Zaawansowane technologie bazodanowe - Indeksy

Zadanie 1(Przygotowanie bazy):

```
psql –f 1-ascii.sql;
psql –f 1-ascii.sql;
```

Zadanie 2:

(Polecenie ANALYZE)

```
kozumar1=> ANALYZE klienci;
ANALYZE
kozumar1=> ANALYZE kompozycje;
ANALYZE
kozumar1=> ANALYZE odbiorcy;
ANALYZE
kozumar1=> ANALYZE zamowienia;
ANALYZE
```

(Polecenie EXPLAIN):

```
kozumar1=> EXPLAIN SELECT * FROM klienci;
                       QUERY PLAN
-------------------------------------------------------------
 Seq Scan on klienci  (cost=0.00..1.50 rows=50 width=121)
(1 row)

kozumar1=> EXPLAIN(VERBOSE) SELECT * FROM klienci;
kozumar1=> EXPLAIN(VERBOSE) SELECT * FROM klienci;
                                    QUERY PLAN
-------------------------------------------------------------------------------------------
 Seq Scan on public.klienci  (cost=0.00..1.50 rows=50 width=121)
   Output: idklienta, haslo, nazwa, miasto, kod, adres, email, telefon, fax, nip, regon
(2 rows)
```

Zadanie 3(indeksy oparte o haszowanie):

```
EXPLAIN SELECT * FROM zamowienia WHERE idkompozycji = 'buk1';
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE idkompozycji = 'buk1';
                          QUERY PLAN
---------------------------------------------------------------
 Seq Scan on zamowienia  (cost=0.00..167.19 rows=424 width=52)
   Filter: (idkompozycji = 'buk1'::bpchar)
(2 rows)
```

```
CREATE INDEX zamowieniaIndex ON zamowienia USING
hash(idkompozycji);

EXPLAIN SELECT * FROM zamowienia WHERE idkompozycji = 'buk1';
```

```
kozumar1=> CREATE INDEX zamowieniaIndex ON zamowienia USING hash(idkompozycji);
WARNING:  hash indexes are not WAL-logged and their use is discouraged
CREATE INDEX
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE idkompozycji = 'buk1';
                                QUERY PLAN
---------------------------------------------------------------------------
 Bitmap Heap Scan on zamowienia  (cost=15.29..87.59 rows=424 width=52)
   Recheck Cond: (idkompozycji = 'buk1'::bpchar)
   ->  Bitmap Index Scan on zamowieniaindex  (cost=0.00..15.18 rows=424 width=0)
         Index Cond: (idkompozycji = 'buk1'::bpchar)
(4 rows)
```

Zadanie 4(Indeksy oparte o b-drzewa):

```
DROP INDEX zamowieniaIndex;
```

```
kozumar1=> DROP INDEX zamowieniaIndex;
DROP INDEX
```

```
CREATE INDEX bTreeIndex ON zamowienia USING btree(idkompozycji);
```

```
kozumar1=> CREATE INDEX bTreeIndex ON zamowienia USING btree(idkompozycji);
CREATE INDEX
```

```
EXPLAIN ANALYZE(SELECT * FROM zamowienia WHERE idkompozycji =
'buk1');
```

```
kozumar1=> EXPLAIN ANALYZE(SELECT * FROM zamowienia WHERE idkompozycji = 'buk1');
                                                        QUERY PLAN
---------------------------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on zamowienia  (cost=11.57..83.87 rows=424 width=52) (actual time=0.204..1.142 rows=424 loops=1)
   Recheck Cond: (idkompozycji = 'buk1'::bpchar)
   Heap Blocks: exact=67
   ->  Bitmap Index Scan on btreeindex  (cost=0.00..11.46 rows=424 width=0) (actual time=0.168..0.168 rows=424 loops=1)
         Index Cond: (idkompozycji = 'buk1'::bpchar)
 Planning time: 0.136 ms
 Execution time: 1.222 ms
(7 rows)
```

Zapytanie wyświetlające zamówienia na wszystkie kompozycje, których ID zaczyna się na litery stojące w alfabecie przed „b”:

```
EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE idkompozycji < 'b';
```

```
kozumar1=> EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE idkompozycji < 'b';
                                      QUERY PLAN
---------------------------------------------------------------------------------------
 Index Scan using btreeindex on public.zamowienia  (cost=0.28..8.30 rows=1 width=52)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Index Cond: (zamowienia.idkompozycji < 'b'::bpchar)
(3 rows)
```

Indeks nie jest wykorzystywany.

Zapytanie o pozostałe kompozycje, czyli „b” i kolejne litery:

```
EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE idkompozycji >= 'b';
```

```
kozumar1=> EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE idkompozycji >= 'b';
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Seq Scan on public.zamowienia  (cost=0.00..167.19 rows=8015 width=52)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Filter: (zamowienia.idkompozycji >= 'b'::bpchar)
(3 rows)
```

Użycie SET ENABLE_SEQSCAN TO OFF oraz powtórzenie powyższych zapytań:

```
kozumar1=> SET ENABLE_SEQSCAN TO OFF;
SET
kozumar1=> EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE idkompozycji < 'b';
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Index Scan using btreeindex on public.zamowienia  (cost=0.28..8.30 rows=1 width=52)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Index Cond: (zamowienia.idkompozycji < 'b'::bpchar)
(3 rows)

kozumar1=> SET ENABLE_SEQSCAN TO OFF;
SET
kozumar1=> EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE idkompozycji >= 'b';
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Bitmap Heap Scan on public.zamowienia  (cost=158.40..325.59 rows=8015 width=52)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Recheck Cond: (zamowienia.idkompozycji >= 'b'::bpchar)
   ->  Bitmap Index Scan on btreeindex  (cost=0.00..156.40 rows=8015 width=0)
         Index Cond: (zamowienia.idkompozycji >= 'b'::bpchar)
(5 rows)
```

Zadanie 5(indeksy a wzorce):

```
CREATE INDEX uwagiIndex ON zamowienia(uwagi);

EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE uwagi LIKE 'do%';
```

```
kozumar1=> CREATE INDEX uwagiIndex ON zamowienia(uwagi);
CREATE INDEX
kozumar1=> EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE uwagi LIKE 'do%';
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Seq Scan on public.zamowienia  (cost=10000000000.00..10000000167.19 rows=11 width=52)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Filter: ((zamowienia.uwagi)::text ~~ 'do%'::text)
(3 rows)
```

```
DROP INDEX uwagiIndex;

CREATE INDEX uwagiIndex2 ON zamowienia(uwagi varchar_pattern_ops);

EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE uwagi LIKE 'do%';
```

```
kozumar1=> DROP INDEX uwagiIndex;
DROP INDEX
kozumar1=> CREATE INDEX uwagiIndex2 ON zamowienia(uwagi varchar_pattern_ops);
CREATE INDEX
kozumar1=> EXPLAIN VERBOSE SELECT * FROM zamowienia WHERE uwagi LIKE 'do%';
                                                QUERY PLAN
-----------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on public.zamowienia  (cost=4.40..35.16 rows=11 width=52)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Filter: ((zamowienia.uwagi)::text ~~ 'do%'::text)
   ->  Bitmap Index Scan on uwagiindex2  (cost=0.00..4.39 rows=11 width=0)
         Index Cond: (((zamowienia.uwagi)::text ~>=~ 'do'::text) AND ((zamowienia.uwagi)::text ~<~ 'dp'::text))
(5 rows)
```

Zadanie 6(indeksy wielokolumnowe):

```
CREATE INDEX indexFor3 ON zamowienia(idklienta, idodbiorcy, idkompozycji);
```

```
kozumar1=> CREATE INDEX indexFor3 ON zamowienia(idklienta, idodbiorcy, idkompozycji);
CREATE INDEX
```

```
EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta = 'msowins' AND idodbiorcy = 1 AND idkompozycji = 'buk1');
```

```
kozumar1=> EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta = 'msowins' AND idodbiorcy = 1 AND idkompozycji = 'buk1');
                                                    QUERY PLAN
--------------------------------------------------------------------------------------------------------------------------------
 Index Scan using indexfor3 on public.zamowienia  (cost=0.28..8.30 rows=1 width=52) (actual time=0.079..0.080 rows=1 loops=1)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Index Cond: (((zamowienia.idklienta)::text = 'msowins'::text) AND (zamowienia.idodbiorcy = 1) AND (zamowienia.idkompozycji = 'buk1'::bpchar))
 Planning time: 0.211 ms
 Execution time: 0.121 ms
(5 rows)
```

```
EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta = 'msowins' OR idodbiorcy = 1 OR idkompozycji = 'buk1');
```

```
kozumar1=> EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta = 'msowins' OR idodbiorcy = 1 OR idkompozycji = 'buk1');
                                                    QUERY PLAN
--------------------------------------------------------------------------------------------------------------------------------
 Seq Scan on public.zamowienia  (cost=0.00..207.26 rows=843 width=52) (actual time=0.048..6.955 rows=841 loops=1)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Filter: (((zamowienia.idklienta)::text = 'msowins'::text) OR (zamowienia.idodbiorcy = 1) OR (zamowienia.idkompozycji = 'buk1'::bpchar))
   Rows Removed by Filter: 7174
 Planning time: 0.225 ms
 Execution time: 7.076 ms
(6 rows)
```

```
EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE

idkompozycji= 'buk1');
```

```
kozumar1=> EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idkompozycji= 'buk1');
                                                    QUERY PLAN
--------------------------------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on public.zamowienia  (cost=11.57..83.87 rows=424 width=52) (actual time=0.202..1.140 rows=424 loops=1)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Recheck Cond: (zamowienia.idkompozycji = 'buk1'::bpchar)
   Heap Blocks: exact=67
   ->  Bitmap Index Scan on btreeindex  (cost=0.00..11.46 rows=424 width=0) (actual time=0.164..0.164 rows=424 loops=1)
         Index Cond: (zamowienia.idkompozycji = 'buk1'::bpchar)
 Planning time: 0.176 ms
 Execution time: 1.219 ms
(8 rows)
```

```
DROP INDEX indexFor3;
```

```
kozumar1=> DROP INDEX indexFor3;
DROP INDEX
```

```
CREATE INDEX indexKlient ON zamowienia (idklienta);

CREATE INDEX indexOdbiorca ON zamowienia (idodbiorcy);
```

```
CREATE INDEX indexKompozycja ON zamowienia (idkompozycji);
```

```
kozumar1=> CREATE INDEX indexKlient ON zamowienia (idklienta);
CREATE INDEX
kozumar1=> CREATE INDEX indexOdbiorca ON zamowienia (idodbiorcy);
CREATE INDEX
kozumar1=> CREATE INDEX indexKompozycja ON zamowienia (idkompozycji);
CREATE INDEX
```

```
EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta =
'msowins' AND idodbiorcy = 1 AND idkompozycji = 'buk1');
```

```
kozumar1=> EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta = 'msowins' AND idodbiorcy = 1 AND idkompozycji = 'buk1');
                                                 QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on public.zamowienia  (cost=23.89..27.91 rows=1 width=52) (actual time=0.334..0.334 rows=1 loops=1)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Recheck Cond: (((zamowienia.idklienta)::text = 'msowins'::text) AND (zamowienia.idodbiorcy = 1) AND (zamowienia.idkompozycji = 'buk1'::bpchar))
   Heap Blocks: exact=1
   ->  BitmapAnd  (cost=23.89..23.89 rows=1 width=0) (actual time=0.319..0.319 rows=0 loops=1)
         ->  Bitmap Index Scan on indexklient  (cost=0.00..5.49 rows=161 width=0) (actual time=0.091..0.091 rows=161 loops=1)
               Index Cond: ((zamowienia.idklienta)::text = 'msowins'::text)
         ->  Bitmap Index Scan on indexodbiorca  (cost=0.00..6.43 rows=287 width=0) (actual time=0.078..0.078 rows=287 loops=1)
               Index Cond: (zamowienia.idodbiorcy = 1)
         ->  Bitmap Index Scan on indexkompozycja  (cost=0.00..11.46 rows=424 width=0) (actual time=0.129..0.129 rows=424 loops=1)
               Index Cond: (zamowienia.idkompozycji = 'buk1'::bpchar)
 Planning time: 0.263 ms
 Execution time: 0.391 ms
(13 rows)
```

```
EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta =
'msowins' OR idodbiorcy = 1 OR idkompozycji = 'buk1');
```

```
kozumar1=> EXPLAIN ANALYZE VERBOSE( SELECT * FROM zamowienia WHERE idklienta = 'msowins' OR idodbiorcy = 1 OR idkompozycji = 'buk1');
                                                 QUERY PLAN
-------------------------------------------------------------------------------------------------------------------------------
 Bitmap Heap Scan on public.zamowienia  (cost=24.02..106.28 rows=843 width=52) (actual time=0.363..1.466 rows=841 loops=1)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
   Recheck Cond: (((zamowienia.idklienta)::text = 'msowins'::text) OR (zamowienia.idodbiorcy = 1) OR (zamowienia.idkompozycji = 'buk1'::bpchar))
   Heap Blocks: exact=67
   ->  BitmapOr  (cost=24.02..24.02 rows=872 width=0) (actual time=0.330..0.330 rows=0 loops=1)
         ->  Bitmap Index Scan on indexklient  (cost=0.00..5.49 rows=161 width=0) (actual time=0.114..0.114 rows=161 loops=1)
               Index Cond: ((zamowienia.idklienta)::text = 'msowins'::text)
         ->  Bitmap Index Scan on indexodbiorca  (cost=0.00..6.43 rows=287 width=0) (actual time=0.078..0.078 rows=287 loops=1)
               Index Cond: (zamowienia.idodbiorcy = 1)
         ->  Bitmap Index Scan on indexkompozycja  (cost=0.00..11.46 rows=424 width=0) (actual time=0.136..0.136 rows=424 loops=1)
               Index Cond: (zamowienia.idkompozycji = 'buk1'::bpchar)
 Planning time: 0.245 ms
 Execution time: 1.602 ms
(13 rows)
```

Zadanie 7(indeksy a sortowanie):

```
EXPLAIN VERBOSE SELECT * FROM zamowienia ORDER BY idkompozycji;
```

```
kozumar1=> EXPLAIN VERBOSE SELECT * FROM zamowienia ORDER BY idkompozycji;
                                      QUERY PLAN
---------------------------------------------------------------------------------------------
 Index Scan using indexkompozycja on public.zamowienia  (cost=0.28..484.02 rows=8015 width=52)
   Output: idzamowienia, idklienta, idodbiorcy, idkompozycji, termin, cena, zaplacone, uwagi
(2 rows)
```

Indeks został wykorzystany

```
DROP INDEX indexKompozycja;
```

```
kozumar1=> DROP INDEX indexKompozycja;
DROP INDEX
```

```
EXPLAIN SELECT * FROM zamowienia ORDER BY idkompozycji;
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia ORDER BY idkompozycji;
                            QUERY PLAN
-------------------------------------------------------------------
 Sort  (cost=666.86..686.90 rows=8015 width=52)
   Sort Key: idkompozycji
   ->  Seq Scan on zamowienia  (cost=0.00..147.15 rows=8015 width=52)
(3 rows)
```

```
DROP INDEX indexKlient;

DROP INDEX indexOdbiorca;
```

```
kozumar1=> DROP INDEX indexKlient;
DROP INDEX
kozumar1=> DROP INDEX indexOdbiorca;
DROP INDEX
```

Zadanie 8(Indeksy częściowe):

```
CREATE INDEX indexKlient ON zamowienia(idklienta) WHERE zaplacone;
```

```
kozumar1=> CREATE INDEX indexKlient ON zamowienia(idklienta) WHERE zaplacone;
CREATE INDEX
```

```
EXPLAIN SELECT * FROM zamowienia WHERE idklienta='msowins' AND zapl
acone;
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE idklienta='msowins' AND zaplacone;
                                QUERY PLAN
---------------------------------------------------------------------------
 Bitmap Heap Scan on zamowienia  (cost=5.53..74.54 rows=161 width=52)
   Recheck Cond: (((idklienta)::text = 'msowins'::text) AND zaplacone)
   ->  Bitmap Index Scan on indexklient  (cost=0.00..5.49 rows=161 width=0)
         Index Cond: ((idklienta)::text = 'msowins'::text)
(4 rows)
```

```
EXPLAIN SELECT * FROM zamowienia WHERE idklienta='msowins' AND NOT
zaplacone;
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE idklienta='msowins' AND NOT zaplacone;
                                QUERY PLAN
---------------------------------------------------------------------------
 Bitmap Heap Scan on zamowienia  (cost=5.49..74.50 rows=1 width=52)
   Recheck Cond: ((idklienta)::text = 'msowins'::text)
   Filter: (NOT zaplacone)
   ->  Bitmap Index Scan on multicolumnindex  (cost=0.00..5.49 rows=161 width=0)
         Index Cond: ((idklienta)::text = 'msowins'::text)
(5 rows)
```

```
EXPLAIN SELECT sum(cena) FROM zamowienia WHERE NOT zaplacone;
```

```
kozumar1=> EXPLAIN SELECT sum(cena) FROM zamowienia WHERE NOT zaplacone;
                            QUERY PLAN
-------------------------------------------------------------------
 Aggregate  (cost=147.17..147.18 rows=1 width=5)
   ->  Seq Scan on zamowienia  (cost=0.00..147.15 rows=7 width=5)
         Filter: (NOT zaplacone)
(3 rows)
```

Nie korzysta z indeksu

Zadanie 9(Indeksy na wyrażeniach):

```
CREATE INDEX indexMiasto ON klienci (lower(miasto) varchar_pattern_ops);
```

```
kozumar1=> CREATE INDEX indexMiasto ON klienci (lower(miasto) varchar_pattern_ops);
CREATE INDEX
```

```
EXPLAIN SELECT * FROM klienci WHERE lower(miasto) LIKE 'krak%';
```

```
kozumar1=> EXPLAIN SELECT * FROM klienci WHERE lower(miasto) LIKE 'krak%';
                        QUERY PLAN
-----------------------------------------------------------
 Seq Scan on klienci  (cost=0.00..1.75 rows=1 width=692)
   Filter: (lower((miasto)::text) ~~ 'krak%'::text)
(2 rows)
```

Zadanie 10(Indeksy GiST):

```
ALTER TABLE zamowienia ADD COLUMN lokalizacja point;
```

```
kozumar1=> ALTER TABLE zamowienia ADD COLUMN lokalizacja POINT;
ALTER TABLE
```

```
UPDATE zamowienia SET lokalizacja=point(random()*100, random()*100)
;
```

```
kozumar1=> UPDATE zamowienia SET lokalizacja=point(random()*100, random()*100);
UPDATE 8015
```

```
EXPLAIN SELECT * FROM zamowienia WHERE lokalizacja <-> point(50,50)
<= 10;
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE lokalizacja <-> point(50,50) <= 10;
                              QUERY PLAN
-----------------------------------------------------------------------
 Seq Scan on zamowienia  (cost=0.00..278.23 rows=2672 width=68)
   Filter: ((lokalizacja <-> '(50,50)'::point) <= '10'::double precision)
(2 rows)
```

```
EXPLAIN SELECT * FROM zamowienia WHERE box'( (0,0), (50,50) )' @> l
okalizacja;
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE box'( (0,0), (50,50) )' @> lokalizacja;
                        QUERY PLAN
-----------------------------------------------------------
 Seq Scan on zamowienia  (cost=0.00..258.19 rows=8 width=68)
   Filter: ('(50,50),(0,0)'::box @> lokalizacja)
(2 rows)
```

```
create index indexLokalizacja ON zamowienia USING GIST (lokalizacja
);
```

```
kozumar1=> create index indexLokalizacja ON zamowienia USING GIST (lokalizacja);
CREATE INDEX
```

```
EXPLAIN SELECT * FROM zamowienia WHERE lokalizacja <-> point(50,50)
<= 10;
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE lokalizacja <-> point(50,50) <= 10;
                              QUERY PLAN
-----------------------------------------------------------------------
 Seq Scan on zamowienia  (cost=0.00..278.23 rows=2672 width=68)
   Filter: ((lokalizacja <-> '(50,50)'::point) <= '10'::double precision)
(2 rows)
```

```
EXPLAIN SELECT * FROM zamowienia WHERE box'( (0,0), (50,50) )' @> l
okalizacja;
```

```
kozumar1=> EXPLAIN SELECT * FROM zamowienia WHERE box'( (0,0), (50,50) )' @> lokalizacja;
                                QUERY PLAN
---------------------------------------------------------------------------
 Bitmap Heap Scan on zamowienia  (cost=4.21..30.91 rows=8 width=68)
   Recheck Cond: ('(50,50),(0,0)'::box @> lokalizacja)
   ->  Bitmap Index Scan on indexlokalizacja  (cost=0.00..4.21 rows=8 width=0)
         Index Cond: ('(50,50),(0,0)'::box @> lokalizacja)
(4 rows)
```