SuperOffice 4.0

Database Manual

Purpose of the manual

This manual describes the internal SuperOffice database. In particular, it explains the naming-conventions which are in use and how the individual database tables are related to each other.

Please note that some of the database tables and records are essential to the correct operation of SuperOffice. You should, therefore, avoid changing these tables and records in any way. These critical tables and records are discussed in sections for Special Tables and Special Records.

The manual is written for the programming user. This manual is not for the typical business user of SuperOffice. That is to say, it has been assumed that you have a basic, working knowledge of using databases. By reading the manual it should be possible to gain a good understanding of the product's database structure.

In particular, it is assumed that the reader is familiar with terms such as table (or file), column (or field), row (or record), relation, checksum and database dictionary. It is also important to have some knowledge of SQL - the Standard Query Language. By using the information in this manual the reader will be able to access, to edit and to insert new records into the SuperOffice database.

Towards the end of the manual a set of examples is(are?) also provided. The examples show how to insert, how to update and how to manipulate data in a SuperOffice database. These include examples of SQL-code for Watcom and Oracle database systems.

The manual is divided into several sections :-

Database conventions and table descriptions
Relationships diagram
Special tables
Other Special tables
Special records
Values needed to access some of the tables
Selection lists in a SuperOffice database
Date format in a SuperOffice database
Note about using Travel function or Corporate version
Set of examples using the SuperOffice database

Database conventions and table descriptions

This section describes all of the available tables and standard conventions used in the SuperOffice database.

The explanations of the naming-convention, the data-types and the key-codes in use throughout the database are covered first.

Then the available tables are listed. For each table, a short description is then provided beside the table's name. Then each field (or column) in that table is explained in turn.

Conventions used throughout the SuperOffice Database

Naming conventions

All tables starts with the first column, called "tablename"+"_id". This uniquely identifies every record entered into the table. This value will stay unchanged for the life of the database. There is one exception to this rule, and that is new records created in a Travel database. Their id's will change when updating the parent database.

All fields referring to other records therefore have names ending with the extension "_id".

All fields selecting an entry from any one of the special lists for business, company interest and so on have names ending with the extension "_idx".

Standard data-types in use

The following datatypes are in use :-

Name	Size	Description
ushort	2	bytes unsigned integer
ulong	4	bytes unsigned integer
long	4	bytes signed integer
date_d	4	byte signed integer, number of seconds since January 1.,1970
date_t	4	byte signed integer, number of seconds since January 1.,1970
longid	4	bytes field, addressed as ulong
shortid	2	bytes field, addressed as ushort
string	1	array of bytes terminated by zero-byte, length includes zero-byte
vstring	1	as string, length specifies maximum length.
lvstring	0	as string, length specifies maximum length.

Standard definitions of field- and key-codes used

Key definitions		Description
Null	N	means no nul values allowed
	N*	user forced to enter a value
Key	P	Primary key
	U	Unique
	S	Secondary key
	C	Combined (described below table)
	F	Foreign key, Relation describes to what table

Descriptions of the SuperOffice database tables

Sequence next_id to be used for each table

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	id	N	PU	
longid	next_id			

Company Information on license and owner of this SuperOffice database

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	company_id	N	PU	
long	version	N		
ushort	revision			
_compres	reserved			

Associate All possible users of SuperOffice

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	associate_id	N	PU	
string[16]	loginname	N	SU	
shortid	group_idx	N	F	ListText
_assocres	reserved			
date_t	lastlogin	N		
date_t	lastlogout	N		
longid	person_id	N	F	Person
date_t	checklistlimit	N		
ushort	type			

Contact *Companies and Organizations*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	contact_id	N	PU	
string[50]	name	N	S	
string[40]	department		S	
string[12]	number1		S	
string[12]	number2		S	
longid	associate_id		F	Associate
longid	country_id	N	F	Country
shortid	business_idx	N*	FS	ListText
shortid	category_idx	N*	FS	ListText
ushort	xstop			
ushort	nomailing			
string[22]	custom1			
string[22]	custom2			
string[22]	custom3			
date_t	registered	N		
longid	registered_associate_id	N	F	Associate
date_t	updated	N		
longid	updated_associate_id	N	F	Associate
longid	text_id		F	Text
longid	mother_id		FS	Contact
longid	userdef_id		F	UDContactX
longid	url_id		F	Text

Person *Persons in a company or an organization*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	person_id	N	PU	
longid	contact_id	N	FC	Contact
ushort	rank	N	C	
string[32]	lastname		S	
string[22]	firstname		S	
string[12]	mrmrs			
string[40]	title			
string[32]	custom1			
string[22]	custom2			
string[22]	custom3			
longid	text_id		F	Text
shortid	position_idx		F	ListText
string[20]	mailstop			
ushort	year_of_birth			
ushort	month_of_birth			
ushort	day_of_birth			
ushort	phone_present			
longid	email_id		F	Text
longid	userdef_id		F	UDPerson
date_t	registered	N		
longid	registered_associate_id	N	F	Associate
date_t	updated	N		
longid	updated_associate_id	N	F	Associate
string[12]	person_number		S	
	cont_id+rank		S	

Address *Contact and Person addresses*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	address_id	N	PU	
longid	owner_id	N	FC	Cont/Pers
shortid	atype_idx	N	FC	ListText
string[12]	zipcode N*	S		
string[32]	city			
string[4]state		S		
string[30]	county			
string[42]	address1			
string[42]	address2			
string[42]	address3			
	owner_id+type_idx		SU	

Phone *Contact and Person phone numbers (+fax)*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	phone_id	N	PU	
longid	owner_id	N	FC	Cont/Pers
shortid	ptype_idx	N	FC	ListText
long	search_phone key	N	S	
string[18]	phone phone			
	owner_id+type_idx		SU	

Appointment *Appointments ,documents or sale*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	appointment_id N	PU		
longid	contact_id		FC	Contact
longid	person_id		F	Person
longid	associate_id	N	FC	Associate
shortid	group_idx	N	FS	ListText
date_t	registered	N		
longid	registered_associate_id	F	Associate	
date_t	done	N	SC	
date_t	do_by			
ushort	duration			
ushort	leadtime			
shortid	task_idx	N	F	ListText
shortid	priority_idx		F	ListText
ushort	type		C	
ushort	status		C	
ushort	private			
ushort	alarm			
longid	text_id		F	Text
longid	project_id		FC	Project
longid	mother_id		FS	Appointment
longid	userdef_id		F	UDAppointment
longid	document_id		F	Document
	cont_id+done		S	
	assoc_id+done+type		S	
	assoc_id+status	S		
	proj_id+done		S	

Document *Has an appointment record as owner*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	document_id	N	PU	
longid	application_id	N	F	Application
string[30]	name	N		
string[60]	header			
string[20]	our_ref			
string[20]	your_ref			
string[20]	searchname		S	
string[30]	attention			
longid	about_id		F	Text

Project

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	project_id	N	PU	
string[50]	name		S	
string[12]	project_number	S		
shortid	type_idx		F	ListText
shortid	status_idx		F	ListText
longid	associate_id		F	Associate
string[22]	custom1			
string[22]	custom2			
string[22]	custom3			
longid	text_id		F	Text
date_t	registered	N		
longid	registered_associate_id	N	F	Associate
date_t	updated	N		
longid	updated_associate_id	N	F	Associate
longid	userdef_id		F	UserDef(project)

ProjectMember

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	projectmember_id	N	PU	
longid	project_id		FC	Project
ushort	ownertype			
longid	owner_id		F	Contact/Person
ushort	rank		C	
shortid	mtype_idx		F	ListText
	proj_id+rank		S	

Sale

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	sale_id	N	PU	
longid	associate_id	N	FC	Associate
shortid	group_idx	N	FC	ListText
longid	contact_id	N	FC	Contact
longid	person_id		FC	Person
date_t	registered			
date_t	saledate	N	C	
long	amount			
ushort	probability_idx	N	F	ListText
longid	appointment_id N	FC	Appointment	
longid	text_id		F	Text
longid	project_id	N	FC	Project
long	earning			
ushort	earning_percent			
longid	userdef_id		F	UserDef
	cont_id+saledate		S	
	assoc_id+saledate		S	
	proj_id+saledate		S	
	grp_idx+cont_id+saledate		S	

ContactInterest

Type	*Field/Index*	*Null*	*Key*	*Relation*
ulong	contactinterest_id	N	PU	
longid	contact_id	N	FS	Contact
shortid	cinterest_idx	N	FS	ListText
date_t	interest_date	N		

PersonInterest

Type	*Field/Index*	*Null*	*Key*	*Relation*
ulong	personinterest_id	N	PU	
longid	person_id	N	FS	Person
shortid	pinterest_idx	N	FS	ListText
date_t	interest_date	N		

ZipToCity

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	ziptocity_id	N	PU	
longid	country_id		FC	Country
string[12]	zipcode N	PC		
string[32]	city	N	S	
	country_id+zipcode		S	

Template *Describes templates available for writing new documents*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	template_id	N	PU	
ushort	machinetype			
ushort	dynamic			
ushort	type		S	
string[32]	templatename			
string[32]	filename			
longid	application_id		F	Application
ulong	refcount			
shortid	template_idx	N	F	ListText
longid	autoevent_id		F	AutoEvent
ushort	generate_sale			

Application

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	application_id	N	PU	
string[70]	name	N		
string[130]	path			
string[62]	filename	N		
string[62]	parameters			
string[62]	working_directory			
string[40]	dde_topic			
string[100]	dde_open			
string[40]	dde_minimize			
string[40]	dde_maximize			
string[40]	dde_restore			
ushort	dde_show			
string[100]	dde_merge			
string[6]	mac_creator			
string[6]	mac_type			
string[4]	win_type			
longid	win_charactermap_id		F	CharacterMap
longid	mac_charactermap_id	F	CharacterMap	

ListText *See also "Selection lists in a SuperOffice database"*

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	listtext_id	N	PU	
ushort	list_id	N	C	
ushort	list_idx	N	C	
ushort	show_idx	N	C	
ushort	parent			
string[30]	text			
	list_id+list_idx		SU	
	list_id+show_idx		S	

RedLetterDay

Type	*Field/Index*		*Null*	*Key*	*Relation*
longid	redletterday_id	N	PU		
date_t	reddate	N	C		
longid	country_id		N	FC	Country
ulong	reds				
ulong	colored				
ushort	color				
string[80]	text				
	cntry_id+reddate			S	

Text

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	text_id	N	PU	
ushort	type		C	
longid	owner_id		FC	Owner
lvstring[2048]	text			
	owner_id+type	S		

Country

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	country_id	N	PU	
string[40]	name		S	
string[40]	english_name		S	
string[10]	phone_prefix			
ulong	layout_id			
ushort	time_offset			
string[10]	time_name			
string[20]	summer_time			
string[20]	winter_time			
string[6]zip_prefix				
ulong	flagres_id			

Preference

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	preference_id	N	PUC	
shortid	type	N	C	
vstring[1024]	pdata			
	type+id	SU		

Selection

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	selection_id	N	PU	
longid	associate_id	N	FS	Associate
string[50]	name	N	S	

SelectionMember

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	selectionmember_id	N	PU	
longid	selection_id	N	FC	Selection
longid	contact_id	N	FC	Contact
	sel_id+cont_id		SU	

UserDefTable

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	userdeftable_id	N	PU	
string[40]	name			
ushort	table_number			
ushort	width			
ushort	length			

UserDefControl

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	userdefcontrol_id	N	PU	
longid	userdeftable_id N	FC	UserDefTable	
ushort	type			
shortid	list_id			
ushort	csize			
string[20]	prefix			
ulong	prefix_pos			
ushort	prefix_width			
ulong	control_pos			
ushort	control_width			
ushort	control_height			
ushort	rank		C	
ushort	field_number			
	udtab_id+rank		S	

UDContact1

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	udcontact1_id	N	PU	

UDContact2

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	udcontact2_id	N	PU	

RecordLink

Type	*Field/Index*	*Null*	*Key*	*Relation*
longid	recordlink_id	N	PU	
ushort	tablenumber		C	
longid	record_id		C	
longid	associate_id		FC	Associate
longid	external_id			
date_t	generatedtime			
ushort	flag			
	tabno+rec_id+assoc_id	S		
	assoc_id+tabno+rec_id	S		

Special tables

Some of the tables in a SuperOffice database are very special.

Firstly, in a SuperOffice database there are "dictionary information tables" which contain a description of the database itself. These tables are not to be changed. They should never be updated or inserted into. The tables make active use of checksum fields, and SuperOffice will not start if these checksums are incorrect.

The dictionary may, however, be changed to adapt to a different table definition than standard SuperOffice. This change can only be done with help and advice from SuperOffice's support.

The following tables are "dictionary information tables" :-

ConceptualDatabase
ConceptualTable
ConceptualField
PhysicalSchema
PhysicalDatabase
PhysicalTable
PhysicalField
Relationship
DicIndex
IndexField

Then there are other important tables to be handled with great care :-

Company This table contains the owner of the database. Tampering with this table may result in SuperOffice not starting any more.

Associate Contains all users or employees in the owner Company. Any change to this table may result in locking users out of the database

Sequence This table contains rows for generating unique id's for each SuperOffice table. This table should never be inserted into or deleted from.
Later we will explain how to use this table.

ActiveUser Any changes made to this table may result in users being logged out automatically.

Further, listed below are a number of tables which may result in strange results if insert, delete or update are used on any of them :-

Mail
MailAppointment
AutoEvent

SuperOffice has special Travel tables containing information on users with the Travel option in SuperOffice. You must never change any of these :-

All tables that contains the names: travel, satellite or area

Other Special tables

Use same format as previous Special tables explanations. ------

One table is used for connecting PDA computers.

RecordLink Contains information on duplicates (i.e. records currently in SuperOffice and the PDA) to correctly update when syncronizing again.

Information on different country specific information is also stored

Country Defines address layout, phone prefixes and other country specific information

The extra dialog you may define for all Customers, store its data in tables which will change whenever you change the definition of this dialog.

UDContact1 and
UDContact2

These tables are used to store rows in the user defined Contact table. To not loose old data, Every time you change the definition of the user defined table, we switch to using the other table. To check which table is the active one, you have to read one row from Preference.

SELECT pdata FROM PREFERENCE WHERE preference_id = 503

pdata will contain '1' if UDContact1 is active, and if it contains '2' UDContact2 is active.

Calendar information regarding which days are to be red are stored in one table

RedLetterDay

This table is used to mark specific days in the calendar as red. The format of some columns in this table are very special. First for every month there whould be a row with reddate = month 1st midnight. This should contain a bitmap of all red days for this month column reds. This 32 bit bitmap is computed using least significant bit as 1st of month and 1 as red 0 as not red.

Example September 1995. Sundays are 3rd, 10th, 17th and 24th. Lets just say that the 15 and 23 are to be marked red in addition to Sundays. This results in the following

```
Sundays
1st                     32nd
  00100000 01000000 10000001 00000000
```

```
Then extra reds
  00000000 00000010 00000010 00000000
```

```
If we OR these together we get the correct bitmap
  00100000 01000010 10000011 00000000
```

Now LSB is to the left, and we split this in 8 bit portions to make it easier. First binary LSB to the right as we are used to

```
  00000100 01000010 11000001 00000000
```

Then type this number in Windows calculator scientific mode using binary numbers. When typed click on decimal mode and you get the value

```
  12665348
```

So to set RedLetterDay for september 1995 you do as follows:

Get unique id for RedLetterDay table (id = 30 in Sequence), and increment.

```
INSERT INTO RedLetterDay (redletterday_id, reddate, country_id, reds)
VALUES (next_id, 809913600, mycountry, 12665348)
```

This will make September 3, 10, 15, 17, 23, 24 red in the calendar.

Special records

Inside the Contact table there is one row describing your own organisation. This row should only by updated using SuperOffice and never be deleted. If you delete or update this row from outside SuperOffice you may not be able to run SuperOffice anymore.

This row usually has an id value of 2, but this may change. You should always check which Contact row is owner company.

The easiest way to test this is to find your row in the Associate table, find releated Person row, and from that one read contact_id.

```
SELECT person_id FROM Associate WHERE loginname = '<MyId>'
SELECT contact_id FROM Person WHERE person_id = person.person_id
```

The Contact row with contact_id = person.contact_id should not be updated or deleted.

Values needed to access some of the tables

Some of the tables must have special values which are not easy to guess correctly. They are listed below with a description of how to use them.

Use of the Sequence table

Table	=	*Entry number*
Company	=	11
Associate	=	12
Contact	=	13
Person	=	14
Address	=	15
Phone	=	16
Appointment	=	17
Document	=	18
Mail	=	19
Project	=	20
ProjectMember	=	21
Sale	=	22
Budget	=	23
ContactInterest	=	24
PersonInterest	=	25
ZipToCity	=	26
Template	=	27
Application	=	28
ListText	=	29
RedLetterDay	=	30
Text	=	31
Country	=	32
Preference	=	33
CharacterMap	=	34
Selection	=	35
SelectionMember	=	36
AutoEvent	=	37
MailAppointment	=	38
AccessRights	=	39
ActiveUser	=	40
TravelTransactionLog	=	41
TravelGeneratedTransaction	=	42
Traveller	=	43
TravelCurrent	=	44
TravelGeneratedDatabase	=	45
UserDefTable	=	46
UserDefControl	=	47
UDContact1	=	48
UDContact2	=	49
RecordLink	=	50
???	=	51
???	=	52
???	=	53
???	=	54
???	=	55

These constants are used when you want to insert records in the database from outside SuperOffice, or using SQL calls into SuperOffice.

When inserting a new row in a table you will need to read the id-value to use for this record. The values above are used to select which table you want the next id-value for.

Reading and updating the next id-value should always be performed as one operation, because if someone else reads and updates this information simultaneously, a database error will occur when inserting the new row.

Example find next id for Contact table (value = 13)

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 13;
SELECT next_id - 1 FROM Sequence WHERE id = 13;
COMMIT TRANSACTION
```

Selection lists in a SuperOffice database

Examples of lists include business-types, categories, appointment-types.
These are implemented in one common table containing all the lists. This table was defined above as ListText. Each entry is selected using two values each consisting of 2 bytes. the first list_id has one value for each list, and the second selects the correct item within the list. Only the scond value is stored inside a row of SuperOffice data.

These lists have the following list_id values.

Business type	=	1
Category	=	2
Appointment type	=	3
Document in	=	4
Document out	=	5
Group	=	6
ContactInterest	=	8
PersonInterest	=	9
PersonPosition	=	10
Address type	=	11
Phone type	=	12
Priority	=	13
Probability	=	14
Project type	=	15
Project status	=	16
Project Member title	=	17

```
SELECT list_idx FROM ListText WHERE list_id = 1 AND text = 'Industry'
```

Date format in a SuperOffice database

The date format in SuperOffice is a 4 byte value containing seconds from 1st of January 1970 00:00 (midnight). This will limit a date to stay within 1st Janary 1970 to 1st January 2038 (This is not precise).
This date value is easily produced using a C function called mktime.
Find Access function name for this and SQL statement for this.
Any field containing only a date (not time) is set to midnight that day.

1st September 1995 midnight = 809913600

One day is 60 * 60 * 24 = 86400

6th September 1995 = 809913600 + (86400 * 5) = 810345600

Add date + time!!!!

Note about using Travel function or Corporate version

SuperOffice makes it posible to update databases in different locations using transaction logs. These are updated when a SuperOffice user changes data. They will not be updated when inserting, updating or deleting from outside SuperOffice.

Maybe we should explain how to enter into transactionlog also

Set of examples using the SuperOffice database

Example of creating a new customer

To create a new customer (Contact table) the following must be done

1. Get the sequence number of for the next id in Contact table

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 13;
SELECT next_id - 1 FROM Sequence WHERE id = 13;
COMMIT TRANSACTION
```

2. Insert a new row into Company (Contact) table.

```
INSERT INTO Contact (contact_id, name, country_id, ....)
VALUES (next_id, "Newname", 81, ...
```

You will have to insert a value in the following fields.
id
name
country_id

The following columns should have a default value
userdef_id = 0
text_id = 0

To follow SuperOffice normal rules you should also insert values into the following filelds
registered = todays date & time

The correct value for country_id is normally the country code for this country when dialing. Whenever two or more countries use the same. You may find this using a select statement on the Country table.
SELECT country_id FROM Country WHERE name = "<country needed>"

Example of inserting a new Customer with Phone, Fax, Address and one Person with direct Phone and Home address.

Find correct country_id

```
ucountry_id = SELECT country_id FROM Country WHERE name = "USA"
```

Get next_id for Contact table

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 13;
SELECT next_id - 1 FROM Sequence WHERE id = 13;
COMMIT TRANSACTION
```

Find Associate_id

```
SELECT assosiate_id FROM Associate WHERE loginname = '<myloginname>'
```

Find correct business_idx

```
SELECT list_idx FROM ListText WHERE list_id = 1 AND text = 'Industry'
```

Find correct category_idx

```
SELECT list_idx FROM ListText WHERE list_id = 1 AND text = 'Industry'
```

Check if Customer number is to be generated and Unique

```
SELECT pdata FROM Preference WHERE preference_id = 202
If pdate == 1 Generate number
SELECT pdata FROM Preference WHERE preference_id = 201
If pdate == 1 Unique numbers
```

Find Customer number value

```
BEGIN TRANSACTION
UPDATE Preference SET pdate = pdata + 1 WHERE preference_id = 100
SELECT pdata - 1 FROM Preference WHERE preference_id = 100
COMMIT TRANSACTION
```

Find date value for today

TBDefined

```
INSERT INTO Contact (contact_id, name, number2, associate_id, country_id,
        business_idx, category_idx, stop, registered,
        registered_associate_id, updated, updated_associate_id,
        text_id, mother_id, userdef_id )
VALUES ( next_id, "MyName", assoc_id, .....)
```

Add one address for this customer

Get next_id for the Address table

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 15;
SELECT next_id - 1 FROM Sequence WHERE id = 15;
COMMIT TRANSACTION
```

Find Address type

Contact	= 0
Person	= 16384

Postal	= 1
Street	= 2
Private	= 3

Contact Postal address atype_idx = Contact + Postal = 0 + 1 = 1

```
INSERT INTO Address (address_id, owner_id, atype_idx, zipcode, city, state,
        address1, address2)
VALUES ( next_id, contact_id, 1, "12345", "Bedford", "MA",
        "The building", "Great Road 123")
```

Add one phone number and one Fax

Get next_id for the Phone table

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 16;
SELECT next_id - 1 FROM Sequence WHERE id = 16;
COMMIT TRANSACTION
```

Find Phone Type

Contact	= 0
Person	= 16384
Phone	= 1
Phone2	= 2
Fax	= 3
Private	= 4
Cellular	= 5
Pager	= 6

Contact Phone = Contact + Phone = 0 + 1

Compute search phone
Use numeric value of phone number.
NB! Keep in mind that phone numbers never include abroad prefix and country code.

```
INSERT INTO Phone (phone_id, owner_id, ptype_idx, search_phone, phone)
VALUES (next_id, owner, 1, 6172752140, (617)275-2140)
```

Then a fax number

Get next_id for the Phone table

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 16;
SELECT next_id - 1 FROM Sequence WHERE id = 16;
COMMIT TRANSACTION
```

```
INSERT INTO Phone (phone_id, owner_id, ptype_idx, search_phone, phone)
VALUES (next_id, owner, 3, 6172752141, '(617)275-2141')
```

Example of adding Persons

Get next_id for the Person table

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 14;
SELECT next_id - 1 FROM Sequence WHERE id = 14;
COMMIT TRANSACTION
```

The Column rank defines the default order for these persons inside the company. It starts on 0.

```
INSERT INTO Person (person_id, contact_id, rank, firstname, lastname, mrmrs,
        title, registered, registered_associate_id,
        updated, updated_associate_id, person_number)
VALUES (next_id, cont_id, 0, "Steve", "Warson", "Mr.", "Car driver" ,
        809697989, asoc_id, 809697989, asoc_id, "<pnumber>")
```

Example of adding info text for a Contact

Get next_id for the Text table

```
BEGIN TRANSACTION
UPDATE Sequence SET next_id = next_id + 1 WHERE id = 31;
SELECT next_id - 1 FROM Sequence WHERE id = 31;
COMMIT TRANSACTION
```

Find Text type

Contact info	= 1
Person Info	= 2
Person eMail	= 3
Appointment text	= 4
Document about	= 5
Project text	= 6
Sale text	= 7
Notepad	= 8
MailAppointment	= 9
URL text (www) = 10	

```
INSERT INTO Text (text_id, type, owner_id, text)
VALUES (next_id, 1, cont_id, "Info text")
```

Then attach this Info text to correct Customer.

```
UPDATE Contact SET text_id = next_id WHERE contact_id = cont_id
```

Example of inserting a Contact Interest

Get unique contactinterest_id for ContactInterest table.
find cinterest_idx

```
INSERT INTO ContactInterest (contactinterest_id, contact_id, cinterest_idx,
          interest_date)
VALUES (next_id, cont_id, interest_idx, today)
```

Example of inserting a new appointment

This will consist of one row in Appointment and one row in Text.

Get unique id (next_id)

Find group_idx

Find done and do_by

Not completed task done == do_by

Completed task and not on diary (no start time) done == checked date
do_by == intended done date.

Find duration
Number of minutes.
MinutesToday(done) + duration < 1440 (24 hrs)

Leadtime is not in use at present

Find task_idx

Find priority_idx

Find Appointment type

Appointment in diary	= 1
Appointment in check list	= 2
Note shown on bottom of daiary(day) screen	= 3
Incoming Document	= 4
Outgoing Document	= 5

Find status type

Not done = 1
Completed = 3

Private not in use at present

Find Alarm
Alarm on = 0x4000 = 16383
Alarm = 16383 + Number of minutes before appnt.
10 minutes before appnt alarm on = 16393.

```
INSERT INTO Appointment
VALUES
```

Get unique next_id for Text table

```
INSERT INTO Text (text_id, type, owner_id, text)
VALUES (next_id, 4, appnt_id, "Appointment text")
UPDATE Appointment SET text_id = next_id, WHERE id = appnt_id
```

Example of inserting a new Document

This will result in one Appointment, one Document and zero or more Text rows.

Column attention in Document table is a text containing eighter some text typed by user, or firstname lastname of selected person, if a person was selected.

Get unique id for Document table

Create document name. Remember that .EXT is going to be used for opening the document later.

```
INSERT INTO Document (document_id, name, header, attetion)
VALUES (next_id, "SUPER008.DOC", "Proposal for 50 user SuperOffice",
        "Steve Warson")
```

If about text is needed you have to create a row in table Text, and then update field about_id in Document row.

Appointment record created as previously with the following chanes:

task_idx dependent on incoming or outgoing document.
type = 5 for outgoing document, 4 for incoming document.
status = 3. Always completed
No alarm.
Document_id will containt the id of the Document row we just created

```
INSERT INTO Appointment (appointment_id, ..... , document_id ....)
VALUES (next_id, ..... , doc_id, ....)
```

Relationships diagram