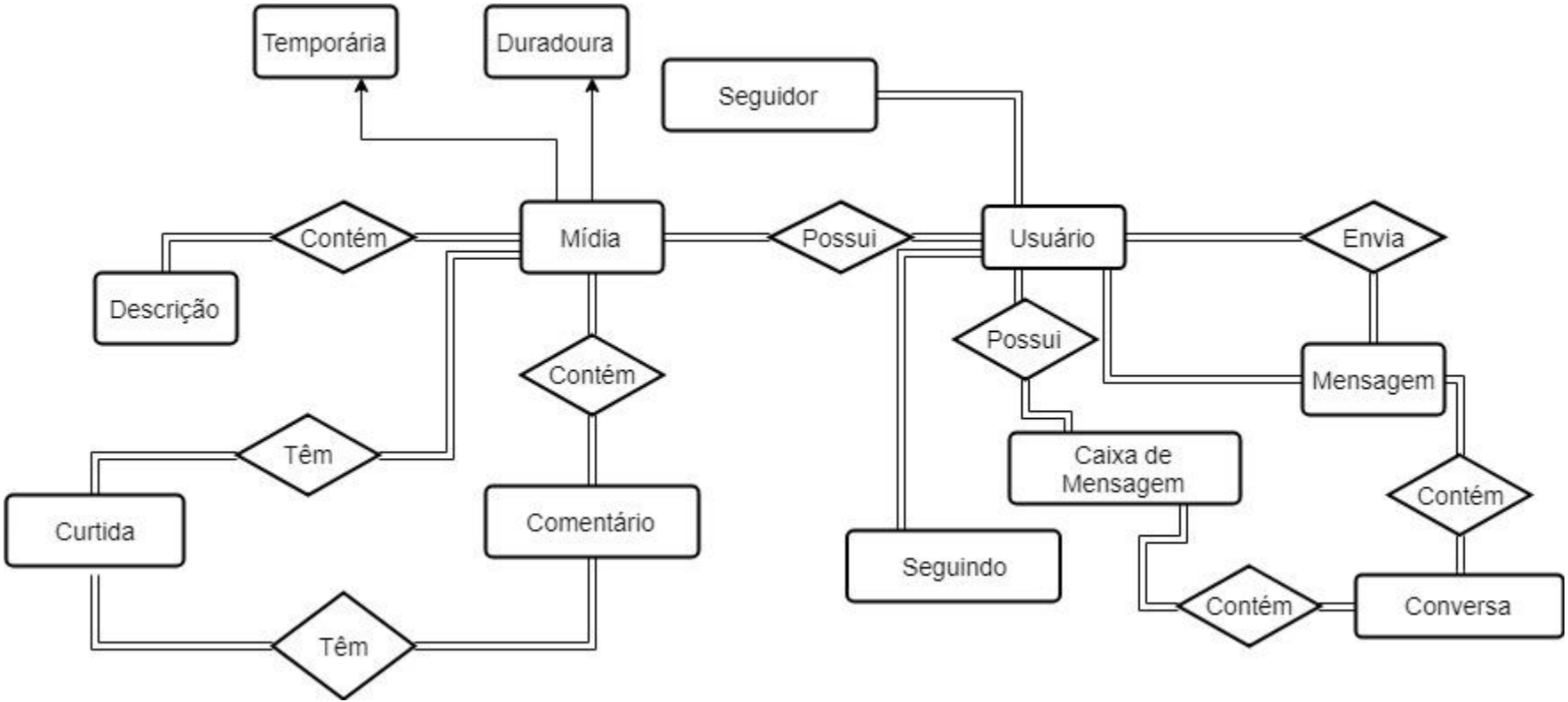
# Trabalho de banco de dados 2

Eduardo Zimelewicz
Mauro Sergio Lopes

# Banco baseado no Instagram

- Rede social de compartilhamento de mídias
- Mídias podem ser permanentes ou temporárias
- Usuários conversam entre si
- Usuários podem ser privados
- Usuários podem ter listas de usuários bloqueados
- Usuários possuem lista de usuários seguidores e usuários que seguem
- Usuário deve autorizar início de conversa com outro usuário que não segue

# Modelo ER


Temporária
Duradoura
Seguidor
Contém
Mídia
Possui
Usuário
Envia
Descrição
Possui
Contém
Mensagem
Têm
Caixa de Mensagem
Contém
Curtida
Comentário
Seguindo
Contém
Conversa
Têm

# Trigger 1 - bloquear_usuario

```
create trigger bloqueia after insert on bloqueado
        for each row execute procedure bloquear_usuario();


create or replace function bloquear_usuario() returns trigger as $$
begin
        delete from seguidor where
                seguidor.usuario_id = new.usuario_id and seguidor.seguidor_id = new.bloqueado_id;
        delete from seguindo where
                seguindo.usuario_id = new.bloqueado_id and seguindo.seguindo_id = new.usuario_id;
        return new;
end;
$$ language plpgsql;
```

# Trigger 2 - create_message_box

```
create trigger new_user after insert on usuario
        for each row execute procedure create_message_box();

create or replace function create_message_box() returns trigger as $$
begin
        insert into caixa_de_mensagem values (default, new.usuario_id);
        return new;
end;
$$ language plpgsql;
```

# Trigger 3 - seguir_usuario

```
create trigger seguir after insert on seguindo
        for each row execute procedure seguir_usuario();

create or replace function seguir_usuario() returns trigger as $$
declare
        rec_usuario usuario;
begin
        select * from usuario into rec_usuario where usuario_id = new.seguindo_id;
        if (rec_usuario.privado = false) then
                insert into seguidor values(new.seguindo_id, new.usuario_id, current_timestamp, false);
                return new;
        end if;
        insert into seguidor values(new.seguindo_id, new.usuario_id, current_timestamp, true);
        return new;
end;
$$ language plpgsql;
```

# Trigger 4 - aceitar_request

```
create trigger aceitar after update of pendente on seguidor
        for each row execute procedure aceitar_request();


create or replace function aceitar_request() returns trigger as $$
begin
        update seguidor set time_stamp = current_timestamp
                where seguidor.usuario_id = new.usuario_id and seguidor.seguidor_id = new.seguidor_id;
        return new;
end;
$$ language plpgsql;
```

# Trigger 5 - stories_check

```
create trigger stories after update on midia
        for each row execute procedure stories_check();

create or replace function stories_check() returns trigger as $$
declare
        midia_rec midia;
        cursor_midia cursor for select * from midia;
begin
        open cursor_midia;
        loop
                fetch cursor_midia into midia_rec;
        exit when not found;

        if ((age(current_timestamp,midia_rec.time_stamp) >= interval '24 hours')
                        and (midia_rec.duradoura = false)) then
                delete from midia where current of cursor_midia;
        end if;

        end loop;
        close cursor_midia;
        return new;
end;
$$ language plpgsql;
```

# Trigger 6 - conversa_request

```
create trigger req_conversa after insert on conversa
        for each row execute procedure conversa_request();

create or replace function conversa_request() returns trigger as $$
declare
        user_rec usuario;
begin
        select * from usuario into user_rec
                where usuario.usuario_id = new.receptor_id;

        if(user_rec.privado = true) then
                insert into mensagem values (default, new.conversa_id, '', true);
                return new;
        end if;
end;
$$ language plpgsql;
```

# Trigger 7 - accept_message_request

```
create trigger accept_message after update of bloqueado on conversa
        for each row execute procedure accept_message_request();

create or replace function accept_message_request() returns trigger as $$
declare
begin
        delete from mensagem where
                mensagem.conversa_id = old.conversa_id and mensagem.request = true;
        return new;
end;
$$ language plpgsql;
```

# Trigger 8 - checa_mensagens_nao_lidas

```
create trigger atualiza_caixa_de_mensagens after insert or update on mensagem
        for each row execute procedure checa_mensagens_nao_lidas();

create or replace function checa_mensagens_nao_lidas() returns trigger as $$
declare
        qtd_conv_nao_lidas integer;
begin

        if(TG_OP = 'INSERT') then
                update conversa set possui_mensagem_nova = true
                        where conversa.conversa_id = new.conversa_id;
        elsif (TG_OP = 'UPDATE') THEN
                update conversa set possui_mensagem_nova = false
                        where conversa.conversa_id = new.conversa_id;
        end if;

        select count(*) into qtd_conv_nao_lidas
                from conversa, caixa_de_mensagem
                where conversa.caixa_id = caixa_de_mensagem.caixa_id
                        and conversa.possui_mensagem_nova = true;

        update caixa_de_mensagem set qtd_conversas_nao_lidas = qtd_conv_nao_lidas;
        return new;
end;
$$ language plpgsql;
```

# Trigger 9 - verifica_tamanho_da_mida

```
create trigger verifica_tamanho_da_mida before insert on midia
        for each statement execute procedure checa_tamanho_midia();

create or replace function checa_tamanho_midia() returns trigger as $$
declare
begin
        if(new.tipo = 'gif') then
                if(new.tamanho > 5000) then
                        raise exception 'Tamanho de gif não suportado!';
                end if;
        elsif(new.tipo = 'imagem') then
                if(new.tamanho > 10000) then
                        raise exception 'Tamanho de imagem não suportado!';
                end if;
        elsif(new.tipo = 'video') then
                if(new.tamanho > 50000) then
                        raise exception 'Tamanho de video não suportado!';
                end if;
        end if;
        return new;
end;
$$ language plpgsql;
```