Overall Design

What problem are we facing?

- The goal of this project is to build a web-based issue tracking system.
- Key function of the tracking system is to create tasks, to report issues for tasks, to assign issues to particular people, to maintain, and be able to change issue status in workflow.
- To achieve the goal, I include four entities in the design: Tasks, Users, Status, Workflows.

Who are the players of the tracking system?

- Around the function of the system, there are several players. Person who creates the tasks is the "reporter". Then he or she can assign the tasks to other fellow workers, they become the "assignees". Assignees and reporters can both change tasks' current status, but only reporters can assign assignees to certain tasks, not the other way around.
- I create a User Table to store basic information of both types of players, such as name, email, display_name and password.
- Users and tasks are connected through the Relation"Assignment". Assignment.uid refers to a tuple of users in Users Table.

What type of tasks should be stored in the database?

- There are two types of tasks, system users create projects, and reporters report issues. Therefore in the Tasks Table, I use an attribute "ttype", "parent_tid" to specify the task type. ttype = P means it's a project, ttype = I means it's an issue. P. parent_tid = NULL, I.parent_tid = 1, which means project is the parent of a certain issue.
- "Parent_tid" is a foreign key referencing to itself tasks (tid).
- Furthermore, we store "wfid" and "status" as foreign keys in the Tasks Table, "wfid" refers to the Workflows Table(wfid), and "status" refers to the Status Table (sid).

Use case analysis

1. Actors

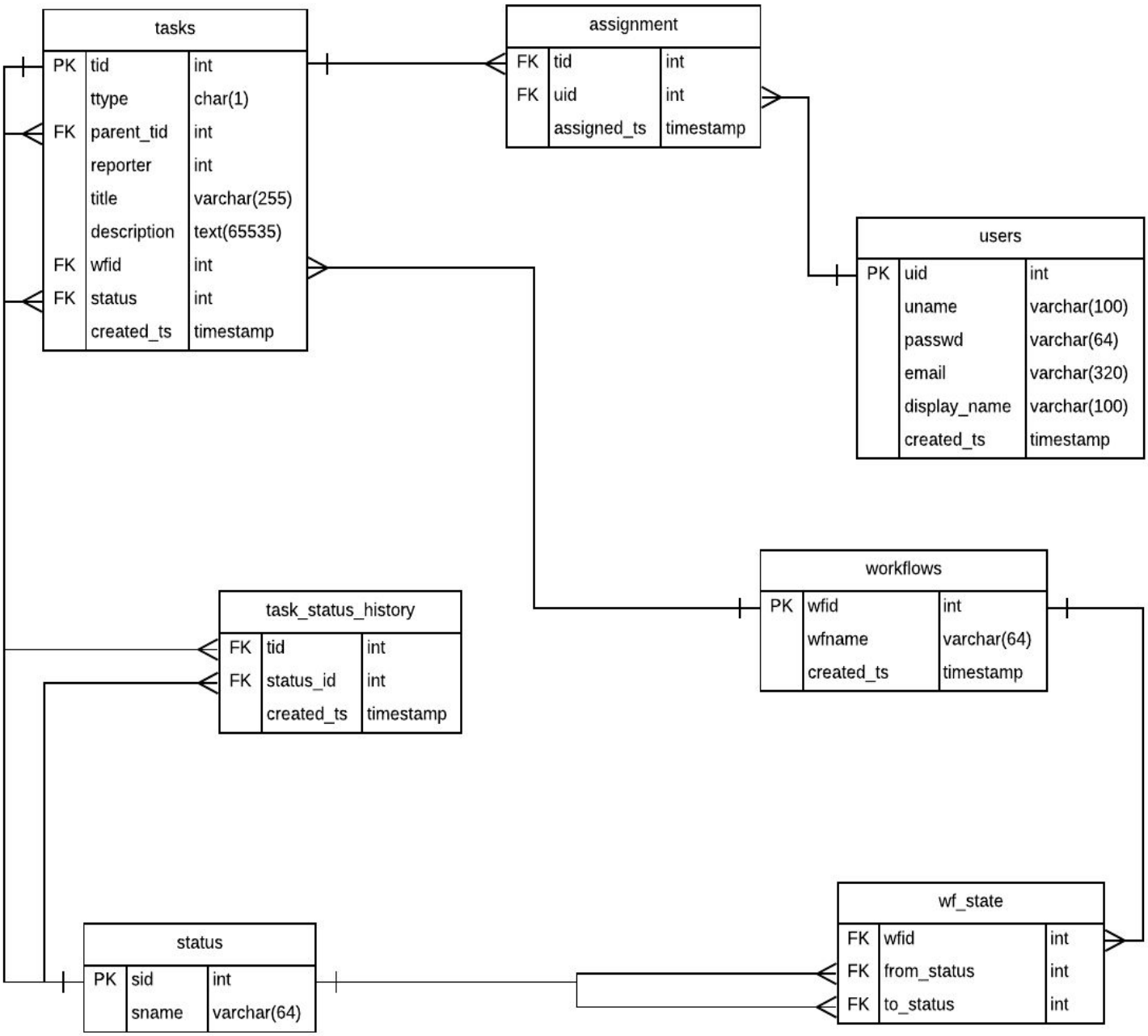
   There are two types of actors in this project, one is the project lead, which can assign tasks to users, can define workflow of projects, create tasks, and update status of the projects. The other one is the normal users (assignees), which can create tasks, and update status of the projects.

2. Use cases
   1) Sign up
      Any user can sign up for the system by providing email, can choose their username, display name, and password (password will be store in SHA 256 shadow)
   2) Login in authentication
      Any user uses a username and password to log in the system. Assuming application will convert user entered password into SHA 256 shadow to compare against the database
   3) Authorization
      There's simple authorization, to check whether a user is authorized to update a project and its issues, to check whether a user is a project lead or assignee to a particular project.
   4) Create tasks
      Users can create projects, which become the project lead of that project.
      Any user can report an issue of a project.
      Both project and issue is a kind of task. Project is a parent of issues.
   5) Assign tasks to users
      Project lead can assign other fellow users to work on his/her project. Only the project lead has this action.
   6) Define workflow of projects
      Project lead needs to specify the workflow for his/her project. The workflow defines the allowed status and change between status.
   7) Update status of the projects
      Both project leads and assignees can change the current status of an issue, based on current status, and the project workflow.

3. Assumption and Scope
   1) No user role based authorization. A user becomes the lead if and only if the user creates a project, there's no role constraint on whether a user can create projects.
   2) For workflow, we assume there's only one default starting status allowed.
   3) Projects and issues have inherently similar attributes, such as the name of project and title of issue are of similar data type and size.

Data Model

1. ER Diagram



2. Relational Schema and Constraints

- tasks(**tid**, ttype, parent_tid, reporter, title, description, wfid, status, created_ts)
- users(**uid**, uname, passwd, email, display_name, created_ts)
- assignment (tid, uid, assigned_ts)
- status (**sid**, sname)
- workflows (**wfid**, wfname, created_ts)
- wf_state (wfid, from_status, to_status)
- task_status_history (tid, status_id, created_ts)

Foreign key -------> Reference table

- tasks(parent_tid) -----> tasks(tid)
- tasks(reporter) -----> users(uid)
- tasks(wfid) -----> workflows(wfid)
- tasks(status) -----> status (sid)
- assignment(tid) -----> tasks(tid)
- assignment(uid) -----> users(uid)
- task_status_history (tid) -----> tasks(tid)
- task_status_history (status_id) -----> status (sid)
- wf_state (wfid) ----> workflows(wfid)
- wf_state (from_status, to_status) -----> status (sid)

Schema DDL

```
use ticket;

drop table if exists tasks, users, assignment, workflows, wf_state, status,
task_status_history;

create table status (
  sid int not null auto_increment,
  sname varchar(64) not null,

  primary key (sid)
);

create table workflows (
  wfid int not null auto_increment,
  wfname varchar(64) not null,
  created_ts timestamp not null,

  key (wfname),
  primary key (wfid)
);

create table wf_state (
  wfid int not null,
  from_status int default null,
  to_status int default null,

  key (wfid, from_status),
  constraint fk_wfs_wfid foreign key (wfid) references workflows (wfid),
  constraint fk_wfs_fsid foreign key (from_status) references status (sid),
  constraint fk_wfs_tsid foreign key (to_status) references status (sid)
);

create table users(
  uid int not null auto_increment,
  uname varchar(100) not null,
  passwd varchar(64) not null,
  email varchar(320) not null,
  display_name varchar(100) not null,
  created_ts timestamp not null,
```

```
  primary key (uid)
);

create table tasks(
  tid int not null auto_increment,
  ttype char not null,
  parent_tid int default null,
  reporter int not null,
  title varchar(255) not null,
  description text default null,
  wfid int not null,
  status int default null,
  created_ts timestamp not null,

  primary key (tid),
  key parent_tid (parent_tid),
  key wfid (wfid),
  key status (status),
  fulltext index title (title),
  fulltext index description(description),
  constraint fk_t_ptid foreign key (parent_tid) references tasks (tid),
  constraint fk_t_wfid foreign key (wfid) references workflows (wfid),
  constraint fk_t_sid foreign key (status) references status (sid)
);

create table assignment (
  tid int not null,
  uid int not null,
  assigned_ts timestamp not null,

  key tid (tid),
  key uid (uid),
  constraint fk_a_tid foreign key (tid) references tasks (tid),
  constraint fk_a_uid foreign key (uid) references users (uid)
);

create table task_status_history (
  tid int not null,
  status_id int not null,
  created_ts timestamp not null,

  key tid (tid),
```

```
  key status_id (status_id),
  constraint fk_tsh_tid foreign key (tid) references tasks (tid),
  constraint fk_tsh_sid foreign key (status_id) references status (sid)
)
```

Test and Verification

1. Test data

```
use ticket;

/* Some Sample Status */
insert into status (sname) values
('OPEN'),
('IN_PROC'),
('REVIEW'),
('QA'),
('CLOSED')
;
select * from status;

/* Sample workflow */
insert into workflows (wfname, created_ts) values
('project1 wf', now()),
('project2 wf', now()),
('project3 wf', now())
;
select * from workflows;

insert into wf_state (wfid, from_status, to_status) values
(1, null, 1),
(1, 1, 2),
(1, 2, 1),
(1, 2, 3),
(1, 3, 2),
(2, 3, 1),
(2, 3, 4),
(2, 4, 3),
(3, 4, 1),
(3, 4, 5),
(3, 5, 1)
;
select * from wf_state;
```

```
/* Sample users */
insert into users (uname, passwd, email, display_name, created_ts) values
('test1','340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a',
'test1@example.com', 'Lucy Lu', now()),
('test2','340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a',
'test2@example.com', 'Mimi Hu', now()),
('test3','340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a',
'test3@example.com', 'Lin Pan', now()),
('test4','340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a',
'test4@example.com', 'Jeff Bezos', now())
;
select * from users;

/* Sample tasks */
insert into tasks (ttype, parent_tid, reporter, title, description, wfid, status,
created_ts ) values
('P', null, 2, 'Solidarity GN', 'Meet at Library', 1, null, now()),
('P', null, 3, 'Social Distancing', 'Stay at home', 2, null, now()),
('I', 1, 1, 'New Roof', 'Fix Library roof', 3, 1, now()),
('I', 1, 4, 'Amazon Kindle screen', 'Make Kindle Great Again', 3, 1, now())
;

select * from tasks;

insert into task_status_history (tid, status_id, created_ts) values
(2, 2, now()),
(1, 1, now()),
(3, 2, now()),
(3, 3, now()),
(3, 4, now()),
(3, 5, now())
;
select * from task_status_history;

/* Sample Assign*/
insert into assignment (tid, uid, assigned_ts) values
(1, 2, now()),
(1, 3, now()),
(2, 1, now()),
(2, 2, now()),
(4, 4, now())
;
select * from assignment;
```

2. Answers to question c (sql query)

1) Before and after creating a new user account

```
5
6    /* (1) Create a new user account, together with email, password, username, and display name.  */
7    insert into users (uname, passwd, email, display_name, created_ts) values
8    ('tester','340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a', 'test4@example.com', 'Jeff Shi', now());
9    select * from users;
10
```

100% 5:9

Result Grid | Filter Rows: Search | Edit: | Export/Import:

uid	uname	passwd	email	display_name	created_ts
1	test1	340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a	test1@example.com	Lucy Lu	2020-04-25 08:55:11
2	test2	340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a	test2@example.com	Mimi Hu	2020-04-25 08:55:11
3	test3	340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a	test3@example.com	Lin Pan	2020-04-25 08:55:11
NULL	NULL	NULL	NULL	NULL	NULL

```
5
6    /* (1) Create a new user account, together with email, password, username, and display name.  */
7    insert into users (uname, passwd, email, display_name, created_ts) values
8    ('tester','340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a', 'test4@example.com', 'Jeff Shi', now());
9    select * from users;
10
```

100% 5:9

Result Grid | Filter Rows: Search | Edit: | Export/Import:

uid	uname	passwd	email	display_name	created_ts
1	test1	340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a	test1@example.com	Lucy Lu	2020-04-25 08:55:11
2	test2	340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a	test2@example.com	Mimi Hu	2020-04-25 08:55:11
3	test3	340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a	test3@example.com	Lin Pan	2020-04-25 08:55:11
4	tester	340c19a145a5232ba0d24a2c91c0ee5bf5a06e2556c83e5072697338891b2e1a	test4@example.com	Jeff Shi	2020-04-25 08:55:...
NULL	NULL	NULL	NULL	NULL	NULL

2) Before and after Creating an issue for a project

```
10
11   /* (2) Create an issue for a project with title and description, and initialize the status of this issue. */
12   insert into tasks (ttype, parent_tid, reporter, title, description, wfid, status, created_ts ) values
13   ('I', 1, 2, 'New Wall', 'Fix Library roof', 1, 1, now());
14
15   set @new_tid = (select LAST_INSERT_ID());
16   insert into task_status_history (tid, status_id, created_ts) values
17   (@new_tid, 1, now())
18   ;
19   select * from tasks;
20
```

100% 4:19

Result Grid | Filter Rows: Search | Edit: | Export/Import:

tid	ttype	parent_tid	reporter	title	description	wfid	status	created_ts
1	P	NULL	2	Solidarity GN	Meet at Library	1	NULL	2020-04-25 08:55:11
2	P	NULL	3	Social Distancing	Stay at home	2	NULL	2020-04-25 08:55:11
3	I	1	1	New Roof	Fix Library roof	3	1	2020-04-25 08:55:11
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
10
11    /* (2) Create an issue for a project with title and description, and initialize the status of this issue. */
12    insert into tasks (ttype, parent_tid, reporter, title, description, wfid, status, created_ts ) values
13    ('I', 1, 2, 'New Wall', 'Fix Library roof', 1, 1, now());
14
15    set @new_tid = (select LAST_INSERT_ID());
16    insert into task_status_history (tid, status_id, created_ts) values
17    (@new_tid, 1, now())
18    ;
19    select * from tasks;
20
```

100% 5:19

Result Grid Filter Rows: Search Edit: Export/Import:

tid	ttype	parent_tid	reporter	title	description	wfid	status	created_ts
1	P	NULL	2	Solidarity GN	Meet at Library	1	NULL	2020-04-25 08:55:11
2	P	NULL	3	Social Distancing	Stay at home	2	NULL	2020-04-25 08:55:11
3	I	1	1	New Roof	Fix Library roof	3	1	2020-04-25 08:55:11
4	I	1	2	New Wall	Fix Library roof	1	1	2020-04-25 08:59:...
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3) Check user authorization, then add an assignee.

```
20
21    /* (3) For a current user and a certain issue, first check if this user is authorized to assign it to other
22    users (i.e., is a lead)-- query result > 1 means there exists such user; then write a query to add an assignee.  */
23
24    select count(a.uid)
25    from tasks p inner join tasks i on p.tid = i.parent_tid
26    inner join assignment a on a.tid = p.tid
27    where i.tid = 3 /*issue id*/
28    and a.uid = 2; /* user id */
29
```

100% 4:24

Result Grid Filter Rows: Search Export:

count(a.uid)
1

```
29
30    insert into assignment (tid, uid, assigned_ts) values
31    (2, 4, now());
32    select * from assignment;
33
```

100% 4:32

Result Grid Filter Rows: Search Export:

tid	uid	assigned_ts
1	2	2020-04-25 08:55:11
1	3	2020-04-25 08:55:11
2	1	2020-04-25 08:55:11
2	2	2020-04-25 08:55:11
2	3	2020-04-25 08:55:11
2	4	2020-04-25 09:03:43
2	4	2020-04-25 09:06:35

4) List all possible next statuses of a certain issue, based on its current status

```
/* (4) List all possible next statuses of a certain issue, based on its current status */
select * from tasks where parent_tid = 1; /*find out there're two issues currently*/

select s.sid, s.sname
from tasks i inner join wf_state ws on i.wfid = ws.wfid
and i.status = ws.from_status inner join status s on ws.to_status = s.sid
where i.tid = 4 /* issue id*/
;
```

100% 3:38

Result Grid | Filter Rows: Search | Export:

sid	sname
2	IN_PROC

5) Show the status change history of a certain issue, time desc

```
/* (5) Show the status change history of a certain issue, sorted by change timestamps in descending
order.  */
select *
from task_status_history
where tid = 3 /* issue id */
order by created_ts desc;
```

100% 4:46

Result Grid | Filter Rows: Search | Export:

tid	status_id	created_ts
3	2	2020-04-25 09:20:46
3	3	2020-04-25 09:20:46
3	4	2020-04-25 09:20:46
3	5	2020-04-25 09:20:46
3	3	2020-04-25 08:55:11

6) List any issues for the project with the name “Amazon Kindle” issue title contains the term “screen”, assignees is “Jeff Bezos”, and the status is “OPEN”.

```
/* (6) List any issues for the project with name “Amazon Kindle” where the issue title contains the
term “screen”, user “Jeff Bezos” is one of the assignees, and the status of the issue is
“OPEN”.  */

select *
from tasks i inner join tasks p on i.parent_tid = p.tid
inner join assignment a on a.tid = i.tid
inner join users u on a.uid = u.uid
inner join status s on i.status = s.sid
where p.title = 'Amazon Kindle'
and MATCH(i.title) against ('screen' IN NATURAL LANGUAGE MODE) and u.uname = 'Jeff Bezos'
;
```

100% 4:55

Result Grid | Filter Rows: Search | Export:

tid	ttype	parent_tid	reporter	title	description	wfid	status	created...	tid	ttype	parent_