Consultingwerk
software architecture and development

True or false: any index will do

Peter Judge, Consultingwerk
peter.judge@consultingwerk.com

Peter Judge

Writing 4GL since 1996, working on a variety of frameworks and applications.

Active member of the OpenEdge community and speaker at international conferences

Focusing on integration with OpenEdge applications, PASOE, software architecture and web technologies

Modernization in Focus

Modernization of Legacy OpenEdge Applications

Deep Technical Expertise

Global IT Partner with Local Presence

More than Consulting – We Deliver Tools & Solutions

Agenda

- Why indexes?
- Simple index selection
- Logic trees with OR
- Temp-tables and other miscellanea

Indexes …. Why do we need 'em?

- Applications run faster
  - Fewer records are read
  - Fewer records sent to the client
  - Client does less – ideally no – additional filtering / selection
- Uniqueness enforcement
- Happier DBAs

Vibe coding is not your friend

Verify that you are using the index(es) you think you are using

- Static queries (FOR, FIND, OPEN QUERY)
  - COMPILE … XREF
- Dynamic queries
  - LOG-MANAGER QryInfo log-entry type

```
log-manager:logfile-name    = "c:/temp/wshop.log"
log-manager:logging-level   = 3.
log-manager:log-entry-types = "4GLTrace,4GLTrans,QryInfo".
```

- Tools like ProTop that read various VSTs

LOG-MANAGER QryInfo output

- For example

```
for each order no-lock where salesrep begins 'd'
```

```
Type: Dynamically Opened Query
PREPARE-STRING: for each order no-lock where salesrep begins 'd'
Prepared at Runtime
Client Sort: N
Scrolling: Y
Table: sports2000.Order
  Indexes: SalesRep
Times prepared: 1
Time to prepare (ms): 0
DB Blocks accessed:
  sports2000 : 366074
DB Reads:
  Table: sports2000.Order : 182913
  Index: Order.SalesRep : UNAVAILABLE
```

Index selection

> The Compiler constructs a logical tree from a query and evaluates both sides of each `AND` or `OR`, looking for index criteria. ABL counts equality, range, and sort matches (for `OR`) and uses them to select and bracket indexes.
>
> The precise rules are numerous and complex, and it is not important to fully understand their details.

https://docs.progress.com/bundle/openedge-abl-database-trigger-and-indexes/page/Background-and-terminology.html

Before we get to the rules…

- This only applies to ABL and not SQL
- Rules are applied in hierarchical order to filter indexes
  - This is important: Each rule is applied and the result is one or more ***remaining*** indexes
  - Use a worksheet approach to make the rules "easy" to apply

- The first 6 rules only apply to a subset of indexes
  - Compiler scans all fields in the query and selects all indexes that have leading components with those fields

Rules continue to be applied until there is only one index left

```
Table: Customer
      sRepStateCity          8         3 + SalesRep  + State + City
w     Comments               8         1 + Comments
      CountryPost            8         2 + Country   + PostalCode
pu    CustNum                8         1 + CustNum
      Name                   8         1 + Name
      SalesRep               8         1 + SalesRep
      sRepState              8         2 + SalesRep  + State
      CustNumUseless         8         2 + CustNum   + SalesRep

FOR EACH Customer WHERE State = 'Leinster' AND City = 'Dublin'

// The sSrepStateCity index is NOT eligible for selection
// NO indices are pre-selected for this query
```

those fields

Rules continue to be applied until there is only one index left

```
Table: Customer
      sRepStateCity           8         3 + SalesRep  + State + City
w     Comments                8         1 + Comments
      CountryPost             8         2 + Country   + PostalCode
pu    CustNum                 8         1 + CustNum
      Name                    8         1 + Name
      SalesRep                8         1 + SalesRep
      sRepState               8         2 + SalesRep  + State
      CustNumUseless          8         2 + CustNum   + SalesRep

FOR EACH Customer WHERE State = 'Leinster' AND SalesRep = 'XXX'

// The sSrepState index IS eligible for selection
```

those fields

Rules continue to be applied until there is only one index left

The Rules

For the first-pass set of indexes, filter using the following rules:

1. Pre-select only indexes with leading components in the where clause
2. If `CONTAINS` use word-index
3. Unique index with all components involved in the equality matches
4. Most active equality matches
5. Most active range matches
6. Most active sort matches

If multiple indexes remain, select one from

7. The primary index
8. First index alphabetically by name

The Rules

For the first-pass set of indexes, filter using the following rules:

1. Pre-select only indexes with leading components in the where clause
2. If `CONTAINS` use word-index
3. Unique index with all components involved in the equality matches
4. Most active equality matches
5. Most active range matches
6. Most active sort matches

>1

If multiple indexes remain, select one from

8. The primary index
9. First index alphabetically by name

Superset Selector
If there are indexes that will select supersets of records that are selected by other indexes, then we eliminate those.

⊇

Sort Selector Skip
Index records are already sorted by the index fields, so we don't need to reevaluate

order Table indexes

```
Flags Index Name    Cnt Field Name
u     CustOrder      2  + CustNum + Ordernum
      OrderDate      1  + OrderDate
pu    OrderNum       1  + Ordernum
      OrderStatus    1  + OrderStatus
      SalesRep       1  + SalesRep
w     SRepW          1  + SalesRep
      SRepDate       2  + SalesRep  + OrderDate
      DateSRep       2  + OrderDate + SalesRep
      SDateOstat     2  + ShipDate  + OrderStatus
```

for each order

Index	U		PU			W				
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat	
If "CONTAINS", use word-index	X	X	X	X	X		X	X	X	8
Unique index with complete equality match	X	X	X	X	X		X	X	X	
Most active equality matches	X	X	X	X	X		X	X	X	
Most active range matches	X	X	X	X	X		X	X	X	
Most active sort matches	X	X	X	X	X		X	X	X	
The primary index			✓							
First index by name										

`where salesrep = 'BBB'`

Index	U		PU			W			
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat
If “CONTAINS”, use word-index					X		X		
Unique index with complete equality match					X		X		
Most active equality matches					✓				
Most active range matches									
Most active sort matches									
The primary index									
First index by name									

2

`where salesrep = "DKP" and orderdate = 09/05/2011`

Index	U		PU			W				
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat	
If “CONTAINS”, use word-index		X			X		X	X		4
Unique index with complete equality match		X			X		X	X		
Most active equality matches							X	X		2
Most active range matches							X	X		
Most active sort matches							X	X		
The primary index										
First index by name								✓		

where salesrep = "DKP" and orderdate = 09/05/2011 by salesrep

Index	U		PU			W				
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat	
If “CONTAINS”, use word-index		X			X		X	X		4
Unique index with complete equality match		X			X		X	X		
Most active equality matches							X	X		2
Most active range matches							X	X		
Most active sort matches							X	X		
The primary index										
First index by name								✓		

```
where salesrep = "BBB" and orderStatus = "Shipped"
```

Index	U		PU			W				
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat	
If “CONTAINS”, use word-index				X	X		X			3
Unique index with complete equality match				X	X		X			
Most active equality matches				✓	✓					>1
Most active range matches										
Most active sort matches										
The primary index										
First index by name										

where salesrep >= "DKP" and orderdate >= 09/05/2011

Index	U		PU			W				
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat	
If “CONTAINS”, use word-index		X			X		X	X		4
Unique index with complete equality match		X			X		X	X		
Most active equality matches		X			X		X	X		
Most active range matches							X	X		
Most active sort matches							X	X		2
The primary index										
First index by name								✓		

`where salesrep >= "DKP" and orderdate >= 09/05/2011 by salesrep`

Index	U		PU			W				
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat	
If “CONTAINS”, use word-index		X			X		X	X		4
Unique index with complete equality match		X			X		X	X		
Most active equality matches		X			X		X	X		
Most active range matches					X		X	X		3
Most active sort matches					X		X			2
The primary index										
First index by name					✓					

where orderdate = 09/05/2011

Index	U		PU			W			
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat
If “CONTAINS”, use word-index		X						X	
Unique index with complete equality match		X						X	
Most active equality matches		X						X	
Most active range matches		X						X	
Most active sort matches		X						X	
The primary index									
First index by name								✓	

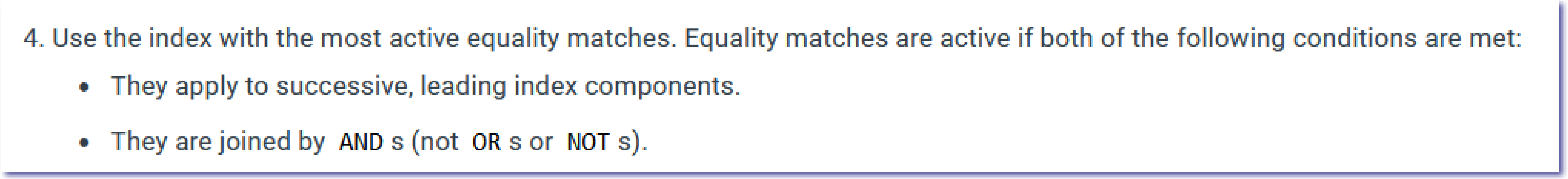
2

where orderdate = 09/05/2011 and shipdate = 10/01/2011

Index	U		PU			W				
Selection Rule	CustOrder	OrderDate	OrderNum	OrderStatus	SalesRep	sRepW	SRepDate	DateSRep	SDateOstat	
If "CONTAINS", use word-index		X						X	X	3
Unique index with complete equality match		X						X	X	
Most active equality matches		✓								
Most active range matches										
Most active sort matches										
The primary index										
First index by name										

Most active equality matches - considerations

- From *General rules for choosing a single index*



4. Use the index with the most active equality matches. Equality matches are active if both of the following conditions are met:
   - They apply to successive, leading index components.
   - They are joined by `AND` s (not `OR` s or `NOT` s).

- There appears to be a "hidden rule" related to the number of fields in an index that are matched – if 1 out of 1 field is matched, and 1 of 2 fields is matched, the index with the 1 / 1 field match is used
  - More proof that you need to check your assumptions

https://docs.progress.com/bundle/openedge-abl-database-trigger-and-indexes/page/General-rules-for-choosing-a-single-index.html

Most active equality matches - considerations

- This can really bite when dealing with parent-child relations

```
for each OrderLine where OrderNum = 42 and ItemNum = 11
```

- We really want this query to use the first 2 indexes, but only the *itemnum* index is used

```
Table: OrderLine

Flags Index Name                       Cnt Field Name
----- -------------------------------- --- ----------------------
      itemnum                          1 + Itemnum

pu    orderline                        2 + Ordernum + Linenum

      OrderLineStatus                  1 + OrderLineStatus
```

Breaking index selection

- Expressions break bracketing

```
FOR EACH order NO-LOCK WHERE MONTH(orderDate) = 1 ...
```

- `BEGINS` does NOT break bracketing
  - Considered a range bracket

```
FOR EACH order NO-LOCK WHERE salesRep BEGINS "D"
```

  - Uses the `order.salesRep` index

- `MATCHES` breaks bracketing

- When in doubt, test and verify

That's all … OR?

customer Table indexes

```
Indexes
Flags Index Name                    Cnt  Field Name
w     Comments               10     1 + Comments
      Country                9      1 + Country
      CountryPost            10     2 + Country
                                      + PostalCode
pu    CustNum                10     1 + CustNum
      Name                   10     1 + Name
      SalesRep               10     1 + SalesRep
      SrepCountryCLimit      9      3 + SalesRep
                                      + Country
                                      + CreditLimit
```

```
for each Customer where Name = "Acme Inc" or SalesRep = "BBB"
```

What indexes are used?

Consultingwerk
software architecture and development

```
for each Customer where Name = "Acme Inc" or SalesRep = "BBB"
```

What indexes are used?

- If you guessed **`Name`** , bravo.
- If you guessed **`SalesRep`** , bravo.
- Validate with `COMPILE ... XREF`

```
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 STRING "Acme Inc" 8 NONE TRANSLATABLE
custom.p 45 STRING "BBB" 5 NONE TRANSLATABLE
custom.p 45 SEARCH sports2000.Customer Name
custom.p 45 SEARCH sports2000.Customer SalesRep
```

```
for each Customer where Name = "Acme Inc" or Name = "RunRun
Roads"
```

What indexes are used?

```
for each Customer where Name = "Acem Inc" or Name = "RunRun Roads"
```

What indexes are used?

- If you guessed **Name** , bravo.

- Validate with `COMPILE ... XREF`

```
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 STRING "Acme Inc" 8 NONE TRANSLATABLE
custom.p 45 STRING "RunRun Roads" 5 NONE TRANSLATABLE
custom.p 45 SEARCH sports2000.Customer Name
custom.p 45 SEARCH sports2000.Customer Name
```

The OR operator makes a (huge) difference

Each side of an OR is its own, distinct index selection operation

```
for each customer where Name = "Acme Inc" or SalesRep = "BBB"
```

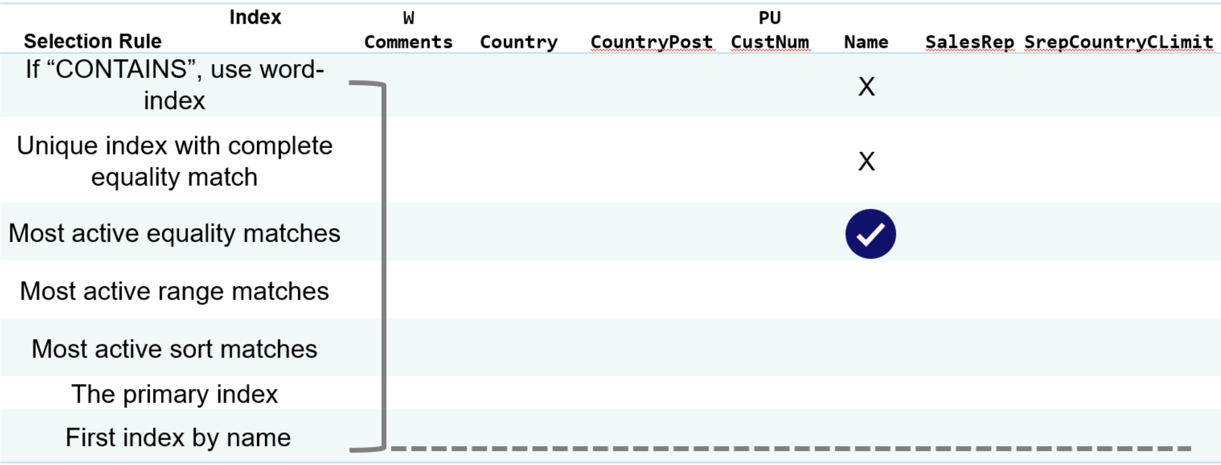
- 2 distinct index selection operations

```
for each customer where Name = "Acme Inc" or Name = "RunRun Roads"
```

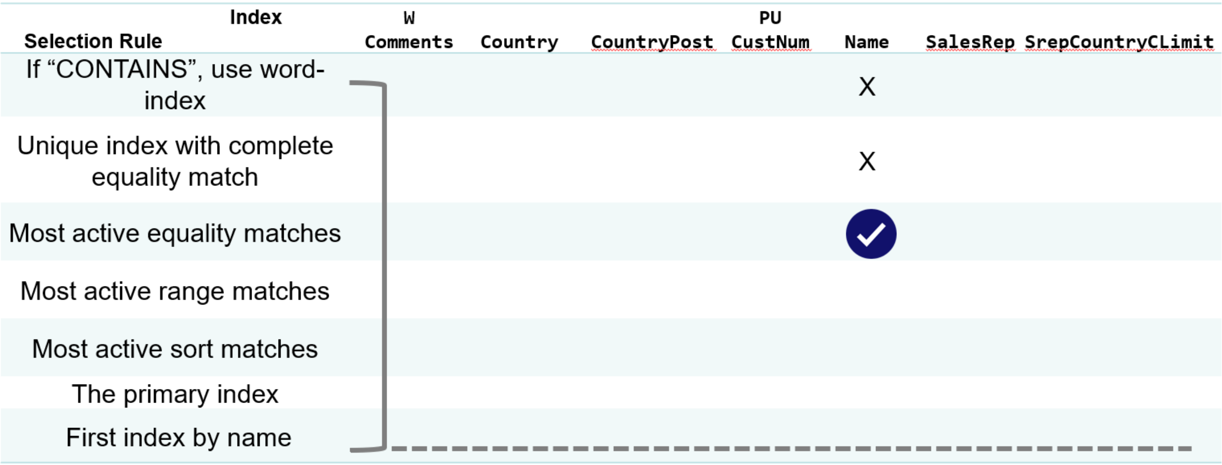
- This is *also* 2 distinct index selection operations

- A "logic tree" is built until we hit an **AND**. Break down the query until we hit **AND**s
- This implies that we can force multiple indexes to be used via **OR**s and parentheses
- Now we apply our rules to the **AND** part of the logic tree, as above

```
for each customer where
customer.Name = "Acme Inc"              or              customer.Name = "RunRun Roads"
```



Selection Rule	Index W Comments	Country	CountryPost	PU CustNum	Name	SalesRep	SrepCountryCLimit
If “CONTAINS”, use word-index					X		
Unique index with complete equality match					X		
Most active equality matches					✓		
Most active range matches							
Most active sort matches							
The primary index							
First index by name							



Selection Rule	Index W Comments	Country	CountryPost	PU CustNum	Name	SalesRep	SrepCountryCLimit
If “CONTAINS”, use word-index					X		
Unique index with complete equality match					X		
Most active equality matches					✓		
Most active range matches							
Most active sort matches							
The primary index							
First index by name							

```
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 STRING "Acme Inc" 8 NONE TRANSLATABLE
custom.p 45 STRING "RunRun Roads" 5 NONE TRANSLATABLE
custom.p 45 SEARCH sports2000.Customer Name
custom.p 45 SEARCH sports2000.Customer Name
```

```
for each Customer where Name = "Acme Inc" or Name = "RunRun
Roads" use-index
```

What indexes are used?

```
for each Customer where Name = "Acme Inc" or Name = "RunRun
Roads" use-index Name
```

What indexes are used?

- If you guessed **Name** , bravo.
- But only one index was used because we told it use one index
- Validate with `COMPILE ... XREF`

```
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 ACCESS sports2000.Customer Name
custom.p 45 STRING "Acme Inc" 8 NONE TRANSLATABLE
custom.p 45 STRING "RunRun Roads" 5 NONE TRANSLATABLE
custom.p 45 SEARCH sports2000.Customer Name WHOLE-INDEX
```

Complex OR with parentheses (1)

How many "simple clauses" are here?

```
for each Customer where
 ( Customer.Name = 'Acme Inc' or
    ( Customer.Country = 'AAA' or
        (  Customer.Name = 'Acme Inc' and Customer.Country = 'DEN')
    )
 )
or
( Customer.Salesrep = 'BBB' and  Customer.Salesrep = 'XXX' )
```

Complex OR with parentheses (1)

```
45 SEARCH sports2000.Customer Name
45 SEARCH sports2000.Customer Country
45 SEARCH sports2000.Customer Country
45 SEARCH sports2000.Customer Name
45 SEARCH sports2000.Customer Srep
```

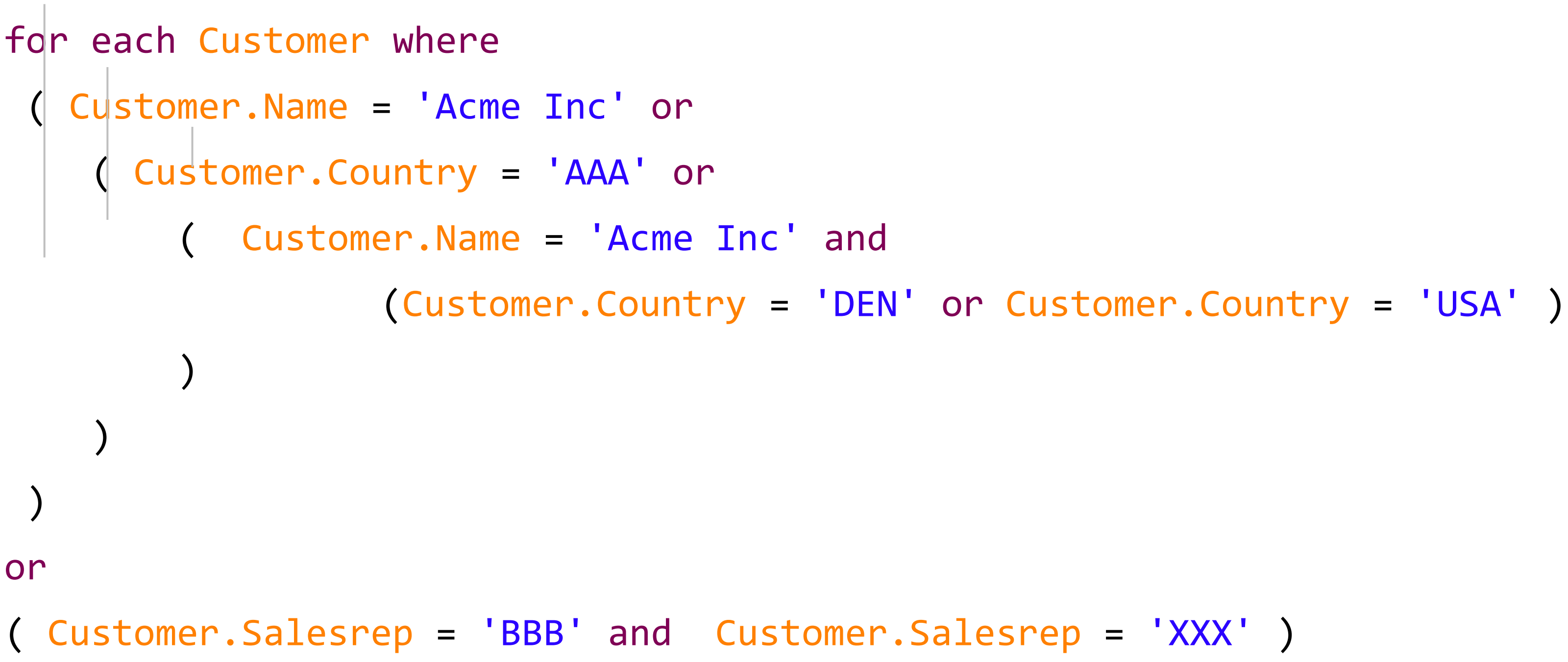
How many "simple clauses" are here?

```
for each Customer where
  ( Customer.Name = 'Acme Inc' or
    ( Customer.Country = 'AAA' or
        (  Customer.Name = 'Acme Inc' and Customer.Country = 'DEN')
    )
  )
or
( Customer.Salesrep = 'BBB' and  Customer.Salesrep = 'XXX' )
```

1	Name
2	Country
3, 4	Country, Name
5	Srep

Complex OR with parentheses (2)

How many "simple clauses" are here?



```
for each Customer where
 ( Customer.Name = 'Acme Inc' or
    ( Customer.Country = 'AAA' or
        (  Customer.Name = 'Acme Inc' and
                  (Customer.Country = 'DEN' or Customer.Country = 'USA' )
        )
    )
 )
or
( Customer.Salesrep = 'BBB' and  Customer.Salesrep = 'XXX' )
```

```
46 SEARCH sports2000.Customer Name
46 SEARCH sports2000.Customer Country
46 SEARCH sports2000.Customer Name
46 SEARCH sports2000.Customer SalesRep
```

Complex OR with parentheses (2)

How many "simple clauses" are here?

```
for each Customer where
 ( Customer.Name = 'Acme Inc' or
    ( Customer.Country = 'AAA' or
        (  Customer.Name = 'Acme Inc' and
                  (Customer.Country = 'DEN' or Customer.Country = 'USA' )
        )
    )
 )
or
( Customer.Salesrep = 'BBB' and  Customer.Salesrep = 'XXX' )
```

1 Name

2 Country

3 Name

4 Srep

But wait! There's more!

Indexes and unknown values

- Unique indexes enforce a constraint of a one value per index-field per table …

  BUT multiple unknown values in ABL are allowed

- If you want truly unique values, mark the field(s) in the index as mandatory.

- Index field(s) containing unknown values sort **higher** than any other value

```
create Customer.
assign Customer.custnum = 1
       Customer.name    = 'Peter'.

create Customer.
assign Customer.custnum = 2
       Customer.name    = 'Paul'.

create Customer.
assign Customer.custnum = ?
       Customer.name    = 'Mary'.


for each Customer by CustNum desc:
```

```
CustNum        Name
-------------- --------------
             ?  Mary
             2  Paul
             1  Peter
```

indexes and case sensitivity

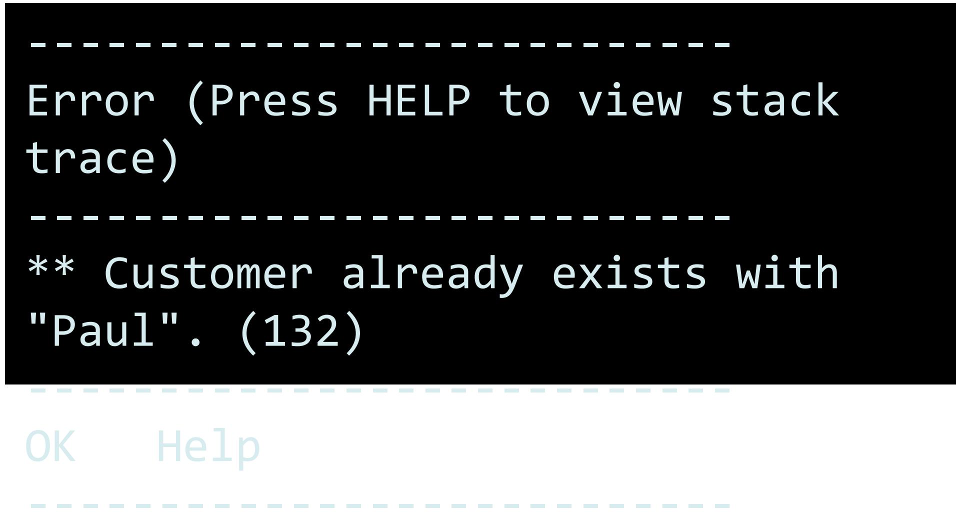
- By default, the AVM doesn't care about case for fields (whether indexed or not)
  - Field (data) values are stored as entered
  - Index field values are stored in UPPER CASE

```
"Paul" = "PAUL" = "paul"
```

- If the fields are case sensitive
  - Field (data) values are stored as entered
  - Index field values are stored as entered

```
"Paul" <> "PAUL" <> "paul"
```

  - Sorting on these fields may differ
  - Word indexed fields are always treated as case-insensitive



```
-----------------------------
Error (Press HELP to view stack
trace)
-----------------------------
** Customer already exists with
"Paul". (132)
-----------------------------
OK    Help
-----------------------------
```

Temp-table anomalies

- The first-alphabetical rule is replaced by a first-defined rule

- Fields cannot be marked as mandatory so uniqueness must be guaranteed by application code

Is the correct index always the right index?

- The compiler does not know how much data is in the DB nor how it is distributed across the tables you are querying

- The developer should have an idea of the data distribution. Just because you use the exact index on a very large table does not mean that inverting a query cannot run faster

The precise rules are numerous and complex, and it is not important to fully understand their details.

- Understanding the index "selection" rules is vital
  - There are subtleties and oddities though

- Get used to validating that your expected index is actually being used
  - COMPILE XREF
  - LOG-MANAGER
  - INDEX-INFORMATION()

- Use ABL Database Triggers and Indexes doc page contains the rules and other useful information
  - https://docs.progress.com/bundle/openedge-abl-database-trigger-and-indexes/page/Database-Index-Usage.html

Conferences in 2025

- ~~OpenEdge World-Tour/PUG UK & Ireland, March 19th/20th~~
- ~~PUG Lithuania in Vilnius, April 9th~~
- ~~OpenEdge World-Tour/PUG Germany in Cologne, May 13th/14th~~
- ~~PUG Baltics in Tallinn, May 15th~~
- ~~PUG France, June 3rd~~
- ~~PUG South Africa, Jo'burg and Cape Town, Sept 3,4 and 8, 9~~
- **PUG Challenge and OpenEdge World-Tour Boston, Sep 28th- Oct 1st**
- **PUG Challenge Valencia, November 5th-7th**

Questions

Consultingwerk

software architecture and development

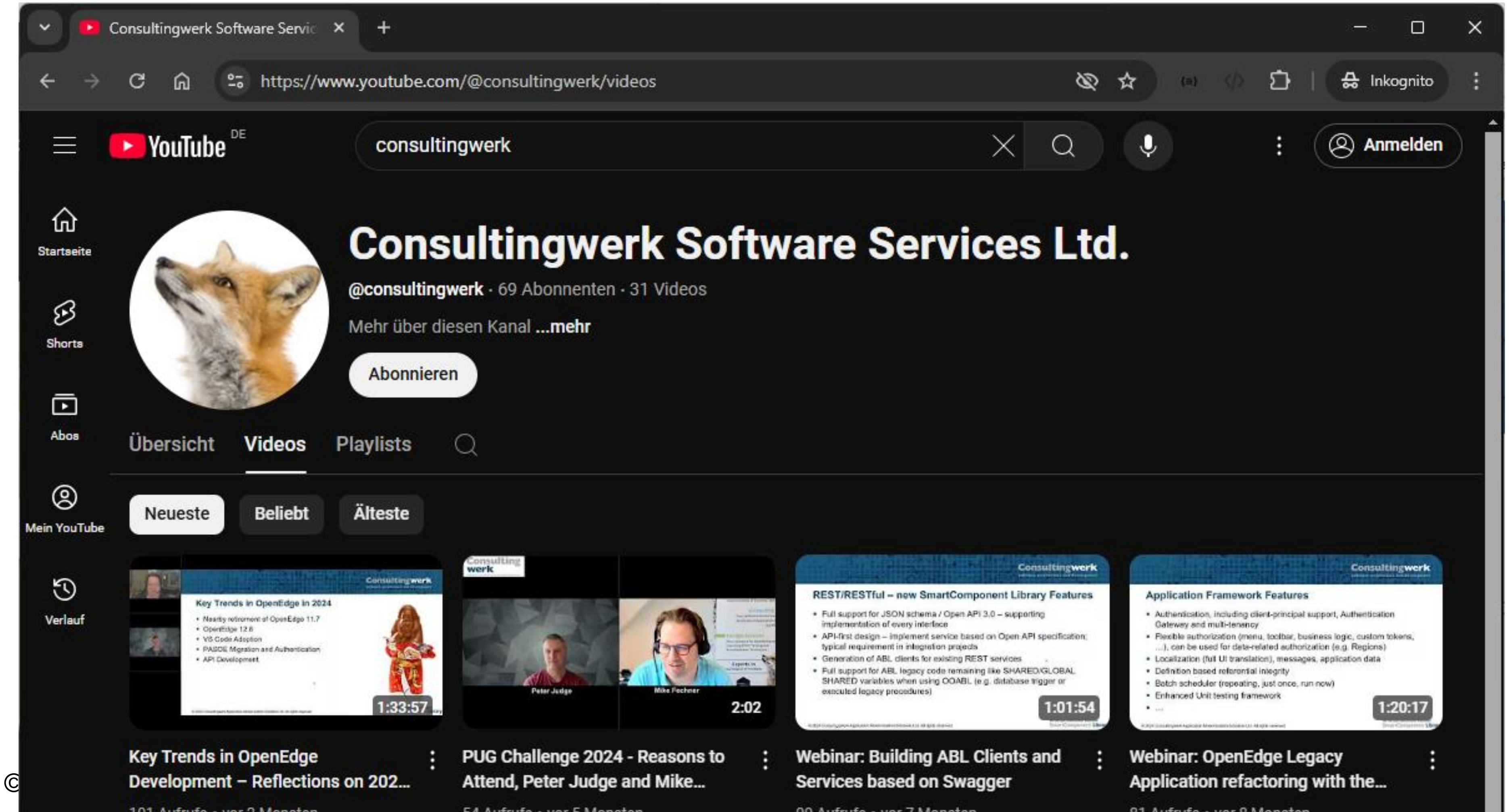
Consultingwerk
software architecture and development
SmartUpdate
Stay Tuned for the Latest News,
Product Updates, and Event Highlights!

Subscribe Now to Consultingwerk's Official Newsletter!

www.consultingwerk.com/newsletter

or marketing@consultingwerk.com

Visit our YouTube channel



Consultingwerk Software Servic

https://www.youtube.com/@consultingwerk/videos

Consultingwerk Software Services Ltd.

@consultingwerk · 69 Abonnenten · 31 Videos

Mehr über diesen Kanal ...mehr

Key Trends in OpenEdge in 2024

- Nearby retirement of OpenEdge 11.7
- OpenEdge 12.8
- VS Code Adoption
- PASOE Migration and Authentication
- API Development

1:33:57

Key Trends in OpenEdge Development – Reflections on 202...

101 Aufrufe · vor 2 Monaten

2:02

PUG Challenge 2024 - Reasons to Attend, Peter Judge and Mike...

54 Aufrufe · vor 5 Monaten

REST/RESTful – new SmartComponent Library Features

- Full support for JSON schema / Open API 3.0 – supporting implementation of every interface
- API-first design – implement service based on Open API specification; typical requirement in integration projects
- Generation of ABL clients for existing REST services
- Full support for ABL legacy code remaining like SHARED/GLOBAL SHARED variables when using OOABL (e.g. database trigger or executed legacy procedures)

1:01:54

Webinar: Building ABL Clients and Services based on Swagger

90 Aufrufe · vor 7 Monaten

Application Framework Features

- Authentication, including client-principal support, Authentication Gateway and multi-tenancy
- Flexible authorization (menu, toolbar, business logic, custom tokens, ...), can be used for data-related authorization (e.g. Regions)
- Localization (full UI translation), messages, application data
- Definition based referential integrity
- Batch scheduler (repeating, just once, run now)
- Enhanced Unit testing framework
- ...

1:20:17

Webinar: OpenEdge Legacy Application refactoring with the...

91 Aufrufe · vor 8 Monaten