Netflix Streaming - SQL & NoSQL DB Schema with Comments

# 1. SQL Tables (Structured Relational Data)

## Table: users

CREATE TABLE users (  
 user\_id UUID PRIMARY KEY, -- Unique ID for each user  
 email VARCHAR(255) UNIQUE NOT NULL, -- User's login email  
 name VARCHAR(255), -- Full name of the user  
 created\_at TIMESTAMP, -- Account creation time  
 last\_login TIMESTAMP -- Last time the user logged in  
);

## Table: videos

CREATE TABLE videos (  
 video\_id UUID PRIMARY KEY, -- Unique ID per uploaded video  
 title VARCHAR(255), -- Name/title of the video  
 description TEXT, -- Summary or description of the video  
 uploaded\_by UUID REFERENCES users(user\_id), -- Uploader's user\_id  
 upload\_time TIMESTAMP, -- Timestamp when the video was uploaded  
 duration INT, -- Length of video in seconds  
 category VARCHAR(100), -- Genre/category (e.g., Comedy, Drama)  
 status VARCHAR(50) -- Current state: transcoding, ready, error  
);

## Table: user\_watch\_history

CREATE TABLE user\_watch\_history (  
 user\_id UUID REFERENCES users(user\_id), -- Who watched  
 video\_id UUID REFERENCES videos(video\_id), -- What video  
 last\_watched TIMESTAMP, -- When was it last played  
 watched\_duration INT, -- Total seconds watched  
 PRIMARY KEY (user\_id, video\_id) -- Unique record per user+video  
);

# 2. NoSQL Tables (Flexible / High Throughput)

These would ideally be implemented using MongoDB, DynamoDB, or BigQuery/ElasticSearch depending on access patterns.

## Collection: video\_chunks

{  
 "chunk\_id": "UUID",  
 "video\_id": "UUID", // Associated video  
 "resolution": "1080p", // 144p, 720p, 1080p, etc.  
 "chunk\_index": 3, // Order of this chunk in playback  
 "url": "https://cdn.net/video/chunk3.ts", // CDN link to chunk  
 "bitrate": 2500, // Bits per second (2500 kbps for 1080p)  
 "duration\_sec": 6 // Chunk duration in seconds (usually ~6)  
}

bitrate explanation: Determines the video quality and bandwidth usage for the chunk. Higher bitrate = better video quality but needs faster internet.

## Collection: manifests

{  
 "manifest\_id": "UUID",  
 "video\_id": "UUID",  
 "resolution": "720p",  
 "manifest\_url": "https://cdn.net/video/manifest720p.m3u8"  
}

Manifest links help video players load the correct resolution and ordering of chunks.

## Collection: stream\_events

{  
 "event\_id": "UUID",  
 "user\_id": "UUID",  
 "video\_id": "UUID",  
 "event\_type": "buffer", // play, pause, quality\_change, etc.  
 "timestamp": "2024-04-24T10:11:12Z",  
 "resolution": "1080p",  
 "chunk\_index": 4  
}

These logs are useful for:  
- Diagnosing streaming issues  
- Playback analytics  
- Quality monitoring (QoE)

# Summary Table

|  |  |  |
| --- | --- | --- |
| Table Name | Type | Purpose |
| users | SQL | Manage user accounts and auth |
| videos | SQL | Store metadata of uploaded videos |
| user\_watch\_history | SQL | Track what user watched and progress |
| video\_chunks | NoSQL | Store metadata for each video chunk |
| manifests | NoSQL | Mapping of resolutions to chunks |
| stream\_events | NoSQL | High-volume real-time playback events |