Problem Session #04 ER Diagram Exercise and Modify Existing Database

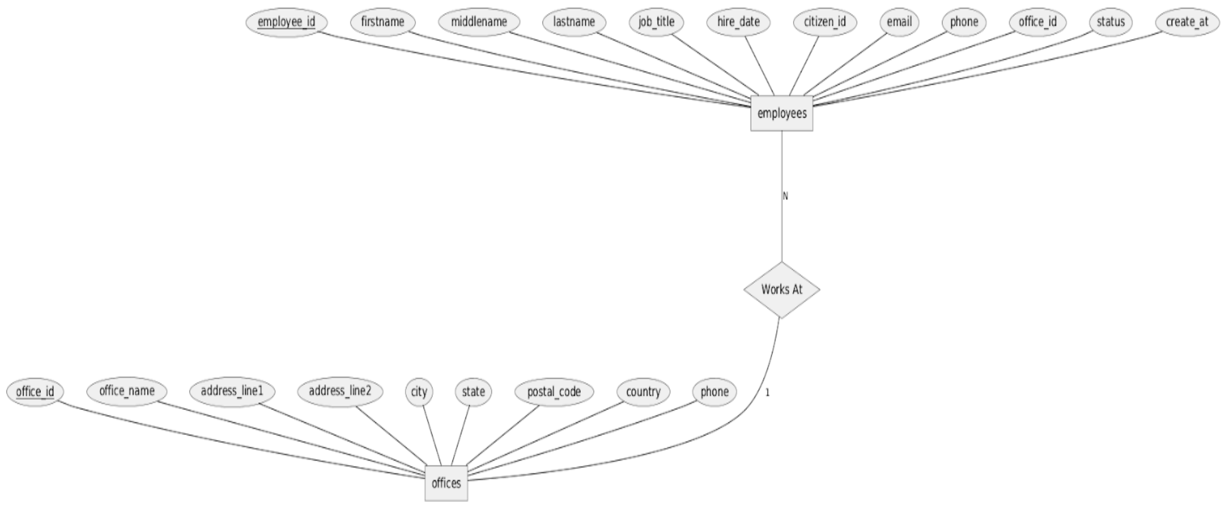
สมาชิกกลุ่ม	
กันต์ธีร์ ดวงมณี	67070501003
นัธทวัฒน์ ปริมสิริคุณาวุฒิ	67070501027
ภานุวัฒน์ บุญศักดิ์	67070501034
รัตนนันทน์ สิริพลวัฒน์	67070501038
วิศิษฐ์ สุวรรณเนาว์	67070501042
ศุภวิชญ์ มารยาท	67070501045
พลวริษฐ์ วัฒนเหมรัตน์	67070501067

Task 1: ER Diagram Exercise

01: A law firm wants to keep employees and their offices.

1. employees(employee_id, firstname, middlename, lastname, job_title, hire_date, citizen_id, email, phone, office_id, status, create_at)
2. offices(office_id, office_name, address_line1, address_line2, city, state, postal_code, country, phone)

Explanation:
None of Weak entities. Because they have own primary key that is globally unique within the system

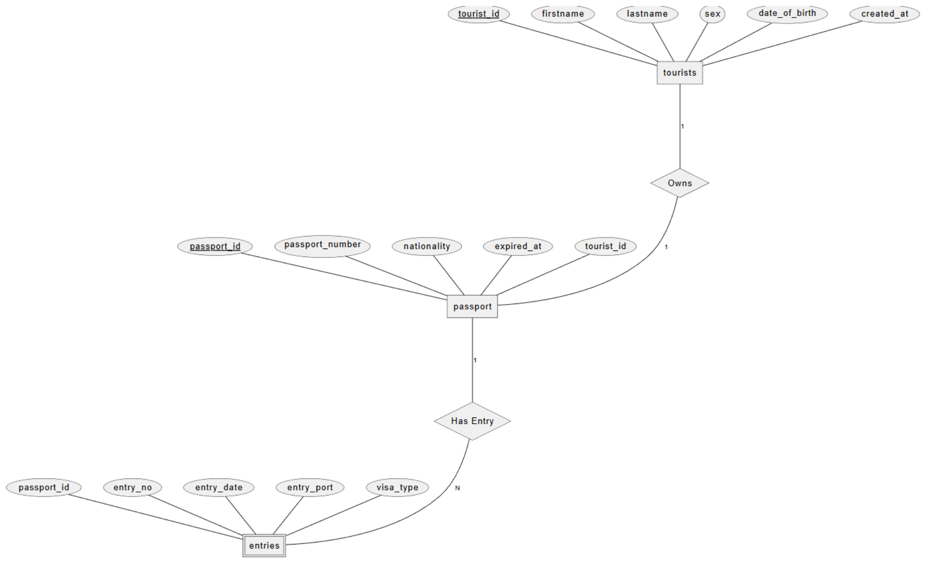


https://drive.google.com/file/d/1dSbx4mv7PEiHi9yLp7nGiUq6Qy-2Ey73/view?usp=sharing
รูปภาพแสดง Chen ERD with Weak Entities ข้อ 1

02: Thai customs want to keep tourists and their passports that they used to enter Thailand.

1.tourists(tourist_id, firstname, lastname, sex, date_of_birth, created_at)
2.passport(passport_id, passport_number, nationality, expired_at, tourist_id)
3.entries(passport_id, entry_no, entry_date, entry_port, visa_type) [Weak-Entities]

Explanation: tourist_id is used as the Primary Key to link with the tourist. It relies on passport_id to identify who it belongs to, and since it cannot exist independently, it is considered a Weak Entity.

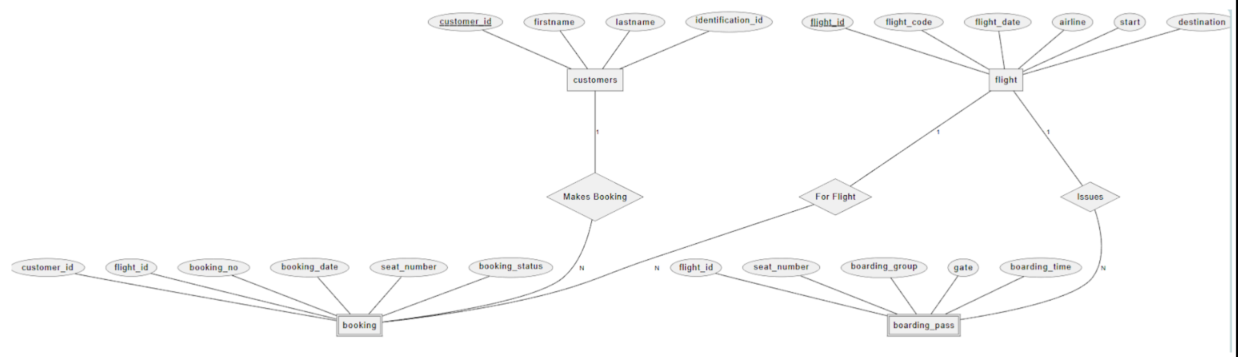


รูปภาพแสดง Chen ERD with Weak Entities ข้อ 2
https://drive.google.com/file/d/1Ybpjk26txMNKj4TjCZWjm6EnUNdB5ADg/view?usp=sharing
URL Chen ERD ข้อ 2

03: The airline wants to retain its customer base and customer's flight booking.

1.customers(customer_id, firstname, lastname, identification_id)
2.flight(flight_id, flight_code, flight_date, airline, start, destination)
3.booking(customer_id, flight_id, booking_no, booking_date, seat_number, booking_status) [Weak-Entities]
4.boarding_pass(flight_id, seat_number, boarding_group, gate, boarding_time)
[Weak-Entities]

Explanation:Customers use customer_id as the Primary Key. Flights use flight_id as the Primary Key. Booking is a Weak Entity because it relies on customer_id and flight_id to identify a reservation. Boarding_pass is a Weak Entity because it relies on flight_id and seat_number to identify a boarding pass.

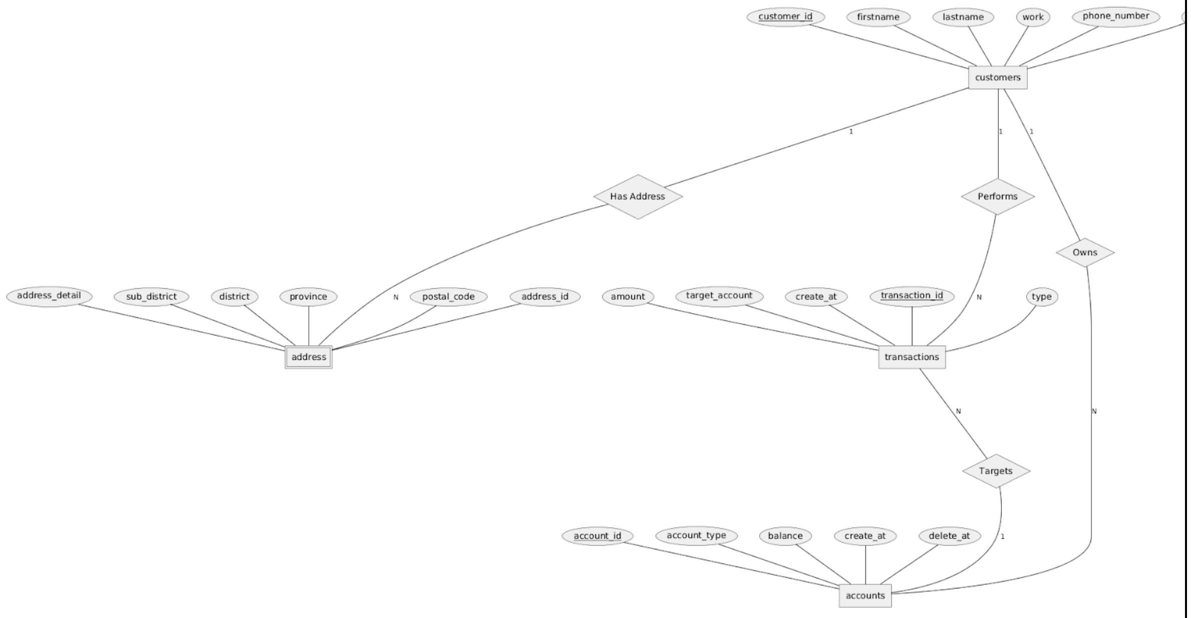


รูปภาพแสดง Chen ERD with Weak Entities ข้อ 3
https://drive.google.com/file/d/107YkrMiagy4IJtcDvWazDUC0Hpdcju-g/view?usp=sharing
URL Chen ERD ข้อ 3

04: A bank wants to keep customers and their accounts.

1. customers(customer_id,firstname,lastname,work,phone_number,email,address)
2. accounts(account_id, account_type, balance, create_at, delete_at)
3. address(address_id, customer_id, address_detail, sub_district, district, province, postal_code) [Weak-Entities] (depends on customers)
4. transaction(transaction_id, customer_id, type, amount, target_account, create_at)

Explanation:
address is a weak entity because it cannot be uniquely identified on its own and depends on **customers**. It uses **customer_id** as a foreign key and is identified in the context of a specific customer, meaning an address cannot exist without an associated customer.

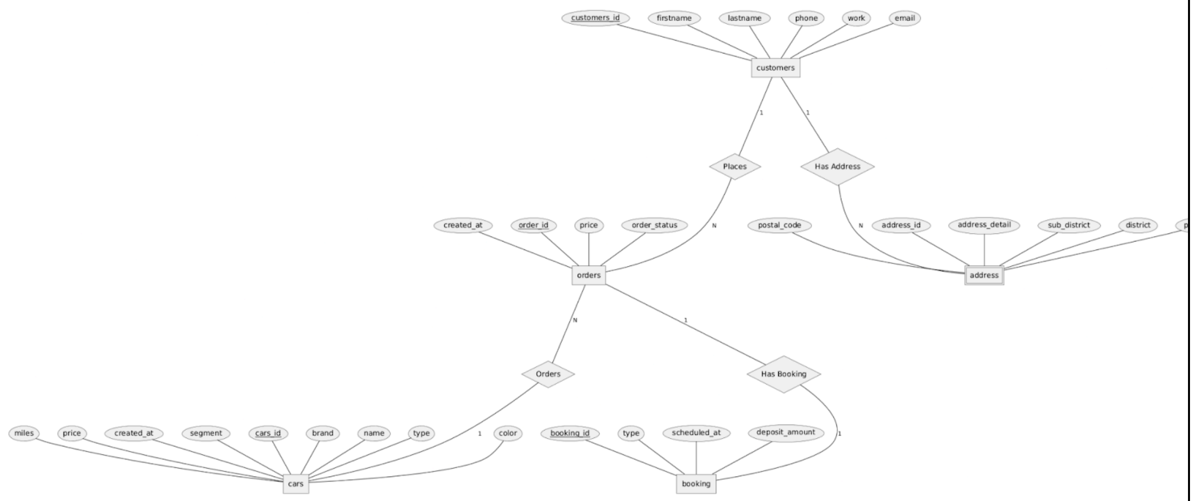


รูปภาพแสดง Chen ERD with Weak Entities ข้อ 4
https://drive.google.com/file/d/15jGXrl-8mggsJeiWuDprxNBSPc6Jf7Pg/view?usp=sharing
URL Chen ERD ข้อ 4

05: A car dealer that wants to store cars, customers, and orders.

1. cars(cars_id, brand, name, type, color, miles, price, created_at, segment)
2. customers(customers_id, firstname, lastname, address, phone, work, email)
3. orders(order_id, customer_name, cars_name, price, order_status, created_at)
4. address(address_id, customer_id, address_detail, sub_district, district, province, postal_code) [Weak-Entities] (depends on customers)
5. booking(booking_id, type, scheduled_at, deposit_amount)

Explanation:
address is a weak entity because it depends on **customers** for its existence. It cannot be uniquely identified without referencing **customers_id**, which acts as its identifying relationship. An address only makes sense in the context of a specific customer, so it cannot exist independently.

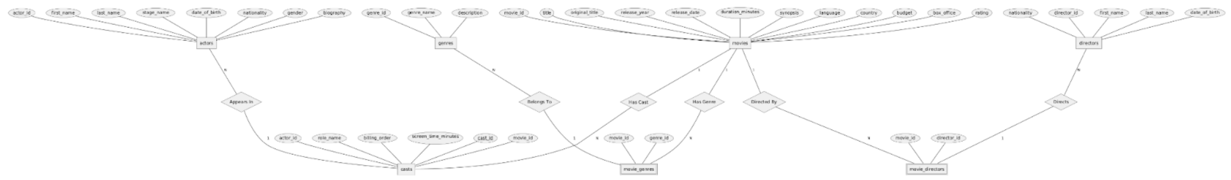


รูปภาพแสดง Chen ERD with Weak Entities ข้อ 5
https://drive.google.com/file/d/104lXJoSWcuIO76LxPaRPmxJ8ZGwRtbX5/view?usp=sharing
URL Chen ERD ข้อ 5

06: A film archive that wants to store movies, actors, and casts.

1. movies(movie_id, title, original_title, release_year, release_date, duration_minutes, synopsis, language, country, budget, box_office, rating)
2. actors(actor_id, first_name, last_name, stage_name, date_of_birth, nationality, gender, biography)
3. directors(director_id, first_name, last_name, date_of_birth, nationality
4. genres(genre_id, genre_name, description)
5. movie_genres(movie_id, genre_id) [Weak-Entities]
6. casts(cast_id, movie_id, actor_id, role_name, billing_order, screen_time_minutes)
7. movie_directors(movie_id, director_id) [Weak-Entities]

Explanation:
movie_genres, **movie_directors** are weak entities, because they don't have their own primary key that depends on movies.



https://drive.google.com/file/d/1s301JUjJqJLanOB73wKj3OQTx-OHEDBG/view?usp=sharing

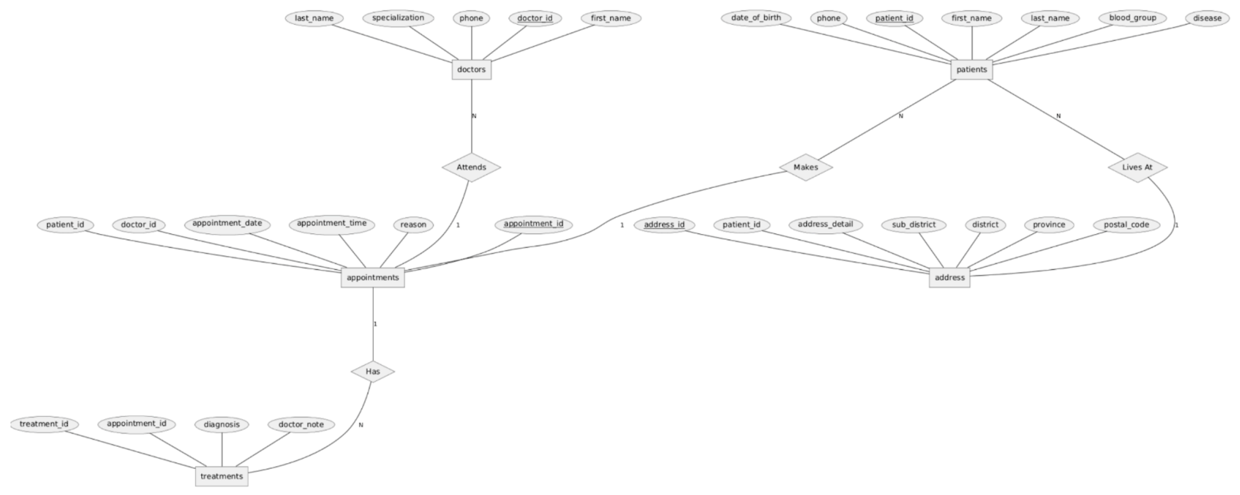
รูปภาพแสดง Chen ERD with Weak Entities ข้อ 6

07: A hospital that wants to store patients, doctors, and appointments.

1. patients (patient_id, first_name, last_name, blood_group, disease, date_of_birth, phone)
2. doctors (doctor_id, first_name, last_name, specialization, phone)
3. appointments (appointment_id, patient_id, doctor_id, appointment_date, appointment_time, reason)
4. address (address_id, patient_id, address_detail, sub_district, district, province, postal_code)
5. treatments (treatment_id, appointment_id, diagnosis, doctor_note)

   [Weak-Entities]

Explanation:
Treatments is weak entities, because a weak entity often relies on a Partial Key (also called a Discriminator)



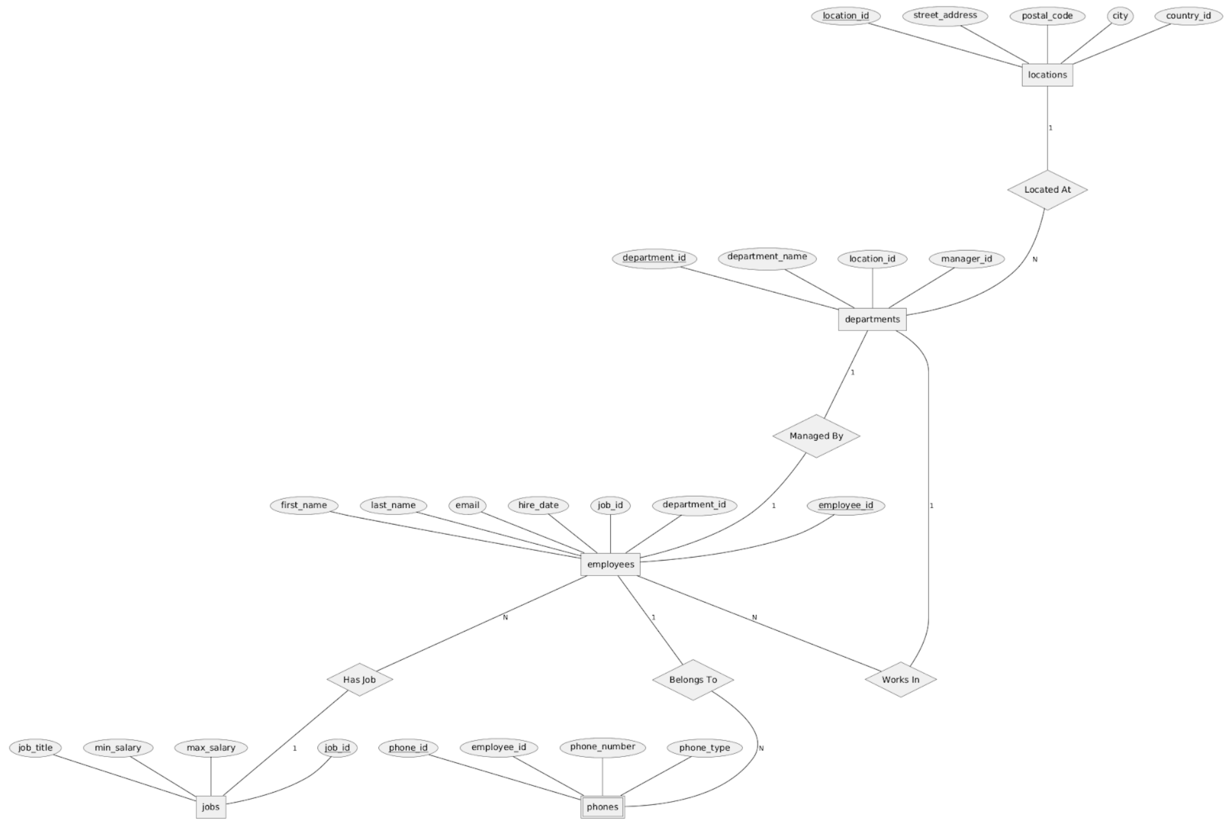
https://drive.google.com/file/d/1jHLA2eESP9K3AsbHUz0YIOy8JTW5d7AG/view?usp=share_link

รูปภาพแสดง Chen ERD with Weak Entities ข้อ 7

08: A company that wants to store departments, employees, employee phone numbers

6. locations (location_id, street_address, postal_code, city, country_id)
7. departments (department_id, department_name, location_id, manager_id)
8. jobs (job_id, job_title, min_salary, max_salary)
9. employees (employee_id, first_name, last_name, email, hire_date, job_id, department_id)
10. phones (phone_id, employee_id, phone_number, phone_type) [Weak-Entities]

Explanation:
phones is weak entities, because a weak entity often relies on a **Partial Key** (also called a **Discriminator**)



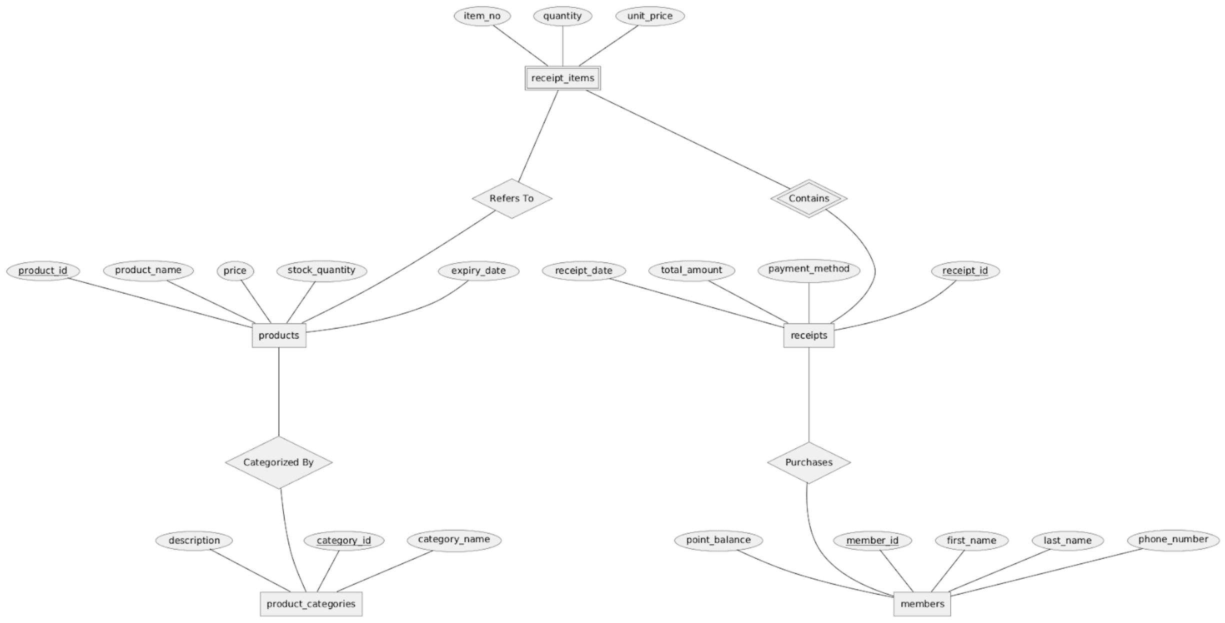
https://drive.google.com/file/d/1_Zi7HDA3rTwhmBjFeM0ZkJSWmgBe1vS2/view?usp=sharing

รูปภาพแสดง Chen ERD with Weak Entities ข้อ 8

09: A supermarket that wants to store products, receipts, and reviews.

1. product_categories (category_id, category_name, description)
2. products (product_id, product_name, category_id, price, stock_quantity, expiry_date)
3. members (member_id, first_name, last_name, phone_number, point_balance)
4. receipts (receipt_id, receipt_date, total_amount, member_id, payment_method)
5. receipt_items (receipt_item_id, receipt_id, product_id, quantity, unit_price)[weak entity]

receipt_items (รายการสินค้าในใบเสร็จ) ไม่สามารถเกิดขึ้นได้หากไม่มี receipts (ใบเสร็จรับเงิน) อ้างอิงอยู่ ดังนั้น receipt_items จึงเป็น Weak Entity ที่ต้องพึ่งพา receipts ในการระบุตัวตน



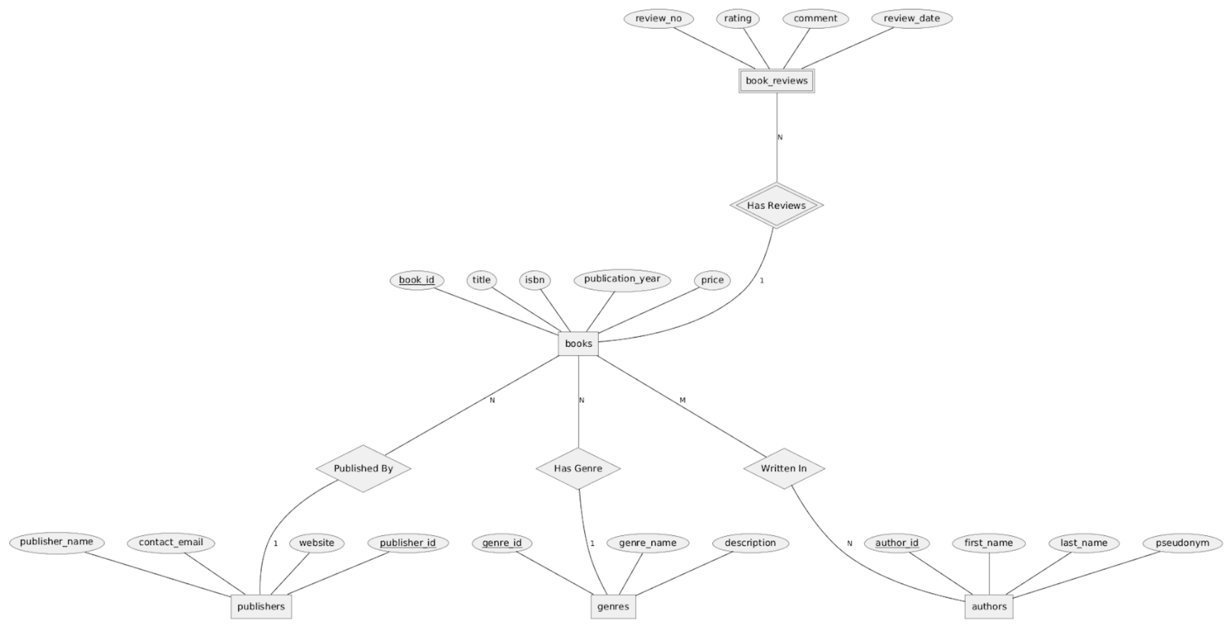
https://drive.google.com/file/d/1Bjn07Sxl9wu3mSEG_HMI8eXbKFJz4ZVh/view?usp=sharing

รูปภาพแสดง Chen ERD with Weak Entities ข้อ 9

10: A bookstore wants to keep books and authors of each book.

1. publishers (publisher_id, publisher_name, contact_email, website)
2. genres (genre_id, genre_name, description)
3. books (book_id, title, isbn, publication_year, price, publisher_id, genre_id)
4. authors (author_id, first_name, last_name, pseudonym)
5. book_authors (id,book_id, author_id)
6. book_reviews (book_id, review_no, rating, comment, review_date) [weak entity]

book_reviews (รีวิวหนังสือ) เข้ามา ซึ่งรีวิวจะเกิดขึ้นไม่ได้ถ้าไม่มีหนังสือเล่มนั้นอยู่ (books) หรืออาจมองเป็น book_copies (เล่มหนังสือคงคลัง) ที่ต้องอิงกับ ISBN ขอ หนังสือหลัก



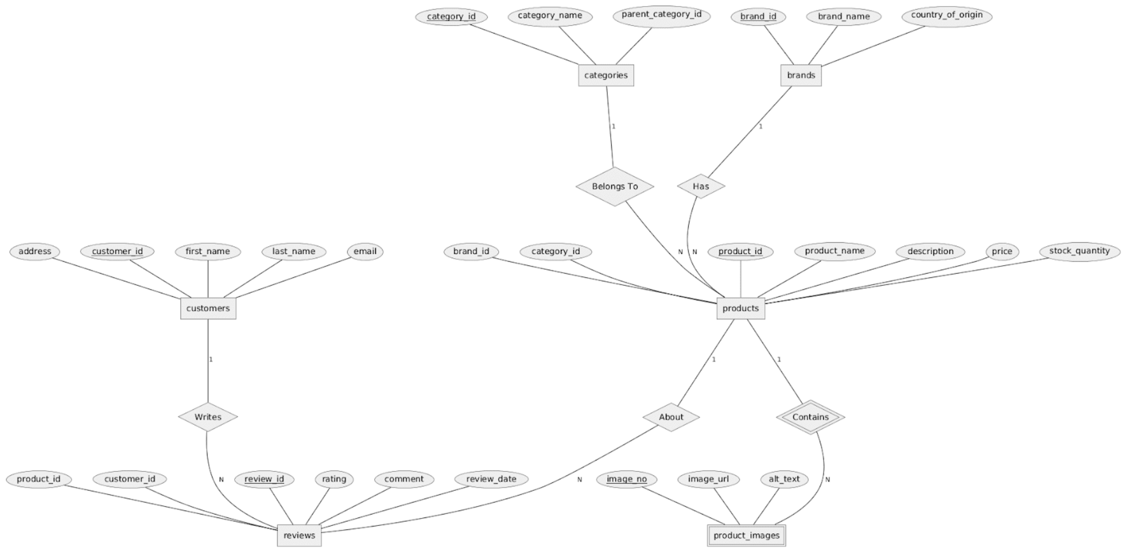
https://drive.google.com/file/d/1zXhUc87qC-1SPAJ7skRCcXUHMkPw-NRn/view?usp=sharing

รูปภาพแสดง Chen ERD with Weak Entities ข้อ 10

11: An online store that wants to store customers, products, and reviews.

1. brands (brand_id, brand_name, country_of_origin)
2. categories (category_id, category_name, parent_category_id)
3. products (product_id, product_name, description, price, stock_quantity, brand_id, category_id)
4. customers (customer_id, first_name, last_name, email, password_hash, address)
5. reviews (review_id, product_id, customer_id, rating, comment, review_date)
6. `product_image(image_no, image_url, alt_text)` [weak entity]

`product_image` เป็นเอนทิตีอ่อนแอ เพราะว่าตัวมันเองไม่มีรหัสที่ไม่ซ้ำกัน (เช่น ในระบบอาจจะมีรูปภาพที่ 1, 2, 3 ของสินค้า A และรูปภาพที่ 1, 2, 3 ของสินค้า B ซึ่งเลขซ้ำกันได้) ดังนั้น มันจึงต้องพึ่งพา products ซึ่งเป็นเอนทิตีแม่ (Owner Entity) เพื่อระบุว่ารูปนี้เป็นของสินค้าชิ้นไหน และถ้าเราลบสินค้าชิ้นนั้นออกจากระบบ รูปภาพทั้งหมดของสินค้านั้นก็ต้องถูกลบตามไปด้วย



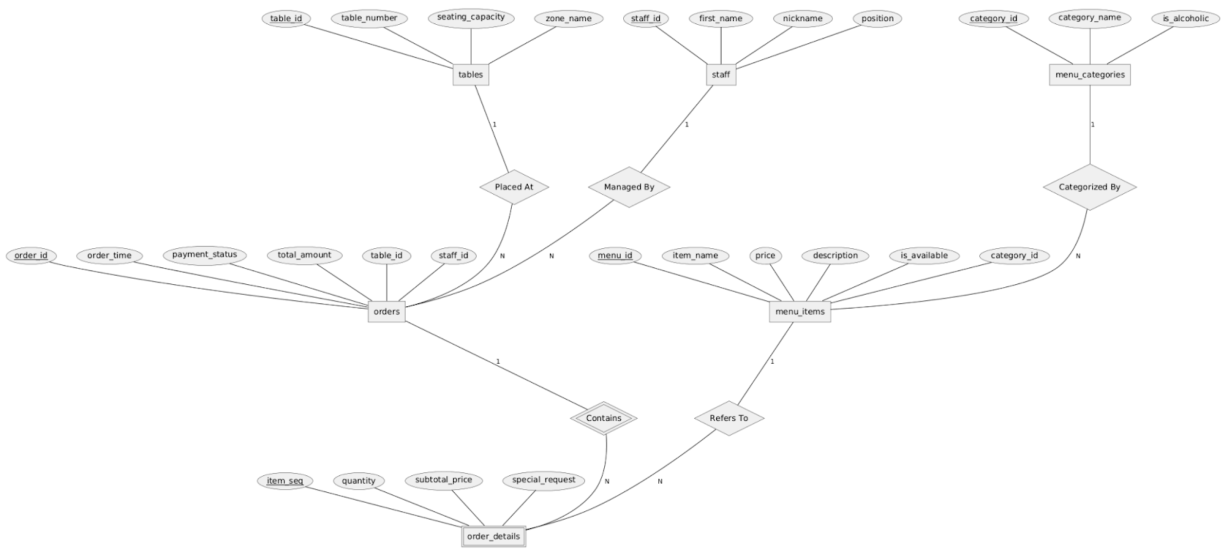
https://drive.google.com/file/d/1qpWjP4XObOoj0udpyVWo9NCOXrzUPAjs/view?usp=sharing

รูปภาพแสดง Chen ERD with Weak Entities ข้อ 11

12: A restaurant wants to keep tables, menus, and order of each table.

1. menu_categories (category_id, category_name, is_alcoholic)
2. menu_items (menu_id, item_name, price, description, category_id, is_available)
3. tables (table_id, table_number, seating_capacity, zone_name)
4. staff (staff_id, first_name, nickname, position)
5. orders (order_id, table_id, staff_id, order_time, payment_status, total_amount)
6. order_details (item_seq,quantity,subtotal_price,special_request) [weak entity]

`Order_details` เป็นเอนทิตีอ่อนแอ เพราะว่าตัวมันเองไม่มีรหัสที่ไม่ซ้ำกัน (เช่น ในออเดอร์หนึ่งอาจมีรายการที่ 1, 2, 3 และในออเดอร์ของอีกโต๊ะก็มีรายการที่ 1, 2, 3 เหมือนกัน ซึ่งเลขซ้ำกันได้) ดังนั้น มันจึงต้องพึ่งพา `orders` ซึ่งเป็นเอนทิตีแม่ (Owner Entity) เพื่อระบุว่ารายการนี้เป็นของบิลใบไหน และถ้าเรายกเลิกหรือลบออเดอร์นั้นออกจากระบบ รายการอาหารทั้งหมดในออเดอร์นั้นก็ต้องถูกลบตามไปด้วย



https://drive.google.com/file/d/1qPAx0jmdW0rr2Z282PK2t6y3uyDQ0nJy/view?usp=sharing

รูปภาพแสดง Chen ERD with Weak Entities ข้อ 12

Task 2: Final Project ERD

Step 1 : Define requirements and business rules of streaming service DB.

Requirements
1. User & Authentication - Users can sign up and log in using email and password - Support third party authentication providers - Users can log out from active sessions - Password reset and verification via token - User sessions are tracked with expiration - Basic user profile management   - Name   - Username   - Profile image 2. Content Catalog - Display a catalog of contents including: - Movies   - Series   - Episodes   - Music - Each content includes metadata: - Title   - Description   - Thumbnail   - Duration   - Release date   - Content type   - Content can be published or unpublished   - Content availability can be checked before playback - Contents can be categorized by: - Genres   - Categories

3. Video Playback
- Users can stream video content
- Playback controls:
- Play
- Pause
- Resume
- Seek forward and backward
- Auto resume from last watched position
- Playback requires a valid streaming token
- Basic error handling for expired or invalid tokens

4. Search & Discovery
- Search content by title
- Browse content by:
    - Genre
    - Category
- Sort content by:
    - Recently added
    - Recently published
- Display popular content based on view count

5. Recommendations (Basic)
- Show “Popular” content based on total views
- Show “Recently Added” content
    - Simple rule based recommendations using:
    - Viewing history
- Popular content

6. Watch Progress
- Track how much of a content a user has watched
- Mark content as completed
- Resume playback from last position
- Show “Continue Watching” list'
- Track watch history across devices

7. Device & Platform Support
- Web based application
- Responsive UI for:
    - Desktop
    - Mobile

- Session based authentication across devices

8. Admin / Content Management
- Admins can upload media files
- Admins can:
  - Add new content
  - Edit content metadata
  - Publish or unpublish content
- Create series playlists
- Add episodes to a series
- Maintain episode order
- Track admin actions for auditing

9. Security & Performance
- Secure video access using token based authorization
- Streaming tokens have expiration
- Support CDN based storage and delivery
- Prevent unauthorized content access

10. Analytics (Minimal)
- Track total views per content
- Track user viewing sessions
- Track active users

11. Playlist

- Group episodes under a playlist (series)
- Display episodes in ordered list
- Each episode includes basic metadata
- Users can:
  - Play episodes directly from playlist
  - Auto play next episode
  - Resume playlist from last watched episode
- Track playlist watch progress
- Purchase playlists or individual content
- Purchased content appears in user library

Business rules
1. User & Account Rules 1. A user must register using a unique email address. 2. A user must verify their email address before being allowed to log in. 3. Users must authenticate using a valid email and password combination. 4. User passwords must be stored in encrypted or hashed form. 5. A user may have only one active session per device at any given time. 6. Only authenticated users may access protected content. 7. Logged out users must not be able to access any protected content. 8. Users must be able to reset their password using an email based verification process. 9. A user may log out at any time, which immediately invalidates the active session token. 10. Users may edit their own basic profile information. 2. Content Catalog Rules 1. Only content marked as published is visible to users. 2. Each content item must contain a title, description, thumbnail image, genre, and duration. 3. Every content item must belong to at least one genre or category. 4. Unpublished content must not appear in search results, browsing views, or recommendations. 5. Content availability must be validated before playback is initiated. 3. Video Playback Rules 1. Only authenticated users are permitted to stream video content. 2. Video playback may begin only if the content is published and available. 3. Video playback must support play, pause, and seek operations. 4. The system must periodically save the user's last watched position during playback. 5. When a user resumes a video, playback must continue from the last saved position. 6. If video playback fails, a clear and user friendly error message must be displayed. 4. Search & Discovery Rules 1. Search results must include only published content. 2. Search functionality must match content titles. 3. Browsing by genre or category must display only content belonging to the

selected filter.

4. Sorting options are limited to popularity and recently added content.
5. When no matching content is found, a “No results found” message must be displayed.

5. Recommendations Rules (Basic)
    1. “Popular” content must be determined based on total view count.
    2. “Trending” content must be determined based on recent viewing activity.
    3. “Recently added” content must be sorted by publish date.
    4. Rule based recommendations may consider genre similarity or overall popularity.

6. Watch Progress Rules
    1. The system must track watch progress for each user and each content item.
    2. Content is considered “In Progress” if less than 90 percent has been watched.
    3. Content is considered “Completed” if 90 percent or more has been watched.
    4. The “Continue Watching” list must display only content marked as “In Progress.”
    5. Completed content must not appear in the “Continue Watching” list.

7. Device & Platform Rules
    1. The application must be accessible through modern web browsers.
    2. The user interface must adapt to both desktop and mobile screen sizes.
    3. Core functionality must behave consistently across supported devices.
    4. Watch progress must synchronize across devices for the same user account.

8. Admin & Content Management Rules
    1. Only users with an administrator role may upload or manage content.
    2. Administrators may create, edit, publish, or unpublish content.
    3. Administrators may create playlists and assign episodes to playlists.
    4. Each episode must belong to exactly one playlist.
    5. Unpublished content or playlists must not be visible to users.
    6. Updates to content metadata must be reflected immediately for users.

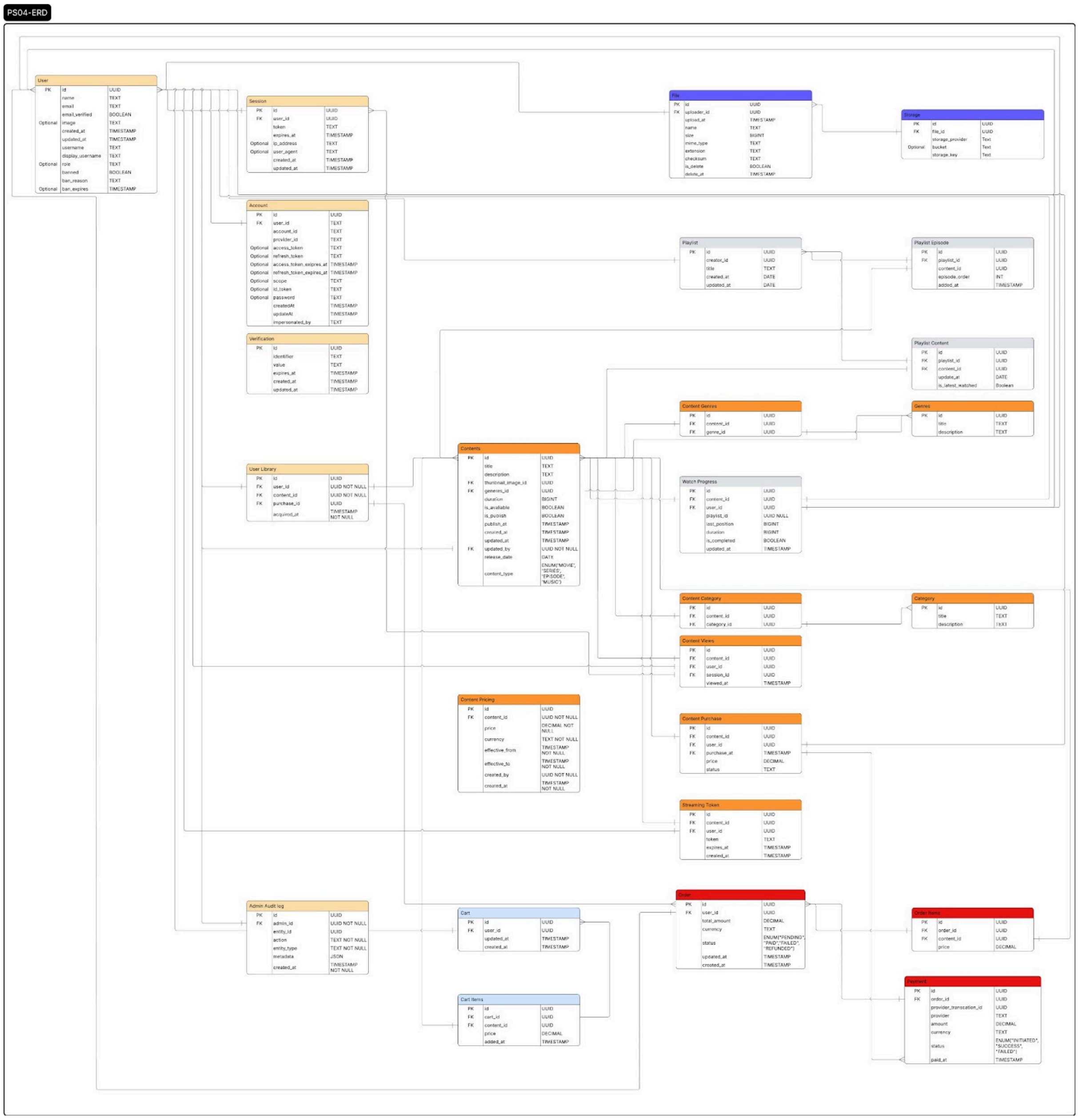
9. Security & Performance Rules
    1. Video streaming must require a valid access token.
    2. Direct access to raw video file URLs must be restricted.
    3. Access tokens must expire after a defined time period.
    4. Video streaming must use a CDN to ensure performance and scalability.
    5. The system must prevent unauthorized sharing or reuse of streaming links.

10. Analytics Rules (Minimal)
    1. Each video playback must increment the content’s view count.
    2. A view must be counted only once per user per session.
    3. Active users must be calculated based on login activity.
    4. Analytics data must be accessible only to administrator users.
    5. Analytics reports must not expose personal user information.

11. Playlist Rules
    1. A playlist groups multiple episodes in a defined sequence.
    2. Episodes within a playlist must be displayed in ascending order.
    3. Each episode must include a title, description, and thumbnail image.
    4. Users may start playback from any episode in a playlist.
    5. When an episode finishes, the next episode must automatically begin playback.
    6. Playlist playback must resume from the last watched episode and timestamp.
    7. The “Continue Watching” list for playlists must resume at the correct episode and position.
    8. Playlists must be purchased before users can access their content.

Step 2 : Design Chen ER & Crow's Foot ER Diagram

Crow's Foot ER Diagram



User

PK	id	UUID
	name	TEXT
	email	TEXT
	email_verified	BOOLEAN
Optional	image	TEXT
	created_at	TIMESTAMP
	updated_at	TIMESTAMP
	username	TEXT
	display_username	TEXT
Optional	role	TEXT
	banned	BOOLEAN
	ban_reason	TEXT
Optional	ban_expires	TIMESTAMP

Session

PK	id	UUID
FK	user_id	UUID
	token	TEXT
	expires_at	TIMESTAMP
Optional	ip_address	TEXT
Optional	user_agent	TEXT
	created_at	TIMESTAMP
	updated_at	TIMESTAMP

Account

PK	id	UUID
FK	user_id	TEXT
	account_id	TEXT
	provider_id	TEXT
Optional	access_token	TEXT
Optional	refresh_token	TEXT
Optional	access_token_expires_at	TIMESTAMP
Optional	refresh_token_expires_at	TIMESTAMP
Optional	scope	TEXT
Optional	id_token	TEXT
Optional	password	TEXT
	createdAt	TIMESTAMP
	updateAt	TIMESTAMP
	impersonated_by	TEXT

Verification

PK	id	UUID
	identifier	TEXT
	value	TEXT
	expires_at	TIMESTAMP
	created_at	TIMESTAMP
	updated_at	TIMESTAMP

File

PK	id	UUID
FK	uploader_id	UUID
	upload_at	TIMESTAMP
	name	TEXT
	size	BIGINT
	mime_type	TEXT
	extension	TEXT
	checksum	TEXT
	is_delete	BOOLEAN
	delete_at	TIMESTAMP

Storage

PK	id	UUID
FK	file_id	UUID
	storage_provider	Text
Optional	bucket	Text
	storage_key	Text

Playlist

PK	id	UUID
	creator_id	UUID
	title	TEXT
	created_at	DATE
	updated_at	DATE

Playlist Episode

PK	id	UUID
FK	playlist_id	UUID
	content_id	UUID
	episode_order	INT
	added_at	TIMESTAMP

Playlist Content

PK	id	UUID
FK	playlist_id	UUID
FK	content_id	UUID
	update_at	DATE
	is_latest_watched	Boolean

Content Genres

PK	id	UUID
FK	content_id	UUID
FK	genre_id	UUID

Genres

PK	id	UUID
	title	TEXT
	description	TEXT

User Library

PK	id	UUID
FK	user_id	UUID NOT NULL
FK	content_id	UUID NOT NULL
FK	purchase_id	UUID
	acquired_at	TIMESTAMP NOT NULL

Contents

PK	id	UUID
	title	TEXT
	description	TEXT
FK	thumbnail_image_id	UUID
FK	generes_id	UUID
	duration	BIGINT
	is_avaliable	BOOLEAN
	is_publish	BOOLEAN
	publish_at	TIMESTAMP
	created_at	TIMESTAMP
	updated_at	TIMESTAMP
FK	updated_by	UUID NOT NULL
	release_date	DATE
	content_type	ENUM('MOVIE', 'SERIES', 'EPISODE', 'MUSIC')

Watch Progress

PK	id	UUID
FK	content_id	UUID
FK	user_id	UUID
	playlist_id	UUID NULL
	last_position	BIGINT
	duration	BIGINT
	is_completed	BOOLEAN
	updated_at	TIMESTAMP

Content Category

PK	id	UUID
FK	content_id	UUID
FK	category_id	UUID

Category

PK	id	UUID
	title	TEXT
	description	TEXT

Content Views

PK	id	UUID
FK	content_id	UUID
FK	user_id	UUID
FK	session_id	UUID
	viewed_at	TIMESTAMP

Content Pricing

PK	id	UUID
FK	content_id	UUID NOT NULL
	price	DECIMAL NOT NULL
	currency	TEXT NOT NULL
	effective_from	TIMESTAMP NOT NULL
	effective_to	TIMESTAMP NOT NULL
	created_by	UUID NOT NULL
	created_at	TIMESTAMP NOT NULL

Content Purchase

PK	id	UUID
FK	content_id	UUID
FK	user_id	UUID
FK	purchase_at	TIMESTAMP
	price	DECIMAL
	status	TEXT

Streaming Token

PK	id	UUID
FK	content_id	UUID
FK	user_id	UUID
	token	TEXT
	expires_at	TIMESTAMP
	created_at	TIMESTAMP

Admin Audit log

PK	id	UUID
FK	admin_id	UUID NOT NULL
	entity_id	UUID
	action	TEXT NOT NULL
	entity_type	TEXT NOT NULL
	metadata	JSON
	created_at	TIMESTAMP NOT NULL

Cart

PK	id	UUID
FK	user_id	UUID
	updated_at	TIMESTAMP
	created_at	TIMESTAMP

Cart Items

PK	id	UUID
FK	cart_id	UUID
FK	content_id	UUID
	price	DECIMAL
	added_at	TIMESTAMP

Order

PK	id	UUID
FK	user_id	UUID
	total_amount	DECIMAL
	currency	TEXT
	status	ENUM("PENDING","PAID","FAILED","REFUNDED")
	updated_at	TIMESTAMP
	created_at	TIMESTAMP

Order Items

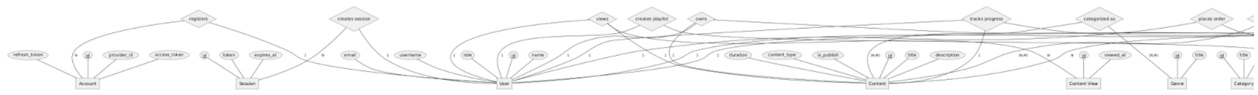
PK	id	UUID
FK	order_id	UUID
FK	content_id	UUID
	price	DECIMAL

Payment

PK	id	UUID
FK	order_id	UUID
	provider_transcation_id	UUID
	provider	TEXT
	amount	DECIMAL
	currency	TEXT
	status	ENUM("INITIATED","SUCCESS","FAILED")
	paid_at	TIMESTAMP

https://drive.google.com/file/d/1s95ex4alMNal9LvFrVZU5OI82a4BSPKO/view?usp=sharing

Chen ER Diagram



https://drive.google.com/file/d/1QnGZIWTkYyx-I9pOuohNMcg4js7IXEka/view?usp=sharing

Step 3 : Explanation

Entity and Relationship Justification

The entities in the ER diagram were identified by analyzing the core business processes and data requirements of a digital streaming platform. The User entity represents individuals who access the system and acts as the central entity connecting authentication, content access, transactions, and viewing activity. To support secure access and external authentication, the Account entity is separated from User, allowing multiple authentication methods to be linked to a single user. The Session entity captures login instances and enforces the one-to-many relationship where a user may have multiple active or historical sessions, but each session belongs to exactly one user.

Content delivery is modeled using the Content entity, which represents all media items available on the platform, such as movies, episodes, or special features. To organize and classify content effectively, Genre and Category entities are introduced with many-to-many relationships to Content, allowing each content item to belong to multiple genres or categories. The Playlist entity is used to represent ordered collections of content, such as episodic series or curated lists, enforcing a one-to-many relationship where a playlist contains multiple content items while each episode belongs to a specific playlist.

User interaction with media is captured through behavioral entities. The Watch Progress entity records playback position, completion status, and timestamps, enabling resume playback and viewing history features. This entity forms a many-to-one relationship with both User and Content, since users may watch many content items and each content item may be watched by many users. Similarly, the Content View entity records viewing events for analytical and reporting purposes, supporting engagement metrics and recommendation logic. Secure media access is enforced using the Streaming Token entity, which links users, sessions, and content through time-limited access credentials.

Monetization and ownership are represented through Order, Payment, and User Library entities. An Order may contain multiple content purchases, while each Payment is associated with exactly one Order, enforcing transactional integrity. The User Library entity establishes a persistent relationship between users and the content they have purchased, independent of viewing activity. Administrative control is modeled using Admin Audit Log, which records system-level actions for accountability and traceability, and File, which represents uploaded media assets associated with content records.

Potential Improvements and Missing Elements

The ER model can be improved by adding a Subscription entity to clearly handle recurring plans, billing periods, and access tiers. Introducing a Device entity linked to sessions would help enforce device limits and improve security tracking. A Rating or Review entity could enhance user engagement and support better recommendations. Stronger constraints, such as limiting active tokens per user and enforcing payment validation, would improve data integrity. Finally, separating Admin into its own entity rather than using role attributes would make permission management clearer and easier to scale.