Q= with keyword?

Ans=with keyword is used for creating the common table expression.A CTE is a temporary result set, That can be referenced with a select , insert ,update ,or delete .

With CTE\_Name (column,colue2){

AS

CTE query

Q= what is merge statement?

Ans= Merge statement is used for inserts update and delete in one statement . that means no longer have to use multiple statement for performing insert update and delete .

Merge into student as t

Using studentsource as s

On t.id = s.id

When matched then

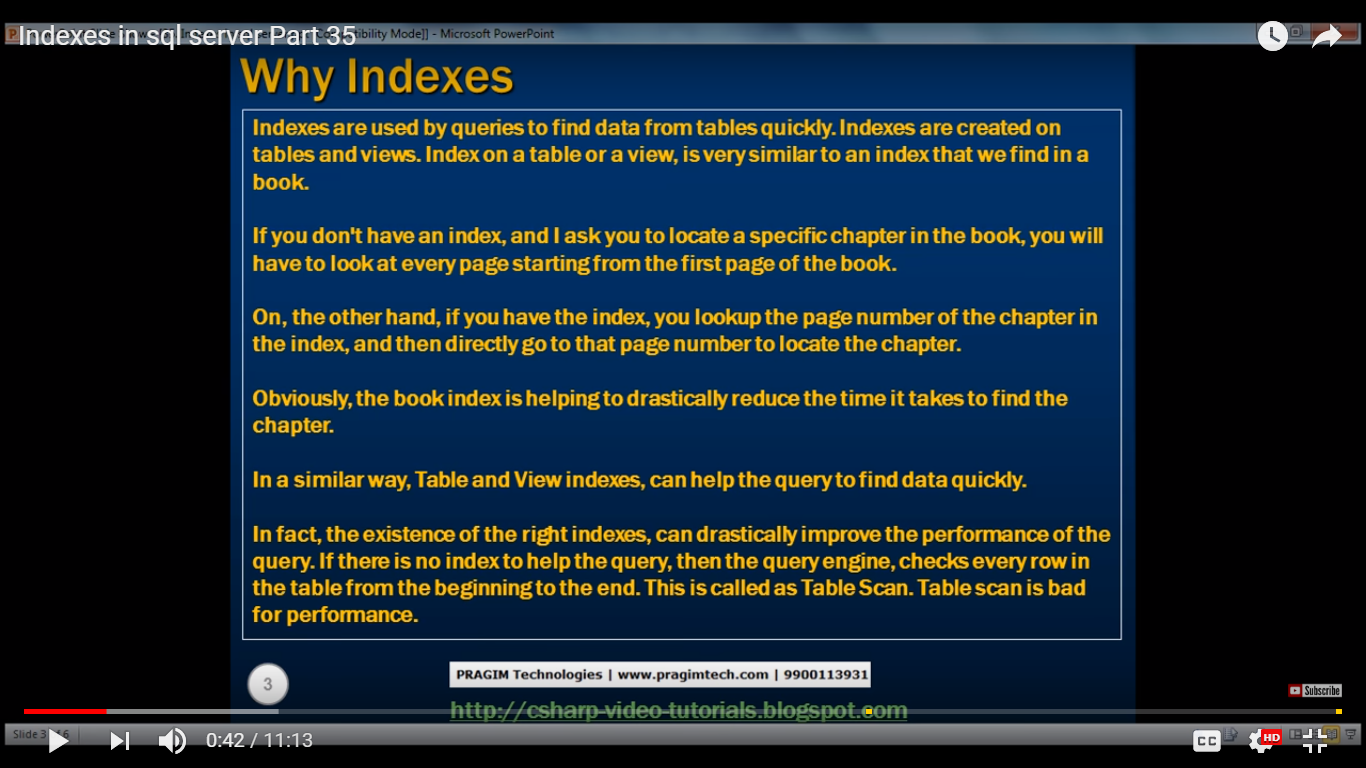
Update set t.name =s.name

When not matched by target then

Insert (id,name) values (s.id,s.name)

When not matched by source then

Delete;



Create index IC\_Tb on Table\_name (salary asc);

references intigrity

Referential integrity (RI) is a relational database concept, which states that table relationships must always be consistent. In other words, any foreign key field must agree with the primary key that is referenced by the foreign key. Thus, any primary key field changes must be applied to all foreign keys, or not at all. The same restriction also applies to foreign keys in that any updates (but not necessarily deletions) must be propagated to the primary parent key.

create table student\_marks (

rollno number references Student\_add,

subject varhcar (20),

marks number );

nth hieghest salary

Select TOP 1 salary From

(select distinct top 2 salary from employee order by salary desc) Result

Order by salary

delete duplicate row except 1 row

With EmployeeCTE AS(

select \*, ROWNUMBER() OVER (OVER PARTITION BY ID ) AS ROWNUMBER FROM EMPLOYEE

)

DELETE FROM EMLOYEECTE WHERE ROWNUMBER > 1

delete from EmpDup where EmpID in(select EmpID from EmpDup group by EmpId having

count(\*) >1)

write a query to get employees detail last 3 month hired

Select \*,DATEDIFF (MONTH,Hiredate,GETDATE()) as diff from Employee

where DATEDIFF(MONTH,hiredate,GETDATE()) between between 1 and 3

order by hiredate desc

//LAST 30 days

Select \*,DATEDIFF (DAY,Hiredate,GETDATE()) as diff from Employee

where DATEDIFF(MONTH,hiredate,GETDATE()) between between 1 and 30

order by hiredate desc

//LAST year

Select \*,DATEDIFF (YEAR,Hiredate,GETDATE()) as diff from Employee

where DATEDIFF(MONTH,hiredate,GETDATE()) between between 0 and 1

order by hiredate desc

What is Piot element

If you are using SQL Server 2005+, then you can use the PIVOT function to transform the data from rows into columns.

It sounds like you will need to use dynamic sql if the weeks are unknown but it is easier to see the correct code using a hard-coded version initially.

select \*

from

(

select store, week, xCount

from yt

) src

pivot

(

sum(xcount)

for week in ([1], [2], [3])

) piv;

get all record where name start with M

select \* from employee where name like 'M%'

select \* from Employee where charIndex('M',Name) = 1

select \* from Employee where Left (Name,1) = 'M'

select \* from Employee where substring (Name,1,1) = 'M'

There are a few technical differences. Abstract classes can still do more in comparison to Java 8 interfaces:

1. Abstract class can have a constructor.
2. Abstract classes are more structured and can hold a state.

Conceptually, main purpose of defender methods is a backward compatibility after introduction of new features (as lambda-functions) in Java 8.

Abstract classes are similar to interfaces. You cannot instantiate them, and they may contain a mix of methods declared with or without an implementation.

However, with abstract classes, you can declare fields that are not static and final, and define public, protected, and private concrete methods.

With interfaces, all fields are automatically public, static, and final, and all methods that you declare or define (as default methods) are public. In addition, you can extend only one class, whether or not it is abstract, whereas you can implement any number of interfaces.

Thread t1 = **new** Thread();

Thread.UncaughtExceptionHandler uncaughtExceptionHandler = **new** Thread.UncaughtExceptionHandler() {

@Override

**public** **void** uncaughtException(Thread t, Throwable e) {

System.***out***.printf("An exception has been captured\n");

System.***out***.printf("Thread: %s\n", t.getId());

System.***out***.printf("Exception: %s: %s\n", e.getClass().getName(), e.getMessage());

System.***out***.printf("Stack Trace: \n");

e.printStackTrace(System.***out***);

System.***out***.printf("Thread status: %s\n", t.getState());

}

};

t1.setUncaughtExceptionHandler(uncaughtExceptionHandler);

Let’s take an example of an **Employee** table:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | CREATE TABLE Employee (      EmployeeID,      EmployeeName,      DOB, -- Date of birth      DOJ, -- Date of joining      SSN, -- Social Security Number      DeptID, -- Department ID      MgrID -- Manager ID  ) |

**1. Candidate Key:** is the attribute/column or a set of attributes/columns in a relation/table that qualifies for uniqueness of each tuple/row. A relation/table can have one or more than one Candidate Keys. A Candidate key is also known as a **minimal Super key**.

Here in **Employee** table columns **EmployeeID** & **SSN** individually can maintain uniqueness in a table, thus are eligible for Candidate keys. The columns **EmployeeName + DOB** combined can also make up a Candidate Key, but there is a narrow chance that 2 Employees with same name can be born in same day.

**2. Primary Key:** is the Candidate key attribute/column that is most suited to maintain uniqueness in a table at the tuple/row level. [More about PK](https://sqlwithmanoj.com/2009/11/03/db-basics-difference-between-primary-key-unique-key/).

Here in **Employee** table you can choose either **EmployeeID** or **SSN** column for a PK, EmployeeID is preferable choice because SSN is a secure (PII) value.

**3. Alternate Key:** are the other Candidate key attribute/columns that you didn’t choose as Primary key column.

Like if you choose **EmployeeID** as a PK then **SSN** would be the Alternate key.

**4. Super Key:** is a superset of Candidate key. If you add any other attribute/column to a Candidate Key then it become a Super Key.

Like **EmployeeID + EmployeeName** is a Super Key.

**5. Composite Key:** If a table do have a single column that qualifies for a Candidate key, then you have to select 2 or more columns to make a row unique.

Like if there is no EmployeeID or SSN columns in Employee table, then you can make **EmployeeName + DOB** as a **Composite Primary Key**. But still there can be a narrow chance of duplicate rows.

<http://javaconceptoftheday.com/java-practice-questions-on-access-modifiers/>