# [What are the Best Practices for performance?](#_6hhxhjlyloyl)

It depends on query performance to improve the performance, properties of the query should be improve those are listed below...

## 1.Optimizing Query Performance

In general, queries that do less work perform better. When evaluating query performance in BigQuery, the amount of work required depends on a number of factors:

* [Input data and data sources (I/O)](https://cloud.google.com/bigquery/docs/best-practices-performance-input): How many bytes does your query read?
* [Communication between nodes (shuffling)](https://cloud.google.com/bigquery/docs/best-practices-performance-communication): How many bytes does your query pass to the next stage? How many bytes does your query pass to each slot?
* [Computation](https://cloud.google.com/bigquery/docs/best-practices-performance-compute): How much CPU work does your query require?
* [Outputs (materialization)](https://cloud.google.com/bigquery/docs/best-practices-performance-output): How many bytes does your query write?
* [Query anti-patterns](https://cloud.google.com/bigquery/docs/best-practices-performance-patterns): Are your queries following SQL best practices?

# Link for optimizing query performance

(https://cloud.google.com/bigquery/docs/best-practices-performance-overview)

## 2.Managing Input Data and Data Sources

When evaluating your input data, consider the required I/O. How many bytes does your query read? Are you properly limiting the amount of input data? Is your data in native BigQuery storage or an external data source? The amount of data read by a query and the source of the data impact query performance and cost.

### Control projection - Avoid SELECT \*

**Best practice:** Control projection - Query only the columns that you need.

Projection refers to the number of columns that are read by your query. Projecting excess columns incurs additional (wasted) I/O and materialization (writing results).

### Prune partitioned queries

**Bestpractice:** When querying a [time-partitioned table](https://cloud.google.com/bigquery/docs/querying-partitioned-tables), use the \_PARTITIONTIME pseudo column to filter the partitions.

### Denormalize data whenever possible

**Best practice:** BigQuery performs best when your data is denormalized. Rather than preserving a relational schema such as a star or snowflake schema, denormalize your data and take advantage of nested and repeated fields. Nested and repeated fields can maintain relationships without the performance impact of preserving a relational (normalized) schema.

#### i) Using nested and repeated fields

BigQuery doesn't require a completely flat denormalization. You can use nested and repeated fields to maintain relationships.

* Nesting data (STRUCT)
* Repeated data (ARRAY)
* Nested and repeated data (ARRAY of STRUCTs)

### Use external data sources appropriately

**Best practice:** If query performance is a top priority, do not use an external data source.

Querying tables in BigQuery managed storage is typically much faster than querying external tables in Google Cloud Storage, Google Drive, or Google Cloud Bigtable.

Use an external data source for these use cases:

* Performing extract, transform, and load (ETL) operations when loading data
* Frequently changing data
* Periodic loads such as recurring ingestion of data from Cloud Bigtable

### Avoid excessive wildcard tables:

**Best practice:** When querying [wildcard tables](https://cloud.google.com/bigquery/docs/querying-wildcard-tables), use the most granular prefix possible.

# Link for Managing i/p data & data sources

(https://cloud.google.com/bigquery/docs/best-practices-performance-input)

## 3. Optimizing Communication Between Slots

When evaluating your communication throughput, consider the amount of shuffling that is required by your query. How many bytes are passed between stages? How many bytes are passed to each slot?

The following best practices provide guidance on controlling communication between slots.

### Reduce data before using a JOIN

**Best practice:** Reduce the amount of data that is processed before a JOIN clause.

### Do not treat WITH clauses as prepared statements

**Best practice:** Use WITH clauses primarily for readability.

### 

### Avoid tables shared by date

**Best practice:** Do not use tables shared by date (also called date-named tables) in place of time-partitioned tables.

### Avoid oversharding tables

**Best practice:** Avoid creating too many table shards. If you are sharding tables by date, use time-partitioned tables instead.

# Link for Optimizing Communication b/w Slots

(<https://cloud.google.com/bigquery/docs/best-practices-performance-communication>)