SQL Query Tuning Hints

Overview of Query Tuning

- Tuning Queries
  - Avoid grabbing too much data
  - *Avoid SELECT **
    - Query performs faster if actual columns names used
  - *Filter GROUP BY queries in WHERE clause*
    - HAVING clause filters data after rows are retrieved

Overview of Query Tuning

- Tuning Queries
  - Limit table scanning and row processing
  - *Avoid NOT IN, NOT LIKE, !=*
    - Re-write for the positive
  - *Avoid using cursors!*
    - Create index on temp tables

Overview of Query Tuning

- Typical instances for query tuning
  - Correlated queries
  - Temporary tables
  - Table variables

  - JOIN conditions
    - choose one that uses a clustering index
    - avoid those that contain string comparisons

Overview of Query Tuning

- Typical instances for query tuning
  - Order of tables in FROM clause may affect JOIN processing
  - Some query optimizers perform *worse* on *nested queries* compared to their equivalent un-nested counterparts
  - Heavy reliance on *views* might do better to persist data

Overview of Query Tuning

- Additional Query Tuning Guidelines
  - Query with multiple selection conditions connected via OR may not be using any index
  - Look to split up suspect queries and express as a union
    - Force a condition on attribute to leverage an index

Overview of Query Tuning

- Additional Query Tuning Guidelines
  - Apply the following transformations
    - NOT condition into a positive expression.
    - Embedded SELECT blocks may be replaced by JOINs
    - Repeat range predicate restriction on JOINs
  - Re-write indexes to ‘cover’ columns in WHERE clause