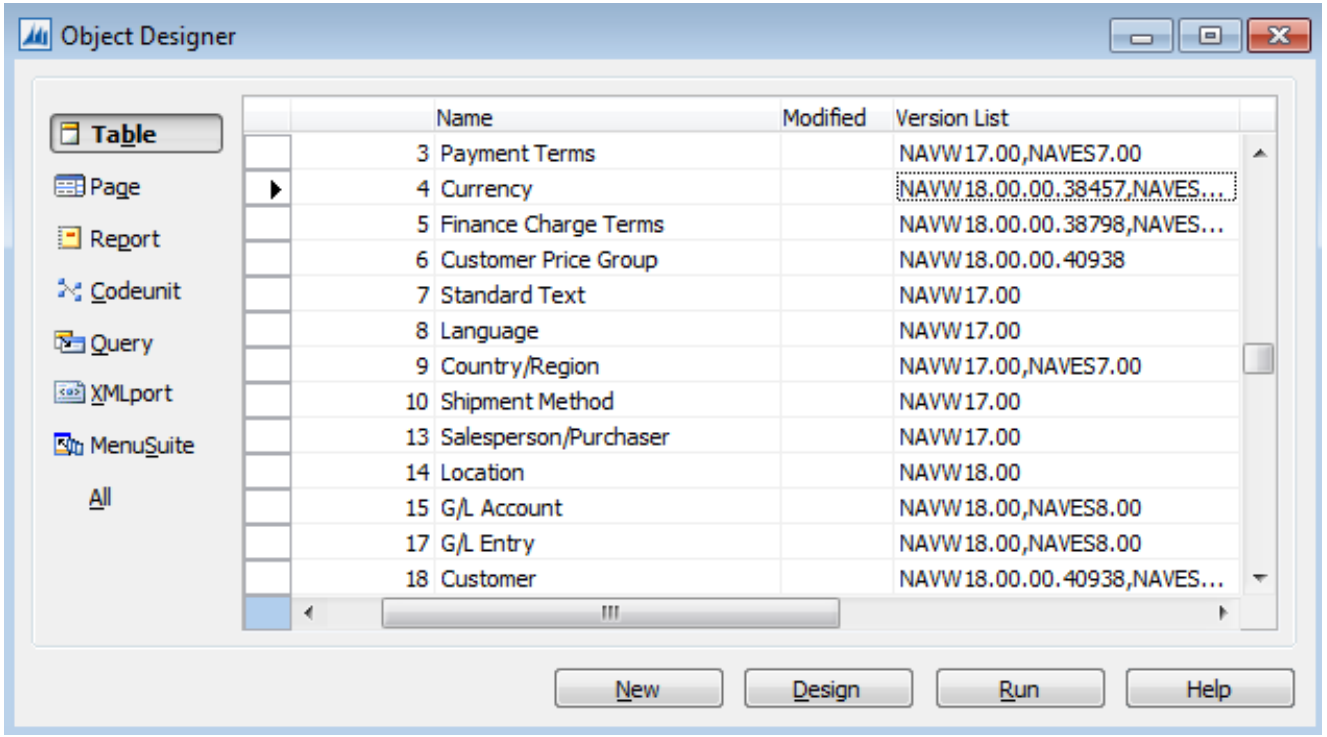
Esteve Terradas et

iris ekamat

# Introduction development

Microsoft Dynamics™

# Designing Application Objects



- Accés a taules
- Filtres
- SumIndexFields i FlowFieldsFields
- Transaccions
- Diàlegs

- **Accés a taules**
- Filtres
- SumIndexFields i FlowFieldsFields
- Transaccions
- Diàlegs

# Tipus de dades record

- Aquest tipus de dades complex correspon a una fila d'una taula . Cada registre consisteix en camps que formen les columnes de la taula.
- Explícit e implícit:
  - Explícit: Generat per usuari
  - Implícit: en disparadors (triggers) de taules. (Rec) registre actual i (xRec) registre anterior.

# Funció GET

- Us: localitza un registre amb un valor específic de la seva clau primària

```
[Ok :=] Record.GET([Value] ,...)
```

# Funció GET

- Exemple

| Variable name | DataType | Subtype |
|---|---|---|
| CustomerRec | Record | Customer |

| Text constant | ENU value |
|---|---|
| Text000 | The record was found. |
| Text001 | The record could not be found. |

# Funció GET

- Exemple

```
CustomerRec.GET('1120');
// This statement causes a run-time error if customer 1120 cannot be
// found. To avoid this, use the following construct:
IF CustomerRec.GET('1120') THEN
  MESSAGE(Text000)
ELSE
  MESSAGE(Text001);
```

# Funció SETCURRENTKEY

- Us: Activa una clau de la taula.

```
[Ok :=] Record.SETCURRENTKEY(Field1, [Field2],...)
```

# Funció SETCURRENTKEY

- Exemple

| Name | DataType | Subtype |
|---|---|---|
| MyCustomer | Record | Customer |

| Name | ConstValue |
|---|---|
| Text000 | The key was successfully selected. |
| Text001 | The key could not be found. |

# Funció SETCURRENTKEY

- Exemple

```
IF MyCustomer.SETCURRENTKEY(Name) THEN
  MESSAGE(Text000)
ELSE
  MESSAGE(Text001);
```

# Funció SETRANGE

Us: Establir un filtre simple sobre un camp per a definir un rang de registres.

```
Record.SETRANGE(Field [,FromValue] [,ToValue])
```

# Funció SETRANGE

- Exemple

| Variable name | DataType | Subtype |
|---|---|---|
| CustomerRec | Record | Customer |

```
CustomerRec.SETRANGE("No.", '100', '200');
```

# Funció SETFILTER

Us: Establir un filtre complex sobre un camp per a definir un rang de registres.

```
Record.SETFILTER(Field, String, [Value],...)
```

# Funció SETFILTER

- Exemple

| Name | DataType | Subtype |
|---|---|---|
| GLAccountRec | Record | G/L Account |

```
// Using a filter with replacement field.
// This filter selects all accounts in the range from 100 to 200
//    and No. 300.
GLAccountRec.SETFILTER("No.", '%1..%2|%3', '100', '200', '300');
// Using a filter entered directly in a string.
// This filter, which is entered as a string, has the same result as
//    the previous example.
//    This filter selects all accounts in the range from 100 to 200 and
//    and No. 300.
GLAccountRec.SETFILTER("No.", '100..200|300');
```

# Funció FIND

- Us: Fa a la variable apunti a un registre del rang.

```
Ok := Record.FIND([Which])
```

# Funció FIND

- Exemple

| Variable name | DataType | Subtype |
|---|---|---|
| ItemRec | Record | Item |

| Text constant | ENU value |
|---|---|
| Text000 | Item No. %1.\Description: %2. Price: $%3. |
| Text001 | The item was not found. |

# Funció FIND

- Exemple

```
ItemRec."No." := '1100';
IF ItemRec.FIND('=') THEN
  MESSAGE(TEXT000,  ItemRec."No.", ItemRec.Description, ItemRec."Unit Price")
ELSE
  MESSAGE(TEXT001);
```

# Funció NEXT

- Us: Fa a la variable avançar "Steps" registres endavant i apuntar a un nou.

```
Steps := Record.NEXT([Steps])
```

# Funció NEXT

- Exemple

| Variable name | DataType | Subtype |
|---|---|---|
| Count | Integer | Not applicable |
| CustomerRec | Record | Customer |

```
Count := 0;
IF CustomerRec.FIND('-') THEN
  REPEAT
    Count := Count + 1;
  UNTIL CustomerRec.NEXT = 0;
```

- Accés a taules
- **Filtres**
- SumIndexFields i FlowFieldsFields
- Transaccions
- Diàlegs

# Funcions per examinar filtres

```
// Write the following code in the OnRun trigger.
CustomerEntry.SETRANGE(Amount, -100, 100);
String := CustomerEntry.GETFILTER(Amount);
MESSAGE(Text000, String);
```

- Funció GETFILTER

Us: retorna en forma de cadena el filtre que esta actiu per al camp

Sintaxis: <String> := <Record>.GETFILTER(<Field>)

- Funció GETFILTERS

Us: retorna en forma de cadena tots els filtres que estan actius per a qualsevol camp

Sintaxis: <String> := <Record>.GETFILTERS

```
CustLedgerEntry.SETRANGE(Amount, -100, 100);
CustLedgerEntry.SETRANGE("Posting Date", 010108D, 123108D);
Str := CustLedgerEntry.GETFILTERS;
MESSAGE(Text000 + '%1', Str);
```

# Funcions per examinar filtres

```
CustomerRec.SETFILTER("No.",'200|300');
Val := CustomerRec.GETRANGEMIN("No.");
MESSAGE(Text000, Val);
```

- Funció GETRANGEMIN

Us: retorna el valor mínim d'un filtre (dona error si el filtre no te forma de interval continu)

Sintaxis: <Value> := Record>.GETRANGEMIN(<Field>)

- Función GETRANGEMAX

Us: retorna el valor máximo de un filtre (dona error si el filtre no te forma d'interval continuo)

Sintaxis: <Value> := Record>.GETRANGEMAX(<Field>)

```
CustomerRec.SETFILTER("No.",'100..200');
Val := CustomerRec.GETRANGEMAX("No.");
MESSAGE(Text000, Val);
```

# Funcions per copiar filtres

```
// Set filters on fields in a Customer record
Customer1.SETFILTER("No.", '<1000');
Customer1.SETRANGE("Credit Limit (LCY)", 10000, 20000);
Customer1.MARKEDONLY(TRUE);
// Apply filters to another record
Customer2.COPYFILTERS(Customer1);
Count := Customer2.COUNT;
```

- Funció COPYFILTER

Us: Aplica a un camp d'un registre el tiltre que esta definit en el mateix camp d'altre registre

- Sintaxis:<Record>.COPYFILTER(<FromField>,<ToRecord>.<ToField>)
- Funció COPYFILTERS
- Us: aplica als camps d'un registre els filtres que estan definits per un altre registre
- Sintaxis: <Record>.COPYFILTERS(<FromRecord>)

```
Customer.SETFILTER("No.", '<1000');
Customer.COPYFILTER("No.", Vendor."No.");
.
.
Count := Vendor.COUNT;
```

# Funció per marcar registres

| | |
|---|---|
| Text000 | Number of records before MARKEDONLY: %1.\ |
| Text001 | Number of records after MARKEDONLY: %2. |

```
CustomerRec.SETCURRENTKEY("No.");
CustomerRec."No." := '10000';
CustomerRec.FIND('=');
CustomerRec.MARK(TRUE); // Mark a record.
No1 := CustomerRec.COUNT;
CustomerRec.MARKEDONLY(TRUE);
No2 := CustomerRec.COUNT;
MESSAGE(Text000 + Text001, No1, No2);
```

- Funció MARKEDONLY
- Us: per a seleccionar només els registres que estan marcats per l'usuari o per el programa
- Sintaxis:

[<IsMarkedOnly>]:=<Record>.MARKEDONLY([<SetMarkedOnly>])

- Funció MARK
- Us: Per a marcar/desmarcar un registre o saber si un registro determinat està marcat
- Sintaxis: [IsMarked] := Record.MARK([SetMarked])

- Accés a taules
- Filtres
- **SumIndexFields i FlowFieldsFields**
- Transaccions
- Diàlegs

# Funcions SumIndexFields i FlowFields

- SumIndexFields
- Funció CALCSUMS
- Us: Retorna la suma per un (o variis) SumIndexField
- Sintaxis: [<Ok> :=] <Record>.CALCSUMS (<Field1>, [<Field2>,...])

```
"Cust. Ledger Entry".SETCURRENTKEY("Customer No.","Date");
"Cust. Ledger Entry".SETRANGE("Customer No.", 'AAA 1050');
"Cust. Ledger Entry".SETRANGE("Date", 010196D, 123196D);
"Cust. Ledger Entry".CALCSUMS("Amount");
```

# Funcions SumIndexFields i FlowFields

- FlowFields

| FlowField type | Field type | Description |
|---|---|---|
| **Sum** | Decimal | The sum of a specified set in a column in a table. |
| **Average** | Decimal | The average value of a specified set in a column in a table. |
| **Exist** | Boolean | Indicates whether any records exist in a specified set in a table. |
| **Count** | Integer | The number of records in a specified set in a table. |
| **Min** | Any | The minimum value in a column in a specified set in a table. |
| **Max** | Any | The maximum value in a column in a specified set in a table. |
| **Lookup** | Any | Looks up a value in a column in another table. |

# Funcions SumIndexFields i FlowFields

Customer (Table data)

| Customer | Name | Country/Region Code | Balance (FlowField) | Any Entries (FlowField) |
|---|---|---|---|---|
| 10000 | Windy City Solutions | US | 60 | Yes |
| 10010 | Modern Cars Inc. | US | 90 | Yes |
| 10020 | | FR | 210 | Yes |
| 10030 | | UK | 0 | No |
| 10040 | La Cuisine Française | FR | 300 | Yes |

Virtual part of the table data

Customer Entry (Table data)

| Customer | Date | Comment | Amount |
|---|---|---|---|
| 10000 | | | 10 |
| 10000 | | | 20 |
| 10000 | | | 30 |
| 10010 | | | |
| 10010 | | | 50 |
| 10020 | | | 60 |
| 10020 | | | 70 |
| 10020 | | | 80 |
| 10040 | | | 90 |
| | | | |
| 10040 | | | 110 |

- Accés a taules
- Filtres
- SumIndexFields i FlowFieldsFields
- **Transaccions**
- Diàlegs

# Transaccions

- Inicialización de registros
  - Función INIT
  - Sintaxis: <Record>.INIT
- Inserción de registros
  - Función INSERT.
  - Sintaxis: [<Ok> :=] <Record>.INSERT([<RunTrigger>])

```
// Scenario 1
CustomerRec.INIT;
CustomerRec."No." := '1120';
CustomerRec.INSERT;
// Scenario 2
CustomerRec."No." := '1120';
CustomerRec.INIT;
CustomerRec.INSERT;
```

# Transaccions

- Esborrar registres:
  - Funció DELETE
    - Sintaxis: [<Ok> :=] <Record>.DELETE([<RunTrigger>])
  - Función DELETEALL
    - Sintaxis: Record.DELETEALL([RunTrigger])

```
// This C/AL code:
WHILE CustomerRec.FIND('-') DO
  CustomerRec.DELETE;
// Performs the same operation as:
CustomerRec.DELETEALL;
```

# Transaccions

- Assignació de valors a camps normals
  - Sentencia de assignació:
    - Sintaxis: <Field> := <Value>
  - Funció VALIDATE
    - Sintaxis: <Record>.VALIDATE(<Field> [,<NewValue>])

```
GeneralLedgerEntry.VALIDATE("G/L AccountNo", '100');
// This corresponds to:
GeneralLedgerEntry."G/L AccountNo" := '100';
GeneralLedgerEntry.VALIDATE("G/L AccountNo");
```

# Transaccions

- Sentencia MODIFY
  - Sintaxis: [<Ok> :=] <Record>.MODIFY([<RunTrigger>])
- Sentencia MODIFYALL
  - Sintaxis: <Record>.MODIFYALL(<Field>,<NewValue> [,<RunTrigger>])
- Sentencia RENAME
  - Sintaxis: [<Ok>:= ]<Record>.RENAME(<Value1>, [<Value2>,...])

```
ItemRec.GET('1150');
ItemRec.RENAME('1105');
```

```
// The result of this statement:
CustomerRec.MODIFYALL("Statistics Group", 2);
// is equivalent to:
IF CustomerRec.FIND('-') THEN
  REPEAT
    CustomerRec."Statistics Group" := 2;
    CustomerRec.MODIFY;
  UNTIL CustomerRec.NEXT = 0;
```

# Transaccions

- Sentencia COMMIT

```
BeginWriteTransactions
(C/AL Statements) // Transaction 1
COMMIT
(C/AL Statements) // Transaction 2
EndWriteTransactions
```

- Accés a taules
- Filtres
- SumIndexFields i FlowFieldsFields
- Transaccions
- **Diàlegs**

# Diàlegs

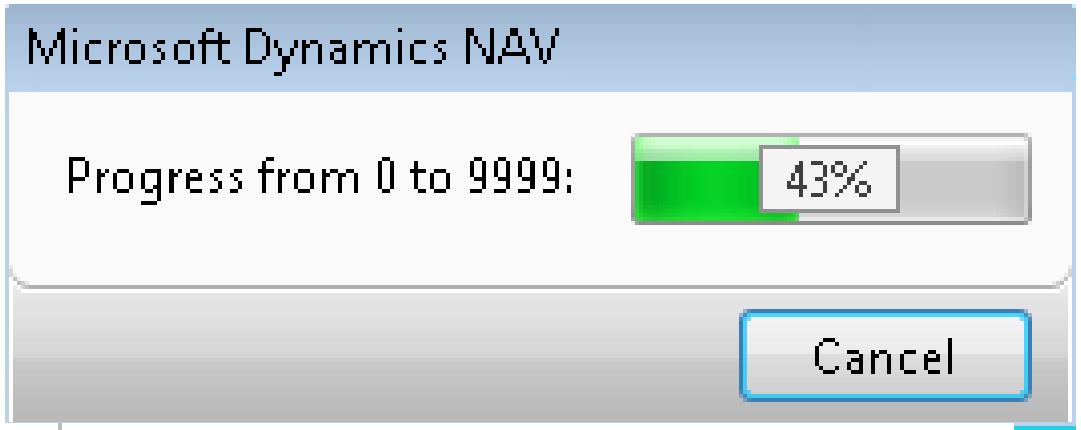
- Funció MESSAGE
  - Muestra un mensaje en pantalla.
  - MESSAGE(String [, Value1, ...])
- Funció CONFIRM
  - Muestra un mensaje en pantalla con opciones Ok y Cancel.
  - <Ok> := CONFIRM(<String>[,<Default>][,<Value1>,...])
- Funció STRMENU
  - Muestra una ventana de opciones.
  - <OptionNumber> := STRMENU(<OptionString>[,<DefaultNumber>])

# Diàlegs

- Funció OPEN
  - Abre una ventana de diálogo
  - <Dialog>.OPEN(<String>[,<Variable1>,…])
- Función UPDATE
  - Actualiza el valor mostrado en un campo de una ventana de diálogo
  - Sintaxis: <Dialog>.UPDATE([<Number>] [,<Value>])
- Función CLOSE
  - Cierra una ventana de dialogo
  - Sintaxis: <Dialog>.CLOSE

# Diàlegs

```
MyNext := 0;
MyDialog.OPEN(Text000,MyNext);
REPEAT
  // Do some processing.
  MyNext := MyNext + 1;
  MyDialog.UPDATE(); // Update the field in the dialog.
UNTIL MyNext = 9999;
SLEEP(1000);
MyDialog.CLOSE()
```




| | |
|---|---|
| Text000 | Progress from 0 to 9999 @1@@@@@ |

# Webgrafia i/o material

**Microsoft Dynamics NAV 2015**

- https://msdn.microsoft.com/en-us/library/dd355032%28v=nav.80%29.aspx