PASS

How to write high performance queries in T-SQL

Sergio Govoni

Agenda

- Piani di Esecuzione in SQL Server
- Scrivere codice performante e misurare le prestazioni
- Set-based vs Iterative Code
- Common Table Expression (CTE)
- OVER
- APPLY
- Sargable Predicate

Ambiente di test

- Eseguire SQL Server Management Studio
  - Download: https://bit.ly/2K7Jzgu
- Connettersi all'istanza SQL Azure
  - **Server**: delphi-day-2018.database.windows.net
  - **User**: TSQLDDay2018
  - **Password**: 94483ycWSmrC22893etWS995!5k88X
  - **Database**: WideWorldImporters, AdventureWorks2017

Indici

I differenti tipi di indici, caratteristiche e struttura

Heap Table

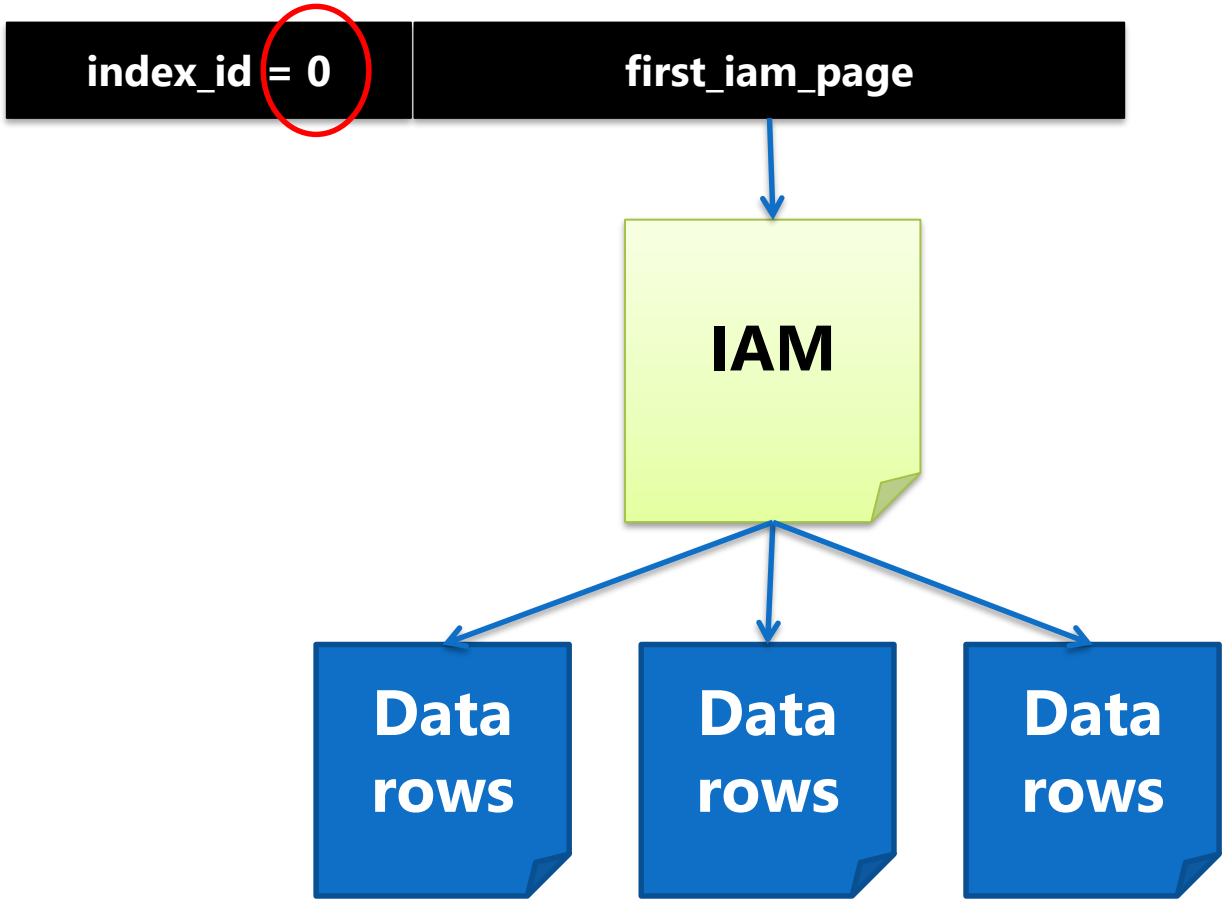
Heap

```
 8  create table dbo.A
 9  (
10    id integer identity(1, 1) not null
11    ,codice varchar(20) not null
12    ,descr varchar(40) not null
13  );
```

```
23  exec sp_helpindex 'dbo.A';
24
```

Messaggi

Non esistono indici per l'oggetto 'dbo.A' oppure non



Tipi di Indici

Clustered
Non-Clustered
Unique
Filtered
Full-Text
Spatial
XML
ColumnStore

Regole: Per ogni tabella

1 Clustered Index

- Up to 900 bytes

999 Non-Clustered (2008+)

- Up to 1.700 bytes

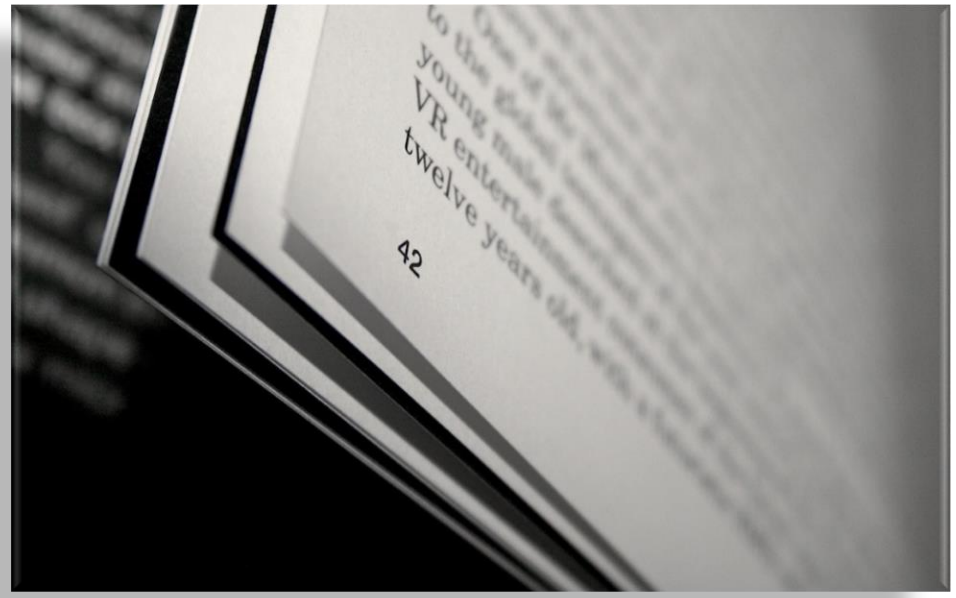
249 XML indexes

Maximum Capacity Specifications

Clustered Index: Come dovrebbe essere

Clustered Index

- Ristretto
- Univoco
- Statico
- Incrementale



Clustered Index: Come dovrebbe essere

Clustered Index

- Ristretto
  - No dati duplicati
  - Dimensione limitata
  - Ha effetto su non-clustered
  - Efficiente per range query

Univoco

Statico

Incrementale

Clustered Index: Come dovrebbe essere

Clustered Index

- Ristretto
- Univoco
  - Non richiesto, ma consigliato
  - Ristretto ricordate? ☺
- Statico
- Incrementale

Clustered Index: Come dovrebbe essere

Clustered Index

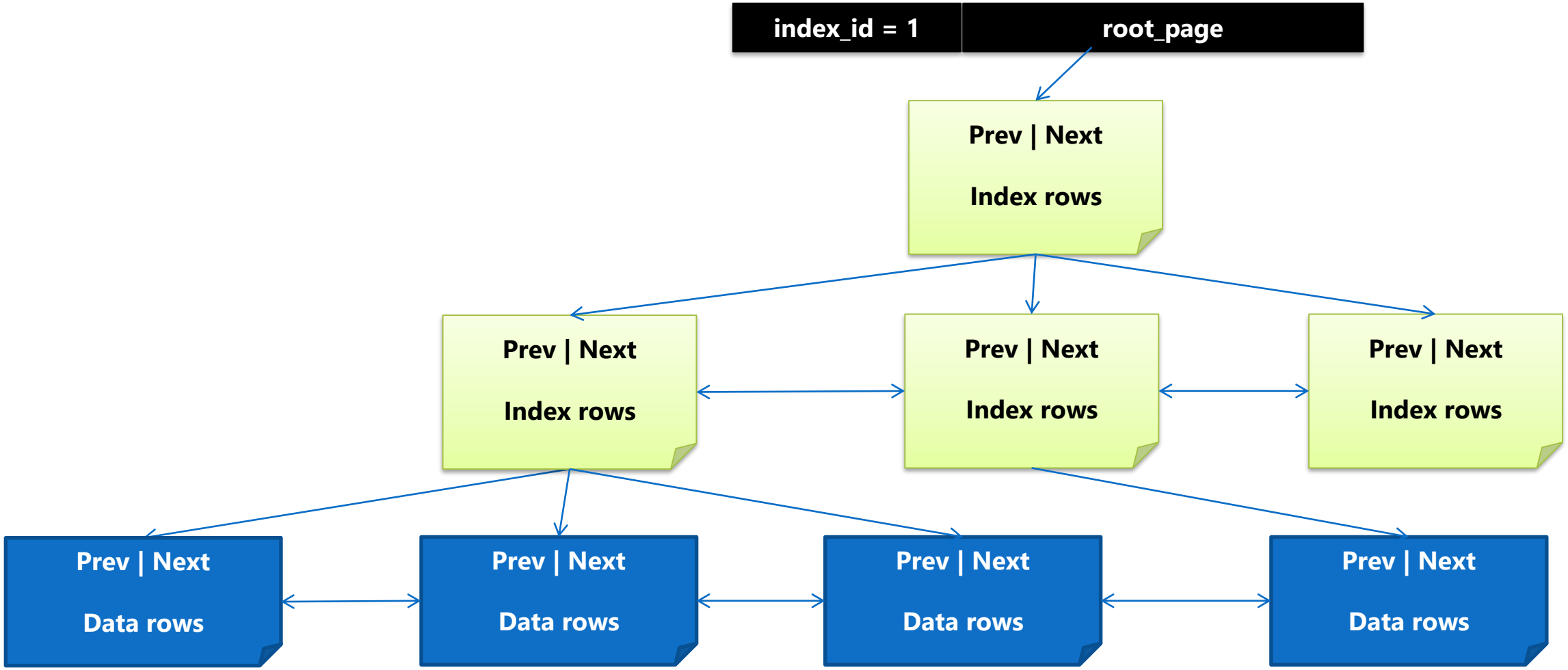
- Ristretto
- Univoco
- Statico
  - Le modifiche provocano overhead aggiuntivo per i Non-Clustered Index
  - Frammentazione dell'indice cluster
- Incrementale

Clustered Index: Come dovrebbe essere

Clustered Index

- Ristretto
- Univoco
- Statico
- Incrementale
  - I dati vengono aggiunti alla fine dell'indice, si riduce il fenomeno noto come Page Split e la frammentazione dell'indice stesso

Clustered index



Clustered index: Guidelines

Un Clustered index è ottimo quando le query selezionano una grande quantità di record adiacenti (range queries)

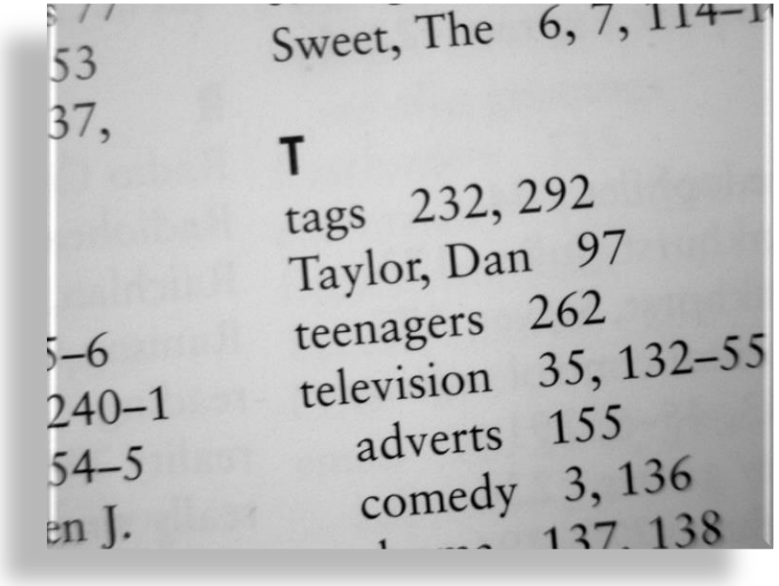
- Tipicamente viene creato sulle colonne più frequentemente utilizzate nelle JOINs e nella clausola WHERE (con “=”, “<”, “>”, “BETWEEN”)
- Sono da preferire le colonne relative a tipi di dato piccoli e le colonne molto selettive

Non-Clustered Index: Caratteristiche

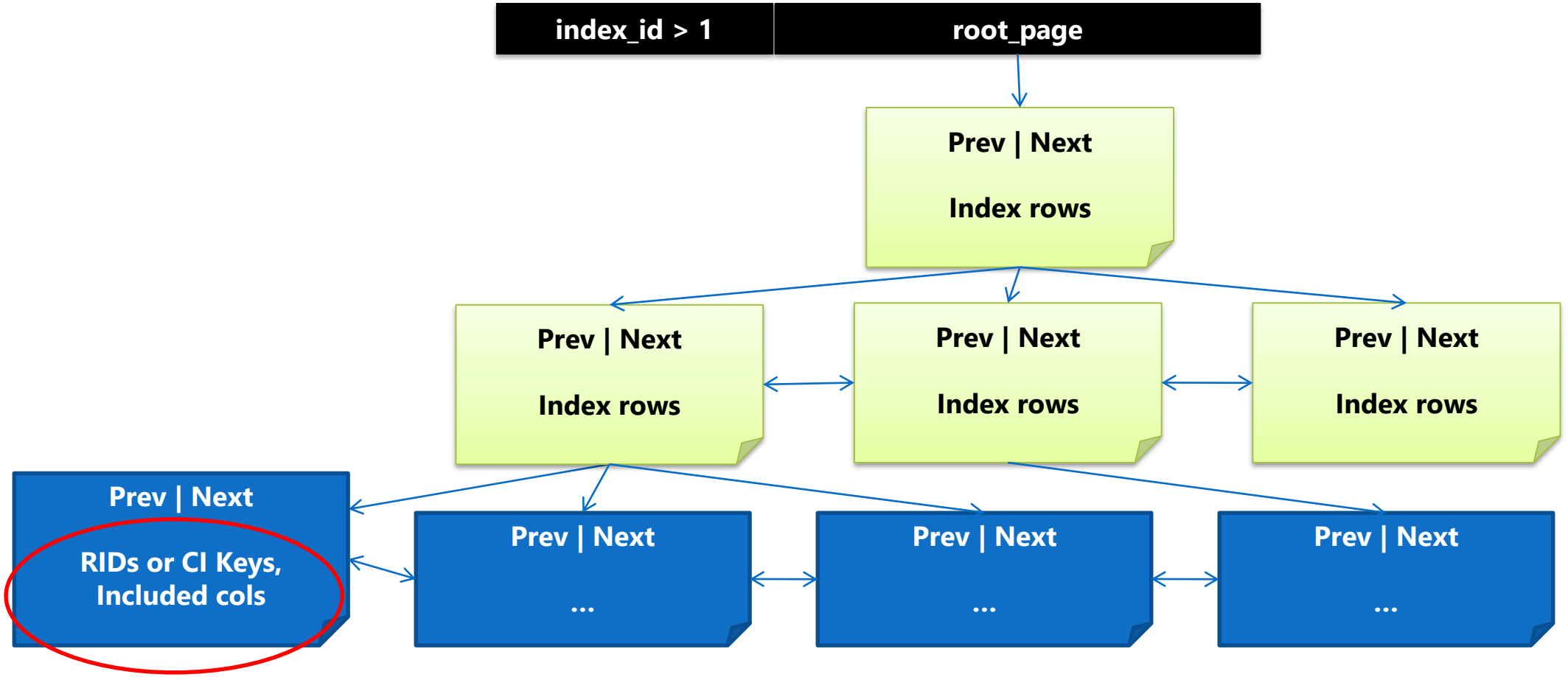
Coprono le query (indici di copertura)

Contengono il collegamento all'indice Clustered

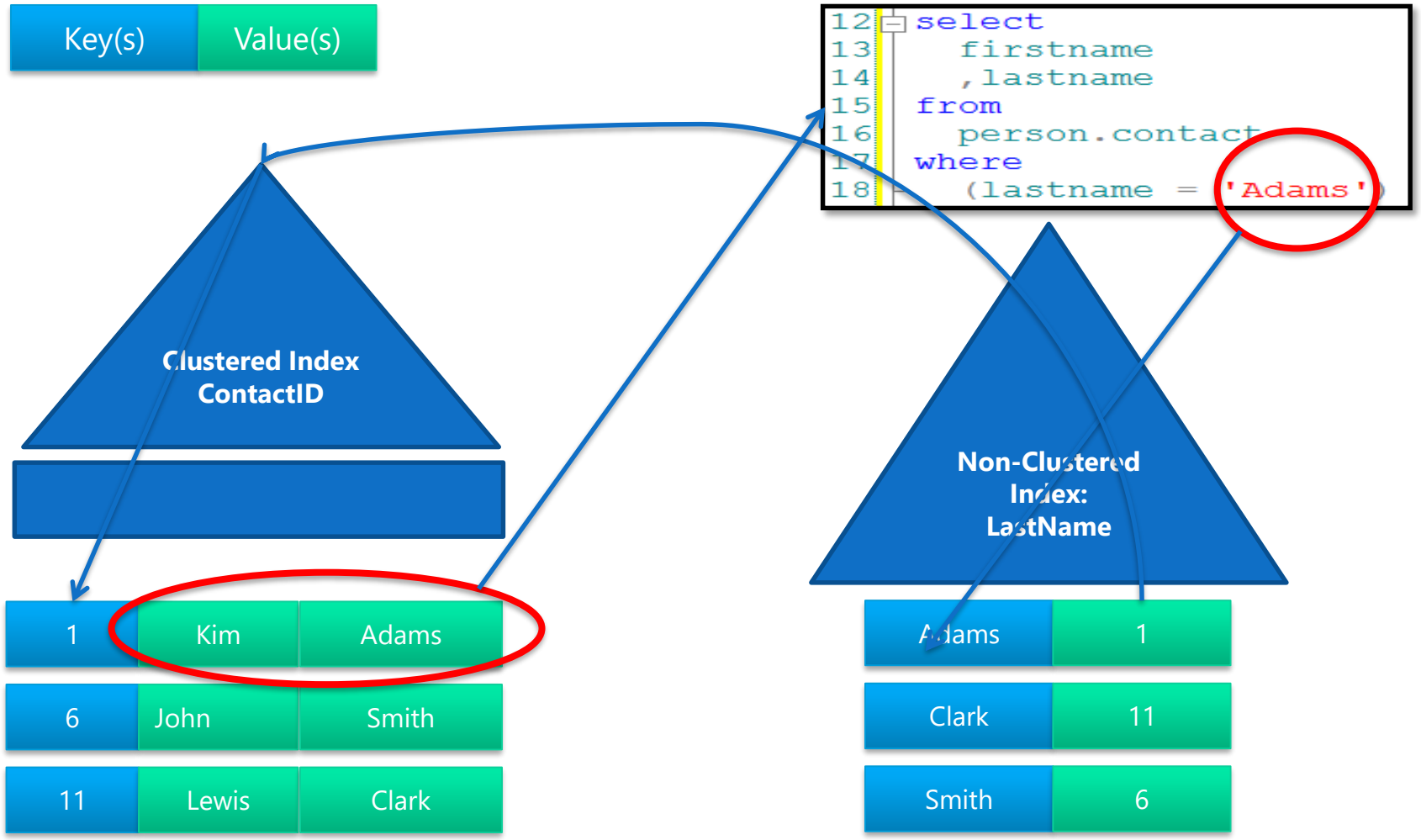
- Lookup – Ricerca chiave



Non-Clustered index



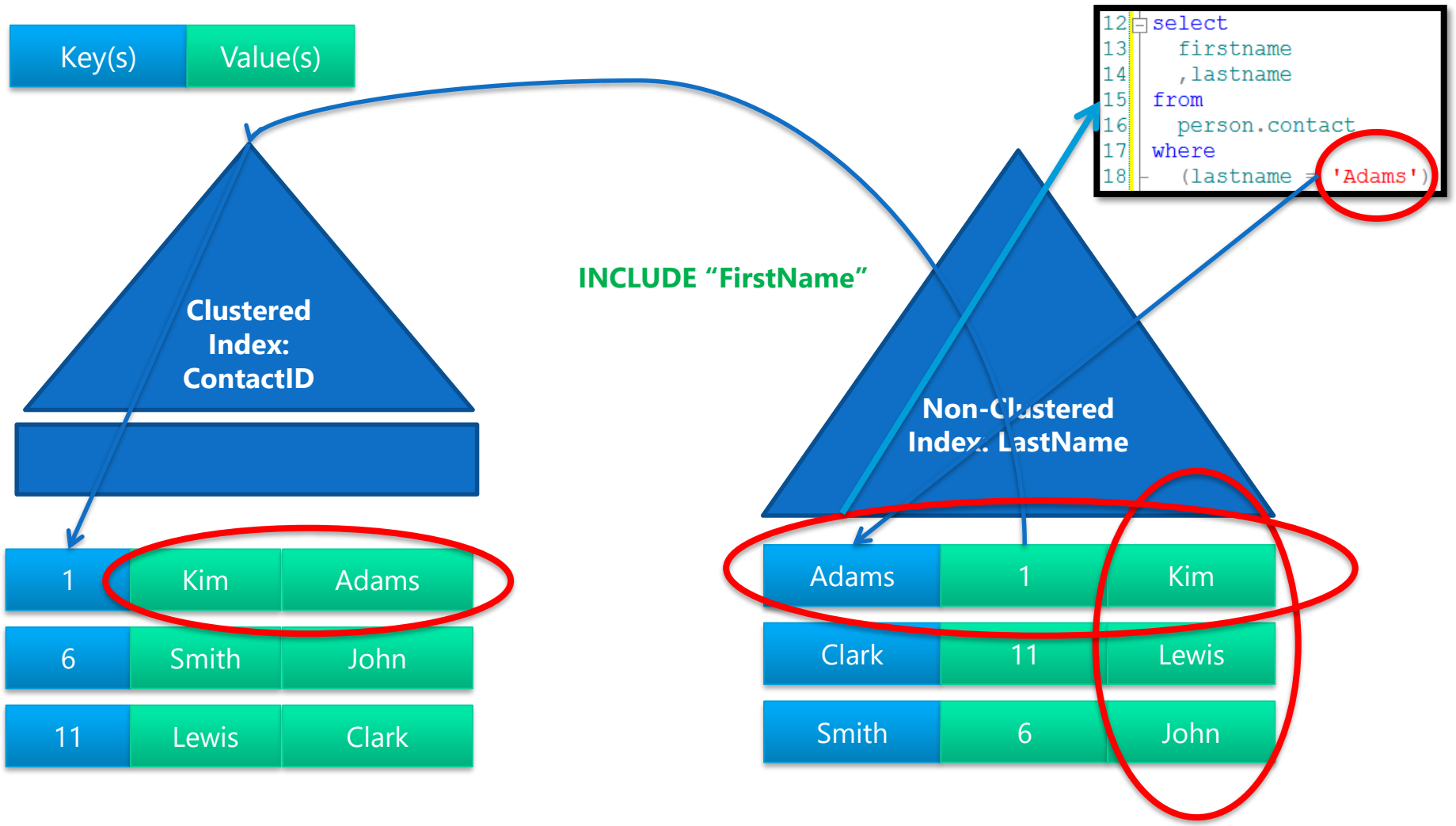
Non-clustering index lookup



Non-Clustered Index con INCLUDE

- La clausola INCLUDE è utilizzata per coprire completamente una query, l'indice è detto "di copertura"
- Duplicazione di dati, ma si evitano le letture sull'indice cluster
- Non influisce sui limiti, 16 colonne in chiave o 900 bytes
- Overhead dovuto alla manutenzione

Non-clustering index lookup

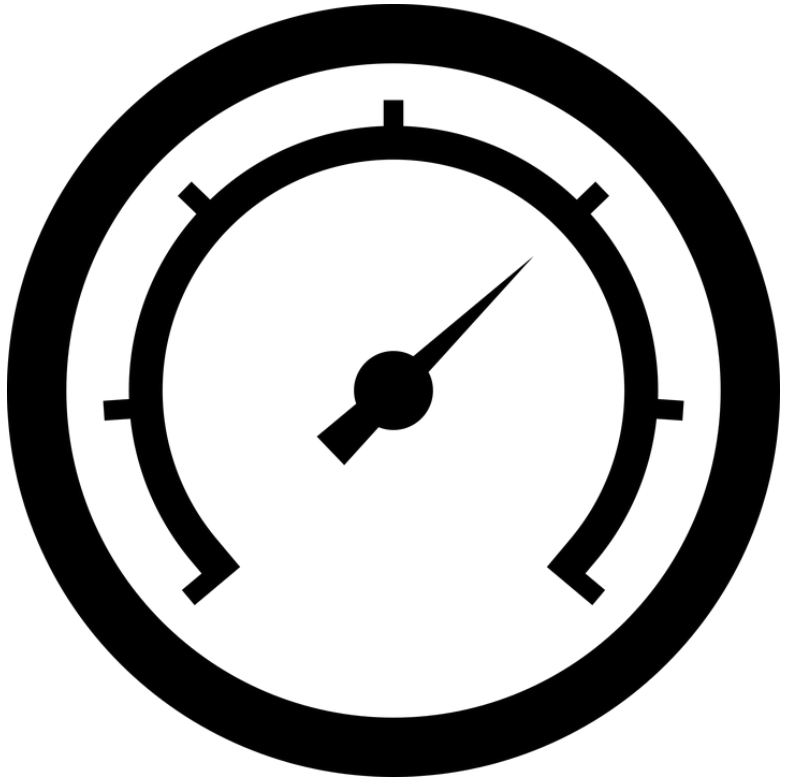


La mia query è lenta: Aiutoooooo!!

- Lento e veloce sono relativi
  - Più lento rispetto a ieri?
  - Più lento della luce? ☺
  - Quanto tempo e I/O impiegati quando la consideravi la query veloce?
- Serve una baseline!

Baseline

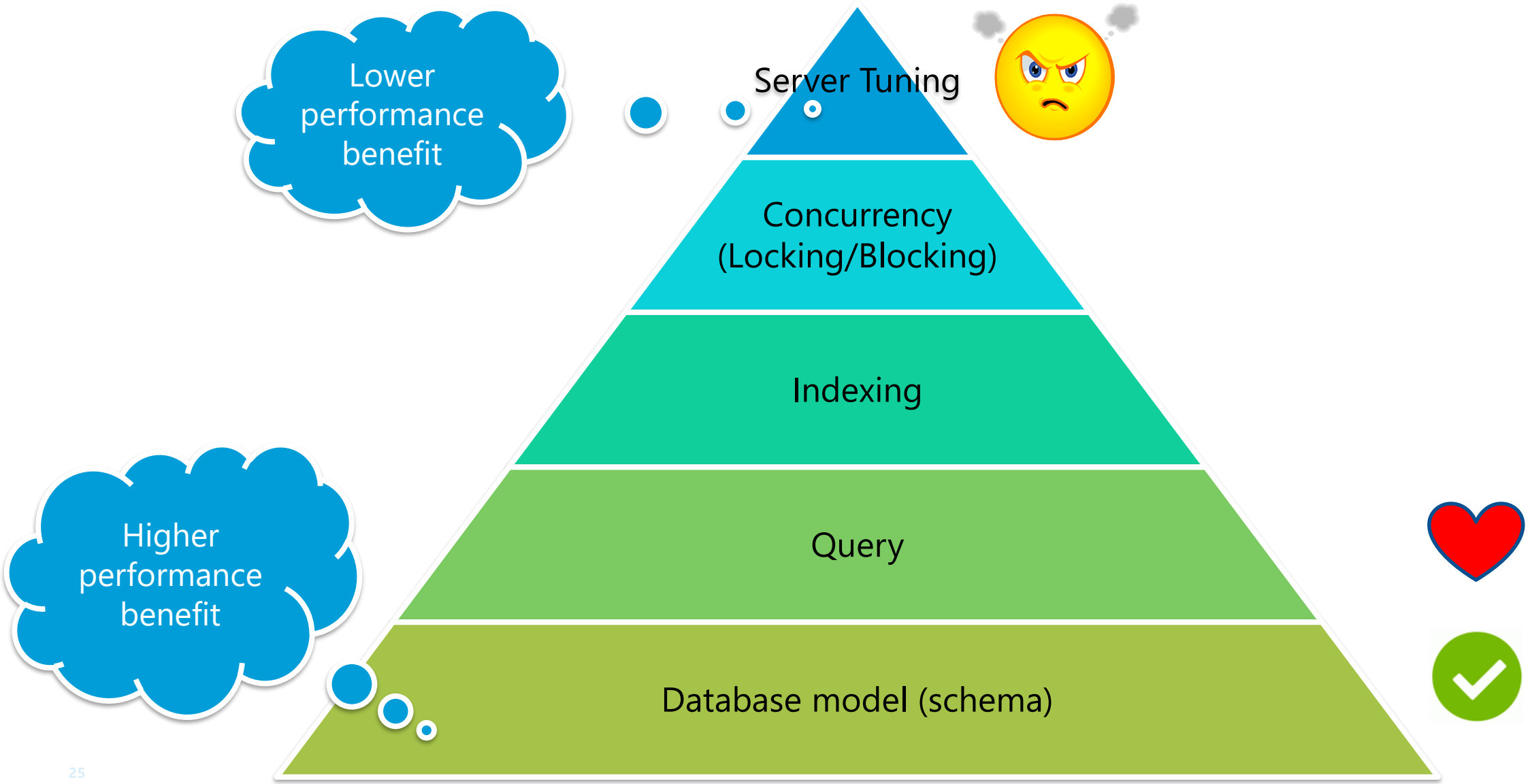
- Descrive le performance delle query in condizioni normali
- Viene utilizzata come termine di paragone
- Per le query
  - Tempo di esecuzione
  - Cicli di CPU
  - Pagine lette/scritte
  - …



Query Baselining

- SET STATISTICS IO ON
- SET STATISTICS TIME ON
- Client Statistics
- SQL Profiler
- DMVs
  - sys.dm_exec_query_stats

La piramide delle performance



Database model (schema)

- Normalizzazione
  - 1NF: Ogni attributo DEVE essere atomico
    - No multi-value!
  - 2NF: Ogni attributo NON chiave dipende in maniera funzionale dalla chiave
  - 3NF: Ogni attributo NON chiave non può descrivere altri attributi non chiave

Database model (schema)

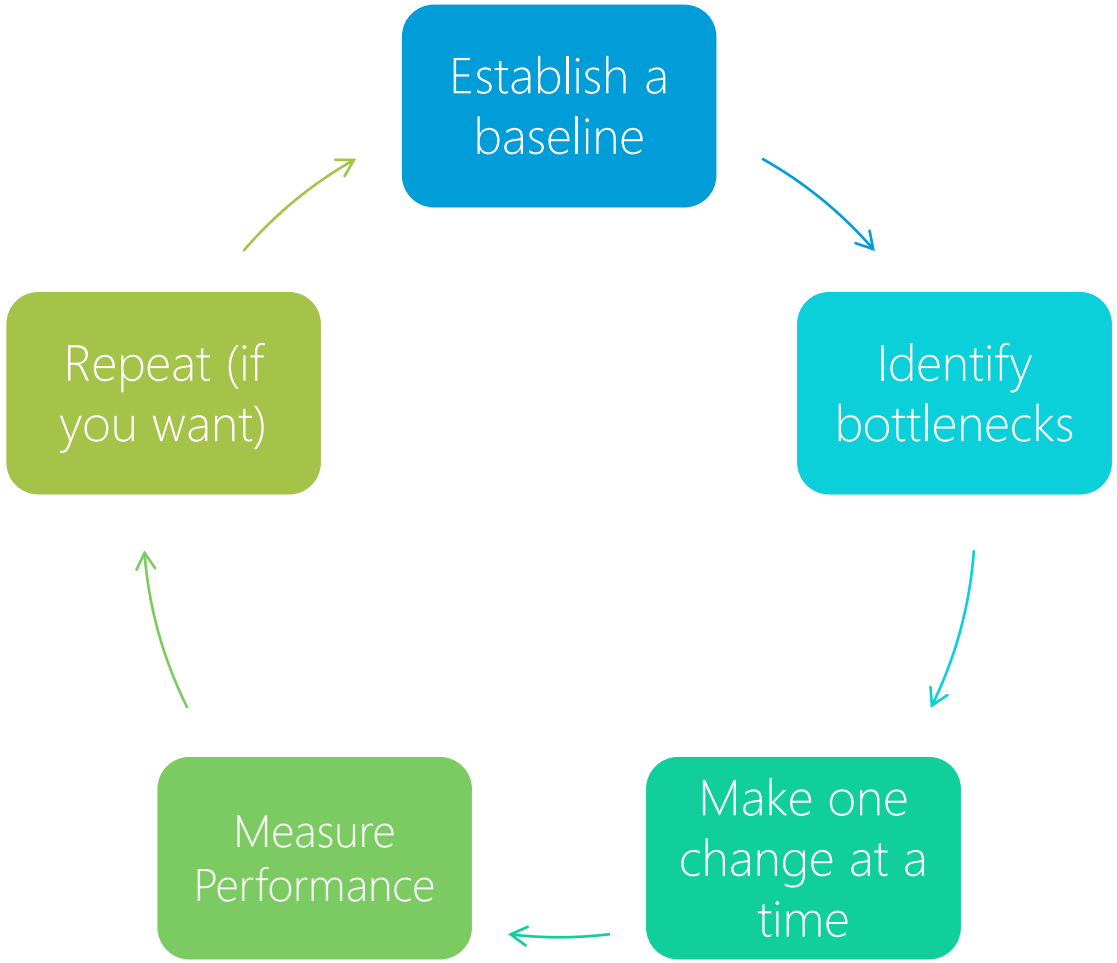
- Segnali preoccupanti
  - Dati ripetuti
  - Dati separati da ","
    - Esempio: COD1, COD2, …, CODN
  - Colonne "elenco" con suffissi numerici
    - Telefono1, Telefono2, Telefono3
  - Colonne di tipo Variant

Database model (schema): Worst Practices

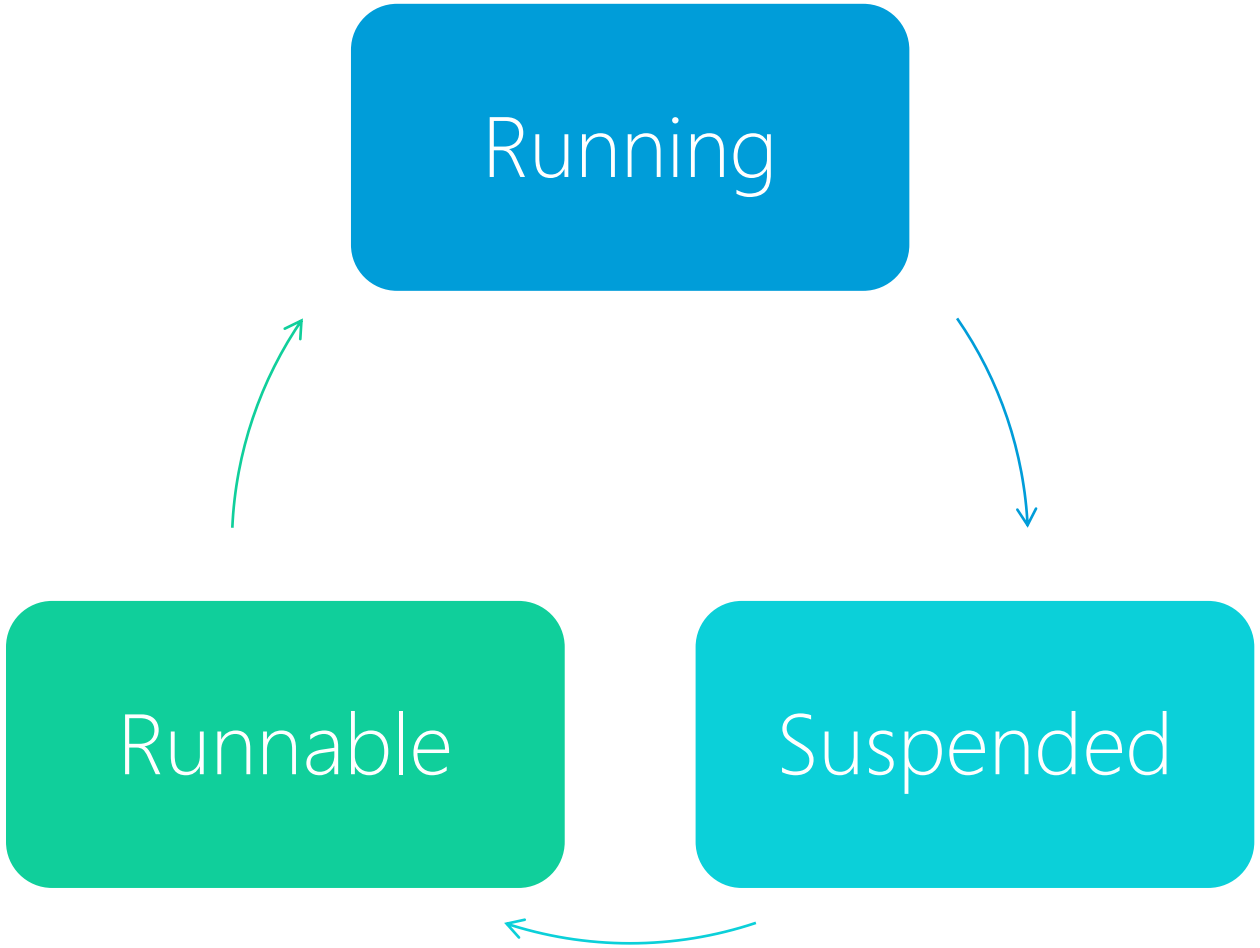
- Nessuna PK o solo chiavi surrogate
- Nessuna FK
- Nessun CHECK CONSTRAINT
  - Presenza di Trigger per verificare l'integrità di dominio (!!)
- Assenza di default
- Utilizzo di NULL dove non necessario

Ottimizzazione del codice

Metodologia di Tuning e Monitoring



Tre possibili stati per una query



Quando una query è “migliore”?

Throughput elevato

Più concorrenza

Meno risorse

Rapida restituzione dei risultati

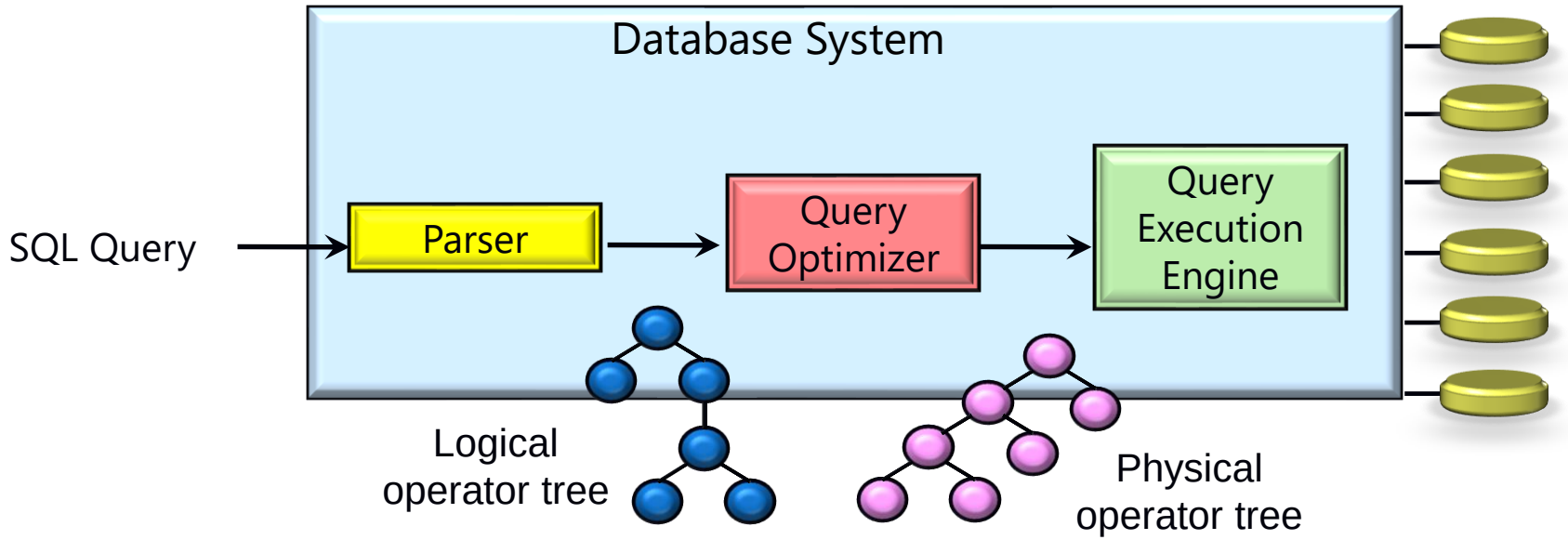
Piani di esecuzione in SQL Server

Execution Plan: Fondamenti

- Risultato del lavoro svolto dal Query Optimizer
  - Contiene le istruzioni per eseguire una query
  - Traccia la strada più efficiente per accedere ai dati
- Perché questa query è così lenta?
  - Gli indici in essere vengono utilizzati?
  - Perché non vengono utilizzati?
- SQL Server riutilizza i piani di esecuzione
  - Plan Cache

Il ruolo del Query Optimizer

Trasforma un query in un piano di esecuzione



Logical operators: **COSA** devo fare
(Union, Selection, Join, Grouping)

Physical operators: **COME** devo farlo
(Nested Loop Join, Sort-Merge Join, Hash Join, Index Join)

Execution Plan: Fondamenti

- Due tipi
  - Estimated Execution Plan
  - Actual Execution Plan
- Tre formati
  - Graphical Plans
    - Semplici da leggere, i dati di dettaglio sono mascherati
  - Text Plans (deprecated)
    - Più complessi da leggere, ma più ricchi di informazioni
  - XML Plans
- Dove li troviamo?
  - SSMS, SQL Profiler, DMVs, Query Store

Il primo operatore nel piano di esecuzione

- SELECT/INSERT/UPDATE/DELETE
- Informazioni non visibili nel tool-tip
  - Parallelism
  - Optimization level
  - Reason for early termination…
  - ANSI options
  - Parameters
  - Warnings

Demo

Il primo operatore

Warnings

- No statistics (ColumnsWithNoStatistics)
- No join predicate (NoJoinPredicate)
- Implicit conversions (PlanAffectingConvert)
- Sort on tempdb (SpillToTempDb)
- No matched indexes (UnmatchedIndexes)
- Wait (WaitWarningType)

- Check the showplanxml.xsd

Demo

Warnings

Extra operators

- Un operatore “extra” è un task che..
  - Non ti aspetti di vedere nel piano di esecuzione
  - Non sai spiegare
- Cosa fare in presenza di un operatore “extra” ?
  - Capire cosa fa l’operatore
  - Capire perché e necessario
    - Se non è necessario, fix it! ☺

Demo

Extra operators

Index Scan vs Index Seek

- Gli Index Seeks non sono necessariamente migliori degli Index Scan
  - It depends!! ☺
- Attenzione a..
  - Statistiche di I/O (SET STATISTICS IO ON)
  - Numero stimato di esecuzioni
  - Numero effettivo di esecuzioni

Demo

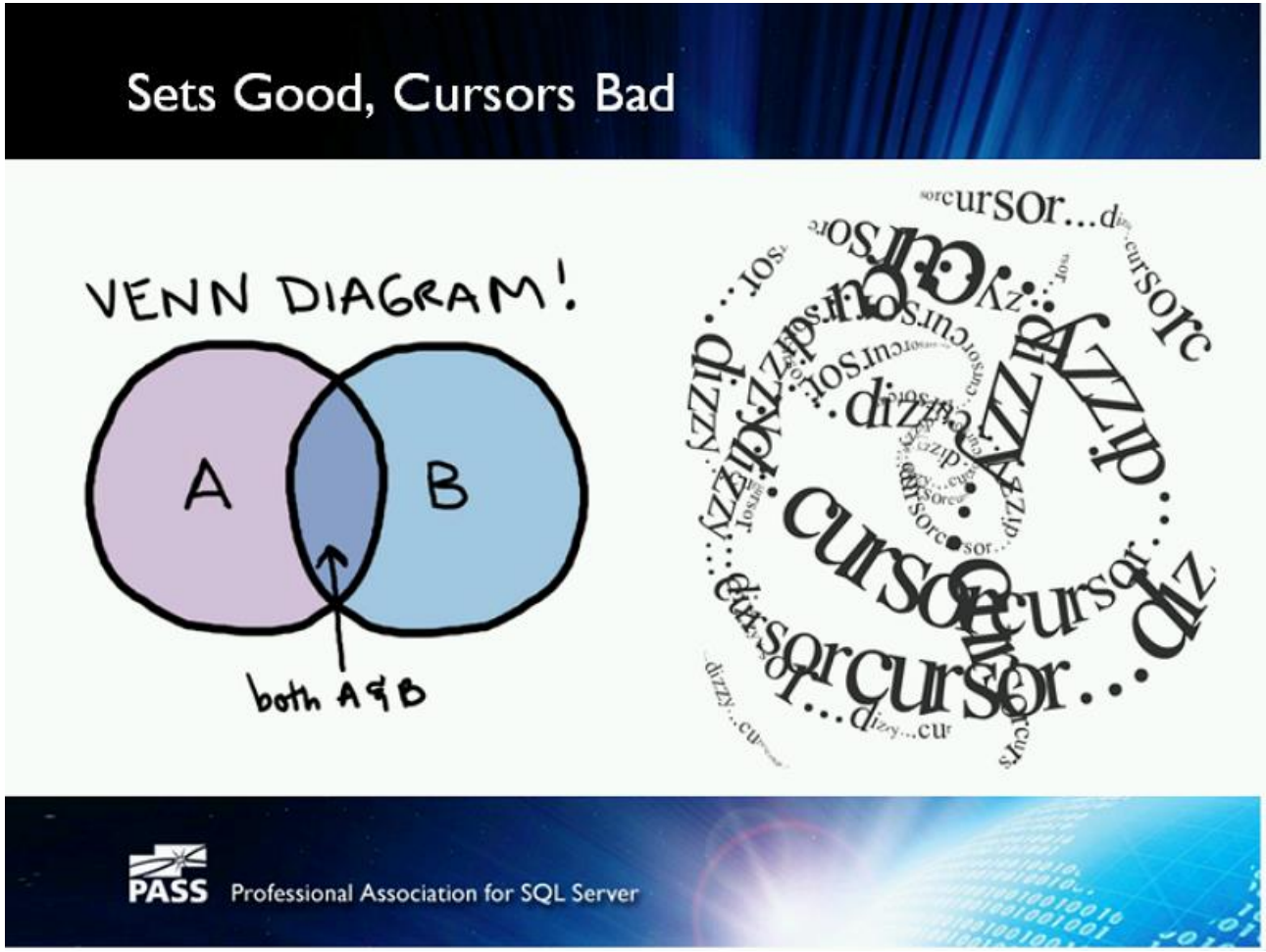
Index Scan vs Index Seek

Il linguaggio T-SQL

Il linguaggio T-SQL

- Deriva dal linguaggio SQL ANSI
- Confrontato con gli altri linguaggi
  - Non è difficile da imparare
  - Può essere molto tollerante
- Chi sviluppa, lo utilizza ogni giorno per
  - SELECT, INSERT, UPDATE, DELETE, CREATE, ecc..
- Solo chi investe più tempo scopre la sua natura dichiarativa!

Come evolve il linguaggio T-SQL?



Come evolve il linguaggio T-SQL?

- Più statement set-based
  - Permetteranno di scrivere query più semplici ed efficienti
- Meno statement “cursor-like”
  - Impongono all’Engine un approccio “row-by-row”
- L’importanza di avere un linguaggio dichiarativo
  - Lo sviluppatore dichiara cosa vuole ottenere
  - Non si preoccupa di come l’engine implementerà la richiesta!

Set-Based o Iterative Code?

- Sono due approcci per sviluppare soluzioni
  - Conosciuti entrambi da chi sviluppa
  - Il più utilizzato, però, è l'approccio procedurale ☹
- Perché l'approccio set-based è consigliato?
- Perché molti sviluppatori utilizzando l'approccio procedurale?
- Quali sono gli ostacoli che impediscono l'utilizzo dell'approccio set-based?

Set-Based o Iterative Code?

- L'approccio set-based non è intuitivo
- Per realizzare soluzioni set-based è necessario conoscere i fondamenti del linguaggio SQL
- Le difficoltà che si incontrano nel pensare set-based derivano spesso dal nostro background

Set-Based o Iterative Code?

Quante volte abbiamo lavorato con un file?

```
open file
fetch first
while not EOF
begin
  process
  fetch next
end
close file
```

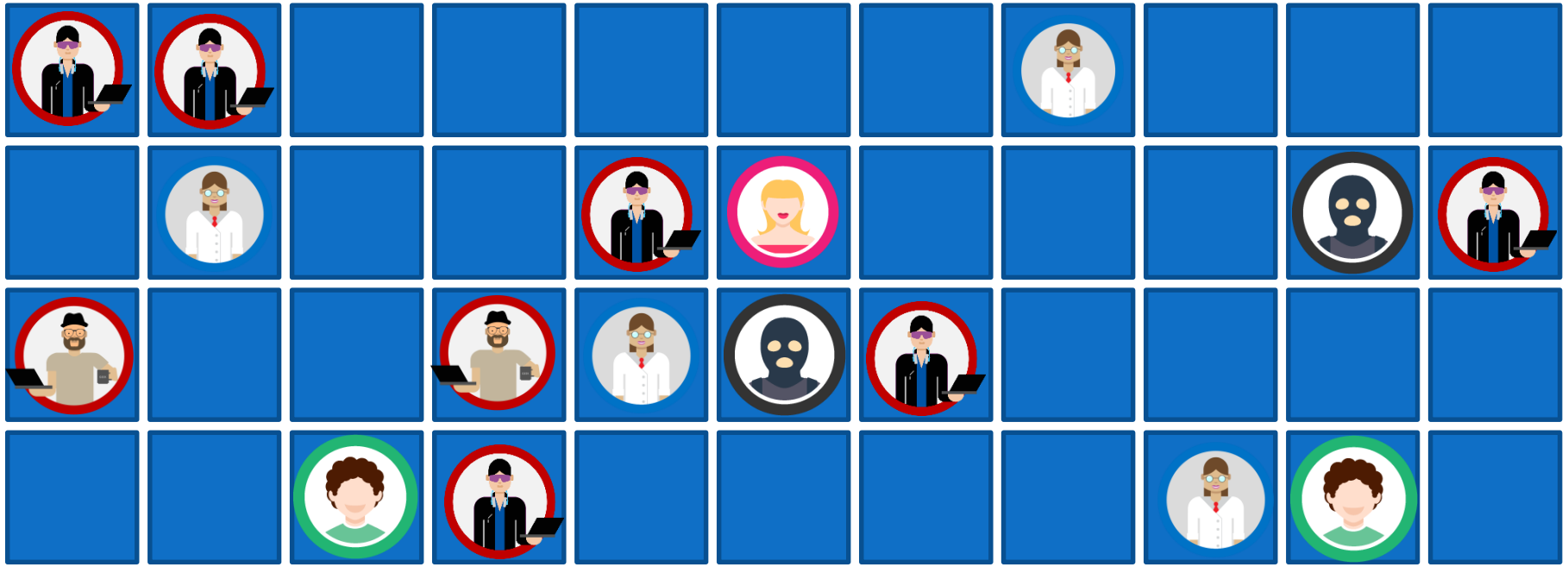
Set-Based o Iterative Code?

- Siamo abituati a pensare che
  - I dati siano memorizzati in un ordine specifico
  - Le operazioni di fetch
    - Garantiscono la restituzione ordinata di porzioni di dati
    - Una porzione alla volta
- La nostra mente è programmata per pensare ai dati in questi termini
  - In modo ordinato
  - Manipolati a porzioni, una porzione alla volta

Vi chiedono di risolvere un problema

Dato un gruppo di persone il cui numero è noto, trovare in quale fila di un cinema (teatro, stadio), vi è sufficiente spazio affinché queste persone possano avere tutti i posti a sedere vicini

Problema: Posti Liberi

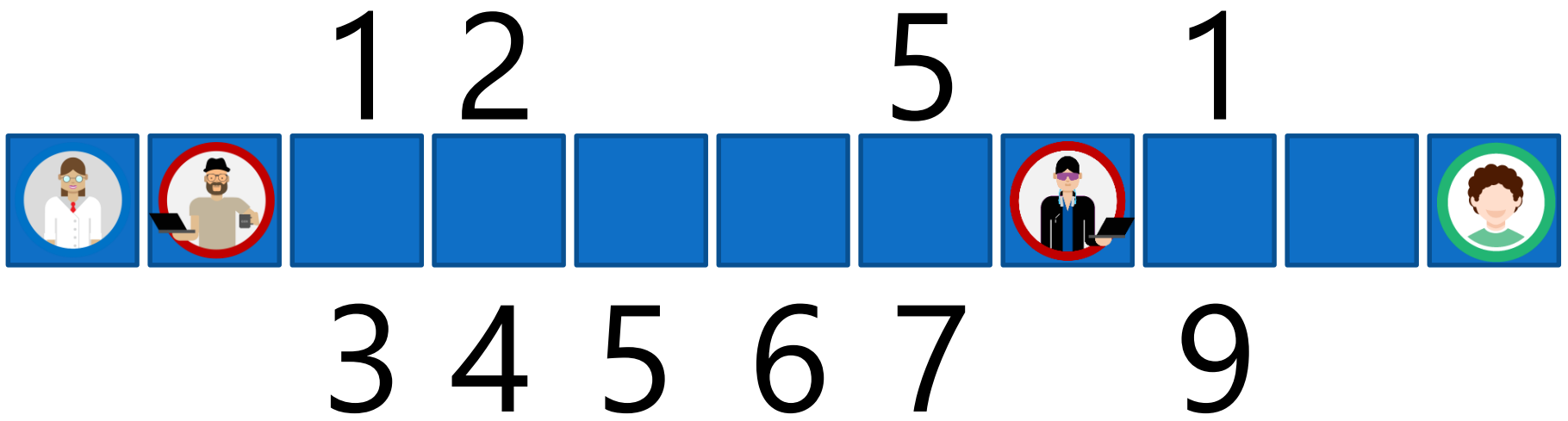


Cerchiamo di risolvere il problema,
In 1 minuto da ora
Pronti.. Via!

Problema Posti Liberi

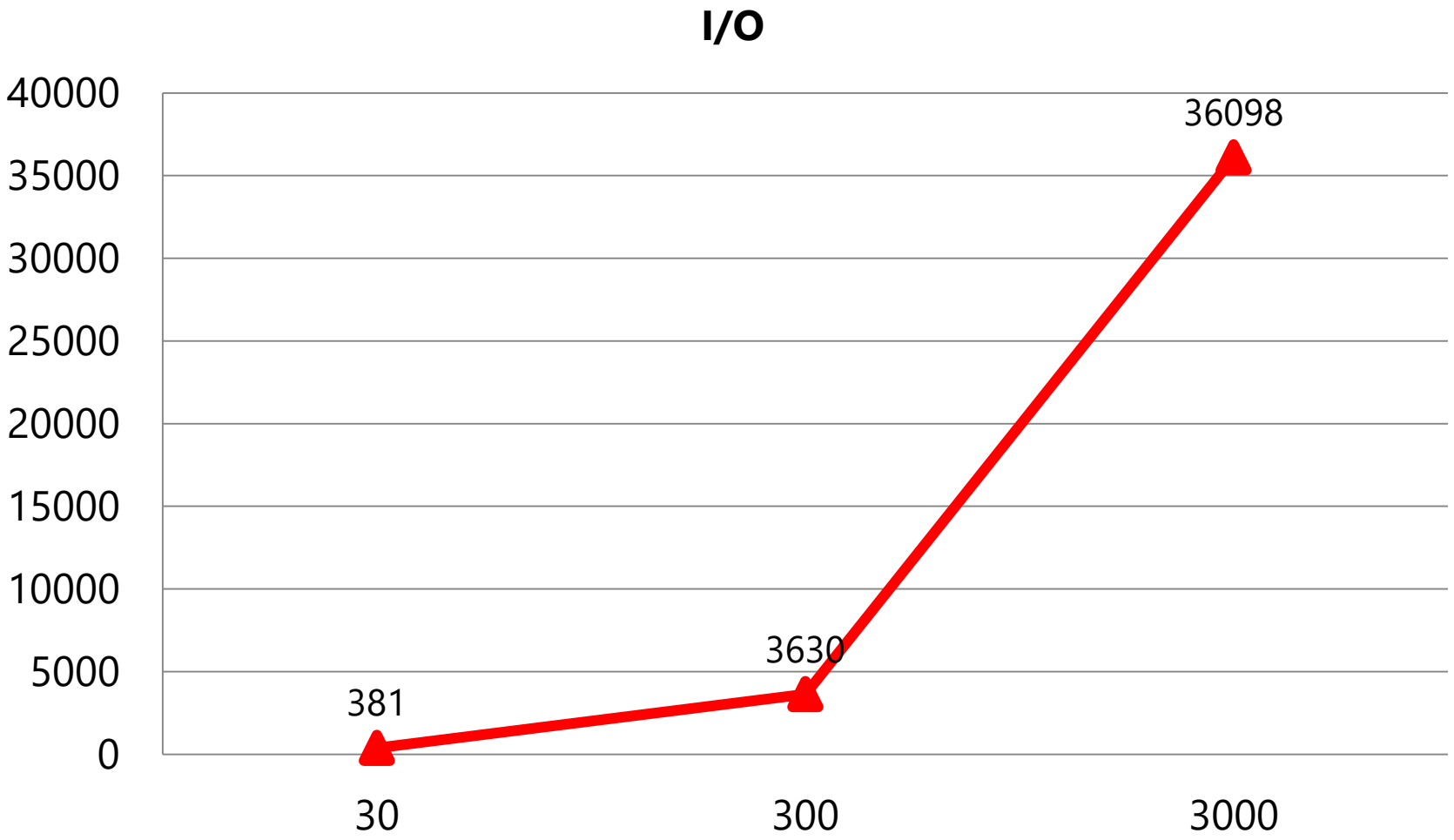
Prima soluzione

Posti Liberi – Prima Soluzione

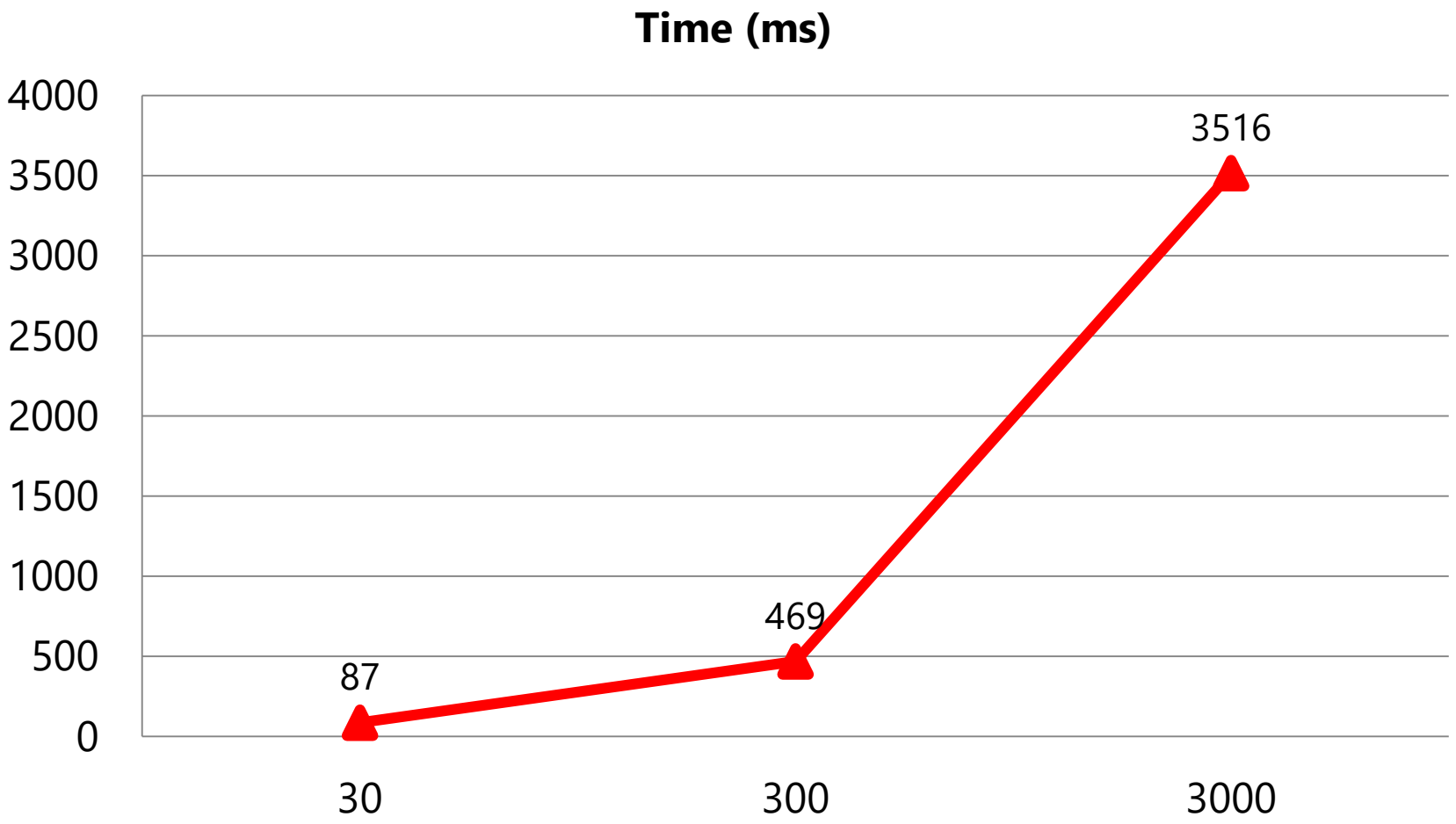


A. Vai al primo posto libero
B. Imposta il contatore posti liberi (NPostiLiberi) a 1
C. Parti dal primo posto libero (n), memorizza questo numero come primo spazio libero (NPrimoLibero)
D. Se il prossimo posto libero è (n+1) allora aumento NPostiLiberi
E. Se il prossimo posto libero non è (n+1) allora conservalo ancora con NPrimoLibero
F. Riparti dal punto A

Prestazioni I/O



Prestazioni Time



La prima soluzione

- Per ogni riga
- E' davvero la migliore? E' la più semplice?
- Non avete la sensazione che ci potrebbe essere qualcosa di meglio? Un'altra soluzione per il problema?
  - Scommetto che avete questa sensazione, ma non sapete come andare verso un'altra soluzione

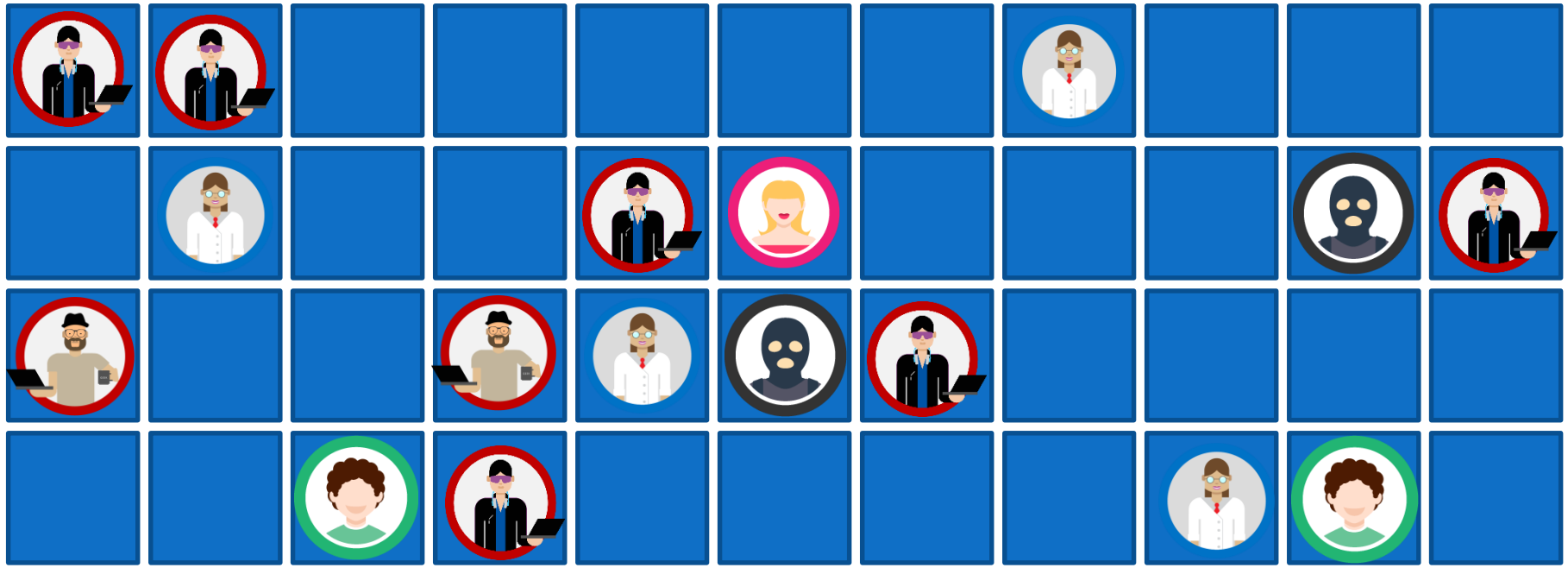
La prima soluzione

- Risultato non così buono ☹
- Non è poi così facile ☹
- Cerchiamo di pensare "al di fuori del codice"
- Il primo passo è: Non di pensare al codice!
  - Il codice è solo uno "strumento"
- Proviamo a trovare la soluzione dal punto di vista puramente logico e poi traduciamo la soluzione in comandi SQL

Pensiamo “Set Based”

- Solitamente iniziamo immediatamente a realizzare la prima soluzione che ci viene in mente..
  - Poi risolviamo tutti i problemi che ne derivano
  - La prima soluzione non sarà in grado di scalare
- Abbandoniamo questo approccio e cerchiamo di trovare il miglior algoritmo (indipendente dal linguaggio) che sia in grado di risolvere il problema

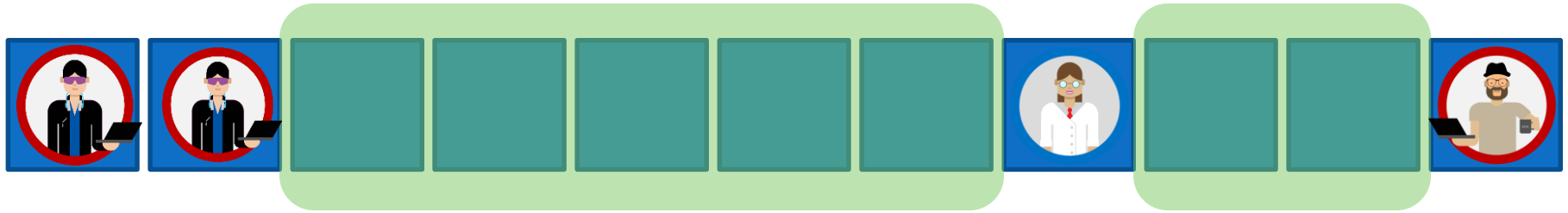
Problema: Posti Liberi



Problema Posti Liberi

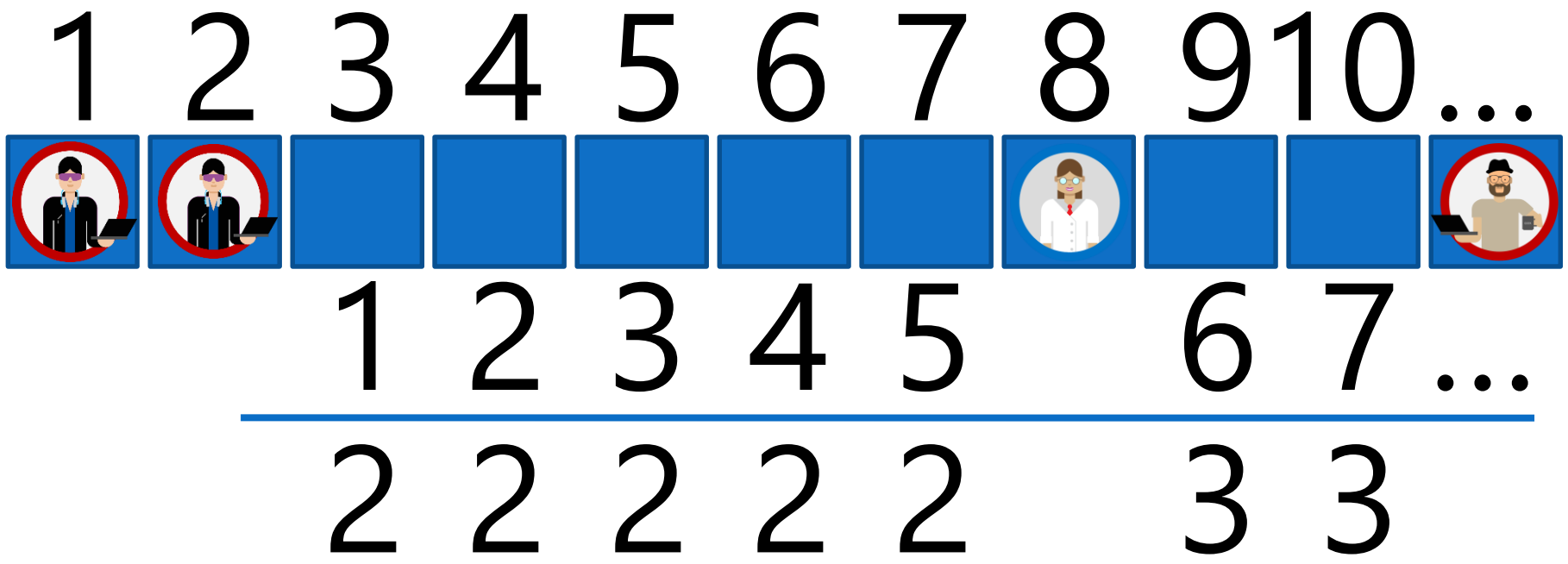
Seconda soluzione

Posti Liberi – Set Based



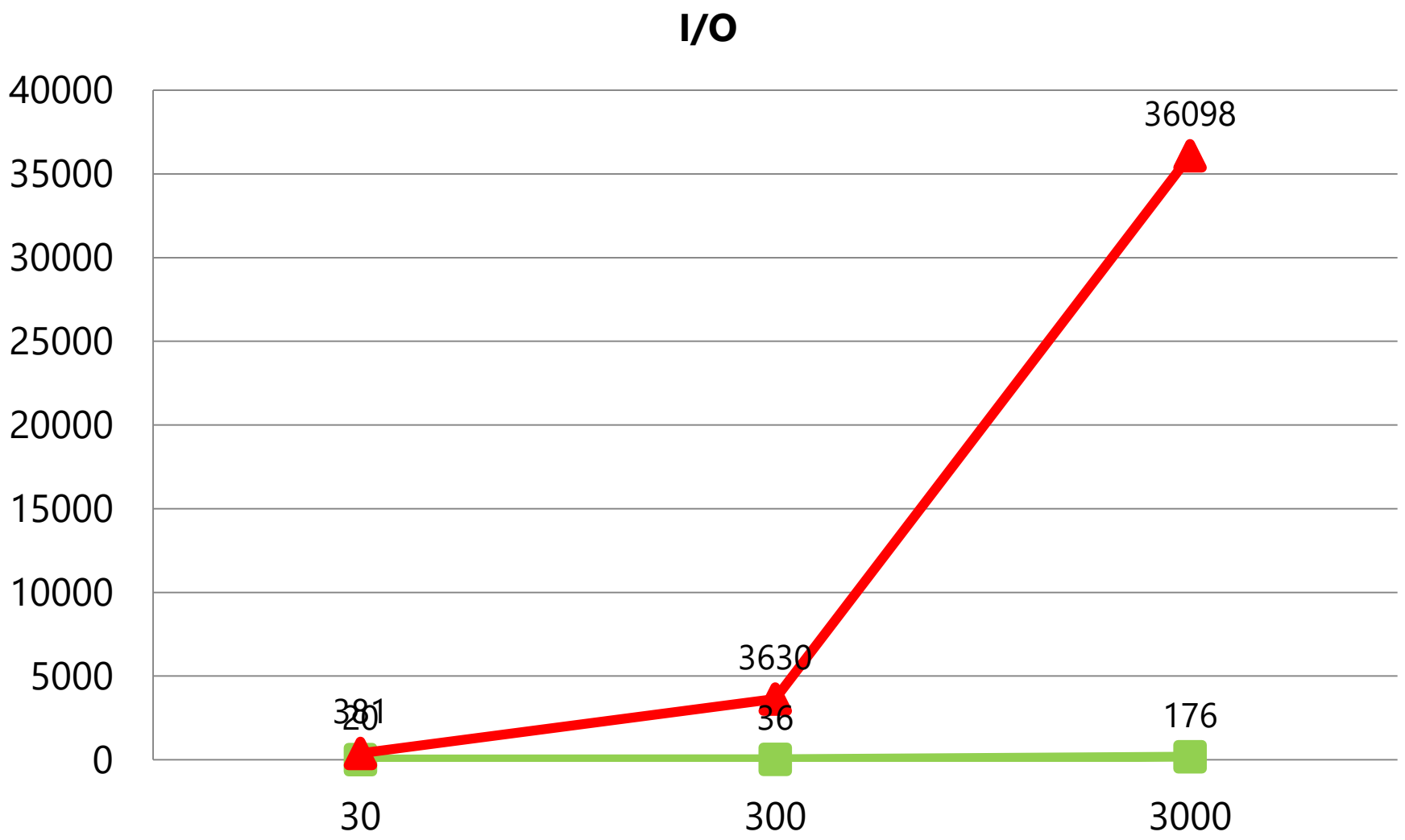
- Se SQL Server avesse gli occhi ☺ sarebbe in grado di osservare che ci sono gruppi di posti liberi
- Si potrebbe semplicemente usare una “GROUP BY”
- Abbiamo bisogno di trasformare questa immagine “solo per chi può vederla” in qualcosa che possa essere gestito da SQL Server

Posti Liberi – Set Based

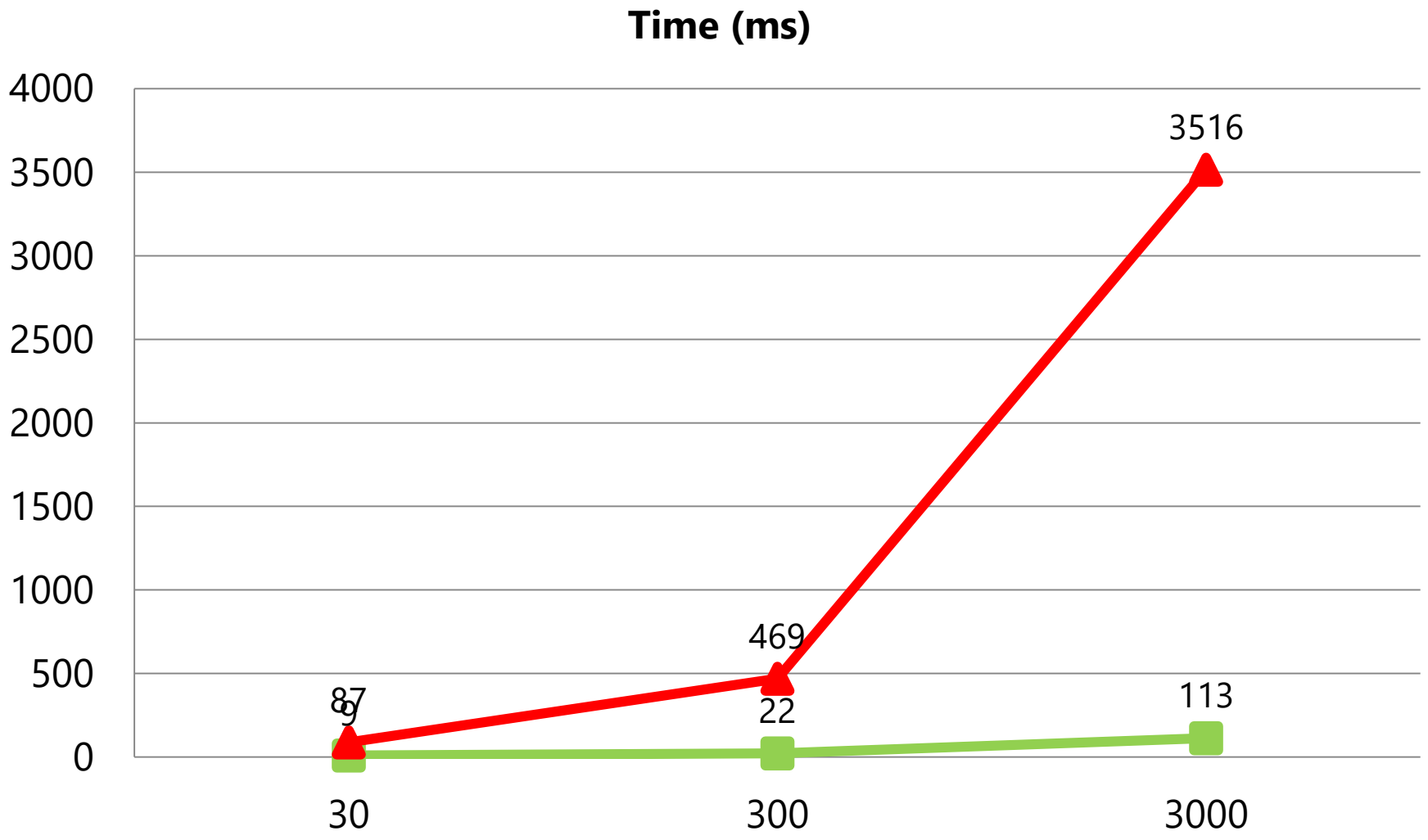


A. Numeriamo tutti i posti (liberi e occupati)
B. Numeriamo tutti i posti liberi
C. Facciamo una semplice sottrazione
D. Ora possiamo utilizzare un GROUP BY

Confronto Prestazioni



Confronto Prestazioni



Common Table Expression

CTE

Cos’è una CTE?

- CTE è l’acronimo di Common Table Expression
  - Tabella derivata, virtuale, originata da una query
  - Può essere citata nella clausola FROM come una qualsiasi altra tabella fisica
  - Visibilità limitata alla query in cui è definita
- Attraverso CTE, SQL Server fornisce la possibilità di scrivere query ricorsive

Demo

Common Table Expression

La clausola OVER

La clausola OVER

- Prima implementazione in SQL Server 2005
- Definisce un sotto insieme di dati (finestra) su cui effettuare una operazione
  - Ranking (ROW_NUMBER, NTILE, …)
  - Aggregazione (MIN, MAX, SUM, AVG, …)
  - Analisi o una funzione di windowing
- Il risultato è incluso in ogni riga del result-set

La clausola OVER: Estensioni

- Da SQL Server 2012, si può manipolare la finestra su cui eseguire le operazioni
- Le opzioni ROWS e RANGE, specificate dopo l'ORDER BY, definiscono in modo preciso l'ampiezza della finestra
  - ROWS (default) riferimento assoluto alla riga
  - RANGE il valore di riferimento per l'ordinamento determina il range

La clausola OVER: Estensioni

- SQL Server 2012 implementa nuove funzioni di Windowing per fare analisi
  - Fondamentali per semplificare alcuni tipi di query
- Funzioni analitiche
  - LEAD, LEG, FIRST_VALUE, LAST_VALUE
- Calcolo dei percentili
- Calcolo della distribuzione cumulativa

Demo

La clausola OVER

L’operatore APPLY

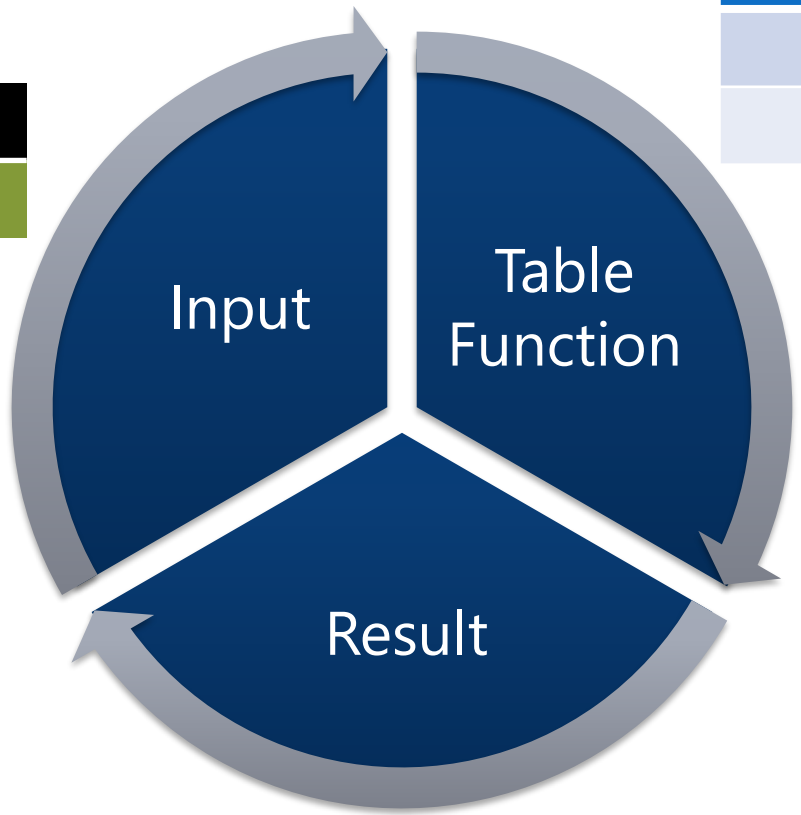
L’operatore APPLY

- Spesso non viene utilizzato al pieno delle sue possibilità
- E’ un operatore tabellare come JOIN, UNION
- E’ in grado di applicare una funzione tabellare ad un set di dati in input

L’operatore APPLY

StockItemName
USB drive

CustomerID	CustomerName
21	Mark
22	Lily



StockItemName	CustomerID	CustomerName
USB drive	21	Mark
USB drive	22	Lily

L’operatore APPLY

- L’uso più comune di APPLY è con Table Valued Functions, chi di voi conosce le DMV

```
SELECT
  QT.text
  ,QS.*
FROM
  sys.dm_exec_query_stats AS QS
CROSS APPLY
  sys.dm_exec_sql_text(QS.sql_handle) AS QT;
```

- Permette di trasformare le MS-T-V-F in funzioni in-line (I-T-V-F)

L’operatore APPLY

- Applicazione di query inline
- L’Engine traduce spesso l’operatore APPLY in un JOIN
  - Quando non è possibile farlo.. In un Nested Loop (JOIN)

```
SELECT
  C.CustomerID
  ,M.MaxOrderDate
  ,M.MaxOrderID
  ,M.PickingCompleted
FROM
  Sales.Customers AS C
CROSS APPLY
  (
    SELECT
      RowNumber = ROW_NUMBER() OVER (PARTITION BY
                                       CustomerID
                                     ORDER BY
                                       OrderDate DESC
                                       ,OrderID DESC)
      ,CustomerID
      ,MaxOrderDate = OrderDate
      ,MaxOrderID = OrderID
      ,PickingCompleted = PickingCompletedWhen
    FROM
      Sales.Orders
  ) AS M
WHERE
  (RowNumber = 1)
  AND (C.CustomerID = M.CustomerID)
ORDER BY
  C.CustomerID;
```

Demo

L’operatore APPLY

SARGable Predicate

SARGable predicate

- Sargable è un termine strano, è una abbreviazione di SARG che sta per **S**earch **ARG**ument
- Sargable significa che il predicato può essere valutato/eseguito utilizzando un Index Seek
- Predicati
  - <expression><operator><expression>
  - <column><operator><expression>

SARGable predicate: Esempi

```
-- SARGable
WHERE StockItemID = 104
-- NON SARGable
WHERE StockItemID + 1 = 104

-- SARGable
WHERE StockItemName = 'DBA joke mug – it'
-- NON SARGable
WHERE UPPER(StockItemName) = UPPER('DBA joke mug - it')

-- ???
WHERE StockItemName LIKE 'DBA%'
```

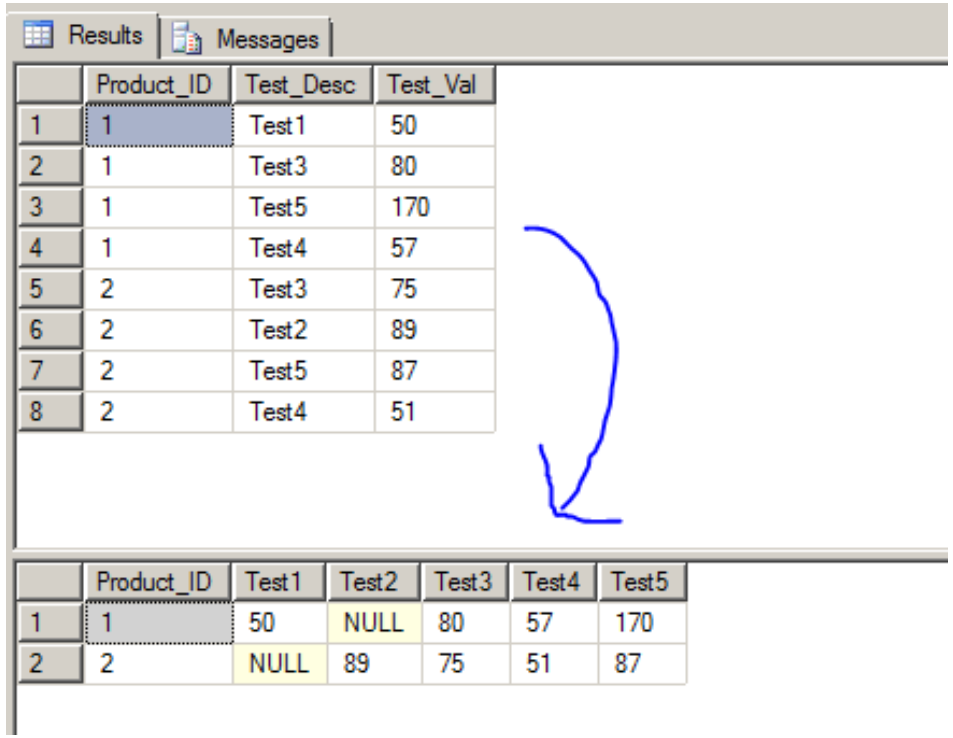
Demo

SARGable Predicate

PIVOT, UNPIVOT

Pivoting

- Il Pivoting è una tecnica che consente di ruotare righe in colonne
- PIVOT (T-SQL)



	Product_ID	Test_Desc	Test_Val
1	1	Test1	50
2	1	Test3	80
3	1	Test5	170
4	1	Test4	57
5	2	Test3	75
6	2	Test2	89
7	2	Test5	87
8	2	Test4	51

	Product_ID	Test1	Test2	Test3	Test4	Test5
1	1	50	NULL	80	57	170
2	2	NULL	89	75	51	87

Unpivoting

- Tecnica inversa al Pivoting ☺
  - Non è esattamente vero, ma pensiamola così per semplicità
- Consente di ruotare colonne in righe
- UNPIVOT (T-SQL)

Demo

PIVOT, UNPIVOT

Summary 1/2

- La mia query è lenta, anzi.. è tutto lento!
  - Serve una Baseline!!!
- La piramide delle performance
- Metodologie di tuning
- Il piano di esecuzione
- Set-Based o Iterative Code?
  - Set-based tutta la vita! ☺

Summary 2/2

- T-SQL
  - Common Table Expression (CTE)
  - OVER e Window Functions
  - APPLY
  - SARGable predicate

Resources

- Libri di Itzik Ben-Gan
  - https://bit.ly/2a0wIPv
  - https://bit.ly/2Jla3hN
  - https://bit.ly/29phxf8
- UGISS – User Group Italiano SQL Server
  - https://www.ugiss.org/
- Canale UGISS su Vimeo
  - https://vimeo.com/ugiss
- SQL Saturday
  - http://sqlsaturday.com/

Explore everything PASS has to offer

24HOURS OF PASS

Free online webinar events

LOCAL GROUPS

Local user groups around the world

SQLSATURDAY PASS

Free 1-day local training events

VIRTUAL GROUPS

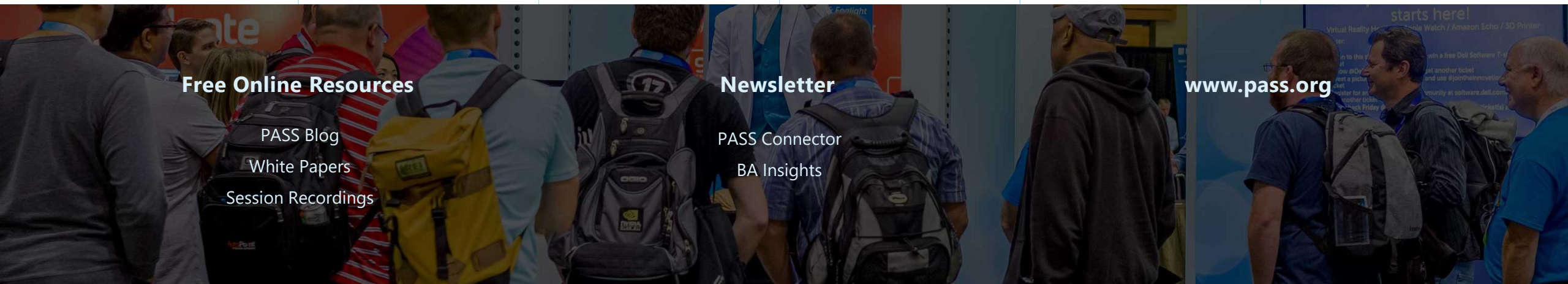
Online special interest user groups

BUSINESS ANALYTICS DAY PASS

Business analytics training

PASS VOLUNTEERS

Get involved



Free Online Resources

PASS Blog

White Papers

Session Recordings

Newsletter

PASS Connector

BA Insights

www.pass.org

Thank You!

/sgovoni

@segovoni