

Trabajo Nociones de Arquitectura de la Información

Versión: 2021.04.25 16:00

RESPONSABLES

Nombre Completo – Documento de Identificación
1. Gladis Carmona - 43983972
2. Camila Arbeláez - 1036959719
3. Juan Pablo Botero - 98714459
REPO EN GITHUB: https://github.com/jpablo146unal/AGD_homework

*Realiza este trabajo considerando los datos que generan los sistemas transaccionales e información no estructurada de tu dominio (si trabajas por ejemplo para TCC tu dominio es la mensajería; también puedes explorar en la página <https://www.kaggle.com/datasets> o <https://arxiv.org/>). Considera tener acceso a esta información, de al menos 10 MB (puede ser uno o varios archivos de texto), y **tener al menos cuatro clases conceptuales. Este documento también debe almacenarse en el REPO. Plazo Máximo de Entrega 23 de Mayo, NO SE recibirá por correo electrónico, envío por <https://forms.gle/h7ty3yZykaUq5m7y6>***

1. COMPRENSIÓN DEL NEGOCIO

1.1. DESCRIPCIÓN DEL CONTEXTO DEL NEGOCIO.

Describe en máximo 250 palabras el contexto en el cual se generan los datos y cuál es el proceso que los genera.

Los Juegos Olímpicos (J.O.O.) son el mayor evento deportivo internacional multidisciplinario en el que participan atletas de distintos países del mundo y son considerados la principal competición deportiva del mundo con la participación de más de 200 naciones. Existen dos tipos: los Juegos Olímpicos de Verano y los Juegos Olímpicos de Invierno.

Sus orígenes se remontan a la antigüedad (776 a. C), sin embargo, la primera edición de la era moderna se llevó a cabo en Atenas el 1896. Desde entonces se han realizado cada 4 años en diversas ciudades del mundo, exceptuando las ediciones de 1916, 1940 y 1944 por motivo de la primera y segunda guerra mundial y postergada la de 2020 debido a la pandemia por COVID-19.

Los datos sobre las olimpiadas son generados y mantenidos por el Comité Olímpico Internacional, comprenden la información de más de un siglo de competencias, desde la edición de 1896 hasta 2016. Se encuentran almacenados en la base de datos "Olympics Athlete Events Analysis" disponible en www.kaggle.com.

1.2. IDENTIFICACIÓN DEL PROBLEMA:

Delimite en máximo 150 palabras la problemática, así como identificar los requisitos, supuestos, restricciones y beneficios de la solución de este.

Se requiere identificar los países más exitosos en las olimpiadas, al igual que los atletas más destacados y sus respectivas disciplinas, sin embargo, se desconoce el patrón de ganadores junto con sus atributos, tampoco se conoce la edad de todos los atletas, por tanto, las no registradas se excluirán de las consultas. Se parte de diferentes suposiciones como que los países latinos no son los principales ganadores en los juegos mientras los más desarrollados son los más destacados, cada juego tiene muchos eventos deportivos, un atleta puede participar en uno o más eventos, un atleta puede representar diferentes países y ganar máximo una medalla por competencia. Todos estos datos permitirían establecer un seguimiento a los patrones de los ganadores que sirva de insumo a los demás países para emprender un plan de mejora en sus entrenamientos y deportes.

1.3. DETERMINACIÓN DE OBJETIVOS:

Describe en máximo 150 palabras las metas a lograr al proponer una solución basada en un modelo de datos o de analítica (cómo y qué tipo de ventaja competitiva se ganará).

El correcto manejo y aprovechamiento de estos datos permitirá obtener las métricas por país (ganadores, no ganadores, disciplina, edad, género) con el fin de promover el intercambio de atletas y la implementación de modelos de entrenamiento más exitosos, así como determinar cuál es el país, la disciplina y el atleta que más han ganado medallas en la historia de los juegos olímpicos. Por otra parte, determinar si el promedio de la edad de los ganadores ha cambiado en el tiempo permitirá establecer con certeza el segmento de la población ideal para destinar más recursos en sus entrenamientos y lograr a futuro un mayor número de competencias ganadas.

1.4. EVALUACIÓN DE LA SITUACIÓN ACTUAL:

Describa en máximo 150 palabras el estado actual antes de implementar la solución de analítica, a fin de tener un punto de comparación que permita medir el grado de éxito de la solución.

El Comité Olímpico Colombiano no cuenta con una base de datos propia que le permita conocer los resultados de interés para las diferentes federaciones que lo conforman y su herramienta de trabajo es Microsoft Excel. La base de datos disponible cuenta con registros faltantes en campos como la edad, peso y altura, que tienen celdas con valores “NA” y el género para algunos atletas varia incluso en el mismo año de competencia, lo cual dificulta el manejo de los datos y la posibilidad de obtener las métricas necesarias para establecer los objetivos planteados. Estos campos serán reemplazados por NULL y no se tendrán en cuenta en las consultas y así garantizar que se trabaja con datos y campos que cumplen con las características adecuadas.

2. COMPRENSIÓN DE LOS DATOS

2.1. RECOLECCIÓN DE DATOS

Describa en máximo 150 palabras los datos a utilizar identificando las fuentes, las técnicas empleadas en su recolección, los problemas encontrados en su obtención y la forma cómo se resolvieron los mismos. Además, adjunte los datos (archivos de texto, etc.) agréguelos en el github (**REPO EN GITHUB**) en un solo archivo, por favor comprímalo(s). Llame el archivo T1.2.1.Datos.zip

Los datos fueron extraídos de <https://www.kaggle.com/samruddhim/olympics-athlete-events-analysis> y cada registro es obtenido de las diferentes competencias llevadas a cabo en los Juegos Olímpicos, datos creados y administrados por el Comité Olímpico Internacional. Los principales problemas encontrados fueron los datos faltantes en las diferentes variables cuantitativas tales como edad, peso y altura, atletas que registraban competencias iguales en el mismo año del evento, además de la comprensión en la disposición de los datos; estos inconvenientes fueron resueltos tras el análisis exhaustivo de los mismos, así como la exclusión de datos atípicos como NULL en campos numéricos y la eliminación de registros duplicados.

2.2. DESCRIPCIÓN DE DATOS (DICCIONARIO):

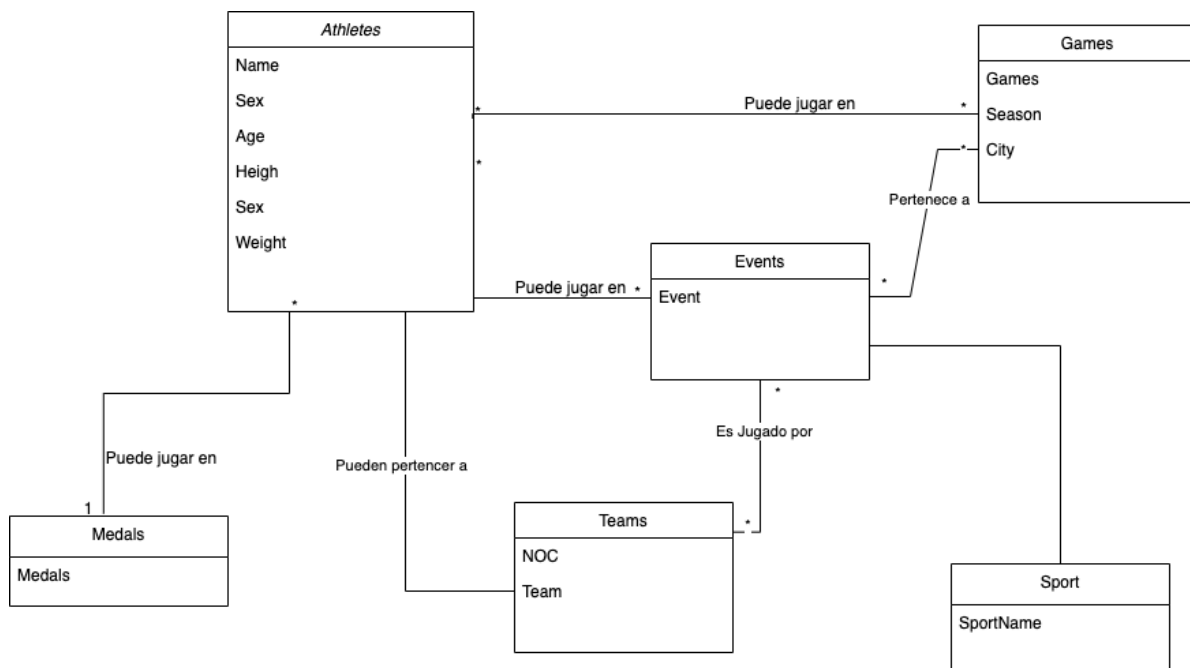
Diligencia la siguiente tabla, puede agregar otra columna si lo considera necesario.

Nombre del atributo / variable	Formato o Tipo de Dato	Descripción
ID_Athlete	Integer	Dato numérico asignado a cada atleta
Name	String	Nombre de cada atleta
Sex	String	Género del atleta
Age	Integer	Edad en años del atleta para el año de competición
Height	Integer	Estatura en centímetros del atleta
Weight	Integer	Peso en kilogramos del atleta
Team	String	Nombre del país que representa el atleta

NOC	String	Nombre del país asignado por el comité olímpico
Games	String	Edición de los juegos Olímpicos basado en la temporada
Year	Integer	Año de competición
Season	String	Temporada en la que se desarrolló la competencia
City	String	Ciudad en que se llevaron a cabo los juegos
Sport	String	Nombre del deporte
Event	String	Nombre de la disciplina olímpica
Medal	String	Tipo de medalla ganada

2.3. MODELO DEL DOMINIO

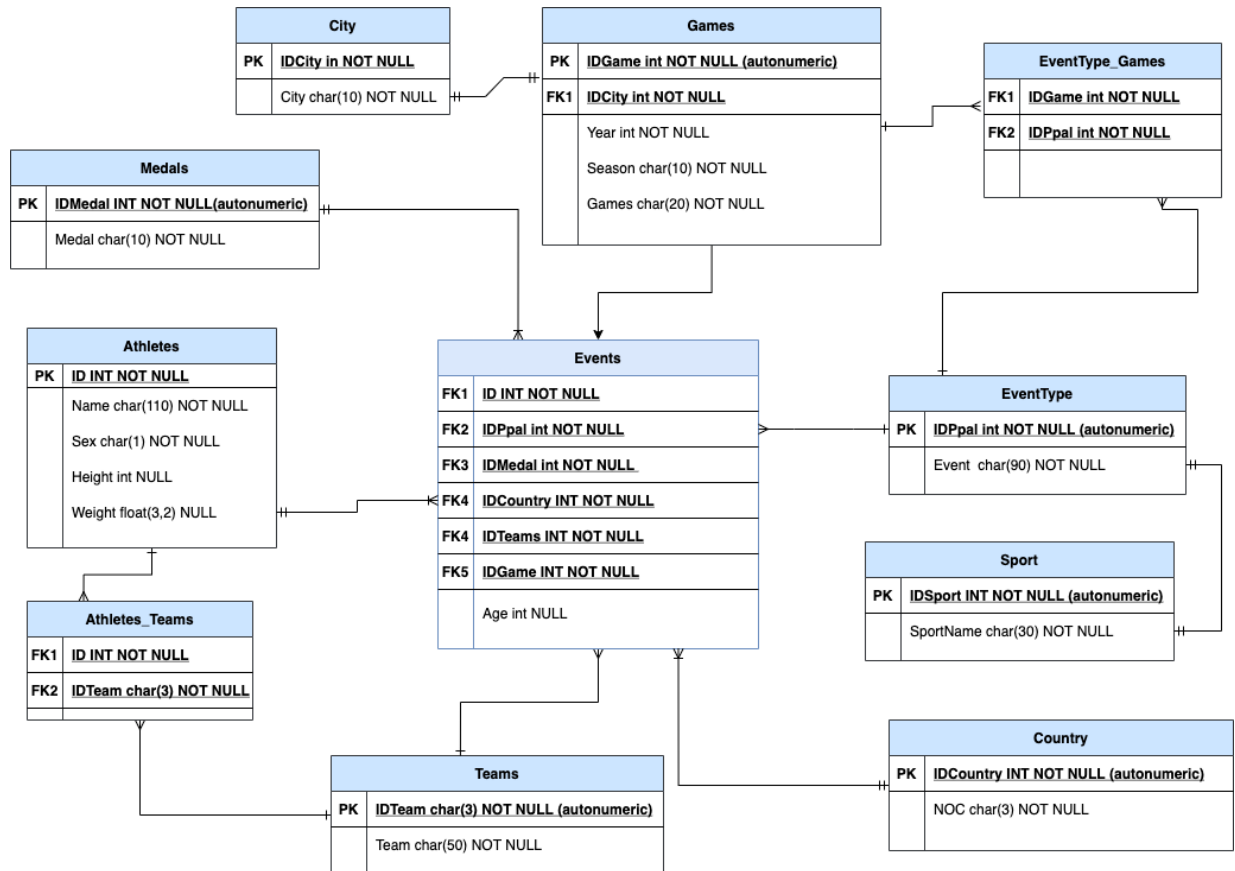
Observación: Incluya el gráfico del modelo del dominio que representa la estructura de datos de su problema.



3. MODELO ENTIDAD-RELACIÓN

3.1. TOMA DE PANTALLA DEL MODELO E-R

Observación: lo que se pide, puede usar <https://draw.io> o Microsoft Visio® y modele usando la notación de Barker.



3.2. SENTENCIA O CONSULTA DE CREACIÓN DEL TABLA(S)

Observación: Escriba el código en el Sistema de Gestión de Bases de Datos Relacionales de su elección (se recomienda SQLite por simplicidad, mediante <https://sqlitebrowser.org/>) para crear las tablas que corresponda con su conjunto de datos específico. Almacene en el repositorio **(REPO EN GITHUB)** el script con el nombre de T1.3.2.Creacion_Tablas.sql

3.3. SENTENCIAS PARA INSERTAR DATOS

Observación: Escriba el código para insertar los datos en cada una de las tablas creadas. Almacene en el repositorio **(REPO EN GITHUB)** el script con el nombre de T1.3.3.Insertar_Datos.sql

3.4. SENTENCIA DE CONSULTA

Observación: realice la exploración básica de los datos, conteos totales y por categorías, máximos, promedio y mínimos. Es decir, aplique estadística descriptiva con el fin de conocer las propiedades de los datos y entenderlos lo mejor posible. Use solamente sentencias SQL. Anexe las tomas de pantalla donde evidencie la sentencia SQL y su correspondiente ejecución. Además, **Almacene en el repositorio (REPO EN GITHUB) el script con el nombre de T1.3.4.Consultar_Datos.sql**

LIMPIEZA DE DATOS

Para tener datos consistentes, borramos los duplicados por evento que había en los datos iniciales:

ID	Name	Sex	Age	Height	Weight	Team	NOG	Games	Year	Season	City	Sport	Event	Medal
2777	Hermann Reinhard Alker	M	43	NA	NA	Germany	GER	1928 Summer	1928	Summer	Amsterdam	Art Competitions	Art Competitions Mixed Architecture, Designs For Town Planning	NA
2777	Hermann Reinhard Alker	M	43	NA	NA	Germany	GER	1928 Summer	1928	Summer	Amsterdam	Art Competitions	Art Competitions Mixed Architecture, Designs For Town Planning	NA
2777	Hermann Reinhard Alker	M	43	NA	NA	Germany	GER	1928 Summer	1928	Summer	Amsterdam	Art Competitions	Art Competitions Mixed Architecture, Architectural Designs	NA
2777	Hermann Reinhard Alker	M	43	NA	NA	Germany	GER	1928 Summer	1928	Summer	Amsterdam	Art Competitions	Art Competitions Mixed Architecture, Architectural Designs	NA
2777	Hermann Reinhard Alker	M	47	NA	NA	Germany	GER	1932 Summer	1932	Summer	Los Angeles	Art Competitions	Art Competitions Mixed Architecture, Designs For Town Planning	NA
2777	Hermann Reinhard Alker	M	51	NA	NA	Germany	GER	1936 Summer	1936	Summer	Berlin	Art Competitions	Art Competitions Mixed Architecture, Unknown Event	NA
2777	Hermann Reinhard Alker	M	51	NA	NA	Germany	GER	1936 Summer	1936	Summer	Berlin	Art Competitions	Art Competitions Mixed Architecture, Unknown Event	NA

Quedando finalmente en la tabla Events los datos sin repeticiones:

```
8 select *
9 from Events e inner join eventType et
10 on e.IDPpal = et.IDPpal
11 where
12 id = 2777
13 -- borramos los repetidos de los datos porque teníamos datos que venía dobles desde
```

ID	IDPpal	IDMedal	IDCountry	IDTeams	IDGame	Age	IDPpal	Event
1	2777	44	1	30	3	14	51	Art Competitions Mixed Architecture, Unknown Event
2	2777	338	1	30	3	10	43	Art Competitions Mixed Architecture, Designs For Town Planning
3	2777	338	1	30	3	12	47	Art Competitions Mixed Architecture, Designs For Town Planning
4	2777	339	1	30	3	10	43	Art Competitions Mixed Architecture, Architectural Designs

Execution finished without errors.
Result: 4 rows returned in 50ms
At line 8:
select *
from Events e inner join eventType et
on e.IDPpal = et.IDPpal
where
id = 2777

```
1 select ID, IDPpal, IDMedal, IDCountry, IDTeams, IDGame, Age, count(0)
2 from Events
3 group by ID, IDPpal, IDMedal, IDCountry, IDTeams, IDGame, Age
4 having count(0) > 1
5 order by count(0)
```

ID	IDPpal	IDMedal	IDCountry	IDTeams	IDGame	Age	count(0)
----	--------	---------	-----------	---------	--------	-----	----------

Execution finished without errors.
Result: 0 rows returned in 3271ms
At line 1:
select ID, IDPpal, IDMedal, IDCountry, IDTeams, IDGame, Age, count(0)
from Events
group by ID, IDPpal, IDMedal, IDCountry, IDTeams, IDGame, Age
having count(0) > 1
order by count(0)

PROMEDIOS

Hay 145 países que, en los datos alguno de sus atletas no registran Edad:

SQL 1	
1	select COUNT(distinct c.NOC) as CountryName
2	from Events e inner join Country c
3	on e.IDCountry = c.IDCountry
4	where Age = 'NULL'
	CountryName
1	145

En el promedio de edades por país, el país que tiene los atletas más jóvenes es: MHL, con 20 años, el país que en promedio tiene los atletas más adultos es: MON, con 30 años

SQL 1	
1	select round(avg(Age)) as AvgAge, c.NOC as CountryName
2	from Events e inner join Country c
3	on e.IDCountry = c.IDCountry
4	where Age <> 'NULL'
5	group by c.NOC
6	order by round(avg(Age)) DESC
7	limit 1
8	
	AvgAge CountryName
1	30.0 MON

SQL 1	
1	select round(avg(Age)) as AvgAge, c.NOC as CountryName
2	from Events e inner join Country c
3	on e.IDCountry = c.IDCountry
4	where Age <> 'NULL'
5	group by c.NOC
6	order by round(avg(Age)) ASC
7	limit 1
8	
	AvgAge CountryName
1	20.0 MHL

Se excluyeron de los análisis aquellos registros con edad Nula.

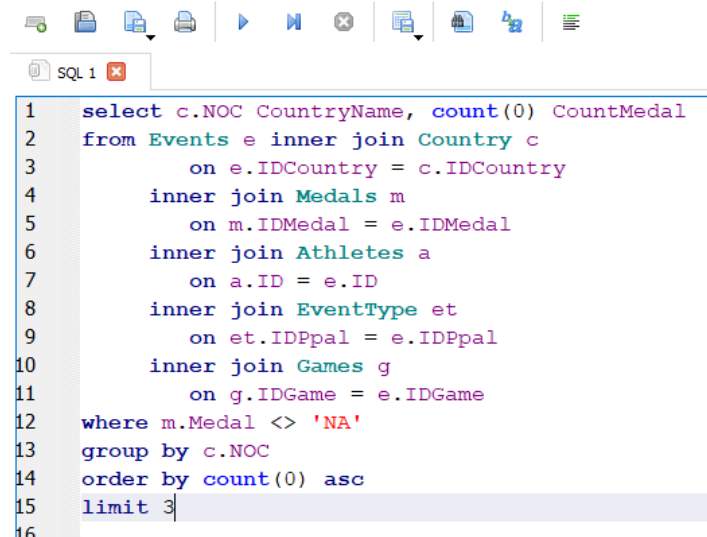
CONTEOS

Los 3 países que más han ganado medallas en la historia son: USA, URS, GER:

```
1 select c.NOC CountryName, count(0) CountMedal
2 from Events e inner join Country c
3     on e.IDCountry = c.IDCountry
4     inner join Medals m
5     on m.IDMedal = e.IDMedal
6     inner join Athletes a
7     on a.ID = e.ID
8     inner join EventType et
9     on et.IDPpal = e.IDPpal
10    inner join Games g
11    on g.IDGame = e.IDGame
12 where m.Medal <> 'NA'
13 group by c.NOC
14 order by count(0) desc
15 limit 3
16
```

	CountryName	CountMedal
1	USA	5637
2	URS	2503
3	GER	2165

Los 3 países que menos medallas han ganado son: AHO,BAR,VER



The screenshot shows a SQL IDE window titled 'SQL 1'. The query editor contains the following SQL code:

```
1 select c.NOC CountryName, count(0) CountMedal
2 from Events e inner join Country c
3     on e.IDCountry = c.IDCountry
4     inner join Medals m
5     on m.IDMedal = e.IDMedal
6     inner join Athletes a
7     on a.ID = e.ID
8     inner join EventType et
9     on et.IDPpal = e.IDPpal
10    inner join Games g
11    on g.IDGame = e.IDGame
12 where m.Medal <> 'NA'
13 group by c.NOC
14 order by count(0) asc
15 limit 3
16
```

Below the query editor, the results are displayed in a table:

	CountryName	CountMedal
1	AHO	1
2	BAR	1
3	BER	1

Los 5 eventos que más han ganado medallas son:

```

1 select et.Event, count(0) CountMedal
2 from Events e inner join Country c
3     on e.IDCountry = c.IDCountry
4     inner join Medals m
5     on m.IDMedal = e.IDMedal
6     inner join Athletes a
7     on a.ID = e.ID
8     inner join EventType et
9     on et.IDPal = e.IDPal
0     inner join Games g
1     on g.IDGame = e.IDGame
2 where m.Medal <> 'NA'
3 group by et.Event
4 order by count(0) desc
5 limit 5

```

	Event	CountMedal
1	Football Men's Football	1269
2	Ice Hockey Men's Ice Hockey	1230
3	Hockey Men's Hockey	1050
4	Water Polo Men's Water Polo	866
5	Rowing Men's Coxed Eights	730

Los 5 Atletas que más han ganado medallas son:

```

1  select a.Name, count(0) CountMedal
2  from Events e inner join Country c
3      on e.IDCountry = c.IDCountry
4      inner join Medals m
5      on m.IDMedal = e.IDMedal
6      inner join Athletes a
7      on a.ID = e.ID
8      inner join EventType et
9      on et.IDPpal = e.IDPpal
10     inner join Games g
11     on g.IDGame = e.IDGame
12 where m.Medal <> 'NA'
13 group by a.Name
14 order by count(0) desc
15 limit 5
16

```

	Name	CountMedal
1	Michael Fred Phelps, II	28
2	Larysa Semenivna Latynina (Diriy-)	18
3	Nikolay Yefimovich Andrianov	15
4	Takashi Ono	13
5	Ole Einar Bjrndalen	13

El promedio de edades por año oscila entre 24 y 26 en promedio

SQL 1		
1	Select round(avg(age)) AVGAge, g.Year	
2	from Events e inner join Games g	
3	on e.IDGame = g.IDGame	
4	where e.age <> 'NULL'	
5	group by g.Year	
	AVGAge	Year
1	24.0	1896
2	29.0	1900
3	27.0	1904
4	27.0	1906
5	27.0	1908
6	28.0	1912
7	29.0	1920
8	28.0	1924
9	28.0	1928
10	30.0	1932
11	27.0	1936
12	28.0	1948
13	26.0	1952
14	26.0	1956
15	25.0	1960
16	25.0	1964
17	24.0	1968
18	24.0	1972
19	24.0	1976
20	24.0	1980

21	24.0	1984
22	24.0	1988
23	24.0	1992
24	24.0	1994
25	25.0	1996
26	25.0	1998
27	25.0	2000
28	26.0	2002
29	26.0	2004
30	26.0	2006
31	26.0	2008
32	26.0	2010
33	26.0	2012
34	26.0	2014
35	26.0	2016

4. MONGODB

4.1. SENTENCIA O CONSULTA DE CREACIÓN DEL DOCUMENTO(S)

Observación: Escriba el código en MongoDB para crear al menos 20 documentos que correspondan a su conjunto de datos específico. Almacene en el repositorio **(REPO EN GITHUB)** el script con el nombre de T1.4.1.Creacion_Documentos.sql

4.2. SENTENCIA DE CONSULTA

Observación: Realice la exploración básica de los datos, conteos totales y por categorías, máximos, promedio y mínimos. Es decir, aplique estadística descriptiva con el fin de conocer las propiedades de los datos y entenderlos lo mejor posible. Use solamente sentencias mongo. Anexe las tomas de pantalla donde evidencie la sentencia mongo y su correspondiente ejecución. Además, Almacene en el repositorio **(REPO EN GITHUB)** el script con el nombre de T1.4.2.Consultar_Datos.sql

```

MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count()
20
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { name: { $exists: true } } )
20
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { Sex: { $exists: true } } )
20
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { Sex: "F" } )
7
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { Sex: "M" } )
13
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { Medal: { $ne: null } } )
11
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.find({Medal:"Gold"}).count()
5
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("name",{Medal:"Gold"})
[
  "Edgar Lindenau Aabye",
  "Jo Qesem Ayela Aleh",
  "Noor Alam",
  "Rebecca 'Becky' Adlington",
  "William Accambray"
]

```

```

MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.find({Medal:null}).count()
9
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.find({Medal:"Gold", Sex: "F"}).count()
2
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.find({Medal:"Gold", Sex: "M"}).count()
3
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { Age: { $ne: null } } )
20
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, promAge: {$avg: "$Age"} } }])
{ "_id" : null, "promAge" : 24.95 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, max: {$max: "$Age"} } }])
{ "_id" : null, "max" : 34 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, min: {$min: "$Age"} } }])
{ "_id" : null, "min" : 16 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", promAge: {$avg: "$Age"} } }])
{ "_id" : "M", "promAge" : 26.307692307692307 }
{ "_id" : "F", "promAge" : 22.428571428571427 }

```

```

MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", min: {$min: "$Height"} } }])
{ "_id" : "F", "min" : 150 }
{ "_id" : "M", "min" : 170 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", prom: {$avg: "$Height"} } }])
{ "_id" : "F", "prom" : 168.28571428571428 }
{ "_id" : "M", "prom" : 184 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { Weight: { $ne: null } } )
18
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, promWeight: {$avg: "$Weight"} } }])
{ "_id" : null, "promWeight" : 72.61111111111111 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, max: {$max: "$Weight"} } }])
{ "_id" : null, "max" : 104 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, min: {$min: "$Weight"} } }])
{ "_id" : null, "min" : 40 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", prom: {$avg: "$Weight"} } }])
{ "_id" : "M", "prom" : 81.9090909090909 }
{ "_id" : "F", "prom" : 58 }

```

```

MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", max: {$max:"$Age"} } }])
{ "_id" : "M", "max" : 34 }
{ "_id" : "F", "max" : 31 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", min: {$min:"$Age"} } }])
{ "_id" : "F", "min" : 16 }
{ "_id" : "M", "min" : 20 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.count( { Height: { $ne: null } } )
18
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, promHeight: {$avg:"$Height"} } }])
{ "_id" : null, "promHeight" : 177.88888888888889 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, max: {$max:"$Height"} } }])
{ "_id" : null, "max" : 198 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:null, min: {$min:"$Height"} } }])
{ "_id" : null, "min" : 150 }
MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", max: {$max:"$Height"} } }])
{ "_id" : "M", "max" : 198 }
{ "_id" : "F", "max" : 179 }

```

```

MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct( "NOC" )
[
  "CHN",
  "COL",
  "DEN",
  "EGY",
  "ESP",
  "FRA",
  "FRG",
  "GBR",
  "GER",
  "GHA",
  "HUN",
  "ITA",
  "NZL",
  "PAK",
  "PUR",
  "ROU",
  "SUI",
  "UKR"
]

```

```
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct( "Year" )
[
  "1900",
  "1952",
  "1956",
  "1960",
  "1964",
  "1972",
  "1984",
  "1992",
  "2002",
  "2004",
  "2008",
  "2012",
  "2016"
]
```

```
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("NOC",{Medal:"Gold"})
[ "DEN", "FRA", "GBR", "NZL", "PAK" ]
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("Team",{Medal:"Gold"})
[ "Denmark/Sweden", "France", "Great Britain", "New Zealand", "Pakistan" ]
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("NOC",{Medal:{ $ne: null }})
[
  "DEN",
  "ESP",
  "FRA",
  "FRG",
  "GBR",
  "GER",
  "NZL",
  "PAK",
  "ROU",
  "SUI"
]
```

```
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", max: {$max:"$Weight"} } }])
{ "_id" : "F", "max" : 72 }
{ "_id" : "M", "max" : 104 }
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.aggregate([{$group: {_id:"$Sex", min: {$min:"$Weight"} } }])
{ "_id" : "F", "min" : 40 }
{ "_id" : "M", "min" : 72 }
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct( "City" )
[
  "Athina",
  "Barcelona",
  "Beijing",
  "Helsinki",
  "Innsbruck",
  "London",
  "Los Angeles",
  "Melbourne",
  "Munich",
  "Paris",
  "Rio de Janeiro",
  "Roma",
  "Salt Lake City",
  "Tokyo"
]
```

```
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("Age",{Medal:{ $ne: null }})
[ 16, 19, 20, 23, 24, 26, 29, 30, 31, 34 ]
```

```

[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("Sport",{Medal:{ $ne: null }}) ]
[
  "Basketball",
  "Cross Country Skiing",
  "Cycling",
  "Equestrianism",
  "Gymnastics",
  "Handball",
  "Hockey",
  "Sailing",
  "Swimming",
  "Tug-Of-War"
]
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("Event",{Medal:{ $ne: null }}) ]
[
  "Basketball Men's Basketball",
  "Cross Country Skiing Women's 4 x 5 kilometres Relay",
  "Cycling Men's Team Pursuit, 4,000 metres",
  "Equestrianism Mixed Jumping, Team",
  "Gymnastics Women's Team All-Around",
  "Handball Men's Handball",
  "Hockey Men's Hockey",
  "Sailing Women's Two Person Dinghy",
  "Swimming Women's 400 metres Freestyle",
  "Tug-Of-War Men's Tug-Of-War"
]

[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("NOC",{Medal:"Gold"}) ]
[ "DEN", "FRA", "GBR", "NZL", "PAK" ]
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("Team",{Medal:"Gold"}) ]
[ "Denmark/Sweden", "France", "Great Britain", "New Zealand", "Pakistan" ]
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.find({Medal:"Silver"}).count() ]
1
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("name",{Medal:"Silver"}) ]
[ "Noor Alam" ]
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.find({Medal:"Bronze"}).count() ]
5
[MongoDB Enterprise atlas-11mihi-shard-0:PRIMARY> db.Olympics.distinct("name",{Medal:"Bronze"}) ]
[
  "Alejandro 'lex' Abrines Redondo",
  "Andreea Roxana Acatrinei",
  "Brigitte Albrecht-Loretan",
  "Christian Ahlmann",
  "Reinhard Alber"
]

```

5. ANÁLISIS DE LECTURA

Observación: Considerando el artículo: “The Definitive Guide to Graph Databases for the RDBMS Developer” de Neo4J. Compartido en las carpeta de lecturas recomendadas. Analice y responda cada pregunta en máximo 150 palabras:

1. ¿Cuáles son las limitaciones, que se pueden inferir de la lectura, para migrar los conjuntos de datos relacionales a NoSQL?

Los seres humanos somos por naturaleza resistentes a los cambios. La limitación más importante corresponde a concebir y aceptar una nueva forma de pensar. NoSQL, en este caso Graph databases, implica la ruptura de paradigmas sobre el modelado y almacenamiento de datos y hasta de la comprensión de las necesidades actuales.

También existen limitaciones técnicas. Algunas bases NoSQL son nuevas y no están completamente maduras; algunas son inestables, inseguras y no son compatibles con ACID.

2. ¿Cuáles limitaciones adicionales que se deben considerar, a parte de las mencionadas en el artículo?

Otras consideraciones de las bases NoSQL son: las nuevas bases de datos utilizan sus propias características en el lenguaje de consulta y no son 100% compatibles con SQL. El soporte a problemas con los queries de trabajo en una base de datos NoSQL es más complicado.

No todas las bases de datos NoSQL contemplan la atomicidad de las instrucciones y la integridad de los datos.

Falta de estandarización. Hay muchas bases de datos NoSQL y aún no hay un estándar como sí lo hay en las bases de datos relacionales.

Herramientas GUI: la mayoría de las bases de datos NoSQL no contienen una interfaz gráfica. Requiere conocimiento especial para poder ejecutar algunas de ellas.

3. ¿Cuáles son las razones (criterios) que se deben considerar para migrar un conjunto de datos relacionados a NoSQL?

La necesidad de manipular datos estrechamente conectados, cada vez más frecuentes y voluminosos en nuestra época, hace que las bases relacionales sean inapropiadas en algunos casos, debido a que no están explícitamente diseñadas para almacenar relaciones entre datos. Existen 5 signos que pueden indicar que el modelo relacional está bajo tensión: un número considerable de JOINS, numerosos Self-JOINS, cambios frecuentes de esquema, lentitud de los queries y pre-cómputo de resultados.

Las bases NoSQL ofrecen ventajas de escalabilidad, mantienen el desempeño de los queries a medida que aumentan los datos, aceleran los ciclos de desarrollo y la capacidad de respuesta a nuevos requerimientos.