



Introduction to the Data Model



December 9, 2013

© Guidewire Software, Inc. 2001-2013. All rights reserved.
Do not distribute without permission.

Guidewire training materials contain Guidewire proprietary information that is subject to confidentiality and non-disclosure agreements. You agree to use the information in this manual solely for the purpose of training to implement Guidewire software solutions. You also agree not to disclose the information in this manual to third parties or copy this manual without prior written consent from Guidewire. Guidewire training may be given only by Guidewire employees or certified Guidewire partners under the appropriate agreement with Guidewire.

Lesson objectives

- By the end of this lesson, you should be able to:
 - Describe the contents of a Guidewire data model
 - Identify information about a given application's data model
 - Reference entity fields using dot notation

This lesson uses the notes section for additional explanation and information.
To view the notes in PowerPoint, select View → Normal or View → Notes Page.
When printing notes, select Note Pages and Print hidden slides.

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

2

G U I D E W I R E



Lesson outline

- Contents of the data model
- The Data Dictionary
- Objects and the data model

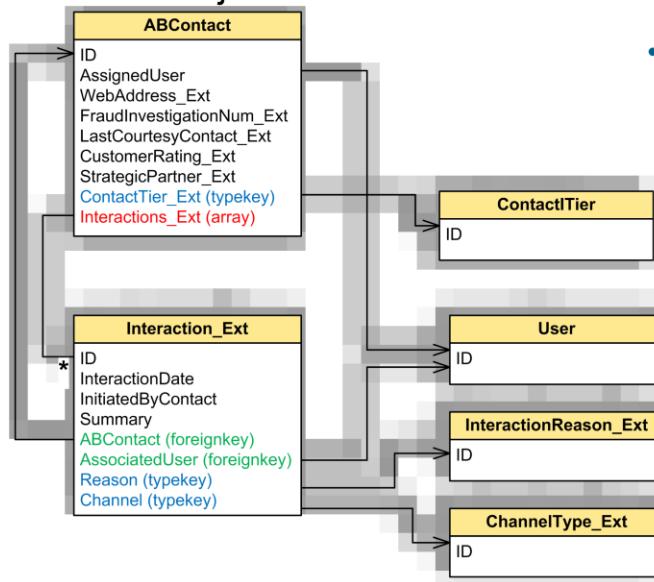
© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

3

G U I D E W I R E

The data model

- For each Guidewire application, the **data model** is the set of data objects and information about their relationships



- Consists of entities, entity fields, typelists and typekeys

- Part of Guidewire data model
- Other data model components include field validators and abstract data types
- Diagram shows work completed by students in course

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

4

G U I D E W I R E

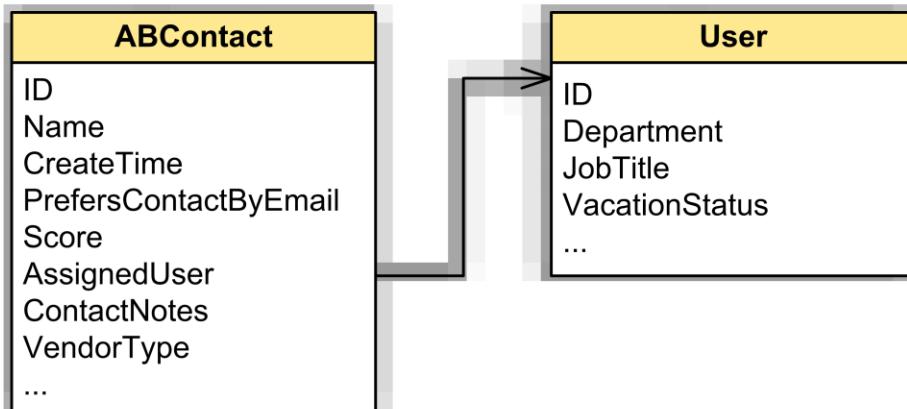
The diagram is a data model for an entity and related entities in TrainingApp that students will create over this course.

There are two other components to a Guidewire data model: field validators and abstract datatypes. A field validator defines the format requirements for specific fields, such as a postal code that must consist of 5 characters, each of which is a digit 0 through 9. An abstract data type defines a datatype that is built on top of a simple data type and is used to ensure the consistent definition of similar fields. For example, an ExchangeRate datatype could be defined as a decimal with a precision of 7 and a scale of 6. Then, any field that is intended to store exchange rate values could use this abstract data type, thereby ensuring that all exchange rate fields in the data model have been declared in the same way.

Field validators are discussed in this course. Abstract data types are not covered in detail. To learn more about abstract data types, please refer to documentation.

Data model entities

- A **data model entity** is an abstract definition of a group of business objects used by the application
- Examples: ABContact and User



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

5

G U I D E W I R E

Data model entities in the database

Table - dbo.ab_abcontact*					
ID	Name	CreateTime	PrefersContact...	Score	AssignedUser
64	United Natural Foods Inc	9/4/2009 1:51:32 PM	False	NULL	2
65	3M	9/4/2009 1:51:32 PM	False	NULL	3
74	Express Auto	9/4/2009 1:51:32 PM	False	81	3
75	European Autoworks	9/4/2009 1:51:32 PM	True		
76	M B Garage	9/4/2009 1:51:32 PM	False		
77	Burlingame Saab	9/4/2009 1:51:32 PM	False		
78	Menlo Park Chevron	9/4/2009 1:51:32 PM	True		
79	Cupertino's Smog Pro and Auto Repair	9/4/2009 1:51:32 PM	False		
80	Meinecke Car Care Center	9/4/2009 1:51:32 PM	False		
81	Morgan Hill Auto Body	9/4/2009 1:51:32 PM	False		
82	Hollister Muffler and Quick Lube	9/4/2009 1:51:32 PM	True		

ABContact

ID
Name
CreateTime
PrefersContactByEmail
Score
AssignedUser
ContactNotes
VendorType
...

- Most entity data is stored in its own database table
 - Some tables are shared
 - Other entities are virtual, non-persistent entities and no data is managed in the database

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

6

G U I D E W I R E

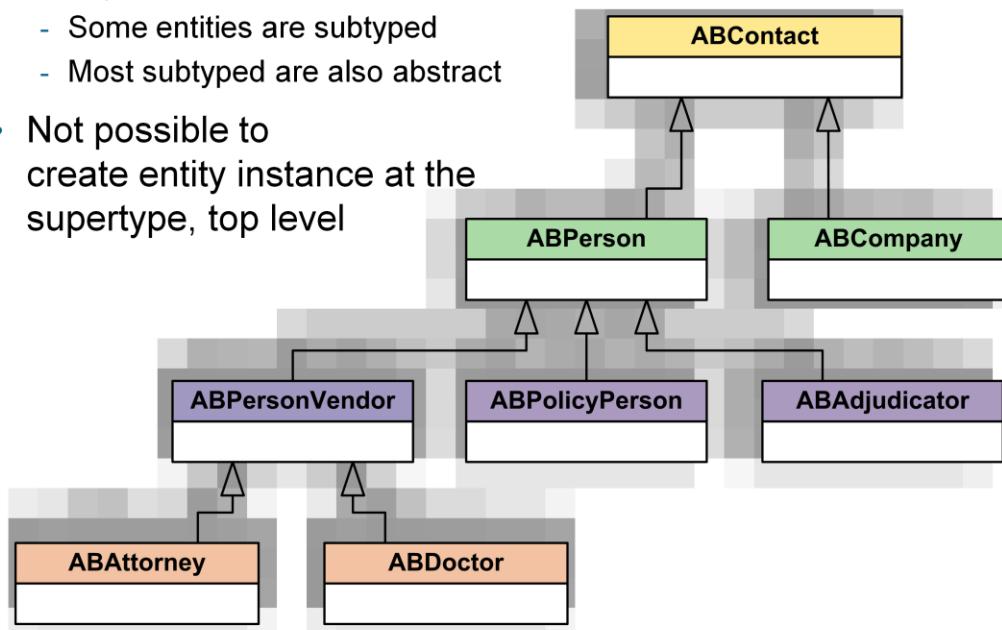
There are two exceptions to the generalization above: virtual and subtype entities.

Virtual entities are entities constructed entirely with code. They exist during run-time, but the data inside them is neither read directly from the database nor written directly to the database. One example of this is BillingCenter's ProducerCodeRoleEntry entity. It is a simple wrapper entity used to support modifying the producer codes by policy role. Its data is assembled from other physical entities.

Subtyped entities are entities that share a parent/child relationship. All fields in the parent entity are inherited by the child entity. A top-level subtyped entity and all of its child subtypes are stored in a single database table, as opposed to each having its own table.

Subtyped entities

- Subtype inherits all fields of its parent
 - Some entities are subtyped
 - Most subtyped are also abstract
- Not possible to create entity instance at the supertype, top level



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

7

GUIDEWIRE

The only major subtyped entity in TrainingApp is **ABContact**, which is abstract. This means that you cannot create instances of **ABContact** at the **ABContact** level. You can create instances at the levels below **ABContact** (such as instances of **ABPerson** or **ABCompany** or **ABPlace**).

- Major subtyped entities in ClaimCenter include: Contact, Incident, Transaction.
- Major subtyped entities in PolicyCenter include: Contact, Job, Modifier, and Workflow.
- Major subtyped entities in BillingCenter include: Contact, Activity, Plan, Invoice, and Workflow.

Subtype entity data

- Supertype parent table stores all instances of itself AND of subtype data
 - Contains parent fields and all subtype fields
 - Irrelevant fields are null for specific subtypes
 - Subtype column identifies subtype
- Example:
 - Firstname and lastname are always null for subtype = 3

SELECT ID, TAXID, NAME, FIRSTNAME, LASTNAME, VENDORTYPE, SUBTYPE FROM AB_ABCONTACT order by SUBTYPE;						
ID	TAXID	NAME	FIRSTNAME	LASTNAME	VENDORTYPE	SUBTYPE
71	3432-32-431134-23-2343	null	James	Smythe	null	1
72	null	null	Paul	Peterson	null	1
67	2096542-3113-2456902	null	Lily	Watson	null	2
68	9983200-5335-0023899	null	James	Andersen	null	2
75	1242577-7777-7752421	Express Auto	null	null	null	3
76	3219251-8888-1529123	European Autoworks	null	null	null	3
77	5040392-9999-2930405	M B Garage	null	null	null	3
78	5647382-7777-2837465	Burlingame Saab	null	null	null	3
79	5646372-4444-2736465	Menlo Park Chevron	null	null	null	3

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

8

G U I D E W I R E

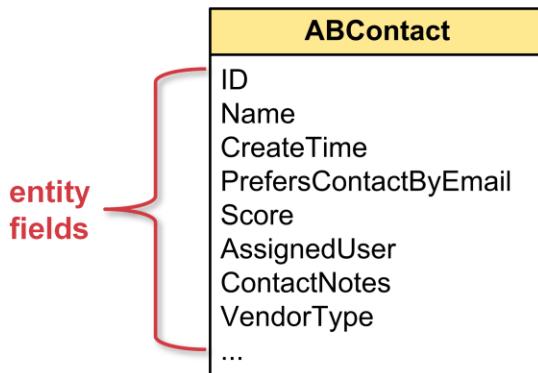
The ab_abcontact table is the parent table or supertype table that stores all instances of ABContact and its subtypes.

The ABContact table has columns for all fields defined at the parent level and therefore relevant to all child subtypes (such as TAXID). It also has columns for all fields defined at any of the subtype levels. This includes columns defined at the ABPerson level (such as FIRSTNAME and LASTNAME) and columns defined at the ABCompany level (such as NAME). Null values appear for columns not relevant to a given row's subtype.

A subtype column row value of 3 is for the subtype ABAutoRepairShop.

Entity fields

- An **entity field** is a value (or set of values) used to define the state or nature of a specific instance of the entity
 - Example: ABContact's Name field
- Four general types of entity fields
 - Data
 - Foreign key
 - Array key
 - Typekey



ABContact
ID
Name
CreateTime
PrefersContactByEmail
Score
AssignedUser
ContactNotes
VendorType
...

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

9

G U I D E W I R E

Data fields

- A **data field** stores a single value that does not reference any other object or table
- Examples of single values:
 - Name is a String
 - CreateTime is a datetime
 - PrefersContactByEmail is a bit
 - Score is an integer

ABContact	
ID	
Name	
CreateTime	
PrefersContactByEmail	
Score	
AssignedUser	
ContactNotes	
VendorType	
...	

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

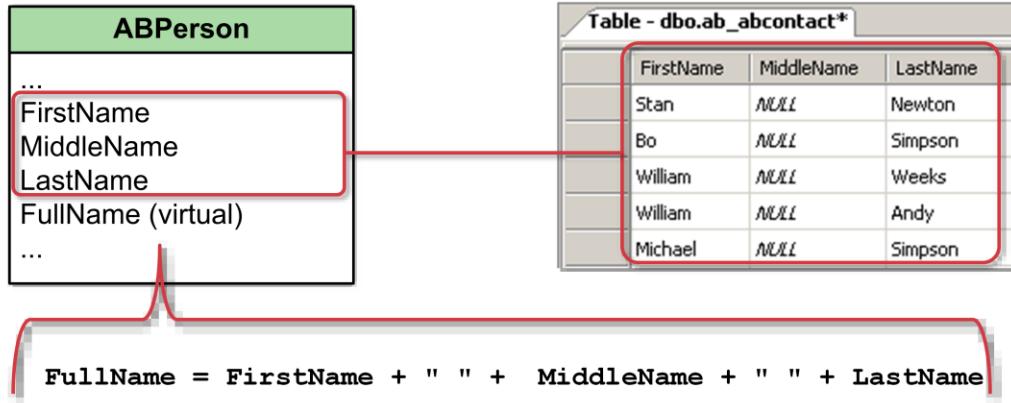
10

G U I D E W I R E

Data fields are defined in XML for an entity file with a <column> element. Therefore, they are sometimes referred to as "column fields". There are other types of fields beyond primitive value fields that are stored in database table columns, however. So one should not assume that the only fields that map to database columns are the "column fields". This course uses the term "data fields" to avoid this possible point of confusion.

Data fields in the database

- Data fields can be physical or virtual
 - Data model defines physical fields as columns in a database table
 - Code defines virtual fields and there is no physical database column



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

11

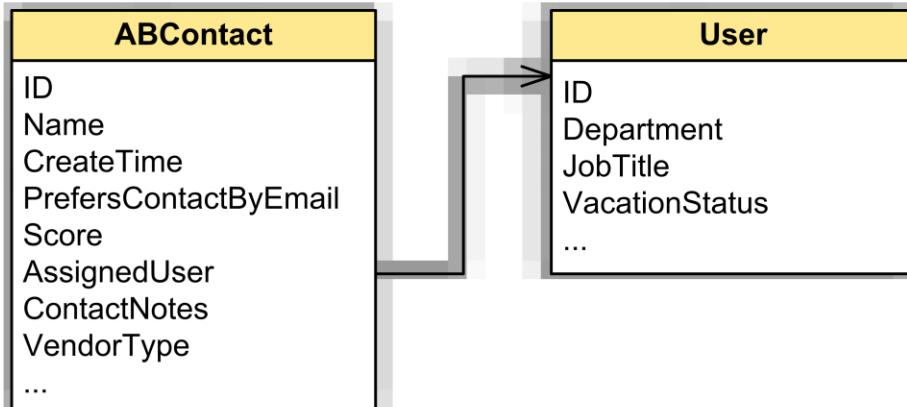
G U I D E W I R E

In the example above, the FullName field is a virtual field or property. FullName concatenates the FirstName, MiddleName, and LastName.

Entity enhancements and are discussed in another lesson.

Foreign key fields

- A **foreign key field** stores a reference to a related object in the data model
- AssignedUser in ABContact is the foreign key field for User



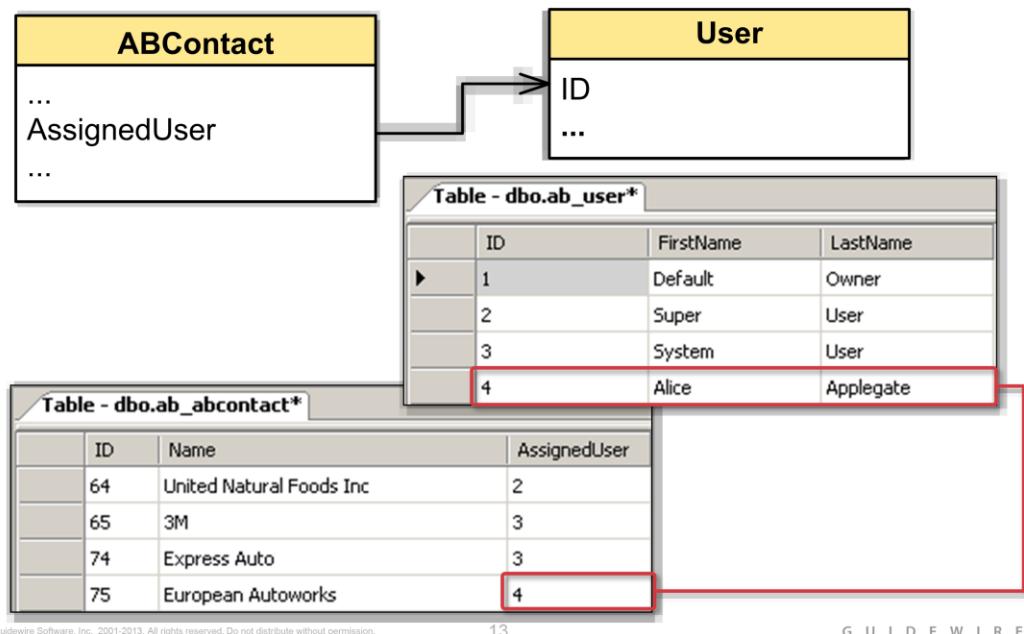
© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

12

G U I D E W I R E

Foreign key fields in the database

- Foreign key fields are stored as foreign key columns



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

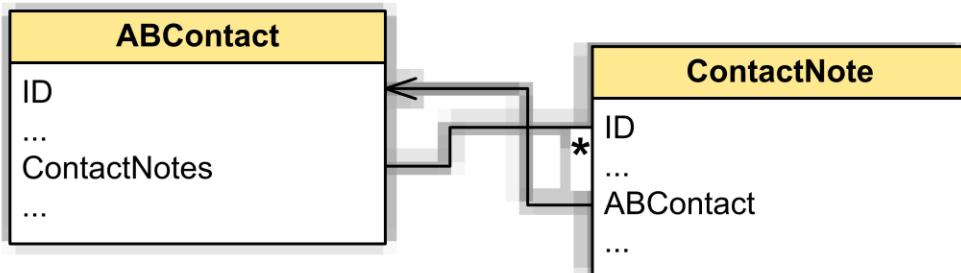
13

G U I D E W I R E

In the slide example, the assigned user (4) for European Autoworks is Alice Applegate. The assigned user (3) for 3M and Express Auto is System User. The assigned user (2) for United Natural Foods Inc is Super User.

Array key fields

- An array key field stores references to a set of related objects in the data model
- Not stored in database
- Populated at runtime by queries



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

14

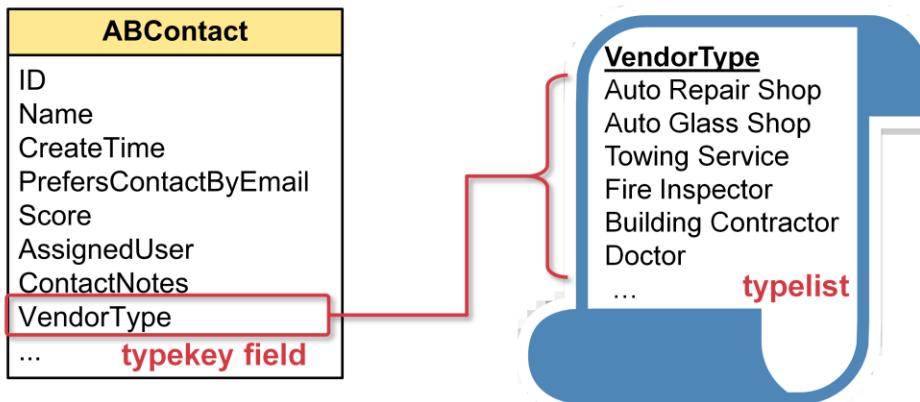
G U I D E W I R E

Array keys are managed during run-time by code that queries the database for objects of a given type with foreign keys that point to the array's parent.

For example, assume you have an ABContact object named ericAndy that stores the ABContact data for Eric Andy. Whenever you reference EricAndy.ContactNotes, Guidewire queries the database for all ContactNote objects whose ABContact field points to Eric Andy and returns those objects in an array.

Typekey fields and typelists

- A **typelist** is a predefined list of values that constrains a field
- A **typekey** field is a field associated with a specific typelist
- The typelist defines the possible values of the typekey field



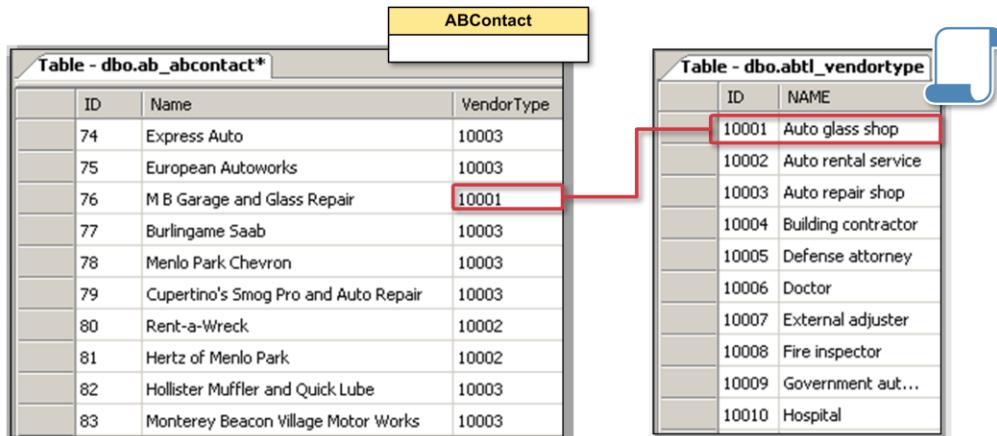
© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

15

G U I D E W I R E

Typelists in the database

- Each type list is stored in its own table
- Each typekey is a foreign key a single row in a type list table



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

16

G U I D E W I R E

Type list tables are named <applicationcode>tl_<typelistname>.

Subtype typelists

- Subtyped entity automatically creates typelist table of all subtypes
 - Table of all subtypes for the supertype parent
 - ID integer value uniquely identifies the subtype
 - Typecode describes subtype
- Subtype column in the supertype table points to the unique ID row in the typelist table

SELECT ID, TYPECODE FROM ABTL_ABCONTACT	
ID	TYPECODE
1	ABAdjudicator
2	ABAAttorney
3	ABAAutoRepairShop

SELECT ID, TAXID, NAME, FIRSTNAME, LASTNAME, VENDORTYPE, SUBTYPE FROM AB_ABCONTACT order by SUBTYPE;					
ID	TAXID	NAME	FIRSTNAME	LASTNAME	VENDORTYPE
71	3432-32-431134-23-2343	null	James	Smythe	null
72	null	null	Paul	Peterson	null
67	2096542-3113-2456902	null	Lily	Watson	null
68	9983200-5335-0023899	null	James	Andersen	null
75	1242577-7777-7752421	Express Auto	null	null	null
76	3219251-8888-1529123	European Autoworks	null	null	null
77	5040392-9999-2930405	M B Garage	null	null	null
78	5647382-7777-2837465	Burlingame Saab	null	null	null
79	5646372-4444-2736465	Menlo Park Chevron	null	null	null

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

17

G U I D E W I R E

Whenever an entity is subtyped, a typelist is automatically created for it. This typelist contains one typecode for every subtype of the parent entity. Typelist tables are prefixed with "xxtl_ ", where "xx" is the two-letter application code.

The ab_abcontact table is the parent or supertype table that stores all instances of ABContact and its subtypes.

Data model configuration tasks

- Creating new entities
- Extending existing entities
- Creating typelists
- Extending existing typelists
- Following lessons discuss data model configuration tasks



Lesson outline

- Contents of the data model
- The Data Dictionary
- Objects and the data model

The Data Dictionary



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

20

G U I D E W I R E

- The **Data Dictionary** documents the entities and typelists in your application
 - Collection of HTML pages
 - Requires that you regenerate data dictionary
- Two primary sections:
 - Data Entities
 - Typelists

The data dictionary is located at: <install directory>\build\dictionary\data\index.html. The entity pages of the Data Dictionary make use of active content.

Following are important notes regarding active content in different browsers:

If you are using Google Chrome, you will not be able to view active content (for example, clicking on the question marks will not show anything).

If you are using Internet Explorer (IE) - IE blocks active content and displays a warning bar that states "Internet Explorer has restricted this webpage from running scripts or ActiveX controls...". You can right-click the alert bar to allow the blocked content to run, but you have to do this every time you navigate to the Data Dictionary. Alternately, you can allow all active content in HTML pages on your machine to run automatically by doing the following:

- Select Tools > Internet Options from the menu bar.
- Click the Advanced tab.
- Scroll down to the Security section. Enable the "Allow active content to run in files on 'My Computer'" option.
- Restart Internet Explorer.

The Data Dictionary's conversion view is designed for data model work when converting data from a legacy application. This view provides a subset of the information in the standard view of the application entities that is more useful for those working on the conversion of legacy data. For more information on the conversion view, refer to the Configuration Guide for your product.

Entities in the Data Dictionary

- Entities section lists each entity and information about it
- To view information about given entity, click its name
- Hyperlink document is fully navigable

The screenshot shows the Guidewire Data Dictionary interface. On the left, there is a sidebar with a list of entities: Home, ABAdjudicator, ABAttorney, ABAutoRepairShop, ABAutoScrapYard_Ext, ABAutoTowingAgcy, ABCompany, ABCOMPANYVendor, ABContact (highlighted with a red box), ABContactAddress, ABContactCategoryScore, ABContactContact, ABContactScoringWorkItem, ABContactSearchCriteria, ABContactSearchCriteriaSpec, ABContactSearchCriteriaTag, ABContactSpecialistService, ABContactTag, ABDoctor. Below this list is a copyright notice: © Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

ABContact (ab_abcontact) (delegates to [ABLinkable](#), [CommonContact](#), [EventAware](#), [GlobalContactName](#), [HistoryEventContainer](#), [Retireable](#), [Validatable](#))

► Description

Attributes: Abstract?, Editable?, Exportable?, Extendable?, Keyed?, Loadable?, Sourceable?, Supertype?, Versionable?

Messaging Events: The application will generate these events for this entity:

- ABContactRemoved
- ABContactAdded
- ABContactChanged
- ABContactPendingChangeRejected
- ABContactResync

Integration programmers can write Event Fired rules to perform actions on these events. Refer to the [Integration Guide](#) for the complete list of events for all entities. Remember to configure a new or existing messaging destination in Studio to listen for desired events.

► Concrete FK references:
[Show user interface details on all fields](#)

GUIDEWIRE

To view the Data Entities section of the data dictionary, click the "Data Entities" link on the Data Dictionary home page.

Entity header

ABContact (ab_abcontact) (delegates to [Validatable](#), [CommonContact](#), [ABLinkable](#), [HistoryEventContainer](#), [Retireable](#), [EventAware](#))

► Description

Attributes: Abstract[?], Editable[?], Exportable[?], Extendable[?], Keyed[?], Loadable[?], Sourceable[?], Supertype[?], Versionable[?]

- Entity name and name of database table
- Delegate entities (if any)
 - Delegates define fields and behaviors needed by multiple entities, such as "Assignable" or "ABLinkable (to other apps)"
- Entity attributes (for definition of each, click " ? ")
- Other entities with foreign key and array references to this entity

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

22

G U I D E W I R E

A delegate is an abstract entity that defines a set of fields and methods needed by other entities to enable a specific type of behavior. Any entity that needs the behavior "delegates to" the delegate entity. For example, ClaimCenter has four entities that can be assigned to users: Claim, Exposure, Activity, and Matter. It also has an "Assignable" delegate entity that contains the fields needed for assignment (such as AssignedUser and AssignedGroup) and methods. The Claim, Exposure, Activity, and Matter entities all delegate to this entity and therefore all have these fields and methods.

The Data Dictionary lists delegates using the term "delegates to". However, when you configure one entity to reference another entity as a delegate, the keyword that you use is "implements". Therefore, the relationship between an entity and its delegate entities can also be described with "implements", as in "ABContact implements CommonContact and ABLinkable".

Subtypes list

Subtypes	
Code	Name
ABCCompany	ABCCompany
ABCCompanyVendor	ABCCompanyVendor
ABAAutoRepairShop	ABAAutoRepairShop
ABAAutoScrapYard_Ext	ABAAutoScrapYard_Ext
ABAAutoTowingAgcy	ABAAutoTowingAgcy
ABLawFirm	ABLawFirm
ABMedicalCareOrg	ABMedicalCareOrg
ABPolicyCompany	ABPolicyCompany
ABPerson	ABPerson
ABAAdjudicator	ABAAdjudicator
ABPersonVendor	ABPersonVendor
ABAAttorney	ABAAttorney
ABDoctor	ABDoctor
ABPropInspector_Ext	ABPropInspector_Ext
ABPolicyPerson	ABPolicyPerson
ABUserContact	ABUserContact
ABPlace	ABPlace
ABLEgalVenue	ABLEgalVenue

- If entity is subtyped, it has subtypes list
- Identifies all child subtypes
- Includes links to each subtype

Fields

name	data type	properties	description
Fields			
CreateTime datetime <small>(non-null)[?]</small>			foreign keys have link to related entity
			Timestamp when the object was created
CreateUser foreign key to User <small>(database column: CreateUserID)</small>			expandable list of UI files that display field (if field is used in UI)
			User who created the object
FaxPhone phone (30) <small>(exportable)[?] (loadable)[?] (writable)[?]</small>			extensions to entity in dark blue
			Fax number associated with the contact.
		► Show user interface details...	
FraudInvestigations_Ext integer <small>(exportable)[?] (loadable)[?] (writable)[?]</small>			virtual properties identified
			Number of fraud investigations
PrimaryPhoneValue Derived property returning java.lang.String <small>(virtual property)[?]</small>			
			Gets the value of the Contact's primary phone number.
VendorType type key to VendorType <small>(exportable)[?] (writable)[?]</small>			typekeys have link to typelist
			The company's vendor type.

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

24

G U I D E W I R E

Both foreign keys and typekeys include links to the related entity or typelist.

If a property is virtual, then its data type is listed as "Derived property returning <data type>" and it has the "virtual property" property. In the example above, PrimaryPhoneValue returns the home phone number if the contact's primary phone is set to "home", returns the work phone number if the contact's primary phone is set to "work", and returns the fax phone number if the contact's primary phone is set to "fax".

Virtual fields are displayed in green, whether they are extensions or part of the base application.

Arrays

- Each array includes a link to an entity stored within an array
- Data dictionary shows the linkage

The screenshot shows the Guidewire Data Dictionary interface. At the top, there is a green header bar with the title "Arrays". Below it is a list of array entities:

- BankAccounts** array key for [BankAccount](#) (exportable)? (triggers validation)? (writable)?
Bank accounts
- Buildings_Ext** array key for [Building_Ext](#) (exportable)? (writable)?
Buildings
- ContactNotes** array key for [ContactNote](#) (exportable)? (writable)?
Notes
- FlagEntries** array key for [FlagEntry](#) (exportable)? (writable)?
Flag entries
- HistoryEntries** array key for [HistoryEntry](#) (exportable)? (writable)?
History entries
- LegalCases** array key for [LegalCase](#) (exportable)? (writable)?
Legal cases

A red box highlights the "ContactNotes" entry. Below this, a detailed view of the "ContactNote" entity is shown in a separate window:

ContactNote (abx_contactnote) (delegates to [Retireable](#))

► Description

Attributes: Editable?, Exportable?, Extendable?, Final?, Keyed?, Loadable?, Sourceable?, Versionable?

▼ Array references:
[ABContact.ContactNotes](#)

Show user interface details on all fields

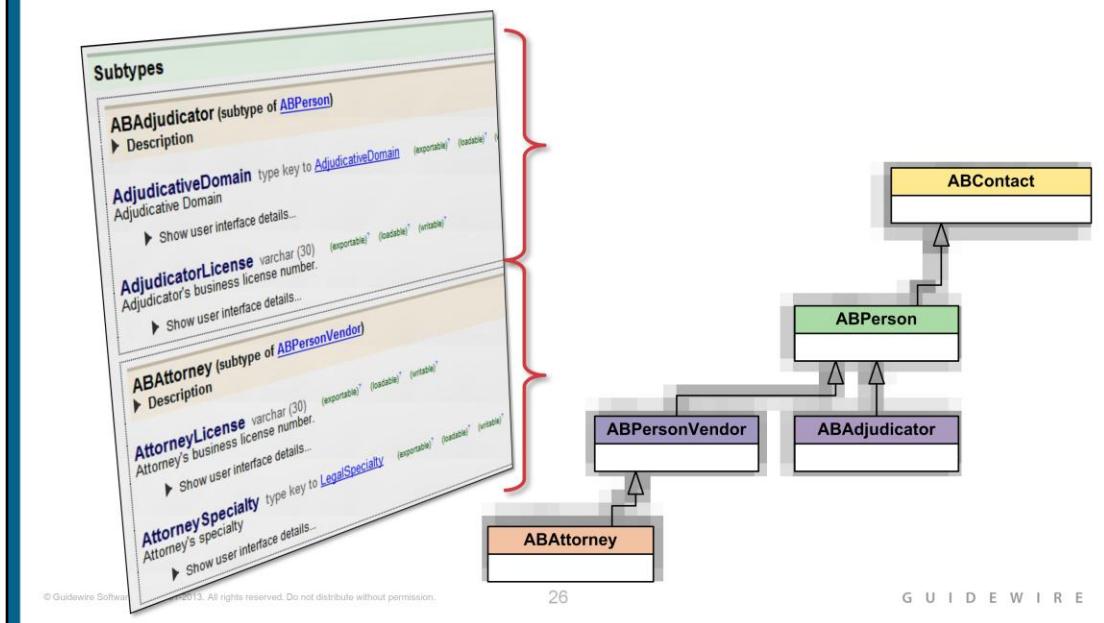
© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

25

G U I D E W I R E

Subtype fields

- If entity is subtyped, fields and arrays declared at each subtype level are listed in their own section



TypeLists in the Data Dictionary

- TypeLists section lists each typeList and its typecodes
 - To view information about given typeList, click its name

The screenshot shows the Guidewire Data Dictionary interface. On the left, a sidebar lists various typeList names. The 'DoctorSpecialtyType' entry is highlighted with a red box and selected, which triggers the main panel to display detailed information about it. The main panel has a title 'DoctorSpecialtyType (abtl_doctorspecialtytype)'. It includes sections for 'Attributes', 'Referenced by' (listing 'ABDoctor Specialty'), and 'Typecodes'. The 'Typecodes' section contains a table with the following data:

Code	Name	Description	Priority	Internal	Retired
CriticalCareMedicine	Critical Care Medicine	Critical Care Medicine			
Dentistry	Dentistry/Oral Surgery	Dentistry/Oral Surgery			
EmergencyMedicine	Emergency Medicine	Emergency Medicine			
FootSurgery	Foot Surgery	Foot Surgery			
GeneralSurgery	General Surgery	General Surgery			

To view the TypeLists section of the data dictionary, click the TypeLists link on the Data Dictionary home page.

You can view the typecode and name for all elements in the list. You can also access typeLists from the main page or from a typekey field.

All fields

The screenshot shows the Guidewire ContactManager Data Dictionary interface. The main window displays the Guidewire logo and the title "ContactManager 8.0.0.908 Data Dictionary". Below the title, it says "Entity model version 224 - Built on Aug 22, 2013 at 2:22:15 PM". Under "Data Entities", there are links for "Data Entities (Database View)" and "Data Entities (Migration View)". A red arrow points from the "All fields" link in the "TypeLists" section to the "All fields" section on the right. The "All fields" section contains the text "(an index of field names to entities containing them)". On the right side, there is a vertical list of entity types with their corresponding attribute fields:

- Attributes**
[User](#)
- Attribute Type**
[AttributeCriteriaElement](#)
- AttributeValue**
[AttributeCriteriaElement](#)
- Author**
[Document](#)
[DocumentSearchCriteria](#)
[Note](#)
[NoteSearchCriteria](#)
- AuthorDenorm**
[Document](#)
- AuthoringDate**
[Note](#)
- AutoGenerated**
[Activity](#)

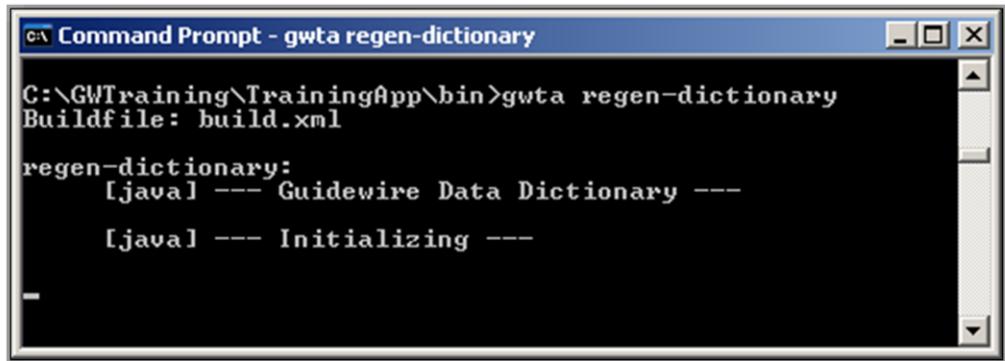
At the bottom left, there is a copyright notice: "© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission." At the bottom center, it says "28". At the bottom right, it says "GUIDEWIRE".

In the example in the screenshot on the right:

- There is an "Attributes" field on the User entity.
- There is an "AttributeType" field on the AttributeCriteriaElement entity.
- There is an "AttributeValue" field on the AttributeCriteriaElement entity.
- There is an "Author" field on the Document, DocumentSearchCriteria, Note, and NoteSearchCriteria entities.

...and so on.

Generating the data dictionary



```
C:\ Command Prompt - gwta regen-dictionary
C:\GWTTraining\TrainingApp\bin>gwta regen-dictionary
Buildfile: build.xml

regen-dictionary:
    [java] --- Guidewire Data Dictionary ---
    [java] --- Initializing ---
```

- You can (re)generate Data Dictionary:
 - After initial install (Data Dictionary is not pre-generated)
 - Any time you extend data model
- To regenerate dictionary, from bin directory, execute:
 - **gwXX regen-dictionary**
 - where **XX** is application's two-letter code

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

29

G U I D E W I R E

The codes for each application are:

- TA for TrainingApp
- CC for ClaimCenter
- PC for PolicyCenter
- BC for BillingCenter
- AB (address book) for ContactManager

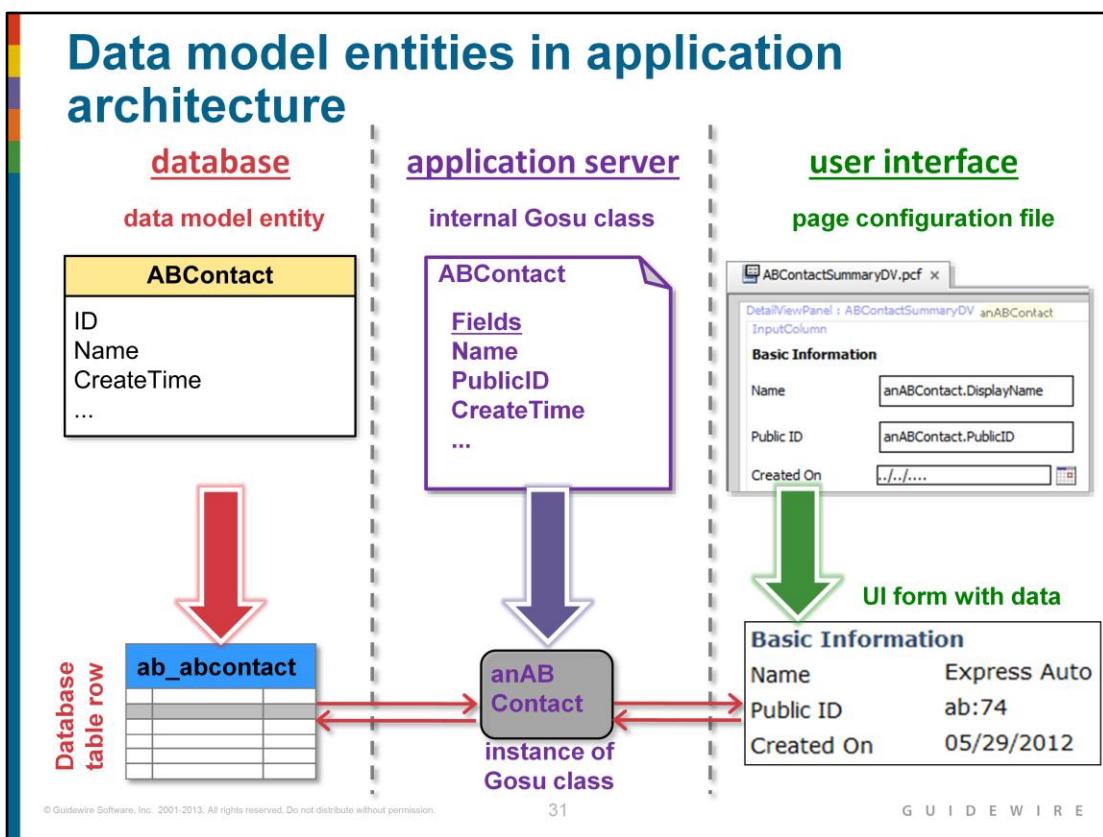
The regen-dictionary command also regenerates the Security Dictionary. This dictionary lists all system permissions and other information that pertains to controlling access to a Guidewire application.



Lesson outline

- Contents of the data model
- The Data Dictionary
- Objects and the data model

Data model entities in application architecture



For every data model entity, the application automatically creates an internal Gosu class with the same name. For every field in the data model entity, there is a field in the corresponding internal Gosu class. For example, the ABContact data model entity has a "FaxPhone" field, and the internal ABContact Gosu class also has a "FaxPhone" field.

The Gosu classes that map to data model entities are internal and cannot be manipulated directly. However, it is possible to add things to them indirectly.

You can add fields to base application entities as discussed in the "Extending Base Entities" lesson.

You can add fields to a custom entity as discussed in the "Creating New Entities" lesson.

You can add methods to any entity as discussed in the "Enhancements" lesson.

Gosu classes that are automatically created from data model entities are sometimes referred to as "database-backed classes" because these classes have corresponding database tables, and instances of these classes can be saved to the database. Whenever a row is read from a database table, an instance of the corresponding Gosu class is created, and the data is placed into that instance. The data in this instance can then be displayed in the user interface and modified by the user prior to saving the data. Whenever the data in a run-time instance of a database-backed class is changed and saved, that data gets written to the corresponding database table. If the data was initially read from the database (such as when a user looks for an existing ABContact and then modifies it), the existing database row is updated to reflect the changes. If the data was created entirely in the run-time environment (such as when a user creates a new ABContact), then a new row is inserted into the database table.

It is also possible to create Gosu classes that are not database-backed. Instances of those classes can be created and modified during runtime, but they cannot be saved to the database. These types of classes are discussed in detail in the "Gosu Classes" lesson.

Dot notation

- **Dot notation** is a Gosu language syntax used to identify data objects and properties
 - Starts with object and ends with field, related object, or related array
- Relies on data model
 - Not used to configure data model
 - Available for User Interface configuration and Gosu classes
- Examples include data fields, object collections, and methods

The screenshot shows the entity definition for 'ABContact'. It includes sections for 'Description', 'Fields', and 'Arrays'. The 'Fields' section contains a table with columns for name, type, description, and various metadata like exportability and loadability. One row is highlighted with a red box: 'FaxPhone phone (30)'.

Name	Type	Description	Metadata
AssignedUser	foreign key to User	(exportable)* (loadable)* (writable)	Assigned user
FaxPhone	phone (30)	(exportable)* (loadable)* (writable)	Fax number associated with the contact.
PreferredCurrency	type key to Currency	(exportable)* (loadable)	The contact's preferred currency.

anABContact.FaxPhone

Dot notation is a language syntax used to identify data elements such as text fields, related objects or collections of objects. The syntax is not used in data model configuration itself. It is useful to mention at this point in time because the data model definition informs the possible dot notation expressions for referencing entity data in code.

Dot notation syntax

- Referencing data field on given object
 - `object.field`

`anABContact`

`FaxPhone`

`anABContact.FaxPhone`

- Referencing field on related object
 - `object.foreignKey.field`

`anABContact`

(User object)

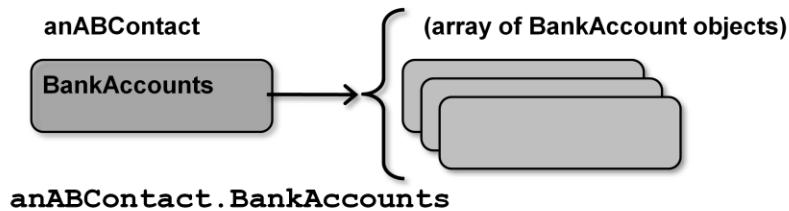
`AssignedUser`

`JobTitle`

`anABContact.AssignedUser.JobTitle`

Dot notation syntax (2)

- Referencing array on given object
 - `object.array`



- Referencing field at subtype level
 - `(object as subtype).field`



© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

34

G U I D E W I R E

Referencing fields at the subtype level is discussed in detail in the "Subtypes" lesson and in the "Atomic Widgets" lesson.

Lesson objectives review

- You should now be able to:
 - Describe the contents of a Guidewire data model
 - Identify information about a given application's data model
 - Reference entity fields using dot notation

Review questions

1. For each of the following, identify if it is stored as a database table, as a table column, or not stored in the database at all:
 - a) An entity (such as ABContact)
 - b) A physical data field (such as ABContact.CreateTime)
 - c) A virtual data field (such as ABContact.FullName)
 - d) A foreign key field (such as ABContact.AssignedUser)
 - e) An array key field (such as ABContact.ContactNotes)
 - f) A typelist (such as VendorType)
 - g) A typekey field (such as ABContact.VendorType)
2. Name two circumstances in which you would execute the regen-dictionary command.

(continued)

Answers

- 1a) A table
1b) A column
1c) Not stored in the database (Technically, the source values are probably stored in the database. But the value of a virtual field itself is not stored as a separate value.)
1d) A column (specifically a foreign key column to the relevant entity's table)
1e) Not stored in the database
1f) A table
1g) A column (specifically a foreign key column to the typelist's table)
2. Possible answers: You would execute regen-dictionary when you install the application and whenever you extend the data model. You could also execute the command when you modify system permissions or some other aspect of access and therefore need to regenerate the Security Dictionary.

Review questions

ABContact (ab_abcontact) (delegates to
► Description

Fields

AssignedUser foreign key to [User](#) (exportable)* (loadable)
Assigned user

FaxPhone [phone](#) (30) (exportable)* (loadable)* (writable)*
Fax number associated with the contact.

PreferredCurrency type key to [Currency](#) (exportable)*
The contact's preferred currency.

Arrays

ContactNotes array key for [ContactNote](#) (exportable)* (loadable)
Notes

ABPerson (subtype of ABContact)
► Description

CellPhone [phone](#) (30) (exportable)* (loadable)* (writable)*
Mobile phone number associated with the contact.

DateOfBirth datetime (exportable)* (loadable)* (writable)*
Date of birth.

3. For the object "myContact" of type ABContact, what is the dot notation for the object's:
- Preferred currency?
 - Notes?
 - Level of experience of myContact's assigned user?
 - Mobile phone (if it is a person)

User (ab_user)

► Description

Fields

BackupUser foreign key to [User](#) (virtual property)* (writable)*
Returns the backup user for this user

ExperienceLevel type key to [UserExperienceType](#) (exportable)* (loadable)* (writable)*
Experience level of the user.

JobTitle [shorttext](#) (255) (exportable)* (loadable)* (writable)*
User's job title.

© Guidewire Software, Inc. 2001-2013. All rights reserved. Do not distribute without permission.

37

G U I D E W I R E

Answers

- myContact.PreferredCurrency
- myContact.ContactNotes
- myContact.AssignedUser.ExperienceLevel
- (myContact as ABPerson).CellPhone

Notices

Copyright © 2001-2013 Guidewire Software, Inc. All rights reserved.

Guidewire, Guidewire Software, Guidewire ClaimCenter, Guidewire PolicyCenter, Guidewire BillingCenter, Guidewire Reinsurance Management, Guidewire ContactManager, Guidewire Vendor Data Management, Guidewire Client Data Management, Guidewire Rating Management, Guidewire InsuranceSuite, Guidewire ContactCenter, Guidewire Studio, Guidewire Product Designer, Guidewire Live, Guidewire ExampleCenter, Gosu, Deliver Insurance Your Way, and the Guidewire logo are trademarks, service marks, or registered trademarks of Guidewire Software, Inc. in the United States and/or other countries. Guidewire products are protected by one or more United States patents.

This material is Guidewire proprietary and confidential. The contents of this material, including product architecture details and APIs, are considered confidential and are fully protected by customer licensing confidentiality agreements and signed Non-Disclosure Agreements (NDAs).

This file and the contents herein are the property of Guidewire Software, Inc. Use of this course material is restricted to students officially registered in this specific Guidewire-instructed course, or for other use expressly authorized by Guidewire. Replication or distribution of this course material in electronic, paper, or other format is prohibited without express permission from Guidewire.