



TP BASE DE DATOS

Competencia *de Esquiadores*

Alumnos:

- › Motta, Marino Juan
- › Galarza, Jeico Tiziano
- › Yucra, Ángel Patricio

Profesora:

- › Sandra Daujan

ÍNDICE

Enunciado	3
Estrategia de Resolución	4
MER y Supuestos	5
Consultas	7
1 - 7 2 - 9 3 - 12 4 - 14 5 - 15 6 - 18 7 - 20 8 - 21	
Situaciones Lógicas Propias	23

Enunciado de la Base de Datos

Como parte de la organización de las próximas olimpiadas de invierno, se decide la creación de un sistema de información para realizar la gestión de las **pruebas de esquí**.

Del análisis realizado se obtiene la siguiente información:

Los **juegos** se componen de una **serie de pruebas**, en cada una de las cuales intervienen **varios participantes**. Cada **participante** en una prueba puede **intervenir a título individual** (**esquiador individual**) o bien **formando parte de un equipo** en cuyo caso el participante será el equipo (**no el esquiador**). De cada **esquiador** (individual o de equipo) se desea tener el **DNI, el nombre completo y la edad**; y en caso de participar de **forma individual** su **nacionalidad**.

A **cada participante** en una prueba (esquiador individual o equipo participante) se le asigna un **código de participación** que identifica a la **sigla del nombre de la prueba junto a un número secuencial**.

De cada **equipo participante** se conoce un **nombre, un entrenador y los esquiadores que lo componen**. El **que un equipo participe en una prueba no significa que todos los esquiadores que lo componen intervengan en la misma**. Un **esquiador que forma parte de un equipo, no podrá cambiarse a otro ni actuar a título individual mientras duran los Juegos**. Tampoco un **esquiador individual no podrá pasar a formar parte de un equipo**.

Existen una serie de **federaciones de esquí**, cada una de las cuales tiene un **nombre y un número de federados**, en las **federaciones se federan todos los esquiadores**.

Por un acuerdo existente entre las distintas federaciones, no se permite que **ningún esquiador se federe en dos federaciones distintas**. Tampoco se admite que **participen esquiadores** (ni a título individual ni formando parte de un equipo) **que no estén federados**.

Cada federación puede o no administrar varias estaciones de esquí y toda estación se administrará al menos por una federación, aun cuando puede haber **estaciones de esquí administradas conjuntamente por** varias federaciones.

Una **estación de esquí** se identifica por un **código, tiene un nombre, unas personas de contacto, una dirección, un teléfono y un número total de kilómetros esquiables**, así como las **pistas de las que dispone**.

Cada **pista** se identifica a partir del código de la estación de esquí y un **número secuencial**. **Se consideran también como pistas** (para la realización de **pruebas de largo recorrido**) a **varias de estas pistas** (**siempre de la misma estación**) que por sus características físicas pudieran enlazarse. Así por ejemplo la **pista diez estaría compuestas por las pistas dos y cuatro**.

Se requiere, para poder **planificar las pruebas** mantener esta **utilización combinada de las pistas**. Para cada **pista se mantiene** también su **longitud en kilómetros y su grado de dificultad (en la escala azul, verde, roja y negra)**.

La realización de **cada prueba** se desarrollará a lo largo de **varios días** utilizando una serie de **pistas de una única estación**. Los **equipos o esquiadores individuales podrán competir en diferentes pruebas**. Para cada **participante** en una prueba (**equipo o esquiador individual**) se registrará la **fecha o fechas en que participa, el tiempo empleado y la posición obtenida**; en el caso de **equipos** estos datos se obtienen de los **correspondientes a cada uno de los esquiadores del equipo que han intervenido en la prueba**. Cada **prueba** se **identifica por un nombre**, será de un **tipo (fondo, slalom, salto)** **tendrá unas fechas previstas de realización y se registrará el participante ganador y el tiempo empleado** por este.

Estrategia para la Resolución

Para hacer la base de datos, primero analizamos el enunciado, y como se puede ver más arriba, está completamente marcado con diferentes cosas, como por ejemplo las partes *así* son simplemente los atributos que de primera mano supimos que deben estar en las tablas, lo que está marcado **así** hace referencia a algunas de las tablas que identificamos en las primeras lecturas, etc.

Tuvimos que pensar cuidadosamente cada estructura y frecuentemente estuvimos debatiendo sobre separar o no toda la parte de los esquiadores relacionada con “Equipo” e “Individual”, finalmente quedamos en que no hay forma de que queden juntos y cumplan la función que necesitamos, así que separados quedaron.

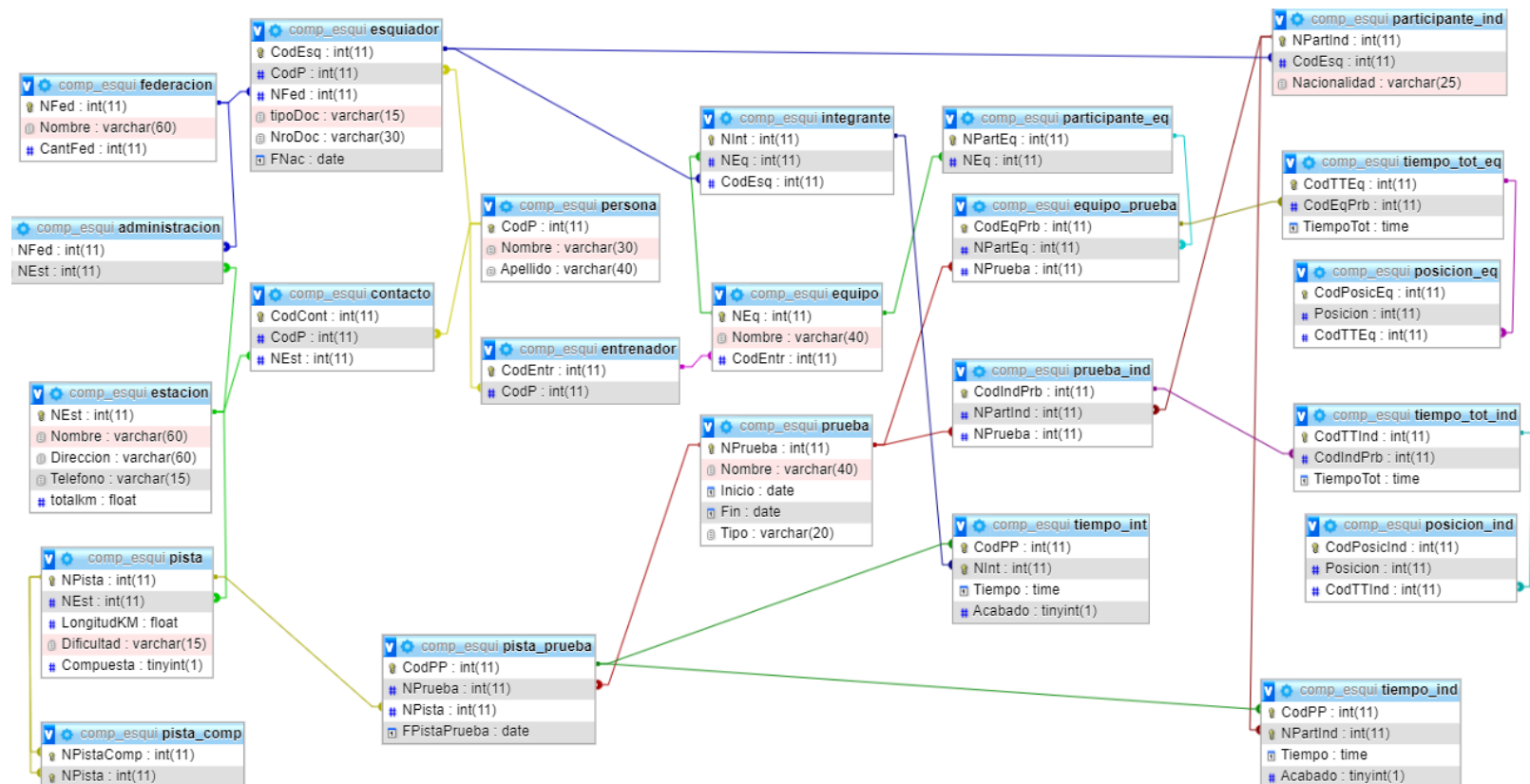
Las pistas también dudamos de separarlas o no, y al final decidimos poner un booleano que diga si son o no compuestas, en el caso de que sí, estarán vinculadas a una tabla dedicada a pistas compuestas.

Para hacer el MER inicial, utilizamos draw.io, y fue la parte más extensa del trabajo por todas las veces que rehicimos ciertas partes. Luego, armamos la base de datos y todo el resto del código (inserts) con el editor de texto Visual Studio Code, ya que estamos más familiarizados y le podemos sacar provecho a algunas funciones que no están en Notepad++ (un ejemplo sería el poder copiar solo ciertas partes de varias filas al mismo tiempo, útil sobre todo en la parte de los inserts para copiar PK y pasarlos como FK a otras tablas). Para comunicarnos durante la resolución utilizamos mayormente la aplicación Discord que tiene chats de texto y de voz, además de que creamos un repositorio en GitHub para poder compartir siempre la última versión del código. (más info en)

Luego, una vez terminada la base de datos CompEsqui, graficamos el MER desde la página de phpmyadmin, y lo ordenamos para que sea legible, puesto que si no se ordena quedan todos encimados uno sobre otro y es imposible trabajarlo. Más adelante mostramos el gráfico del MER.

Durante la parte de los inserts, en las primeras 10 tablas insertamos los datos en conjunto, pero luego ya nos dividimos formalmente uno en toda la parte de Equipos, uno en las tablas de Individual, y uno que avance con el documento para entregar, obviamente todo consultándolo entre nosotros, pero el “trabajo duro” (escribir cada cosa en realidad) quedó dividido de esa manera.

Gráfico Modelo-Entidad-Relación



Supuestos

- Tengamos en cuenta que los contactos solo pueden servir en una única estación a la vez.
- Como una Federación puede administrar varias estaciones y una estación puede ser administrada en conjunto, la tabla “Administración” vincula ambas tablas.
- El “tipo” en pistas dice si la pista es compuesta o no. Si es compuesta, ese número de pista estará reflejado en la tabla PistaComp en NPistaComp, y todas las pistas que contenga estarán en NPista de esa tabla.
- Las pruebas se desarrollarán a lo largo de varios días, y cada pista de la prueba tiene en Pistas_Prueba su FPruebaPista correspondiente.
- En Tiempo_Integrante se registrarán los tiempos de cada *Integrante del equipo* (NInt) que participen en la *pista N* de la *prueba N* (CodP_{pista}P_{prueba}). Si un integrante hace más de una prueba o corre en más de una pista, sus tiempos tendrán distintos CodPistaPrueba, y si en una misma pista hay más de un integrante (relevos u otros equipos), habrá tuplas con distintos NInt.
- El Tiempo_Ind se aplica de la misma manera que Tiempo_Int, habrá un CodPP en cada de *Pista N* en *Prueba N*, y cada participante tendrá su propio NPartInd para los tiempos.

- El Acabado en Tiempo_Int y Tiempo_Ind es un booleano que se pone en 1 cuando se inserta el último tiempo de un integrante del equipo/participante individual en una misma prueba. Mientras alguien lo tenga en esa prueba, nos sirve para poder identificar si ese equipo completó la prueba en su totalidad, y si nadie el equipo lo tiene en esa prueba, no se debe contar para los resultados (porque si no el equipo tendría menos tiempo porque nunca hizo X pista y ganaría sin completar la prueba).
- Cada Pista en la que se deba competir POR PRUEBA va a ser transitada un solo día, es decir que en la misma prueba no habrá varios días para la misma pista y los esquiadores individuales o en equipo participarán solo ese día. Cabe destacar que si OTRA PRUEBA requiere correr en esa pista, no hay problema en que se programe otra fecha para la misma pista siempre que estemos en otra prueba.
- Para buscar a los ganadores (calculable) buscamos a los que tengan posición 1.

Consultas

Consulta 1

Informar ganador y tiempo utilizado de la prueba que se desarrolló en la mayor cantidad de jornadas.

Antes de proyectar los ganadores, hay que identificar la “mayor cantidad de jornadas”, y luego la prueba que cumpla con las condiciones.

```
select count(CodPP)
  from Pista_Prueba
  group by NPrueba;
```

count (CodPP)
3
2
2
4
2
2
3
2
3
2

Con la cantidad de jornadas, ahora hace falta obtener el máximo y a que prueba pertenece

```
select max(e)
  from (select count(CodPP) as "e"
        from Pista_Prueba
        group by NPrueba) A;
```

max (e)
4

```
select NPrueba
  from Pista_Prueba
  group by NPrueba
  having count(CodPP) = (
    select max(e)
      from (select count(CodPP) as "e"
            from Pista_Prueba
            group by NPrueba) A);
```

NPrueba
304

Y para un código más limpio guardaremos el resultado en una variable

```
SET @NPrueba = (  
select NPrueba  
from Pista_Prueba  
group by NPrueba  
having count(CodPP) = (  
select max(e)  
from (select count(CodPP) as "e"  
from Pista_Prueba  
group by NPrueba) A));  
select @NPrueba
```

```
+-----+  
| @NPrueba |  
+-----+  
|      304 |  
+-----+
```

Ahora buscaremos el ganador de esa prueba, vinculando varias tablas a la vez

```
select PI.NPrueba, P.Posicion, NPartInd  
from Prueba_Ind PI, Tiempo_Tot_Ind TTI, Posicion_Ind P  
where Posicion = 1  
and PI.NPrueba = @NPRUEBA  
and PI.CodIndPrb = TTI.CodIndPrb  
and TTI.CodTTInd = P.CodTTInd;
```

```
+-----+-----+-----+  
| NPrueba | Posicion | NPartInd |  
+-----+-----+-----+  
|      304 |         1 |      2133 |  
+-----+-----+-----+
```

Con ese resultado, podemos simplemente sacar el NPartInd a una variable para facilitar buscar los datos del ganador, y lo mismo para el tiempo

```
SET @NPartInd = (  
select NPartInd  
from Prueba_Ind PI, Tiempo_Tot_Ind TTI, Posicion_Ind P  
where Posicion = 1  
and PI.NPrueba = @NPRUEBA  
and PI.CodIndPrb = TTI.CodIndPrb  
and TTI.CodTTInd = P.CodTTInd);  
select @NPartInd;
```

```
+-----+  
| @NPartInd |  
+-----+  
|      2133 |  
+-----+
```

```
SET @TiempoTot = (  
select TTI.TiempoTot  
from Prueba_Ind PI, Tiempo_Tot_Ind TTI, Posicion_Ind P  
where Posicion = 1  
and PI.NPrueba = @NPRUEBA  
and PI.CodIndPrb = TTI.CodIndPrb  
and TTI.CodTTInd = P.CodTTInd);  
select @TiempoTot;
```

```
+-----+  
| @TiempoTot |  
+-----+  
| 00:53:12 |  
+-----+
```


El resultado final queda entonces:

```
select Concat(Nombre, " ", Apellido) as "Nombre y Apellido", @NPrueba, @TiempoTot
  from Persona P, Esquiador E, Participante_Ind PI
  where PI.NPartInd = @NPartInd
  and PI.CodEsq = E.CodEsq
  and E.CodP = P.CodP;
```

Nombre y Apellido	@NPrueba	@TiempoTot
Saray Duran	304	00:53:12

Eso serviría para la parte individual, pero si quisiéramos sacar los equipos si bien hay que cambiar los nombres de las tablas y de los atributos, la lógica se aplica de la misma manera. De todas formas, en esta prueba solo participaron esquiadores individuales y no hay equipos compitiendo en la misma, por lo tanto, la resolución finaliza aquí.

Consulta 2

Listar cada una de las pruebas junto el identificador de pista, la dificultad y en caso de estar compuesta por otras pistas la cantidad de pistas que la componen.

Primero obtenemos una tabla con las pruebas y los demás datos

```
select PP.NPrueba, PP.Npista, P.Dificultad, Compuesta as "Comp"
  from pista_prueba PP
  inner join pista P
  on PP.NPista = P.NPista;
```

NPrueba	Npista	Dificultad	Comp
301	202	Rojos	0
301	203	Negro	0
301	204	Verde	1
302	214	Negro	1
302	220	Verde	0
303	240	Azul	0
303	244	Rojos	1
304	252	Rojos	0
304	253	Negro	0
304	262	Rojos	0
304	263	Rojos	0
305	282	Rojos	0
305	283	Rojos	0
306	301	Verde	0
306	302	Rojos	0
307	320	Azul	0
307	321	Verde	0
307	324	Negro	1
308	330	Azul	0
308	330	Azul	0
309	361	Verde	0
309	362	Rojos	0
309	363	Negro	0
310	381	Verde	0
310	382	Rojos	0

Luego, averiguamos cuantas pistas forman las compuestas

```
select NPistaComp, count(NPista) as "CantPistas"
from Pista_Comp
group by NPistaComp
having NPistaComp in (select PP.Npista
from pista_prueba PP
inner join pista P
on PP.NPista = P.NPista
and compuesta = 1);
```

NPistaComp	CantPistas
204	2
214	3
244	2
324	2

Ahora agregamos la cantidad de pistas con el mismo formato que el primer paso, agregando la cantidad de pistas

```
select PP.NPrueba, PP.Npista, P.Dificultad, Compuesta as "Comp", CantPistas
from pista_prueba PP, pista P, (select NPistaComp, count(NPista) as "CantPistas"
from Pista_Comp
group by NPistaComp
having NPistaComp in (select PP.Npista
from pista_prueba PP
inner join pista P
on PP.NPista = P.NPista
and compuesta = 1)) A
where PP.NPista = P.NPista
and P.NPista = A.NPistaComp;
```

NPrueba	Npista	Dificultad	Comp	CantPistas
301	204	Verde	1	2
302	214	Negro	1	3
303	244	Rojo	1	2
307	324	Negro	1	2

Llamaremos a esta tabla
“CONSULTA 2.1”

Ahora, unimos la tabla anterior con la del primer paso, pero filtrando las compuestas ya que las incorporamos en la segunda tabla. Tal vez es medio confuso y las subconsultas son bastante grandes, así que lo explicamos mediante las imágenes

```
select PP.NPrueba, PP.Npista, P.Dificultad, Compuesta as "Comp", @NULL as "CantPistas"
from pista_prueba PP, pista P
where PP.NPista = P.NPista
and compuesta = 0;
```

NPrueba	Npista	Dificultad	Comp	CantPistas
301	202	Rojo	0	NULL
301	203	Negro	0	NULL
302	220	Verde	0	NULL
303	240	Azul	0	NULL
304	252	Rojo	0	NULL
304	253	Negro	0	NULL
304	262	Rojo	0	NULL
304	263	Rojo	0	NULL
305	282	Rojo	0	NULL
305	283	Rojo	0	NULL
306	301	Verde	0	NULL
306	302	Rojo	0	NULL
307	320	Azul	0	NULL
307	321	Verde	0	NULL
308	330	Azul	0	NULL
308	330	Azul	0	NULL
309	361	Verde	0	NULL
309	362	Rojo	0	NULL
309	363	Negro	0	NULL
310	381	Verde	0	NULL
310	382	Rojo	0	NULL

PRIMERA TABLA
FILTRADA Y MODIFICADA
la llamaremos “CONSULTA 2.2”

Finalmente, utilizamos el operador UNION para juntar los datos de las dos tablas y los ordenamos por el Número de Prueba.

```
> (select PP.NPrueba, PP.Npista, P.Dificultad, Compuesta as "Comp", @NULL as "CantPistas"
UNION
> (select PP.NPrueba, PP.Npista, P.Dificultad, Compuesta as "Comp", CantPistas...
order by NPrueba;|
```

CONSULTA 2.2

CONSULTA 2.1


NPrueba	Npista	Dificultad	Comp	CantPistas
301	202	Rojo	0	NULL
301	203	Negro	0	NULL
301	204	Verde	1	2
302	220	Verde	0	NULL
302	214	Negro	1	3
303	240	Azul	0	NULL
303	244	Rojo	1	2
304	252	Rojo	0	NULL
304	253	Negro	0	NULL
304	262	Rojo	0	NULL
304	263	Rojo	0	NULL
305	282	Rojo	0	NULL
305	283	Rojo	0	NULL
306	301	Verde	0	NULL
306	302	Rojo	0	NULL
307	320	Azul	0	NULL
307	321	Verde	0	NULL
307	324	Negro	1	2
308	330	Azul	0	NULL
309	361	Verde	0	NULL
309	362	Rojo	0	NULL
309	363	Negro	0	NULL
310	381	Verde	0	NULL
310	382	Rojo	0	NULL

Consulta 3

Informar los datos de los esquiadores que pertenecen a los equipos que superen el promedio de integrantes de esquiadores de todos los equipos.


Primero sacamos la cantidad de integrantes por cada equipo, y luego el promedio de esos datos lo guardamos en una variable

```
select count(*) "a"
  from Integrante
  group by NEq;
```



a
5
5
5
6
5
5
4
4
4
5
5
5
6


```
set @PromedioInt = (
  select round (avg(a), 0)
    from (select count(*) "a"
          from Integrante
          group by NEq) a);
select @PromedioInt;
```



@PromedioInt
5

Ahora buscamos los equipos que superen ese promedio

```
select NEq
  from Integrante
  group by NEq
  having count(NEq) > @PromedioInt;
```



NEq
503
512

Y por último volcamos la información pertinente de los esquiadores pertenecientes a esos equipos

```
select concat(Nombre, " ", Apellido) as "Nombre Completo", NEq, TipoDoc, NroDoc
  from Integrante I
    inner join Esquiador E on E.CodEsq = I.CodEsq
    inner join Persona P on P.codP = E.codP
    where I.NEq in(select NEq
                    from Integrante
                    group by NEq
                    having count(NEq) > @PromedioInt);
```

Nombre Completo	NEq	TipoDoc	NroDoc
Mariam Tortosa	512	PC	D01542123
Luis Torres	512	PC	E00149283
Amaia Salmeron	512	PC	E02123421
Lucas Manuel	512	PC	D17385921
Iker Carreño	512	PC	D01837529
Javier Alegria	512	PC	C03243212
Celeste Fuertes	503	CNI	32069231
Sergio Rosa	503	CNI	32960212
Ivan Giulkovich	503	CNI	35109415
Leonel Lopez	503	CNI	33209212
Mireia Hinojosa	503	CNI	36231019
Juliana Patiño	503	CNI	35212669

Consulta 4

Mostrar a cada entrenador junto a la cantidad de equipos que entrena en las olimpiadas.

Primero vamos a la tabla de equipo y como los entrenadores ya están vinculados directamente a los equipos en esta tabla, solamente hay que hacer un count desde la tabla Equipo.

```
select CodEntr, count(NEq)
  from Equipo
  group by CodEntr;
```



CodEntr	count (NEq)
60	1
61	1
62	1
63	2
64	1
65	1
66	1
67	1
68	1
69	2
70	1

Ya que tenemos los códigos de los entrenadores y a cuantos equipos entrena cada uno, necesitamos los datos personales del entrenador, por lo tanto, recurrimos a Entrenador y a Persona:

```
select Nombre, Apellido, CodEntr
  from Persona P
  inner join Entrenador E
  on P.CodP = E.CodP;
```

Nombre	Apellido	CodEntr
Loana	Albear	60
Samuel	Ontiveros	61
Alan	Gomez	62
Maximiliano	Ibarra	63
Sara	Gustani	64
Maria	Martines	65
Luciano	Florian	66
Maximiliano	Power	67
Abel	Muruchi	68
Samuel	Condori	69
Nicolas	Marquez	70

Para finalizar, simplemente juntamos esas consultas en una sola y vinculamos los resultados:

```
select concat(P.Nombre, " ", P.Apellido) AS "Entrenador", E.CodEntr, count(NEq) as "Cant Equipos"
from Entrenador E
    inner join Equipo Eq on Eq.CodEntr = E.CodEntr
    inner join Persona P on E.CodP = P.CodP
    group by CodEntr;
```

Entrenador	CodEntr	Cant Equipos
Loana Albear	60	1
Samuel Ontiveros	61	1
Alan Gomez	62	1
Maximiliano Ibarra	63	2
Sara Gustani	64	1
Maria Martines	65	1
Luciano Florian	66	1
Maximiliano Power	67	1
Abel Muruchi	68	1
Samuel Condori	69	2
Nicolas Marquez	70	1

Consulta 5

Listar nombre y federación de todos los participantes (individual o en equipo) que compitieron en la inauguración de las olimpiadas.

Analizando la base de datos, podemos deducir que la fecha de inauguración de las olimpiadas aparece tanto en Prueba como en Pista_Prueba, pero a nosotros nos sirve directamente apuntar a Pista_Prueba para saber quienes son los que corrieron el primer día en esa pista, ya que si utilizamos solo la prueba nos daría los resultados de los que corrieron en toda la jornada de la misma.

Entonces busquemos la menor fecha y el identificador de la pista que se corrió ese mismo día

```
set @FInaug = (
select min(FPistaPrueba)
from Pista_Prueba);
select @FInaug;
```



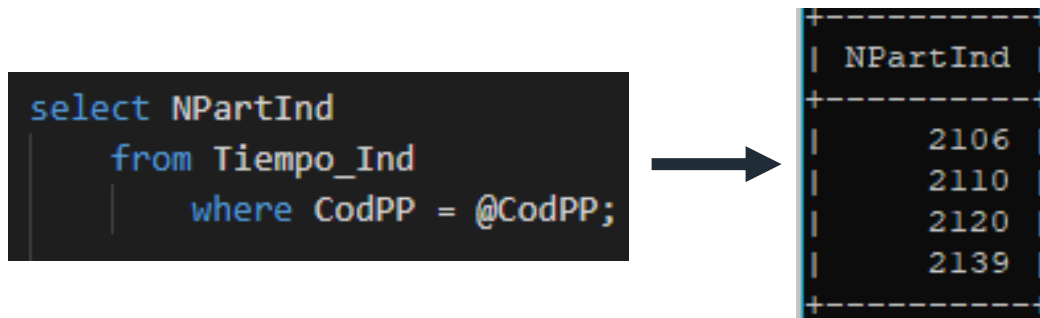
```
+-----+
| @FInaug |
+-----+
| 2020-08-04 |
+-----+
```

```
set @CodPP = (
select CodPP
from Pista_Prueba
where FPistaPrueba = @FInaug);
select @CodPP;
```



```
+-----+
| @CodPP |
+-----+
| 1200 |
+-----+
```


Ahora solo hay que ver quienes son los que corrieron allí, y puesto que en cada pista (por prueba) se corre en un solo día, solo necesitamos los tiempos (prueba de que corrieron) de esa pista que los obtendremos desde Tiempo_Ind y Tiempo_Int, pero como esta Pista en esa fecha es solamente utilizada por los participantes individuales, no es necesario buscar los tiempos de los integrantes de equipos (de todas formas si lo hiciéramos simplemente seguimos el mismo procedimiento con distintos nombres/tablas y al final juntamos ambos resultados con union).



Ahora ya con la identificación de los participantes, simplemente debemos vincular a las demás tablas y mostrar los datos pertinentes

```
select concat(P.Nombre, " ", Apellido) as "Nombre y Apellido", F.Nombre "Federacion"
from Participante_Ind PI
inner join Esquiador E on PI.CodEsq = E.CodEsq
inner join Persona P on E.CodP = P.CodP
inner join Federacion F on E.NFed = F.NFed
where PI.NPartInd in (select NPartInd
from Tiempo_Ind
where CodPP = @CodPP);
```

Nombre y Apellido	Federacion
Feliciano Valdivia	Federacion Argentina de Ski y Andinismo
Mihaela Zheng	Federacion de Ski y Snowboard de Chile
Carlos Moreno	U.S. Ski and Snowboard Association
Irene Novoa	Real Federacion Española de Deportes de Invierno

Como dijimos previamente, ningún integrante de equipo participó en esa prueba ni mucho menos en esa pista el primer día, así que acá terminaría la resolución de la consulta. Sin embargo, como extra, podemos realizar el mismo procedimiento para los integrantes de equipos que hayan participado en el cierre (último día) de la competencia (por dar un ejemplo con equipos).

```
set @FCierre = (
    select max(FPistaPrueba)
    from Pista_Prueba);
select @FCierre;
```

@FCierre
2020-10-19

```
set @CodPP = (
    select CodPP
    from Pista_Prueba
    where FPistaPrueba = @FCierre);
select @CodPP;
```

@CodPP
1282

```
select NInt
    from Tiempo_Int
    where CodPP = @CodPP;
```

NInt
612
622
632
661

```
select concat(P.Nombre, " ", Apellido) as "Nombre y Apellido", F.Nombre "Federacion"
    from Integrante I
    inner join Esquiador E on I.CodEsq = E.CodEsq
    inner join Persona P on E.CodP = P.CodP
    inner join Federacion F on E.NFed = F.NFed
    where I.NInt in (select NInt
        from Tiempo_Int
        where CodPP = @CodPP);
```

Nombre y Apellido	Federacion
Olivia Gomis	Federacion de Ski y Snowboard de Chile
Hernan Gomez	U.S. Ski and Snowboard Association
Daniela Escudero	U.S. Ski and Snowboard Association
Modesto Diaz	Real Federacion Espa?ola de Deportes de Invierno

Consulta 6

Identificar los esquiadores que al final de la competencia no participaron en ninguna prueba y formaban parte de un equipo, junto al nombre del equipo; ordenado por equipo.

Para identificar a los esquiadores que no participaron y estaban en un equipo, primero debemos buscar los que sí participaron, y luego usar ese resultado como filtro para los integrantes.

Así que busquemos a todos los integrantes que tengan tiempos registrados (compitieron).

```
select NInt
      from Tiempo_Int
      group by NInt;
```

(es bastante largo por eso hay varias columnas)

The diagram illustrates a data flow process. On the left, a table with 16 rows is shown, with the first row labeled 'NInt'. An arrow points from this table to a middle table with 36 rows. Another arrow points from the middle table to a right table with 36 rows, which is labeled '36 rows in set' at the bottom.

NInt
600
601
603
604
605
606
610
611
612
615
616

620
621
622
623
624
625
626
630
631
632
633
634
635
636


641
642
645
646
649
650
653
654
659
660
661

36 rows in set

Ahora averigüemos todos los integrantes que NO aparezcan en esa lista, utilizándola como filtro

```
select NInt, NEq
  from Integrante
  where NInt not in (select NInt
                    from Tiempo_Int
                    group by NInt);
```

NInt	NEq
602	500
607	501
608	501
609	501
662	502
663	502
655	503
656	503
657	503
658	503
613	504
614	504
617	505



618	505
619	505
643	506
644	506
647	507
648	507
651	508
652	508
627	510
628	510
629	510
637	512
638	512
639	512
640	512

28 rows in set

Ahora ya teniendo identificados a los esquiadores junto con el Número de equipo en el que están, con un par de inner joins y un ordenar por NEquipo obtendremos el resultado esperado.

```
select I.CodEsq, concat(P.Nombre, " ", P.Apellido) as "Nombre y Apellido", EQ.Nombre "Equipo"
  from Integrante I
 inner join Esquiador E on I.CodEsq = E.CodEsq
 inner join Persona P on E.CodP = P.CodP
 inner join Equipo EQ on I.NEq = EQ.NEq
    where Nint not in (select NInt
                      from Tiempo_Int
                      group by NInt)
    order by EQ.NEq;
```

CodEsq	Nombre y Apellido	Equipo
802	Agustin Servantes	Ushuaia Canopy
807	Mihaela Jim?nez	Rainbow7
808	Gabriel Dominguez	Rainbow7
809	Juan Motta	Rainbow7
900	Barbara Jaime	Vaevictis
901	Brenda Lagos	Vaevictis
888	Ivan Giulkovich	Heretics
889	Leonel Lopez	Heretics
890	Mireia Hinojosa	Heretics
891	Juliana Pati?o	Heretics
821	Saray Borrego	Gen G
822	Claudia Montilla	Gen G
825	Sofia Moreyra	Santiago Wanderers
826	Jimi Hendrix	Santiago Wanderers
827	Adrian Ojeda	Santiago Wanderers
867	Thiago Fernandez	Rebirth
868	Manuel Suarez	Rebirth
871	Raul Romero	Infinity
872	Gheorghe Giner	Infinity
875	Gemma Alves	Damwon
876	Matias Antros	Damwon
840	Felix Galvan	Giants
841	Fabian Gutierrez	Giants
842	Agustin Saponare	Giants
850	Amaia Salmeron	Fnatic
851	Lucas Manuel	Fnatic
852	Iker Carre?o	Fnatic
853	Javier Alegria	Fnatic

28 rows in set

Consulta 7

Identificar a las estaciones de esquí que sean administradas por más de una federación, indicando nombre, km esquiables y cantidad de pistas, ordenadas alfabéticamente.

Para encontrar las estaciones deseadas, hay que filtrar las que sean administradas por una sola federación y usar las restantes.

```
select NEst
  from administracion
  group by NEst
  having count(NFed) > 1;
```



NEst
155
160
161
162
163
164
165
166
167
168

Ahora teniendo identificadas a las estaciones que necesitamos, hace falta proyectar los demás datos pertinentes, vinculandolos en base a los Números de Estación de la consulta anterior

```
select E.Nombre, E.NEst, count(NPista) as "Cantidad de Pistas", E.TotalKM
  from pista P, estacion E
  where E.NEst = P.NEst
  group by NEst
  having NEst in (select NEst
                  from administracion
                  group by NEst
                  having count(NFed) > 1)
  order by Nombre;
```

NEst	Nombre	Cantidad de Pistas	TotalKM
162	Beaver Creek	5	18.9
167	Formigal	5	13
160	Glenshee	3	10.1
164	La Norma	3	8.4
155	La Parva	6	20
168	La Pinilla	3	10.8
161	Nevis Range	2	12.4
165	Saint-Lary-Soulan	2	9.3
166	Sierra Nevada	7	27
163	Tignes	4	10.8

Consulta 8 – Situación Lógica

Dada la cantidad de participantes en las olimpiadas se decidió habilitar una nueva estación de esquí llamada “Sur/Norte”. Para administrar se pensó en la federación que tiene asignada una sola estación junto a la federación que tiene la mayor cantidad de estaciones asignadas. Para determinar cuántos km esquiables tendrá se pensó en tomar el promedio de km totales que tiene toda la competencia. Las pistas aún no serán asignadas porque deben evaluar si hay algún recorte de las ya existentes.

Primero averiguaremos el promedio de km esquiables de todas las estaciones (que coinciden con los de todas las pistas) para luego poner los datos en la estación mediante una variable. Con otra variable, necesitaremos saber cual es el identificador de la última estación registrada y sumarle uno.

```
set @TotalKM = (
    select round(avg(TotalKM), 1)
    from Estacion);
select @TotalKM;
```

→

@TotalKM
12.2


```
set @NEstNuevo = (
    select max(NEst) +1
    from Estacion);
select @NEstNuevo;
```

→

@NEstNuevo
169

Ahora creamos la estación mediante un insert (los demás datos son inventados)

```
insert into Estacion values
(@NEstNuevo, "Sur / Norte", "Av. El Bosque 86, Las Condes", "+56957638114", @TotalKM);
```

NEst	Nombre	Direccion	Telefono	totalkm
150	Catedral Alta Patagonia	Alicia M. de Justo 1848	2944409000	16.3
151	Las Leñas	RP222	26274711000	14.3
152	Cerro Castor	Ruta Nacional 3	2901499301	3.9
153	El Colorado	Av. Kennedy 9070	78977534	5.6
154	Freire	Camino los Condores 1800	66484295	15.4
155	La Parva	Luis Carrera 1263	29642100	20
156	Vail	Vail Resorts Management Company	+19708455745	13.6
157	Aspen Highlands	P.O.Box 1248	+19709251220	2.1
158	Hurricane Ridge	PMB 218	+13605653131	13.9
159	Glencoe Mountain	Glencoe Mountain	+441855851226	5.8
160	Glenshee	Glenshee Ski Centre Cairnwell	+441339741320	10.1
161	Nevis Range	Nevis Range Torlundy	+441397705825	12.4
162	Beaver Creek	P.O. Box 7 81658	+19707544636	18.9
163	Tignes	73321 Tignes	+33479400440	10.8
164	La Norma	73500 La Norma France	+33479203146	8.4
165	Saint-Lary-Soulan	65170 Saint-Lary	+33562395081	9.3
166	Sierra Nevada	Plaza de Andalucia	958708090	27
167	Formigal	Urbanización Formigal 22640	+349744490000	13
168	La Pinilla	Cerezo de Arriba 40592	+34921551113	10.8
169	Sur / Norte	Av. El Bosque 86, Las Condes	+56957638114	12.2

Ahora para determinar quienes estarán a cargo de esa estación, hay que buscar la federación que administre una sola y la que administre más. Primero buscamos el máximo, y luego que federación tiene esa cantidad.

```

set @MaxF =(
select max(F)
  from (select count(NFed) as "F"
        from Administracion A
        inner join Estacion E on A.NEst = E.NEst
        group by A.NFed)A);
select @MaxF;

```

@MaxF
9

```

set @NFedMayor = (
select NFed
  from Administracion
  group by NFed
  having count(NFed) = @MaxF);
select @NFedMayor;

```

@NFedMayor
103

Y ahora la federación que administre una sola estación

```

set @NFedMenor = (
select NFed
  from Administracion
  group by NFed
  having count(NFed) = 1);
select @NFedMenor;

```

@NFedMenor
101

Ahora que tenemos todos los datos, simplemente hay que cargarlos a la tabla Administración

```

insert into Administracion values
(@NFedMenor,@NEstNuevo),
(@NFedMayor,@NEstNuevo);

```

NFed	NEst
100	150
100	151
100	152
100	153
100	154
100	155
101	155
101	169
102	156
102	157
102	158
102	159
103	160
103	161

103	162
103	163
103	164
103	165
103	166
103	167
103	168
103	169
104	160
104	161
104	162
104	163
104	164
104	165
105	160
105	161
105	162
105	166
105	167
105	168

Situación Lógica A

Debido a los decepcionantes resultados de la ultima prueba, los administradores del equipo que quedó en ultimo puesto han decidido cambiar su entrenador, y contratar al entrenador del equipo que resultó ganador, y de esta manera en un futuro conseguir mejores resultados

Primero buscaremos cual es la última prueba, teniendo en cuenta que sea la que comenzó más tarde

```
SET @UltNPrueba = (  
    select NPrueba  
    from Prueba  
    where Inicio =(select max(Inicio)  
                   from Prueba));  
select @UltNPrueba;
```

@UltNPrueba
310

Luego buscamos los equipos que participaron en esa prueba

```
select CodEqPrb  
from equipo_prueba  
where NPrueba = @UltPrueba;
```

CodEqPrb
11002
11006
11009
11013

Con los equipos identificados, ahora hace falta ver quien es el ganador y quien es el perdedor que despide a su entrenador, por eso ahora mostraremos las posiciones

```
select P.Posicion, P.CodTTEq, CodEqPrb  
from Posicion_Eq P  
inner join Tiempo_Tot_Eq TT on P.CodTTEq = TT.CodTTEq  
where CodEqPrb in (select CodEqPrb  
                   from equipo_prueba  
                   where NPrueba = @UltPrueba)  
order by Posicion;
```

Posicion	CodTTEq	CodEqPrb
1	25006	11006
2	25013	11013
3	25002	11002
4	25009	11009

Una vez encontrados, solo tenemos que buscar cual es la última posición y con eso sabremos que equipo perdió (el que ganó ya se sabe que quedó en 1^{ra} posición)

```
set @UltPosic =(  
select max(P.Posicion)  
from Posicion_Eq P  
inner join Tiempo_Tot_Eq TT on P.CodTTEq = TT.CodTTEq  
where CodEqPrb in (select CodEqPrb  
                   from equipo_prueba  
                   where NPrueba = @UltPrueba)  
order by Posicion); select @UltPosic;
```

@UltPosic
4

Ahora hay que identificar a esos equipos

```
set @NEqPerdedor = (
select E.NEq
  from Posicion_Eq P
    inner join Tiempo_Tot_Eq TT on P.CodTTEq = TT.CodTTEq
    inner join Equipo_Prueba EP on TT.CodEqPrb = EP.CodEqPrb
    inner join Participante_Eq PE on EP.NPartEq = PE.NPartEq
    inner join Equipo E on PE.NEq = E.NEq
  where EP.CodEqPrb in (select CodEqPrb
                        from equipo_prueba
                        where NPrueba = @UltPrueba)
    and Posicion = @UltPosic
); select @NEqPerdedor;
```



@NEqPerdedor
506

```
set @NEqGanador = (
select E.NEq
  from Posicion_Eq P
    inner join Tiempo_Tot_Eq TT on P.CodTTEq = TT.CodTTEq
    inner join Equipo_Prueba EP on TT.CodEqPrb = EP.CodEqPrb
    inner join Participante_Eq PE on EP.NPartEq = PE.NPartEq
    inner join Equipo E on PE.NEq = E.NEq
  where EP.CodEqPrb in (select CodEqPrb
                        from equipo_prueba
                        where NPrueba = @UltPrueba)
    and Posicion = 1
); select @NEqGanador;
```



@NEqGanador
504

Una vez que identificamos a los equipos, solo nos queda averiguar los entrenadores y luego hacer el update correspondiente

```
set @EntrPerdedor = (
  select CodEntr
    from Equipo
   where NEq = @NEqPerdedor);
select @EntrPerdedor;
```



@EntrPerdedor
65

```
set @EntrGanador = (
  select CodEntr
    from equipo
   where NEq = @NEqGanador);
select @EntrGanador;
```



@EntrGanador
63

Ahora con todos los datos hacemos el update de los equipos, y listo

```
update Equipo
  set CodEntr = @EntrGanador
  where CodEntr = @EntrPerdedor;
```

NEq	Nombre	CodEntr
500	Ushuaia Canopy	60
501	Rainbow7	61
502	Vaevictis	62
503	Heretics	69
504	Gen G	63
505	Santiago Wanderers	64
506	Rebirth	65
507	Infinity	66
508	Damwon	67
509	Ice Phoenix	68
510	Giants	63
511	Cloud9	69
512	Fnatic	70




NEq	Nombre	CodEntr
500	Ushuaia Canopy	60
501	Rainbow7	61
502	Vaevictis	62
503	Heretics	69
504	Gen G	63
505	Santiago Wanderers	64
506	Rebirth	63
507	Infinity	66
508	Damwon	67
509	Ice Phoenix	68
510	Giants	63
511	Cloud9	69
512	Fnatic	70

Situación lógica B

Debido que hay entrenadores que por un largo tiempo no lograron estar en un equipo, han estado en una situación económica bastante inestable y se les presentó la oportunidad de trabajar como contactos, laburando en la estación con la menor cantidad de contactos.

Primero hay que ver cuantos contactos hay en cada estación


```
select Nest, count(*) "CantContact"
      from contacto
      group by Nest;
```



Nest	CantContact
150	2
151	2
152	2
153	1
154	2
155	2
156	3
157	2
158	2
159	2
160	2
161	2
162	2
163	3
164	3
165	3
166	2
167	2
168	4

Y luego buscamos el mínimo


```
set @MinContact = (select min(CantContact)
                  from (select Nest, count(*) "CantContact"
                        from contacto
                        group by Nest)a);
select @MinContact;
```



@MinContact
1

Ahora identificamos la Estación


```
set @NEst = (
  select Nest
    from contacto
   group by Nest
  having count(*) = @MinContact);
select @NEst;
```



@NEst
153

Posteriormente debemos conseguir a los entrenadores que NO tienen equipo, así que utilizaremos como filtro los que ya están en un equipo.

```
select E.CodP
from entrenador E
where E.CodEntr not in (select EQ.CodEntr
                        from Equipo EQ);
```




CodP
50
56
57
58

En un principio utilizaríamos el operador IF para procesar esos cuatro códigos e insertarlos directamente, pero como el InnoDB no lo soporta, debemos hacerlo uno por uno...

Para cada uno guardaremos el CodP en una variable y luego lo borramos de entrenador

```
set @CodPers1 = (select E.CodP
                 from entrenador E
                 where E.CodEntr not in (select EQ.CodEntr
                                         from Equipo EQ)limit 1);
select @CodPers1;


delete from entrenador where
CodP = @CodPers1;
```



@CodPers1
50

```
set @CodPers2 = (select E.CodP
                 from entrenador E
                 where E.CodEntr not in (select EQ.CodEntr
                                         from Equipo EQ)limit 1);
select @CodPers2;

delete from entrenador where
CodP = @CodPers2;
```



@CodPers2
56

```
set @CodPers3 = (select E.CodP
                 from entrenador E
                 where E.CodEntr not in (select EQ.CodEntr
                                         from Equipo EQ)limit 1);
select @CodPers3;

delete from entrenador where
Codp = @CodPers3;
```



@CodPers3
57

```
set @CodPers4 = (select E.CodP
                 from entrenador E
                 where E.CodEntr not in (select EQ.CodEntr
                                         from Equipo EQ)limit 1);
select @CodPers4;

delete from entrenador where
CodP = @CodPers4;
```




@CodPers4
58

Y ahora solo queda insertarlos, como la tabla de contactos ya tiene su auto_increment, no hace falta saber quién es el último contacto, se suma uno automáticamente.

```
insert into contacto values
('','','@CodPers1,@Nest),
('','','@CodPers2,@Nest),
('','','@CodPers3,@Nest),
('','','@CodPers4,@Nest);
```

CodCont	CodP	NEst
200	1	150
201	2	150
202	3	151
203	4	151
204	5	152
205	6	152
206	7	153
207	8	154
208	9	154
209	10	155
210	11	155
211	12	156
212	13	156
213	14	156
214	15	157
215	16	157
216	17	158
217	18	158
218	19	159
219	20	159
220	21	160
221	22	160
222	23	161



223	24	161
224	25	162
225	26	162
226	27	163
227	28	163
228	29	163
229	30	164
230	31	164
231	32	164
232	33	165
233	34	165
234	35	165
235	36	166
236	37	166
237	38	167
238	39	167
239	40	168
240	41	168
241	43	168
242	44	168
244	50	153
245	56	153
246	57	153
247	58	153