

# Main SJ Tables Explained

*The contents of this document are proprietary information and Copyrighted © 2004 by Tahoe Management Systems, Inc. and SalesJunction.com. This information may not be reprinted, disseminated or represented in any way without prior express written permission by Tahoe Management Systems, Inc.*

## Overview

This document will explain the key SalesJunction.com table relations. The following tables will be explained:

- CRDACCOUNT
- CRDCONTACT
- CRDOPP
- CRDCASE
- CRDACTIVITIES
- CRDNOTE

User Defined fields are kept in secondary tables that join with their main table counterparts. The Following User Defined fields tables will be defined along with their primary tables:

- CRDUSERFLDS\_A
- CRDUSERFLDS\_C
- CRDUSERFLDS\_O
- CRDUSERFLDS\_T (cases)

The [second section](#) will give example T-SQL and Queries to:

- Rename an Account or merge into another
- Change all Contacts to another Account
- Reassign main ownership of an Account

## Tables Explained

Complete field [documentation](#) along with an [Entity Relationship Diagram](#) can be seen by clicking on [this link](#). This documentation is very detailed and can be used by technical database administrators. The following is a general explanation of the tables and their relations.

### CRDACCOUNT

This is the accounts table. This is the main table and most other tables relate back to this table in some way. The AccountID field in this table is used as a key in all other tables. The AccountID field is unique in this table.

The CRDUSERFLDS\_A table contains the user-defined fields. This table is always joined with CRDACCOUNT with a one-to-one relationship with AccountID and A\_AccountID. The following SQL can be used to join the two:

```
SELECT a.*,au.* FROM CRDACCOUNT a LEFT JOIN CRDUSERFLDS_A au ON
a.AccountID=au.A_AccountID
```

The record for the matching account in CRDUSERFLDS\_A is not always created when a new account is created in CRDACCOUNT. Use the LEFT SQL keyword to make sure you get all records in CRDACCOUNT. If there is no matching record in the user defined table you will have NULL value fields.

\* Pro users also should pay attention to CRDACCTUSERS. This table contains all accounts along with the authority settings for users. You can join this table when you want to apply authority filtering to users. For example if you have a specific user you would like to create a report for and you are a Pro customer. Join this table in using the AccountID and UserID and if the account record appears in the result-set then the user will have at least read authority to the account. The same hold true for all other table that have an AccountID key in them such as CRDCONTACT and CRDOPP.

## CRDCONTACT

This is the contacts table. Every account can have many contacts. Every contact must have an account assigned to it. The unique key in this table is AccountID and Name with AccountID matching a valid account from CRDACCOUNT.

The CRDUSERFLDS\_C table contains the user-defined fields. This table is always joined with CRDCONTACT with a one-to-one relationship with AccountID to C\_AccountID and Name to C\_Name. The following SQL can be used to join the two:

```
SELECT c.*,cu.* FROM CRDCONTACT c LEFT JOIN CRDUSERFLDS_C cu ON
c.AccountID=cu.C_AccountID AND c.Name=cu.C_Name
```

The record for the matching contact in CRDUSERFLDS\_C is not always created when a new contact is created in CRDCONTACT. Use the LEFT SQL keyword to make sure you get all records in CRDCONTACT. If there is no matching record in the user defined table you will have NULL value fields.

## CRDOPP

This is the opportunities table. Every account can have many opportunities. Every opportunity must have an account assigned to it. The unique key in this table is AccountID and Name with AccountID matching a valid account from CRDACCOUNT.

The CRDUSERFLDS\_O table contains the user-defined fields. This table is always joined with CRDOPP with a one-to-one relationship with AccountID to O\_AccountID and Name to O\_Name. The following SQL can be used to join the two:

```
SELECT o.*,ou.* FROM CRDOPP o LEFT JOIN CRDUSERFLDS_O ou ON
o.AccountID=ou.O_AccountID AND o.Name=ou.O_Name
```

The record for the matching opportunity in CRDUSERFLDS\_O is not always created when a new opportunity is created in CRDOPP. Use the LEFT SQL keyword to make sure you get all records in CRDOPP. If there is no matching record in the user defined table you will have NULL value fields.

## CRDCASE

This is the case table. Every account can have many cases. Every case must have an account assigned to it. The unique key in this table is AccountID and CaseRefNum. CaseRefNum is also unique in the table.

The CRDUSERFLDS\_T table contains the user-defined fields. This table is always joined with CRDCASE

with a one-to-one relationship with AccountID to T\_AccountID and the CaseRefNum to T\_CaseRefNum. The following SQL can be used to join the two:

```
SELECT t.*,tu.* FROM CRDCASE t LEFT JOIN CRDUSERFLDS_T tu ON
t.AccountID=tu.T_AccountID and t.CaseRefNum=tu.T_CaseRefNum
```

## CRDACTIVITIES

This table contains all the activities for accounts, contacts and opportunities. An activity must have a related AccountID from CRDACCOUNT. Activities can be assigned to contacts and opportunities. An account, contact or opportunity may have many activities. The unique key for this record is ActRefNum. This field is automatically created by the system and is numeric.

Since an activity can have related opportunities and contacts, the AccountID is used to relate them along with the Contact and/or Opportunity field in the table. Note that only contacts and opportunities from the same account the activity is assigned to may be related to an activity.

Since multiple users can be assigned to activities, there is a sub table to hold their entries. The CRDACTIVITIES\_USER table will contain all assigned users to an activity. The relationship is one to many from CRDACTIVITIES to CRDACTIVITIES\_USER with the joining key being ActRefNum.

The RecType field in this table contains one of the following characters: M=meeting, C=call, O=other, T=todo, E=email.

The Status field will contain a C if the activity is completed, otherwise it will be blank.

For Pro users, the FollowUpID and FollowUpStep may contain data that relates to their tables.

The following SQL will return activities along with any related account, contact and opportunity records:

```
SELECT v.*,a.*,c.*,o.* FROM CRDACTIVITIES v LEFT JOIN CRDACCOUNT a ON
a.AccountID=v.AccountID LEFT JOIN CRDCONTACT c on c.AccountID=v.AccountID and
c.Name=v.Contact LEFT JOIN CRDOPP o ON o.AccountID=v.Opportunity
```

Doing such a large join isn't recommended but does illustrate the relations with the tables. Usually you'll just want to join in one or two tables that you may want to get data from.

## CRDNOTE

This table contains all notes. A note can be related to an account, contact, or opportunity. All notes must be related to a valid account with a valid AccountID in the CRDACCOUNT table. The note table also contains document attachment references. The unique key for the notes table is: RECID. This field is standard unique number in all tables and is an auto incremented number created by the system.

A note or attachment must belong to an account, but can also be related to a contact or opportunity. The Type field tells you what kind of relation exists. It will contain one of the following characters: A=Account, C=Contact, O=Opportunity. If the note is not an A type, the AccountID/Name combination serves as the key to the related table. An example SQL to pull all notes related to a single contact is:

```
SELECT * FROM CRDNOTE WHERE AccountID='My Account' AND Name='Contact Name'
AND Type='C'
```

To identify a note from a file attachment record you can look at the Size, OriginalFilename, and Filename fields. For notes, those fields will contain 0, blank, and blank. If the size is not zero then it is a file attachment and the Description field will contain the description of the file if any.

An example SQL to pull all account notes with the related account record is:

```
SELECT n.*, a.* FROM CRDNOTE n JOIN CRDACCOUNT a ON a.AccountID=n.AccountID
WHERE n.Type='A'
```

An example SQL to pull all contact notes with the related contact record is:

```
SELECT n.*, c.* FROM CRDNOTE n JOIN CRDCONTACT c ON c.AccountID=n.AccountID
AND c.Name=n.Name WHERE n.Type='C'
```

## Common T-SQL and Queries

Note that renaming accounts, contacts and other fields causes filters to become invalid. If you rename fields and have users report that they are no longer getting records in their views, have them recreate the views.

**NOTE:** The database is under constant upgrade to add features for our users. These queries were written on 5/09/2004 and may be out-of-date at anytime in the future. Please be sure to consult the tables and their definitions when making changes like the following.

### Rename an Account

```
declare @oldacctname as varchar(30);
declare @newacctname as varchar(30);

set @oldacctname = 'OldNameID'
set @newacctname= 'NewNameID'

UPDATE CRDACCOUNT SET AccountID=@newacctname WHERE AccountID=@oldacctname
UPDATE CRDUSERFLDS_A SET A_AccountID=@newacctname WHERE A_AccountID=@oldacctname
UPDATE CRDOPP SET AccountID=@newacctname WHERE AccountID=@oldacctname
UPDATE CRDUSERFLDS_O SET O_AccountID=@newacctname WHERE O_AccountID=@oldacctname
UPDATE CRDCONTACT SET AccountID=@newacctname WHERE AccountID=@oldacctname
UPDATE CRDUSERFLDS_C SET C_AccountID=@newacctname WHERE C_AccountID=@oldacctname
UPDATE CRDCASE SET AccountID=@newacctname WHERE AccountID=@oldacctname
UPDATE CRDUSERFLDS_T SET T_AccountID=@newacctname WHERE T_AccountID=@oldacctname
UPDATE CRDNOTE SET AccountID=@newacctname WHERE AccountID=@oldacctname
UPDATE CRDACTIVITIES SET AccountID=@newacctname WHERE AccountID=@oldacctname
UPDATE CRDCHANGELOG SET AccountID=@newacctname WHERE AccountID=@oldacctname
UPDATE CRDACCTUSERS SET AccountID=@newacctname WHERE AccountID=@oldacctname --Pro
```

### Merge Account Into Another

```
declare @oldacctname as varchar(30);
declare @newacctname as varchar(30);
```

```
set @oldacctname = 'OldNameID'
set @newacctname= 'NewNameID'
```

```
UPDATE CRDOPP SET AccountID=@newacctname WHERE AccountID=@oldacctname
```

```
UPDATE CRDUSERFLDS_O SET O_AccountID=@newacctname WHERE O_AccountID=@oldacctname
```

```
UPDATE CRDCONTACT SET AccountID=@newacctname WHERE AccountID=@oldacctname
```

```
UPDATE CRDUSERFLDS_C SET C_AccountID=@newacctname WHERE C_AccountID=@oldacctname
```

```
UPDATE CRDCASE SET T_AccountID=@newacctname WHERE AccountID=@oldacctname
```

```
UPDATE CRDUSERFLDS_T SET T_AccountID=@newacctname WHERE T_AccountID=@oldacctname
```

```
UPDATE CRDNOTE SET AccountID=@newacctname WHERE AccountID=@oldacctname
```

```
UPDATE CRDACTIVITIES SET AccountID=@newacctname WHERE AccountID=@oldacctname
```

```
UPDATE CRDCHANGELOG SET AccountID=@newacctname WHERE AccountID=@oldacctname
```

```
DELETE CRDACCTUSERS WHERE AccountID=@oldacctname --Pro
```

```
DELETE CRDACCOUNT WHERE AccountID=@oldacctname
```

```
DELETE CRDUSERFLDS_A WHERE A_AccountID=@oldacctname
```

## Change All Contacts to another Account

```
UPDATE CRDCONTACT SET AccountID='NewAcctID' WHERE AccountID='OldAcctID'
```

```
UPDATE CRDUSERFLDS_C SET C_AccountID='NewAcctID' WHERE C_AccountID='OldAcctID'
```

```
UPDATE CRDNOTE SET AccountID='NewAcctID' WHERE AccountID='OldAcctID' AND Name='Contact' AND Type='Contact'
```

```
UPDATE CRDACTIVITIES SET AccountID='NewAcctID' WHERE AccountID='OldAcctID' AND Contact='Contact'
```

## Reassign Main Ownership of an Account

```
declare @newuser as varchar(25);
declare @olduser as varchar(25);
```

```
set @newuser = 'New UserID'
set @olduser = 'Old UserID'
```

```
UPDATE CRDACCOUNT SET OwnerUserID=@newuser WHERE OwnerUserID=@olduser
```

```
UPDATE CRDCONTACT SET OwnerUserID=@newuser WHERE OwnerUserID=@olduser
```

```
UPDATE CRDACTIVITIES SET CrtUserID=@newuser WHERE CrtUserID=@olduser
```

```
UPDATE CRDACTIVITIES_USER SET ActUserID=@newuser WHERE ActUserID=@olduser
```

```
UPDATE CRDOPP SET OwnerUserID=@newuser WHERE OwnerUserID=@olduser
```

```
UPDATE CRDNOTE SET CrtUserID=@newuser WHERE CrtUserID=@olduser
```

```
UPDATE CRDACCTUSERS SET UserID=@newuser WHERE UserID=@olduser
```

### Set ALL Records to a New User

```
UPDATE CRDACCOUNT SET OwnerUserID='newuser'
```

```
UPDATE CRDCONTACT SET OwnerUserId='newuser'
```

```
UPDATE CRDACTIVITIES SET CrtUserID='newuser'
```

```
UPDATE CRDACTIVITIES_USER SET ActUserID='newuser'
```

```
UPDATE CRDOPP SET OwnerUserID='newuser'
```

```
UPDATE CRDCASE SET AssignedUserID='newuser'
```

```
UPDATE CRDNOTE SET CrtUserID='newuser'
```

```
UPDATE CRDACCTUSERS SET UserID='newuser'
```

---