

The New Face of MS SQL Server DataServer Deployment

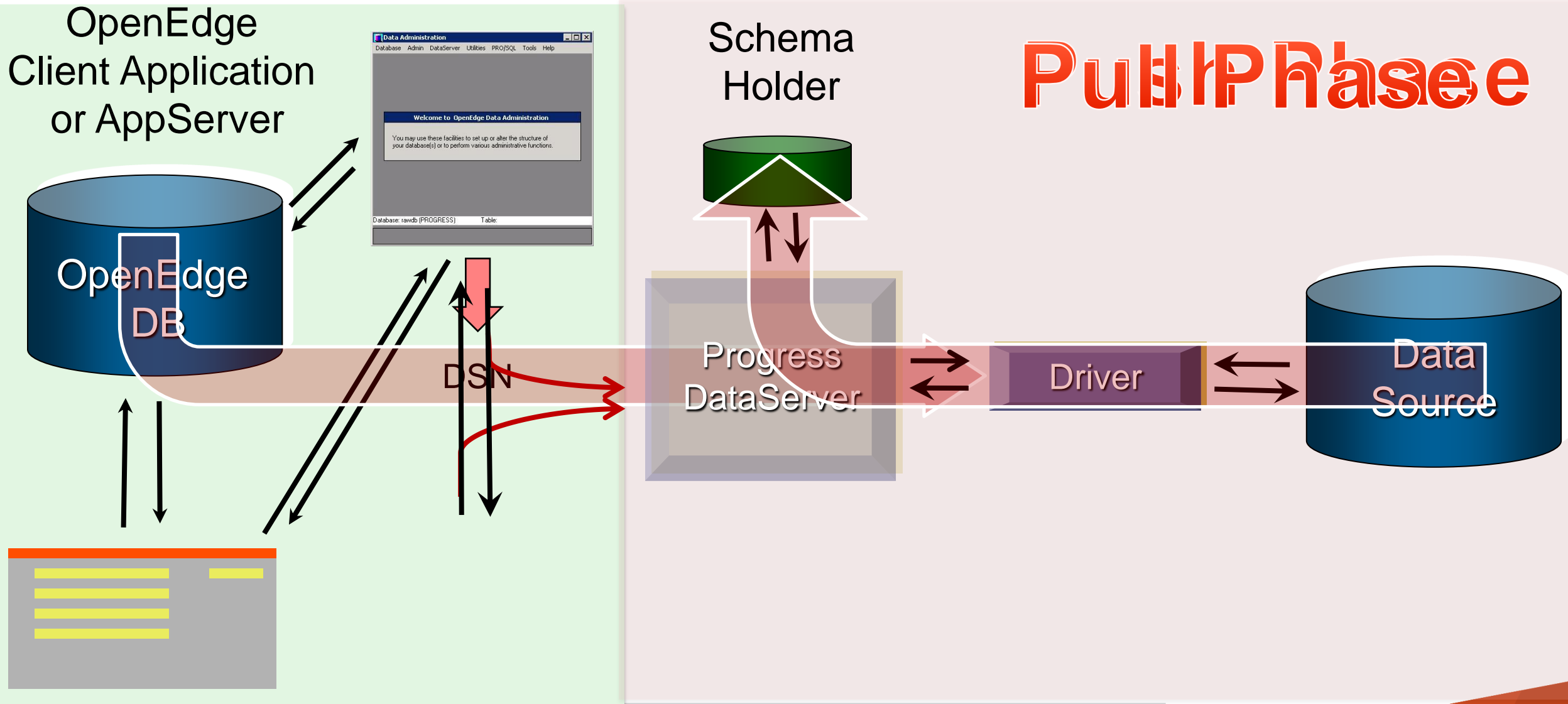
David Moloney and Sachin Garg
Demo delivered by Anil Shukla
OpenEdge DS Product owners
October 2013

PROGRESS
EXCHANGE 2013
DISCOVER. DEVELOP. DELIVER.

Agenda

- Intro. and New Deployment Strategy Goals
- Implementation Strategy and Approach
- Advanced Migration Capabilities
- Multi Pass Strategy – a Recommendation
- Bonus Slides

Conceptual Overview of the Migration Process



New Deployment Goal

- **Main objective:** Improve overall DataServer performance by optimizing data access
 - The Best Solution:
Primary constraint → sets **Clustered index** (*implicitly*) → sets **ROWID index** selection
- Why ROWID? ROWID: Uniquely identifies each row used by the DataServer Application
 - Improving the DataServer's record access via ROWID improves all aspects of performance:
 - Locking
 - Deletions & Updates
 - FINDs, Queries, Browsers
 - Cursor positioning, including INDEXED-REPOSITION
 - RECID/ROWID function
 - LOB operations

New Deployment Goal

What do we want when mapping ROWID?

- **Clustered Index characteristics** – provides the efficiency we want for DataServer ROWID
 - Table records are physically ordered to match the index; Index stores the actual data.
 - Fast index scans because physically adjacent rows have sequences index keys
 - A good clustered index has:
 - Frequently searched columns
 - High degree of uniqueness
 - Often accessed sequentially for range queries
 - Monotonic, incremental, unique (distinct)
 - Narrowly sized, non-composite
 - Relatively static; Infrequently changed
- **Primary Index** – provides uniqueness for row identification we need for DataServer ROWID
 - Uniquely identifies all table records
 - Must contain a unique value for each row of data
 - Cannot contain NULL data (i.e., mandatory, that is, must be defined for each row)

Agenda

- Intro. and New Deployment Strategy Goals
- Implementation Strategy and Approach
- Advanced Migration Capabilities
- Multi Pass Strategy – a Recommendation
- Bonus Slides

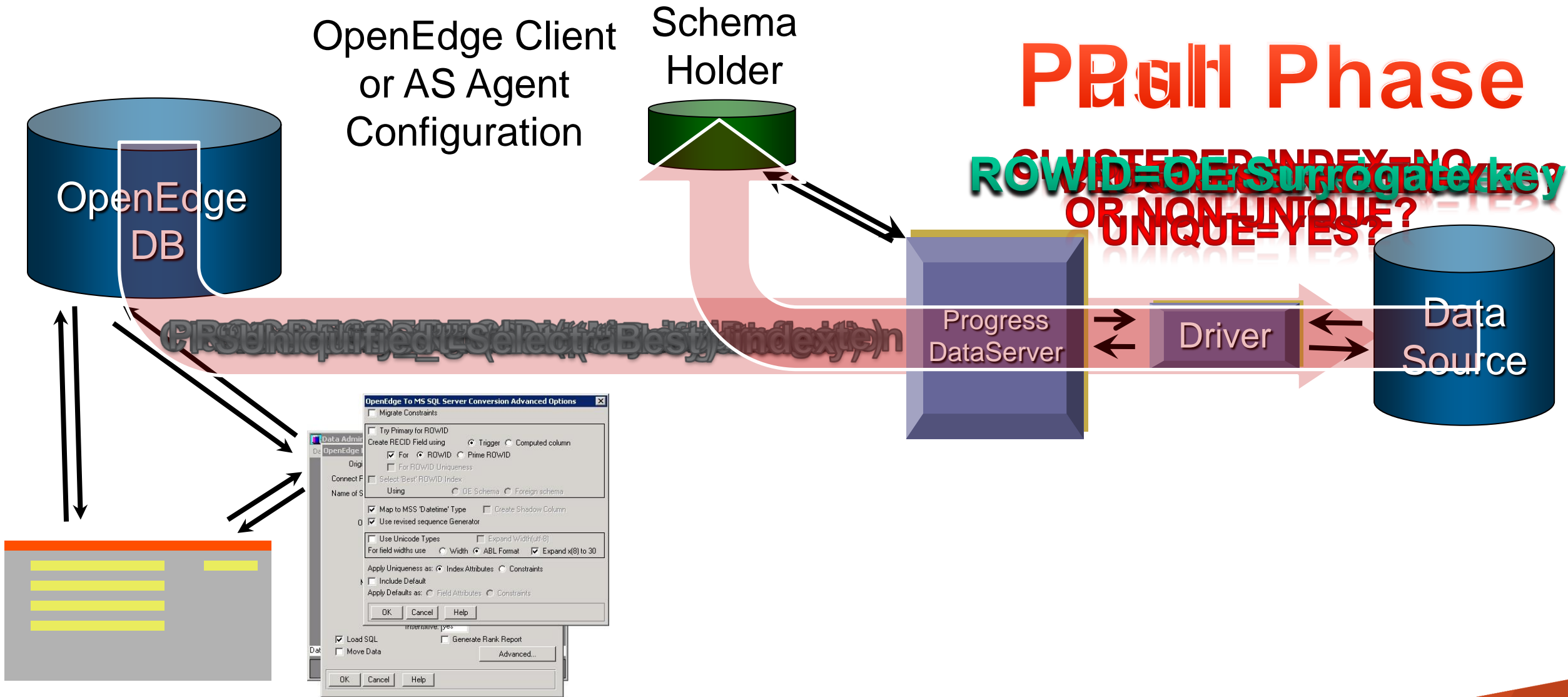
Implementation Strategy and Approach

- What is the best way to enable and deploy this objective since OpenEdge does not support constraints?
 - Allow the server constraint to be defined directly for the migration?
 - Map the server constraint to something in OpenEdge?
 - Search for or derive index candidates?
- How do we compensate for differences in form and function?

	Singular ?	Required?	NULL- Constrained?	Unique- Constrained?
OE Prime Index	X	X		
MSS Prime Index	X	Recommended	X	X
MSS Clustered Index	X	Recommended*		
DS ROWID Index	X	X	Recommended	X

* Required for SQL Azure

Conceptual Overview of the New Migration Options



Agenda

- Intro. and New Deployment Strategy Goals
- Implementation Strategy and Approach
- Advanced Migration Capabilities
- Multi Pass Strategy – a Recommendation
- Bonus Slides

The New Face of MS SQL Server DataServer Deployment

New Migration window selections

OpenEdge DB to MS SQL Server Conversion

Original OpenEdge Database:

Connect Parameters for OpenEdge:

Name of Schema Holder Database:

Logical Database Name:

ODBC Data Source Name:

Username:

User's Password:

Connect Parameters:

Maximum Varchar Length:

Codepage:

Collation:

Insensitive:

Load SQL

Move Data

Generate Rank Report

OpenEdge To MS SQL Server Conversion Advanced Options

Migrate Constraints

Try Primary for ROWID

Create RECID Field using Trigger Computed column

For ROWID Prime ROWID

For ROWID Uniqueness

Select 'Best' ROWID Index

Using OE Schema Foreign schema

Map to MSS 'Datetime' Type Create Shadow Column

Use revised sequence Generator

Use Unicode Types Expand Width(utf-8)

For field widths use Width ABL Format Expand x(8) to 30

Apply Uniqueness as: Index Attributes Constraints

Include Default

Apply Defaults as: Field Attributes Constraints

Visit the "Advanced.." is optional. It is set to provide backward compatible, default behavior.

Rules: Migrate Constraints



Rules:

Primary constraint : Unique and Mandatory
Unique Clustered “constraint:” given priority over Primary constraints for ROWID candidacy.

Primary always becomes clustered implicitly except when an explicit “clustered” constraint definition supersedes it.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		Bidx1	Bidx2



Rules: Migrate Constraints



Rules:
Primary constraint : Unique and Mandatory
Unique Clustered “constraint:” given priority over Primary constraints for ROWID candidacy.
Primary always becomes clustered implicitly except when an explicit “clustered” constraint definition supersedes it.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		Bidx1	Bidx2

Unique
 Mandatory
 Non-clustered

Rules: Migrate Constraints



Rules:
Primary constraint : Unique and Mandatory
Unique Clustered “constraint:” given priority over Primary constraints for ROWID candidacy.

Primary always becomes clustered implicitly except when an explicit “clustered” constraint definition supersedes it.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		Bidx1	Bidx2

Unique
 Mandatory
 Non-clustered

Rules: Migrate Constraints



ROWID

Rules:

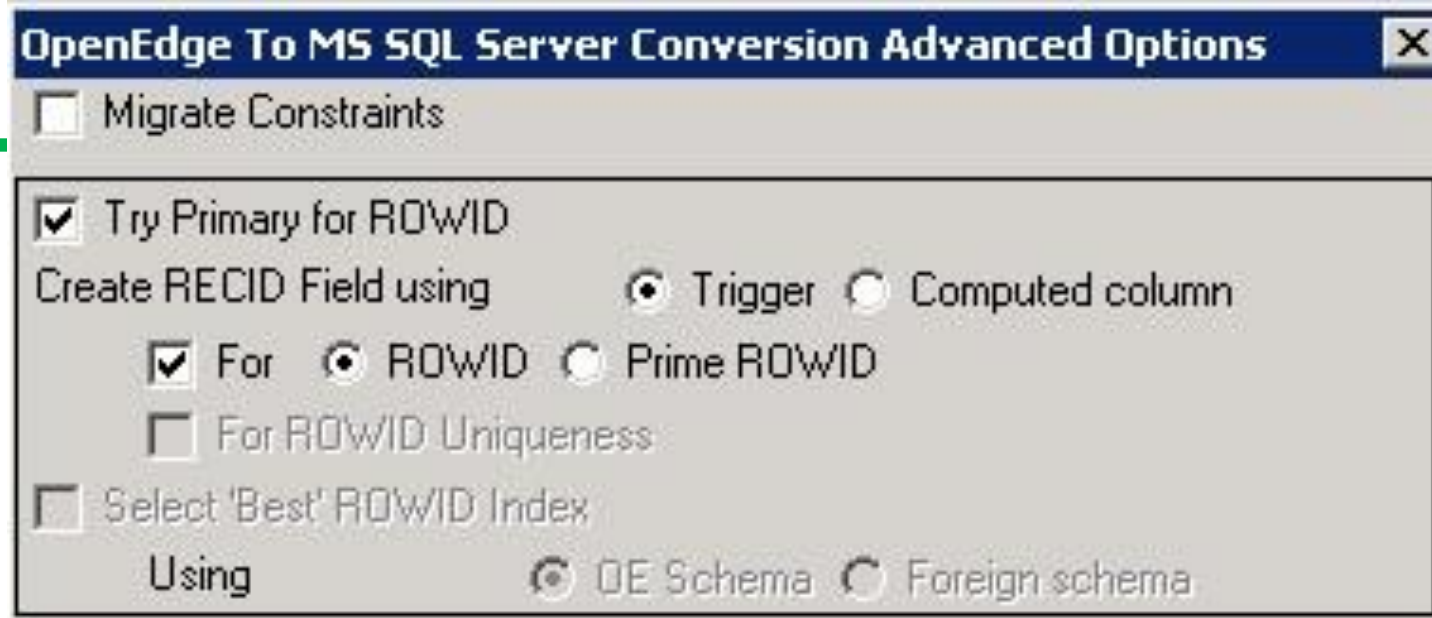
Primary constraint : Unique and Mandatory
Unique Clustered “constraint:” given priority over Primary constraints for ROWID candidacy.

Primary always becomes clustered implicitly except when an explicit “clustered” constraint definition supersedes it.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		Bidx1	Bidx2

Unique	Mandatory	∅	Non-clustered
--------	-----------	---	---------------

Rules: Try Primary for ROWID



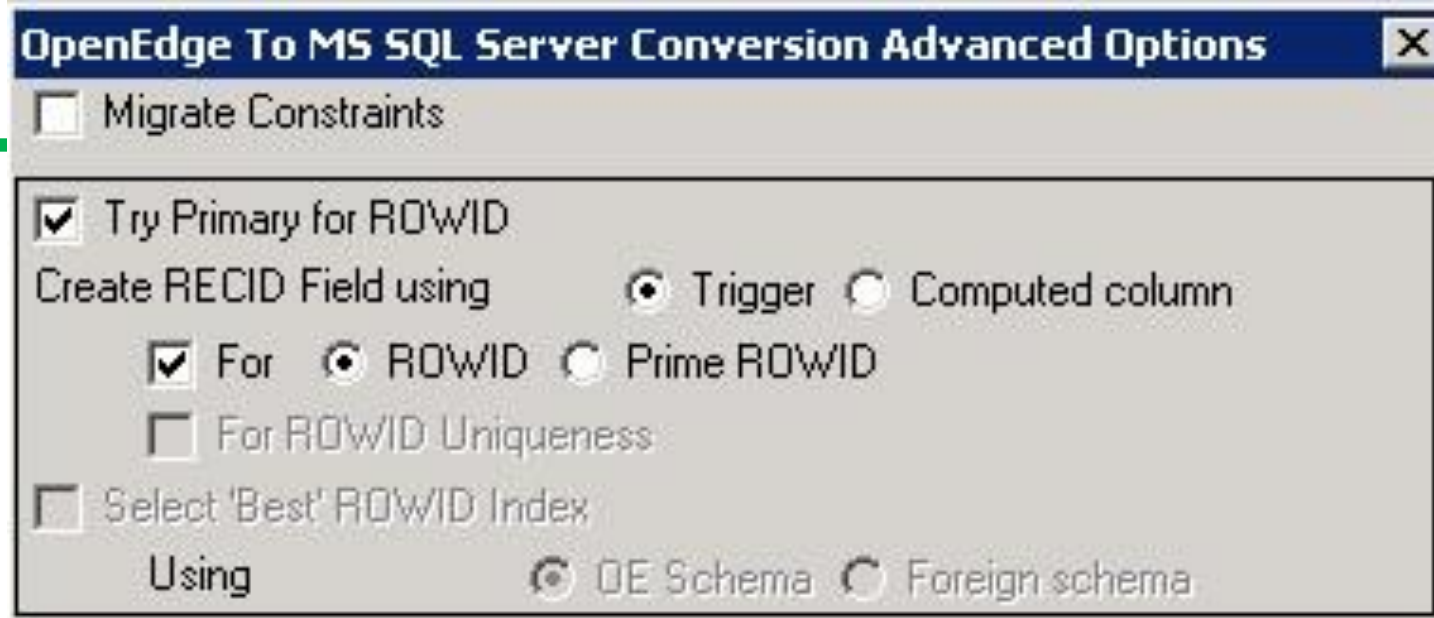
Rules:
If OE Primary is mandatory and Unique – it can become MSS primary

MSS primary is made implicit clustered if no explicit clustered.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx3	Aidx3
B	Ø Bidx1	Bidx2	Bidx3	Bidx4		Bidx3	Bidx3
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3

Unique
 Mandatory
 Ø Non-clustered

Rules: Try Primary for ROWID



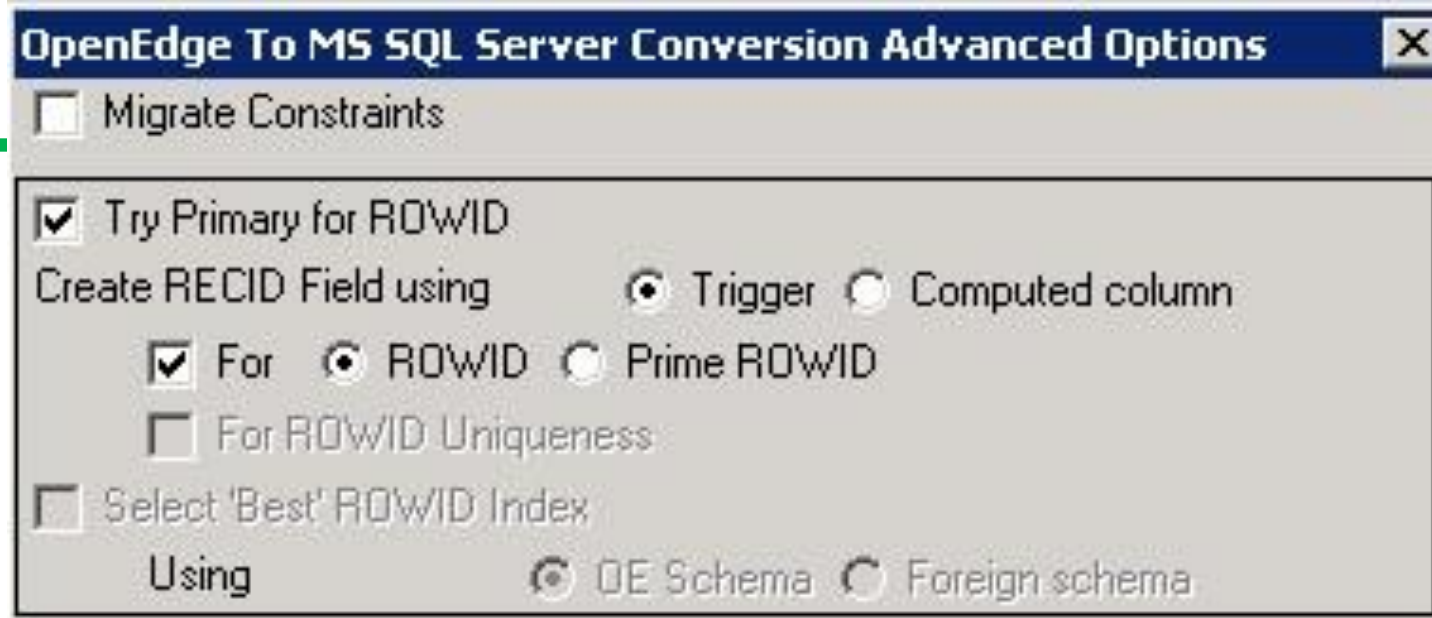
Rules:
If OE Primary is mandatory and Unique – it can become MSS primary

MSS primary is made implicit clustered if no explicit clustered.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx3	Aidx3
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		Bidx3	Bidx3
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3

Unique
Mandatory
∅ Non-clustered

Rules: Try Primary for ROWID



ROWID
↑

Rules:
If OE Primary is mandatory and Unique – it can become MSS primary

MSS primary is made implicit clustered if no explicit clustered.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx3	Aidx3
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		Bidx3	Bidx3
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3

Unique
Mandatory
∅ Non-clustered

Rules: Create RECID Field

OE 11.3 Migration Dialog

OE 10.2B Migration Dialog

11.3 →
Default

Migrate Constraints

Try Primary for ROWID

Create RECID Field using Trigger Computed column

For ROWID Prime ROWID

For ROWID Uniqueness

Select 'Best' ROWID Index

Using OE Schema Foreign schema

Create RECID Field Load SQL Move Data

Create Shadow Columns Include Defaults Use Unicode Types

Expand Width (utf-8) Use Revised Sequence Generator

Map to MSS 'Datetime' Type

For field widths use: Width ABL Format Expand x(8) to 30

For Create RECID use: Trigger Computed column

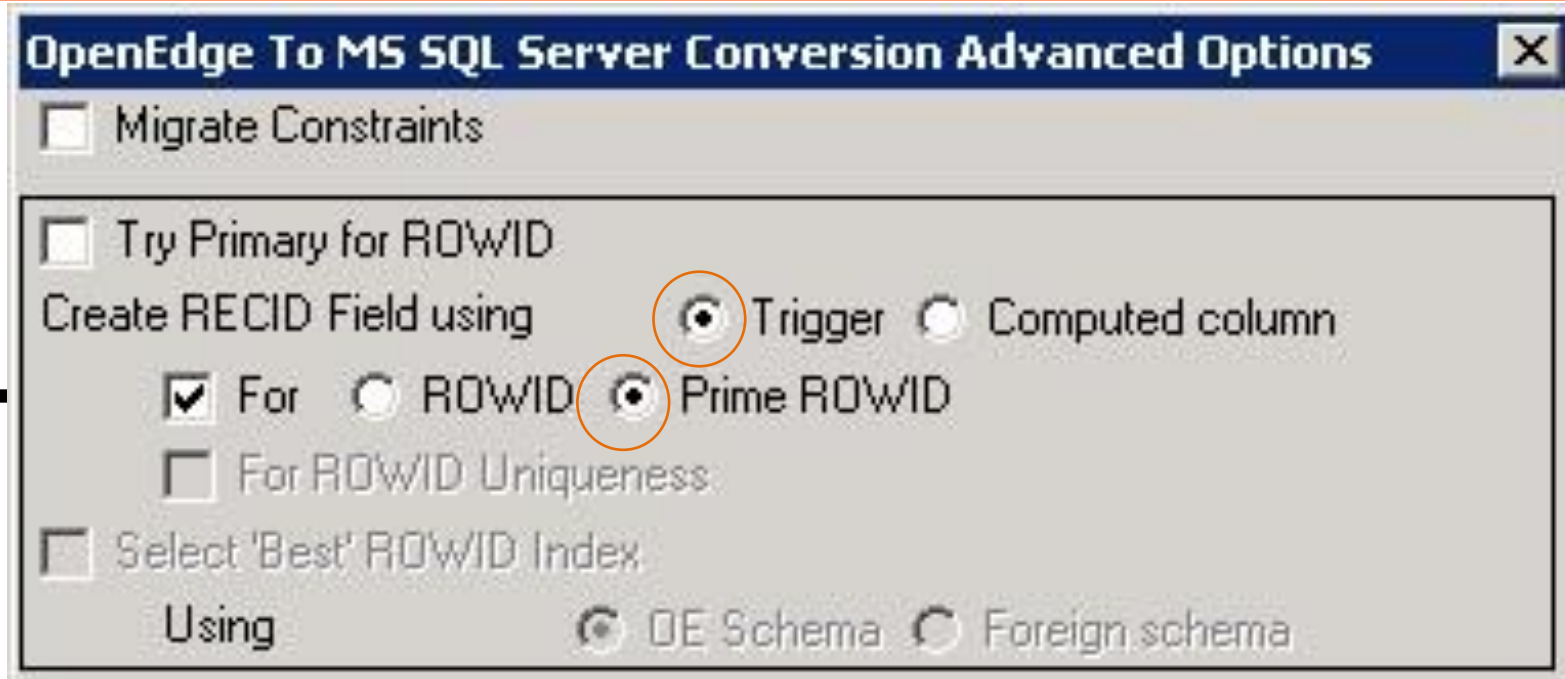
Rule:
PROGRESS_RECID is the Default ROWID/RECID and is always used for ROWID/RECID if it exists.

Either INSERT Trigger or Computed Column can be used to generate the PROGRESS_RECID value.
Backward-compatible behavior is to add a non-clustered hidden index

Table	Constraint Defs.		OpenEdge Indexes			Added Indexes	
	Primary	Clustered	Primary	Other Indexes			Other Indexes
A	Aidx1		Aidx3	Aidx4	Aidx5	∅	PROGRESS_RECID
B	Bidx1	Bidx2	Bidx3	Bidx4		∅	PROGRESS_RECID
C			Cidx3	Cidx4	Cidx5	∅	PROGRESS_RECID

Unique
 Mandatory
 ∅
 Non-clustered

Rules: Create RECID Field With Prime ROWID Using Trigger



RECID/
ROWID



Rule:

The insert trigger creates a PROGRESS_RECID field that is unique but non-mandatory.

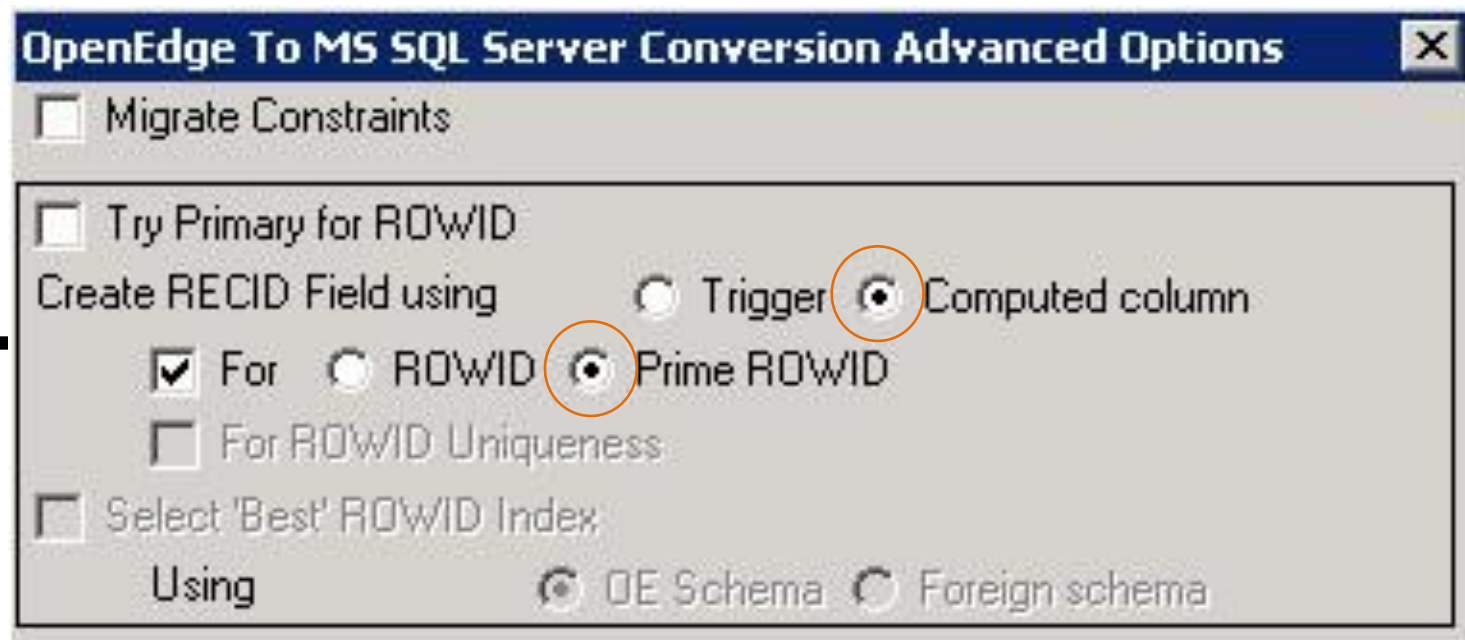
Only mandatory index components can be made primary.

Trigger can derive a unique clustered index for ROWID/RECID

Table	Constraint Defs.		OpenEdge Indexes			Added Indexes	
	Primary	Clustered	Primary	Other Indexes			Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	PROGRESS_RECID	
B	Ø Bidx1	Bidx2	Bidx3	Bidx4		PROGRESS_RECID	
C			Cidx3	Cidx4	Cidx5	PROGRESS_RECID	

Unique	Mandatory	Ø	Non-clustered
--------	-----------	---	---------------

Rules: Create RECID Field With Prime ROWID Using Computed Column



Rule:

The computed column option creates a **PROGRESS_RECID** field that is unique and mandatory.

A computed column can derived a **PROGRESS_RECID** that can be made MSS primary and implicit clustered).

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	P_RECID	P_RECID
B	Bidx1	Bidx2	Bidx3	Bidx4		P_RECID	P_RECID
C			Cidx3	Cidx4	Cidx5	P_RECID	P_RECID

Unique	Mandatory	∅	Non-clustered
--------	-----------	---	---------------

**RECID/
ROWID**

Rules: Triple Combination



Rule:

Precedence of options is top down
 Lower precedence option are unused if ROWID derivations were established at higher levels.
 Only a single component binary integer supports RECID

Table	MSS Constraints		OE Indexes			MSS Attributes	
	Primary	Clust.	Primary	Other indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅Bidx1	Bidx2	Bidx3	Bidx4		∅Bidx1	Bidx2
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3
D			Didx3	Didx4			P RECID

Unique
Mandatory
∅ Non-clustered

RECID

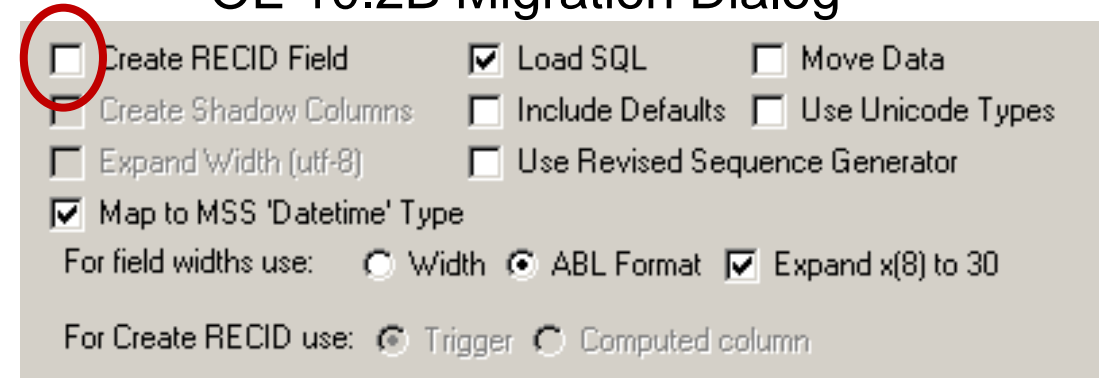
ROWID

Rules: Select 'Best' ROWID Index

OE 11.3 Migration Dialog

OE 10.2B Migration Dialog

11.3
Default



Rule:

As long as none of the other OE 11 options for ROWID designation are set.

OE 10: Un Checking "Create ROWID Field" \equiv OE 11: Un Checking "Create ROWID Field"

OE 10: Checking "Create ROWID Field" \equiv OE 11: Checking "Create ROWID Field using" => "FOR" -> "ROWID"

But uses new Algorithm if one of the new option is selected

The 'Select Best' Index Selection Criteria

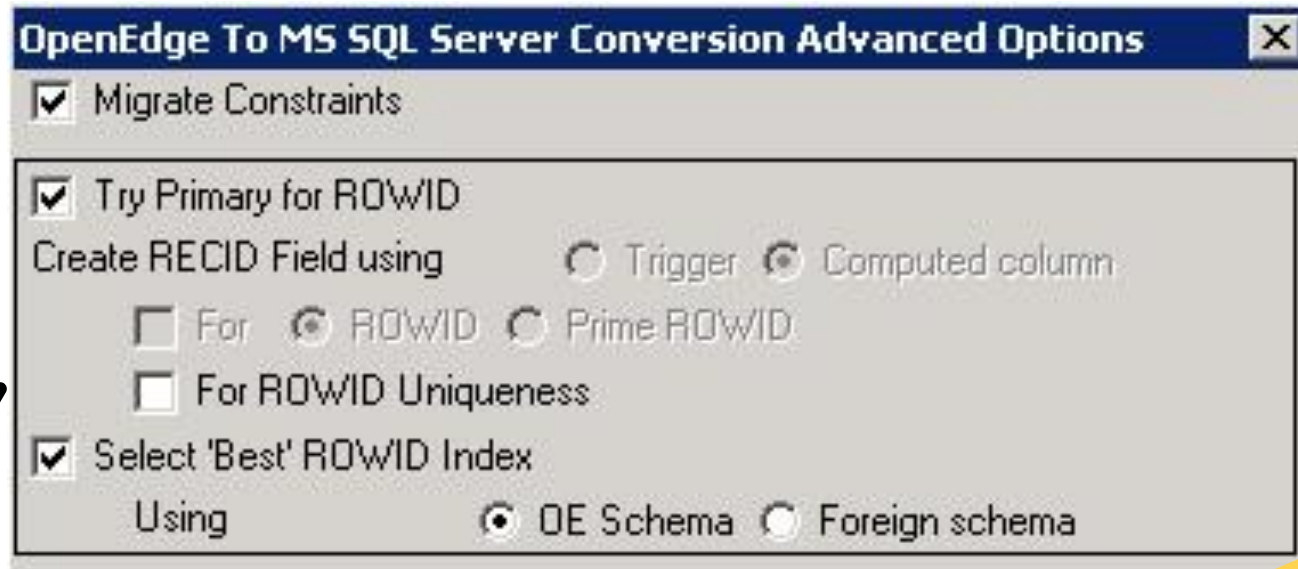
- A single component, unique, integer, binary key is always the preferred choice
- Compact key indexes in terms of components and size are preferred
- “Mandatory” is not a required attribute but definitely a preferred attribute
- Level of an index is a priority assigned to Indexes. Levels based on
 - “Mandatory” status of an index
 - Indexes with all mandatory columns are always ranked higher than columns without mandatory columns.
 - Number of Index components
 - Data types of components
- Ranking and designation based on index weight
 - Index level combined with Index size decide the weight of an index
 - Lowest weight is best among all candidate indexes
 - Selected ROWID candidate index is made Primary constraint if unique otherwise clustered – provided there are no explicit constraint definitions

How Do We Choose an ROWID Index ?

ROWID Designation Rules

- Prefer Existing indexes over derived one
- Seek a Primary first with an implied clustered
- If a clustered index is unique, it is first choice for the ROWID designation
- Indexes with mandatory components are preferred over non-mandatory indexes
- If PROGRESS_RECID is found in the table, it becomes ROWID/RECID no matter what other options are available
- Make unique option “For ROWID Uniqueness” is prevented from consideration at lower precedent levels if higher precedent level can handle uniqueness requirement of ROWID

Rule: Select All Options With a “Select ‘Best’ ROWID Index” Catch-All



	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1		Aidx3	Aidx4	Aidx5
B	∅ Bidx1	Bidx2	Bidx3	Bidx4	
B1		B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3		B3idx2	B3idx3	B3idx4	
C			Cidx3	Cidx4	Cidx5
C1			C1idx3	C1idx4	
C2			C2idx3	C2idx4	
C3			C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rules:

- Precedence given to constraints
- Non-unique indexes cannot serve as ROWID

Rule: Select All Options With a “Select ‘Best’ ROWID Index” Catch-All



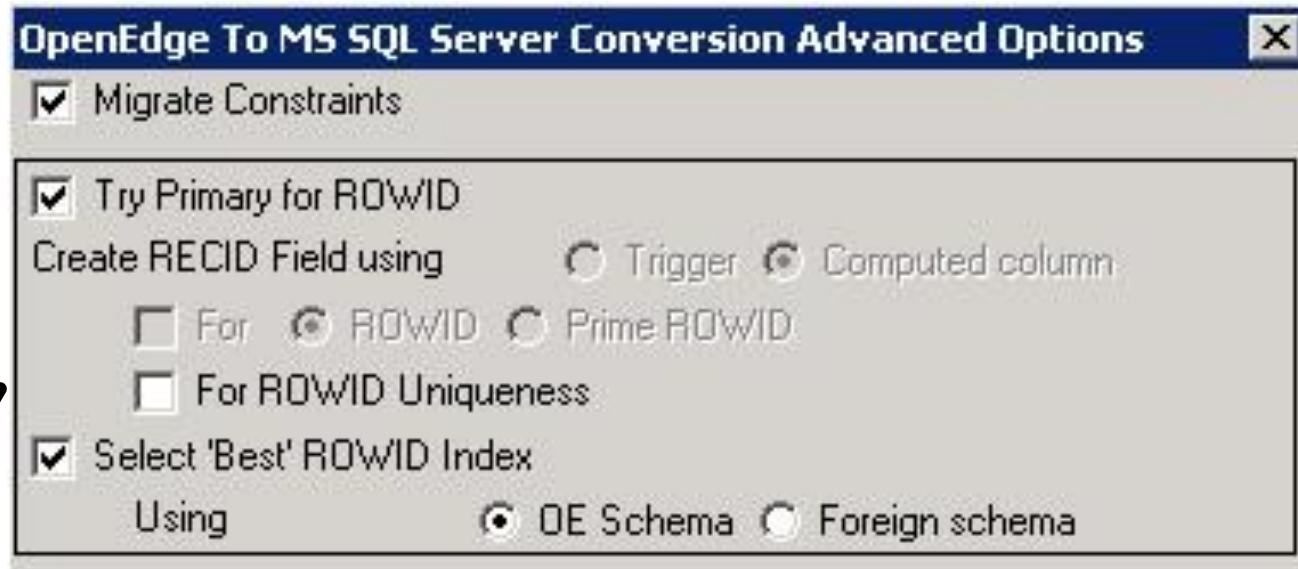
	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1	Aidx1	Aidx3	Aidx4	Aidx5
B	∅ Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx4	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3	∅ B3idx1	B3idx2	B3idx3	B3idx4	
C			Cidx3	Cidx4	Cidx5
C1			C1idx3	C1idx4	
C2			C2idx3	C2idx4	
C3			C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rules:

- Any unique clustered index can serve as ROWID
- When ROWID is derived from the MSS primary, we stop the ROWID search. When derived from clustered, continue to search for a MSS primary

Rule: Select All Options With a “Select ‘Best’ ROWID Index” Catch-All



	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Rules:

- Precedence of options is top down
- Non-mandatory, unique, clustered can serve as ROWID
- “Select ‘Best’ ROWID Index” catch-all finds a non-clustered MSS primary

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1	Aidx1	Aidx3	Aidx4	Aidx5
B	Bidx1	Bidx2	Bidx3	Bidx4	
B1	B1idx4	B1idx2	B1idx3	B1idx4	
B2	B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	C1idx4	C1idx3	C1idx3	C1idx4	
C2			C2idx3	C2idx4	
C3			C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rule: Select All Options With a “Select ‘Best’ ROWID Index” Catch-All



	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1	Aidx1	Aidx3	Aidx4	Aidx5
B	∅ Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx4	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	∅ C1idx4	C1idx3	C1idx3	C1idx4	
C2	C2idx4	C2idx4	C2idx3	C2idx4	
C3	C3idx3	C3idx3	C3idx3	C3idx4	
C4		C4idx3	C4idx3	C4idx4	C5idx5

Rules:

- Non-unique indexes are ineligible for ROWID
- When ROWID is derived from the MSS primary, we stop the ROWID search. When derived from clustered, continue to search for a MSS primary.
- ROWID derived from primary is preferred over a clustered derivation

For ROWID Uniqueness: a Better “Select ‘Best’ ROWID Index” Catch-All



	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1	Aidx1	Aidx3	Aidx4	Aidx5
B	Bidx1	Bidx2	Bidx3	Bidx4	
B1		B1idx2	B1idx3	B1idx4	
B2	B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	C1idx4	C1idx3	C1idx3	C1idx4	
C2			C2idx3	C2idx4	
C3			C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rules:

- ROWID uniqueness is applied in order of precedence (except at top-level “Migrate Constraints”) and no longer applied once constraints are successfully derived
- Can be applied to any other non-unique index being evaluated for primary, clustered and/or ROWID

For ROWID Uniqueness: a Better “Select ‘Best’ ROWID Index” Catch-All



	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1		Aidx3	Aidx4	Aidx5
B	∅ Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx3	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	∅ C1idx4	C1idx3	C1idx3	C1idx4	
C2			C2idx3	C2idx4	
C3			C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rules:

- ROWID uniqueness is applied in order of precedence but is not applied to top-level “Migrate Constraints”
- ROWID uniqueness is not applied to OE constraint definitions that are migrated. Constraint definitions are honored in the form the user provided them in

For ROWID Uniqueness: a Better “Select ‘Best’ ROWID Index” Catch-All



	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1		Aidx3	Aidx4	Aidx5
B	∅ Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx3	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	∅ C1idx4	C1idx3	C1idx3	C1idx4	
C2	C2idx3	C2idx3	C2idx3	C2idx4	
C3	∅ C3idx4	C3idx3	C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rules:

- ROWID uniqueness is applied in order of precedence
- Uniqueness is applied to non-unique indexes at each level until constraints are successfully derived

For ROWID Uniqueness: a Better “Select ‘Best’ ROWID Index” Catch-All



	Unique		Mandatory
	Non-Unique		Non-Mandatory
∅	Non-clustered		

Table	MSS Constraints		OE Indexes		
	Primary	Clust.	OE Primary	Other indexes	
C4			C4idx3	C4idx4	C4idx4
C5			C5idx3	C5idx4	C5idx5

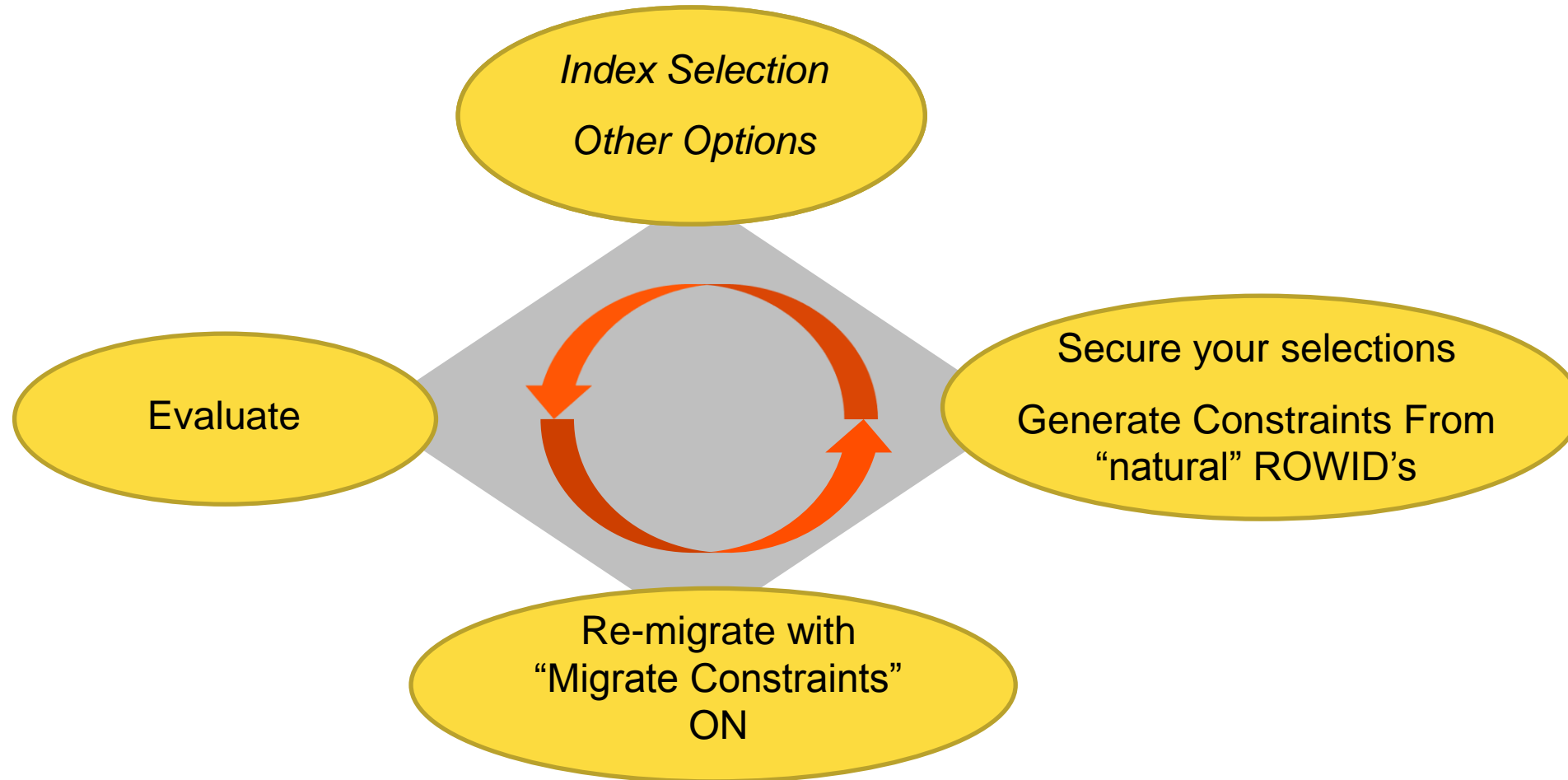
Rules:

- Existing index is preferred over derived index
- Uniqueness is applied to non-unique indexes at each level until constraints are successfully derived

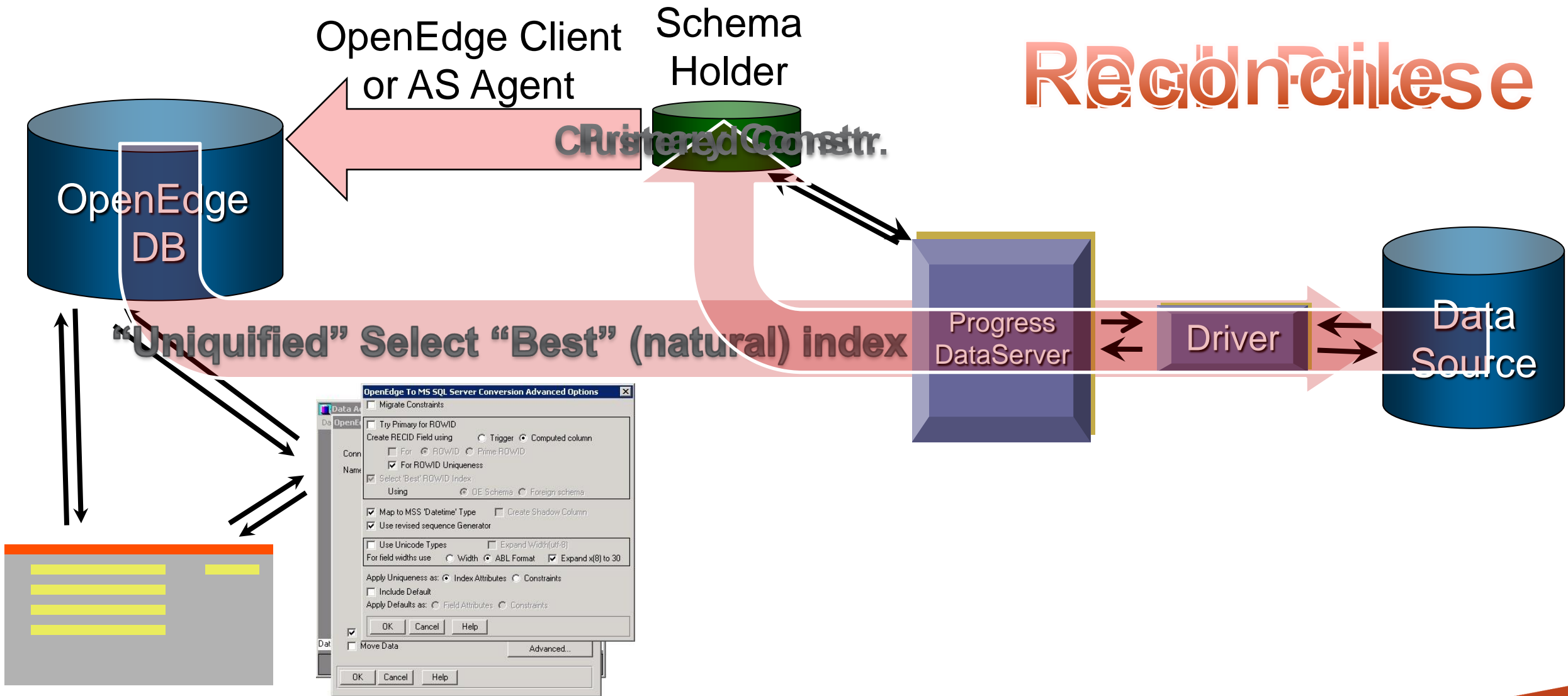
Agenda

- Intro. and New Deployment Strategy Goals
- Implementation Strategy and Approach
- Advanced Migration Capabilities
- Multi Pass Strategy – a Recommendation
- Bonus Slides

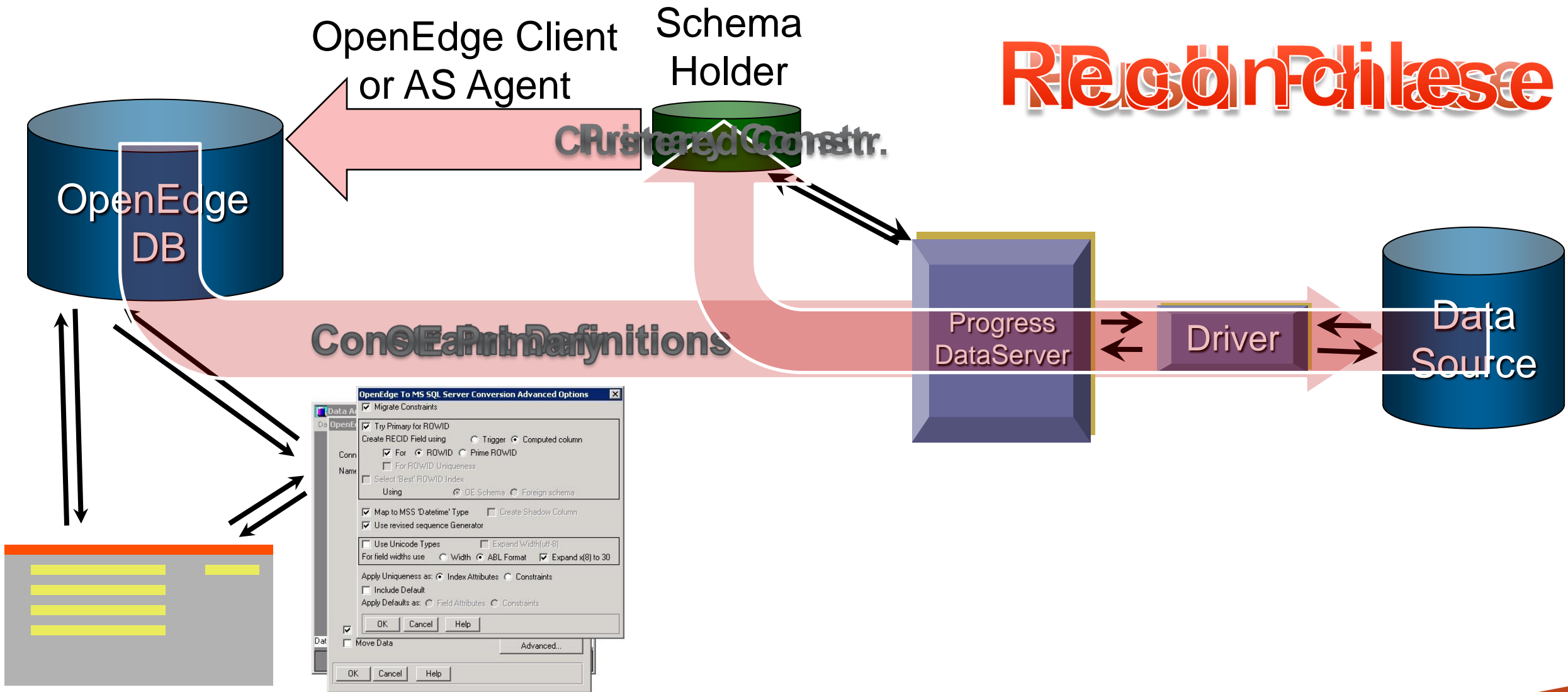
ROWID Designation Multi Pass Strategy



New Migration – Iteration 1



New Migration – Iteration 2



Summary

- **Main objective:** Improve overall DataServer performance by optimizing data access
 - The Best Solution:
Primary constraint → sets Clustered index (implicitly) → sets ROWID index selection.
- Remember
 - This is an “automated” tool: fuzzy inputs may cause fuzzy results.
 - The tool cannot replace your critical knowledge of both database and application



PROGRESS

Agenda

- Intro. and New Deployment Strategy Goals
- Implementation Strategy and Approach
- Advanced Migration Capabilities
- Multi Pass Strategy – a Recommendation
- Bonus Slides

Implementation Considerations

- OpenEdge primary can be any index whereas MSS primary must be unique/mandatory
- What if a particular table requires a different primary constraint from the clustered index?
- We want clustered index for ROWID performance even if the primary constraint is different
But we want primary keys unique and mandatory attributes for ROWID.
- If ROWID candidate must be unique, can we increase the pool of ROWID candidates by attaching uniqueness to them?
- What if I need several methodologies of locating ROWID candidates. Can we take an iterative approach?
- How do I select the “Best” ROWID candidate?
- Should I use natural or surrogate keys?
- There is no substitute for knowing your database and applications.

Application Optimization – Clustering OE Primary as ROWID

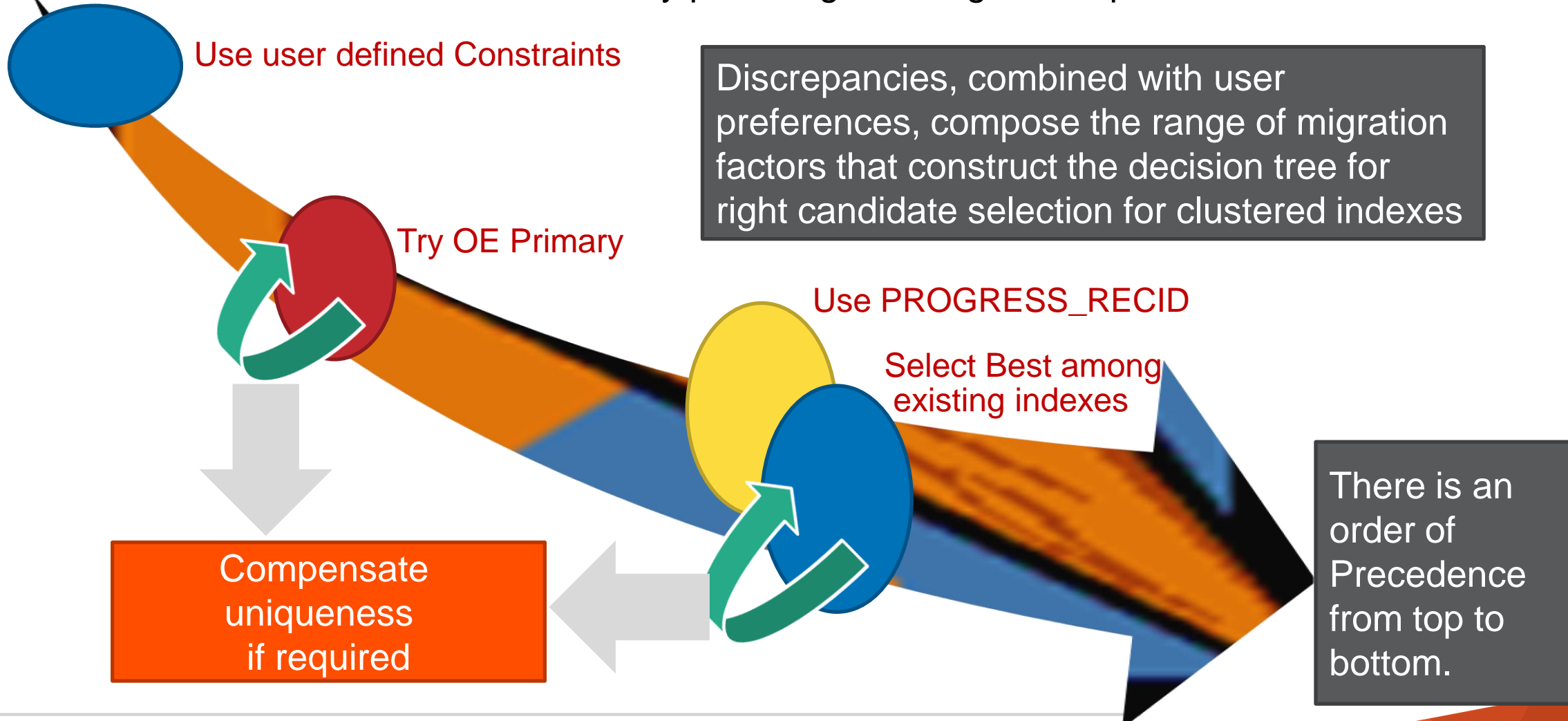
■ Clustering OE Primary indexes

- All OE DB files capable of being migrated have a prime index.
- OE primary mainly supplies a default sort order for queries whose sort criteria are not otherwise specified.
- SQL Server's clustered index sorts and stores the physical content of data rows based on its key values.
- Since there are characteristic differences between SQL and OE primary keys, the server mapping may or may not be exact in nature.

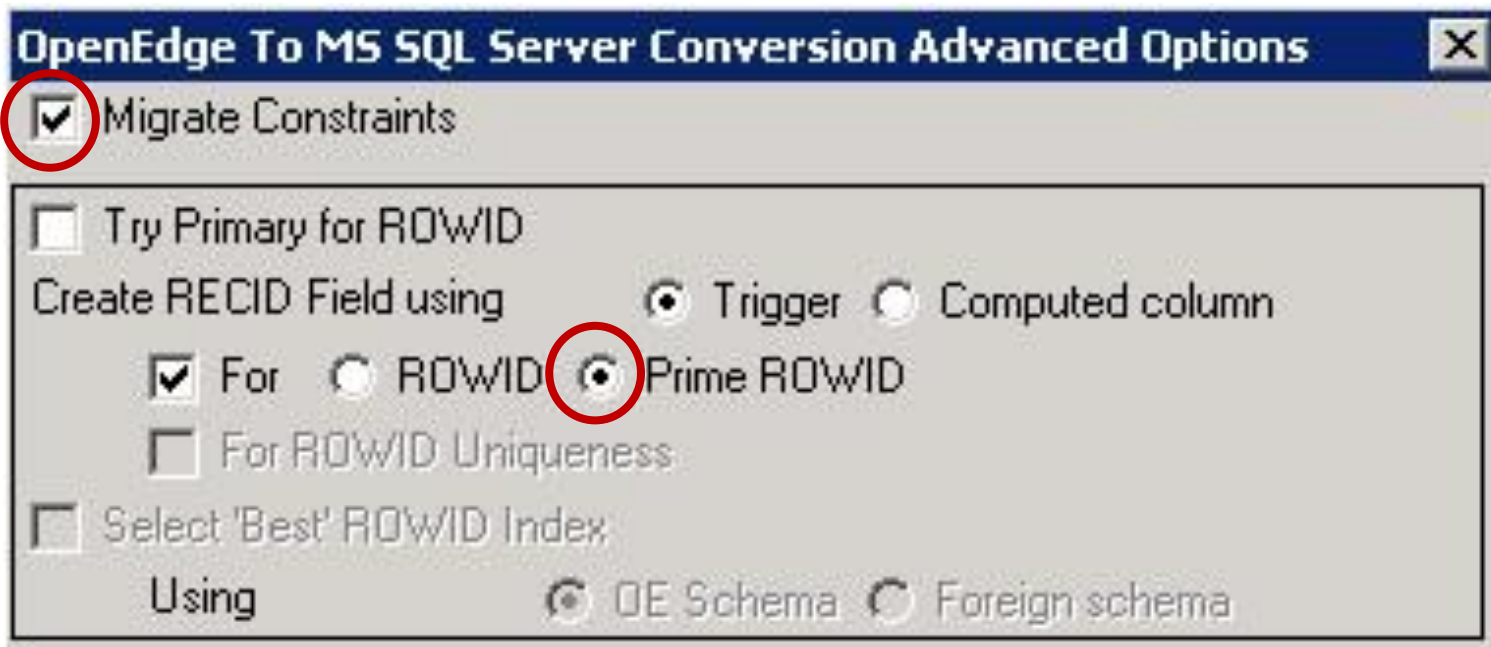
	Singular ?	NULL-Constrained ?	Unique-Constrained ?
OE Prime Index	X		
MSS Prime Index	X	X	X
MSS Clustered Index	X		
DataServer ROWID Index	Preferred	Preferred	Required

Advanced Migration Capabilities

Facilitate finer control on ROWID selection by providing new migration options



Rules: Combining Migrate Constraints with Create ROWID Field



Rule:

- Precedence of options is top down

Any unique clustered will be assigned ROWID.

Only a single component binary integer supports RECID

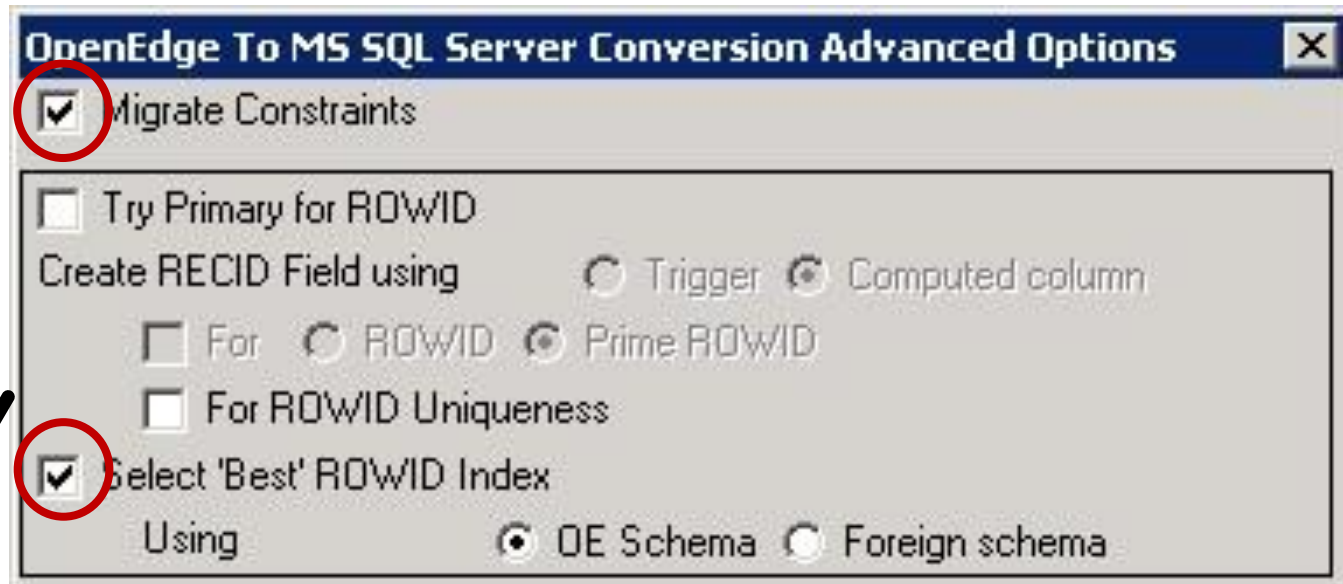
Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		∅ Bidx1	Bidx2
C			Cidx3	Cidx4	Cidx5		P_RECID

Legend: Unique (Green), Mandatory (Blue), ∅ (White), Non-clustered (Grey)

ROWID



Rules: “Migrate Constraints” with “Select ‘Best’ ROWID Index”



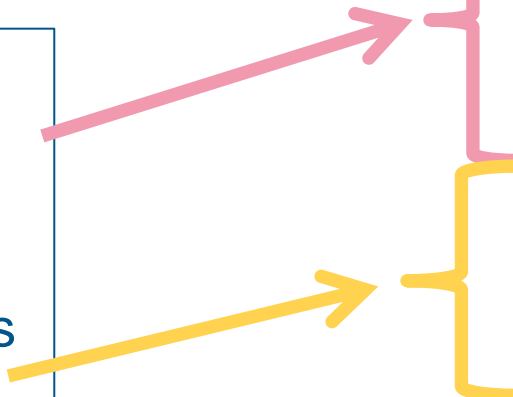
■	Unique	■	Mandatory
■	Non-Unique	■	Non-Mandatory
∅	Non-clustered		



Rules:

- Precedence of options is top down
- “Select ‘Best’ ROWID Index” can act like a catch-all (provided there is always an existing, eligible unique index to server as ROWID)

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	■ Aidx1		■ Aidx3	■ Aidx4	■ Aidx5
B	■ ∅ Bidx1	■ Bidx2	■ Bidx3	■ Bidx4	
B1		■ B1idx2	■ B1idx3	■ B1idx4	
B2	■ ∅ B2idx1	■ B2idx2	■ B2idx3	■ B2idx4	
B3		■ B3idx2	■ B3idx3	■ B3idx4	
C			■ Cidx3	■ Cidx4	■ Cidx5
C1			■ C1idx3	■ C1idx4	
C2			■ C2idx3	■ C2idx4	
C3			■ C3idx3	■ C3idx4	



Rule: Select Options all with a “Select ‘Best’ ROWID Index” Catch-All



█	Unique	█	Mandatory
█	Non-Unique	█	Non-Mandatory
█	∅	█	Non-clustered

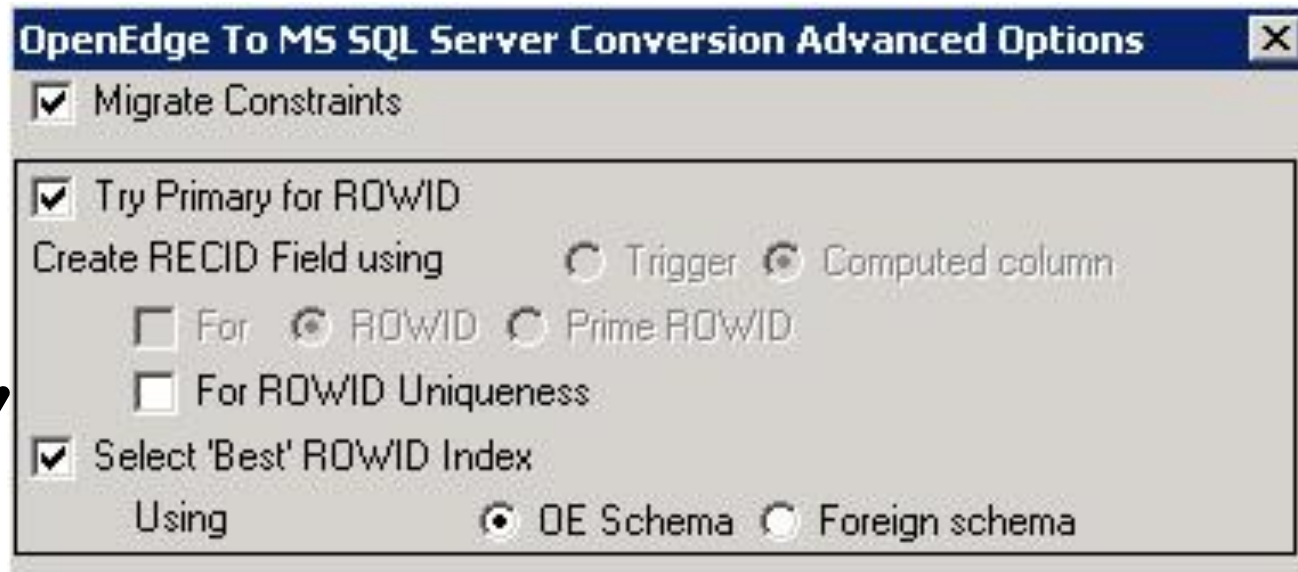


Rules:

- Non-unique indexes cannot serve as ROWID

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1	Aidx1	Aidx3	Aidx4	Aidx5
B	∅ Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx4	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3		B3idx2	B3idx3	B3idx4	
C			Cidx3	Cidx4	Cidx5
C1			C1idx3	C1idx4	
C2			C2idx3	C2idx4	
C3			C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rule: Select all Options with a “Select ‘Best’ ROWID Index” Catch-All



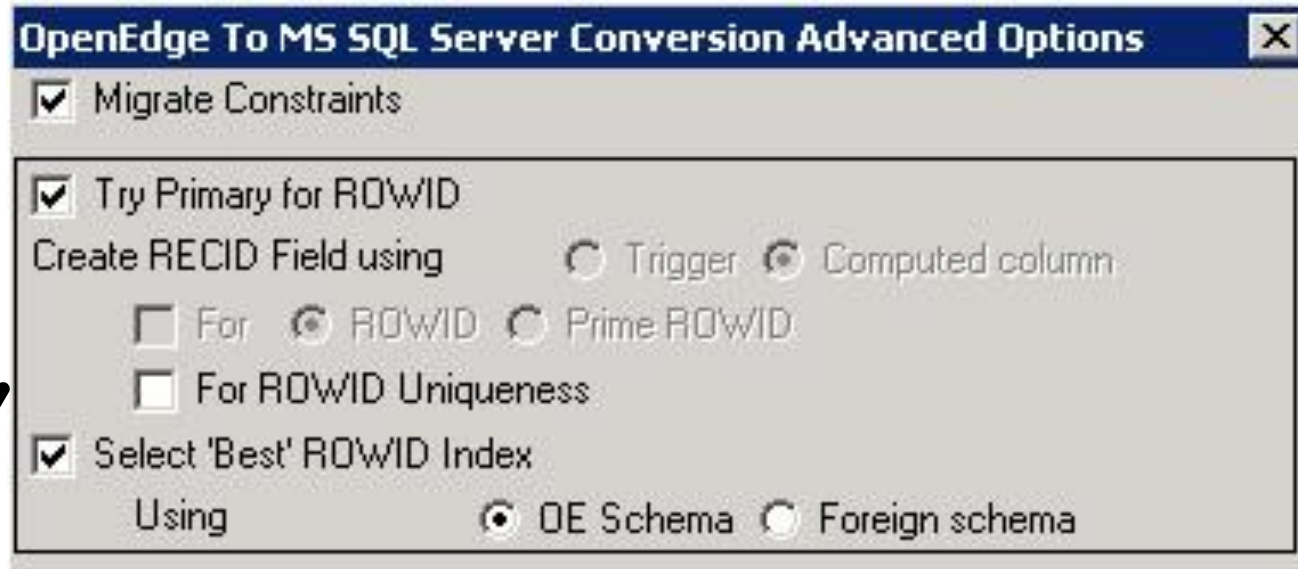
 Unique	 Mandatory
 Non-Unique	 Non-Mandatory
 ∅ Non-clustered	

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1	Aidx1	Aidx3	Aidx4	Aidx5
B	Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx4	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	∅ C1idx4	C1idx3	C1idx3	C1idx4	
C2	C2idx4	C2idx4	C2idx3	C2idx4	
C3		C3idx4	C3idx3	C3idx4	
C4			C4idx3	C4idx4	C5idx5

Rules:

- Non-unique indexes are ineligible for ROWID
- Any unique index can be made clustered and serve as ROWID
- When ROWID is derived from the MSS primary, we stop the ROWID search. When derived from clustered, continue to search for a MSS primary

Rule: Select all Options with a “Select ‘Best’ ROWID Index” Catch-All



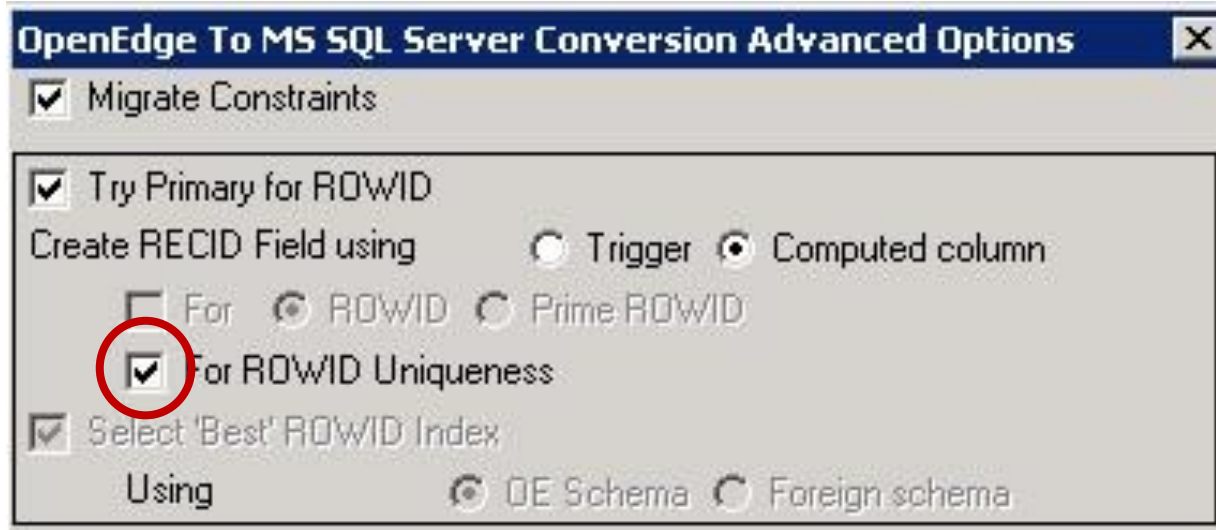
█	Unique	█	Mandatory
█	Non-Unique	█	Non-Mandatory
█	∅	█	Non-clustered

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1	Aidx1	Aidx3	Aidx4	Aidx5
B	Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx4	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	C1idx4	C1idx3	C1idx3	C1idx4	
C2	C2idx4	C2idx4	C2idx3	C2idx4	
C3		C3idx4	C3idx3	C3idx4	
C4	∅	C4idx3	C4idx3	C4idx4	C5idx5

Rules:

- Any unique index can be made clustered and serve as ROWID
- Non-unique indexes are ineligible for ROWID
- When ROWID is derived from the MSS primary, we stop the ROWID search
When derived from clustered, continue to search for a MSS primary

For ROWID Uniqueness: a Better “Select ‘Best’ ROWID Index” Catch-All



 Unique	 Mandatory
 Non-Unique	 Non-Mandatory

∅ Non-clustered

Table	Constraint Defs.		OpenEdge Indexes		
	Primary	Clustered	Primary	Other Indexes	
A	Aidx1		Aidx3	Aidx4	Aidx5
B	∅ Bidx1	Bidx2	Bidx3	Bidx4	
B1	∅ B1idx3	B1idx2	B1idx3	B1idx4	
B2	∅ B2idx1	B2idx2	B2idx3	B2idx4	
B3	B3idx3	B3idx2	B3idx3	B3idx4	
C	Cidx3	Cidx3	Cidx3	Cidx4	Cidx5
C1	∅ C1idx4	C1idx3	C1idx3	C1idx4	
C2	C2idx3	C2idx3	C2idx3	C2idx4	
C3		C3idx3	C3idx3	C3idx4	
C4	∅ C4idx5	C4idx3	C4idx3	C4idx4	C5idx5

Rules:

- ROWID uniqueness is applied in order of precedence
- Uniqueness is applied to non-unique indexes at each level until constraints are successfully derived
- Only indexes used for constraint derivation retain uniqueness

The 'Select Best' Index Evaluation

- Level and weight calculation

idx1		idx2		idx3	
f1	char(30)	f1	int	f1	int
f2	char(20)	f2	int	f2	int
		f3	char(20)		
Uniq and Mand		Uniq and mand		Uniq nonMand	

Idx1 size = 30 + 20 = 50
Exponent = TRUNCATE(Index_size / 8, 0) = 50/8 = 6
Remainder = Index_size MOD 8. = 50 Mod 8 = 2
Multiplier = Exponent + (IF remainder > 0 then 1 else 0) = 6 + 1 = 7
Weight = (Index_level + No_of_idx_fields) * multiplr.
 = (9 + 2) * 7 = 77

Idx2 Weight = 60

Idx3 Weight = 32

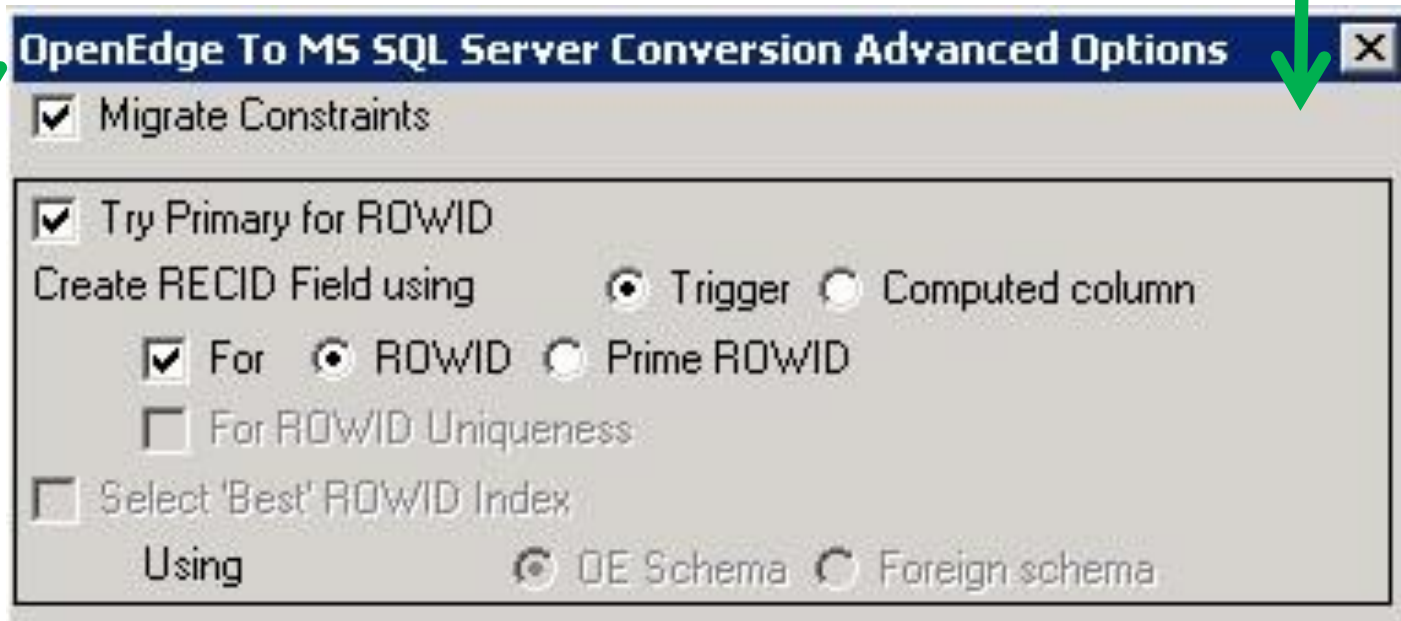
- Indexes are designated based on level followed by weight if there is a tie
- "Idx1" and idx2 are at same level (9) but idx2 has less weight, hence gets the preference over idx3

Unique Index Levels

Lvl	DataType	#
1	int only	=1
2	BIGINT only	=1
3	int,int	2
4	anything except date float	=1
5	anything except float	=1
6	anything	=1
7	int only	>2
8	BIGINT only	>1
9	anything except date float	>1
10	anything except float	>1
11	anything	>1
non mandatory		
12	int only	=1
13	BIGINT only	=1
14	int,int	2

is Number of index fields

Rules: Combining “Migrate Constraints” with “Try Primary for ROWID”



Rules:
Precedence of options is top down
When constraints are migrated, they are always explicit changes to the server

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		∅ Bidx1	Bidx2
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3

Unique
 Mandatory
 Non-clustered

Use Cases – ROWID Designation

Case-1: Customer			
		idxname(c-firstname,c-lastname)	
custno	int64	idxlocation(area-code,regn-code)	
c-firstname	x(20)	idxcustno(custno)	
c-lastname	x(20)	<div style="background-color: yellow; padding: 5px;"> 102B: idxname 11.x : idxcustno </div>	
Area-code	int		
Regn-code	int		

Case-3 : Salesrep		salesqty(sales-rep,salesquota)	
sales-rep	x(10)	salesarea(area-code,sname)	
salesquota	int	<div style="background-color: yellow; padding: 5px;"> 102B: no index designated 11.x : Uniquify option chosen will designate ROWID </div>	
Area-code	int		
sname	x(15)		
sales-term	date		

Case-2: Department		dept-ident(cost-centre-code, deptno)	
cost-centre-code	x(10)	dept-desc (deptname, location-code)	
Deptno	int	<div style="background-color: yellow; padding: 5px;"> 102B: no index designated 11.x : dept-desc </div>	
Deptname	x(25)		
location-code	int		

	Unique Index
	Mandatory index
	Non-Unique index
	Non-Mandatory index





Use Cases

Case-4:Dept		dept-ident(cost-centre-code,deptno)	
cost-centre-code	x(10)	dept-desc(deptname,location-code)	
Deptno	int	Enforcing a ROWID designation	
Deptname	x(25)		
location-code	int		

102B: User has no control, whichever index is first in sequence will get designated
 11.x : User can enforce an index to be designated as ROWID in case it meets eligibility criteria

Case-5 : Item		Itemstock(Item-num, on-hand)	
Item-num	int64	ItemSubsStock(item-num, on-hand,subs-item)	
idesc	x(40)	Leverage foreign DB recommendation of ROWID candidate	
subs-item	int		
cost	decimal		
loc	char		
on-hand	int		

102B: Itemstock
 11.x: OE algorithm will designate ItemStock, But MSS SQL server will designate ItemSubsStock

	Unique Index		Non-Unique index
	Mandatory index		Non-Mandatory index

ROWID Designation Multi Pass Strategy

Evaluate transactional Performance improvement. Determine acceptable performance

In Ver. 11.x, Progress recommends that the “best index” approach be favored as a paradigm shift from RECID generation process of the legacy migration model where ROWID was derived from “Create RECID Field” default selection. It is also a good way to find a “natural key” that likely already exists in the foreign data source

Index Selection
Select ‘Best’ ROWID Index’

Evaluate

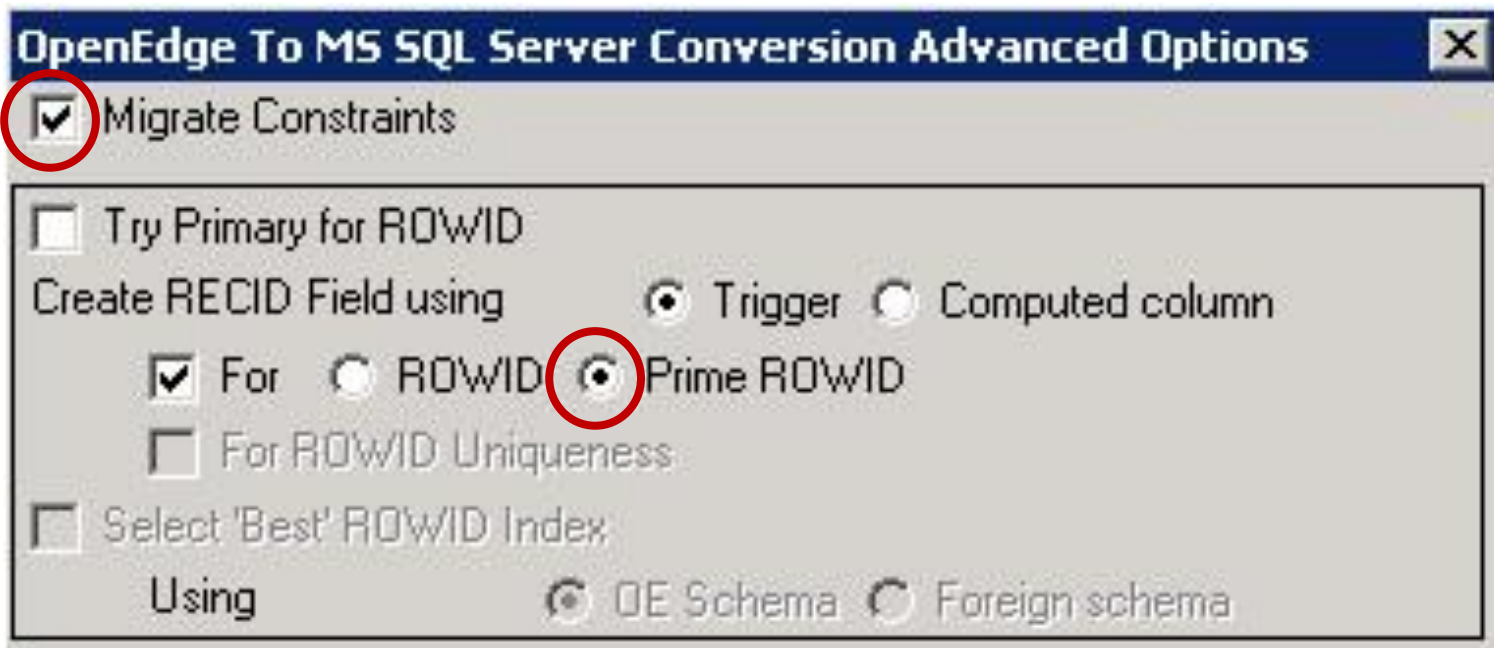
Secure your selections
Generate Constraints From
“natural” ROWID’s

Apply business requirements to algorithmic selections and evaluate performance

Re-migrate with
“Migrate Constraints”
ON

To get the performance benefits of the ROWID choices derived from the “Select ‘Best’ ROWID Index’ option, the ROWID choice would need to be utilized as primary and clustered indexes wherever possible

Rules: Combining Migrate Constraints with Create RECID Field



RECID

ROWID

Rule:

Precedence of options is top down

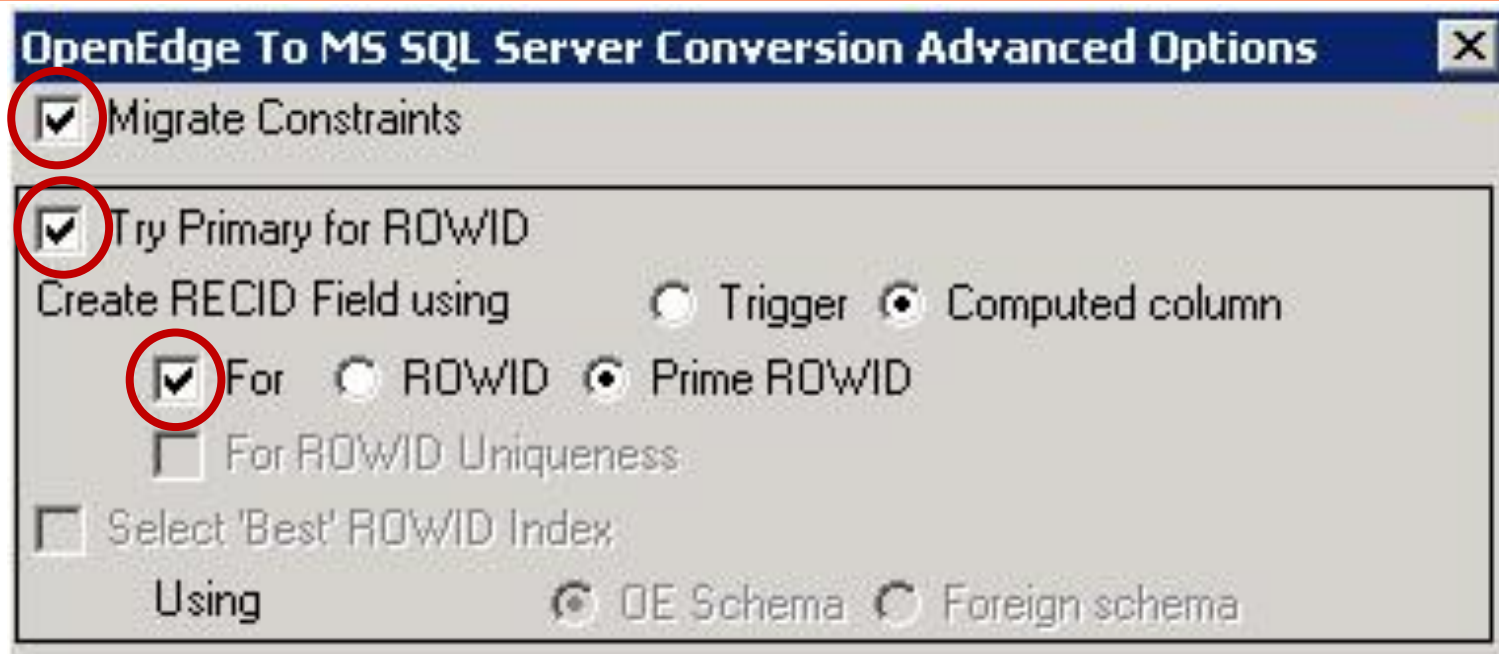
Any unique clustered will be assigned ROWID.

Only a single component binary integer supports RECID

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		∅ Bidx1	Bidx2
C			Cidx3	Cidx4	Cidx5		P_RECID

Unique
Mandatory
∅ Non-clustered

Rules: Triple Combination

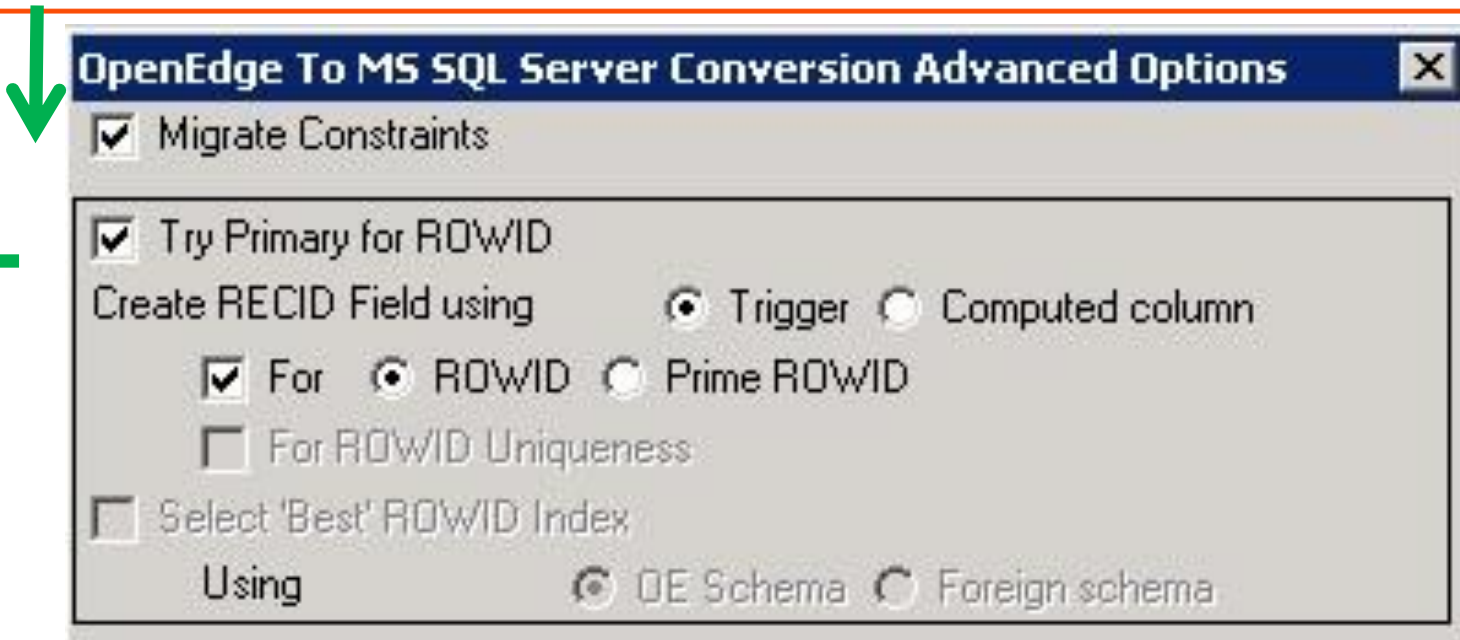


Rule:
 Precedence of options is top down
 Lower precedence option are unused if ROWID derivations were established at higher levels.

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		∅ Bidx1	Bidx2
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3

Unique
Mandatory
∅ Non-clustered

Rules: Combining “Migrate Constraints” with “Try Primary for ROWID”

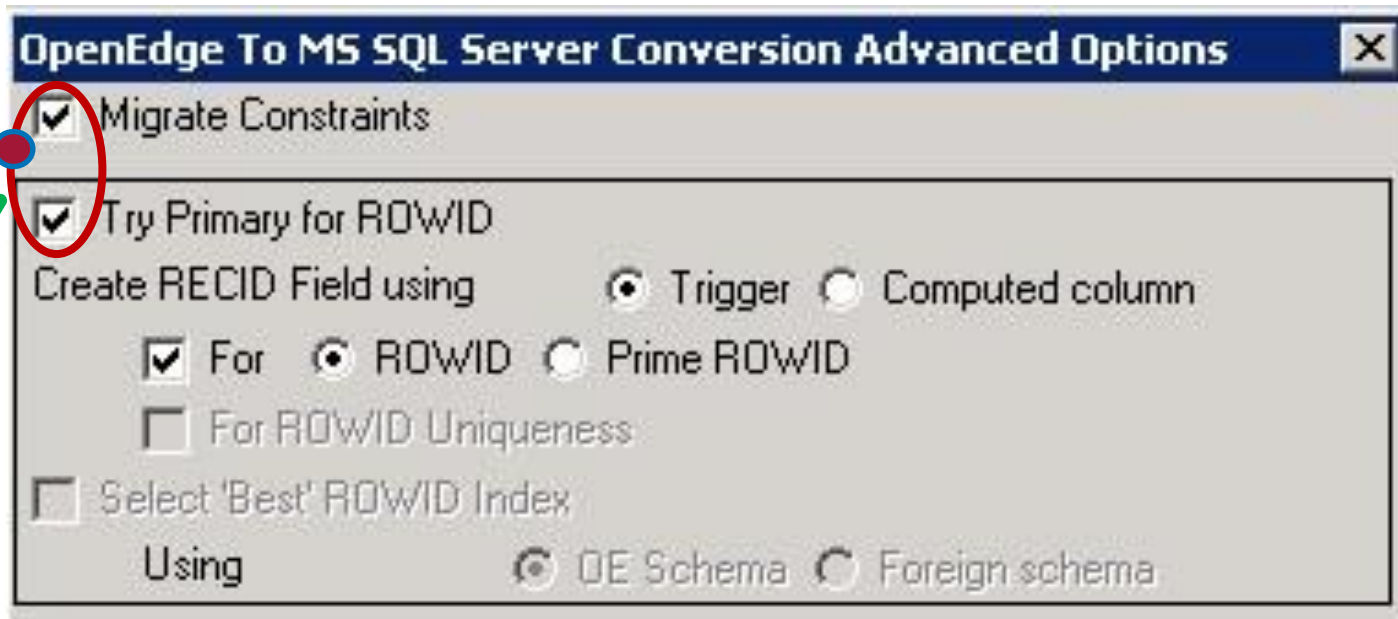


Rules:
Precedence of options is top down
When constraints are migrated, they are always explicit changes to the server
Constraint designations from other sources are implicit and/or derived

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		∅ Bidx1	Bidx2
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3

Unique
 Mandatory
 Non-clustered

Rules: Combining “Migrate Constraints” with “Try Primary for ROWID”



Rules:
Precedence of options is top down
When constraints are migrated, they are always explicit changes to the server
Constraint designations from other sources are implicit and/or derived

Table	Constraint Defs.		OpenEdge Indexes			MSS Attributes	
	Primary	Clustered	Primary	Other Indexes		Primary	Clustered
A	Aidx1		Aidx3	Aidx4	Aidx5	Aidx1	Aidx1
B	∅ Bidx1	Bidx2	Bidx3	Bidx4		∅ Bidx1	Bidx2
C			Cidx3	Cidx4	Cidx5	Cidx3	Cidx3

Unique Mandatory



PROGRESS