

**Reporte Segunda Entrega Proyecto DBS**

**Fecha de entrega: 09/11/2025**

**Samuel Borrás Mahecha**

**Luis Esteban Chaustre Garzón**

**Daniel Prieto Ordoñez**

**ANDRES OSWALDO CALDERON ROMERO, Ph.D.**

**PONTIFICIA UNIVERSIDAD JAVERIANA**

**FACULTAD DE INGENIERÍA**

**BASES DE DATOS**

**BOGOTÁ D.C**

**2025**

## Contenido

1. Introducción .....	3
2. Suposiciones y Requerimientos.....	3
3. Diagrama Entidad Relación .....	4
4. Diccionario de Datos, FKs, PKs.....	5
5. Justificación de la Normalización hasta 3NF.....	7
6. Implementación de DDL en PostgreSQL .....	8
7. Población de las relaciones UXCuestionario y Pregunta.....	9
8. Inserción de Datos Restantes y Relación Telemetría.....	9

## **1. Introducción**

El presente reporte corresponde a la Primera Entrega (Parte A: Conceptual y Diseño Lógico) del proyecto “Designing a Telemetry and UX Database for Chocolate-Doom Research”. Este proyecto tiene como propósito el diseño de una base de datos relacional que permita almacenar, organizar y analizar datos de telemetría y experiencia de usuario (UX) obtenidos de sesiones de juego del motor modificado Chocolate-Doom.

El objetivo principal es proporcionar una estructura de datos sólida que soporte la integración de información proveniente de múltiples partidas, incluyendo posiciones de jugadores, estadísticas de combate, niveles, mapas y respuestas a instrumentos de experiencia de usuario como PENS, GUESS o BANGS. Esto permitirá al grupo de investigación realizar análisis sobre patrones de movimiento, cooperación y comportamiento de los jugadores, manteniendo al mismo tiempo la calidad, consistencia y ética en el manejo de los datos recolectados. En esta primera fase se abordan tres componentes fundamentales del diseño de la base de datos:

- Definición de supuestos y requisitos, tanto funcionales como no funcionales, junto con las consideraciones éticas relacionadas con la recopilación y manejo de datos personales.
- Elaboración del diagrama entidad-relación (E-R) con sus respectivas cardinalidades, atributos clave y vínculos entre las entidades principales (usuarios, jugadores, partidas, eventos de telemetría, instrumentos y respuestas UX).
- Derivación del esquema relacional, especificando llaves primarias (PK), foráneas (FK) y restricciones, además de justificar el proceso de normalización aplicado para asegurar la integridad y eficiencia de la base de datos.

Esta etapa constituye la base conceptual del sistema, sobre la cual se construirá la implementación y el proceso de ingestión de datos en fases posteriores. Su correcta elaboración garantiza que el diseño cumpla los objetivos de investigación, facilite consultas analíticas avanzadas y respete las normas de ética y privacidad establecidas.

## **2. Suposiciones y Requerimientos**

A continuación, se presentan las suposiciones, requerimientos funcionales y no funcionales que orientan el diseño de la base de datos propuesta para el proyecto. Estas consideraciones establecen el marco técnico y operativo bajo el cual se desarrollará el sistema, definiendo las condiciones iniciales asumidas para su funcionamiento, las capacidades que debe ofrecer para cumplir con los objetivos de investigación, y los criterios de desempeño, integridad y privacidad que deben mantenerse durante su ejecución. En conjunto, estos elementos garantizan que la solución planteada sea coherente, escalable y éticamente responsable en el manejo de los datos de telemetría y experiencia de usuario.

### *Suposiciones:*

- Se asume que los archivos de telemetría se reciben en formato estándar con estructura fija.
- Se asume que cada registro contiene un identificador de partida, jugador y tiempo.
- Se asume que la frecuencia de captura es constante durante toda la sesión.
- Se asume que cada jugador pertenece a un único usuario.
- Se asume que los datos de los cuestionarios UX están completos y validados antes de su carga.
- Se asume que todos los identificadores de las entidades son únicos.
- Se asume que los datos personales se almacenan de forma separada a las respuestas de los cuestionarios UX para preservar el anonimato.
- Se asume que el sistema operará en PostgreSQL como gestor principal.
- Se asume que las consultas analíticas se ejecutan con volúmenes moderados de datos.
- Se asume que el usuario del sistema cuenta con permisos adecuados para ejecutar el script de carga.

### *Requerimientos funcionales:*

- El sistema debe registrar la información de usuarios.
- El sistema debe cargar archivos a las tablas del modelo.
- El sistema debe validar los datos cargados.
- El sistema debe eliminar registros duplicados.
- El sistema debe registrar errores en una tabla de control.
- El sistema debe ejecutar consultas analíticas sobre trayectorias, proximidad, cooperación o desempeño.
- El sistema debe relacionar respuestas UX con datos de telemetría.

### *Requerimientos no funcionales:*

- El sistema debe mantener integridad referencial en todas las tablas.
- El sistema debe garantizar privacidad de los datos personales.
- El sistema debe soportar veinte mil registros de telemetría sin pérdida de rendimiento.
- El sistema debe incluir índices para optimizar las consultas.
- El sistema debe permitir la reconstrucción completa mediante un script de carga.

## **3. Diagrama Entidad Relación**

El siguiente diagrama entidad–relación (E-R) representa, de manera inicial y sujeta a modificaciones, la estructura conceptual y lógica del sistema diseñado para el proyecto “Designing a Telemetry and UX Database for Chocolate-Doom Research”. Este modelo fue desarrollado utilizando la herramienta Visual Paradigm, lo que permitió una representación clara, estructurada y normalizada de las entidades y sus relaciones.

En el diagrama se pueden observar las cardinalidades que definen el tipo de relación entre las entidades (uno a uno, uno a muchos o muchos a muchos), así como los atributos que deben ser únicos para garantizar la integridad de los datos y aquellos que admiten valores nulos cuando la información es opcional o depende del contexto de captura. Asimismo, se destacan las llaves primarias que identifican de manera exclusiva cada registro dentro de una tabla y las llaves foráneas que establecen los vínculos entre las diferentes entidades, asegurando la integridad referencial del modelo. En conjunto, este diagrama constituye la base estructural sobre la cual se construirá el esquema relacional y los procesos de carga, validación y análisis de los datos.

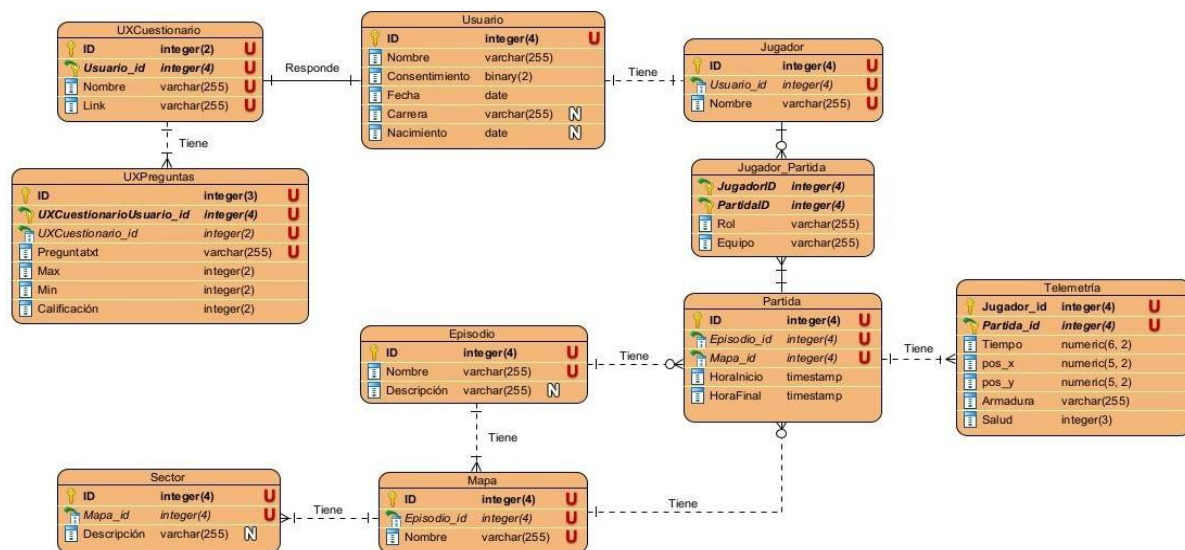


Fig 1. Primer Diagrama E-R.

#### 4. Diccionario de Datos, FKs, PKs

El diccionario de datos presentado a continuación complementa el modelo relacional propuesto para el proyecto. En este documento se especifican, para cada tabla del sistema, los atributos que la componen junto con su respectiva variable, nombre de la variable, tipo de dato, valores permitidos, condición de nulabilidad, y la identificación de si actúan como llave primaria (PK) o llave foránea (FK). Además, se incluye una descripción detallada del propósito y uso de cada campo dentro del modelo. Este nivel de detalle permite comprender de manera precisa la estructura lógica de la base de datos, facilita la implementación en el gestor PostgreSQL y asegura la correcta validación, consistencia e integridad de los datos durante los procesos de carga, análisis y consulta.

<b>Usuario</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Usuario	ID	Integer	0001-9999	No	PK	Valor numérico asignado a un usuario
Nombre Usuario	Nombre	Varchar	Texto Libre	No		Nombre del usuario
Fecha Nacimiento	Nacimiento	mm/dd/yyyy	01-12/01-31/1990 - now()	Si		Edad del Usuario
Carrera	Carrera	Varchar	Lista de Carreras	Si		Carrera que estudia el usuario
Consentimiento	Consentimiento	Booleano	Si/No	No		Consentimiento del usuario para tratar sus datos
Fecha Creación	Fecha	mm/dd/yy	07/10/25 - now()	No		Fecha de creacion del usuario en la base de datos
<b>Jugador</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Jugador	ID	Integer	0001-9999	No	PK	Valor numérico asignado a un Jugador
Nombre Jugador	Nombre	Varchar	Texto Libre	No		Nombre creado para el jugador
ID del Usuario	Usuario_id	Integer	0001-9999	No	FK	Id del usuario al que pertenece el jugador
<b>Partida</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Partida	ID	Integer	0001-9999	No	PK	Valor numérico asignado a una partida
Hora Inicio	HoraInicio	hh:mm	00-23:00-59	No		Hora de inicio de una partida
Hora Final	HoraFinal	hh:mm	00-23:00-59	No		Hora en que termina la partida
ID del episodio	Episodio_id	Integer	0001-9999	No	FK	Id del episodio en el que se jugó
ID del mapa	Mapa_id	Integer	0001-9999	No	FK	Id del mapa en el que se jugó
<b>Episodio</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Episodio	ID	Integer	0001-9999	No	PK	Valor numérico asignado a un episodio
Nombre Episodio	Nombre	Varchar	Lista de Episodios	No		Nombre del episodio
Descripción	Descripción	Varchar	Texto Libre	Si		Descripción del episodio
<b>Mapa</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Mapa	ID	Integer	0001-9999	No	PK	Valor numérico asignado a un mapa
Nombre Mapa	Nombre	Varchar	Lista de Mapas	No		Nombre del mapa
ID del episodio	Episodio_id	Integer	0001-9999	No	FK	Id del episodio al que pertenece el mapa
<b>Sector</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Sector	ID	Integer	0001-9999	No	PK	Valor numérico adignado a un sector
Descripción	Descripción	Varchar	Texto Libre	Si		Descripción del sector
ID del mapa	Mapa_id	Integer	0001-9999	No	FK	Id del mapa al que pertenece el sector
<b>UXCuestionario</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Cuestionario	ID	Integer	001-003	No	PK	Valor numérico asignado a los cuestionarios UX
ID del Usuario	Usuario_id	Integer	0001-9999	No	PK/FK	Id del usuario que contesta el cuestionario
Nombre Cuestionario	Nombre	Varchar	Lista de Cuestionarios	No		Nombre del cuestionario
Link Cuestionario	Link	Varchar	Direcciones URL	No		Link en el que se encuentra el cuestionario
<b>UXPreguntas</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID Pregunta	ID	Integer	001-999	No	PK	Valor numérico asignado a una pregunta del cuestionario
ID Cuestionario	UXCuestionario_id	Integer	001-003	No	FK	Id del cuestionario donde esta la pregunta
ID del Usuario	Usuario_id	Integer	0001-9999	No	PK/FK	Id del usuario que responde la pregunta
Pregunta	Preguntabt	Varchar	Texto Libre	No		Pregunta que se realiza al usuario
Escala Máxima	Max	Integer	5,10	No		Valor máximo con el que se puede calificar la pregunta
Escala Mínimo	Min	Integer		0	No	Valor mínimo con el que se puede calificar la pregunta
Calificación	Calificación	Integer	[Min,Max]	No		Calificación con la que el usuario respondió la pregunta
<b>Telemetría</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID del Jugador	Jugador_id	Integer	0001-9999	No	PK/FK	Id del jugador que esta jugando una partida
ID de la Partida	Partida_id	Integer	0001-9999	No	PK/FK	Id de la partida que se juega
Tiempo	Tiempo	Numeric	0001-9999	No		Cuanto tiempo ha transcurrido en la partida
Posición en eje x	pos_x	Numeric	001-999	No		Posición en eje x que tiene el jugador
Posición en eje y	pos_y	Numeric	001-999	No		Posición en eje y que tiene el jugador
Armadura	Armadura	Varchar	Lista de Armaduras	No		Con que tipo de armadura esta jugando el jugador
Salud	Salud	Integer	000-200	No		Valor de la salud del jugador
<b>Jugador_Partida</b>						
Variable	Nombre Variable	Tipo de Dato	Valores Permitidos	Nulo	PK/FK	Descripción
ID del Jugador	JugadorID	Integer	0001-9999	No	PK/FK	Id del jugador que esta jugando una partida
ID de la Partida	PartidaID	Integer	0001-9999	No	PK/FK	Id de la partida que se juega
Rol	Rol	Varchar	Lista de Roles	No		Cual es el rol del jugador en la partida
Equipo	Equipo	Varchar	Texto Libre	No		A qué equipo pertenece el jugador

**Tabla 1.** Diccionario de datos.

## 5. Justificación de la Normalización hasta 3NF

El esquema propuesto de manera inicial para la base de datos cumplía satisfactoriamente con los criterios de las tres primeras formas normales. Sin embargo, se realizaron cambios en algunas relaciones y atributos con el fin de reducir la complejidad del sistema, el diagrama Entidad-Relación que nace de estas modificaciones se presenta a continuación:

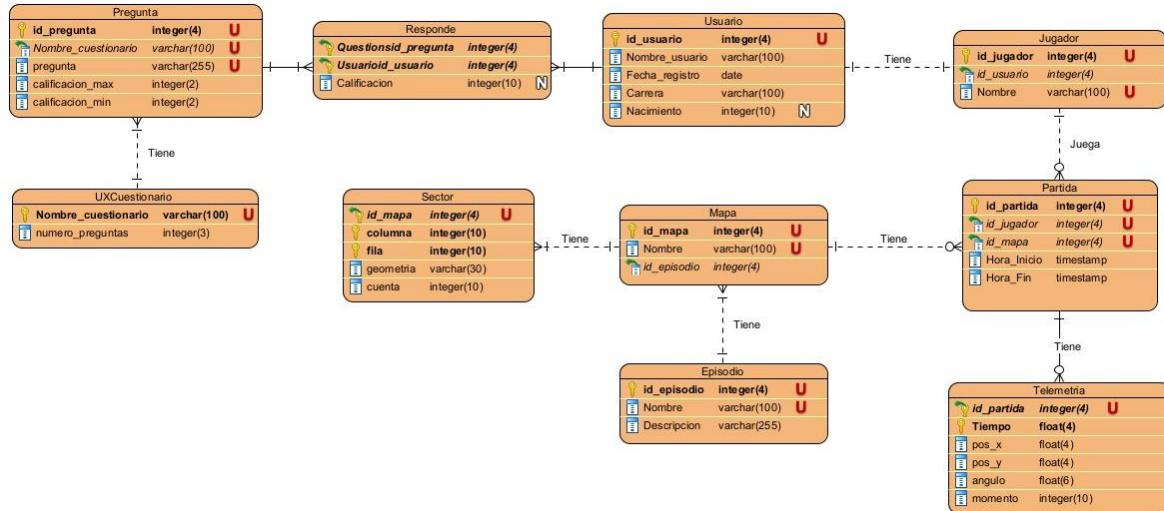


Fig 2. Segundo Diagrama E-R.

Para este planteamiento de diseño también se cumplen las primeras tres formas normales. En Primera Forma Normal (1NF), todas las tablas cuentan con valores atómicos, no se identifican grupos repetitivos y se establece una clave primaria claramente definida en cada entidad. La tabla Sector, por ejemplo, utiliza una clave compuesta (id\_mapa, columna, fila) que garantiza la unicidad de cada sector dentro de un mapa, evitando redundancias.

Para la Segunda Forma Normal (2NF), todas las tablas que emplean claves compuestas tienen dependencias funcionales completas. En el caso ya mencionado de Sector, ningún atributo depende parcialmente de la clave compuesta, sino que todos se derivan de la combinación completa de los campos que la conforman. El resto de entidades usan claves simples, por lo que al estar ya en 1NF pasan automáticamente a 2NF.

En relación con la Tercera Forma Normal (3NF), no se observan dependencias transitivas que comprometan la estructura. Por ejemplo, en Jugador el nombre del jugador depende directamente del identificador del jugador y no del usuario, aun cuando exista una relación uno a uno entre ambas entidades. Asimismo, la entidad Pregunta vincula correctamente el cuestionario mediante una clave foránea sin introducir atributos que dependan indirectamente de otra entidad.



## 6. Implementación de DDL en PostgreSQL

El proceso de construcción de la base de datos *doomProject* partió del diseño conceptual representado en el diagrama entidad-relación, el cual permitió identificar de manera clara las entidades principales, sus atributos y las relaciones existentes entre ellas. Dicho modelo se utilizó como base para elaborar un diccionario de datos, en el que se definieron de forma estructurada los nombres de las tablas, los tipos de datos, las claves primarias, las claves foráneas, las restricciones de unicidad y los dominios permitidos para cada atributo. Este diccionario constituyó la guía técnica para garantizar la correcta correspondencia entre el modelo lógico y su implementación física en el sistema gestor de bases de datos PostgreSQL.

En el diagrama, las entidades Usuario, Jugador, Episodio, Mapa, Sector, Partida, Telemetría, UXCuestionario, Pregunta y Responde se interconectan mediante relaciones uno a uno, uno a muchos y muchos a muchos, reflejando la estructura jerárquica y funcional del proyecto. Por ejemplo, un Usuario puede tener un Jugador asociado, un Jugador participa en múltiples Partidas, y cada Partida genera datos de Telemetría. Asimismo, la parte relacionada con la experiencia del usuario se organiza mediante Cuestionarios compuestos por Preguntas, que a su vez son respondidas por diferentes Usuarios, estableciendo una relación de tipo muchos a muchos materializada en la tabla Responde.

Por su parte, el diccionario de datos permitió estandarizar la nomenclatura de los atributos y definir sus tipos conforme a las necesidades del sistema: se emplearon tipos SERIAL para las claves primarias autoincrementales, VARCHAR para los campos textuales de longitud variable, y FLOAT para la información numérica de precisión variable en la telemetría. Además, se especificaron las reglas de integridad referencial que garantizan la coherencia entre las entidades dependientes, como las claves foráneas entre Mapa y Episodio, o entre Pregunta y UXCuestionario.

Con base en esa documentación, se elaboró el DDL que materializó la estructura lógica del modelo en el entorno físico de PostgreSQL. Cada tabla del DDL se corresponde directamente con una entidad del diagrama y respeta las especificaciones consignadas en el diccionario de datos. Las relaciones entre entidades fueron implementadas mediante sentencias FOREIGN KEY, asegurando la integridad y consistencia del conjunto de datos.

En conclusión, la implementación del DDL fue el resultado de un proceso sistemático de traducción del modelo conceptual a un modelo físico, en el que el diagrama entidad-relación definió la estructura general del sistema, el diccionario de datos precisó los detalles técnicos de cada componente, y el DDL consolidó ambos elementos en una base de datos funcional, eficiente y coherente con los principios del diseño relacional.

En conclusión, la implementación del DDL fue el resultado de un proceso sistemático de traducción del modelo conceptual a un modelo físico, en el que el diagrama entidad-relación definió la estructura general del sistema, el diccionario de datos precisó los detalles técnicos de cada componente, y el DDL consolidó ambos elementos en una base de datos funcional, eficiente y coherente con los principios del diseño relacional. Asimismo, se



adjunta en el entregable un respaldo completo de la base de datos, con el fin de garantizar la trazabilidad del desarrollo, la posibilidad de verificación de su correcta implementación y la facilidad para su futura restauración o despliegue en diferentes entornos de trabajo.

## 7. Población de las relaciones UXCuestionario y Pregunta

El proceso de población de las relaciones UXCuestionario y Pregunta se realizó tomando como referencia el instrumento GUESS, ampliamente utilizado para la evaluación de experiencias de usuario en entornos interactivos. En primera instancia, se registró en la tabla *UXCuestionario* una única entrada correspondiente al instrumento aplicado, identificada con el nombre "GUESS" y con el valor 18 en el campo *numero\_preguntas*, representando el total de ítems que lo componen. Esta inserción permitió establecer el vínculo entre el cuestionario y las preguntas que conforman su estructura, sirviendo como punto de referencia para la posterior carga de datos en la relación *Pregunta*.

Posteriormente, se procedió a poblar la tabla *Pregunta* con las 18 preguntas del instrumento GUESS, como se observa en la Figura 3, asignando a cada una un identificador incremental generado automáticamente mediante la propiedad *SERIAL* del campo *id\_pregunta*. Cada pregunta fue registrada asociada al cuestionario "GUESS", con los valores 1 y 7 en los campos *calificacion\_min* y *calificacion\_max*, respectivamente. Estos valores representan la escala de respuesta tipo Likert utilizada en el instrumento, donde 1 corresponde a "Strongly Disagree" y 7 corresponde a "Strongly Agree", reflejando la intensidad del acuerdo del usuario con cada afirmación. De esta manera, la estructura de datos quedó preparada para almacenar las respuestas de los usuarios en la tabla *Responde*, asegurando coherencia con el modelo teórico del cuestionario y consistencia con los principios del diseño relacional previamente establecidos.

id_pregunta	nombre_cuestionario	pregunta	calificacion_max	calificacion_min
1	GUESS	I find the controls of the game to be straightforward.	7	1
2	GUESS	I find the game's interface to be easy to navigate.	7	1
3	GUESS	I am captivated by the game's story from the beginning.	7	1
4	GUESS	I enjoy the fantasy or story provided by the game.	7	1
5	GUESS	I feel detached from the outside world while playing the game.	7	1
6	GUESS	I do not care to check events that are happening in the real world during the game.	7	1
7	GUESS	I think the game is fun.	7	1
8	GUESS	I feel bored while playing the game.	7	1
9	GUESS	I feel the game allows me to be imaginative.	7	1
10	GUESS	I feel creative while playing the game.	7	1
11	GUESS	I enjoy the sound effects in the game.	7	1
12	GUESS	I feel the game's audio (e.g., sound effects, music) enhances my gaming experience.	7	1
13	GUESS	I am very focused on my own performance while playing the game.	7	1
14	GUESS	I want to do as well as possible during the game.	7	1
15	GUESS	I find the game supports social interaction (e.g., chat) between players.	7	1
16	GUESS	I like to play this game with other players.	7	1
17	GUESS	I enjoy the game's graphics.	7	1
18	GUESS	I think the game is visually appealing.	7	1

Fig 3. Tuplas de la Tabla Pregunta.

## 8. Inserción de Datos Restantes y Relación Telemetría

El proceso de población de las tablas Usuario, Jugador, Responde, Mapa y Episodio se llevó a cabo mediante la inserción manual de los registros, utilizando sentencias SQL del tipo INSERT INTO. Dado que estas relaciones no manejan grandes volúmenes de datos, su llenado fue realizado sin inconvenientes, permitiendo mantener un control preciso sobre la

integridad de los valores insertados. En la tabla Usuario se registraron los datos básicos de identificación de los jugadores, mientras que en Jugador se vinculó cada usuario con su respectiva identidad dentro del sistema de juego, a manera de ejemplo las relaciones se muestran en las Figuras 4 y 5, respectivamente. De igual forma, las tablas Mapa y Episodio fueron pobladas con los nombres y descripciones de los niveles disponibles en el entorno, manteniendo la coherencia entre las claves foráneas y las relaciones establecidas en el diagrama entidad-relación.

```
doomproject=# select * from usuario;
```

id_usuario	nombre_usuario	fecha_registro	carrera
1	EstebanChaustre	2025-11-06	IngSistemas
2	SamuelBorras	2025-11-06	IngSistemas
3	DanielPrieto	2025-11-06	IngSistemas
4	JulioTriana	2025-11-07	Negocios
5	JuanMateus	2025-11-07	Administracion
6	AlejandraPerdomo	2025-11-08	DisenoIndustrial

(6 rows)

*Fig 4. Tuplas de la Tabla Usuario.*

```
doomproject=# select * from jugador;
```

id_jugador	id_usuario	nombre
123	1	EstebiPRO
456	2	Borrium
789	3	Danidih
444	4	Jeelian
100	5	Juanmatro
126	6	Alepero

(6 rows)

*Fig 5. Tuplas de la Tabla Jugador.*

Posteriormente, cada usuario participó en una sesión de juego y, al finalizar, diligenció el cuestionario GUESS. Las calificaciones proporcionadas en este instrumento se registraron en la tabla Responde, respetando el formato definido en el modelo relacional. En cada registro se asoció el identificador del usuario con el de la pregunta correspondiente, junto con la calificación otorgada. Este proceso permitió consolidar la información de la experiencia del usuario de manera ordenada y normalizada, garantizando que las respuestas se vincularan correctamente al cuestionario y al usuario que las emitió.

Por otro lado, el proceso de inserción de datos para las tablas Partida y Telemetría fue considerablemente más complejo y automatizado. Cuando un jugador iniciaba una nueva partida, el archivo modificado proporcionado por el profesor comenzaba a captar los datos de telemetría de forma automática durante toda la sesión. El primer registro obtenido se consideró como la hora de inicio y el último como la hora de finalización de la partida, valores

que fueron posteriormente almacenados en la tabla Partida, como se ve reflejado en la Figura 6. De esta manera, se logró mantener la trazabilidad temporal de cada sesión de juego, vinculando al jugador con su actividad en un mapa determinado.

```
doomproject=# select * from partida;
```

id_partida	id_jugador	id_mapa	hora_inicio	hora_fin
1	456	5	2025-11-08 01:37:41	2025-11-08 03:05:11
2	126	3	2025-11-08 14:32:13	2025-11-08 14:47:52
3	100	3	2025-11-08 16:12:20	2025-11-08 16:24:11
4	123	7	2025-11-08 16:52:42	2025-11-08 18:46:28
5	444	5	2025-11-08 19:10:40	2025-11-08 20:25:40
6	789	7	2025-11-08 21:03:55	2025-11-08 22:35:08

(6 rows)

*Fig 6. Tuplas de la Tabla Partida.*

El llenado de la tabla Telemetría requirió un procedimiento más elaborado, debido al gran volumen de datos generados por el sistema. Para ello, se desarrolló un script en lenguaje C++, con apoyo de herramientas de inteligencia artificial, que automatizó la conversión del formato original a uno compatible con las sentencias SQL de inserción. El código leía los datos desde un archivo de entrada con formato:

*timestamp tic x y z angle momx momy momz*

Y los transformaba al formato deseado:

*(id\_partida, 'timestamp', tic, x, y, z, angle, momx, momy),*

Eliminando la última columna y agregando comillas al valor de tipo timestamp.

El uso de esta herramienta resultó fundamental, ya que permitió convertir miles de registros de telemetría de forma rápida y precisa, generando automáticamente las tuplas en el formato adecuado para los INSERTS dentro de PostgreSQL. Sin esta automatización, el proceso habría sido sumamente tedioso y propenso a errores humanos, pues habría requerido modificar manualmente cada línea del archivo fuente, que alcanzo más de veinte mil tuplas, como se muestra en la consulta ejecutada en la Figura 7. En consecuencia, el script desarrollado facilitó de manera significativa la población masiva y estructurada de la tabla Telemetría, asegurando la coherencia y eficiencia del proceso de carga de datos.

```
doomproject=# select count (*) from telemetria;
```

count
21706

(1 row)

*Fig 7. Conteo de Tuplas de la Tabla Telemetría.*

Finalizado el proceso de inserción de todas las tuplas en las tablas correspondientes, se generó una copia de respaldo de la base de datos utilizando la documentación oficial de PostgreSQL y el comando:

```
pg_dump doomproject > doomproject.sql
```

Este procedimiento permitió obtener un volcado completo con la estructura y los datos del proyecto, garantizando su preservación y posibilidad de restauración futura. El archivo resultante, doomproject.sql, se encuentra adjunto dentro del archivo comprimido .zip entregado junto con el presente trabajo. Por su parte, otros elementos relevantes como el script programado en C++ y los datos de telemetría por partida se encuentran disponibles en el repositorio de GitHub, el cual fue compartido previamente con el profesor para su revisión.