# Data Warehousing and BI Analytics

## Scenario

You are a data engineer hired by a solid waste management company. The company collects and recycles solid waste across major cities in the country of Brazil. The company operates hundreds of trucks of different types to collect and transport solid waste. The company would like to create a data warehouse so that it can create reports like

- total waste collected per year per city
- total waste collected per month per city
- total waste collected per quarter per city
- total waste collected per year per trucktype
- total waste collected per trucktype per city
- total waste collected per trucktype per station per city

You will use your data warehousing skills to design and implement a data warehouse for the company.

## Objectives

In this assignment you will:

- Design a Data Warehouse
- Load data into Data Warehouse
- Write aggregation queries
- Create MQTs
- Create a Dashboard

# Exercise 1 - Design a Data Warehouse

The solid waste management company has provied you the sample data they wish to collect.

| Trip number | Waste Type | Waste Collected in tons | Collection Zone | City | Date |
|---|---|---|---|---|---|
| 1 | Dry | 45.23 | South | Sao Paulo | 23-Jan-20 |
| 2 | Wet | 43.12 | Central | Rio de Janeiro | 24-Jan-20 |
| 3 | Electronic | 40.19 | South | Sao Paulo | 23-Jan-20 |
| 4 | Plastic | 34.87 | West | Rio de Janeiro | 24-Jan-20 |
| 5 | Wet | 45.34 | West | Rio de Janeiro | 23-Jan-20 |

You will start your project by designing a Star Schema warehouse by identifying the columns for the various dimension and fact tables in the schema.

### Task 1 - Design the dimension table MyDimDate

Write down the fields in the MyDimDate table in any text editor one field per line. The company is looking at a granularity of day. Which means they would like to have the ability to generate the report on yearly, monthly, daily, and weekday basis.

MyDimDate
- dateid
- date
- Year
- Quarter
- QuarterName
- Month
- Monthname
- Day
- Weekday
- WeekdayName

### Task 2 - Design the dimension table MyDimWaste

Write down the fields in the MyDimWaste table in any text editor one field per line.

MyDimWaste
- wasteid
- wastetype

### Task 3 - Design the dimension table MyDimZone

Write down the fields in the MyDimZone table in any text editor one field per line.

MyDimzone
- zoneid
- collectionzone
- city

### Task 4 - Design the fact table MyFactTrips

Write down the fields in the MyFactTrips table in any text editor one field per line.

MyFactTrips
- tripid
- dateid
- wasteid
- zoneid
- wastecollected

# Exercise 2 - Create schema for Data Warehouse on PostgreSQL

### Task 5 - Create the dimension table MyDimDate

Create the MyDimDate table.

WasteManagement/postgres@postgres

Query Editor  Query History

```
CREATE TABLE public."MyDimDate"
(
    dateid integer NOT NULL,
    date date,
    day integer,
    year integer,
    weekday integer,
    weekname character(50),
    month integer,
    monthname character(50),
    quarter integer,
    quartername character(50),
    CONSTRAINT "MyDimDate.pkey" PRIMARY KEY (dateid)
)
```

### Task 6 - Create the dimension table MyDimWaste

Create the MyDimWaste table.

WasteManagement/postgres@postgres

Query Editor  Query History

```
CREATE TABLE public."MyDimWaste"
(
    wasteid integer NOT NULL,
    wastetype character(50),
    CONSTRAINT "MyDimWaste.pkey" PRIMARY KEY (wasteid)
)
```

### Task 7 - Create the dimension table MyDimZone

Create the MyDimZone table.

WasteManagement/postgres@postgres

Query Editor  Query History

```
CREATE TABLE public."MyDimZone"
(
    zoneid integer NOT NULL,
    collectionzone character(50),
    city character(50),
    CONSTRAINT "MyDimZone.pkey" PRIMARY KEY (zoneid)
)
```

### Task 8 - Create the fact table MyFactTrips

Create the MyFactTrips table.

WasteManagement/postgres@postgres

Query Editor  Query History

```
CREATE TABLE public."MyFactTrips"
(
    tripid integer NOT NULL,
    dateid integer NOT NULL,
    wasteid integer NOT NULL,
    zoneid integer NOT NULL,
    wastecollected numeric,
    CONSTRAINT "MyFactTrips.pkey" PRIMARY KEY (tripid)
    CONSTRAINT "MyFactTrips.fkeydate" FOREIGN KEY (dateid) REFERENCES public."MyDimDate" (dateid),
    CONSTRAINT "MyFactTrips.fkeyzone" FOREIGN KEY (zoneid) REFERENCES public."MyDimZone" (zoneid),
    CONSTRAINT "MyFactTrips.fkeywaste" FOREIGN KEY (wasteid) REFERENCES public."MyDimWaste" (wasteid)
)
```

# Exercise 3 - Load data into the Data Warehouse

In this exercise you will load the data into the tables.

After the initial schema design, you were told that due to opertional issues, data could not be collected in the format initially planned.

You will load the data provided by the company in csv format.

### Task 9 - Load data into the dimension table DimDate

Download the data from https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0260EN-SkillsNetwork/labs/Final%20Assignment/DimDate.csv

Load this data into DimDate table.

WasteManagement/postgres@postgres

Query Editor | Query History

```
CREATE TABLE public."DimDate"
(
    dateid integer NOT NULL,
    date date,
    Year integer,
    Quarter integer,
    QuarterName character(50),
    Month integer,
    Monthname character(50),
    Day integer,
    Weekday integer,
    WeekdayName character(50),
    CONSTRAINT "DimDate.pkey" PRIMARY KEY (dateid)
)
```

Data Output | Explain | Messages | Notifications

```
CREATE TABLE

Query returned successfully in 609 msec.
```

WasteManagement/postgres@postgres

Query Editor | Query History | Scratch

```
SELECT * FROM public."DimDate" LIMIT 5;
```

Data Output | Explain | Messages | Notifications

| | dateid [PK] integer | date date | year integer | quarter integer | quartername character (50) | month integer | monthname character (50) | day integer | weekday integer | weekdayname character (50) |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2019-03... | 2019 | 1 | Q1 | 3 | March | 9 | 7 | Sunday |
| 2 | 2 | 2019-03... | 2019 | 1 | Q1 | 3 | March | 10 | 1 | Monday |
| 3 | 3 | 2019-03... | 2019 | 1 | Q1 | 3 | March | 11 | 2 | Tuesday |
| 4 | 4 | 2019-03... | 2019 | 1 | Q1 | 3 | March | 12 | 3 | Wednesday |
| 5 | 5 | 2019-03... | 2019 | 1 | Q1 | 3 | March | 13 | 4 | Thursday |

### Task 10 - Load data into the dimension table DimTruck

Download the data from https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0260EN-SkillsNetwork/labs/Final%20Assignment/DimTruck.csv

Load this data into DimTruck table.

WasteManagement/postgres@postgres

Query Editor | Query History

```
SELECt * FROM public."DimTruck" LIMIT 5;
```

Data Output | Explain | Messages | Notifications

| | truckid [PK] integer | trucktype character (50) |
|---|---|---|
| 1 | 115 | Volvo |
| 2 | 120 | Scania |
| 3 | 121 | Volvo |
| 4 | 122 | Scania |
| 5 | 125 | Volvo |

### Task 11 - Load data into the dimension table DimStation

Download the data from https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0260EN-SkillsNetwork/labs/Final%20Assignment/DimStation.csv

Load this data into DimStation table.

WasteManagement/postgres@postgres

Query Editor | Query History

```
CREATE TABLE public."DimStation"
(
    Stationid integer NOT NULL,
    City character(50),
    CONSTRAINT "DimStation.pkey" PRIMARY KEY (Stationid)
)
```

Data Output | Explain | Messages | Notifications

```
CREATE TABLE

Query returned successfully in 578 msec.
```

WasteManagement/postgres@postgres

Query Editor | Query History

```
SELECt * FROM public."DimStation" LIMIT 5;
```

Data Output | Explain | Messages | Notifications

| | stationid [PK] integer | city character (50) |
|---|---|---|
| 1 | 19 | Sao Paulo |
| 2 | 21 | Sao Paulo |
| 3 | 31 | Rio de Janeiro |
| 4 | 32 | Rio de Janeiro |
| 5 | 40 | Brasilia |

### Task 12 - Load data into the fact table FactTrips

Download the data from https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0260EN-SkillsNetwork/labs/Final%20Assignment/FactTrips.csv

Load this data into FactTrips table.

WasteManagement/postgres@postgres

Query Editor | Query History

```
CREATE TABLE public."FactTrips"
(
    Tripid integer NOT NULL,
    Dateid integer NOT NULL,
    Stationid integer NOT NULL,
    Truckid integer NOT NULL,
    Wastecollected numeric,
    CONSTRAINT "FactTrips.pkey" PRIMARY KEY (Tripid),
    CONSTRAINT "FactTrips.fkeydate" FOREIGN KEY (Dateid) REFERENCES public."DimDate" (dateid),
    CONSTRAINT "FactTrips.fkeystation" FOREIGN KEY (Stationid) REFERENCES public."DimStation" (Stationid),
    CONSTRAINT "FactTrips.fkeytruck" FOREIGN KEY (Truckid) REFERENCES public."DimTruck" (Truckid)
)
```

WasteManagement/postgres@postgres

Query Editor | Query History

```
SELECT * FROM public."FactTrips" LIMIT 5;
```

Data Output | Explain | Messages | Notifications

| | tripid [PK] integer | dateid integer | stationid integer | truckid integer | wastecollected numeric |
|---|---|---|---|---|---|
| 1 | 23475 | 1 | 71 | 133 | 33.36 |
| 2 | 23476 | 1 | 46 | 162 | 34.88 |
| 3 | 23477 | 1 | 40 | 134 | 34.69 |
| 4 | 23478 | 1 | 43 | 148 | 30.01 |
| 5 | 23479 | 1 | 46 | 169 | 37.47 |

# Exercise 4 - Write aggregation queries and create MQTs

### Task 13 - Create a grouping sets query

Create a grouping sets query using the columns stationid, trucktype, total waste collected.

WasteManagement/postgres@postgres

Query Editor | Query History

```
SELECT "DimStation".stationid, "DimTruck".trucktype, sum(wastecollected) AS totalwastecollected
FROM "FactTrips"
LEFT JOIN "DimStation"
ON "FactTrips".stationid = "DimStation".stationid
LEFT JOIN "DimTruck"
ON "FactTrips".truckid="DimTruck".truckid
GROUP BY GROUPING SETS("DimStation".stationid, "DimTruck".trucktype);
```

Data Output

| | stationid integer | trucktype character (50) | totalwastecollected numeric |
|---|---|---|---|
| 1 | 40 | [null] | 163671.71 |
| 2 | 43 | [null] | 166730.64 |
| 3 | 48 | [null] | 254136.86 |
| 4 | 57 | [null] | 164839.47 |
| 5 | 19 | [null] | 332816.78 |
| 6 | 81 | [null] | 247492.72 |
| 7 | 86 | [null] | 163940.76 |
| 8 | 31 | [null] | 167061.68 |
| 9 | 47 | [null] | 248569.92 |
| 10 | 71 | [null] | 165131.77 |
| 11 | 84 | [null] | 166212.16 |
| 12 | 97 | [null] | 165984.20 |
| 13 | 44 | [null] | 166456.22 |
| 14 | 82 | [null] | 246759.30 |
| 15 | 77 | [null] | 165914.25 |
| 16 | 21 | [null] | 333142.01 |
| 17 | 83 | [null] | 332035.37 |
| 18 | 46 | [null] | 169264.68 |
| 19 | 32 | [null] | 167764.84 |
| 20 | [null] | Volvo | 1908758.30 |

### Task 14 - Create a rollup query

Create a rollup query using the columns year, city, stationid, and total waste collected.

WasteManagement/postgres@postgres

Query Editor | Query History

```
SELECT "DimDate".year, "DimStation".city, "DimStation".stationid, sum(wastecollected) AS totalwastecollected
FROM "FactTrips"
LEFT JOIN "DimStation"
ON "FactTrips".stationid = "DimStation".stationid
LEFT JOIN "DimDate"
ON "FactTrips".dateid="DimDate".dateid
GROUP BY ROLLUP("DimDate".year, "DimStation".city, "DimStation".stationid)
ORDER BY year, city, stationid;
```

Data Output

| | year integer | city character (50) | stationid integer | totalwastecollected numeric |
|---|---|---|---|---|
| 1 | 2019 | Brasilia | 40 | 138850.54 |
| 2 | 2019 | Brasilia | 43 | 141503.54 |
| 3 | 2019 | Brasilia | 46 | 144418.78 |
| 4 | 2019 | Brasilia | 71 | 141856.38 |
| 5 | 2019 | Brasilia | 77 | 141627.10 |
| 6 | 2019 | Brasilia | 97 | 140485.97 |
| 7 | 2019 | Brasilia | [null] | 848742.31 |
| 8 | 2019 | Rio de Janeiro | 31 | 143821.05 |
| 9 | 2019 | Rio de Janeiro | 32 | 142766.60 |
| 10 | 2019 | Rio de Janeiro | 44 | 141766.33 |
| 11 | 2019 | Rio de Janeiro | 57 | 141405.86 |

### Task 15 - Create a cube query

Create a cube query using the columns year, city, stationid, and average waste collected.

WasteManagement/postgres@postgres

Query Editor  Query History

```
SELECT "DimDate".year, "DimStation".city, "DimStation".stationid, AVG(wastecollected) AS averagewastecollected
FROM "FactTrips"
LEFT JOIN "DimDate"
ON "FactTrips".dateid = "DimDate".dateid
LEFT JOIN "DimStation"
ON "FactTrips".stationid="DimStation".stationid
GROUP BY CUBE("DimDate".year, "DimStation".city, "DimStation".stationid)
ORDER BY year,city, stationid;
```

Data Output

| | year integer | city character (50) | stationid integer | averagewastecollected numeric |
|---|---|---|---|---|
| 1 | 2019 | Brasilia ... | 40 | 37.3756500672947510 |
| 2 | 2019 | Brasilia ... | 43 | 37.5042512589451365 |
| 3 | 2019 | Brasilia ... | 46 | 37.5016307452609712 |
| 4 | 2019 | Brasilia ... | 71 | 37.4587747557433325 |
| 5 | 2019 | Brasilia ... | 77 | 37.4674867724867725 |
| 6 | 2019 | Brasilia ... | 97 | 37.5430171031533939 |
| 7 | 2019 | Brasilia ... | [null] | 37.4753757506181561 |
| 8 | 2019 | Rio de Janeiro ... | 31 | 37.4631544673091951 |
| 9 | 2019 | Rio de Janeiro ... | 32 | 37.5207884362680683 |
| 10 | 2019 | Rio de Janeiro ... | 44 | 37.4547767503302510 |
| 11 | 2019 | Rio de Janeiro ... | 57 | 37.4188568404339772 |

### Task 16 - Create an MQT

Create an MQT named max_waste_stats using the columns city, stationid, trucktype, and max waste collected.

WasteManagement/postgres@postgres

Query Editor  Query History

```
CREATE MATERIALIZED VIEW  max_waste_stats
AS (SELECT "DimStation".city, "DimStation".stationid, "DimTruck".trucktype, MAX(wastecollected) AS maxwastecollected
FROM "FactTrips"
LEFT JOIN "DimStation"
ON "FactTrips".stationid = "DimStation".stationid
LEFT JOIN "DimTruck"
ON "FactTrips".truckid = "DimTruck".truckid
GROUP BY GROUPING SETS ("DimStation".city, "DimStation".stationid, "DimTruck".trucktype));
```

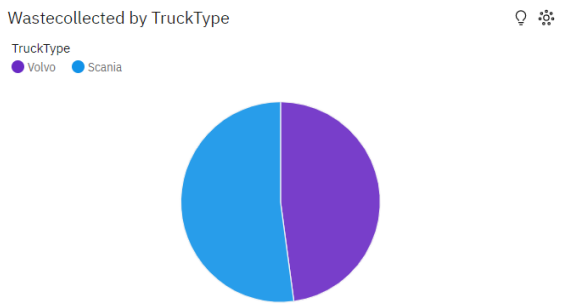
# Exercise 5 - Create a dashboard using Cognos Analytics

Download the data from DataForCognos_date

Use the DataForCognos_date.csv file to generate the following charts.

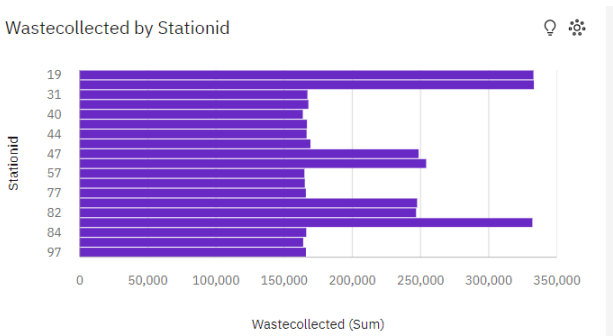
### Task 17 - Create a pie chart in the dashboard

Create a pie chart that shows the waste collected by truck type.



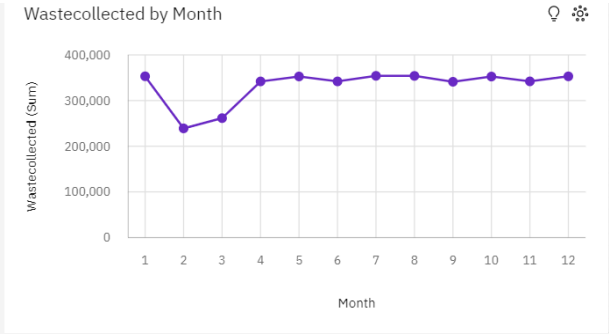

### Task 18 - Create a bar chart in the dashboard

Create a bar chart that shows the waste collected station wise.




### Task 19 - Create a line chart in the dashboard

Create a line chart that shows the waste collected by month wise.



**Task 20 - Create a pie chart in the dashboard**

Create a pie chart that shows the waste collected by city.