Functions:

1

**create** **function** increase\_year(pname **varchar**(30))

**returns** **void** **as** **$$**

**begin**

**if**(**select** p.years\_served **from** president p **where** p."name" = pname) > 8 **then**

**raise** **exception** 'Years are already 8';

**else** **update** president

**set** years\_served = years\_served + 1

**where** "name" = pname;

**end** **if**;

**end**; **$$** **language** plpgsql;

2

**create** **function** add\_hobby(pid **int**, nhobby **varchar**(30))

**returns** **void** **as** **$$**

**begin**

**if**(**select** **count**(p.hobby) **from** pres\_hobby p **where** p.hobby = nhobby) = 0 **then**

**raise** **exception** 'Hobby does not exist';

**else**

**insert** **into** pres\_hobby(pres\_id, hobby) **values** (pid, nhobby);

**end** **if**;

**end**; **$$** **language** plpgsql;

3

**create** **function** add\_tenure(pid **int**, adminnr **int**, presid **int**, thisyear **int**)

**returns** **void** **as** **$$**

**begin**

**if**(**select** **count**(a.admin\_nr) **from** administration a **where** a.admin\_nr + 1 = adminnr) = 1 **then**

**insert** **into** administration (id, admin\_nr, pres\_id, year\_inaugurated)

**values** (pid, adminnr, presid, thisyear);

**else** **raise** **exception** 'Admin nr already exists';

**end** **if**;

**end**; **$$** **language** plpgsql;

4

**create** **function** add\_child(pid **int**, sname **varchar**(30), uchildren **int**)

**returns** **void** **as** **$$**

**begin**

**if**(**select** p.nr\_children **from** pres\_marriage p **where** p.pres\_id = pid) + uchildred <= p.nr\_children **then**

**raise** **exception** 'Cannot work';

**else** **update** pres\_marriage pm

**set** pm.nr\_children = pm.nr\_children + uchildren

**where** pm.pres\_id = pid;

**end** **if**;

**end**; **$$** **language** plpgsql;

Cursor

1

**CREATE** **OR** **replace** **FUNCTION** viceAndPres()

**RETURNS** **TABLE** (v\_pres\_name **varchar**(255)) **AS** **$$**

**DECLARE**

v\_pres **CURSOR** **FOR**

**SELECT** **distinct** vice\_pres\_name

**FROM** admin\_vpres

**order** **by** vice\_pres\_name;

pres admin\_vpres.vice\_pres\_name%**type**;

**BEGIN**

**create** **temp** **table** **if** **not** **exists** temp\_table (

**name** **varchar**(255)

);

**open** v\_pres;

**fetch** v\_pres **into** pres;

**while** **found** **loop**

**if** pres **in** (**select** **distinct** **name** **from** president **order** **by** **name**) **then**

**insert** **into** temp\_table

**values** (pres);

**end** **if**;

**fetch** v\_pres **into** pres;

**end** **loop**;

**close** v\_pres;

**return** QUERY

**select** **name**

**from** temp\_table;

**DROP** **TABLE** temp\_table;

**END**; **$$** **language** plpgsql;

**select** viceAndPres();

2

**create** **type** unm\_ret **as** (pres\_name **varchar**(20), birth\_year **int4**);

**create** **or** **replace** **function** unmarriedPres()

**returns** **setof** unm\_ret **as** **$$**

**declare**

m\_pres **cursor** **for**

**select** **distinct** id, **name**, p.birth\_year

**from** president p

**order** **by** id;

p\_id president.id%**type**;

p\_name president.**name**%**type**;

p\_birth president.birth\_year%**type**;

**begin**

**create** **temp** **table** **if** **not** **exists** notMarried\_temp (

presname **varchar**(20),

birth\_yr **int4**

);

**OPEN** m\_pres;

**FETCH** m\_pres **INTO** p\_id, p\_name, p\_birth;

**while** **FOUND** **loop**

**if** **not** p\_id **IN** (**SELECT** **DISTINCT** pres\_id **FROM** pres\_marriage pm) **THEN**

**INSERT** **INTO** notMarried\_temp

**VALUES** (p\_name, p\_birth);

**END** **IF**;

**FETCH** m\_pres **INTO** p\_id, p\_name, p\_birth;

**END** **loop**;

**CLOSE** m\_pres;

**RETURN** QUERY

**SELECT** presname, birth\_yr

**FROM** notMarried\_temp;

**drop** **table** notMarried\_temp;

**end**; **$$** **language** plpgsql;

**select** unmarriedPres();

3

**create** **type** chi\_ret **as** (pres\_name **varchar**(20), children **int4**);

**create** **or** **replace** **function** presWithChildren()

**returns** **setof** chi\_ret **as** **$$**

**declare**

p\_pres **cursor** **for**

**select** id, **name**

**from** president

**order** **by** id;

p\_name president.**name**%**type**;

p\_id president.id%**type**;

**begin**

**create** **temp** **table** **if** **not** **exists** presAndChildren\_temp (

presname **varchar**(20),

nr\_children **int4**

);

**open** p\_pres;

**fetch** p\_pres **into** p\_id, p\_name;

**while** **found** **loop**

**if** **not** p\_id **in** (**select** **distinct** pres\_id **from** pres\_marriage) **then**

**insert** **into** presAndChildren\_temp

**values** (p\_name, 0);

**else**

**insert** **into** presAndChildren\_temp

**values** (p\_name, (**select** **sum**(nr\_children) **from** pres\_marriage **where** pres\_id = p\_id **group** **by** p\_id));

**end** **if**;

**fetch** p\_pres **into** p\_id, p\_name;

**end** **loop**;

**close** p\_pres;

**return** query

**select** \*

**from** presAndChildren\_temp;

**drop** **table** presAndChildren\_temp;

**end**; **$$** **language** plpgsql;

**select** presWithChildren();

4

**create** **type** el\_ret **as** (pres\_name **varchar**(20), el\_count **int4**);

**create** **or** **replace** **function** presElections()

**returns** **setof** el\_ret **as** **$$**

**declare**

p\_pres **cursor** **for**

**select** **name**

**from** president

**order** **by** id;

p\_name president.**name**%**type**;

**begin**

**create** **temp** **table** **if** **not** **exists** presAndelection\_temp (

presname **varchar**(20),

elections **int4**

);

**open** p\_pres;

**fetch** p\_pres **into** p\_name;

**while** **found** **loop**

**insert** **into** presAndelection\_temp

**values** (p\_name, (**select** **count**(\*) **from** election **where** candidate = p\_name));

**fetch** p\_pres **into** p\_name;

**end** **loop**;

**close** p\_pres;

**return** query

**select** \*

**from** presAndelection\_temp;

**drop** **table** presAndelection\_temp;

**end**; **$$** **language** plpgsql;

**select** presElections();

Trigger

1

**create** **function** checkwin()

**returns** **trigger** **as** **$$**

**begin**

**if** (**select** **count**(e.winner\_loser\_indic) **from** election e **where** winner\_loser\_indic = 'W' **and** election\_year = **new**.election\_year) > 0 **then**

**raise** **exception** 'Number of winners invalid';

**end** **if**;

**return** **new**;

**end**; **$$** **language** plpgsql;

**create** **trigger** checkwin

**before** **insert** **or** **update** **on** election

**for** **each** **row** **execute** **procedure** checkwin();

2

?

3

**create** **function** checkHobby()

**returns** **trigger** **as** **$$**

**begin**

**if** 'TOUCH FOOTBALL' **in** (**select** ph.hobby **from** pres\_hobby ph **inner** **join** president p **on** ph.pres\_id = p.id **where** p.birth\_year < '1800') **then**

**raise** **exception** 'No touch football in those times';

**end** **if**;

**return** **new**;

**end**; **$$** **language** plpgsql;

**create** **trigger** chechHobby

**before** **insert** **or** **update** **on** pres\_hobby

**for** **each** **row** **execute** **procedure** checkHobby();

4

**create** **function** checkPres()

**returns** **trigger** **as** **$$**

**begin**

**if**((**select** **count**(a.pres\_id) **from** administration a **inner** **join** president p **on** a.pres\_id = p.id **where** a.year\_inaugurated <= p.birth\_year + 21) < (**select** **count**(pres\_id) **from** administration)) **then**

**raise** **exception** 'President too young';

**end** **if**;

**return** **new**;

**end**; **$$** **language** plpgsql;

**create** **trigger** checkPres

**before** **insert** **or** **update** **on** administration

**for** **each** **row** **execute** **procedure** checkPres();

5

**create** **function** checkYearsServed()

**returns** **trigger** **as** **$$**

**begin**

**if**(**new**.years\_served < **old**.years\_served) **then**

**update** president

**set** years\_served = **old**.years\_served;

**raise** **notice** 'Cannot decrease years served';

**end** **if**;

**return** **new**;

**end**; **$$** **language** plpgsql;

**create** **trigger** checkYearsServed

**after** **update** **of** years\_served **on** president

**for** **each** **row** **execute** **procedure** checkYearsServed();