Meta Stack Exchange is a question and
answer site for meta-discussion of the
Stack Exchange family of Q&A websites.
It only takes a minute to sign up.

×

Sign up to join this community

Anybody can ask a question

Anybody can answer

The best answers are voted
up and rise to the top

Database schema documentation for the public data dump and SEDE

Asked 10 years, 10 months ago   Active yesterday   Viewed 70k times

▲

322

▼

Stack Exchange releases "data dumps" of all its publicly available content roughly every three months via archive.org, and also makes that information queryable over the Internet at the Stack Exchange Data Explorer (SEDE) (updated weekly, on Sunday). Over time, as new features and other bits of data are introduced to Stack Exchange sites, the "anatomy" of the data dump is likely to change.

372

- What is the current structure of the public data dump?

- Is there a diagram of the schema?

Return to FAQ index

`support` `faq` `data-explorer` `data-dump`

edited Mar 21 at 12:31

community wiki
22 revs, 9 users 28%
Peter Mortensen

2 Answers

Active Oldest Votes

▲

About this list:
foreign key fields are formatted as links to their parent table.

Posts / **PostsWithDeleted**

You find in `Posts` all non-deleted posts. `PostsWithDeleted` includes rows with deleted posts while sharing the same columns with `Posts` but [for deleted posts only a few fields populated](#) which are marked with a [1] below.

- `Id` [1]
- `PostTypeId` [1] *(listed in the [PostTypes] table)*
  - 1 = Question
  - 2 = Answer
  - 3 = Orphaned tag wiki
  - 4 = Tag wiki excerpt
  - 5 = Tag wiki
  - 6 = Moderator nomination
  - 7 = "Wiki placeholder" *(seems to only be the [election description])*
  - 8 = Privilege wiki
- [AcceptedAnswerId] *(only present if `PostTypeId = 1`)*
- [ParentId] [1] *(only present if `PostTypeId = 2`)*
- `CreationDate` [1]
- `DeletionDate` [1] *(only non-null for the SEDE `PostsWithDeleted` table. Deleted posts are not present on `Posts` . Column not present on data dump.)*
- `Score` [1]
- `ViewCount` *(nullable)*
- `Body` *([as rendered HTML](#), not Markdown)*
- [OwnerUserId] *(only present if user has not been deleted; always -1 for tag wiki entries, i.e. the community user owns them)*
- `OwnerDisplayName` *(nullable)*
- [LastEditorUserId] *(nullable)*
- `LastEditorDisplayName` *(nullable)*
- `LastEditDate` (e.g. `2009-03-05T22:28:34.823`) - *the date and time of the most recent edit to the post (nullable)*
- `LastActivityDate` (e.g. `2009-03-11T12:51:01.480`) - *datetime of the post's most recent activity*
- `Title` *(nullable)*
- `Tags` [1] *(nullable)*
- `AnswerCount` *(nullable)*
- `CommentCount` *(nullable)*
- `FavoriteCount` *(nullable)*
- `ClosedDate` [1] *(present only if the post is closed)*
- `CommunityOwnedDate` *(present only if post is community wiki'd)*

Users

- Reputation

- CreationDate

- DisplayName

- LastAccessDate *([Datetime user last loaded a page; updated every 30 min at most](#))*

- WebsiteUrl

- Location

- AboutMe

- Views *([Number of times the profile is viewed](#))*

- UpVotes *([How many upvotes the user has cast](#))*

- DownVotes

- ProfileImageUrl

- EmailHash *(now always blank)*

- AccountId *(User's Stack Exchange Network profile ID)*

Comments

- Id

- [PostId](#)

- Score

- Text *(Comment body)*

- CreationDate

- UserDisplayName

- [UserId](#) *(Optional. Absent if user has been deleted)*

Badges

- Id

- [UserId](#)

- Name *(Name of the badge)*

- Date (e.g. 2008-09-15T08:55:03.923)

- Class

  1 = Gold

  2 = Silver

  3 = Bronze

- TagBased = *True* *if badge is for a tag, otherwise it is a named badge*

CloseAsOffTopicReasonTypes

- Id

- IsUniversal

- `MarkdownInputGuidance`

- `MarkdownPostOwnerGuidance`

- `MarkdownPrivilegedUserGuidance`

- `MarkdownConcensusDescription` (sic)

- `CreationDate`

- `CreationModeratorId`

- `ApprovalDate`

- `ApprovalModeratorId`

- `DeactivationDate`

- `DeactivationModeratorId`

PendingFlags

Despite the name, this table in fact contains close-related flags and votes.

- `Id`

- `FlagTypeId` *(listed in the* `FlagTypes` *table)*

  13 = canned flag for closure

  14 = vote to close

  15 = vote to reopen

- `PostId`

- `CreationDate`

- `CloseReasonTypeId` *(listed in the* `CloseReasonTypes` *table)*

- `CloseAsOffTopicReasonTypeId` , *if* `CloseReasonTypeId = 102 (off-topic)` *(listed in the* `CloseAsOffTopicReasonTypes` *table)*

- `DuplicateOfQuestionId` , *if* `CloseReasonTypeId` *is 1 or 101 (old duplicate or current duplicate)*

- `BelongsOnBaseHostAddress` , *for votes to close and migrate*

PostFeedback

Collects up and down votes from anonymous visitor and/or unregistered users. See [here](#)

- `Id`

- `PostId`

- `IsAnonymous`

- `VoteTypeId` *(listed in the* `VoteTypes` *table)*

  2 = UpMod

  3 = DownMod

- `CreationDate`

PostHistory

- `PostHistoryTypeId` *(listed in the* `PostHistoryTypes` *table)*

  1 = Initial Title - initial title *(questions only)*

  2 = Initial Body - initial post raw body text

  3 = Initial Tags - initial list of tags *(questions only)*

  4 = Edit Title - modified title *(questions only)*

  5 = Edit Body - modified post body (raw markdown)

  6 = Edit Tags - modified list of tags *(questions only)*

  7 = Rollback Title - reverted title *(questions only)*

  8 = Rollback Body - reverted body (raw markdown)

  9 = Rollback Tags - reverted list of tags *(questions only)*

  10 = Post Closed - post voted to be closed

  11 = Post Reopened - post voted to be reopened

  12 = Post Deleted - post voted to be removed

  13 = Post Undeleted - post voted to be restored

  14 = Post Locked - post locked by moderator

  15 = Post Unlocked - post unlocked by moderator

  16 = Community Owned - post now community owned

  17 = Post Migrated - post migrated - *now replaced by 35/36 (away/here)*

  18 = Question Merged - question merged with deleted question

  19 = Question Protected - question was protected by a moderator.

  20 = Question Unprotected - question was unprotected by a moderator.

  21 = Post Disassociated - OwnerUserId removed from post by admin

  22 = Question Unmerged - answers/votes restored to previously merged question

  24 = Suggested Edit Applied

  25 = Post Tweeted

  31 = Comment discussion moved to chat

  33 = Post notice added - `comment` *contains foreign key to PostNotices*

  34 = Post notice removed - `comment` *contains foreign key to PostNotices*

  35 = Post migrated away - *replaces id 17*

  36 = Post migrated here - *replaces id 17*

  37 = Post merge source

  38 = Post merge destination

  50 = Bumped by Community User

  52 = Question became hot network question

  53 = Question removed from hot network questions by a moderator

  Additionally, in [older dumps](#) *(all guesses, all seem no longer present in the wild):*

  23 = Unknown dev related event

  26 = Vote nullification by dev *(ERM?)*

  27 = Post unmigrated/hidden moderator migration?

  28 = Unknown suggestion event

  29 = Unknown moderator event *(possibly de-wikification?)*

  30 = Unknown event *(too rare to guess)*

- `PostId`

- `RevisionGUID` : At times more than one type of history record can be recorded by a single action. All of these will be grouped using the same RevisionGUID

- `CreationDate` (e.g. `2009-03-05T22:28:34.823`)

- `UserDisplayName` : populated if a user has been removed and no longer referenced by user Id

- `Comment` : This field will contain the comment made by the user who edited a post.

  - If PostHistoryTypeId = 10, this field contains the CloseReasonId of the close reason (listed in `CloseReasonTypes`):

    Old close reasons:

    1 = Exact Duplicate

    2 = Off-topic

    3 = Subjective and argumentative

    4 = Not a real question

    7 = Too localized

    10 = General reference

    20 = Noise or pointless (Meta sites only)

    Current close reasons:

    101 = Duplicate

    102 = Off-topic

    103 = Unclear what you're asking

    104 = Too broad

    105 = Primarily opinion-based

  - If `PostHistoryTypeId in (33,34)` this field contains the `PostNoticeId` of the `PostNotice`

- `Text` : A raw version of the new value for a given revision

  - If `PostHistoryTypeId in (10,11,12,13,14,15,19,20,35)` this column will contain a JSON encoded string with all users who have voted for the `PostHistoryTypeId`

  - If it is a duplicate close vote, the JSON string will contain an array of original questions as `OriginalQuestionIds`

  - If `PostHistoryTypeId = 17` this column will contain migration details of either `from <url>` or `to <url>`

PostLinks

- `Id` primary key

- `CreationDate` when the link was created

- `PostId` id of source post

- `RelatedPostId` id of target/related post

- `LinkTypeId` type of link

  1 = Linked (`PostId` contains a link to `RelatedPostId`)

  3 = Duplicate (`PostId` is a duplicate of `RelatedPostId`)

PostNotices

- `Id`

- `PostId`

- `PostNoticeTypeId`

      3 = Insufficient explanation

      10 = Current answers are outdated

      11 = Draw attention

      12 = Improve details

      13 = Authoritative reference needed

      14 = Canonical answer required

      15 = Reward existing answer

      20 = Content dispute

      21 = Offtopic comments

      22 = Historical significance

      23 = Wiki Answer

- `CreationDate`

- `DeletionDate`

- `ExpiryDate`

- `Body` *(when present contains the custom text shown with the notice)*

- `OwnerUserId`

- `DeletionUserId`

PostNoticeTypes

- `Id`

- `ClassId`

      1 = Historical lock

      2 = Bounty

      4 = Moderator notice

- `Name`

- `Body` *(contains the default notice text)*

- `IsHidden`

- `Predefined`

- `PostNoticeDurationId`

     -1 = No duration specified

     1 = 7 days (bounty)

PostTags

- `PostId`

- `TagId`

ReviewRejectionReasons

Canned rejection reasons for suggested edits. See [Show all review rejection reasons](#)

- `Id`

- `PostTypeId` (for reasons that apply to Wiki (5) or Excerpt (6) post types only, otherwise null)

ReviewTaskResults

- `Id`
- `ReviewTaskId`
- `ReviewTaskResultTypeId` (listed in `ReviewTaskResultTypes`)

  1 = Not Sure

  2 = Approve (suggested edits)

  3 = Reject (suggested edits)

  4 = Delete (low quality)

  5 = Edit (first posts, late answers, low quality)

  6 = Close (close, low quality)

  7 = Looks OK (low quality)

  8 = Do Not Close (close)

  9 = Recommend Deletion (low quality answer)

  10 = Recommend Close (low quality question)

  11 = I'm Done (first posts)

  12 = Reopen (reopen)

  13 = Leave Closed (reopen)

  14 = Edit and Reopen (reopen)

  15 = Excellent (community evaluation)

  16 = Satisfactory (community evaluation)

  17 = Needs Improvement (community evaluation)

  18 = No Action Needed (first posts, late answers)

- `CreationDate` date only (`2018-07-31 00:00:00`); time data looks to be purposefully removed to protect user privacy
- `RejectionReasonId` (for suggested edits; listed in `ReviewRejectionReasons`)
- `Comment`

ReviewTasks

- `Id`
- `ReviewTaskTypeId` (listed in `ReviewTaskTypes`)

  1 = Suggested Edit

  2 = Close Votes

  3 = Low Quality Posts

  4 = First Post

  5 = Late Answer

  6 = Reopen Vote

  7 = Community Evaluation

  8 = Link Validation

  9 = Flagged Posts

  10 = Triage

- `DeletionDate` date only (`2018-07-31 00:00:00`)

- `ReviewTaskStateId` (listed in [ReviewTaskStates](#))

    1 = Active

    2 = Completed

    3 = Invalidated

- [PostId](#)

- [SuggestedEditId](#) (for suggested edits, which have their own numbering for historical reasons)

- [CompletedByReviewTaskId](#) id associated to the ReviewTaskResult that stores the outcome of a completed review.

SuggestedEdits

If both approval and rejection date are null then this edit is still in review (and its corresponding entry in `ReviewTasks` will have an active state as well).

- `Id`

- [PostId](#)

- `CreationDate`

- `ApprovalDate` - NULL if not approved (yet).

- `RejectionDate` - NULL if not rejected (yet).

- [OwnerUserId](#)

- `Comment`

- `Text`

- `Title`

- `Tags`

- `RevisionGUID`

SuggestedEditVotes

- `Id`

- [SuggestedEditId](#)

- [UserId](#)

- `VoteTypeId` (listed in the [VoteTypes](#) table)

    2 = Approve (technically UpMod)

    3 = Reject (technically DownMod)

- `CreationDate`

- [TargetUserId](#)

- `TargetRepChange`

Tags

- TagName

- Count

- ExcerptPostId *(nullable) Id of Post that holds the excerpt text of the tag*

- WikiPostId *(nullable) Id of Post that holds the wiki text of the tag*

TagSynonyms

- Id

- SourceTagName

- TargetTagName

- CreationDate

- OwnerUserId

- AutoRenameCount

- LastAutoRename

- Score

- ApprovedByUserId

- ApprovalDate

Votes

- Id

- PostId

- VoteTypeId *(listed in the VoteTypes table)*

  1 = AcceptedByOriginator

  2 = UpMod *(AKA upvote)*

  3 = DownMod *(AKA downvote)*

  4 = Offensive

  5 = Favorite *(AKA bookmark; UserId will also be populated)*

  6 = Close (effective 2013-06-25: Close votes are **only** stored in table: PostHistory)

  7 = Reopen

  8 = BountyStart *(UserId and BountyAmount will also be populated)*

  9 = BountyClose *(BountyAmount will also be populated)*

  10 = Deletion

  11 = Undeletion

  12 = Spam

  15 = ModeratorReview

  16 = ApproveEditSuggestion

- UserId *(present only if VoteTypeId in (5,8) ; -1 if user is deleted)*

- CreationDate Date only (2018-07-31 00:00:00 *time data is purposefully removed to protect user privacy)*

- BountyAmount (present only if VoteTypeId in (8,9))

✕

Not listed here:

- `xxxTypes` tables which list (Id, Name) pairs for `Posts.PostTypeId` , `Votes.VoteTypeId` , etc.
See [Show all types](#) for an up-to-date list of all types.

All Tables/Columns/Type

Find the exact [T-SQL datatype](#) and length/precision of each specific column in this query:

[List all Fields in all Tables on SEDE](#)

TIMESTAMPS

All timestamps are `UTC` , default format: `yyyy-MM-dd hh:mm:ss` (stored with milliseconds).

Example of conversion current time to `PST` (including [DST](#)) using `At Time Zone` :

```
SELECT GetDate() At Time Zone 'UTC' At Time Zone 'Pacific Standard Time'
```

To **list time zones**: `SELECT * FROM sys.time_zone_info`

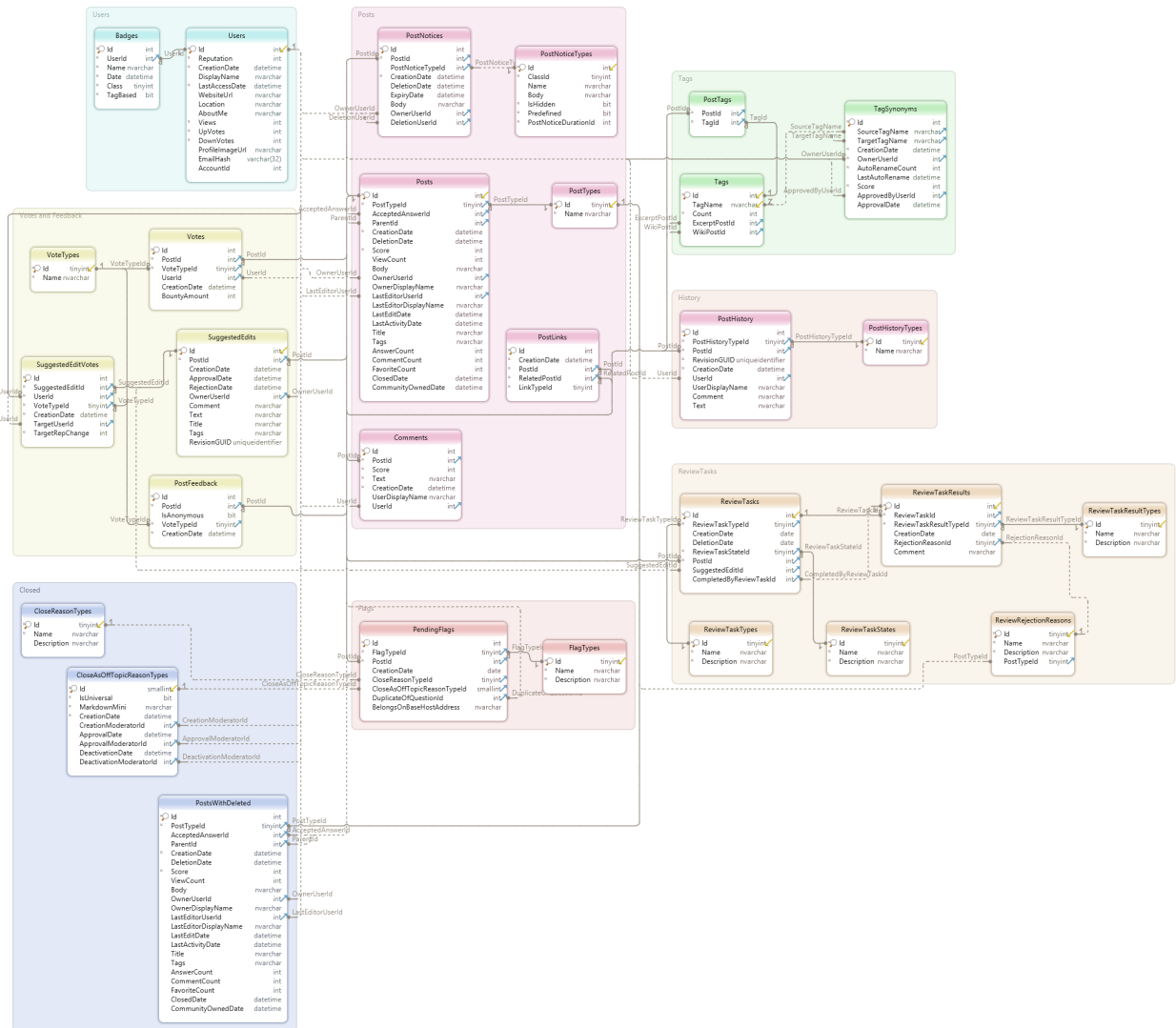edited yesterday   community wiki
                                                             96 revs, 35 users 30%
                                                             rene

The Schema Visualized

▲

20

▼

↺

The Entity Relationship Diagram (ERD) above was constructed using the data found in the *information_schema.columns* table. (Thank you to [ashleedawg](#) for paring the table down to its bare essentials).

All the table, column and key information were compiled into a SQL script using python3. The results were cross validated with the accepted answer above, and the few idiosyncratic foreign key formations left over were hardcoded in at the end.)

The ERD was then reverse engineered using *dbschema*.

Please find the script that wrote the SQL and other related files in my [GitHub repository](#).

The interactive HTML version of the ERD is posted here: https://sedeschema.github.io/

edited Jun 24 '19 at 5:02

community wiki
8 revs, 2 users 97%
leerssej