Week 6: Query Optimization

Database System Structure
- Data is stored as **blocks** (regions of data, usually 4K or 8K) in **pages** (for the intents and purposes of this class, just bigger regions of data for lookup) on **disk.**
- Data is accessed in a database by SQL Commands, which are in turn parsed by the database system
- **Database Index** - metadata in the database that keeps an ordered list of specific data in a table to optimize queries
    - Query optimizer works behind the scenes to decide which index to use given a query to optimize the time for fetching data.
    - *Ex.* a **primary key** is one example of what a database may index
    - Aggregate columns **cannot** be indexed
    - Columns of multiple tables **also cannot** be indexed together
- **B-Tree** - a type of self-balancing tree (a tree that keeps its children balanced at each level) that keeps track of ordering at each level, allowing for fast lookup
    - **Leaf nodes** (nodes with no children) are connected by a **doubly linked list**
    - Indexing in a B-Tree works like so:
        - Start at the **root** (the top node that does not have a parent)
        - With your **search key**, or the value you are searching for, look at the current node's key. For your search key **s** and the current node's key **k**,
            - If s < k, go down the left child
            - If s >= k, go down the right child
        - Continue until you reach the leaves, which should contain your data. If it doesn't, then your search query returned no results.

Query Analysis in Postgres
- **EXPLAIN ANALYZE** - used to see the statistics of a certain query; that is, the steps the database went through to perform the query, and how long it took.

```
EXPLAIN ANALYZE SELECT menu_item FROM Kerbey_Lane WHERE
cost = 'cheap'
```

- ○ *Ex.* this query will show me how the database pulled cheap menu items from Kerbey Lane, and how long it took. Because of the nature of Kerbey Lane, the SELECT query will probably not return anything.
- **CREATE INDEX** - used to create user-defined indexes on columns

```
CREATE INDEX index_name ON table_name(column_name);
```

- ○ Indexes like these will speed up performance on queries that reference the column by precalculating certain values (i.e sorting them)