Advanced Indexing: Takeaways

Syntax

- Creating an index:

```
CREATE INDEX state_index ON homeless_by_coc(state)
```

- Creating a multi-column index:

```
CREATE INDEX state_year_index ON homeless_by_coc(state, year)
```

- Creating a partial index:

```
CREATE INDEX state_count_index ON homeless_by_coc(state) WHERE count > 0
```

- Creating an index on a function:

```
CREATE INDEX measures_index ON homeless_by_coc(LOWER(measures))
```

Concepts

- Indexes create a B-tree structure on a column, which allows filtered queries to perform binary search.
- Indexes can significantly reduce query speeds.
- A `Bitmap Heap Scan` is used when Postgres sees that a lot of heap pages will need to be loaded. It works by loading each heap page containing results only once. These pages are quickly identified using the index.
- A `Bitmap Heap Scan` is more efficient than a pure `Seq Scan` because the number of filtered rows in an index will always be less than or equal to the number of rows in the full table.
- Postgres allows up to 32 columns to be indexed.
- Along with passing in additional options when creating an index, an index can be created using any Postgres function.

- Partial indexes restrict an index to a range of data and can be independent from the column that is being queried.

Resources

- Partial indexes
- Indexes on expressions