Freeminders Data Structure Definition

This is to provide a high level explanation of the data structure and how it should be used. As we begin to use the structure and find problems, we will obviously change it to make it work.

There are many columns missing that need to be copied from the old data structure. No other tables should need to be copied, just columns. When adding the columns, be sure they are correctly named per the coding standards document.

|  |  |
| --- | --- |
| **User** | This is a standard Parse table with a few things added to it |
| userLocations | A relationship to the UserLocation table. A user will be able to maintain a list of locations, like an address book. |
| userContacts | A relationship to the UserContact table. A user will be able to maintain a list of contacts, think of a phone book. The contacts will be either email addresses or phone numbers (used for future SMS functionality) |
| userPurchases | A relationship to the UserPurchase table. All of the items a user has purchased can be found thu this relationship |
| remainingSMS | **Future Functionality** – A user will be able to buy a block of SMS messages. Whenever a Reminder triggers that has an SMS contact associated with it, the code will check if the user has any SMS “credits” remaining and if so, it will send an SMS message and then decrement this counter. |
| remainingEmail | The number of remaining email credits they have. Once this hits 0 and if they are \*not\* flag as having unlimited email, they will no longer get email reminders |
| hasUnlimitedEmail | If this flag is set, we do not use, nor do we decrement the remainingEmail count. The user has no limits to the email that is sent from Reminders. |

|  |  |
| --- | --- |
| **DateTimeTrigger** | DateTime related trigger information will be here.  Add columns as needed. |

|  |  |
| --- | --- |
| **LocationTrigger** | Location related trigger information will be here.  Add columns as needed. |
| userLocation | A pointer to an entry in the UserLocation table. This value is mutually exclusive with the field location. See UserLocation for more information. |
| location | A Parse GeoPoint specifying the Longitude and Latitude of a location. This is used for an ad-hoc location, that is, a location that is not in the UserLocation “address book” for the user. |

|  |  |
| --- | --- |
| **Reminder** | Indivdual Reminder information.  Add columns as needed. |
| name | The name of the reminder |
| dateTimeTriggers | A relationship to DateTimeTrigger for this reminder. |
| locationTriggers | A relationship to LocationTrigger for this reminder. |
| weatherTriggers | A relationship to WeatherTrigger for this reminder. |
| reminderSteps | Array of Pointers – Should point to ReminderStep objects |
| userContacts | A relationship to UserContact for this reminder. If this relationship is NULL, that means the application should use the contacts specified on the Reminder’s ReminderGroup. Think of it as inheritance. If Reminder does not explicity set the contacts to use, it will use the parent ReminderGroup. |
| isActive | Flag to indicate if the Reminder is currently active. |
| parentReminders | Array of Pointers – Exercise caution if you include/includeKeys when loading as you could get back a lot of data.  If there are 1 or more “parent reminders”, those reminders must be inactive (isActive = false) before this one can be active.  This value can be NULL, which means there is no relationship to other reminders. |
| note | text |

|  |  |
| --- | --- |
| **ReminderGroup** | A Freeminder Reminder Group  Add columns as needed. |
| name | The name of the group |
| userContacts | A relationship to the UserContact for this group. Unless overridden on the child reminder, whenever a child reminder triggers sends an email/sms notification, it will send to all those specified in this list. |
| reminders | Array of Pointers – Should point to Reminder objects |
| storeReminderGroup | Whenever a user “buys” a group from the store, a copy of that group, including reminders will be created. A pointer back to the original store reminder group will be added here so we know this is a copy of a store group, and, which group it is a copy of. |
| description | text |
| owner | Pointer to the User that owns this group. |

|  |  |
| --- | --- |
| **ReminderStep** | The reminder step table |
| name | The name of the step |
| order | The order number of the step. When listing steps for a reminder, they will be sorted on this. |
| isComplete | A flag indicating if the step is complete. |

|  |  |
| --- | --- |
| **StoreCategory** | A category, used to classify/categories an item in the store. |
| name | The category name |
| description | A description of the category |

|  |  |
| --- | --- |
| **StoreItem** | An item that will be listed in the in app store.  Add columns as needed |
| name | The display name of the item. |
| description | The description of the item. |
| isEnabled | A flag to indicate the item is visible in the store. |
| price | The price of the item. |
| salePrice | The sale price of the item. |
| storeCategories | A relationship to StoreCategory. The StoreItem can be in 1 or many categories. |
| reminderGroups | A relationship to ReminderGroup. This could be NULL, see other fields below. If populated, this is the list of ReminderGroups they get when they make the purchase. |
| countSMS | If this store item is a consumable, that is, buying it gives them, say 100 SMS messages, this property will tell the code how many remainingSMS needs to be incremented on the User object/table. |
| countEmail | A consumable number of email messages a user is allowed to send. This number gets added to remainingEmail in the User table. |
| unlimitedEmail | A flag that, if true, sets the hasUnlimitedEmail flag on the User table. Allows the user to send an unlimited number of email notifications. |

|  |  |
| --- | --- |
| **UserContact** | Think of this as a phone book. Initially it will allow the user to maintain a list of email addresses and phone numbers (for SMS). In the future, it also include device information for Push Notifications (future functionality) |
| name | A user defined name for this contact. |
| typeCode | 1 = email  2 = sms |
| value | If typeCode = 1, then this is an email address  If typeCode = 2, then this is a phone number (all formatting from stored phone number should be removed) |

|  |  |
| --- | --- |
| **UserLocation** | An “address” book. Currently used by LocationTrigger and WeatherTrigger. |
| name | A user entered name for the location |
| address | The address of the location.  Note: We may want to break this into address, city, state. |
| zipCode | A zip code if provided. I think WeatherTriggers need zipCode. If it is not needed, it can be removed. |
| location | A Parse GeoPoint. This will probably be either calculated based on the address/zipCode, or provided by the phone. |

|  |  |
| --- | --- |
| **UserPurchase** | A record of purchases made by the user thru the store. Add columns as needed. |
| storeItem | A pointer to the StoreItem table. Allows us to know what they purchased. |
| amountPaid | How much the custom paid for the item. |
| amountReceived | How much we got from the App Store for this. Note: Not sure if we can get that info from them, or if this is something we want. It may be removed. |
| receiptId | The receiptId from the App store |

|  |  |
| --- | --- |
| **WeatherTrigger** | Weather trigger related information. Add other columns as needed. |
| userLocation | A pointer back to named location in the users “address” book. This value is mutually exclusive. If it is defined, nothing else should be in the location or zipCode field. |
| location | A GeoPoint. Used for an ad-hoc location. |
| zipCode | Not sure if this is needed. |