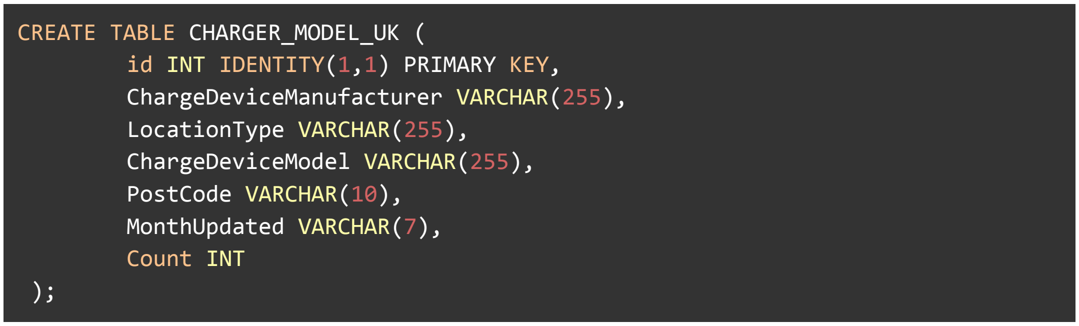
Proposal for integrating and updating data from AWS S3 to RDMS

The file "**charger_models_UK.csv**" hosted in AWS S3 could be used with an RDBMS like Amazon RDS. We will create a table in RDBMS with the same columns as the CSV file (we can add some additional column for the primary key) and then load the CSV file data to the table.

I will take SQL Server to demonstrate the SQL steps:

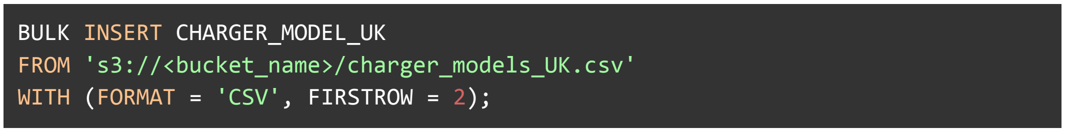
1. Create a new database in the Amazon RDS, if one does not already exist, and connect to it using a SQL tool.

2. Create the new table CHARGER_MODEL_UK, with following columns: ChargeDeviceManufacturer, LocationType, ChargeDeviceModel, PostCode, MonthUpdated, Count



```
CREATE TABLE CHARGER_MODEL_UK (
        id INT IDENTITY(1,1) PRIMARY KEY,
        ChargeDeviceManufacturer VARCHAR(255),
        LocationType VARCHAR(255),
        ChargeDeviceModel VARCHAR(255),
        PostCode VARCHAR(10),
        MonthUpdated VARCHAR(7),
        Count INT
);
```

3. Load the CSV file from S3 bucket to the new table. id column will be populated parallelly.



```
BULK INSERT CHARGER_MODEL_UK
FROM 's3://<bucket_name>/charger_models_UK.csv'
WITH (FORMAT = 'CSV', FIRSTROW = 2);
```

Once the data is loaded into the RDBMS, it can be queried and analyzed using SQL.

When the source data suffers regular changes

In case the source data would suffer regular changes, the data in the RDBMS table can be kept up to date by periodically re-loading the CSV file into the table.

We can do this by automating the script, daily or weekly, using an autosys, cron job, or AWS Lambda.

We can reload the data using multiple ways:

1. We can either add a step to the existing script or create a new one to perform a full load of the data, for reloading them, by truncating the table and using BULK Load.
2. We can also use AWS Glue service by creating a Glue job that will extract csv data from S3 and load only new or updated records in the table.
3. We can create a staging table and load the updated CSV data. Then by using the MERGE SQL command, we can incrementally load the data to our main table. Below is the SQL code for the same:

```
 MERGE INTO CHARGER_MODEL_UK AS target
 USING CHARGER_MODEL_UK_staging AS source
 ON target.ChargeDeviceManufacturer = source.ChargeDeviceManufacturer
 AND target.LocationType = source.LocationType
 AND target.ChargeDeviceModel = source.ChargeDeviceModel
 AND target.PostCode = source.PostCode
 AND target.MonthUpdated = source.MonthUpdated
 WHEN NOT MATCHED BY TARGET THEN
      INSERT (ChargeDeviceManufacturer, LocationType, ChargeDeviceModel,
PostCode, MonthUpdated, count)
       VALUES (source.ChargeDeviceManufacturer, source.LocationType,
source.ChargeDeviceModel, source.PostCode, source.MonthUpdated, source.count)
 WHEN MATCHED THEN
       UPDATE SET target.Count = source.Count;
```

Moreover, we can also store a timestamp or version number of the last update, and use that to filter out any data that has already been processed.

Overall, integrating the CSV file with a RDBMS like Amazon RDS can help us perform more complex queries and analysis on the data, while also providing a scalable and secure storage solution for the data.