Impeller

Database programming language

Introduction

```
connect("website.org")              //available in subproc too
var1←tab #tab{fld1+fld2}>10         //assign
var1←var1⌄tab #{fld1+fld2}<5        //append
var2←tab #tab{fld1}=7
var3←var1\var2                      //"not", i.e. except
var3←var1&var2                      //"and", i.e. intersect
var3←var1⌄var2                      //"or", i.e. union
var1:tab{fld2}←7                    //request inside variable
←var1                               //duplicate tuples, i.e. copy
var1↓                               //save
var1↑                               //delete
rollback()
```

Departments

_departments //system table in one scheme

  name~text

  id≈uni2

  skin~nat16 //guid

tab //user table

  d~**dep** //fictional field

  id≈uni8

_permissions4deps //system table in one scheme

  d≈_departments

  t≈_tables

  r≈_roles

  permissions~nat2[5, 3, 2] //[In,Ed,As,De,Pi][Mb,Mb/s,Mb/s][u, r]

Corks

```
_ciphers                    //alter-or-create: create
    skin~nat16                  //guid
    hort~nat16                  //visible only by administrator
    encrypt, decrypt~bin            //visible only by administrator

_shortCorks, _longCorks         //alter-or-create: create table
    hort≈rand16                 //guid
    forEncrypt≈bool             //encryption vs. decryption
    u~_users                //dominant field: login identifier

_shortCorks                 //alter-or-create: add field
    cork~text

_longCorks                  //alter-or-create: add field
    cork~bin
```

Prop

```
tab                     //user table
    u~prop32                //32-bytes field
    id≈uni8

___props                    //system table on USB-flash
    s≈__schemes
    t≈_tables
    pk≈bob
    prop~varbyte

_permissions4outprops           //system table in one scheme
    d≈_departments
    t≈_tables
    r≈_roles
    permissions~nat2[3, 2]      //[Pi][Mb,Mb/s,Mb/s][u, r]
```

Twist

```
tab                           //alter-or-create: create
    fld1≈uni4                 //primary key
    fld2/ fld3~bin fld4~tab4       //overlap
    fld5~___twists                 //dominant field: twist identifier

___twists                     //system table on USB-flash
    id≈ser2                   //non-unique field
    s~__schemes
    forEncrypt~bool               //encryption vs. decryption
    twist~bin                 //like “long cork”
```

Sharding

```
//triggers "⬚In, ⬚Ed, ⬚As,
//           ⬚De, ⬚Va, ⬚Pi, ☐Sha"


_bases
    id≈uni2
    address~bob//IP-address, DNS-name, etc
```

Neutralization

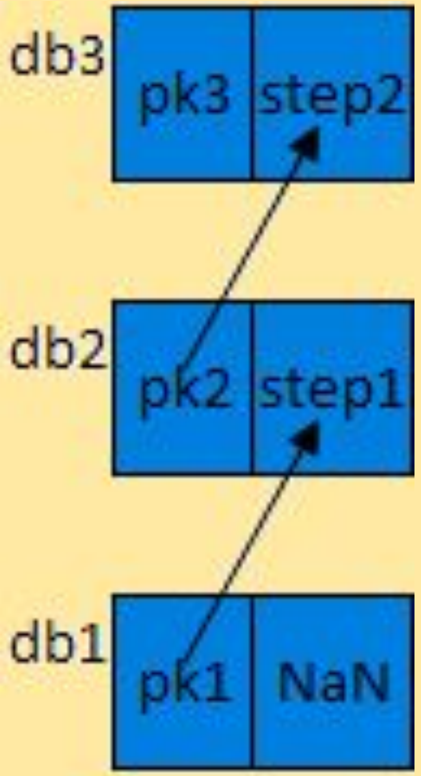
//**neu** datatype: 239…128 > 127…64 > **63…0 > 239…128**

```
_bases
    usr, pwd, cookie~text

bases2⊃_bases       //testator key

tab⇒bases2          //acceptor key
    fld1≈uni2       //primary key
    fld2~neu1       //version of tuple
```

Steping

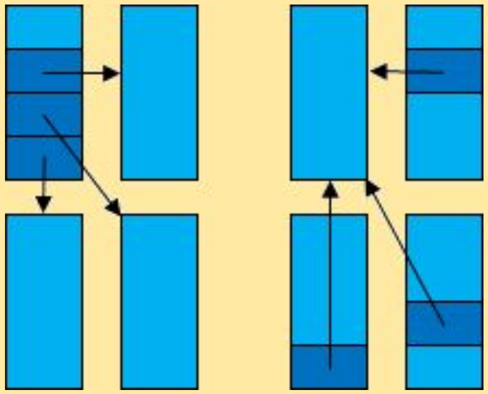
bases3⊃_bases //**testator** key

tabl⇒bases3 //**acceptor**
key
    fld1≈uni2
    fld2~**step**



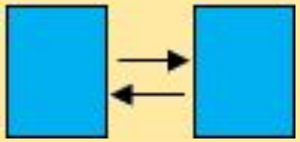
Palpation by nisba, by token

```
tab←db1@tab                    //by nisba db1 to default base
tab←m1@tab                     //db1,db2 to default


_tokens⊃_bases                 //testator key
    id≈ser2                //null allowed
    nisba~text≠"all"           //constraint

_tokens←{1,"db1",}{2,"db2",}{3,"db3"}{4,"db4"}
    $@@tab←db1@tab #fld=$//by token
```

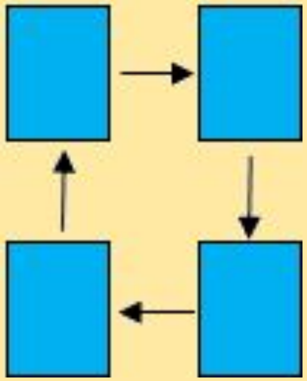


Palpation by mobs

```
_mobs                    //system table in one scheme
nisba≈text≠"all"         //constraint
dest≈text←"all"          //default value
mob≈text ≠"all"          //constraint
```



```
_mobs←{"db1", "m1"}{"db2", "m1"}{"db1", "m2"}{"db3", "m2"}{"db4", "m2"}
m1@tab ←m1@tab   //db1,db2 to db1,db2
m1@tab ←m1@@tab       //db1 to db2, db2 to db1
```



```
delete(_mobs)            //delete all tuples
_mobs←{"db1", "m3", "db2"}{"db2", "m3", "db3"}
      {"db3", "m3", "db4"}{"db4", "m3", "db1"}
m3@tab ←m3@tab   //db1 to db2, db2 to db3, db3 to db4, db4 to db1
```

Mailing

bases4⊃**_bases** //testator is “_bases”

tab
    fld1≈uni2 //primary key
    fld2~bases4 //**destination** for tuple

Stack, queue

var2:{fld2}↫var1:{fld1} //gaff tuple, roll tuple

//var1:{fld1}⟵var2:{fld2}

//var2:{fld2}⟵var1:{pk}

var1:{fld1}⤺var2:{fld2}

//var1⟵var2:{fld2}

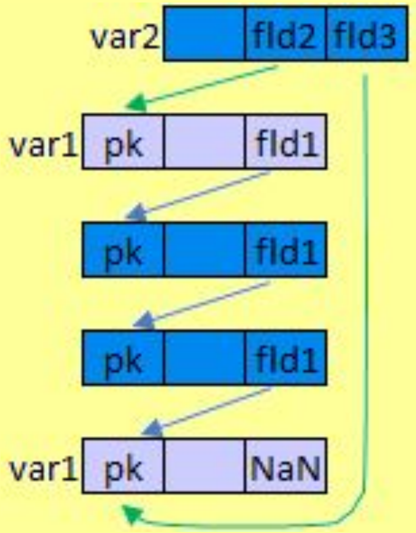
//var2{fld2}⟵var1:{fld1}

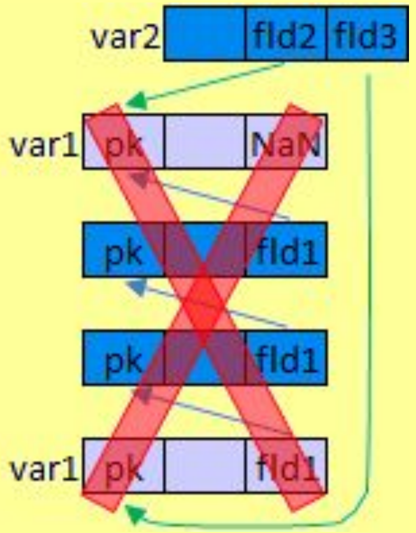
//var1:{fld1}⟵Null

var1:{fld1}↲var2:{fld3}

//var1⟵var2:{fld3}

//var2:{fld3}⟵tab1{pk} #tab1{fld1}=var1:{pk}





Medal (datatype)

```
tab                         //alter-or-create: create table
     fld1, fld2, fld3~int4      //three fields of same datatype
     fld4~5                 //5-bits field
     fld5~tab5{fld5}            //field with datatype of another field
     fld6~tab6                  //regular key to "tab6"
     fld7~own                   //regular key to "tab" itself
     fld8~db1@sch^tab8      //"tab8" in scheme "sch" of base "db1"
```

Variables

```
var1~int2
var4~7              //7-bits variable
var5~tab99{fld99} //variable with datatype of field
var6~tab6         //variable may refer only to “tab6”
var10⟵10          //minimal datatype: nat1
```

4 types of foreign keys

```
tab                //alter-or-create: alter table: add fields
    fld9~tab9 #    //bit key to "tab9"
    fld10~own #    //bit key to "tab" itself

tab1               //alter-or-create: create table
    fld1~any       //spur key to simple primary key
    fld2~_tables   //dominant field: where is primary key from

tab3               //alter-or-create: create table
    fld1~any       //flan key to some column
    fld2~_fields   //dominant field : where is column from
```

3 types of table reference

tab1⇒**tab2, tab3** //add **acceptor** reference
tab1⇏tab2 //remove acceptor reference

tab1⇐**tab4, tab5** //add **donor** reference
tab1⇍tab4 //remove donor reference

tab1⊃**tab6, tab7** //add **testator** reference
tab1⊅tab6 //remove testator reference

Tankers (clouds, stocks, socnets) for foto and video

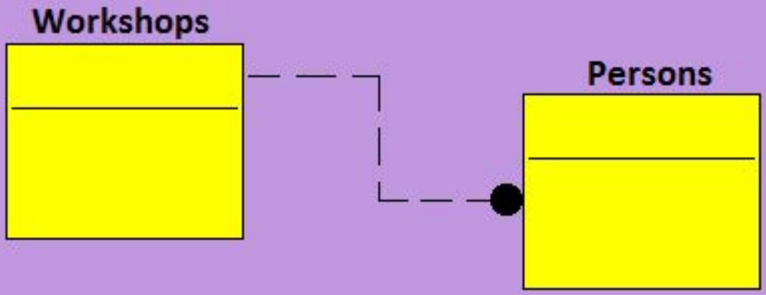
```
_tankers ⊃_bases          //testator key
    album~text            //folder, album, playlist
    overall, vacant~float4     //read-only, filled by plugin: space
    onlyFoto, onlyVideo~bool     //read-only, filled by plugin
    pingBeforeReturn~bool⟵true
    pingAfterUpload, invisibleInAlbum, stopUsing~bool⟵false

film⇒_tankers             //acceptor reference
    id≈uni8
    clip~link

film{clip} ⤥ head.* #head{pk=8}
head.* ⤦ film{clip} #film{id=10}
```

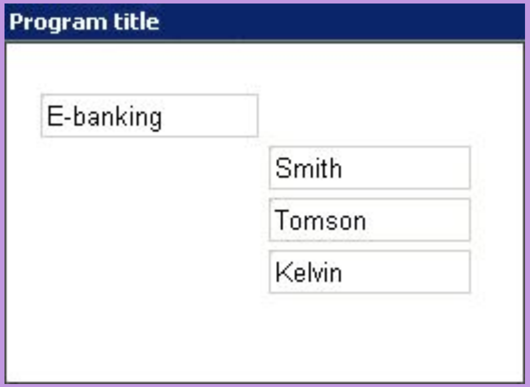
Fonding for departments and whole shemes

```
_bases5⊂*_bases
    dep~_departments
    period~datatime[3]      //for blocks, for log, and for snapshot

fond("depname")
flash("depname")
stop("depname")
```

Several tables on the screen – operator "snake" for tree

Workshops.Persons↯ # {idshop}=2





E-banking

Smith

Tomson

Kelvin

ER-modeller operator “snake” for system tables

```
_tables            //system table in one scheme
     name~text
     id~~uni4
     host~step        //result of “alter table”

_fields            //system table in same scheme
     order~nat1      //serial number inside table, e.g. offset
     name~text

_tables._fields↯
```

Electronic table – operator “snake” for field “any”

```
sheets                  //predistant table
    name~text               //name of sheet
    id~~uni1

cells                   //distant table
    value~any               //value of pilule
    a1, a2~nat2             //x-, y- coordinate of pilule
    s~sheets

⟨a.b.c. ⟩ sheets.cells ↯
a.b.c.sheets.cells ↯            //the same
a.* ↯                   //the same
```

Syndicates

⊂	⊂•	⊂/•	-⊂	-⊂/	⊂-	⊂/-	⊂-/	@/	⊂|	⊂|•	⊂:	=-|	=-|/
⊂	⊃	⊅	∈	∉	⊆	⊈	⊊	∅	∪	∩	∀	∃	∄
–<	+-<+	+-=<*	∇<	•-<	|-<	/<	@<	//<	•<|⊂	•<	•<•	•<•⊂	•<|
↞	⇱	⇛	⇽	↢	⇤	↼	⟜	↯	⤵	↶	↷	⤻	⤹
-:<	-::<	-=:<	∇∇=<:|•	∇∇=<::|•	|/<	-/<	-@<•		-<~	+<	+<•	+<|	+<|•
⇷	⇺	⇍	⇧	⇮	⤙	↽	⥈		↫	↲	↳	↴	⤴
-<:	<:	-=<	-=<::	∇=<::•	∇=<:	/</	–<@|•	<∇	<∇∇	⊂<	⊂<•	⊂<|	⊂<|•
↤	⊣	⇐	⇚	⇨	⇦	↜	⥉	⤚	⤜	⤤	⤣	⤥	⤦
-</	-<-//•	-=</	∇•=•<|	-=-<	-==<|	/<-/	-~<	-~~<	-~~~<•	-.<	-,<	-//	-//@@
↚	⇸	⇍	⬇	⋹	⤋	⥊	⬿	⬳	⟿	⇠	⇠	×	⊗
-<	-<•	-<|	-<|•	-<+	-<•+	-<|+	-<|•+	-<<	-<<|+	-<<<	-<•-<	-///	-///@@
←	→	↓	↑	↖	↘	↙	↗	↔	⤢	⤨	⇄	×	⨂
–::<∇•	-:<∇•	-=:<<	–,<∇•	•-<:	•<@	•<@•	@<:|	@<<	-@<<	-~<<	⊂-<•	-/-/	-//@@@
⤕	⤀	⇼	⤖	⬻	⊷	⊶	⫰	⧟	⥈	↭	⥹	⊛	⦻
<•	<-	<=	<-•	=-	==	~-	~=	~~	=/	=//	~/=	~=/	+-
>	≤	≦	≥	≡	≣	≃	≅	≈	≠	#	≇	≆	±
:/	+/	•/	∇/	∇•/∇•|	-/	~/	|/	+@@	/@@	|-@@	|-	||//	,,
⧉	Σ	Π	√	⧗	\	&	v	⊕	∅	⊖	⊥	⎍	;
::/	++/	••/	∇∇/	∇∇	@@	@@@	/-	+||	-//||	||@@	||	||/	/||
⌘	♡	^	∂	Δ	∞	℃	∠	⊞	⊠	⦷	∥	∦	⧄
~|	~|@	~|:	/∇	∇•/∇•	@/@	@@@/	/–	+∇∇	-//∇∇	<@@	@∇	⊂∇	||∇
∫	∮	$	⊲	⧓	%	℉	∡	⨹	⨻	⧀	○	⌂	⬚
|~	|~∇	|~-	|~|~	∇•/∇	∇•/<	<•/∇•	]∇/∇]	∇∇•	/∇•	/∇••	@|	•|	•/
⌒	▿	≏	⇳	⧑	⧔	⧕	⟗	▲	◀	▶	◇	⧫	■

Speaker seeks

free help of software engineers:

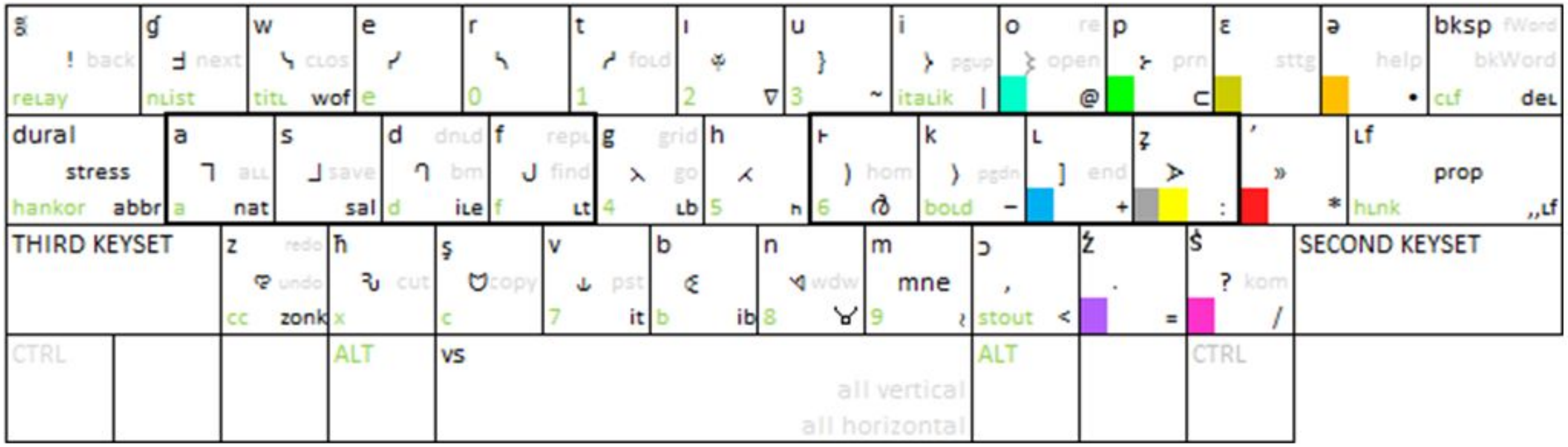
to implement GPL compiler;
to write schemes for foto, video, audio formats;
to create ttf-file with syndicates (it will be like font “Fira Code”)

Write a letters

EUR
ASIA
AFRICA
Pacific Ocean
VK.COM/DIMA.TURIN
VK.COM/DOMKA.TOLMACH
DIMA.TURIN@QQ.COM
DIMA.TURIN@GMAIL.COM

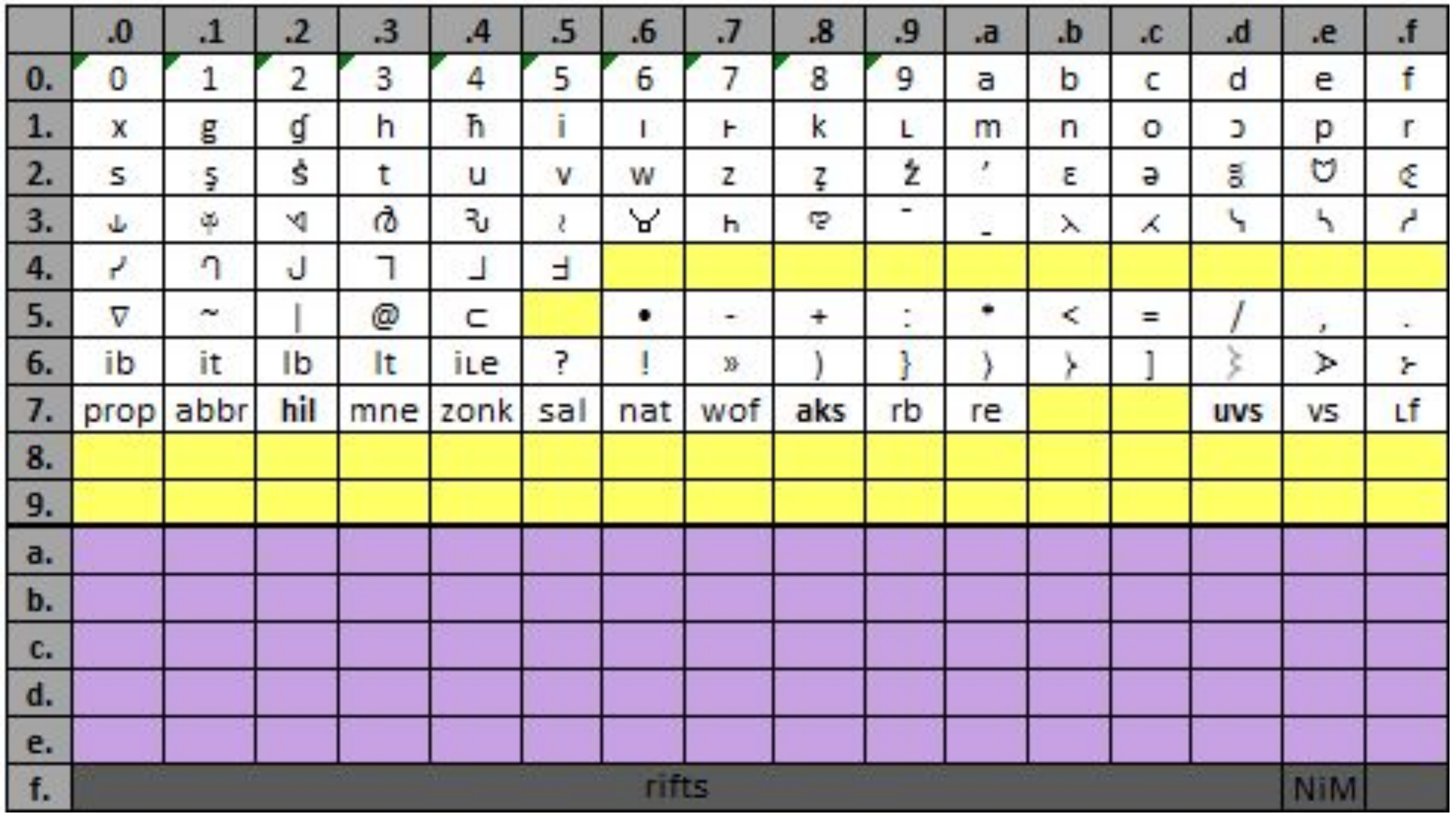
The end

Wolei (WOfal LAYout)



Details of Wolei are in separate project
Of World Phonemic Alphabet “Wofal”

BAM (BAiteme-Morpheme encoding)



	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.a	.b	.c	.d	.e	.f
0.	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
1.	x	g	ɠ	h	ħ	i	ı	⊦	k	ʟ	m	n	o	ɔ	p	r
2.	s	ş	ṡ	t	u	v	w	z	ẓ	ż	ʼ	ɛ	ə	ȭ	ʊ	ɛ̨
3.	⫝	ɸ	⊲	ð	ʋ	≀	ʊ	ɦ	ʘ	¯	_	⅄	⅄	ʕ	ʕ	ɾ
4.	ɾ	ʔ	ᴊ	˥	˩	Ⅎ										
5.	∇	~	|	@	⊂		•	-	+	:	•	<	=	/	,	.
6.	ib	it	lb	lt	iʟe	?	!	»	)	}	⟩	≻	]	⦄	⋗	⊱
7.	prop	abbr	**hil**	mne	zonk	sal	nat	wof	**aks**	rb	re			**uvs**	vs	ʟf
8.																
9.																
a.																
b.																
c.																
d.																
e.																
f.	rifts														NiM	

Details of BAM are also in separate project of World Phonemic Alphabet “Wofal”