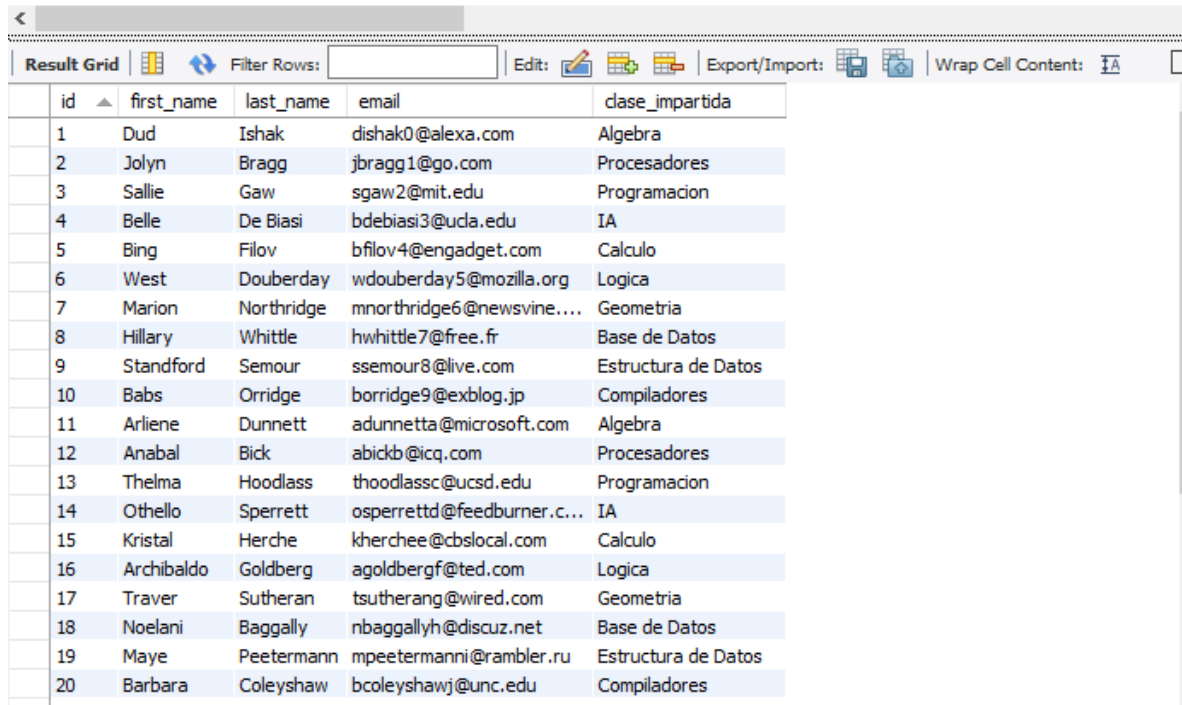


[illegible]

BD:

```
1 • SELECT * FROM web_student_tracker.profesor;
```



The screenshot shows a database query result grid with the following data:

id	first_name	last_name	email	clase_impartida
1	Dud	Ishak	dishak0@alexa.com	Algebra
2	Jolyn	Bragg	jbragg1@go.com	Procesadores
3	Sallie	Gaw	sgaw2@mit.edu	Programacion
4	Belle	De Biasi	bdebiasi3@ucd.edu	IA
5	Bing	Filov	bfilov4@engadget.com	Calculo
6	West	Douberday	wdouberday5@mozilla.org	Logica
7	Marion	Northridge	mnorthridge6@newsvine....	Geometria
8	Hillary	Whittle	hwhittle7@free.fr	Base de Datos
9	Standford	Semour	ssemour8@live.com	Estructura de Datos
10	Babs	Orridge	borridge9@exblog.jp	Compiladores
11	Arlene	Dunnett	adunnetta@microsoft.com	Algebra
12	Anabal	Bick	abickb@icq.com	Procesadores
13	Thelma	Hoodlass	thoodlassc@ucsd.edu	Programacion
14	Othello	Sperrett	osperrettd@feedburner.c...	IA
15	Kristal	Herche	kherchee@cbslocal.com	Calculo
16	Archibaldo	Goldberg	agoldbergf@ted.com	Logica
17	Traver	Sutheran	tsutherang@wired.com	Geometria
18	Noelani	Baggally	nbaggallyh@discuz.net	Base de Datos
19	Maye	Peetermann	mpeetermanni@rambler.ru	Estructura de Datos
20	Barbara	Coleyshaw	bcoleyshawj@unc.edu	Compiladores

3.- Explicación del diagrama de uso de Spring Batch:

R: Se adjunta Imagen con Explicación

4.- Explicación de los comandos de git en línea de comando:

R:

Pull Request: Una solicitud para que el código modificado pueda fusionarse con el código del repositorio sobre el que se ha trabajado.

Fork: Tiene la función de crear la copia de un determinado repositorio en la cuenta de usuario.

Este repositorio copiado será igual al repositorio desde el que se realiza el fork en Git.

A pesar de esto, cuando se cree la copia, cada repositorio se ubicará en espacios diferentes y tendrán la posibilidad de evolucionar de forma diferente, de acuerdo a las acciones del usuario en cada uno de estos recursos de Git.

Esto implica que los cambios que se lleven a cabo en el repositorio original no se

transmitirán de forma automática al repositorio copia ni tampoco sucederá que las modificaciones en el fork afectarán el proyecto original.

Rebase: Un comando que reproduce commits o confirmaciones de cambio una por una, en la parte superior de una determinada rama.

Git Rebase nombre_rama_que_tendra_rebase

Stach: El comando **git stash** almacena temporalmente (o guarda en un stash) los cambios que hayas efectuado en el código en el que estás trabajando para que puedas trabajar en otra cosa y, más tarde, regresar y volver a aplicar los cambios más tarde. Guardar los cambios en stashes resulta práctico si tienes que cambiar rápidamente de contexto y ponerte con otra cosa, pero estás en medio de un cambio en el código y no lo tienes todo listo para confirmar los cambios.

Puedes volver a aplicar los cambios de un stash mediante el comando **git stash pop**

Al hacer pop del stash, se eliminan los cambios de este y se vuelven a aplicar en el código en el que estás trabajando.

Otra opción es volver a aplicar los cambios en el código en el que estás trabajando y conservarlos en tu stash mediante el comando **git stash apply**

Clean: El comando **git clean** opera en archivos sin seguimiento. Los archivos sin seguimiento son archivos que se han creado en el directorio de trabajo del repositorio, pero que no se han añadido al índice de seguimiento del repositorio con git add.

La configuración global de Git obliga a usar la opción “force” con el comando git clean para que este se inicie. Es un mecanismo de seguridad importante. Cuando se ejecuta el comando git clean, no se puede deshacer.

Cuando se ejecuta por completo, git clean hará una eliminación permanente del sistema de archivos

Cherry-pick: **git cherry-pick** es un potente comando que permite que las confirmaciones arbitrarias de Git se elijan por referencia y se añadan al actual HEAD de trabajo. La ejecución de cherry-pick es el acto de elegir una confirmación de una rama y aplicarla a otra. git cherry-pick puede ser útil para deshacer cambios.

Por ejemplo, supongamos que una confirmación se aplica accidentalmente en la rama equivocada. Puedes cambiar a la rama correcta y ejecutar cherry-pick en la confirmación para aplicarla a donde debería estar.

Cuando se detecta un error, es importante ofrecer una solución a los usuarios finales cuanto antes. Por ejemplo, supongamos que un desarrollador ha

comenzado a trabajar en una nueva función. Durante su desarrollo, identifica un error preexistente. Entonces, el desarrollador crea una confirmación explícita para aplicar una solución al error. Esta nueva confirmación de aplicación de solución se puede elegir mediante cherry-pick directamente en la rama principal para corregir el error antes de que afecte a más usuarios.

git cherry-pick commitSha (Numero de identificador del commit)