Exercise 03.03: Extend the KillrVideo Logical Model

In this exercise, you will:

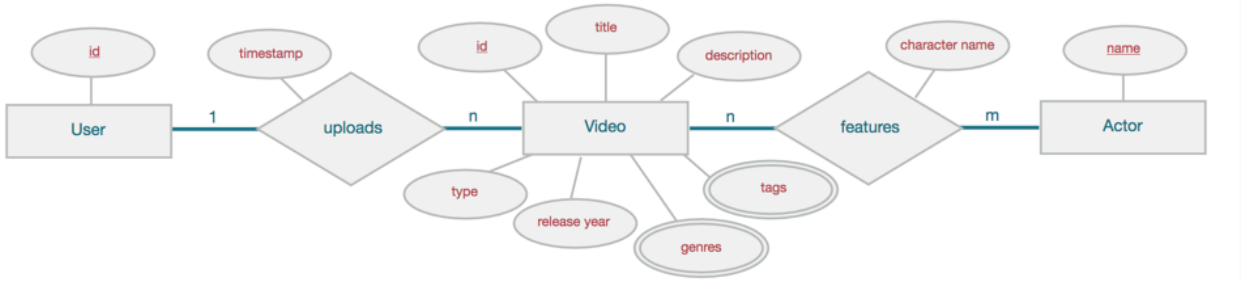
- Add tables to the logical model to support additional queries
- Ensure that the mapping rules are applied appropriately

Background

The KillrVideo product team wants to add additional query capabilities to the application. Specifically, the following queries will need to be supported:

- **Q1.** Find all user videos that match a specific tag (show the most recent uploaded videos first)
- **Q2.** Find all movies that features a specific actor and release year range (show the most recent videos first, and then sorted by title)
- **Q3.** Find all movies that features a specific actor, genre and release year range (show the most recent videos first, and then sorted by title)

The conceptual model outlines the available attributes for the appropriate entities and relationships.



Steps

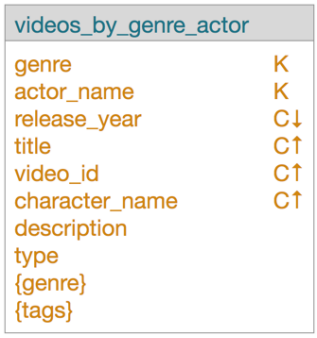
1. On paper or in a text editor, create a logical Chebotko diagram that can support Q1.
2. On paper or in a text editor, create a logical Chebotko diagram that can support Q2.
3. On paper or in a text editor, create a logical Chebotko diagram that can support Q3.
4. For each query, answer the following questions:
    a. What entity or relationship type is being stored in a partition or row?
    b. What are the key attribute(s) for this table?
    c. What attribute is used for the partition key(s) that enabled the equality query?
    d. What attribute is used for the cluster column(s) that enables the inequality / range scan?
    e. What are the clustering column(s) and ordering supporting the required results?

Q2

- What entity or relationship type is being stored in a partition or row?
    - Each partition represents a `Video features Actor` relationship.
    - Each row represents information for a video the actor was featured in.
- What are the key attribute(s) for this table?
    - The key attributes are `actor_name` and `video_id`.
- What attribute is used for the partition key(s) that enabled the query?
    - The equality attribute is `actor_name`.
- What attribute is used for the cluster column(s) that enables the inequality / range scan?
    - The inequality attribute is `release_year`.

- What are the clustering column(s) and ordering that support the required results?
    - The clustering columns that support ordering is `release_year` (DESC) and `title` (ASC).

Q3



- What entity or relationship type is being stored in a partition or row?
    - Each partition represents a `Video features Actor` relationship.
    - Each row represents video information for a specific genre and actor.
- What are the key attribute(s) for this table?
    - The key attributes are `actor_name` and `video_id`.
- What attribute is used for the partition key(s) that enabled the query?
    - The equality attributes are `genre` and `actor_name`.
- What attribute is used for the cluster column(s) that enables the inequality / range scan?
    - The inequality attribute is `release_year`.
- What are the clustering column(s) and ordering that support the required results?
    - The clustering columns that support ordering is *`release_year` (DESC)* and *`title` (ASC)*.