Creating a database for a messaging app like WhatsApp requires a well-structured schema to manage users, messages, groups, permissions, and chat features. Below is the **database design** along with **table structures and explanations**.

Database Name: `messaging_app`

1. Users Table

Stores user details.

Table: users

Column Name	Data Type	Constraints	Description
`user_id`	INT (PK)	AUTO_INCREMENT	Unique user ID
`username`	VARCHAR(50)	UNIQUE, NOT NULL	Username of the user
`email`	VARCHAR(100)	UNIQUE, NOT NULL	Email of the user
`phone`	VARCHAR(15)	UNIQUE, NOT NULL	Phone number
`password`	VARCHAR(255)	NOT NULL	Hashed password
`status`	VARCHAR(255)	NULL	Status message
`profile_pic`	TEXT	NULL	Profile picture URL
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Account creation date

---

2. Contacts Table

Stores user contacts.

Table: contacts

Column Name	**Data Type**	**Constraints**	**Description**
`contact_id`	INT (PK)	AUTO_INCREMENT	Unique contact ID
`user_id`	INT (FK)	NOT NULL	Reference to `users` table
`contact_user_id`	INT (FK)	NOT NULL	Reference to `users` table (contact person)
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Added date

Constraints:

- `user_id` and `contact_user_id` together should be unique to prevent duplicate contacts.

---

3. Messages Table

Stores user messages (one-to-one and group).

Table: messages

Column Name	Data Type	Constraints	Description
`message_id`	INT (PK)	AUTO_INCREMENT	Unique message ID
`sender_id`	INT (FK)	NOT NULL	User sending the message
`receiver_id`	INT (FK)	NULL	User receiving the message (NULL if group)
`group_id`	INT (FK)	NULL	Reference to group chat (NULL for one-to-one)
`message_text`	TEXT	NULL	Message text
`message_type`	ENUM	('text','image','video','audio','file')	Type of message
`media_url`	TEXT	NULL	URL of media file (if applicable)
`is_read`	BOOLEAN	DEFAULT FALSE	Read status
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Sent time

Constraints:

- Either `receiver_id` (for one-to-one chat) or `group_id` (for group chat) must be provided.
- Foreign keys reference the `users` and `groups` tables.

---

4. Groups Table

Stores group details.

Table: groups

Column Name	Data Type	Constraints	Description
`group_id`	INT (PK)	AUTO_INCREMENT	Unique group ID
`group_name`	VARCHAR(255)	NOT NULL	Name of the group
`admin_id`	INT (FK)	NOT NULL	User ID of the group admin
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Creation date

---

5. Group Members Table

Stores group participants.

Table: group_members

Column Name	Data Type	Constraints	Description
`member_id`	INT (PK)	AUTO_INCREMENT	Unique member ID
`group_id`	INT (FK)	NOT NULL	Reference to `groups` table
`user_id`	INT (FK)	NOT NULL	User ID of the participant
`role`	ENUM	('admin','member')	User role in the group
`joined_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Joined time

Constraints:

- A user can be part of multiple groups.
- Each group must have at least one admin.

---

6. Chat Settings Table

Stores chat-specific settings for users.

Table: chat_settings

Column Name	Data Type	Constraints	Description
`setting_id`	INT (PK)	AUTO_INCREMENT	Unique setting ID
`user_id`	INT (FK)	NOT NULL	Reference to `users` table
`chat_type`	ENUM	('one_to_one', 'group', 'secret')	Type of chat
`chat_id`	INT (FK)	NOT NULL	Chat reference (group_id or user_id)
`muted`	BOOLEAN	DEFAULT FALSE	Chat muted status
`archived`	BOOLEAN	DEFAULT FALSE	Chat archived status

Constraints:

- Helps in managing chat visibility, mute settings, etc.

---

7. Secret Chats Table

Stores secret chats with encryption.

Table: secret_chats

Column Name	Data Type	Constraints	Description
`chat_id`	INT (PK)	AUTO_INCREMENT	Unique secret chat ID
`user_one`	INT (FK)	NOT NULL	First user ID
`user_two`	INT (FK)	NOT NULL	Second user ID
`encryption_key`	TEXT	NOT NULL	Encryption key for chat

---

8. Blocked Users Table

Stores blocked contacts.

Table: blocked_users

Column Name	Data Type	Constraints	Description
`block_id`	INT (PK)	AUTO_INCREMENT	Unique block ID
`user_id`	INT (FK)	NOT NULL	User who blocked someone
`blocked_user_id`	INT (FK)	NOT NULL	Blocked user ID
`blocked_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Blocked time

Database Relationships

1. **Users Table** → Connected with `messages`, `contacts`, `blocked_users`, `groups`, and `chat_settings`.
2. **Messages Table** → Connected with `users` (sender, receiver) and `groups`.
3. **Groups Table** → Connected with `group_members`, `messages`.
4. **Secret Chats** → Uses encryption for private messaging.
5. **Blocked Users** → Stores blocked contacts.
6. **Chat Settings** → Stores muted/archived chats.

Summary

This database structure supports: ✅ User management
✅ One-to-one messaging
✅ Group messaging
✅ Secret chats with encryption
✅ Message media support (images, videos, files)
✅ Mute, archive, and disable chat options
✅ User blocking system

Yes! These are the **core database models** needed to build a messaging app like WhatsApp. However, depending on additional features you may want, we can extend this schema. Below are some **additional models** that could enhance the app:

Additional Database Models (Optional Enhancements)

9. Message Reactions Table

Stores reactions (like 👍❤□😂) to messages.

Table: message_reactions

Column Name	Data Type	Constraints	Description
`reaction_id`	INT (PK)	AUTO_INCREMENT	Unique reaction ID
`message_id`	INT (FK)	NOT NULL	Reference to `messages` table
`user_id`	INT (FK)	NOT NULL	User who reacted
`reaction`	ENUM	('like','love','haha','wow','sad','angry')	Type of reaction
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Time of reaction

10. Message Read Receipts Table

Stores read/delivered status of messages.

Table: message_read_receipts

Column Name	Data Type	Constraints	Description
`receipt_id`	INT (PK)	AUTO_INCREMENT	Unique ID
`message_id`	INT (FK)	NOT NULL	Reference to `messages` table
`user_id`	INT (FK)	NOT NULL	User who read the message
`status`	ENUM	('delivered', 'read')	Read or delivered status
`timestamp`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Time of status update

11. Calls Table

Stores audio/video call logs.

Table: calls

Column Name	Data Type	Constraints	Description
`call_id`	INT (PK)	AUTO_INCREMENT	Unique call ID
`caller_id`	INT (FK)	NOT NULL	Caller user ID
`receiver_id`	INT (FK)	NOT NULL	Receiver user ID
`call_type`	ENUM	('audio', 'video')	Type of call
`call_status`	ENUM	('missed', 'completed', 'declined')	Status of the call
`duration`	INT	DEFAULT 0	Duration in seconds
`started_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Call start time

---

12. Call Participants Table

Stores participants for group calls.

Table: call_participants

Column Name	Data Type	Constraints	Description
`participant_id`	INT (PK)	AUTO_INCREMENT	Unique participant ID
`call_id`	INT (FK)	NOT NULL	Reference to `calls` table
`user_id`	INT (FK)	NOT NULL	User ID of participant
`joined_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	When they joined

---

13. Notifications Table

Stores push notifications for messages, calls, etc.

Table: notifications

Column Name	Data Type	Constraints	Description
`notification_id`	INT (PK)	AUTO_INCREMENT	Unique notification ID
`user_id`	INT (FK)	NOT NULL	User receiving the notification
`type`	ENUM	('message', 'call', 'group_invite')	Type of notification
`message`	TEXT	NOT NULL	Notification content
`is_read`	BOOLEAN	DEFAULT FALSE	Read status
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Time of notification

---

14. Group Invites Table

Stores pending group invitations.

Table: group_invites

Column Name	Data Type	Constraints	Description
`invite_id`	INT (PK)	AUTO_INCREMENT	Unique invite ID
`group_id`	INT (FK)	NOT NULL	Reference to `groups` table
`inviter_id`	INT (FK)	NOT NULL	User who sent the invite
`invitee_id`	INT (FK)	NOT NULL	User receiving the invite
`status`	ENUM	('pending', 'accepted', 'declined')	Invite status
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Time of invite

---

15. Status Updates Table (Like WhatsApp Stories)

Stores status updates (text, images, videos).

Table: status_updates

Column Name	Data Type	Constraints	Description
`status_id`	INT (PK)	AUTO_INCREMENT	Unique status ID
`user_id`	INT (FK)	NOT NULL	User who posted the status
`status_text`	TEXT	NULL	Text status (optional)
`media_url`	TEXT	NULL	URL of image/video
`visibility`	ENUM	('public', 'contacts', 'private')	Who can see it
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Time of posting
`expires_at`	TIMESTAMP	NULL	Expiry time (24 hours default)

---

16. Status Views Table

Stores who viewed a status.

Table: status_views

Column Name	Data Type	Constraints	Description
`view_id`	INT (PK)	AUTO_INCREMENT	Unique view ID
`status_id`	INT (FK)	NOT NULL	Reference to `status_updates` table
`viewer_id`	INT (FK)	NOT NULL	User who viewed the status
`viewed_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Time of viewing

---

17. Admin Reports Table

Stores reports about users, groups, or messages.

Table: reports

Column Name	Data Type	Constraints	Description
`report_id`	INT (PK)	AUTO_INCREMENT	Unique report ID
`reported_by`	INT (FK)	NOT NULL	User who filed the report
`reported_user`	INT (FK)	NULL	Reported user (if applicable)
`reported_group`	INT (FK)	NULL	Reported group (if applicable)
`reported_message`	INT (FK)	NULL	Reported message (if applicable)
`reason`	TEXT	NOT NULL	Reason for the report
`status`	ENUM	('pending', 'resolved', 'dismissed')	Current status
`created_at`	TIMESTAMP	DEFAULT CURRENT_TIMESTAMP	Time of report

Final Thoughts

- The **original core models** cover essential **messaging features**.
- The **additional models** handle **reactions, calls, notifications, statuses, and admin reports**.
- This design is **scalable**, meaning you can **add more features** in the future without breaking existing functionality.