# Memoria

Hemos considerado usar el campo Email como equivalente al UserID en la base de datos relacional.

En la tabla Hall\_of\_fame se ha utilizado Email como clave de clustering key para diferenciar casos en los que dos usuarios del mismo país completen la misma mazmorra con el mismo tiempo. El nombre de la mazmorra es un campo estático, porque cada partición compuesta por una mazmorra id corresponde siempre al mismo nombre de mazmorra. Además, los tiempos se ordenan de forma ascendente para posteriormente seleccionar los 5 mejores tiempos.

En la tabla top\_horde se ha utilizado como clave de partición el evento\_id junto con el País, ya que el mismo evento se repite en distintos países. La clave de clustering es N\_killed con orden descendente, priorizando a los jugadores con mayor número de enemigos derrotados. Hemos considerado no implementar N\_killed como clave de clustering para facilitar las actualizaciones de los datos, pero como la prioridad es la velocidad en lectura, que estén ordenados los datos acelera mucho la velocidad de lectura, ralentizando un poco las escrituras al tener que borrar e insertar los enemigos asesinados cuando se complete una horda (ya que no se puede modificar un dato que compone la clave primaria).

Se ha creado la tabla Usuarios para facilitar las escrituras, ya que algunas solicitudes de inserción requieren datos adicionales como el nombre de usuario, que no siempre se proporcionan en la consulta.

Los datos se han importado desde archivos CSV generados a partir del modelo relacional. El archivo *query\_sql.ipynb* crea las tablas csv desde el modelo relacional y posteriormente se copian los datos en la base de datos Cassandra. Además, se han eliminado las filas con tiempos iguales a 0 para solventar errores ya que en la tabla top\_horde hemos guardado únicamente 5 usuarios por cada mazmorra de cada país.

Para levantar el cluster local de 3 nodos hemos usado un Docker-compose. Nuestra base de datos solo tiene un rack y un centro de datos por lo que no es estrictamente necesario usar NetworkTopologyStrategy, de hecho, SimpleStrategy es suficiente, ya que está diseñado para clústeres con un solo centro de datos según lo que solicita el enunciado, pero como teóricamente estamos implementado una NoSQL para una gran empresa de videojuegos, consideramos que tendrán más de un centro de datos, por eso usamos NetworkTopologyStrategy.

Hemos implementado 3 querys, las querys hall of fame y user statistics requieren de mayor consistencia, sin priorizar la velocidad, por tanto, en ellas hemos cambiado el nivel de consistencia a All. Para la query top horde, no se da tanta importancia a la consistencia sobre la velocidad de lectura, por tanto, hemos usado un nivel de consistencia de Local\_One. Sobre las escrituras hemos usado un nivel Any para que con tal de que un nodo confirme escritura se finalice la escritura, al completar una horda se activa una función que borra y reescribe el numero de monstruos eliminados en la horda.