67-262 Database Design and Development

Phase 2:
Re-Engineering the Early Stages of Zoom's Database

Team C6
Maya Pandurangan, Marko Heinen

Table of Contents

Updated User Stories

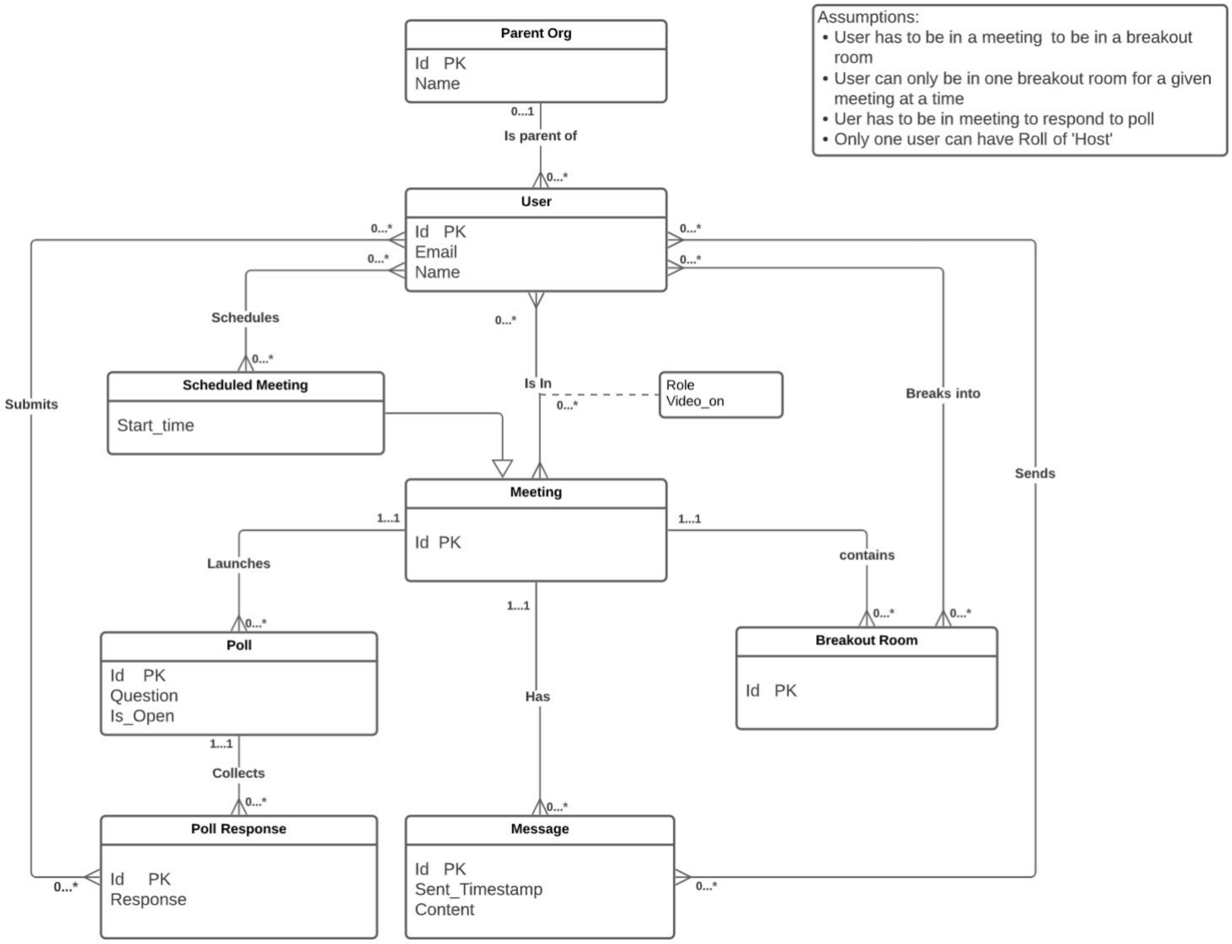
Based on our research on Zoom, we have identified three stakeholders to model user stories around: a student, a teacher, and a project manager. The user stories we have developed (Fig. 1) describe how a user would interact with the database. They are critical as they act as the foundation for the subsequent steps.

Figure 1. Updated User Stories

ID	Event type	As a <role>	I want <goal>	So that <reason>
US1	Complex	Project Manager	I want to be able to find my parent organization.	So that I can know what perks I have.
US2	Analytical	Teacher	I want to be able to count the number or people who responded 'Yes' to my poll	So that I can easily interpret the results and see how many people said yes.
US3	Analytical	Teacher	I want to be able to find the breakout room filled with the most people.	So that I can make sure there aren't too many people in each breakout room at any given point.
US4	Complex	Teacher	I want to find the name of the last person who sent a message.	So that I can see who's been active in the chat most recently.
US5	Complex	Project Manager	I want to create a scheduled meeting	So that I can plan for future events
US6	Complex	Teacher	I want to create a list of people's names who have responded to the poll	So that I can see who participated in the poll
US7	Complex	Student	I want to be able to leave the meeting.	So that I can go attend to other tasks.
US8	Complex	Project Manager	I want to create a message	So that I can talk to others.
US9	Simple	Teacher	I want to be able to edit the start time of a scheduled meeting	So that I can postpone my meeting time
US10	Analytical	Teacher	I want to find the number of people in a given meeting with their camera on. (*new*)	So that I can track video participation in my class.

Updated Conceptual Model

The conceptual model (CM) is a blueprint for how the user stories can be implemented as a database. In our CM (Fig. 2) we have converted all the user stories into entities and relations. The CM is a great planning tool, but without any constraints the CM fails to ensure the integrity of the proposed database. To address this, we have included a set of assumptions about how a user can interact with the database. The assumptions effectively ensure that the integrity of the database will be preserved.



Assumptions:
- User has to be in a meeting to be in a breakout room
- User can only be in one breakout room for a given meeting at a time
- Uer has to be in meeting to respond to poll
- Only one user can have Roll of 'Host'

Figure 2. Updated Conceptual Model

Relational Model

For each entity in our conceptual model we have developed an equivalent relation as listed below.

Relations

Breaks_Into (**person.id**, **breakout_room.id**)

Breakout_Room (**id**, meeting.id)

Is_In (role, video_on, **person.id**, **meeting.id**)

Meeting (**id**)

Message (**id**, sent_timestamp, content, meeting.id)

Parent_Org (**id**, name)

Person (**id**, email, name, background, parent_org.id)

Poll (**id**, question, is_open, meeting.id)

Poll_Response (**id**, response, poll.id)

Schedules (**person.id**, **meeting.id**)

Scheduled_Meeting (**id**, start_time)

Sends (**person.id**, **message.id**)

Submits (**person.id**, **poll_response.id**)

Absorption Process

All entities have associations between them. When an association has a multiplicity of either 0 or 1 at one end then it can be absorbed into the relation of the entity at the opposite end of the association. In this section we indicate how the absorption has been done in our relational model.

Association	**Absorption Process**
Breaks_Into	becomes an entity
Collects	absorbs into Poll_Response (**id**, response, poll.id)
Contains	absorbs into Breakout_Room (**id**, meeting.id)
Has	absorbs into Message (**id**, sent_timestamp, content, meeting.id)
Is_In	becomes an entity
Is_Parent_Of	absorbs into Person (**id**, email, name, background, parent_org.id)
Launches	absorbs into Poll (**id**, question, is_open, meeting.id)
Schedules	becomes an entity
Sends	becomes an entity
Submits	becomes an entity

Functional Dependencies

For each relation, we have listed their functional dependencies and normal form below.

Relation	Functional Dependencies	Normal Form
Breaks_Into (**person.id, breakout_room.id**)	none	BCNF
Breakout_Room (**id**, meeting.id)	{id -> meeting.id}	BCNF
Is_In (role, video_on, **person.id, meeting.id**)	{person.id, meeting_id -> role}	3NF
Meeting (**id**)	none	BCNF
Message (**id**, sent_timestamp, content, meeting.id)	{id-> meeting.id, content, sent_timestamp}	BCNF
Parent_Org (**id**, name)	{id->name}	BCNF
Person (**id**, email, name, background, parent_org.id)	{id->email, name, background, parentOrg.id}	BCNF
Poll (**id**, question, is_open, meeting.id)	{id -> question, is_open, meeting.id}	BCNF
Poll_Response (**id**, response, poll.id)	{id -> poll.id, response}	BCNF
Schedules (**person.id, meeting.id**)	none	BCNF
Scheduled_Meeting (**id**, start_time)	{id -> start_time}	BCNF
Sends (**person.id**, **message.id**)	none	BCNF
Submits (**person.id**, **poll_response.id**)	none	BCNF

Normalized Schema

Using the relations and functional dependencies we made in the previous sections, we then normalized our relations to bring them into BCNF.

Normalizations

Relation	Normalization
Breaks_Into (**person.id**, **breakout_room.id**)	{person.id, breakout_room.id}$^+$ = {person.id, breakoutRoom.id} Is in BCNF because (person.id, breakout_room.id) is a super key for the relation.
Breakout_Room (**id**, meeting.id)	{id}$^+$ = {id,meeting.id} Is in BCNF because id is a super key for the relation.

Is_In (role, video_on, **person.id**, **meeting.id**)	{person.id, meeting.id}$^+$ = {role, person.id, meeting.id}
	
	Since the FD is not a super key, we have to decompose the Is_In relation.
	
	R1(role, video_on, **person.id**, **meeting.id**)
	
	R2(**person.id**, **meeting.id**, role)
	{person.id, meeting.id}$^+$ = {role, person.id, meeting.id}
	Is in BCNF
	
	R3(**person.id**, **meeting.id**, video_on)
	{person.id, meeting.id}$^+$ = {video_on, person.id, meeting.id}
	Is in BCNF
	
	R2 and R3 are the final, decomposed relations in BCNF.

Meeting (**id**)	{id}$^+$ = {id} Is in BCNF because id is a super key for the relation.
Message (**id**, sent_timestamp, content, meeting.id)	{id}$^+$ = {id,meeting.id, content, sent_timestamp} Is in BCNF because id is a super key for the relation.
Parent_Org (**id**, name)	{id}$^+$ = {id, name} Is in BCNF because id is a super key for the relation.

Person (**id**, email, name, background, parent_org.id)	{id}+ = {id, email, name, background, parent_org.id} Is in BCNF because id is a super key for the relation.
Poll (**id**, question, is_open, meeting.id)	{id}$^+$ = {id, question, is_open, meeting.id} Is in BCNF because id is a super key for the relation.
Poll_Response (**id**, response, poll.id)	{id}$^+$ = {id, poll.id, response} Is in BCNF because id is a super key for the relation.
Schedules (**person.id**, **meeting.id**)	{person.id, meeting.id}$^+$ = {person.id, meeting.id} Is in BCNF because (person.id, meeting.id) is a super key for the relation.
Scheduled_Meeting (**id**, start_time)	{id}$^+$ = {id, start_time} Is in BCNF because id is a super key for the relation.
Sends (**person.id**, **message.id**)	{person.id, message.id}$^+$ = {person.id, message.id} Is in BCNF because (person.id, message.id) is a super key for the relation.
Submits (**person.id**, **poll_response.id**)	{person.id, poll_response.id}$^+$ = {person.id, pollresponse.id} Is in BCNF because (person.id, poll_response.id) is a super key for the relation.

Acknowledgements

This project would not have been possible without the support of the 67-626 faculty and staff. We would like to thank Professors Raja and Tu for not just helping us build our understanding of database design and development, but also reinforcing the idea of self-directed learning. Despite the hurdles that the global pandemic has presented, Professors Raja and Tu have managed to deliver a high-quality course. Further, they were there at every step ready to offer any necessary support. We’d also like to thank all the 67-262 Teaching Assistants, especially Matt and Joyce, for their help.