

# The Pharmacy Demo

---

## Overview

### Aim

The aim of this demo is to show how InterBase change views work using an example of a Pharmacy.

### How

Using a simple example of medicine data that is update centrally, and then collected by remote pharmacies as data deltas.

This is achieved by using 2 applications. 1) an application to modify the central data 2) a pharmacy application (with each pharmacy being identified by a unique ID in the pharmacy table) to collect the data.

## Example Details

### User story

The pharmacy use data from the pharmaceutical companies to create labels for the drugs being dispensed. It is important this data is correct.

While this data generally stays the same, there are times that some data is updated. There are a high number of distributed end points to get the data to.

There is a high volume of distributed end points, many connecting via cellular connections. To service each end point appropriately there needs to be consideration to update data quickly and as cheaply as possible.

### Requirements

To keep data costs to a minimum you need to reduce network traffic by passing only delta packets.

Knowing which sites are updated and which ones have not connected in for some time is important as this is life saving information.

### Solution

Central InterBase database with Change View created. This Change view is then connected to from the remote locations and used to collect the data from the central database.

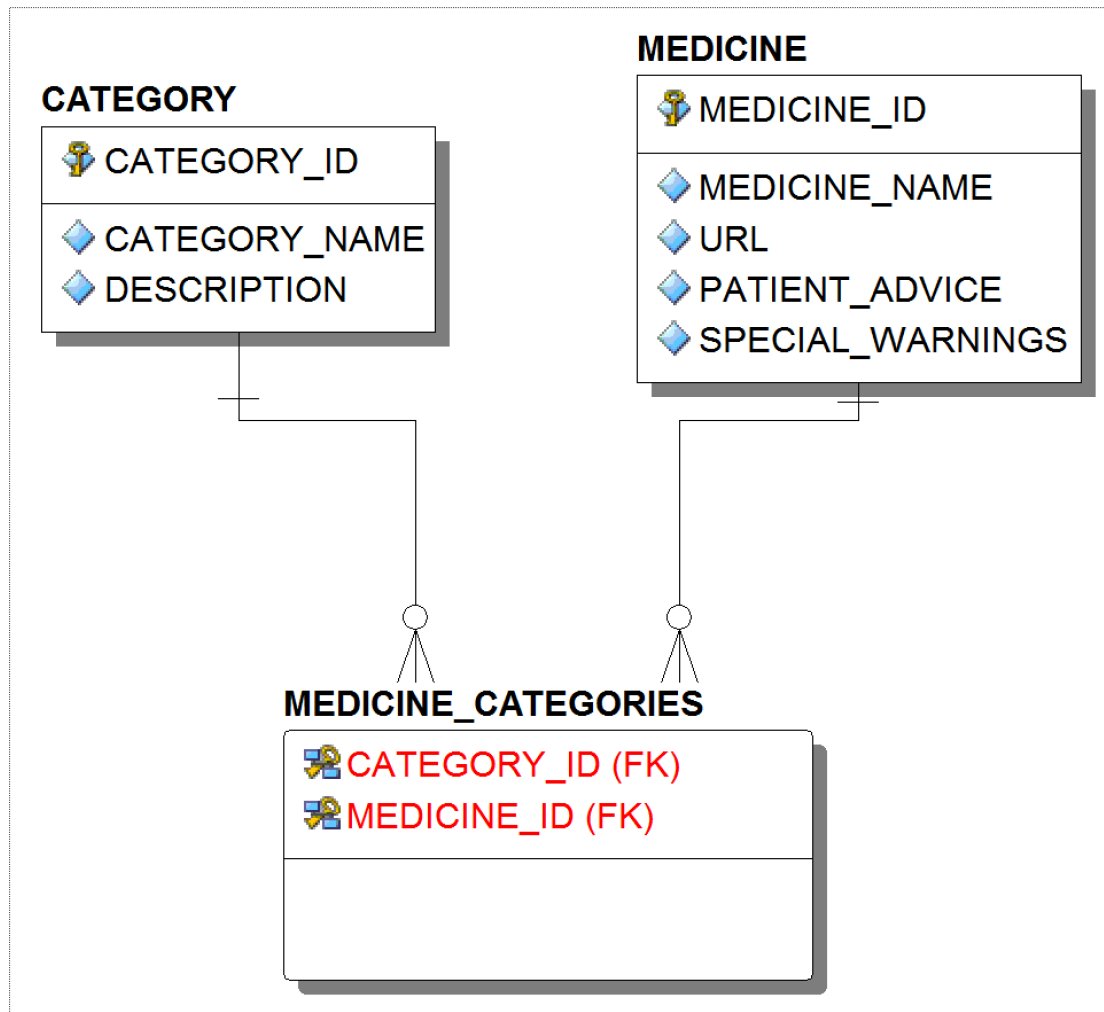
Change Views internal tables can be queried to find out which end points are up to date.

## Demo elements and setup

### Medicine Data Model

Medicine data is searched on through selecting a category to see a list of medicines in that category. Medicines can belong to more than 1 category.

This is achieved through a list of categories, a list of medicines and a link table that defines which category a medicine belongs to.



### Databases

The demo works with two databases.

1. Medicine.ib
2. Pharmacy.ib (there is also a Pharmacy2.ib with ID 2 for quick testing of two sites)

### Medicine.ib

The medicine database contains a change view that remote clients are able to subscribe to. The change view tracks inserts, updates and deletes to all 3 tables in this model.

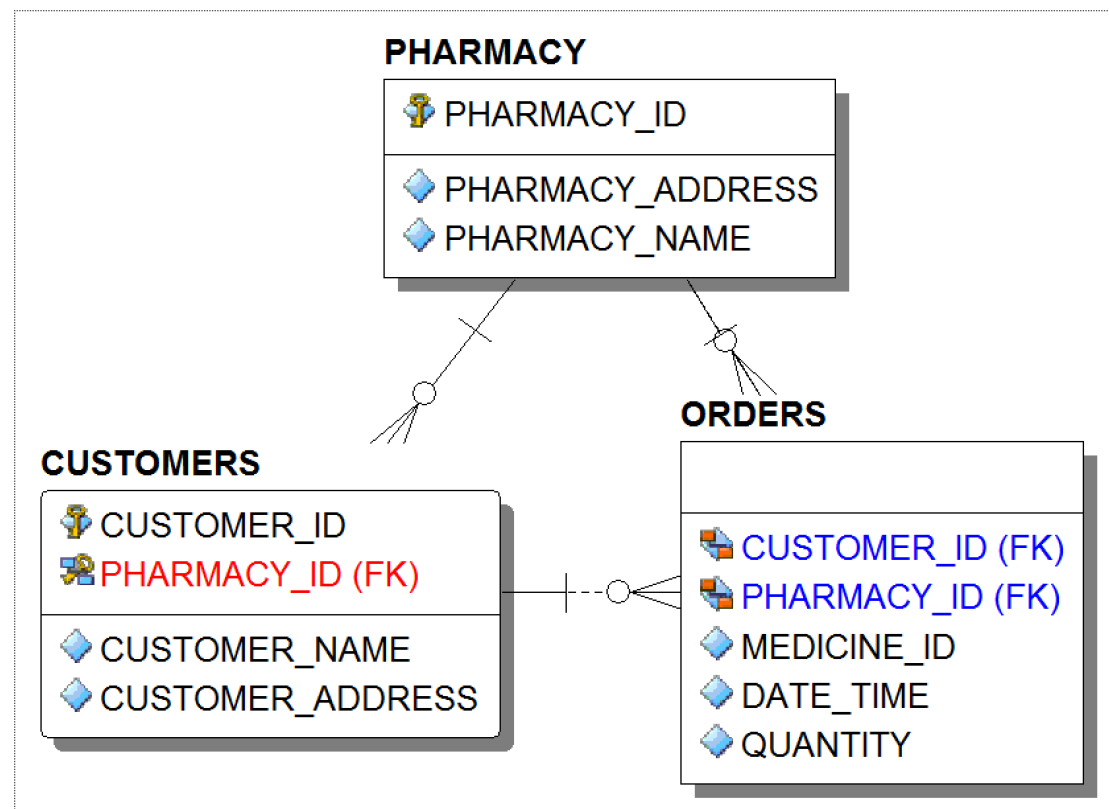
```

CREATE SUBSCRIPTION SUB_MEDICINEUPDATES ON
  CATEGORY FOR ROW (INSERT, UPDATE, DELETE),
  MEDICINE FOR ROW (INSERT, UPDATE, DELETE),
  MEDICINE_CATEGORIES FOR ROW (INSERT, UPDATE, DELETE)
DESCRIPTION 'Track inserts, updates and deletes to
medicine data';

```

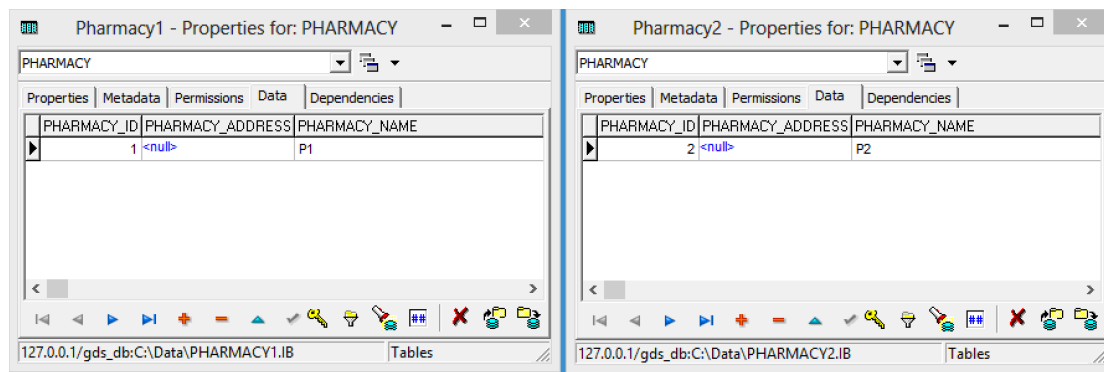
### Pharmacy.ib

For the purpose of this demo, the Pharmacy also has additional tables for collecting data for customers and orders and to identify the pharmacy itself. *Future work will be to refactor these for bi-di movement.*



A change view requires a <device id> to specify the end location asking for the changes. Using this ID it can provide custom delta's for multiple devices and they can be tracked centrally.

The PHARMACY\_ID is used by the example application to build the request sent to the medicine database when activating a Change View Subscription; Therefore to test with multiple pharmacies you need to, copy the Pharmacy.ib and update the PHARMACY.PHARMACY\_ID



The image above shows two connections to two databases (P1 and P2) – Database user/password is SYSBDA / masterkey.

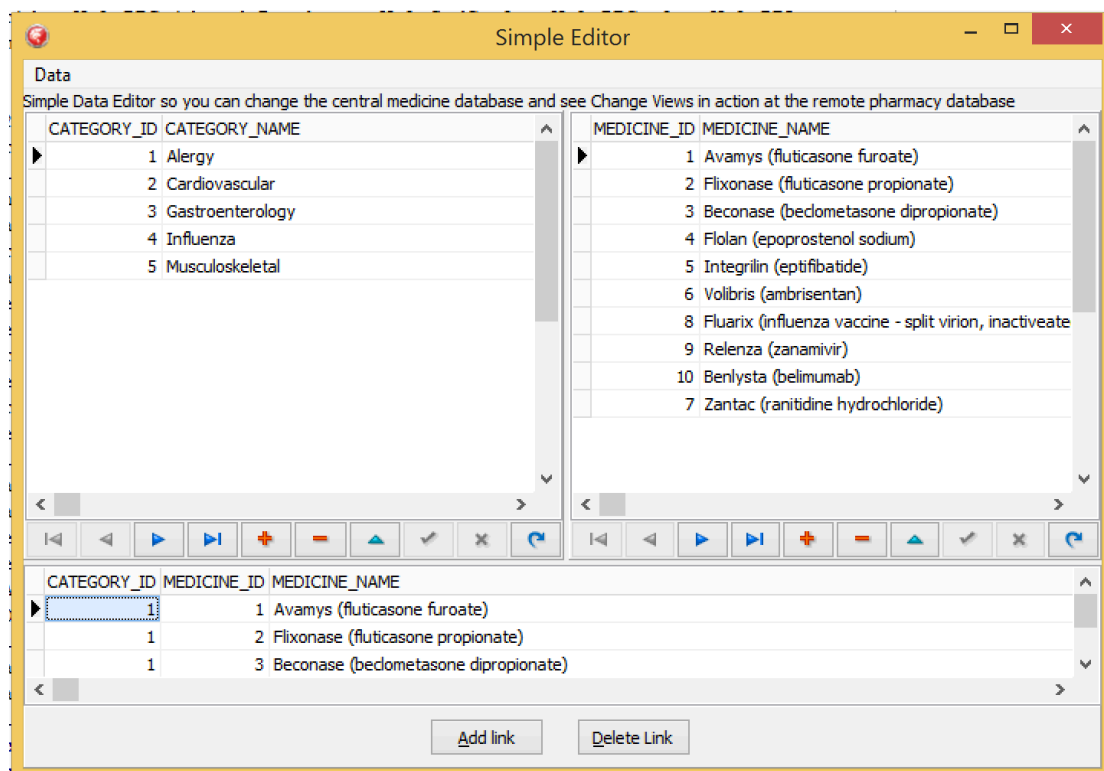
To make the demo compatible with all editions this database has not been encrypted.

### Using the Demo

Create a folder called c:\data and place the database in that folder.

The basic concept is to update the data in the central database (medicines) and then see you can identify the changes in the remote (pharmacy) database.

The two parts in essence are the app to edit the medicine database and then the pharmacy database application.



The medicine database editor is a very simple application and is shipped in the Object Pascal version of the demo. (CentralDataViewer)

The PharmacyApp reads an INI file that states which pharmacy.ib database to connecting to locally and also where the medicine.ib (remote) database is.

The INI should be in the application folder and called "PharmacyApp.Ini"  
*See TdmPharmacyController.Create*

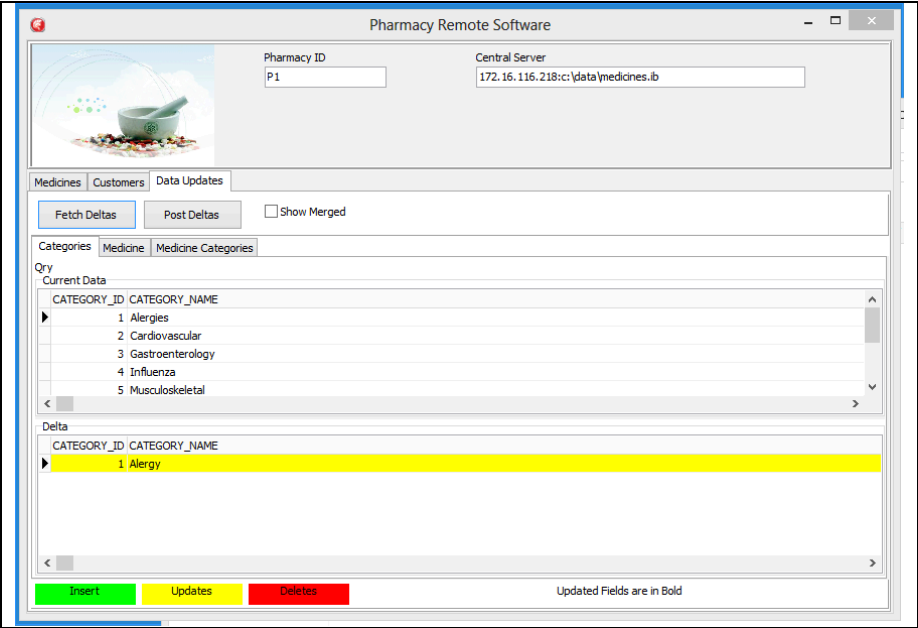
```
[Database]
Local=c:\data\pharmacy1.ib
Remote=127.0.0.1:c:\data\medicines.ib
```

Open 1 or more PharmacyApp instances (each using a different ID's) The follow is examples of what you will see depending on if there are changes to fetch or not.

Once data is modified in the medicine database pushing "Fetch Deltas" on the PharmacyApp will show the deltas; Post Deltas will post and refresh the datasets. Example screens below.

Fetch Deltas

Example shows delta waiting for the category table



Tick Show Merged and run Fetch Deltas and you can now see local data + delta with edited fields in Bold

Pharmacy Remote Software

Pharmacy ID: P1

Central Server: 172.16.116.218:c:\data\medicines.ib

Medicines Customers Data Updates

Fetch Deltas Post Deltas ☒ Show Merged

Categories Medicine Medicine Categories

Qry Current Data

CATEGORY_ID	CATEGORY_NAME
1	Allergies
2	Cardiovascular
3	Gastroenterology
4	Influenza
5	Musculoskeletal

Delta

CATEGORY_ID	CATEGORY_NAME
1	<b>Allergy</b>
2	Cardiovascular
3	Gastroenterology
4	Influenza
5	Musculoskeletal

Insert Updates Deletes Updated Fields are in Bold

After you Post Deltas you will see that the data is up to date either with an empty delta window or no highlighted rows when you show merged.

Pharmacy Remote Software

Pharmacy ID: P1

Central Server: 172.16.116.218:c:\data\medicines.ib

Medicines Customers Data Updates

Fetch Deltas Post Deltas ☒ Show Merged

Categories Medicine Medicine Categories

Qry Current Data

CATEGORY_ID	CATEGORY_NAME
1	Allergy
2	Cardiovascular
3	Gastroenterology
4	Influenza
5	Musculoskeletal

Delta

CATEGORY_ID	CATEGORY_NAME
1	Allergy
2	Cardiovascular
3	Gastroenterology
4	Influenza
5	Musculoskeletal

Insert Updates Deletes Updated Fields are in Bold