**Tvrtka analizira podatke kako bi se identificirali ključni klijenti koji su generirali najveće troškove najma opreme i vozila tijekom odabrane vremenske periode. Analiza ponašanja visokopotrosačkih klijenata može pružiti dublji uvid u trendove na tržištu najma vozila i opreme te pomoći u donošenju informiranih poslovnih odluka.**

1. **PRIKAŽI KLIJENTE S NAJVIŠE POTROŠENOG NOVCA NA NAJAM VOZILA I OPREME**:

Da bismo pronašli klijenta s najviše potrošenog novca na najam vozila i opreme, možemo koristiti SQL upit koji će zbrojiti iznose transakcija za najam vozila i opreme za svakog klijenta te ih sortirati silazno prema ukupnom iznosu. Evo kako biste to mogli učiniti:

SELECT

k.id AS id\_klijent,

CONCAT(k.ime, ' ', k.prezime) AS ime\_prezime\_klijenta,

SUM(t.iznos) AS ukupni\_iznos

FROM

klijent k

INNER JOIN

najam\_vozila nv ON k.id = nv.id\_klijent\_najam

INNER JOIN

transakcija t ON nv.id\_transakcija\_najam = t.id

GROUP BY

k.id

ORDER BY

ukupni\_iznos DESC

LIMIT 1;

Ovaj upit će izračunati ukupni iznos potrošenog novca na najam vozila i opreme za svakog klijenta, grupirati rezultate po klijentu, sortirati ih silazno prema ukupnom iznosu te vratiti informacije o klijentu s najviše potrošenog novca na vrhu rezultata.

U ovom upitu koriste se INNER JOIN i GROUP BY.

Evo korak-po-korak objašnjenja upita:

SELECT: Prvo definiramo što želimo izvući iz baze podataka. U ovom slučaju, želimo dobiti ID klijenta, ime i prezime klijenta te ukupni iznos potrošenog novca.

FROM: Zatim navodimo iz kojih tablica želimo dohvatiti podatke. Koristimo tablice klijent, najam\_vozila i transakcija.

JOIN: Povezujemo tablice koristeći JOIN kako bismo dobili sve potrebne informacije. Spajamo klijent tablicu s najam\_vozila tablicom prema ID-u klijenta te zatim najam\_vozila tablicu s transakcija tablicom prema ID-u transakcije.

GROUP BY: Grupiramo rezultate po ID-u klijenta. To znači da će se sve transakcije za istog klijenta zbrojiti zajedno.

SUM: Koristimo funkciju SUM kako bismo zbrojili iznose transakcija za svakog klijenta.

ORDER BY: Sortiramo rezultate prema ukupnom iznosu silazno (od najvećeg prema najmanjem). To nam omogućuje da prvi rezultat bude klijent s najviše potrošenog novca.

LIMIT: Na kraju ograničavamo broj rezultata koje želimo vratiti na 1, jer nas zanima samo klijent s najviše potrošenog novca.

LIMIT 1;

Koristimo ORDER BY da sortiramo rezultate prema ukupnom iznosu u silaznom redoslijedu.

LIMIT 1 se koristi kako bismo ograničili rezultate samo na prvi redak, što će nam dati klijenta s najvećim ukupnim iznosom.

Ovaj SQL upit će vratiti informacije o klijentu koji je potrošio najviše novca na najam vozila i opreme. Evo što nam konkretno vraća:

id\_klijent: ID klijenta koji je potrošio najviše novca na najam vozila i opreme.

ime\_prezime\_klijenta: Ime i prezime klijenta koji je potrošio najviše novca.

ukupni\_iznos: Ukupni iznos novca koji je klijent potrošio na najam vozila i opreme.

Ovaj upit identificira klijenta koji je potrošio najviše novca na najam vozila i opreme tijekom određenog vremenskog razdoblja:

**Tvrtka želi redovito analizirati svoje poslovanje iz prethodnih godina kako bi dobila uvid u trendove i promjene u potražnji za najmom vozila. Žele znati kako im je išlo poslovanje u godinama prije ove da vide kako su napredovali.**

1. **PRIKAZI NAJMOVE VOZILA KOJI SU SE IZVRŠILI U POSLJEDNJIH 5 GODINA**

Ovaj upit je moguće modificirati da se prikažu svi najmovi vozila koji su se održali u posljednjih nekoliko godina samo promjenom znamenke u INTERVAL YEAR u upitu

– u ovom upitu smo selektirali sve atribute iz tablice najam\_vozila čiji atribut datum je za 5 godina

manji od današnjeg datuma.

SELECT \*

FROM najam\_vozila

WHERE datum\_pocetka > DATE\_SUB(NOW(), INTERVAL 5 YEAR);

Ovaj upit će izlistati sve najmove vozila koji su se dogodili u posljednjih 5 godina.

Ovaj upit selektira sve atribute iz tablice najam\_vozila gdje je atribut datum\_pocetka veći od datuma koji je dobiven oduzimanjem od trenutnog datuma intervala određenog brojem godina. Evo detaljnijeg opisa:

SELECT \*: Ova linija označava da želimo selektirati sve atribute iz tablice najam\_vozila.

FROM najam\_vozila: Navodi se ime tablice iz koje se selektiraju podaci, u ovom slučaju, najam\_vozila.

WHERE datum\_pocetka > DATE\_SUB(NOW(), INTERVAL NEKOLIK YEAR): Ovo je uvjetna klauzula koja ograničava rezultate na one redove gdje je vrijednost atributa datum\_pocetka veća od rezultata dobivenog oduzimanjem od trenutnog datuma intervala određenog brojem godina. NOW() vraća trenutni datum i vrijeme, a DATE\_SUB() funkcija oduzima od tog trenutnog datuma interval određen brojem godina.

**Unutar Tvrtke se sustavno analizirati broj najmova vozila po zaposleniku kako bi identificirala zaposlenike s najvećim rezultatima. Tvrtka želi prepoznati zaposlenika s najvećim brojem najmova vozila kao potencijalnog lidera ili mentora unutar tima jer posjeduju vještine i osobine koje su ključne za uspješno vođenje i poticanje drugih na postizanje rezultata. Također će ga adekvatno nagraditi u obliku povišice.**

1. **PRIKAŽI ZAPOSLENIKA KOJI JE IZVRŠIO NAJVEĆI BROJ NAJMA VOZILA**

SELECT z.\*

FROM najam\_vozila AS najam

RIGHT JOIN zaposlenik AS z ON z.id = najam.id\_zaposlenik

GROUP BY najam.id\_zaposlenik

ORDER BY COUNT(\*) DESC

LIMIT 1;

SELECT najam.id\_zaposlenik

FROM najam\_vozila AS najam

INNER JOIN zaposlenik AS z ON z.id = najam.id\_zaposlenik

GROUP BY najam.id\_zaposlenik

ORDER BY COUNT(\*) DESC

LIMIT 1;

Ovaj SQL upit prikazuje sve atribute zaposlenika koji je izvršio najveći broj najmova vozila. Evo detaljnijeg objašnjenja koraka:

SELECT najam.id\_zaposlenik: Ovaj dio upita određuje koje atribute želimo prikazati iz tablice zaposlenik.

FROM najam\_vozila AS najam: Ovdje se navodi glavna tablica iz koje želimo dohvatiti podatke, a to je najam\_vozila. Kao alias za ovu tablicu koristi se najam.

INNER JOIN zaposlenik AS z ON z.id = najam.id\_zaposlenik: Koristi se INNER JOIN kako bismo spojili tablice najam\_vozila i zaposlenik. zaposlenik AS z je još jedan alias koji daje tablici zaposlenik. Na kraju ove linije navodi se kako se ove dvije tablice povezuju. To znači da su redci u tablici najam\_vozila spojeni s redcima u tablici zaposlenik gdje je id u tablici zaposlenik jednak id\_zaposlenik u tablici najam\_vozila.

GROUP BY najam.id\_zaposlenik: Ova linija grupira rezultate prema atributu id\_zaposlenik iz tablice najam\_vozila. To osigurava da se brojanje vrši po zaposleniku.

ORDER BY COUNT(\*) DESC: Sortira rezultate silazno prema broju najmova vozila (izračunatom pomoću COUNT(\*)), kako bismo zaposlenika s najvećim brojem najmova stavili na vrh.

Koristimo ORDER BY kako bi sortirali rezultate. COUNT(\*) se koristi za brojanje redaka u svakoj grupi. DESC znači "silazno", što znači da želimo sortirati rezultate od najvećeg broja prema najmanjem.

LIMIT 1: Ograničava broj rezultata na jedan, što znači da će se prikazati samo zaposlenik s najvećim brojem najmova vozila.

**4.PRIKAŽI SVE KAMIONE KOJI SU BILI IZNAJMLJENI BAREM DVA PUTA**

SELECT k.\*

FROM vozilo\_na\_najmu vn

INNER JOIN kamion k ON vn.id\_vozilo = k.id\_vozilo

GROUP BY k.id

HAVING COUNT(\*) >= 2

Ovaj upit će vratiti sve kamione koji su bili iznajmljeni barem dva puta.

1. **NAPISI ZAPOSLENIKE POD NAZIVOM MARKO I KOJI JE PO ZANIMANJU EKONOMISTA**

SELECT z.\*, zan.opis\_zanimanja

FROM zaposlenik AS z

INNER JOIN zanimanje AS zan ON z.id\_zanimanje = zan.id

WHERE z.ime = 'Marko' AND zan.opis\_zanimanja = 'ekonomista';

**Tvrtka analizira transakcije izvršene putem kriptovaluta, pri čemu želi identificirati one koje nisu Bitcoin i čiji ukupni iznos plaćanja ne prelazi 90% ukupnog iznosa transakcije. Tvrtka se fokusira na praćenje financijskih operacija izvan tradicionalnih platnih sustava poput Bitcoin-a te analizira korištenje ostalih kriptovaluta u transakcijama, pružajući uvid u raznolikost kriptovalutnog tržišta i preferencije kupaca pri plaćanju. Dodatno, spajanje s tablicom najma vozila omogućuje tvrtki bolje razumijevanje konteksta transakcija, posebno u slučaju kada su kriptovalute korištene za plaćanje najma vozila, što može biti ključno za praćenje financijskih tokova i optimizaciju poslovanja.**

**6.PRIKAŽI SVE TRANSAKCIJE IZVRŠENE PUTEM KRIPTOVALUTA KOJE NISU BITCOIN, A UKUPNI IZNOS PLAĆANJA NIJE VEĆI OD 90% UKUPNOG IZNOSA TRANSAKCIJE:**

SELECT t.\*

FROM transakcija t

JOIN kriptovalutno\_placanje krp ON t.id = krp.id\_transakcija\_kripto

JOIN najam\_vozila nv ON t.id = nv.id\_transakcija\_najam

GROUP BY t.id

HAVING SUM(t.placeno) < 0.9 \* SUM(t.iznos) AND krp.kriptovaluta <> 'Bitcoin';

* **SELECT t.\*:** Ovaj dio odabire sve stupce iz tablice transakcija.
* **FROM transakcija t:** Definira se glavna tablica kao "t" koja je tablica transakcija.
* **JOIN kriptovalutno\_placanje krp ON t.id = krp.id\_transakcija\_kripto:** Spaja se tablica kriptovalutnih plaćanja na tablicu transakcija putem odgovarajućeg identifikatora transakcije.
* **JOIN najam\_vozila nv ON t.id = nv.id\_transakcija\_najam:** Dodaje se spajanje s tablicom najma vozila kako bi se dodao kontekst najma vozila transakcijama.
* **GROUP BY t.id:** Grupiranje rezultata prema identifikatoru transakcije.
* **HAVING SUM(t.placeno) < 0.9 \* SUM(t.iznos) AND krp.kriptovaluta <> 'Bitcoin':** Postavlja se uvjet koji provjerava da je ukupno plaćanje manje od 90% ukupnog iznosa transakcije i da kriptovaluta nije Bitcoin.

**Tvrtka prati kartične transakcije putem Visa ili Mastercard radi boljeg razumijevanja financijskih tokova. Posebno se fokusira na transakcije koje nisu povezane s pravnim osobama banke te izdvaja iznose veće od 500 eura radi detaljnijeg praćenja većih transakcija. Ova analiza omogućuje precizno knjiženje konta, otkrivanje trendova potrošnje te prilagodbu poslovnih strategija prema potrebama tržišta.**

**7.PRIKAŽI KOLIKO JE SVE SKUPA TRANSAKCIJA IZVRŠENO PUTEM KARTICE VISA ILI MASTERCARD, KOJE NISU POVEZANE S PRAVNOM OSOBOM BANKE I ČIJI JE UKUPNI IZNOS VEĆI OD 500EUR:**

SELECT SUM(t.iznos) AS ukupni\_iznos

FROM transakcija t

JOIN karticno\_placanje kp ON t.id = kp.id\_transakcija\_kartica

LEFT JOIN pravna\_osoba po ON kp.id\_pravna\_osoba\_banka = po.id

WHERE (kp.tip\_kartice = 'Visa' OR kp.tip\_kartice = 'Mastercard')

AND po.id IS NULL

GROUP BY t.id

HAVING SUM(t.iznos) > 500;

Ovaj upit izračunava sumu iznosa svih transakcija koje su obavljene putem kartica Visa ili Mastercard, ali nisu povezane s pravnom osobom banke. Klauzula LEFT JOIN koristi se kako bi se spojila tablica karticno\_placanje s tablicom pravna\_osoba kako bi se provjerilo jesu li transakcije povezane s bankom. U WHERE uvjetima specificiramo da su tip kartice Visa ili Mastercard i da pravna osoba banke nije pridružena transakciji. Nakon toga, koristimo GROUP BY da bismo grupirali rezultate prema identifikatoru transakcije, a HAVING klauzula osigurava da se uzimaju u obzir samo transakcije čiji je ukupni iznos veći od 1000 kuna.

Koristimo tablicu pravna\_osoba kako bismo provjerili je li transakcija povezana s pravnom osobom banke. Kada korisnik koristi karticu za plaćanje, ta transakcija može biti povezana s bankom ili nekim drugim pravnim subjektom koji pruža karticu, poput banke.

Kada tvrtka želi analizirati transakcije putem kartica Visa ili Mastercard koje nisu povezane s pravnom osobom banke, to može ukazivati na transakcije koje nisu tipične kartične transakcije, poput plaćanja putem internetskih platformi ili usluga koje ne koriste klasične bankarske institucije za obradu plaćanja. Stoga je provjera pravne osobe važna kako bi se identificirale takve transakcije. Ako pravna osoba nije pridružena transakciji, to znači da transakcija nije izvršena putem banke, što je bitno za analizu troškova i poslovnih aktivnosti tvrtke.

**7.IZLISTAJ SVE TRANSAKCIJE IZVRŠENE PUTEM KARTICE VISA ILI MASTERCARD, KOJE NISU POVEZANE S PRAVNOM OSOBOM BANKE I ČIJI JE UKUPNI IZNOS VEĆI OD 1000 KUNA: ??? ovo je staro**

SELECT t.\*

FROM transakcija t

LEFT JOIN karticno\_placanje kp ON t.id = kp.id\_transakcija\_kartica

LEFT JOIN gotovinsko\_placanje gp ON t.id = gp.id\_transakcija\_gotovina

JOIN najam\_vozila nv ON t.id = nv.id\_transakcija\_najam

GROUP BY t.id

HAVING (kp.tip\_kartice = 'Visa' OR kp.tip\_kartice = 'Mastercard')

AND kp.id\_pravna\_osoba\_banka IS NULL

AND SUM(t.iznos) > 1000;

* **SELECT t.\*:** Opet se odabiru svi stupci iz tablice transakcija.
* **FROM transakcija t:** Glavna tablica je opet tablica transakcija.
* **LEFT JOIN karticno\_placanje kp ON t.id = kp.id\_transakcija\_kartica:** Ovaj lijevi JOIN spaja transakcije s podacima o kartičnom plaćanju.
* **LEFT JOIN gotovinsko\_placanje gp ON t.id = gp.id\_transakcija\_gotovina:** Još jedan lijevi JOIN dodaje podatke o gotovinskom plaćanju.
* **JOIN najam\_vozila nv ON t.id = nv.id\_transakcija\_najam:** Dodaje se spajanje s tablicom najma vozila.
* **GROUP BY t.id:** Grupiranje rezultata prema identifikatoru transakcije.
* **HAVING (kp.tip\_kartice = 'Visa' OR kp.tip\_kartice = 'Mastercard') AND kp.id\_pravna\_osoba\_banka IS NULL AND SUM(t.iznos) > 1000:** Ovdje se postavljaju uvjeti za odabir transakcija koje koriste kartice Visa ili Mastercard, nisu povezane s pravnom osobom banke i imaju ukupni iznos veći od 1000 kuna.

**Tvrtka želi analizirati svoje poslovne troškove kako bi bolje razumjela gdje se trošio novac tokom prva 3mjeseca poslovanja u 2024.godini i kako bi identificirala područja u kojima može uštedjeti ili optimizirati troškove.**

**8. PRIKAŽI SVE POSLOVNE TROŠKOVE ZA REZERVACIJE KOJE SU SE DOGODILE U PRVOM TROMJESEČJU 2024.GOD:**

SELECT pt.\*

FROM poslovni\_trosak pt

JOIN transakcija t ON pt.id\_transakcija\_poslovnog\_troska = t.id

WHERE (MONTH(t.datum) = 1 OR MONTH(t.datum) = 2 OR MONTH(t.datum) = 3) AND YEAR(t.datum) = 2024;

Ovaj upit ima sljedeće korake:

1. **SELECT pt.\*:** Ovaj dio označava da želimo izlistati sve stupce (**\***) iz tablice **poslovni\_trosak**.
2. **FROM poslovni\_trosak pt:** Ovdje navodimo glavnu tablicu iz koje želimo dohvatiti podatke, a to je **poslovni\_trosak**. Kao alias za ovu tablicu koristimo **pt**.
3. **JOIN transakcija t ON pt.id\_transakcija\_poslovnog\_troska = t.id:** Ovo je ključni dio upita gdje spajamo tablicu **poslovni\_trosak** s tablicom **transakcija** na temelju njihovih ključeva. Spajamo ih tako da se vrijednost **id\_transakcija\_poslovnog\_troska** iz tablice **poslovni\_trosak** podudara s vrijednošću **id** u tablici **transakcija**.
4. **WHERE (MONTH(t.datum) = 1 OR MONTH(t.datum) = 2 OR MONTH(t.datum) = 3) AND YEAR(t.datum) = 2024;:** Ovo ograničenje filtrira rezultate prema datumu transakcije. Konkretno, želimo samo prvo tromjesečje u 2024. godina. Koristimo funkcije **MONTH(t.datum)**  **I YEAR(t.datum)** koje predstavljaju stupac datuma u tablici transakcija
5. **Upit koji izlistava klijente koji su najčešće iznajmljivali određenu vrstu opreme:**

SELECT k.ime, k.prezime, COUNT(r.id\_klijent) AS broj\_iznajmljivanja

FROM klijent k

INNER JOIN najam\_vozila nv ON k.id = nv.id\_klijent\_najam

INNER JOIN oprema\_na\_najmu r ON nv.id = r.id\_najam

INNER JOIN oprema o ON r.id\_oprema = o.id

WHERE o.naziv = 'Naziv\_određene\_vrste\_opreme'

GROUP BY k.id

ORDER BY broj\_iznajmljivanja DESC;

U ovom upitu:

* **klijent**, **najam\_vozila** i **oprema\_na\_najmu** tablice su spojene kako bi se došlo do informacija o klijentima koji su iznajmljivali određenu vrstu opreme.
* **oprema** tablica se koristi za filtriranje određene vrste opreme prema nazivu.
* **COUNT()** funkcija koristi se za brojanje koliko puta je svaki klijent iznajmio određenu vrstu opreme.
* **GROUP BY** klauzula grupira rezultate prema ID-u klijenta.
* **ORDER BY** klauzula sortira rezultate po broju iznajmljivanja od najvišeg prema najnižem.

1. **Upit koji pronalazi najpopularniji grad za najam vozila:**

SELECT z.grad\_radno\_mjesto AS Najpopularniji\_grad, COUNT(\*) AS Broj\_najmova

FROM zaposlenik z

INNER JOIN najam\_vozila nv ON z.id = nv.id\_zaposlenik\_najam

GROUP BY z.grad\_radno\_mjesto

ORDER BY Broj\_najmova DESC

LIMIT 1;

U ovom upitu:

* **zaposlenik** tablica se koristi kako bi se došlo do informacija o gradovima gdje se nalaze lokacije poslovnica tvrtke za najam vozila.
* **najam\_vozila** tablica se koristi kako bi se povezali gradovi sa stvarnim najmovima vozila.
* **COUNT(\*)** funkcija koristi se za brojanje koliko puta se svaki grad pojavljuje u rezultatima, odnosno koliko je najmova bilo u svakom gradu.
* **GROUP BY** klauzula grupira rezultate po gradovima.
* **ORDER BY** klauzula sortira rezultate po broju najmova od najvećeg prema najmanjem.
* **LIMIT 1** ograničava rezultate samo na prvi redak, tj. najpopularniji grad.

1. **Upit koji izlistava najam vozila koji je generirao najveći prihod u određenom razdoblju:**

SELECT nv.id AS ID\_najma,

nv.datum\_pocetka AS Datum\_pocetka,

nv.datum\_zavrsetka AS Datum\_zavrsetka,

SUM(t.iznos) AS Ukupni\_prihod

FROM najam\_vozila nv

JOIN transakcija t ON nv.id\_transakcija\_najam = t.id

WHERE nv.datum\_pocetka BETWEEN '2024-01-01' AND '2024-12-31'

GROUP BY nv.id, nv.datum\_pocetka, nv.datum\_zavrsetka

ORDER BY Ukupni\_prihod DESC

LIMIT 1;

U ovom upitu:

* **najam\_vozila** tablica se koristi kako bi se došlo do informacija o najmovima vozila.
* **transakcija** tablica se koristi kako bi se povezali najmovi vozila s transakcijama i izračunao ukupni prihod.
* **SUM(t.iznos)** funkcija koristi se za zbrajanje iznosa svih transakcija povezanih s pojedinim najmom vozila, što daje ukupni prihod.
* **WHERE** klauzula koristi se za filtriranje rezultata prema određenom razdoblju, odnosno datuma početka i završetka najma vozila.
* **GROUP BY** klauzula grupira rezultate po ID-u najma vozila, datumu početka i datumu završetka.
* **ORDER BY** klauzula sortira rezultate po ukupnom prihodu od najvećeg prema najmanjem.
* **LIMIT 1** ograničava rezultate samo na prvi redak, tj. najam s najvećim prihodom u određenom razdoblju.

**Prikaži ime grada i broj najmova za sve gradove koji imaju više od jednog najma vozila. ?????**

Ovaj upit će vratiti ime grada i broj najmova za sve gradove koji imaju više od jednog najma vozila. RIGHT JOIN se koristi kako bi se osiguralo da se u rezultat uključe i gradovi koji nemaju nikakvih najmova vozila, ali im je moguće pridružiti podatke iz tablice najam\_vozila. HAVING COUNT(\*) > 1 ograničava rezultate na one gradove koji imaju više od jednog najma vozila.

SELECT g.ime\_grada, COUNT(\*) AS broj\_najmova

FROM grad g

RIGHT JOIN najam\_vozila nv ON g.id\_grad = nv.id\_grad

GROUP BY g.ime\_grada

HAVING COUNT(\*) > 1;

Ovaj upit koristi RIGHT JOIN kako bi spojio tablicu **grad** s tablicom **najam\_vozila**, uzimajući sve retke iz tablice **najam\_vozila** i podudarajuće retke iz tablice **grad**.

Nakon spajanja, upit koristi GROUP BY kako bi grupirao rezultate po imenu grada (**ime\_grada**).

Zatim se koristi HAVING COUNT(\*) > 1 kako bi se ograničili rezultati na one gradove koji imaju više od jednog najma vozila. Ovo osigurava da se uključe samo gradovi koji su imali više od jednog najma vozila.

Konačno, SELECT naredba vraća ime grada (**ime\_grada**) i broj najmova vozila (**broj\_najmova**) za sve gradove koji zadovoljavaju kriterije.