Proyecto de Diseño de una Base de Datos de Telemetría y UX para la Investigación de Chocolate-Doom: Diseño Conceptual y Lógico (DBS 2025-30)

Colnexus

Integrantes:

- Juan Felipe Vela Jimenez
- Jose Alejandro Contreras Obregón
- Daniel Alejandro Duarte Duarte
- Monica Maria Castro Benitez

1. Introducción y contexto del proyecto

El presente informe detalla el diseño conceptual y lógico para un sistema de base de datos (BD) destinado a recopilar datos de telemetría de juego y experiencia de usuario (UX) generados por el puerto fuente *Chocolate-Doom*. Este proyecto se enmarca dentro de un esfuerzo de investigación para analizar las dinámicas de juego y el rendimiento en un entorno deliberadamente diseñado para emular la tecnología de los años 90 y generar una propuesta que atienda la consultoría la cual nos fue presentada.

1.1. Contextualización de la consultoría Chocolate-Doom

El puerto fuente *Chocolate-Doom* es fundamental para esta investigación debido a su filosofía de diseño conservadora. Este software no sólo reproduce la apariencia y la sensación de los juegos originales (Doom, Heretic, Hexen, Strife), sino que, de manera crucial para el análisis, preserva las limitaciones y errores (*bugs*) de las versiones vanilla de DOS. Este enfoque establece un objetivo único para el repositorio de datos: debe funcionar no solo como un sistema de *big data* para métricas de rendimiento y juego (telemetría), sino también como una herramienta de clasificación y análisis para eventos que son resultado directo de la emulación del código original.

La implicación directa de este enfoque genera unos requerimientos específicos para el modelado de datos, esta base de datos debe estar estructurada para permitir a los investigadores correlacionar eventos de juego (Como un *crash*, un comportamiento inesperado de un enemigo, o la muerte del jugador) con variables de la sesión, la versión específica del software y las características inherentes del juego fuente (Doom o Strife). Esto requiere el uso de dimensiones de referencia detalladas para JuegoFuente y VersionSoftware que contenga metadatos explícitos sobre las características *vanilla* que se preservan.

1.2. Estructura del informe (Modelo de datos)

Con base en las prácticas estándares en cuanto a la arquitectura de bases de datos, este documento se centra en la primera etapa del diseño, buscando diferenciar claramente el modelo conceptual (orientado a la representación de entidades del dominio y sus relaciones, crucial para la comunicación con expertos del dominio y partes interesadas) del modelo lógico (orientado a la implementación relacional, tipificación de datos, integridad y normalización). Este análisis proporciona los cimientos técnicos necesarios para que los administradores de bases de datos (DBA) y los desarrolladores puedan proceder con la fase de modelo físico.

2. Supuestos clave y marco de requisitos

Para garantizar la viabilidad del diseño lógico, se establecen supuestos operacionales y se definen los requisitos esenciales, categorizados como funcionales, no funcionales y éticos.

2.1. Supuestos operacionales y tecnológicos

- **Fuente de datos:** Se asume que los datos son generados por un módulo modificado de *Chocolate-Doom* que asegura que cada evento se registre con atomicidad y con una marca de tiempo precisa, garantizando el orden cronológico dentro de cada sesión de juego.
- **Volumen y rendimiento:** El sistema manejará un volumen de datos de alta velocidad, característico de los sistemas de telemetría, donde la tasa de inserción es prioritaria. Este volumen justifica la adopción de claves auto-incrementales para las tablas de hechos, mejorando la eficiencia de los *joins* y la integridad referencial.
- **Calidad de UX:** Los datos de UX, aunque representan un volumen menor, son determinantes. Se requiere una vinculación estricta con las métricas de rendimiento para análisis correlacionales; por ejemplo, determinar si una configuración gráfica particular o un mapeo de control se correlaciona con un rendimiento de juego superior o inferior.

2.2. Requisitos funcionales (RF)

Los requisitos funcionales describen las acciones, comportamientos e interacciones específicas que el sistema debe ejecutar.

- **Registro de sesión completo:** El sistema tiene la obligación de registrar de forma exhaustiva el ciclo de vida de cada SesionDeJuego, incluyendo el momento preciso de inicio y fin, la duración total, la versión del software utilizada y el estado final de la sesión

(e.g., victoria, derrota, desconexión por error, salida normal).

- **Eventos granulares de telemetría:** El sistema debe capturar eventos de juego atómicos y detallados. Esto incluye, pero no se limita a, Muertes, Disparos, Recolección de Ítems, Cambios de Nivel y el uso de *power-ups*. La naturaleza de la investigación de *Chocolate-Doom*, centrada en los límites del sistema vanilla , exige que estos eventos incluyan metadatos detallados, estos metadatos pueden abarcar coordenadas espaciales del jugador (X, Y, Z), dirección de la visión y, si es posible, el estado interno del *tick* del juego al momento exacto del evento. Para manejar esta heterogeneidad de información contextual, se empleará un atributo tipo JSON o CLOB para almacenar datos cualitativos específicos.
- **Registro de configuración estática:** Es indispensable almacenar la ConfiguracionUX utilizada por el jugador al comienzo de cada sesión. Esto asegura la trazabilidad de los parámetros ambientales, tales como la resolución de pantalla, los ajustes de sonido y el mapeo de controles, permitiendo correlacionar el diseño de la interfaz con los resultados de la telemetría.

2.3. Requisitos no funcionales (RNF)

Los requisitos no funcionales definen los atributos de calidad, como el rendimiento y la escalabilidad, del sistema.

- **Tasa de inserción:** El sistema debe estar optimizado para la inserción masiva. Se requiere una latencia de inserción mínima que permita soportar una tasa agregada de al menos 100,000 registros de eventos por segundo durante picos de actividad de la comunidad de jugadores/investigadores.
- **Escalabilidad de almacenamiento:** Dado el potencial crecimiento exponencial de los datos de telemetría, el diseño lógico debe facilitar la escalabilidad horizontal. La elección de la Tercera Forma Normal (3NF) y la adecuada separación de las dimensiones (tablas de referencia) de las tablas de hechos de alto volumen (eventos) es esencial para minimizar la redundancia y optimizar el espacio de almacenamiento, apoyando la estrategia de escalabilidad.
- **Seguridad:** Se requiere la implementación de controles de acceso robustos para restringir la manipulación y la consulta de datos, especialmente aquellos que, incluso tras la anonimización, pudieran considerarse sensibles o de acceso restringido para fines de investigación.

2.4. Requisitos éticos y de gobernanza de datos

La gestión ética de los datos es un pilar fundamental en cualquier proyecto de investigación. Estos requisitos se traducen directamente en restricciones obligatorias de diseño.

- **Anonimización obligatoria:** Cualquier información que permita la identificación directa de un jugador (PII), incluyendo direcciones IP o identificadores persistentes de dispositivos, debe ser sometida a un proceso riguroso de anonimización o pseudoanonimización inmediatamente después de la ingesta.
- **Segregación de PII:** Para proteger la privacidad y simplificar la gobernanza, la información que, aunque anonimizada o indirecta (como el Alias único), aún permita una vinculación personal, debe almacenarse en una tabla segregada del núcleo de la base de datos de telemetría. Esta segregación de la PII implica que la tabla principal JUGADOR utilizará una clave primaria opaca, el Jugador_ID (una clave sucedánea), que es independiente de cualquier identificador externo sensible. Solo una tabla auxiliar y de acceso restringido, referenciada por este Jugador_ID, contendrá los metadatos potencialmente sensibles.

3. Diseño conceptual: el modelo entidad-relación (DER)

El Modelo Entidad-Relación (DER) proporciona una representación visual y conceptual de las estructuras de datos y sus interconexiones, utilizando entidades y relaciones definidas por cardinalidades.

3.1. Entidades fundamentales

El modelo conceptual identifica las siguientes siete entidades primarias que capturan los elementos centrales del dominio *Chocolate-Doom* y la actividad de recopilación de telemetría:

Entidad	Descripción
User	Representa al estudiante voluntario que participa en el estudio. Contiene información demográfica y el registro de consentimiento.
Player	Identidad utilizada dentro del juego. Puede estar vinculada 1:1 con un User, o permitir múltiples alias para un mismo usuario.
Game (Sesión de Juego)	Instancia única de juego con inicio y fin definidos. Registra contexto, versión del software y las condiciones bajo las cuales se jugó.

Time/Tic	Unidad temporal mínima emitida por el motor del juego. Permite ordenar los eventos en secuencia y reconstruir trayectorias.
Episode / Map / Sector	Estructura espacial jerárquica del juego. Los episodios agrupan mapas y cada mapa se divide en sectores que representan regiones navegables.
TelemetryEvent	Registro atómico capturado en un tic, que describe el estado del jugador (posición, orientación, estadísticas, acciones u otras métricas).
UXInstrument	Instrumento estandarizado para medir la experiencia de usuario (como PENS, GUESS o BANGS), incluyendo su definición y estructura.
UXResponse	Respuestas proporcionadas por un participante a un UXInstrument, permitiendo el análisis subjetivo de experiencia y su relación con el comportamiento observable en el juego.

3.2. Relaciones y cardinalidades clave

Las cardinalidades describen la interacción entre dos elementos de la base de datos, vitales para el diseño conceptual.

Relación	Cardinalidad	Descripción
User — Player	1 : N	Un jugador puede iniciar múltiples sesiones, pero cada Player pertenece estrictamente a un único User.
Player — Game (Sesión de Juego)	1 : N	Un jugador puede participar en múltiples sesiones de juego. Cada sesión es realizada por un solo jugador.
Game — Map	N : N	Una sesión puede recorrer varios mapas, y un mapa puede aparecer en distintas sesiones.
Map — Sector	1 : N	Cada mapa está compuesto de múltiples sectores. Un sector pertenece únicamente a un mapa.
Game — Time/Tic	1 : N	Una sesión de juego se compone de una secuencia de tics (o frames lógicos), los cuales ordenan temporalmente la telemetría.
Time/Tic — TelemetryEvent	1 : 1	Cada tic puede registrar uno o más eventos de telemetría asociados a un jugador en ese momento.
TelemetryEvent — Sector	N : 1	Cada evento ocurre dentro de un sector particular del mapa, lo que permite análisis espaciales.
User — UXResponse	1 : N	Un usuario puede responder uno o varios instrumentos UX a lo largo del estudio.
UXInstrument — UXResponse	1 : N	Cada respuesta corresponde a un instrumento UX específico aplicado al usuario (como PENS, GUESS o BANGS).

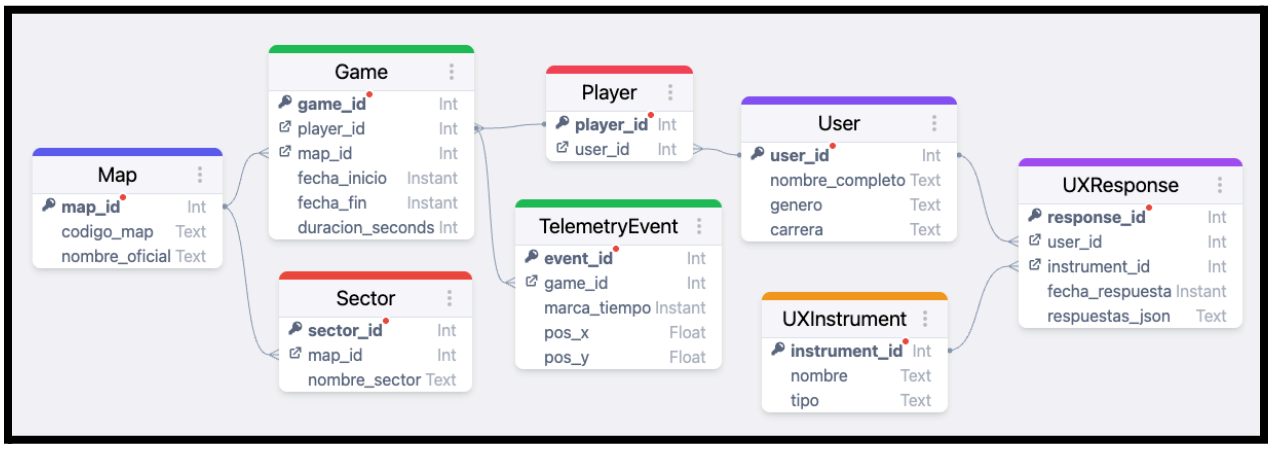
El siguiente extracto representa las entidades clave y sus atributos primarios en el modelo conceptual:

Tabla 3.3: Entidades Clave y Atributos Primarios:

Entidad	Atributos Principales	PK	Relación clave
User_PII	user_id, nombre_completo, correo, fecha_nacimiento	user_id	1 : N con Player
Player	player_id, user_id(FK), alias, fecha_creacion	player_id	N : N con Game (via GameParticipant)
Game	game_id, version_software, fuente, fecha_inicio, fecha_fin, duracion_minutos	game_id	1 : N con Tic
GameParticipant	game_id(FK), player_id(FK)	(game_id, player_id)	Resuelve N:N entre Game y Player
Map	map_id, codigo_map, nombre_oficial	map_id	1 : N con Sector
Sector	sector_id, map_id(FK), nombre_sector	sector_id	N : 1 con Map
Tic	tic_id, game_id(FK), numero_tic	tic_id	1 : 1 con TelemetryEvent
TelemetryEvent	event_id, tic_id(FK), player_id(FK), sector_id(FK), tipo_evento, coords, datos_json	event_id	N : 1 con Player, Sector
UXInstrument	instrument_id, nombre, tipo	instrument_id	1 : N con UXItem
UXItem	item_id, instrument_id(FK), texto_pregunta	item_id	1 : N con UXResponseItem
UXResponse	response_id, user_id(FK), instrument_id(FK), fecha_respuesta	response_id	1 : N con UXResponseItem
UXResponseItem	response_id(FK), item_id(FK), valor	(response_id, item_id)	N : 1 con UXItem

3.3. Esquema E-R

Se genera el esquema correspondiente usando Azimutt



4. Diseño lógico: esquema relacional, normalización y diccionario de datos

El diseño lógico transforma el modelo conceptual en un esquema relacional específico, asegurando la integridad de los datos mediante la normalización y la definición rigurosa de claves y restricciones.

4.1. Derivación y justificación de normalización (3NF)

El objetivo primario de la normalización es eliminar la redundancia y prevenir las anomalías de actualización, inserción y borrado. Para un sistema de investigación, donde la coherencia y la precisión de los datos son imperativas, se justifica la adhesión estricta a la Tercera Forma Normal (3NF). La 3NF asegura que los atributos no clave dependan directamente de la clave primaria completa y no de otros atributos no clave (eliminando dependencias transitivas).

Se ha optado por el uso sistemático de claves autoincrementales (BIGINT auto-incrementales) como claves primarias para las tablas de hechos (SESION_DE_JUEGO, EVENTO_DE_TELEMETRIA). Este enfoque es crítico para satisfacer el RNF de alto rendimiento (RNF-PER01), ya que las claves autoincrementales son pequeñas, deterministas y optimizan la velocidad de la ingesta de datos y las operaciones de *join*.

4.2. Esquema relacional normalizado (3NF)

A continuación, se presenta el esquema relacional en notación formal, donde los atributos

subrayados (\underline{\text{Atributo}}) representan la Clave Primaria (PK) y los atributos seguidos de FK son Claves Foráneas.

1. **JUEGO_FUENTE** (\underline{Juego_ID}, Nombre, Lanzamiento_Original)
2. **VERSION_SOFTWARE** (\underline{Version_ID}, Juego_ID (FK), Nombre_Puerto, Numero_Version, Preserva_Bugs)
3. **MAPA** (\underline{Mapa_ID}, Codigo_Mapa, Nombre_Oficial, Juego_ID (FK))
4. **JUGADOR** (\underline{Jugador_ID}, Alias_Anonimizado (UQ), Fecha_Registro)
5. **SESION_DE_JUEGO** (\underline{Sesion_ID}, Jugador_ID (FK), Version_ID (FK), Hora_Inicio, Duracion_Segundos, Estatus_Final)
6. **CONFIGURACION_UX** (\underline{Sesion_ID} (PK, FK), Resolucion_Pantalla, Detalles_Controles, Otros_Ajustes)
7. **CAUSA_EVENTO_REF** (\underline{Causa_ID}, Tipo_Causa, Descripcion_Corta)
8. **SESION_MAPA_TRANSICION** (\underline{Sesion_ID} (FK), \underline{Mapa_ID} (FK), \underline{Momento_Entrada}, Momento_Salida)
9. **EVENTO_DE_TELEMETRIA** (\underline{Evento_ID}, Sesion_ID (FK), Mapa_ID (FK), Causa_ID (FK), Marca_Tiempo_Absoluta, Tipo_Evento, Coordenadas_XYZ, Datos_JSON_Contexto)

4.3. Análisis de dependencias funcionales y justificación de 3NF

La estructura propuesta garantiza el cumplimiento de la 3NF, esencial para la consistencia en el análisis de datos históricos.

- **Cumplimiento de 2NF:** Todas las tablas con claves compuestas, como SESION_MAPA_TRANSICION, aseguran que todos los atributos no clave (Momento_Salida) dependen de la clave compuesta completa (\underline{Sesion_ID}, \underline{Mapa_ID}, \underline{Momento_Entrada}).
- **Cumplimiento de 3NF (Eliminación de Transitividad):** La separación entre JUEGO_FUENTE y VERSION_SOFTWARE ejemplifica la eliminación de la dependencia transitiva. Si el nombre del juego fuente (Nombre) estuviera almacenado directamente en la tabla SESION_DE_JUEGO, se violaría la 3NF porque Nombre dependería de Juego_ID, que a su vez depende de Version_ID, que es la FK directa. Al normalizar y aislar las características del juego fuente, cualquier cambio en el nombre o propiedades de un juego no requiere actualizaciones masivas en las tablas de telemetría (hechos),

previniendo anomalías de actualización y manteniendo la integridad y la velocidad del sistema.

4.4. Diccionario de datos y restricciones de integridad

El Diccionario de Datos establece los detalles técnicos de cada atributo y sus restricciones, asegurando la calidad de los datos recopilados.

Tabla 4.4: Diccionario de datos del esquema lógico

Tabla	Atributo	Tipo de Dato	PK/FK/UQ	Restricciones de Integridad	Notas Operacionales
jugador	Jugador_ID	INT	PK	not null. auto_increment.	Clave Sucedánea opaca.
jugador	Alias_Anonimizado	VARCHAR (50)	FQ	not null.	Unicidad para identificación de usuario anonimizado.
sesion_de_juego	Sesion_ID	BIGINT	PK	not null. auto_increment.	Clave de hecho principal.
sesion_de_juego	Jugador_ID	INT	FK	not null. referencia jugador.	ON DELETE RESTRICT (Protección de datos de sesión).
evento_de_telemetria	Evento_ID	BIGINT	PK	not null. auto_increment.	Mayor volumen de datos (RNF-PER01).
evento_de_telemetria	Sesion_ID	BIGINT	FK	not null. referencia sesion_de_juego.	ON DELETE CASCADE (Dependencia existencial).
evento_de_telemetria	Tipo_Evento	VARCHAR (30)		not null. check (in 'muerte', 'disparo', 'bug',...).	Restricción de comprobación para tipología.
evento_de_telemetria	Datos_JSON_Contexto	JSON/CLOB		nullable.	Almacenamiento de metadatos específicos del *tick* y coordenadas.
configuracion_ux	Sesion_ID	BIGINT	PK, FK	not null. referencia sesion_de_juego.	Garantiza la relación 1:1 de configuración por sesión.

4.5. Integridad referencial y restricciones operacionales

La implementación de la integridad referencial a través de Claves Foráneas (FK) es vital para asegurar que las referencias a las dimensiones sean válidas.

- **Restricciones de integridad referencial:**
    - Para la relación entre SESION_DE_JUEGO y EVENTO_DE_TELEMETRIA, se aplica ON DELETE CASCADE. Esto significa que si se elimina una sesión (por ejemplo, por corrupción de datos), todos sus eventos asociados deben eliminarse automáticamente, ya que no tienen sentido fuera de ese contexto.

    - Sin embargo, para las referencias a las dimensiones principales (JUGADOR, VERSION_SOFTWARE, MAPA), se aplica ON DELETE RESTRICT (o NO ACTION). Esto evita la eliminación accidental de datos de dimensión que son referenciados por millones de filas de hechos, preservando la coherencia analítica.
- **Restricciones de comprobación:** Se deben utilizar restricciones de comprobación para validar la calidad de los datos de investigación. Por ejemplo, en la tabla de referencia CAUSA_EVENTO_REF, se puede imponer un CHECK para asegurar que el Tipo_Causa solo pueda ser 'Bug_Vanilla', 'Limite_Tecnico', o 'Error_Jugador', lo que facilita la clasificación necesaria para el análisis de la autenticidad retro de *Chocolate-Doom*.

5. Consideraciones avanzadas y conclusiones

5.1. Implementación de requisitos éticos en el diseño lógico

El requisito ético de segregación de la PII se resuelve mediante un diseño en capas que aísla la información sensible. La clave principal de la tabla JUGADOR es el Jugador_ID, una clave sucedánea opaca que no contiene información directamente identificable.

Para cumplir con la gobernanza de datos , se debe introducir una tabla auxiliar de alto control de acceso:

- **Tabla lógica adicional (PII):** JUGADOR_PII (\underline{Jugador_ID} (PK, FK), IP_Hashed, Email_Hashed, Fecha_Consentimiento, Identificador_Persistente).

La tabla JUGADOR_PII mantiene el mapeo a los identificadores originales, pero se almacena en un repositorio de datos separado con acceso altamente restringido. Dado que la inmensa mayoría de la base de datos de telemetría solo referencia el Jugador_ID (la clave sucedánea sin significado externo), el diseño garantiza que el núcleo de la investigación es anónimo, reduciendo la exposición a riesgos legales y de privacidad.

5.2. Preparación para la fase de diseño físico

Aunque el diseño lógico es conceptualmente independiente del Sistema de Gestión de Bases de Datos (DBMS), la necesidad de cumplir con la alta tasa de inserción (RNF-PER01) requiere consideraciones para la optimización física.

- **Indexación:** Se recomienda la indexación intensiva de todas las claves foráneas en las tablas de hechos de alta cardinalidad, especialmente Sesion_ID, Mapa_ID, y Causa_ID en EVENTO_DE_TELEMETRIA. La velocidad de las consultas analíticas depende

directamente de la eficiencia de los *joins* a través de estas claves.

- **Particionamiento:** Dada la naturaleza de serie temporal de la telemetría, la tabla EVENTO_DE_TELEMETRIA debe ser particionada. El particionamiento basado en la fecha o el rango de la Marca_Tiempo_Absoluta o la Hora_Inicio de la sesión mejorará significativamente el rendimiento para consultas de análisis de tendencias.

5.3. Conclusiones del diseño lógico

El diseño lógico para la Base de Datos de Telemetría y UX de *Chocolate-Doom* se ha desarrollado con un enfoque en la rigurosidad científica y la escalabilidad operativa. La adhesión a la Tercera Forma Normal (3NF) garantiza la integridad y la coherencia de los datos históricos, esenciales para la investigación. La utilización estratégica de claves auto-incrementales aborda los requisitos no funcionales de rendimiento en entornos de alta ingesta de datos.

Finalmente, el diseño incorpora directamente las necesidades específicas del dominio, asegurando la capacidad de clasificar los eventos de juego en función de las características preservadas del código *vanilla* (mediante las dimensiones VERSION_SOFTWARE y CAUSA_EVENTO_REF). Al segregar la información personal identificable, el esquema cumple con los requisitos éticos de gobernanza de datos, lo que resulta en un modelo de datos robusto, extensible y apto para la publicación académica (DBS 2025-30).

Fuentes citadas

Chocolate Doom - The Doom Wiki at DoomWiki.org, https://doomwiki.org/wiki/Chocolate_Doom 2. Chocolate Doom, https://www.chocolate-doom.org/wiki/index.php/Chocolate_Doom 3. Explicación del modelado de datos: Conceptual, físico y lógico - Couchbase, https://www.couchbase.com/blog/es/conceptual-physical-logical-data-models/ 4. Modelado dimensional: Claves primarias y foráneas - IBM, https://www.ibm.com/docs/es/ida/9.1.2?topic=entities-primary-foreign-keys 5. Modelo de datos — documentación de Bases de Datos I | Proyecto II - 0.0.1, https://manual-tecnico-bd-oracle.readthedocs.io/es/latest/Modelo%20de%20datos.html 6. La gestión ética de los datos - IADB Publications, https://publications.iadb.org/es/la-gestion-etica-de-los-datos 7. Create an Entity Relationship Diagram in Lucidchart - Lucid Help Center, https://help.lucid.co/hc/en-us/articles/16471565238292-Create-an-Entity-Relationship-Diagram-in-Lucidchart 8. Clave primaria, integridad referencial, comprobación y restricciones de unicidad - IBM, https://www.ibm.com/docs/es/db2/11.1.0?topic=concepts-primary-key-referential-integrity-check-unique-constraints

AML CODE:

```
User
  user_id integer pk
  nombre_completo text
  correo text
  genero text
  carrera text

Player
  player_id integer pk
  user_id integer -> User(user_id)
  alias varchar(100)

Game
  game_id integer pk
  player_id integer -> Player(player_id)
  map_id integer -> Map(map_id)
  version_software varchar(100)
  fecha_inicio timestamp
  fecha_fin timestamp
  duracion_seconds integer

Map
  map_id integer pk
  codigo_map varchar(50)
  nombre_oficial varchar(200)
```

Sector
  sector_id integer pk
  map_id integer -> Map(map_id)
  nombre_sector varchar(150)

TelemetryEvent
  event_id integer pk
  game_id integer -> Game(player_id)
  marca_tiempo timestamp
  tipo_evento varchar(80)
  pos_x numeric
  pos_y numeric
  resultados text

UXInstrument
  instrument_id integer pk
  nombre varchar(100)
  tipo varchar(50)

UXResponse
  response_id integer pk
  user_id integer -> User(user_id)
  instrument_id integer -> UXInstrument(instrument_id)
  fecha_respuesta timestamp
  respuestas_json text