92- NOTE: Math operators: +, -, *, /, %, ^.

93- NOTE: Maths functions:
- abs(x)/@ x : absolute value of x.
- round(x,d) : rounded value of x till d decimal places.
- ceiling(x) : round up x.
- floor(x)   : round down x.
- factorial(x): x!
- x & y   : bitwise AND
- x | y   : bitwise OR
- ~x   : bitwise NOT
- x # y   : bitwise XOR
- x << y   : bitwise left shift
- x >> y   : bitwise right shift

94- Display list of films where rental rate of a film is less than 4% of its replacement cost from table(film).

```
SELECT
film_id
,title
,ROUND((rental_rate*100)/replacement_cost,2) as percentage
FROM film
WHERE ((rental_rate*100)/replacement_cost) < 4.00
ORDER BY percentage DESC
```

NOTE: found out that WHERE clause executes before aliases are given so WHERE clause DOES'T work with alias.

95- NOTE: Conditional statement

```
CASE
   WHEN condition_01 THEN result_01
   WHEN condition_02 THEN result_02
   WHEN condition_03 THEN result_03
   ELSE result_04
END
```

=> Priority 01>02>03>04

96- NOTE: You can give alias to Conditional statement (CASE/WHEN) after the 'END' keyword.
97- NOTE: You can group rows (GROUP BY) using alias given to conditional statement (CASE/WHEN).

98- Display flight information using given instructions:

1. Actual departure time not available - 'TBD'
2. Flight is on time or <5 mins late - 'On Schedule'
3. Flight is <= 1 hr late - '1 hr Delay'
4. Flight is >1 hr late - 'Sorry for the inconvenience.'

from table(flights).

```
SELECT
flight_no as "Flight Number"
, departure_airport as "From"
, arrival_airport as "To"
, (actual_departure - scheduled_departure) as "Expected Delay"
,CASE
WHEN actual_departure IS NULL THEN 'TBD'
WHEN (actual_departure - scheduled_departure) <= '00:05:00' THEN 'On Schedule'
```

WHEN (actual_departure - scheduled_departure) <= '01:00:00' THEN 'Upto 1 hr Delay'
ELSE 'Sorry for the inconvenience'
END as "Info"
FROM flights
ORDER BY scheduled_departure

98- Display count of flights in different catgory of delay(GROUP BY) using given instructions:

1. Actual departure time not available - 'TBD'
2. Flight is on time or <5 mins late - 'On Schedule'
3. Flight is <= 1 hr late - '1 hr Delay'
4. Flight is >1 hr late - 'Sorry for the inconvenience.'

from table(flights).

```
SELECT
COUNT(flight_id)
,CASE
WHEN actual_departure IS NULL THEN 'TBD'
WHEN (actual_departure - scheduled_departure) <= '00:05:00' THEN 'On Schedule'
WHEN (actual_departure - scheduled_departure) <= '01:00:00' THEN 'Upto 1 hr Delay'
ELSE 'Sorry for the inconvenience'
END as delay
FROM flights
GROUP BY delay
ORDER BY COUNT(flight_id) DESC
```

99- Find the count of low, mid and high price tickets from table(bookings).

=> Low ticket price: amount<20,000
=> Mid ticket price: amount between 20,000 and 150,000
=> High ticket price: amount greater than 150,000

```
SELECT
COUNT(book_ref)
,CASE
    WHEN total_amount<20000 THEN 'Low Price Ticket'
    WHEN total_amount<150000 THEN 'Mid Price Ticket'
    WHEN total_amount>=150000 THEN 'High Price Ticket'
    ELSE 'Unclassified'
END AS ticket_price
FROM bookings
GROUP BY ticket_price
```

100- Find the count of flights departed in each 4 season from table(flights).

=> Winter: December, January, February
=> Spring: March, April, May
=> Summer: June, July, August
=> Winter: September, October, November

```
SELECT
COUNT(flight_no)
,CASE
WHEN EXTRACT(month FROM scheduled_departure) IN (1,2,12) THEN 'Winter'
WHEN EXTRACT(month FROM scheduled_departure) IN (3,4,5) THEN 'Spring'
WHEN EXTRACT(month FROM scheduled_departure) IN (6,7,8) THEN 'Summer'
WHEN EXTRACT(month FROM scheduled_departure) IN (9,10,11) THEN 'Fall'
ELSE NULL
END AS season
```

```
FROM flights
GROUP BY season
```

101- Divide the film from table(film) in following tier:

=> Great rating or long (tier 1): rating is PG or PG-13 or film length is
greater than 210 mins.
=> Long Drama (tier 2): 'Drama' in description and length>90.
=> Short Drama (tier 3): 'Drama' in description and length<=90.
=> Very cheap (tier 4): Rent rate < $1.

```
SELECT
title
,CASE
    WHEN (rating IN ('PG','PG-13') OR length>210) THEN 'Great rating or long (tier 1)'
    WHEN (description LIKE '%Drama%' AND length>90) THEN 'Long Drama (tier 2)'
    WHEN (description LIKE '%Drama%' AND length<=90) THEN 'Short Drama (tier 3)'
    WHEN (rental_rate<1) THEN 'Very cheap (tier 4)'
    ELSE 'Uncategorized (tier 5)'
END AS tier
FROM film
ORDER BY title
```

102- Make column aliases: PG,PG-13,NC-17,R,G and display count of films under these
rating types from table(film).

```
SELECT
SUM(CASE WHEN rating='PG' THEN 1 ELSE 0 END) as "PG"
,SUM(CASE WHEN rating='PG-13' THEN 1 ELSE 0 END) as "PG-13"
,SUM(CASE WHEN rating='NC-17' THEN 1 ELSE 0 END) as "NC-17"
,SUM(CASE WHEN rating='R' THEN 1 ELSE 0 END) as "R"
,SUM(CASE WHEN rating='G' THEN 1 ELSE 0 END) as "G"
FROM film
```

This type of use of CASE clause is used to pivot/rotate table

103- NOTE: Coalesce (dictionary meaning: to come together to form one larger group, substance, etc.).

104- NOTE COALESCE(column_name_1, column_name_2, ..., ..., '2020-01-01 00:00:00') returns value of column_name_1 if its value IS NOT NULL otherwise column_name_2 value if NOT NULL otherwise .... up until last value '2020-01-01 00:00:00'.

105- Static values / Alternate column names passed in COALESCE function must be of same data type as the value of first most column name passed in COALESCE function.

105- Display flight number and actual departure time for all flights of table(flight), if
actual_departure is NULL for a flight than show scheduled_departure.

```
SELECT
flight_no
,COALESCE(actual_departure,scheduled_departure) as departure_time
FROM flights
ORDER BY departure_time
```

106- NOTE: CAST(something AS data_type): Used to change data type.
(Not every value/column can be changed to other data types.)

105- Display flight number and actual departure time for all flights of table(flight), if
actual_departure is NULL for a flight than a message 'No information'.

```
SELECT
flight_no
,COALESCE(CAST(actual_departure AS VARCHAR),'No Information') as departure_time
FROM flights
ORDER BY flight_no
```

106- Display rental_id, customer_id, rental date, return_date for each rentals from table(rental) use message 'Not Returned' where return_date is NULL.
Order by return_date in descending order, use rental_date in places where return_date is NULL.

```
SELECT
rental_id
,customer_id
,rental_date
,COALESCE(CAST(return_date AS VARCHAR),'Not Returned') return_info
FROM rental
ORDER BY COALESCE(return_date,rental_date) DESC
```

107- NOTE: REPLACE(column, original_text, new_text), replaces all occurences of original_text in given string to new_text.

108- Remove spaces from passenger_id and convert it to BIGINT data type.

```
SELECT
ticket_no
,CAST(REPLACE(passenger_id,' ','') AS BIGINT) as psg_id
FROM tickets
```