



# 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, Splunk>, Listen to Your Data, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners. © 2018 Splunk Inc. All rights reserved.

# Agenda

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





# HANI ATALLA

Splunk Engineer



# So Why Ditch DB Connect v1?





# 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



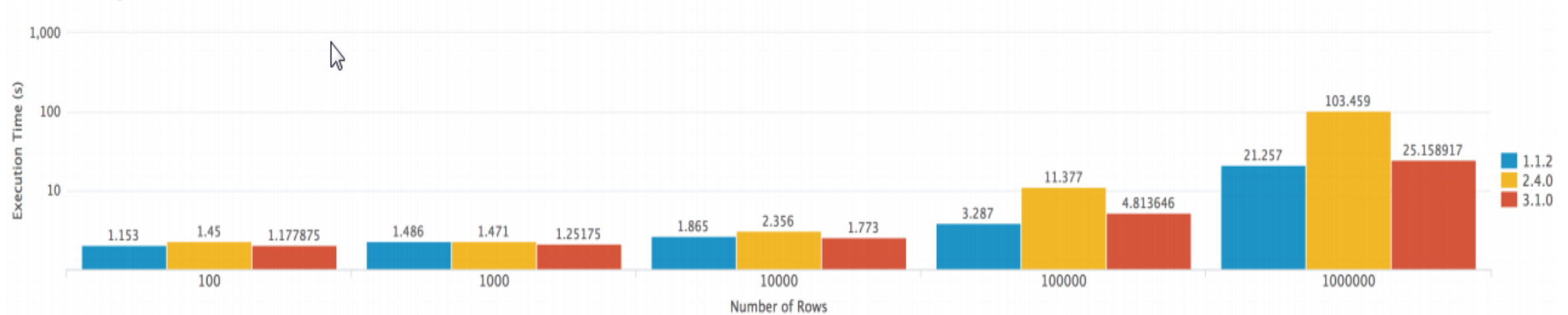
Flexibility

Stored procedures, 14 supported databases, Linux and Windows platform

# Performance - Queries

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

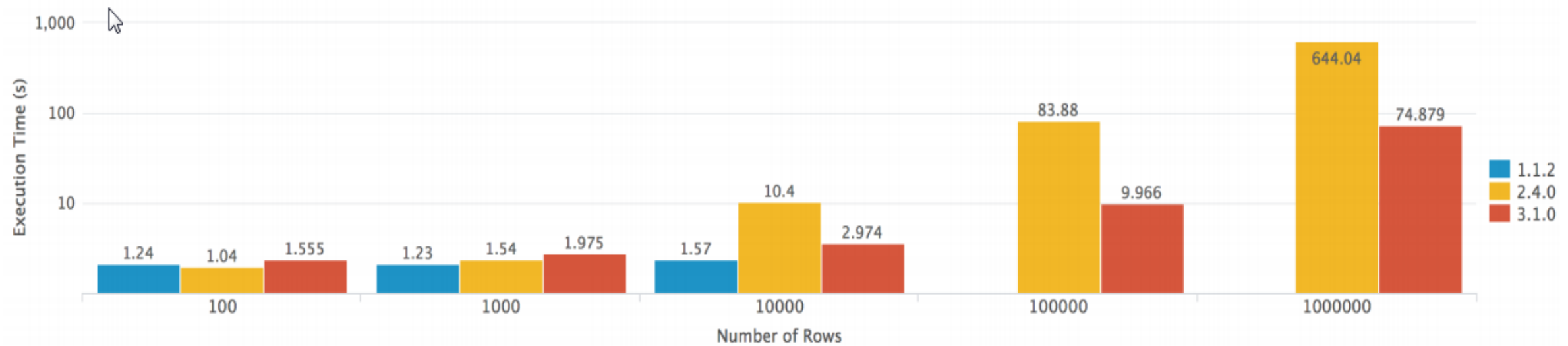
DB Connect Query Performance



# Performance - Outputs

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

DB Connect Output Performance





# 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]

130.60.4 - - [07/Jan 18:10:57:153] "GET /category.screen?category\_id=GIFTS&SESSIONID=5D1SLAFF10ADFF10 HTTP 1.1" 404 720 "http://buttercup-shopping.com/cart.do?action=view&itemId=EST-6&product\_id=FI-SW-01"  
128.241.220.82 - - [07/Jan 18:10:57:123] "GET /product.screen?product\_id=FL-DSH-01&SESSIONID=5D5SL7FF6ADFF9 HTTP 1.1" 404 3322 "http://buttercup-shopping.com/category.screen?category\_id=GIFTS"  
317 27.160.0.0 - - [07/Jan 18:10:56:156] "GET /oldlink?item\_id=EST-26&SESSIONID=5D5SL9FF1ADFF3 HTTP 1.1" 200 1318 "http://buttercup-shopping.com/cart.do?action=purchase&itemId=EST-16&product\_id=RP-LI-02"  
100 125.17 14.14.14.14 [07/Jan 18:10:55:187] "GET /category.screen?category\_id=FLOWERS&SESSIONID=5D5SL8FF1ADFF6 HTTP 1.1" 200 3885 "http://buttercup-shopping.com/category.screen?category\_id=FLOWERS&SESSIONID=5D5SL8FF1ADFF6"  
100 125.17 14.14.14.14 [07/Jan 18:10:55:188] "GET /category.screen?category\_id=FLOWERS&SESSIONID=5D5SL8FF1ADFF6 HTTP 1.1" 200 3885 "http://buttercup-shopping.com/category.screen?category\_id=FLOWERS&SESSIONID=5D5SL8FF1ADFF6"

# On DB Connect v1.x



# Option 1

The Long Path



# Dual Install

splunk> Apps Administrator Messages Settings Activity Help Find

## Apps

Browse more apps Install app from file Create app

Showing 1-18 of 18 items Results per page 25

Name	Folder name	Version	Update checking	Visible	Sharing	Status	Actions
SplunkForwarder	SplunkForwarder		Yes	No	App   Permissions	Disabled   Enable	
SplunkLightForwarder	SplunkLightForwarder		Yes	No	App   Permissions	Disabled   Enable	
Log Event Alert Action	alert_logevent	6.3.1511	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Webhook Alert Action	alert_webhook	6.3.1511	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Apps Browser	appsbrowser	6.3.1511	Yes	Yes	App   Permissions	Enabled	Launch app   Edit properties   View objects
Splunk DB Connect v1	dbx	1.2.2	Yes	Yes	App   Permissions	Enabled   Disable	Set up   Edit properties   View objects   View details on SplunkApps
framework	framework		Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Getting started	gettingstarted	1.0	Yes	Yes	App   Permissions	Disabled   Enable	
introspection_generator_addon	introspection_generator_addon	6.3.1511	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Home	launcher		Yes	Yes	App   Permissions	Enabled	Launch app   Edit properties   View objects
learned	learned		Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
legacy	legacy		Yes	No	App   Permissions	Disabled   Enable	
sample data	sample_app		Yes	No	App   Permissions	Disabled   Enable	
Search & Reporting	search	6.3.1511	Yes	Yes	App   Permissions	Enabled	Launch app   Edit properties   View objects
Splunk DB Connect v2	splunk_app_db_connect	2.1.0	Yes	Yes	App   Permissions	Enabled   Disable	Launch app   Edit properties   View objects   View details on SplunkApps
Splunk Archiver App	splunk_archiver	1.0	Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects   View details on SplunkApps
splunk_httpinput	splunk_httpinput		Yes	No	App   Permissions	Enabled   Disable	Edit properties   View objects
Distributed Management Console	splunk_management_console	6.3.1511	Yes	Yes	App   Permissions	Enabled	Launch app   Edit properties   View objects

About Support File a Bug Documentation Privacy Policy

© 2005-2015 Splunk Inc. All rights reserved.

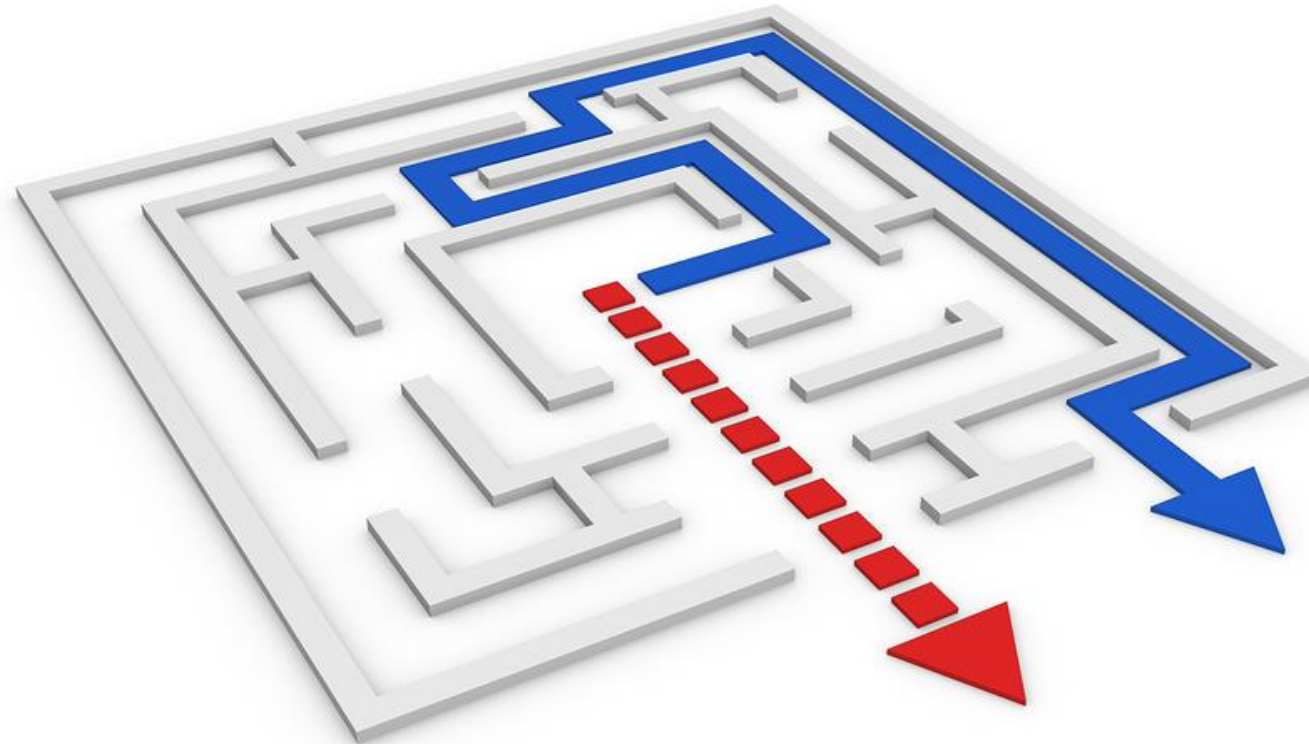
# The Long Path to v3 from v1 [ Part I ]

- ▶ 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]





# 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**

☒ 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-MM-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]

Edit Input

Set SQL Query

Set Properties

Complete

<

Next

Cancel

Choose Table

Connection  
IPTSTEST\_INTERFACE

Catalog  
Select...

Schema  
Select...

Table

Preview Data

SQL Editor

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

Format Execute SQL

Settings

Template  
mssql:audit

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

Query Timeout  
30  
Enter the number of seconds to wait for the



From [v1]

A Database input will fetch data from a SQL database.

Input Type  
Tail (Follow based on increasing value)

Database  
[Select]

☒ 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-MM-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 [v3]

Edit Input

Set SQL Query Set Properties Complete

Choose Table Preview Data Settings

Connection  
IPIS\_TEST\_INTERFACE

Catalog  
Select

Schema  
Select

Table

SQL Editor

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

Format Execute SQL

Template  
mssql:audit

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

Query Timeout  
30

Enter the number of seconds to wait for the

# DB Input [Components]

A Database input will fetch data from a SQL database.

## Input Type

Tail (Follow based on increasing value)

## Database

☒ 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

yyy-MM-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

# 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;
```



< Hide Fields    All Fields

Interesting Fields

- # action 3
- a action\_name 3
- # os\_process 100+
- # port 100+
- a protocol 1
- # sessionid 100+
- a timestamp 100+
- a userhost 20

+ Extract New Fields



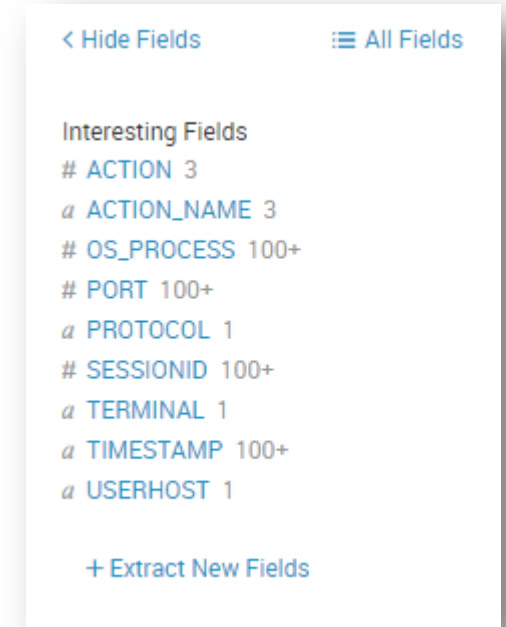
# SQL: Upper / Lower Case Behavior v3

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;
```

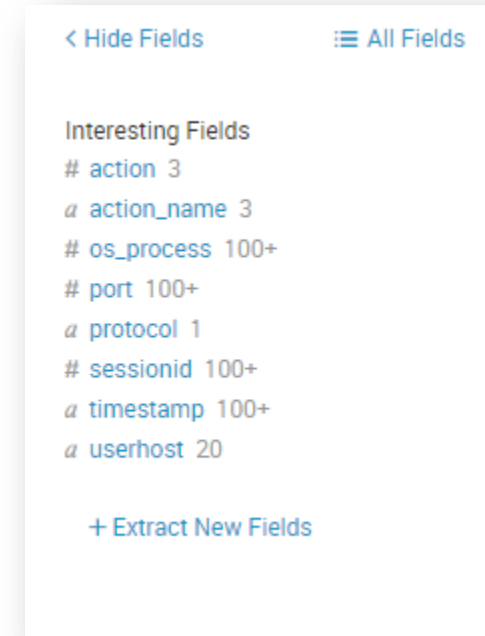
```
SELECT  
  ACTION,  
  ACTION_NAME,  
  PROTOCOL,  
  SESSIONID,  
  PORT,  
  USERHOST,  
  OS_PROCESS,  
  TERMINAL,  
  TIMESTAMP  
FROM Audit_Trail;
```



# 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



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

8/22/2018 20:58:30.890

Timestamp

☒ Current Index Time ☐ Choose Column

Column

event\_time ▼



# DB Input [DB Connect v1]

Grep is Your Best Friend

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

```
:/opt/splunk/var/lib/splunk/persistentstorage/dbx $ ls
08ab6e930c7767e7f0ce29bb332f0eee 42434a40f2c8c41c6359a407829f7c7f 69570151749c9e7374356db043daa48a 7bb64863f78827a573a4a18c35bb330b acfbefda60ba4eb642b92fa8141507b2 d64779ac26265bf0ecdba797b9ac3a87 ee5f54f24f5b6adb5d2940b62b209367
10d8ee7d5604f759b18369f28afe6f9c 43ee8ac778f4a1b30fbf48714b846f3f 6c83478b2b776013f7ea53a3e776187d 7d4d12a0176b43d1f1eebd792d9a1fc3 aeb1921fd4c6adbde8d00f895b79a6a8 d6f719336a5dc70ea29bfb048bf4d73 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 fcf7c2ae597ddba921b67b9b7881869
1f5be5408b27d0f70f350f288693cfff 5963b5008b5045046234dbb514bd987e 76063dd69d24751724fad74d7d685064 8752c73929b0c83bc0ca4416ec99844f bdacc562021a640020a701450a8eef4e e2e919c7933152b06209b64b2b0cdc67 fe75d6db5d6c4b59ec79a6b391908363
22374d872f8a0861fd24f5380da0f3a9 59677d24b83328f3db4489e16ed7bda1 771d8096d00d67d9a155a66c286f4a81 9e2daa25c45133d4ab2188e6b6c83d4e cdbaaa393c888b8828f66f3a24f33a56 eb86b0c5c99ab70ceb97bf304a6884bc global
334b78d4596a7f054d41fdd6a90d7fa0 5b688b7e57674193c0e1c38ea0966f5c 793785b5f0cd64f3f2cf4af9d64772a3 a132a60a0e16ddbfb2b01d1618aa73f9c d022ff3232435c5d4d4c9f1386c9c8b5 ecbe5c1cfbb932e18b1c8d234d2464e7
3c4a47d27733e606ddc7bdbbf0d819a1 642d10b7035b6cee8e808b69dfb2b31f 7994ff22d6c6370a11d8f612ede12154 ab42494b296d2fa268c6c71cf64b14c6 d5f05ae7dd75d57d6a5e7750f6a84403 edf2b2b854b6158e5668cbab36c9fb37

:/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>
```

```
:/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
:/opt/splunk/var/lib/splunk/persistentstorage/dbx/4a753be34e0e9edd59c8d94de461d3e2 $
```

# 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

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

# Set Checkpoint Value

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



Rising Column  
DATETIME

Checkpoint Value  
6/7/2018 12:10:07.268

Timestamp  
Current Index Time Choose Column

Column  
DATETIME

Query Timeout  
30  
Enter the number of seconds to wait for the query to complete. The default is 30 if you leave it blank.

# Time Zone

Data Lab Configuration Health ▾ Search

Splunk DB Connect

## Edit Input

Set SQL Query

Set Properties

Complete



Next

Cancel

## Choose Table

## Connection

## Catalog

## Schema

## 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

## 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

## Checkpoint Value

Unlock &amp; Edit

## Timestamp

# 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.



## New Connection

Settings

Permissions

Connection Name

Identity

postgres\_user

Connection Type

Select...

Timezone

UTC : +00:00

The time zone used by DB Connect to read time-related fields. By default the JVM time zone setting is used.

[Learn More](#)

### JDBC URL Settings

Host

Port

Default Database

The usage and meaning of this parameter varies between database vendors. [Learn More](#)

☐ Enable SSL

This is a DB driver flag and may not be supported by all JDBC drivers. [Learn More](#)

### Advanced Settings

☐ Read Only

Use a read-only database connection to ensure that data cannot be altered. This is a DB driver flag and not guarantee to work for all drivers.



## 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

**.conf18**

**splunk>**