Migrating Your DB Inputs

From DB Connect v1 to v3

Hani Atalla

October 2018

Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC.

The forward-looking statements made in this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward-looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

splunk> .conf18

Agenda

- Disclaimer
- Why Ditch DB Connect v1?
- The Long Path
- The Short Path
- Steps to Migrate
- Takeaways

Disclaimer

- I am not here on behalf of my employer as the views of my employer do not conform to my views, or to any accepted standard of logic that the Greeks thought up some 2000 years ago.

splunk> .conf18

HANI ATALLA

Splunk Engineer

splunk> .conf18

So Why Ditch DB Connect v1?

splunk> .conf2017

DB Connect Is Back

…and it is better than ever

Tyler Muth | Denis Vergnes

September 2017 | Washington, DC

Why v3?

- New UI to manage inputs, outputs, lookups
  - Wizard based
  - Type ahead dropdowns
  - More .conf options: query timeout
- Input templates
  - Add-on for Oracle, MS SQL Servers, McAfee, Nagios
- DB Connect health checks
  - Pre-built panels to monitor DB Connect
- Improved Performance
- Flexibility
  - | dbxquery procedure="{call <procedure-name>}"

Why v3?

Ease of use

WYSIWYG SQL and SPL editors, new UI, input bulk operations, input template

Performance

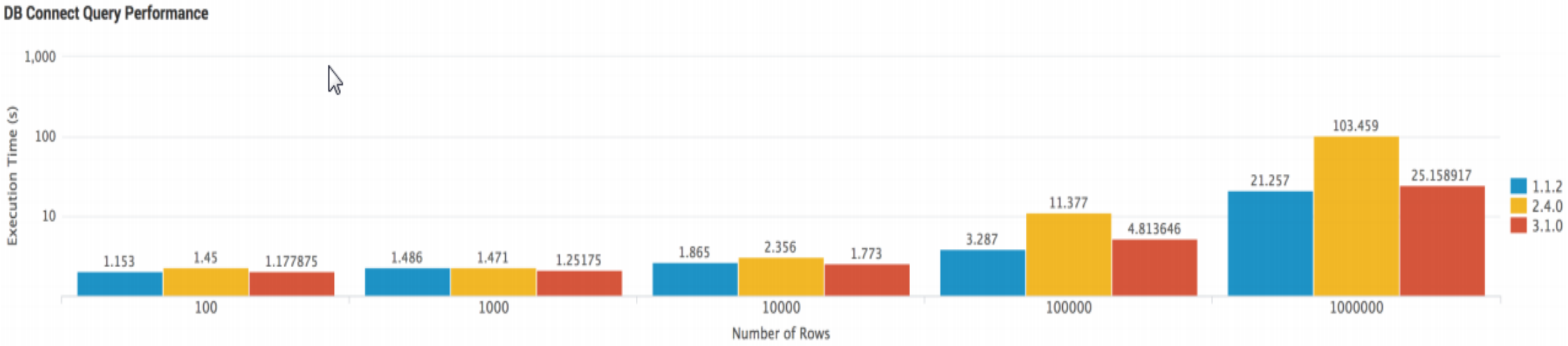
Performance boost up to 10x, vertical scalability

Flexibility

Stored procedures, 14 supported databases, Linux and Windows platform

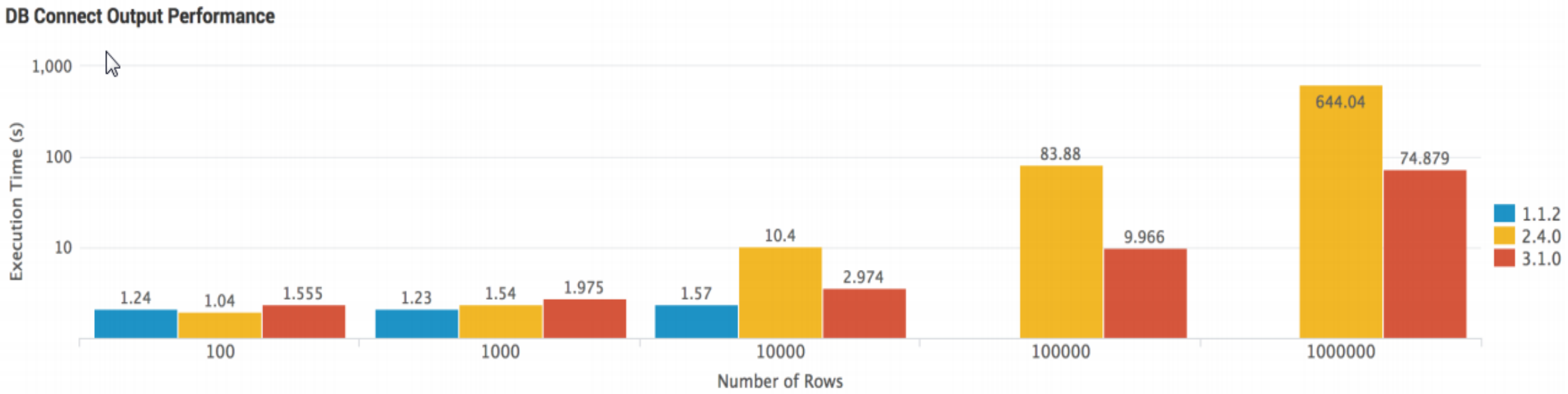
Performance - Queries

Improvement increases with dataset size, up to 4x faster from 2.x



Performance - Outputs

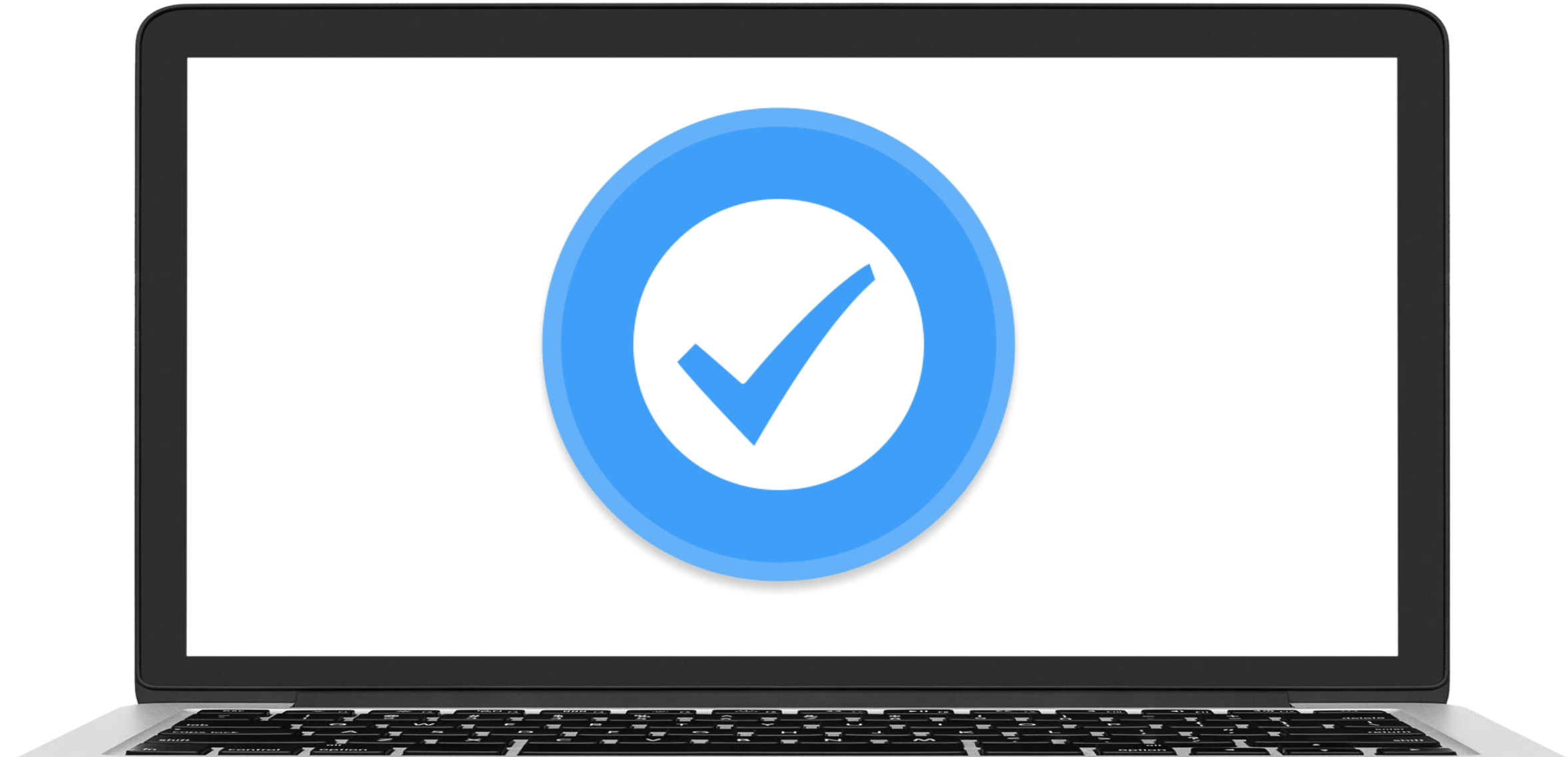
Large datasets are output 2x to 9x faster than v2.4x



But Why Really Ditch DB Connect v1?

Our Options?

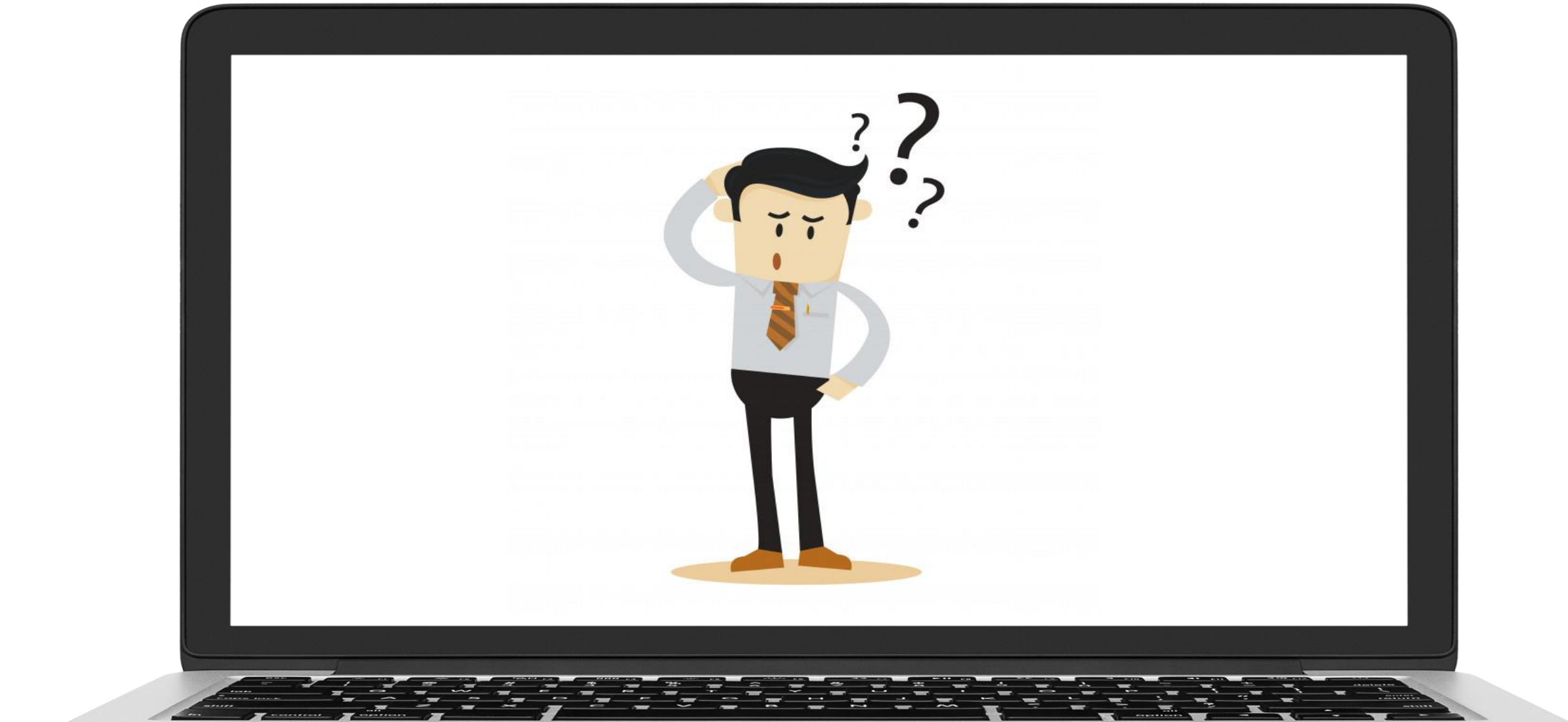
Already on DB Connect v2.x



You Are Set

- Install latest DB Connect v3 on a Heavy Forwarder
  - [v 3.1.3]
- Run migration script
  - [app_migration.py]

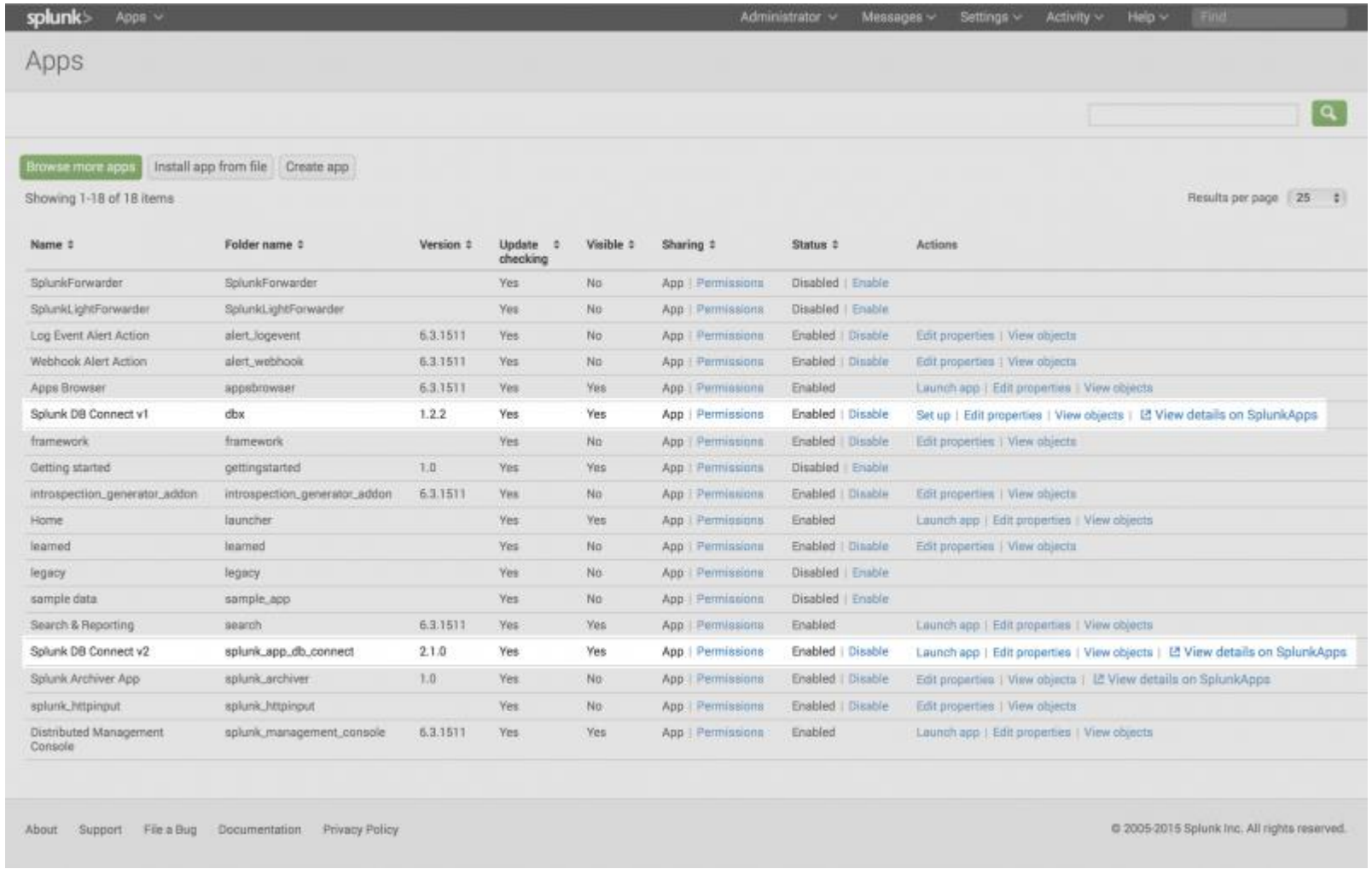
On DB Connect v1.x



Option 1

The Long Path

Dual Install



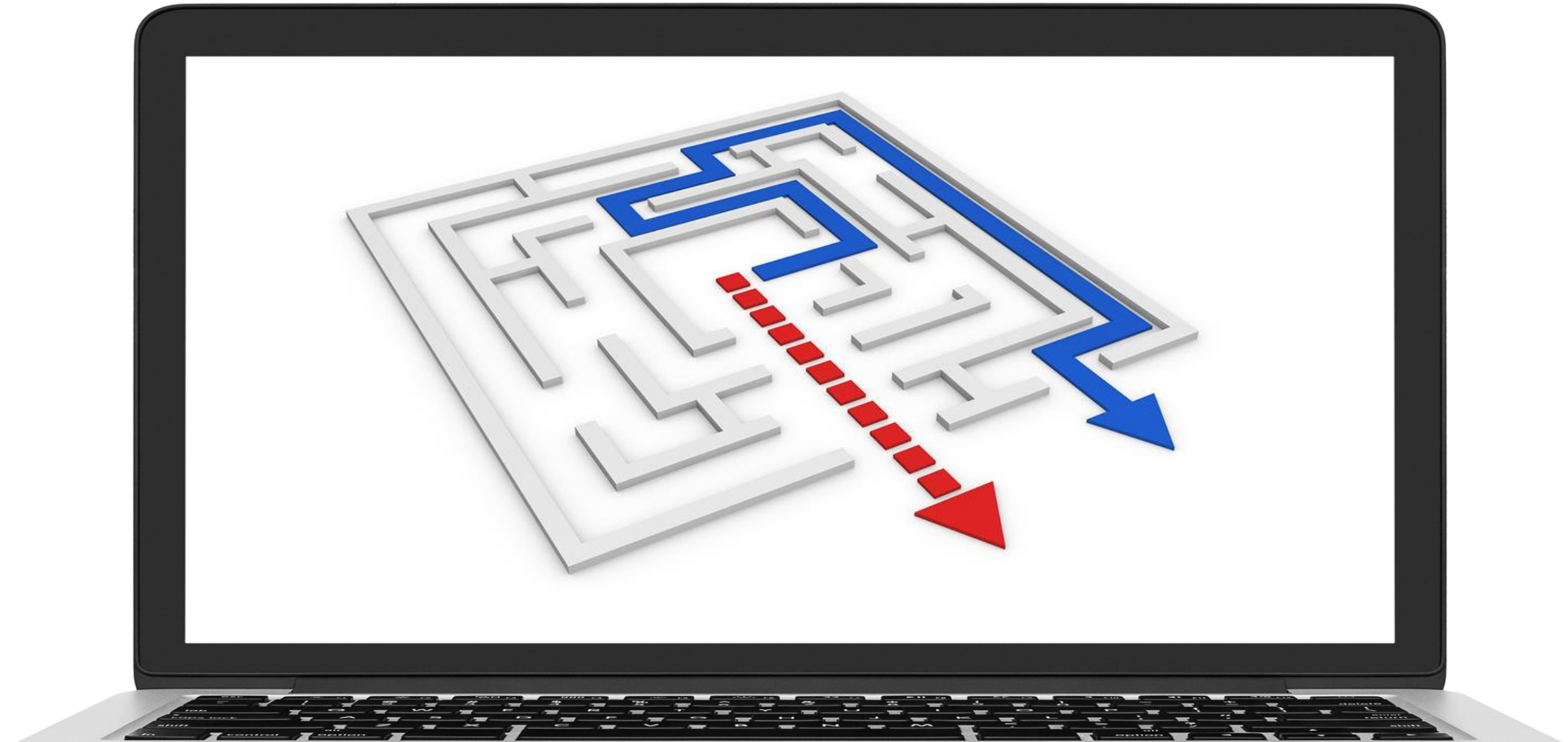
The Long Path to v3 from v1 [**Part I**]

- Ensure you are running latest DB Connect v1
  - [v1.2.2]
- Update your Java install
  - [Java 8]
- Install DB Connect v2
  - [2.4.0 or later]
- Copy database drivers
  - [.jar files **from** /etc/apps/dbx/bin/lib **to** /etc/apps/splunk_app_db_connect/bin/lib]
- Run the connections/identities migration script to v2
  - [splunk cmd python dbx_migrate_connections.py]
- Run the DB inputs migration script to v2
  - [splunk cmd python dbx_migrate_inputs.py]

The Long Path to v3 [**Part II**]

- Run the Lookups migration script to v2
  - [splunk cmd python dbx_migrate_lookups.py]
- Install latest DB Connect v3 release
  - [v 3.1.3]
- Run migration script (drivers/identifies/connections/inputs/lookups) [v2 to v3]
  - [splunk cmd python SPLUNK_HOME/etc/apps/splunk_app_db_connect/bin/app_migration.py]
- Move or delete DB Connect v1
  - [$SPLUNK_HOME/etc/apps/dbx]

Perhaps There is a Shorter Path



Option 2

A Shorter Path

From: DB Input [DB Connect v1]

A Database input will fetch data from a SQL database.

Input Type

Tail (Follow based on increasing value)

Database

[redacted]

Specify SQL query

SQL Query *

```
SELECT field1, field2, field3 from table where
{{AND $rising_column$ > ?}}
ORDER BY trim(upper(field1))
```

You can specify the SQL query that is executed against the database yourself. For information on how to specify such a query, see Splunk DB Connect documentation. Example:
`SELECT * FROM my_table {{WHERE $rising_column$ > ?}}`

Tail input settings

Rising Column *

myTimestampColumn

Choose a column with an increasing value. Such as a creation or modification timestamp or a sequential identifier. You can also create a trigger to synthesize such a value.

Sourcetype

mySplunkSourceType

Splunk Index

mySplunkIndex

Host Field value

myDBBHost

Output

Output Format *

Multi-line Key-Value format

Specify how the event text content is generated.

Output timestamp

Timestamp column

myTimestampColumn

Select a column from the given table/query which should be used for the timestamp value.

Timestamp format

yyyy-MMM-dd HH:mm:ss.SSS

The timestamp format expressed as a Java SimpleDateFormat pattern. The default format is configured in the Splunk DB Connect app setup.

Interval

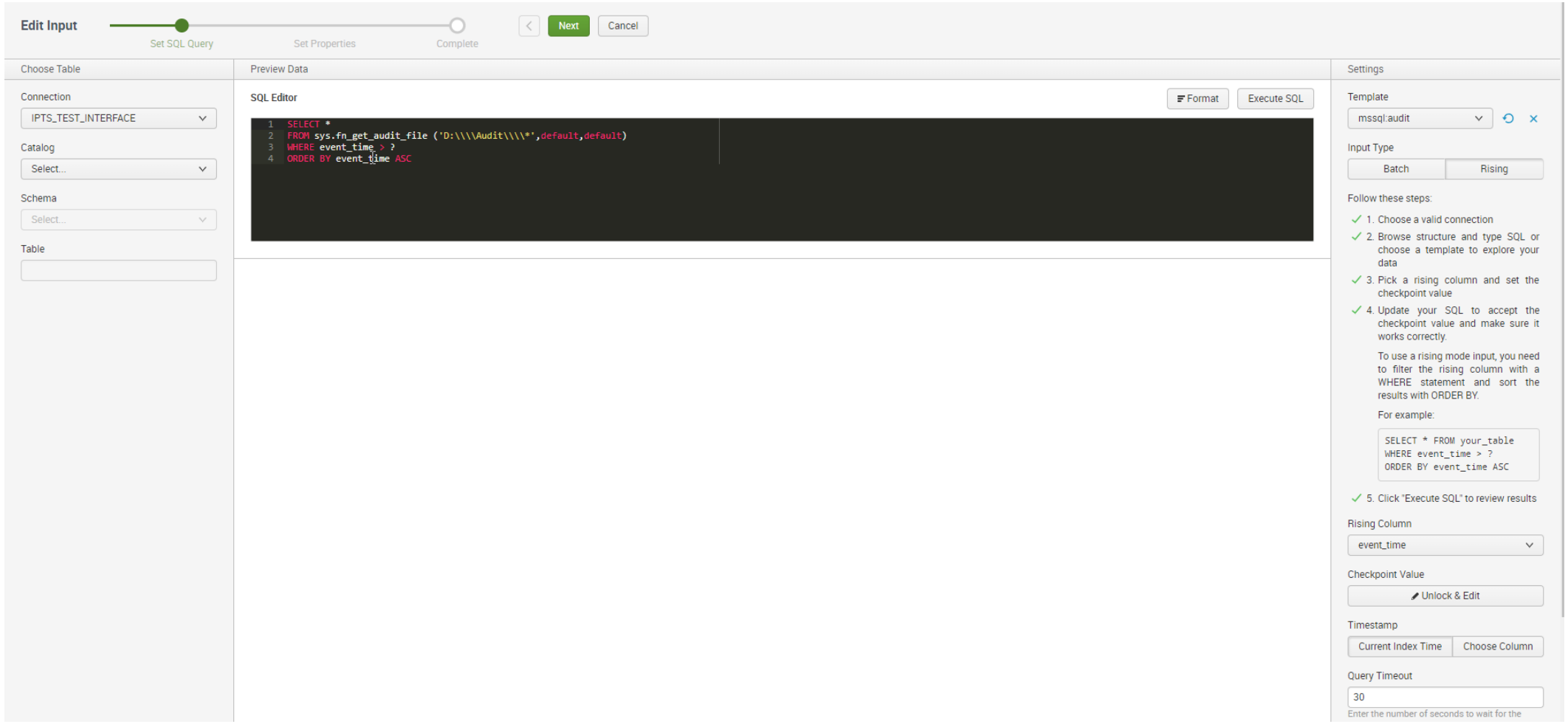
300

The interval can either be a valid cron expression or a relative time expression to wait between each run. Leave empty to let dbmon choose an appropriate interval automatically depending on the amount of data fetched.

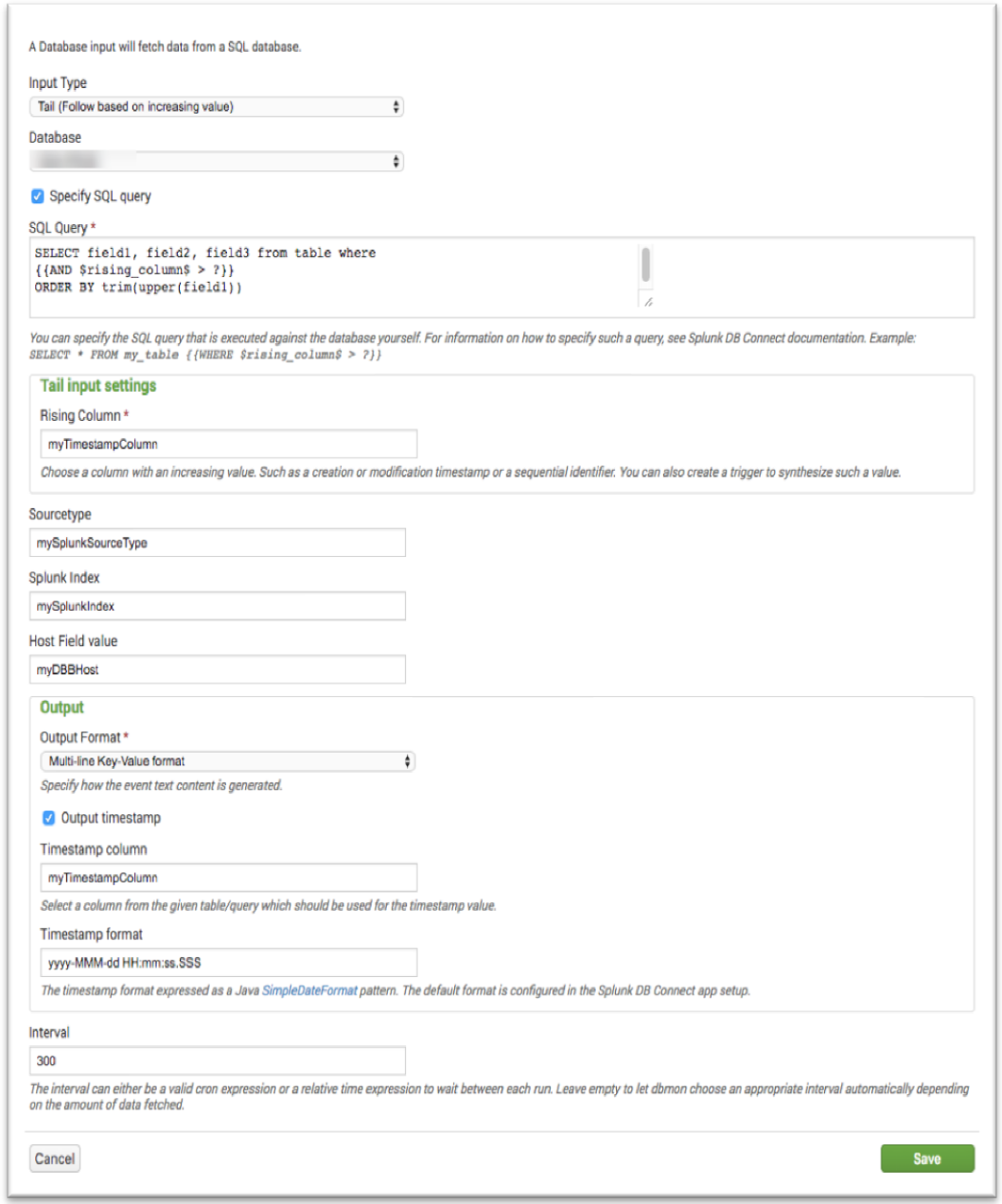
Cancel

Save

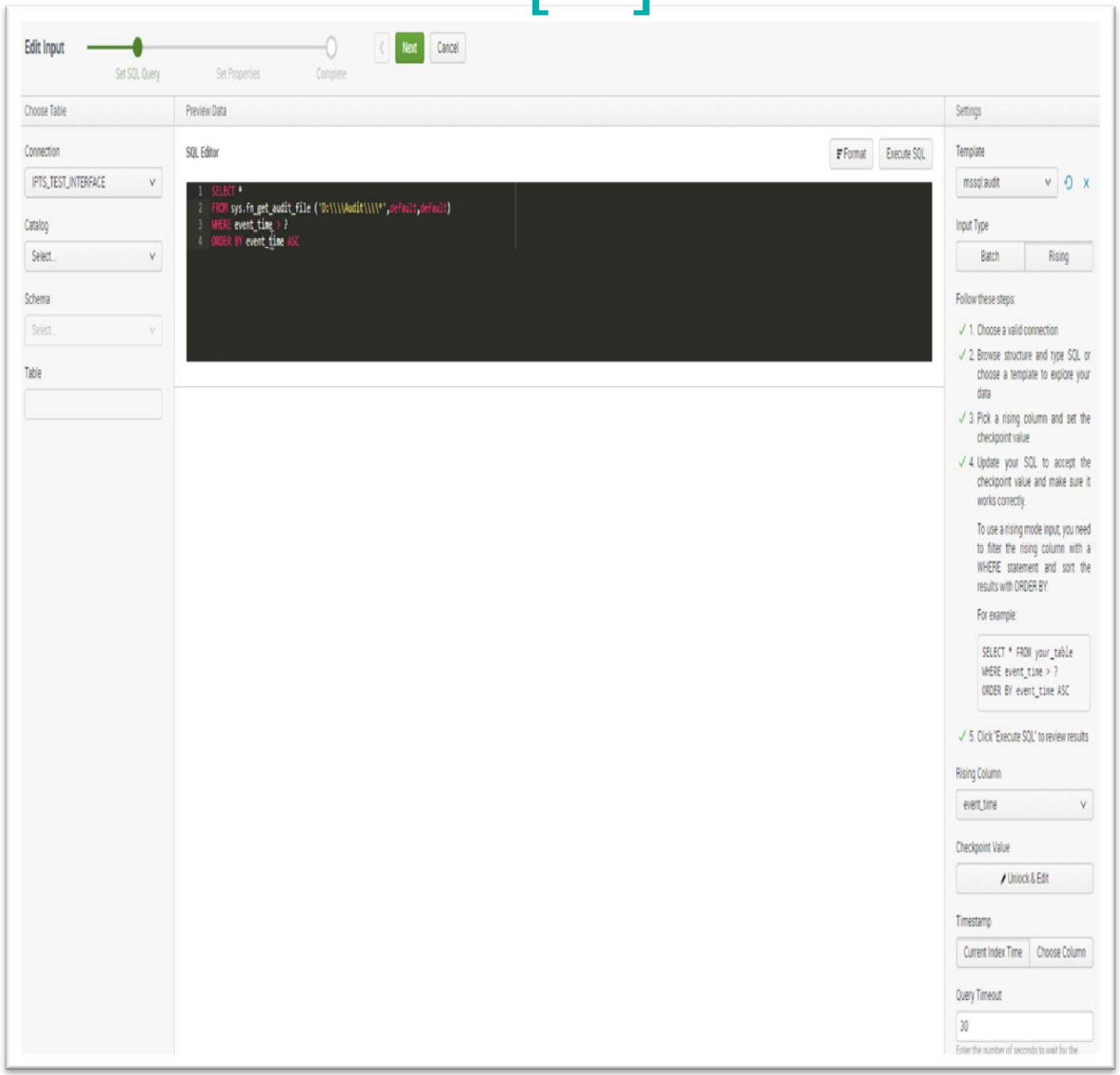
To: DB Input [DB Connect v3]



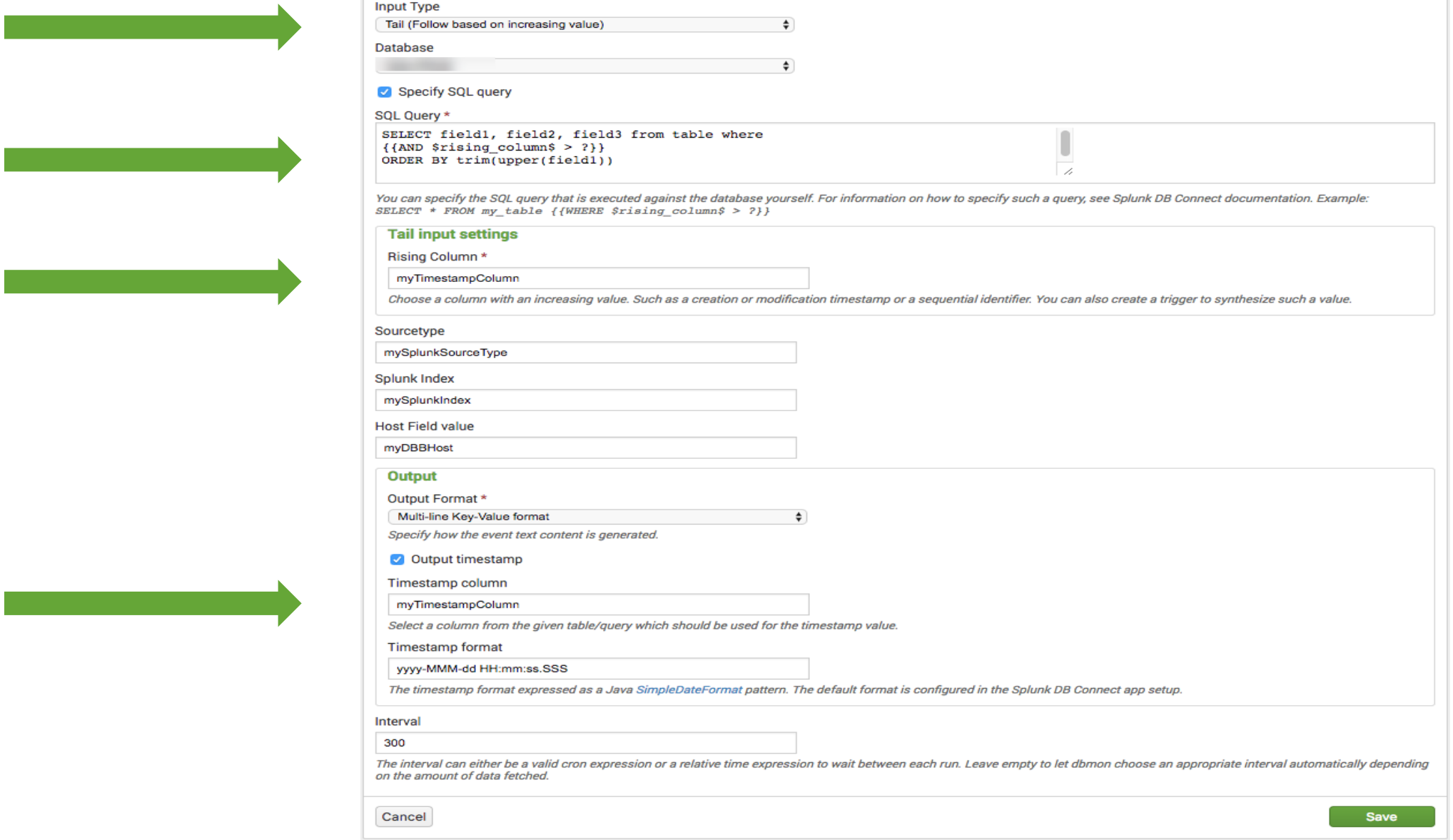
From [v1]



To [v3]



DB Input [Components]



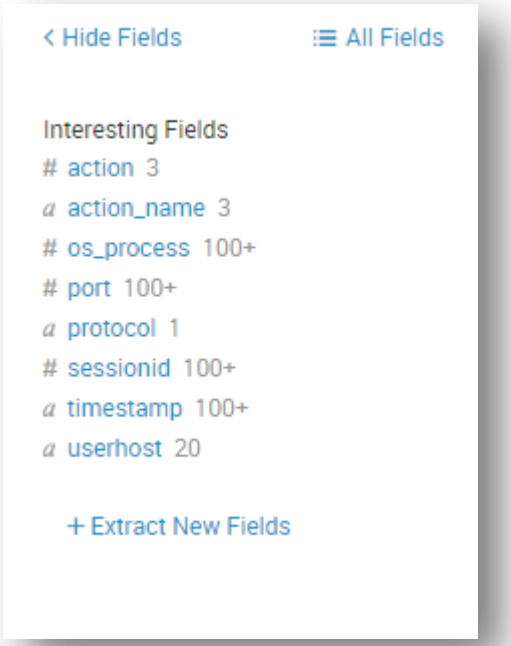
SQL: Upper / Lower Case Behavior v1

Converts to lowercase regardless of SQL

```
SELECT
  action,
  action_name,
  protocol,
  sessionid,
  port,
  userhost,
  os_process,
  terminal,
 timestamp
FROM Audit_Trail;
```

```
SELECT
  Action,
  Action_Name,
  Protocol,
  SessionID,
  Port,
  UserHost,
  OS_Process,
  Terminal,
  TimeStamp
FROM Audit_Trail;
```

```
SELECT
  ACTION,
  ACTION_NAME,
  PROTOCOL,
  SESSIONID,
  PORT,
  USERHOST,
  OS_PROCESS,
  TERMINAL,
  TIMESTAMP
FROM Audit_Trail;
```



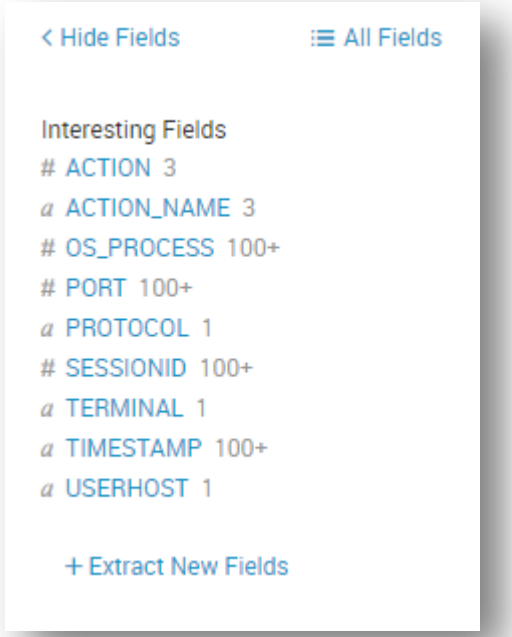
SQL: Upper / Lower Case Behavior v3

Converts to uppercase regardless of SQL

```
SELECT
  action,
  action_name,
  protocol,
  sessionid,
  port,
  userhost,
  os_process,
  terminal,
 timestamp
FROM Audit_Trail;
```

```
SELECT
  Action,
  Action_Name,
  Protocol,
  SessionID,
  Port,
  UserHost,
  OS_Process,
  Terminal,
  TimeStamp
FROM Audit_Trail;
```

```
SELECT
  ACTION,
  ACTION_NAME,
  PROTOCOL,
  SESSIONID,
  PORT,
  USERHOST,
  OS_PROCESS,
  TERMINAL,
  TIMESTAMP
FROM Audit_Trail;
```

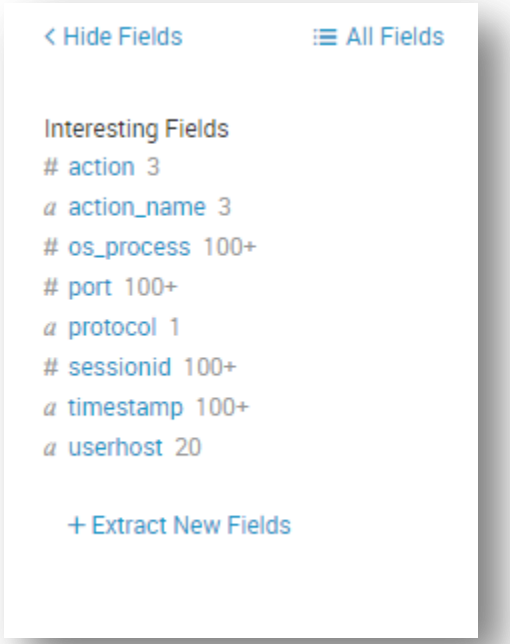


splunk> .conf18

DB Input – DB Connect v3

Use "AS" clause to force required case

```
SELECT
        action AS "action" ,
        action_name AS "action_name" ,
        protocol AS "protocol",
        sessionID AS "sessionid",
        port AS "port",
        userhost AS "userhost",
        os_process AS "os_prcess",
        terminal AS "terminal",
        timestamp AS "timestamp"
FROM Audit_Trail;
```



Make It Simple

```
SELECT
  e.employee_id AS "Employee", e.first_name || ' ' || e.last_name AS "Name“, e.email AS "Email“, e.phone_number AS "Phone“, TO_CHAR(e.hire_date, 'MM/DD/YYYY'),
TO_CHAR(e.salary, 'L99G999D99', 'NLS_NUMERIC_CHARACTERS = ".," NLS_CURRENCY = "$"'), e.commission_pct AS "Comission %", || j.job_title || ' in ' || d.department_name || '
department (manager: ‘     || dm.first_name || ' ' || dm.last_name || ') and immediate supervisor: ' || m.first_name || ' ' || m.last_name , TO_CHAR(j.min_salary, 'L99G999D99',
'NLS_NUMERIC_CHARACTERS = ".," NLS_CURRENCY = "$"') || ' - ' ||    TO_CHAR(j.max_salary, 'L99G999D99', 'NLS_NUMERIC_CHARACTERS = ".," NLS_CURRENCY = "$"')
  , l.street_address || ', ' || l.postal_code || ', ' || l.city || ', ' || l.state_province || ', '
    || c.country_name || ' (' || r.region_name || ')'
  , jh.job_id AS "History Job ID"
  , 'worked from ' || TO_CHAR(jh.start_date, 'MM/DD/YYYY') || ' to ' || TO_CHAR(jh.end_date, 'MM/DD/YYYY') ||
    ' as ' || jj.job_title || ' in ' || dd.department_name || ' department‘
FROM employees e
JOIN jobs j
    ON e.job_id = j.job_id
LEFT JOIN employees m
    ON e.manager_id = m.employee_id
lEFT JOIN departments d
    ON d.department_id = e.department_id
LEFT JOIN employees dm
    ON d.manager_id = dm.employee_id
LEFT JOIN locations l
    ON d.location_id = l.location_id
  LEFT JOIN countries c
    ON l.country_id = c.country_id
  LEFT JOIN regions r
    ON c.region_id = r.region_id
LEFT JOIN job_history jh
    ON e.employee_id = jh.employee_id
LEFT JOIN jobs jj
    ON jj.job_id = jh.job_id
LEFT JOIN departments dd
    ON dd.department_id = jh.department_id
ORDER BY e.employee_id;
```

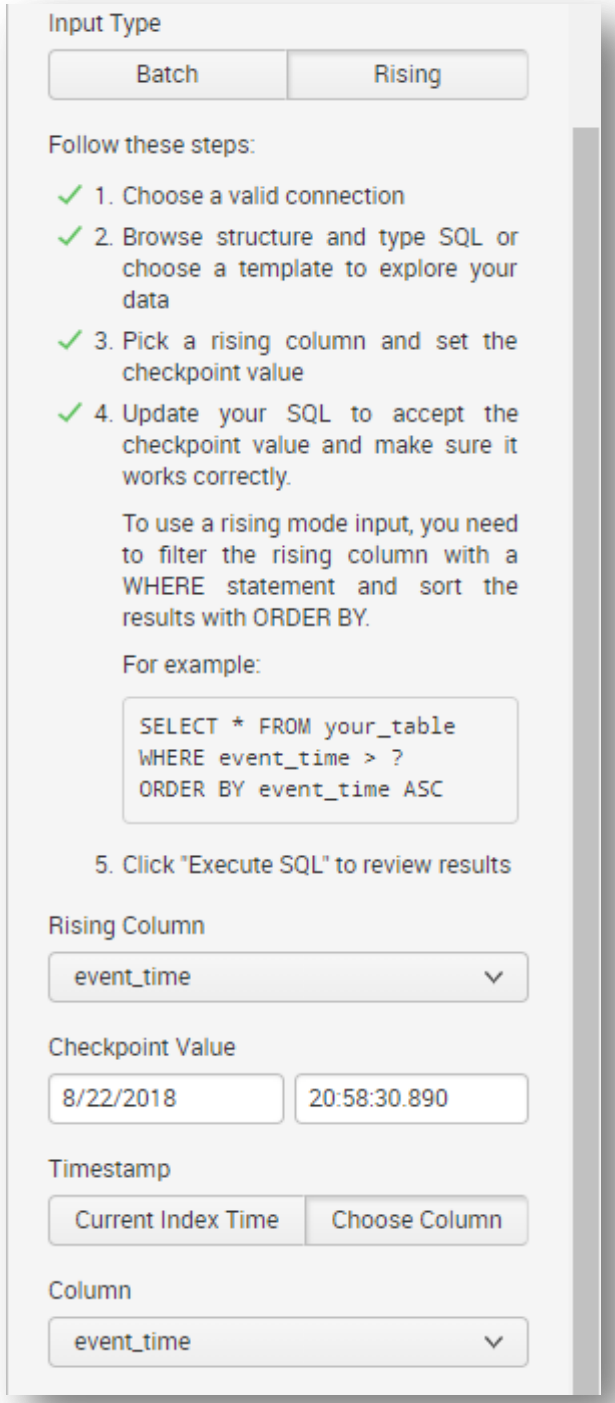
Make It Simple

Obfuscate SQL complexity in database views

```
CREATE VIEW View_Employees_Info AS SELECT
  e.employee_id AS "Employee #"
  , e.first_name || ' ' || e.last_name AS "Name"
  , e.email AS "Email"
  , e.phone_number AS "Phone"
  , TO_CHAR(e.hire_date, 'MM/DD/YYYY') AS "Hire Date"
  , TO_CHAR(e.salary, 'L99G999D99', 'NLS_NUMERIC_CHARACTERS
= ".," NLS_CURRENCY = "$"") AS "Salary"
  , e.commission_pct AS "Comission %“,
 , e.timestamp AS “TimeStamp"
FROM employees e
  JOIN jobs j
    ON e.job_id = j.job_id
  LEFT JOIN employees m
    ON e.manager_id = m.employee_id
  LEFT JOIN departments d
    ON d.department_id = e.department_id
  LEFT JOIN employees dm
    ON d.manager_id = dm.employee_id
  LEFT JOIN locations l
    ON d.location_id = l.location_id
  LEFT JOIN countries c
    ON l.country_id = c.country_id
  LEFT JOIN regions r
    ON c.region_id = r.region_id
  LEFT JOIN job_history jh
    ON e.employee_id = jh.employee_id
  LEFT JOIN jobs jj
    ON jj.job_id = jh.job_id
  LEFT JOIN departments dd
    ON dd.department_id = jh.department_id
ORDER BY e.employee_id;
```

```
SELECT
        Employee AS “employee” ,
        First_Name AS “first_name”,
        Phone_Number AS “phone_number”,
        Salary AS “salary”,
        Commission_PCT AS “commission_pct”,
        TimeStamp AS “timestamp”
FROM View_Employees_Info
Order by TimeStamp asc;
```

Rising Column Checkpoint Value



splunk> .conf18

DB Input [DB Connect v1]

Grep is Your Best Fried

$SPLUNK_HOME/var/lib/splunk/persistentstorage/dbx/<hash>

```
[redacted]:/opt/splunk/var/lib/splunk/persistentstorage/dbx $ ls
08ab6e930c7767e7f0ce29bb332f0eee  42434a40f2c8c41c6359a407829f7c7f  69570151749c9e7374356db043daa48a  7bb64863f78827a573a4a18c35bb330b  acfbedfa60ba4eb642b92fa8141507b2  d64779ac26265bf0ecdba797b9ac3a87  ee5f54f24f5b6adb5d2940b62b209367
10d8ee7d5604f759b18369f28afe6f9c  43ee8ac778f4a1b30fbf48714b846f3f  6c83478b2b776013f7ea53a3e776187d  7d4d12a0176b43d1f1eebd792d9a1fc3  aeb1921fd4c6adbde8d00f895b79a6a8  d6f719336a5dc70ea29bffb048bf4d73  ef3402a2d90a20fb81e52d347404b382
13004ba2dc1850849105132ba40cb683  45ce21388780934d0df8d4c9da33fe9f  6f2bca58cee97738af1d4f13eb7f264d  7ec20b489b114c5e2ecb9731f8b39528  aeefe9a2380efee63abca732b8d3d7d2  d9710c7862bc72eaeef34ed330eb2d54  f44681b3e0f3cf870a3c620da13bea05
13df2ca5c729dd3c5b9c1205d944bfe6  4a753be34e0e9edd59c8d94de461d3e2  6f50e20273298689d48ce9b53e8e80ac  83cb5c54409c4d73b50c4ea2b054137a  aff5372fda9b56f47ba9281a14d90d96  dc962d2b445023b3f45674a07a2415da  f98b505f60247260e42f6fcf2000040c
1e12a2df166c486fc05955c346be6cbc  4f4fbfa13166e4ccf707283a9773cf2e  72f3be6fb7e618e0cea2abf6ceb4b6d8  85f2b73a0a7bacc3ce200c8396285bd8  b5b2e5b3e100880f9dd12524d98f9465  dcbe847ce33ba0c6e2dc424e375ec899  fb5be9552b50c9c2d68440bc734f6391
1e9f538e8047326b715163d113ae2e44  506e859fa056f8f7ec5ae0001532834e  751accbd74296a96b4864c3818d6318f  86a7e4837b9e0dec36e2e0b3b8317823  bd00613e227dbac68d7815fb8511448a  dd01117fa92c5d893bc8184ad78d47a0  fcfd7c2ae597ddba921b67b9b7881869
1f5be5408b27d0f70f350f288693cfff  5963b5008b5045046234dbb514bd987e  76063dd69d24751724fad74d7d685064  8752c73929b0c83bc0ca4416ec99844f  bdacc562021a640020a701450a8eef4e  e2e919c7933152b06209b64b2b0cdc67  fe75d6db5d6c4b59ec79a6b391908363
22374d872f8a0861fd24f5380da0f3a9  59677d24b83328f3db4489e16ed7bda1  771d8096d00d67d9a155a66c286f4a81  9e2daa25c45133d4ab2188e6b6c83d4e  cdbaaa393c888b8828f66f3a24f33a56  eb86b0c5c99ab70ceb97bf304a6884bc  global
334b78d4596a7f054d41fdd6a90d7fa0  5b688b7e57674193c0e1c38ea0966f5c  793785b5f0cd64f3f2cf4af9d64772a3  a132a60a0e16ddbf2b01d1618aa73f9c  d022ff3232435c5d4d4c9f1386c9c8b5  ecbe5c1cfbb932e18b1c8d234d2464e7
3c4a47d27733e606ddc7bdbbf0d819a1  642d10b7035b6cee8e808b69dfb2b31f  7994ff22d6c6370a11d8f612ede12154  ab42494b296d2fa268c6c71cf64b14c6  d5f05ae7dd75d57d6a5e7750f6a84403  edf2b2b854b6158e5668cbab36c9fb37
[redacted]:/opt/splunk/var/lib/splunk/persistentstorage/dbx $
```

DB Input [DB Connect v1]

“grep” is your best friend

```
hostname:/opt/splunk/var/lib/splunk/persistentstorage/dbx/grep –irl <db-input-name>
```

```
[redacted]:/opt/splunk/var/lib/splunk/persistentstorage/dbx/4a753be34e0e9edd59c8d94de461d3e2  $ ls -la
total 16
drwx--x---.  2 splunk splunk 4096 Jun  7 11:53 .
drwx--x---. 70 splunk splunk 4096 Oct  5  2017 ..
-rw-------.  1 splunk splunk  158 Jun  1  2016 manifest.properties
-rw-------.  1 splunk splunk  248 Jun  7 12:31 state.xml
[redacted]:/opt/splunk/var/lib/splunk/persistentstorage/dbx/4a753be34e0e9edd59c8d94de461d3e2  $
```

splunk> .conf18

manifest.properties

$SPLUNK_HOME/var/lib/splunk/persistentstorage/dbx/4a753be34ertrsaf42fsdfds/manifest.properties

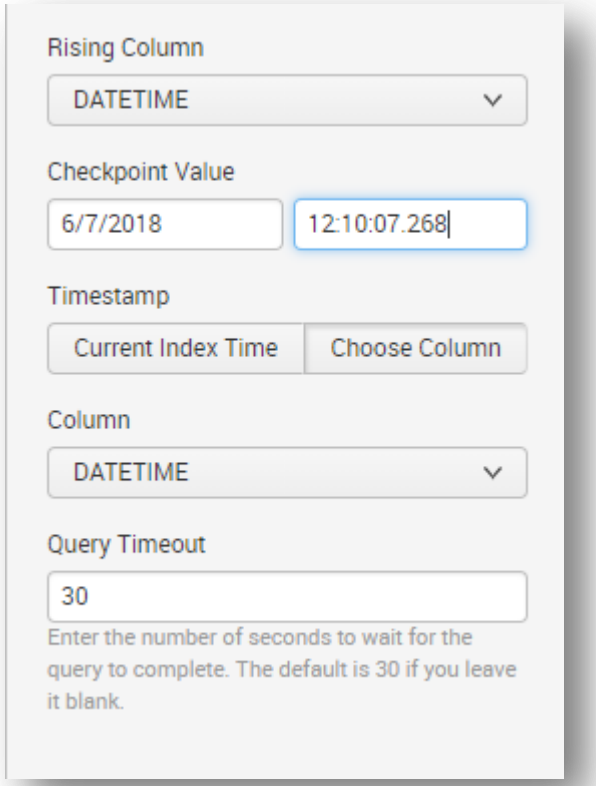
```
#Created at Wed Jun 01 14:32:02 EDT 2016
#Wed Jun 01 14:32:02 EDT 2016
version=1
name=dbmon-tail\://DB-Connection/DB-Input
type=xstream
created=1464805922416
```

state.xml

$SPLUNK_HOME/var/lib/splunk/persistentstorage/dbx/4a753be34ertrsaf42fsdfds/state.xml

Set Checkpoint Value

```
<list>
  [redacted]
  <value key="latest.myRisingColumnFieldName">
    <value class="sql-timestamp">2018-06-07 12:10:07.268782068</value>
  </value>
</list>
```



Time Zone

Data Lab | Configuration | Health | Search — Splunk DB Connect

Edit Input — Set SQL Query → Set Properties → Complete

Next | Cancel

Choose Table

Connection

Catalog: Select...

Schema: Select...

Table

Preview Data

SQL Editor — Format | Execute SQL

```
1  SELECT event_time
2  FROM sys.fn_get_audit_file ('D:\\\\Audit\\\\*',default,default)
3  WHERE event_time > ?
4  ORDER BY event_time ASC
```

	event_time
1	2018-08-23 16:49:34.6652473
2	2018-08-23 16:49:34.7702485
3	2018-08-23 16:49:34.7712493
4	2018-08-23 16:49:34.7742486
5	2018-08-23 16:49:34.7772266
6	2018-08-23 16:54:34.7930838
7	2018-08-23 16:54:34.8990835
8	2018-08-23 16:54:34.9000873
9	2018-08-23 16:54:34.9030323
10	2018-08-23 16:54:34.9060457

Settings

Template: Select...

Input Type: Batch | Rising

Follow these steps:

- ✓ 1. Choose a valid connection
- ✓ 2. Browse structure and type SQL or choose a template to explore your data
- ✓ 3. Pick a rising column and set the checkpoint value
- ✓ 4. Update your SQL to accept the checkpoint value and make sure it works correctly.

  To use a rising mode input, you need to filter the rising column with a WHERE statement and sort the results with ORDER BY.

  For example:

```
SELECT * FROM your_table
WHERE event_time > ?
ORDER BY event_time ASC
```

- ✓ 5. Click "Execute SQL" to review results

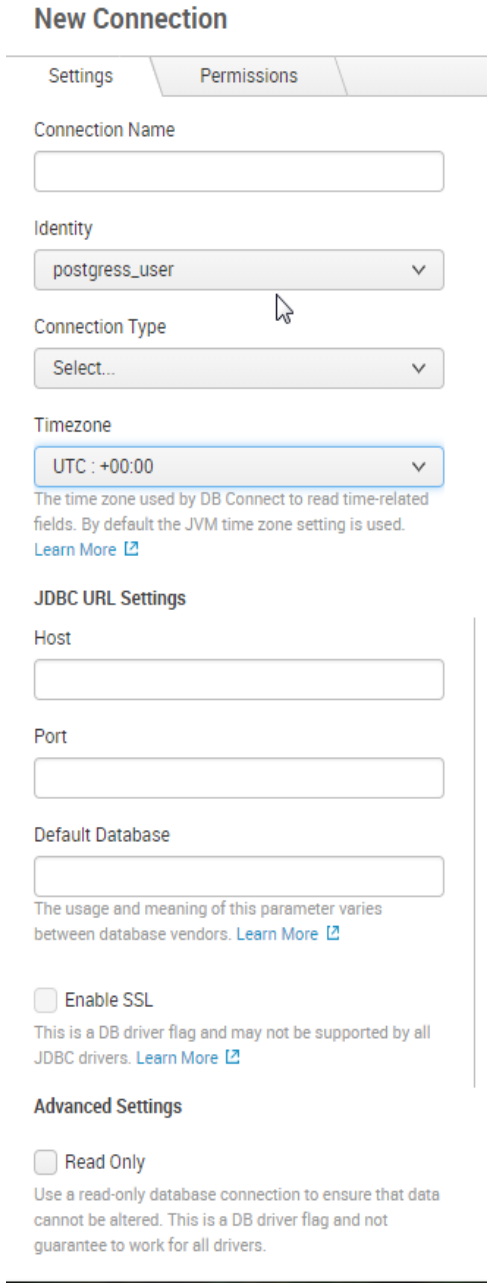
Rising Column: event_time

Checkpoint Value: Unlock & Edit

Timestamp: Current Index Time | Choose Column

Time Zone

- DB Connect v3 ignores TZ settings in props.conf on sourcetype
- Set “Timezone” in the DB input connection.
- IF 2 DB inputs are using same connection info but timestamps are in different TZ, you’ll need to have 2 connections.



Key Takeaways

1. Case matters so use “AS” clause
2. Use database views
3. Checkpoint value is in DB input state.xml
4. Set TZ in the DB connection settings NOT props.conf
5. Source syntax is different between DBX3
6. Ensure kv_mode=auto
7. Verify: ingest in a temp index with same sourcetype as events from DBX1

Thank You

Don't forget to rate this session in the .conf18 mobile app