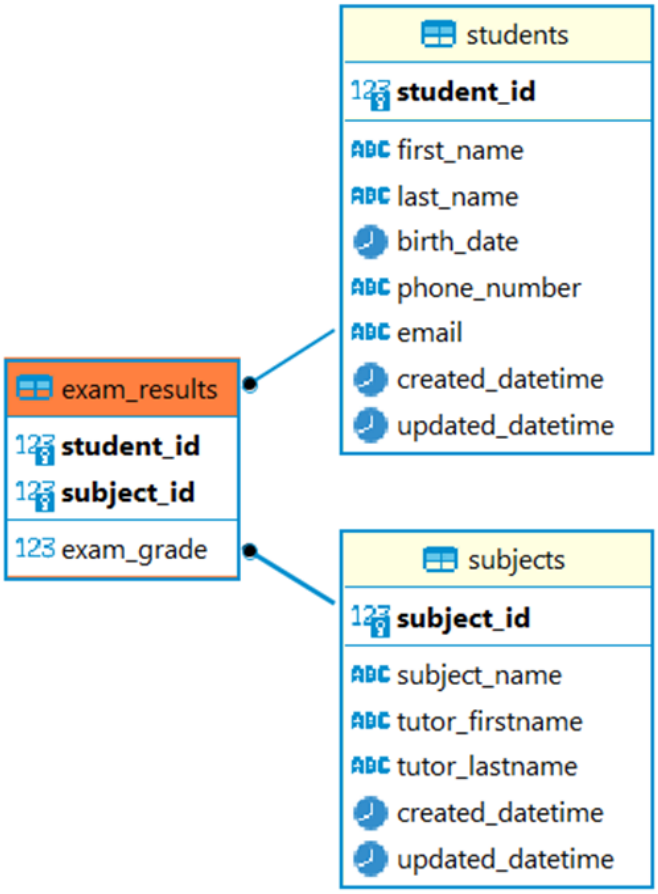
DB module

DB schema



Without indexes

```
explain analyze select * from students where first_name = 'Danny';

Seq Scan on students  (cost=0.00..2846.00 rows=29 width=94) (actual time=0.555..28.908
rows=38 loops=1)
  Filter: ((first_name)::text = 'Danny'::text)
  Rows Removed by Filter: 99962
Planning Time: 1.338 ms
Execution Time: 28.932 ms
```

```
explain analyze select * from students where last_name like '%Devona%';
```

```
Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.151..9.647
rows=26 loops=1)
  Filter: ((last_name)::text ~~ '%Devona%'::text)
  Rows Removed by Filter: 99974
Planning Time: 0.108 ms
Execution Time: 9.663 ms
```

```
explain analyze select * from students where phone_number like '%0886%';

Seq Scan on students  (cost=0.00..2846.00 rows=10 width=94) (actual time=0.186..12.378
rows=8 loops=1)
  Filter: ((phone_number)::text ~~ '%0886%'::text)
  Rows Removed by Filter: 99992
Planning Time: 0.045 ms
Execution Time: 12.392 ms
```

```
explain analyze select st, er.exam_grade from students st, exam_results er
        where st.student_id = er.student_id
        and st.last_name like '%Nicky%'
        and er.exam_grade > 3;

Nested Loop  (cost=4.51..3271.02 rows=36 width=122) (actual time=1.084..159.404 rows=288
loops=1)
  ->  Seq Scan on students st  (cost=0.00..2846.00 rows=9 width=122) (actual
time=0.225..9.777 rows=74 loops=1)
        Filter: ((last_name)::text ~~ '%Nicky%'::text)
        Rows Removed by Filter: 99926
  ->  Bitmap Heap Scan on exam_results er  (cost=4.51..47.18 rows=4 width=8) (actual
time=0.640..2.018 rows=4 loops=74)
        Recheck Cond: (student_id = st.student_id)
        Filter: (exam_grade > 3)
        Rows Removed by Filter: 6
        Heap Blocks: exact=730
        ->  Bitmap Index Scan on exam_results_pkey  (cost=0.00..4.51 rows=11 width=0)
(actual time=0.248..0.248 rows=10 loops=74)
              Index Cond: (student_id = st.student_id)
Planning Time: 2.410 ms
Execution Time: 159.492 ms
```

B-Tree index after data loaded

```
CREATE INDEX students_first_name_idx ON students (first_name);
CREATE INDEX students_last_name_idx ON students (last_name);
CREATE INDEX students_phone_number_idx ON students (phone_number);

students_first_name_idx Rel size: 752K
students_last_name_idx Rel size: 752K
students_phone_number_idx Rel size: 3.1M
```

```
explain analyze select * from students where first_name = 'Danny';

Bitmap Heap Scan on students  (cost=4.52..109.15 rows=29 width=94) (actual
time=0.021..0.057 rows=38 loops=1)
  Recheck Cond: ((first_name)::text = 'Danny'::text)
  Heap Blocks: exact=38
  ->  Bitmap Index Scan on students_first_name_idx  (cost=0.00..4.51 rows=29 width=0)
(actual time=0.016..0.016 rows=38 loops=1)
        Index Cond: ((first_name)::text = 'Danny'::text)
Planning Time: 0.211 ms
Execution Time: 0.070 ms
```

```
explain analyze select * from students where last_name like '%Devona%';

Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.103..9.949
rows=26 loops=1)
  Filter: ((last_name)::text ~~ '%Devona%'::text)
  Rows Removed by Filter: 99974
Planning Time: 0.087 ms
Execution Time: 9.963 ms
```

```
explain analyze select * from students where phone_number like '%0886%';

Seq Scan on students  (cost=0.00..2846.00 rows=10 width=94) (actual time=0.210..12.954
rows=8 loops=1)
  Filter: ((phone_number)::text ~~ '%0886%'::text)
  Rows Removed by Filter: 99992
Planning Time: 0.052 ms
Execution Time: 12.982 ms
```

```
explain analyze select st, er.exam_grade from students st, exam_results er
        where st.student_id = er.student_id
```

```
        and st.last_name like '%Nicky%'
        and er.exam_grade > 3;

Nested Loop  (cost=4.51..3271.02 rows=36 width=122) (actual time=0.238..10.616 rows=288
loops=1)
  ->  Seq Scan on students st  (cost=0.00..2846.00 rows=9 width=122) (actual
time=0.225..9.640 rows=74 loops=1)
        Filter: ((last_name)::text ~~ '%Nicky%'::text)
        Rows Removed by Filter: 99926
  ->  Bitmap Heap Scan on exam_results er  (cost=4.51..47.18 rows=4 width=8) (actual
time=0.006..0.012 rows=4 loops=74)
        Recheck Cond: (student_id = st.student_id)
        Filter: (exam_grade > 3)
        Rows Removed by Filter: 6
        Heap Blocks: exact=730
        ->  Bitmap Index Scan on exam_results_pkey  (cost=0.00..4.51 rows=11 width=0)
(actual time=0.003..0.003 rows=10 loops=74)
              Index Cond: (student_id = st.student_id)
Planning Time: 0.198 ms
Execution Time: 10.642 ms
```

B-Tree index before data loaded

```
All tables are created but data not imported

CREATE INDEX students_first_name_idx ON students (first_name);
CREATE INDEX students_last_name_idx ON students (last_name);
CREATE INDEX students_phone_number_idx ON students (phone_number);

students_first_name_idx Rel size: 1M
students_last_name_idx Rel size: 1.1M
students_phone_number_idx Rel size: 4.5M
```

```
explain analyze select * from students where first_name = 'Danny';

Bitmap Heap Scan on students  (cost=4.52..109.15 rows=29 width=94) (actual
time=0.022..0.057 rows=38 loops=1)
  Recheck Cond: ((first_name)::text = 'Danny'::text)
  Heap Blocks: exact=38
  ->  Bitmap Index Scan on students_first_name_idx  (cost=0.00..4.51 rows=29 width=0)
(actual time=0.016..0.016 rows=38 loops=1)
        Index Cond: ((first_name)::text = 'Danny'::text)
Planning Time: 0.061 ms
Execution Time: 0.071 ms
```

```
explain analyze select * from students where last_name like '%Devona%';

Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.107..10.084
rows=26 loops=1)
  Filter: ((last_name)::text ~~ '%Devona%'::text)
  Rows Removed by Filter: 99974
Planning Time: 0.066 ms
Execution Time: 10.099 ms
```

```
explain analyze select * from students where phone_number like '%0886%';

Seq Scan on students  (cost=0.00..2846.00 rows=10 width=94) (actual time=0.190..12.448
rows=8 loops=1)
  Filter: ((phone_number)::text ~~ '%0886%'::text)
  Rows Removed by Filter: 99992
Planning Time: 0.054 ms
Execution Time: 12.462 ms
```

```
explain analyze select st, er.exam_grade from students st, exam_results er
        where st.student_id = er.student_id
        and st.last_name like '%Nicky%'
        and er.exam_grade > 3;

Nested Loop  (cost=4.51..3271.02 rows=36 width=122) (actual time=0.247..12.122 rows=288
loops=1)
  ->  Seq Scan on students st  (cost=0.00..2846.00 rows=9 width=122) (actual
time=0.222..9.608 rows=74 loops=1)
        Filter: ((last_name)::text ~~ '%Nicky%'::text)
        Rows Removed by Filter: 99926
  ->  Bitmap Heap Scan on exam_results er  (cost=4.51..47.18 rows=4 width=8) (actual
time=0.012..0.033 rows=4 loops=74)
        Recheck Cond: (student_id = st.student_id)
        Filter: (exam_grade > 3)
        Rows Removed by Filter: 6
        Heap Blocks: exact=730
        ->  Bitmap Index Scan on exam_results_pkey  (cost=0.00..4.51 rows=11 width=0)
(actual time=0.006..0.006 rows=10 loops=74)
              Index Cond: (student_id = st.student_id)
Planning Time: 0.310 ms
Execution Time: 12.156 ms
```

Hash index after data loaded

```
All tables are seeded, previous index dropped

drop index if exists students_first_name_idx;
drop index if exists students_last_name_idx;
drop index if exists students_phone_number_idx;

create index students_first_name_hash_idx ON students USING HASH (first_name);
create index students_last_name_hash_idx ON students USING HASH (last_name);
create index students_phone_number_hash_idx ON students USING HASH (phone_number)

students_first_name_hash_idx Rel size: 4M
students_last_name_hash_idx Rel size: 4.1M
students_phone_number_hash_idx Rel size: 4M
```

```
explain analyze select * from students where first_name = 'Danny';

Bitmap Heap Scan on students  (cost=4.22..108.86 rows=29 width=94) (actual
time=0.025..0.069 rows=38 loops=1)
  Recheck Cond: ((first_name)::text = 'Danny'::text)
  Heap Blocks: exact=38
  ->  Bitmap Index Scan on students_first_name_hash_idx  (cost=0.00..4.22 rows=29 width=0)
(actual time=0.017..0.017 rows=38 loops=1)
        Index Cond: ((first_name)::text = 'Danny'::text)
Planning Time: 0.140 ms
Execution Time: 0.099 ms
```

```
explain analyze select * from students where last_name like '%Devona%';

Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.224..10.341
rows=26 loops=1)
  Filter: ((last_name)::text ~~ '%Devona%'::text)
  Rows Removed by Filter: 99974
Planning Time: 0.092 ms
Execution Time: 10.356 ms
```

```
explain analyze select * from students where phone_number like '%0886%';

Seq Scan on students  (cost=0.00..2846.00 rows=10 width=94) (actual time=0.202..12.683
rows=8 loops=1)
```

```
  Filter: ((phone_number)::text ~~ '%0886%'::text)
  Rows Removed by Filter: 99992
Planning Time: 0.054 ms
Execution Time: 12.697 ms
```

```
explain analyze select st, er.exam_grade from students st, exam_results er
        where st.student_id = er.student_id
        and st.last_name like '%Nicky%'
        and er.exam_grade > 3;

Nested Loop  (cost=4.51..3271.02 rows=36 width=122) (actual time=0.227..10.786 rows=288
loops=1)
  ->  Seq Scan on students st  (cost=0.00..2846.00 rows=9 width=122) (actual
time=0.214..9.655 rows=74 loops=1)
        Filter: ((last_name)::text ~~ '%Nicky%'::text)
        Rows Removed by Filter: 99926
  ->  Bitmap Heap Scan on exam_results er  (cost=4.51..47.18 rows=4 width=8) (actual
time=0.007..0.014 rows=4 loops=74)
        Recheck Cond: (student_id = st.student_id)
        Filter: (exam_grade > 3)
        Rows Removed by Filter: 6
        Heap Blocks: exact=730
        ->  Bitmap Index Scan on exam_results_pkey  (cost=0.00..4.51 rows=11 width=0)
(actual time=0.004..0.004 rows=10 loops=74)
              Index Cond: (student_id = st.student_id)
Planning Time: 0.189 ms
Execution Time: 10.812 ms
```

Hash index before data loaded

```
All tables are created -> indexes added -> data seeded

create index students_first_name_hash_idx ON students USING HASH (first_name);
create index students_last_name_hash_idx ON students USING HASH (last_name);
create index students_phone_number_hash_idx ON students USING HASH (phone_number)

students_first_name_hash_idx Rel size: 4.7M
students_last_name_hash_idx Rel size: 4.71M
students_phone_number_hash_idx Rel size: 4.6M
```

```
explain analyze select * from students where first_name = 'Danny';
```

```
Bitmap Heap Scan on students  (cost=4.22..108.86 rows=29 width=94) (actual
time=0.031..0.141 rows=38 loops=1)
  Recheck Cond: ((first_name)::text = 'Danny'::text)
  Heap Blocks: exact=38
  ->  Bitmap Index Scan on students_first_name_hash_idx  (cost=0.00..4.22 rows=29 width=0)
(actual time=0.020..0.020 rows=38 loops=1)
        Index Cond: ((first_name)::text = 'Danny'::text)
Planning Time: 0.319 ms
Execution Time: 0.166 ms
```

```
explain analyze select * from students where last_name like '%Devona%';

Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.109..10.695
rows=26 loops=1)
  Filter: ((last_name)::text ~~ '%Devona%'::text)
  Rows Removed by Filter: 99974
Planning Time: 0.060 ms
Execution Time: 10.711 ms
```

```
explain analyze select * from students where phone_number like '%0886%';

Seq Scan on students  (cost=0.00..2846.00 rows=10 width=94) (actual time=0.220..11.996
rows=8 loops=1)
  Filter: ((phone_number)::text ~~ '%0886%'::text)
  Rows Removed by Filter: 99992
Planning Time: 0.069 ms
Execution Time: 12.010 ms
```

```
explain analyze select st, er.exam_grade from students st, exam_results er
        where st.student_id = er.student_id
        and st.last_name like '%Nicky%'
        and er.exam_grade > 3;

Gather  (cost=1000.42..5877.02 rows=387 width=122) (actual time=0.370..9.549 rows=288
loops=1)
  Workers Planned: 1
  Workers Launched: 1
  ->  Nested Loop  (cost=0.42..4838.32 rows=228 width=122) (actual time=0.327..6.617
rows=144 loops=2)
        ->  Parallel Seq Scan on students st  (cost=0.00..2331.29 rows=56 width=122)
(actual time=0.297..5.612 rows=37 loops=2)
              Filter: ((last_name)::text ~~ '%Nicky%'::text)
```

```
              Rows Removed by Filter: 49963
        ->  Index Scan using exam_results_pkey on exam_results er  (cost=0.42..44.73
rows=4 width=8) (actual time=0.010..0.026 rows=4 loops=74)
              Index Cond: (student_id = st.student_id)
              Filter: (exam_grade > 3)
              Rows Removed by Filter: 6
Planning Time: 0.241 ms
Execution Time: 9.584 ms
```

GIN index

```
All tables are created -> data seeded -> indexes create

CREATE EXTENSION pg_trgm; //Using of gin indexes with varchar data type requires extension

create index students_first_name_gin_idx ON students USING GIN (first_name gin_trgm_ops);
create index students_last_name_gin_idx ON students USING GIN (last_name gin_trgm_ops);
create index students_phone_number_gin_idx ON students USING GIN (phone_number
gin_trgm_ops);

students_first_name_gin_idx Rel size: 2.5M
students_last_name_gin_idx Rel size: 2.4M
students_phone_number_gin_idx Rel size: 3.5M
```

```
explain analyze select * from students where first_name = 'Danny';

Bitmap Heap Scan on students  (cost=52.22..156.86 rows=29 width=94) (actual
time=0.182..0.227 rows=38 loops=1)
  Recheck Cond: ((first_name)::text = 'Danny'::text)
  Heap Blocks: exact=38
  ->  Bitmap Index Scan on students_first_name_gin_idx  (cost=0.00..52.22 rows=29 width=0)
(actual time=0.175..0.175 rows=38 loops=1)
        Index Cond: ((first_name)::text = 'Danny'::text)
Planning Time: 0.132 ms
Execution Time: 0.240 ms
```

```
explain analyze select * from students where last_name like '%Devona%';

Bitmap Heap Scan on students  (cost=36.07..70.16 rows=9 width=94) (actual
time=0.059..0.095 rows=26 loops=1)
  Recheck Cond: ((last_name)::text ~~ '%Devona%'::text)
  Heap Blocks: exact=26
```

```
  ->  Bitmap Index Scan on students_last_name_gin_idx  (cost=0.00..36.07 rows=9 width=0)
(actual time=0.049..0.050 rows=26 loops=1)
        Index Cond: ((last_name)::text ~~ '%Devona%'::text)
Planning Time: 0.109 ms
Execution Time: 0.116 ms
```

```
explain analyze select * from students where phone_number like '%0886%';

Bitmap Heap Scan on students  (cost=20.08..57.83 rows=10 width=94) (actual
time=0.042..0.053 rows=8 loops=1)
  Recheck Cond: ((phone_number)::text ~~ '%0886%'::text)
  Rows Removed by Index Recheck: 1
  Heap Blocks: exact=9
  ->  Bitmap Index Scan on students_phone_number_gin_idx  (cost=0.00..20.07 rows=10
width=0) (actual time=0.036..0.036 rows=9 loops=1)
        Index Cond: ((phone_number)::text ~~ '%0886%'::text)
Planning Time: 0.077 ms
Execution Time: 0.067 ms
```

```
explain analyze select st, er.exam_grade from students st, exam_results er
        where st.student_id = er.student_id
        and st.last_name like '%Nicky%'
        and er.exam_grade > 3;

Nested Loop  (cost=29.12..4321.82 rows=359 width=122) (actual time=0.104..1.837 rows=288
loops=1)
  ->  Bitmap Heap Scan on students st  (cost=28.69..316.87 rows=89 width=122) (actual
time=0.087..0.246 rows=74 loops=1)
        Recheck Cond: ((last_name)::text ~~ '%Nicky%'::text)
        Heap Blocks: exact=73
        ->  Bitmap Index Scan on students_last_name_gin_idx  (cost=0.00..28.67 rows=89
width=0) (actual time=0.065..0.066 rows=74 loops=1)
              Index Cond: ((last_name)::text ~~ '%Nicky%'::text)
  ->  Index Scan using exam_results_pkey on exam_results er  (cost=0.42..44.96 rows=4
width=8) (actual time=0.008..0.021 rows=4 loops=74)
        Index Cond: (student_id = st.student_id)
        Filter: (exam_grade > 3)
        Rows Removed by Filter: 6
Planning Time: 0.556 ms
Execution Time: 1.918 ms
```

GIsT index

```
All tables are created -> data seeded -> indexes create

CREATE EXTENSION pg_trgm; //Using of gin indexes with varchar data type requires extension

create index students_first_name_gist_idx ON students USING GIST (first_name
gist_trgm_ops);
create index students_last_name_gist_idx ON students USING GIST (last_name gist_trgm_ops);
create index students_phone_number_gist_idx ON students USING GIST (phone_number
gist_trgm_ops);

students_first_name_gist_idx Rel size: 5.9M
students_last_name_gist_idx Rel size: 5.8M
students_phone_number_gist_idx Rel size: 8.7M
```

```
explain analyze select * from students where first_name = 'Danny';

Bitmap Heap Scan on students  (cost=4.50..109.14 rows=29 width=94) (actual
time=2.845..2.878 rows=38 loops=1)
  Recheck Cond: ((first_name)::text = 'Danny'::text)
  Heap Blocks: exact=38
  ->  Bitmap Index Scan on students_first_name_gist_idx  (cost=0.00..4.50 rows=29 width=0)
(actual time=2.832..2.832 rows=38 loops=1)
        Index Cond: ((first_name)::text = 'Danny'::text)
Planning Time: 0.069 ms
Execution Time: 2.896 ms
```

```
explain analyze select * from students where last_name like '%Devona%';

Bitmap Heap Scan on students  (cost=4.35..38.43 rows=9 width=94) (actual time=2.509..2.543
rows=26 loops=1)
  Recheck Cond: ((last_name)::text ~~ '%Devona%'::text)
  Heap Blocks: exact=26
  ->  Bitmap Index Scan on students_last_name_gist_idx  (cost=0.00..4.35 rows=9 width=0)
(actual time=2.494..2.495 rows=26 loops=1)
        Index Cond: ((last_name)::text ~~ '%Devona%'::text)
Planning Time: 0.094 ms
Execution Time: 2.567 ms
```

```
explain analyze select * from students where phone_number like '%0886%';

Bitmap Heap Scan on students  (cost=4.36..42.11 rows=10 width=94) (actual
time=4.106..4.117 rows=8 loops=1)
  Recheck Cond: ((phone_number)::text ~~ '%0886%'::text)
  Rows Removed by Index Recheck: 1
  Heap Blocks: exact=9
  ->  Bitmap Index Scan on students_phone_number_gist_idx  (cost=0.00..4.36 rows=10
width=0) (actual time=4.093..4.093 rows=9 loops=1)
        Index Cond: ((phone_number)::text ~~ '%0886%'::text)
Planning Time: 0.081 ms
Execution Time: 4.144 ms
```

```
explain analyze select st, er.exam_grade from students st, exam_results er
        where st.student_id = er.student_id
        and st.last_name like '%Nicky%'
        and er.exam_grade > 3;

Nested Loop  (cost=5.39..4298.10 rows=359 width=122) (actual time=3.383..4.590 rows=288
loops=1)
  ->  Bitmap Heap Scan on students st  (cost=4.97..293.14 rows=89 width=122) (actual
time=3.367..3.509 rows=74 loops=1)
        Recheck Cond: ((last_name)::text ~~ '%Nicky%'::text)
        Heap Blocks: exact=73
        ->  Bitmap Index Scan on students_last_name_gist_idx  (cost=0.00..4.95 rows=89
width=0) (actual time=3.342..3.343 rows=74 loops=1)
              Index Cond: ((last_name)::text ~~ '%Nicky%'::text)
  ->  Index Scan using exam_results_pkey on exam_results er  (cost=0.42..44.96 rows=4
width=8) (actual time=0.006..0.014 rows=4 loops=74)
        Index Cond: (student_id = st.student_id)
        Filter: (exam_grade > 3)
        Rows Removed by Filter: 6
Planning Time: 0.194 ms
Execution Time: 4.621 ms
```

Conclusion

Index sizes

Field	B-tree	Hash	GIN	GiST
first_name	752K	4.7M	2.5M	5.9M
last_name	752K	4.71M	2.4M	5.8M
phone_number	3.1M	4.6M	3.5M	8.7M

Index sizes varies depending on was it created before or after data was seeded. Didn't found explanation for that.

Index types

- *B-Tree*is the default that you get when you do CREATE INDEX. Virtually all databases have some B-tree indexes. B-trees attempt to remain balanced, with the amount of data in each branch of the tree being roughly the same. Therefore the number of levels that must be traversed to find rows is always in the same ballpark. B-tree indexes can be used for equality and range queries efficiently. They can operate against all datatypes, and can also be used to retrieve NULL values. B-trees are designed to work very well with caching, even when only partially cached.
- *Hash Indexes*pre-Postgres 10 are only useful for equality comparisons, but you never want to use them since they aren't transaction safe, must be manually rebuilt after crashes, and aren't replicated to followers. So, the advantage over using a B-tree is rather small. In Postgres 10 and above, hash indexes are now write-ahead logged and replicated to followers.
- *Generalized Inverted Indexes (GIN)*are useful when an index must map many values to one row. Whereas B-tree indexes are optimized for when a row has a single key value. GINs are good for indexing array values as well as for implementing full-text search.
- *Generalized Search Tree (GiST)*indexes allow you to build general balanced tree structures, and can be used for operations beyond equality and range comparisons. They're used to index the geometric data types, as well as full-text search.

Notes

B-Tree and Hash indexes do not help you with partial search. To solve this it's possible to use GIN or GiST.

Without index	`explain analyze select * from students where last_name like '%Devona%';`
	
	`Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.107..10.084 rows=26 loops=1)`
	`  Filter: ((last_name)::text ~~ '%Devona%'::text)`
	`  Rows Removed by Filter: 99974`
	`Planning Time: 0.066 ms`
	`Execution Time: 10.099 ms`

B-Tree	`explain analyze select * from students where last_name like '%Devona%';`
	
	`Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.103..9.949 rows=26 loops=1)`
	`  Filter: ((last_name)::text ~~ '%Devona%'::text)`
	`  Rows Removed by Filter: 99974`
	`Planning Time: 0.087 ms`
	`Execution Time: 9.963 ms`

Hash	`explain analyze select * from students where last_name like '%Devona%';`
	
	`Seq Scan on students  (cost=0.00..2846.00 rows=9 width=94) (actual time=0.109..10.695 rows=26 loops=1)`
	`  Filter: ((last_name)::text ~~ '%Devona%'::text)`
	`  Rows Removed by Filter: 99974`
	`Planning Time: 0.060 ms`
	`Execution Time: 10.711 ms`

GIN	`explain analyze select * from students where last_name like '%Devona%';`
	
	`Bitmap Heap Scan on students (cost=36.07..70.16 rows=9 width=94) (actual time=0.059..0.095 rows=26 loops=1)`
	`  Recheck Cond: ((last_name)::text ~~ '%Devona%'::text)`
	`  Heap Blocks: exact=26`
	`  ->  Bitmap Index Scan on students_last_name_gin_idx (cost=0.00..36.07 rows=9 width=0) (actual time=0.049..0.050 rows=26 loops=1)`
	`  Index Cond: ((last_name)::text ~~ '%Devona%'::text)`
	`Planning Time: 0.109 ms`
	`Execution Time: 0.116 ms`

GiST	`explain analyze select * from students where last_name like '%Devona%';`
	
	`Bitmap Heap Scan on students (cost=4.35..38.43 rows=9 width=94) (actual time=2.509..2.543 rows=26 loops=1)`
	`  Recheck Cond: ((last_name)::text ~~ '%Devona%'::text)`
	`  Heap Blocks: exact=26`
	`  ->  Bitmap Index Scan on students_last_name_gist_idx (cost=0.00..4.35 rows=9 width=0) (actual time=2.494..2.495 rows=26 loops=1)`
	`  Index Cond: ((last_name)::text ~~ '%Devona%'::text)`
	`Planning Time: 0.094 ms`
	`Execution Time: 2.567 ms`

From the table above we can see that GIN and GiST indexes provides much more performance for the partial search.

At the same time for the exact matching B-Tree and Hash showing the better results.

Without index	`explain analyze select * from students where first_name = 'Danny';`
	
	`Seq Scan on students  (cost=0.00..2846.00 rows=29 width=94) (actual time=0.555..28.908 rows=38 loops=1)`
	`  Filter: ((first_name)::text = 'Danny'::text)`
	`  Rows Removed by Filter: 99962`
	`Planning Time: 1.338 ms`
	`Execution Time: 28.932 ms`

B-Tree	`explain analyze select * from students where first_name = 'Danny';`
	
	`Bitmap Heap Scan on students  (cost=4.52..109.15 rows=29 width=94) (actual time=0.022..0.057 rows=38 loops=1)`
	`  Recheck Cond: ((first_name)::text = 'Danny'::text)`
	`  Heap Blocks: exact=38`
	`  ->  Bitmap Index Scan on students_first_name_idx (cost=0.00..4.51 rows=29 width=0) (actual time=0.016..0.016 rows=38 loops=1)`
	`  Index Cond: ((first_name)::text = 'Danny'::text)`
	`Planning Time: 0.061 ms`
	`Execution Time: 0.071 ms`

Hash	`explain analyze select * from students where first_name = 'Danny';`
	
	`Bitmap Heap Scan on students  (cost=4.22..108.86 rows=29 width=94) (actual time=0.031..0.141 rows=38 loops=1)`
	`  Recheck Cond: ((first_name)::text = 'Danny'::text)`
	`  Heap Blocks: exact=38`
	`  ->  Bitmap Index Scan on students_first_name_hash_idx  (cost=0.00..4.22 rows=29 width=0) (actual time=0.020..0.020 rows=38 loops=1)`
	`  Index Cond: ((first_name)::text = 'Danny'::text)`
	`Planning Time: 0.319 ms`
	`Execution Time: 0.166 ms`

GIN	`explain analyze select * from students where first_name = 'Danny';`
	
	`Bitmap Heap Scan on students (cost=52.22..156.86 rows=29 width=94) (actual time=0.182..0.227 rows=38 loops=1)`
	`  Recheck Cond: ((first_name)::text = 'Danny'::text)`
	`  Heap Blocks: exact=38`
	`  ->  Bitmap Index Scan on students_first_name_gin_idx (cost=0.00..52.22 rows=29 width=0) (actual time=0.175..0.175 rows=38 loops=1)`
	`  Index Cond: ((first_name)::text = 'Danny'::text)`
	`Planning Time: 0.132 ms`
	`Execution Time: 0.240 ms`

GiST	`explain analyze select * from students where first_name = 'Danny';`
	
	`Bitmap Heap Scan on students (cost=4.50..109.14 rows=29 width=94) (actual time=2.845..2.878 rows=38 loops=1)`
	`  Recheck Cond: ((first_name)::text = 'Danny'::text)`
	`  Heap Blocks: exact=38`
	`  ->  Bitmap Index Scan on students_first_name_gist_idx (cost=0.00..4.50 rows=29 width=0) (actual time=2.832..2.832 rows=38 loops=1)`
	`  Index Cond: ((first_name)::text = 'Danny'::text)`
	`Planning Time: 0.069 ms`
	`Execution Time: 2.896 ms`

The main point is that the type of index should be chosen carefully depending on task you are going to solve.