Praćenje mobilnih objekata u stvarnom vremenu zasnovano na prostornoj bazi podataka

Katalinić, David

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Preddiplomski studij računarstva

Završni rad

Praćenje mobilnih objekata u stvarnom vremenu zasnovano na prostornoj bazi podataka

Rijeka, rujan 2022.

David Katalinić
0069087868

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Preddiplomski studij računarstva

Završni rad

Praćenje mobilnih objekata u stvarnom vremenu zasnovano na prostornoj bazi podataka

Mentor: doc.dr.sc. Sandi Ljubić

Rijeka, rujan 2022.

David Katalinić
0069087868

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
POVJERENSTVO ZA ZAVRŠNE ISPITE

Rijeka, 3. ožujka 2022.

Zavod: **Zavod za računarstvo**
Predmet: **Baze podataka**
Grana: **2.09.02 Informacijski sustavi**

ZADATAK ZA ZAVRŠNI RAD

Pristupnik: **David Katalinić (0069087868)**
Studij: Preddiplomski sveučilišni studij računarstva

Zadatak: **Praćenje mobilnih objekata u stvarnom vremenu zasnovano na prostornoj bazi podataka / Real-time Tracking of Mobile Objects based on a Spatial Database**

Opis zadatka:

U radu je potrebno analizirati postojeća proširenja baza podataka opće namjene za podršku za rad s prostornim podacima. Opisati karakteristične tipove podataka koji u takvim proširenjima pružaju temeljnu apstrakciju i modeliraju strukturu geometrijskih objekata s njihovim odgovarajućim odnosima i operacijama u prostornom okruženju. Istražiti mogućnosti specifičnih prostornih upita koji su podržani u prostornoj bazi podataka. Na temelju usvojenih znanja, implementirati sustav za praćenje mobilnih objekata u stvarnom vremenu koji na poslužiteljskoj strani koristi model prostorne baze podataka. Na klijentskoj strani potrebno je implementirati odgovarajući vizualizacijski modul za prikaz mobilnih objekata na geografskoj karti. Trajektorije i brzine kretanja mobilnih objekata mogu se zadati direktno putem sučelja sustava i putem konfiguracijskih datoteka. Sustav mora podržati i detekciju ciljanih događaja zasnovanih na međusobnom prostornom odnosu mobilnih objekata.

Rad mora biti napisan prema Uputama za pisanje diplomskih / završnih radova koje su objavljene na mrežnim stranicama studija.

Katalinić

Zadatak uručen pristupniku: 21. ožujka 2022.

Mentor:

Doc. dr. sc. Sandi Ljubić

Predsjednik povjerenstva za završni ispit:

Prof. dr. sc. Kristijan Lenac

Izjava o samostalnoj izradi rada

Izjavljujem da sam samostalno izradio ovaj rad.

Rijeka, rujan 2022.

David Katalinić

Sadržaj

Popis slika

Poglavlje 1

Uvod

Glavna tema koja će se obraditi u ovom radu biti će prostorne baze podataka (engl. spatial database) i njihove značajke. Prostorne baze podataka su baze koje omogućuju spremanje prostornih podataka (engl. spatial data) te pružaju niz funkcija i mogućnosti za rad s tim prostornim podatcima. Neke od primjena prostornih baza su u nadzoru i praćenju objekata, urbanom planiranju ili za praćenje gustoće stanovništva na nekom području.

Prije govora o samim prostornim bazama podataka potrebno je definirati geoinformacijske sustave koji su obrađeni u sljedećem poglavlju. Nakon toga u trećem poglavlju analizirati će se 3 proširenja za rad s prostornim podatcima: PostGIS, Microsoft SQL Server i MySQL Spatial. U sljedećem poglavlju detaljno će se obraditi svi karakteristični tipovi podataka koji u prostornim bazama podataka omogućavaju apstrakciju i modeliraju strukturu geometrijskih objekata. U petom poglavlju govori se o specifičnim prostornim upitima (engl. spatial quries) te su dani primjeri upita u SQL-u. Osim prostornih upita u ovom poglavlju opisani su i prostorni indeksi (engl. spatial index). Naposljetku u zadnjem poglavlju opisani je razvoj i značajke aplikacije za praćenje mobilnih objekata u stvarnom vremenu kao i tehnološki stog aplikacije, model baze podataka te karakteristični slučajevi korištenja sustava na klijentskoj (engl. frontend) strani.

Poglavlje 2

Geoinformacijski sustavi (GIS)

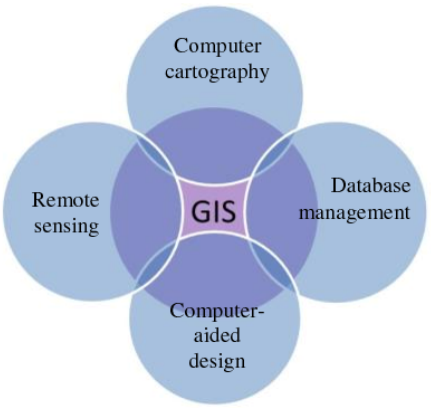
Postoje različite definicije geoinformacijskih sustava (engl. geographic information system) gdje se svaka razvila iz određene perspektive ili discipline. Geoinformacijski sustav možemo formalno definirati kao informacijski sustav za upravljanje, analizu, vizualiziranje i distribuiranje informacija o objektima i pojavama, čiji referentni sustav je definiran na Zemlji. Sastoji se od sklopovlja (engl. hardware), računalnih programa, podataka, ljudi, organizacija i institucijskih aranžmana za skupljanje, pohranjivanje, analiziranje i diseminaciju informacija o područjima na Zemlji.

Kod geoinformacijskih sustava definiraju se četiri glavna podsustava [1]:

- Sustav za unos podataka: Ovaj podsustav bavi se prikupljanjem i/ili procesiranjem geoprostornih podataka.
- Sustav za pohranjivanje i dohvat podataka: organizira i pohranjuje podatke u obliku koji omogućuje brz dohvat za buduće analize kao i brza ažuriranja.
- Sustav za rukovanje i analizu podataka: obavlja različite zadatke kao što su promjena formata podataka.
- Sustav za izvještavanje: prikazuje cijelu ili dio baze podataka u tabličnom ili kartografskom obliku.

Za određivanje definicije geoinformacijskog sustava važno je definirati odnos između GIS-a i oblikovanja pomoću računala (engl. computer aided design), sustava za upravljanje bazama podataka (engl. database management), računalne kartogra-

fije (engl. computer cartography) i upravljanje udaljenim osjetilima (engl. remote sensing) (Slika 2.1). Često se tvrdi da je GIS podskup ili nadskup ovih pojedinih dijelova.



Slika 2.1 *Odnos između GIS-a i pojedinih sustava* [2]

GIS zajedno sa svim svojim povezanim tehnologijama može odgovoriti na pet pitanja [1]:

- Što se nalazi na nekoj lokaciji?
- Gdje je nešto?
- Što se promijenilo od?
  Npr. Šumska područja čija se površina smanjila u zadnjih 10 godina?
- Koji geoprostorni podatci postoje?
  a) Koji su odnosi između dva skupa podataka koji se odnose na istu lokaciju?
  b) Koje geografske varijacije postoje s obzirom na prostor?
- Što ako? Npr. Što se događa s populacijom na nekom području ako se kroz to područje sagradi autocesta?

Razvoj u računalnoj tehnologiji tijekom 60.-ih godina 20. stoljeća dovodi do naglog razvoja geoinformacijskih sustava. Danas geoinformacijski sustavi imaju široku primjenu u različitim industrijama. Neke od primjena su: kontroliranje kvalitete vode, automatsko mapiranje, nadzor i praćenje objekata, urbano planiranje.

Poglavlje 3

Proširenja baze podataka opće namjene za podršku za rad s prostornim podatcima

Danas postoje mnogobrojna proširenja te sustavi za upravljanje bazama podataka (SUBP) (engl. database management system) koji pružaju podršku za rad s prostornim podatcima. U ovom poglavlju detaljno će se analizirati tri različita proširenja/SUBP-a: PostGIS, MySQL Spatial i Microsoft SQL Server. Za svako proširenje opisati će se vrste podataka koje podržava, prostorni upiti i prostorni indeksi.

3.1 PostGIS

PostGIS je besplatno proširenje otvorenog tipa za PostgresSQL koji je razvijeno sukladno standardima propisanim od strane *Open Geospatial Consortium-a* (OGC). Glavni razlog zašto je PostGIS razvijen kao proširenje za PostgresSQL je velika fleksibilnost koju PostgresSQL pruža u razvoju novih proširenja te ujedno veliki broj mogućnosti koje PostgresSql pruža.

3.1.1 Povijest razvoja PostGIS-a

PostGIS je započeo kao projekt kompanije Refractions Research koja se bavi geoprostornim savjetovanjem. U svibnju 2001, objavljena je prva verzija PostGIS-a, PostGIS 0.1 koja je imala objekte, indekse i mali broj funkcija. Rezultat ove prve verzija je bila baza podataka koja je mogla samo spremati i vraćati podatke, ali ne i analizirati podatke.

Kako se povećavao broj funkcija javila se potreba za jasnom organizacijskom strukturom. Rješenje za ovaj problem nudi *Open Geospatial Consortium* u obliku dokumenta *Simple Features for SQL*. *Simple Features for SQL* je specifikacija koja definira tipove i funkcije koje čine standardnu bazu podataka. Sa daljnjim razvojem rastao je broj dostupnih funkcija, ali svejedno mogućnosti i sposobnosti PostGIS-a ostale su značajno limitirane. Mnoge važne funkcije poput unije, presjeka, dodirivanja nisu bile dostupne te je njihova implementacija bila komplicirana i vremenski zahtjevna. Srećom, ubrzo je razvijen projekt pod nazivom *Geometry Engine, Open Source - GEOS* koji je sadržavao potrebne algoritme za potpunu implementaciju *Simple Features for SQl* specifikacija.

3.1.2 Vrste prostornih podataka

U PostGIS-u postoje četiri glavne vrste prostornih podataka: geometrija, geografija, topologija (engl. topology) i rešetka (engl. raster). Geometrijska vrsta podataka prisutna je od samog početka dok PostGIS 1.5 uvodi geografski tip podataka. Topološku vrstu podataka i rešetke uvodi PostGIS 2.0. Sve četiri vrste podataka mogu postojati unutar iste baze podataka, čak mogu postojati unutar istih tablica u različitim stupcima.

Geometrija je planarski (engl. planar) tip podataka. Iako je prva vrsta podataka dostupna u PostGIS-u, još uvijek je najpopularniji tip podataka koji PostGIS podržava te je temelj svih ostalih vrsta podataka. Svijet u geometrijskom tipu podataka promatra se kao ravna kartezijanska mreža u kojem je najkraći put između dvije točke ravna linija. U 2D modelu sve entitete možemo prikazati pomoću tri glavne vrste geometrije: točkom (engl. point), poligonom (engl. polygon) i nizom povezanih

točaka (engl. linestring).

Geografski tip podataka temelji se na sfernom modelu svijeta u kojem je najkraći put između dviju točaka na sferi kružni luk. Primjenom sfernog modela svijeta funkcije nad geografskim podatcima daju točnije rezultate. Geografske koordinate su sferne koordinate izražene u stupnjevima. Zbog složene matematike manji je broj dostupnih funkcija te brzina izračuna sporija.

Topologija je relacijski model podataka. Topologija modelira svijet kao mrežu međusobno povezanih čvorova, rubova i lica. Objekti koji se sastoje od tih elemenata (povezanih čvorova, rubova i lica) mogu dijeliti te elemente s drugim objektima. Kao rezultat toga, ako se u topološkoj mreži promjeni rub nekog objekta, onda će se svi objekti, koji dijele taj rub, promijeniti sukladno tome.

Rešetke sukladno svojem imenu modeliraju svijet kao mrežu pravokutnih ćelija gdje svaka ćelija sadrži numerički niz vrijednosti. Podatci rešetka mogu se najbolje definirati kao mozaik piksela. Najbolji primjer rešetke je televizija koja nije ništa više nego jedna velika rešetka s više od dva milijuna piksela.

Sukladno *OGC Simple Feature* specifikaciji PostGIS implementira dva formata za predstavljanje geometrijskih vrijednost za vanjsku upotrebu: Well Known Text (WKT) i Well Known Binary (WKB). WKT pruža standardnu tekstualnu reprezentaciju prostornih podataka dok WKB pruža prijenosnu, preciznu reprezentaciju prostornih podataka kao binarnih podataka. Kako je *OGC Simple Feature* specifikacija u početku podržavala samo 2D geometriju te nije sadržavala podatke o identifikatoru referentnog prostornog sustava (engl. spatial reference identifier (SRID)), PostGIS je razvio Extended Well Known Text (EWKT) i Extended Well Known Binary (EWKB) formate. Ova dva nova formata pružaju podršku za 3D i 4D koordinate te sadržavaju SRID informaciju.

3.1.3 Prostorni indeksi

Prostorni indeksi su ključni dio prostornih baza jer nam omogućavaju korištenje velikih skupova podataka u prostornim bazama. Bez prostornih indeksa svaka pretraga baze zahtijevala bi sekvencijalno skeniranje svih zapisa u bazi što zahtijeva veliku količinu vremena. Jedna od glavnih prednosti PostgresSQL-a za rukovanje prostornim podatcima je ta što nudi više vrsta metoda indeksiranja koje rade s višedimenzionalnim podatcima. Glavne metode indeksiranja dostupne u PostGIS-u su: GiST, BRIN i SP-GiST.

GiST je najkorištenija metoda indeksiranja za multidimenzionalne podatke te se može koristiti nad različitim vrstama podataka. PostGIS još dodatno implementira R stablo povrh GiST-a kako bi omogućio indeksiranje prostornih podataka. GiST je najkorištenija metoda indeksiranja te pruža izrazito kvalitetne perfomanse upita.

BRIN indeks za svaki raspon blokova (engl. block range) sprema određeni sažetak informacija. U PostgresSQL-u blok definira osnovnu jedinicu za spremanje podataka te je 8kB velika dok raspon blokova označava grupu stranica koje su fizički susjedne u tablici. BRIN je prikladan samo za podatke koji su prostorno sortirani te se rijetko ili nikad ažuriraju. U zamjenu za svoju limitiranost, što se tiče vrste podataka za koju se može primijeniti, BRIN pruža puno veću brzinu indeksa te znatno manju veličinu indeksa.

SP-GiST je generička metoda indeksiranja koja podržava particionirana stabla pretrage kao što su *quad stabla*, k-d stabla i *radix stabla*. Kao što samo ime sugerira, ova metoda dijeli podatkovne strukture u particije koje ne moraju biti jednake veličine. SP-GiST je najbolje koristiti u geometrijama koje se ne preklapaju.

3.1.4 Prostorni upiti

PostGIS korisniku na raspolaganje daje veliki broj funkcija i operatora. Osnovne funkcije koje PostGIS podržava su: ST_Contains(), ST_Crosses(), ST_Disjoint(), ST_Equals(), ST_Intersects(), ST_Overlaps(), ST_Touches(), ST_Within() te su one definirane prema *OGC Simple Feature* specifikaciji i njihova je uloga određivanje uobičajenih prostornih odnosa. Navedene funkcije uobičajeno se koriste kao uvjeti u

SQL WHERE i JOIN naredbama.

Prilikom kreiranja prostornog upita, kako bi se osiguralo najbrže moguće izvođenje, potrebno je koristiti prostorni indeks ako on postoji. Kako bi se koristio prostorni indeks u WHERE ili ON uvjetu, mora se koristiti ili prostorni operator (engl. spatial operator) ili funkcija svjesna indeksa (engl. index-aware function). Prostorni operatori uključuju operatore ograničavajućeg pravokutnika (engl. bounding box operators) od kojih je najpopularniji && te operatori daljine koji se koriste u upitima najbližeg susjeda. Funkcije koje su svjesne indeksa automatski dodaju operatore ograničavajućeg pravokutnika. U funkcije koje automatski dodaju operatore pripadaju gore navedene funkcije te uz njih još pripadaju funkcije: ST_DWithin, ST_DFullyWithin, ST_3DDFullyWithin, i ST_3DDWithin. ST_Distance je primjer funkcije koja ne koristi indekse za optimizaciju pretrage. Zbog toga, sljedeći bi upit (Ispis 3.1) bio izrazito spor na velikoj tablici podataka:

```
1 SELECT geom
2 FROM geom_table
3 WHERE ST_Distance( geom, 'SRID=312;POINT(100000 200000)') < 100
```

Ispis 3.1 Primjer upita koji ne koristi indeks

Ovaj upit (Ispis 3.1) je izrazito spor jer računa udaljenost svake točke u tablici od specificirane točke, odnosno, izračun se izvršava na svakom redu tablice. Broj izračuna može se znatno smanjiti korištenjem funkcije ST_DWithin što je vidljivo na upitu Ispis 3.2.

```
1 SELECT geom
2 FROM geom_table
3 WHERE ST_DWithin( geom, 'SRID=312;POINT(100000 200000)', 100 )
```

Ispis 3.2 Primjer upita koji koristi indeks

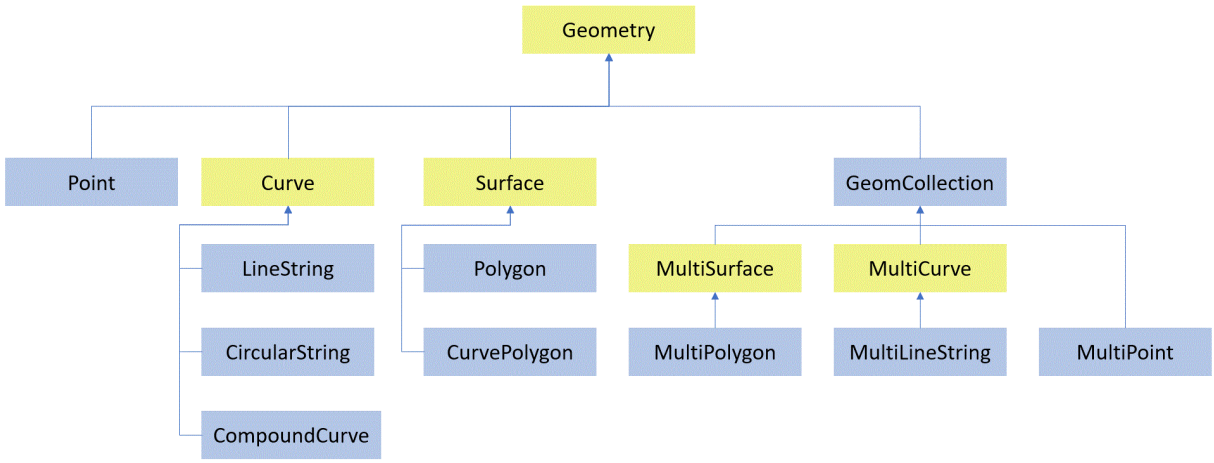
Ako postoji prostorni indeks, on omogućava da se koriste samo oni zapisi s geometrijama čiji se ograničavajući pravokutnici preklapaju s proširenim opsegom pretrage.

3.2 Microsoft SQL Server

Podršku za rad s prostornim podatcima uvedena je objavom SQL Server 2008 koji je kao i PostGIS razvijen u skladu s *OGC Simple Feature* specifikacijom. Za razliku od PostGIS-a, koji je proširenje za PostgresSQL koje je potrebno posebno instalirati, podrška za rad s prostornim podatcima je osnovna komponenta Microsoft SQL poslužitelja (engl. server).

3.2.1 Vrste prostornih podataka

SQL poslužitelj, za razliku od PostGIS-a, ne podržava četiri vrste prostornih podataka, nego podržava dva tipa podataka: geometriju i geografiju. Geometrija i geografija podržavaju šesnaest objekata prostornih podataka od kojih se njih jedanaest može instancirati. Na slici ispod (Slika 3.1) prikazana je geometrijska hijerarhija na temelju koje su geometrijski i geografski tipovi podataka bazirani. Vrste podataka koje se mogu instancirati označene su plavom bojom.



Slika 3.1 *Geometrijska hijerarhija* [7]

Geometrija i geografija često se ponašaju slično, ali svejedno postoji nekoliko ključnih razlika u tome kako se podatci spremaju i analiziraju. Jedna od glavnih razlika je način mjerenja prostornih podataka. U geometriji, koja je planarnog tipa,

udaljenosti i površine uvijek se računaju u mjernoj jedinici u kojoj su zadane koordinate dok su kod geografije, koja je elipsoidnog sustava, koordinate zadane u stupnjevima, udaljenosti i površine se računaju u metrima i kvadratnim metrima. Nadalje još jedna značajna razlika, a tiče se samih objekata prostornih podataka, je orijentacija prostornih podataka. Primjer ove razlike je orijentacija prstenova u poligonu. Orijentacija prstenova u poligonu u planarnom prstenu nije važan faktor dok u elipsoidnom sustavu poligon bez orijentacije nema nikakvog smisla ili je dvosmislen.

Sukladno *OGC Simple Feature* specifikaciji SQL poslužitelj podržava WKT i WKB formate za vanjsku upotrebu, ali za razliku od PostGIS-a nema definirane EWKT i EWKB. Razlog toga je taj što je podrška za prostorne podatke razvijena tek 2008. godine nakon što je *OGC Simple Feature* specifikacija počela podržavati 3D i 4D geometriju.

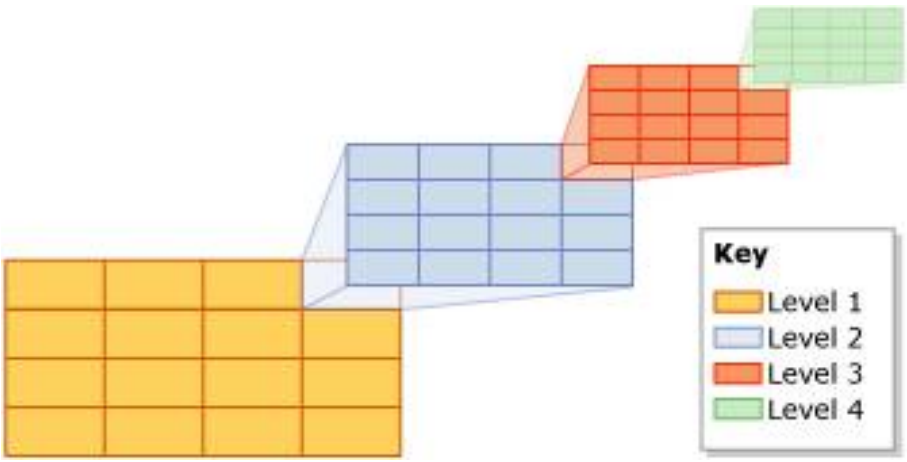
3.2.2 Pohrana prostornih podataka

Kada je u pitanju pohrana podataka, geometrija i geografija su tipovi podataka promjenjive duljine. To znači da količina prostora potrebna za njihovu pohranu ovisi o složenosti objekta kojeg ti podatci opisuju. Prostorni podatci su u potpunom kontrastu s tipovima podataka kao što su *int* ili *datetime* koji su vrsta podataka fiksne veličine neovisno o objektu kojeg opisuju. Kada su u pitanju podatci geometrijskog i geografskog tipa, oni se spremaju u obliku toka binarnih podataka. Svaki tok započinje zaglavljem koji sadrži osnovne informacije poput vrste geometrije, broja točaka u cijelom objektu i prostornog referentnog sustava koji se koristi. Nakon zaglavlja odmah slijede x i y koordinate u 8-bitnom binarnom zapisu. Ovisno o broju točaka u objektu povećava se veličina toka binarnih podataka, a samim time i količina potrebnog prostora za pohranu podataka.

3.2.3 Prostorni indeksi

U SQL poslužitelju svi indeksi su kreirani koristeći B-stabla, što znači da indeksi moraju predstavljati 2-dimenzionalne prostorne podatke u linearnom poretku B-stabala. Proces kreiranja indeksa dekomponira prostor u hijerarhijsku mrežu (engl.

grid hierarchy,) s četiri razine. Svaka sljedeća razina dalje razlaže razinu iznad sebe te stoga svaka ćelija razine iznad sadrži potpunu mrežu razine ispod. Ukupan broj ćelija jednak je na svim razinama te su sve ćelije jednake veličine. Na Slici 3.2 može se vidjeti primjer dekompozicije jedne ćelije na svim razinama (engl. levels) mreže.



Slika 3.2 *Dekompozicija jedne ćelije po svim razinama* [7]

Nakon završetka procesa dekompozicije prostora, prostorni indeks čita prostorne podatke iz tablice red po red. Po završetku čitanja podataka za prostorni objekt prostorni indeks pokreće proces teselacije (engl. tessellation) za taj objekt. U procesu teselacije objekt se uklapa u hijerarhiju mreže pridružujući objekt skupu ćelija mreže koje taj objekt dodiruje. Izlaz procesa teselacije je skup ćelija koje se dodiruju te se one spremaju u indeks za taj objekt. Koristeći ove spremljene ćelije prostorni indeks može locirati objekt u prostoru u odnosu na druge objekte u prostornom stupcu koji su također pohranjeni u indeksu.

Za razumijevanje načina funkcioniranja prostornih indeksa u upitima trebamo prvo objasniti kako funkcioniraju upiti u SQL poslužitelju. SQL poslužitelj koristi upite s dvije faze: primarni filtar i sekundarni filtar. Primarni filtar je brzi aproksimirajući upit koji odabire potencijalne podatke koje onda šalje sekundarnom filtru. Među tim potencijalnim podatcima samo jedan dio će završiti u rezultatu te se taj dio mora filtrirati u sekundarnom filtru. Sekundarni filtar je precizan, ali spor upit koji se izvršava nad podatcima koji su odabrani u primarnom filtru.

U slučaju da postoji prostorni indeks nad podatcima, onda se taj indeks može koristiti umjesto primarnog filtra te tako smanjiti količinu podataka koju je potrebno testirati sporijim sekundarnim filtrom. Prostorni indeksi podržavaju sljedeće geometrijske metode: STContains(), STDistance(), STEquals(), STIntersects(), STOverlaps(), STTouches() i STWithin() koje se moraju koristiti u WHERE ili JOIN naredbama. Od geografskih metoda podržane su STIntersects(), STEquals() i STDistance(). Geografske metode, za razliku od geometrijskih, mogu se koristiti samo unutar WHERE naredbe, ako se želi koristiti prostorni indeks. Primjer upita za najbližeg susjeda prikazan je ispod (Ispis 3.3).

```
SELECT TOP(K) [WITH TIES] *
FROM <Table> AS T [WITH(INDEX(<SpatialIndex>))]
WHERE <SpatialColumn>.STDistance(@reference_object) IS NOT NULL
ORDER BY <SpatialColumn>.STDistance(@reference_object) [;]
```

Ispis 3.3 Upit za pronalazak najbližeg susjeda

3.3 MySQL

Podrška za prostorne tipove podataka počela je 2004. s MySQL 4.1. Performanse i preciznost ovih prostornih podataka i funkcija bile su izrazito loše te je MySQL-u trebalo sve do 2013. godine i verzije 5.6 da može uspješno procesirati prostorne podatke i odnose. Unatoč tome, broj dostupnih značajki još je uvijek bio mali te je MySQL tek 2018. počeo podržavati neke ključne značajke poput višestrukih referentnih sustava i geografskih izračuna.

3.3.1 Vrste prostornih podataka

Za razliku od PostGIS-a i Microsfot SQL poslužitelja, MySQL podržava samo jednu vrstu podataka, a to je geometrija. MySQL smatra geometriju, odnosno geografsko obilježje i geoprostorno obilježje kao sinonime te definira geometriju kao točku ili skup točaka koje predstavljaju bilo što u svijetu što ima lokaciju.

Geometrijski razred služi kao osnovni, apstraktni razred koji se ne može instancirati. Podrazredi geometrije koji se mogu instancirati ograničene su na nulte, jednodimenzionalne i dvodimenzionalne geometrijske objekte koji postoje u dvodimenzionalnom koordinatnom sustavu. Osnovna geometrijska klasa ima potklase za:

- točke: predstavljaju nultodimenzionalne objekte.
- krivulje: predstavljaju jednodimenzionalne objekte s potklasom niz povezanih točaka.
- površine: koriste se za dvodimenzionalne objekte te sadrži potklasu poligon.
- geometrijske kolekcije (engl. GeometryCollection): sadrži specijalizirane nulte (kolekcija točaka), jednodimenzionalne (kolekcija linija) i dvodimenzionalne (kolekcija poligona) klase.

Uz navedene vrste prostornih podataka MySQL, kao i PostGIS i Microsoft SQL poslužitelj, podržava WKT i WKB za spremanje podataka i vanjsku upotrebu.

3.3.2 Dobro oblikovana geometrija i valjanost

Za geometrijske vrijednosti MySQL razlikuje dobro oblikovanu geometriju i valjanost geometrije. Da bi se geometrija smatrala dobro oblikovanom mora zadovoljiti sljedeće uvjete:

- niz povezanih točaka mora imati barem dvije točke
- poligon mora imati barem jedan prsten
- prsten poligona mora biti zatvoren (prva i zadnja točka moraju biti iste)
- prsteni poligona moraju imati barem četiri točke
- kolekcije ne smiju biti prazne

Geometrija se smatra geometrijski valjana ako zadovoljava uvjete dobro oblikovane geometrije i ako zadovoljava sljedeće uvjete:

- unutarnji prstenovi poligona moraju biti unutar vanjskog prstena
- poligoni ne sijeku sami sebe

- multipoligoni (engl. Multipolygons) ne smiju imati poligone koji se preklapaju

Prostorne funkcije javit će grešku ako geometrije nisu dobro oblikovane. Isto vrijedi i za funkcije uvoza podataka koje koriste WKT i WKB.

3.3.3 Prostorni indeksi

Prostorni indeksi važan su dio MySQL optimizacije jer mogu znatno ubrzati proces izvršavanja prostornih upita. Prilikom kreiranja prostornih indeksa na prostornim stupcima MySQL koristi R stablo. Za kreiranje prostornih indeksa koristi se minimalno ograničavajući pravokutnik.

Naredba za kreiranje indeksa slična je naredbi za kreiranje normalnih indeksa samo što se još koristi naredba SPATIAL.
Primjeri naredbi za kreiranje prostornog indeksa:

- S naredbom CREATE

```
1 CREATE TABLE geom (g GEOMETRY NOT NULL, SPATIAL INDEX(g));
```

Ispis 3.4 Kreiranje prostornog indeksa s CREATE

- S naredbom ALTER

```
1 CREATE TABLE geom (g GEOMETRY NOT NULL);
2 ALTER TABLE geom ADD SPATIAL INDEX(g);
```

Ispis 3.5 Kreiranje prostornog indeksa s ALTER

Prilikom stvaranja prostornih indeksa postoji nekoliko ograničenja:

- prostorni indeksi dostupni su samo na MyISAM i InnoDB tablicama
- stupci nad kojima su stvoreni indeksi ne smiju biti NULL
- puna širina svakog stupca je indeksirana.

Prilikom izvršavanja prostornog upita optimizator provjerava postoje li prostorni indeksi koji se mogu iskoristiti u funkcijama koje su u WHERE naredbi. Za razliku od prostornih upita u Microsoft SQL poslužitelju, koji se sastoji od dva filtra gdje je prvi

filtar brz te pronalazi moguća rješenja koja se onda u sekundarnom filtru dodatno provjeravaju, u MySQL upitu uvijek se provjeravaju svi zapisi u tablici. Ovdje se može vidjeti velika važnost prostornih indeksa u upitima jer znatno smanjuju broj podataka koji se moraju provjeriti.

3.3.4 Rukovanje argumentima u prostornim funkcijama

Svaka geometrijska vrijednost povezana je s određenim referentnim prostornim sustavom za geografske lokacije. Za izračune na višestrukim geometrijskim vrijednostima, sve vrijednosti moraju biti u istom referentnom prostornom sustavu ili će doći do pogreške. Geometrija koja je rezultat prostorne funkcije nalazi se u istom referentnom prostornom sustavu kao i argumenti funkcije jer rezultantne geometrije nasljeđuju identifikator prostorne reference koji određuje u kojem se referentnom prostornom sustavu geometrija nalazi.

Poglavlje 4

Vrste prostornih podataka

U ovom poglavlju opisuju se karakteristični tipovi podataka koji su dostupni u proširenjima za rad s prostornim podacima zajedno sa njihovim podvrstama. Sve navedene vrste prostornih podataka nisu dostupni u svim proširenjima, ali su dostupni u PostGIS proširenju koje je korišteno u izradi aplikacije.

4.1 Geometrija i geografija

Geometrija i geografija su dvije najosnovnije i najkorištenije vrste prostornih podataka koje su dostupne u većini proširenja. U ovom potpoglavlju detaljno će se obraditi oba tipa podataka te će se opisati kada je bolje koristiti geometriju, a kada geografiju.

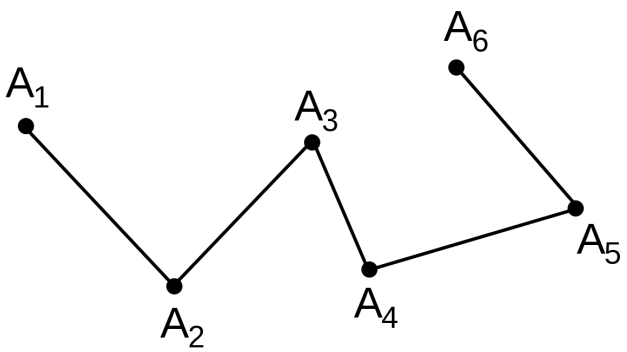
4.1.1 Geometrija

Kada se govori o vrstama prostornih podataka, prvo je potrebno spomenuti geometriju koja je osnovni tip prostornih podataka koji je dostupan u svim proširenjima za podršku za rad s prostornim podatcima. Geometrija je planarski tip podataka u kojem se svijet modelira kao dvodimenzionalna kartezijanska mreža. Geometrijske vrijednosti povezane su s referentnim prostornim sustavom koji se određuje pomoću SRID broja. Mjerne jedinice x i y osi određene su referentnim prostornim

sustavom. U geometriji svijet modeliramo pomoću sljedećih geometrijskih tipova podataka: točka, niz povezanih točaka, poligona i kolekcija.

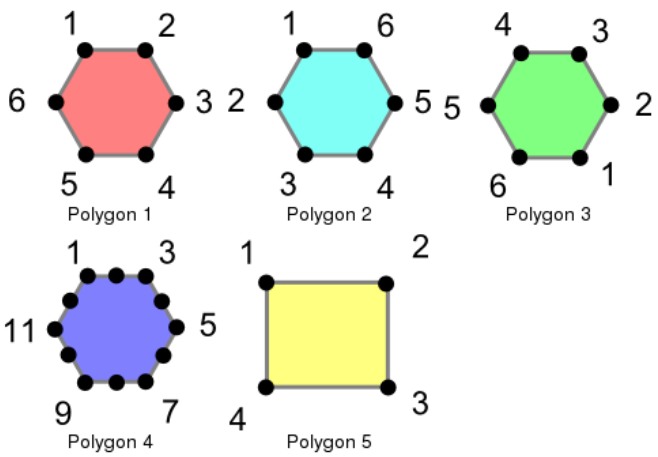
Točka je nultodimenzionalna geometrija što znači da nema dužinu u ni jednom smjeru te predstavlja jednu lokaciju u koordinatnom prostoru. Točka ne sadržava nikakve granice te je unutrašnjost točke ujedno sama točka. Zbog svih ovih karakteristika točka se definira kao jednostavna geometrija. Točka se definira s x i y koordinatama za dvodimenzionalne točke, a za trodimenzionalne točke dodaje se još i z koordinata. Za 4D točke dodaje se M koordinata. U prostornim podatcima točka se generalno primjenjuje za prikaz točne lokacije u prostoru. Ta lokacija može biti adresa neke ulice, lokacija neke zgrade, grada, i slično.

Niz povezanih točaka je jednodimenzionalna linija koja se sastoji od kontinuiranog niza linijskih segmenata. Svaki segment sastoji se od dvije točke gdje je krajnja točka jednog segmenta početna točka drugog segmenta. Prema *OGC Simple Feature* specifikaciji, niz povezanih točaka može imati ili 0 točaka ili 2 točke ili više od 2 točke. Niz povezanih točaka koji ne siječe samog sebe smatra se jednostavnim. Kada su početna i zadnja točka jednake, onda se niz povezanih točaka smatra zatvorenim, a ako je niz povezanih točaka zatvoren i jednostavan, onda se zove prstenom. Glavna primjena niza povezanih točki je prikaz ulica, cesta i rijeka na mapi. Na Slici 4.1) može se vidjeti vizualni primjer jednostavnog niza povezanih točaka.



Slika 4.1 *Niz povezanih točaka.*

Poligon je dvodimenzionalna geometrija koja ima pridruženu dužinu i površinu. Oblik poligona definiran je vanjskom granicom koja se sastoji od niza povezanih točaka koje tvore prsten. Za razliku od niza povezanih točaka koji definira samo točke koje leže na prstenu, poligon sadrži i sve točke unutar prstena. Zbog toga se poligon koristi za prikaz područja. Osim vanjskog prstena koji određuje njegov oblik, poligon može imati i 0 ili više unutarnjih prstenova koji su zapravo rupe unutar poligona. Poligon se uglavnom koristi za prikaz objekata čija su veličina i oblik bitni. Granice gradova, parkovi i vodene površine često se prikazuju pomoću poligona. Na Slici 4.2 mogu se vidjeti vizualni prikazi 5 različitih poligona.



Slika 4.2 *Primjeri poligona* [4]

Najčešći prostorni upit koji se provodi s poligonima je izračun površine poligona. Na primjeru ispod (Ispis 4.1) može se vidjeti upit za izračun površine svih poligona u tablici u PostGIS proširenju za PostgresSQL.

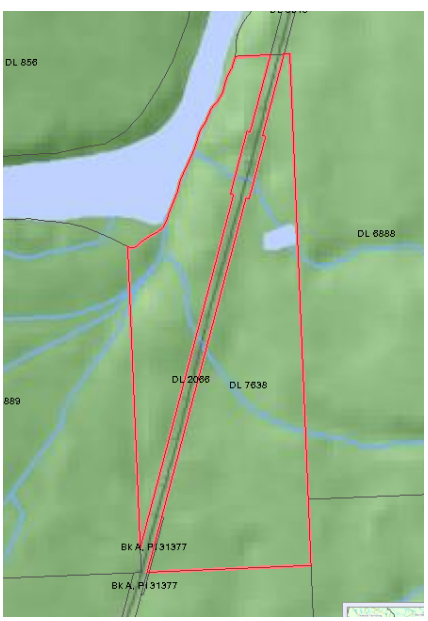
```
1  SELECT name, ST_Area(geom)
2  FROM geometries
3  WHERE name LIKE 'Polygon%';
```

Ispis 4.1 Izračun površine poligona

Kolekcije grupiraju više jednostavnih geometrija u setove. Postoje 4 glavne vrste kolekcija:

- kolekcija točaka (engl. MultiPoint)
- kolekcija nizova povezanih točaka (engl. MultiLinestring)
- kolekcija poligona (engl. MultiPolygon)
- geometrijska kolekcija (engl. GeometryCollection): heterogena kolekcija bilo kojih geometrija. Za razliku od ostalih kolekcija koje su istog tipa podataka, geometrijska kolekcija može sadržavati i točke i nizove povezanih točaka i poligone te njihove kolekcije.

Kolekcije su korisne za modeliranje stvarnih objekata u svijetu kao prostornih objekata. Na primjer, kako modelirati zemljište koje je podijeljeno cestom? Modelirat ćemo ga koristeći kolekciju poligona gdje je svaka strana poseban poligon (Slika 4.3).



Slika 4.3 *Zemljište modelirano pomoću kolekcije poligona* [4]

4.1.2 Geografija

U geografskom tipu podataka koristi se sferni model svijeta. Geografske koordinate nisu kartezijevog tipa jer ove koordinate ne prikazuju linearnu distancu između dvije točke. Umjesto kartezijevih koordinata koriste se sferne koordinate koje mjere kutnu distancu te su mjerne jedinice u stupnjevima. Kao i geometrijski tip podataka i geografski tip je povezan s referentnim prostornim sustavom koji se određuje pomoću SRID broja.

4.1.3 Razlike između geometrije i geografije

Prije donošenja odluke o tome hoće li se koristiti geometrijski ili geografski tip podataka postoji nekoliko stvari koje je potrebno sagledati. Geografske metode su preciznije nego geometrijske, ali posljedica toga je manji broj dostupnih funkcija te veća količina potrebnog vremena za izvršenje tih funkcija. Na primjeru prostornog upita ispod (Ispis 4.2) može se vidjeti razlika u izračunatoj distanci koristeći geometriju i geografiju.

```
SELECT ST_Distance(
  ST_GeometryFromText('Point(-118.4079 33.9434)'),
  ST_GeometryFromText('Point(139.733 35.567)'))
  AS geometry_distance,
ST_Distance(
  ST_GeographyFromText('Point(-118.4079 33.9434)'),
  ST_GeographyFromText('Point(139.733 35.567)'))
  AS geography_distance;

 geometry_distance | geography_distance
-------------------+--------------------
  258.146005837336 |   8833954.76996256
```

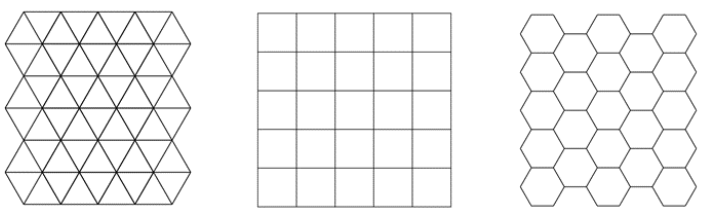
Ispis 4.2 Računanje udaljenosti koristeći geometriju i geografiju

Glavno pitanje je kada je više potrebna veća preciznost geografskih metoda, a kada je više potrebna optimalnost i fleksibilnost geometrijskog tipa podataka. Kada su u pitanju podatci koji su geografski kompaktni odnosno sadržani unutar jedne

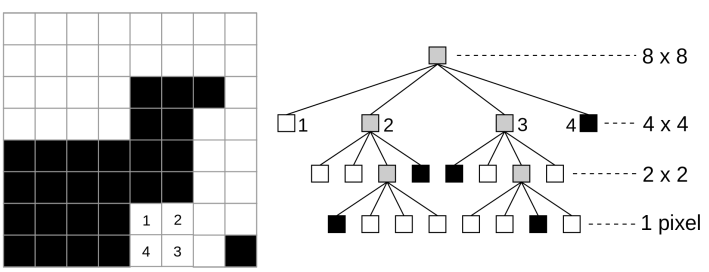
države, županije ili grada, onda je bolje koristiti geometriju s kartezijanskom projekcijom. Ako su podatci geografski raspršeni, odnosno prekrivaju veliki dio svijeta, onda je potrebno koristiti geografski tip podataka.

4.2 Rešetke

Vrste podataka koje su do sada spominjane bile su vektorskog tipa, ali postoji još jedna vrsta podataka koja se često koristi, a to su rešetke. U rešetkama cijeli prostor dijeli se na disjunktne ćelije, gdje svaka ćelija sadržava vrijednost koja vrijedi na području cijele ćelije. Razlikujemo dvije vrste rešetki na temelju oblika ćelija od kojih su sagrađene: ravnomjerne rešetke (Slika 4.4) i neravnomjerne rešetke (Slika 4.5). U ravnomjernim rešetkama sve ćelije su istog oblika i veličine. Primjer ove vrste rešetki su: kvadrat, trokut, šesterokut itd. Kod neravnomjernih rešetki ćelije mogu varirati oblikom i tako efikasnije aproksimirati pojavu, a primjer ove vrste rešetke je *quad* stablo.



Slika 4.4 *Ravnomjerne rešetke* [10]

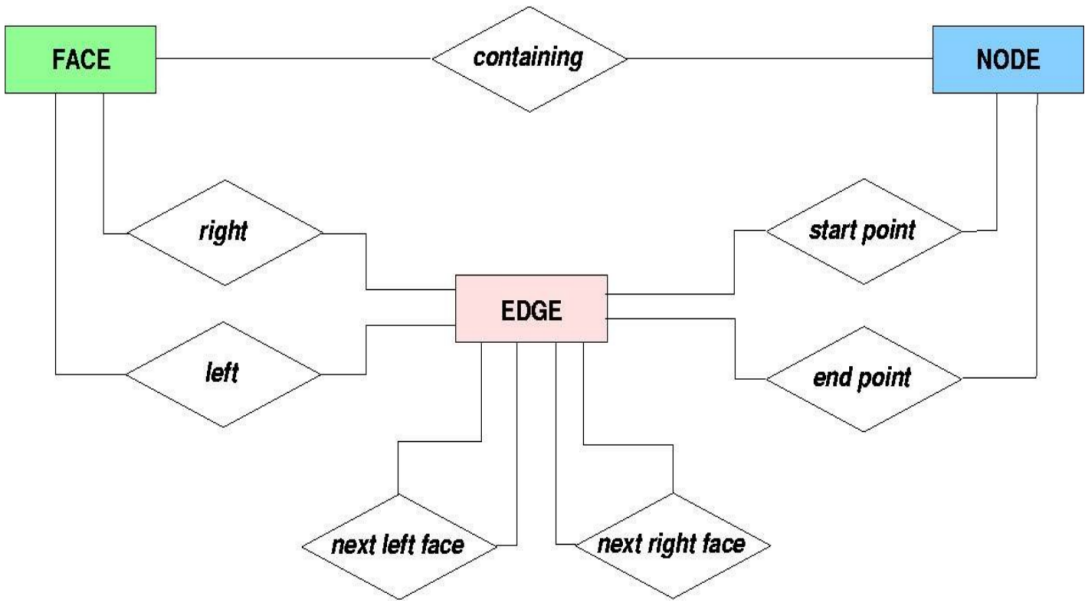


Slika 4.5 *Neravnomjerne rešetke* [1]

Neke od prednosti korištenja rešetki naspram vektorskog tipa podataka je: laka razumljivost rešetki, brzo procesiranje, dodatne funkcije za obradu te oblik podataka. Rešetke imaju i neke nedostatke poput izgleda, preciznosti i veličine samih rešetki.

4.3 Topologija

Topologija je relacijski model podataka. Kao i rešetke, topologija je dostupna samo u PostGIS proširenju. U topologiji svijet se modelira kao mreža povezanih čvorova, rubova i lica. Svi objekti sastavljeni su od ovih elemenata te ih mogu dijeliti s drugim objektima što dovodi do smanjenja prostora potrebnog za pohranu jer se granice moraju pohraniti samo jednom. Jedna od karakteristika topologije su eksplicitni odnosi objekata jer se za svaki rub (engl. edge) zna lijevo i desno lice (engl. face) te se za svaki čvor (engl. node) zna kojem licu pripada. Svaki rub sadrži i sljedeće lijevo i desno lice (engl. next left/right face) te sadrži početnu (engl. start point) i krajnju točku (engl. end point) čvora.



Slika 4.6 *Konceptualni model topologije* [1]

U topologiji postoje dva povezana koncepta: mreže i usmjeravanje. Topološka mreža osigurava da kada se promijeni rub jednog objekta sukladno se tomu promijene svi objekti koji dijele taj rub. Usmjeravanje ne brine samo povezanost, nego i cijena te povezanosti. Uglavnom se koristi za razvoj navigacijskih aplikacija.

Poglavlje 5

Prostorni upiti

Prostorni upiti su ključni dio proširenja za rad s prostornim podacima jer nam oni omogućavaju utvrđivanje odnosa između prostornih objekata. U ovom poglavlju opisane su neke od karakterističnih funkcija u prostornim upitima te je dano nekoliko primjera prostornih upita u SQL-u. Sve funkcije i primjeri navedeni u ovom poglavlju odnose se na PostGIS proširenje jer je ono korišteno u izradi aplikacije, ali sve ove funkcije dostupne i u ostalim proširenjima samo sa drugačijom sintaksom.

5.1 Prostorni odnosi

Prije govora o samim prostornim upitima potrebno je opisati načine određivanja prostornih odnosa. Prema *OGC Simple Feature* specifikaciji osnovni pristup usporedbi dviju geometrija je napraviti testove po parovima sjecišta između unutrašnjosti, granica i eksterijera dviju geometrija i klasificirati odnos između dviju geometrija na temelju unosa u rezultirajućoj matrici sjecišta.

Točke u geometriji koje su u dvodimenzionalnom prostoru mogu se podijeliti u tri grupe:

- Granica: granica geometrije je skup geometrija koji se nalazi u nižoj dimenziji. Za točke koje su dimenzije 0, granica je prazan skup, dok za niz povezanih točaka granica su krajnje točke, a za poligon, granica je skup vanjskih i unu-

tarnjih prstenova.

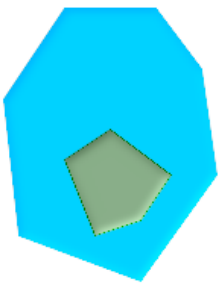
- Unutrašnjost: unutrašnjost geometrije su točke koje nisu u granici. Unutrašnjost točke je sama točka, dok unutrašnjost niza povezanih točaka su sve točke između dvije krajnje točke, a za poligon je to površina unutar poligona.
- Eksterijer: eksterijer geometrije su sve točke koje nisu na granici ili u unutrašnjosti geometrije.

5.2 Funkcije u prostornim upitima

Sva proširenja za podršku za rad s prostornim podatcima nude mnogobrojne funkcije od kojih će se samo nekoliko najosnovnijih i najkorištenijih funkcija obraditi u ovom poglavlju.

5.2.1 ST_Contains i ST_Within

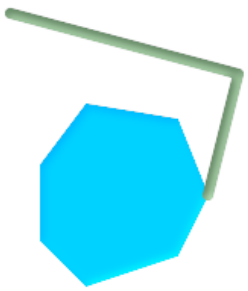
ST_Contains kao argumente prima geometriju A i geometriju B i vraća istinu, ako je geometrija B potpuno unutar geometrije A. Geometrija A sadržava geometriju B, ako niti jedna točka geometrije B ne leži u eksterijeru geometrije A (Slika 5.1). Funkcija ST_Within je inverz od funkcije ST_Contains pa ako promijenimo redoslijed argumenata vrijedi ST_Contains(A,B) = ST_Within(B,A). Ove funkcije automatski koriste prostorne indekse ako oni postoje za tu geometriju.



Slika 5.1 *Geometrija B (zelena) u potpunosti sadržana u geometriji A (plava)* [3]

5.2.2 ST_Touches

Ova funkcija vraća istinu ako se geometrije A i B sijeku, ali bez da se i njihove unutrašnjosti sijeku. Odnosno, to znači da geometrije moraju imati barem jednu zajedničku točku, ali zajedničke točke moraju biti u presjeku njihovih granica te ne smiju nikako biti u presjeku njihovih unutrašnjosti (Slika 5.2).



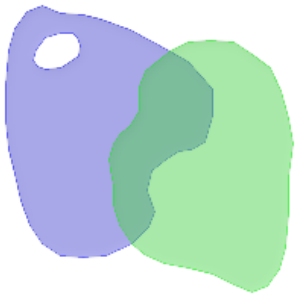
Slika 5.2 *Geometrija A (plava) i B (zelena) se dodiruju* [3]

5.2.3 ST_DistanceSphere

ST_DistanceSphere računa udaljenost između dvije točke u metrima. Točke su zadane u obliku zemljopisne dužine i širine (engl. longitude, latitude). Za razliku od funkcije ST_Distance funkcije, ST_DistanceSphere koristi za izračun sferni oblika zemlje i radijus izveden iz sferoida koji je definiran SRID-om. Zbog ovog načina izračuna ST_DistanceSphere je puno precizniji od ST_Distance funkcije, ali kao rezultat toga je sporija od ST_Distance funkcije.

5.2.4 ST_Overlaps

Kao i prijašnje funkcije, funkcija ST_Overlaps kao argumente prima dvije geometrije te vraća istinu ako se geometrije prostorno preklapaju. Razlika između ove funkcije i funkcije ST_Contains je ta što se geometrije sijeku, ali jedna geometrija ne smije u potpunosti sadržavati drugu geometriju (Slika 5.3). Prostorni indeksi se automatski primjenjuju ako postoje.



Slika 5.3 *Geometrija A (ljubičasta) i B (zelena) se prostorno preklapaju* [3]

5.2.5 ST_Intersects

Funkcija ST_Intersects kao argumente može primiti ili dvije geometrije ili dvije geografije. Dvije geometrije ili geografije se sijeku ako dijele bilo koji dio prostora, odnosno, ako funkcija ST_Contains ili ST_Overlaps ili ST_Touches vrati istinu. Za geografiju, tolerancija je 0.00001 metara. Ova funkcija, kao i prije navedene funkcije, automatski koristi prostorne indekse ako su dostupni.

5.3 Primjeri prostornih upita u SQL-u

U ovom potpoglavlju prikazat će se primjeri nekoliko najčešćih i najosnovnijih prostornih upita. Koristiti će se funkcije koje su prije spomenute u poglavlju kao i neke nove funkcije.

Upit za pronalazak najbližeg susjeda je najosnovniji i najčešći upit u prostornim bazama te je ovaj upit korišten i u izradi same aplikacije. Cilj ovog prostornog upita (Ispis 5.1) je pronaći zračnu luku koja je najbliža trenutnoj lokaciji zrakoplova. U upitu prvo se odabire identifikator i udaljenost svih zračnih luka od zrakoplova te se nakon toga rezultati sortiraju uzlazno po udaljenosti pa se na kraju odabire samo prvi zapis.

```
1 SELECT id,
2 ST_DistanceSphere($airplane_location, airport_location)
```

```
AS udaljenost
FROM geoms ORDER BY udaljenost ASC LIMIT 1
```

Ispis 5.1 Upit za pronalazak najbližeg susjeda

Cilj sljedećeg upita (Ispis 5.2) je pronaći koliko je dug tok rijeke Save koji prolazi kroz Hrvatsku.

```
SELECT
sum(ST_Length(r.the_geom))/1000 AS kilometers
FROM
  rijeke r,
  drzave d
WHERE
  r.name = 'Sava'
  AND d.name = 'Hrvatska'
  AND ST_Intersects(d.the_geom, r.the_geom);
```

Ispis 5.2 Pronalazak duljine toka rijeke Save koji prolazi kroz Hrvatsku

U upitu se uz pomoć funkcije ST_Length računa duljina rijeke u kilometrima koja ima ime Sava i koja siječe državu s imenom Hrvatska, odnosno koja dijeli prostor s Hrvatskom.

Cilj novog upita (Ispis 5.3) je pronaći državu u kojoj se nalazi rijeka Mississippi. Za provjeru je li jedna geometrija sadržana u drugoj geometriji koristi se funkcija ST_Within ili ST_Contains.

```
SELECT r.name, d.name
FROM
    rijeke r,
    drzave d
WHERE
    r.name = 'Mississippi'
    AND ST_Within(r.the_geom, d.the_geom)
```

Ispis 5.3 Pronalazak države kroz koju teče Mississipi

Upit prikazan na Ispisu 5.4 pronalazi sve rijeke koje teku kroz Hrvatsku, ali i kroz ostale zemlje odnosno pronalaze se sve rijeke čiji tok nije potpuno sadržan u Hrvatskoj. To se radi uz pomoć funkcije ST_Overlaps koja provjerava sijeku li se geometrije bez da je bilo koja rijeka u potpunosti sadržana unutar Hrvatske.

```
1 SELECT r.name
2 FROM rijeke r, drzave d
3 WHERE
4 d.name = 'Hrvatska'
5 AND ST_Overlaps(d.the_geom, r.the_geom);
```

Ispis 5.4 Pronalazak svih rijeka čiji tok nije potpuno sadržan u Hrvatskoj

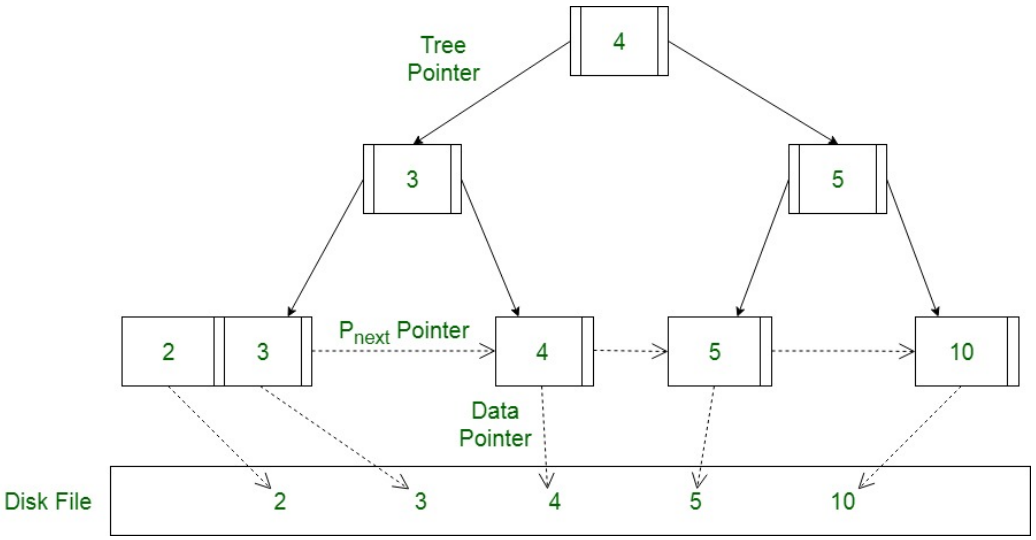
5.4 Prostorni indeksi

Prostorni indeksi su važan dio prostornih upita jer nam oni omogućavaju da optimiziramo brzinu izvršavanja pojedinih upita. U ovom potpoglavlju opisana su stabla pretrage koja čine temelj indeksa koji se koriste u prethodno opisanim proširenjima.

5.4.1 B+ stablo

B stablo je jedno od prvih razvijenih stabala za pretragu. Zbog toga što je uravnoteženo stablo visine, odlično je za primjenu kod prijenosa podataka u ulazu/izlazu diska gdje se prijenos vrši u jedinici stranice. Sastoji se od korijena, unutarnjih čvorova i listova stabla. U B+ stabla podatci se spremaju samo u listove stabla gdje su listovi stabla međusobno povezani u vezanim listama. Zbog toga što su podatci spremljeni samo u listovima stabla pojednostavljuje pretragu stabla.

Na Slici 5.4 može se vidjeti struktura B stabla pretraživanja gdje svaki čvor ima pokazivač (engl. Tree pointer) na sljedeći čvor osim listova koji imaju pokazivače na datoteke (engl. Data pointer) na disku (engl. Disk file) i na susjedne listove (engl. P Pointer).



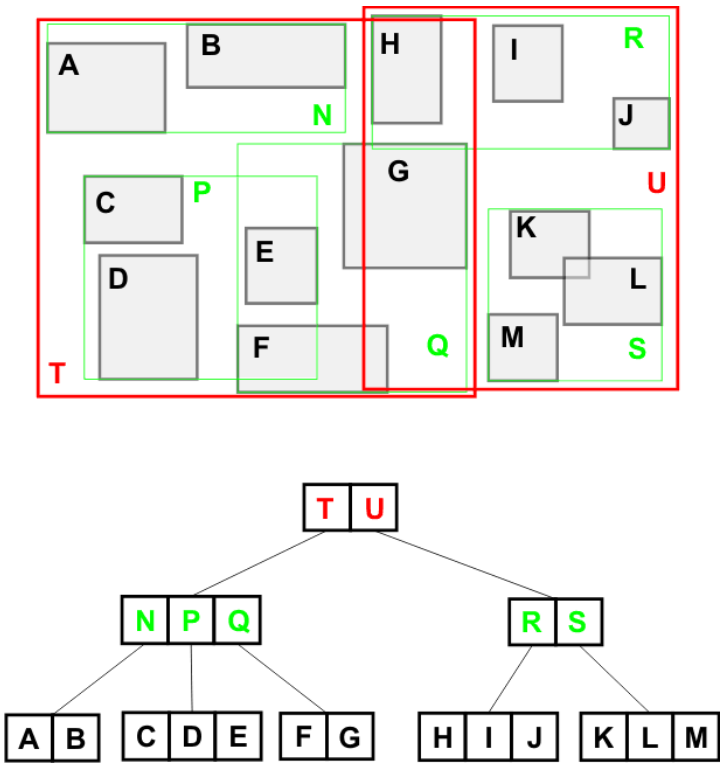
Slika 5.4 *B+ stablo pretraživanja*

5.4.2 R stablo

R stablo temelji se na B stablu, ali za razliku od B stabla koje je jednodimenzionalno, R stablo je multidimenzionalno. Pošto se temelji na B stablu, R stablo je isto balansirano stablo pretrage, odnosno, svi listovi čvorova nalaze se na istoj dubini. Glavni dio R stabla čine minimalno ograničavajući pravokutnici koje se koriste za grupiranje objekata koji su blizu jedan drugome. U listovima čvorovi se sastoje od identifikatora podatkovnih objekta i minimalno ograničavajućeg pravokutnika. U čvorovima koji nisu listovi nalaze se pokazivači koji pokazuju na niže čvorove u R stablu i minimalno ograničavajući pravokutnici koji sadrže sve minimalno ograničavajuće pravokutnike iz nižih čvorova.

Algoritam pretrage za R stablo je izrazito jednostavan. Pretraga stabla kreće od korijena te se rekurzivno pretražuju podstabla minimalno ograničavajućih pravokutnika (Slika 5.5). Pretražuju se samo oni minimalno ograničavajući pravokutnici koji se sijeku s traženim pravokutnikom koji, kada se dođe do lista čvora, testiraju s traženim pravokutnikom.

R stabla su najpraktičnija kad se radi o velikom broju podataka te je ne moguće spremiti cijelo stablo u memoriju jer R stablo učitava čvorove samo onda kad su oni



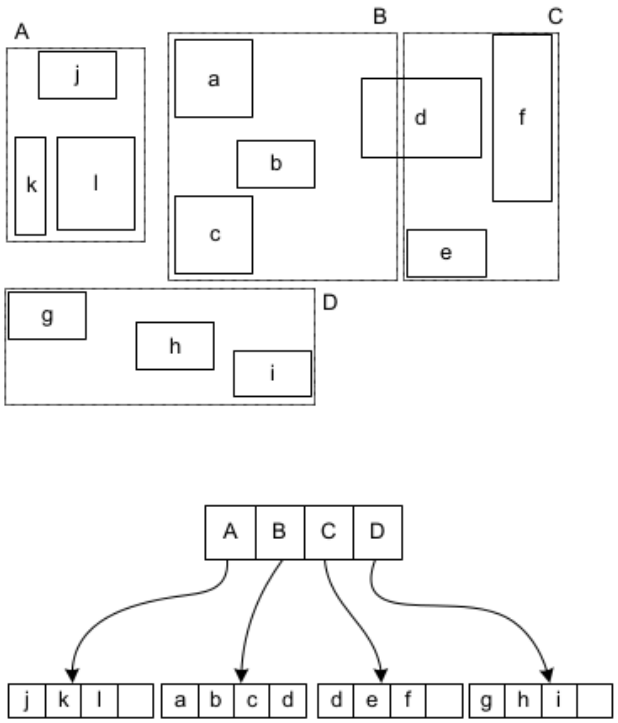
Slika 5.5 *R stablo pretraživanja*

potrebni. R stablo unatoč svim svojim prednostima nije bez nedostataka. Glavni problemi su preklapanje čvorova, površina prekrivena čvorom i veličina pravokutnika pojedinog čvora. Za rješenje ovih problema pojavilo se nekoliko varijanti R stabla od kojih su najvažnije dvije vrste: R+ i R* stablo.

5.4.3 R+ stablo

R+ stablo javilo se kao rješenje za problem preklapanja minimalno ograničavajućih pravokutnika. R+ ima nekoliko ograničenja koja sprečavaju mogućnost preklapanja, a to su: čvorovi ne moraju biti do pola puni, identifikator objekta može biti spremljen u više listova i unosi unutarnjih čvorova se ne preklapaju. Spremanje objektnih identifikatora na više mjesta sprečava preklapanje objekata. Pošto nema preklapanja čvorova, performanse upita su bolje te se prati samo jedan put i posjeti se manji

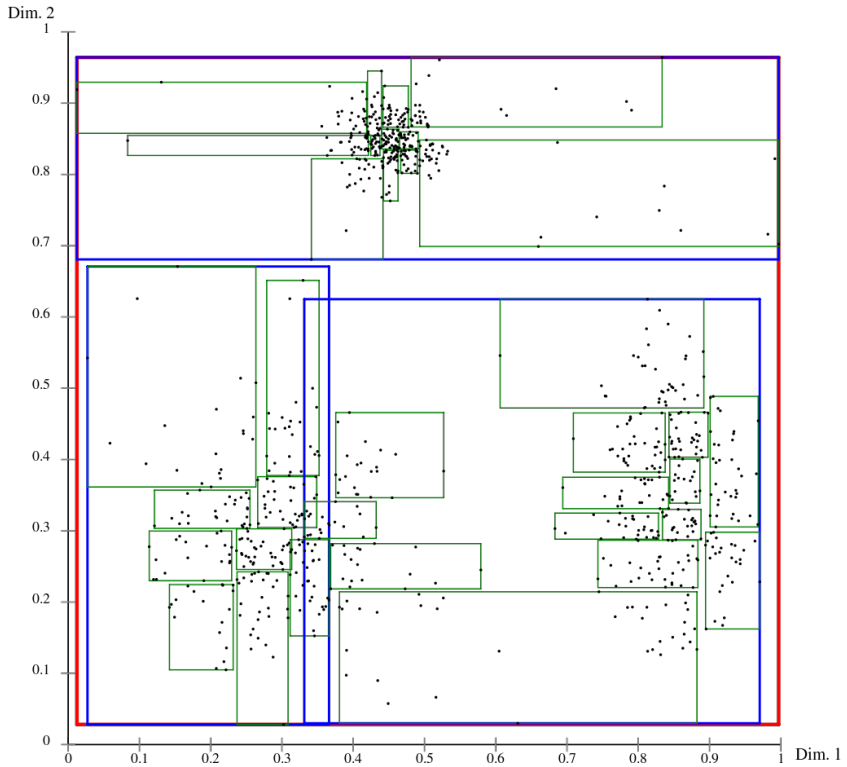
broj čvorova. Posljedica svega toga je veća cijena pretrage R+ stabla u odnosu na R stablo. Na Slici 5.6 prikazano je R+ stablo pretraživanja.



Slika 5.6 *R+ stablo pretraživanja*

5.4.4 R* stablo

R* stablo pokušava poboljšati R stablo uz pomoć novog algoritma za podjelu čvorova te prisiljavanjem ponovnog umetanja istih objekata kada dođe do ažuriranja strukture podataka. R* stablo podjelu čvorova vrši na jednoj osi te istražuje sve moguće objekte iznad i ispod te osi. Primjena metode ponovnog umetanja objekata poveća složenost stabla, ali ujedno i optimizira postojeće stablo. Na Slici 5.7 dan je vizualni prikaz R* stabla pretraživanja.



Slika 5.7 *R* stablo pretraživanja*

Poglavlje 6

Izrada sustava za praćenje mobilnih objekata u stvarnom vremenu zasnovano na prostornoj bazi podataka

Na temelju usvojenih znanja implementiran je sustav za praćenje mobilnih objekata u stvarnome vremenu. U ovom poglavlju opisan je tehnološki stog aplikacije, model baze podataka koji je korišten te karakteristični slučajevi korištenja sustava na klijentskoj strani.

6.1 Tehnološki stog

U razvoju aplikacije koristile su se različite tehnologije. Na poslužiteljskoj strani za sustav za upravljanje bazom podataka korišten je PostgreSQL 14 te, kako bi se omogućila podrška za rad s prostornim podatcima, proširenje PostGIS. Prilikom razvoja aplikacije za administraciju baze podataka korištena je *pgAdmin4* platforma. Kao poslužitelj za web stranicu korišten je *Apache server* koji je instaliran kao dio *Xampp* paketa. *Xampp* aplikacija dolazi sa već zadanim sustavom za upravljanje bazom podataka. Kako bi se omogućilo korištenje PostgreSQL-a uz *Xampp* paket

potrebno je PostgreSQL instalirati unutar *Xampp* direktorija. Nadalje ujedno je potrebno i konfigurirati php.ini datoteku u *Xampp* direktoriju. U *Xampp* paketu uključen je i PHP jezik čija je najnovija verzija korištena za izradu poslužiteljske strane aplikacije.

Na klijentskoj strani od programskih jezika korišteni su HTML, CSS i Javascript. U Javascriptu potrebno je naglasiti dvije važne knjižice: *Leaflet* i *Jquery*. *Leaflet* knjižica se koristila za prikaz karte svijeta i ujedno za prikaz oznaka zrakoplova i određivanje njihovih akcija na klijentskoj strani. U *Jquery* knjižici najbitnije su *ajax* funkcije koje su korištene za slanje poziva na poslužiteljsku stranu. Za izvor karte koja se prikazuje na klijentskoj strani korištena je stranica *Mapbox*.

6.2 Model baze podataka

Prije opisa svih funkcionalnosti aplikacije potrebno je prvo opisati konačni model baze podataka. Konačni model baze podataka sastoji se od četiri tablice: *airplane*, *airport*, *country_borders* i *no_fly_zone*.

Tablica airplane ukupno sadrži devet stupaca: id, name, speed, start_point, current_point, end_point, time, new_zone i old_zone. Među svim stupcima tablice korisnik zadaje podatke samo za name, speed, start_point i end_point prilikom kreiranja oznaka zrakoplova na klijentskoj strani. Vrijeme se računa preko brzine i udaljenosti između početne i krajnje točke dok se current_point periodično ažurira. Stupci new_zone i old_zone sadržavaju ime zone te služe za praćenje je li zrakoplov ušao u zonu sa zabranom leta. U ovoj tablici potrebno je izdvojiti start_point, current_point i end_point stupce jer su oni geometrijskog tipa, odnosno, oni su točke u prostoru.

Sljedeća tablica je airport koja je jednostavna tablica s četiri stupca: id, name, city i location. Za razliku od tablice airplane ovdje korisnik nema pravo kreiranja novih zračnih luka, nego sve dolaze unaprijed zadane s aplikacijom. Ovdje je stupac location geometrijskog tipa te je isto točka u prostoru.

Tablica country_borders kao što samo ime sugerira definira granice svih država u svijetu. Za kreiranje ove tablice korištena je datoteka sa skupom podataka koji

definira granice svih država u svijetu. Potrebno je izdvojiti stupac geom koji je kako samo ime sugerira geometrijskog tipa te određuje poligon koji definira granice pojedine države.

Na kraju imamo tablicu no_fly_zone u kojoj kao i u tablici airport korisnik može sam stvarati nove zone sa zabranom leta preko sučelja na klijentskoj strani. Tablica sadrži najmanje stupaca od svih tablica s ukupno tri stupca: id, name i location. Ovdje je stupac location geometrijskog tipa te definira poligon koji označava zonu sa zabranom leta.

6.3 Karakteristični slučajevi korištenja sustava na klijentskoj strani

Aplikacija podržava detekciju ciljanih događaja zasnovanih na međusobnom prostornom odnosu mobilnih objekata poput: provjere u kojoj državi se zrakoplov nalazi, koja je najbliža zračna luka za dani zrakoplov, provjera je li zrakoplov ušao u zonu sa zabranom leta... U potpoglavljima su opisani svi događaji koji su podržani te je uz svaki odgovarajući događaj dan vizualni prikaz te prostorni upit koji je na poslužiteljskoj strani.

6.3.1 Zadavanje rute

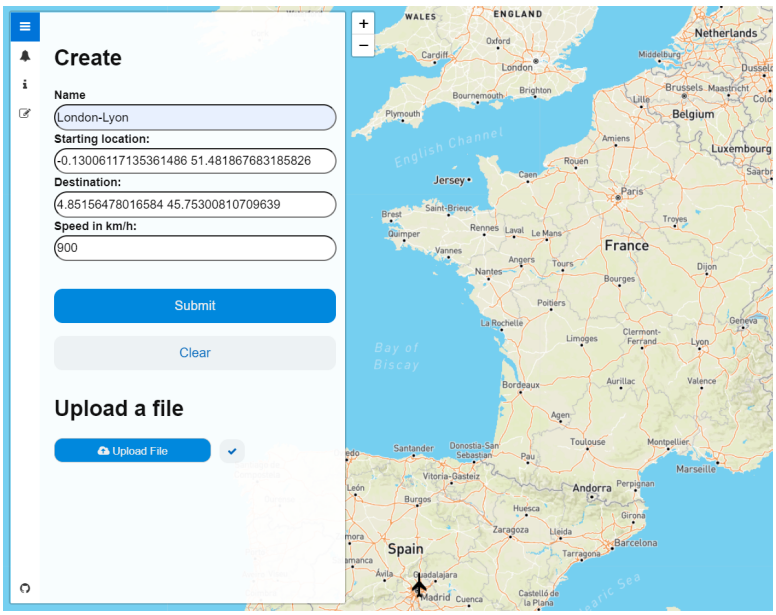
Prvi slučaj korištenja sustava na klijentskoj strani koji se detaljnije objašnjava je zadavanje rute. Zadavanje rute zrakoplova izvršava se prilikom kreiranja oznaka odnosno zrakoplova putem sučelja na klijentskoj strani. Prilikom stvaranja zrakoplova korisnik mu zadaje ime, početnu lokaciju, krajnju lokaciju i brzinu. Sve ove podatke korisnik može unijeti ručno upisivanjem vrijednosti u odgovarajuća polja za unose, ali korisnik može početnu i krajnju lokaciju unijeti i klikom na mapu. Kada korisnik klikne na mapu, sustav automatski zapisuje zemljopisnu širinu i dužinu u odgovarajuće polje. Nakon što korisnik ispuni sva polja i klikne *Submit* gumb na mapi, nastaje novi zrakoplov čija je ruta ravna linija između početne i krajnje točke. Osim putem sučelja za kreiranje oznaka zrakoplova korisnik može zadati rute i putem GeoJSON

datoteke koju može učitati putem *Upload* gumba.

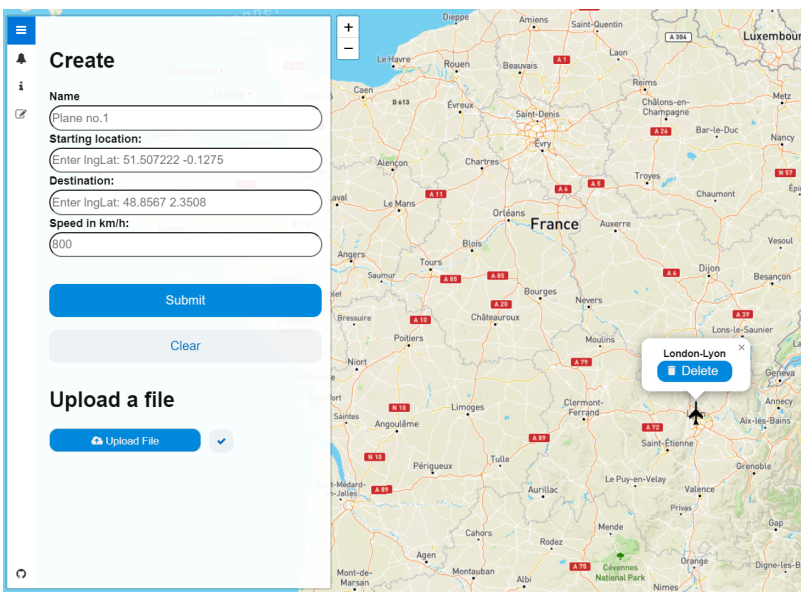
U pozadini, proces stvaranja novog zrakoplova sastoji se od tri upita (Ispis 6.1). U prvom upitu podatci koje je korisnik definirao (Slika 6.1), spremaju se u bazu, dok se u drugom upitu postavlja vrijednost *time* stupca čija se vrijednost računa preko distance između početne i odredišne točke te brzine zrakoplova. To određuje trajanje leta zrakoplova. Na kraju se u trećem upitu odabire novo ažurirano vrijeme kako bi se na klijentskoj strani mogla kreirati oznaka zrakoplova na mapi (Slika 6.2).

```
$query ="INSERT INTO
airplane(name,speed,start_point,end_point)
VALUES('$name','$speed',ST_GeomFromText('POINT($start_point)'),
ST_GeomFromText('POINT($end_point)'));";

$query .= "UPDATE airplane SET
time = ((ST_DistanceSphere(start_point, end_point)/1000)
/ (speed)) * 3600000 WHERE name = '$name';";

$query .= "SELECT time FROM airplane WHERE name='$name';";
```

Ispis 6.1 Upit za kreiranje novog zrakoplova



Slika 6.1 *Primjer kreiranja zrakoplova putem sučelja*



Slika 6.2 *Rezultantna oznaka zrakoplova*

6.3.2 Definiranje zona sa zabranom leta i detekcija događaja u tim zonama

Aplikacija dolazi s već predefiniranim zonama sa zabranom leta, ali korisnik može stvoriti vlastite zone putem sučelja u aplikaciji. Prilikom stvaranja nove zone korisnik mora definirati ime i točke poligona koji određuje oblik zone sa zabranom leta (Slika 6.3). Ovdje, za razliku od sučelja gdje korisnik stvara zrakoplove, korisnik mora ručno unijeti sve točke poligona. Prilikom unošenja točki poligona korisnik mora paziti da su početna i zadnja točka jednake, odnosno da je poligon zatvoren jer tako PostGIS definira poligon. Upit za stvaranje novih zona je jednostavan te se može vidjeti ispod (Ispis 6.2).

```
1 $query = "INSERT INTO no_fly_zone(name,location)
2 VALUES('$name',ST_GeomFromText('POLYGON(($polygon))') );";
```

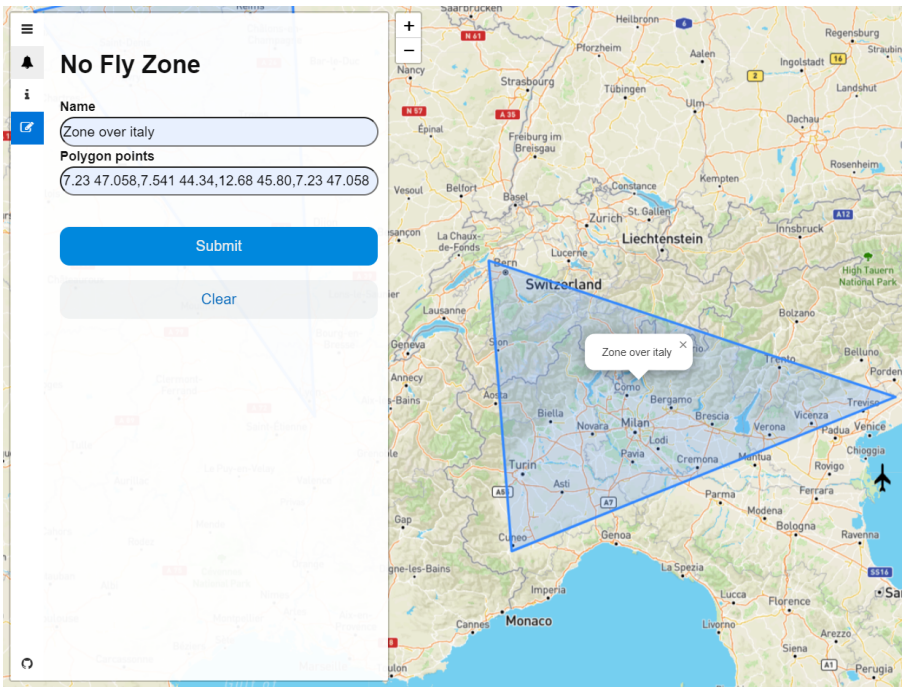
Ispis 6.2 Upit za kreiranje novih zona sa zabranom leta

Osim definiranja zona sa zabranom leta potrebno je spomenuti kako se očitavaju ulasci zrakoplova u zonu sa zabranom leta. Detekcija ulaska zrakoplova u zonu vrši se periodično kroz tri upita (Ispis 6.3). U prvom upitu vrijednost stupca *old_zone* ažurira se s vrijednosti new_zone stupca. U drugom upitu uz pomoć funkcije ST_Within provjerava se nalazi se zrakoplov u zoni sa zabranom leta. U slučaju da se zrakoplov nalazi u zoni ime u new_zone stupcu postavi se na ime trenutne zone, inače vrijednost se postavi na NULL. U trećem upitu odabire se ime zrakoplova i ime iz new_zone stupca ako vrijednost u new_zone stupcu nije NULL ili je vrijednost new_zone stupca različita od vrijednosti old_zone stupca. Ova provjera se izvršava zato što se provjera ulaska zrakoplova u zonu vrši periodično pa da se korisnika ne obavještava (Slika 6.4) više puta da se isti zrakoplov nalazi u istoj zoni sa zabranom leta.

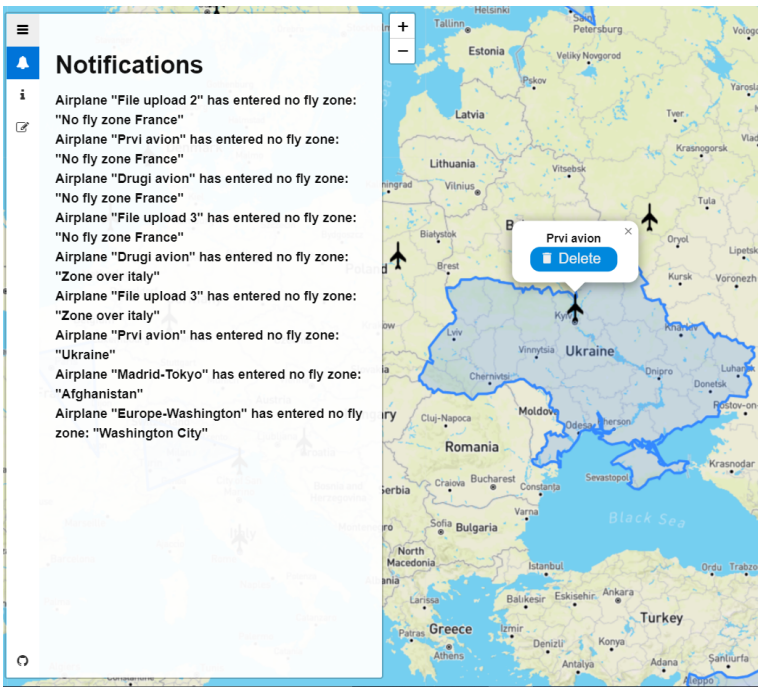
```
1 $query = "UPDATE airplane SET
2 old_zone = new_zone WHERE name = '$name';";
3
4 $query .= "UPDATE airplane C
5 SET new_zone = (SELECT B.name FROM airplane AS A,
6 no_fly_zone AS B WHERE A.name='$name'
```

```
AND ST_Within(A.current_point, B.location) is true )
WHERE C.name = '$name';";
$query .= "SELECT name, new_zone FROM airplane
WHERE name = '$name' AND new_zone IS NOT NULL  AND
(old_zone IS NULL OR new_zone NOT IN
(SELECT old_zone FROM airplane WHERE name='$name'));";
```

Ispis 6.3 Upit za detekciju ulaska zrakoplova u zonu sa zabranom leta



Slika 6.3 *Stvaranje zone sa zabranom leta i rezultantne zone*



Slika 6.4 *Obavijesti o ulasku zrakoplova u zonu sa zabranom leta*

6.3.3 Konkurentno praćenje objekata

Najvažniji dio aplikacije je konkurentno praćenje zrakoplova na mapi. Ovaj problem se rješava uz pomoć *Leaflet* funkcije getLatLng(). Za svaki objekt periodično se zove ova funkcija koja vraća trenutnu zemljopisnu širinu i dužinu objekta koja se nakon toga uz pomoć *ajax* poziva šalje na poslužiteljsku stranu aplikacije gdje se uz pomoć jednog upita (Ispis 6.4) ažurira vrijednost current_point stupca za svaki objekt.

Vrijednost current_stupca je ključni dio aplikacije jer se on koristi u svim ostalim upitima poput provjere nalazi li se zrakoplov u zoni sa zabranom leta, provjere iznad koje države zrakoplov trenutno leti, koja je najbliža zračna luka i slično.

```
1 $query = "UPDATE airplane set
2 current_point = 'POINT($lng $lat)' WHERE name='$name';";
```

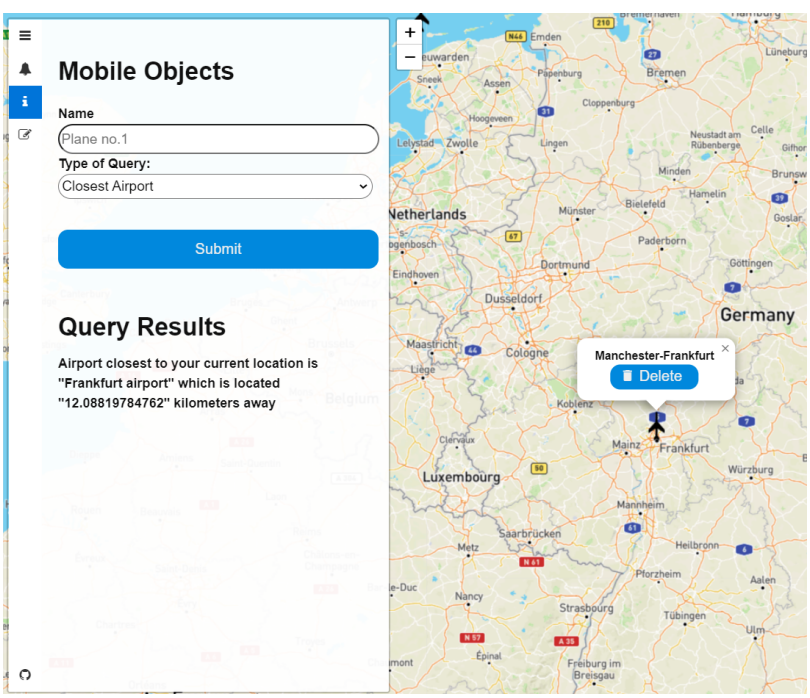
Ispis 6.4 Upit za ažuriranje trenutne pozicije

6.3.4 Pronalazak najbliže zračne luke

Jedna od funkcionalnosti koju aplikacija nudi je mogućnost pronalaska najbliže zračne luke za trenutnu poziciju zrakoplova. Pronalazak zračne luke obavlja se uz pomoć jednog upita (Ispis 6.5) s funkcijom ST_DistanceSphere() koja mjeri udaljenost između dvije točke odnosno udaljenost između trenutne lokacije zrakoplova i lokacije svih zračnih luka. Nakon toga podatci se sortiraju uzlazno po udaljenosti te se odabire prva zračna luka (Slika 6.5).

```
1 $query = "SELECT (ST_DistanceSphere(current_point, location))
     /1000 AS distance,
2 B.name FROM airplane AS A, airport AS B WHERE A.name='$name'
3 ORDER BY distance ASC LIMIT 1";
```

Ispis 6.5 Upit za pronalazak najbliže zračne luke



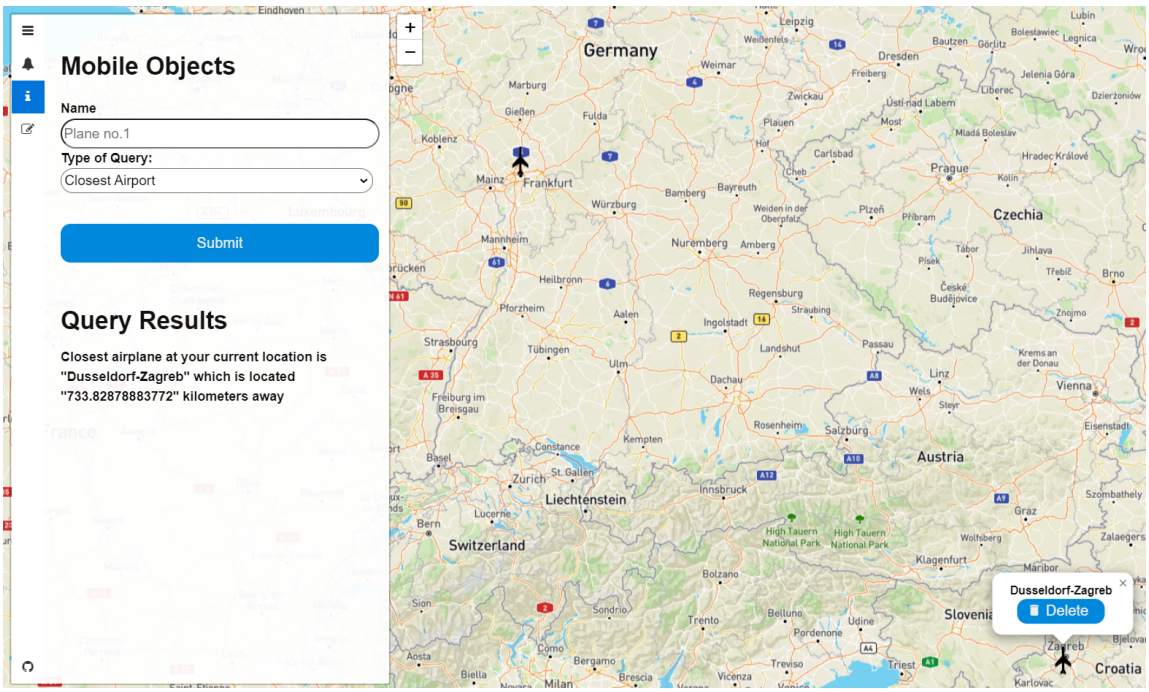
Slika 6.5 *Rezultat pronalaska najbliže zračne luke*

6.3.5 Pronalazak najbližeg susjednog zrakoplova

Ovaj slučaj korištenja sustava sličan je prijašnje opisanom slučaju pronalaska najbliže zračne luke, osim što se ne računa udaljenost između trenutne lokacije zrakoplova i lokacije zračnih luka, nego se računa udaljenost između trenutne pozicije odabranog zrakoplova i trenutne pozicije svih ostalih zrakoplova (Ispis 6.6). Na Slici 6.6 može se vidjeti rezultat upita.

```
$query = "SELECT (ST_DistanceSphere(A.current_point, B.
    current_point))/1000
AS distance, B.name FROM airplane AS A, airplane AS B
WHERE A.name='$name' ORDER BY distance ASC LIMIT 1 OFFSET 1;";
```

Ispis 6.6 Upit za pronalazak najbližeg zrakoplova



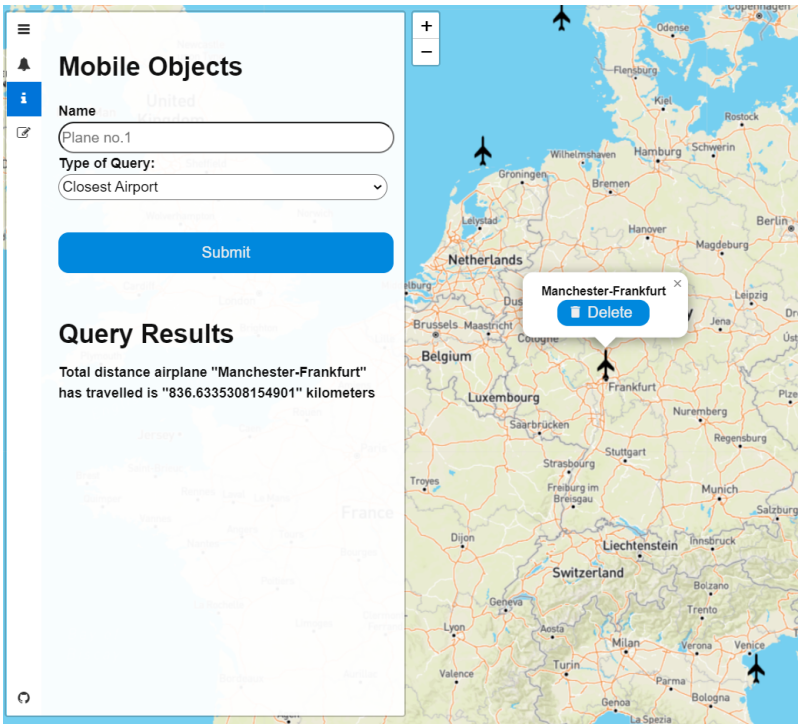
Slika 6.6 *Rezultat pronalaska najbližeg susjednog zrakoplova*

6.3.6 Udaljenost koju je zrakoplov prešao do određenog trenutka

Sukladno svojem imenu, ovaj slučaj korištenja računa koliku udaljenost je zrakoplov prešao do određenog trenutka. Ovaj slučaj je riješen uz pomoć jednostavnog upita (Ispis 6.7) koji je sličan dvama prethodnim upitima, ali je samo jednostavniji. U ovom upitu s funkcijom ST_DistanceSphere() računa se udaljenost od trenutne lokacije zrakoplova i njegove početne lokacije koja se onda dijeli s 1000 kako bi se dobila vrijednost u kilometrima (Slika 6.7).

```
1 $query = "SELECT (ST_DistanceSphere(start_point, current_point)
     )/1000
2 AS distance, name FROM airplane WHERE name='$name';";
```

Ispis 6.7 Upit za izračun udaljenosti koju je zrakoplov prešao



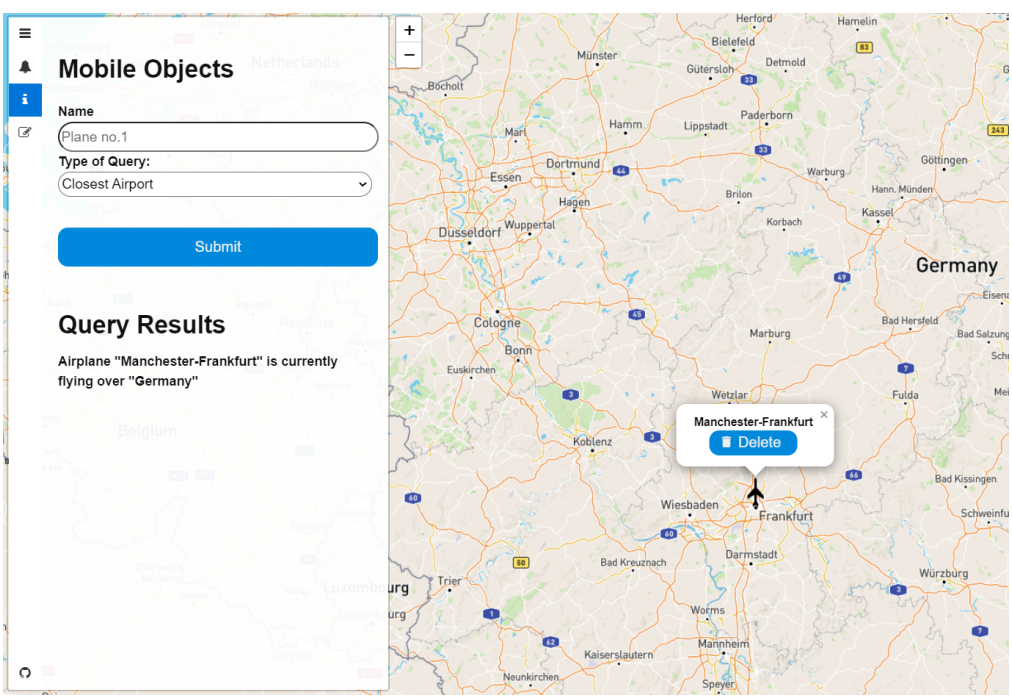
Slika 6.7 *Udaljenost koju je zrakoplov prešao*

6.3.7 Provjera u kojoj državi se zrakoplov nalazi

Način pronalaska države u kojoj se zrakoplov nalazi sličan je načinu provjere je li zrakoplov ušao u zonu sa zabranom leta, ali je jednostavniji. Jednostavniji je jer se upit (Ispis 6.8) za pronalazak države izvršava samo onda kad to korisnik odluči, dok se za provjeru zona sa zabranom leta upit periodično poziva. Pošto se upit ne izvršava periodično, nema potrebe za trima upitima niti pamtiti staru i novu vrijednost kao kod provjere zona. Dosta je jedan upit s funkcijom ST_Within() koji provjerava u kojoj državi se zrakoplov nalazi (Slika 6.8).

```
$query = "SELECT A.name ,B.name ,ST_Within(A.current_point, B.
    geom) as zemlja
FROM airplane AS A, country_borders AS B WHERE A.name='$name'
ORDER BY zemlja DESC LIMIT 1;";
```

Ispis 6.8 Upit za pronalazak u kojoj državi se zrakoplov nalazi



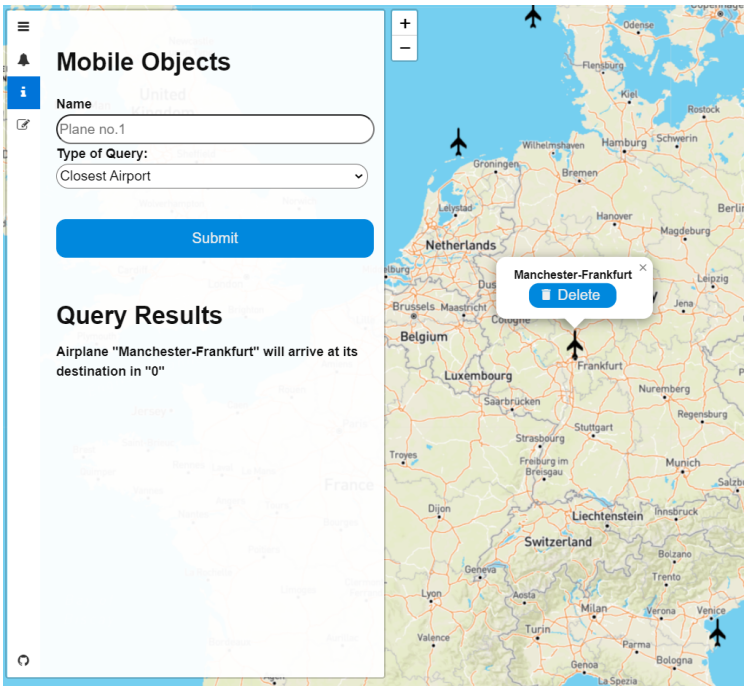
Slika 6.8 *Provjera u kojoj državi se zrakoplov nalazi*

6.3.8 Određivanje preostalog vremenskog trajanja leta

Zadnji karakteristični slučaj korištenja sustava na klijentskoj strani koji se razmatra je određivanje preostalog vremenskog trajanja leta (Slika 6.9). U ovom slučaju vrijeme se računa kao omjer udaljenosti od trenutne i konačne lokacije izračunate uz pomoć ST_DistanceSphere() funkcije i brzine (Ispis 6.9).

```
$query = "SELECT name,
((ST_DistanceSphere(current_point, end_point)/1000)/speed)
AS vrijeme FROM airplane WHERE name='$name';";
```

Ispis 6.9 Upit za određivanje preostalog vremena trajanja leta



Slika 6.9 *Preostalo vrijeme do kraja leta*

Poglavlje 7

Zaključak

Nakon što je u radu dan pregled karakterističnih prostornih podataka i specifičnih prostornih upita, moguće je uvidjeti prednosti korištenja prostornih baza podataka. Korištenje prostornih podataka omogućuje nam reprezentaciju prostornih objekata i njihovih odnosa kakvu ne možemo dobiti korištenjem standardne relacijske baze podataka. Nadalje, funkcije u prostornim bazama podataka omogućuju nam razinu analize podataka koju nikako ne možemo postići korištenjem baza podataka opće namjene. Na primjer, odgovor na pitanje kolika je površina dijela poligona A koja se nalazi unutar poligona B ne možemo nikako dobiti u općoj bazi jer funkcije potrebne za realizaciju ovog upita jednostavno nisu definirane.

Na temelju usvojenih znanja, implementirana je aplikacija za praćenje mobilnih objekata koja na poslužiteljskoj strani koristi prostornu bazu podataka. Za izradu aplikacije, od tri analizirana proširenja u trećem poglavlju, izabran je PostGIS jer nudi puno veći broj dostupnih vrsta podataka i funkcija. Osim toga, važno je istaknuti da PostGIS nudi brže performanse u odnosu na druga dva proširenja te nudi puno veću fleksibilnost u optimizaciji prostornih upita zbog većeg broja prostornih indeksa. Još jedan važan razlog za odabir PostGIS-a je puno veći doprinos zajednice koja je nastala oko PostGIS-a, a što omogućava lakše pronalaženje materijala za učenje i odgovora na moguće probleme. Detaljno su opisani slučajevi korištenja sustava na klijentskoj strani gdje su, uz vizualne primjere, dani i prostorni upiti koji se izvršavaju u pozadini. Na temelju ovih upita mogu se lakše razumjeti prednosti korištenja prostornih baza podataka u uspostavljanju odnosa između prostornih objekata.

Bibliografija

[1] ZPR-FER Napredni modeli i baze podataka 2018/2019 5. "Geoprostorne baze podataka", s Interneta, https://www.fer.unizg.hr/_download/repository/4._GIS.pdf, 30.7.2022

[2] Maguire, D.J.: "An overview and definition of GIS", in Geographical Information Systems: Principles and Applications, D. J.Maguire; M.F. Goodchild; D. Rhind, Editors, Wiley and Sons, Inc.: New York 1991.

[3] "PostGIS 3.2.4dev Manual", s Interneta, https://postgis.net/docs, 30.7.2022

[4] "Introduction to PostGIS", s Interneta, https://postgis.net/workshops/postgis-intro/, 30.7.2022.

[5] Obe, Regina O.; Hsu, Leo S.: "PostGIS in Action", Manning Publications, , Stamford, Connecticut, 2011.

[6] Aitchison, Alastair: "Beginning Spatial with SQL Server 2008", Springer Verlag, New York 2009.

[7] "Spatial Data SQL Server", s Interneta, https://docs.microsoft.com/en-us/sql/relational-databases/spatial/spatial-data-sql-server?view=sql-server-ver15, 2.8.2022.

[8] Aitchison, Alastair: "Pro Spatial with SQL Server 2012", Springer Verlag and Business Media New York, New York 2012.

[9] "MySQL 8.0 Reference Manual 11.4 Spatial Data Types", s Interneta, MySQL8.0ReferenceManual, 4.8.2022.

[10] "DM-07 - The Raster Data Model", s Interneta https://gistbok.ucgis.org/bok-topics/raster-data-model, 9.8.2022.

[11] Bertino, Elisa; Chin Ooi, Beng; Sacks-Davis, Ron i dr.: "Indexing Techinque For Advanced Database Systems", Springer Science and Business Media New York, New York 1997.

[12] Guttman, Antonin: "R Trees: A Dynamic Index Structure for Spatial Searching", s Interneta `https://www.researchgate.net/publication/221213205_R_Trees_A_Dynamic_Index_Structure_for_Spatial_Searching`, 11.8.2022

[13] "DM-66 - Spatial Indexing", s Interneta `https://gistbok.ucgis.org/bok-topics/spatial-indexing#Different`, 11.8.2022.

[14] Graefe, Goetz: "Modern B-Tree Techniques", Foundations and Trends in Databases, Vol. 3, No. 4, pp. 203–402, 2010.

Sažetak

U ovome radu analizirana su postojeća proširenja baza podataka opće namjene za podršku za rad s prostornim podacima. Opisani su karakteristični tipovi prostornih podataka te prostorni upiti i potporni indeksi koji se koriste u takvim proširenjima. Na temelju usvojenih znanja, implementiran je sustav za praćenje mobilnih objekata u stvarnom vremenu koji na poslužiteljskoj strani koristi model prostorne baze podataka zasnovan na proširenju PostGIS. Na klijentskoj strani realiziran je odgovarajući broj karakterističnih slučajeva korištenja koji demonstriraju mogućnosti upravljanja putanjama mobilnih objekata te detekcije ciljanih događaja zasnovanih na položaju i međusobnom prostornom odnosu mobilnih objekata.

Ključne riječi — **Prostorne baze podataka, prostorni podatci, prostorni upiti, PostGIS, praćenje mobilnih objekata**

Abstract

In this thesis, the existing extensions of general-purpose databases for working with spatial data are analyzed. Characteristic spatial data types, spatial queries, and the underlying spatial indexes utilized in such extensions are described. Based on the acquired knowledge, a system for real-time tracking of mobile objects was implemented, with its server side relying on a spatial database model based on the PostGIS extension. On the client side, an appropriate number of characteristic use cases have been provided that demonstrate the possibilities of managing the trajectories of mobile objects and detecting target events based on the position and spatial relationship of mobile objects.

Keywords — **Spatial databases, spatial data, spatial queries, PostGIS, mobile objects tracking**