EPAM Systems, RD Dep., RD Dep.

POSTGRESQL DB FOR DWH AND ETL BUILDING

Transaction

1. Create a table called employee with columns id serial, name varchar, status varchar.
2. Replicate the example given in the lecture with the code below:

-- first transaction
begin;
 	
select txid_current();
 	
insert into public.employee ("name", status)
values ('Alice', 'Not fired');
 	
select *, xmin, xmax
from public.employee e;
 	
commit;
	-- first transaction
	begin;
	
	select *, xmin, xmax from public.employee e;
	
	commit;

-- second transaction
begin;
 	
select *, xmin, xmax
from public.employee e;
 	
select *, xmin, xmax
from public.employee e;
 	
commit;
	-- second transaction
	begin;
	
	select txid_current();
	
	delete from public.employee
	where id = 1;
	
	select *, xmin, xmax
	from public.employee e;
	
	commit;

insert into public.employee ("name", status) values ('Alice', 'Not fired');	

-- third transaction
begin;
 	
select *, xmin, xmax
from public.employee e;
 	
select *, xmin, xmax
from public.employee e;
 	
commit;
	-- third transaction
	begin;
	
	select txid_current();
	
	update public.employee
	set status = 'Fired'
	where id = 2;
	
	select *, xmin, xmax
	from public.employee e;

	commit;

3. Run the command set transaction isolation level repeatable read.
4. Check your current isolation level in each session with show transaction isolation level.
5. Recreate employee table and redo the second task but modify the code so that select statements would now include **cmin** and **cmax** system columns. What changed?
6. * Try to cause a *serialization anomaly* on the employee table (add more data if necessary). Change your isolation level to serializable and try to cause *serialization anomaly* one more time. What happened?
7. * Set your isolation level to read committed. Try to cause a *lost update* database anomaly on the employee table (add more data if necessary). What happened? What do you think are the downsides of the approach that Postgres took to handle this anomaly?