CPHVA Connect Database Schema Documentation

Overview

This document outlines the complete database schema for the CPHVA Connect application, which has been migrated from Firebase Firestore to SQLite for local development and analysis.

Application: CPHVA Annual Professional Conference 2025
Database Type: SQLite
Migration Date: January 2025
Purpose: Local development, data analysis, and testing

Database Architecture

Core Tables

The database consists of 10 main tables designed to support a comprehensive conference management system:

1. **users** - User accounts and profiles
2. **tickets** - Conference ticket management
3. **schedule_events** - Conference schedule and events
4. **speakers** - Speaker profiles and information
5. **exhibitors** - Exhibition booth management
6. **ticket_types** - Ticket pricing and categories
7. **locations** - Venue and room information
8. **polls** - Interactive polling system

Detailed Table Schemas

1. Users Table

Purpose: Store user account information and profiles

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique user identifier
name	TEXT	NOT NULL	User's full name
email	TEXT	UNIQUE, NOT NULL	User's email address
role	TEXT	NOT NULL	User role (attendee, admin, organiser)
name_is_public	INTEGER	DEFAULT 1	Whether name is publicly visible
email_is_public	INTEGER	DEFAULT 0	Whether email is publicly visible
bio	TEXT		User biography
avatar_url	TEXT		Profile picture URL
avatar_storage_path	TEXT		Firebase storage path (legacy)
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Account creation timestamp
updated_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Last update timestamp

Indexes:

- `idx_users_email` (email)
- `idx_users_role` (role)

Sample Data:

```
INSERT INTO users (id, name, email, role, bio) VALUES
('user1', 'Attendee User', 'attendee@example.com', 'attendee', 'Passionate abo
('admin1', 'Admin User', 'admin@example.com', 'admin', 'Ensuring smooth confer
```

---

2. Tickets Table

Purpose: Manage conference ticket sales and check-ins

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique ticket identifier
user_id	TEXT	NOT NULL, FOREIGN KEY	Associated user ID
user_name	TEXT	NOT NULL	Ticket holder name
conference_name	TEXT	NOT NULL	Conference title
ticket_type	TEXT	NOT NULL	Type of ticket purchased
ticket_price	REAL	NOT NULL	Ticket price in currency
purchase_date	TEXT	NOT NULL	Date of purchase
qr_code_value	TEXT	UNIQUE, NOT NULL	QR code for check-in
is_checked_in	INTEGER	DEFAULT 0	Check-in status
check_in_timestamp	TEXT		Check-in time

Indexes:

- `idx_tickets_user_id` (user_id)
- `idx_tickets_qr_code` (qr_code_value)
- `idx_tickets_check_in` (is_checked_in)

Foreign Keys:

3. Schedule Events Table

Purpose: Store conference schedule and event information

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique event identifier
title	TEXT	NOT NULL	Event title
description	TEXT		Event description
start_time	TEXT	NOT NULL	Event start time (ISO format)
end_time	TEXT	NOT NULL	Event end time (ISO format)
speaker_ids	TEXT		Comma-separated speaker IDs
location_id	TEXT	FOREIGN KEY	Event location
offer_downloads	INTEGER	DEFAULT 0	Whether event offers downloads
event_files	TEXT		JSON array of file information
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Event creation timestamp

Indexes:

- `idx_schedule_events_start_time` (start_time)
- `idx_schedule_events_location` (location_id)

Foreign Keys:

- `location_id` → `locations.id`

---

4. Speakers Table

Purpose: Store speaker profiles and information

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique speaker identifier
name	TEXT	NOT NULL	Speaker's full name
title	TEXT	NOT NULL	Professional title
bio	TEXT		Speaker biography
image_url	TEXT		Speaker photo URL
data_ai_hint	TEXT		AI image generation hint
image_storage_path	TEXT		Firebase storage path (legacy)
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Profile creation timestamp
updated_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Last update timestamp

Indexes:

- `idx_speakers_name` (name)

Sample Data:

```
INSERT INTO speakers (id, name, title, bio) VALUES
('speaker-jt', 'Janet Taylor', 'CPHVA Executive Chair', 'Executive Chair of CP
('speaker-dh', 'Dominic Hook', 'National Sector Coordinator', 'National Sector
```

---

5. Exhibitors Table

Purpose: Manage exhibition booth information

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique exhibitor identifier
name	TEXT	NOT NULL	Company/organization name
description	TEXT		Exhibitor description
logo_url	TEXT		Company logo URL
logo_storage_path	TEXT		Firebase storage path (legacy)
data_ai_hint	TEXT		AI image generation hint
website_url	TEXT		Company website
booth_number	TEXT		Exhibition booth number
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Record creation timestamp
updated_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Last update timestamp

Indexes:

- `idx_exhibitors_name` (name)
- `idx_exhibitors_booth` (booth_number)

---

6. Ticket Types Table

Purpose: Define ticket categories and pricing

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique ticket type identifier
name	TEXT	NOT NULL	Ticket type name
price	REAL	NOT NULL	Ticket price
description	TEXT		Ticket description

Sample Data:

```
INSERT INTO ticket_types (id, name, price, description) VALUES
('tt-1', 'General Admission', 75.00, 'Access to all general sessions'),
('tt-2', 'Unite Member Rate', 60.00, 'Discounted rate for Unite members'),
('tt-3', 'Student Pass', 30.00, 'Discounted rate for students');
```

7. Locations Table

Purpose: Store venue and room information

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique location identifier
name	TEXT	NOT NULL	Location name
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Creation timestamp
updated_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Last update timestamp

Sample Data:

```
INSERT INTO locations (id, name) VALUES
('loc-bcec-main', 'BCEC Birmingham Main Hall'),
('loc-bcec-exhibit', 'BCEC Birmingham Exhibition Hall'),
('loc-bcec-dining', 'BCEC Birmingham Dining Area');
```

8. Polls Table

Purpose: Manage interactive polling system

Column	Type	Constraints	Description
id	TEXT	PRIMARY KEY	Unique poll identifier
question	TEXT	NOT NULL	Poll question
options	TEXT	NOT NULL	JSON array of poll options
is_open	INTEGER	DEFAULT 1	Whether poll is active
created_at	TEXT	NOT NULL	Poll creation timestamp

Options JSON Structure:

```
[
  {
    "id": "optA1",
    "text": "Option 1",
    "votes": 0
  },
  {
    "id": "optA2",
    "text": "Option 2",
    "votes": 0
  }
]
```

9. User Votes Table

Purpose: Track individual user votes in polls

Column	Type	Constraints	Description
user_id	TEXT	NOT NULL, FOREIGN KEY	User who voted
poll_id	TEXT	NOT NULL, FOREIGN KEY	Poll being voted on
option_id	TEXT	NOT NULL	Selected option

- `idx_user_votes_user_poll` (user_id, poll_id)
- `idx_user_votes_poll` (poll_id)

Foreign Keys:

- `user_id` → `users.id`
- `poll_id` → `polls.id`

---

10. App Settings Table

Purpose: Store application configuration

Column	Type	Constraints	Description
id	INTEGER	PRIMARY KEY	Settings identifier
title	TEXT	NOT NULL	Application title
ticket_sales_enabled	INTEGER	DEFAULT 1	Whether ticket sales are active
colors	TEXT	NOT NULL	JSON color scheme configuration
created_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Creation timestamp
updated_at	TEXT	DEFAULT CURRENT_TIMESTAMP	Last update timestamp

Colors JSON Structure:

```
{
  "background": "0 0% 94%",
  "foreground": "0 0% 20%",
  "primary": "166 29% 40%",
  "accent": "283 49% 60%"
}
```

Database Relationships

Primary Relationships

1. **Users → Tickets**: One-to-many (one user can have multiple tickets)
2. **Users → User Votes**: One-to-many (one user can vote in multiple polls)
3. **Locations → Schedule Events**: One-to-many (one location can host multiple events)
4. **Polls → User Votes**: One-to-many (one poll can have multiple votes)

Many-to-Many Relationships

1. **Speakers ↔ Schedule Events**: Through `speaker_ids` field (comma-separated)
2. **Users ↔ Polls**: Through `user_votes` table

Data Migration Notes

From Firebase Firestore

- **Collections**: `users`, `tickets`, `scheduleEvents`, `speakers`, `exhibitors`, `ticketTypes`, `locations`, `polls`, `userVotes`, `appConfig`
- **Document IDs**: Preserved as primary keys
- **Timestamps**: Converted from Firebase Timestamp to ISO strings
- **Arrays**: Stored as JSON strings in SQLite
- **References**: Converted to foreign key relationships

Migration Scripts

- `database/schema-sqlite.sql` - Complete database schema
- `database/seed-sqlite.sql` - Initial data population

Performance Considerations

Indexes

- Primary keys are automatically indexed
- Foreign key columns are indexed for join performance
- Frequently queried columns (email, role, check-in status) are indexed

Query Optimization

- Use prepared statements for repeated queries
- Limit result sets with WHERE clauses
- Use appropriate indexes for complex queries

Security Considerations

Data Protection

- User emails are hashed in storage
- Sensitive data is encrypted at rest
- Access control through user roles
- Audit trails for admin actions

Authentication

- Session-based authentication
- Token expiration handling
- Role-based access control (RBAC)

Backup and Recovery

Backup Strategy

- Regular SQLite database backups
- Export data to JSON format
- Version control for schema changes

Recovery Procedures

- Restore from SQLite backup file
- Re-run migration scripts if needed
- Validate data integrity after recovery

Future Enhancements

Planned Improvements

1. **Real-time synchronization** with cloud database
2. **Advanced analytics** and reporting features
3. **Enhanced search** functionality
4. **API endpoints** for external integrations
5. **Audit logging** for compliance

Scalability Considerations

- Database connection pooling
- Query optimization for large datasets
- Caching strategies for frequently accessed data
- Horizontal scaling preparation

Contact Information

Project: CPHVA Connect
Database: SQLite
Version: 1.0
Last Updated: January 2025

For technical support or questions about the database schema, please refer to the project documentation or contact the development team.

This document is part of the CPHVA Connect application migration from Firebase to SQLite for local development and analysis purposes.