

**دانشکده ریاضی و علوم کامپیوتر**

**برنامه‌سازی پیشرفته و کارگاه**

**دیتابیس مقدماتی ۲**

استاد درس

دکتر مهدی قطعی

استاد دوم

بهنام یوسفی مهر

نگارش

سیدآرمان حسینی

بهار ۱۴۰۳

joinها

‏foreign keyها رو یادتونه؟ بهتون اجازه می‌دادن که بین tableهای مختلف، ارتباط برقرار کنید. یکی از دستوراتی که بهتون اجازه می‌ده از این foreign keyها استفاده کنید، دستورات خانوادهٔ joinئه و توی این بخش، می‌خوایم با اون‌ها یه خورده ور بریم.

اولین join

کوئری‌هایی که تا حالا زدین، فقط با recordهای یه جدول درگیر بودن. مثلا کوئری زیر، فقط رکوردهای جدول tasks رو خروجی می‌داد:

select id, employee\_id, name from tasks;

یا دستور زیر، فقط stepهای تسک شمارهٔ ۳ رو نشون می‌داد:

select name from steps  
where task\_id = 3;

حالا فرض کنید که بخوایم توی یه کوئری، هم اطلاعات taskها رو نشون بدیم، هم اطلاعات stepها رو. چه کار می‌کنیم؟

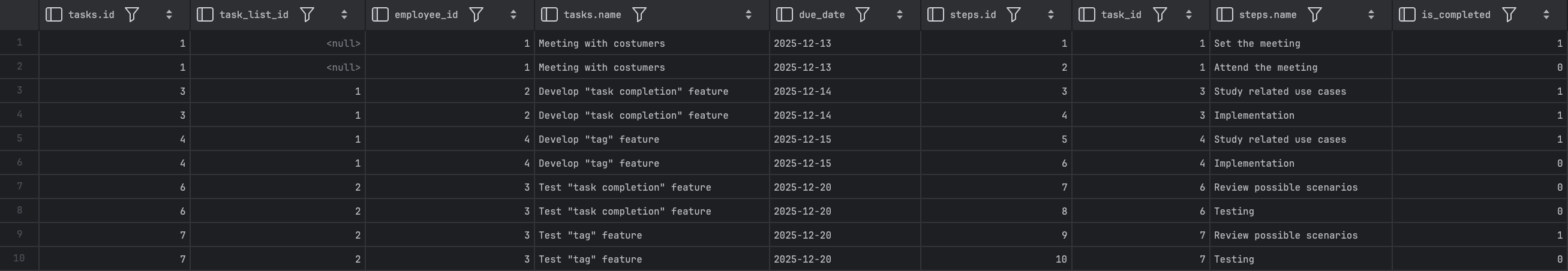
این‌جاست که دستور join به کارمون میاد. دستور join، ستون‌های دو یا چند جدول رو با هم ترکیب می‌کنه. به عنوان مثال، دستور زیر رو توی consoleتون اجرا کنید:

select \* from tasks  
join steps on tasks.id = steps.task\_id;

این دستور، به mysql همچین چیزی می‌گه:

1. اول، همه رکوردهای جدول tasks رو بخون.
2. بعدش، همهٔ رکوردهای جدول steps رو بخون.
3. حالا، هر step رو، کنار taskای نشون بده که idی اون task، برابر task\_idی stepمونه.

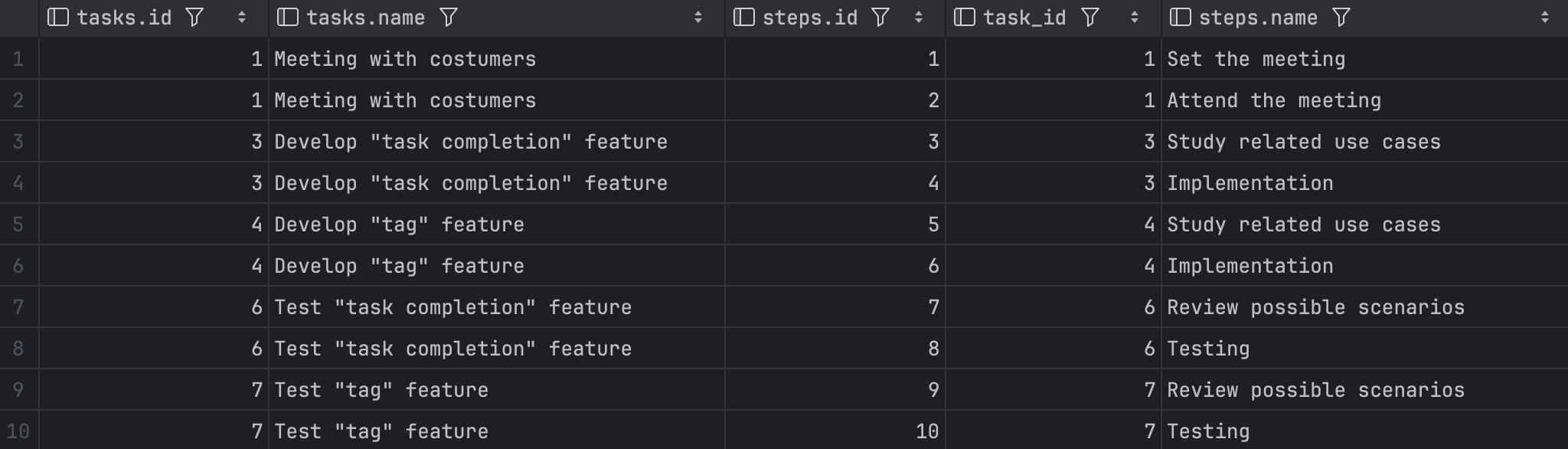
یه نگاه به خروجی‌ این دستور بندازید. خروجی‌ش، یه table گنده‌ست که همهٔ ستون‌های جداول tasks و steps رو داره. این table، یه همچین شکلی داره:



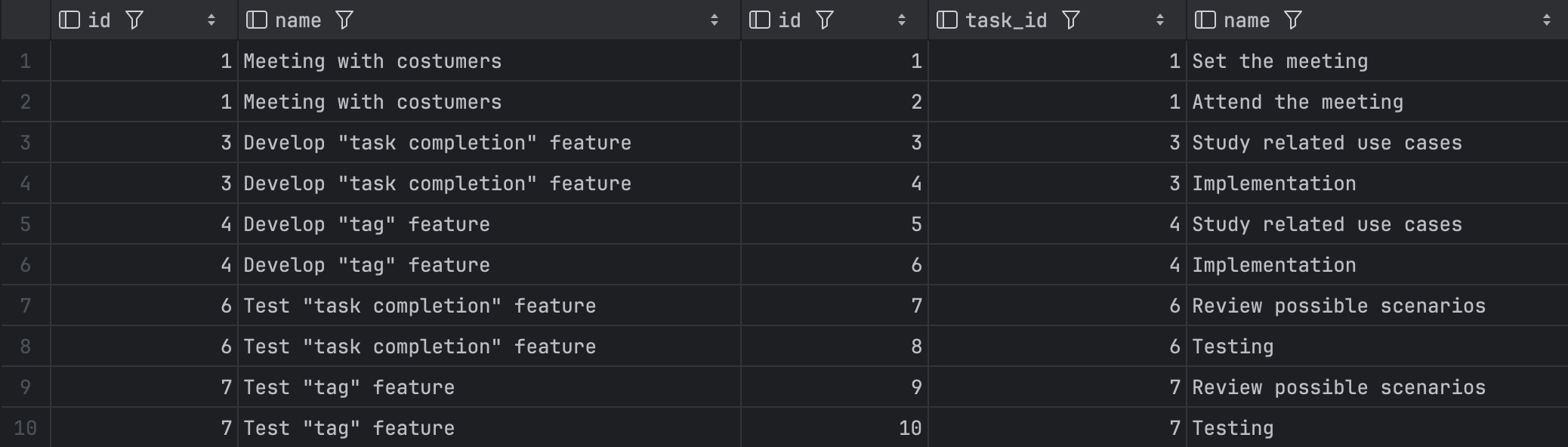
خوندن اطلاعات جدول بالا سخته، بیاین توی selectمون، از شر یه سری از ستون‌هاش خلاص بشیم تا بتونیم بهتر بررسی‌ش کنیم:

select tasks.id, tasks.name, steps.id, task\_id, steps.name from tasks  
join steps on tasks.id = steps.task\_id;

اگر دوباره این دستور رو اجرا کنید، خروجی زیر رو می‌بینید:

همون‌طور که می‌بینید، توی جدول بالا هر step به همراه task مربوط بهش دیده می‌شه. بعضی از taskها، دو بار توی خروجی دیده می‌شن، چون که دوتا step داشتن! ولی بعضی taskها هم اصلا توی این جدول دیده نمی‌شن، چون به کل stepای نداشتن.

اگر به جدول بالا دقت کنید، یه چیز جالب توی اون می‌بینید. این که ستون‌های id، به عنوان tasks.id و steps.id نمایش داده شدن. فرض کنید mysql این کار رو نمی‌کرد:



توی جدول بالا، کدوم ستون id مربوط به taskئه و کدوم مربوط به stepئه؟ ستون name چطور؟

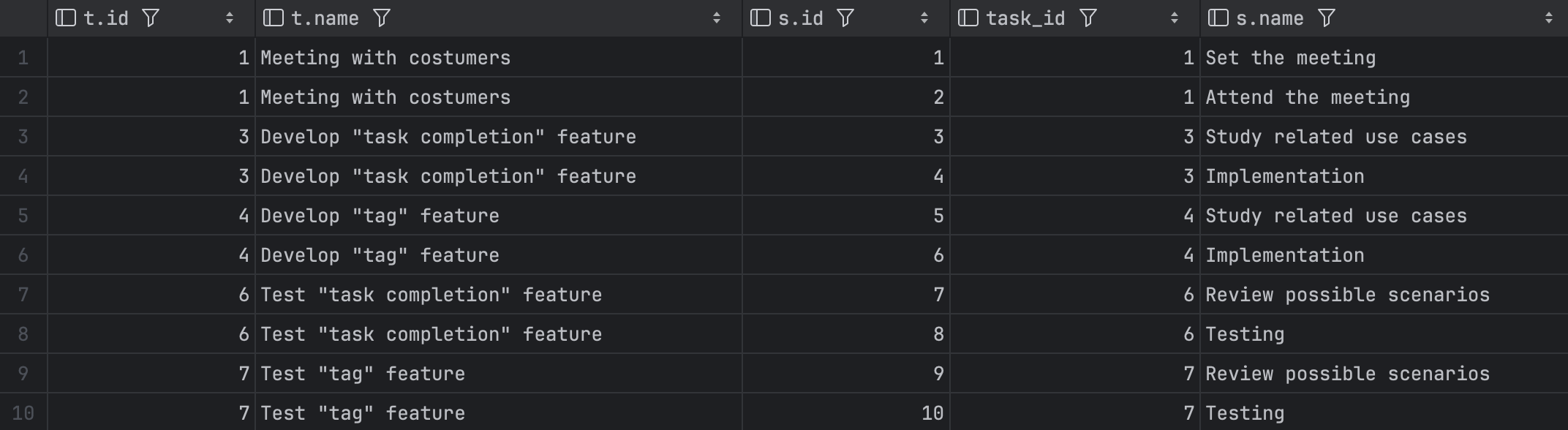
در واقع، چون این ستون‌ها توی هر دو جدول مشترکن، اسم جدول اون‌ها هم بالای ستونشون اومده. اگر دقت کنید، حتی توی کوئری‌مون هم هر جایی ما از این دو ستون استفاده کردیم، مجبور شدیم قبل اون‌ها اسم جدول‌شون رو بنویسیم:

select tasks.id, tasks.name, steps.id, task\_id, steps.name from tasks  
join steps on tasks.id = steps.task\_id;

ما می‌تونستیم این کوئری رو کوتاه‌تر هم بنویسیم:

select t.id, t.name, s.id, task\_id, s.name from tasks t  
join steps s on t.id = s.task\_id;

توی کوئری جدیدمون، به جای tasks از t و به جای steps از s استفاده کردیم. برای این که mysql بدونه منظورمون از t و s چیه، جلوی اسم این جداول هر کدوم رو مشخص کردیم. خروجی این کوئری هم مثل کوئری قبلیه، با این تفاوت که mysql هم از t و s استفاده کرده:

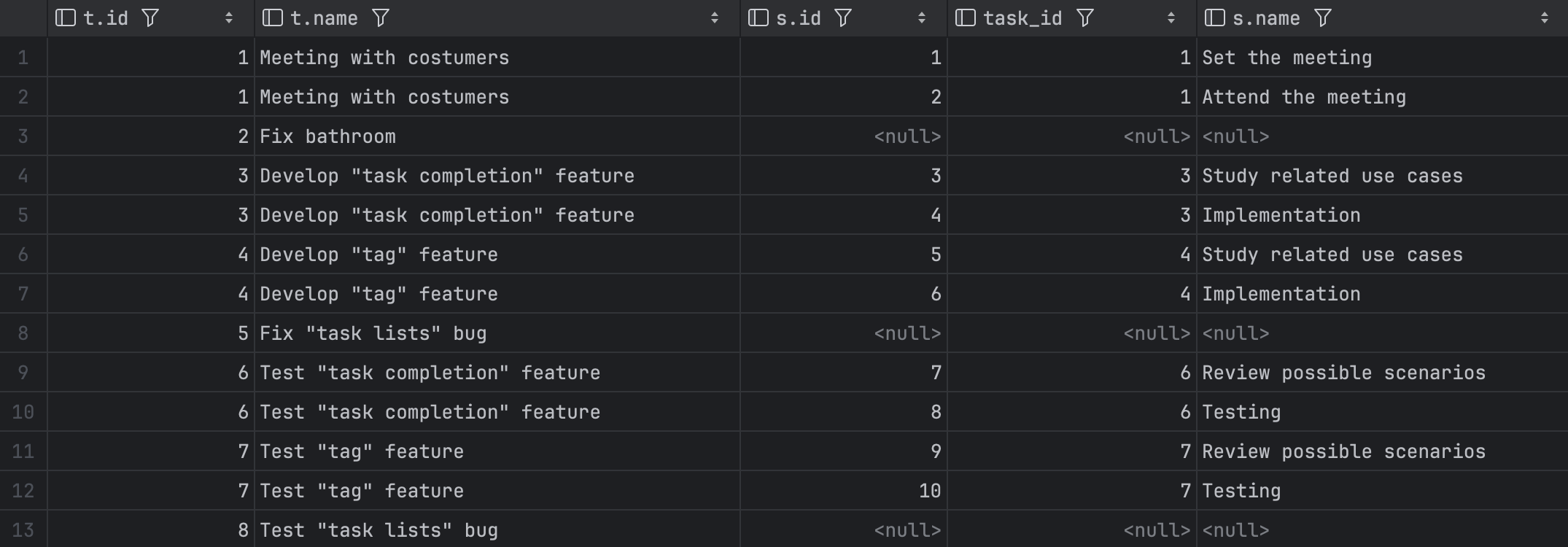


left join

اگر بخوایم taskهایی که stepای ندارن هم توی خروجی بالا نمایش داده بشن، از left join استفاده می‌کنیم:

select t.id, t.name, s.id, task\_id, s.name from tasks t  
left join steps s on t.id = s.task\_id;

خروجی دستور بالا، به این شکله:



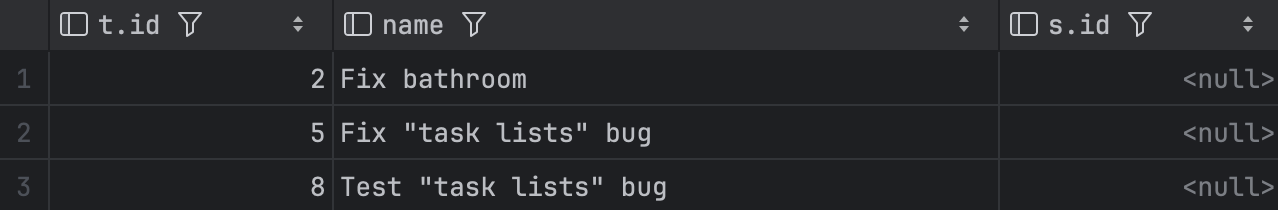
همون‌طور که می‌بینید، taskهای بدون step هم توی کوئری بالا نمایش داده شدن. توی سطرهای این taskها ستون‌های s.id، task\_id و s.name مقدار null رو گرفتن.

کوئری زدن روی نتایج

ما می‌تونیم مثل جدول‌های قبلی‌مون، روی این جدول هم where و group by و کوئری‌های مختلف بزنیم. مثلا توی کوئری زیر، taskهایی که هیچ stepای ندارن رو نشون می‌دیم:

select t.id, t.name, s.id from tasks t  
left join steps s on t.id = s.task\_id  
where s.id is null;

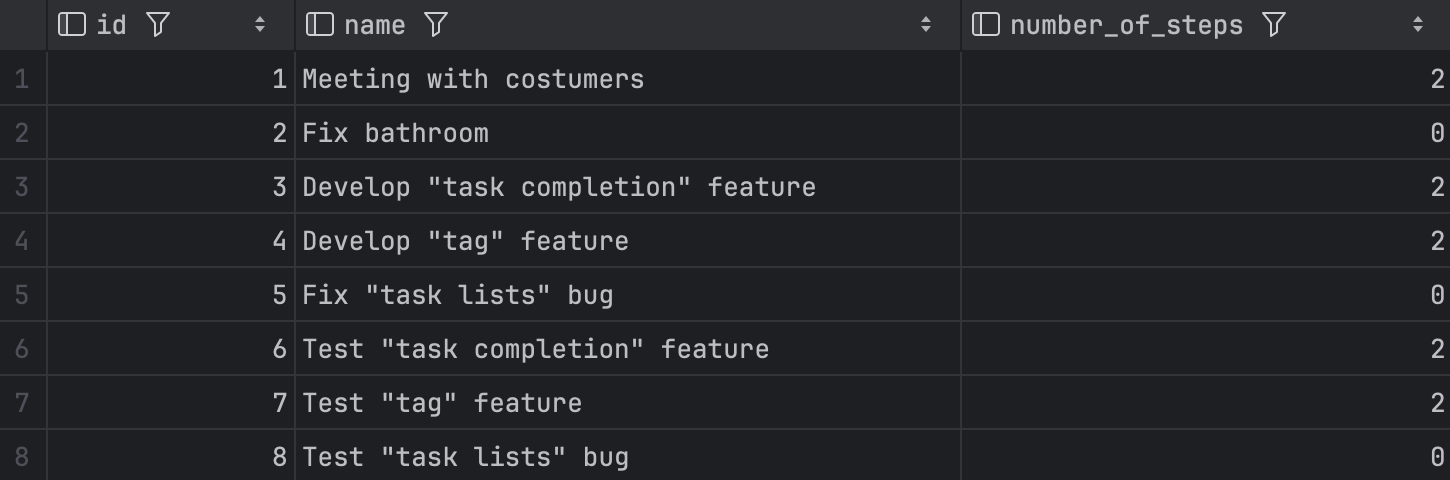
خروجی این دستور، به همچین شکلیه:



یا می‌تونیم ببینیم که هر task چند step داره:

select t.id, t.name, *count*(s.id) as number\_of\_steps from tasks t  
left join steps s on t.id = s.task\_id  
group by t.id;

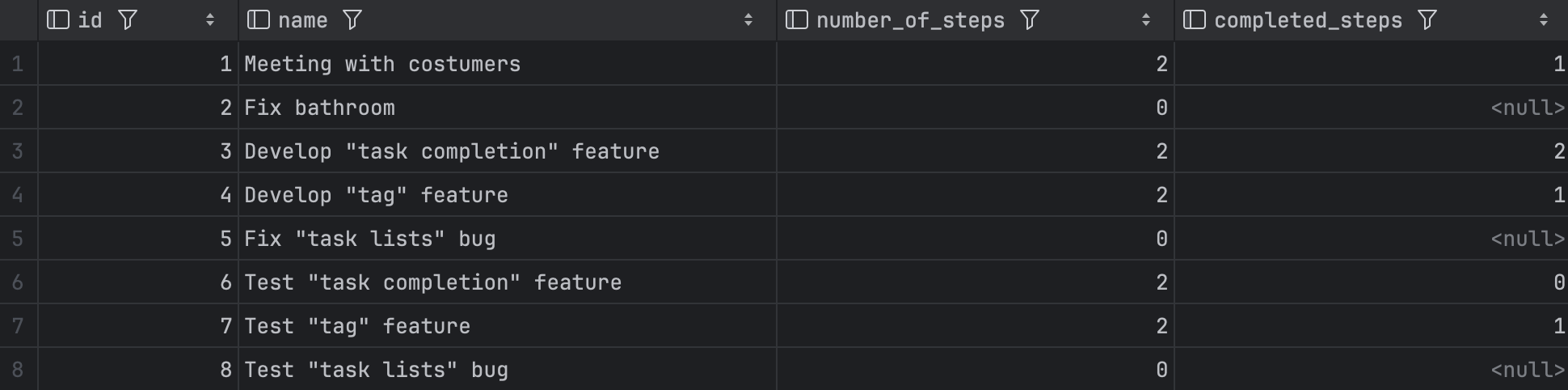
خروجی دستور بالا به شکل زیره:



حتی می‌تونیم ببینیم که از هر task، چند step تموم شده:

select   
 t.id,   
 t.name,   
 *count*(s.id) as number\_of\_steps,   
 *sum*(s.is\_completed = true) as completed\_steps  
from tasks t  
left join steps s on t.id = s.task\_id  
group by t.id;

خروجی کد بالا هم به شکل زیره:



همون‌طور که می‌بینید، ستون completed\_steps برای taskهای بدون step به جای 0، null گرفته. برای حل این مشکل از ifnull استفاده می‌کنیم. اگر کد زیر براتون واضح نیست، یه سرچ کوچیک راجع به این تابع بکنید:

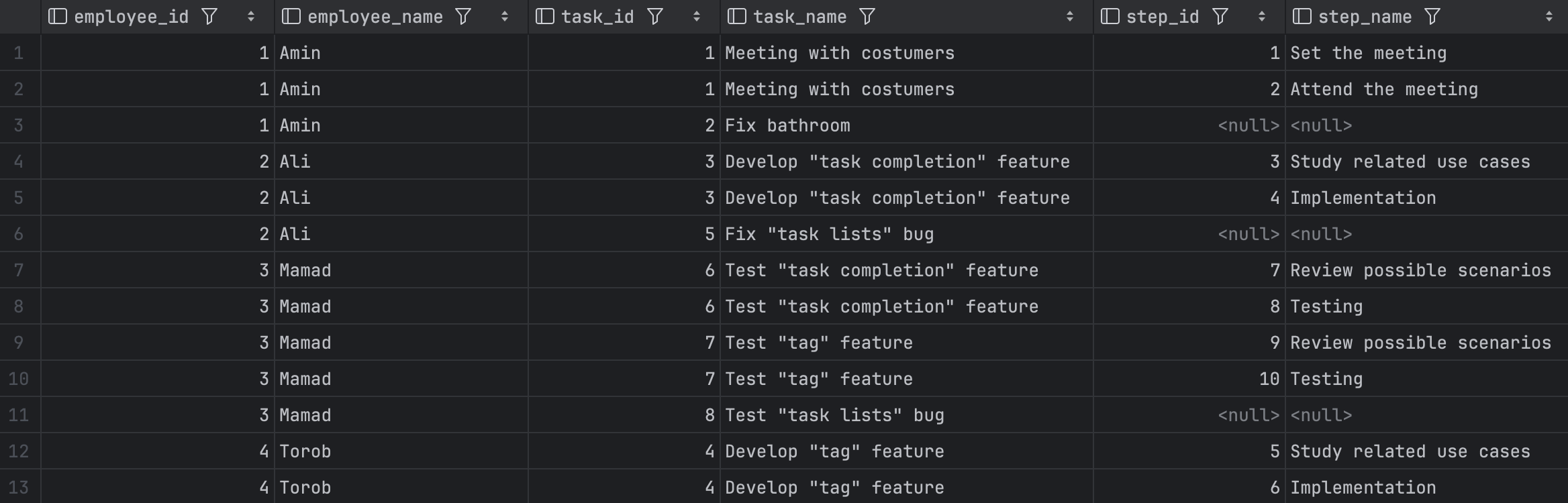
select  
 t.id,  
 t.name,  
 *count*(s.id) as number\_of\_steps,  
 *ifnull*(*sum*(s.is\_completed = true), 0) as completed\_steps  
from tasks t  
left join steps s on t.id = s.task\_id  
group by t.id;

join چند جدول

ما حتی می‌تونیم بیشتر از دو جدول رو با هم join کنیم. مثل بیاید جدول employee هم با task و step جوین کنیم:

select   
 e.id as employee\_id,   
 e.name as employee\_name,   
 t.id as task\_id,   
 t.name as task\_name,   
 s.id as step\_id,   
 s.name as step\_name  
from employees e  
left join tasks t on t.employee\_id = e.id  
left join steps s on s.task\_id = t.id;

خروجی دستور بالا، به این شکله:



با استفاده از جدول بالا هم می‌تونیم کوئری‌های خوبی راجع به کارکنان شرکت بزنیم. مثلا این که هر کارمند، چند قدم از کارهاش رو انجام داده:

select  
 e.id as employee\_id,  
 e.name as employee\_name,  
 *count*(s.id) as total\_number\_of\_steps,  
 *ifnull*(*sum*(s.is\_completed = true), 0) as completed\_steps  
from employees e  
left join tasks t on t.employee\_id = e.id  
left join steps s on s.task\_id = t.id  
group by e.id;

خروجی این کوئری هم به شکل زیره:

