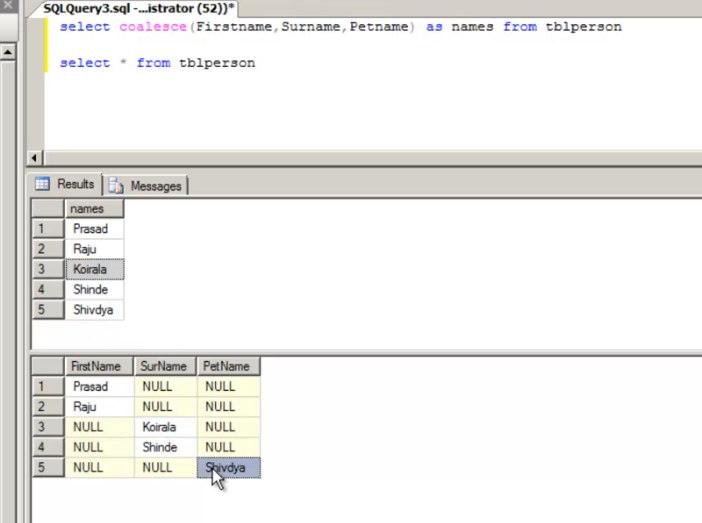
**Rank:**

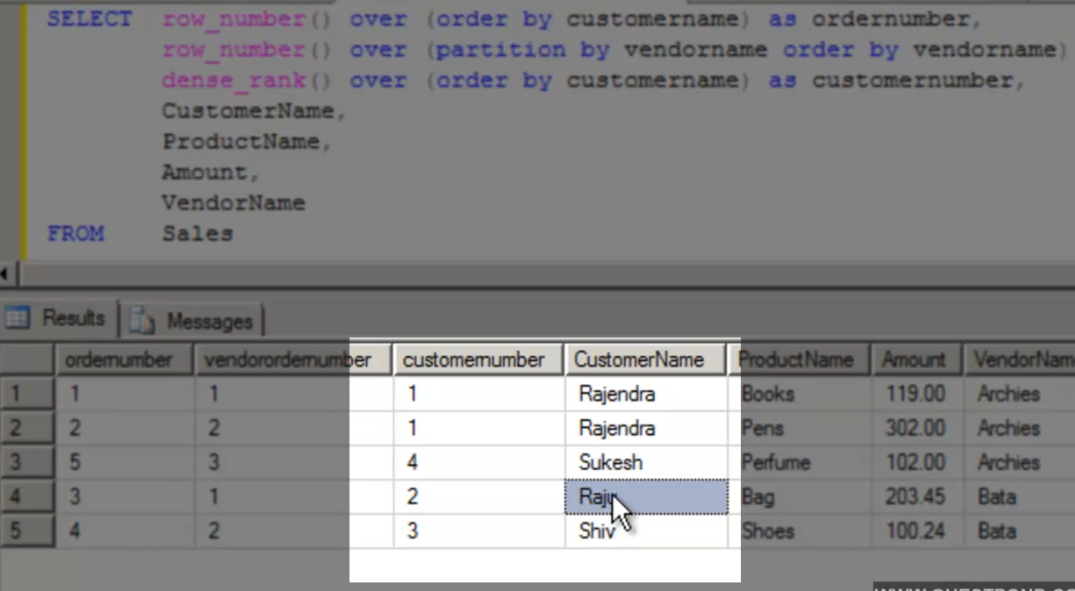
With this we can generate the unique number for the particular column value and it generated sane number if the value repeats.

But here we have a problem when the same record value repeats. For the next unique value It skips the number incremented and assign the respective value generated from the increment.

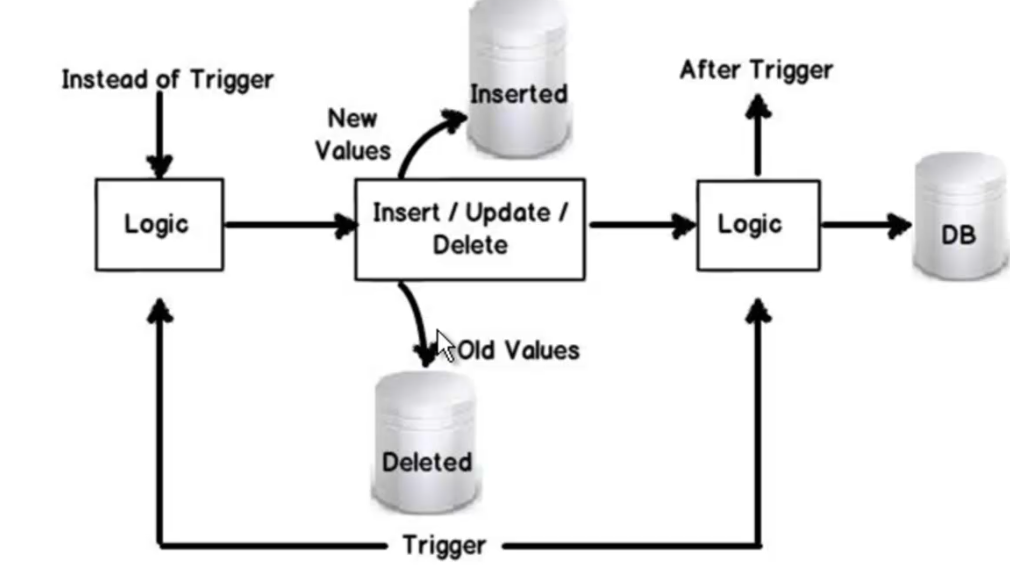
**Dense\_Rank():**

Dense rank is similar to the rank but it generated the unique numbers in sequence and will not skip any numbers.





Triggers:



Similar to Sp that run some logic. But this execution of the logic happens before or after execution on the insert or update

2 type:

1.Instead of trigger (logic is executed before the record is inserted or updated)

2. After triggers (executed when record is inserted or updated on the table)

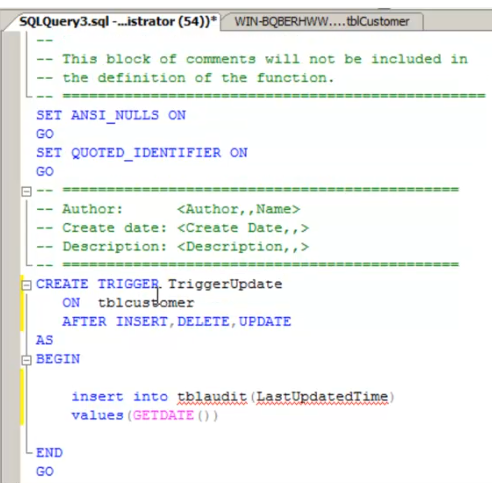
While crating trigger

We need name

On which table

When it has to executed

And for which operation

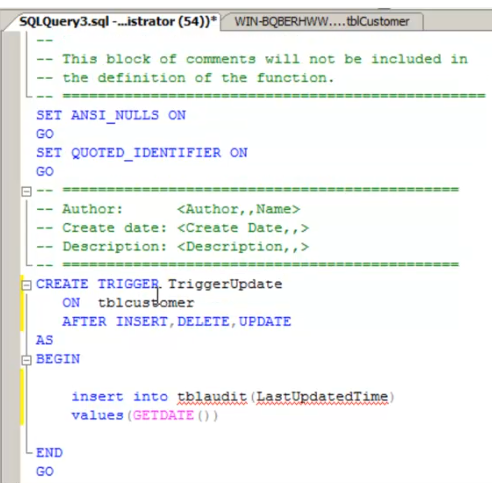


Temporary table created by sql server to find the record being inserted or deleted so that we can track the changes made to the record.

INSRTED: new record will be here when insert/update fires

DELETED: Old record which is deleted will be here

**Instead of trigger and after trigger:**



Instead of trigger will override the operations mentioned in the trigger

Instead of Insert, update, delete

This says instead of above operations you run the logic define in the trigger. So it will override the default action of query we are responsible for handling the above operation in our triggers.

Used in the scenarios where in you want to perform some alternative action before insert or update basically validation of the fields values and reject if there is error.

In delete operation suppose if you want to do cascade delete or delete child records when the parent is deleted then we go for the delete triggers.

Denormalization:

Used for the olap system where processing and generating the report is important so we violate all the normalization principle to improve the search result.

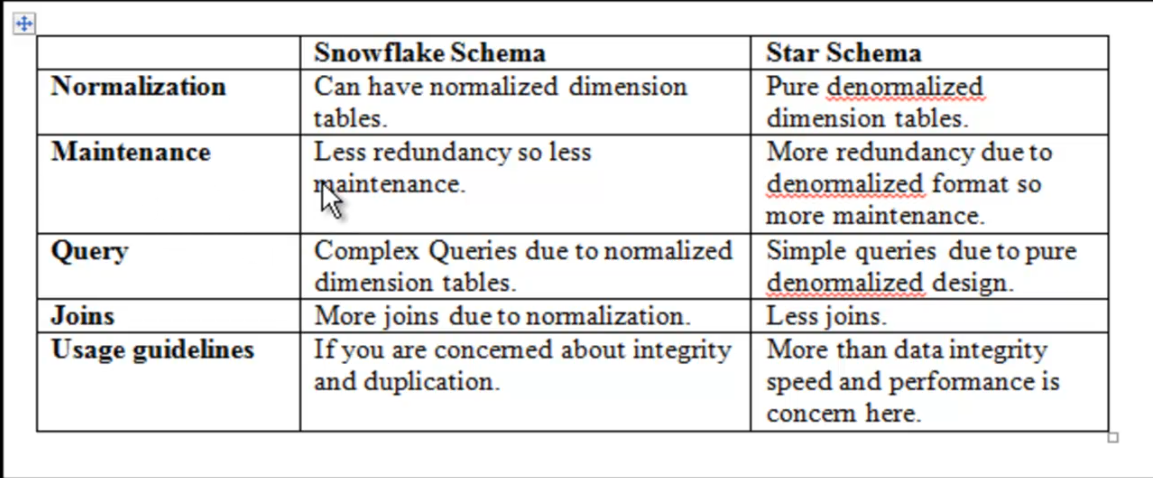
OLTP: online transaction processing. Deals with storage and performing transaction

OLAP: online Analytical processing: used only for the reporting and data generation for the report so quick out put need to be generated from this.

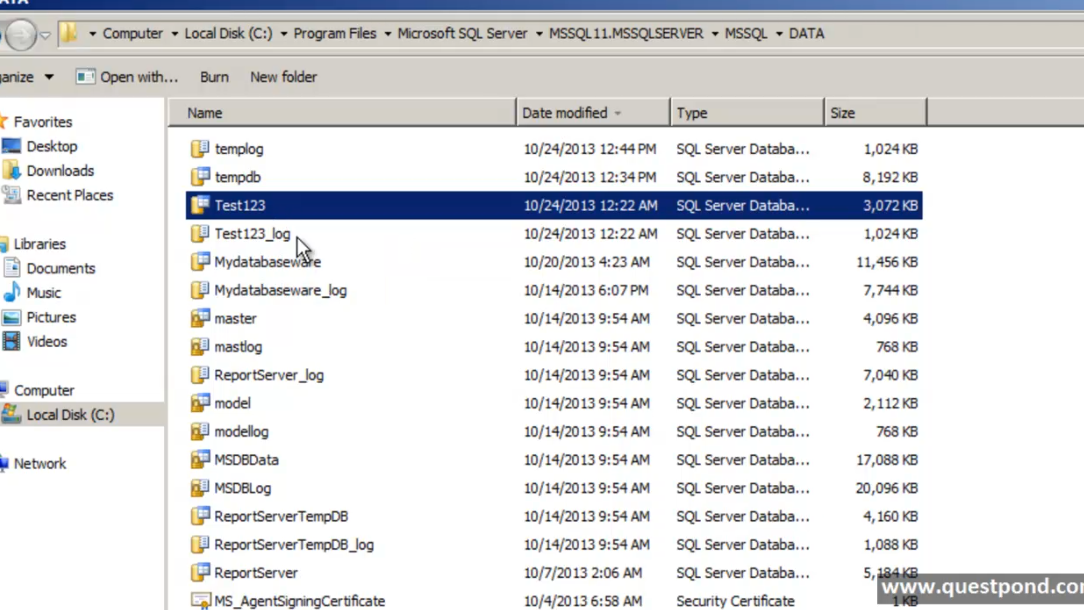
Star Schema VS snow flaks

Dimensions: More info about measures.

Measures: numbers for

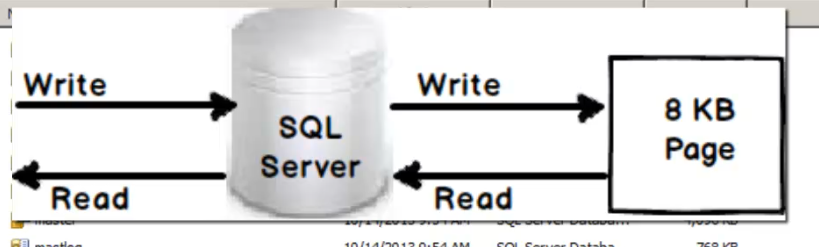


How Sql server store manages data internally



Data is present in .mdf file

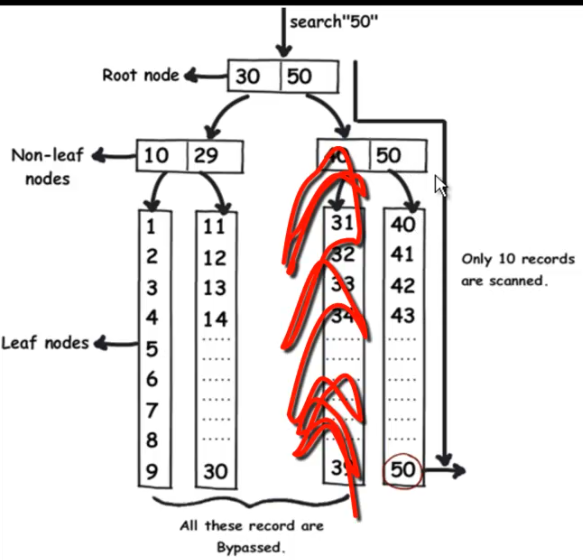
Log in .ldf



Indexing in DB:

Main goal of the index is to make the search operation faster by building the B-tree or balanced tree structure in DB.

Balanced tree:



Without index search is called table scan

With index search is called as index seek.

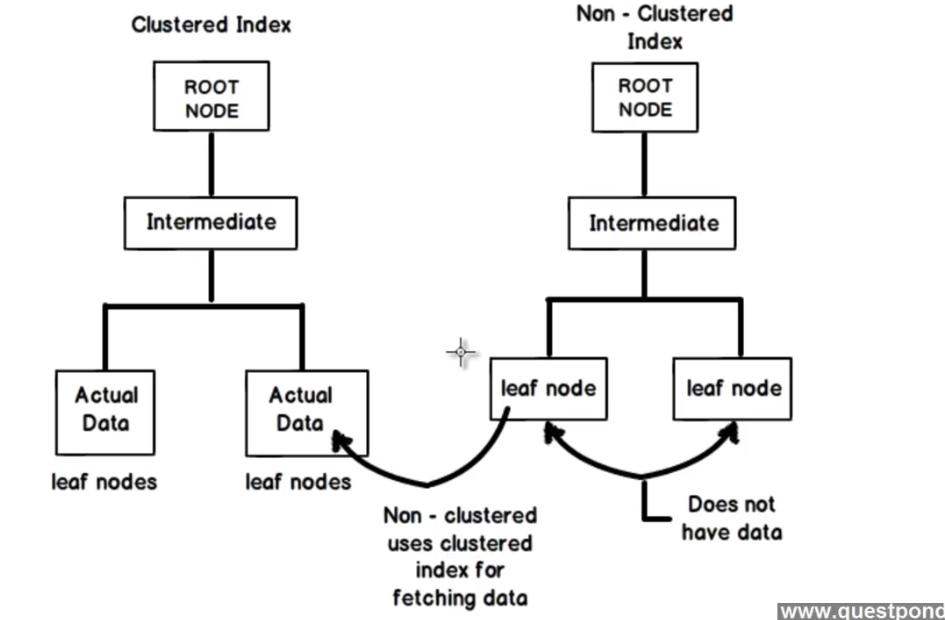
Impact of Indexing on insert update and delete:

Types of indexing:

1. Clustered index (point towards actual data)
2. Non clustered index (point to clustered index to fetch the data)

B tee structure is same for both

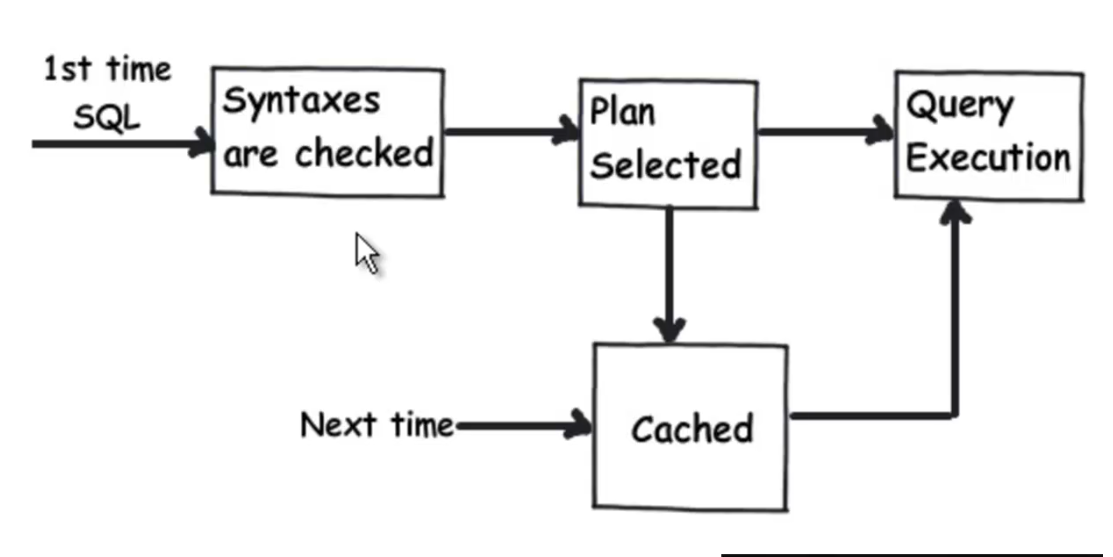
Difference is the ways leaf node works



Stored procedure:

When we run any Sql statement following actions will be executed in sequence

1. Check for syntax
2. Select the execution plan. I.e basically tells if we need to go for table scan or index seek once the plan is finalized
3. Then it will run

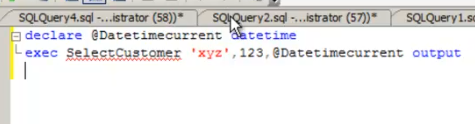


Advantages:

1. Performance (syntax check is done t creation time, execution plan is cached at first run)
2. Code is centralized changes done here will no affect the code done in programming language.

Parameter to SP:

1. Input parameter: data passed to SP
2. Output parameter: get the data back from SP



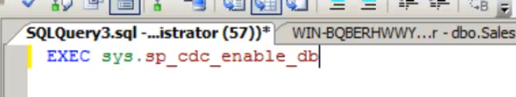
1. Return parameter: to return the value from the SP.

CDC: change data capture

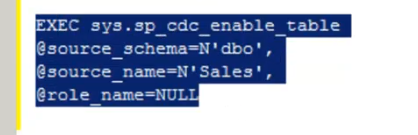
Helps you to capture the changes in data when any record is updated or delete or inserted into the db.

To use this first we need to enable cdc on the table or Db

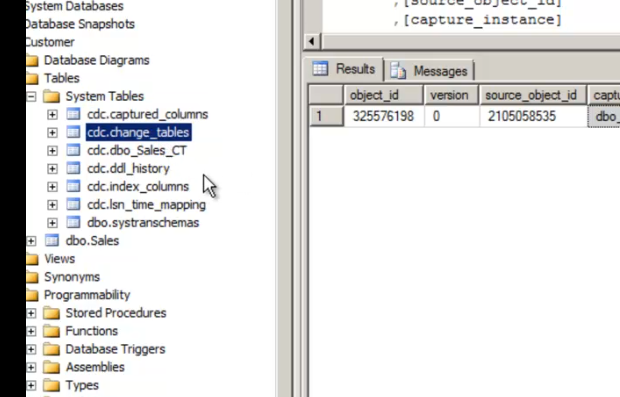
Use is auditing purpose



For table level



Exec SP\_CDC\_Enable\_Db



When cdc enable on a table 3 main tables come into picture are

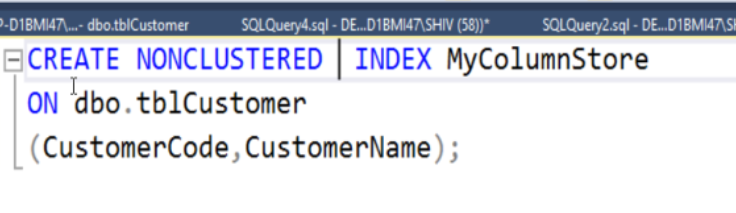
Captured\_Columns: has the list of all columns under capture

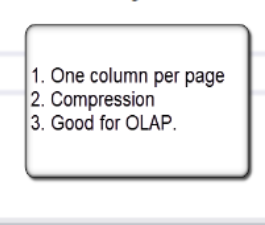
Change\_tables: this has list all the tables enabled for the change capture.

Table\_CT: it has change value record captured as a separate row in the table.

Column store index:

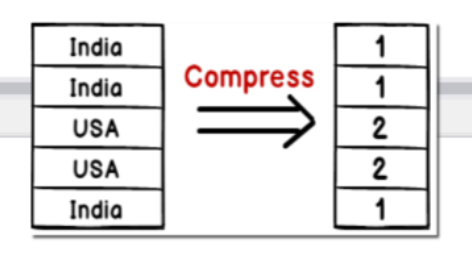
Io call comes down and performance of the query will be improved.





Here we fix e column into one page instead of the putting entire row into one page

Scope for compression

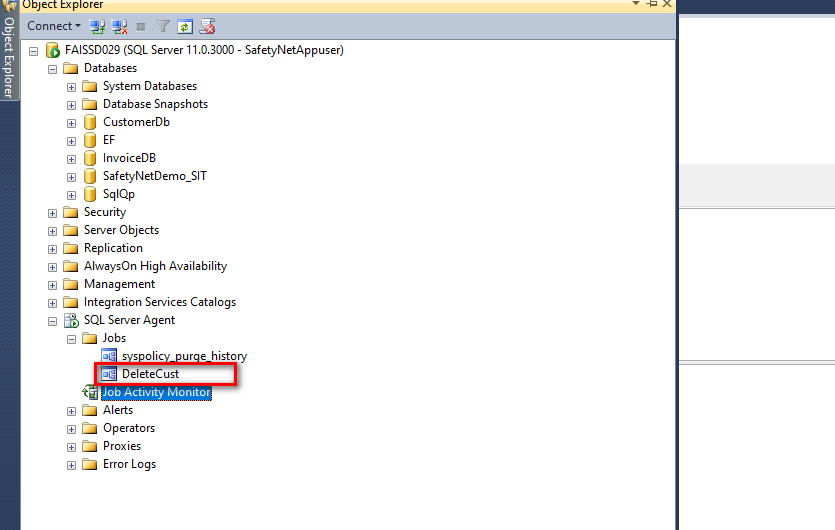


No suitable for oltp suystem

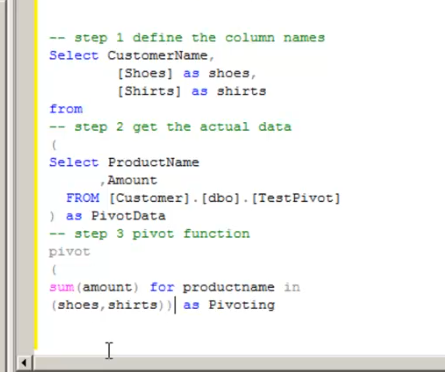
Sql server agent:

Used to run a specific job after specific interval of time. Sql server agent is used.

Say you want to run a backup job every evening



Pivot and unpivot



Sql query performance:

1. Unique key improves the table scan performance

If duplicate value end up in multiple check for the same value.

1. Table scan for small table Seek scan for table with huge data