What are indexes?

- Indexes are database objects that can make queries against your tables faster.
- They do this by sorting the data in the fields they apply to – either in the table itself, or in a separate data structure.
- This sorting allows the database engine to locate records within a table without having to search through the table row-by-row.
- There are two types of indexes – clustered and non-clustered.

Clustered Indexes

- The rows of a table with a clustered index are physically sorted based on the field or fields the index is applied to.
- A table with a primary key is given a clustered index (based on the primary key field) by default
- Most tables should have at least a clustered index, as queries against tables with a clustered index generally tend to be faster.
- A table may only have **one** clustered index.

Clustered Indexes: Strategies

- Apply a clustered index to whatever field – or fields - are most likely to be used in a join against the table.
- Ideally this field (or combination of fields) should also be the one that most *uniquely* defines a record in the table.
- Whatever field would be a good candidate for a primary key of a table, is usually also a good candidate for a clustered index.

Non-clustered Indexes

- A table may have many *non-clustered* indexes.
- Non-clustered indexes do not physically sort the data in a table like clustered indexes do.
- The sorted order of the field or fields non-clustered indexes apply to is stored in an external data structure, which works like a kind of phone book.

Non-Clustered Indexes: Strategies

- If you will be joining your table on fields besides the one "covered" by the clustered index, consider non-clustered indexes on those fields.
- You can add as many non-clustered indexes as we like, but should be judicious in doing so.
- Fields covered by a non-clustered index should still have a high level of uniqueness.

Indexes: General Approach

- It's how our table utilized in joins that should drive our use and design of indexes.
- You should generally add a clustered index first, and then layer in non-clustered indexes as needed to "cover" additional fields used in joins against our table.
- Indexes take up memory in the database, so only add them when they are really needed.
- They also make inserts to tables take longer, so you should generally add indexes *after* data has been inserted to the table.