FINISHED

Story

Acme Inc started business in 2013 and immediately started onboarding members.

Enrollment rules:

- You must be 18 years or older to have an account at Acme Inc.
- You must provide valid identifiers (email, zip code, phone number) during enrollment.
- Members are never removed, just cancelled.

Issue

A developer wrote a script that accidentally messed up some data and there have been a few bugs over the years that could have caused issues.

Task

The goal is to use Spark to identify the data that is out of these bounds.

Schema

```
create table members (
id int not null auto_increment,
first_name varchar(255) not null,
last_name varchar(255) not null,
email varchar(255) not null,
phone int(10) not null,
status enum('active', 'cancelled') not null,
zip5 int(5) not null,
created_at datetime,
updated_at datetime,
birth_date date not null)
```

Took 0 sec. Last updated by chime at March 05 2019, 4:54:48 PM.

Define file location

READY

```
val workingDir = "file:///opt/data"
val sourceFilePath = s"${workingDir}/data.json"
val jsonAnomaliesPath = s"${workingDir}/anomalies"
val parquetPath = s"${workingDir}/parquet"
val validOutPath = s"${workingDir}/validoutput"
```

```
workingDir: String = file:///opt/data
sourceFilePath: String = file:///opt/data/data.json
jsonAnomaliesPath: String = file:///opt/data/anomalies
parquetPath: String = file:///opt/data/parquet
validOutPath: String = file:///opt/data/validoutput
```

Display schema structure

READY

```
spark.read.json(sourceFilePath).printSchema
// All columns are String since the data is defined as string in json file. It does not match
```

```
root
 |-- birth_date: string (nullable = true)
 |-- created_at: string (nullable = true)
 |-- email: string (nullable = true)
 |-- first_name: string (nullable = true)
 |-- id: string (nullable = true)
 |-- last_name: string (nullable = true)
 |-- phone: string (nullable = true)
 |-- status: string (nullable = true)
 |-- updated_at: string (nullable = true)
 |-- zip4: string (nullable = true)
```

Create a members table from the given file

READY

```
sql(s"CREATE TABLE IF NOT EXISTS members USING org.apache.spark.sql.json OPTIONS (path '$sour
```

```
res24: org.apache.spark.sql.DataFrame = []
```

Create validation functions

READY

```
// validate emails
def isValidEmail(email : String): Boolean = if("""^\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+

println(s"test isValidEmail function with different input options: ${isValidEmail("ethan@chim


// validate zip code - the strict 5 digit lenght is based on the mysql schema zip5 column len
def isValidZipCode(zipCode: String): Boolean = if("""^[0-9]{5}$""".r.findFirstIn(zipCode) == N

println(s"test isValidZipCode function with different input options: ${isValidZipCode("36066"

// Assuming that the database schema phone column length is 10 Int, this is for phone numbers
def isValidPhoneNumber(phone: String): Boolean = if("""^[0-9]{10}$""".r.findFirstIn(phone) ==

println(s"test isValidPhoneNumber function with different input options: ${ isValidPhoneNumbe
```

```
test isValidEmail function with different input options: List(true, true, false, false, false,
false, false)
test isValidZipCode function with different input options: List(true, false, false, true)
test isValidPhoneNumber function with different input options: List(false, true, false)
isValidEmail: (email: String)Boolean
isValidZipCode: (zipCode: String)Boolean
isValidPhoneNumber: (phone: String)Boolean
```

Register validation functions as User Defined Functions for SQL queries

READY

```
sqlContext.udf.register("is_valid_email", (email: String) => isValidEmail(email))
sqlContext.udf.register("is_valid_zip_code", (zip: String) => isValidZipCode(zip))
sqlContext.udf.register("is_valid_phone", (phone: String) => isValidPhoneNumber(phone))
```

```
res26: org.apache.spark.sql.expressions.UserDefinedFunction = UserDefinedFunction(<function1>,
BooleanType,Some(List(StringType)))
```

Data bounds validation

READY

```
sql("""SELECT
    |is_valid_email(email) as valid_email,
    |is_valid_zip_code(zip4) as valid_zip_code,
    |is_valid_phone(phone) as valid_phone,
    |if(datediff( to_date(created_at, 'yyyy-mm-dd'),to_date(birth_date, 'mm/dd/yyyy')) - 6570
    |count(*) record_count
    |   FROM members
    |   GROUP BY
    |valid_email,
    |valid_zip_code,
    |valid_phone,
    |valid_age""".stripMargin).show(120,220,false)

// display cube based bounds validation
sql("""SELECT
    |is_valid_email(email) as valid_email,
    |is_valid_zip_code(zip4) as valid_zip_code,
    |is_valid_phone(phone) as valid_phone,
    |if(datediff( to_date(created_at, 'yyyy-mm-dd'),to_date(birth_date, 'mm/dd/yyyy')) - 6570
    |count(*) record_count
    |   FROM members
    |   GROUP BY
    |valid_email,
    |valid_zip_code,
    |valid_phone,
    |valid_age WITH CUBE""".stripMargin).show(120,220,false)
```

```
+-----------+--------------+-----------+---------+------------+
|valid_email|valid_zip_code|valid_phone|valid_age|record_count|
+-----------+--------------+-----------+---------+------------+
|       true|         false|       true|     true|           3|
|       true|          true|       true|     true|        9350|
|      false|          true|       true|     true|           3|
|       true|          true|       true|    false|         607|
+-----------+--------------+-----------+---------+------------+


+-----------+--------------+-----------+---------+------------+
|valid_email|valid_zip_code|valid_phone|valid_age|record_count|
+-----------+--------------+-----------+---------+------------+
|       null|          true|       true|    false|         607|
|       true|         false|       true|     true|           3|
|       true|          null|       null|     null|        9960|
|       true|         false|       true|     null|           3|
|       true|          true|       true|    false|         607|
|      false|          true|       null|     null|           3|
```

Find duplicated records based on email,first name, last name

READY

```
sql("SELECT members.*,t1.dist_cnt FROM members JOIN (SELECT email,first_name, last_name,count
```

```
+----------+-------------------+-------------------------+----------+----+----------+------
----+---------+-------------------+-----+--------+
|birth_date|         created_at|                    email|first_name|  id| last_name|     p
hone|   status|         updated_at| zip4|dist_cnt|
+----------+-------------------+-------------------------+----------+----+----------+------
----+---------+-------------------+-----+--------+
| 5/18/1981|2017-11-14T10:55:23|  AbbigailLawyer@gmail.com|  Abbigail|7978|    Lawyer|212686
7491|   active|2019-01-14T00:37:34|46655|       2|
|12/12/1987|2013-04-20T08:39:29|  AbbigailLawyer@gmail.com|  Abbigail| 320|    Lawyer|262361
3090|   active|2018-09-05T02:39:06|29992|       2|
| 10/8/1976|2013-06-26T13:37:14|        AbbyBush@gmail.com|      Abby| 610|      Bush|128291
4404|cancelled|2018-12-25T15:23:54|24192|       2|
|  2/4/1979|2014-02-27T03:11:14|        AbbyBush@gmail.com|      Abby|1711|      Bush|282348
3323|cancelled|2018-09-17T14:29:55|25138|       2|
|12/30/1967|2017-04-06T22:25:44|     AbdielBliss@gmail.com|    Abdiel|6991|     Bliss|426326
4902|   active|2018-10-28T21:34:46|78383|       2|
|  8/9/1982|2017-08-23T17:19:50|     AbdielBliss@gmail.com|    Abdiel|7623|     Bliss|241459
0050|   active|2018-12-05T14:03:07|65364|       2|
```

Print members schema

READY

```
sql("describe formatted members").show
```

```
+--------------------+--------------------+-------+
|            col_name|           data_type|comment|
+--------------------+--------------------+-------+
|          birth_date|              string|   null|
|          created_at|              string|   null|
|               email|              string|   null|
|          first_name|              string|   null|
|                  id|              string|   null|
|           last_name|              string|   null|
|               phone|              string|   null|
|              status|              string|   null|
|          updated_at|              string|   null|
|                zip4|              string|   null|
|                    |                    |       |
|# Detailed Table ...|                    |       |
|            Database|             default|       |
|               Table|             members|       |
|        Created Time|Sat Feb 16 10:55:   |       |
```

Active/canceled records based on age validation

READY

```
// 18 year = 6570 days
// age threshold
sql("SELECT status,if(datediff( to_date(created_at, 'yyyy-mm-dd'),to_date(birth_date, 'mm/dd/y
```

```
+---------+---------+------------+
|   status|valid_age|record_count|
+---------+---------+------------+
|   active|     true|        5705|
|cancelled|     true|        3651|
|cancelled|    false|         236|
|   active|    false|         371|
+---------+---------+------------+
```

Final result set based on data bounds validation

READY

```
sql("""SELECT
         |members.id,
         |members.first_name,
         |members.last_name,
         |members.email,
         |members.phone,
         |members.status,
         |members.zip4,
         |members.created_at,
         |members.updated_at,
         |members.birth_date,
         |is_valid_email(members.email) as valid_email,
         |is_valid_zip_code(members.zip4) as valid_zip_code,
         |is_valid_phone(members.phone) as valid_phone,
         |if(datediff( to_date(members.created_at, 'yyyy-mm-dd'),to_date(members.birth_date,
       |FROM members
         |WHERE
         |!is valid email(members.email) or !is valid zip code(members.zip4) or !is valid phon
```

```
+---+----------+---------+------------------------+---------+---------+-----+-----------
--------+------------------+---------+----------+------------+----------+---------+
| id|first_name| last_name|                        email|     phone|   status| zip4|         cr
eated_at|         updated_at|birth_date|valid_email|valid_zip_code|valid_phone|valid_age|
+---+----------+---------+------------------------+---------+---------+-----+-----------
--------+------------------+---------+----------+------------+----------+---------+
|  9|   Jayvion|    Marion|   JayvionMarion@gmail.com|3240113946|   active|64132|2013-02-13T
11:12:45|2018-06-03T05:15:29| 2/13/1999|       true|           true|         true|    false|
| 17|   Casimir|   Hershel|  CasimirHershel@gmail.com|7316490247|   active|77632|2013-02-14T
11:19:46|2018-03-17T02:02:07| 2/13/1996|       true|           true|         true|    false|
| 20|      Taya|      Inga|        TayaInga@gmail.com|7350406982|cancelled|72879|2013-02-14T
19:32:48|2018-10-14T15:38:53| 2/12/1997|       true|           true|         true|    false|
| 25|     Lonie|    Elaina|     LonieElaina@gmail.com|2715734326|cancelled|24609|2013-02-16T
14:11:11|2018-07-28T02:45:54| 2/10/1997|       true|           true|         true|    false|
| 31|    Elmina|     Dolph|     ElminaDolph@gmail.com|4891431775|   active|38575|2013-02-18T
00:07:36|2018-12-08T03:13:43|  2/8/1997|       true|           true|         true|    false|
| 34|    Ephram|     Bjorn|     EphramBjorn@gmail.com|3341831232|   active|86958|2013-02-19T
04:43:19|2018-11-15T17:30:53|  2/7/1997|       true|           true|         true|    false|
```

Save records with anomalies to the json file

READY

```
sql("""SELECT
       |members.id,
       |members.first_name,
       |members.last_name,
       |members.email,
       |members.phone,
       |members.status,
       |members.zip4,
       |members.created_at,
       |members.updated_at,
       |members.birth_date,
       |is_valid_email(members.email) as valid_email,
       |is_valid_zip_code(members.zip4) as valid_zip_code,
       |is_valid_phone(members.phone) as valid_phone,
       |if(datediff( to_date(members.created_at, 'yyyy-mm-dd'),to_date(members.birth_date, 'mm
       |FROM members
       |WHERE !is_valid_email(members.email) or !is_valid_zip_code(members.zip4) or !is_valid_
       save(jsonAnomaliesPath)
```

Save valid records to conform the data types

READY

```
sql("""select
       |cast(members.id as int) as id,
       |members.first_name,
       |members.last_name,
       |members.email,
       |cast(members.phone as long) as phone,
       |members.status,
       |cast(members.zip4 as int) zip5,
       |cast(members.created_at as timestamp) as created_at,
       |cast(members.updated_at as timestamp) as updated_at,
       |to_date(members.birth_date, 'mm/dd/yyyy') as birth_date
```

```
        |from members
        |where is_valid_email(members.email) and is_valid_zip_code(members.zip4) and is_valid_r
```

Read valid records from parquet files and print updated schema datatypes

READY

```
val memebersParquet = spark.read.parquet(s"$parquetPath")
memebersParquet.createOrReplaceTempView("members_parquet")
memebersParquet.printSchema
```

```
root
 |-- id: integer (nullable = true)
 |-- first_name: string (nullable = true)
 |-- last_name: string (nullable = true)
 |-- email: string (nullable = true)
 |-- phone: long (nullable = true)
 |-- status: string (nullable = true)
 |-- zip5: integer (nullable = true)
 |-- created_at: timestamp (nullable = true)
 |-- updated_at: timestamp (nullable = true)
 |-- birth_date: date (nullable = true)

memebersParquet: org.apache.spark.sql.DataFrame = [id: int, first_name: string ... 8 more fiel
ds]
```

Sample data from the temporary table based on the proper datatype schema

READY

```
sql("SELECT * FROM members_parquet limit 10").show(120,220,false)
```

```
+---+----------+---------+--------------------------+----------+---------+-----+------------
-------+-------------------+----------+
| id|first_name|last_name|                     email|     phone|   status| zip5|         cre
ated_at|         updated_at|birth_date|
+---+----------+---------+--------------------------+----------+---------+-----+------------
-------+-------------------+----------+
|  1|   Carissa|    Mearl|    CarissaMearl@gmail.com|4180777944|   active|47308|2013-02-12 0
3:35:13|2018-07-10 15:53:41|1975-01-20|
|  2|   Claudio|  Saundra|  ClaudioSaundra@gmail.com|6345775307|   active|72938|2013-02-12 0
4:44:38|2018-10-03 08:08:34|1966-01-22|
|  3|    Sophie|   Jarrod|    SophieJarrod@gmail.com|8342083684|   active|86025|2013-02-12 0
6:49:58|2018-11-30 14:54:27|1960-01-24|
|  4|    Dellia|   Kolten|    DelliaKolten@gmail.com|9908183568|   active|76377|2013-02-12 1
2:37:51|2018-07-28 11:56:23|1958-01-24|
|  5|   Towanda|  Donovan|  TowandaDonovan@gmail.com|9200652274|   active|57803|2013-02-12 1
5:27:37|2018-04-06 05:51:38|1968-01-22|
|  6|     Ebony|   Maryjo|     EbonyMaryjo@gmail.com|3162608332|cancelled|56469|2013-02-13 0
3:58:22|2018-10-25 21:57:34|1978-01-18|
```

Save the valid data to json files with updated datatypes and column names - original json has zip4, but mysql schema has zip5 as a column

READY

name

```
sql("SELECT * FROM members_parquet").coalesce(1).write.mode("overwrite").format("org.apache.sp
       save(validOutPath)
```

Display json dataset schema

READY

```
val validMembers = spark.read.json(validOutPath)
validMembers.printSchema
```

```
root
 |-- birth_date: string (nullable = true)
 |-- created_at: string (nullable = true)
 |-- email: string (nullable = true)
 |-- first_name: string (nullable = true)
 |-- id: long (nullable = true)
 |-- last_name: string (nullable = true)
 |-- phone: long (nullable = true)
 |-- status: string (nullable = true)
 |-- updated_at: string (nullable = true)
 |-- zip5: long (nullable = true)

validMembers: org.apache.spark.sql.DataFrame = [birth_date: string, created_at: string ... 8 m
ore fields]
```

Display valid json dataset

READY

```
validMembers.show(120,220,false)
```

```
+----------+------------------------+----------------------------+----------+---+----------+
----------+---------+------------------------+-----+
|birth_date|              created_at|                       email|first_name| id| last_name|
phone|   status|              updated_at| zip5|
+----------+------------------------+----------------------------+----------+---+----------+
----------+---------+------------------------+-----+
|1975-01-20|2013-02-12T03:35:13.000Z|      CarissaMearl@gmail.com|   Carissa|  1|     Mearl|
4180777944|   active|2018-07-10T15:53:41.000Z|47308|
|1966-01-22|2013-02-12T04:44:38.000Z|    ClaudioSaundra@gmail.com|   Claudio|  2|   Saundra|
6345775307|   active|2018-10-03T08:08:34.000Z|72938|
|1960-01-24|2013-02-12T06:49:58.000Z|      SophieJarrod@gmail.com|    Sophie|  3|    Jarrod|
8342083684|   active|2018-11-30T14:54:27.000Z|86025|
|1958-01-24|2013-02-12T12:37:51.000Z|      DelliaKolten@gmail.com|    Dellia|  4|    Kolten|
9908183568|   active|2018-07-28T11:56:23.000Z|76377|
|1968-01-22|2013-02-12T15:27:37.000Z|    TowandaDonovan@gmail.com|   Towanda|  5|   Donovan|
9200652274|   active|2018-04-06T05:51:38.000Z|57803|
|1978-01-18|2013-02-13T03:58:22.000Z|       EbonyMaryjo@gmail.com|     Ebony|  6|    Maryjo|
3162608332|cancelled|2018-10-25T21:57:34.000Z|56469|
```