Describe the differences between the various types of indexes presented during lecture and identify why each is preferred in certain read or write scenarios.

1. Difference between clustered/non-clustered index is at the LEAF node!!!!
   1. Clustered index has the data at leaf node
   2. Non-clustered index has a pointer at leaf node; this pointer is to the root node/entry point of the clustered index (this entry point is called the ‘clustering key’)
2. What is the structure of an index (structure of clustered and non-clustered are similar)
   1. ROOT
      1. entry point of any index
      2. millions of entry points in a typical index
   2. BRANCH
      1. As we add rows to index we broaden the number of trees (Root-Branch-Leaf)
      2. more than 2.37+- BILLION rows in a table…we add a second BRANCH
   3. LEAF
      1. lowest point in an index
      2. Clustered index 🡪 the actual data resides at this level
      3. Non-clustered index 🡪 has a pointer to the entry-point/ROOT node of appropriate clustered index

Indexing is critical to the overall performance of a database

…specifically READ performance

* Indexes hinder/harm performance of WRITES
* Therefore OLTP environments have few indexes (maybe only 1 or 2 per table)
* Remember, goal of indexes is to improve READ activity (SELECT statements)
  + Where do we place them?
  + Important characteristics for CLUSTERED index (only one per table):
    - **Narrow** (8 bytes or fewer 🡪 INTEGER, datetime)
      * All non-clustered indexes reference the clustered index
        + All indexed queries traversing a non-clustered index then traverse the clustered index
      * The smaller amount it takes the more rows fit in memory
      * Data found in memory is 1000 times faster to retrieve (mechanical drives)
      * SSD (solid state drives behave like memory)
    - **Sequential** 🡪 Primary key auto-increment is fantastic; also timestamp to hundredths/sec
    - **Never-changing** 🡪 PK should never change
    - \*\*automatically assigned by the system to the primary key by default\*\*
    - Preferred to be on the PK in a write-centric environment
    - Preferred to be on a collection of columns in a READ –centric environment
      * all dimPK/FKs in a fact table in dimensional model
      * all columns in a SELECT, FROM and WHERE clause in reporting server
  + NON-CLUSTERED (many per table):
    - Referenced in WHERE clause (usually ‘name’ columns)
    - Referenced in FROM clause (Foreign Keys mostly)
    - Referenced in GROUP BY
    - Preferred in READ-centric environments (reporting or data warehousing)
    - Not preferred in any WRITE-centric environment

1. *Explain the differences between an index seek and index scan and address when each is preferred.*
2. SEEK
   1. Query is engaging the index
   2. Preferred when there are greater than 10,000 rows in a table
   3. WHY?
      1. It takes time to reference query statistics and build an execution plan
      2. It often quicker to simply scan the entire table if less than 1000 rows
3. SCAN
   1. Query is NOT engaging the index
   2. Preferred when there less than 1000 rows in a table
   3. Possibly due to a crappy query 🡪 look at what is being retrieved…re-write?