Form Submissions

Data for submitted forms is stored in *runtime-form-data* collection. For each form submission, a document is inserted in the collection.

Forms submitted on a specific date

To fetch documents created on a particular date, for example, following query outputs the forms' data submitted on 3rd October, 2022:

```
db["runtime-form-data"].find({
    dateCreated: {
        $gte: ISODate('2022-10-03'), // YYYY-MM-DD
        $lt: ISODate('2022-10-04') //defaults to 2022-10-04 00:00
    }
}); //output - records
```

Forms submitted on a specific date

To fetch documents created on a particular date, for example, following query outputs the forms' data submitted on 3rd October, 2022:

```
db["runtime-form-data"].find({
    dateCreated: {
        $gte: ISODate('2022-10-03'), // YYYY-MM-DD
        $lt: ISODate('2022-10-04') //defaults to 2022-10-04 00:00
    }
}); //output - records
```

Alternate query to the above query in which you can define date format too:

```
db["runtime-form-data"].aggregate([
    {
        $addFields: {
            createdOnDate: {
                $dateToString: {
                    format: '%d-%m-%Y',
                    date: '$dateCreated'
                }
            }
        }
    }, {
        $match: {
            createdOnDate: '03-10-2022'
        }
    }
]); //output - records
```

Count - number of forms submitted on a specific date

we can modify above queries as following:

```
db["runtime-form-data"].countDocuments({
    dateCreated: {
        $gte: ISODate('2022-10-03'), // YYYY-MM-DD
        $lt: ISODate('2022-10-04')
    }
}); //Output - 17
```

```
db["runtime-form-data"].aggregate
([
        {
            $addFields: {
                createdOnDate: {
                    $dateToString: {
                        format: '%d-%m-%Y',
                        date: '$dateCreated'
                    }
                }
            }
        }, {
            $match: {
                createdOnDate: '03-10-2022'
            }
        }, {
            $count: "Forms Submitted"
        }
    ]); //Output - { 'Forms Submitted': 17 }
```

Forms submitted for a range of dates

To get documents created from 3rd October, 2022 till 15th October 2022, including both dates.

```
db["runtime-form-data"].find({
    dateCreated: {
        $gte: ISODate('2022-10-03'), // yyyy-mm-ddThh:mm:ss.<milliseconds>Z defaults as 2022-10-03T
        $lte: ISODate('2022-10-15T23:59:59.999Z') // or ISODate('2022-10-15 23:59:59.999')
    }
}); //Output - records
```

Count - number of forms submitted for a range of dates

Above filter can be used in countDocuments():

```
db["runtime-form-data"].countDocuments({
    dateCreated: {
        $gte: ISODate('2022-10-03'), // start date
        $lte: ISODate('2022-10-15T23:59:59.999Z') // end Date
    }
}); //output: 97
```

Form submission counts group by form submission type

This query is suitable when we want form type wise individual form count

```
db["runtime-form-data"].aggregate
([
    {$match:{dateCreated:{$gte:ISODate('2023-01-04'),$lte:ISODate('2023-01-04T23:59:59.999Z')}}},
```

```
    {$group :{_id:{Form:'$formDefinitionReference.uuid'},Total:{$sum:1}}}
]);

/* Output: { _id: 'RB_UnableRegOnlineBankingBusCust', recordCount: 67 }
           { _id: 'RB_BiometricRemoval', recordCount: 11 }
           { _id: 'RB_ATMClaim', recordCount: 11 }
           { _id: 'RB_DebitCardAccTakeover', recordCount: 8 } */
```

Form submission counts group by submission date

This query is suitable when we want date wise total form count

```
db["runtime-form-data"].aggregate
([
  {$match:{dateCreated:{$gte:ISODate('2022-12-01'),$lte:ISODate('2022-12-19T23:59:59.999Z')}}},
  {$group:{ _id:{Date:{$dateToString:{format: "%Y-%m-%d", date: "$dateCreated"}}}, Total:{ $sum: 1
  {$sort: {_id:1}}
]);

/* Output: { _id: { Date: '2022-12-01' }, Total: 3 }
           { _id: { Date: '2022-12-02' }, Total: 7 }
           { _id: { Date: '2022-12-04' }, Total: 1 }
           { _id: { Date: '2022-12-05' }, Total: 4 } */
```

Form submission counts group by form type and submission date

This query is suitable when we want date wise individual form count

```
db["runtime-form-data"].aggregate
([
  {$match:{dateCreated:{$gte:ISODate('2022-12-04'),$lte:ISODate('2022-12-04T23:59:59.999Z')}}},
  {$group:{ _id:{Date:{$dateToString:{format:"%Y-%m-%d",date:"$dateCreated" }},Form:'$formDefiniti
  {$sort: {_id:1}}
]);

/* Output: { _id: { Date: '2022-12-04', Form: 'RB_BiometricRemoval' },Total: 1 }
           { _id: { Date: '2022-12-05', Form: 'RB_BiometricRemoval' },Total: 1 }
           { _id: { Date: '2022-12-05', Form: 'RB_DebitCardAccTakeover' },Total: 3 }
           { _id: { Date: '2022-12-06', Form: 'RB_ATMClaim' }, Total: 1 }
           { _id: { Date: '2022-12-06', Form: 'RB_DebitCardAccTakeover' },Total: 1 }  */
```

Form submission counts for a month group by hour

This query is suitable when we want hour wise total form count for a month

```
db["runtime-form-data"].aggregate([{
    $match: { dateCreated: {
            $gte: ISODate('2023-01-01T00:00:00.000Z'),
            $lt: ISODate('2023-02-01T00:00:00.000Z')   }
            }
},
    { $project: { hour: { $hour: '$dateCreated' }, _id: 1 }
},
    { $group: { _id: '$hour', count: { $sum: 1 } }
},
    { $sort: { _id: 1    }
```

```
}]);

/* Output: { _id: 0, count: 7 }
{ _id: 1, count: 4 }
{ _id: 4, count: 1 }
{ _id: 5, count: 3 }
{ _id: 6, count: 4 } */
```

Form submission counts for a month group by day

This query is suitable when we want day wise total form count

```
db["runtime-form-data"].aggregate ([
    { $match: { dateCreated: { $gte: ISODate('2023-01-01T00:00:00.000Z'),
            $lte: ISODate('2023-01-31T23:59:59.999Z') }
    }
},
     { $group: { _id: { month: { $month: '$dateCreated' }, Day: { $dayOfMonth: '$dateCreated' } },
        count: { $sum: 1 } }
},
    { $project: { _id: 0, Day: '$_id.Day', count: 1    }
},
    { $sort: { Day: 1 }
}]);

/* Output:  { count: 1, Day: 2 }
            { count: 2, Day: 3 }
            { count: 70, Day: 4 }
            { count: 71, Day: 5 }
            { count: 70, Day: 6 } */
```

To get Count of Form Submission on Day wise for a week

This Query will give you the count of form submission on day wise.

```
db["runtime-form-data"].aggregate(

[{ $match: { dateCreated: {

            $gte: ISODate('2023-07-31'),

            $lte: ISODate('2023-08-06T23:59:59.999Z')

        } }

}, {

    $group: {

        _id: { $dayOfWeek: '$dateCreated'    },

        count: {    $sum: 1 }

```

```
        }

}, {

    $project: {  day :{  $switch : { branches:[

{case : { $eq : ['$_id',1] }, then: "Sunday"},

{case : { $eq : ['$_id',2] }, then: "Monday"},

{case : { $eq : ['$_id',3] }, then: "Tuesday"},

{case : { $eq : ['$_id',4] }, then: "Wednesday"},

{case : { $eq : ['$_id',5] }, then: "Thursday"},

{case : { $eq : ['$_id',6] }, then: "Friday"},

{case : { $eq : ['$_id',7] }, then: "Saturday"},

]  }

},

count:1, _id:0}

}, { $sort: {   _id: 1  }  }]  );
```

Events

When a form is submitted, a series of events are triggered. The events/documents(records) are stored in *event-store* collection. For a complete flow, starting with form submission to sending mails, there should be 5 documents per form submission(per submission uuid) in event-store.

Successful submissions on a specific date

> Following query filters events created on a specific date, counts them for each uuid and outputs only the uuid for which both the internal and external emails were sent successfully.

```
    db["event-store"].aggregate([
        {
            $addFields: {
                createdOnDate: {
                    $dateToString: {
                        format: '%d-%m-%Y',
                        date: '$header.created'
                    }
                }
            }
        }, {
            $match: {
                'body.eventClass': 'SEND_SUCCESS',
                createdOnDate: '03-10-2022'
            }
        }, {
            $group: {
```

```
                _id: '$body.uuid',
                success_mails_count: {
                    $count: {}
                }
            }
        }, {
            $match: {
                success_mails_count: 2
            }
        }
    ]);


/*output: { _id: '00c6bb7b-af47-4f78-a2a5-b41af4dbc593', success_mails_count: 2 }
          { _id: '3ebbc5f0-398d-40e1-89a6-903e805e489p', success_mails_count: 2 }
          { _id: '3ebbc5f0-398d-40e1-89a6-9037355e443c', success_mails_count: 2 }
          { _id: '3ebbc5f0-398d-40e1-89a6-90e805e4442f', success_mails_count: 2 } */
```

Email sending failed for forms submitted on a specific date

For a specific date, following query outputs the uuids(submission ids) for which internal and/or external emails were sent but the send failed for internal or external or both. For example, in the below query, we are looking if mails were sent on 29th August, 2022 for the forms submitted and if yes, did they fail? It will list the failed uuids. The example output tells us that for uuid *c4549e1a-9011-4e22-9abe-e7b57f253056,* both internal and external email sending failed.

```
    db["event-store"].find({
        "body.eventClass": "SEND_FAILURE",
        "header.created": {
            $gte: ISODate('2022-08-29'),
            $lt: ISODate('2022-08-30')
        }
    }, {
            "body.uuid": 1,
            "_id": 0
        }
    );

/* Output:
 { body: { uuid: 'c4549e1a-9011-4e22-9abe-e7b57f253056' } }
 { body: { uuid: 'c4549e1a-9011-4e22-9abe-e7b57f253056' } }
{ body: { uuid: 'c4549e1a-9011-4e22-9abe-e7b57f253056' } }
 { body: { uuid: 'c4549e1a-9011-4e22-9abe-e7b57f253056' } } */
```

All the submissions irr-respective of Email success & failure

LIFECYCLE_EVENT: ['SUBMITTED', 'SEND_SUCCESS'] means Email Triggered Successfully

LIFECYCLE_EVENT: ['SUBMITTED'] means form submitted but Email is not Triggered

This query is suitable when we want all records related to email trigger

```
db["runtime-form-data"].aggregate([
{$match:{dateCreated:{$gte:ISODate('2023-01-03'),$lte:ISODate('2023-01-03T23:59:59.999Z')}}},
{ $lookup:
        {
```

```
          from: "event-store",
          localField: "uuid",
          foreignField: "references.uuid",
          as: "event"
        }
},
{$group:{_id:{Date:{$dateToString:{format:"%Y-%m-%d",date:"$dateCreated"}},uuid:'$uuid',Form:'$for
{$sort: {_id:1}}
]);

/* Output:

{ _id:
   { Date: '2022-12-18',
     uuid: 'a955d963-2d0b-4c18-81ab-abf460ae93fb',
     Form: 'RB_DebitCardAccTakeover',
     LIFECYCLE_EVENT: [ 'SUBMITTED', 'SEND_SUCCESS' ] } }
{ _id:
   { Date: '2022-12-18',
     uuid: 'c00428ce-e42f-4c1b-b364-92b79f06dbd6',
     Form: 'RB_DebitCardAccTakeover',
     LIFECYCLE_EVENT: [ 'SUBMITTED' ] } }
{ _id:
   { Date: '2022-12-18',
     uuid: 'edd89433-992e-4dc5-8a78-f0e26a902f55',
     Form: 'RB_DebitCardAccTakeover',
     LIFECYCLE_EVENT: [ 'SUBMITTED', 'SEND_SUCCESS' ] } } */
```

FORM SUCCESS AND FAILURE COUNT

LIFECYCLE_EVENT: ['SUBMITTED', 'SEND_SUCCESS'] means Email Triggered Successfully

LIFECYCLE_EVENT: ['SUBMITTED'] means form submitted but Email is not Triggered

This query is suitable when we want the failed Email triggered count and successful Email triggered count

```
db["runtime-form-data"].aggregate([
{$match:{dateCreated:{$gte:ISODate('2022-12-20'),$lte:ISODate('2022-12-26T23:59:59.999Z')}}},
{ $lookup:
        {
          from: "event-store",
          localField: "uuid",
          foreignField: "references.uuid",
          as: "event"
        }
},
{$group :{_id:{LIFECYCLE_EVENT:'$event.body.eventClass'},Total:{$count:{}}}}
]);

/*Output: { _id: { LIFECYCLE_EVENT: [ 'SUBMITTED' ] }, Total: 2 }
          { _id: { LIFECYCLE_EVENT: [ 'SUBMITTED', 'SEND_SUCCESS' ] },Total: 19 }
```

Only Success Record

This query is suitable when we want the successful Email triggered Record

```
db["runtime-form-data"].aggregate([
{$match:{dateCreated:{$gte:ISODate('2022-12-20'),$lte:ISODate('2022-12-26T23:59:59.999Z')}}},
{ $lookup:
       {
          from: "event-store",
          localField: "uuid",
          foreignField: "references.uuid",
          as: "event"
       }
},
{ $match:{"event.body.eventClass":"SEND_SUCCESS" }},
{$group:{_id:{Date:{$dateToString:{format:"%Y-%m-%d",date:"$dateCreated"}},uuid:'$uuid',Form:'$for
{$sort: {_id:1}}
]);

/* Output:

{ _id:
   { Date: '2022-12-18',
     uuid: 'a955d963-2d0b-4c18-81ab-abf460ae93fb',
     Form: 'RB_DebitCardAccTakeover',
     LIFECYCLE_EVENT: [ 'SUBMITTED', 'SEND_SUCCESS' ] } }
{ _id:
   { Date: '2022-12-18',
     uuid: 'edd89433-992e-4dc5-8a78-f0e26a902f55',
     Form: 'RB_DebitCardAccTakeover',
     LIFECYCLE_EVENT: [ 'SUBMITTED', 'SEND_SUCCESS' ] } } */
```

Only Failure Record

This query is suitable when we want the failed Email triggered Record

```
db["runtime-form-data"].aggregate([
{$match:{dateCreated:{$gte:ISODate('2023-01-04'),$lte:ISODate('2023-01-04T23:59:59.999Z')}}},
{ $lookup:
       {
          from: "event-store",
          localField: "uuid",
          foreignField: "references.uuid",
          as: "event"
       }
},
{$match:{$and:[{"event.body.eventClass":'SUBMITTED' },{"event.body.eventClass":{$ne:'SEND_SUCCESS'
{$group:{_id:{Date:{$dateToString:{format:"%Y-%m-%d-T%H:%M:%S",date:"$dateCreated"}},uuid:'$uuid',
{$sort: {_id:1}}
]);

/* Output:

{ _id:
   { Date: '2023-01-16-T08:51:30',
     uuid: 'fc34d4a1-8a16-4c2f-9b95-6e4ff5d45b0c',
     Form: 'RB_CustomerSupportSpecialist',
     Event_Failure: [ 'SUBMITTED', 'SEND_FAILURE' ] } }
{ _id:
```

```
  { Date: '2023-01-16-T14:45:04',
    uuid: 'd5bd4e57-9f0c-4aba-bbea-750b33f16839',
    Form: 'RB_SkillsCoachingObservation',
    Event_Failure: [ 'SUBMITTED', 'SEND_FAILURE' ] } } */
```

Mapping of form name & Kafka topic.

```
db["forms-rule-template-data"].aggregate([
{$match:{$and:[{ruleTemplateName:'kafka-event-partitioning'},{"rows":{"$not":{"$size":0}}}]}},
{$group :{_id:{Form:'$formDefinitionReference.uuid', KafkaTopic:{$last:{$arrayElemAt:['$rows',-1]}}
]);
```

Number of messages published from one form for specific time period.

```
db["event-store"].aggregate([
{$match:{$and:[{"header.schemaId":'KAFKA_PUBLISH_EVENT'},{"body.eventClass":'SENT'},
{"references.uuid":'RB_BDN'},
{"header.created":{$gte:ISODate('2023-08-01'),$lte:ISODate('2023-08-31T23:59:59.999Z')}}
]}},
{ $unwind: "$references" },
{$match:{"references.name":{$ne:'EFORM'}}},
{$group :{_id:{Form:'$references.uuid',Date:{$dateToString:{format:"%Y-%m-%d",date:"$header.created
]);
```

Number of messages published on Kafka on a particular day.

```
db["event-store"].aggregate([
{$match:{$and:[{"header.schemaId":'KAFKA_PUBLISH_EVENT'},{"body.eventClass":'SENT'},
{"header.created":{$gte:ISODate('2023-08-01'),$lte:ISODate('2023-08-31T23:59:59.999Z')}}
]}},
{ $unwind: "$references" },
{$match:{"references.name":{$ne:'EFORM'}}},
{$group :{_id:{Form:'$references.uuid',Date:{$dateToString:{format:"%Y-%m-%d",date:"$header.created
]);
```

Number of messages published on Kafka on a particular day on a specific Kafka topic

```
db["event-store"].aggregate([
{$match:{$and:[{"header.schemaId":'KAFKA_PUBLISH_EVENT'},{"body.eventClass":'SENT'},
{"body.topic":'eforms.rb.uat.events.formlifecycle.dbn'},
{"header.created":{$gte:ISODate('2023-08-01'),$lte:ISODate('2023-08-31T23:59:59.999Z')}}
]}},
{ $unwind: "$references" },
{$match:{"references.name":{$ne:'EFORM'}}},
{$group :{_id:{Form:'$references.uuid',topic:'$body.topic',Date:{$dateToString:{format:"%Y-%m-%d",c
]);
```