Universidad ORT Uruguay

Facultad de Ingeniería

Base de datos 2

Obligatorio 2

Sebastián Méndez 143403

Matías Regaló 308940

Facundo Martínez 342426

Docentes: Leonardo Barcelo - Danilo Fraque

2025

Índice

1) Correcciones

Dado el feedback del obligatorio 1, se hicieron modificaciones.

A continuación, detallaremos los comentarios de la devolución con su correspondiente solución:

- ¿En dónde se almacenan las construcciones y recursos disponibles del juego?

El modelo actual almacena recursos específicos de cada país en cada partida, pero no tiene una tabla catálogo de "tipos de recursos disponibles en el juego”.

Se soluciono creando una tabla RECURSO_PARTIDA y modificando RECURSO para que haga referencia al tipo.

-En la tabla JUGADOR, cantidad_medallas_obtenidas está en la entidad JUGADOR, pero las medallas se derivan de los logros y la tabla JUGADOR_MEDALLA. Mantener este campo puede generar inconsistencias (violación de la 3FN). Se recomienda eliminarlo y calcularlo mediante consulta.

Se elimina cantidad_medallas_obtenidas y la tabla JUGADOR_MEDALLA, se crea una vista.

-En la tabla COMERCIO, id_recurso solo referencia el recurso del país origen. No se contempla el caso de trueques de varios recursos o recursos de ambos países. El modelo no permite registrar múltiples recursos por transacción, lo que limita la representación del escenario de trueque.

Se detectaron limitaciones a la hora de hacer un trueque donde participen los países.

Agregamos a la tabla COMERCIO el país de origen y destino. Creamos la tabla COMERCIO_RECURSO que se detalla los recursos que ofrece el país origen y destino.

-En la tabla CONSTRUCCION_TIPO, el campo max_unidades_por_ronda se aplica a todos los tipos de construcción, pero no todos los tipos requieren este atributo (según el enunciado). Debería permitirse NULL o separar los atributos específicos en tablas especializadas.

La solución más simple era permitir NULL que fue la que se implementó. Se discutió otra posible solución de hacer especialización para tener tablas separadas.

-En la tabla RONDA, el campo número no es UNIQUE por partida, lo que puede permitir rondas duplicadas con el mismo número en una partida. Se recomienda agregar UNIQUE (codigo_partida, numero).

Se implementó la solución sugerida.

Se deja adjunta DDL con los cambios implementados.

Restricciones no estructurales.

En la entrega pasada se identificaron 13 restricciones no estructurales.

RESTRICCION 1: Si la producción supera el límite, suspender siguiente ronda

RESTRICCION 2: Suspender produccion si el almacenamiento pasa el maximo

RESTRICCION 3: Producción automatica de construcciones (usina/represa/plantación)

RESTRICCION 4: Cada país produce 10.000 unidades de su PBN cada 10 rondas

RESTRICCIONES 5, 6, 7: Consumo obligatorio de PBN y generación de deuda

RESTRICCION 8: Antes de construir (costos iniciales), el pais debe tener recursos

RESTRICCION 9: Un pais no puede tener dos construcciones del mismo tipo

RESTRICCION 10: Cantidad a comerciar NO puede superar capacidad del transporte

RESTRICCION 11 y 12: Consumo basico (alimentos y energia) y eliminacion por incumplimientos

RESTRICCION 13: Un país solo puede alcanzar un logro si cumple condiciones específicas

Para poder implementar una solución para cada restricción se hizo uso de Funciones auxiliares,Triggers,Procedimientos,Secuencias.

Se adjunta archivo de la implementación de las restricciones no estructurales.

Para la verificación se usó un conjunto de datos de prueba().

Se siguieron los siguientes pasos de ejecución para realizar estas verificaciones:

1:DDL.

2:Datos_prueba.

3:RestriccionesNoEstructurales.

2) Requerimientos

Los scripts con los requerimientos de la parte 2 fueron adjuntados en la entrega.

3) Gestión de chats

Colección 1: Chats

Guardamos un documento por cada partida y almacena la configuración y las estadísticas generales de la sala de chat de esa partida.

_id: (String) Clave primaria. Es el mismo codigo_partida (ejemplo. "PARTIDA_2025_001").

fecha_creacion: (Date) Cuándo se creó la sala de chat de esta partida.

configuracion: (Objeto) Contiene las reglas del chat para esta partida.

- configuracion.permitir_mensajes_privados: (Boolean) Si los jugadores pueden enviarse mensajes privados.
- configuracion.moderacion_activa: (Boolean) Si el chat tiene moderación activa.
- configuracion.max_caracteres_mensaje: (Number) Límite de caracteres por mensaje.
- configuracion.emojis_permitidos: (Boolean) Si se pueden usar emojis.

participantes: (Array de Objetos) Lista de jugadores que están en el chat de esta partida.

- participantes[].alias: (String) El alias del jugador.
- participantes[].fecha_union: (Date) Cuándo se unió el jugador al chat.
- participantes[].activo: (Boolean) Si el jugador está actualmente en la partida/chat.
- participantes[].denuncias_recibidas: (Number) Contador de denuncias que ha recibido este jugador.
- participantes[].mensajes_enviados: (Number) Contador de mensajes que ha enviado este jugador.

estadisticas: (Objeto) Contadores y metadatos agregados de la partida.

- estadisticas.total_mensajes: (Number) Total de mensajes enviados en el chat.
- estadisticas.total_mensajes_publicos: (Number) Subtotal de mensajes públicos.
- estadisticas.total_mensajes_privados: (Number) Subtotal de mensajes privados.
- estadisticas.total_likes: (Number) Total de "Me Gusta" dados en la partida.
- estadisticas.total_denuncias: (Number) Total de denuncias recibidas en la partida.
- estadisticas.ultimo_mensaje: (Date) Timestamp del último mensaje enviado.

version_esquema: (String) Para control de versiones del documento (ej. "2.0").

fecha_ultima_actualizacion: (Date) Cuándo se actualizó por última vez este documento .

EJEMPLO


```
{
 "_id": "PARTIDA_2025_001",
 "fecha_creacion": "2025-11-01T10:00:00Z",
 "configuracion": {
  "permitir_mensajes_privados": true,
  "moderacion_activa": true,
  "max_caracteres_mensaje": 500,
  "emojis_permitidos": true
 },
 "participantes": [
        {
   "alias": "jugador1",
   "fecha_union": "2025-11-01T10:05:00Z",
```

```
      "activo": true,
      "denuncias_recibidas": 0,
      "mensajes_enviados": 45
            },
            {
      "alias": "jugador2",
      "fecha_union": "2025-11-01T10:06:00Z",
      "activo": true,
      "denuncias_recibidas": 1,
      "mensajes_enviados": 38
            }
  ],
  "estadisticas": {
    "total_mensajes": 156,
    "total_mensajes_publicos": 134,
    "total_mensajes_privados": 22,
    "total_likes": 89,
    "total_denuncias": 3,
    "ultimo_mensaje": "2025-11-01T12:45:30Z"
  },
  "version_esquema": "2.0",
  "fecha_ultima_actualizacion": "2025-11-01T12:45:30Z"
}
```


Colección 2: Mensajes

Esta colección tiene un documento por cada mensaje individual enviado. Es donde se almacena la gran mayoría de los datos.

_id: (ObjectId) Clave primaria. El identificador único de este mensaje.

chat_id: (String) Clave foránea. El _id de la colección Chats (ej. "PARTIDA_2025_001") para vincular el mensaje a su partida.

timestamp: (Date) Fecha y hora exactas en que se envió el mensaje. Se usa para ordenar.

tipo_mensaje: (String) Define la visibilidad: "publico" o "privado".

remitente_alias: (String) Alias del jugador que envió el mensaje.

destinatario_alias: (String) Si tipo_mensaje es "privado", el alias del receptor. Si no, es null.

contenido: (Objeto) El sub-documento flexible que contiene el mensaje en sí.

- contenido.tipo: (String) Qué tipo de contenido es: "texto", "accion_juego", "propuesta_comercio", "votacion", "estrategia_compartida".
- contenido.texto: (String) El texto principal visible para el usuario.
- contenido.emojis: (Array de Strings) (Opcional) Lista de emojis usados (ej. ["🔧", "🏭"]).

- contenido.texto_original_hash: (String) (Opcional) Hash del texto original, usado solo en mensajes moderados (estado: "moderado") para auditoría.
- contenido.propuesta: (Objeto) (Opcional) Solo existe si contenido.tipo es "propuesta_comercio".
    - contenido.propuesta.recursos_ofrecidos: (Array de Objetos)
    - contenido.propuesta.recursos_solicitados: (Array de Objetos)
    - contenido.propuesta.medio_transporte: (String)
    - contenido.propuesta.validez_turnos: (Number)

contenido.accion: (Objeto) (Opcional) Solo existe si contenido.tipo es "accion_juego".

- contenido.accion.tipo_accion: (String)
- contenido.accion.id_construccion: (Number)
- contenido.accion.nombre_construccion: (String)
- contenido.accion.recursos_utilizados: (Array de Objetos)
- contenido.accion.beneficios: (Objeto)

contenido.estrategia: (Objeto) (Opcional) Solo existe si contenido.tipo es "estrategia_compartida".

- contenido.estrategia.turnos: (Array de Objetos)
- contenido.estrategia.objetivos_finales: (Array de Strings)

- contenido.votacion: (Objeto) (Opcional) Solo existe si contenido.tipo es "votacion".
    - contenido.votacion.pregunta: (String)
    - contenido.votacion.opciones: (Array de Objetos)
        - contenido.votacion.opciones[].id: (Number)
        - contenido.votacion.opciones[].texto: (String)
        - contenido.votacion.opciones[].votantes: (Array de Strings) (NO guardamos contador)
    - contenido.votacion.estado: (String) (ej. "abierta", "cerrada")
    - contenido.votacion.fecha_cierre: (Date)
    - contenido.votacion.requiere_unanimidad: (Boolean)
    - contenido.votacion.umbral_aprobacion: (Number) (ej. 0.66)

interacciones: (Objeto) Agrupa todas las reacciones de otros jugadores a este mensaje.

- interacciones.likes_usuarios: (Array de Strings) Lista de alias de jugadores que dieron "Me Gusta".
- interacciones.denuncias: (Array de Objetos) Cada objeto es una denuncia individual.
    - interacciones.denuncias[].por_alias: (String) Quién denunció.
    - interacciones.denuncias[].motivo: (String) Razón (ej. "spam").
    - interacciones.denuncias[].descripcion: (String) (Opcional) Texto libre.
    - interacciones.denuncias[].timestamp: (Date) Cuándo denunció.
- interacciones.comentarios: (Array de Objetos) (Opcional) Para respuestas directas al mensaje.
- interacciones.comentarios[].autor_alias: (String)
- interacciones.comentarios[].texto: (String)
- interacciones.comentarios[].timestamp: (Date)

- interacciones.reacciones_personalizadas: (Array de Objetos) (Opcional) Para emojis de reacción.
    - interacciones.reacciones_personalizadas[].alias: (String) Quién reaccionó.

        - interacciones.reacciones_personalizadas[].reaccion: (String) El emoji (ej. "🎉").
    - interacciones.respuesta_comercio: (Objeto) (Opcional) Solo existe en "propuesta_comercio".
        - interacciones.respuesta_comercio.estado: (String) (ej. "pendiente", "aceptada")
        - interacciones.respuesta_comercio.fecha_respuesta: (Date)

estado: (String) El estado de visibilidad del mensaje: "activo", "moderado", "eliminado".
moderacion: (Objeto) (Opcional) Solo existe si estado es "moderado".

- moderacion.fecha_moderacion: (Date)
- moderacion.moderador_alias: (String)
- moderacion.razon: (String)
- moderacion.accion_tomada: (String) (ej. "ocultacion_contenido")
- moderacion.sancion_aplicada: (Objeto)
    - moderacion.sancion_aplicada.tipo: (String) (ej. "advertencia")
    - moderacion.sancion_aplicada.duracion_silencio: (Number) (o null)

Mensajes

// EJEMPLO 1: Mensaje de público


```
{
 "_id": { "$oid": "690a0569b225368faaaeb904" },
 "chat_id": "PARTIDA_2025_001",
 "timestamp": "2025-11-01T11:30:15Z",
 "tipo_mensaje": "publico",
 "remitente_alias": "jugador1",
 "destinatario_alias": null,
 "contenido": {
  "tipo": "texto",
  "texto": "¿Alguien quiere comerciar hierro? 🔧",
  "emojis": ["🔧"]
 },
 "interacciones": {
  "likes_usuarios": [
   "jugador2",
   "jugador3",
   "jugador4"
        ],
  "denuncias": []
 },
 "estado": "activo"
}
```


// EJEMPLO 2: Mensaje privado


```
{
 "_id": { "$oid": "690a0569b225368faaaeb905" },
 "chat_id": "PARTIDA_2025_001",
 "timestamp": "2025-11-01T11:32:00Z",
 "tipo_mensaje": "privado",
 "remitente_alias": "jugador1",
```

```
  "destinatario_alias": "jugador2",
  "contenido": {
    "tipo": "propuesta_comercio",
    "texto": "Te ofrezco 100 de hierro por 50 de oro",
    "propuesta": {
      "recursos_ofrecidos": [
            { "id_recurso": 1, "nombre": "Hierro", "cantidad": 100 }
            ],
      "recursos_solicitados": [
            { "id_recurso": 3, "nombre": "Oro", "cantidad": 50 }
            ],
      "medio_transporte": "barco",
      "validez_turnos": 2
            }
  },
  "interacciones": {
    "likes_usuarios": [],
    "denuncias": [],
    "respuesta_comercio": {
      "estado": "pendiente",
      "fecha_respuesta": null
            }
  },
  "estado": "activo"
}
```

// EJEMPLO 3: Mensaje con acción de juego

```
{
  "_id": { "$oid": "690a0569b225368faaaeb906" },
  "chat_id": "PARTIDA_2025_001",
  "timestamp": "2025-11-01T11:35:20Z",
  "tipo_mensaje": "publico",
  "remitente_alias": "jugador3",
  "destinatario_alias": null,
  "contenido": {
    "tipo": "accion_juego",
    "texto": "He construido una nueva fábrica de acero 🏭",
    "accion": {
      "tipo_accion": "construccion",
      "id_construccion": 42,
      "nombre_construccion": "Fábrica de Acero",
      "recursos_utilizados": [
            { "id_recurso": 1, "nombre": "Hierro", "cantidad": 50 },
            { "id_recurso": 2, "nombre": "Carbón", "cantidad": 30 }
            ],
      "beneficios": {
        "produccion_adicional": { "id_recurso": 5, "nombre": "Acero", "cantidad": 20 }
            }
            },
    "emojis": ["🏭"]
  },
```

```
  "interacciones": {
    "likes_usuarios": [
      "jugador1",
      "jugador2",
      "jugador4",
      "jugador5",
      "jugador6"
          ],
    "denuncias": [],
    "reacciones_personalizadas": [
          { "alias": "jugador2", "reaccion": "🎉" },
          { "alias": "jugador4", "reaccion": "🎉" },
          { "alias": "jugador5", "reaccion": "😮" }
          ]
  },
  "estado": "activo"
}
```

// EJEMPLO 4: Mensaje con contenido multimedia y datos complejos

```
{
  "_id": { "$oid": "690a0569b225368faaaeb908" },
  "chat_id": "PARTIDA_2025_001",
  "timestamp": "2025-11-01T11:40:00Z",
  "tipo_mensaje": "publico",
  "remitente_alias": "jugador4",
  "destinatario_alias": null,
  "contenido": {
    "tipo": "estrategia_compartida",
    "texto": "Mi plan para los próximos 3 turnos 📊",
    "estrategia": {
      "turnos": [
          {
        "numero": 5,
        "acciones": [
          { "tipo": "construccion", "objetivo": "Mina de Oro" },
          { "tipo": "comercio", "socio": "jugador2", "recurso": "Hierro" }
          ]
          },
          {
        "numero": 6,
        "acciones": [
          { "tipo": "expansion", "territorio": "Norte" },
          { "tipo": "defensa", "puntos_asignados": 150 }
          ]
          },
          {
        "numero": 7,
```

```
        "acciones": [
          { "tipo": "logro", "objetivo": "Industrialización Completa" }
          ]
          }
          ],
    "objetivos_finales": ["Dominio Económico", "Alianza con jugador2 y jugador5"]
          },
   "emojis": ["📊", "🎯"]
  },

  "interacciones": {
   "likes_usuarios": [
    "jugador2",
    "jugador5"
          ],
   "denuncias": [],
   "comentarios": [
          {
     "autor_alias": "jugador2",
     "texto": "Buena estrategia, cuenten conmigo 💪",
     "timestamp": "2025-11-01T11:42:00Z"
          }
          ]
  },
  "estado": "activo"
}
// EJEMPLO 5: Mensaje denunciado y moderado
{
  "_id": { "$oid": "690a0569b225368faaaeb909" },
  "chat_id": "PARTIDA_2025_001",
  "timestamp": "2025-11-01T11:50:00Z",
  "tipo_mensaje": "publico",
  "remitente_alias": "jugador6",
  "destinatario_alias": null,
  "contenido": {
   "tipo": "texto",
   "texto": "[CONTENIDO MODERADO]",
   "texto_original_hash": "sha256_hash_del_contenido"
  },
  "interacciones": {
   "likes_usuarios": [],
   "denuncias": [
          {
     "por_alias": "jugador1",
     "motivo": "lenguaje_ofensivo",
     "descripcion": "Insultos hacia otros jugadores",
     "timestamp": "2025-11-01T11:51:00Z"
          },
          {
     "por_alias": "jugador2",
```

```
      "motivo": "lenguaje_ofensivo",
      "timestamp": "2025-11-01T11:51:30Z"
          },
          {
      "por_alias": "jugador3",
      "motivo": "spam",
      "timestamp": "2025-11-01T11:52:00Z"
          }
          ]
  },

  "estado": "moderado",

  "moderacion": {
    "fecha_moderacion": "2025-11-01T11:55:00Z",
    "moderador_alias": "admin1",
    "razon": "Violación de normas de conducta",
    "accion_tomada": "ocultacion_contenido",
    "sancion_aplicada": {
      "tipo": "advertencia",
      "duracion_silencio": null
          }
  }
}
```

// EJEMPLO 6: Mensaje con datos de votación (tipo agregado dinámicamente)

```
{
  "_id": { "$oid": "690a0569b225368faaaeb910" },
  "chat_id": "PARTIDA_2025_001",
  "timestamp": "2025-11-01T12:00:00Z",
  "tipo_mensaje": "publico",
  "remitente_alias": "jugador1",
  "destinatario_alias": null,
  "contenido": {
    "tipo": "votacion",
    "texto": "¿Deberíamos formar una alianza económica? 🗳️",
    "votacion": {
      "pregunta": "¿Alianza económica entre países del norte?",
      "opciones": [
          { "id": 1, "texto": "A favor", "votantes": ["jugador2", "jugador3", "jugador4"] },
          { "id": 2, "texto": "En contra", "votantes": ["jugador5"] },
          { "id": 3, "texto": "Abstención", "votantes": ["jugador6"] }
          ],
      "estado": "abierta",
      "fecha_cierre": "2025-11-01T18:00:00Z",
      "requiere_unanimidad": false,
      "umbral_aprobacion": 0.66
          },
    "emojis": ["🗳️"]
```

```
  },
  "interacciones": {
    "likes_usuarios": [
      "jugador2",
      "jugador3",
      "jugador4",
      "jugador5"
            ],
    "denuncias": []
  },

  "estado": "activo"
}
```

ÍNDICES CREADOS

chats

```
// para ordenar chats por fecha de creación
db.chats.createIndex(
  { fecha_creacion: 1 }
);

// para listar chats por último mensaje
db.chats.createIndex(
  { "estadisticas.ultimo_mensaje": -1 }
);

// buscar participantes activos
db.chats.createIndex(
  { "participantes.alias": 1 },
  { partialFilterExpression: { "participantes.activo": true } }
);

 //Indice único para evitar alias duplicados dentro de un chat
db.chats.createIndex(
  { _id: 1, "participantes.alias": 1 },
  { unique: true }
);
```

mensajes

```
// mensajes por chat ordenados por timestamp
db.mensajes.createIndex(
  { chat_id: 1, timestamp: 1 }
);

//Mensajes privados: chat + tipo + remitente + destinatario
db.mensajes.createIndex(
```

```
  { chat_id: 1, tipo_mensaje: 1, remitente_alias: 1, destinatario_alias: 1 }
);

// todos los mensajes enviados por un usuario
db.mensajes.createIndex(
  { remitente_alias: 1, timestamp: -1 }
);

// todos los mensajes recibidos por un usuario
db.mensajes.createIndex(
  { destinatario_alias: 1, timestamp: -1 }
);



// contar mensajes enviados por usuario dentro de un chat
db.mensajes.createIndex(
  { chat_id: 1, remitente_alias: 1 }
);

// Mensajes privados por destinatario  y fecha
db.mensajes.createIndex(
  { destinatario_alias: 1, timestamp: 1 }
);

// Por modo de moderación (buscar mensajes moderados/eliminados)
db.mensajes.createIndex(
  { estado: 1 }
);

//Consultas por tipo de contenido (texto, propuesta, votación, acción...)
db.mensajes.createIndex(
  { "contenido.tipo": 1 }
);
```

VALIDADORES

```
db.createCollection("chats", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: [
        "_id",
        "fecha_creacion",
        "configuracion",
        "participantes",
        "estadisticas",
        "version_esquema",
        "fecha_ultima_actualizacion"
      ],
      properties: {
        _id: {
          bsonType: "string",
          description: "Identificador único de la partida (String requerido)"
        },
        fecha_creacion: {
          bsonType: "date",
          description: "Fecha de creación de la partida (Date requerido)"
        },
        configuracion: {
          bsonType: "object",
          required: [
            "permitir_mensajes_privados",
            "moderacion_activa",
            "max_caracteres_mensaje",
            "emojis_permitidos"
          ],
          properties: {
            permitir_mensajes_privados: {
              bsonType: "bool",
              description: "Si se permiten mensajes privados (Boolean requerido)"
```

```
    },
    moderacion_activa: {
      bsonType: "bool",
      description: "Si la moderación está activa (Boolean requerido)"
    },
    max_caracteres_mensaje: {
      bsonType: "int",
      minimum: 1,
      description: "Máximo de caracteres por mensaje (Integer requerido, mínimo 1)"
    },
    emojis_permitidos: {
      bsonType: "bool",
      description: "Si se permiten emojis (Boolean requerido)"
    }
  },
  additionalProperties: false
},
participantes: {
  bsonType: "array",
  minItems: 0,
  items: {
    bsonType: "object",
    required: [
      "alias",
      "fecha_union",
      "activo",
      "denuncias_recibidas",
      "mensajes_enviados"
    ],
    properties: {
      alias: {
        bsonType: "string",
        description: "Alias único del participante (String requerido)"
      },
      fecha_union: {
        bsonType: "date",
        description: "Fecha de unión a la partida (Date requerido)"
      },
      activo: {
        bsonType: "bool",
        description: "Si el participante está activo (Boolean requerido)"
      },
      denuncias_recibidas: {
        bsonType: "int",
        minimum: 0,
        description: "Total de denuncias recibidas (Integer requerido, mínimo 0)"
      },
      mensajes_enviados: {
        bsonType: "int",
        minimum: 0,
```

```
        description: "Total de mensajes enviados (Integer requerido, mínimo 0)"
      }
    },
    additionalProperties: false
  },
  description: "Array de participantes (Array requerido)"
},
estadisticas: {
  bsonType: "object",
  required: [
    "total_mensajes",
    "total_mensajes_publicos",
    "total_mensajes_privados",
    "total_likes",
    "total_denuncias",
    "ultimo_mensaje"
  ],
  properties: {
    total_mensajes: {
      bsonType: "int",
      minimum: 0,
      description: "Total de mensajes (Integer requerido, mínimo 0)"
    },
    total_mensajes_publicos: {
      bsonType: "int",
      minimum: 0,
      description: "Total de mensajes públicos (Integer requerido, mínimo 0)"
    },
    total_mensajes_privados: {
      bsonType: "int",
      minimum: 0,
      description: "Total de mensajes privados (Integer requerido, mínimo 0)"
    },
    total_likes: {
      bsonType: "int",
      minimum: 0,
      description: "Total de likes dados (Integer requerido, mínimo 0)"
    },
    total_denuncias: {
      bsonType: "int",
      minimum: 0,
      description: "Total de denuncias recibidas (Integer requerido, mínimo 0)"
    },
    ultimo_mensaje: {
      bsonType: "date",
      description: "Timestamp del último mensaje (Date requerido)"
    }
  },
  additionalProperties: false
},
```

```
      version_esquema: {
        bsonType: "string",
        description: "Versión del esquema (String requerido)"
      },
      fecha_ultima_actualizacion: {
        bsonType: "date",
        description: "Fecha de última actualización (Date requerido)"
      }
    },
    additionalProperties: false
  }
 }
});

db.createCollection("mensajes", {
 validator: {
  $jsonSchema: {
    bsonType: "object",
    required: [
      "_id",
      "chat_id",
      "timestamp",
      "tipo_mensaje",
      "remitente_alias",
      "contenido",
      "interacciones",
      "estado"
    ],
    properties: {
      _id: {
        bsonType: "objectId",
        description: "Identificador único del mensaje (ObjectId requerido)"
      },
      chat_id: {
        bsonType: "string",
        description: "ID de la partida (String requerido)"
      },
      timestamp: {
        bsonType: "date",
        description: "Fecha y hora del mensaje (Date requerido)"
      },
      tipo_mensaje: {
        enum: ["publico", "privado"],
        description: "Tipo de mensaje: publico o privado (String requerido)"
      },
      remitente_alias: {
        bsonType: "string",
        description: "Alias del remitente (String requerido)"
      },
      destinatario_alias: {
```

```
  bsonType: ["string", "null"],
  description: "Alias del destinatario (String o null)"
},
contenido: {
  bsonType: "object",
  required: ["tipo", "texto"],
  properties: {
    tipo: {
      enum: ["texto", "accion_juego", "propuesta_comercio", "votacion", "estrategia_compartida"],
      description: "Tipo de contenido (String requerido)"
    },
    texto: {
      bsonType: "string",
      description: "Texto del mensaje (String requerido)"
    },
    emojis: {
      bsonType: "array",
      items: { bsonType: "string" },
      description: "Array de emojis (Array opcional)"
    },
    texto_original_hash: {
      bsonType: "string",
      description: "Hash del texto original para mensajes moderados (String opcional)"
    },
    propuesta: {
      bsonType: "object",
      required: ["recursos_ofrecidos", "recursos_solicitados", "medio_transporte", "validez_turnos"],
      properties: {
        recursos_ofrecidos: {
          bsonType: "array",
          items: {
            bsonType: "object",
            required: ["id_recurso", "nombre", "cantidad"],
            properties: {
              id_recurso: { bsonType: "int" },
              nombre: { bsonType: "string" },
              cantidad: { bsonType: "int", minimum: 0 }
            }
          }
        },
        recursos_solicitados: {
          bsonType: "array",
          items: {
            bsonType: "object",
            required: ["id_recurso", "nombre", "cantidad"],
            properties: {
              id_recurso: { bsonType: "int" },
              nombre: { bsonType: "string" },
              cantidad: { bsonType: "int", minimum: 0 }
            }
```

```
      }
    },
    medio_transporte: { bsonType: "string" },
    validez_turnos: { bsonType: "int", minimum: 0 }
  },
  description: "Propuesta de comercio (Objeto opcional)"
},
accion: {
  bsonType: "object",
  required: ["tipo_accion"],
  properties: {
    tipo_accion: { bsonType: "string" },
    id_construccion: { bsonType: "int" },
    nombre_construccion: { bsonType: "string" },
    recursos_utilizados: {
      bsonType: "array",
      items: {
        bsonType: "object",
        required: ["id_recurso", "nombre", "cantidad"],
        properties: {
          id_recurso: { bsonType: "int" },
          nombre: { bsonType: "string" },
          cantidad: { bsonType: "int" }
        }
      }
    },
    beneficios: { bsonType: "object" }
  },
  description: "Acción de juego (Objeto opcional)"
},
estrategia: {
  bsonType: "object",
  required: ["turnos", "objetivos_finales"],
  properties: {
    turnos: {
      bsonType: "array",
      items: {
        bsonType: "object",
        required: ["numero", "acciones"],
        properties: {
          numero: { bsonType: "int" },
          acciones: { bsonType: "array" }
        }
      }
    },
    objetivos_finales: {
      bsonType: "array",
      items: { bsonType: "string" }
    }
  },
```

```
      description: "Estrategia compartida (Objeto opcional)"
    },
    votacion: {
      bsonType: "object",
      required: ["pregunta", "opciones", "estado"],
      properties: {
        pregunta: { bsonType: "string" },
        opciones: {
          bsonType: "array",
          items: {
            bsonType: "object",
            required: ["id", "texto", "votantes"],
            properties: {
              id: { bsonType: "int" },
              texto: { bsonType: "string" },
              votantes: {
                bsonType: "array",
                items: { bsonType: "string" }
              }
            }
          }
        },
        estado: { enum: ["abierta", "cerrada"] },
        fecha_cierre: { bsonType: "date" },
        requiere_unanimidad: { bsonType: "bool" },
        umbral_aprobacion: { bsonType: "double", minimum: 0, maximum: 1 }
      },
      description: "Votación (Objeto opcional)"
    }
  },
  additionalProperties: true
},
interacciones: {
  bsonType: "object",
  required: ["likes_usuarios", "denuncias"],
  properties: {
    likes_usuarios: {
      bsonType: "array",
      items: { bsonType: "string" },
      description: "Array de alias que dieron like (Array requerido)"
    },
    denuncias: {
      bsonType: "array",
      items: {
        bsonType: "object",
        required: ["por_alias", "motivo", "timestamp"],
        properties: {
          por_alias: { bsonType: "string" },
          motivo: { bsonType: "string" },
          descripcion: { bsonType: "string" },
```

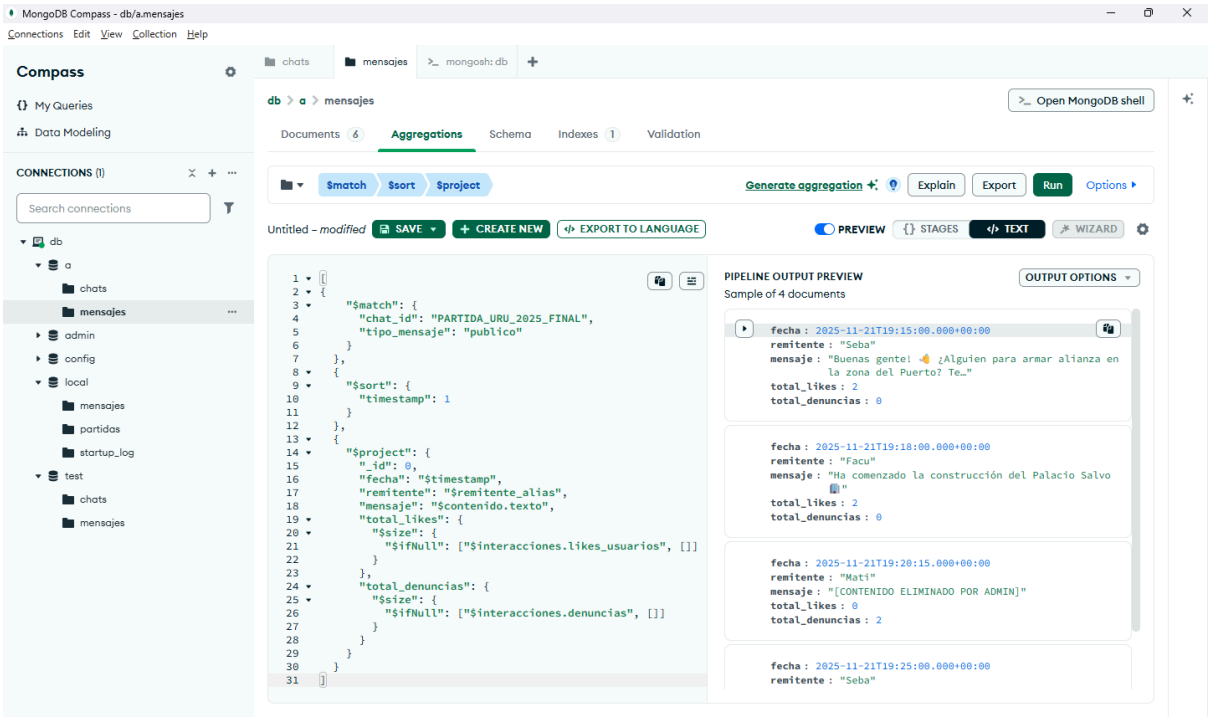
```
            timestamp: { bsonType: "date" }
           }
          },
          description: "Array de denuncias (Array requerido)"
         },
         comentarios: {
          bsonType: "array",
          items: {
           bsonType: "object",
           required: ["autor_alias", "texto", "timestamp"],
           properties: {
            autor_alias: { bsonType: "string" },
            texto: { bsonType: "string" },
            timestamp: { bsonType: "date" }
           }
          },
          description: "Array de comentarios (Array opcional)"
         },
         reacciones_personalizadas: {
          bsonType: "array",
          items: {
           bsonType: "object",
           required: ["alias", "reaccion"],
           properties: {
            alias: { bsonType: "string" },
            reaccion: { bsonType: "string" }
           }
          },
          description: "Array de reacciones personalizadas (Array opcional)"
         },
         respuesta_comercio: {
          bsonType: "object",
          required: ["estado"],
          properties: {
           estado: { enum: ["pendiente", "aceptada", "rechazada"] },
           fecha_respuesta: { bsonType: ["date", "null"] }
          },
          description: "Respuesta a propuesta de comercio (Objeto opcional)"
         }
        },
        additionalProperties: true
       },
       estado: {
        enum: ["activo", "moderado", "eliminado"],
        description: "Estado del mensaje (String requerido)"
       },
       moderacion: {
        bsonType: "object",
        required: ["fecha_moderacion", "moderador_alias", "razon", "accion_tomada",
"sancion_aplicada"],
```

```
        properties: {
          fecha_moderacion: { bsonType: "date" },
          moderador_alias: { bsonType: "string" },
          razon: { bsonType: "string" },
          accion_tomada: { bsonType: "string" },
          sancion_aplicada: {
            bsonType: "object",
            required: ["tipo"],
            properties: {
              tipo: { bsonType: "string" },
              duracion_silencio: { bsonType: ["int", "null"] }
            }
          }
        },
        description: "Información de moderación (Objeto opcional)"
      },
    },
    additionalProperties: false
  }
 }
});
```

4.1)

```
[
{
  "$match": {
    "chat_id": "PARTIDA_URU_2025_FINAL",
    "tipo_mensaje": "publico"
  }
 },
 {
  "$sort": {
    "timestamp": 1
  }
 },
 {
  "$project": {
    "_id": 0,
    "fecha": "$timestamp",
    "remitente": "$remitente_alias",
    "mensaje": "$contenido.texto",
    "total_likes": {
      "$size": {
        "$ifNull": ["$interacciones.likes_usuarios", []]
      }
    },
    "total_denuncias": {
      "$size": {
```

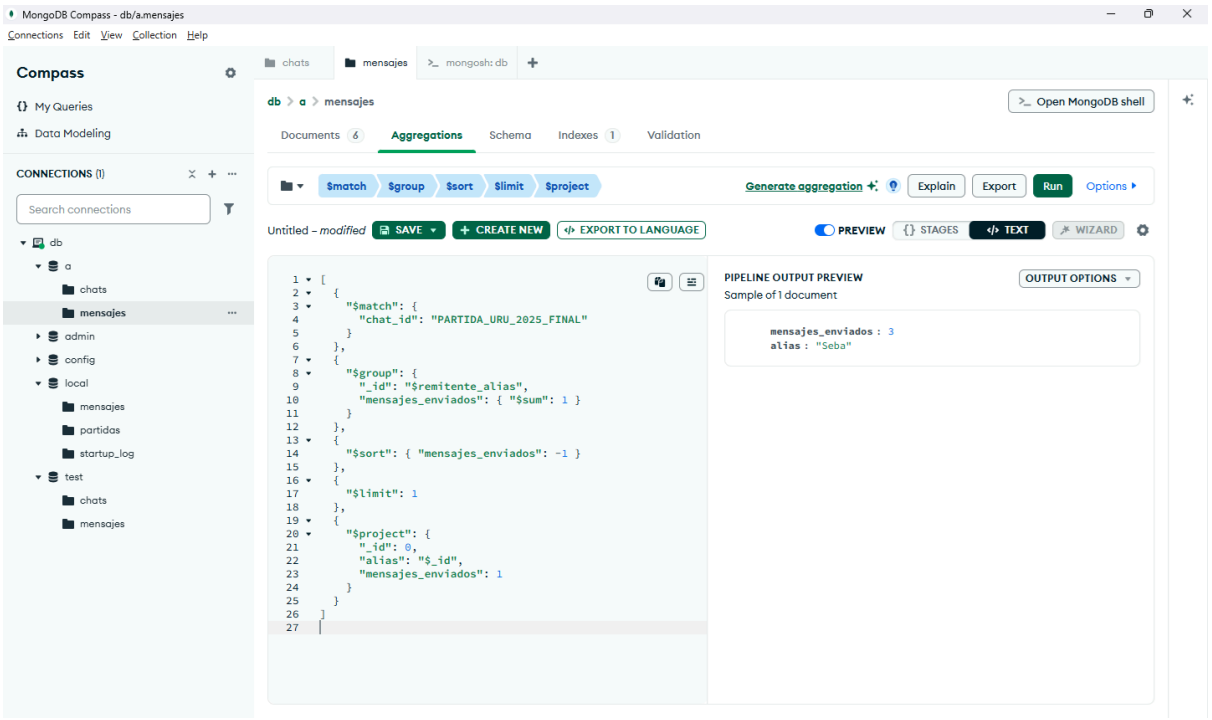
```
        "$ifNull": ["$interacciones.denuncias", []]
      }
    }
  }
 }
]
```



4.2)

```
[
  {
   "$match": {
    "chat_id": "PARTIDA_URU_2025_FINAL"
   }
  },
  {
   "$group": {
    "_id": "$remitente_alias",
    "mensajes_enviados": { "$sum": 1 }
   }
  },
  {
   "$sort": { "mensajes_enviados": -1 }
  },
  {
   "$limit": 1
```

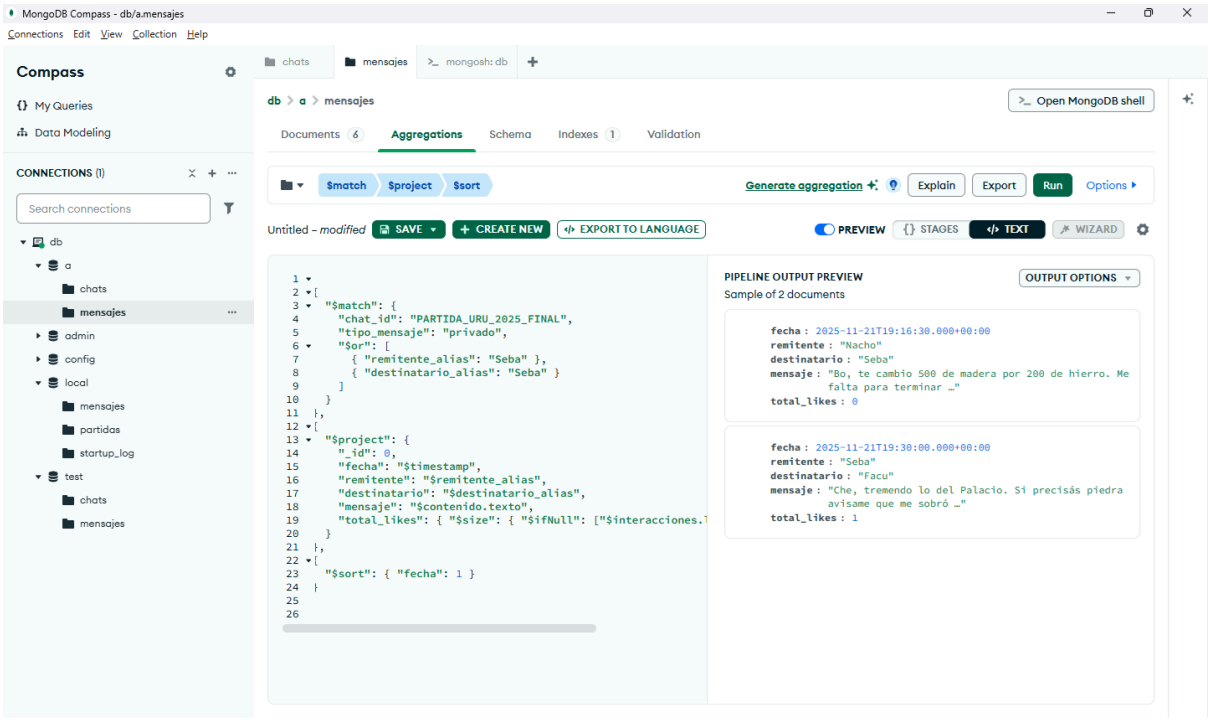
```
  },
  {
   "$project": {
    "_id": 0,
    "alias": "$_id",
    "mensajes_enviados": 1
   }
  }
 ]
```



4.3)

```
[
  {
   "$match": {
    "chat_id": "PARTIDA_URU_2025_FINAL",
    "tipo_mensaje": "privado",
    "$or": [
     { "remitente_alias": "Seba" },
     { "destinatario_alias": "Seba" }
    ]
   }
  },
  {
   "$project": {
    "_id": 0,
    "fecha": "$timestamp",
    "remitente": "$remitente_alias",
    "destinatario": "$destinatario_alias",
```

```
      "mensaje": "$contenido.texto",
      "total_likes": { "$size": { "$ifNull": ["$interacciones.likes_usuarios", []] } }
    }
  },
  {
    "$sort": { "fecha": 1 }
  }
]
```



5)

1. ¿Cómo aplicaron los conceptos del material de estudio en su modelado de MongoDB?

A diferencia de Oracle, donde hubiéramos necesitado al menos 6 o 7 tablas (Mensajes, Chats, ChatParticipantes, Likes, Denuncias, Reacciones, etc.) con JOINs complejos, aquí modelamos el sistema en solo dos colecciones (Chats y Mensajes).

El Esquema Flexible es el concepto más importante que usamos.

Por ejemplo el campo contenido en la colección Mensajes. En un mismo chat, este campo puede contener un documento simple ({ tipo: "texto", texto: "..." }) o un sub-documento complejo ({ tipo: "votacion", votacion: {...} }) o ({ tipo: "accion_juego", accion: {...} }). En Oracle, esto habría sido imposible sin recurrir a columnas JSON.

Aplicamos una decisión de diseño híbrida basada en la cardinalidad de la relación.

Por ejemplo Incrustamos los arrays interacciones.likes_usuarios y interacciones.denuncias dentro del documento Mensajes. Como un mensaje rara vez tendrá millones de "me gusta" (es una relación "uno-a-pocos"), es mucho más rápido leerlos de una sola vez que hacer un JOIN.

No insertamos los mensajes dentro de la partida. Creamos una colección Mensajes separada que "referencia" al chat con chat_id. Hicimos esto porque la relación es "uno-a-muchos" (un chat puede llegar a tener miles de mensajes).

En cuanto a la desnormalización para rendimiento de lectura, se da que este concepto es opuesto a lo que buscamos en Oracle, pero es vital en NoSQL.

Por ejemplo para la parte 4.1 (obtener likes/denuncias), usamos el operador $size sobre los arrays incrustados. Esto es un cálculo instantáneo en memoria. En Oracle, habríamos necesitado un COUNT con GROUP BY sobre una tabla de Likes, lo cual es más lento.

Para la parte 4.2 (usuario con más mensajes), mantuvimos un contador desnormalizado (mensajes_enviados) en el array Chats.participantes. La lectura es instantánea, a costa de una escritura más compleja (usando $inc). En Oracle, esto habría sido un costoso COUNT GROUP BY ORDER BY LIMIT 1 sobre toda la tabla de mensajes.

2. ¿Consideran que su modelado puede ser mejorado en algún aspecto?

Sí, todo modelado implica tomar decisiones y el nuestro tiene puntos de mejora o debate:

1. **Consistencia de Contadores:** Como mencionamos, los contadores en la colección Chats (estadisticas.total_mensajes o participantes.mensajes_enviados) se actualizan por separado de la inserción del mensaje. Esto es rápido, pero no es una operación atómica. Si la inserción en Mensajes tiene éxito pero falla la actualización en Chats (un $inc), los contadores quedarían desincronizados.

   Podríamos usar Transacciones de MongoDB para envolver la inserción y la actualización, garantizando la consistencia ACID que tenemos en Oracle, aunque esto sacrificaría algo de rendimiento.

2. Datos Históricos Estancados: En los mensajes de tipo propuesta_comercio, guardamos el nombre del recurso (ej. "nombre": "Hierro"). Si el administrador del juego cambia el nombre "Hierro" por "Acero" en la base de datos principal (SQL), nuestros logs de chat en MongoDB mostrarán el nombre antiguo.

   Podríamos guardar solo el id_recurso y consultarlo, pero eso viola el principio de desnormalización y velocidad. Para un log histórico, nuestro diseño es aceptable, pero es una limitación a reconocer.

3. Ventajas y desventajas de haber utilizado MongoDB en este subsistema

Ventajas

- **Flexibilidad de Esquema:** Manejar textos, votaciones, acciones y emojis en una estructura relacional rígida como la de Oracle sería complejo. Pero en MongoDB se hace simple debido a que fue diseñado para esto.
- **Rendimiento de Lectura:** Para la UI del chat, necesitamos cargar mensajes y sus reacciones (likes, denuncias) rápido. Al incrustar estos datos, MongoDB realiza una sola operación de

lectura de disco por mensaje. Oracle requeriría múltiples JOINs (a Likes, a Denuncias, etc.) por cada mensaje.

- **Escalabilidad Horizontal:** Un chat es un subsistema de alta escritura (muchos mensajes por segundo). MongoDB está diseñado para escalar horizontalmente (agregar más servidores) mucho más fácil y económicamente que Oracle (que escala verticalmente, es decir, comprando un servidor más grande y caro).

Desventajas

- Falta de Consistencia: A diferencia de Oracle, que es ACID por defecto, MongoDB prioriza la velocidad. Como ya hicimos referencia los contadores pueden desincronizarse. Para un chat es aceptable; para el subsistema de COMERCIO o RECURSO sería fatal.
- Redundancia de Datos: Almacenamos el remitente_alias en cada documento de mensaje. Si un jugador envía 10.000 mensajes, su alias está repetido 10.000 veces. En Oracle, solo guardaríamos un jugador_id, que es mucho más eficiente en almacenamiento.
- Ausencia de JOIN: La falta de JOIN nos obliga a desnormalizar datos. Esto hace que las consultas de lectura sean simples, pero las de escritura más complejas (actualizar en dos lugares por ejemplo) y limita la capacidad de hacer consultas exploratorias que en SQL son triviales.

4. ¿Encuentran algún otro subsistema que podría haberse usado este tipo de base de datos?

1. REGISTRO_RECURSO_RONDA: Este es el candidato más fuerte. Esta tabla es un log de eventos o una base de datos de serie temporal. Registra el estado de los recursos de un país en cada ronda. Esta tabla crecerá infinitamente y rápido. Las consultas siempre serán sobre codigo_partida y id_ronda. Un modelo NoSQL (un documento por país/partida con un array de registros_ronda simila la chat) sería eficiente para leer la historia de un país, evitando JOINs y particionando los datos por partida.
2. Preferencias de Usuario: La tabla USUARIO es rígida. Si quisiéramos guardar configuraciones del jugador (ej. "tema_ui: oscuro", "notificaciones_email: false", "atajos_teclado: {}"), estos son datos no estructurados. En lugar de agregar una columna JSON en Oracle, una colección PreferenciasUsuario en MongoDB sería un caso de uso adecuado.