

# HOW TO create application

|  |    |
|--|----|
| Tables and indexes .....               | 2  |
| Tables .....                           | 2  |
| Indexes .....                          | 3  |
| Forms, create simple form .....        | 5  |
| Default.....                           | 5  |
| Edit.....                              | 6  |
| Generate .....                         | 7  |
| Forms, create second form.....         | 11 |
| Forms, create master/detail form ..... | 16 |
| Report.....                            | 20 |
| Report with group and summarize.....   | 22 |
| Menu of application .....              | 24 |
| Resume.....                            | 27 |

Ovo je praktično uputstvo kako napraviti aplikaciju KUĆNI BUDŽET

Podaci se nalaze u četiri tabele

Aplikacija bi trebalo imati dve/tri forme i dva izveštaja

Ovde će biti pokazano kako uraditi osnovne stvari koje su neophodne da se napravi aplikacija, biće napravljen rečnik podataka, tri forme, dva izveštaja i za sve to napraviti meni

This is a practical guide HOW TO make an application to the HOUSEHOLD BUDGET

The data contained in the four tables

The application should have two forms and two reports

Here you will learn how to do basic things that are necessary to make the application will be made data dictionary, the three forms, the two reports and for all to make me

## **Tabele i indeksi**

Osnova svake aplikacije us podaci

Tabele koje su potrebne aplikaciji su:  
COST – troškovi  
PAYMENT – plaćanja  
REPORT i REPORT2 – podaci za izveštaj

Indeksi su  
Za COST : COST i COSTN  
Za PAYMENT : PAYMENT  
Za REPORT : REPORT  
Za REPORT2 : REPORT2

### ***Tabele***

Pokrenite program HMGCASE

Pogledajte Tables->View i tu biste trebali videti strukture svih tabela

Ako je sve kao na slici izaberite Tables->Generate

## **Tables and indexes**

The basis of each application are data

Tables that are required applications are:  
COST - Costs  
PAYMENT - Payment  
REPORT and REPORT2 - report data

Indexes are  
For COST: COST and COSTN  
For PAYMENT: PAYMENT  
For REPORT: REPORT  
For REPORT2 : REPORT2

### ***Tables***

Start the HMGCASE

See Tables-> View and there you should see the structure of table

If all like the picture, select Tables->Generate

The screenshot shows the 'DBF tables structure' window in HMG Case Utility. It displays a list of tables and their fields. The table has columns: DBF name, Seq, Name, Type, Len, and Dec.

| DBF name | Seq | Name     | Type | Len | Dec |
|----------|-----|----------|------|-----|-----|
| COST     | 1   | PAY      | N    | 3   | 0   |
| COST     | 2   | DESCRIPT | C    | 20  | 0   |
| PAYMENT  | 1   | DATE1    | D    | 8   | 0   |
| PAYMENT  | 2   | PAY_ID   | N    | 3   | 0   |
| PAYMENT  | 3   | DESCRIPT | C    | 20  | 0   |
| PAYMENT  | 4   | EXPENSE  | N    | 10  | 2   |
| PAYMENT  | 5   | PAYMENT  | N    | 10  | 2   |
| PAYMENT  | 6   | DATE2    | D    | 8   | 0   |
| REPORT   | 1   | DATE1    | D    | 8   | 0   |
| REPORT   | 2   | PAY_ID   | N    | 3   | 0   |
| REPORT   | 3   | DESCRIPT | C    | 20  | 0   |
| REPORT   | 4   | EXPENSE  | N    | 10  | 2   |
| REPORT   | 5   | PAYMENT  | N    | 10  | 2   |
| REPORT   | 6   | DATE2    | D    | 8   | 0   |
| REPORT   | 7   | SALDO    | N    | 10  | 2   |
| REPORT   | 8   | PAYDESC  | C    | 20  | 0   |

Keyboard status: NumLock CapsLock Insert 14.05.2014 15:03

## Indeksi

Pogledajte Index->Edit i videćete indekse

Tabele su dobro učitane, indekse ćete morati malo doterati, samo ispravite nazive tabela

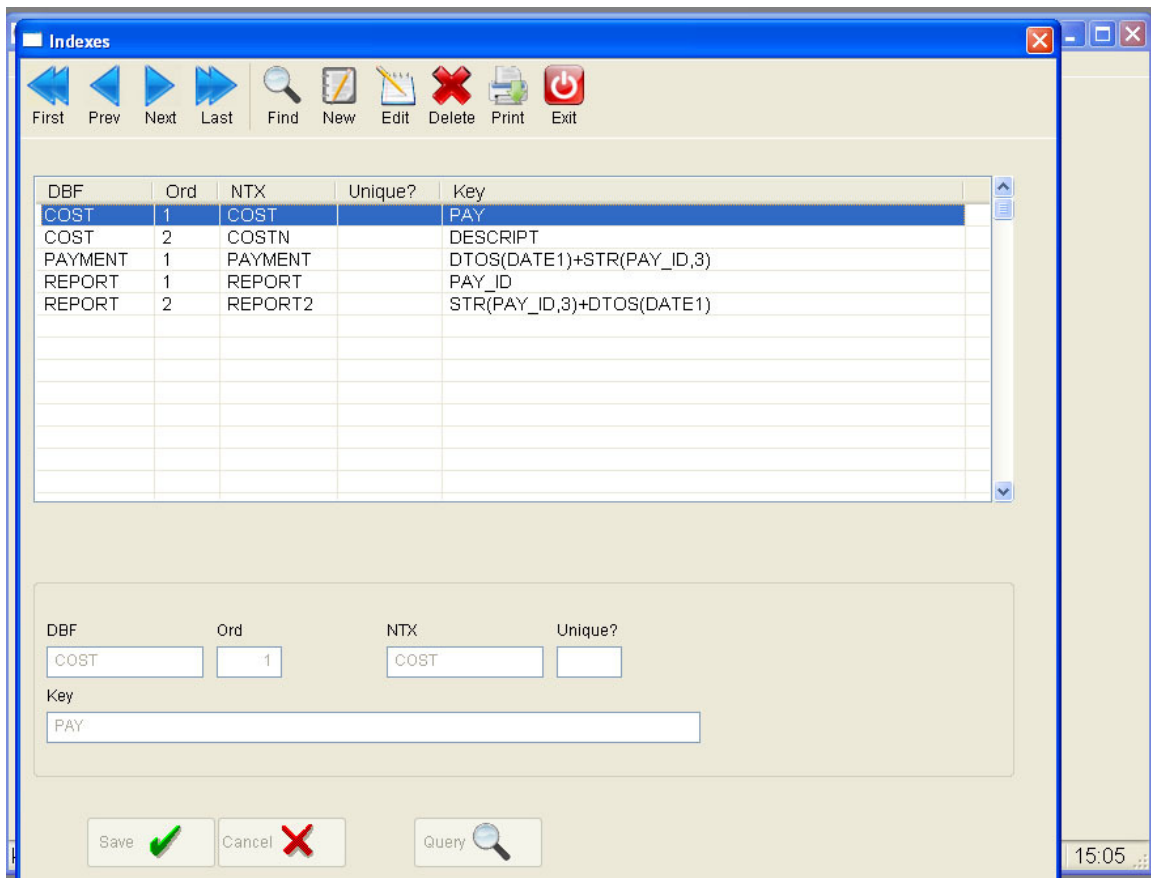
Izmena je jednostavna kliknite na red koji treba da ispravite ili pritisnite ENTER i dobićete mogućnost da ispravite podatke, konkretno treba popraviti polje DBF

## Indexes

See Index-> Edit and you will see the index

Tables are well loaded, the index will need to add some flavor, just correct table names

Editing is a simple click on the line that should be corrected or press ENTER and you will be able to correct the data, specifically to repair DBF field



Kada to uradite izaberite Index->Generate

When you do select Index-> Generate

Ako izađete iz aplikacije videćete da su kreirana 3 programa, **OPEN\_DBF**, **OPEN\_NTX** i **USE\_DBF**

If you exit the application, you will see that they created three programs, **OPEN\_DBF**, **OPEN\_NTX** and **USE\_DBF**

Videćete da ima nekoliko DBF tabela koje počinju sa „\_“, to su „sistemske“ i nemojte ih dirati

You will see that there are several DBF tables that start with "\_", there are "system tables" and do not touch them

## Forms, create simple form

Uobičajeno je da se za unos podataka u neku tabelu napravi program koji služi za unos, izmenu i brisanje.

Ovde imamo dve tabele od kojih je jedna **COST.DBF** sa podacima o troškovima

Da bismo napravili program za unos potrebno je uradimo

1. izaberemo tabelu i napravimo default formu
2. izmenimo ono što želimo
3. generišemo program
4. kompiliramo program

Pokrenite program

### ***Default***

Izaberite Forms->Default

It is common to input data into a table create a program that is used to enter, modify and delete.

Here we have two tables, one of which is **COST.DBF** data on costs

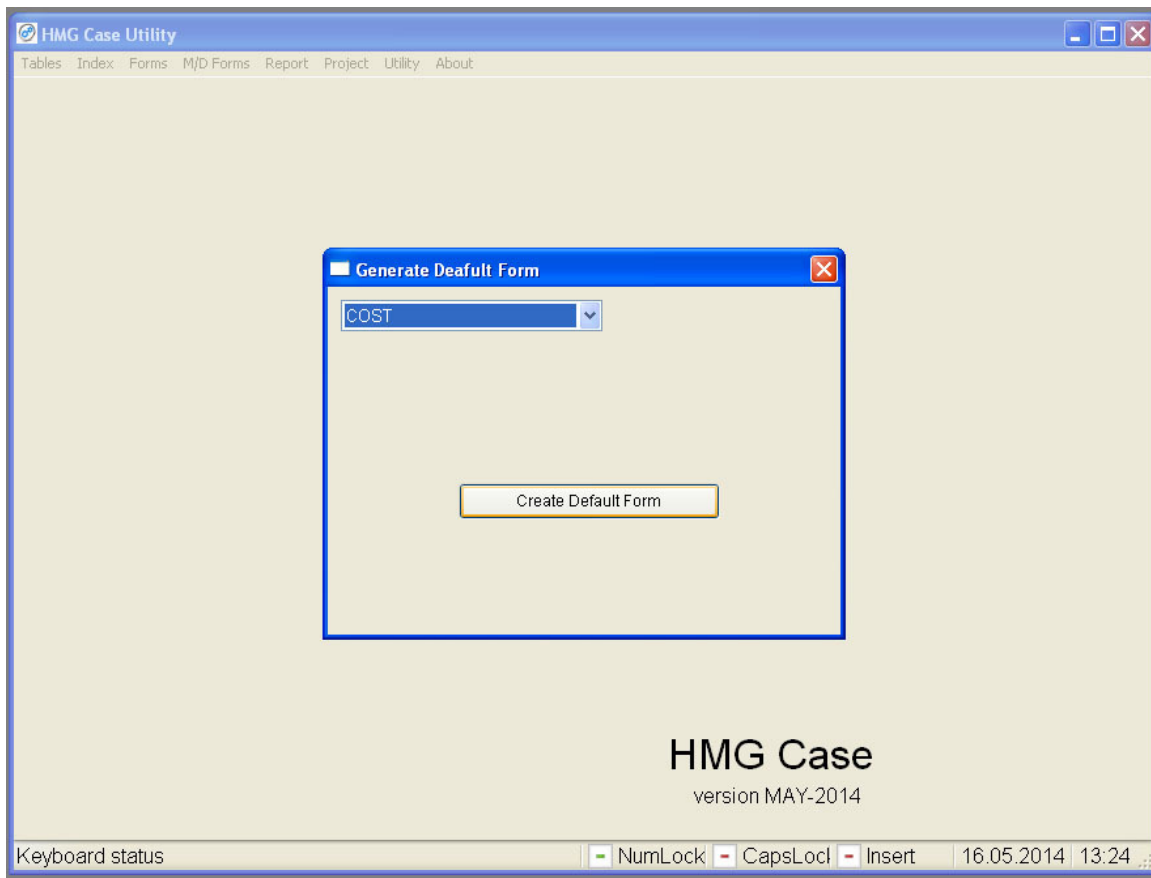
To create a data entry program needs to do

1. select the table and make a default form
2. change that we want
3. generate program
4. compile program

Run the program

### ***Default***

Select Forms-> Default



Kada ovo uradite sledeći korak je izabrato  
Forms->Edit i dobićete ekran kao na slici

***Edit***

When you do this the next step is to choose  
Forms-> Edit and you will get a screen as  
shown in Figure

***Edit***

[illegible]

Pošto je ovo primer ne morate ništa da menjate, samo napustite

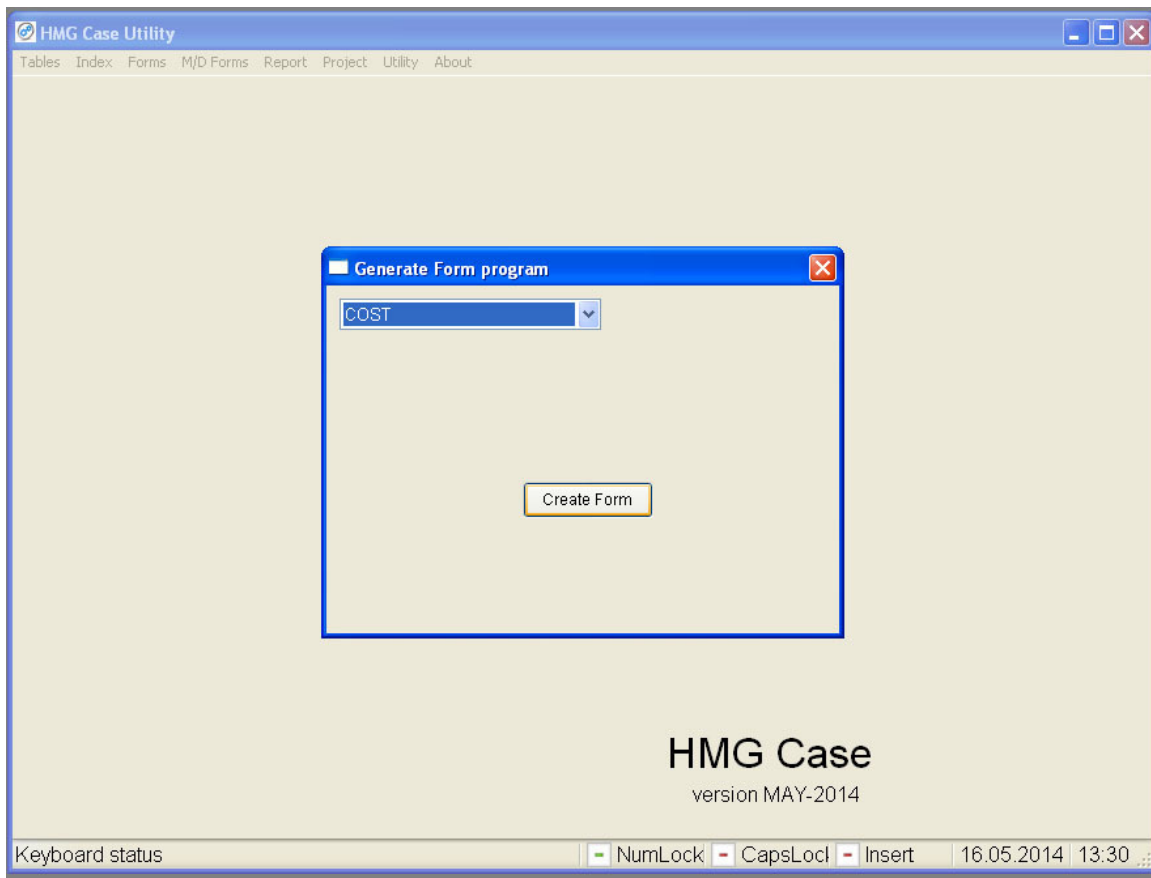
## Generate

Sledeći korak je izabrati Forms->Generate

Since this example does not need to change anything, just leave

## Generate

The next step is to choose Forms->Generate



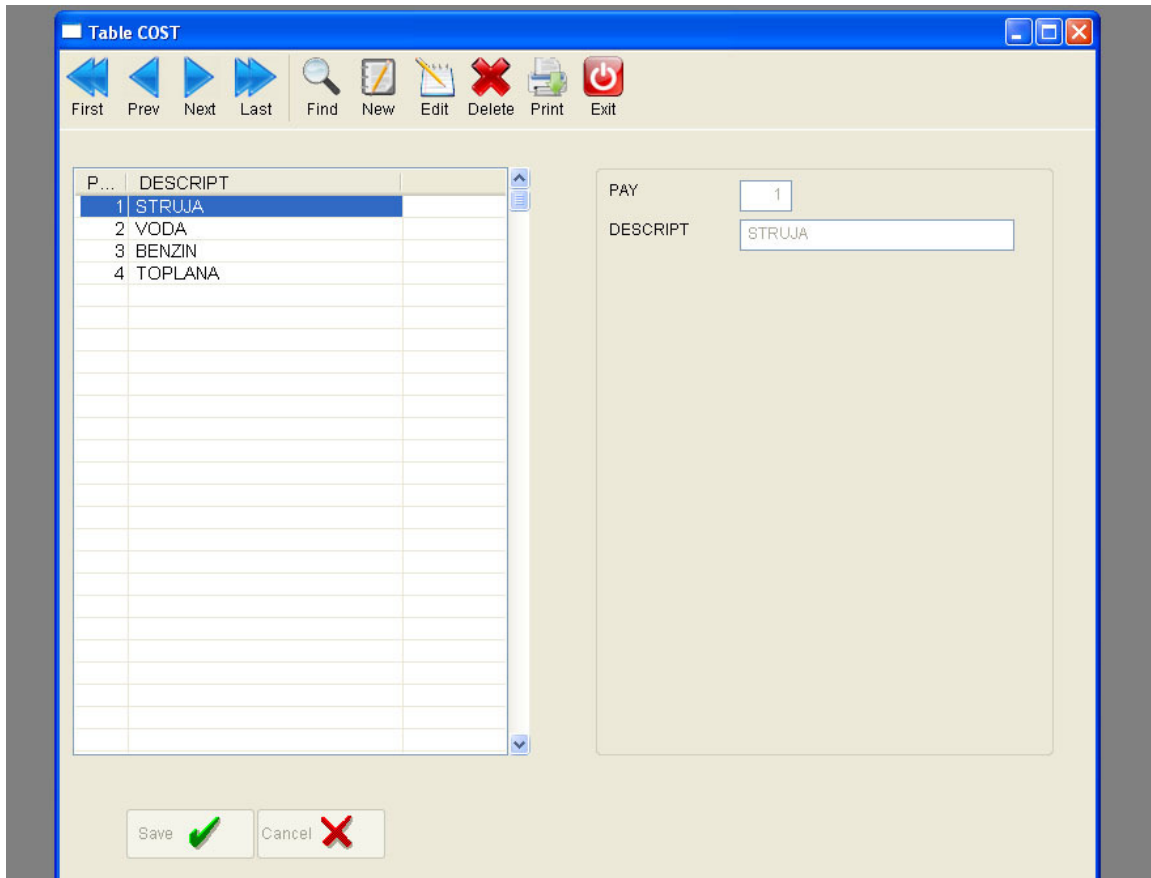
Kliknite na Create Form i sledeća poruka je **Generate edit\_1.prg !**

Izađite iz programa na radnom folderu gde je program otkucajte *build edit\_1* , sačekajte biće pokrenut program koji bi trebao izgledati kao na slici

Click the Create Form and the following message is a **Generate edit\_1.prg !**

Exit to the working directory where the program type *build edit\_1* wait will be running a program should look like this





Program je potpuno funkcionalan

The program is fully functional

Napomena: ovo je primer koji bi trebalo doraditi

Note: this is an example that should be further work

Vrednost polja PAY bi trebalo da bude ključna, bez ponavljanja.

PAY field value should be the key, without repetition.

Proceduru **NewRecord\_nnnn** promenite ovako ili na neki drugi način da obezbedite jedinstvenu vrednost polja

Procedure **NewRecord\_nnnn** change one way or the other way to provide a unique value field

```
PROCEDURE NewRecord_nnnn
```

```
Win_1.StatusBar.Item(1) := "Editing"

SET ORDER TO 1
dbGoTop()
dbGoBottom()
_next = pay + 1

Win_1.mPAY.Value          := _next
Win_1.mDESCRIPT.Value    := space(20)
```

```
EnableField_1484()  
  
Win_1.mDESCRIPT.SetFocus  
  
RETURN
```

Napravite kompilaciju i probajte. Trebalo bi da prilikom unosa nove šifre dobijete sledeći broj, unestite samo naziv

Kada je to u redu, kopirajte program u **ED\_COST.PRG** i malo ga izmenite jer je ovakav napravljen da se startuje kao glavni

Promenite ime procedure MAIN u EDIT\_COST

Izbišite nekoliko redova na početku programa koji se završavanju sa **// for test**

Make a compilation and try. You should enter a new payment when you get the next number, just enter a name

When it's right, copy the program to **ED\_COST.PRG** and little modify it because this was made to start a main

Change the name of the procedure MAIN in EDIT\_COST

Then delete few lines at the beginning of the program which is ending with a **// for test**

## Forms, create second form

Sledeća forma bi trebalo biti za unos podataka u tabelu **PAYMENT**

Kreirajte default formu za tabelu **PAYMENT**

Izberite Forms->Default

Izaberite tabelu **PAYMENT** i kliknite na **Create Default Form**

Sledeće je da je doterate na Forms->Edit i izabrite tabelu **PAYMENT**

To bi trebalo da izgleda ovako

The following form should be to enter data into a table **PAYMENT**

Create a default form for the table **PAYMENT**

Choose Forms->Default

Select the **PAYMENT** table and click on the Create Default Form

The following is the tweak to Forms-> Edit and chose a table **PAYMENT**

It should look like this

| Form    | DBF     | No | Field Name |
|---------|---------|----|------------|
| PAYMENT | PAYMENT | 1  | DATE1      |
| PAYMENT | PAYMENT | 2  | PAY_ID     |
| PAYMENT | PAYMENT | 3  | DESCRIPT   |
| PAYMENT | PAYMENT | 4  | EXPENSE    |
| PAYMENT | PAYMENT | 5  | PAYMENT    |
| PAYMENT | PAYMENT | 6  | DATE2      |

Form: COST  
DBF: COST  
No: 1  
Field Name: PAY  
Label: PAY  
Field Type: N  
Field Len: 3  
Field Dec:   
Row: 100  
Col: 520  
Picture: 999  
Default: 0

☒ Browse ☒ Display ☐ DBF  
☒ Input ☒ Edit ☐ Key  
☒ Query ☒ Print ☐ Valid  
☐ Display ☐ Validate ☐ Display

Save ☒ Cancel ☒ Query ☒ Preview form

Polja možete rasporediti kako želite, pozicionirate se na red i pritskom na ENTER ili dvoklik i imate mogućnost da

The fields you can arrange how you want to position the red and by pressing the ENTER key or double-clicking, and you

promenite attribute polja, kad nešto promenite obavezno završite klikom na **SAVE** da bi bilo sačuvano, a možete i odustati od promene klikom na **CANCEL**, dugme **PREVIEW FORM** ćete videti kako će biti raspoređena na ekranu

Imate mogućnost da postavite kontrole, atribut **Validate**. Tu imate mogućnost da kontrolišete unos u polje i mora imati kao vrednost polja iz tabele

Vežba: dodati polje u koje će se upisivati opis ključnog polja

Kliknite na **NEW** pa na **SAVE**, videćete da je formirano polje **DISPLAY**, *CHAR* dužine 10, atribut **Display** (samo za prikaz)

Vežba: napravite kontrolu na polju **PAY\_ID** da može imati vrednost kao polje **PAY** iz tabele **COST** i opis stavite u polje **DISPLAY**

To bi sad trebalo izgledati kao na slici

can change attributes field, when something changes required are finished, click on **SAVE** to be saved, and you can cancel the changes, click **Cancel**, click **PREVIEW FORM** you will see how they will be distributed on the screen

You have the ability to set the control attribute **Validate**. There you have the ability to control entry into the field and have a field value from a table

Exercise: add a field that will be saved ključnog description fields

Click on **NEW** then on **SAVE**, you will see that it has formed the **DISPLAY** field, *CHAR* length 10, **Display** attribute (for display only)

Exercise: Create a control on the field **PAY\_ID** that may have value as an array **PAY** from the table **COST** and put the description in the **DISPLAY**

It should now look like this

Forms

First Prev Next Last Find New Edit Delete Print Exit

PAYMENT

| Form    | DBF     | No | Field Name |
|---------|---------|----|------------|
| PAYMENT | PAYMENT | 1  | DATE1      |
| PAYMENT | PAYMENT | 2  | PAY_ID     |
| PAYMENT | PAYMENT | 3  | DESCRIPT   |
| PAYMENT | PAYMENT | 4  | EXPENSE    |
| PAYMENT | PAYMENT | 5  | PAYMENT    |
| PAYMENT | PAYMENT | 6  | DATE2      |
| PAYMENT | \$      | 99 | DISPLAY    |

Form: PAYMENT

DBF: PAYMENT

No: 2

Field Name: PAY\_ID

Label: PAY\_ID

Field Type: N

Field Len: 3

Field Dec: 0

Row: 130

Col: 520

Picture: 999

Default: 0

☒ Browse ☒ Display ☒ Edit ☒ Print ☒ Validate

☐ Input ☐ Key ☐ Valid ☐ Display

DBF: COST

Key: PAY

Valid: DESCRIPT

Display: DISPLAY

Save Cancel Query Preview form

13:26

Napravite Forms->Generate i biće kreiran program **EDIT\_n.PRG**

Create Forms-> Generate and will be created program **EDIT\_n.PRG**

Ukucajte **built EDIT\_n.PRG**

Type **built EDIT\_n.PRG**

Kada se program kompilira i startuje izgledaće kao na slici

When the program compiles and starts to look like the picture



PROCEDURE PaintDisplay\_nnnn

```
@ 440, 10 FRAME Frame_2 WIDTH 750 HEIGHT 120

@ 450, 20 LABEL Label_1 VALUE "DATUM"
@ 450, 110 LABEL Label_2 VALUE "PAY"
@ 450, 200 LABEL Label_3 VALUE "DESCRIPTION"
@ 450, 380 LABEL Label_4 VALUE "EXPENSE"
@ 450, 490 LABEL Label_5 VALUE "PAYMENT"
@ 450, 600 LABEL Label_6 VALUE "DATE2"
*@ 280, 420 LABEL Label_7 VALUE "DISPLAY"

@ 470, 20 TEXTBOX mDATE1 WIDTH 80 DATE
@ 470, 110 TEXTBOX mPAY_ID WIDTH 40 NUMERIC INPUTMASK "999" on
enter valid_2_2("E") on lostfocus valid_2_2("F")
@ 500, 110 TEXTBOX dDISPLAY WIDTH 150
@ 470, 200 TEXTBOX mDESCRIPT WIDTH 170 INPUTMASK
"!!!!!!!!!!!!!!!!!!!!!!"
@ 470, 380 TEXTBOX mEXPENSE WIDTH 100 NUMERIC INPUTMASK
"999,999.99"
@ 470, 490 TEXTBOX mPAYMENT WIDTH 100 NUMERIC INPUTMASK
"999,999.99"
@ 470, 600 TEXTBOX mDATE2 DATE
```

Promenite proceduru MAIN u  
Edit\_payment

Change in procedure MAIN in  
Edit\_payment

Obrišite nekoliko redova na početku koji se  
završavaju sa **// for test**

Erase a few lines at the beginning of the  
end with **// for test**

Snimite program kao **ED\_PAY.PRG**

Save program as **ED\_PAY.PRG**

## Forms, create master/detail form

Ovaj program će vam omogućiti i mogućnost kreiranja forme za rad sa dve tabele vezane ključnim poljima

This program will allow you the ability to create forms to work with two tables related to key areas

Za početak napravite default master/detail formu

Start by making a default master/detail form

Izaberite M/D Forms->Default i dobićete ekran kao na slici

Select M/D Forms-> Default and get the screen as shown in Figure

**Master** tabela je prva i biće prikazan jedan slog

**Master** table first and be shown a record

**Detail** tabela je druga i biće prikazano više slogova

**Detail** table is different and will be shown more syllables

**Relation** su polja koja su zajednička i povezuju tabele, može ih biti više

**Relation** fields that are common and associated tables, there may be more

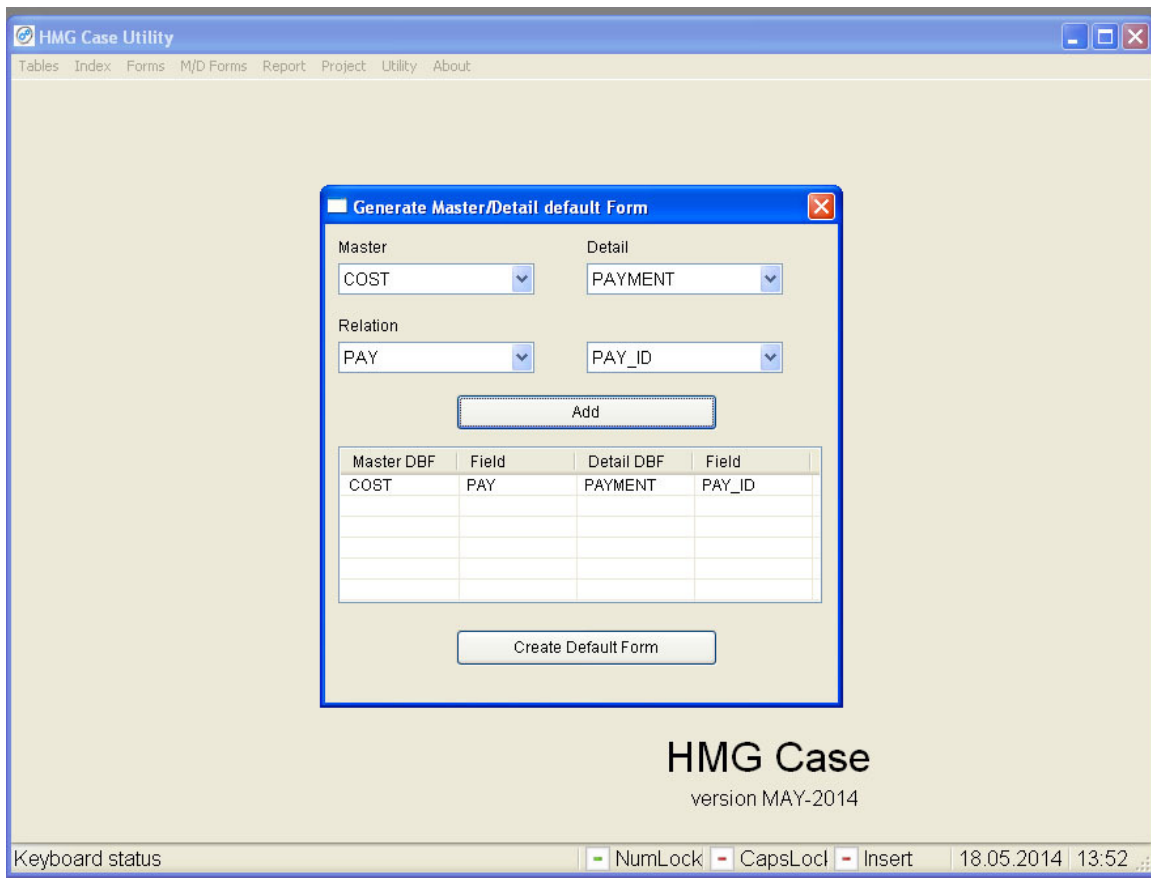
Napomena: ako ih ima više izaberite jedan par i kliknite na dugme **ADD**, izaberite sledeći par i kliknite na dugme **ADD** dok ne povežete sve, zatim kliknite na dugme **Create Default Forms**

Note: If you have more than one pair of select and click the **ADD** button, choose next pair and click the **ADD** button to not connect everything, and then click **Create Default Forms**

U ovom primeru polja koja čine relaciju su COST->PAY i PAYMENT->PAY\_ID

In this example, the fields that make up the relation COST-> PAY and PAYMENT-> PAY\_ID





Sledeći korak je izmena forme

The next step is a modified form

U ovom primeru ne morate ništa menjati

In this example, you do not need to change

Napomena: ako želite da nešto menjate promeniti atribut polja **PAY\_ID** i isključiti ih sve, kod ovakvih formi ključno polje druge tabele dobija vrednost prema pravilima i ne treba da se vidi ne ekranu

Note: If you want to change something change field attributes **PAY\_ID** and exclude them, in such form another key field of the table gets value according to the rules and should not be seen does not display

Napomena: polja **EXPENSE** i **PAYMENT** bi trebalo sumirati na ekranu i to ćete učiniti ako im uključite atribut **Sum**

Note: Fields **EXPENSE** and **PAYMENT** should be summarized on the screen and you will do if they turn attribute **Sum**

Sledeći korak je kreiranje programa koje se zove **EDMD\_n.PRG**

The next step is to create a program called **EDMD\_n.PRG**

Table FMD\_FLD

First Prev Next Last Find New Edit Delete Print Exit

COST

| Form | Bl... | DBF     | No | Field Name |
|------|-------|---------|----|------------|
| COST | 1     | COST    | 1  | PAY        |
| COST | 1     | COST    | 2  | DESCRIPT   |
| COST | 2     | PAYMENT | 3  | DATE1      |
| COST | 2     | PAYMENT | 4  | DESCRIPT   |
| COST | 2     | PAYMENT | 5  | EXPENSE    |
| COST | 2     | PAYMENT | 6  | PAYMENT    |
| COST | 2     | PAYMENT | 7  | DATE2      |
| COST | 2     | PAYMENT | 8  | PAY_ID     |

Form: COST

Block: 2

DBF: PAYMENT

No: 8

Field Name: PAY\_ID

Label: PAY\_ID

Field Type: N

Field Len: 3

Field Dec: 0

Row: 130

Col: 100

Picture: 999

Default: 0

☐ Browse
 ☐ Display
 ☐ DBF

☐ Input
 ☐ Edit
 ☐ Key

☐ Sum
 ☐ Query
 ☐ Valid

☐ Display
 ☐ Validate
 ☐ Display

Save Cancel Query

14:38

Potrebno ga je kompilirati i kad se pokrene izgledaće ovako

It is to compile and when run will look like this

Ako imate SUM polja moraćete im opis i poziciju promeniti ručno, ti redovi se nalaze ispod definicije MD\_Grid

If you have a SUM field will need their description and position change manually, these lines are located under the definition MD\_Grid

```
@ 200,10 GRID MD_Grid_1 ;
  WIDTH 750 ;
  HEIGHT 250 ;
  HEADERS { Recno", "DATE1", "DESCRIPT", "EXPENSE", "PAYMENT", "DATE2" } ;
  WIDTHS { 0,110,210,110,110,110 } ;
  JUSTIFY { 1,0,0,1,1,0 }

@ 460, 300 LABEL lSum value "SUM"
@ 460, 350 TEXTBOX nSum_1 WIDTH 95 NUMERIC INPUTMASK "999,999.99"
@ 460, 460 TEXTBOX nSum_2 WIDTH 95 NUMERIC INPUTMASK "999,999.99"
```

## Report

Sastavni deo svake aplikacije su izveštaji i to je sledeći deo koji ćemo napraviti, jedan jednostavan i jedan sa zbirom i međuzbirom

An integral part of each application reports and that's the next part that we do, one simple and one with the sum and subtotal

Da bi primer bio jednostavan napravljena je tabela iz koje će se praviti izveštaji, ali je pre izveštaja napuniti podacima

To make a simple example created a table from which to make statements, but rather reports the data to fill

Program PRE\_REP1.PRG je primer kako izgleda funkcija koja sumira troškove i plaćanja po vrstama

Program PRE\_REP1.PRG is an example of what it looks like a function that summarizes the charges and payments by type

Takvu funkciju ubacite u program za izveštaj ali da se izvršava pre kreiranja izveštaja.

Such functions include in the report but the program to be executed before creating the report.

Za početak napravimo program koji će prikazati stanja na današnji dan

To start a program that will display the balances of today

Pokrenite program HMGCASE i kliknite na Report->Default

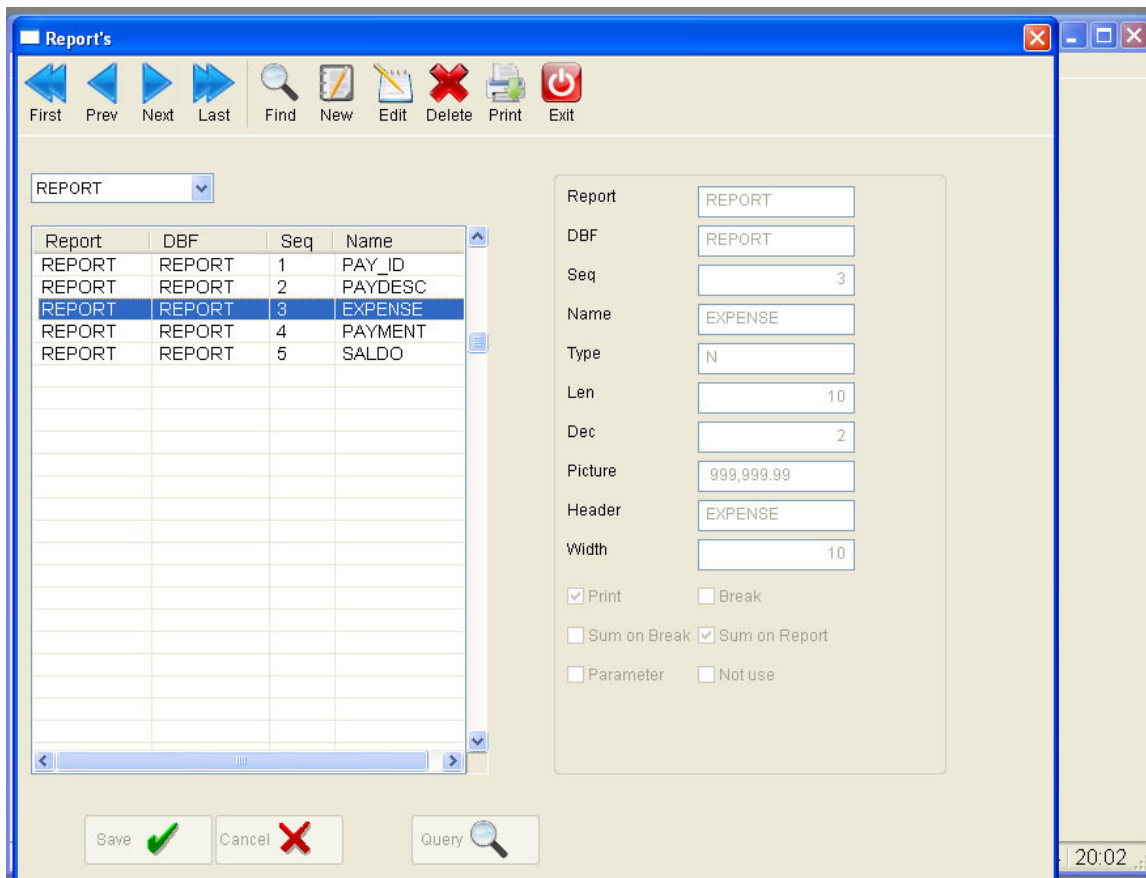
Run the program and click on HMGCASE Report-> Default

Izaberite tabelu REPORT i kliknite na Create Default Report

Select the table REPORT and click Create Default Report

Izaberite Report->Edit i videćete

Select Report-> Edit and you will see



Za početak ne morate ništa menjati, samo pređite na sledeću opciju Report->Generate i biće kreiran program **REP\_n.PRG**

Ukucajte *build REP\_n.PRG* i program će biti pokrenut, pitaće vas na koji štampač da šalje štampu, izaberite neki i pritisnite ENTER. Izveštaj će biti prikazan na ekranu, pogledajte i možete ga odštampati

Ono što možete doraditi kod opcije *Edit* je promena zaglavlja, formata i možete zadati sumiranje polja

U ovom primeru možete za polja **EXPENSE**, **PAYMENT** i **SALDO** izabrati format prikaza „999,999.99” i uključiti atribut **Sum on Report**

Generišite program i napravite kompilaciju

Izveštaj će prikazati po vrstama troškove,

For starters do not have to change anything, just go to the next option Report->Generate and will be created program **REP\_n.PRG**

Type *build REP\_n.PRG* and program will be launched, it will ask you which printer to send print, select one and press ENTER. The report will be displayed on the screen, and you can see it printed

What can further work with options *Edit* changes the header, you can specify the format and summarizing field

In this example, the fields can **EXPENSE**, **PAYMENT** and **SALDO** choose a display format, "999,999.99" and include attribute **Sum on Report**

Generate a program and make a compilation

The report will show the types of expenses,

plaćanja i saldo, i na kraju to sve sabrati

payments and balance, and in the end it all add up

## Report with group and summarize

Prethodni primer je bio jednostavan

The previous example was simple

Složeniji izveštaji omogućavaju grupisanje podataka i naravno sumiranje

More complex reports provide data grouping and summarizing of course

Uradimo takav primer

Let's do it

Imamo tabelu REPORT2, koju pre kreiranja izveštaja treba napuniti podacima i za to imamo primer PRE\_REP2.PRG koji treba ubaciti u program

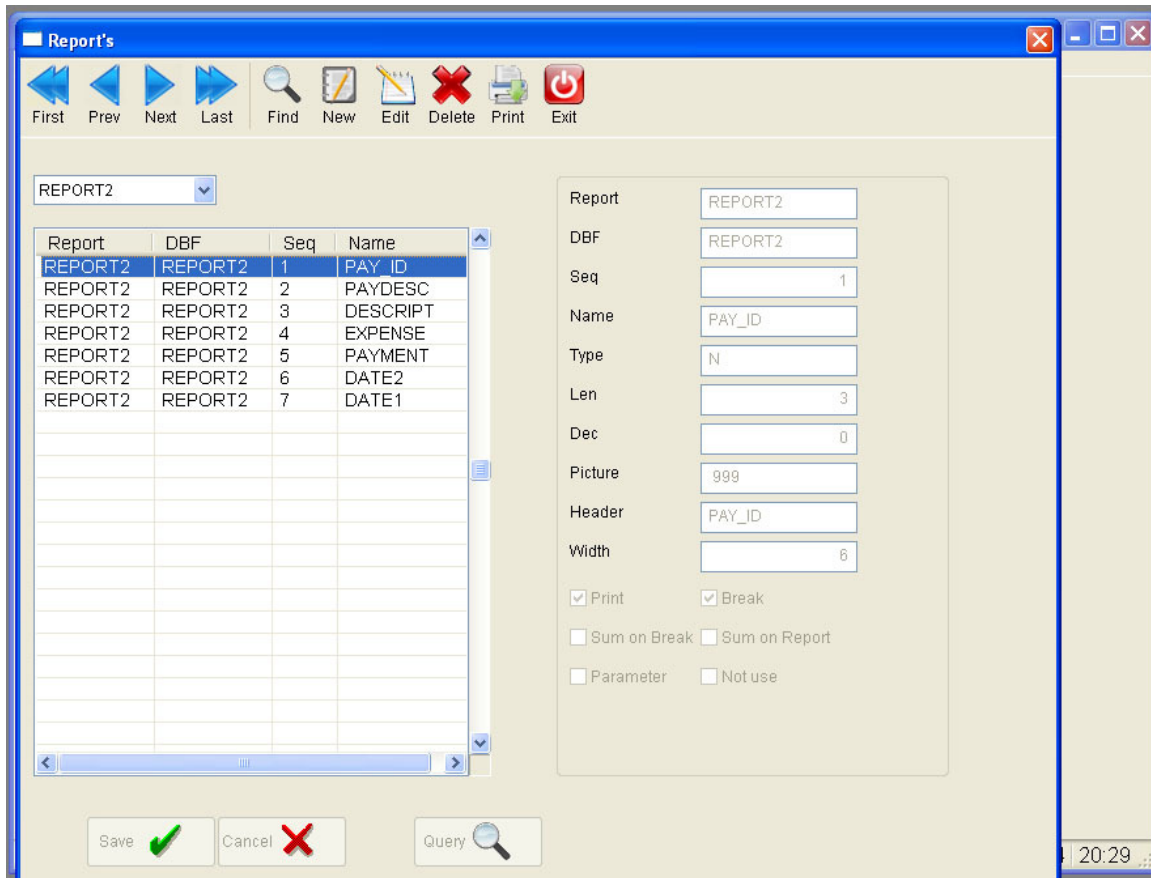
We have a table Report2, that before creating the report should fill the data and to have such PRE\_REP2.PRG that should be included in the program

Kliknite na opciju Report->Default i izaberite tabelu REPORT2, i kliknite na Create Default Form

Click on the option Report-> Default and select the table Report2, and click the Create Default Form

Kliknite na Report->Edit i videćete slično kao na slici

Click Report-> Edit and you will see a similar picture in



Sada moramo nešto da izmenimo da bi program radio ono što treba

Polje **PAY\_ID** uključite atribut *Break*  
 Polja **EXPENSE** i **PAYMENT** uključite atribut *Sum on Break* i *Sum On Report*, format prikaza stavite "999,999.99"  
 Polje **DATE1** i **DATE2** uključite atribut *Parametar*

Izađite iz programa, uradite Report->Generate i biće kreiran program **REP\_n.PRG**, za kompilaciju ukucajte build **REP\_n:PRG**

Napomena: atribut *Sum on Break* će napraviti međuzbir na poljima koja su tako označena kada se promeni vrednost polja gde je uključen atribut *Break*

Atribut *Sum on Report* će napraviti zbir na poljima koja ste označili, ovo se može

Now we need something to change to make the program do what you need

Field **PAY\_ID** turn attribute *Break*  
 Fields **EXPENSE** and **PAYMENT** include attribute on *Sum on Break* and *Sum On Report*, the display format put "999,999.99"

Field **DATE1** i **DATE2** turn attribute *Parameter*

Exit the program, do the Report-> Generate and will be created **REP\_n.PRG** program, the compilation type **build REP\_n:PRG**

Note: attribute *Sum on Break* will make the subtotal fields are marked so you can change the value of which is included attribute *Break*

Attribute *Sum on Report* will create a set of fields that are marked, it can be used even

koristiti i kada nijedno polje nije označeno atributom Break

none field is marked attributes *Break*

## Menu of application

Do sada smo napravili nekoliko program i to

So far, we have made several programs to

Za rečnik podataka

OPEN\_DBF

OPEN\_NTX

USE\_BDF

Data dictionary

OPEN\_DBF

OPEN\_NTX

USE\_BDF

Za forme

ED\_COST.PRG

ED\_PAY.PRG

ED\_MD\_CP.PRG

Forms

ED\_COST.PRG

ED\_PAY.PRG

ED\_MD\_CP.PRG

Za izveštaje

REP\_A.PRG

REP\_B.PRG

Reports

REP\_A.PRG

REP\_B.PRG

Sledeći korak je kreiranje glavnom programa koji će sadržavati meni

The next step is to create a main program that will include menu

Izaberite Project->Aplication i tamo unesite ime i opis aplikacije, širinu i visinu prozora i dimenzije menija

Select Project-> Aplication and then enter the name and description of the application, the width and height dimensions of the window and menu

Project->Seting je opcija gde ćete uneti setovanja za vaše programe

Project-> Seting option is where you enter the settings for your programs

Project->Menu bi trebao izgledati kao na slici

Project-> Menu should look like the picture





Obrisati sve redove koje se završavaju sa **// test**

Delete all lines that end with **// test**

Procedure su MAIN i to treba promeniti, može biti samo jedna procedura MAIN u aplikaciji, postoji komentar u nastavku

Procedures are MAIN and it needs to be changed, it can only be a MAIN procedure in the application, there is a comment in nastavku

Definicija DEFINE WINDOW sadrži red MAIN koji treba promeniti u MODAL, postoji komentar u nastavku

The definition of DEFINE WINDOW MAIN contains line that should change in MODAL, there is a comment below

## Resume

I na kraju da napišem da je moja ideja oduvek bila da krajnji rezultat dobijem vrlo brzo, što podrazumeva izvorni kod koji se može kompilirati i probati, ako radite samo default i generate rezultat se dobije za minut ili brže

Koliko će ovo pomoći početnicima? Nekima malo, nekima mnogo, sve zavisi od toga koliko će truda uložiti da nauče Harbour i HMG

Možda će nekom smetati što nema kontrola koje često koristi, ali ovo je namenjeno početnicima da im napravi osnovu programa koju dalje mogu da menjaju po potrebi

Finally I write that my idea has always been that the end result is a very fast, which means the source code that can be compiled and try, if you are running a default and generate results within a minute or faster

How will this help beginners? Some little, so much, it all depends on how much effort to invest in learning Harbour and HMG

Perhaps a mind that has no controls often used, but this is intended for beginners to make them the basis of the program that could still change if necessary