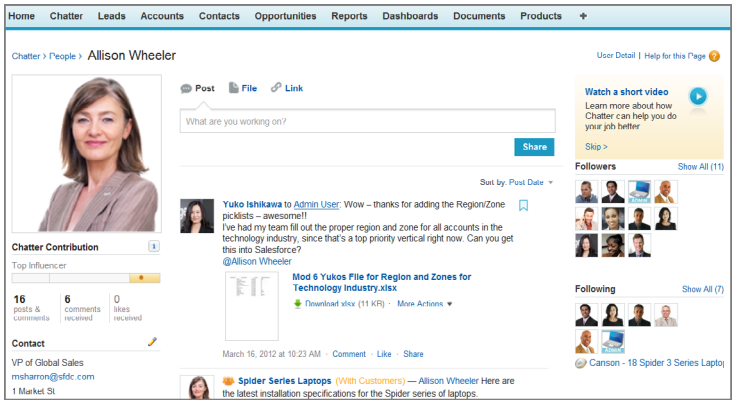
Chatter SOAP API Cheat Sheet

Overview

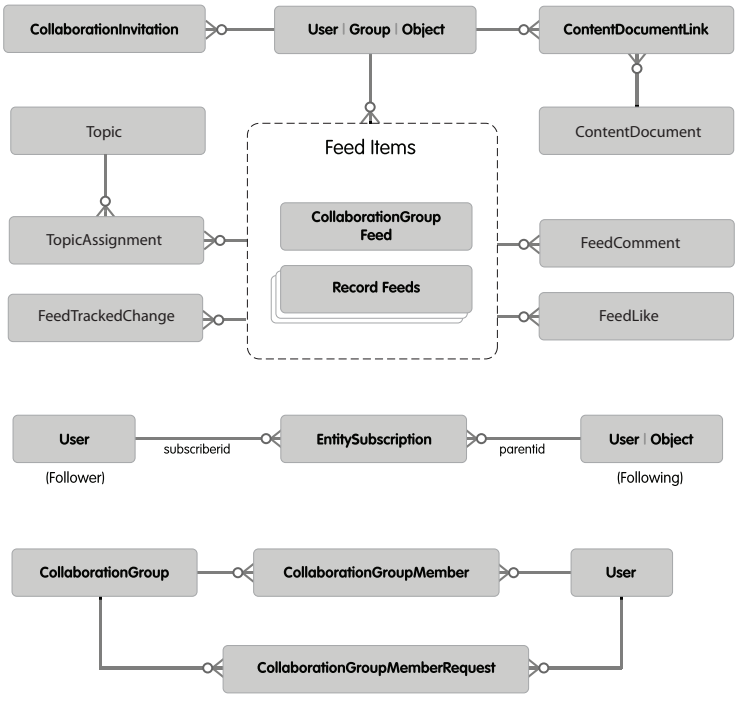
This cheat sheet describes the data model that supports the Salesforce Chatter collaboration platform. Chatter provides profiles, posts, comments, feeds, and groups.

Each of these items can be found in the data model, exposed on Force.com. Using this data model, you can extend your own applications with Chatter support, or build new ones with the Chatter platform.



Chatter Data Model

The data model diagram shows the most important objects found in the Chatter data model.



Feed Items

Feed Items is a collection of objects that store certain feed data. These primary objects provide a route to most Chatter data. Feed items aggregate various feeds. For example, CollaborationGroupFeed aggregates posts for groups a user is a member of and record feeds aggregate posts made on that record and changes to tracked record fields. There is a record feed for each object that is enabled for feed tracking.

The Type field of the object indicates whether the feed item represents a UserStatus, TextPost, LinkPost, ContentPost, DashboardComponentSnapshot, ApprovalPost, CollaborationGroupCreated, or TrackedChange. If the type is TrackedChange, the FeedTrackedChange object stores additional data. Any other type means that the feed stores the additional data, such as AccountFeed or CollaborationGroupFeed.

Each feed item represents either a set of changes on a specific record or a post to a particular user or record, as indicated by the ParentId field. For instance, when you post to a feed, the ParentId of the resulting feed item, holds your UserId. Note that some queries and statements, for example adding a comment, require the ID of a feed item.

Fields include:

- *Type*: The type of feed item. Values are UserStatus, TextPost, LinkPost, ContentPost, DashboardComponentSnapshot, ApprovalPost, or TrackedChange.
- *CreatedDate*: The date and time when the feed item was created.
- *CreatedById*: The ID of the user who created the feed item.
- *InsertedById*: If the CreatedById specifies a different user than the user who is currently logged in (only possible via API inserts), the InsertById is the ID of the user who created the record. This is useful for integrations and data imports.
- *ParentId*: The ID of the user, record, or group that is tracked in the feed.

When feed tracking is enabled for an object, the object's record feed aggregrates changes too the tracked fields by feed items. Record feeds for standard objects are named after the object. For example, an AccountFeed object represents the record feed for Account. For custom objects like Foo__c the record feed name is Foo__Feed. The FeedTrackedChange object stores the field change details.

OpportunityFeed Query

```
ID OpportunityId = <Id of an opportunity>;
List<OpportunityFeed> oRFeed = [
   SELECT Id, Type, CreatedBy.Name, Parent.Name, Body
   FROM OpportunityFeed WHERE ParentId = :OpportunityId
   ORDER BY CreatedDate DESC, Id DESC LIMIT 20
];
```

This query displays the 20 most recent opportunity feed items.

Record Detail Page Query

```
ID accountId = <Id of an Account>;
List<AccountFeed> aRFeed = [
   SELECT Id, Type, CreatedDate, CreatedById, CreatedBy.Name,
      ParentId, Parent.Name, FeedItemId, Body,
      Title, LinkUrl,
      (SELECT Id, FieldName, OldValue, NewValue
         FROM FeedTrackedChanges),
      (SELECT Id, CreatedDate, CreatedById,
            CreatedBy.Name, CommentBody
         FROM FeedComments ORDER BY CreatedDate DESC)
   FROM AccountFeed WHERE ParentId = :accountId
   ORDER BY CreatedDate DESC, Id DESC LIMIT 10
];
```

This query displays the items on the detail page for an account record. You can modify the query to work with other record feeds by replacing AccountFeed with a different record feed.

FeedTrackedChange

The FeedTrackedChange object stores individual field changes on records that belong to objects that have feed tracking enabled. Query this object only through a record feed.

Fields: *Id*, *FieldName* (name of the field that was changed), *OldValue*, *NewValue*, *FeedItemId* (ID of the feed item that represents the record feed item with the changed field)

Query for Tracked Changes to a Record

```
ID OpportunityId = <Id of an opportunity>;
List<OpportunityFeed> oEFeedT = [
   SELECT Id, Type, CreatedBy.Name, Parent.Name,
      (SELECT OldValue, NewValue FROM FeedTrackedChanges)
   FROM OpportunityFeed
   WHERE ParentId = :OpportunityId AND Type = 'TrackedChange'
   ORDER BY CreatedDate DESC, Id DESC LIMIT 20
];
```

In this example ParentId points to an opportunity record and the query returns all OpportunityFeed (a record feed) items about that opportunity.

CollaborationGroup

The CollaborationGroup object represents Chatter groups.

Fields: *Name*, *Description*, *OwnerId*, *CollaborationType* (Public or Private), *CanHaveGuests*

The CollaborationGroup object has an associated record feed called CollaborationGroupFeed, which aggregates the feed items associated with a group. Use CollaborationGroupFeed like any other record feed.

Groups Query

```
List<CollaborationGroup> oG = [SELECT Name, Description,
CollaborationType, OwnerId from CollaborationGroup];
```

The query returns a list of all Chatter groups. The CollaborationType field is either Private or Public.

Group's Feed Query

```
ID cgid = <Id of a group>;
List<CollaborationGroupFeed> ic = [
   SELECT Id, Type, ParentId, Parent.Name, Body, Title,
      (SELECT Id, FieldName, OldValue, NewValue
         FROM FeedTrackedChanges ORDER BY Id DESC),
    (SELECT Id, CommentBody, CreatedDate, CreatedById,
         CreatedBy.FirstName, CreatedBy.LastName
         FROM FeedComments
         ORDER BY CreatedDate DESC, Id DESC LIMIT 4)
   FROM CollaborationGroupFeed WHERE ParentId = :cgid
   ORDER BY CreatedDate DESC, Id DESC LIMIT 20
];
```

Changes to a group's name, description or similar fields appear as tracked changes on the group's feed. The query retrieves the tracked change values as well as comments on posts.

CollaborationGroupMemberRequest

The CollaborationGroupMemberRequest object represents a request to join a private Chatter group.

Fields: *CollaborationGroupId* (ID of the CollaborationGroup), *RequesterId* (ID of the user requesting to join the group), *ResponseMessage* (optional message), *Status* (status of the request)

On `create()`, an email is sent to the owner and managers of the private group. When they accept or decline the request, the `Status` changes to `Accepted` or `Declined`, and an email is sent to notify the requester. If the request is declined, a `ResponseMessage` can be included to provide additional details.

EntitySubscription

The EntitySubscription object represents subscriptions. Subscriptions specify which users or records a user is following, or who is following a user or record.

Fields: *ParentId* (ID of the user or record being followed), *SubscriberId* (ID of the user following the record)

Query for Followers of a User or Record

```
ID uid = <Id of User or Record>;
List<EntitySubscription> followers = [
   SELECT Id, SubscriberId, Subscriber.Name
   FROM EntitySubscription WHERE ParentId = :uid
];
```

Query for Users and Records I Follow

```
ID uid = <Id of User>;
EntitySubscription[] followingES = [
   SELECT Id, ParentId, SubscriberId, Parent.Name
   FROM EntitySubscription WHERE SubscriberId = :uid
];
```

To only show users, add Parent.Type='User' to the WHERE clause.

Query for Subscriptions to Records

```
ID uid = <Id of User>;
List<EntitySubscription> les = [
   SELECT Id, Parent.Name FROM EntitySubscription
   WHERE SubscriberId = :uid AND ParentId IN
      (SELECT Id FROM Account WHERE NumberOfEmployees >= 10)
];
```

This query lists accounts a user follows that have more than ten employees.

FeedComment

The FeedComment object stores comments and is a child object of an associated record feed item. Each FeedComment is associated with a feed item. Query this object only a record feed.

Fields: *CommentBody* (the string representing the comment's text), *FeedItemId* (ID of the feed item on which the comment was made), *ParentId* (ID of the record associated with the comment), *CreatedById* (ID of the user who added this item)

CreatedById can be modified to the ID of the original poster during migration. ParentId either points to a record that received a comment, or to a user whose post received a comment. The functionally is equivalent to the associated feed item's parent.

Creating a Comment

```
FeedComment fcomment = new FeedComment();
fcomment.FeedItemId = <feedItemId>;
fcomment.CommentBody = 'This is a profound comment.';
insert fcomment;
```

You need the ID of a feed item to create comment. All the queries on record feeds return feed item records.

ContentDocumentLink

The ContentDocumentLink object represents the link between a Salesforce CRM Content document or Chatter file and where it's shared. A file can be shared with other users, Chatter groups, records, and Salesforce CRM Content workspaces.

The following query retrieves the Title and Description of the latest version of a document loaded via Chatter:

```
SELECT ContentDocument.LatestPublishedVersion.Title,
       ContentDocument.LatestPublishedVersion.Description
       FROM ContentDocumentLink
       WHERE LinkedEntityid = '<entity-id>'
```

CollaborationGroupMember

The CollaborationGroupMember object associates Chatter group members with Chatter groups.

Fields: *CollaborationGroupId* (ID of the CollaborationGroup), *MemberId* (ID of the user who is a member)

Members of a Group Query

```
ID cg = <Id of a Chatter group>;
List<CollaborationGroupMember> oM = [
   SELECT MemberId, CollaborationGroup.Name
   FROM CollaborationGroupMember WHERE CollaborationGroupId = :cg
];
```

The query returns the IDs of members for a particular Chatter group.

Adding a User to a Group

```
ID uid = <Id of a user>;
ID cgid = <Id of a group>;
CollaborationGroupMember cgm = new CollaborationGroupMember(
   memberid = uid, collaborationgroupid = cgid);
insert cgm;
```

CollaborationInvitation

Use the CollaborationInvitation object to create or delete (cancel) invitations to join Chatter. You can invite a user to join Chatter directly or as part of a CollaborationGroup. When users accept your CollaborationGroup invitation, they join the CollaborationGroup and Chatter as well.

Invited users can view profiles, post on their feed, and join groups, but they can't see your data or records.

DisableFeedTrackingHeader

The DisableFeedTrackingHeader is a SOAP header that specifies whether the changes made in the current call are tracked in feeds. When a user subscribes to a record, changes to the record display in the Chatter feed on the user's home page.

Use this header if you want to process a large number of records without tracking the changes in various feeds related to the records. This header is available if the Chatter feature is enabled for your organization.

Fields: *disableFeedTracking* (If `true`, the changes made in the current call are not tracked in feeds.)

Writing Efficient SOQL Queries

- ALWAYS use a LIMITS clause. Feeds generally show 20 items per page.
- Use efficient filters:
  - ParentId, Id
  - ParentType
  - CreateById
- `For record feeds, include ParentID=<recordID>`
- `Use the ORDER BY CreateDate DESC, ID DESC format to show most recent feed items.`

Creating a Text Post

```
FeedItem fItem = new FeedItem();
fItem.Type = 'TextPost';
fItem.ParentId = <Id of User or Record>;
fItem.Body = 'The mice will see you now';
insert fItem;
```

Note that you can't create a post of type UserStatus.

See the following queries for additional examples:

- Chatter Tab Query
- Record Detail Page Query
- Profile Tab Query

Chatter Triggers

You can create Apex triggers on the FeedItem and FeedComment objects.You can't create triggers on record feeds, but you can have triggers run based on events in a record feed. Triggers only support inserts and deletes. Only FeedItems of Type TextPost, LinkPost, and ContentPost can be inserted.

Apex code uses additional security when executing in a Chatter context.

To post to a private group, the user running the code must be a member of that group. If the user isn't a member, you can set the CreatedById field to be a member of the group in the FeedItem record.

This example is a trigger that takes an action everytime someone posts a specific key word to an entity feed. For example, you may want to perform an action on a record based on a special keyword (e.g a '!close' post closes an Opportunity).

```
Trigger CheckChatterPostsOnOpportunity on FeedItem (before insert)
{
   // Get the key prefix for the Opportunity object
   // using a describe call.
   String oppKeyPrefix = Opportunity.sObjectType.
getDescribe().getKeyPrefix();
      for (FeedItem f: trigger.new)
         {
         String parentId = f.parentId;
         // Compare the start of the 'parentID' field
         // to the Opportunity key prefix to
         // restrict the trigger to act on posts made to the
         // Opportunity object.
            if (parentId.startsWith(oppKeyPrefix) &&
               f.Body == '!close')
            {
               // Add your business logic here
            }
         }
      }
```

Writing Efficient Chatter Triggers

- Remember all triggers run in bulk.
- Use isEmpty to restrict logic execution.
- Refactor common logic out of a trigger.
  - Most triggers work with FeedItem or FeedComment.
- Remember that UserStatus is on the User Object.
- Filter FeedItem execution:
  - On Post types of triggers: TextPost, LinkPost, ContentPost.
  - On Object types of triggers, use this format:
    `<object>.sObjectType.getDescribe().getKeyPrefix();`


salesforce
force.com