Chosen Business: Discord

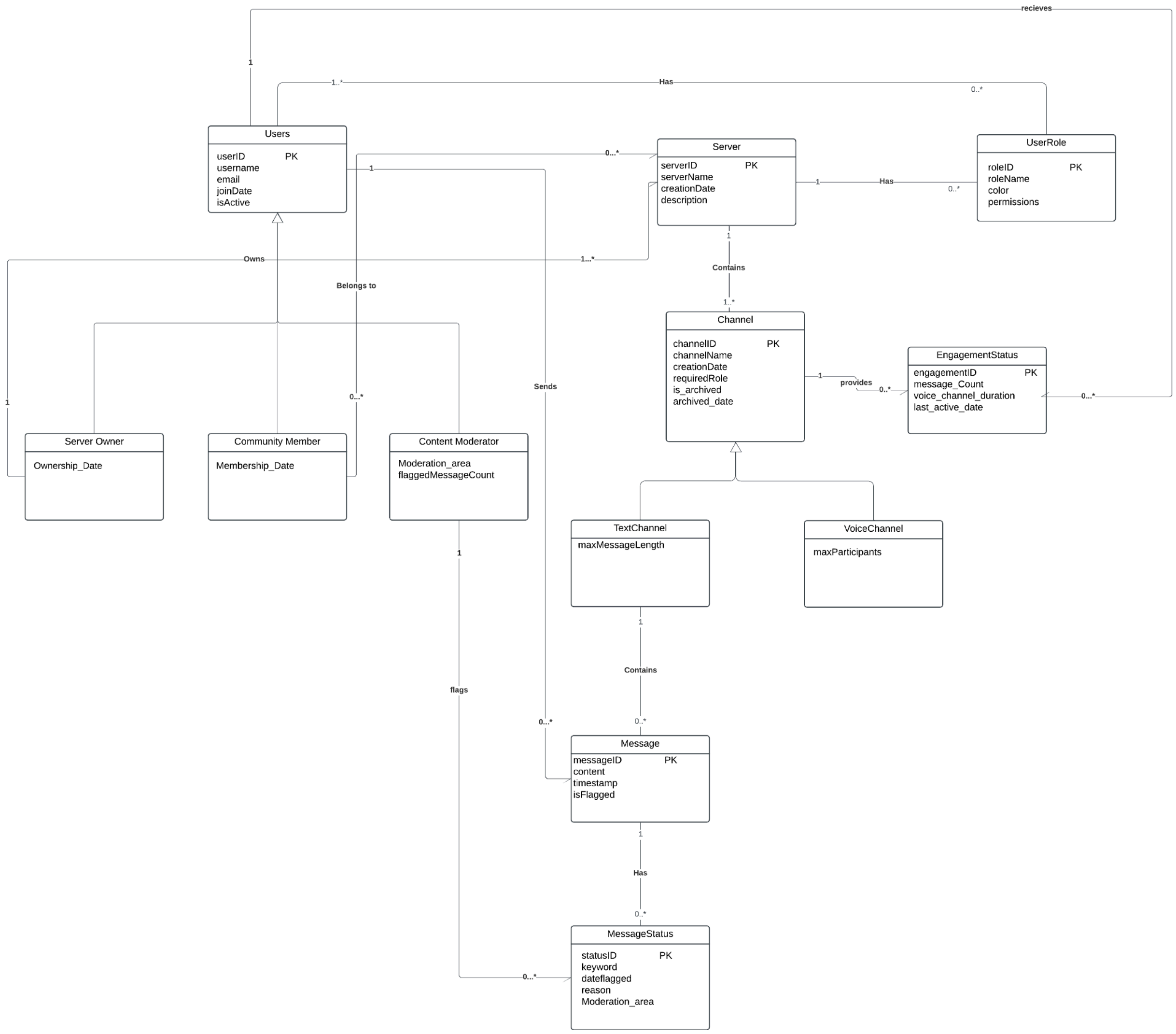
User Stories:

ISD	Verb+Noun	As a <role>	I want <goal>	So that <reason>	Simple / Complex	Operational / Analytical
US1	rank channels	Member	I want to view the most active channels in the server	I can join and participate in high-engagement areas	simple	Analytical
US2	Update permissions within Members	Server Owner	I want to grant or revoke specific permissions for members	so I can quickly manage user capabilities	Simple	Operational
US3	Check voice channel availability	Member	I want to see if there is enough space in a voice channel for new participants	I can decide whether to join the channel or not	Complex	Operational
US4	Aggregate flagged keywords by count	Content Moderator	I want to track the most common keywords flagged in messages across servers	I can monitor them more closely	Simple	Analytical
US5	Identify and move inactive channels to an archive area	Server Owner	I want to automatically archive and categorize inactive channels	I can prevent overcrowding and keep active discussions more accessible	Complex	Operational
US6	Aggregate message counts per channel over a specified period	Server Owner	I want to analyze message volume per channel over time to identify	I can better tailor the server's organization	Simple	Analytical

			which topics drive the most interactions			
US7	Search messages across channels	Member	I want to search for specific keywords within all channels of a server	I can find relevant discussions	Complex	Operational
US8 **(NEW)**	Aggregate message counts and voice channel durations across multiple servers	Member	I want to see my engagement summary across servers	I can track and manage my activity across communities	Complex	Analytical
US9	Flag inappropriate messages and assign them to a moderator based on channel type Content Moderator	Content Moderator	I want to To flag inappropriate messages and assign them to a moderator based on the channel type	Each flagged message is handled by the appropriate moderator responsible for that type	Complex	Operational
US10	Aggregate flagged content across servers and rank by occurrence	Content Moderator	I want to identify servers with the highest rate of flagged content	I can focus my attention on high-risk areas	Complex	Analytical

***Note:** User story 8 represents the new functionality that our team added.

UML Diagram


recieves
1
1..*
Has
0..*
Users
userID PK
username
email
joinDate
isActive
0...*
1
Server
serverID PK
serverName
creationDate
description
1
Has
0..*
UserRole
roleID PK
roleName
color
permissions
Owns
1...*
Belongs to
1
Contains
1..*
Channel
channelID PK
channelName
creationDate
requiredRole
is_archived
archived_date
1
provides
0..*
EngagementStatus
engagementID PK
message_Count
voice_channel_duration
last_active_date
0...*
Sends
1
0...*
Server Owner
Ownership_Date
Community Member
Membership_Date
Content Moderator
Moderation_area
flaggedMessageCount
TextChannel
maxMessageLength
VoiceChannel
maxParticipants
1
1
Contains
flags
0...*
0..*
Message
messageID PK
content
timestamp
isFlagged
1
Has
0..*
0...*
MessageStatus
statusID PK
keyword
dateflagged
reason
Moderation_area

US1:

Relational Model:

Users(**userID**, username, email, joinDate, isActive)
ServerOwner(**userID**, Ownership_Date)
CommunityMember(**userID**, Membership_Date)
ContentModerator(**userID**, Moderation_Area,flaggedMessageCount)
Server(**serverID**, serverName, creationDate, description, userID)
UserRole(**roleID**, roleName, color, permissions, serverID)
EngagementStatus(**engagementID**, message_Count, voice_channel_duration, last_active_date, userID, channelID)
Channel(**channelID**, channelName, creationDate, requiredRole,is_archived, archived_date serverID)
TextChannel(**channelID**, maxMessageLength)
VoiceChannel(**channelID**, maxParticipants)
Message(**messageID**, content, timestamp, isFlagged, userID, channelID)
MessageStatus(**statusID**, keyword, dateflagged, reason, Moderation_Area, Moderator_ID, messageID)
UserRolesAndUser(**userID,roleID**)
CommunityMemberAndServer(**userID,ServerID**)

Note*: we used bold text for the Primary key. Single underline for foreign key. Bold text + single underline means it is both a Primary key and a Foreign key.

Functional Dependencies:

Users:
Relation: Users(**userID**, username, email, joinDate, isActive)
FDs:

- userID → username, email, joinDate, isActive

Assumptions:

- The userID can uniquely identify all necessary information about the user.
- All attributes on the RHS are entirely dependent on the userID.

Normal Form: BCNF

- Because userID is the PK, it is also a superkey. The other attributes of the Users relation are all functionally dependent on userID. Additionally, there are no other FDs from the other relations that overlap with this one to violate BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

ServerOwner:

Relation: ServerOwner(**userID**, serverID, Ownership_Date)

FDs:

- userID, serverID → Ownership_Date

Assumptions:

- The same user can own more than one server, but each server is owned by one unique owner. Therefore, the unique user and unique server together can determine the ownership date (userID, serverID → Ownership_Date).

Normal Form: BCNF

- Because userID and serverID make up the composite PK, they are each a superkey. The other attribute in the ServerOwner relation is Ownership_Date, which is fully functionally dependent on userID and serverID. Additionally, there are no other FDs that have a determinant that is not a superkey and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

CommunityMember:

Relation: CommunityMember(**userID**, Membership_Date)

FDs:

- userID → Membership_Date

Assumptions:

- A community member can only be uniquely identified by their userID.
- Membership_Date is fully dependent on the userID.

Normal Form: BCNF

- Because userID is the PK, it is also a superkey. Membership_Date is the other attribute in the CommunityMember relation, and it is fully functionally dependent on userID. Additionally, there are no other FDs in the relation that overlap with this one to violate BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

ContentModerator:

Relation: ContentModerator(**userID**, Moderation_Area,flaggedMessageCount)

FDs:

- userID → Moderation_Area,flaggedMessageCount

Assumptions:

- A moderator can only be uniquely identified by their userID.
- Moderation_Area and flaggedMessageCount is fully dependent on the userID.

Normal Form: BCNF

- Because userID is the PK, it is also a superkey. Moderation_Area is the other attribute in the ContentModerator relation, and it is fully functionally dependent on userID. Additionally, there are no other FDs in the relation that overlap with this one to violate

BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

Server:

Relation: Server(**serverID**, serverName, creationDate, description, *userID*)

FDs:

- serverID → serverName, creationDate, description, userID

Assumptions:

- A server can be uniquely identified by its serverID.
- Because userID correlates to the owner of that server and a server is owned by only one user, serverID can determine userID.

Normal Form: BCNF

- Because serverID is the PK, it is also a superkey. The other attributes in the Server relation are fully functionally dependent on serverID. Additionally, there are no other FDs with a determinant that is not a superkey. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

UserRole:

Relation: UserRole(**roleID**, roleName, color, permissions, serverID)

FDs:

- roleID → roleName, color, permissions, serverID

Assumptions:

- A role can be uniquely identified by its roleID.
- Each role is correlated with a specific server.

Normal Form: BCNF

- Because roleID is the PK, it is also a superkey. The other attributes in the UserRole relation are fully functionally dependent on roleID. Additionally, there are no other FDs that overlap and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

EngagementStatus:

Relation: EngagementStatus(**engagementID**, message_Count, voice_channel_duration, last_active_date, userID, channelID)

FDs:

- engagementID → message_Count, voice_channel_duration, last_active_date, userID, channelID

Assumptions:

- Engagement status can be uniquely identified by its engagementID.
- By knowing the specific user and the specific channel, we can determine the engagement status.

Normal Form: BCNF

- Because engagementID is the PK, it is also a superkey. The other attributes in the EngagementStatus relation are fully functionally dependent on engagementID. Additionally, there are no other FDs that overlap and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

<u>Channel:</u>

Relation: Channel(**channelID**, channelName, creationDate, requiredRole, is_archived, archived_date, <u>serverID</u>)

FDs:

- channelID → channelName, creationDate, requiredRole, serverID

Assumptions:

- A channel can be uniquely identified by its channelID.
- Each unique channel belongs to one unique server.

Normal Form: BCNF

- Because channelID is the PK, it is also a superkey. The other attributes in the Channel relation are fully functionally dependent on roleID. Additionally, there are no other FDs that have a determinant that is not a superkey and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

<u>TextChannel:</u>

Relation: TextChannel(<u>**channelID**</u>, maxMessageLength)

FDs:

- channelID → maxMessageLength

Assumptions:

- A channel can be uniquely identified by its channelID.
- The maxMessageLength depends on the unique text channel that it belongs to.

Normal Form: BCNF

- Because channelID is the PK, it is also a superkey. The only other attribute is maxMessageLength in the TextChannel relation, which is fully functionally dependent on channelID. Additionally, there are no other FDs that have a determinant that is not a superkey and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

<u>VoiceChannel:</u>

Relation: VoiceChannel(<u>**channelID**</u>, maxParticipants)

FDs:

- channelID → maxParticipants

Assumptions:

- A channel can be uniquely identified by its channelID.
- The maxParticipants depends on the unique voice channel that it belongs to.

Normal Form: BCNF

- Because channelID is the PK, it is also a superkey. The only other attribute is maxParticipants in the VoiceChannel relation, which is fully functionally dependent on channelID. Additionally, there are no other FDs that have a determinant that is not a superkey and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

Message:

Relation: Message(**messageID**, content, timestamp, isFlagged, userID, channelID)

FDs:

- messageID → content, timestamp, isFlagged, userID, channelID

Assumptions:

- A message can be uniquely identified by its messageID.
- The specific message can have only been sent by one user, identified by userID, in one specific channel, identified by channelID.

Normal Form: BCNF

- Because messageID is the PK, it is also a superkey. All the other attributes in the Message relation are fully functionally dependent on messageID. Additionally, there are no other FDs that have a determinant that is not a superkey and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

MessageStatus:

Relation: MessageStatus(**statusID**, keyword, dateflagged, reason, Moderation_Area, Moderator_ID, messageID)

FDs:

- statusID → keyword, dateflagged, reason, Moderator_ID, messageID

Assumptions:

- A message's status can be uniquely identified by the statusID.
- If a certain message has a status associated with it, the messageID can determine the associated unique status (messageID → statusID).

Normal Form: BCNF

- Because statusID is the PK, it is also a superkey. All the other attributes in the MessageStatus relation are fully functionally dependent on statusID. Additionally, there are no other FDs that have a determinant that is not a superkey and violate the BCNF. There are no partial or transitive dependencies. Therefore, this relation is in BCNF because the FD has a determinant that is a superkey.

Normalization:

If BCNF *is written next to the name of the relation, it means that the relation is already in BCNF and does not require any decomposition.*

- **Users:** BCNF
- **ServerOwner:** BCNF
- **CommunityMember:** BCNF
- **ContentModerator:** BCNF
- **Server:** BCNF
- **UserRole:** BCNF
- **EngagementStatus:** BCNF
- **Channel:** BCNF
- **TextChannel:** BCNF
- **VoiceChannel:** BCNF
- **Message:** BCNF
- **MessageStatus:** BCNF