International Space Parking Station

Popis

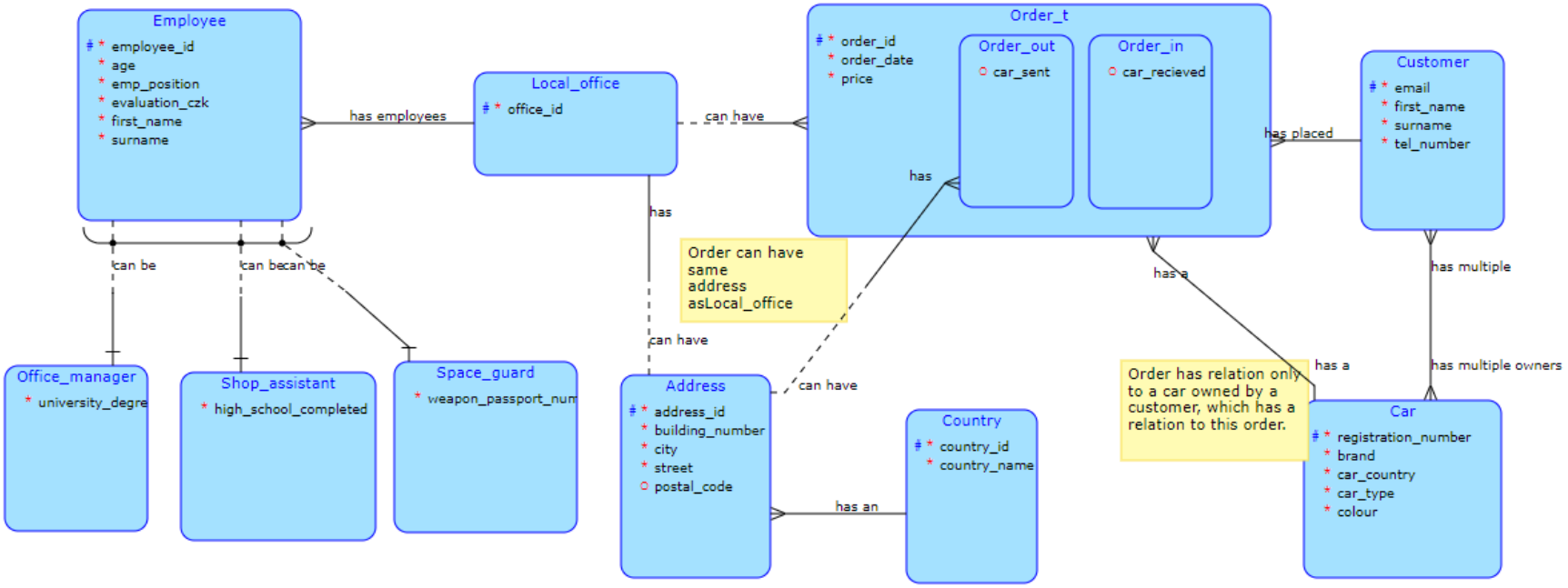
International Space Parking Station (ISPS) je služba pro zákazníky z celého světa. Mezinárodní vesmírná parkovací stanice nabízí službu parkoviště.

Prvním krokem je registrace zákazníka na jedné z poboček na Zemi. Pobočky jsou ve všech světových zemích a proto si firma ISPS vede registr svých poboček (a jejich adres) a zaměstnanců. U zaměstnanců povinně evidujeme věk, výplatu (tu známe okamžitě při nástupu), jméno a příjmení. Pro zaměstnance existují tři pracovní pozice. Manažer pobočky, který musí mít vysokoškolské vzdělání. Pracovník na pobočce, který musí mít maturitu (v jiných zemích ekvivalent vzdělání na této úrovni). A vesmírný hlídač, který musí mít zbrojní průkaz.

Zákazník během své registrace předá informace o své osobě. Povinně evidujeme email (podle kterého je identifikujeme), jméno, příjmení a telefonní číslo. Zákazník zároveň registruje vozidlo u kterého povinně evidujeme registrační značku (podle které identifikujeme), značku, zemi původu, barvu a typ. Zákazník může mít více než jedno vozidlo a vozidlo může vlastnit více lidí (např. auto v rodině používá manžel, manželka a syn).

Každý zákazník vytváří objednávku. Objednávky jsou dvojího typu, objednávka při které zákazník předá auto a my ho uložíme (order_in) a objednávka kdy vracíme auto z vesmírné stanice zákazníkovi na adresu (order_out). U objednávek evidujeme datum objednávky, cenu objednávky a kontrolujeme zda auto jsme převzali (order_in) nebo odeslali zpět (order_out). Každá objednávka order_out má adresu. U adres povinně evidujeme číslo popisné, ulici, město, kraj a zemi.

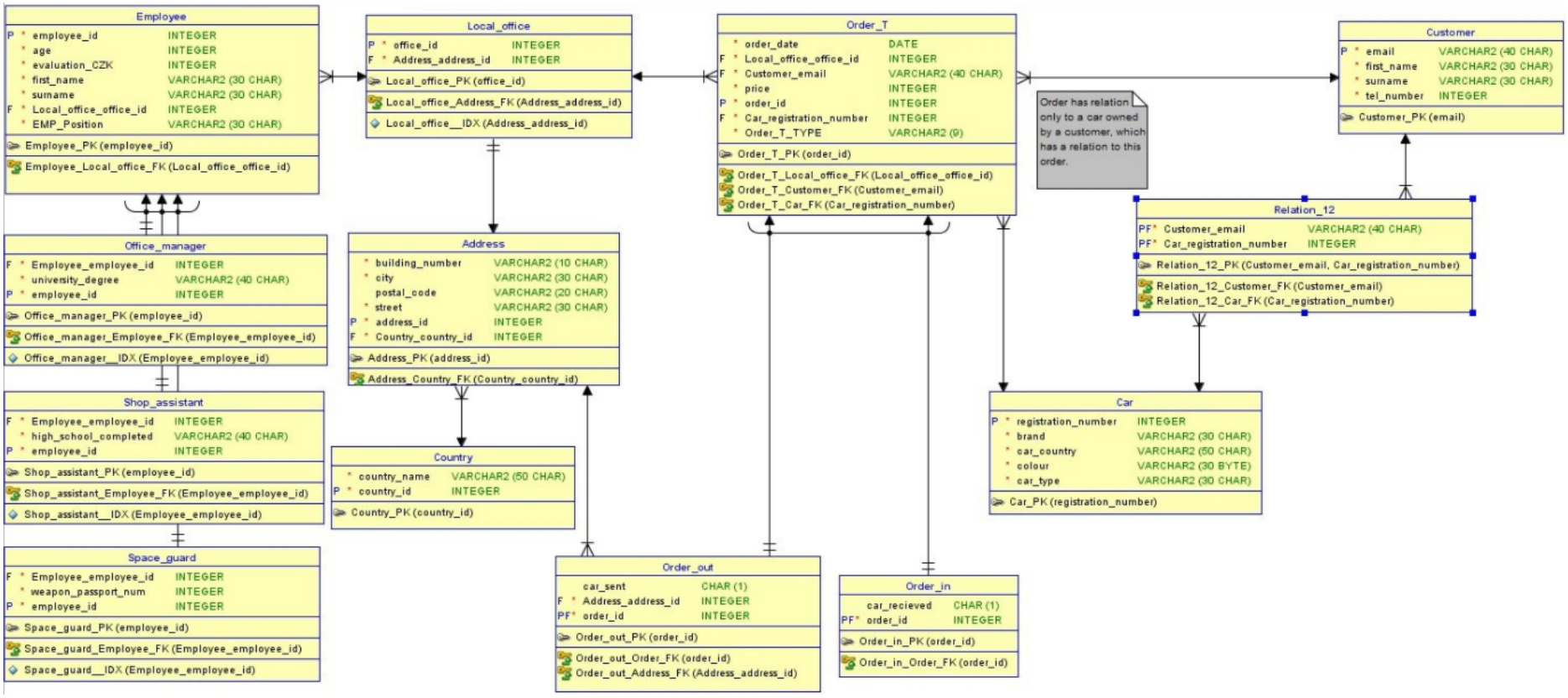
Konceptuální schéma



Diskuze smyček

- **Smyčka (Address, Local_office, Order(Order_out)):**
  Smyčka nevadí, objednávka může být na stejnou adresu kde je Local_office.
- **Smyčka (Order, Customer, Car):**
  Smyčka nevadí, protože každé auto musí mít majitele(zákazníka) i objednávku.
  Každá objednávka musí mít zákazníka i auto. A zákazník musí mít objednávku a auto.
  To že zákazník a objednávka mají stejné auto nevadí, pouze to znamená že ta objednávka patří k tomu zákazníkovi.
  Daná objednávka se váže pouze k takovému autu, které vlastní zákazník navázaný na tuto objednávku.

Relační schéma



Dotazy

D1	Ukaž mi česká města v databázi.
RA	`{country(country.country_name='Ceska republika')[country_id->k1][k1=k2]address[country_country_id->k2, city]}[city]`

SQL	`select distinct city`
	`from`
	`(`
	`  (select distinct country_id as k1`
	`  from country`
	`  where country.country_name = 'Ceska republika')`
	`  join`
	`  (select distinct country_country_id as k2, city`
	`  from address) on k1 = k2`
	`);`

D2	ID poboček, které nemají adresu "Praha".

RA	`{{local_office}`
	`\`
	`{address(city='Praha')[Address_id=address_address_id>local_office}}[office_id]`

SQL	`select distinct office_id`
	`from(`
	`  (select distinct * from local_office)`
	`  minus`
	`  (select distinct office_id, address_address_id from local_office`
	`  join ( select distinct * from address where city = 'Praha') on address_id=address_address_id)`
	`);`

D3	Ukaž mi majitele aut, kteří mají pouze auta typu 'Sedan'.

RA	{
	{ car(car_type='Sedan') [registration_number=car_registration_number> relation_12 [customer_email=email> customer }
	\
	{ car(car_type!='Sedan')
	[registration_number=car_registration_number> relation_12
	[customer_email=email> customer }
	}

SQL	select * from customer c where exists (select * from relation_12 r join car on r.car_registration_number = car.registration_number where (r.customer_email=c.email) and (car.car_type = 'Sedan')) and not exists (select * from relation_12 r join car on r.car_registration_number = car.registration_number where (r.customer_email=c.email) and (car.car_type != 'Sedan'));

D4	Zákazníci, kteří měli objednávku na všech pobočkách.
RA	{{{order_t[customer_email,local_office_office_id->office_id]÷local_office[office_id]} * customer}[customer_email]}[customer_email = email> customer
SQL	SELECT * FROM customer c WHERE NOT EXISTS(SELECT office_id FROM local_office lo WHERE NOT EXISTS (SELECT customer_email FROM order_t o WHERE o.customer_email = c.email AND o.local_office_office_id = lo.office_id));

D5	Kontrola výsledku dotazu z kategorie D1.
SQL	(SELECT DISTINCT office_id, address_address_id FROM local_office) MINUS (SELECT DISTINCT office_id, address_address_id FROM local_office RIGHT JOIN (SELECT DISTINCT * FROM order_t RIGHT JOIN (SELECT email FROM customer c WHERE NOT EXISTS (SELECT office_id FROM local_office lo WHERE NOT EXISTS (SELECT customer_email FROM order_t o WHERE o.customer_email = c.email AND o.local_office_office_id = lo.office_id))) ON email = customer_email) ON local_office_office_id = office_id);

D6	Všechny možné kombinace zaměstnanců a poboček.
RA	{Employee × Local_office}[first_name, surname, office_id]
SQL	(SELECT DISTINCT first_name, surname, office_id FROM Employee CROSS JOIN Local_office);

D7	Seřazená unikátní jména zákázníků kteří zaplatili za objednávky typu order_in více než 40000.
SQL	select first_name from customer c join order_t o on o.customer_email=c.email where o.order_t_type='Order_in' group by first_name having sum(o.price)>40000 order by first_name

D8	Vyber zákazníky a ukaž mi na kterých pobočkách vytvořili objednávky.

SQL	`select distinct c.first_name as customer, o.local_office_office_id as local_office`
	`from customer c`
	`full outer join order_t o on c.email=o.customer_email`
	`order by customer`

D9	Ukaž mi zákazníka který zaplatil 10566.

SQL	`select first_name`
	`from customer`
	`join order_t on customer_email = email`
	`where price=10566;`
	
	`select distinct first_name`
	`from customer`
	`right join (`
	`(select distinct order_date, local_office_office_id, customer_email, price, order_id, car_registration_number, order_t_type`
	`from customer`
	`natural join order_t`
	`)`
	`minus`
	`(select distinct order_date, local_office_office_id, customer_email, price, order_id, car_registration_number, order_t_type`
	`from customer`
	`natural join`
	`(select distinct *`
	`from order_t`
	`where price != 10566`
	`)`
	`)`
	`)`
	`on CUSTOMER_EMAIL = email;`
	
	`select distinct first_name`
	`from customer`
	`right join`
	`(select distinct *`
	`from order_t`
	`where price = '10566'`
	`)`
	`on customer_email = email;`

D10	Ukaž mi počet objednávek zákazníků, seřazeno dle počtu sestupně.

SQL	`select z.first_name, (select count(1) from ORDER_t o where o.CUSTOMER_EMAIL=z.EMAIL)`
	`as POCET_OBJ`
	`from CUSTOMER z`
	`order by 2 desc;`

D11	Zaměstnanci, jejichž jméno začíná na "A" spolu se zaměstnanci, kteří pracují na pobočce číslo 15.

SQL	`select z.first_name, p.office_id as local_office`
	`from employee z`
	`  join local_office p on p.office_id=z.local_office_office_id`
	`where z.first_name like 'A%'`
	`union`
	`select z.first_name, p.office_id as local_office`
	`from employee z`
	`  join local_office p on p.office_id=z.local_office_office_id and`
	`p.office_id=15;`

D12	Ukaž mi zaměstance kteří se jmenují Alan a zároveň mají příjmení Sadílek.

RA	`employee(first_name='Alan')`
	`∩`
	`employee(surname='Sadílek')`

SQL	`(SELECT DISTINCT *`
	` FROM employee`
	` WHERE first_name = 'Alan') INTERSECT`
	`(SELECT DISTINCT *`
	` FROM employee`
	` WHERE surname = 'Sadílek');`

D13	Vytvoř pohled a ukaž mi počet objednávek od roku 2000.

SQL	`create view V_POCET_SLUZEB as`
	`  select extract(YEAR from o.ORDER_DATE) as ROK, count(1) as POCET`
	`  from order_t o`
	`  group by extract(YEAR from o.ORDER_DATE);`
	
	`select * from V_POCET_SLUZEB`
	`where rok >1999`
	`order by ROK;`
	
	`drop view V_POCET_SLUZEB;`

D14	Emil Volf se stal vlastníkem všech BMW.

SQL	`INSERT INTO relation_12`
	`(customer_email, car_registration_number) SELECT`
	`'Emil.Volf@seznam.cz', registration_number FROM car WHERE brand='BMW'`
	`;`
	
	`rollback;`

D15	Všem objednávkám Nely Veselkové sniž cenu o 10%.

SQL	`update ORDER_T set PRICE=PRICE*0.9`
	`where ORDER_ID in`
	`  (select o.ORDER_ID`
	`  from ORDER_T o`
	`  join CUSTOMER c on c.email=o.customer_email`
	`  and c.first_name='Nela'`
	`  and c.surname='Veselková'`
	`);`
	
	`rollback;`

D16	Smaž všechna auta zákazníka 'Emil Volf'.

SQL	`delete from relation_12`
	`where car_registration_number in (`
	`  select distinct car_registration_number`
	`  from relation_12 r`
	`  join customer c on r.customer_email = c.email`
	`  where c.first_name = 'Emil' and surname = 'Volf'`
	`);`
	`rollback;`

D17	Ukaž mi zelená auta.
RA	`car(colour='zelená')`

SQL	`SELECT DISTINCT *`
	`FROM car`
	`WHERE colour = 'zelená';`

D18	Ukaž mi všechny majitele BMW SUV.

RA	`{car(brand='BMW' ∧ car_type='SUV')`
	`[registration_number=car_registration_number>`
	`relation_12}[customer_email=email>customer`

SQL	`select distinct email, first_name, surname, tel_number`
	`from customer c`
	`join relation_12 r on c.email=r.customer_email`
	`join car on r.car_registration_number=car.registration_number`
	`where car.brand = 'BMW' and car.car_type = 'SUV'`
	`--order by email asc`
	`;`

D19	Všechny možné kombinace zákazníků a registračních značek aut.
RA	`{customer × relation_12}[email, car_registration_number]`

SQL	`(SELECT DISTINCT email, car_registration_number`
	`FROM customer`
	`CROSS JOIN relation_12);`

D20	Seznam zaměstnanců a jejich pozic, kteří mají plat nad 30 000.
RA	`{employee (evaluation_czk > 30000) } [first_name, surname, emp_position]`

SQL	`(SELECT DISTINCT first_name, surname, emp_position`
	`FROM employee`
	`WHERE evaluation_czk > 30000);`

D21	Všechna auta která nejsou SUV nebo Coupe.
RA	`car (car_type <> 'SUV' ∨ car_type <> 'Coupe' )`

SQL	`SELECT DISTINCT *`
	`FROM car`
	`WHERE car_type <> 'SUV'`
	`  OR car_type <> 'Coupe';`

D22	Ukaž mi všechny manažery kteří vystudovali CVUT a mají plat nad 26000.
RA	`{Office_manager(university_degree='CVUT')  *> employee}(evaluation_czk > 26000)`

SQL	`(SELECT DISTINCT employee_id, age, evaluation_czk, first_name, surname, local_office_office_id, emp_position`
	`FROM Office_manager`
	`NATURAL JOIN employee`
	`WHERE evaluation_czk > 26000 and university_degree = 'CVUT');`

D23	Ukaž mi všechny české adresy.
RA	`Country (country_name='Ceska republika')` `[country_id=country_country_id> address`

SQL	`select distinct building_number, city, postal_code, street, address_id, country_country_id`
	`from address a`
	`join country c on a.country_country_id = c.country_id`
	`where c.country_name = 'Ceska republika'`
	`;`

D24	Ukaž mi zákazníky, kteří si mají objednávky z let 2010 až 2012.
RA	`order_t (order_date > '1.1.2009' ∧ order_date < '1.1.2013') [customer_email=email>customer`
SQL	`select distinct email, first_name, surname, tel_number from customer c join order_t o on c.email = o.customer_email where order_date > TO_DATE('1.1.2009','dd.mm.yyyy') and order_date < TO_DATE('1.1.2013','dd.mm.yyyy') order by email ;`

D25	Zákazníci kteří nemají BMW ani Audi.
RA	`{customer} \ {{car(brand='BMW' ∨ brand='Audi')[registration_number=car_registration_number>relation_12 }[customer_email=email>customer}`
SQL	`(select distinct email, first_name, surname, tel_number from customer) minus (select distinct email, first_name, surname, tel_number from customer right join (select distinct * from relation_12 right join (select distinct * from car where brand = 'BMW' or brand = 'Audi') on registration_number = car_registration_number) on email = customer_email);`

Závěr

Semestrální práce mi pomohla nahlédnout do záludností kolem databází a uvědomit si že databáze asi nejsou úplně pro mě. Nástroje které jsem používal během této semestrální práce bych nehodnotil velmi dobře, hlavně SQL Developer. Nejvíce mě motivovalo vybrané téma. Bohužel, koncem semestru času na DBS je málo a myslím si že by neuškodilo kdyby byly konzultace častěji.