Trabajos de prácticas

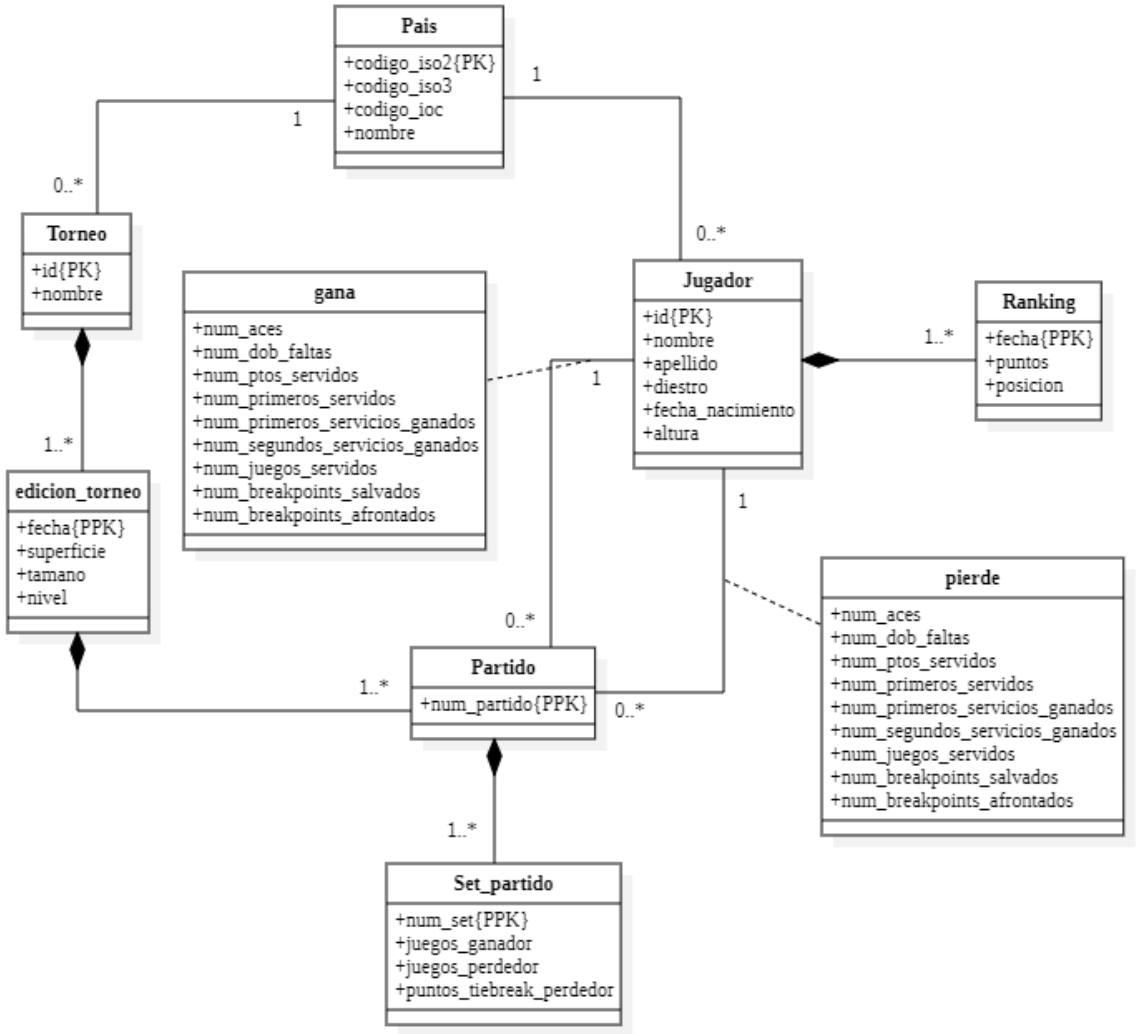
El alumnado debe organizarse en parejas para realizar las tareas descritas en este documento (se pueden hacer excepciones para realizar el trabajo de forma individual, pero ha de consultarse antes con el profesorado). Cada pareja ha de entregar un único archivo .zip en el campus virtual con el siguiente contenido:

- Un archivo PDF con explicaciones de cada una de las tareas y consultas realizadas. En este archivo se debe explicar con el suficiente nivel de detalle el código ejecutado y se deben de mostrar las salidas obtenidas. Ha de realizarse una maquetación apropiada para facilitar la legibilidad del documento.
- Archivos de texto con todo el código ejecutado (consultas, comandos, etc.). Se recomienda generar un archivo para cada una de las secciones siguientes.

La fecha de entrega está indicada en el elemento habilitado para la misma en el campus virtual.

1.- Bases de datos relacionales (PostgreSQL)

Importa en una base de datos relacional PostgreSQL los datos proporcionados en el campus virtual sobre tenis y resuelve las siguientes consultas. El modelo de esta base de datos puede verse en la siguiente imagen.



***Q1**: Muestra todos los ganadores del torneo "Wimbledon" (Nombre, apellidos y año). Ordena el resultado por año.*

	nombre	apellido	ano
1	Michael	Stich	1.991
2	Andre	Agassi	1.992
3	Pete	Sampras	1.993
4	Pete	Sampras	1.994
5	Pete	Sampras	1.995
6	Richard	Krajicek	1.996
7	Pete	Sampras	1.997
8	Pete	Sampras	1.998
9	Pete	Sampras	1.999
10	Pete	Sampras	2.000
11	Goran	Ivanisevic	2.001
12	Lleyton	Hewitt	2.002
13	Roger	Federer	2.003
14	Roger	Federer	2.004
15	Roger	Federer	2.005
16	Roger	Federer	2.006
17	Roger	Federer	2.007
18	Rafael	Nadal	2.008
19	Roger	Federer	2.009
20	Rafael	Nadal	2.010
21	Novak	Djokovic	2.011
22	Roger	Federer	2.012
23	Andy	Murray	2.013
24	Novak	Djokovic	2.014
25	Novak	Djokovic	2.015
26	Andy	Murray	2.016
27	Roger	Federer	2.017
28	Novak	Djokovic	2.018
29	Novak	Djokovic	2.019
30	Novak	Djokovic	2.021
31	Novak	Djokovic	2.022
32	Carlos	Alcaraz	2.023

***Q2**: Muestra los años en los que Roger Federer ganó algún torneo de nivel Gran Slam (G) o Master 1000 (M). Para cada año, muestra el número de torneos y lista sus nombres (ordenados por la fecha de celebración). Ordena el resultado por el año*

	ano	torneos	torneos
1	2.002	1	Hamburg Masters
2	2.003	1	Wimbledon
3	2.004	6	Australian Open, Indian Wells Masters, Hamburg Masters, Wimbledon, Canada Masters, US Open
4	2.005	6	Miami Masters, Indian Wells Masters, Hamburg Masters, Wimbledon, Cincinnati Masters, US Open
5	2.006	7	Australian Open, Miami Masters, Indian Wells Masters, Wimbledon, Canada Masters, US Open, Madrid Masters
6	2.007	5	Australian Open, Hamburg Masters, Wimbledon, Cincinnati Masters, US Open
7	2.008	1	US Open
8	2.009	4	Madrid Masters, Roland Garros, Wimbledon, Cincinnati Masters
9	2.010	2	Australian Open, Cincinnati Masters
10	2.011	1	Paris Masters
11	2.012	4	Indian Wells Masters, Madrid Masters, Wimbledon, Cincinnati Masters
12	2.014	2	Cincinnati Masters, Shanghai Masters
13	2.015	1	Cincinnati Masters
14	2.017	5	Australian Open, Indian Wells Masters, Miami Masters, Wimbledon, Shanghai Masters
15	2.018	1	Australian Open
16	2.019	1	Miami Masters

***Q3**: Muestra los partidos de semifinales (ronda='SF') y final (ronda = 'F') del torneo de "Roland Garros" del 2018. Para cada partido muestra la ronda, el tipo de desenlace, el nombre y apellidos del ganador y el nombre y apellidos del perdedor y el resultado con el número de juegos del ganador y del perdedor en cada set, y opcionalmente en paréntesis el número de juegos del perdedor en el tie break.*

ronda	desenlace	ganador	perdedor	resultado
F	N	Rafael Nadal	Dominic Thiem	6-4 6-3 6-2
SF	N	Dominic Thiem	Marco Cecchinato	7-5 7-6(10) 6-1
SF	N	Rafael Nadal	Juan Martin del Potro	6-4 6-1 6-2

***Q4**: Muestra la lista de jugadores españoles (ES) que ganaron algún torneo de nivel Gran Slam (G). Para cada jugador muestra los siguientes datos resumen de todos sus partidos: número de partidos jugados, porcentaje de victorias, porcentaje de aces, porcentaje de dobles faltas, porcentaje de servicios ganados, porcentaje de restos ganados, porcentaje de break points salvados (de los sufridos en contra), porcentaje de break points ganados (de los provocados a favor).*

jugador	partidos	pcje victorias	pcje aces	pcje dobles faltas	pcje servicios ganados	pcje restos ganados	pcje break salvados	pcje break ganados
Carlos Moya	873	63.9	6.4	3.2	63.4	39	62.8	40.5
Carlos Alcaraz	103	79.8	4.6	3	65.4	41.6	63.8	41.1
Juan Carlos Ferrero	717	64.3	5.1	2.8	63.7	39.7	62.5	40.1
Rafael Nadal	1.276	82.3	4.3	2.3	67.3	42.4	66.1	44.9
Albert Costa	648	58.5	4.5	2.6	62.5	39.5	60.3	41

***Q5**: Lista los jugadores que fueron derrotados (en algún partido del 2018) por el rival de Rafael Nadal de la primera ronda (R128) de Roland Garros de 2018*

jugador	pais
Joao Domingues	PT
Federico Delbonis	AR
Pablo Cuevas	UY
Roberto Carballes Baena	ES
Diego Schwartzman	AR
Denis Istomin	UZ

2.- Datos agregados en SQL (PostgreSQL)

Crea un nuevo esquema llamado “Agregados” en la base de datos en la que has importado los datos de tenis.

Crea los tipos de dato compuestos que necesites para poder almacenar en una sola tabla todos los datos de los partidos (países, torneos, ediciones, partidos, sets y jugadores). Las estructuras deben de poder resolver de forma eficiente las consultas Q1 y Q3 de la sección anterior. Importa los datos de la base de datos de tenis a esta nueva tabla y resuelve las 5 consultas de sección anterior.

Almacena en una única tabla, ahora utilizando estructuras JSON, todos los datos de los partidos (países, torneos, ediciones, partidos, sets y jugadores). Ahora, la estructura diseñada para la tabla debe de ser eficiente para resolver las consultas Q2 y Q4. Importa los datos de tenis en esta nueva tabla y resuelve las 5 consultas de la sección anterior.

3.- Bases de datos distribuidas con SQL (CITUS Data)

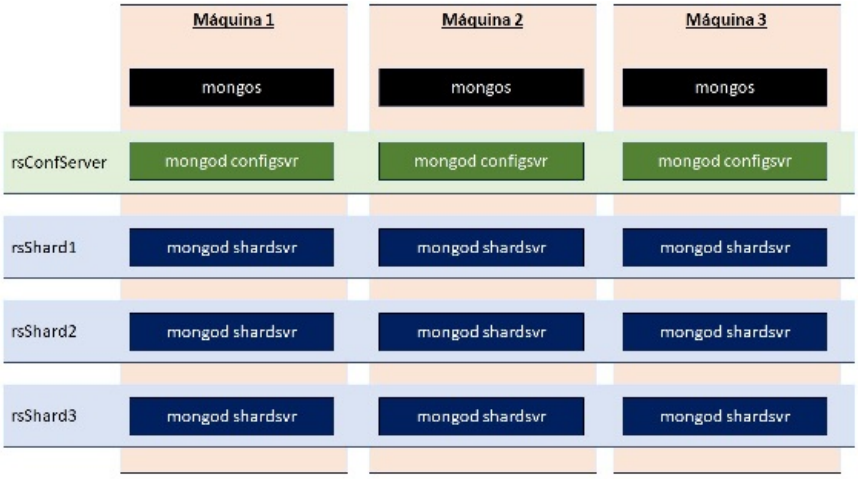
Crea un cluster de Citus con tres nodos, un nodo coordinador (usando la máquina GreiBD) y dos nodos worker (usando clones de la máquina GreiBDServerBase). Importa los datos de tenis en esta base de datos y distribúyelos de la forma que creas más apropiada para obtener un buen rendimiento cuando la base de datos aumente de tamaño (la base de datos aumentará de tamaño al incorporar nuevos partidos correspondientes a nuevas ediciones de los torneos). Prueba las 5 consultas de la sección 1 en esta base de datos. Comprueba que al detener uno de los worker las consultas no funcionan.

Busca información en la web de citus sobre como configurar la replicación CITUS (variable citus.shard_replication_factor). Cambia el factor de replicación a 2, borra las tablas y vuelve a importarlas. Comprueba ahora que ocurre con las consultas al detener uno de los workers.

Busca información sobre el formato columnar para las tablas en Citus. Elimina las tablas que has importado. Importa las tablas de nuevo, pero esta vez no las distribuyas y configura un formato columnar para la tabla de partidos. Compara los tiempos de ejecución de las 5 consultas de la sección 1 entre el formato por filas y el formato columnar.

4.- Bases de datos NoSQL: Documentales (MongoDB)

Crea un cluster de MongoDB siguiendo la arquitectura que puedes ver en la imagen siguiente:



Puedes utilizar la máquina GreiBD y dos clones de la máquina GreiBDServerBase. Una vez hayas creado el cluster, importa los datos de los partidos de la base de datos de tenis en una colección de MongoDB. Diseña el modelo de datos de la colección de manera que sean eficientes las consultas Q1 y Q3. Indexa y particiona la colección de la forma que creas más apropiada. Resuelve las 5 consultas de la sección 1. Detén una de las máquinas y comprueba si todavía puedes ejecutar las consultas. Detén dos máquinas y comprueba de nuevo si puedes ejecutar las consultas. Prueba con distintas opciones de las preferencias y del compromiso de lectura para ver si con dos máquinas detenidas puedes ejecutar las consultas.

4.- Bases de datos NoSQL: Grafos (Neo4J)

Diseña un modelo de datos de grafos para representar la información de los partidos de tenis. Importa todos los datos de la base de datos de tenis en Neo4j directamente desde PostgreSQL usando una conexión JDBC. Una vez tengas el grafo con los nodos y arcos creados, resuelve cada una de las consultas de la sección 1.

5.- Bases de datos NoSQL: Column Family (HBase) [OPCIONAL]

Diseña un modelo de datos de tipo Column Family para representar la información de los partidos de tenis. Realiza una aplicación JAVA que permita importar los datos de los partidos de tenis desde PostgreSQL y que permita ejecutar y ver los resultados de las 5 consultas de la sección 1. Cuando programes el código de las consultas, intenta delegar todo lo que puedas el filtrado de datos a Hbase, utilizando los filtros que sean necesarios.