Video Processing Pipeline

El proyecto consiste en el desarrollo de una base de datos para un Sistema de Procesamiento de Video que automatiza la transcodificación y conversión de archivos multimedia. Este sistema permite recibir videos en diversos formatos, procesarlos según diferentes perfiles de calidad y generar múltiples versiones optimizadas para distintos dispositivos y plataformas de streaming.

Objetivo

El objetivo principal es diseñar e implementar una base de datos que permita:

Gestión de archivos fuente: Almacenar metadatos de videos originales
Control de trabajos de codificación: Administrar la cola de procesamiento
Seguimiento de calidad: Registrar métricas de rendimiento y calidad
Monitoreo de recursos: Controlar la utilización de servidores de procesamiento
Trazabilidad completa: Desde el archivo original hasta las versiones finales

SITUACIÓN PROBLEMÁTICA

Necesidad del Negocio

Las empresas de streaming y producción multimedia enfrentan el desafío de procesar grandes volúmenes de contenido de video para múltiples plataformas. Sin un sistema de base de datos estructurado, surgen problemas como:

Pérdida de trazabilidad en el procesamiento de archivos
Duplicación de trabajo por falta de control de estado
Ineficiencia en la asignación de recursos computacionales
Dificultad para medir calidad y rendimiento del procesamiento
Falta de visibilidad en los tiempos de procesamiento

Solución Propuesta

La implementación de esta base de datos permite:

Centralizar toda la información del pipeline de procesamiento
Automatizar la gestión de colas de trabajo
Proporcionar métricas detalladas de calidad y rendimiento
Optimizar la utilización de recursos computacionales

MODELO DE NEGOCIO

Descripción de la Organización

El sistema está diseñado para organizaciones que manejan procesamiento de video a gran escala:

Tipos de Usuarios

OTTs: Suben videos originales a la plataforma
System Administrators: Configuran perfiles de codificación y monitorean workers
Quality Analysts: Revisan métricas de calidad y rendimiento

Flujo de Trabajo

Ingesta: Los archivos de video se cargan y analizan automáticamente
Encolado: Se crean trabajos de codificación según perfiles activos
Procesamiento: Workers disponibles procesan los videos
Análisis: Se calculan métricas de calidad para cada salida
Distribución: Los archivos procesados están listos para distribución

Relaciones Principales
Un archivo fuente puede generar múltiples trabajos de codificación
Cada trabajo se asigna a un worker y usa un perfil específico
Cada trabajo genera una salida transcodificada con métricas de calidad
Los workers tienen registros de rendimiento para diferentes códecs

LISTADO DE TABLAS

SOURCE_FILES Almacena información de archivos de video originales

source_file_id - BIGINT - Identificador único - CLAVE PRIMARIA
filename - VARCHAR(255) - Nombre del archivo
file_path - VARCHAR(500) - Ruta del archivo - UNIQUE
file_size_bytes - BIGINT - Tamaño en bytes
duration_seconds - DECIMAL(10,3) - Duración del video
upload_timestamp - TIMESTAMP - Fecha de carga
checksum_md5 - CHAR(32) - Hash MD5 - UNIQUE
original_codec - VARCHAR(50) - Códec original
original_resolution - VARCHAR(20) - Resolución original
original_bitrate - BIGINT - Bitrate original
original_framerate - DECIMAL(5,2) - FPS original
audio_channels - TINYINT - Canales de audio
audio_sample_rate - INT - Frecuencia de audio
container_format - VARCHAR(20) - Formato contenedor
metadata_json - JSON - Metadatos adicionales
created_at - TIMESTAMP - Fecha creación
updated_at - TIMESTAMP - Fecha actualización

ENCODING_PROFILES Define perfiles de codificación disponibles

profile_id - INT - Identificador único - CLAVE PRIMARIA
profile_name - VARCHAR(100) - Nombre del perfil - UNIQUE
target_codec - VARCHAR(50) - Códec destino
target_resolution - VARCHAR(20) - Resolución objetivo
target_bitrate - BIGINT - Bitrate objetivo
target_framerate - DECIMAL(5,2) - FPS objetivo
audio_codec - VARCHAR(50) - Códec de audio
audio_bitrate - INT - Bitrate de audio
container_format - VARCHAR(20) - Formato final
ffmpeg_preset - VARCHAR(20) - Preset FFmpeg
quality_setting - DECIMAL(4,2) - Configuración calidad
profile_description - TEXT - Descripción
created_date - TIMESTAMP - Fecha creación
is_active - BOOLEAN - Perfil activo

created_at - TIMESTAMP - Fecha creación
updated_at - TIMESTAMP - Fecha actualización

PROCESSING_WORKERS Servidores disponibles para procesamiento

worker_id - INT - Identificador único - CLAVE PRIMARIA
worker_name - VARCHAR(100) - Nombre del worker - UNIQUE
server_hostname - VARCHAR(255) - Nombre servidor
cpu_cores - TINYINT - Núcleos CPU
ram_gb - SMALLINT - Memoria RAM en GB
gpu_model - VARCHAR(100) - Modelo GPU
gpu_memory_gb - SMALLINT - Memoria GPU en GB
max_concurrent_jobs - TINYINT - Máximo trabajos concurrentes
current_load - TINYINT - Carga actual
worker_status - ENUM - Estado worker
supported_codecs - JSON - Códecs soportados
last_heartbeat - TIMESTAMP - Último heartbeat
worker_type - ENUM - Tipo worker
created_at - TIMESTAMP - Fecha creación
updated_at - TIMESTAMP - Fecha actualización

ENCODING_JOBS Tabla central que gestiona trabajos de codificación

job_id - BIGINT - Identificador único - CLAVE PRIMARIA
source_file_id - BIGINT - Referencia archivo - CLAVE FORÁNEA → source_files
profile_id - INT - Referencia perfil - CLAVE FORÁNEA → encoding_profiles
worker_id - INT - Worker asignado - CLAVE FORÁNEA → processing_workers
job_priority - TINYINT - Prioridad trabajo
job_status - ENUM - Estado trabajo
queued_timestamp - TIMESTAMP - Momento encolado
started_timestamp - TIMESTAMP - Momento inicio
completed_timestamp - TIMESTAMP - Momento finalización
estimated_duration - INT - Duración estimada
actual_duration - INT - Duración real
retry_count - TINYINT - Número reintentos
max_retries - TINYINT - Máximo reintentos
created_by_user - VARCHAR(100) - Usuario creador
error_message - TEXT - Mensaje error
progress_percentage - DECIMAL(5,2) - Porcentaje progreso
created_at - TIMESTAMP - Fecha creación
updated_at - TIMESTAMP - Fecha actualización

TRANSCODED_OUTPUTS Archivos generados tras codificación

output_id - BIGINT - Identificador único - CLAVE PRIMARIA
job_id - BIGINT - Referencia trabajo - CLAVE FORÁNEA → encoding_jobs
output_filename - VARCHAR(255) - Nombre archivo generado
output_path - VARCHAR(500) - Ruta archivo - UNIQUE
output_size_bytes - BIGINT - Tamaño en bytes
actual_bitrate - BIGINT - Bitrate real
actual_resolution - VARCHAR(20) - Resolución real
actual_framerate - DECIMAL(5,2) - FPS real
encoding_time_seconds - INT - Tiempo codificación

compression_ratio - DECIMAL(8,4) - Ratio compresión
output_checksum - CHAR(32) - Hash MD5 - UNIQUE
created_at - TIMESTAMP - Fecha creación
QUALITY_METRICS Métricas de calidad calculadas
metric_id - BIGINT - Identificador único - CLAVE PRIMARIA
output_id - BIGINT - Referencia salida - CLAVE FORÁNEA → transcoded_outputs
psnr_value - DECIMAL(6,3) - Valor PSNR
ssim_value - DECIMAL(6,4) - Valor SSIM
vmaf_score - DECIMAL(6,3) - Puntuación VMAF
bitrate_efficiency - DECIMAL(8,4) - Eficiencia bitrate
visual_quality_score - DECIMAL(5,2) - Calidad visual
audio_quality_score - DECIMAL(5,2) - Calidad audio
analysis_timestamp - TIMESTAMP - Momento análisis
metric_version - VARCHAR(20) - Versión algoritmo
analysis_duration_seconds - INT - Tiempo análisis

FFMPEG_COMMANDS Comandos FFmpeg utilizados

command_id - BIGINT - Identificador único - CLAVE PRIMARIA
job_id - BIGINT - Referencia trabajo - CLAVE FORÁNEA → encoding_jobs, UNIQUE
full_command_string - TEXT - Comando completo
input_filters - JSON - Filtros entrada
video_filters - JSON - Filtros video
audio_filters - JSON - Filtros audio
output_parameters - JSON - Parámetros salida
hardware_acceleration - VARCHAR(50) - Aceleración hardware
thread_count - TINYINT - Número hilos
memory_usage_mb - INT - Uso memoria
command_version - VARCHAR(20) - Versión comando
created_at - TIMESTAMP - Fecha creación

PROCESSING_ERRORS Errores durante procesamiento

error_id - BIGINT - Identificador único - CLAVE PRIMARIA
job_id - BIGINT - Referencia trabajo - CLAVE FORÁNEA → encoding_jobs
error_code - VARCHAR(50) - Código error
error_message - TEXT - Mensaje error
error_timestamp - TIMESTAMP - Momento error
ffmpeg_output - TEXT - Salida FFmpeg
system_resources_at_error - JSON - Estado recursos
recovery_action - VARCHAR(100) - Acción recuperación
is_resolved - BOOLEAN - Error resuelto
resolved_timestamp - TIMESTAMP - Momento resolución
resolved_by - VARCHAR(100) - Usuario resolvió

CODEC_PERFORMANCE Métricas rendimiento de códecs

performance_id - BIGINT - Identificador único - CLAVE PRIMARIA
codec_name - VARCHAR(50) - Nombre códec
worker_id - INT - Referencia worker - CLAVE FORÁNEA → processing_workers
input_resolution - VARCHAR(20) - Resolución entrada

encoding_speed_fps - DECIMAL(8,3) - Velocidad codificación
cpu_utilization - DECIMAL(5,2) - Utilización CPU
memory_usage_mb - INT - Uso memoria
gpu_utilization - DECIMAL(5,2) - Utilización GPU
benchmark_date - TIMESTAMP - Fecha benchmark
test_duration_seconds - INT - Duración prueba
quality_setting - DECIMAL(4,2) - Configuración calidad
sample_file_type - VARCHAR(50) - Tipo archivo prueba

Objetos de Base de Datos
VISTAS
1. vw_dashboard_trabajos_activos
Objetivo: Proporcionar una vista consolidada del estado actual de todos los trabajos en procesamiento para monitoreo en tiempo real.
Tablas involucradas:

encoding_jobs
source_files
encoding_profiles
processing_workers

Descripción: Combina información de trabajos, archivos fuente, perfiles y workers. Incluye cálculo de tiempo de procesamiento y filtra por trabajos activos ('queued', 'processing'). Ordenado por prioridad descendente y fecha de encolado.
Campos principales:

Estado del trabajo y progreso
Información del archivo origen
Perfil de codificación aplicado
Worker asignado y su estado
Tiempos estimados y reales

2. vw_metricas_calidad_completas
Objetivo: Consolidar todas las métricas de calidad con información del trabajo y archivo original para análisis comparativo.
Tablas involucradas:

quality_metrics
transcoded_outputs
encoding_jobs
source_files
encoding_profiles

Descripción: Permite análisis comparativo de calidad entre diferentes perfiles y códecs. Incluye categorización automática de calidad basada en VMAF (Excelente, Muy Buena, Buena, Aceptable, Baja). Solo muestra trabajos completados.
Campos principales:

Métricas PSNR, SSIM, VMAF
Comparación archivo original vs transcodificado

Eficiencia de bitrate y ratio de compresión
Categoría de calidad calculada

3. vw_rendimiento_workers
Objetivo: Analizar el rendimiento y utilización de cada worker en el sistema.
Tablas involucradas:

processing_workers
encoding_jobs
transcoded_outputs

Descripción: Agrega estadísticas de trabajos completados, tiempos promedio y eficiencia por worker. Calcula tasa de éxito, horas totales de procesamiento y estado de conexión (minutos desde último heartbeat).
Campos principales:

Especificaciones del worker (CPU, RAM, GPU)
Total de trabajos procesados
Tasa de éxito/fallo
Tiempos promedio de codificación
Estado de disponibilidad

4. vw_historial_errores_recientes
Objetivo: Monitorear errores recientes para identificar patrones y problemas sistemáticos.
Tablas involucradas:

processing_errors
encoding_jobs
source_files
processing_workers

Descripción: Muestra errores de los últimos 30 días con información contextual del trabajo. Incluye cálculo de tiempo sin resolver y acciones de recuperación.
Campos principales:

Código y mensaje de error
Archivo y worker afectado
Estado de resolución
Tiempo transcurrido sin resolver

FUNCIONES
1. fn_calcular_eficiencia_compresion(job_id)
Retorna: DECIMAL(5,2) - Porcentaje de reducción de tamaño
Objetivo: Calcular el porcentaje de reducción de tamaño tras la compresión comparando el archivo original con el transcodificado.
Parámetros:

p_job_id (BIGINT): ID del trabajo de codificación

Lógica:

Obtiene tamaño del archivo original desde source_files
Obtiene tamaño del archivo transcodificado desde transcoded_outputs
Calcula: ((tamaño_original - tamaño_comprimido) / tamaño_original) * 100
Retorna 0.00 si hay valores NULL o inválidos

Ejemplo de uso:
```
SELECT job_id, fn_calcular_eficiencia_compresion(job_id) AS
eficiencia_porcentaje
FROM encoding_jobs
WHERE job_status = 'completed';
```

2. fn_obtener_tiempo_espera_estimado(prioridad)
Retorna: INT - Minutos estimados de espera
Objetivo: Estimar tiempo de espera para trabajos en cola según su prioridad.
Parámetros:

p_prioridad (TINYINT): Prioridad del trabajo (1-10)

Lógica:

Cuenta trabajos en cola con prioridad >= a la especificada
Obtiene capacidad total de workers activos
Calcula tiempo promedio de procesamiento (últimos 7 días)
Estima: (trabajos_adelante * tiempo_promedio) / (capacidad_total * 60)

Ejemplo de uso:
```
SELECT fn_obtener_tiempo_espera_estimado(8) AS
minutos_espera_prioridad_8;
```

3. fn_verificar_disponibilidad_worker(worker_id)
Retorna: TINYINT - 1 (disponible) o 0 (no disponible)
Objetivo: Verificar si un worker específico está disponible para aceptar nuevos trabajos.
Parámetros:

p_worker_id (INT): ID del worker a verificar

Lógica:

Obtiene estado, carga actual y capacidad máxima del worker
Verifica si estado = 'active' y carga_actual < max_jobs
Retorna 1 si disponible, 0 en caso contrario

Ejemplo de uso:
```
SELECT worker_id, worker_name,
fn_verificar_disponibilidad_worker(worker_id) AS disponible
FROM processing_workers;
```

STORED PROCEDURES
1. sp_asignar_trabajo_a_worker(job_id)

Objetivo: Asignar automáticamente un trabajo en cola al worker más apropiado.
Parámetros:

p_job_id (BIGINT): ID del trabajo a asignar

Tablas afectadas:

encoding_jobs (UPDATE)
processing_workers (UPDATE - incrementa current_load)

Descripción:
Selecciona el worker óptimo basándose en:

Estado activo
Capacidad disponible
Soporte del códec requerido
Mayor capacidad disponible (max_jobs - current_load)
Mayor número de CPU cores

Actualiza el trabajo a estado 'processing' y asigna el worker, incrementando su carga.
Ejemplo de uso:
CALL sp_asignar_trabajo_a_worker(13);
Salida:

resultado: Mensaje de éxito o error
worker_id: ID del worker asignado (o NULL si no hay disponibles)

2. sp_reporte_rendimiento_periodo(fecha_inicio, fecha_fin)
Objetivo: Generar reporte completo de rendimiento del sistema en un período específico.
Parámetros:

p_fecha_inicio (DATETIME): Fecha inicio del período
p_fecha_fin (DATETIME): Fecha fin del período

Tablas consultadas:

encoding_jobs
transcoded_outputs
quality_metrics
encoding_profiles
processing_workers

Descripción:
Retorna 4 conjuntos de resultados:

RESUMEN_TRABAJOS: Total, completados, fallidos, cancelados, duración promedio, tasa de éxito
METRICAS_CALIDAD: Promedios de VMAF, PSNR, SSIM, calidad visual, eficiencia
RENDIMIENTO_POR_PERFIL: Estadísticas por perfil de codificación

TOP_WORKERS: Top 5 workers más productivos

Ejemplo de uso:
CALL sp_reporte_rendimiento_periodo('2025-01-15 00:00:00', '2025-01-30 23:59:59');

3. sp_reintentar_trabajos_fallidos(horas_limite)
Objetivo: Reintentar automáticamente trabajos fallidos que no hayan superado el límite de reintentos.
Parámetros:

p_horas_limite (INT): Límite de horas desde el fallo para considerar el reintento

Tablas afectadas:

encoding_jobs (UPDATE)

Descripción:
Resetea el estado de trabajos fallidos elegibles:

Cambia status a 'queued'
Limpia worker_id y started_timestamp
Incrementa retry_count
Resetea progress_percentage a 0
Solo procesa trabajos donde retry_count < max_retries

Ejemplo de uso:
CALL sp_reintentar_trabajos_fallidos(48);
Salida:

trabajos_reintentados: Número de trabajos reintentados
mensaje: Mensaje descriptivo

TRIGGERS
1. trg_validar_worker_antes_asignacion
Tipo: BEFORE UPDATE
Tabla: encoding_jobs
Objetivo: Validar que el worker asignado esté activo y tenga capacidad disponible antes de asignar un trabajo.
Descripción:
Se ejecuta antes de actualizar un encoding_job. Valida:

Worker tiene estado 'active'
Worker tiene capacidad disponible (current_load < max_concurrent_jobs)

Genera error (SQLSTATE '45000') si alguna validación falla, cancelando la operación.
Casos de uso:

Prevenir asignación a workers en mantenimiento
Prevenir sobrecarga de workers
Garantizar integridad en la asignación automática

2. trg_actualizar_carga_worker_completado
Tipo: AFTER UPDATE
Tabla: encoding_jobs
Objetivo: Decrementar automáticamente la carga del worker cuando un trabajo se completa.
Descripción:
Se ejecuta después de actualizar un encoding_job. Cuando un trabajo pasa de 'processing' a estado final ('completed', 'failed', 'cancelled'):

Decrementa current_load del worker en 1
Usa GREATEST para evitar valores negativos
Actualiza updated_at del worker

Beneficio: Mantiene automáticamente la carga de workers actualizada sin intervención manual.

3. trg_registrar_error_trabajo_fallido
Tipo: AFTER UPDATE
Tabla: encoding_jobs
Objetivo: Registrar automáticamente un error cuando un trabajo cambia a estado fallido.
Descripción:
Se ejecuta después de actualizar un encoding_job. Cuando un trabajo cambia a estado 'failed':

Crea registro en processing_errors
Usa el error_message del trabajo o mensaje por defecto
Determina recovery_action según retry_count vs max_retries
Marca error como no resuelto (is_resolved = FALSE)

Beneficio: Trazabilidad completa de errores sin necesidad de código adicional.

4. trg_validar_metricas_calidad
Tipo: BEFORE INSERT
Tabla: quality_metrics
Objetivo: Validar que las métricas de calidad estén dentro de rangos válidos antes de insertarlas.
Descripción:
Se ejecuta antes de insertar en quality_metrics. Valida:

PSNR: 0-100 (típicamente 20-60 dB)
SSIM: 0-1
VMAF: 0-100

Establece metric_version = '1.0' por defecto si es NULL.
Genera error (SQLSTATE '45000') si valores están fuera de rango.
Beneficio: Garantiza integridad de datos de calidad y previene valores imposibles.