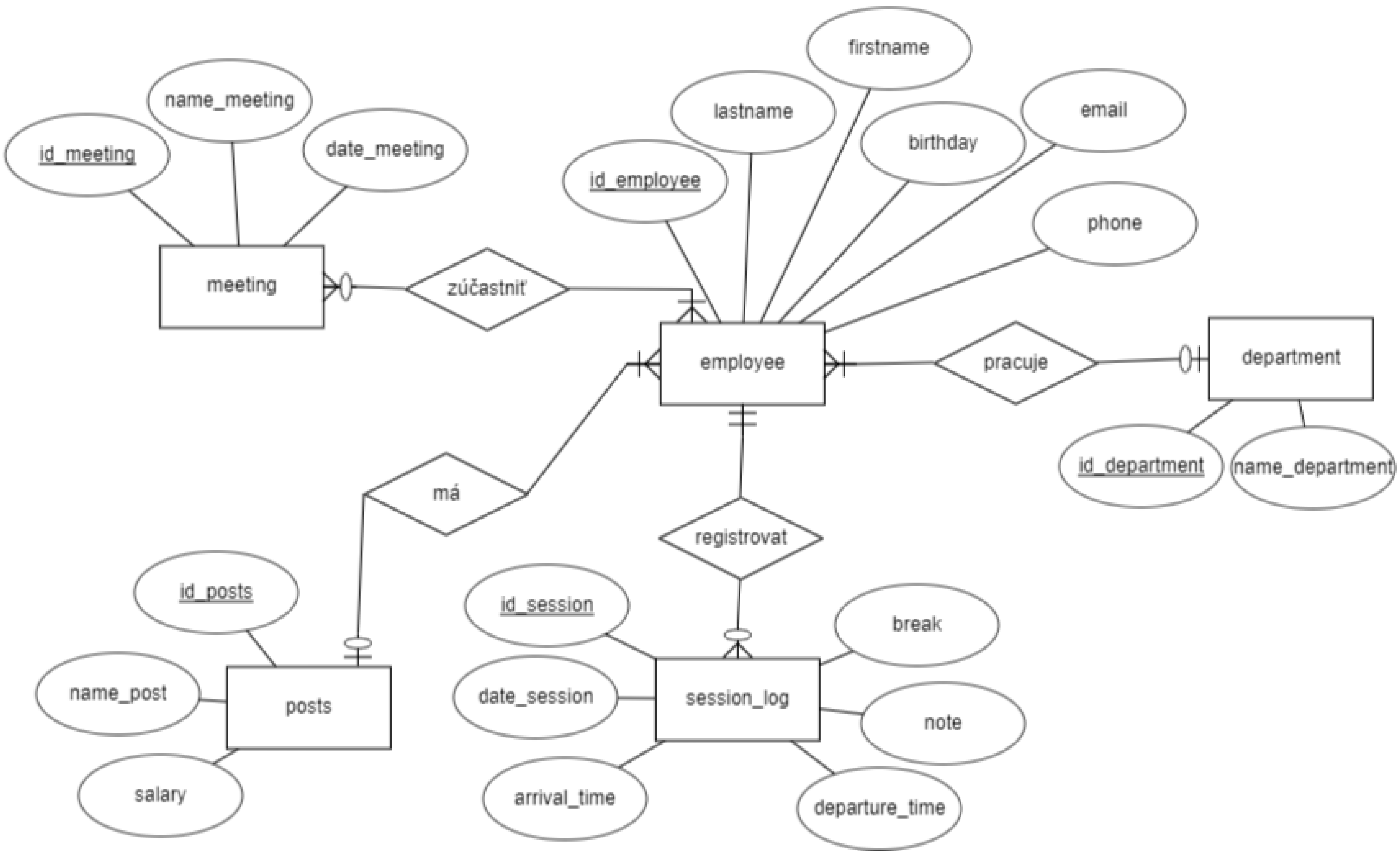
TECHNICKÁ UNIVERZITA KOŠICE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

DATABÁZA
Evidencia dochádzky zamestnancov

odovzdávka č.3

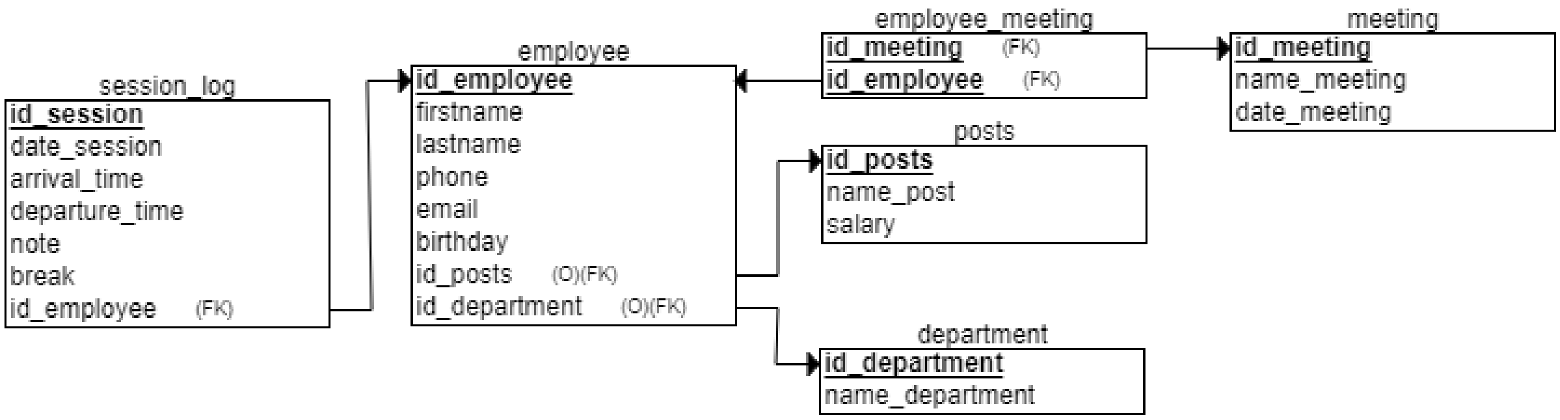
Ladislav Dono

Entitno-relačný model



Entities	Link	Link type	Link content
employee-department	pracuje	M-1	mnoho zamestnancov môže pracovať v jednom oddelení
employee-posts	má	M-1	mnohí zamestnanci môžu pracovať na rovnakej pozícii
employee-session_log	registrovat	1-M	jeden zamestnanec môže mať veľa záznamov
employee-meeting	zúčastniť	M-N	mnoho používateľov sa môže zúčastniť mnohých udalostí

Logický relačný model



SQL skript

```
--department
create table department
(
    id_department   serial not null unique primary key,
    name_department varchar(50) not null
);

--posts
create table posts
(
    id_posts  serial not null unique primary key,
    name_post varchar(50) not null,
    salary    integer     not null
);

--employee
create table employee
(
    id_employee      serial not null unique primary key,
    firstname        varchar(15) not null,
    lastname         varchar(15) not null,
    phone            varchar(15) not null,
    email            varchar(50) not null CHECK (email ~* '^[A-Za-z0-9._%-]+@[A-Za-z0-9.- ]+.[a-zA-Z]{2,3}$'),
    birthday         date        not null,
    id_posts_fk      integer     not null,
    id_department_fk integer     not null,
    foreign key (id_posts_fk) references posts(id_posts),
    foreign key (id_department_fk) references department(id_department)

);

--session_log
create table session_log
(
    id_session     serial not null unique primary key,
    date_session   date    not null,
    arrival_time   time    not null,
    departure_time time    not null,
    break          time,
    note           varchar(100),
    id_employee_fk integer not null,
    foreign key (id_employee_fk) references employee(id_employee)
);

--meeting
create table meeting
(
    id_meeting     serial not null unique primary key,
    name_meeting   varchar(100)    not null,
    date_meeting   date    not null
);

--employee_meeting
create table employee_meeting
(
    id_meeting_fk    int not null,
    id_employee_fk   int not null,
    foreign key (id_meeting_fk) references meeting(id_meeting),
    foreign key (id_employee_fk) references employee(id_employee)

);
```

SQL skript

```
insert into department(id_department, name_department)
values (1, 'Directorate'),
      (2, 'Warehouse'),
      (3, 'Accounting'),
      (4, 'Personnel Department'),
      (5, 'IT Department');

insert into posts(id_posts, name_post, salary)
values (1, 'Administrator', 4100),
      (2, 'Director', 5000),
      (3, 'Analyst', 3200),
      (4, 'Secretary', 1900),
      (5, 'Manager', 2000),
      (6, 'Software engineer', 3200),
      (7, 'Sales', 1000);

insert into employee(id_employee, firstname, lastname, phone, email, birthday, id_posts_fk, id_department_fk)
values (1, 'Miroslan', 'Tan', '095995300', 'Miroslan@mail.com', '2000-02-01', 2 ,1),
      (2, 'Maximilián', 'Samson', '095995356', 'Maximilián@mail.com', '1999-05-06', 1 ,1),
      (3, 'Móric', 'Čaplovič', '095995365', 'Móric@mail.com', '2001-01-08', 3 ,1),
      (4, 'Koloman', 'Bielik', '095995874', 'Koloman@mail.com', '1997-09-02', 4 ,3),
      (5, 'Bruno', 'Bella', '095995436', 'Bruno@mail.com', '1994-11-19', 5 ,4),
      (6, 'Blahoslav', 'Mojžiš', '095995909', 'Blahoslav@mail.com', '1998-12-16', 6 ,5),
      (7, 'Ervín', 'Puškáš', '095995778', 'Ervín@mail.com', '1993-02-11', 7 ,2);

insert into meeting(id_meeting, name_meeting, date_meeting)
values (1,'Operational meeting', '2022-04-01'),
      (2,'Innovative Assembly', '2022-04-02'),
      (3,'Strategic Meeting', '2022-04-03'),
      (4,'Informal meeting', '2022-04-04'),
      (5,'Meeting with partners', '2022-04-05');

insert into employee_meeting(id_meeting_fk, id_employee_fk)
values (1,1),
      (1,2),
      (1,3),
      (1,4),
      (2,1),
      (2,2),
      (3,1),
      (3,6),
      (3,5),
      (4,5),
      (4,7),
      (4,4),
      (4,3),
      (5,1),
      (5,2);

insert into session_log(id_session, date_session, arrival_time, departure_time, break, note, id_employee_fk)
values (1, '2022-04-01', '12:00:00', '18:00:00', null, null, 1),
      (3, '2022-04-01', '08:00:00', '18:00:00', '00:30:00', null, 2),
      (4, '2022-04-01', '08:00:00', '17:00:00', '00:10:00', 'Meškal 13 minút', 3),
      (5, '2022-04-01', '10:00:00', '19:00:00', '00:10:00', null, 4),
      (6, '2022-04-02', '11:00:00', '19:00:00', null, 'Meškal 10 minút', 7),
      (7, '2022-04-02', '12:00:00', '19:00:00', '00:15:00', null, 5),
      (8, '2022-04-02', '09:00:00', '17:00:00', '00:30:00', null, 2),
      (9, '2022-04-03', '08:00:00', '17:00:00', '00:20:00', 'Prišiel som bez preukazu', 6);
```

SQL skript

```
--zobraziť zamestnancov, ktorí sa narodili pred rokom 1999
create view name_birthday as
select e.firstname || ' ' || e.lastname as name, e.birthday
from employee e
where extract(year from e.birthday) <1999
order by e.birthday desc;

--zobrazenie názvu schôdze a dátumu schôdze, kde je názov schôdze 'Strategic'
create view name_date_meeting as
select m.name_meeting, m.date_meeting
from meeting m
where m.name_meeting like 'Strategic%';

--ukázať zamestnancom ich postavenie a plat, Zoradiť podľa platu
create view name_post_salary as
select e.firstname || ' ' || e.lastname as name, p.name_post, p.salary
from employee e
inner join posts p on e.id_posts_fk = p.id_posts
order by p.salary desc ;

--ukázať zamestnancom ich pozíciu oddelenia, kde pracujú, ako aj meno a dátum stretnutia
create view name_department_post_meeting as
select e.firstname || ' ' || e.lastname as name, d.name_department || ' ' || p.name_post as department_post,
    m.name_meeting || ' ' || m.date_meeting as meeting
from employee e
left outer join department d on d.id_department = e.id_department_fk
left outer join posts p on p.id_posts = e.id_posts_fk
right outer join employee_meeting em on e.id_employee = em.id_employee_fk
right outer join meeting m on m.id_meeting = em.id_meeting_fk
where m.name_meeting = 'Operational meeting';

--zobraziť úplné informácie o príchode do práce
create view name_department_post_session as
select e.firstname || ' ' || e.lastname as name, d.name_department || ' ' || p.name_post as department_post,
    sl.date_session, sl.note
from employee e
full outer join session_log sl on e.id_employee = sl.id_employee_fk
full outer join posts p on p.id_posts = e.id_posts_fk
full outer join department d on d.id_department = e.id_department_fk;

--zobraziť minimálny, maximálny a priemerný plat
create view salary_info as
select min(p.salary) as min_salary,
    max(p.salary) as max_salary,
    round(avg(p.salary), 2) as avg_salary
from posts p;

--zobraziť počet návštev zamestnancov úradu v '2022-04-01'
create view count_session as
select count(*)
from session_log sl
where sl.date_session = '2022-04-01';

---------------Finálne odovzdávanie zadania---------------
----1 pohľad s použitím množinových operácií
--zobraziť zamestnancov oddelenia 'Directorate' ktorí majú plat viac ako 4000
--a oddelenia 'IT Department', 'Accounting','Personnel Department' ktorí majú plat viac=  ako 2000
create view employeesDepPost as
select e.lastname ||' '|| e.firstname as employee_, d.name_department||' '|| p.name_post as department_post, p.salary as salary_
from employee e
join department d on d.id_department = e.id_department_fk
join posts p on p.id_posts = e.id_posts_fk
where name_department in('Directorate') and salary >4000
union
select e.lastname ||' '|| e.firstname as employee_, d.name_department||' '|| p.name_post as department_post, p.salary as salary_
from employee e
join department d on d.id_department = e.id_department_fk
join posts p on p.id_posts = e.id_posts_fk
where name_department in('IT Department', 'Accounting','Personnel Department') and salary >=2000
order by salary_ desc;
```

SQL skript

```
--zobraziť zamestnancov, ktorí boli na pracovisku 01 a 02
create view employeesWork as
select e.id_employee, concat(e.lastname, ' ', e.firstname) as name
from employee e
join session_log sl on e.id_employee = sl.id_employee_fk
where sl.date_session = '2022-04-01'
INTERSECT
select e.id_employee, concat(e.lastname, ' ', e.firstname) as name
from employee e
join session_log sl on e.id_employee = sl.id_employee_fk
where sl.date_session = '2022-04-02';
----2 pohľady s použitím vnorených poddopytov.
--zobraziť zamestnancom, ktorí boli najčastejšie na stretnutiach s '2022-04-03' - '2022-04-05'
create view topEmployeeMeeting as
select e.firstname || ' '||e.lastname as name, e.phone, e.birthday, e.email
from employee e
where id_employee in (
   select top.id_e
   from ( select id_employee_fk as id_e, count(id_employee_fk) as ccoutn
       from employee_meeting
       join meeting m on m.id_meeting = employee_meeting.id_meeting_fk
       where date_meeting between '2022-04-03' and '2022-04-05'
       group by id_employee_fk
       order by ccoutn desc
       limit 3
    ) as top );
--zobraziť zamestnancom, ktorí nikdy ne meškali na prácu
create view employeeNieMeskal as
select e.firstname || ' '||e.lastname as name, e.phone, e.birthday, e.email
from employee e
where not exists(
   select *
   from session_log sl
   where sl.note like '%Meškal%' and e.id_employee = sl.id_employee_fk )
order by name;

----skript na vytvorenie triggeru/triggerov, ktoré budú implementovať autoinkrementáciu umelých kľúčov
--trigger pre tabuľky department z autoinkrementácim id_department
drop trigger autoIncDepartment on department;
create trigger autoIncDepartment before insert on department
   for each row
   execute procedure func_autoinc();

create or replace function func_autoinc() returns trigger as
$$
   declare
      max_id_department int;
   begin
      select max(id_department) into max_id_department from department;

      --skontrolujme, či je tabuľka prázdna
      if (max_id_department) is null and new.id_department is null then
         new.id_department = 1;
      end if;
      --skontrolujme, či 'new.id_department' prázdny
      if new.id_department is null then
         new.id_department = (max_id_department)+1;
      end if;
      --skontrolujme, či 'new.id_department' uz je v tabulke
      if( select id_department from department
      where new.id_department  in(id_department)) is not null then
         new.id_department = (max_id_department)+1;
      end if;
      return new;
   end;
$$
language plpgsql;

insert into department(id_department, name_department)  values (1,'Equipment');
insert into department(name_department) values ('Recreation');
```

SQL skript

```
----skript na aspoň dva zmysluplné triggre (okrem triggerov na autoinkrementáciu)
--trigger pre view employee_meeting_view ktorý pridá nový riadok do tabuľky employee_meeting
drop view employee_meeting_view;
create view employee_meeting_view as
select concat(e.firstname, ' ', e.lastname) as employee_name, e.email, m.date_meeting, m.name_meeting
from employee e
left join employee_meeting em on e.id_employee = em.id_employee_fk
left join meeting m on m.id_meeting = em.id_meeting_fk
order by m.date_meeting;

create or replace function add_emp_meet_func() returns trigger as
$$
    declare
        id_e int;
        id_m int;
    begin
        select id_employee into id_e from employee where email = new.email;
        select id_meeting into id_m from meeting where date_meeting = new.date_meeting and name_meeting= new.name_meeting;
        insert into employee_meeting (id_employee_fk,id_meeting_fk) values (id_e,id_m);
        return new;
    end;
$$
language plpgsql;
create trigger add_emp_meet instead of insert on employee_meeting_view
    for each row
    execute procedure add_emp_meet_func();
insert into employee_meeting_view(employee_name, email, date_meeting, name_meeting)
values ('Maximilián Samson', 'Maximilián@mail.com', '2022-04-04', 'Informal meeting');

--trigger pre view employee_session_view ktorý bude aktualizovať riadok tabuľky session_log
create view employee_session_view as
select concat(e.firstname, ' ', e.lastname) as employee_name, e.email, sl.date_session, sl.arrival_time, sl.note
from employee e
full join session_log sl on e.id_employee = sl.id_employee_fk;

create or replace function udp_employee_session_func() returns trigger as
$$
    declare
        id_e int;
    begin
        select id_employee into id_e from employee where email = new.email;
        update session_log set date_session = new.date_session,
                              arrival_time = new.arrival_time,
                              note = new.note
        where id_employee_fk = id_e and date_session = old.date_session;
        return new;
    end;
$$
language plpgsql;
create trigger udp_employee_session instead of update on employee_session_view
    for each row
    execute procedure udp_employee_session_func();

update employee_session_view set note = 'Meškal 1 minútu'  where email = 'Maximilián@mail.com' and date_session = '2022-04-01';

----skript na aspoň jednu storovanú procedúru a jednu funkciu
---1--- -procedúra ktora zvýši alebo zníži platy všetkých zamestnancov o určité percento
create or replace procedure udp_salary(procent int, direction varchar(10)) as
$$
    begin
        if direction = 'increase' then
            update posts set salary = salary + salary*procent/100;
        elseif direction = 'decrease' then
            update posts set salary = salary - salary*procent/100;
        end if;
    end;
$$
language plpgsql;

call udp_salary(1, 'decrease');
call udp_salary(2, 'increase');
```

SQL skript

```
--funkcia, ktorá vráti tabuľku s oddeleniami a pozíciami, ktoré pracovali v určitý deň
create or replace function get_post(_date_session date)
returns table(
                department varchar,
                post varchar
             ) as
$$
    begin
        return query select
        d.name_department, p.name_post
        from
        session_log sl
        join employee e on e.id_employee = sl.id_employee_fk
        join department d on d.id_department = e.id_department_fk
        join posts p on p.id_posts = e.id_posts_fk
        where sl.date_session = _date_session
        order by d.name_department;
    end;
$$
language plpgsql;

select get_post('2022-04-01');
```