**SELECT task\_number\_id, avg(attempt\_num) as average\_number\_of\_attempts**

**from model.submission\_of\_task**

**where result\_flg = 1**

**group by task\_number\_id;**

**SELECT ach.achievement\_id**

**from model.achievements as ach**

**INNER JOIN (select \***

**from model.achieve\_child as achch**

**INNER JOIN model.child as ch on achch.kid\_id = ch.child\_id) as q**

**on q.achievement\_id = ach.achievement\_id**

**where RIGHT(ach.achievement\_desc, LENGTH(ach.achievement\_desc) - 11) = CONCAT(' ', 2);**

**SELECT full\_name\_nm, achievement\_desc**

**from model.achievements as ach**

**INNER JOIN (select \***

**from model.achieve\_child as achch**

**INNER JOIN model.child as ch on achch.kid\_id = ch.child\_id) as q**

**on q.achievement\_id = ach.achievement\_id;**

**SELECT full\_name\_nm, avg(attempt\_num)**

**from model.submission\_of\_task as s**

**INNER JOIN model.teacher as t on**

**s.teacher\_id = t.teacher\_id**

**where result\_flg = 1**

**group by full\_name\_nm;**

**SELECT task\_number\_id, count(result\_flg) as completed\_by**

**from model.submission\_of\_task**

**where result\_flg = 1**

**group by task\_number\_id;**

**SELECT full\_name\_nm, count(result\_flg) as number\_of\_completed\_tasks**

**from model.child**

**INNER JOIN model.submission\_of\_task**

**ON model.child.child\_id = model.submission\_of\_task.child\_id**

**where result\_flg = 1**

**group by full\_name\_nm;**

**SELECT team\_id, count(result\_flg) as number\_of\_completed\_tasks**

**from model.child as c**

**INNER JOIN model.submission\_of\_task as s**

**on c.child\_id = s.child\_id**

**where result\_flg = 1**

**group by team\_id;**

**SELECT team\_id, avg(attempt\_num) as average\_number\_of\_attempts**

**from model.child as c**

**INNER JOIN model.submission\_of\_task as s**

**on c.child\_id = s.child\_id**

**where result\_flg = 1**

**group by team\_id;**

**------------------------------------------------------------------------------------------------------------------------**

**DROP SCHEMA IF EXISTS project\_views CASCADE;**

**create schema project\_views;**

**create view project\_views.v\_children as**

**select full\_name\_nm, grade\_num, gender\_txt, room\_id as room\_number, team\_id as team\_number**

**from model.child;**

**create view project\_views.v\_achievements as**

**select achievement\_desc, issue\_dttm**

**from model.achievements;**

**create view project\_views.v\_classroom as**

**select classroom\_id as class\_number, number\_of\_chairs\_cnt, number\_of\_tables\_cnt**

**from model.classroom;**

**create view project\_views.v\_room as**

**select room\_number\_nm, number\_of\_beds\_cnt**

**from model.room;**

**create view project\_views.v\_task as**

**select\***

**from model.task;**

**create view project\_views.v\_teacher as**

**select full\_name\_nm, theme\_txt, room\_id, gender\_txt**

**from model.teacher;**

**create view project\_views.v\_team as**

**select \***

**from model.team;**

**create view project\_views.submission as**

**SELECT full\_name\_nm, achievement\_desc**

**from model.achievements as ach**

**INNER JOIN (select \***

**from model.achieve\_child as achch**

**INNER JOIN model.child as ch on achch.kid\_id = ch.child\_id) as q**

**on q.achievement\_id = ach.achievement\_id;**

**create view project\_views.completed\_by as**

**SELECT full\_name\_nm, count(result\_flg) as number\_of\_completed\_tasks**

**from model.child**

**INNER JOIN model.submission\_of\_task**

**ON model.child.child\_id = model.submission\_of\_task.child\_id**

**where result\_flg = 1**

**group by full\_name\_nm;**

**create view project\_views.average\_attempt\_in\_team as**

**SELECT team\_id, avg(attempt\_num) as average\_number\_of\_attempts**

**from model.child as c**

**INNER JOIN model.submission\_of\_task as s**

**on c.child\_id = s.child\_id**

**where result\_flg = 1**

**group by team\_id;**

**create view project\_views.teacher\_average\_attempt as**

**SELECT full\_name\_nm, avg(attempt\_num)**

**from model.submission\_of\_task as s**

**INNER JOIN model.teacher as t on**

**s.teacher\_id = t.teacher\_id**

**where result\_flg = 1**

**group by full\_name\_nm;**

**create view project\_views.v\_ratings\_in\_teams as**

**SELECT child.full\_name\_nm,**

**child.team\_id,**

**(sum(task.difficulty\_rate) OVER (PARTITION BY child.team\_id, child.full\_name\_nm )) as rating,**

**submission\_dttm,**

**(sum(task.difficulty\_rate)**

**OVER (PARTITION BY child.team\_id, child.full\_name\_nm ORDER BY submission\_dttm)) as rating\_during\_time**

**from model.child as child**

**JOIN model.submission\_of\_task as sub on child.child\_id = sub.child\_id**

**JOIN model.task as task on task.task\_number\_id = sub.task\_number\_id**

**JOIN project\_views.completed\_by as completed on completed.full\_name\_nm = child.full\_name\_nm**

**where sub.result\_flg = 1;**

**create view project\_views.v\_rating as**

**SELECT full\_name\_nm, sum(raiting\_in\_team)**

**from (**

**SELECT full\_name\_nm, avg(rating) as raiting\_in\_team**

**from project\_views.v\_rating**

**group by full\_name\_nm, team\_id) as raitings\_in\_teams**

**group by full\_name\_nm;**

**create view project\_views.v\_grade\_3 as**

**SELECT full\_name\_nm, number\_of\_completed\_tasks from**

**project\_views.completed\_by**

**group by full\_name\_nm**

**having avg(number\_of\_completed\_tasks) >= 6 and avg(number\_of\_completed\_tasks) < 10;**

**create view project\_views.v\_grade\_4 as**

**SELECT full\_name\_nm, number\_of\_completed\_tasks from**

**project\_views.completed\_by**

**group by full\_name\_nm**

**having avg(number\_of\_completed\_tasks) >= 10 and avg(number\_of\_completed\_tasks) < 20;**

**create view project\_views.v\_grade\_3 as**

**SELECT full\_name\_nm, number\_of\_completed\_tasks from**

**project\_views.completed\_by**

**group by full\_name\_nm**

**having avg(number\_of\_completed\_tasks) >= 20 and avg(number\_of\_completed\_tasks) < 30;**

**drop function if exists *get\_next\_achievement\_index*;**

**create function *get\_next\_achievement\_index*()**

**returns Integer as**

**$$**

**declare**

**index Integer;**

**begin**

**index = 0;**

**if exists(select achievement\_id from model.achievements) then**

**index = (select max(achievement\_id) from model.achievements);**

**end if;**

**index = index + 1;**

**return index;**

**end;**

**$$ LANGUAGE plpgsql;**

**create or replace function *add\_achievement*() returns trigger as**

**$$**

**declare**

**new\_achieve\_index\_id INTEGER default (1);**

**achievement\_\_id Integer;**

**task\_time\_dttm timestamp(8);**

**new\_task\_id Integer;**

**new\_child\_id Integer;**

**new\_time\_dttm timestamp(8);**

**result Integer;**

**task\_count Integer;**

**raiting Integer;**

**begin**

**new\_achieve\_index\_id = *get\_next\_achievement\_index*();**

**new\_task\_id = (select task\_number\_id**

**from model.submission\_of\_task**

**where submission\_dttm = new.submission\_dttm**

**and task\_number\_id = new.task\_number\_id**

**and child\_id = new.child\_id**

**and teacher\_id = new.teacher\_id**

**and attempt\_num = new.attempt\_num);**

**new\_child\_id = (select child\_id**

**from model.submission\_of\_task**

**where submission\_dttm = new.submission\_dttm**

**and task\_number\_id = new.task\_number\_id**

**and child\_id = new.child\_id**

**and teacher\_id = new.teacher\_id**

**and attempt\_num = new.attempt\_num);**

**new\_time\_dttm = (select submission\_dttm**

**from model.submission\_of\_task**

**where submission\_dttm = new.submission\_dttm**

**and task\_number\_id = new.task\_number\_id**

**and child\_id = new.child\_id**

**and teacher\_id = new.teacher\_id**

**and attempt\_num = new.attempt\_num);**

**result = (select result\_flg**

**from model.submission\_of\_task**

**where submission\_dttm = new.submission\_dttm**

**and task\_number\_id = new.task\_number\_id**

**and child\_id = new.child\_id**

**and teacher\_id = new.teacher\_id**

**and attempt\_num = new.attempt\_num);**

**if result = 0 then**

**return new;**

**end if;**

**if exists(SELECT ach.achievement\_id**

**from model.achievements as ach**

**INNER JOIN (select \***

**from model.achieve\_child as achch**

**INNER JOIN model.child as ch on achch.kid\_id = ch.child\_id) as q**

**on q.achievement\_id = ach.achievement\_id**

**where RIGHT(ach.achievement\_desc, LENGTH(ach.achievement\_desc) - 11) = CONCAT(' ', new\_task\_id)) then**

**achievement\_\_id = (select (ach.achievement\_id)**

**from model.achievements as ach**

**INNER JOIN (select \***

**from model.achieve\_child as achch**

**INNER JOIN model.child as ch on achch.kid\_id = ch.child\_id) as q**

**on q.achievement\_id = ach.achievement\_id**

**where RIGHT(ach.achievement\_desc, LENGTH(ach.achievement\_desc) - 11) =**

**CONCAT(' ', new\_task\_id));**

**else**

**achievement\_\_id = new\_achieve\_index\_id;**

**end if;**

**if exists(select issue\_dttm**

**from model.achievements**

**where achievement\_id = achievement\_\_id**

**and SUBSTRING(achievement\_desc, 1, 11) = 'first\_issue') then**

**task\_time\_dttm = (select (issue\_dttm)**

**from model.achievements**

**where achievement\_id = achievement\_\_id**

**and SUBSTRING(achievement\_desc, 1, 11) = 'first\_issue');**

**if tg\_op = 'INSERT' and new\_time\_dttm < task\_time\_dttm then**

**update model.achieve\_child**

**set kid\_id = new\_child\_id**

**where achievement\_id = achievement\_\_id;**

**update model.achievements**

**set issue\_dttm = new\_time\_dttm**

**where achievement\_id = achievement\_\_id;**

**end if;**

**else**

**insert into model.achievements(achievement\_id, achievement\_desc, issue\_dttm)**

**values (new\_achieve\_index\_id, CONCAT('first\_issue', ' ', new\_task\_id), new\_time\_dttm);**

**insert into model.achieve\_child(kid\_id, achievement\_id) values (new\_child\_id, new\_achieve\_index\_id);**

**end if;**

**new\_achieve\_index\_id = *get\_next\_achievement\_index*();**

**task\_count = 0;**

**task\_count =**

**(select count(result\_flg) from model.submission\_of\_task where child\_id = new\_child\_id and result\_flg = 1);**

**if (task\_count = 20) then**

**insert into model.achievements(achievement\_id, achievement\_desc, issue\_dttm)**

**values (new\_achieve\_index\_id, 'Impressive result, keep going', new\_time\_dttm);**

**insert into model.achieve\_child(kid\_id, achievement\_id) values (new\_child\_id, new\_achieve\_index\_id);**

**end if;**

**return new;**

**end;**

**$$**

**language plpgsql;**

**create trigger add\_achievements**

**after insert**

**on model.submission\_of\_task**

**for each row**

**execute procedure *add\_achievement*();**

**create or replace function *alter\_child\_before*() returns trigger as**

**$$**

**declare**

**new\_team\_ INTEGER;**

**new\_room\_ INTEGER;**

**class\_size INTEGER;**

**room\_size INTEGER;**

**chairs INTEGER;**

**tables INTEGER;**

**beds INTEGER;**

**begin**

**if tg\_op = 'UPDATE' and new.child\_id != old.child\_id or new.valid\_from\_dt != old.valid\_from\_dt then**

**raise exception 'You should del and ins';**

**end if;**

**new\_team\_ = (select team\_id from model.child where child\_id = new.child\_id and valid\_from\_dt = new.valid\_from\_dt);**

**new\_room\_ = (select new\_room\_ from model.child where child\_id = new.child\_id and valid\_from\_dt = new.valid\_from\_dt);**

**if (old.team\_id != new.team\_id) then**

**class\_size = (select count(child\_id) from model.child where team\_id = new\_team\_) + 1;**

**chairs = (select number\_of\_chairs\_cnt**

**from model.team as t**

**INNER JOIN model.classroom as c on c.classroom\_id = t.classroom\_id**

**where t.team\_id = new\_team\_);**

**tables = (select number\_of\_tables\_cnt**

**from model.team as t**

**INNER JOIN model.classroom as c on c.classroom\_id = t.classroom\_id**

**where t.team\_id = new\_team\_);**

**if (class\_size > chairs or class\_size > tables) then**

**raise exception 'Not enough Furniture';**

**end if;**

**end if;**

**if (old.room\_id != new.room\_id) then**

**room\_size = (select count(child\_id) from model.room where room.room\_id = new\_room\_) + 1;**

**beds = (select number\_of\_beds\_cnt**

**from model.room**

**where model.room.room\_id = new\_room\_);**

**if (room\_size > beds) then**

**raise exception 'Not enough Furniture';**

**end if;**

**end if;**

**new.valid\_from\_dt = now();**

**return new;**

**end;**

**$$**

**language plpgsql;**

**create or replace function *alter\_child\_after*() returns trigger as**

**$$**

**begin**

**old.valid\_to\_dt = now();**

**insert into model.child(child\_id, full\_name\_nm, grade\_num, valid\_from\_dt, valid\_to\_dt, gender\_txt, room\_id, team\_id)**

**values (old.child\_id, old.full\_name\_nm, old.grade\_num, old.valid\_from\_dt, old.valid\_to\_dt, old.gender\_txt,**

**old.room\_id, old.team\_id);**

**return old;**

**end;**

**$$**

**language plpgsql;**

**create trigger children\_alteration\_before**

**BEFORE update**

**on model.child**

**for each row**

**execute procedure *alter\_child\_before*();**

**create trigger children\_alteration\_after**

**AFTER update**

**on model.child**

**for each row**

**execute procedure *alter\_child\_after*();**

**update model.child**

**set team\_id = 4**

**where child\_id = 1;**

**create index on model.child (valid\_to\_dt);**

**create index on model.submission\_of\_task (result\_flg);**