Week IX

Capstone Project

Intermediate Data Engineer in Azure
Trainer: Balazs Balogh

CUBIX
INSTITUTE OF TECHNOLOGY

Medallion Architecture II. – Silver Layer - Sales

```python
import pyspark.sql.functions as sf
from pyspark.sql import DataFrame

SALES_MAPPING = {
    "son": "SalesOrderNumber",
    "orderdate": "OrderDate",
    "pk": "ProductKey",
    "ck": "CustomerKey",
    "dateofshipping": "ShipDate",
    "oquantity": "OrderQuantity"
}


def get_sales(sales_raw: DataFrame) -> DataFrame:
    """Map and filtered Sales data.

    :param sales_raw:   Raw Sales data.
    :return:            Mapped and filtered Sales data.
    """

    return (
        sales_raw
        .select(
           sf.col("son"),
           sf.col("orderdate").cast("date"),
           sf.col("pk").cast("int"),
           sf.col("ck").cast("int"),
           sf.col("dateofshipping").cast("date"),
           sf.col("oquantity").cast("int"),
        )
        .withColumnsRenamed(SALES_MAPPING)
        .dropDuplicates()
    )
```

Create the Sales transformations, and unit test. Don’t forget to commit and push from time to time.

Medallion Architecture II. – Silver Layer - Customers

```python
import pyspark.sql.functions as sf
from pyspark.sql import DataFrame

CUSTOMERS_MAPPING = {
    "ck": "CustomerKey",
    "name": "Name",
    "bdate": "BirthDate",
    "ms": "MaritalStatus",
    "gender": "Gender",
    "income": "YearlyIncome",
    "childrenhome": "NumberChildrenAtHome",
    "occ": "Occupation",
    "hof": "HouseOwnerFlag",
    "nco": "NumberCarsOwned",
    "addr1": "AddressLine1",
    "addr2": "AddressLine2",
    "phone": "Phone",
}
```

Next up is the Customers transformation, and unit test.

```python
def get_customers(customers_raw: DataFrame) -> DataFrame:
    """Transform and filter Customers data.

    1. Selecting needed columns.
    2. Apply the column name mapping.
    3. Transform MaritalStatus.
    4. Transform Gender.
    5. Create FullAddress column.
    6. Create IncomeCategory column.
    7. Create BithYear column.
    8. Drop duplicates.

    :param customers_raw:   Raw Customers data
    :return:                Cleaned, filtered, and transformed Customers data.
    """

    return (
        customers_raw
        .select(
            sf.col("ck").cast("int"),
            sf.col("name"),
            sf.col("bdate").cast("date"),
            sf.col("ms"),
            sf.col("gender"),
            sf.col("income").cast("int"),
            sf.col("childrenhome").cast("int"),
            sf.col("occ"),
            sf.col("hof").cast("int"),
            sf.col("nco").cast("int"),
            sf.col("addr1"),
            sf.col("addr2"),
            sf.col("phone")
        )
```

Medallion Architecture II. – Silver Layer - Customers

```
.withColumnsRenamed(CUSTOMERS_MAPPING)
        .withColumn(
            "MaritalStatus",
            sf.when(sf.col("MaritalStatus") == "M", 1)
            .when(sf.col("MaritalStatus") == "S", 0)
            .otherwise(None)
            .cast("int")
        )
        .withColumn(
            "Gender",
            sf.when(sf.col("Gender") == "M", 1)
            .when(sf.col("Gender") == "F", 0)
            .otherwise(None)
            .cast("int")
        )
        .withColumn(
            "FullAddress",
            sf.concat_ws(", ", sf.col("AddressLine1"), sf.col("AddressLine2"))
        )
        .withColumn(
            "IncomeCategory",
            sf.when(sf.col("YearlyIncome") <= 50000, "Low")
            .when(sf.col("YearlyIncome") <= 100000, "Medium")
            .otherwise("High")
        )
        .withColumn(
            "BirthYear",
            sf.year(sf.col("BirthDate"))
            .cast("int")
        )
        .dropDuplicates()
    )
```

Medallion Architecture II. – Silver Layer – Products

```python
import pyspark.sql.functions as sf
from pyspark.sql import DataFrame
from pyspark.sql.types import DecimalType

PRODUCTS_MAPPING = {
    "pk": "ProductKey",
    "psck": "ProductSubCategoryKey",
    "name": "ProductName",
    "stancost": "StandardCost",
    "dealerprice": "DealerPrice",
    "listprice": "ListPrice",
    "color": "Color",
    "size": "Size",
    "range": "SizeRange",
    "weight": "Weight",
    "nameofmodel": "ModelName",
    "ssl": "SafetyStockLevel",
    "desc": "Description"
}
```

Continue with the Products transformation, and unit test.

```python
def get_products(products_raw: DataFrame) -> DataFrame:
    """Transform and filter Products data.

    1. Select needed columns, and cast data types.
    2. Rename columns according to mapping.
    3. Create "ProfitMargin".
    4. Replace "NA" values with None.
    5. Drop duplicates.

    :param products_raw:    Raw Products data
    :return:                Cleaned, filtered, and transformed Products data.
    """

    return (
        products_raw
        .select(
            sf.col("pk").cast("int"),
            sf.col("psck").cast("int"),
            sf.col("name"),
            sf.col("stancost").cast(DecimalType(10, 2)).alias("stancost"),
            sf.col("dealerprice").cast(DecimalType(10, 2)).alias("dealerprice"),
            sf.col("listprice").cast(DecimalType(10, 2)).alias("listprice"),
            sf.col("color"),
            sf.col("size").cast("int"),
            sf.col("range"),
            sf.col("weight").cast(DecimalType(10, 2)).alias("weight"),
            sf.col("nameofmodel"),
            sf.col("ssl").cast("int"),
            sf.col("desc")
        )
        .withColumnsRenamed(PRODUCTS_MAPPING)
        .withColumn("ProfitMargin", sf.col("ListPrice") - sf.col("DealerPrice"))
        .replace("NA", None)
        .dropDuplicates()
    )
```

Medallion Architecture II. – Silver Layer – Product Subcategory

```
import pyspark.sql.functions as sf
from pyspark.sql import DataFrame

PRODUCT_SUBCATEGORY_MAPPING = {
    "psk": "ProductSubcategoryKey",
    "pck": "ProductCategoryKey",
    "epsn": "EnglishProductSubcategoryName",
    "spsn": "SpanishProductSubcategoryName",
    "fpsn": "FrenchProductSubcategoryName",
}


def get_product_subcategory(products_subcategory_raw: DataFrame) -> DataFrame:
    """Transform and filter Product Subcategory data.

    1. Select needed columns, and cast data types.
    2. Rename columns.

    :param products_subcategory_raw:    Raw Product Subcategory data
    :return:                            Cleaned, filtered, and transformed Product
Subcategory data.
    """

    return (
        products_subcategory_raw
        .select(
            sf.col("psk").cast("int"),
            sf.col("pck").cast("int"),
            sf.col("epsn"),
            sf.col("spsn"),
            sf.col("fpsn")
        )
        .withColumnsRenamed(PRODUCT_SUBCATEGORY_MAPPING)
        .dropDuplicates()
    )
```

Finish with the Product Subcategory transformation and unit test.

CUBIX
INSTITUTE OF TECHNOLOGY