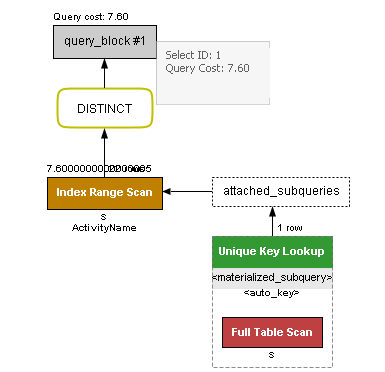
**Query 1: (Best Performance)**

Execution Plan:



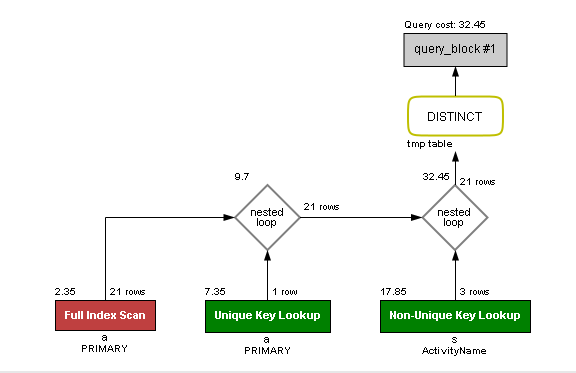
Explain Analyze:



This explain analyze seems to be optimized like the execution plan. It is denoted by “<in\_optimizer>” and contains the index skip scan. It has no nested loops, the sub query is only executed once, and it is traversing through less relations than other queries. This is the most efficient of the 3 queries.

**Query 2:**

Execution Plan:



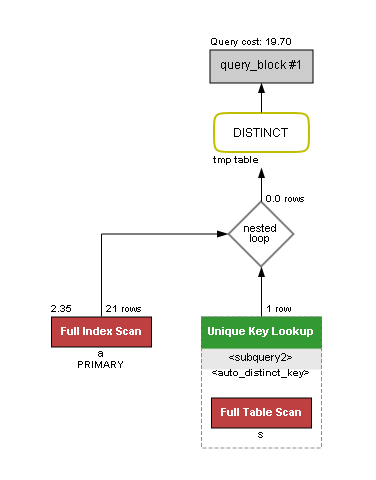
Explain Analyze:



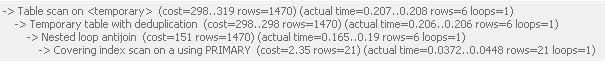
Comparing the query 2 execution plan and the explain analyze, the execution plan seems to be optimized since it has a full index scan while the explain analyze has a table scan and temporary table with duplication which seems more computationally expensive. This is by far the least efficient of the 3 queries.

**Query 3:**

Execution Plan



Explain Analyze:



This execution plan had a full index scan and a full table scan but only a single nested loop. Both have an index scan using a primary key, only with a temporary table. The explain analyze scan I on the temporary table. This query does not have the best performance, but it falls in the middle of the first and second.

Without Index

Execution Plan:

A screenshot of a computer

Description automatically generated

Explain Analyze:

A white rectangular sign with black text

Description automatically generated

With Index:

Execution Plan:

A screenshot of a computer

Description automatically generated

Explain Analyze:

A white rectangular box with black text

Description automatically generated

Between the two I see no performance benefits from the index in this instance since it may not be necessary to use. This is likely because our dataset is so small, and it will not be noticeable.

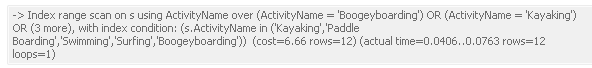
1. Explain Analyze:

* Activities while camping:



This table is efficient because it filters ParkName in the SummerPlan relation and only checks those matching rows. No nested loops. Query uses distinct to not return duplicates.

* Water Activities:



Has an index scan on ActivityName to only select matching rows. No nested loops. Query uses distinct to not return duplicates.

* Where to watch sports:



Indexes on ParkName to make sure it matches the items in the tuple. No nested loops. Query uses distinct to not return duplicates.

A diagram of a table scan

Description automatically generated

A diagram of a table scan

Description automatically generated

A diagram of a function

Description automatically generated

A diagram of a function

Description automatically generated