Observaciones

Me parece raro que en la base de datos no hayan insertado el id de la máquina, eso sería lo más lógico, si entiendo bien, le están enviado al cliente el identificador del servidor, algo que hayo más o menos inapropiado.

Respuesta 1:

Basta con que ambos servidores tomen al mismo tiempo la data de tickets a procesar y ambos enviaran en algún momento el mismo mail, pero con diferente servidor al cliente.

Respuesta 2:

1.- Se consume más recurso

2.- Aumenta la confiabilidad del sistema, debido a que, si alguno de ellos se cae, el otro puede suplir este problema

3.- Si los servidores no se encuentran en la misma región, podríamos hablar de alta disponibilidad, en caso de que en algún momento alguno de ellos no pueda ingresar a la base de datos, por algún como lo seria de un mantenimiento

4.- (opinión) La verdad es que, si la base de datos es centralizada, no hay mucha diferencia, debido a que dependemos de ella, pero si esta tuviera replicas en diversas zonas con algún tipo de sistema maestro esclavo, creo que tendría mayor sentido, o bien que fuera escalable.

Respuesta 3:

La mayor falla, es que no hay un agente intermediario que señale si los datos ya están siendo procesados por otro servidor, de manera de que el segundo no tenga que tomar estos datos hasta que el otro termine.

Respuesta 4:

La solución desde mi punto de vista, asumiendo que el proceso que toma mayor tiempo es el envío de un mail, sería dejar una sola maquina procesando y en el fragmento:

…

if(sendMail(data))

…

Lo cambiaría por un

…

Thread => sendMail()

…

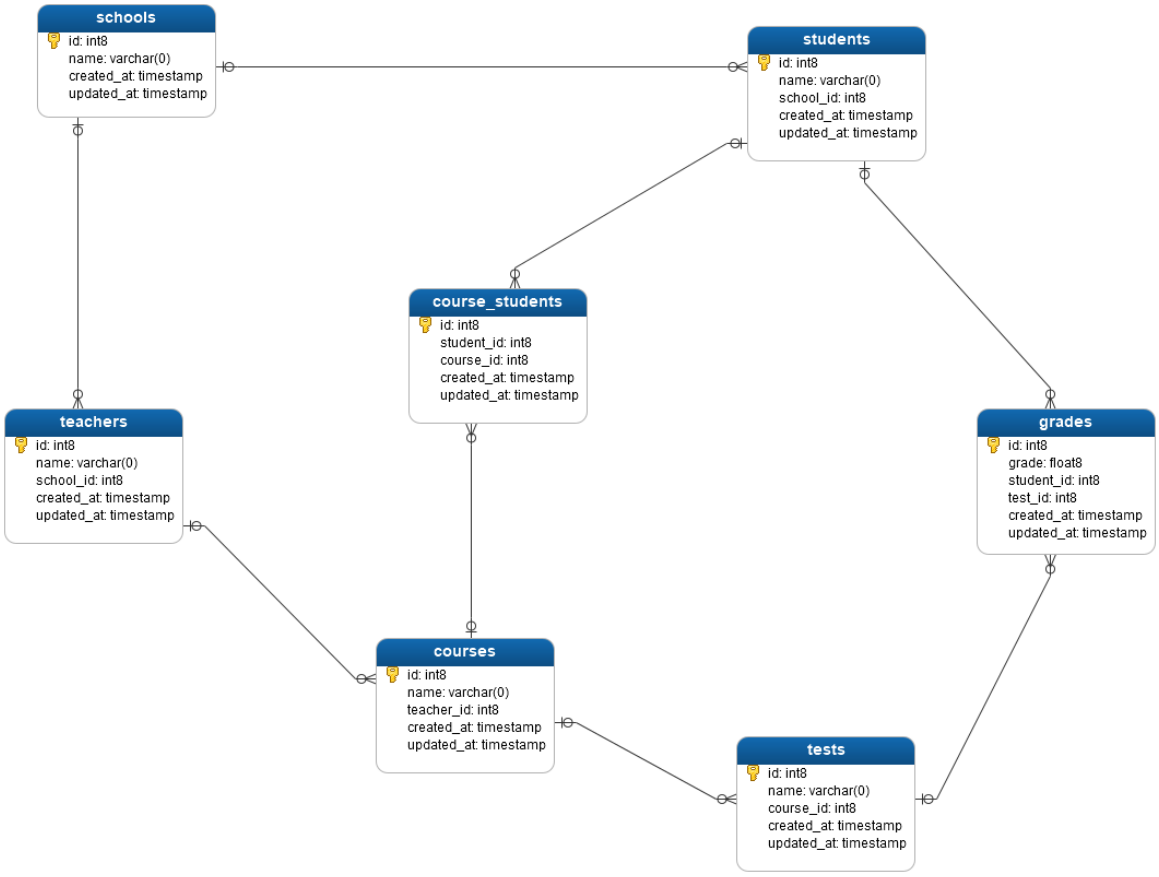
Naturalmente primero tendría que hacerse una prueba de estrés, para saber cuántos mails se pueden procesar al mismo tiempo, después de eso, el fragmento de “forech”, habría que darle un Sleep() de una cantidad de segundo cada x cantidad de mails enviados a procesar.

Yo para este caso, estoy asumiendo que el usar dos servidores, fue una decisión creada por alguien que está acostumbrado a procesar en un único hilo, por lo cual efectivamente este proceso sería bastante lento, debido a que posiblemente el sendMail, al igual que el UPDATE uno a uno, va a ser algo que tome bastante tiempo.

Otra solución desde otro punto de vista

Si la idea de usar dos servidores es por la disponibilidad, entonces habría que implementar un servicio, que funcione de semáforo, para que no permita procesar los datos en caso de que ya haya otro servidor (servicio, contenedor, script, etc) procesando.

Respuesta 1:



Respuesta 2

```sql
select s.name
from course_students cs
join courses c on c.id = cs.course_id
join students s on s.id = cs.student_id
where c.name = 'programación'
```

Respuesta 3

```sql
select s.name, avg(g.grade)
from grades g
join students s on s.id = g.student_id
join tests t on t.id = g.test_id
join courses c on c.id = t.course_id
where c.name = 'programación' and s.name = 'Sebastian'
group by s.name, s.name
```

Respuesta 4

```sql
select s.name student, c.name course, avg(g.grade) average_grade
from grades g
join students s on s.id = g.student_id
join tests t on t.id = g.test_id
join courses c on c.id = t.course_id
group by c.name, s.name
```

Respuesta 5

```sql
select failed_course.student from

(
    select s.name student, c.name course, avg(g.grade) average_grade
    from grades g
    join students s on s.id = g.student_id
    join tests t on t.id = g.test_id
    join courses c on c.id = t.course_id
    group by c.name, s.name
    having avg(g.grade) < 4
) failed_course

group by failed_course.student
having count(average_grade) > 1
```