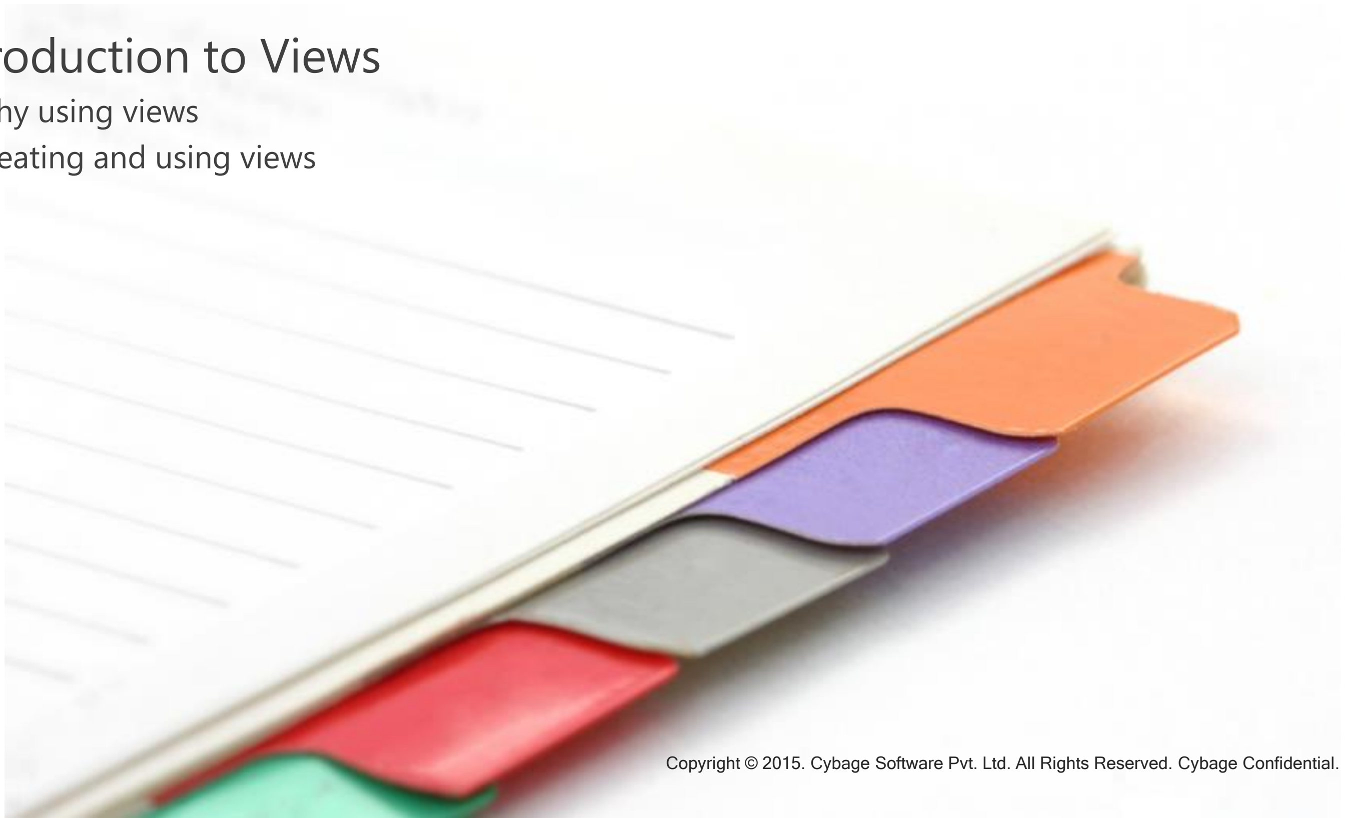
Index

Authored and Presented by : Sushant B.
Extn: 7221, Email: sushantba@cybage.com

Agenda

- Introduction to Indexes
  - Why using Index
  - Types of indexes
  - Creating indexes

- Introduction to Views
  - Why using views
  - Creating and using views

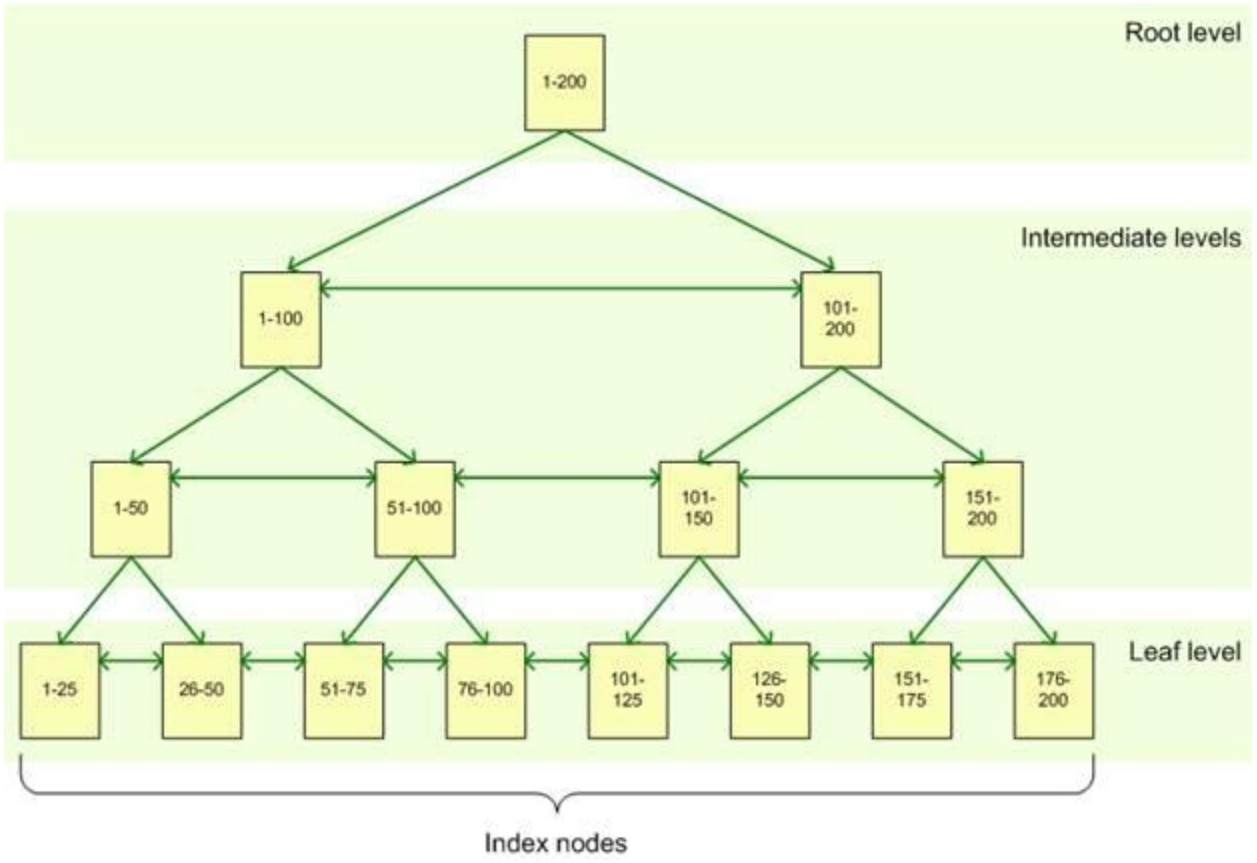


What is an Index

- Helps to search data quickly
- Consists of key and pointers
- Helps improve the performance
- Enforce data integrity.

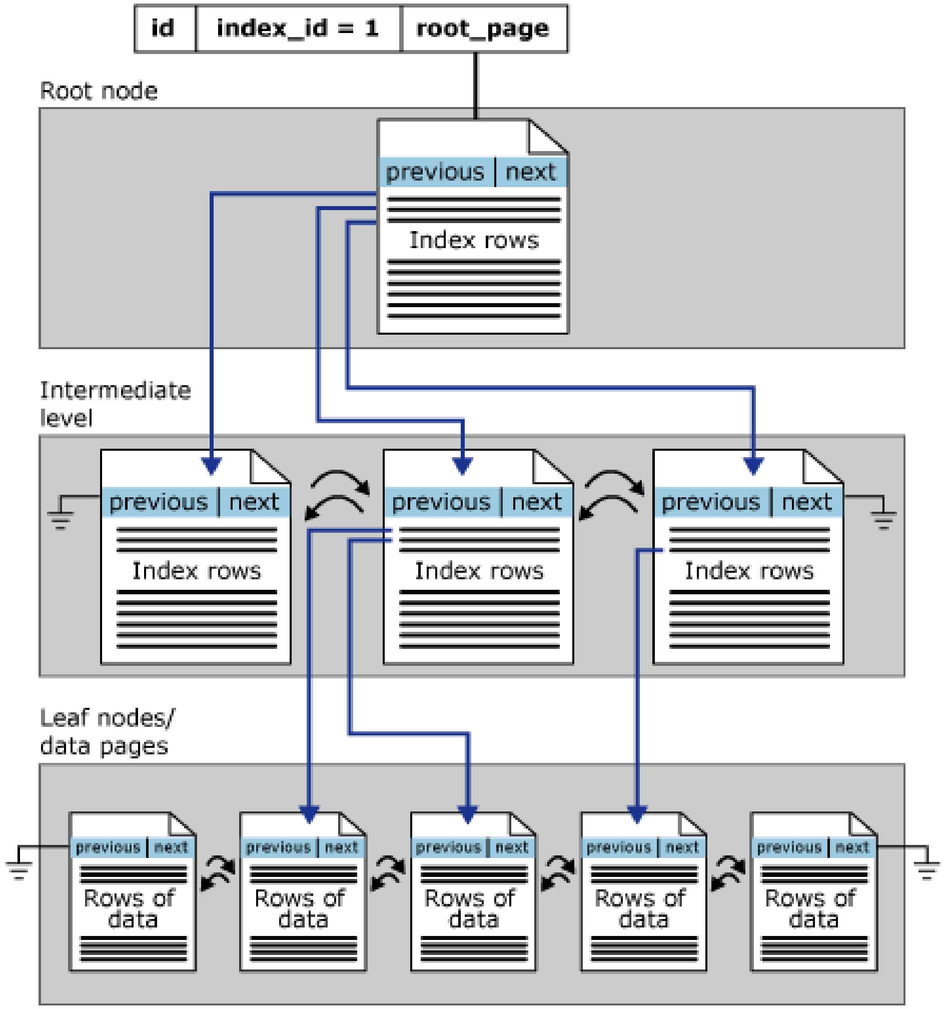
Index B-Tree

- Index pages are organized in B-Tree



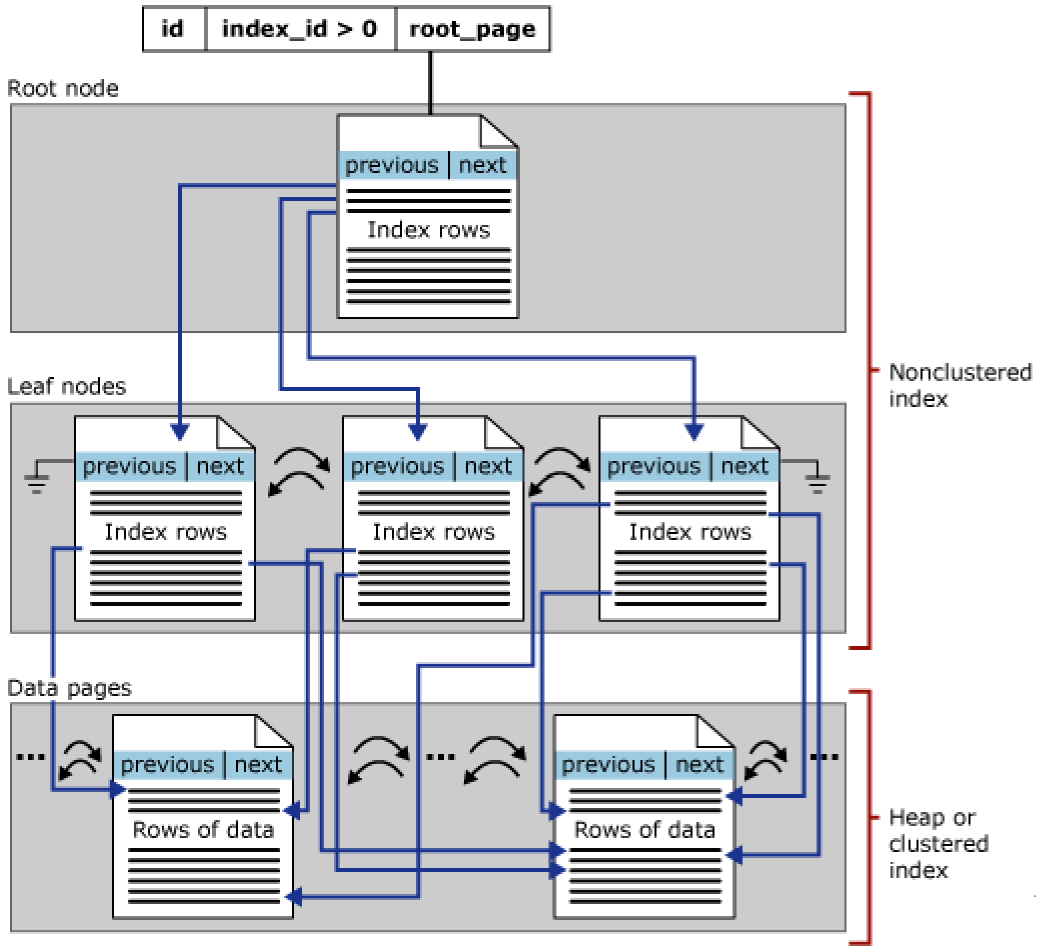
Clustered Index

- Stores actual data rows
- Keeps data in sorted order
- Only one per table



Non-Clustered Index

- Contains Row Locators/Row ID/RID
- A Row Locator either points to
  - A clustered table
  - Or to heap



Other Indexes

- Composite Index
  - Can include more than one column
  - Created when we create composite primary keys
- Unique Index
  - Maintains unique values only
  - Created when we create unique constraint
- Both can be either clustered or non clustered.

When Clustered Index

- Suitable column for clustered index is
  - That provides high degree of uniqueness
  - That can be used in range queries
  - That is defined as IDENTITY
  - That is used to sort data in the result set.

When Non-Clustered Index

- Suitable column for non-clustered index is
  - That involves in join queries
  - That is used in grouping data
  - That is used to search data.

Creating Index

```
--Create a unique clustered index
CREATE UNIQUE CLUSTERED INDEX uciProductId
ON Product(ProductId)

--Create nonclustered index
CREATE NONCLUSTERED INDEX nciProductName
ON Product(ProductName)

--View all indexes created on a table
SP_HELPINDEX Product
```

Summary

- Clustered index
- Non-clustered index

Views

What is View

- A virtual table and does not store data
- Retrieves data dynamically from base table
- Can include maximum 1024 columns

Why Views?

- To represent focused, simplified, and customized data
  - Each user view only the data he or she supposed to view
- As a security mechanism
  - By allowing users to access data through the view
  - Without granting the users permissions to directly access the underlying base tables.
- To provide a backward compatibility
  - No need to update a view for a table whose schema has changed.

Creating and Modifying View

- A simple view

```
CREATE VIEW vwEmployee
AS
SELECT EmployeeId, FirstName FROM Employee
```

- Modifying a view

```
--Modifying the select statement
ALTER VIEW vwEmployee
AS
SELECT iEmployeeId, cFirstName, cLastName , JobTitle
FROM Employee
```

Renaming and Deleting View

- Renaming a view

```
--Rename View(SP_RENAME Old_View_Name, New_View_Name)
SP_RENAME vwEmployee, vwEmp
```

- Delete a view

```
DROP VIEW vwEmployee
```

Modifying Data Using Views

- Following operations are possible through views :
- Inserting data
- Updating data
- Deleting data

SCHEMABINDING Clause

- Binds the view to the base table
- Can not drop base table
- Uses schema.tablename as syntax

```
CREATE VIEW vwEmployee
WITH SCHEMABINDING
AS
SELECT EmpId, Name FROM dbo.Employee
```

WITH CHECK OPTION

- Ensures data remain visible to view
- Checks only when data modified using view
- Does not work if data is directly modified in base table.

```
CREATE VIEW vwTestEmp
AS
SELECT EmployeeId, Name FROM Employee
WHERE EmployeeId = 2
WITH CHECK OPTION
```

Hiding Details of a View

- View text can be encrypted
- Prevents publishing the view in replication

```
CREATE VIEW vwEmployee
WITH ENCRYPTION
AS
SELECT EmpId, Name FROM Employee
```

Nested Views

- You can create a view based on another view
- A nested view access data through another view
- SQL Server supports 32 levels of nesting views.

Bibliography, Important Links

- https://msdn.microsoft.com/en-us/library/ms175049(v=sql.110).aspx

- https://msdn.microsoft.com/en-us/library/ms190174(v=sql.110).aspx

Cybage®
Delivering Value. Scientifically.

Any Questions?

Thank you!