PassQueen
a.k.a dumpsqueen.com

Splunk Core Certified Advanced Power User Exam

Splunk SPLK-1004

Version Demo

Total Demo Questions: 13

Total Premium Questions: 137

Buy Premium PDF

https://passqueen.com

support@passqueen.com

Topic Break Down

Topic	No. of Questions
Topic 1, Advanced Searching and Reporting	53
Topic 2, Search Optimization	22
Topic 3, Data Models	7
Topic 4, Dashboards and Visualizations	31
Topic 5, Knowledge Objects	24
Total	137

QUESTION NO: 1

A Simple XML form contains an input with the token name `host` and uses a submit button. Which statements accurately describe the form and submitted tokens? (Select two.)

A. `$form.host$` reflects the current input value before the form is submitted.

B. `$host$` remains at its previously submitted value until the form is submitted again.

C. Submitting the form clears `$form.host$`.

D. `$form.host$` cannot be referenced in a dashboard search.

E. Only drilldown interactions can set form tokens.

ANSWER: A B

Explanation:

An input updates its form token as the user changes the input value. Therefore, `$form.host$` represents the current input value before submission. When a form uses a submit button, `$host$` represents the submitted value and is updated when the user submits the form.

Submitting a form does not clear the form token, and form tokens can be referenced where dashboard token substitution is supported. This separation allows a dashboard author to control whether panel searches update immediately or only after submission.

QUESTION NO: 2

What arguments are required when using the spath command?

A. input, output, index

B. input, output path

C. No arguments are required.

D. field, host, source

ANSWER: C

Explanation:

No arguments are required. The `spath` command can be run by itself because Splunk supplies sensible defaults for its basic behavior. With no parameters, `spath` reads structured JSON or XML content from the event's `_raw` field and extracts discoverable elements into search-time fields. This makes a simple pipeline such as `| spath` sufficient when the raw event contains valid structured data and the goal is to expose its available keys for searching, filtering, or display.

The command's documented syntax presents `input`, `output`, and `path` as optional parameters. They are used only when a search needs more control: `input` selects a source field other than `_raw`; `path` targets a particular location in the JSON or XML structure; and `output` names the field that receives a specifically targeted extraction. Splunk also notes that automatic extraction without a specified path examines structured data in the beginning of the input value, subject to the command's extraction limits. Therefore, none of these parameters is mandatory for invoking the command. See the official spath command reference and Splunk's structured-data field extraction documentation.

QUESTION NO: 3

Which statements are correct about event types and tags in Splunk? (Select two.)

A. An event type modifies the raw event and stores the modified event in the index.

B. An event type is a named search that classifies matching events.

C. A tag can be applied only to indexed fields.

D. A tag replaces the underlying field value with the tag name.

E. Tags can be associated with field/value pairs and event types.

ANSWER: B E

Explanation:

An event type is a named search that classifies events matching defined search criteria. Tags can be associated with field/value pairs and can also be assigned to event types. Tags provide an additional descriptive label that can be searched and used to organize related events without modifying the raw event data.

QUESTION NO: 4

When should the fill_summary_index.py script be used?

A. To create a summary index.

B. To backfill gaps in a summary index.

C. To reset a summary index that includes overlapping data.

D. To populate a summary index from a saved report.

ANSWER: B

Explanation:

The correct use is **"To backfill gaps in a summary index."** Splunk summary indexing stores the results of scheduled reporting searches in a separate index, allowing subsequent searches and dashboards to query precomputed data rather than repeatedly processing the full underlying data set. If a scheduled summary-indexing search does not run—for example, because of a scheduler interruption, maintenance period, or configuration issue—the expected summary data can be missing for the affected time window. The `fill_summary_index.py` utility addresses this situation by rerunning the saved search over a specified historical interval, using the scheduled execution times that would normally have populated the summary index. It can also be used to populate historical summary data when a summary-indexing search is first deployed. Before running it, confirm the saved search is configured for summary indexing and select a time range that represents the missing historical coverage. Splunk documents this utility as the supported mechanism for filling collection gaps in a summary index; see Use summary indexing for increased search efficiency and the summary-index gap management documentation.

QUESTION NO: 5

Which element attribute is required for event annotation?

A. < search type="event_annotation" >

B. < search style="annotation" >

C. < search type=$annotation$ >

D. < search type="annotation" >

ANSWER: D

Explanation:

`<search type="annotation">` is the required search-element attribute and value for defining an event annotation in a Splunk Simple XML dashboard. An annotation search is a specialized search associated with a time-based visualization, such as a chart, and its returned results identify significant points or intervals to display as visual markers. Splunk distinguishes this specialized search from the primary search used to populate the visualization by setting the search element's `type` attribute to `"annotation"`. The annotation-search results must provide the expected time information so Splunk can place annotation markers at the appropriate locations on the visualization timeline. This declaration is placed on the dashboard's `search` element, allowing the visualization to treat its results as annotations rather than as the main series of chart data. Splunk's dashboard visualization documentation describes event annotations and the required annotation search configuration, including use of `type="annotation"`: Splunk Docs: Add event annotations to charts. For broader Simple XML dashboard syntax and search configuration context, see Splunk Docs: Build dashboards with Simple XML.

QUESTION NO: 6

When running a search, which Splunk component retrieves the individual results?

A. Indexer

B. Search head

C. Universal forwarder

D. Master node

ANSWER: A

Explanation:

Indexer is correct because indexers perform the data retrieval work for searches. An indexer stores indexed event data and processes the relevant portions of a search against the data it holds. In a distributed Splunk deployment, the search head distributes the search to the appropriate indexers, and each indexer searches its local data and returns matching events or intermediate results. Thus, the component that actually retrieves the individual search results from indexed data is the indexer.

Splunk describes indexers as components that index incoming data and also search the data in response to search requests. Search processing is commonly divided between the search head and indexers: the search head coordinates the search and combines returned results, while indexers carry out the retrieval and much of the search processing close to where the data resides. This architecture allows Splunk to search large volumes of distributed data efficiently. See Splunk's documentation on deployment components and how search works.

QUESTION NO: 7

Which statements are correct about the `depends` and `rejects` attributes in a Simple XML dashboard? (Select two.)

A. `depends` prevents a panel from running when its token value is empty, but the panel remains visible.

B. `depends` displays an element only when the specified token is set.

C. `rejects` clears the specified token whenever a panel is displayed.

D. `rejects` prevents an element from being displayed when the specified token is set.

E. Both attributes can be used only on dashboard root elements.

ANSWER: B D

Explanation:

The `depends` attribute controls conditional display by requiring the specified token to be set before an element is displayed. The `rejects` attribute has the opposite condition: it prevents display when the specified token is set. These attributes are commonly used with tokens set by inputs or drilldowns to control panel visibility.

QUESTION NO: 8

A field named `recipients` is multivalued. Which statements are correct about multivalue evaluation functions? (Select two.)

A. `mvfilter()` creates a separate result event for every retained value.

B. `mvfind()` returns every matching value as a new multivalue field.

C. `mvfilter()` returns values from a multivalue field that satisfy a condition.

D. `mvindex()` permanently removes all values except the requested index.

E. `mvfind()` returns the index of the first value that matches a regular expression.

ANSWER: C E

Explanation:

The `mvfilter()` function evaluates a condition against the values in a multivalue field and returns only values that satisfy that condition. The `mvfind()` function returns the index of the first multivalue element that matches a regular expression. Neither function expands an event into multiple result events; that operation is performed by `mvexpand`.

QUESTION NO: 9

What type of drilldown passes a value from a user click into another dashboard or external page?

A. Visualization

B. Event

C. Dynamic

D. Contextual

ANSWER: D

Explanation:

Contextual is correct because a contextual drilldown uses the context of what a user clicked—such as a field value, table cell, chart series, or data point—to create a destination link. Splunk exposes clicked values as drilldown tokens, which can then be inserted into the target dashboard URL, a search, or an external URL. For example, clicking a host value in a visualization can open a second dashboard with that host supplied as an input token, allowing the destination dashboard to run searches scoped to the selected host. The same token-based approach can construct a URL to an external system, passing the selected value as a query parameter. This makes contextual drilldowns a key dashboard interaction feature: the destination is not merely a fixed link, but receives information derived from the user's specific selection. Splunk's dashboard documentation describes drilldown interactions and the use of tokens to capture and pass values from user actions. See the Splunk Dashboard Studio drilldown documentation and the Splunk overview of drilldown actions.

QUESTION NO: 10

How can the inspect button be disabled on a dashboard panel?

A. Set inspect.link.disabled to 1

B. Set link.inspect.visible to 0

C. Set link.inspectSearch.visible to 0

D. Set link.search.disabled to 1

E. Set link.inspect to 0

ANSWER: E

Explanation:

Set `link.inspect` to `0`. In a Simple XML dashboard, this is configured as a panel or visualization option, for example: `<option name="link.inspect">0</option>`. The `link.inspect` option controls whether Splunk displays the Inspect control for that panel's search job. Assigning it the value `0` suppresses the control in the rendered dashboard panel; assigning `1` enables its display. This is a presentation setting for the panel, so it can be applied only where needed rather than changing behavior globally for every dashboard. The setting is particularly useful when building streamlined operational dashboards where the author does not want to expose the panel's inspection interface to dashboard viewers. Splunk's Simple XML panel reference documents `link.inspect` as the option used to control this link, and the dashboard development documentation describes placing visualization and panel settings in `<option>` elements. See the Splunk Simple XML panel reference and Splunk dashboard development documentation.

QUESTION NO: 11

Which statements are accurate about the `dedup` command? (Select two.)

A. It retains the first result encountered for each unique field value.

B. It always retains the event with the latest `_time` value for each field value.

C. It combines duplicate events into one event and merges their field values.

D. With `consecutive=true`, it removes only adjacent duplicate values.

E. It requires a transforming command immediately before it.

ANSWER: A D

Explanation:

The `dedup` command retains the first result encountered for each unique field value, so the ordering of incoming results determines which duplicate is retained. The `consecutive=true` option changes the behavior so that only adjacent duplicate values are removed. It does not remove matching values that are separated by other results.

QUESTION NO: 12

What is the value of**base lispy**in the Search Job Inspector for the search**index=web clientip=76.169.7.252**?

A. [index::web AND 169 252 7 76]

B. [AND 169 252 7 76 index::web]

C. [169 AND 252 AND 7 AND 76 index::web]

D. [index::web 169 AND 252 AND 7 AND 76]

ANSWER: B

Explanation:

The correct base lispy value is **[AND 169 252 7 76 index::web]**. Search Job Inspector exposes the internal, optimized form of the portion of a search that Splunk can use to locate candidate events. This representation uses a Lisp-like prefix structure: the Boolean operator appears immediately after the opening bracket, followed by the searchable terms. Therefore, the conjunction is represented as **AND** in the leading position.

For this search, **index=web** becomes the indexed restriction **index::web**. The dotted IP address is segmented into searchable terms, producing **76**, **169**, **7**, and **252**; the base lispy expression lists these terms with the index constraint under the same implicit conjunction. This form is used by Splunk's search pipeline and indexing strategy to determine which buckets and events are candidates before later search processing evaluates field extraction and additional conditions. Search Job Inspector is specifically intended to help inspect search execution details and optimization behavior. See the Search Job Inspector reference and Splunk's guidance on writing better searches.

QUESTION NO: 13

Two multivalue fields, `product` and `quantity`, contain corresponding values in the same order. Which statements are correct about the `mvzip` function? (Select two.)

A. It pairs values from the two multivalue fields according to their positions.

B. It can use a specified delimiter between each paired value.

C. It creates one separate event for every paired value.

D. It sorts both multivalue fields before combining them.

E. It converts both multivalue fields into numeric fields.

ANSWER: A B

Explanation:

The `mvzip` function combines values from two multivalue fields by position. For example, `mvzip(product,quantity,"|")` creates a multivalue result in which the first product is paired with the first quantity, the second product with the second quantity, and so on. A delimiter can be specified to separate the paired values.

`mvzip` creates a multivalue field; it does not create separate result rows. Creating separate events for multivalue values requires `mvexpand`. It also does not sort either input field before pairing values.