# Assumptions/Constraints

For this program to work properly, certain assumptions must be true and certain constraints must be met.

1. No two contacts will share the same phone number. Every contact has a unique phone number.

2. Contact names (called person\_name in the database) can be the same.

# Database Design

The major issues I faced when trying to distinguish one contact from another were the following:

1. You may have more than one contact with exactly the same name.
2. A phone number can be shared among contacts.
3. Contact names can change.

These are things that my phone allows. However, for this program to work, I had to assume that contacts would not share the same phone number.

The key to overcoming these issues is that **the phone number a text message is sent from/to will never change. The timestamp and message text will never change**. The name of the contact that a message is sent from/to could change. So, in the event that I had received a text from “John” and his number is 234-456-7890, and then later I change his name to “Juan” and receive another text from him, the below entry would appear in the contacts table:

|  |  |
| --- | --- |
| **Person\_name** | **Phone\_number** |
| Juan | 234-456-7890 |

This is because changing a contact name will work retroactively on old texts in the XML file. When the XML file is created, it will update contact names accordingly.

Now suppose I had received a text from “Susan” and her phone number is 111-000-4444. I then change her phone number to 333-555-6767 and receive another text from her. I will have the below entries in the contacts table:

|  |  |
| --- | --- |
| **Person\_name** | **Phone\_number** |
| Susan | 111-000-4444 |
| Susan | 333-555-6767 |

This is because the phone number from which a text is sent can never change. While a contact’s name can be changed (even the name assigned to an old text), the phone number that a text is sent from cannot.

Now, suppose we have the situation in which both the name and the phone number are changed. We have a text from a contact named “Ashley” and her phone number is 098-123-2222. Then, I change her name to “Ashley Smith” and change her phone number to 999-888-7777. We will have the below two entries in the contacts table:

|  |  |
| --- | --- |
| **Person\_name** | **phone\_number** |
| Ashley Smith | 098-123-2222 |
| Ashley Smith | 999-888-7777 |

Why is this? It is because Ashley’s new name would appear in the XML file. But, the phone number that the old texts (texts sent prior to Ashley’s name and number change) wouldn’t change. It is always the same. Texts sent to/from Ashley before I update her phone number to 999-888-7777 will always show up in the XML file as coming from 098-123-2222 (although the contact name will change from “Ashley” to “Ashley Stevens”).

As is apparent by the above situation, certain assumptions will have to be made regarding how the user organizes his/her contacts:

1. When changing the name of a contact, the user does not create a new contact. So, in the John example above, the contact would update the “John” contact to “Juan” instead of having two contacts, one named “John”, and another named “Juan”.
2. If a contact’s phone number is changed, the old phone number is not deleted. Instead, it is left associated with the contact. This way, the old phone number as well as the new phone number will still be associated with the person’s name, as in the Susan example above. (This constraint need not apply to a phone number for which you have never texted or called. In that case, there would not be any records to change.)

How would we allow two contacts to have the same phone number? This is not a problem, until one of their names is changed. If both received or sent a text or phone call from me before the change, then all the program has to do is check the timestamps and figure out which contact’s name changed. Group text messages will not be affected by this strategy, because recipient contact names are not stored in group text messages (only phone numbers are).

Below is the schema for the tables.

**Contacts**(phone\_number, person\_name)

**Phone\_calls**(id, duration, call\_timestamp, contact\_phone\_number, call\_type)

**Text\_messages**(id, msg\_text, msg\_timestamp, sender\_phone\_number, text\_only)

**Text\_message\_recipients**(contact\_phone\_number, text\_message\_id)

**Last\_backup\_date\_time**(backup\_name, backup\_timestamp)

The text\_message\_recipients table will be the multiway relationship between text\_messages and contacts. This table is needed to store the recipient(s) of a text message. (If all texts had only one recipient, then the recipient phone number/name could simply be stored in the text\_messages table. However, you must also account for group texts. That is why I created the text\_message\_recipients table.)

The contacts table has the following constraint: all (phone\_number, person\_name) pairs must be unique. This table is in 2NF and 3NF, because a phone\_number does not determine the person\_name (the name of the contact), and person\_name does not determine phone\_number. It is possible that two people share the same phone number. Furthermore, it is possible that there are two contacts named “John”. These two columns combined represent a distinct contact.

(The columns person\_name, backup\_timestamp, backup\_name, msg\_text, and msg\_timestamp were so named to avoid using a SQL keyword.)

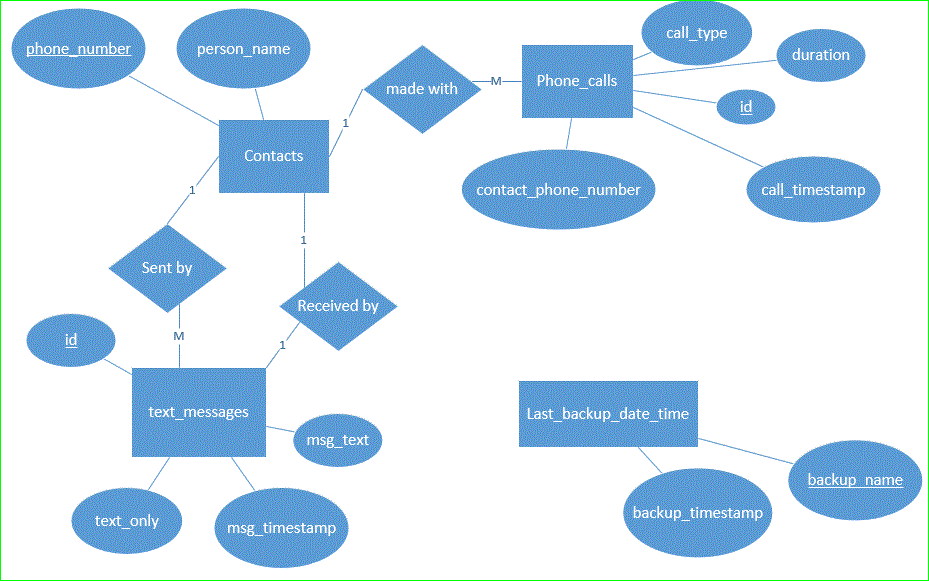


Figure Database Diagram

# Data Dictionary

**backup\_timestamp (timestamp)** – the time that the backup occurred.

**backup\_name (varchar)** – the type of backup performed at the time specified by the backup\_timestamp (acceptable values: “calls” or “texts”).

**call\_timestamp (timestamp)** – the date and time of a phone call. This is when the call first came through to my cell phone (if it was an incoming call) or when I first dialed the call (if it was an outgoing call).

**call\_type (tinyint)** – the type of phone call. If call\_type = 1, then the phone call was incoming and I picked up the phone; if call\_type = 2, then I dialed the number; call\_type = 3 means that the call was incoming and I did not answer the phone.

**contact\_phone\_number (bigint)** – foreign key pointing to the id of the contact that the phone call was made with.

**duration (integer)** – the duration, in seconds, of a phone call. For example: a duration of 131 would mean a phone call that lasted 2 minutes, 11 seconds.

**id (integer)** – the id of a table is an integer that auto increments. It is the primary key. The id of the contacts table, for example, will be unique and will distinguish each contact.

**msg\_text (text)** – the text contained in a text message.

**person\_name (string)** – this is the name of a contact. For example: “Phyllis Milton” could be a person\_name. The person\_name, in combination with the phone\_number, is a unique pair.

**phone\_number (bigint)** – the phone number is an integer. Every contact has at least one phone\_number. A phone\_number can be shared between two contacts. The phone\_number, in combination with the person\_name, is a unique pair.

**msg\_timestamp (timestamp)** – the date and time a text message was sent (for outgoing texts) or received (for incoming texts).

**sender\_phone\_number (bigint)** – foreign key pointing to the id of a contact that sent the text message. If I sent the text message, then this would be my id; if someone sent the text to me, then this would be their id.

**text\_message\_id (integer)** – the id of a text message. In the text\_message\_recipients table, this identifies the text message that a recipient received.