1. Describe los motivos por los que se utilizan *pool* de conexiones.

Un pool de conexiones es un conjunto limitado de conexiones a una base de datos, que es manejado por un servidor de aplicaciones de forma tal, que dichas conexiones pueden ser reutilizadas por los diferentes usuarios. Este pool es administrado por un servidor de aplicaciones que va asignando las conexiones a medida que los clientes van solicitando consultas o actualizaciones de datos.

2. Enumera los distintos tipos de `rowset`.

- JdbcRowSet
- CachedRowSet
- JoinRowSet
- FilteredRowSet
- WebRowSet

3. Según la arquitectura vista en clase *SL_TS_TDG* enumera las distintas capas que la componen y explica la finalidad de cada una.

- Service Layer: Define los límites de la aplicación con una capa de servicios que establece el conjunto de operaciones disponible. Desacopla presentación de negocio y además ayuda en el control de transacciones y en la seguridad.
- Transaction Script: Es un patrón para organizar la lógica de negocio o dominio. Utiliza procedimientos de forma que cada uno se encarga de procesar una petición desde presentación. Están separados de presentación y persistencia, por eso están en la capa de negocio. Una clase por uso.
- Table Data GateWay: Gestiona por completo el acceso a los datos (persistencia) de una entidad. Hace de pasarela. Puede ser Row o Table. Hay una clase por cada entidad o tabla. "AveriasGatewayImpl". En las clases de negocio (TS) "AddMechanic" etc. se invoca a las clases de persistencia mediante la factoría "PersistenceFactory" que nos dará la Gateway que necesitamos

4. Enumera y explica los distintos niveles de aislamiento.

- Read uncommited: Permite los tres tipos de violaciones de aislamiento. MySQL
- Read Committed: evita lectura sucia. Lo tiene Oracle y PostgreSQL y MySQL
- Repeatable read: Evita lectura sucia y no repetible . Lo tiene MySQL
- Serializable: Evita Todas las violaciones de aislamiento. Esta en Oracle y PostgreSQL y MySQL

5. Según la arquitectura vista en clase *SL_TS_TDG* dibuja el diagrama de secuencia de la operación "añadir un contrato".

6. ¿En qué estado están los objetos devueltos por una consulta JQL? ¿y el que devuelve el método `find`?

En estado persistent.

Una consulta JQL devuelve una lista de resultados de la clase que pedimos en la consulta con getResultList () o bien un solo resultado con getSingleResult().

El método find() Buscar por clave primaria, usando las propiedades especificadas. Búsqueda de una entidad de la clase especificada y la clave primaria. Si la instancia de la entidad está contenida en el contexto de persistencia, se devuelve desde allí.

7. Suponiendo configuración de mapeo por defecto, ¿funcionará el código mostrado a continuación? En caso negativo, explica el por qué.

Capa de presentación

```
AdminService as = ServiceFactory.adminServices();
for (Vehiculo v: c.getVehiculos()){
     Printer.print(v);
}
…
```

Capa de servicio

```
class AdminServicesImpl{
     …
     Cliente findClienteById(Long id){
          EntityManager em = getEntityManagerAndOpenTrx();
          try{
               return em.find(Cliente.class, id);
          } finally {
               closeTrxEntityManager();
          }
     }
…
```

No funcionara porque hemos cargado el objeto c pero no los vehiculos que tiene porque no estamos en el context de persistencia. No hemos cargado el grafo en findClienteById.

8. Sobre *CarWorkShop*, ¿es correcta esta consulta JQL? En caso negativo explica por qué no.

```
select f.averias.vehiculo.cliente
from Facturas f
where f.fecha = '12/12/2014'
```

No es correcta por que El final del camino NO puede ser multivaluado. Solo se permiten en caminos (path) que pasen a través de asociaciones manyto-one o one-to-one.

9. Tenemos una aplicación desarrollada con mapeador JPA Hibernate funcionando contra una base de datos MySQL. Por alguna razón hay que cambiar de servidor de base de datos, ahora usaremos una Oracle. ¿Cómo lo resolvemos?

Modficando los datos de persistence.xml

10. Describe brevemente las clases (o interfaces) principales que ofrece el API JPA: cuáles son, que misión tienen y como se interrelacionan.

- EntitymanagerFactory: Con el creo el entityManager. Se obtiene mediante Persistence.CreateEntityManagerFactory()
- EntityManager: con el obtengo la transacción que voy a utilizar.
- EntityTransaction: La transacción obtenida mediante manager.getTransaction();

11. Añadiendo anotaciones de mapeo, ¿cómo se vinculan los dos extremos de una asociación bidireccional? ¿Qué pasa si no se hace?

Sera con la etiqueta @OneToMany en el extremo uno de la relación (donde el set) y la etiqueta @ManyToOne en el otro extremo de la relación. En la etiqueta @OneToMany debemos indicar (mappedBy="nombre atributo en la manytoone")

Otra bidireccional puede ser @ManyToMany en los dos extremos de la relación.Tendremos dos set, y pondremos el mappedBy en uno de ellos.

Si no se indica mappedBy se interpreta como dos asociaciones unidireccionales separadas.

12. ¿Qué devuelve esta expresión?

```
let $i := (1, 2, 3, 4)
return <sal>{$i}</sal>
```

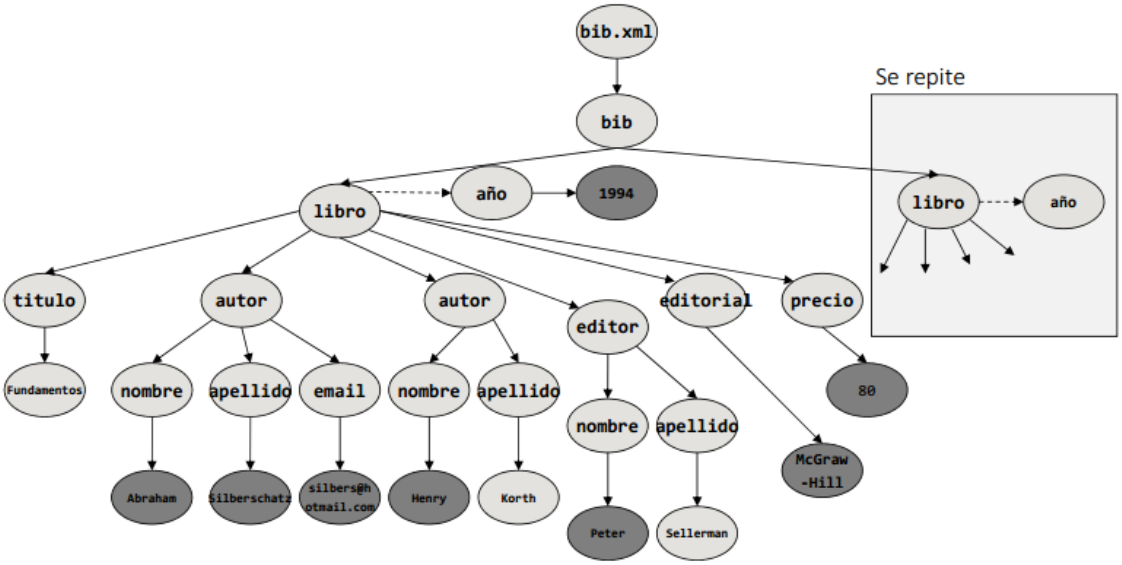
<sal>1, 2, 3, 4</sal>

13. ¿Qué devuelve esta expresión?

```
let $i in (1, 2, 3, 4)
for $j in $i
return <sal>{($i, $j)}</sal>
```

<sal>1 2 3 4 1</sal>
<sal>1 2 3 4 2</sal>
<sal>1 2 3 4 3</sal>
<sal>1 2 3 4 4</sal>

Para las preguntas 14, 15 y 16 considérese el siguiente grafo.



14. Listar un título de cada libro junto con el número de autores.

```
for $a in doc("bib.xml")//libro
let $b := count($a/autor)
return
   <titulo> {$a/titulo, $b}</titulo>
```

15. listar por orden alfabético los nombres de todos los autores y editoriales.

```
for $a in doc("bib.xml")//nombre
order by $a ascending
return
   <nombre> {$a} </nombre>
```

16. mostrar los libros cuyo precio no supera la media.

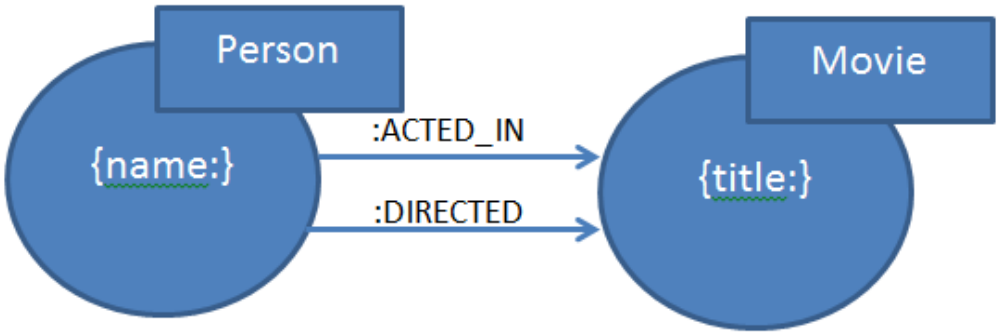
```
for $a in doc("bib.xml")//libro
let $b := $a
where $b/precio <= avg($a/precio)
return $b
```

17. ¿Además de las bases de datos en grafos, que otros tipos de bases de datos se incluyen habitualmente dentro del grupo de sistemas NoSQL?

18. Resumen brevemente al menos dos inconvenientes que se señalan habitualmente como típicas de las bases de datos NoSQL?

19. ¿Qué alternativas existen, para realizar la distribución de datos de una base de datos en un *cluster* de servidores?

20. ¿Qué es la consistencia eventual?



21. Con el grafo de películas, indica brevemente que devuelve esta consulta Cypher:

```
match (p:Person)-[:ACTED_IN*1…3]-(m:Movie)<-[ACTED_IN]-(a:Person)
where p.name='Kevin Bacon'
return m.title, a.name, count(*)
```

22. con el grafo anterior de películas, escribir una consulta en Cypher que devuelva las películas que tienen al menos dos directores, directores que a su vez han actuado en la película.

23. Explica brevemente el teorema CAP.

24. Describe brevemente que son y para qué sirven los *Quorum* de lectura y escritura en un *cluster* de servidores.