****

**Academia SQL + Teradata**

Após termos verificado a qualidade dos dados identificando suas inconsistências, agora iremos transformá-los adequando-os às necessidades técnicas e de negócio. Neste segundo desafio, o objetivo é portanto realizar algumas limpezas ou padronizações e enriquecer as mesmas 3 tabelas anteriormente utilizadas (airports, planes e flights), criando ou atualizando campos.

Airports Dataset



Tarefas

1. Para todo valor da coluna tz dentro do intervalo [−7,−5], substitua o valor na coluna dst pelo horário de verão US/Canada. Não atualizar a coluna quando não for necessário.

*--51 linhas afetadas*

**update** airports **set** dst = 'A'

**where** airports.tz between -7 and -5

**and** dst <> 'A'**;**

1. Para todo valor da coluna dst que seja igual a U, substitua por A.

*--8 linhas afetadas*

**update** airports **set** dst = 'A'

**where** dst = 'U'**;**

1. Crie a coluna region e atribua os valores de acordo com as condições abaixo:

**alter** **table** airports

**add** region VARCHAR(20)**;**

Obs: a região principal dos EUA está no intervalo de longitude [−124,−50]

ALASKA : Quando a longitude for menor que -124.

OFFSHORE : Quando a longitude for maior que -50 ou a latitude for menor que 24.

WEST : Quando a longitude for menor ou igual -95 na região principal dos EUA.

EAST : Quando a longitude for maior que -95 na região principal dos EUA.

NaN : Caso não atenda nenhuma das condições acima.

Consultar a frequência dos valores do novo campo (contagem de cada valor).

**update** airports **set** region =

(

**case**

**when** lon < -124 **then** 'ALaska'

**when** lon > -50 **and** alt < 24 **then** 'OFFSHORE'

**when** lon <= -95 **then** 'WEST'

**when** lon > -95 **then** 'EAST'

**else** 'NAN'

**end**

)

**where** lon between -124 and -50

*--1132 alteraçoes*

**select** region **from** airports

1. Crie a coluna tipo e atribua os valores a partir de subtrings identificadas na coluna name de acordo com as condições abaixo:

Obs: a prioridade das correspondências deve ser a mesma listada abaixo. Por exemplo, caso seja identificado "Airport" e "Field" na mesma string, o valor atribuido deve ser AP.

**alter** **table** airports

**add** tipo CHAR(5)**;**

AP : "Airport", "Tradeport", "Heliport", "Airpor", ou "Arpt"

AD : "Aerodrome"

AK : "Airpark" ou "Aero Park"

AS : "Station" ou "Air Station"

FL : "Field" ou "Fld"

NaN : Caso não atenda nenhuma das condições acima. Consultar a frequência dos valores do novo campo.

**Select** **name**, tipo,

(

**case**

**WHEN** **name** = 'Airport|Tradeport|Heliport|Airpor|Arpt' **THEN** 'AP'

**WHEN** **name** = 'Aerodrome' **THEN** 'AD'

**WHEN** **name** = 'Airpark|Aero Park' **THEN** 'AK'

**WHEN** **name** = 'Station|Air Station' **THEN** 'AS'

**WHEN** **name** = 'Field|Fld' **THEN** 'FL'

**ELSE** **NULL**

**END**

)

**from** airports**;**

**UPDATE** airports **SET** tipo =

(

**CASE**

**WHEN** **name** = 'Airport|Tradeport|Heliport|Airpor|Arpt' **THEN** 'AP'

**WHEN** **name** = 'Aerodrome' **THEN** 'AD'

**WHEN** **name** = 'Airpark|Aero Park' **THEN** 'AK'

**WHEN** **name** = 'Station|Air Station' **THEN** 'AS'

**WHEN** **name** = 'Field|Fld' **THEN** 'FL'

**ELSE** 'Nan'

**END**

)

**;** *--1397*

1. Crie a coluna military e atribua os valores a partir de subtrings identificadas ao final da coluna name de acordo com as condições abaixo:

**alter** **table** airports

**add** military CHAR(8)**;**

Y: "Base", "Aaf", "AFs", "Ahp", "Afb", "LRRS", "Lrrs", "Arb", "Naf", "NAS", "Nas", "Jrb", "Ns", "As", "Cgas", "Angb".

N: Caso nenhuma substring acima seja identificada.

Consultar a frequência dos valores do novo campo.

**select** \* **from** airports**;**

*----1397*

**select** **name**, military,

(

**CASE**

**WHEN** **name** LIKE '%Base' **THEN** 'Y'

**WHEN** **name** LIKE '%Aaf' **THEN** 'Y'

**WHEN** **name** LIKE '%AFs' **THEN** 'Y'

**WHEN** **name** LIKE '%Ahp' **THEN** 'Y'

**WHEN** **name** LIKE '%Afb' **THEN** 'Y'

**WHEN** **name** LIKE '%LRRS' **THEN** 'Y'

**WHEN** **name** LIKE '%Lrrs' **THEN** 'Y'

**WHEN** **name** LIKE '%Arb' **THEN** 'Y'

**WHEN** **name** LIKE '%Naf' **THEN** 'Y'

**WHEN** **name** LIKE '%NAS' **THEN** 'Y'

**WHEN** **name** LIKE '%Nas' **THEN** 'Y'

**WHEN** **name** LIKE '%Jrb' **THEN** 'Y'

**WHEN** **name** LIKE '%Ns' **THEN** 'Y'

**WHEN** **name** LIKE '%As' **THEN** 'Y'

**WHEN** **name** LIKE '%Cgas' **THEN** 'Y'

**WHEN** **name** LIKE '%Angb' **THEN** 'Y'

**ELSE** 'N'

**END**

)

**from** airports**;**

**update** airports **set** military =

(

**CASE**

**WHEN** **name** LIKE '%Base' **THEN** 'Y'

**WHEN** **name** LIKE '%Aaf' **THEN** 'Y'

**WHEN** **name** LIKE '%AFs' **THEN** 'Y'

**WHEN** **name** LIKE '%Ahp' **THEN** 'Y'

**WHEN** **name** LIKE '%Afb' **THEN** 'Y'

**WHEN** **name** LIKE '%LRRS' **THEN** 'Y'

**WHEN** **name** LIKE '%Lrrs' **THEN** 'Y'

**WHEN** **name** LIKE '%Arb' **THEN** 'Y'

**WHEN** **name** LIKE '%Naf' **THEN** 'Y'

**WHEN** **name** LIKE '%NAS' **THEN** 'Y'

**WHEN** **name** LIKE '%Nas' **THEN** 'Y'

**WHEN** **name** LIKE '%Jrb' **THEN** 'Y'

**WHEN** **name** LIKE '%Ns' **THEN** 'Y'

**WHEN** **name** LIKE '%As' **THEN** 'Y'

**WHEN** **name** LIKE '%Cgas' **THEN** 'Y'

**WHEN** **name** LIKE '%Angb' **THEN** 'Y'

**ELSE** 'N'

**END**)**;**

1. Crie a coluna administration e atribua os valores a partir de substrings identificadas na coluna name de acordo com as condições abaixo:

**alter** **table** airports

**add** administration CHAR(3)**;**

Obs: a prioridade das correspondências deve ser a mesma listada abaixo. Por exemplo, caso seja identificada "City" e "International" na mesma string, o valor atribuido deve ser I.

I : "International", "Intl", ou "Intercontinental"

N : "National", "Natl"

R : "Regional", "Reigonal", "Rgnl", "County", "Metro" ou "Metropolitan"

M : "Municipal", "Muni", ou "City"

NaN : Caso não atenda nenhuma das condições acima.

Consultar a frequência dos valores do novo campo.

**Select** **name**, administration,

(

**case**

**WHEN** **name** LIKE '%International%' **OR** **name** LIKE '%Intl%' **OR** **name** LIKE '%Intercontinental%' **THEN** 'I'

**WHEN** **name** LIKE '%National%' **OR** **name** LIKE '%Natl%' **THEN** 'N'

**WHEN** **name** LIKE '%Regional%' **OR** **name** LIKE '%Reigonal%' **OR** **name** LIKE '%Rgnl%' **OR** **name** LIKE '%County%' **OR** **name** LIKE '%Metro%' **OR** **name** LIKE '%Metropolitan%' **THEN** 'R'

**WHEN** **name** LIKE '%Municipal%' **OR** **name** LIKE '%Muni%' **OR** **name** LIKE '%City%' **THEN** 'M'

**ELSE** 'NaN'

**END**

)

**from** airports**;**

**UPDATE** airports **SET** administration =

(

**CASE**

**WHEN** **name** LIKE '%International%' **OR** **name** LIKE '%Intl%' **OR** **name** LIKE '%Intercontinental%' **THEN** 'I'

**WHEN** **name** LIKE '%National%' **OR** **name** LIKE '%Natl%' **THEN** 'N'

**WHEN** **name** LIKE '%Regional%' **OR** **name** LIKE '%Reigonal%' **OR** **name** LIKE '%Rgnl%' **OR** **name** LIKE '%County%' **OR** **name** LIKE '%Metro%' **OR** **name** LIKE '%Metropolitan%' **THEN** 'R'

**WHEN** **name** LIKE '%Municipal%' **OR** **name** LIKE '%Muni%' **OR** **name** LIKE '%City%' **THEN** 'M'

**ELSE** 'NaN'

**END**

)

**;**

**select** \* **from** airports**;**

**SELECT** administration, COUNT(\*) **AS** frequency

**FROM** airports

**GROUP** **BY** administration**;**

Planes Dataset



Tarefas

1. Na coluna ano, substitua todo valor inconsistente por 1996.

**select ano from planes;**

**---61 alteraçoes**

**select ano from planes**

**where (ano is null**

**or trim(ano) = ''**

**or length(trim(ano)) = 0**

**or trim(ano) = 'NA'**

**);**

**UPDATE planes**

**SET ano =**

**CASE**

**WHEN COALESCE(ano,'') = '' THEN '1996'**

**WHEN REGEXP\_REPLACE(ano, '[^0-9]+', '') <> ano THEN '1996'**

**ELSE ano**

**END;**

**SELECT ano, COUNT(\*) AS frequency**

**FROM planes**

**GROUP BY ano;**

1. Crie a coluna age e atribua a idade do avião com base na coluna ano e o ano atual. Verificar os resultados por amostra e de forma agregada utilizando soma e média.

**select age from planes;**

**select age from planes;**

**alter table planes drop age;**

**alter table planes**

**add age Byteint;**

**update planes set age = EXTRACT(YEAR from CURRENT\_DATE) - TO\_NUMBER(ano);**

**--46 resultados**

**select model, age from planes sample 10;**

**select model, age from planes;**

**select Sum(age) as Soma, Avg(age) as Media from planes;**

1. Para todo valor da coluna manufacturer substitua o valor na coluna manufacturer de modo que existam somente os valores únicos listados abaixo. Para simplificar a atualização recomenda-se utilizar uma tabela “de-para”.

AIRBUS

BOEING

BOMBARDIER

CESSNA

EMBRAER

SIKORSKY

CANADAIR

PIPER

MCDONNELL DOUGLAS

CIRRUS

BELL

KILDALL GARY

LAMBERT RICHARD

BARKER JACK

ROBINSON HELICOPTER

GULFSTREAM

MARZ BARRY

Verificar frequência resultante dos valores do campo após atualização, ordenando pelos mais frequentes.

**select** **Distinct**(manufacturer) **from** planes **ORDER** **BY** manufacturer**;**

**update** planes **set** manufacturer = 'BOMBARDIER'

**where** manufacturer = 'BOMBARDIER'**;**

**drop** **table** manufacturer\_de\_para**;**

**select** \* **from** manufacturer\_de\_para**;**

**create** **table** manufacturer\_de\_para(

old\_manufacturer Varchar(50),

new\_manufacturer Varchar(50)

)**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES** ('AIRBUS', 'AIRBUS')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('BOEING', 'BOEING')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('BOMBARDIER', 'BOMBARDIER')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('CESSNA', 'CESSNA')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('EMBRAER', 'EMBRAER')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('SIKORSKY', 'SIKORSKY')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('CANADAIR', 'CANADAIR')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('PIPER', 'PIPER')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('MCDONNELL', 'MCDONNELL DOUGLAS')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('CIRRUS', 'CIRRUS')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('BELL', 'BELL')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('KILDALL', 'KILDALL GARY')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES** ('LAMBERT', 'LAMBERT RICHARD')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('BARKER', 'BARKER JACK')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES** ('ROBINSON', 'ROBINSON HELICOPTER')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('GULFSTREAM', 'GULFSTREAM')**;**

**INSERT** **INTO** manufacturer\_de\_para(old\_manufacturer, new\_manufacturer)

**VALUES**('MARZ', 'MARZ BARRY')**;**

*---atualizar a coluna 2628 atualizadas*

**update** planes **set** manufacturer = (

**select** new\_manufacturer

**from** manufacturer\_de\_para

**where** old\_manufacturer = planes.manufacturer

)**;**

*-- contar frequencia*

**SELECT** manufacturer, COUNT(\*) **AS** frequency

**FROM** planes

**GROUP** **BY** manufacturer

**ORDER** **BY** frequency **DESC;**

1. Remova todos caracteres entre parenteses da coluna model mantendo somente os valores fora do parenteses.

**select** model **from** planes**;**

*select model from planes;*

*update planes set model = REGEXP\_REPLACE(model, '\(.\*?\)', '')*

*WHERE model LIKE '%(%' OR model LIKE '%)%';*

*--46 resultados*

*UPDATE planesSET model = OREPLACE(model, model, STRTOK(model, '(', 1))*

*WHERE model LIKE '%(%';*

1. Impute os valores não existentes na coluna speed com base na fórmula: seats / 0,36.

Verificar os resultados por amostra (top 5) e de forma agregada utilizando soma e média.

**select speed from planes;**

**select speed**

**from planes**

**where speed = 'NA';**

**update planes**

**set speed = TO\_CHAR(seats / 0.36)**

**where speed is null**

**or speed = ''**

**or speed = 'NA';**

**select TOP 5 speed**

**From planes;**

**--1.139.524,00**

**select SUM(speed) as 'Soma Speed'**

**From planes;**

**select AVG(speed) as 'Meida Speed'**

**From planes;**

**--433,61**

Flights Dataset



## Tarefas

1. Impute os valores não existentes nas colunas hora e minuto com 0, e os valores iguais a 24 da coluna hora por 0.

**select** hora, minuto

**from** flights

**where** hora = 'NA'

**or** hora **is** **null**

**and** minuto = 'NA'

**and** minuto = ''

**;**

*--29*

**update** flights **set**

hora = '0'

**where** hora = 'NA'

**or** hora **is** **null**

**;**

**update** flights **set** minuto = '0'

**where** minuto = 'NA'

**and** minuto = ''

**;**

**select** hora

**from** flights

**where** hora = '24'

**;**

*--1 linha*

**update** flights **set** hora = '0'

**where** hora = '24'

**;**

1. Crie a coluna dep\_datetime (timestamp) usando as colunas ano, mes, dia, hora, minuto no formato YYYY-MM-DD HH:MM:00.

Validar que a conversão foi ok comparando a frequência do campo mes com o campo mês da nova coluna, e vendo se os valores mínimo e máximo estão dentro do esperado.

**ALTER TABLE flights**

**ADD dep\_datetime TIMESTAMP(0);**

**UPDATE** flights

**SET** dep\_datetime =

CAST(ano **AS** VARCHAR(4))||'-'||

CAST(cast(mes **AS** **FORMAT**'9(2)')**as** char(2))||'-'||

CAST(cast(dia **AS** **FORMAT**'9(2)')**as** char(2))||' '||

CAST(cast(to\_number(hora) **AS** **FORMAT**'9(2)')**as** char(2))||':'||

CAST(cast(to\_number(minuto) **AS** **FORMAT**'9(2)')**as** char(2))||':'||'00'

**;**

---------------------------marcos----------------------------------

alter table flights add dep\_datetime timestamp;

update flights set dep\_datetime =

--sel ano,mes,dia,hora,minuto,

cast(trim(ano) || '-' ||

CAST(CAST(mes AS FORMAT '9(2)') AS CHAR(2)) || '-' ||

CAST(CAST(dia AS FORMAT '9(2)') AS CHAR(2))

||' '||

cast(CAST(to\_number(hora) AS FORMAT '9(2)') as char(2)) || ':' ||

cast(CAST(to\_number(minuto) AS FORMAT '9(2)') as char(2)) || ':00'

as timestamp);

--from flights;

select min(dep\_datetime),max(dep\_datetime) from flights;

select extract(month from dep\_datetime), count(1) from flights group by 1 order by 1;

select mes, count(1) from flights group by 1 order by 1;

1. Impute os valores não existentes na coluna dep\_time usando as colunas hora, minuto de acordo com o formato esperado pela coluna dep\_time.

**ALTER TABLE flights**

**ADD dep\_time TIME(0);**

**UPDATE flights**

**SET dep\_time = CAST(cast(to\_number(hora) AS FORMAT'9(2)')as char(2))||':'||**

**CAST(cast(to\_number(minuto) AS FORMAT'9(2)')as char(2))||':'||'00'**

**;**

**select dep\_time, hora, minuto from flights;**

**----------------------------------------------------MARCOS----------**

**select distinct hora, minuto from flights**

**where dep\_time = 'NA'**

**order by 1, 2;**

**update flights set dep\_time = '000'**

**where dep\_time = 'NA'; -- 29**

1. Impute os valores não existentes nas colunas dep\_delay e arr\_delay por 0.

**select** dep\_delay, arr\_delay

**from** flights

**where** dep\_delay = 'NA'

**or** dep\_delay **is** **null**

**and** arr\_delay = 'NA'

**and** arr\_delay = ''

**;**

*---29 alteraçoes*

**update** flights **set** dep\_delay = '0'

**where** dep\_delay = 'NA'

**or** dep\_delay **is** **null**

**;**

**update** flights **set** arr\_delay = '0'

**where** arr\_delay = 'NA'

**and** arr\_delay = ''

**;**

1. Crie a coluna air\_time\_projected de acordo com a fórmula distance x 0.1 + 20. Validar usando valores mínimo, máximo, soma e média.

**alter** **table** flights

**add** air\_time\_projected Decimal(18,2)**;**

**UPDATE** flights **SET** air\_time\_projected = distance \* 0.1 + 20**;**

**SELECT** MIN(air\_time\_projected) **as** Minino, MAX(air\_time\_projected) **as** Maximo, SUM(air\_time\_projected) **as** Soma , AVG(air\_time\_projected) **as** Media

**FROM** flights**;**

1. Crie a coluna air\_time\_expected de acordo com a média de valores dos voos com mesma origem e destino. Sugestão: usar uma tabela volátil para cálculo das médias. Validar usando valores mínimo, máximo, soma e média.

**CREATE** **VOLATILE** **TABLE** vt\_airtime

**AS** (

**SELECT** origin, dest, AVG(to\_number(air\_time)) **AS** avg\_air\_time

**FROM** flights

**GROUP** **BY** origin, dest

) **WITH** **DATA** **PRIMARY** **INDEX** (origin, dest) **ON** **COMMIT** **PRESERVE** **ROWS;**

**SELECT** f.\*, v.avg\_air\_time **AS** air\_time

**FROM** flights **AS** f

**INNER** **JOIN** vt\_airtime **AS** v

**ON** f.origin = v.origin **AND** f.dest = v.dest**;**

**SELECT** MIN(air\_time\_expected), MAX(air\_time\_expected), SUM(air\_time\_expected), AVG(air\_time\_expected)

**FROM** (

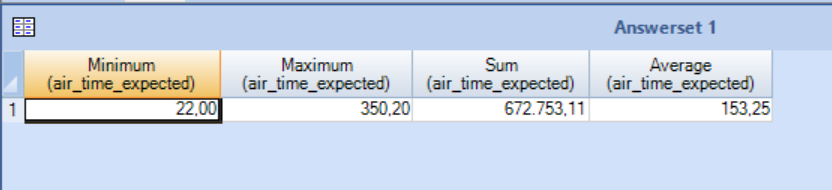
**SELECT** f.\*, v.avg\_air\_time **AS** air\_time\_expected

**FROM** flights **AS** f

**INNER** **JOIN** vt\_airtime **AS** v

**ON** f.origin = v.origin **AND** f.dest = v.dest

) **AS** t**;**



1. Impute os valores não existentes na coluna air\_time de acordo com o maior valor entre air\_time\_projected e air\_time\_expected, utilizando apenas a parte inteira do valor. Garantir que não sobraram linhas sem valor no air\_time. Verificar se os valores ficaram de acordo com a regra, usando o indicador de qualidade correspondente para filtrar.

**sel** air\_time,air\_time\_projected,air\_time\_expectedfrom flights **where** qa\_air\_time **is** **not** **null;**

**ALTER** **TABLE** flights

**ADD** air\_time\_expected float**;**

**insert** **into** flights (air\_time\_expected)

**SELECT** v.avg\_air\_time **AS** air\_time\_expected

**FROM** flights **AS** f

**INNER** **JOIN** vt\_airtime **AS** v

**ON** f.origin = v.origin **AND** f.dest = v.dest**;**

**select** air\_time\_expected **from** flights**;**

**UPDATE** flights

**SET** air\_time = CAST(

**CASE** **WHEN** air\_time\_projected **IS** **NOT** **NULL** **AND** air\_time\_expected **IS** **NOT** **NULL**

**THEN** GREATEST(air\_time\_projected, air\_time\_expected)

**ELSE** COALESCE(air\_time\_projected, air\_time\_expected)

**END** **AS** INTEGER

)

**WHERE** air\_time **IS** **NULL;**

1. Impute os valores não existentes na coluna arr\_time de acordo com a fórmula dep\_time + air\_time, utilizando a nova coluna dep\_datetime e o tipo INTERVAL MINUTE (https://[www.docs.teradata.com/r/Teradata-Database-SQL-Data-Types-and-](http://www.docs.teradata.com/r/Teradata-Database-SQL-Data-Types-and-) Literals/June-2017/DateTime-and-Interval-Data-Types/INTERVAL-MINUTE-Data- Type). Garantir ao final que não sobraram linhas sem valor no arr\_time e que a transformação foi efetiva (pode ser filtrado através do indicador de qualidade).

**alter** **table** flights

**add** arr\_time INTERVAL MINUTE(2)**;**

**update** flights

**set** arr\_time = (dep\_time + air\_time) MINUTE**;**

*--UPDATE fligthsSET arr\_time = dep\_datetime + INTERVAL '1' MINUTE \* air\_time WHERE arr\_time IS NULL;*

**select** COUNT(\*) **from** flights

**where** arr\_time **is** **null;**

1. Crie a coluna haul\_duration com base na coluna air\_time de acordo com as regras abaixo:
   * SHORT: 20 min – 2h59
   * MEDIUM: 3 horas – 5h59
   * LONG: 6+ horas

Verificar frequência do novo campo, validar verificando as faixas mínima e máxima de duração das categorias.

Obs.: recomendado utilizar COMPRESS.

**select** air\_time,

**CASE**

**WHEN** COMPRESS(air\_time) BETWEEN '20' AND '179' **THEN** 'SHORT'

**WHEN** COMPRESS(air\_time) BETWEEN '180' AND '359' **THEN** 'MEDIUM'

**ELSE** 'LONG'

**END** **AS** haul\_duration

**FROM** flights

**;**

**select** haul\_duration, COUNT(\*)

**from**(

**select** air\_time,

**Case**

**WHEN** COMPRESS(air\_time) BETWEEN '20' AND '179' **THEN** 'SHORT'

**WHEN** COMPRESS(air\_time) BETWEEN '180' AND '359' **THEN** 'MEDIUM'

**ELSE** 'LONG'

**END** **AS** haul\_duration

**FROM** flights

) t

**group** **by** haul\_duration

**;**

**select** haul\_duration, MIN(air\_time), MAX(air\_time)

**from**(

**select** air\_time,

**Case**

**WHEN** COMPRESS(air\_time) BETWEEN '20' AND '179' **THEN** 'SHORT'

**WHEN** COMPRESS(air\_time) BETWEEN '180' AND '359' **THEN** 'MEDIUM'

**ELSE** 'LONG'

**END** **AS** haul\_duration

**FROM** flights

) t

**group** **by** haul\_duration

**;**

**select** haul\_duration **from** flights**;**

1. Crie a coluna dep\_season de acordo com as regras abaixo:
   * WINTER : De 21 de Dez às 09:48 PM até 20 de Mar às 03:33 PM.
   * SPRING : De 20 de Mar às 03:34 PM até 21 de Jun às 10:14 AM.
   * SUMMER : De 21 de Jun às 10:15 AM até 23 de Set às 02:04 AM.
   * AUTUMN : De 23 de Set às 02:05 AM até 21 de Dez às 09:47 PM.

Verificar frequência do novo campo, validar verificando as faixas mínimas e máximas de duração das categorias.

Obs.: recomendado utilizar COMPRESS.

**SELECT** \*,

**CASE**

**WHEN** DEPARTURE\_DATE BETWEEN '2022-12-21 09:48:00' **AND** '2023-03-20 15:33:00' **THEN** 'WINTER'

**WHEN** DEPARTURE\_DATE BETWEEN '2023-03-20 15:34:00' **AND** '2023-06-21 10:14:00' **THEN** 'SPRING'

**WHEN** DEPARTURE\_DATE BETWEEN '2023-06-21 10:15:00' **AND** '2023-09-23 02:04:00' **THEN** 'SUMMER'

**WHEN** DEPARTURE\_DATE BETWEEN '2023-09-23 02:05:00' **AND** '2022-12-21 21:47:00' **THEN** 'AUTUMN'

**ELSE** 'OUT OF SEASON'

**END** **AS** DEP\_SEASON

**FROM** flights

**;**

**SELECT** DEP\_SEASON, COUNT(\*) **AS** COUNT

**FROM** (

**SELECT** \*,

**CASE**

**WHEN** DEPARTURE\_DATE BETWEEN '2022-12-21 09:48:00' **AND** '2023-03-20 15:33:00' **THEN** 'WINTER'

**WHEN** DEPARTURE\_DATE BETWEEN '2023-03-20 15:34:00' **AND** '2023-06-21 10:14:00' **THEN** 'SPRING'

**WHEN** DEPARTURE\_DATE BETWEEN '2023-06-21 10:15:00' **AND** '2023-09-23 02:04:00' **THEN** 'SUMMER'

**WHEN** DEPARTURE\_DATE BETWEEN '2023-09-23 02:05:00' **AND** '2022-12-21 21:47:00' **THEN** 'AUTUMN'

**ELSE** 'OUT OF SEASON'

**END** **AS** DEP\_SEASON

**FROM** flights

) **AS** T

**GROUP** **BY** DEP\_SEASON

**;**

**SELECT** DEP\_SEASON **from** flights**;**

1. Crie a coluna dep\_delay\_category com base na coluna dep\_delay de acordo com as regras abaixo:
   * ANTEC : Menor que 0.
   * INTIME : Igual a 0.
   * MINOR : Maior que 0 e menor que 60.
   * MAJOR : Maior ou igual a 60.

Verificar frequência do novo campo, validar verificando as faixas mínimas e máximas de atraso das categorias.

Obs.: recomendado utilizar COMPRESS.

Além das transformações nas 3 tabelas, vamos já deixar criada uma tabela de domínio para o campo carrier (cia aérea), com os valores abaixo:

AA : American Airlines AS : Alaska Airlines B6 : Jetblue Airways DL : Delta Airlines

F9 : Frontier Airlines HA : Hawaiian Airlines OO : Skywest Airlines UA : United Airlines US : US Airways

VX : Virgin America

WN : Southwest Airlines

**alter** **table** flights

**add** dep\_delay\_category char(6) **format** $6.**;**

**update** flights

**set** dep\_delay\_category = compress(

**case**

**when** dep\_delay < 0 **then** 'ANTEC'

**when** dep\_delay = 0 **then** 'INTIME'

**when** dep\_delay > 0 **and** dep\_delay < 60 **then** 'MINOR'

**when** dep\_delay >= 60 **then** 'MAJOR'

**else** ''

**end**

)**;**

**select** dep\_delay\_category **from** flights**;**