DataBase Final Project—Crowdfunding Platform　　Read ME

109403533 資管二 B 林采璇

【通則】

因為一些概念性上的細節，會同時影響到 ER Diagram 與 Store Procedure 的設計，故先將相關事項列於通則。

一、關於刪除 Delete

〈基本概念〉

將 Delete 區分成以下兩種類型:

1) 從資料庫確實移除。
2) 並非將資料從資料庫內部刪除，僅只是不顯示在 user 介面那端，形同已經 Delete 的狀態，相關資料仍儲存在資料庫內部。實作上，會在資料庫那邊的資料加入一個欄位名為 IsDeleted，用以判斷是否「已刪除」。

〈運作細節〉

1) 刪除「member」，即刪除整個帳號
   - 此屬於第一類型的刪除。
   - 會連帶將 member 相關的紀錄 Membercredential、Followingrecord、SponsorRecord、Comment、ProposalMember，全數從資料庫移除。以上幾個 Table 與 member 的 Foreign Key Option 設為 Cascade。
   - FAQ 中若剛好為 Last Editor, Foreign Key Option 設定其值改為 NULL。
   - 若此 member 屬於仍某提案的提案負責人(ProposalMember)，即 ProposalMember 的 IsDeleted 欄位仍為 0，則不可刪除帳號。

2) 刪除「SponsorRecord 、FollowingRecord、Comment、FAQ」
   - 「紀錄」的刪除，屬於第二類型的刪除。
   - 保存的資料，可用於日後分析資訊、回朔使用者「刪除」的資料等，給予使用者更良好的使用者體驗。

3) 刪除「Proposal、ProposalOption」
   - Proposal 的 delete 形同被 deactivated，於 Proposal 裡的欄位紀錄 IsDeactivated。此外，Proposal Option 因為屬於 Proposal 底下細分之提案，故不可單獨被 delete。

4) 刪除「ProposalMember」
   - 屬於第二類型的刪除。
   - 若身為「主要提案者」(IsMainProposer)，則不可退出該專案，直到該專案被停用（IsDeactivated)。若該提案已被停用（IsDeactivated)，會將其所有 ProposalMember 的 IsDeleted 欄位設為 1，連動「刪除」ProposalMember 的身分。

二、補充情境

1) 關於 ProposalMember 的運作

- 每個提案只會有一位「主要提案者」(IsMainProposer)，並預設為最早創建者。
- 身為主要發起人，不可退出該專案，直到該專案停用已被停用（IsDeactivated)。若該提案已被停用（IsDeactivated)，會將其所有 ProposalMember 的 IsDeleted 欄位設為 1，連動「刪除」ProposalMember 的身分。
- 若需更動 IsMainProposer，平台前端會發出訊息供需更動的雙方進行驗證。對於後臺的資料庫而言，若要實作這部分的內容，僅需接收要將哪位 ProposalMember 的 IsMainProposer 欄位改為 1，並把原本 isMainProposer 的欄位改為 0。

【ER Diagram】_ 總共 10 個 Entity

- **TB_Member**

Attribute	Type	Remark
MemberID	INT	[PK][AI][NN]
Name	VARCHAR(64)	[NN]
Phone	VARCHAR(64)	[NN]
Email	VARCHAR(64)	[UQ][NN]
Salt	CHAR(64)	[NN]
Address	VARCHAR(255)	[NN]

- **TB_MemberCredential**

Attribute	Type	Remark
MemberCredentialID	INT	[PK][AI][NN]
MemberID	INT	[FK][NN]
HashedPassword	VARCHAR(200)	[NN]

- **TB_ProposalMember**

Attribute	Type	Remark
ProposalMemberID	INT	[PK][AI][NN]
ProposalID	INT	[FK][NN]
MemberID	INT	[FK][NN]
ProposalMember CreateTime	DATETIME	[NN]
IsMainProposer	TINYINT	[NN]
IsDeleted	TINYINT	[NN]

IsMainProposer 的相關情境，詳見通則的補充情境

- **TB_Proposal**

Attribute	Type	Remark
ProposalID	INT	[PK][AI][NN]
CategoryID	INT	[FK][NN]
ProposalTitle	VARCHAR(120)	[NN]
Content	TEXT	
AccumulatedAmount	INT	[UN][NN]
Goal	INT	[UN][NN]
Status	INT	[UN][NN]
ViewedClicks	INT	[UN][NN]
ProposalCreateTime	DATETIME	[NN]
OnGoingTime	DATETIME	
DueTime	DATETIME	
IsPostponed	TINYINT	[NN]
IsDeactivated	TINYINT	[NN]

- **TB_ProposalOption**

Attribute	Type	Remark
ProposalOption ID	INT	[PK][AI][NN]
ProposalID	INT	[FK][NN]
ProposalOption Title	VARCHAR(120)	[NN]
PricePerEach	INT	[NN]
Description	TEXT	[NN]
QuantityLimit	INT	

- **TB_ProposalOption**

Attribute	Type	Remark
CategoryID	INT	[PK][AI][NN]
CategoryTitle	VARCHAR(120)	[FK][NN]

- **TB_Comment**

Attribute	Type	Remark
CommentID	INT	[PK][AI][NN]
ProposalID	INT	[FK][NN]
CommentMemberID	INT	[FK][NN]
CommentCreateTime	DATETIME	[NN]
MemberComment	TEXT	[NN]
ProposerResponse	TEXT	
IsDeleted	TINYINT	[NN]

- **TB_FAQ**

Attribute	Type	Remark
FAQID	INT	[PK][AI][NN]
ProposalID	INT	[FK][NN]
LastEditorID	INT	[FK]
Question	TEXT	[NN]
Answer	TEXT	[NN]
FAQCreateTime	DATETIME	[NN]
LastUpdateTime	DATETIME	[NN]
IsDeleted	TINYINT	[NN]

∵ 若刪除member時，剛好此member是LastEditor，則LastEditorID SET NULL
∴ LastEditorID can be NULL

- **TB_SponserRecord**

Attribute	Type	Remark
SponserRecordID	INT	[PK][AI][NN]
MemberID	INT	[FK][NN]
ProposalOptionID	INT	[FK][NN]
Amount	INT	[NN]
SponserRecord CreateTime	DATETIME	[NN]
IsDeleted	TINYINT	[NN]

- **TB_FollowingRecord**

Attribute	Type	Remark
Following RecordID	INT	[PK][AI][NN]
ProposalID	INT	[FK][NN]
MemberID	INT	[FK][NN]
FollowingRecord CreateTime	DATETIME	[NN]
IsDeleted	TINYINT	[NN]

【Store Procedure】_ 總共 12 個

① sp_GetFollowedProposalsByMember

- 因為在 QA 中有表示需輸出 proposal_id，但規格書未更動到。故在此表示，有新增輸出 proposal_id，最終共輸出五欄，member_id、proposal_id、proposal_title、proposal_amount、proposal_goal。

- 程式碼邏輯：透過 inner join 搭配相關條件，找出資料。

② sp_RegisterMember

- 因為使用 transaction rollback 機制，故新增一輸出參數，用於回報 error message：OUT outMessage TEXT
  故包含原本規格書要求，最終共有兩個輸出參數如下：
  OUT outAffectedRowNum int,
  OUT outMessage TEXT

- 程式碼邏輯：先確認是否有相同存在的帳號（email），若沒有，再開始創建，創建過程使用 transaction rollback 機制。

③ sp_UpdatePwd

- 因為使用 transaction rollback 機制，故新增一輸出參數，用於回報 error message：OUT outMessage TEXT
  故包含原本規格書要求，最終共有兩個輸出參數如下：
  OUT outAffectedRowNum int,
  OUT outMessage TEXT

- 程式碼邏輯: 先確認是否存在的帳號，接著確認輸入的 salt 與密碼是否更新，若皆達成條件，則更新資料，更新過程使用 transaction rollback 機制。

④ sp_Login

- 程式碼邏輯: 先確認是否存在的帳號，接著確認輸入的密碼是否正確，並依情形輸出 status code。

⑤ sp_GetProposalsByKeyword

- 程式碼邏輯：透過 LIKE concat 進行模糊查詢。

⑥ sp_UpdateProposalStatus

- 程式碼邏輯：因為 Status 只能按順序更新，故更新只有由 1 變 2，或者是 2 變 3，共兩種可能。另外，若更新為 status 2 則需加入 ongoingdate 與 duedate，duedate 為 ongoingdate 加 90 天。

⑦ sp_GetHistorySponsorByMember

- 程式碼邏輯：透過 inner join 搭配相關條件，找出資料。

⑧ sp_GetUnrepliedComments

- 程式碼邏輯：透過 inner join 搭配相關條件，找出資料。

⑨ sp_GetProposalByCompletionRate

- 程式碼邏輯：透過 derived table 搭配相關條件，找出資料，以 Desc 由大到小排列 Ratio。Ratio 以 TRUNCATE((AccumulatedAmount / Goal), 2) 取小數點後兩位，不四捨五入。

⑩ sp_CreateProposal

- 因為剛創建，故 status 為 1，ongoing_date、due_date 為 null，若需更新，則詳見第六個 sp，即 sp_UpdateProposalStatus。
- 因為使用 transaction rollback 機制，故新增一輸出參數，用於回報 error message：OUT outMessage TEXT
  故包含原本規格書要求，最終共有兩個輸出參數如下：

OUT outAffectedRowNum int,
OUT outMessage TEXT

⑪ sp_GetRecommendedProposals

- 程式碼邏輯：透過 derived table 與 inner join 搭配相關條件，找出符合條件的推薦。先確認有沒有存在符合條件的推薦，若有，則輸出點閱率由高至低排列；若無則輸出推薦點閱率 (viewed_num) 最多的前五項提案並請按照 viewed_num 由高到低排列。

⑫ sp_DeleteMember

- 程式碼邏輯：先確認帳號存在，接著確認該帳號並非仍然屬於 proposal member 之一，若皆符合條件，則開始從資料庫刪除。
- 因為使用 transaction rollback 機制，故新增一輸出參數，用於回報 error message：OUT outMessage TEXT
  故包含原本規格書要求，最終共有兩個輸出參數如下：
  OUT outAffectedRowNum int,
  OUT outMessage TEXT
- 刪除帳號連帶刪除的各項紀錄如下(ER 設計為 CASCADE):
  Membercredential、Followingrecord、SponsorRecord、Comment、ProposalMember。
- FAQ 中若剛好為 Last Editor，設定其值改為 NULL(ER 設計為 SET NULL)。