



Amazon Web Services Data Engineering Immersion Day

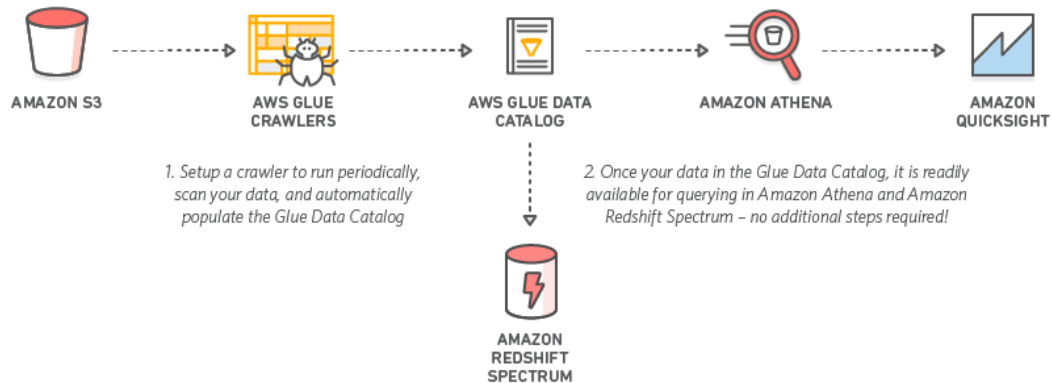
Exploring Data Lake with Amazon Athena and Amazon
Quicksight
Jun 2019

Table of Contents

<i>Introduction.....</i>	<i>2</i>
Prerequisites.....	2
Getting Started.....	2
<i>Query Data with Amazon Athena</i>	<i>3</i>
<i>Build an Amazon QuickSight Dashboard.....</i>	<i>7</i>
Set up QuickSight.....	7
Create QuickSight Charts.....	12
Create QuickSight Parameters.....	15
Create a QuickSight Filter.....	18
Add Calculated Fields.....	20
<i>Amazon QuickSight ML-Insights (Optional).....</i>	<i>24</i>

Introduction

This lab introduces you to AWS Glue, Amazon Athena, and Amazon QuickSight. AWS Glue is a fully managed data catalog and ETL service; Amazon Athena queries data; and Amazon QuickSight provides visualization of the data you import.



Prerequisites

The DMS Lab and Glue ETL lab is a prerequisite for this lab.

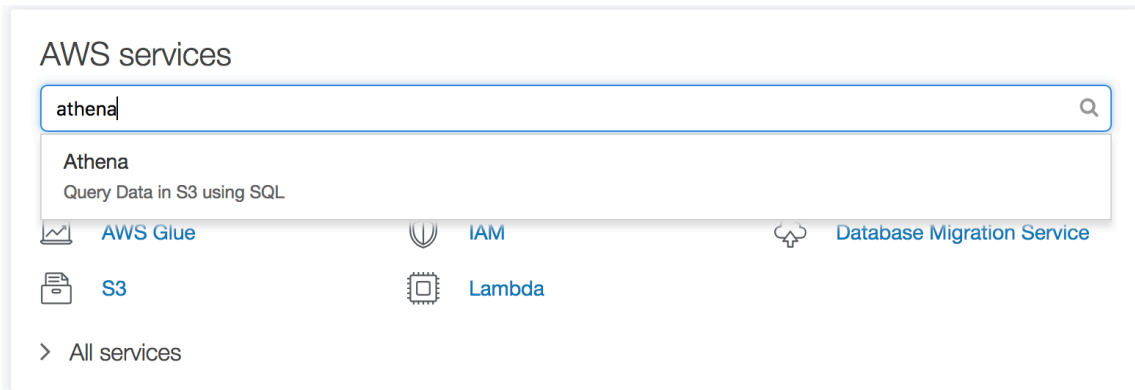
Getting Started

In this lab, you will complete the following tasks:

1. [Query data and create a view with Amazon Athena](#)
2. [Build a dashboard with Amazon QuickSight](#)

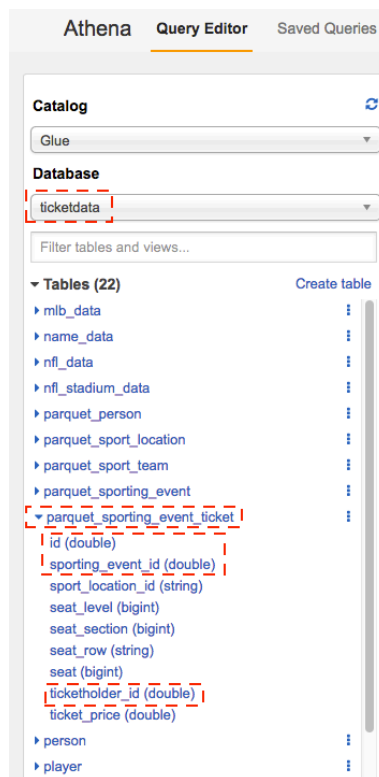
Query Data with Amazon Athena

1. In the AWS services console, search for **Athena**.



2. In the Query Editor, select your newly created database e.g., "ticketdata".
3. Click the table named "parquet_sporting_event_ticket" to inspect the fields.

Note: The type for fields id, sporting_event_id and ticketholder_id should be (double).

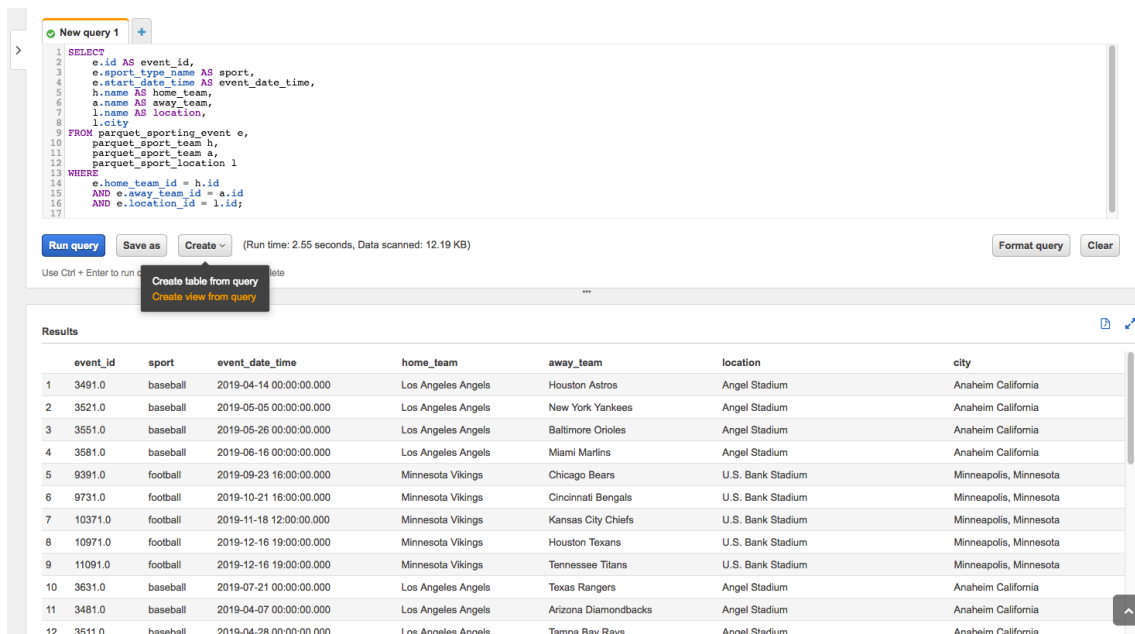


Next, we will query across tables `parquet_sporting_event`, `parquet_sport_team`, and `parquet_sport_location`.

4. Copy the following SQL syntax into the New Query 1 tab and click **Run Query**.

```
SELECT
    e.id AS event_id,
    e.sport_type_name AS sport,
    e.start_date_time AS event_date_time,
    h.name AS home_team,
    a.name AS away_team,
    l.name AS location,
    l.city
FROM parquet_sporting_event e,
    parquet_sport_team h,
    parquet_sport_team a,
    parquet_sport_location l
WHERE
    e.home_team_id = h.id
    AND e.away_team_id = a.id
    AND e.location_id = l.id;
```

The results appear beneath the query window.



The screenshot shows a SQL query editor interface. The query window displays the SQL query from the previous block. Below the query window, there are buttons for 'Run query', 'Save as', and 'Create'. The 'Run query' button is highlighted. Below the buttons, there is a dropdown menu with options 'Create table from query' and 'Create view from query'. The 'Results' section shows a table with 12 rows of data. The table has columns: event_id, sport, event_date_time, home_team, away_team, location, and city. The data is as follows:

	event_id	sport	event_date_time	home_team	away_team	location	city
1	3491.0	baseball	2019-04-14 00:00:00.000	Los Angeles Angels	Houston Astros	Angel Stadium	Anaheim California
2	3521.0	baseball	2019-05-05 00:00:00.000	Los Angeles Angels	New York Yankees	Angel Stadium	Anaheim California
3	3551.0	baseball	2019-05-26 00:00:00.000	Los Angeles Angels	Baltimore Orioles	Angel Stadium	Anaheim California
4	3581.0	baseball	2019-06-16 00:00:00.000	Los Angeles Angels	Miami Marlins	Angel Stadium	Anaheim California
5	9391.0	football	2019-09-23 16:00:00.000	Minnesota Vikings	Chicago Bears	U.S. Bank Stadium	Minneapolis, Minnesota
6	9731.0	football	2019-10-21 16:00:00.000	Minnesota Vikings	Cincinnati Bengals	U.S. Bank Stadium	Minneapolis, Minnesota
7	10371.0	football	2019-11-18 12:00:00.000	Minnesota Vikings	Kansas City Chiefs	U.S. Bank Stadium	Minneapolis, Minnesota
8	10971.0	football	2019-12-16 19:00:00.000	Minnesota Vikings	Houston Texans	U.S. Bank Stadium	Minneapolis, Minnesota
9	11091.0	football	2019-12-16 19:00:00.000	Minnesota Vikings	Tennessee Titans	U.S. Bank Stadium	Minneapolis, Minnesota
10	3631.0	baseball	2019-07-21 00:00:00.000	Los Angeles Angels	Texas Rangers	Angel Stadium	Anaheim California
11	3481.0	baseball	2019-04-07 00:00:00.000	Los Angeles Angels	Arizona Diamondbacks	Angel Stadium	Anaheim California
12	3511.0	baseball	2019-04-28 00:00:00.000	Los Angeles Angels	Tampa Bay Rays	Angel Stadium	Anaheim California

5. As shown above Click **Create** and then select **Create view from query**

6. Name the view "sporting_event_info" and click **Create**.

Create view

Views are updated each time you run a query

Name

Cancel Create

Your new view is created

Athena Query Editor

Saved Queries History AWS Glue Data Catalog Workgroup : primary

Catalog

Glue

Database

ticketdata

Filter tables and views...

Tables (22) Create table

Views (1) Create view

▼ sporting_event_info

event_id (double)

sport (string)

event_date_time (timestamp)

home_team (string)

away_team (string)

location (string)

city (string)

New query 1

```

1 CREATE OR REPLACE VIEW "sporting_event_info" AS
2 SELECT
3     e.id AS event_id,
4     e.sport_type_name AS sport,
5     e.start_date_time AS event_date_time,
6     h.name AS home_team,
7     a.name AS away_team,
8     l.name AS location,
9     l.city
10 FROM parquet_sporting_event e,
11      parquet_sport_team h,
12      parquet_sport_team a,
13      parquet_sport_location l
14 WHERE
15     e.home_team_id = h.id
16     AND e.away_team_id = a.id
17     AND e.location_id = l.id;
18

```

Run query Save as Create (Run time: 1.16 seconds, Data scanned: 0 KB)

Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

7. Copy the following SQL syntax into the New Query 2 tab and click **Run Query**.

```

SELECT t.id AS ticket_id,
       e.event_id,
       e.sport,
       e.event_date_time,
       e.home_team,
       e.away_team,
       e.location,
       e.city,
       t.seat_level,
       t.seat_section,
       t.seat_row,

```

```

t.seat,
t.ticket_price,
p.full_name AS ticketholder
FROM sporting_event_info e,
parquet_sporting_event_ticket t,
parquet_person p
WHERE
t.sporting_event_id = e.event_id
AND t.ticketholder_id = p.id

```

The results appear beneath the query window.

The screenshot shows the Athena Query Editor interface. At the top, there's a navigation bar with 'Athena', 'Query Editor', 'Saved Queries', 'History', 'AWS Glue Data Catalog', and 'Workgroup: primary'. Below this, there's a toolbar with 'New query 1', 'New query 2', 'Run query', 'Save as', 'Create', 'Format query', and 'Clear'. The main area contains a SQL query that selects ticket information from 'sporting_event_info', 'parquet_sporting_event_ticket', and 'parquet_person' tables. Below the query, there's a 'Results' section displaying a table with 12 rows of data. The table columns are: ticket_id, event_id, sport, event_date_time, home_team, away_team, location, city, seat_level, seat_section, seat_row, seat, ticket_price, and ticketholder. The data shows tickets for Philadelphia Eagles games against the Cincinnati Bengals at Lincoln Financial Field.

ticket_id	event_id	sport	event_date_time	home_team	away_team	location	city	seat_level	seat_section	seat_row	seat	ticket_price	ticketholder	
1	4320911.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	B	3	44.64	Corinne Buck
2	4320921.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	B	2	44.64	Corinne Buck
3	4320931.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	B	1	44.64	Corinne Buck
4	4330681.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	B	2	133.92	Corinne Buck
5	4317081.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	2	44.64	Corinne Buck
6	4317091.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	1	44.64	Corinne Buck
7	4320961.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	1	44.64	Corinne Buck
8	4320951.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	2	44.64	Corinne Buck
9	4320941.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	3	44.64	Corinne Buck
10	4330611.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	2	133.92	Corinne Buck
11	4330601.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	3	133.92	Corinne Buck
12	4330621.0	9881.0	football	2019-10-21 16:00:00.000	Philadelphia Eagles	Cincinnati Bengals	Lincoln Financial Field	Philadelphia, Pennsylvania	2	11	A	1	133.92	Corinne Buck

- As shown above Click **Create view from query**.
- Name the view "sporting_event_ticket_info" and click **Create**.

Create view

Views are updated each time you run a query

Name

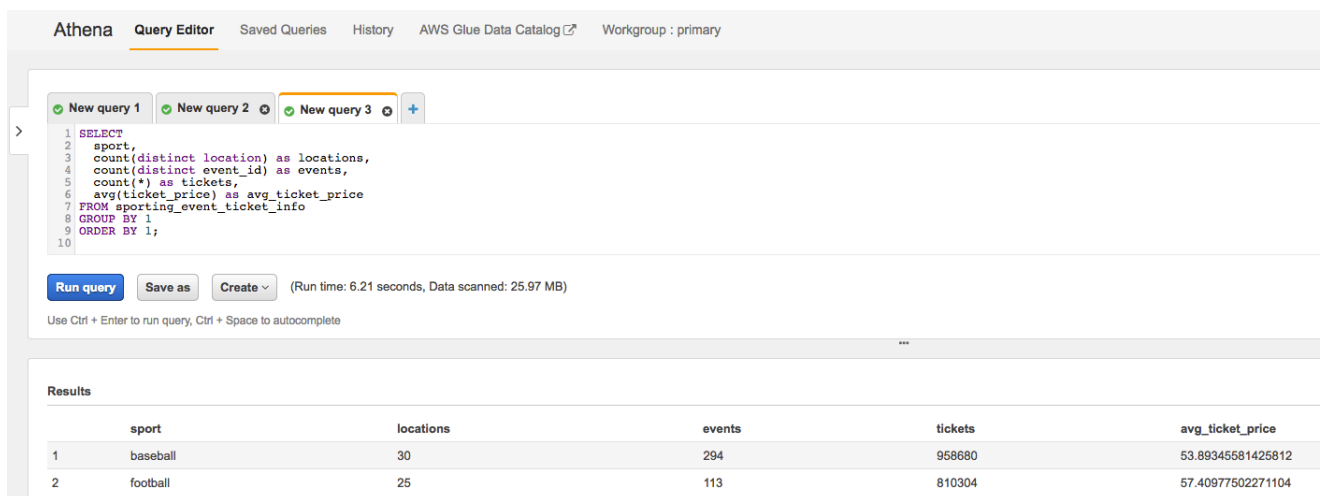
Cancel

Create

10. Copy the following SQL syntax into the New Query 3 tab and click **Run Query**.

```
SELECT
  sport,
  count(distinct location) as locations,
  count(distinct event_id) as events,
  count(*) as tickets,
  avg(ticket_price) as avg_ticket_price
FROM sporting_event_ticket_info
GROUP BY 1
ORDER BY 1;
```

Your query returns two results in approximately five seconds. The query scans 25MB of data, which prior to converting to parquet, would have been 1.59GB of CSV files.



The screenshot shows the AWS Athena Query Editor interface. At the top, there are tabs for 'New query 1', 'New query 2', and 'New query 3'. The 'New query 3' tab is active, displaying the following SQL query:

```
1 SELECT
2   sport,
3   count(distinct location) as locations,
4   count(distinct event_id) as events,
5   count(*) as tickets,
6   avg(ticket_price) as avg_ticket_price
7 FROM sporting_event_ticket_info
8 GROUP BY 1
9 ORDER BY 1;
```

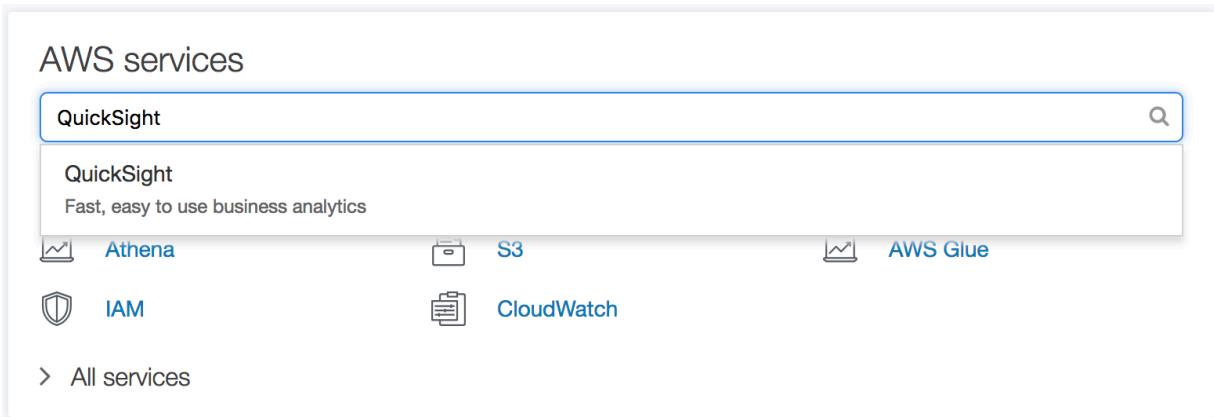
Below the query editor, there are buttons for 'Run query', 'Save as', and 'Create'. A status bar indicates '(Run time: 6.21 seconds, Data scanned: 25.97 MB)'. Below the query editor, there is a 'Results' section showing a table with 5 columns: 'sport', 'locations', 'events', 'tickets', and 'avg_ticket_price'. The table contains two rows of data:

	sport	locations	events	tickets	avg_ticket_price
1	baseball	30	294	958680	53.89345581425812
2	football	25	113	810304	57.40977502271104

Build an Amazon QuickSight Dashboard

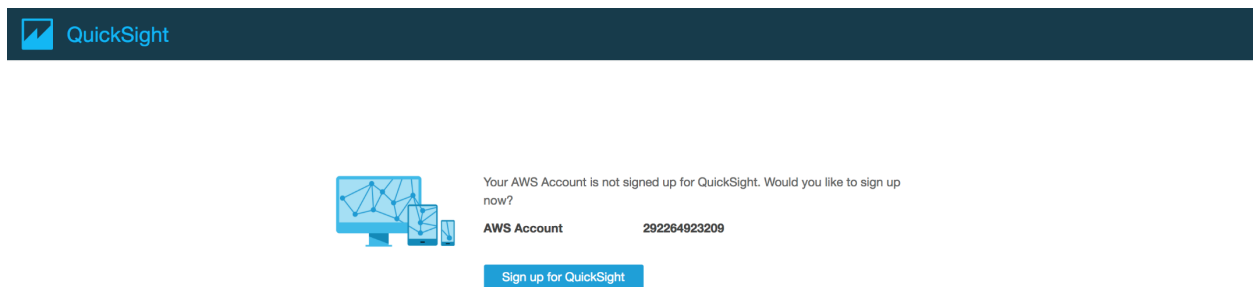
Set up QuickSight

1. In the AWS services console, search for **QuickSight**.



If this is the first time you have used QuickSight, you are prompted to create an account.

2. Click **Sign up for QuickSight**.



3. For account type, choose **Standard**. If you plan to complete the Bonus Exercise, please choose **Enterprise Version**
4. Click **Continue**.

Create your QuickSight account

Edition	<input checked="" type="radio"/> Standard	<input type="radio"/> Enterprise
First author with 1GB SPICE	FREE	FREE
Team trial for 60 days (4 authors)*	FREE	FREE
Additional author per month (yearly)**	\$9	\$18
Additional author per month (monthly)**	\$12	\$24
Additional readers (Pay-per-Session)	N/A	\$0.30/session (max \$5/reader/month) ****
Additional SPICE per month	\$0.25 per GB	\$0.38 per GB
Single Sign On with SAML or OpenID Connect	✓	✓
Connect to spreadsheets, databases & business apps	✓	✓
Access data in Private VPCs		✓
Row-level security for dashboards		✓
Hourly refresh of SPICE data		✓
Secure data encryption at rest		✓
Connect to your Active Directory		✓
Use Active Directory Groups ***		✓

* Trial authors are auto-converted to month-to-month subscription upon trial expiry
 ** Each additional author includes 10GB of SPICE capacity
 *** Active Directory groups are available in accounts connected to Active Directory
 **** Sessions of 30-minute duration. Total charges for each reader are capped at \$5 per month. [Conditions](#) apply

[Continue](#)

- On the Create your QuickSight account page, fill out your name and email address.
- Select region and the check boxes to enable autodiscovery, Amazon Athena, and Amazon S3.
- Click **Choose S3 buckets** and select your DMS bucket (e.g., "dms-lab-george").
- Click **Finish**.

Create your QuickSight account

Edition

Standard

QuickSight account name

Glue-Lab-George

You will need this for you and others to sign in.

Notification email address

julbrigh+dataenglab@amazon.com

For QuickSight to send important notifications.

QuickSight capacity region

US East (N. Virginia)

Select a region.

☒ Enable autodiscovery of data and users in your Amazon Redshift, Amazon RDS and AWS IAM services.

☒ Amazon Athena
Enables QuickSight access to Amazon Athena databases

Please ensure the right Amazon S3 buckets are also enabled for QuickSight.

☒ Amazon S3 (1 bucket)
Enables QuickSight to auto-discover your Amazon S3 buckets

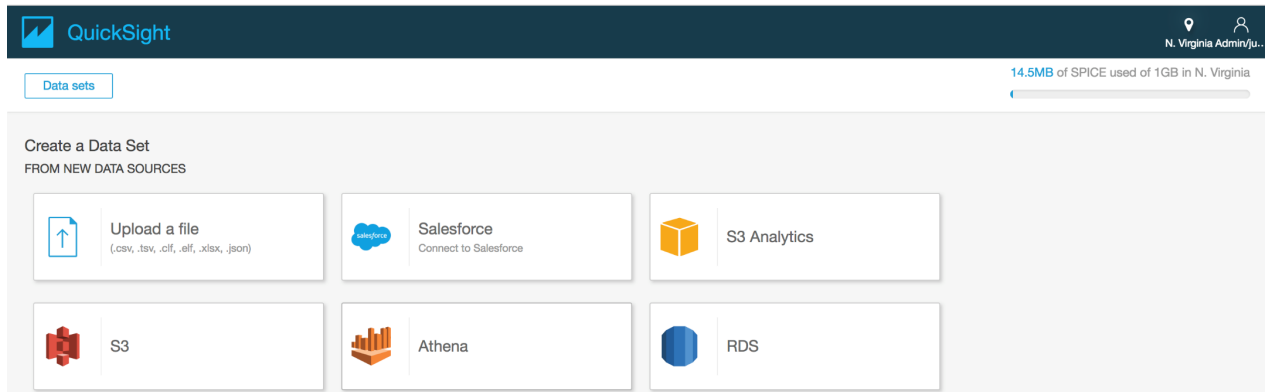
[Choose S3 buckets](#)

☐ Amazon S3 Storage Analytics
Enables QuickSight to visualize your S3 Storage Analytics data

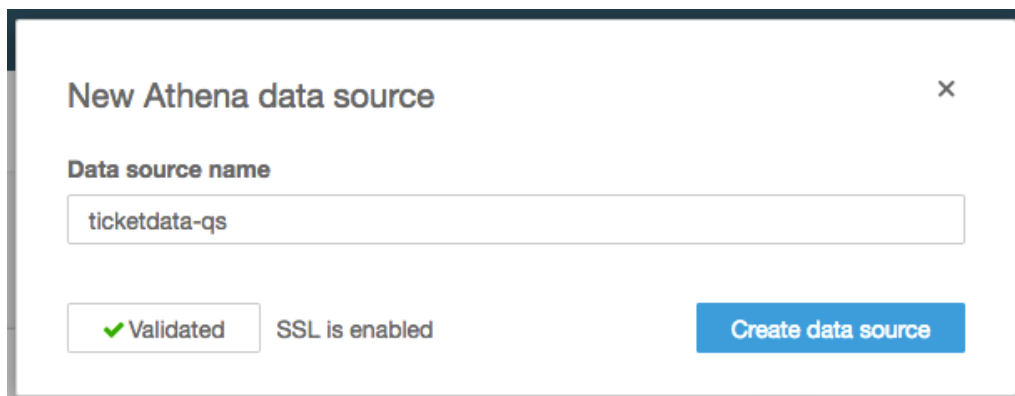
☐ Amazon IoT Analytics
Enable QuickSight to visualize your IoT Analytics data

Finish

9. On the QuickSight landing page, click **Manage Data**.
10. Click **New Data Set**.
11. On the Create a Data Set page, select **Athena** as the data source.



12. For Data source name, type "ticketdata-qs" and click **Validate connection**.
13. Click **Create data source**.



14. In the Database drop-down list, select the database name you created in the AWS Glue lab.
15. Choose the "sporting_event_ticket_info" table and click **Select**.

Choose your table

ticketdata-qs

Database: contain sets of tables.

ticketdata

Tables: contain the data you can visualize.

☐ sporting_event_ticket

☐ sporting_event_ticket_1bb4a008b349ed873527a4c2b9f8ac5f

☒ sporting_event_ticket_info

☐ ticket_purchase_hist

☐ ticket_purchase_hist_95f83e3d847527d7c4e84a4949d62d2b

Edit/Preview data

Use custom SQL

Select

16. To finish data set creation, choose the option **Import to SPICE for quicker analytics** and click **Visualize**.

Finish data set creation

Table: sporting_event_ticket_info
Data source: ticketdata-qs
Schema: ticketdata

☒ Import to SPICE for quicker analytics

✓ 20.9GB available

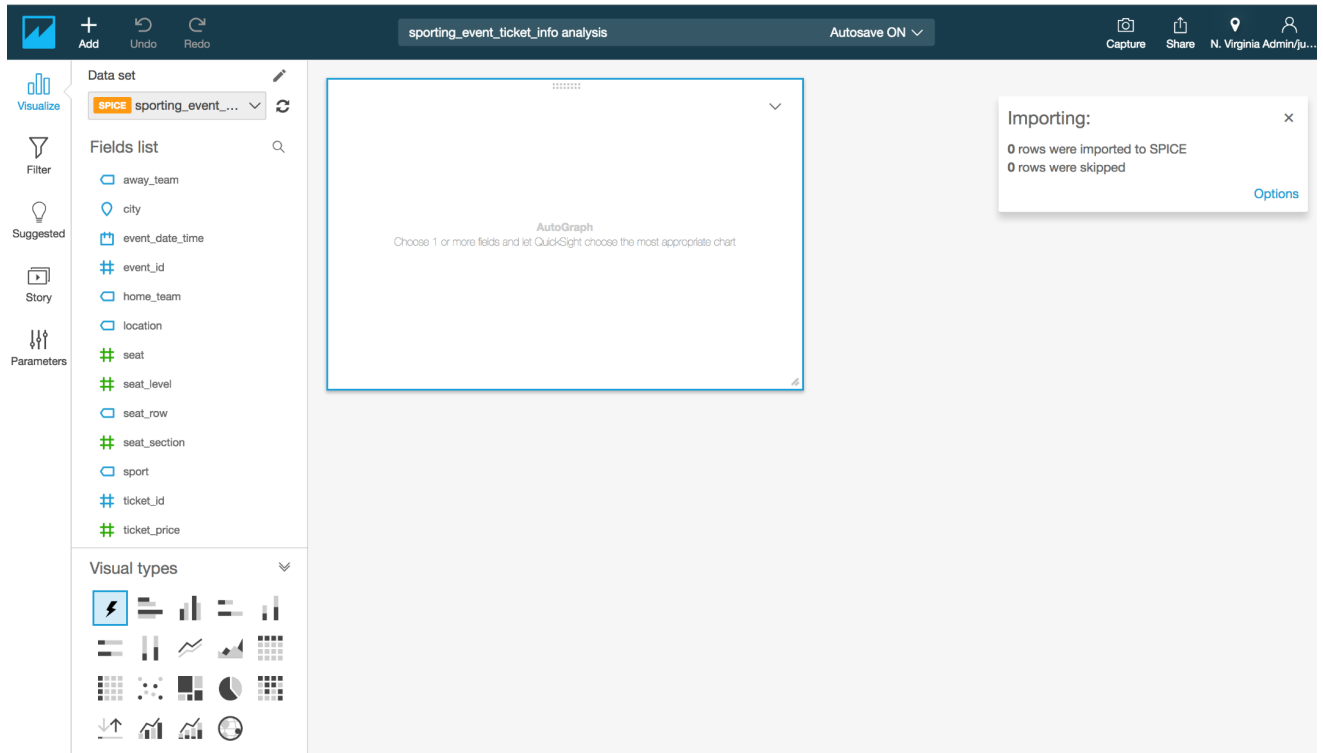
SPICE

☐ Directly query your data

Edit/Preview data

Visualize

You will now be taken to the QuickSight Visualize interface where you can start building your dashboard.

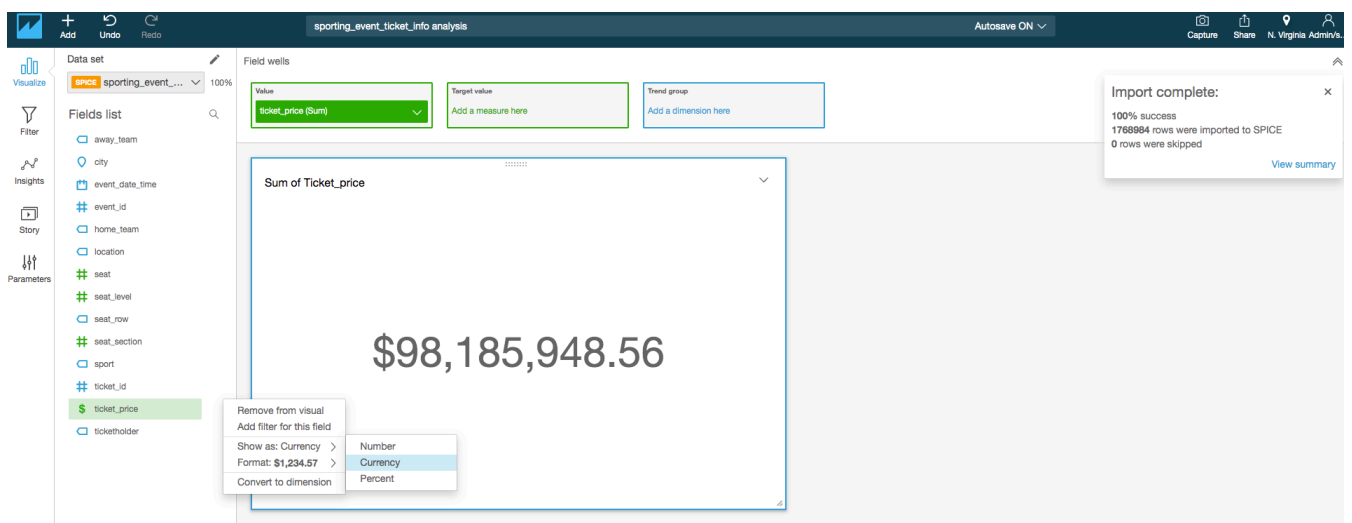


Note: The SPICE dataset will take a few minutes to be built, but you can continue to create some charts on the underlying data.

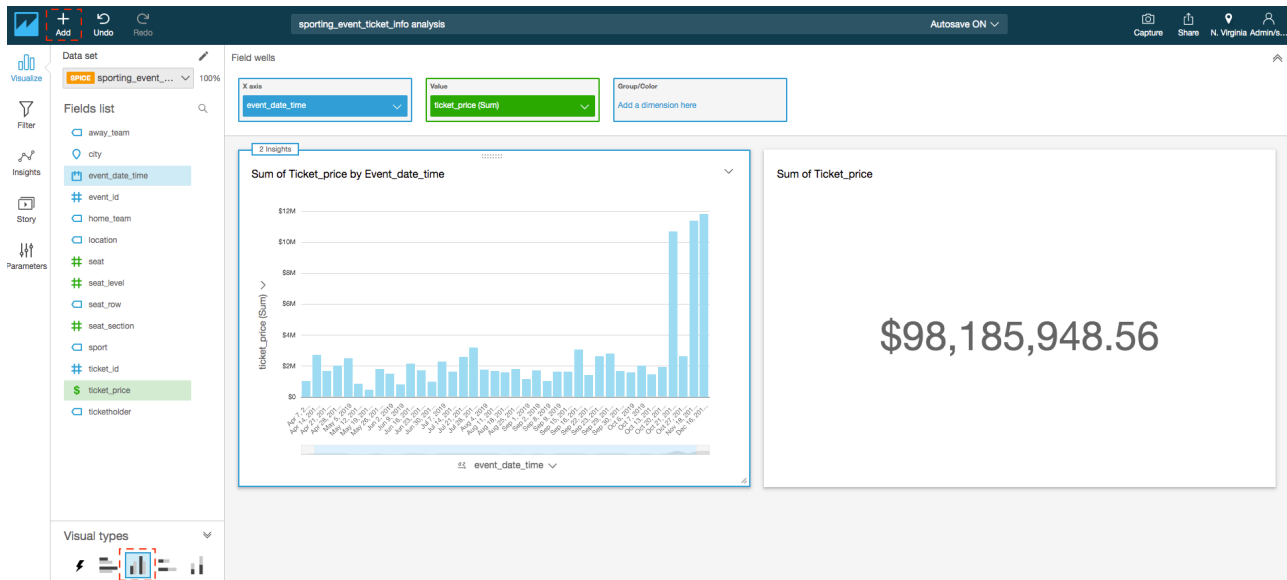
Create QuickSight Charts

In this section we will take you through some of the different chart types.

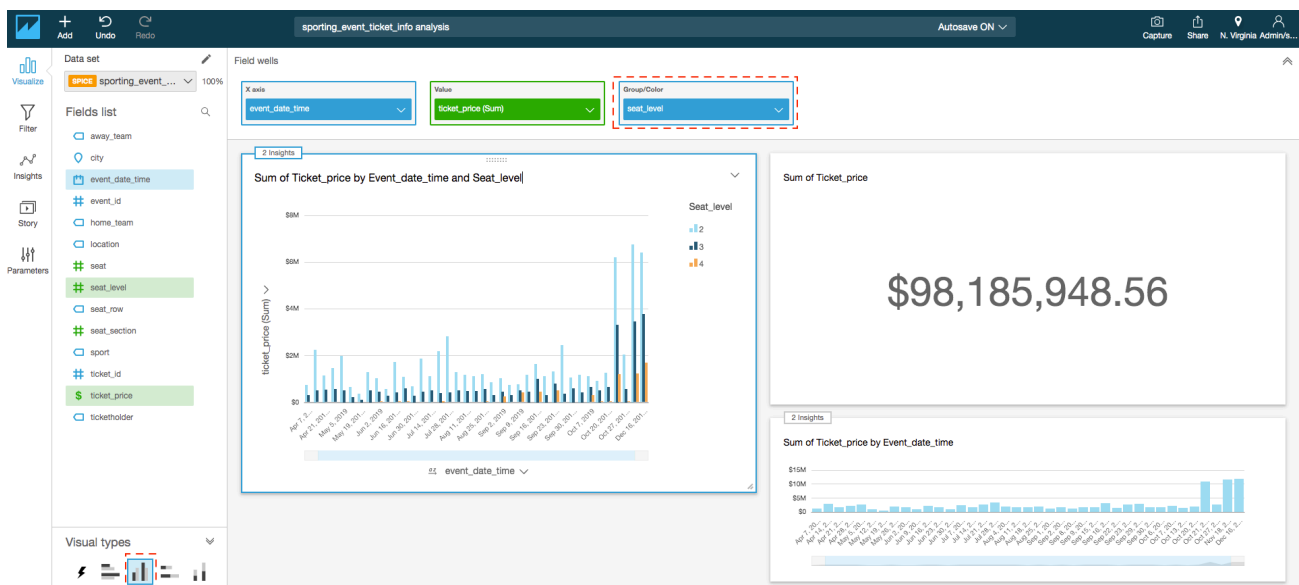
1. In the Fields list, click the "ticket_price" column to populate the chart.
2. Click the expand icon in corner of "ticket_price" field and select format as currency to show numbers in dollar amount.



3. You can add new visual and keep building your dashboard by clicking Add button at top left corner of screen. In the **Visual types** area, choose the **Vertical bar chart** icon. This layout requires a value for the X-axis. Click the "event_date_time" field and you should see the visualization update.

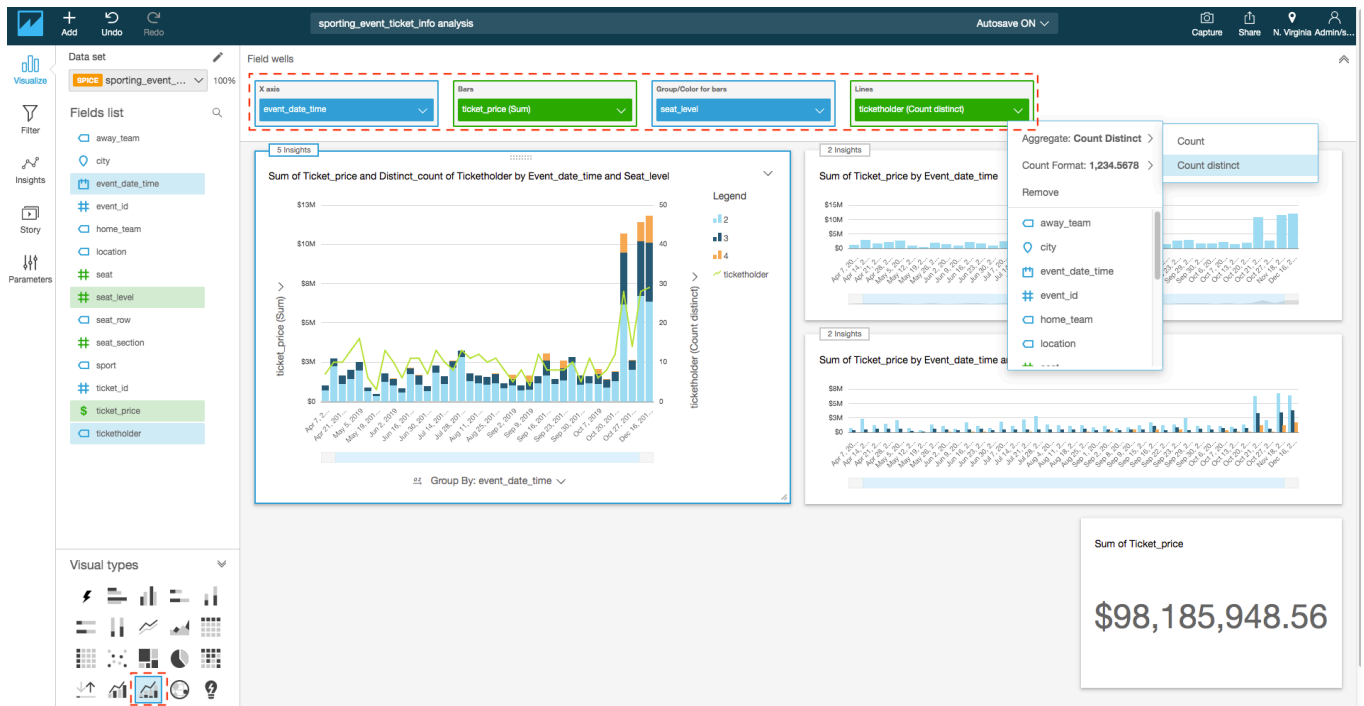


4. Add new Visual and you can drag and move other visuals to adjust space in dashboard. In the Fields list, click and drag the **seat_level** field to the Group/Color box in the Field wells pane. You can also use the slider below the x axis to fit all of the data.

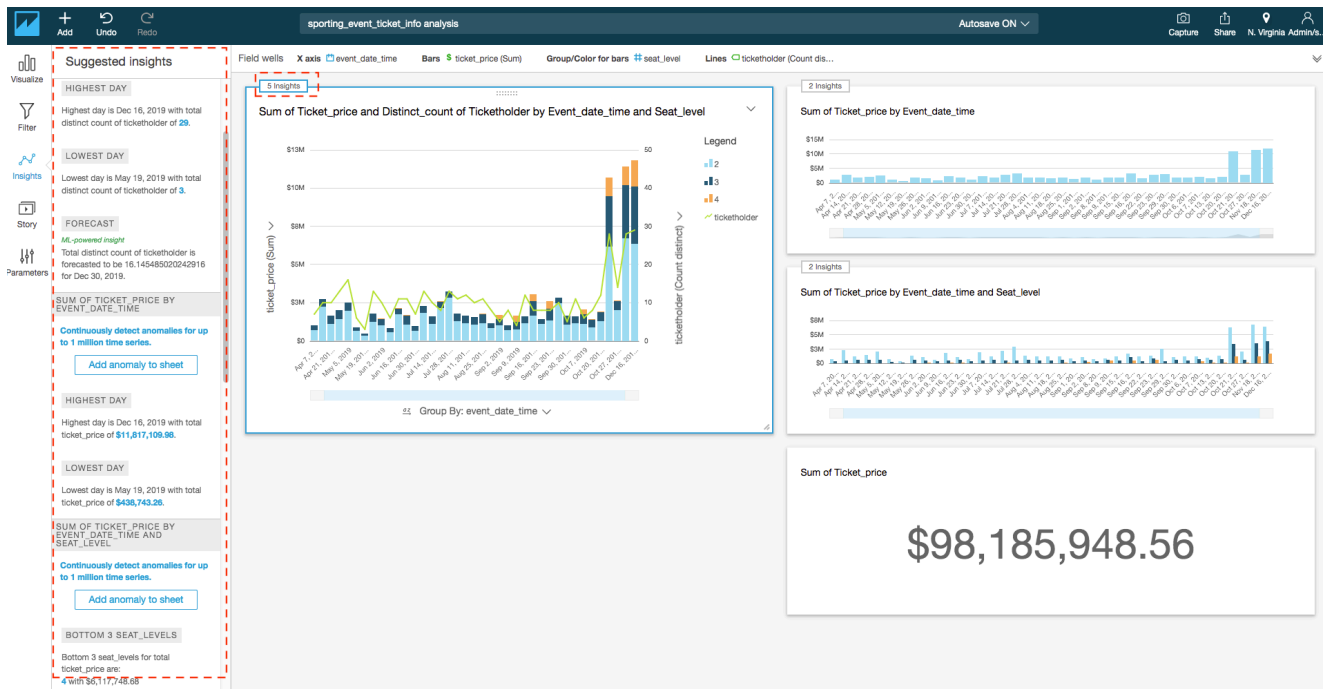


Let's build on this one step further by changing the chart type to "Clustered bar combo chart" and adding in the ticketholder for the Lines.

5. In the Visual types area, choose the Clustered bar combo chart icon.
6. In the Fields list, click and drag the **ticketholder** field to the Lines box in the Field wells pane.
7. In the Field wells pane, click the Lines box and choose **Count Distinct** for Aggregate. You can then see the y-axis update on the right-hand side.



8. Click on insight icon on top of each chart and explore insight information in right hand pan in simple English.

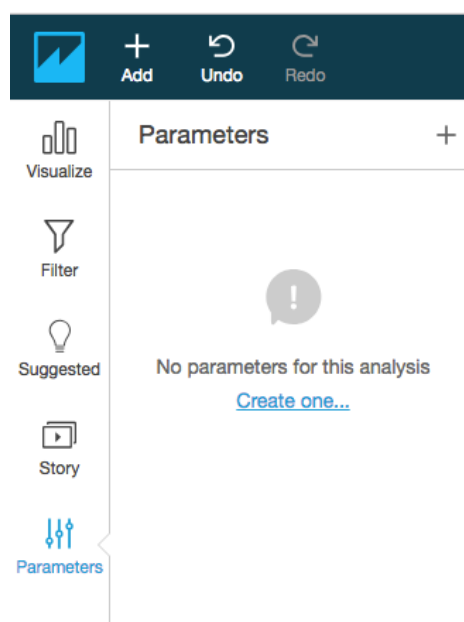


Feel free to experiment with other chart types and different fields to get a sense of the data.

Create QuickSight Parameters

In the next section we are going to create some parameters with controls for the dashboard, then assign these to a filter for all the visuals.

1. In the left navigation menu, select **Parameters**.



2. Click **Create one** to create a new parameter with a Name.
3. For Name, type **EventFrom**.
4. For Data type, choose **Datetime**.
5. For Default value, select the value from calendar as start date available in your graph for event_date_time. For example, **2018-01-01 00:00**.
6. Click **Create**, and then close the Parameter Added dialog box.

Create new parameter

Use parameters to dynamically control values in your fields, filters and sheet

Name

EventFrom

Data type (Not alterable after creation)

Datetime

Default value

2018-01-01 00:00

Set a dynamic default

Cancel Create

7. Create another parameter with the following attributes:
 - a. Name: EventTo
 - b. Data type: Datetime
 - c. For Default value, select the value from calendar as end date available in your graph for event_date_time. For example, 2019-01-01 00:00

Create new parameter

Use parameters to dynamically control values in your fields, filters and sheet

Name

EventTo

Data type (Not alterable after creation)

Datetime

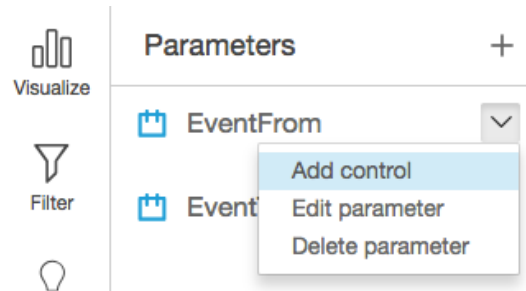
Default value

2019-01-01 00:00

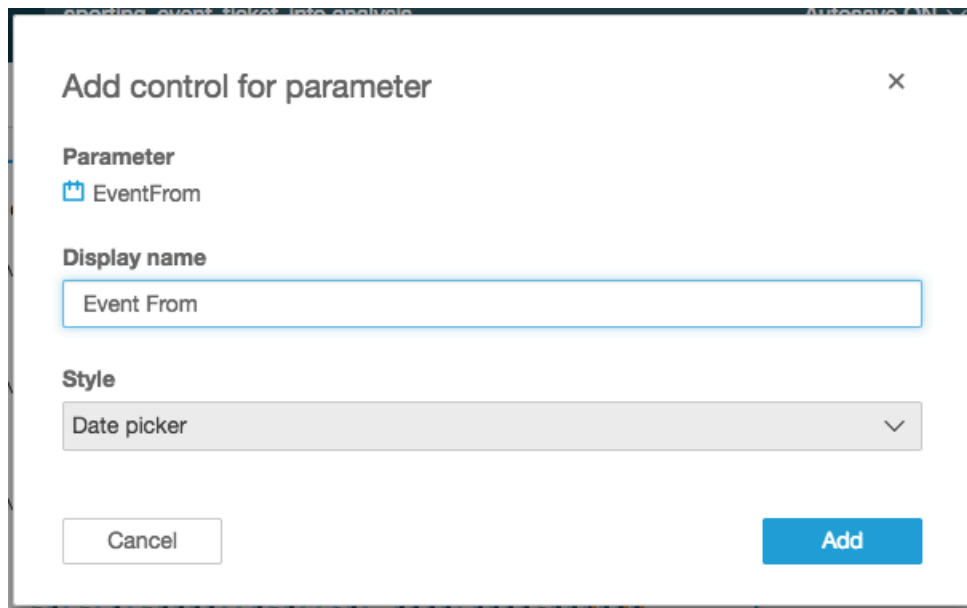
Set a dynamic default

Cancel Create

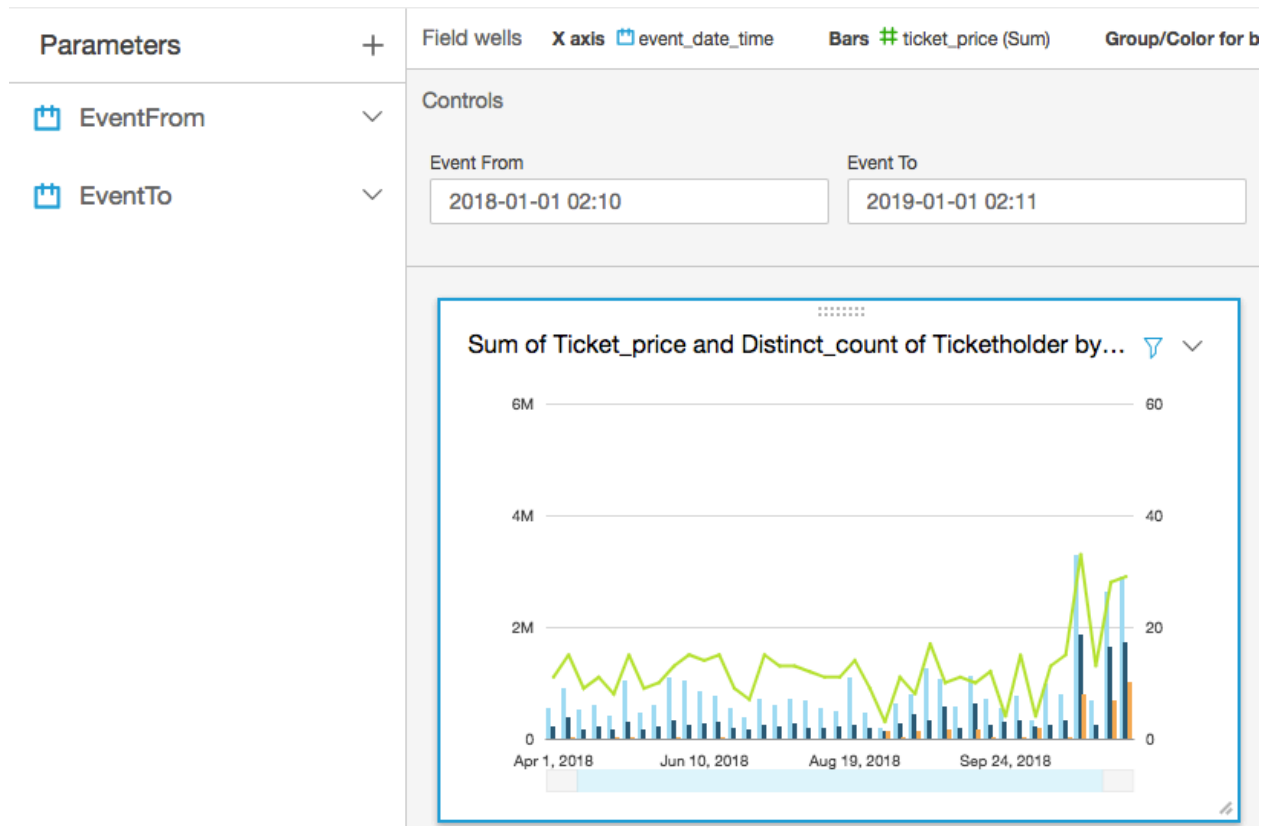
8. Click **Create**.
9. In the Parameter Added dialog box, click **Filter** and then click **Close**.
10. Click the drop-down menu for the EventFrom parameter and choose **Add control**.



11. For Display name, specify **Event From** and click **Add**.



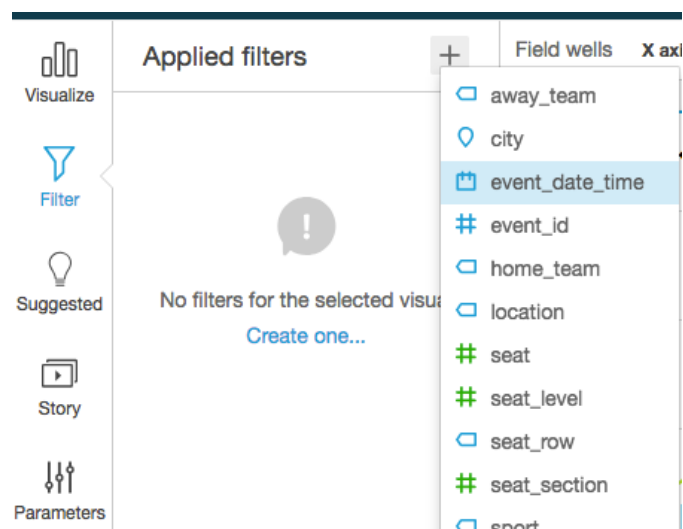
12. Repeat the process to add a control for **EventTo** with display name **Event To**.
- You should now be able to see and expand the Controls section above the chart.



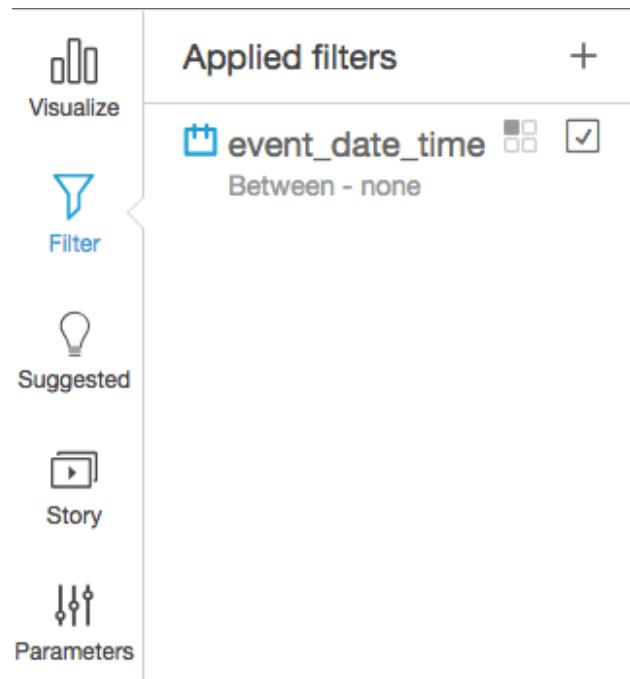
Create a QuickSight Filter

To complete the process, we will wire up a filter to these controls for all visuals.

1. In the left navigation menu, choose Filter.
2. Click the plus icon (+) to add a filter for the field "event_date_time".



3. Click this filter to edit the properties.



4. Choose to make this filter apply to **All visuals**.
5. For Filter type, choose **Time range** and **Between**.
6. Select option Use **Parameter**.
7. For Start date parameter, choose **EventFrom**.
8. For **End date parameter**, choose **EventTo**.
9. Click **Apply**.

Visualize

Filter

Suggested

Story

Parameters

Edit filter

All visuals

event_date_time
Between - none

Filter type
Time range

Between

☒ Use parameters

Start date parameter
EventFrom

End date parameter
EventTo

OR

Add filter condition

Note: Filters in a group can be either aggregated or non-aggregated, but not both.
[Learn more](#)

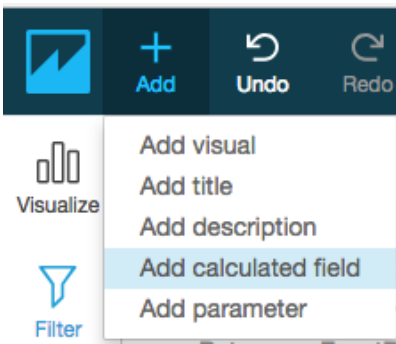
Apply

Close Delete filter

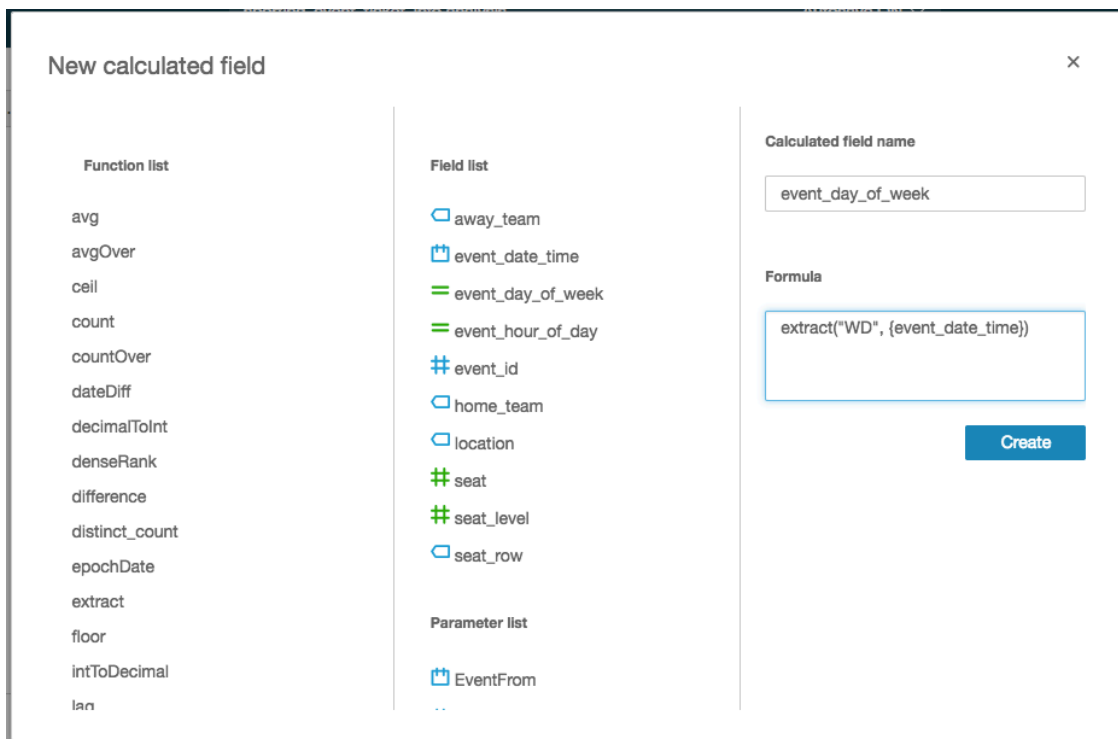
Add Calculated Fields

In the next section, You will learn, how to add calculated fields for "day of week" and "hour of day" to your dataset and a new scatter plot for these two dependent variables.

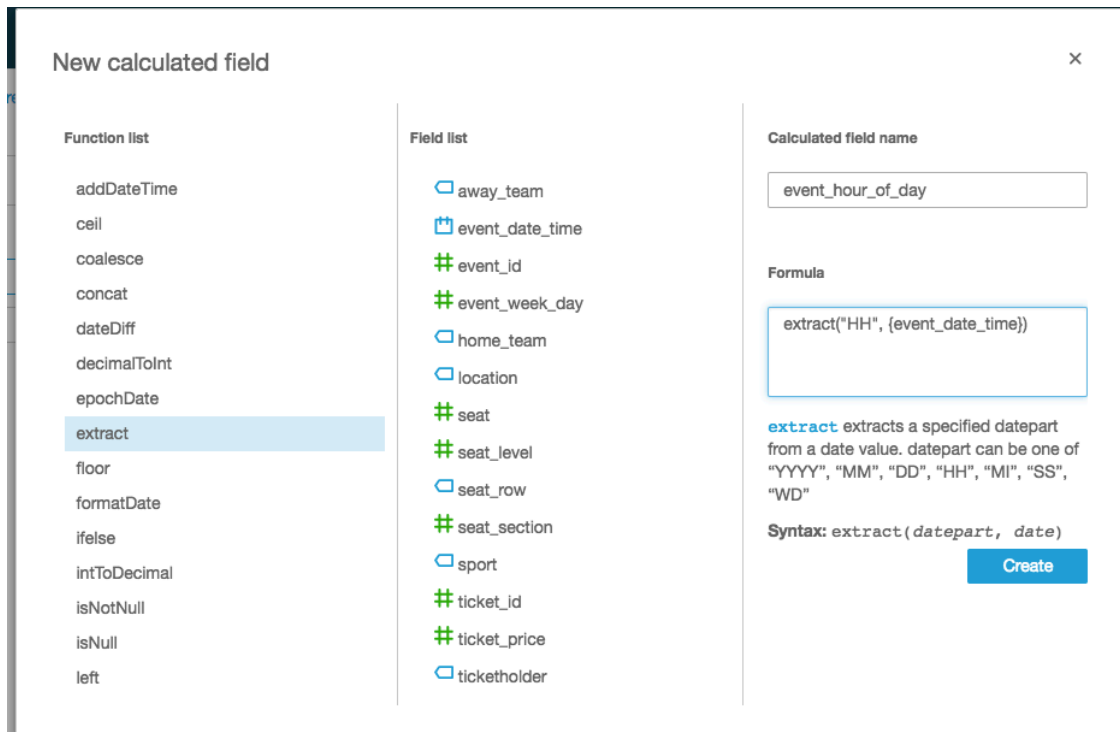
1. Click the Add button on the top left and select **Add a calculated field**.



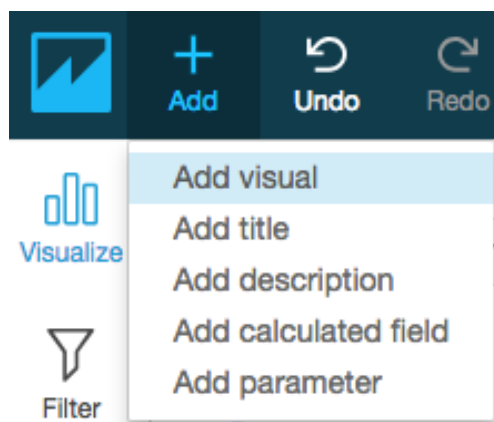
2. For **Calculated field name** type "event_day_of_week".
3. For Formula, type `extract("WD", {event_date_time})`
 Note: extract returns a specified portion of a date value. Requesting a time-related portion of a date that doesn't contain time information returns 0. WD: This returns the day of the week as an integer, with Sunday as 1.
4. Click **Create**.



5. Add another calculated field with the following attributes:
 - a. Calculated field name: "event_hour_of_day"
 - b. Formula: `extract("HH", {event_date_time})`
 Note: HH: This returns the hour portion of the date.

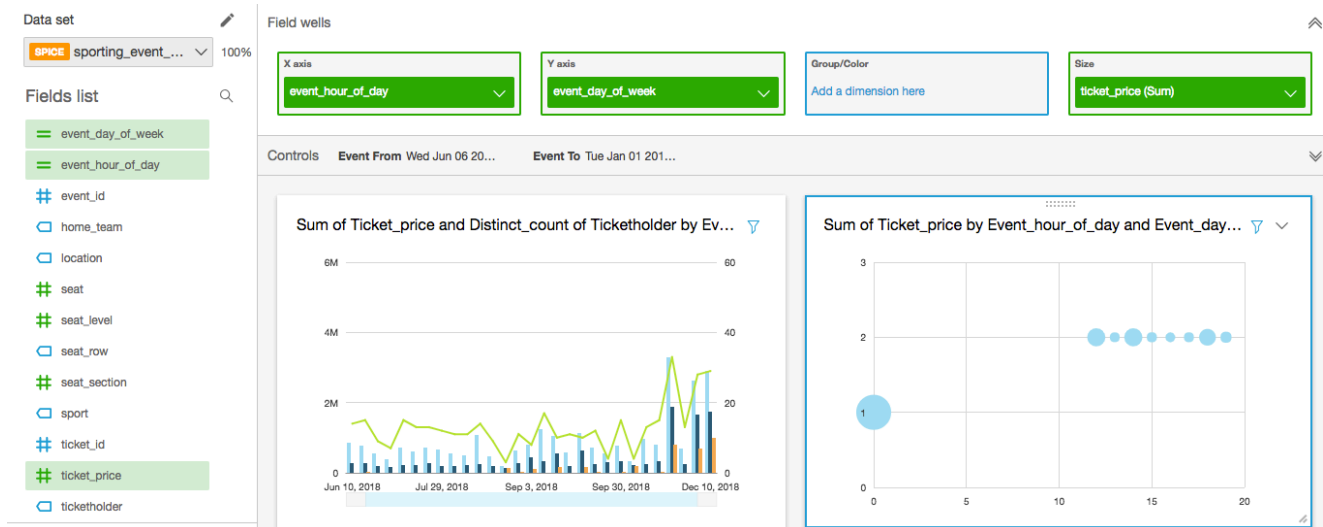


6. Click Add button in the top left and choose **Add visual**.

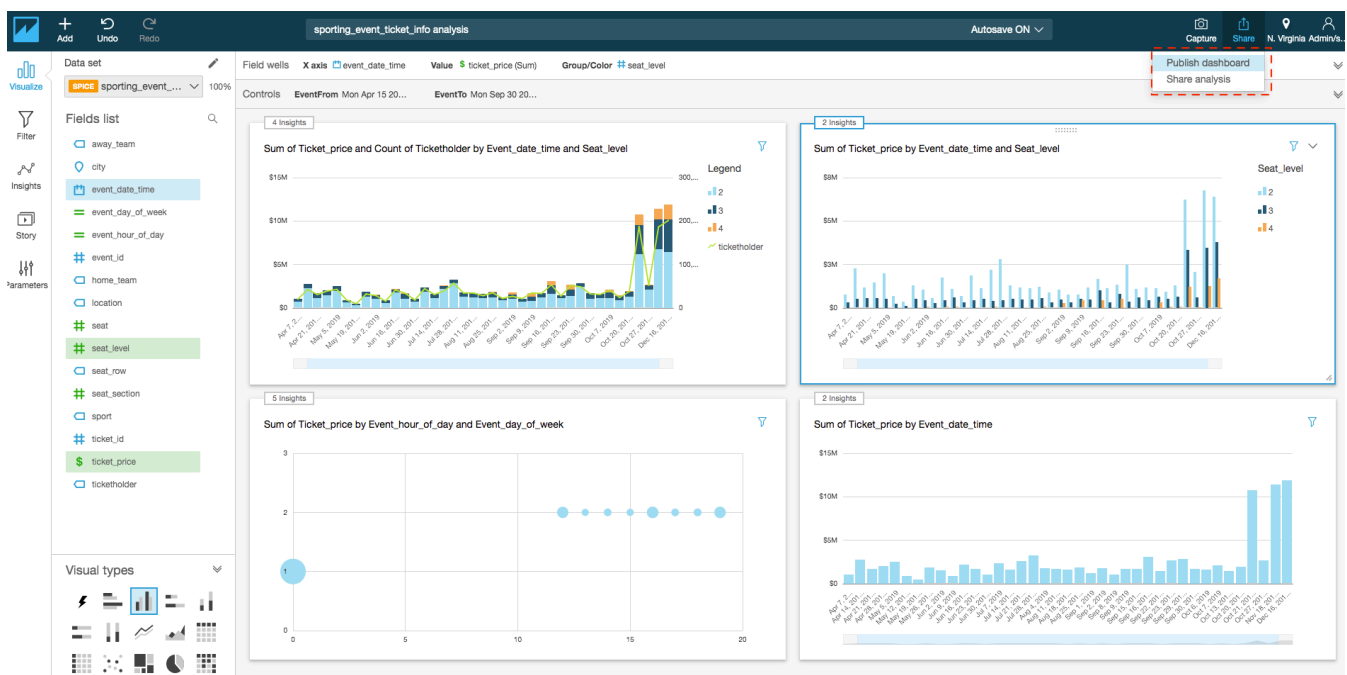


7. For field type, select the scatter plot.

8. In the Fields list, select and drag the following attributes to the Field wells pane to set the graph attributes:
- X-axis: "event_hour_of_day"
 - Y-axis: "event_day_of_week"
 - Size: "ticket_price"



Since now you have completed your dashboard then you can publish it by clicking on top right corner of screen.



A *dashboard* is a read-only snapshot of an analysis that you can share with other Amazon QuickSight users for reporting purposes. In Dashboard other users can still play with visuals and data but that will not modify dataset.

You can share an analysis with one or more other users with whom you want to collaborate on creating visuals. Analysis provide other uses to write and modify data set.

Amazon QuickSight ML-Insights (Optional)

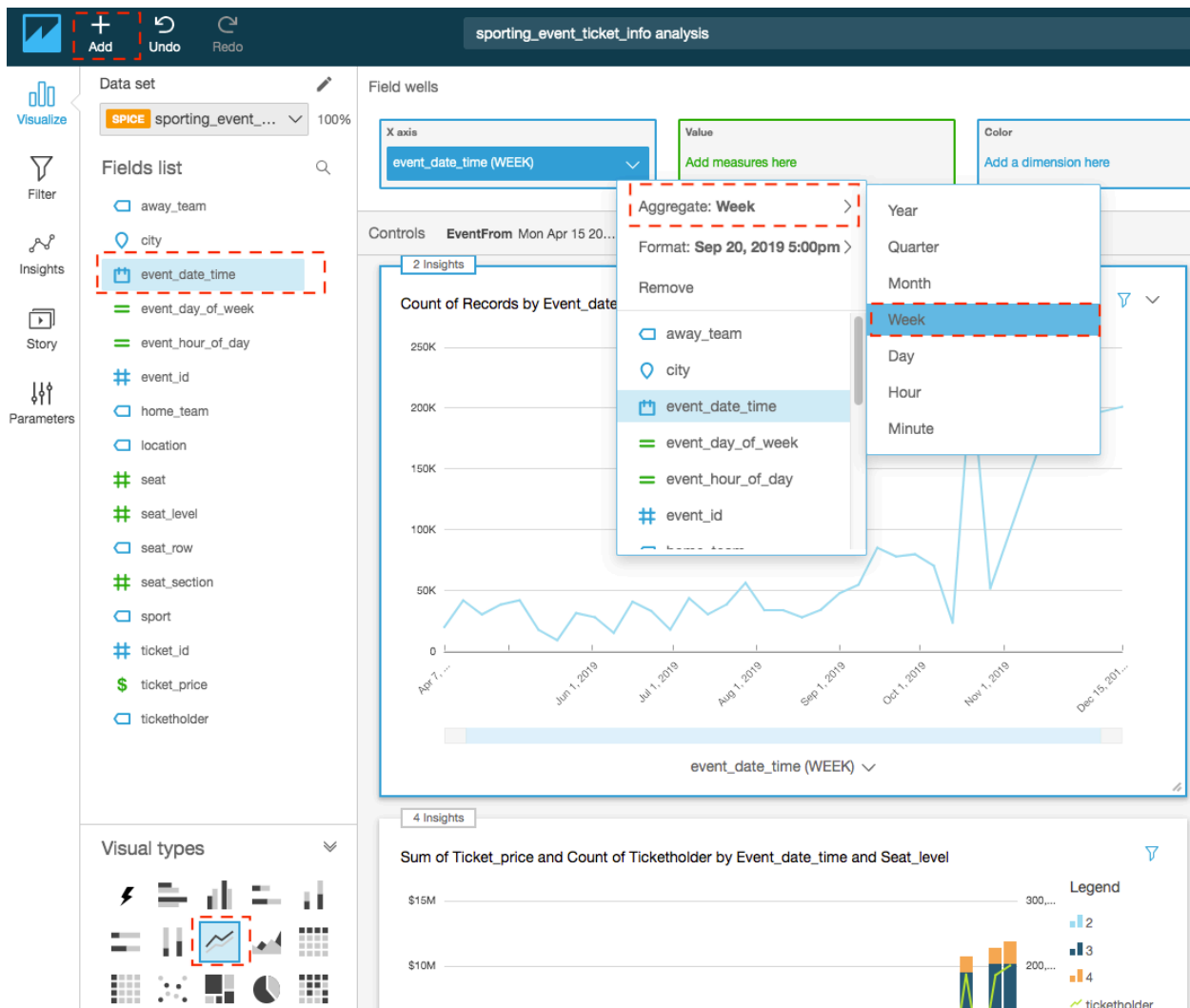
With Amazon QuickSight, you can add Machine Learning capabilities to your visuals, easily, with one click action. There are 3 types of Machine Learning Insights

- Narrative
- Anomaly Detection
- Forecasting

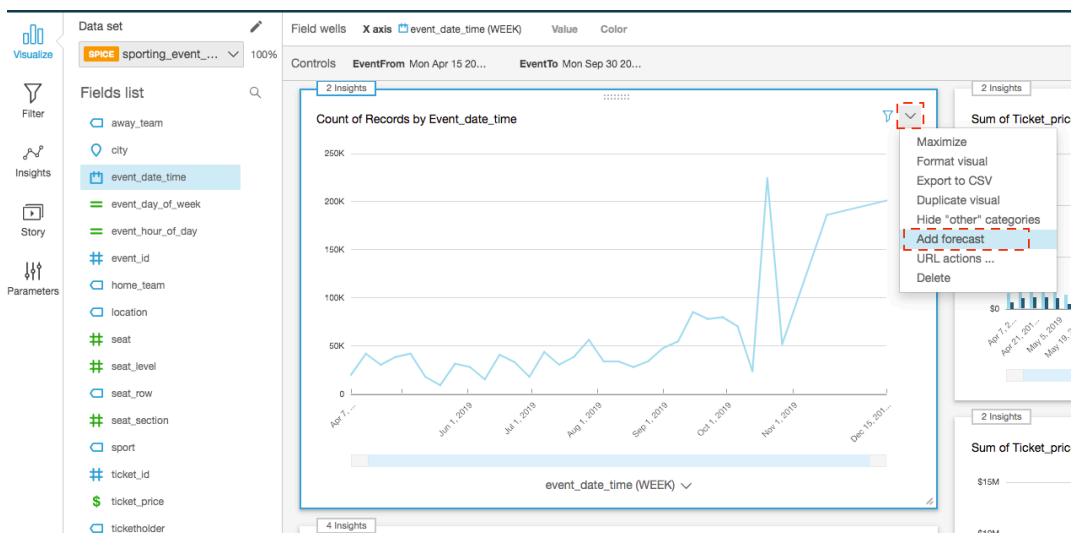
ML-Insights is only available to enterprise version of QuickSight. You will need to upgrade to Enterprise Edition before you start with the task. To upgrade your Amazon QuickSight Subscription from Standard Edition to Enterprise Edition please follow this guide <https://docs.aws.amazon.com/quicksight/latest/user/upgrading-subscription.html>

Let's see how we can add a bit of forecasting in our dashboard. Forecasting works with timeseries, which is better represented with a line graph. Let's first create a line graph.

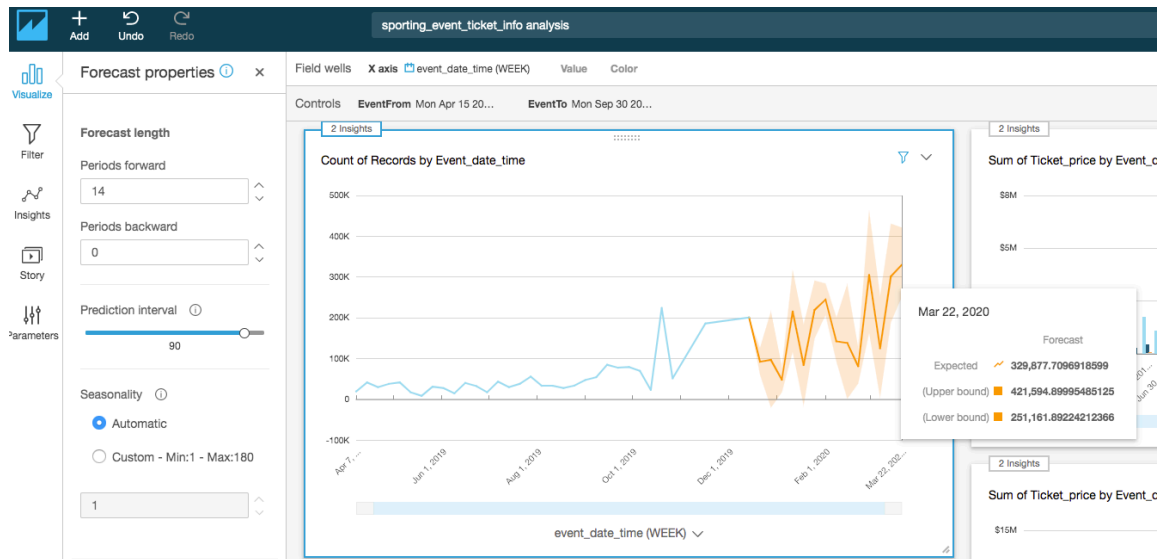
1. Click **add Visual at top left corner of screen**, and select **Line Chart** and add the **event_date_time** as the x-axis and **aggregate by week**. As shown in below screenshot



2. Add forecasting to the visual. To do that, click on the drop-down arrow on the top right corner of the visual, and then click **Add forecast**.



The visual will add forecast , you can hover over and explore forecasted data as shown below:



Feel free to explore with the properties of the forecast algorithm.