# Query Optimization Practice

1. Big Joins vs Nested Subqueries

Using BookBiz, create two separate queries, one using several joins and the other using multiple nested subqueries that will find the name of the author(s) who wrote the book “Secrets of Silicon Valley”. Create a plan for each of these queries and compare them.

1. Subqueries vs direct table access

Find authors who have books. Do you think it would be faster to find distinct auid values in the TitleAuthor table or a different query that uses one of the approaches from class (for example joining author to titleauthor and distinct, or exists in a correlated subquery). Try several approaches and compare the execution plans.

1. Create a query using INTERSECT to find publishers in the same city as authors and a query that gets the same information but uses a join on city. See which plan offers better results. If the two have the same cost, which do you think might have the edge based on the plan listed?
   1. Create an index on author city, then create two new plans for the above queries. What has changed?
   2. Create a second index on publisher city, then create two new plans. What has changed?
   3. What can be inferred from this regarding indexes and set operators like intersect?
2. What is the implication on cost for sorting on multiple columns?
   1. Create a plan for a query that shows all titles, sorted by Title.
   2. Create a plan for a similar query, this time sort on type, then title
   3. Create yet another query that sorts on price, then type, then title
   4. Review the plans for each.
   5. Create a query that finds Title publishers (show Title and Publisher name) using a join. Create plans for each of the following, do the costs differ?
      1. Sort by Title
      2. Sort by Publisher Name
      3. Sort by Title then publisher
      4. Sort by Publisher, then Title