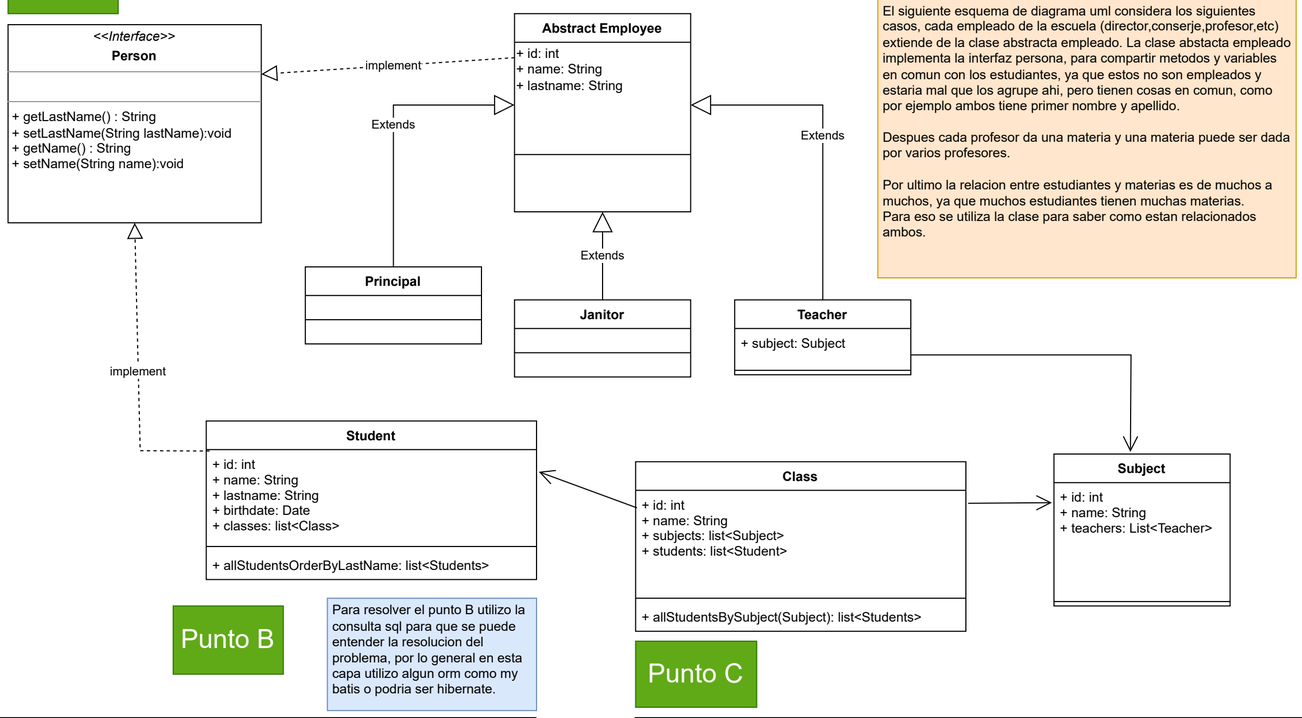
Punto A



Nota

El siguiente esquema de diagrama uml considera los siguientes casos, cada empleado de la escuela (director,conserje,profesor,etc) extiende de la clase abstracta empleado. La clase abstacta empleado implementa la interfaz persona, para compartir metodos y variables en comun con los estudiantes, ya que estos no son empleados y estaria mal que los agrupe ahi, pero tienen cosas en comun, como por ejemplo ambos tiene primer nombre y apellido.

Despues cada profesor da una materia y una materia puede ser dada por varios profesores.

Por ultimo la relacion entre estudiantes y materias es de muchos a muchos, ya que muchos estudiantes tienen muchas materias.
Para eso se utiliza la clase para saber como estan relacionados ambos.

Punto B

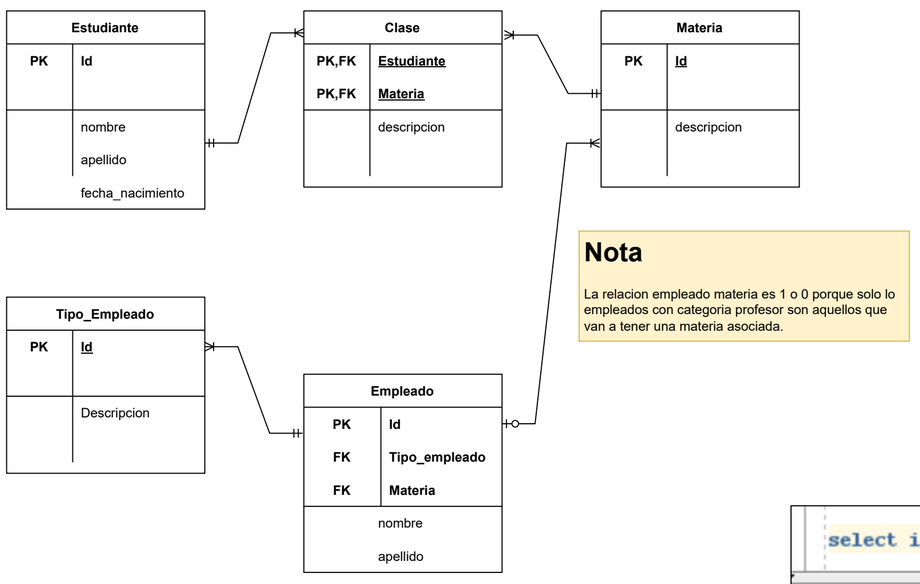
Para resolver el punto B utilizo la consulta sql para que se puede entender la resolucion del problema, por lo general en esta capa utilizo algun orm como my batis o podria ser hibernate.

```java
public List<Student> allStudentsOrderByLastName() {

    List<Student> studentList = new ArrayList<Student>();

    String query = "SELECT SUBSTR(last_name, 1, 1) AS firstLetter,last_name"
            + " FROM student GROUP BY SUBSTR(last_name, 1, 1),last_name";

    // Create the database connection & statement somehow...
    Statement stmt = createStatement();

    try {
        ResultSet results = stmt.executeQuery(query);

        while (results.next()) {
            Student student = new Student();
            student.setLastName(results.getString(2));
            studentList.add(student);
        }
    } finally {
        stmt.close();
    }

    return studentList;
}
```

Punto C

```java
public List<Student> allStudentsBySubject(Subject subject) {

    List<Student> studentList = new ArrayList<Student>();

    String query = "SELECT id,name,last_name"
            + " FROM student std,subject sub,class class WHERE"
            + " class.subject = sub.id AND class.student = std.id AND sub.id=" + subject.getId();

    // Create the database connection & statement somehow...
    Statement stmt = createStatement();

    try {
        ResultSet results = stmt.executeQuery(query);

        while (results.next()) {
            Student student = new Student();
            student.setLastName(results.getString(2));
            studentList.add(student);
        }
    } finally {
        stmt.close();
    }

    return studentList;
}
```

Punto D



Nota

La relacion empleado materia es 1 o 0 porque solo lo empleados con categoria profesor son aquellos que van a tener una materia asociada.

Punto E

Nota

La query del punto E es lenta porque utiliza dos inner join, estas operaciones son mas costosas para la base de datos, en el esquema que defini en el punto anterior al no tener una tabla por cada empleado y teniendo una sola para todos, solo se precisaria hacer un select para obtener el nombre del conserje.

Esto se podria realizar haciendo un select de la tabla empleado donde el tipo de empleado se corresponda con el id del conserje.

Esto tambien responda los pros y los contras del punto anterior. Donde en un ambiente en el cual las tablas se encuentran duplicadas el costo operativo es mas alto que en un entorno normalizado.

Punto F

Nota

En este punto no estoy tan seguro de la respuesta, pero creo que la solucion podria venir por utilizar una vista.

Punto H

Nota

La solucion para prescindir de una aplicacion java y solamente utilizar base de datos podrian ser la creacion procedimientos o funciones de base de datos.

La contra es que solo lo van a poder usar usuarios que con un nivel mas avanzado de expertise, osea que conozcan de base de datos.

Lo pro, podria ser que el desarrollo es mas veloz que una aplicacion java, porque tiene menos capas.

Punto G

Nota

En las siguientes dos imagenes se muestra, primero la lista total de estudiantes con su edad calculada usando su cumpleaños como referencia y en la segunda imagen se muestra el filtro de edad entre 19 y 21 años.

```sql
select id,name,lastname,birthdate,TIMESTAMPDIFF(YEAR, birthdate, CURDATE()) as age from students;
```

Salida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 5 en 0,189 segundos

	id	name	lastname	birthdate	age
1	1	ross	(null)	2001-11-23	20
2	2	monica	(null)	2002-10-28	19
3	3	chandler	(null)	2001-07-14	20
4	4	phoebe	(null)	2003-12-05	18
5	5	rachel	(null)	2004-04-09	18

```sql
select id,name,lastname,birthdate,TIMESTAMPDIFF(YEAR, birthdate, CURDATE()) as age from students
where TIMESTAMPDIFF(YEAR, birthdate, CURDATE()) between 19 and 21;
```

alida de Script | Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 3 en 0,171 segundos

	id	name	lastname	birthdate	age
1	1	ross	(null)	2001-11-23	20
2	2	monica	(null)	2002-10-28	19
3	3	chandler	(null)	2001-07-14	20