Whether the instrument is enabled. The value is `YES` or `NO`. A disabled instrument produces no events. This column can be modified, although setting `ENABLED` has no effect for instruments that have already been created.

- `TIMED`

  Whether the instrument is timed. The value is `YES`, `NO`, or `NULL`. This column can be modified, although setting `TIMED` has no effect for instruments that have already been created.

  A `TIMED` value of `NULL` indicates that the instrument does not support timing. For example, memory operations are not timed, so their `TIMED` column is `NULL`.

  Setting `TIMED` to `NULL` for an instrument that supports timing has no effect, as does setting `TIMED` to non-`NULL` for an instrument that does not support timing.

  If an enabled instrument is not timed, the instrument code is enabled, but the timer is not. Events produced by the instrument have `NULL` for the `TIMER_START`, `TIMER_END`, and `TIMER_WAIT` timer values. This in turn causes those values to be ignored when calculating the sum, minimum, maximum, and average time values in summary tables.

- `PROPERTIES`

  The instrument properties. This column uses the `SET` data type, so multiple flags from the following list can be set per instrument:

  - `global_statistics`: The instrument produces only global summaries. Summaries for finer levels are unavailable, such as per thread, account, user, or host. For example, most memory instruments produce only global summaries.
  - `mutable`: The instrument can “mutate” into a more specific one. This property applies only to statement instruments.
  - `progress`: The instrument is capable of reporting progress data. This property applies only to stage instruments.
  - `singleton`: The instrument has a single instance. For example, most global mutex locks in the server are singletons, so the corresponding instruments are as well.
  - `user`: The instrument is directly related to user workload (as opposed to system workload). One such instrument is `wait/io/socket/sql/client_connection`.

- `VOLATILITY`

  The instrument volatility. Volatility values range from low to high. The values correspond to the `PSI_VOLATILITY_`*`xxx`* constants defined in the `mysql/psi/psi_base.h` header file:

```
#define PSI_VOLATILITY_UNKNOWN 0
#define PSI_VOLATILITY_PERMANENT 1
#define PSI_VOLATILITY_PROVISIONING 2
#define PSI_VOLATILITY_DDL 3
#define PSI_VOLATILITY_CACHE 4
#define PSI_VOLATILITY_SESSION 5
#define PSI_VOLATILITY_TRANSACTION 6
#define PSI_VOLATILITY_QUERY 7
#define PSI_VOLATILITY_INTRA_QUERY 8
```

  The `VOLATILITY` column is purely informational, to provide users (and the Performance Schema code) some hint about the instrument runtime behavior.