# Project – 2
# Group – 3

Step 1:

1.) Schema Design

a) Entities, Attributes and Primary keys:

| S.No. | Entities | Attributes | Primary Key |
|---|---|---|---|
| 1 | Hotel | 1. hotel_id, 2. hotel | hotel_id |
| 2 | Room | 1. room_id, 2. assigned_room_type | room_id |
| 3 | Booking | 1. booking_id, 2. booking_date, 3. lead_time, 4. arrival_date, 5. meal, 6. agent, 7. days_in_waiting_list, 8. required_car_parking_spaces, 9. total_of_special_requests, 10. deposit_type, 11. total_occupants_per_stay, 12. booking_changes, 13. arrival_date_year, 14. arrival_date_month, 15. arrival_date_week_number, 16. arrival_date_day_of_month 17. stays_in_weekend_nights, 18 stays_in_week_nights, | booking_id |
| 4 | Customer | 1. customer_id, 2. adults, 3. children, 4. babies, 5. customer_type, 6. is_repeated_guest, 7. previous_cancellations, 8. previous_bookings_not_canceled,9. total_occupants_per_stay, 10. country | customer_id |
| 5 | Reservation | 1. reservation_id, 2. hotel_id, 3. room_id, 4. is_canceled, 5. reservation_status, 6. reservation_status_date, 7. assigned_room_type, 8. reserved_room_type, 9. market_segment, 10. distribution_channel | reservation_id |
| 6 | Revenue | 1. revenue_id, 2. adr, 3. est_tot_rev_stay, 4. revenue_per_occupant | revenue_id |

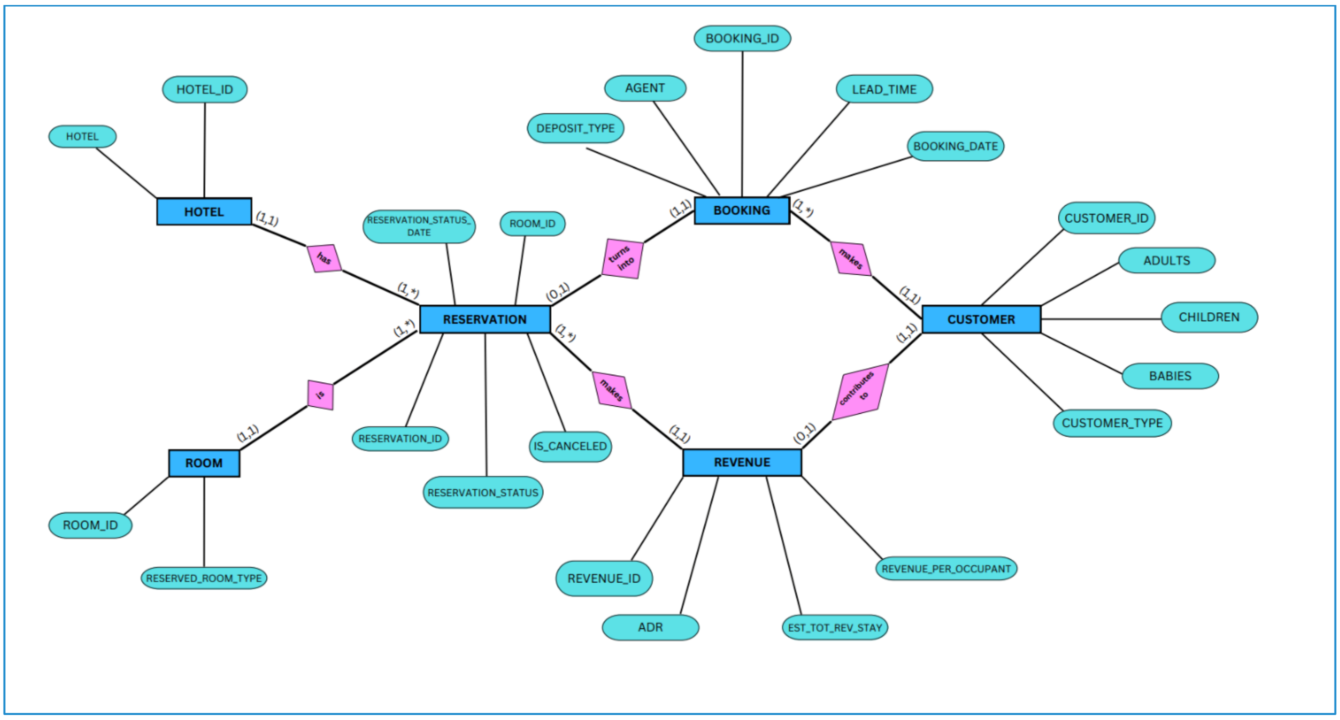
b) Relationships:

- Hotel has reservations.
- Reservations for rooms
- Customers make bookings
- Reservation generate revenue
- Customers pay for booking that makes revenue
- Booking turns into reservation.

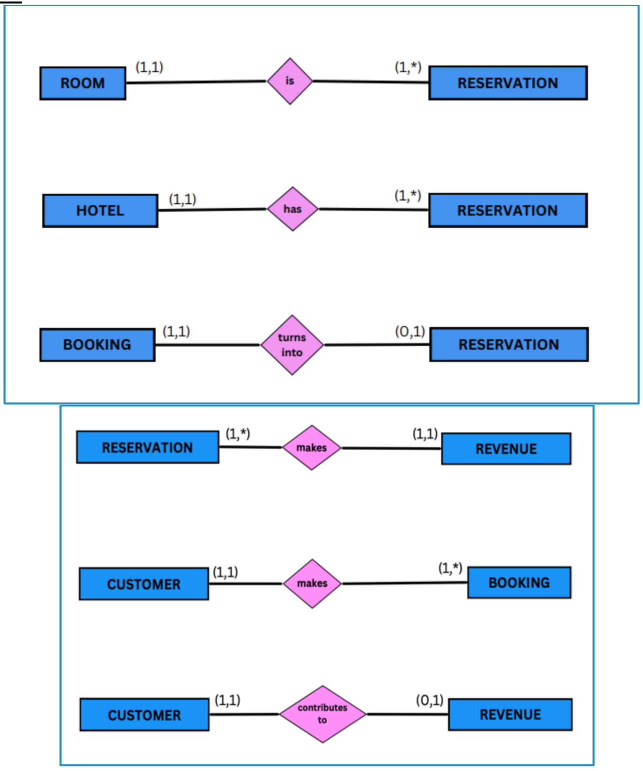
c) Constraints:

- One hotel can have multiple reservations.
- One room can be associated with multiple reservations.
- One customer can have multiple reservations.
- One reservation can have one associated revenue record.
- One customer will only associate to one revenue record.
- One booking can turn into zero or one reservation at most.

d) ER Diagram:



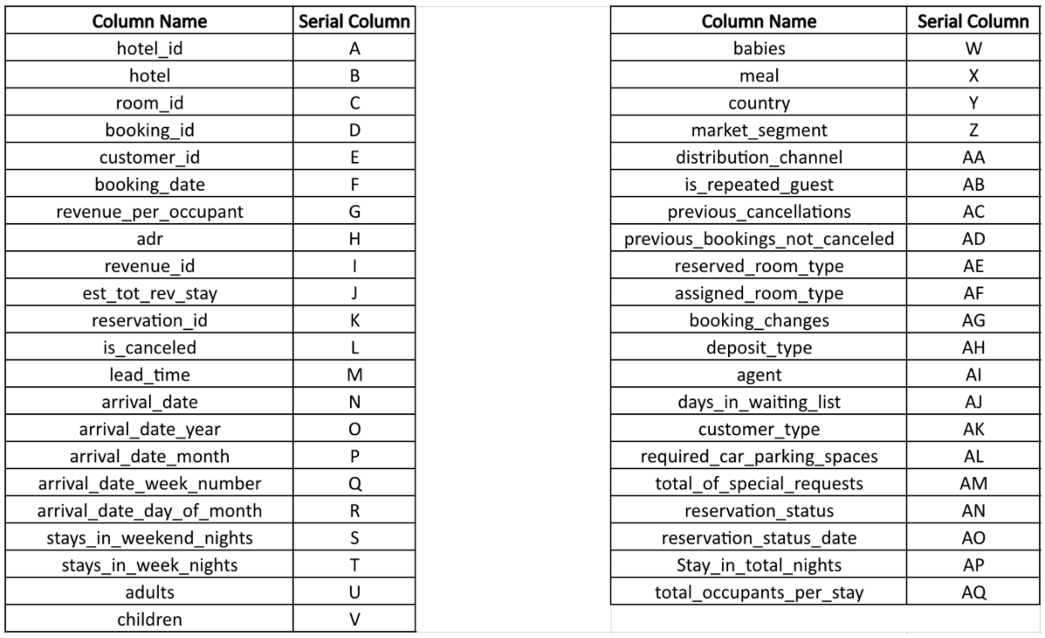
e) Relationships:



2.) Schema Normalization

a) Functional Dependencies:

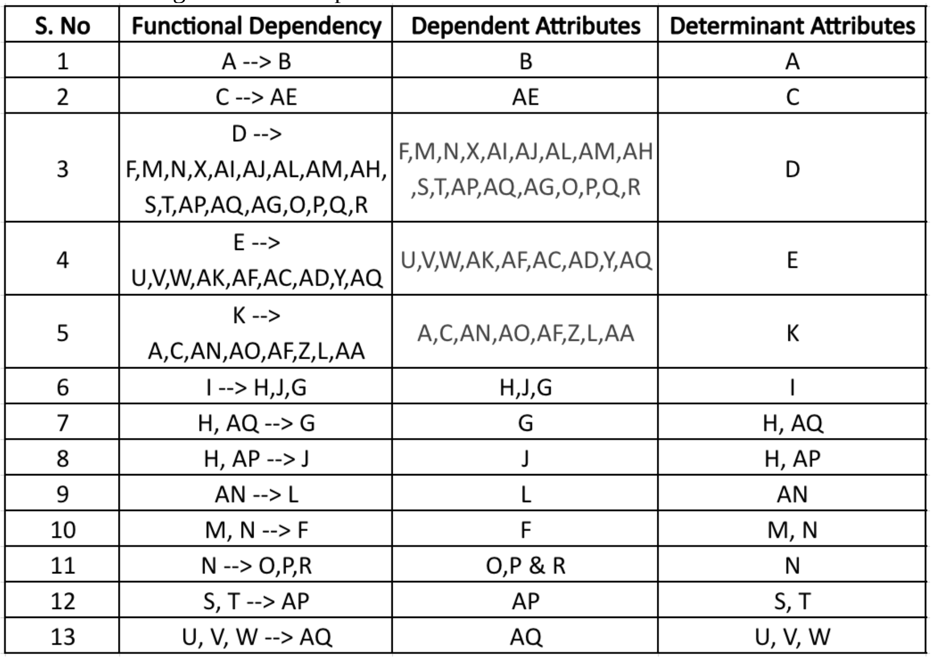
Let us consider the following denotations:



| Column Name | Serial Column |
|---|---|
| hotel_id | A |
| hotel | B |
| room_id | C |
| booking_id | D |
| customer_id | E |
| booking_date | F |
| revenue_per_occupant | G |
| adr | H |
| revenue_id | I |
| est_tot_rev_stay | J |
| reservation_id | K |
| is_canceled | L |
| lead_time | M |
| arrival_date | N |
| arrival_date_year | O |
| arrival_date_month | P |
| arrival_date_week_number | Q |
| arrival_date_day_of_month | R |
| stays_in_weekend_nights | S |
| stays_in_week_nights | T |
| adults | U |
| children | V |

| Column Name | Serial Column |
|---|---|
| babies | W |
| meal | X |
| country | Y |
| market_segment | Z |
| distribution_channel | AA |
| is_repeated_guest | AB |
| previous_cancellations | AC |
| previous_bookings_not_canceled | AD |
| reserved_room_type | AE |
| assigned_room_type | AF |
| booking_changes | AG |
| deposit_type | AH |
| agent | AI |
| days_in_waiting_list | AJ |
| customer_type | AK |
| required_car_parking_spaces | AL |
| total_of_special_requests | AM |
| reservation_status | AN |
| reservation_status_date | AO |
| Stay_in_total_nights | AP |
| total_occupants_per_stay | AQ |

So, we have the following functional dependencies:



| S. No | Functional Dependency | Dependent Attributes | Determinant Attributes |
|---|---|---|---|
| 1 | A --> B | B | A |
| 2 | C --> AE | AE | C |
| 3 | D --> F,M,N,X,AI,AJ,AL,AM,AH,S,T,AP,AQ,AG,O,P,Q,R | F,M,N,X,AI,AJ,AL,AM,AH,S,T,AP,AQ,AG,O,P,Q,R | D |
| 4 | E --> U,V,W,AK,AF,AC,AD,Y,AQ | U,V,W,AK,AF,AC,AD,Y,AQ | E |
| 5 | K --> A,C,AN,AO,AF,Z,L,AA | A,C,AN,AO,AF,Z,L,AA | K |
| 6 | I --> H,J,G | H,J,G | I |
| 7 | H, AQ --> G | G | H, AQ |
| 8 | H, AP --> J | J | H, AP |
| 9 | AN --> L | L | AN |
| 10 | M, N --> F | F | M, N |
| 11 | N --> O,P,R | O,P & R | N |
| 12 | S, T --> AP | AP | S, T |
| 13 | U, V, W --> AQ | AQ | U, V, W |

(b) Assume (A) is a set of all the attributes for the Revenue Table:

K = Revenue_id

K+ = {revenue_id, adr, est_tot_rev_stay, revenue_per_occupant}

Therefore, K+ = A

Hence, K is the primary key for the revenue table.

(c) Since K is a single attribute (an ID attribute) so we can surely prove that we cannot make K any smaller than 1 attribute.
Hence, K which is selected as the primary key be deduced as a minimal key.

Similarly, assume that (A) is a set of all the attributes for each table and K is the primary key for each table respectively.
So, using the above proofs regarding the primary key, we can prove the following findings of the table:

| S.No. | Entities | Primary Key Selected (K) | (b) Is ***K***+ = A ? (A is set of all attributes in the respective entity) | (b) Is condition for primary key satisfied? | (c) Can we remove any sets of attributes from (K) to make a smaller set of attributes for key ? | (c) Hence, is the selected primary key minimal ? |
|---|---|---|---|---|---|---|
| 1 | Hotel | hotel_id | Yes | Yes | No | Yes |
| 2 | Room | room_id | Yes | Yes | No | Yes |
| 3 | Booking | booking_id | Yes | Yes | No | Yes |
| 4 | Guest/Customer | customer_id | Yes | Yes | No | Yes |
| 5 | Reservation | reservation_id | Yes | Yes | No | Yes |
| 6 | Revenue | revenue_id | Yes | Yes | No | Yes |

d) <u>Original Schema design and Analysis for BCNF (Boyce Codd Normal Form)</u> -

Booking:

- booking_id (D)
- lead_time (M)
- arrival_date (N)
- agent (AI)
- days_in_waiting_list (AJ)
- required_car_parking_spaces (AL)
- total_of_special_requests (AM)
- stays_in_weekend_nights (S)
- stays_in_week_nights (T)
- Stay_in_total_nights (AP)
- booking_changes (AG)
- arrival_date_week_number (Q)
- arrival_date_day_of_month (R)
- arrival_date_year (O)
- arrival_date_month (P)
- booking_date (F)

Customer:

- customer_id (E)
- adults (U)
- children (V)
- babies (W)
- customer_type (AK)
- is_repeated_guest (AF)
- previous_cancellations (AC)
- previous_bookings_not_canceled (AD)
- country (Y)
- total_occupants_per_stay (AQ)

Hotel:

- hotel_id (A)
- hotel (B)

Reservation:

- reservation_id (K)
- hotel_id (A)
- room_id (C)
- reservation_status (AN)
- reservation_status_date (AO)
- assigned_room_type (AF)
- market_segment (Z)
- is_canceled (L)
- distribution_channel (AA)

Room:

- room_id (C)
- reserved_room_type (AE)

Revenue:

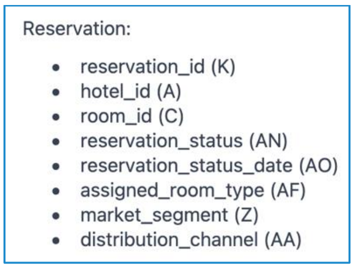
- revenue_id (I)
- adr (H)
- est_tot_rev_stay (J)
- revenue_per_occupant (G)

| S. No | Functional Dependency | Analysis on current schema | Result for BCNF |
|---|---|---|---|
| 1 | A --> B | The determinant A (hotel_id) is a primary key in the Hotel table. | This dependency **satisfies** BCNF. |
| 2 | C --> AE | The determinant C (room_id) is a primary key in the Room table. | This dependency **satisfies** BCNF. |
| 3 | D --> F,M,N,X,AI,AJ,AL,AM,AH,S,T,AP,AQ,AG,O,P,Q,R | The determinant D (booking_id) is a primary key in the Booking table. | This dependency **satisfies** BCNF. |
| 4 | E --> U,V,W,AK,AF,AC,AD,Y,AQ | The determinant E (customer_id) is a primary key in the Customer table. | This dependency **satisfies** BCNF. |
| 5 | K --> A,C,AN,AO,AF,Z,L,AA | The determinant K (reservation_id) is a primary key in the Reservation table. | This dependency **satisfies** BCNF. |
| 6 | I --> H,J,G | The determinant I (revenue_id) is a primary key in the Revenue table. | This dependency **satisfies** BCNF. |
| 7 | H, AQ --> G | The attributes H,AQ & G are not all in the same table. | This dependency **satisfies** BCNF. |
| 8 | H, AP --> J | The attributes H,AP & J are not all in the same table. | This dependency **satisfies** BCNF. |
| 9 | AN --> L | Both AN & L are in the same table and AN is not a key in the Reservation table. | This dependency **violates** BCNF. |
| 10 | M, N --> F | All three of M,N & F are in the same table (Booking table). | This dependency **violates** BCNF. |
| 11 | N --> O, P, R | All five of O,P,Q,R & S are in the same table (Booking table). | This dependency **violates** BCNF. |
| 12 | S, T --> AP | All three of S,T & AP are in the same table (Booking table). | This dependency **violates** BCNF. |
| 13 | U, V, W --> AQ | All four of U,V,W & AQ are in the same table (Customer table). | This dependency **violates** BCNF. |

e) <u>Decomposing the schema:</u>

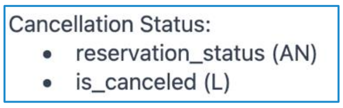
- <u>AN → L:</u>

Decomposing the Reservation table into new tables:


Reservation:

- reservation_id (K)
- hotel_id (A)
- room_id (C)
- reservation_status (AN)
- reservation_status_date (AO)
- assigned_room_type (AF)
- market_segment (Z)
- distribution_channel (AA)

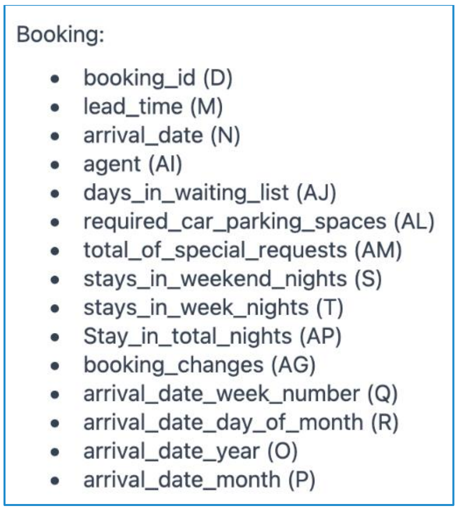
+


Cancellation Status:

- reservation_status (AN)
- is_canceled (L)

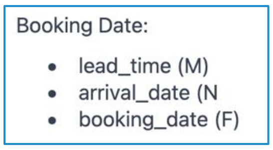
- <u>M, N → F:</u>

  Decomposing the Booking table into new tables:


Booking:

- booking_id (D)
- lead_time (M)
- arrival_date (N)
- agent (AI)
- days_in_waiting_list (AJ)
- required_car_parking_spaces (AL)
- total_of_special_requests (AM)
- stays_in_weekend_nights (S)
- stays_in_week_nights (T)
- Stay_in_total_nights (AP)
- booking_changes (AG)
- arrival_date_week_number (Q)
- arrival_date_day_of_month (R)
- arrival_date_year (O)
- arrival_date_month (P)

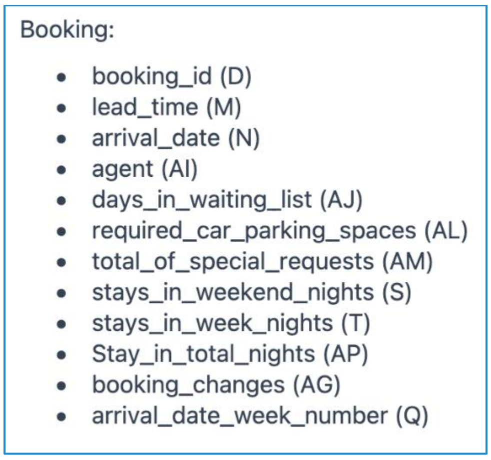
+


Booking Date:

- lead_time (M)
- arrival_date (N
- booking_date (F)

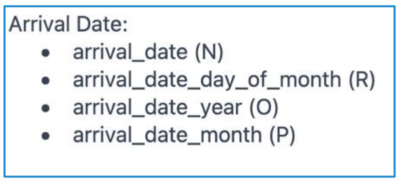
- <u>N → O, P & R:</u>

  Decomposing into new table using the new Booking table:


Booking:

- booking_id (D)
- lead_time (M)
- arrival_date (N)
- agent (AI)
- days_in_waiting_list (AJ)
- required_car_parking_spaces (AL)
- total_of_special_requests (AM)
- stays_in_weekend_nights (S)
- stays_in_week_nights (T)
- Stay_in_total_nights (AP)
- booking_changes (AG)
- arrival_date_week_number (Q)

+


Arrival Date:

- arrival_date (N)
- arrival_date_day_of_month (R)
- arrival_date_year (O)
- arrival_date_month (P)

- <u>S, T → AP:</u>

  Decomposing into new table using the new Booking table:

Booking:

- booking_id (D)
- lead_time (M)
- meal (X)
- arrival_date (N)
- agent (AI)
- days_in_waiting_list (AJ)
- required_car_parking_spaces (AL)
- total_of_special_requests (AM)
- deposit_type (AH)
- stays_in_weekend_nights (S)
- stays_in_week_nights (T)
- total_occupants_per_stay (AQ)
- booking_changes (AG)
- arrival_date_week_number (Q)

+

Total Nights:

- stays_in_weekend_nights (S)
- stays_in_week_nights (T)
- Stay_in_total_nights (AP)

- U, V, W → AQ:

Decomposing the Customer table into two tables:

Customer:

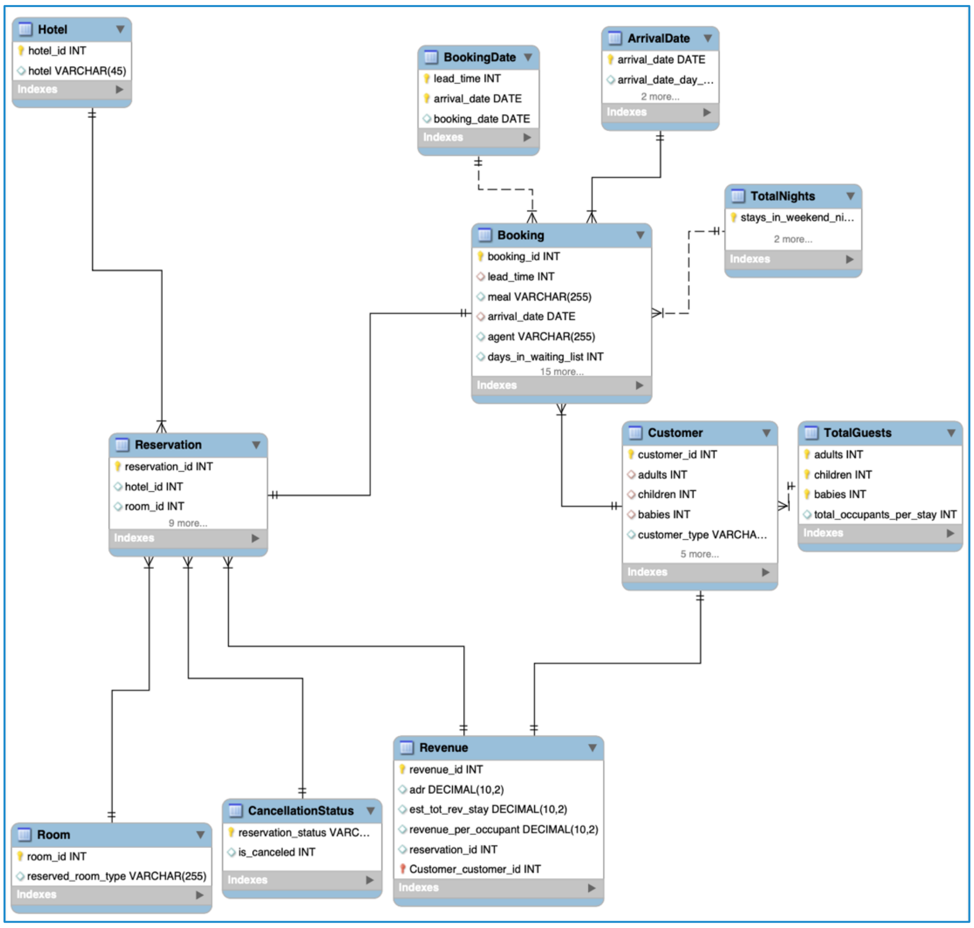
- customer_id (E)
- adults (U)
- children (V)
- babies (W)
- customer_type (AK)
- is_repeated_guest (AF)
- previous_cancellations (AC)
- previous_bookings_not_canceled (AD)
- country (Y)

+

Total Guests:

- adults (U)
- children (V)
- babies (W)
- total_occupants_per_stay (AQ)

f)



3) A database was created using the latest schema.

The tables after removing all functional dependencies that violate BCNF are:

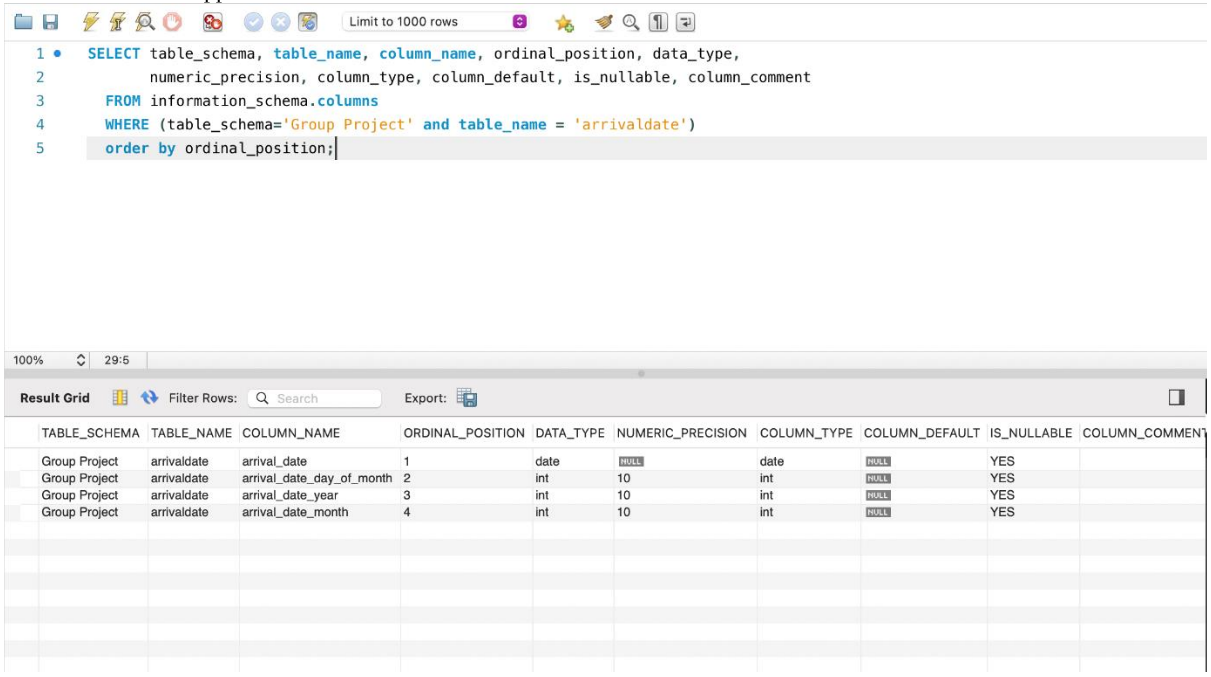
| Serial Number | Table Name | Columns | Primary Key |
|---|---|---|---|
| 1 | Hotel | hotel_id, hotel | hotel_id |
| 2 | Room | room_id, reserved_room_type | room_id |
| 3 | Booking | booking_id, lead_time, meal, arrival_date, agent, days_in_waiting_list, required_car_parking_spaces, total_of_special_requests, deposit_type, stays_in_weekend_nights, stays_in_week_nights, total_occupants_per_stay, booking_changes, arrival_date_week_number | booking_id |
| 4 | Arrival Date | arrival_date, arrival_date_day_of_month, arrival_date_year, arrival_date_month | arrival_date |
| 5 | Total Nights | stays_in_weekend_nights, stays_in_week_nights, Stay_in_total_nights | stays_in_weekend_nights, stays_in_week_nights |
| 6 | Booking Date | lead_time, arrival_date, booking_date | lead_time, arrival_date |
| 7 | Revenue | revenue_id, adr, est_tot_rev_stay, revenue_per_occupant,reservation_id, customer_id | revenue_id |
| 8 | Customer | customer_id, adults, children, babies, customer_type, is_repeated_guest, previous_cancellations, previous_bookings_not_canceled, country, booking_id | customer_id |
| 9 | Total Guests | adults, children, babies, total_occupants_per_stay | adults, children, babies |
| 10 | Reservation | reservation_id, hotel_id, room_id, reservation_status, reservation_status_date, assigned_room_type, market_segment, distribution_channel,booking_id | reservation_id |
| 11 | Cancellation Status | reservation_status, is_canceled | reservation_status |

4) The tables were separated and were saved as separate csv files. These files were then imported into MySQL Workbench using Table Import Wizard and no issues were faced during this process.

**Step 2:**

**Step 2.1: Schema Information**

The below image shows the schema information for the arrivaldate table. The schema information for the rest of the tables can be found in the appendix.



```
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'arrivaldate')
  order by ordinal_position;
```

| TABLE_SCHEMA | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Project | arrivaldate | arrival_date | 1 | date | NULL | date | NULL | YES | |
| Group Project | arrivaldate | arrival_date_day_of_month | 2 | int | 10 | int | NULL | YES | |
| Group Project | arrivaldate | arrival_date_year | 3 | int | 10 | int | NULL | YES | |
| Group Project | arrivaldate | arrival_date_month | 4 | int | 10 | int | NULL | YES | |

## Step 2.2 b, c, d, e

Arrivaldate Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| arrival_date | 793 | 2015-07-01 | 2017-08-31 | 20130 | 20,161,419.4161 | 7,126.659 |
| arrival_date_day_of_month | 793 | 1 | 31 | 30 | 15.7591 | 8.815 |
| arrival_date_year | 793 | 2015 | 2017 | 2 | 2016.0744 | 0.7300 |
| arrival_date_month | 793 | 1 | 12 | 11 | 6.5965 | 3.325 |

Booking Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| booking_id | 119,390 | 10000101 | 10119490 | 119,389 | 10,059,795.500 | 34,464.924 |
| lead_time | 119,390 | 0 | 737 | 737 | 104.0114 | 106.863 |
| arrival_date | 119,390 | 2015-07-01 | 2017-08-31 | 20,130 | 20,162,236.5881 | 6,916.604 |
| agent | 103,050 | 1 | 535 | 534 | 86.6934 | 110.774 |
| days_in_waiting_list | 119,390 | 0 | 391 | 391 | 2.3211 | 17.595 |
| required_car_parking_spaces | 119,390 | 0 | 8 | 8 | 0.0625 | 0.2453 |
| total_of_special_requests | 119,390 | 0 | 5 | 5 | 0.5714 | 0.7928 |
| stays_in_weekend_nights | 119,390 | 0 | 19 | 19 | 0.9276 | 0.9986 |
| stays_in_week_nights | 119,390 | 0 | 50 | 50 | 2.5003 | 1.908 |
| total_occupants_per_stay | 119,386 | 0 | 55 | 55 | 1.9682 | 0.7224 |
| booking_changes | 119,390 | 0 | 21 | 21 | 0.2211 | 0.6523 |
| arrival_date_week_number | 119,390 | 1 | 53 | 52 | 27.1652 | 13.605 |

Bookingdate Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| lead_time | 46,922 | 0 | 737 | 737 | 93.0904 | 84.065 |
| arrival_date | 46,922 | 2015-07-01 | 2017-08-31 | 20,130 | 20,163,157.2530 | 6,655.881 |
| booking_date | 46,922 | 2013-06-24 | 2017-08-31 | 40,207 | 20,161,193.6768 | 6,631.339 |

Cancellationstatus Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| is_canceled | 3 | 0 | 1 | 1 | 0.6667 | 0.4714 |

Customer Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| customer_id | 119,390 | 100001 | 219390 | 119,389 | 159,695.500 | 34,464.924 |
| adults | 119,390 | 0 | 55 | 55 | 1.8564 | 0.579 |
| children | 119,386 | 0 | 10 | 10 | 0.1039 | 0.399 |
| babies | 119,390 | 0 | 10 | 10 | 0.0079 | 0.0974 |
| is_repeated_guest | 119,390 | 0 | 1 | 1 | 0.0319 | 0.1758 |
| previous_cancellations | 119,390 | 0 | 26 | 26 | 0.0871 | 0.8441 |
| previous_bookings_not_canceled | 119,390 | 0 | 72 | 72 | 0.1371 | 1.497 |
| booking_id | 119,390 | 10000101 | 10119490 | 119,389 | 10,059,795.5000 | 34,464.924 |

Hotel Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| hotel_id | 2 | 1 | 2 | 1 | 1.5000 | 0.5 |

Reservation Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| reservation_id | 119,390 | 1000001 | 1119390 | 119,389 | 1,059,695.50 | 34,464.924 |
| hotel_id | 119,390 | 1 | 2 | 1 | 1.3355 | 0.4722 |
| room_id | 119,390 | 1 | 10 | 9 | 1.9895 | 1.697 |
| reservation_status_date | 119,390 | 2014-10-17 | 2017-09-14 | 29897 | 20,161,586.5109 | 6,994.239 |
| booking_id | 119,390 | 10000101 | 10119490 | 119,389 | 10,059,795.5000 | 34,464.924 |

Revenue Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| revenue_id | 119,359 | 00000101-03 | 00119490-01 | 119,389 | 59,786.8874 | 34,463.441 |
| adr | 119,359 | -6.38 | 5,400 | 5,406.38 | 101.8418 | 50.531 |
| est_tot_rev_stay | 119,359 | -63.8 | 7,590 | 7,653.8 | 357.8628 | 335.933 |
| revemue_per_occupant | 119,206 | -3.19 | 2,700 | 2,703.19 | 54.9903 | 29.058 |
| reservation_id | 119,359 | 1000001 | 1119390 | 119,389 | 1,059,686.8874 | 34,463.441 |
| customer_id | 119,359 | 100001 | 219390 | 119,389 | 159,686.8874 | 34,463.441 |

Room Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| room_id | 10 | 1 | 10 | 9 | 5.5000 | 2.872 |

Totalguests Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| adults | 43 | 0 | 55 | 55 | 7.0000 | 12.888 |
| children | 41 | 0 | 10 | 10 | 1.0244 | 1.718 |
| babies | 43 | 0 | 10 | 10 | 0.7674 | 2.010 |
| total_occupants_per_stay | 41 | 0 | 55 | 55 | 9.0488 | 12.597 |

Totalnights Table (2.2 b, c, d, e):

| Numeric Column Name | # observations in column | Min of column | Max of column | Range of the column | Mean of column | Standard deviation of column |
|---|---|---|---|---|---|---|
| stays_in_weekend_nights | 85 | 0 | 19 | 19 | 5.7176 | 4.335 |
| stays_in_week_nights | 85 | 0 | 50 | 50 | 14.2235 | 10.686 |
| stays_in_total_nights | 85 | 0 | 69 | 69 | 19.9412 | 14.937 |

## Step 2.2 f, g

**Bookingtable(f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| booking_id | 10,000,101 | 1 | 10029948 | 10059795.5 | 10089642 | 10113520 | 0 | 0 | 119390 |
| lead_time | 0 | 6345 | 18 | 69 | 160 | 320 | 0 | 0 | 119390 |
| arrival_date | 2015-12-05 | 448 | 2016-03-13 | 2016-09-06 | 2017-03-18 | 2017-07-25 | 0 | 0 | 119390 |
| agent | 9 | 31961 | 14 | 250 | 9 | 9 | 16340 | 0.1369 | 103050 |
| days_in_waiting_list | 0 | 115692 | 0 | 0 | 0 | 0 | 0 | 0 | 119390 |
| required_car_parking_spaces | 0 | 111974 | 0 | 0 | 0 | 1 | 0 | 0 | 119390 |
| total_of_special_requests | 0 | 70318 | 0 | 0 | 1 | 2 | 0 | 0 | 119390 |
| stays_in_weekend_nights | 0 | 51998 | 0 | 1 | 2 | 2 | 0 | 0 | 119390 |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| stays_in_week_nights | 2 | 33684 | 1 | 2 | 3 | 5 | 0 | 0 | 119390 |
| total_occupants_per_stay | 2 | 82048 | 2 | 2 | 2 | 3 | 4 | 0 | 119386 |
| booking_changes | 0 | 101314 | 0 | 0 | 0 | 1 | 0 | 0 | 119390 |
| arrival_date_week_number | 33 | 3580 | 16 | 28 | 38 | 49 | 0 | 0 | 119390 |

**<u>Bookingdatetable</u> (f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| lead_time | 0 | 778 | 25 | 69 | 142 | 260 | 0 | 0 | 469222 |
| arrival_date | 2016-08-08 | 123 | 2016-04-28 | 2016-10-04 | 2017-04-20 | 2017-08-06 | 0 | 0 | 469222 |
| booking_date | 2016-01-18 | 159 | 2016-01-24 | 2016-07-12 | 2017-01-05 | 2017-05-18 | 0 | 0 | 469222 |

**Cancellationstatus Table (f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| is_canceled | 1 | 2 | 0 | 1 | 1 | 1 | 0 | 0 | 3 |

**<u>Customertable</u> (f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| customer_id | 100,001 | 1 | 129848 | 159695 | 189542 | 213420 | 0 | 0 | 119390 |
| adults | 2 | 89680 | 2 | 2 | 2 | 3 | 0 | 0 | 119390 |
| children | 0 | 110796 | 0 | 0 | 0 | 1 | 4 | 0 | 119386 |
| babies | 0 | 118473 | 0 | 0 | 0 | 0 | 0 | 0 | 119390 |
| is_repeated_guest | 0 | 115580 | 0 | 0 | 0 | 0 | 0 | 0 | 119390 |
| previous_cancellations | 0 | 112906 | 0 | 0 | 0 | 0 | 0 | 0 | 119390 |
| previous_bookings_not_canceled | 0 | 115770 | 0 | 0 | 0 | 0 | 0 | 0 | 119390 |
| booking_id | 10,000,101 | 1 | 10029948 | 10059795 | 10089642 | 10113520 | 0 | 0 | 119390 |

**<u>Hoteltable(f,g)</u>**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| hotel_id | 2 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 2 |

**Reservationtable(f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| reservation_id | 1,000,001 | 1 | 1029848 | 1059695 | 1089542 | 1113420 | 0 | 0 | 119390 |
| hotel_id | 1 | 79330 | 1 | 1 | 2 | 2 | 0 | 0 | 119390 |
| room_id | 1 | 85994 | 1 | 1 | 4 | 5 | 0 | 0 | 119390 |
| reservation_status_date | 2015-10-21 | 1,461.00 | 2016-02-01 | 2016-08-07 | 2017-02-08 | 2017-07-14 | 0 | 0 | 119390 |
| booking_id | 10,000,101 | 1 | 10029948 | 10059795 | 10089642 | 10113520 | 0 | 0 | 119390 |

**Revenuetable (f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| revenue_id | 00000101-03 | 1 | 00027048-01 | 00054000-01 | 00080950-01 | 00102522-01 | 0 | 0 | 107792 |
| adr | 62 | 3754 | 66 | 90 | 121 | 191.14 | 0 | 0 | 107792 |
| est_tot_rev_stay | 124 | 2908 | 139 | 258.68 | 438.6 | 1020 | 0 | 0 | 107792 |
| revenue_per_occupant | 31 | 3724 | 36 | 47.6 | 65 | 109.8 | 131 | 0.0012 | 107661 |
| reservation_id | 1,000,001 | 1 | 1026948 | 1053900 | 1080850 | 1102422 | 0 | 0 | 107792 |
| customer_id | 100,001 | 1 | 126948 | 153900 | 180850 | 202422 | 0 | 0 | 107792 |

**Roomtable(f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| room_id | 3 | 1 | 3 | 5 | 7 | 9 | 0 | 0 | 10 |

**Totalgueststable (f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| adults | 2 | 12 | 1 | 2 | 4 | 25 | 0 | 0 | 43 |
| children | 0 | 21 | 0 | 0 | 1 | 3 | 2 | 0.0465 | 41 |
| babies | 0 | 29 | 0 | 0 | 1 | 2 | 0 | 0 | 43 |
| total_occupants_per_stay | 4 | 8 | 3 | 4 | 5 | 27 | 2 | 0.0465 | 41 |

**Totalnightstable(f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| stays_in_weekend_nights | 2 | 11 | 2 | 5 | 8 | 14 | 0 | 0 | 85 |
| stays_in_week_nights | 5 | 5 | 6 | 12 | 19 | 34 | 0 | 0 | 85 |
| stays_in_total_nights | 2 | 3 | 9 | 17 | 27 | 48 | 0 | 0 | 85 |

**Arrivaldatetable (f,g)**

| Numeric Column Name | Mode | Mode Frequency | 25% | 50% | 75% | 95% | Missing qty | Rate | non null count |
|---|---|---|---|---|---|---|---|---|---|
| arrival_date | 2015-07-01 | 1 | 2016-01-15 | 2016-07-31 | 2017-02-14 | 2017-07-22 | 0 | 0 | 793 |
| arrival_date_day_of_month | 1 | 26 | 8 | 16 | 23 | 29 | 0 | 0 | 793 |
| arrival_date_year | 2016 | 366 | 2016 | 2016 | 2017 | 2017 | 0 | 0 | 793 |
| arrival_date_month | 7 | 93 | 4 | 7 | 9 | 12 | 0 | 0 | 793 |

**2.2 h**
All the values calculated and found in the above steps (b,c,d,e,f,g) match with project 1 for all our numeric tables and there were no discrepancies found.

## Step 2.3 a, b, c, d

Hotel Table (2.3 a, b, c, d):

Hotel column (derived from hotel table and reservation table)

A)

| hotel | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_Frequency |
|---|---|---|---|---|
| Resort Hotel | 40060 | 34% | 40060 | 34% |
| City Hotel | 79330 | 66% | 119390 | 100% |

B)

The highest frequency hotel is a city hotell the lowest frequency hotel is the resort hotel

C)

There are no missing values in this column

D)

These values match the values from project 1

ReservationTable (2.3 a, b, c, d):

Reservation_Status column

A)

| Reservation_status | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_Frequency |
|---|---|---|---|---|
| No-Show | 1207 | 1% | 1207 | 1% |
| Canceled | 43017 | 36% | 44224 | 37% |
| Check-Out | 75166 | 63% | 119390 | 100% |

B)

The highest frequency reservation status is checkout with 75166 values. The lowest frequency reservation status is no show with only 1207 values.

C)

There are no missing values in this column.

D)

Values are the same as in project 1.

Assigned_Room_Type column

A)

| Assigned_room_type | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_Frequency |
|---|---|---|---|---|
| L | 1 | 0% | 1 | 0% |
| P | 12 | 0% | 13 | 0% |
| K | 279 | 0% | 292 | 0% |
| I | 363 | 0% | 655 | 1% |
| H | 712 | 1% | 1367 | 1% |
| B | 2163 | 2% | 3530 | 3% |
| C | 2375 | 2% | 5905 | 5% |
| G | 2553 | 2% | 8458 | 7% |
| F | 3751 | 3% | 12209 | 10% |
| E | 7806 | 7% | 20015 | 17% |
| D | 25322 | 21% | 45337 | 38% |
| A | 74053 | 62% | 119390 | 100% |

B)

The highest frequency room type in this table is room type A, with 74053 values. The least frequent room type is type L, with 1 value.

C)

There are no missing values in this table

D)
These values are the same as in project 1. This column is being used instead of reserved room type column which was used in project 1 which had two columns, one for reserved room type and one for assigned room type.

Market_Segment column

A)

| Market_segment | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_Frequency |
|---|---|---|---|---|
| Undefined | 2 | 0% | 2 | 0% |
| Aviation | 237 | 0% | 239 | 0% |
| Complementary | 743 | 1% | 982 | 1% |
| Corporate | 5295 | 4% | 6277 | 5% |
| Direct | 12606 | 11% | 18883 | 16% |
| Groups | 19811 | 17% | 38694 | 32% |
| Offline TA/TO | 24219 | 20% | 62913 | 53% |
| Online TA | 56477 | 47% | 119390 | 100% |

B)

The highest frequency market segment is the Online TA segment with 56477 values; the lowest frequency market segment is the undefined segment, with only 2 values.

C)

There are no missing values in this table.

D)

These values are the same as in project 1.

Distribution_Channel column

A)

| Distribution_channel | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_ Frequency |
|---|---|---|---|---|
| Undefined | 5 | 0% | 5 | 0% |
| GDS | 193 | 0% | 198 | 0% |
| Corporate | 6677 | 6% | 6875 | 6% |
| Direct | 14645 | 12% | 21520 | 18% |
| TA/TO | 97870 | 82% | 119390 | 100% |

B)

The highest frequency distribution channel is TA/TO with 97870 values; the lowest frequency distribution channel is undefined with 5 values.

C)

There are no missing values in this table.

D)

These values are the same as in project 1.

CustomerTable (2.3 a, b, c, d):

Customer_Type column

A)

| Customer_type | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_ Frequency |
|---|---|---|---|---|
| Group | 577 | 0% | 577 | 0% |
| Contract | 4076 | 3% | 4653 | 4% |
| Transient-Party | 25124 | 21% | 29777 | 25% |
| Transient | 89613 | 75% | 119390 | 100% |

B)

The highest frequency customer type is transient with 89613 values; the lowest frequency customer type is group, with only 577 values.

C)

There are no missing values in this column.

D)

These values are the same as in project 1.

Is_Repeated_Guest Column

A)

| Is_repeated_guest | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_ Frequency |
|---|---|---|---|---|
| 1 | 3810 | 3% | 3810 | 3% |
| 0 | 115580 | 97% | 119390 | 100% |

B)

The highest frequency occurrence in this column is 0, or not a repeated guest, with 115580 values whereas the lowest frequency value is yes, with 3810 occurences.

C)
There are no missing values in this column

D)
These values are the same as in project 1.

Country Column

A)

| Country | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_ Frequency |
|---|---|---|---|---|
| PRT | 48590 | 41% | 119390 | 100% |
| GBR | 12129 | 10% | 70800 | 59% |
| FRA | 10415 | 9% | 58671 | 49% |
| ESP | 8568 | 7% | 48256 | 40% |
| DEU | 7287 | 6% | 39688 | 33% |
| ITA | 3766 | 3% | 32401 | 27% |
| IRL | 3375 | 3% | 28635 | 24% |
| BEL | 2342 | 2% | 25260 | 21% |
| BRA | 2224 | 2% | 22918 | 19% |
| NLD | 2104 | 2% | 20694 | 17% |
| Other | 18590 | 16% | 18590 | 16% |

B)

The highest frequency country in this column is PRT; whereas the lowest frequency country is a tie between 30 different countries each having 1 observation in this column.

C)

There are 488 missing values in this column which are labeled as null in the dataset.

D)
These values match those of project 1.

BookingTable (2.3 a, b, c, d):

Arrival_Month (derived from arrival date table via join to booking table)

A)

| Arrival_date_month | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_ Frequency |
|---|---|---|---|---|
| 1 | 5929 | 5% | 5929 | 5% |
| 12 | 6780 | 6% | 12709 | 11% |
| 11 | 6794 | 6% | 19503 | 16% |

| | | | | |
|---|---|---|---|---|
| 2 | 8068 | 7% | 27571 | 23% |
| 3 | 9794 | 8% | 37365 | 31% |
| 9 | 10508 | 9% | 47873 | 40% |
| 6 | 10939 | 9% | 58812 | 49% |
| 4 | 11089 | 9% | 69901 | 59% |
| 10 | 11160 | 9% | 81061 | 68% |
| 5 | 11791 | 10% | 92852 | 78% |
| 7 | 12661 | 11% | 105513 | 88% |
| 8 | 13877 | 12% | 119390 | 100% |

B)
The highest frequency month is August with 13877 values; the lowest frequency month is January, with 5929 values.

C)
There are no missing values in this derived column

D)
There is no difference between these values and the values in project 1

Meal Column

A)

| Meal | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_ Frequency |
|---|---|---|---|---|
| FB | 798 | 1% | 798 | 1% |
| Undefined | 1169 | 1% | 1967 | 2% |
| SC | 10650 | 9% | 12617 | 11% |
| HB | 14463 | 12% | 27080 | 23% |
| BB | 92310 | 77% | 119390 | 100% |

B)

The highest frequency meal is BB with 92310 values; the lowest frequency meal is FB with only 798 values.

C)

There are no missing values in this column.

D)

These values match the values of those in project 1.

Agent Column

A)

| Agent | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_ Frequency |
|---|---|---|---|---|
| 9 | 31961 | 27% | 119390 | 100% |
| Missing | 16340 | 14% | 87429 | 73% |
| 240 | 13922 | 12% | 71089 | 60% |
| 1 | 7191 | 6% | 57167 | 48% |
| 14 | 3640 | 3% | 49976 | 42% |
| 7 | 3539 | 3% | 46336 | 39% |
| 6 | 3290 | 3% | 42797 | 36% |
| 250 | 2870 | 2% | 39507 | 33% |
| 241 | 1721 | 1% | 36637 | 31% |

| | | | | |
|---|---|---|---|---|
| 28 | 1666 | 1% | 34916 | 29% |
| Other | 33250 | 28% | 33250 | 28% |

B)

The highest frequency agent is agent 9 with a total of 31961 values; the lowest frequency agent is a tie between 50 different agents with the value 1.

C)

There were 16340 missing values in this column.

D)
These values match the values in project 1.

Deposit_Type Column

A)

| Deposit_type | Frequency | Frequency_Percentile | Cumulative_Total | Cumulative_Percent_Frequency |
|---|---|---|---|---|
| Refundable | 162 | 0% | 162 | 0% |
| Non Refund | 14587 | 12% | 14749 | 12% |
| No Deposit | 104641 | 88% | 119390 | 100% |

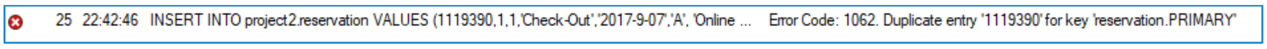
B)

The highest frequency deposit type is No Deposit with 104641 values whereas the lowest frequency deposit type is refundable with 162 values.

C)

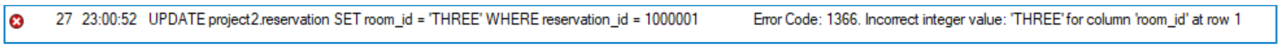
There are no missing values in this column

D)
These values match the values in project 1
See the appendix for SQL commands for each step.
**Step 2.4 a:**
When attempting to add a duplicate row to the revenue table, SQL delivered the following error. Note that reservation_id is the primary key for this table.



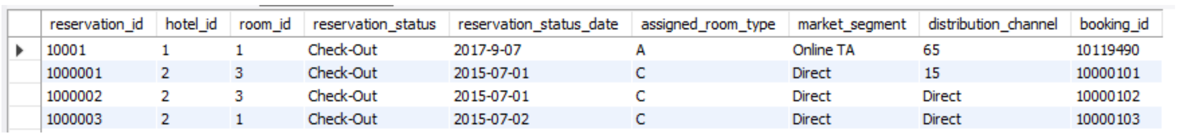
```
25  22:42:46  INSERT INTO project2.reservation VALUES (1119390,1,1,'Check-Out','2017-9-07','A', 'Online ...    Error Code: 1062. Duplicate entry '1119390' for key 'reservation.PRIMARY'
```

**Step 2.4 b:**
When attempting to add a character value to a numerical column in the reservation table, SQL delivered the following error.



```
27  23:00:52  UPDATE project2.reservation SET room_id = 'THREE' WHERE reservation_id = 1000001    Error Code: 1366. Incorrect integer value: 'THREE' for column 'room_id' at row 1
```

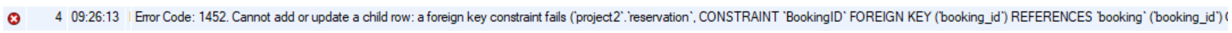
**Step 2.4 c:**
SQL did allow a numerical value to be added in the distribution_channel column, which is set to the variable type 'text'. Two methods were used to test adding the numeric value; by adding a new row and by modifying an existing row. Here are the first few rows of the reservation table showing that numerical values were added to the character column.



| reservation_id | hotel_id | room_id | reservation_status | reservation_status_date | assigned_room_type | market_segment | distribution_channel | booking_id |
|---|---|---|---|---|---|---|---|---|
| 10001 | 1 | 1 | Check-Out | 2017-9-07 | A | Online TA | 65 | 10119490 |
| 1000001 | 2 | 3 | Check-Out | 2015-07-01 | C | Direct | 15 | 10000101 |
| 1000002 | 2 | 3 | Check-Out | 2015-07-01 | C | Direct | Direct | 10000102 |
| 1000003 | 2 | 1 | Check-Out | 2015-07-02 | C | Direct | Direct | 10000103 |

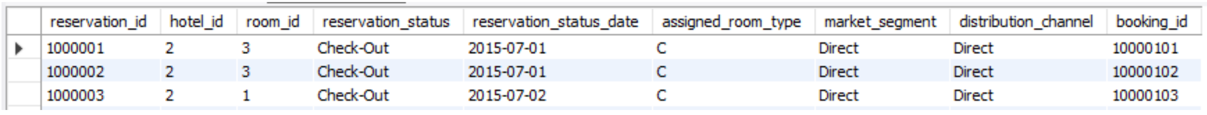
**Step 2.4 d:**
The booking_id column in the reservation table is a foreign key. When testing this column, the following results occurred.

<u>Add a value:</u>
Attempted to add a row with the value 10000100 in the booking_id column (and with a unique value in the primary key column).
Result: Error code 1452.



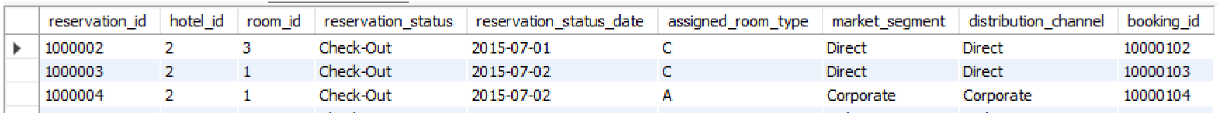
```
4 09:26:13 Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('project2'.'reservation', CONSTRAINT 'BookingID' FOREIGN KEY ('booking_id') REFERENCES 'booking' ('booking_id') (
```

<u>Remove a value:</u>
Attempted to delete a row with the value 10000101 in the booking_id column.
Result: SQL deleted the row.
Before the remove command:



| reservation_id | hotel_id | room_id | reservation_status | reservation_status_date | assigned_room_type | market_segment | distribution_channel | booking_id |
|---|---|---|---|---|---|---|---|---|
| 1000001 | 2 | 3 | Check-Out | 2015-07-01 | C | Direct | Direct | 10000101 |
| 1000002 | 2 | 3 | Check-Out | 2015-07-01 | C | Direct | Direct | 10000102 |
| 1000003 | 2 | 1 | Check-Out | 2015-07-02 | C | Direct | Direct | 10000103 |

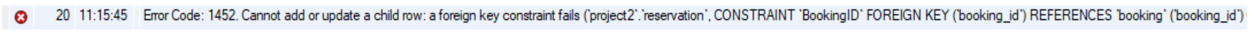
After the remove command:



| reservation_id | hotel_id | room_id | reservation_status | reservation_status_date | assigned_room_type | market_segment | distribution_channel | booking_id |
|---|---|---|---|---|---|---|---|---|
| 1000002 | 2 | 3 | Check-Out | 2015-07-01 | C | Direct | Direct | 10000102 |
| 1000003 | 2 | 1 | Check-Out | 2015-07-02 | C | Direct | Direct | 10000103 |
| 1000004 | 2 | 1 | Check-Out | 2015-07-02 | A | Corporate | Corporate | 10000104 |

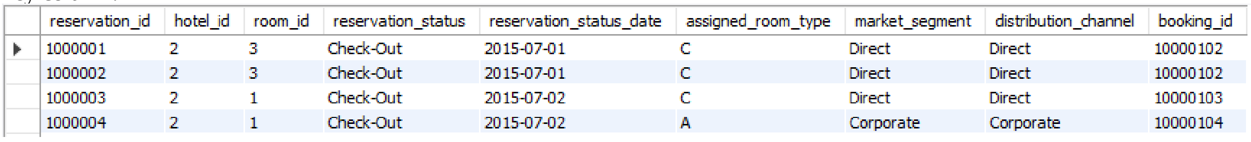
After the row was deleted, it was added back to the table. This indicates that SQL will allow values to be added to the foreign key column if those values exist in the reference column of the parent table.

<u>Change a value:</u>
Attempted to change the value of an entry in the booking_id column to a value that <u>does not</u> exist in the parent table reference column.
Result: Error code 1452



```
20 11:15:45 Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('project2'.'reservation', CONSTRAINT 'BookingID' FOREIGN KEY ('booking_id') REFERENCES 'booking' ('booking_id')
```

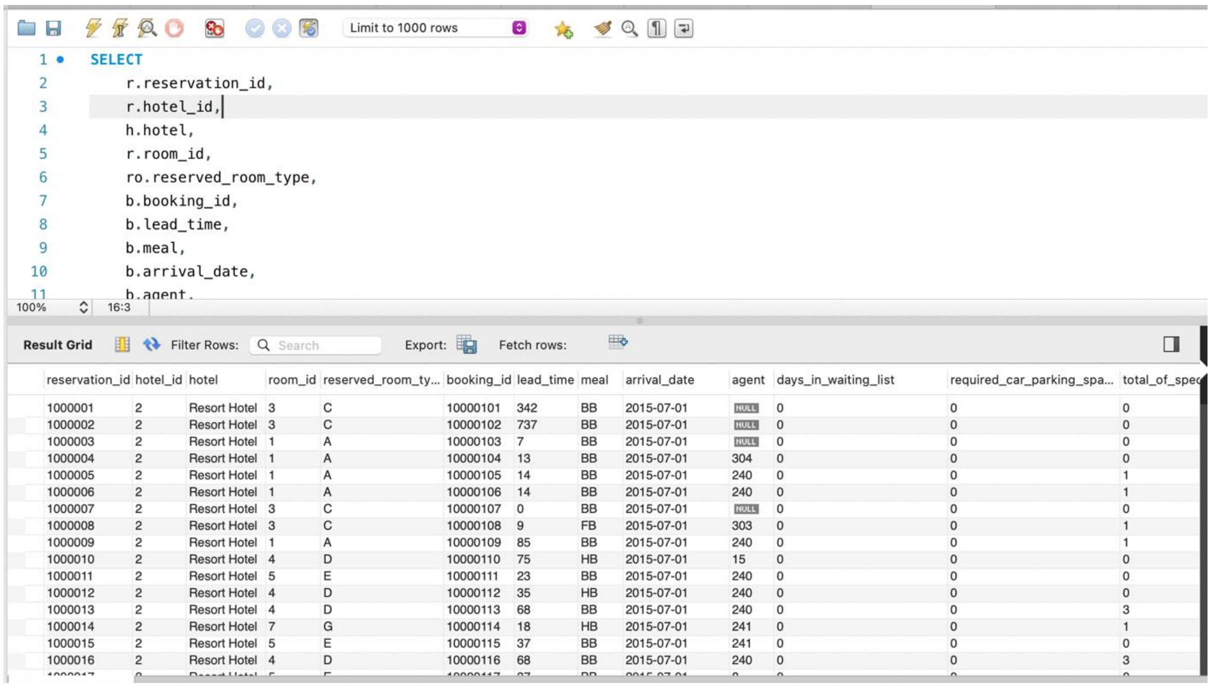
Then, attempted to change the value of an entry in the booking_id column to a value that <u>does</u> exist in the parent table reference column.
Result: SQL allowed the change to occur, even though there are now 2 rows with the same value in the booking_id foreign key column.



| reservation_id | hotel_id | room_id | reservation_status | reservation_status_date | assigned_room_type | market_segment | distribution_channel | booking_id |
|---|---|---|---|---|---|---|---|---|
| 1000001 | 2 | 3 | Check-Out | 2015-07-01 | C | Direct | Direct | 10000102 |
| 1000002 | 2 | 3 | Check-Out | 2015-07-01 | C | Direct | Direct | 10000102 |
| 1000003 | 2 | 1 | Check-Out | 2015-07-02 | C | Direct | Direct | 10000103 |
| 1000004 | 2 | 1 | Check-Out | 2015-07-02 | A | Corporate | Corporate | 10000104 |

**<u>Step 2.4 e:</u>**
We used a few SQL commands to rejoin the 11 tables which resulted in the original 43 columns. Below are the first 10 rows of the completed dataset.

The entirety of the SQL code to return these results can be found in the Appendix.



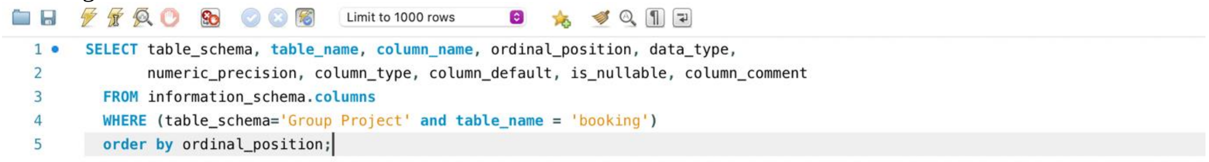
```
SELECT
    r.reservation_id,
    r.hotel_id,
    h.hotel,
    r.room_id,
    ro.reserved_room_type,
    b.booking_id,
    b.lead_time,
    b.meal,
    b.arrival_date,
    b.agent,
```

| reservation_id | hotel_id | hotel | room_id | reserved_room_ty... | booking_id | lead_time | meal | arrival_date | agent | days_in_waiting_list | required_car_parking_spa... | total_of_spec |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1000001 | 2 | Resort Hotel | 3 | C | 10000101 | 342 | BB | 2015-07-01 | NULL | 0 | 0 | 0 |
| 1000002 | 2 | Resort Hotel | 3 | C | 10000102 | 737 | BB | 2015-07-01 | NULL | 0 | 0 | 0 |
| 1000003 | 2 | Resort Hotel | 1 | A | 10000103 | 7 | BB | 2015-07-01 | NULL | 0 | 0 | 0 |
| 1000004 | 2 | Resort Hotel | 1 | A | 10000104 | 13 | BB | 2015-07-01 | 304 | 0 | 0 | 0 |
| 1000005 | 2 | Resort Hotel | 1 | A | 10000105 | 14 | BB | 2015-07-01 | 240 | 0 | 0 | 1 |
| 1000006 | 2 | Resort Hotel | 1 | A | 10000106 | 14 | BB | 2015-07-01 | 240 | 0 | 0 | 1 |
| 1000007 | 2 | Resort Hotel | 3 | C | 10000107 | 0 | BB | 2015-07-01 | NULL | 0 | 0 | 0 |
| 1000008 | 2 | Resort Hotel | 3 | C | 10000108 | 9 | FB | 2015-07-01 | 303 | 0 | 0 | 1 |
| 1000009 | 2 | Resort Hotel | 1 | A | 10000109 | 85 | BB | 2015-07-01 | 240 | 0 | 0 | 1 |
| 1000010 | 2 | Resort Hotel | 4 | D | 10000110 | 75 | HB | 2015-07-01 | 15 | 0 | 0 | 0 |
| 1000011 | 2 | Resort Hotel | 5 | E | 10000111 | 23 | BB | 2015-07-01 | 240 | 0 | 0 | 0 |
| 1000012 | 2 | Resort Hotel | 4 | D | 10000112 | 35 | HB | 2015-07-01 | 240 | 0 | 0 | 0 |
| 1000013 | 2 | Resort Hotel | 4 | D | 10000113 | 68 | BB | 2015-07-01 | 240 | 0 | 0 | 3 |
| 1000014 | 2 | Resort Hotel | 7 | G | 10000114 | 18 | HB | 2015-07-01 | 241 | 0 | 0 | 1 |
| 1000015 | 2 | Resort Hotel | 5 | E | 10000115 | 37 | BB | 2015-07-01 | 241 | 0 | 0 | 0 |
| 1000016 | 2 | Resort Hotel | 4 | D | 10000116 | 68 | BB | 2015-07-01 | 240 | 0 | 0 | 3 |

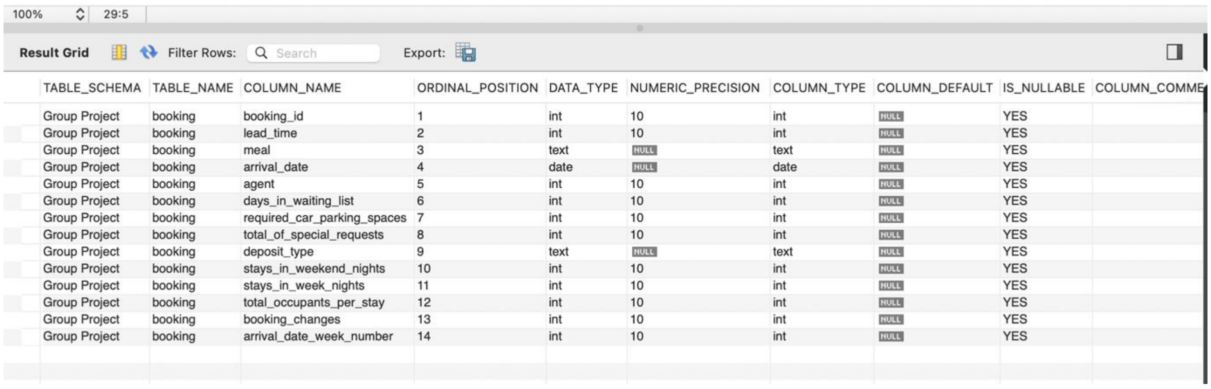
# Appendix

## 2.1 Schema Information Cont’d

Booking Table:

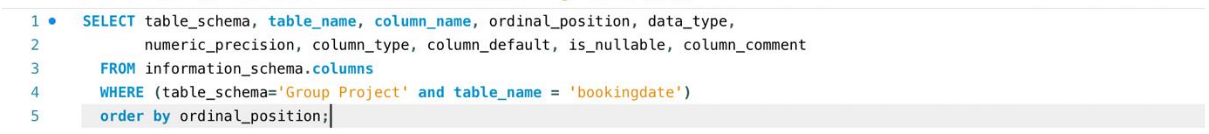


```
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'booking')
  order by ordinal_position;
```

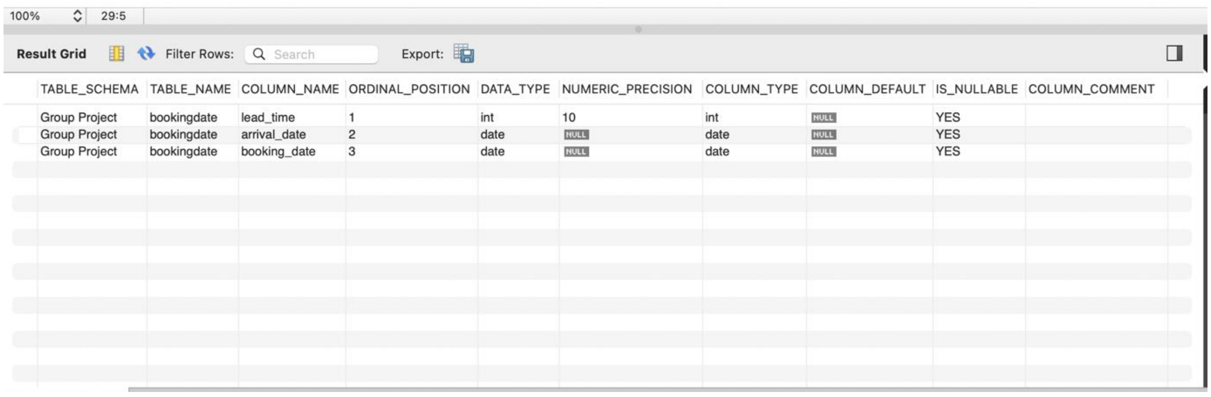


| TABLE_SCHEMA | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMME |
|---|---|---|---|---|---|---|---|---|---|
| Group Project | booking | booking_id | 1 | int | 10 | int | NULL | YES | |
| Group Project | booking | lead_time | 2 | int | 10 | int | NULL | YES | |
| Group Project | booking | meal | 3 | text | NULL | text | NULL | YES | |
| Group Project | booking | arrival_date | 4 | date | NULL | date | NULL | YES | |
| Group Project | booking | agent | 5 | int | 10 | int | NULL | YES | |
| Group Project | booking | days_in_waiting_list | 6 | int | 10 | int | NULL | YES | |
| Group Project | booking | required_car_parking_spaces | 7 | int | 10 | int | NULL | YES | |
| Group Project | booking | total_of_special_requests | 8 | int | 10 | int | NULL | YES | |
| Group Project | booking | deposit_type | 9 | text | NULL | text | NULL | YES | |
| Group Project | booking | stays_in_weekend_nights | 10 | int | 10 | int | NULL | YES | |
| Group Project | booking | stays_in_week_nights | 11 | int | 10 | int | NULL | YES | |
| Group Project | booking | total_occupants_per_stay | 12 | int | 10 | int | NULL | YES | |
| Group Project | booking | booking_changes | 13 | int | 10 | int | NULL | YES | |
| Group Project | booking | arrival_date_week_number | 14 | int | 10 | int | NULL | YES | |

Bookingdate Table:

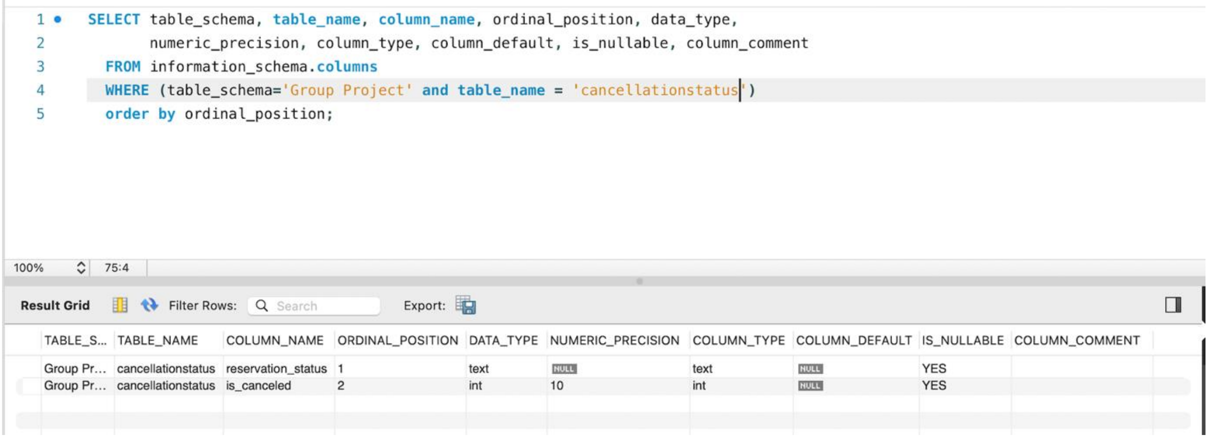


```
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'bookingdate')
  order by ordinal_position;
```



| TABLE_SCHEMA | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Project | bookingdate | lead_time | 1 | int | 10 | int | NULL | YES | |
| Group Project | bookingdate | arrival_date | 2 | date | NULL | date | NULL | YES | |
| Group Project | bookingdate | booking_date | 3 | date | NULL | date | NULL | YES | |

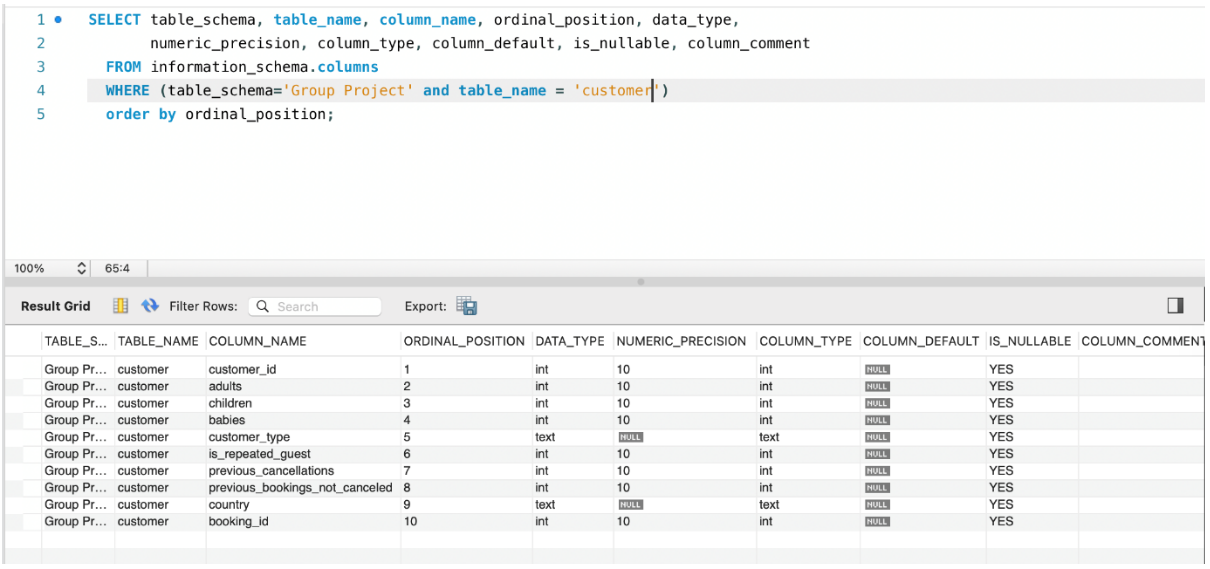
Cancellationstatus Table



```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
        numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'cancellationstatus')
  order by ordinal_position;
```

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | cancellationstatus | reservation_status | 1 | text | NULL | text | NULL | YES | |
| Group Pr... | cancellationstatus | is_canceled | 2 | int | 10 | int | NULL | YES | |

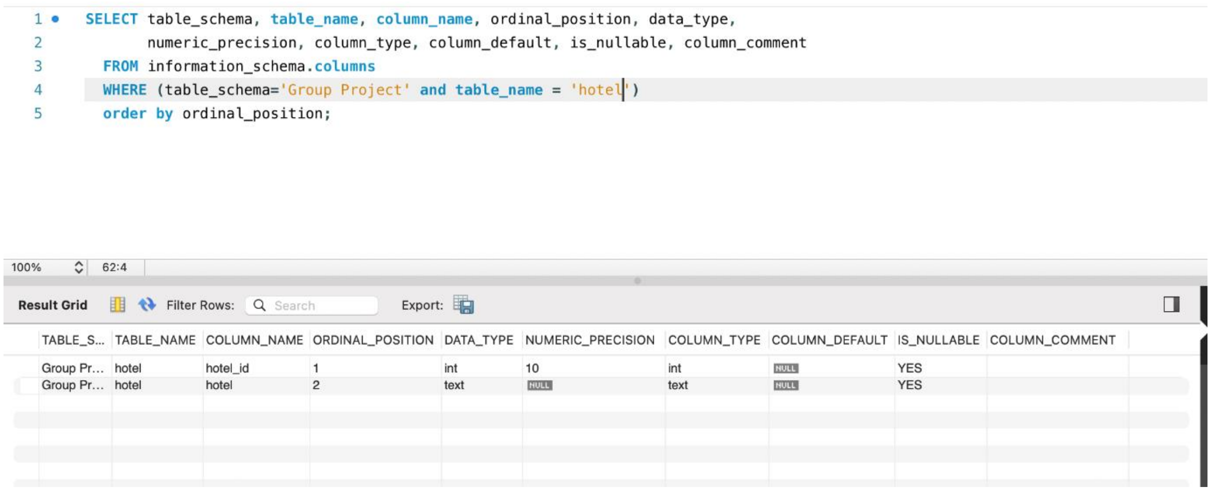
Customer Table



```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
        numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'customer')
  order by ordinal_position;
```

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | customer | customer_id | 1 | int | 10 | int | NULL | YES | |
| Group Pr... | customer | adults | 2 | int | 10 | int | NULL | YES | |
| Group Pr... | customer | children | 3 | int | 10 | int | NULL | YES | |
| Group Pr... | customer | babies | 4 | int | 10 | int | NULL | YES | |
| Group Pr... | customer | customer_type | 5 | text | NULL | text | NULL | YES | |
| Group Pr... | customer | is_repeated_guest | 6 | int | 10 | int | NULL | YES | |
| Group Pr... | customer | previous_cancellations | 7 | int | 10 | int | NULL | YES | |
| Group Pr... | customer | previous_bookings_not_canceled | 8 | int | 10 | int | NULL | YES | |
| Group Pr... | customer | country | 9 | text | NULL | text | NULL | YES | |
| Group Pr... | customer | booking_id | 10 | int | 10 | int | NULL | YES | |

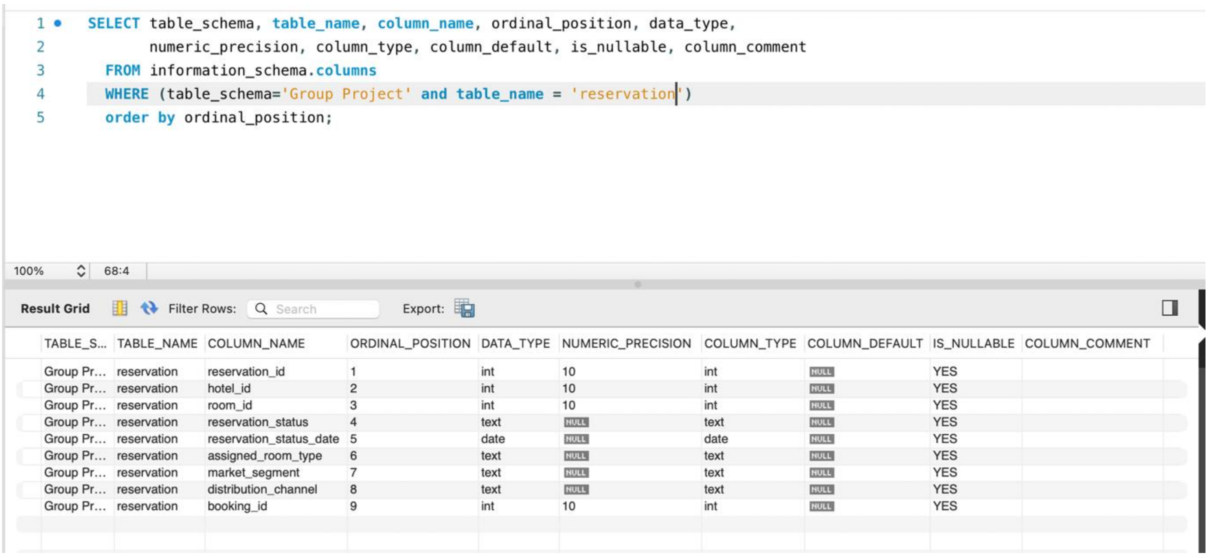
Hotel Table



```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
        numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'hotel')
  order by ordinal_position;
```

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | hotel | hotel_id | 1 | int | 10 | int | NULL | YES | |
| Group Pr... | hotel | hotel | 2 | text | NULL | text | NULL | YES | |

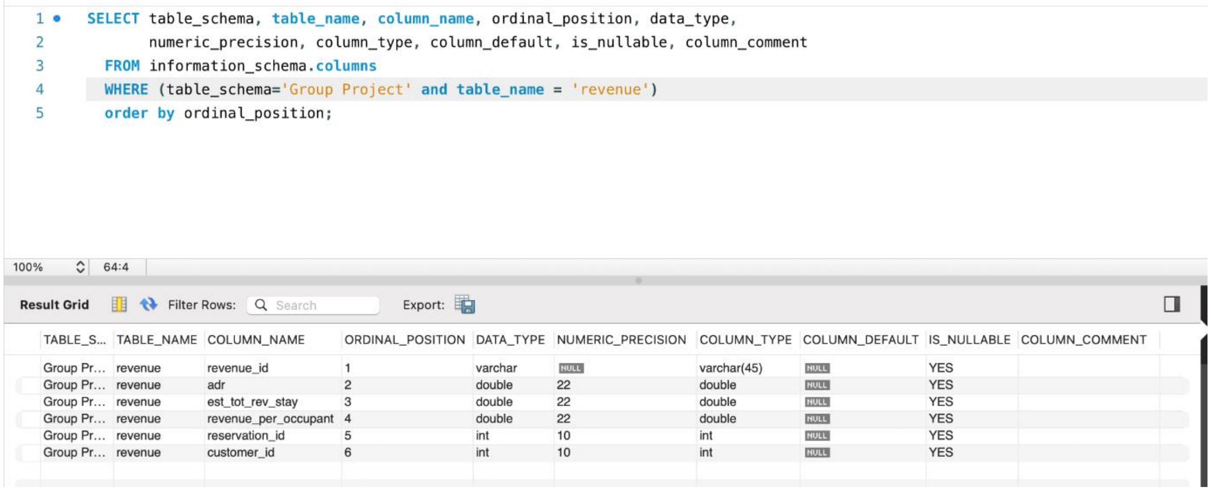
Reservation Table:



```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'reservation')
  order by ordinal_position;
```

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | reservation | reservation_id | 1 | int | 10 | int | NULL | YES | |
| Group Pr... | reservation | hotel_id | 2 | int | 10 | int | NULL | YES | |
| Group Pr... | reservation | room_id | 3 | int | 10 | int | NULL | YES | |
| Group Pr... | reservation | reservation_status | 4 | text | NULL | text | NULL | YES | |
| Group Pr... | reservation | reservation_status_date | 5 | date | NULL | date | NULL | YES | |
| Group Pr... | reservation | assigned_room_type | 6 | text | NULL | text | NULL | YES | |
| Group Pr... | reservation | market_segment | 7 | text | NULL | text | NULL | YES | |
| Group Pr... | reservation | distribution_channel | 8 | text | NULL | text | NULL | YES | |
| Group Pr... | reservation | booking_id | 9 | int | 10 | int | NULL | YES | |

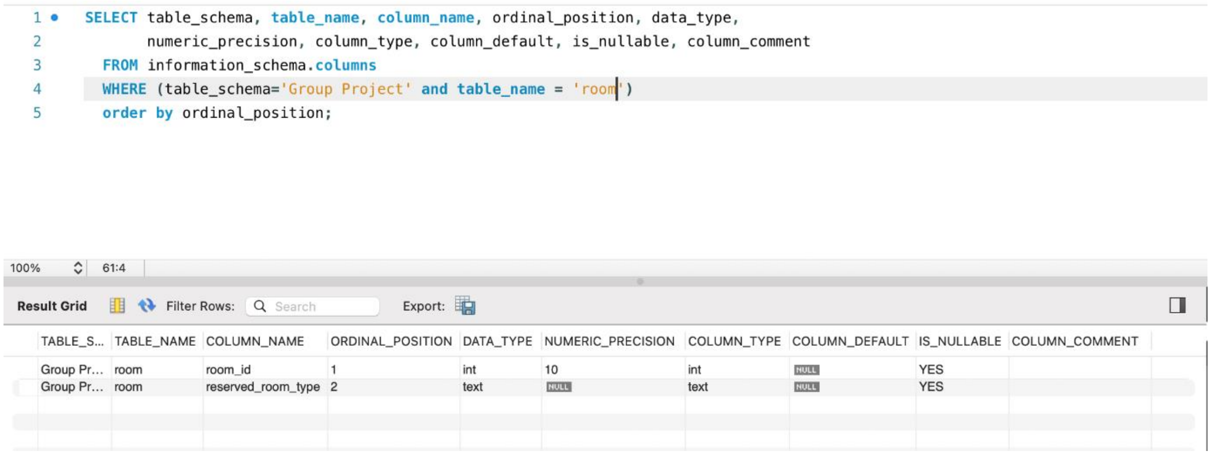
Revenue Table:



```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'revenue')
  order by ordinal_position;
```

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | revenue | revenue_id | 1 | varchar | NULL | varchar(45) | NULL | YES | |
| Group Pr... | revenue | adr | 2 | double | 22 | double | NULL | YES | |
| Group Pr... | revenue | est_tot_rev_stay | 3 | double | 22 | double | NULL | YES | |
| Group Pr... | revenue | revenue_per_occupant | 4 | double | 22 | double | NULL | YES | |
| Group Pr... | revenue | reservation_id | 5 | int | 10 | int | NULL | YES | |
| Group Pr... | revenue | customer_id | 6 | int | 10 | int | NULL | YES | |

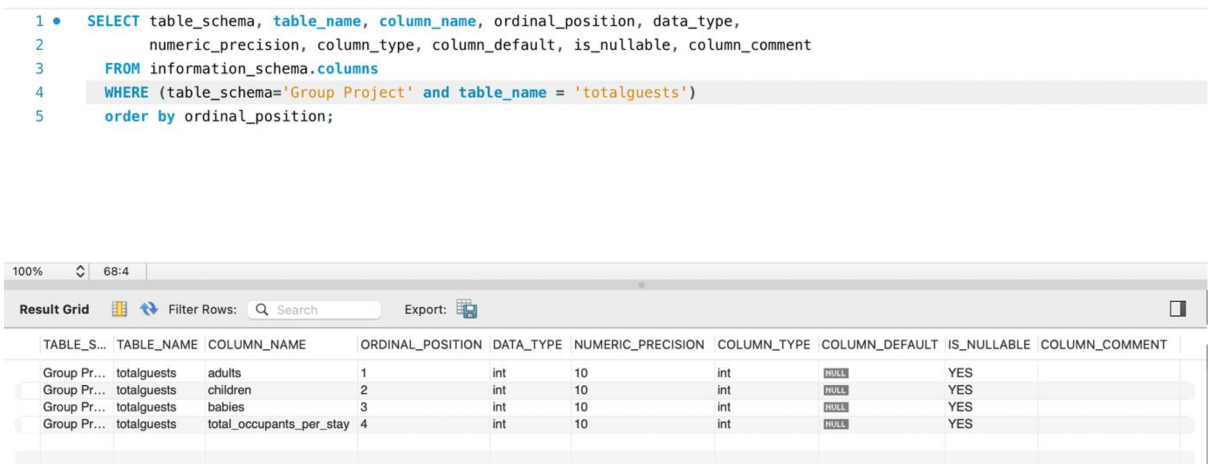
Room Table:



```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'room')
  order by ordinal_position;
```

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | room | room_id | 1 | int | 10 | int | NULL | YES | |
| Group Pr... | room | reserved_room_type | 2 | text | NULL | text | NULL | YES | |

Totalguests Table:

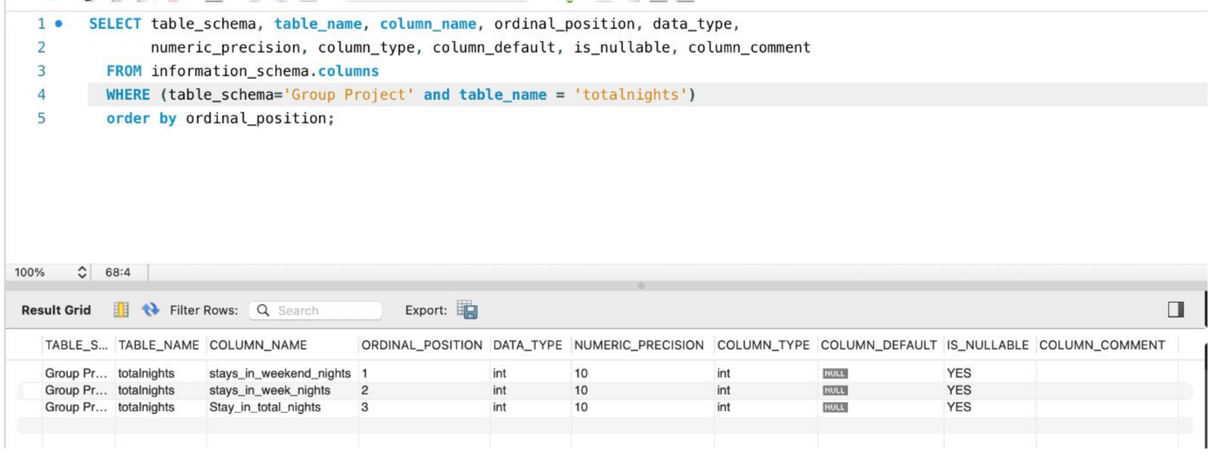


```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'totalguests')
  order by ordinal_position;
```

Result Grid | Filter Rows: | Export:

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | totalguests | adults | 1 | int | 10 | int | NULL | YES | |
| Group Pr... | totalguests | children | 2 | int | 10 | int | NULL | YES | |
| Group Pr... | totalguests | babies | 3 | int | 10 | int | NULL | YES | |
| Group Pr... | totalguests | total_occupants_per_stay | 4 | int | 10 | int | NULL | YES | |

Totalnights Table:



```sql
SELECT table_schema, table_name, column_name, ordinal_position, data_type,
       numeric_precision, column_type, column_default, is_nullable, column_comment
  FROM information_schema.columns
  WHERE (table_schema='Group Project' and table_name = 'totalnights')
  order by ordinal_position;
```

Result Grid | Filter Rows: | Export:

| TABLE_S... | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | DATA_TYPE | NUMERIC_PRECISION | COLUMN_TYPE | COLUMN_DEFAULT | IS_NULLABLE | COLUMN_COMMENT |
|---|---|---|---|---|---|---|---|---|---|
| Group Pr... | totalnights | stays_in_weekend_nights | 1 | int | 10 | int | NULL | YES | |
| Group Pr... | totalnights | stays_in_week_nights | 2 | int | 10 | int | NULL | YES | |
| Group Pr... | totalnights | Stay_in_total_nights | 3 | int | 10 | int | NULL | YES | |

## 2.2 SQL Commands (a, b, c, d, e)

Arrivaldate table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| arrival_date | SELECT count(arrival_date), min(arrival_date), max(arrival_date), max(arrival_date)-min(arrival_date), avg(arrival_date), stddev(arrival_date) FROM arrivaldate; |
| arrival_date_day_of_month | SELECT count(arrival_date_day_of_month), min(arrival_date_day_of_month), max(arrival_date_day_of_month), max(arrival_date_day_of_month)-min(arrival_date_day_of_month), avg(arrival_date_day_of_month), stddev(arrival_date_day_of_month) FROM arrivaldate; |
| arrival_date_year | SELECT count(arrival_date_year), min(arrival_date_year), max(arrival_date_year), max(arrival_date_year)-min(arrival_date_year), avg(arrival_date_year), stddev(arrival_date_year) FROM arrivaldate; |
| arrival_date_month | SELECT count(arrival_date_month), min(arrival_date_month), max(arrival_date_month), max(arrival_date_month)-min(arrival_date_month), avg(arrival_date_month), stddev(arrival_date_month) FROM arrivaldate; |

Booking table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| booking_id | SELECT count(booking_id), min(booking_id), max(booking_id), max(booking_id)-min(booking_id), avg(booking_id), stddev(booking_id) FROM booking; |
| lead_time | SELECT count(lead_time), min(lead_time), max(lead_time), max(lead_time)-min(lead_time), avg(lead_time), stddev(lead_time) |

|  |  |
|---|---|
|  | FROM booking; |
| arrival_date | SELECT count(arrival_date), min(arrival_date), max(arrival_date), max(arrival_date)-min(arrival_date), avg(arrival_date), stddev(arrival_date) FROM booking; |
| agent | SELECT count(agent), min(agent), max(agent), max(agent)-min(agent), avg(agent), stddev(agent) FROM booking; |
| days_in_waiting_list | SELECT count(days_in_waiting_list), min(days_in_waiting_list), max(days_in_waiting_list), max(days_in_waiting_list)-min(days_in_waiting_list), avg(days_in_waiting_list), stddev(days_in_waiting_list) FROM booking; |
| required_car_parking | SELECT count(required_car_parking_spaces), min(required_car_parking_spaces), max(required_car_parking_spaces), max(required_car_parking_spaces)-min(required_car_parking_spaces), avg(required_car_parking_spaces), stddev(required_car_parking_spaces) FROM booking; |
| total_of_special_requests | SELECT count(total_of_special_requests), min(total_of_special_requests), max(total_of_special_requests), max(total_of_special_requests)-min(total_of_special_requests), avg(total_of_special_requests), stddev(total_of_special_requests) FROM booking; |
| stays_in_weekend_nights | SELECT count(stays_in_weekend_nights), min(stays_in_weekend_nights), max(stays_in_weekend_nights), max(stays_in_weekend_nights)-min(stays_in_weekend_nights), avg(stays_in_weekend_nights), stddev(stays_in_weekend_nights) FROM booking; |
| stays_in_week_nights | SELECT count(stays_in_week_nights), min(stays_in_week_nights), max(stays_in_week_nights), max(stays_in_week_nights)-min(stays_in_week_nights), avg(stays_in_week_nights), stddev(stays_in_week_nights) FROM booking; |
| total_occupants_per_stay | SELECT count(total_occupants_per_stay), min(total_occupants_per_stay), max(total_occupants_per_stay), max(total_occupants_per_stay)-min(total_occupants_per_stay), avg(total_occupants_per_stay), stddev(total_occupants_per_stay) FROM booking; |
| booking_changes | SELECT count(booking_changes), min(booking_changes), max(booking_changes), max(booking_changes)-min(booking_changes), avg(booking_changes), stddev(booking_changes) FROM booking; |
| arrival_date_week_number | SELECT count(arrival_date_week_number), min(arrival_date_week_number), max(arrival_date_week_number), max(arrival_date_week_number)-min(arrival_date_week_number), avg(arrival_date_week_number), stddev(arrival_date_week_number) FROM booking; |

Booking table step 2.2 b, c, d. e

| Numeric Column Name | SQL Command |
|---|---|
| lead_time | SELECT count(lead_time), min(lead_time), max(lead_time), max(lead_time)-min(lead_time), avg(lead_time), stddev(lead_time) FROM bookingdate; |
| arrival_date | SELECT count(arrival_date), min(arrival_date), max(arrival_date), max(arrival_date)-min(arrival_date), avg(arrival_date), stddev(arrival_date) FROM bookingdate; |
| booking_date | SELECT count(booking_date), min(booking_date), max(booking_date), max(booking_date)-min(booking_date), avg(booking_date), stddev(booking_date) FROM bookingdate; |

Cancellationstatus table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|

| | |
|---|---|
| is_canceled | SELECT count(is_canceled), min(is_canceled), max(is_canceled), max(is_canceled)-min(is_canceled), avg(is_canceled), stddev(is_canceled) FROM cancellationstatus; |

Customer table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| customer_id | SELECT count(customer_id), min(customer_id), max(customer_id), max(customer_id)-min(customer_id), avg(customer_id), stddev(customer_id) FROM customer; |
| adults | SELECT count(adults), min(adults), max(adults), max(adults)-min(adults), avg(adults), stddev(adults) FROM customer; |
| children | SELECT count(children), min(children), max(children), max(children)-min(children), avg(children), stddev(children) FROM customer; |
| babies | SELECT count(babies), min(babies), max(babies), max(babies)-min(babies), avg(babies), stddev(babies) FROM customer; |
| is_repeated_guest | SELECT count(is_repeated_guest), min(is_repeated_guest), max(is_repeated_guest), max(is_repeated_guest)-min(is_repeated_guest), avg(is_repeated_guest), stddev(is_repeated_guest) FROM customer; |
| previous_cancellations | SELECT count(previous_cancellations), min(previous_cancellations), max(previous_cancellations), max(previous_cancellations)-min(previous_cancellations), avg(previous_cancellations), stddev(previous_cancellations) FROM customer; |
| previous_bookings_not_canceled | SELECT count(previous_bookings_not_canceled), min(previous_bookings_not_canceled), max(previous_bookings_not_canceled), max(previous_bookings_not_canceled)-min(previous_bookings_not_canceled), avg(previous_bookings_not_canceled), stddev(previous_bookings_not_canceled) FROM customer; |
| booking_id | SELECT count(booking_id), min(booking_id), max(booking_id), max(booking_id)-min(booking_id), avg(booking_id), stddev(booking_id) FROM customer; |

Hotel table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| hotel_id | SELECT count(hotel_id), min(hotel_id), max(hotel_id), max(hotel_id)-min(hotel_id), avg(hotel_id), stddev(hotel_id) FROM hotel; |

Reservation table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| reservation_id | SELECT count(reservation_id), min(reservation_id), max(reservation_id), max(reservation_id)-min(reservation_id), avg(reservation_id), stddev(reservation_id) FROM reservation; |
| hotel_id | SELECT count(hotel_id), min(hotel_id), max(hotel_id), max(hotel_id)-min(hotel_id), avg(hotel_id), stddev(hotel_id) FROM reservation; |
| room_id | SELECT count(room_id), min(room_id), max(room_id), max(room_id)-min(room_id), avg(room_id), stddev(room_id) FROM reservation; |
| reservation_status_date | SELECT count(reservation_status_date), min(reservation_status_date), max(reservation_status_date), max(reservation_status_date)- |

| | |
|---|---|
| | min(reservation_status_date), avg(reservation_status_date), stddev(reservation_status_date) FROM reservation; |
| booking_id | SELECT count(booking_id), min(booking_id), max(booking_id), max(booking_id)-min(booking_id), avg(booking_id), stddev(booking_id) FROM reservation; |

Revenue table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| revenue_id | SELECT count(revenue_id), min(revenue_id), max(revenue_id), max(revenue_id)-min(revenue_id), avg(revenue_id), stddev(revenue_id) FROM revenue; |
| adr | SELECT count(adr), min(adr), max(adr), max(adr)-min(adr), avg(adr), stddev(adr) FROM revenue; |
| est_tot_rev_stay | SELECT count(est_tot_rev_stay), min(est_tot_rev_stay), max(est_tot_rev_stay), max(est_tot_rev_stay)-min(est_tot_rev_stay), avg(est_tot_rev_stay), stddev(est_tot_rev_stay) FROM revenue; |
| revemue_per_occupant | SELECT count(revenue_per_occupant), min(revenue_per_occupant), max(revenue_per_occupant), max(revenue_per_occupant)-min(revenue_per_occupant), avg(revenue_per_occupant), stddev(revenue_per_occupant) FROM revenue; |
| reservation_id | SELECT count(reservation_id), min(reservation_id), max(reservation_id), max(reservation_id)-min(reservation_id), avg(reservation_id), stddev(reservation_id) FROM revenue; |
| customer_id | SELECT count(customer_id), min(customer_id), max(customer_id), max(customer_id)-min(customer_id), avg(customer_id), stddev(customer_id) FROM revenue; |

Room table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| room_id | SELECT count(room_id), min(room_id), max(room_id), max(room_id)-min(room_id), avg(room_id), stddev(room_id) FROM room; |

Totalguests table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| adults | SELECT count(adults), min(adults), max(adults), max(adults)-min(adults), avg(adults), stddev(adults) FROM totalguests; |
| children | SELECT count(children), min(children), max(children), max(children)-min(children), avg(children), stddev(children) FROM totalguests; |
| babies | SELECT count(babies), min(babies), max(babies), max(babies)-min(babies), avg(babies), stddev(babies) FROM totalguests; |
| total_occupants_per_stay | SELECT count(total_occupants_per_stay), min(total_occupants_per_stay), max(total_occupants_per_stay), max(total_occupants_per_stay)-min(total_occupants_per_stay), avg(total_occupants_per_stay), stddev(total_occupants_per_stay) FROM totalguests; |

Totalnights table step 2.2 b, c, d, e

| Numeric Column Name | SQL Command |
|---|---|
| stays_in_weekend_nights | SELECT count(stays_in_weekend_nights), min(stays_in_weekend_nights), max(stays_in_weekend_nights), max(stays_in_weekend_nights)-min(stays_in_weekend_nights), avg(stays_in_weekend_nights), stddev(stays_in_weekend_nights) FROM totalnights; |

| | |
|---|---|
| stays_in_week_nights | SELECT count(stays_in_week_nights), min(stays_in_week_nights), max(stays_in_week_nights), max(stays_in_week_nights)-min(stays_in_week_nights), avg(stays_in_week_nights), stddev(stays_in_week_nights)<br>FROM totalnights; |
| stays_in_total_nights | SELECT count(stay_in_total_nights), min(stay_in_total_nights), max(stay_in_total_nights), max(stay_in_total_nights)-min(stay_in_total_nights), avg(stay_in_total_nights), stddev(stay_in_total_nights)<br>FROM totalnights; |

## 2.2 SQL Commands (f, g, h)

2.2 f

Arrivaldate table

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| arrival_date | SELECT arrival_date AS numeric_column_name, COUNT(*) as frequency FROM arrivaldate GROUP BY arrival_date ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM arrivaldate) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM arrivaldate) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM arrivaldate) t | "SELECT DISTINCT ""95th Percentile"" AS ""Percentile"", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM arrivaldate) t " | SELECT AVG(arrival_date) AS median FROM ( SELECT arrival_date, ROW_NUMBER() OVER (ORDER BY arrival_date) AS row_num, COUNT(*) OVER () AS total_rows FROM arrivaldate ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

| | | | | | | |
|---|---|---|---|---|---|---|
| arrival_date_day_of_month | SELECT arrival_date_day_of_month AS numeric_column_name, COUNT(*) as frequency FROM arrivaldate GROUP BY arrival_date_day_of_month ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(arrival_date_day_of_month) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT arrival_date_day_of_month, percent_rank() OVER (ORDER BY arrival_date_day_of_month) p FROM arrivaldate)  t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(arrival_date_day_of_month) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT arrival_date_day_of_month, percent_rank() OVER (ORDER BY arrival_date_day_of_month) p FROM arrivaldate)  t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(arrival_date_day_of_month) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT arrival_date_day_of_month, percent_rank() OVER (ORDER BY arrival_date_day_of_month) p FROM arrivaldate)  t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(arrival_date_day_of_month) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT arrival_date_day_of_month, percent_rank() OVER (ORDER BY arrival_date_day_of_month) p FROM arrivaldate)  t | SELECT AVG(arrival_date_day_of_month) AS median FROM ( SELECT arrival_date_day_of_month, ROW_NUMBER() OVER (ORDER BY arrival_date_day_of_month) AS row_num, COUNT(*) OVER () AS total_rows FROM arrivaldate ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| arrival_date_year | SELECT arrival_date_year AS numeric_column_name, COUNT(*) as frequency FROM arrivaldate GROUP BY arrival_date_year ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(arrival_date_year) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT arrival_date_year, percent_rank() OVER (ORDER BY arrival_date_year) p FROM arrivaldate)  t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(arrival_date_year) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT arrival_date_year, percent_rank() OVER (ORDER BY arrival_date_year) p FROM arrivaldate)  t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(arrival_date_year) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT arrival_date_year, percent_rank() OVER (ORDER BY arrival_date_year) p FROM arrivaldate)  t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(arrival_date_year) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT arrival_date_year, percent_rank() OVER (ORDER BY arrival_date_year) p FROM arrivaldate)  t | SELECT AVG(arrival_date_year)  AS median FROM ( SELECT arrival_date_year, ROW_NUMBER() OVER (ORDER BY arrival_date_year) AS row_num, COUNT(*) OVER () AS total_rows FROM arrivaldate ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| arrival_date_month | SELECT arrival_date_month AS numeric_column_name, COUNT(*) as frequency FROM arrivaldate GROUP BY arrival_date_month ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(arrival_date_month) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT arrival_date_month, percent_rank() OVER (ORDER BY | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(arrival_date_month) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT arrival_date_month, percent_rank() OVER (ORDER | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(arrival_date_month) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT arrival_date_month, percent_rank() OVER (ORDER BY | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(arrival_date_month) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT arrival_date_month, percent_rank() OVER (ORDER BY | SELECT AVG(arrival_date_month)  AS median FROM ( SELECT arrival_date_month, ROW_NUMBER() OVER (ORDER BY arrival_date_month) AS row_num, COUNT(*) OVER () AS |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | arrival_date_month) p FROM arrivaldate) t | BY arrival_date_month) p FROM arrivaldate) t | arrival_date_month) p FROM arrivaldate) t | arrival_date_month) p FROM arrivaldate) t | total_rows FROM arrivaldate ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

**2.2 f Totalnights table**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| stays_in_weekend_nights | SELECT stays_in_weekend_nights AS numeric_column_name, COUNT(*) as frequency FROM totalnights GROUP BY stays_in_weekend_nights ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM totalnights) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM totalnights) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM totalnights) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM totalnights) t | SELECT AVG(stays_in_weekend_nights) AS median FROM ( SELECT stays_in_weekend_nights, ROW_NUMBER() OVER (ORDER BY stays_in_weekend_nights) AS row_num, COUNT(*) OVER () AS total_rows FROM totalnights ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| stays_in_week_nights | SELECT stays_in_week_nights AS numeric_column_name, COUNT(*) as frequency FROM totalnights GROUP BY stays_in_week_nights ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM totalnights) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM totalnights) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM totalnights) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM totalnights) t | SELECT AVG(stays_in_week_nights) AS median FROM ( SELECT stays_in_week_nights, ROW_NUMBER() OVER (ORDER BY stays_in_week_nights) AS row_num, COUNT(*) OVER () AS total_rows FROM totalnights ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | CEIL((total_rows + 1) / 2)); |
| stays_in_total_nights | SELECT stay_in_total_nights AS numeric_column_name, COUNT(*) as frequency FROM totalnights GROUP BY stay_in_total_nights ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(stay_in_total_nights) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT stay_in_total_nights, percent_rank() OVER (ORDER BY stay_in_total_nights) p FROM totalnights) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(stay_in_total_nights) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT stay_in_total_nights, percent_rank() OVER (ORDER BY stay_in_total_nights) p FROM totalnights) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(stay_in_total_nights) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT stay_in_total_nights, percent_rank() OVER (ORDER BY stay_in_total_nights) p FROM totalnights) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(stay_in_total_nights) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT stay_in_total_nights, percent_rank() OVER (ORDER BY stay_in_total_nights) p FROM totalnights) t | SELECT AVG(stay_in_total_nights) AS median FROM ( SELECT stay_in_total_nights, ROW_NUMBER() OVER (ORDER BY stay_in_total_nights) AS row_num, COUNT(*) OVER () AS total_rows FROM totalnights ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| **2.2f Totaguests table** | | | | | | |
| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
| adults | SELECT adults AS numeric_column_name, COUNT(*) as frequency FROM totalguests GROUP BY adults ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM totalguests) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM totalguests) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM totalguests) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM totalguests) t | SELECT AVG(adults) AS median FROM ( SELECT adults, ROW_NUMBER() OVER (ORDER BY adults) AS row_num, COUNT(*) OVER () AS total_rows FROM totalguests ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| children | SELECT children AS numeric_column_name, COUNT(*) as frequency FROM totalguests GROUP BY | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS | SELECT AVG(children) AS median FROM ( SELECT children, ROW_NUMBER() OVER (ORDER BY children) AS |

| | | | | | | |
|---|---|---|---|---|---|---|
| | children<br>ORDER BY<br>frequency<br>DESC<br>LIMIT 1; | value<br>FROM (SELECT<br>children,<br>percent_rank()<br>OVER (ORDER<br>BY children) p<br>FROM<br>totalguests) t | value<br>FROM (SELECT<br>children,<br>percent_rank()<br>OVER (ORDER<br>BY children) p<br>FROM<br>totalguests) t | value<br>FROM (SELECT<br>children,<br>percent_rank()<br>OVER (ORDER<br>BY children) p<br>FROM<br>totalguests) t | value<br>FROM (SELECT<br>children,<br>percent_rank()<br>OVER (ORDER<br>BY children) p<br>FROM<br>totalguests) t | row_num,<br>COUNT(*)<br>OVER () AS<br>total_rows<br>FROM<br>totalguests<br>) AS subquery<br>WHERE<br>row_num IN<br>(FLOOR((total<br>_rows + 1) / 2),<br>CEIL((total_ro<br>ws + 1) / 2)); |
| babies | SELECT<br>babies AS<br>numeric_col<br>umn_name,<br>COUNT(*)<br>as<br>frequency<br>FROM<br>totalguests<br>GROUP BY<br>babies<br>ORDER BY<br>frequency<br>DESC<br>LIMIT 1; | SELECT<br>DISTINCT "25th<br>Percentile" AS<br>"Percentile",<br>first_value(babies<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.25 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>babies,<br>percent_rank()<br>OVER (ORDER<br>BY babies) p<br>FROM<br>totalguests) t | SELECT<br>DISTINCT "50th<br>Percentile" AS<br>"Percentile",<br>first_value(babies<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.50 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>babies,<br>percent_rank()<br>OVER (ORDER<br>BY babies) p<br>FROM<br>totalguests) t | SELECT<br>DISTINCT "75th<br>Percentile" AS<br>"Percentile",<br>first_value(babies<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.75 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>babies,<br>percent_rank()<br>OVER (ORDER<br>BY babies) p<br>FROM<br>totalguests) t | SELECT<br>DISTINCT "95th<br>Percentile" AS<br>"Percentile",<br>first_value(babies<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.95 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>babies,<br>percent_rank()<br>OVER (ORDER<br>BY babies) p<br>FROM<br>totalguests) t | SELECT<br>AVG(babies)<br>AS median<br>FROM (<br>SELECT<br>babies,<br>ROW_NUMBE<br>R() OVER<br>(ORDER BY<br>babies) AS<br>row_num,<br>COUNT(*)<br>OVER () AS<br>total_rows<br>FROM<br>totalguests<br>) AS subquery<br>WHERE<br>row_num IN<br>(FLOOR((total<br>_rows + 1) / 2),<br>CEIL((total_ro<br>ws + 1) / 2)); |
| total_occup<br>ants_per_sta<br>y | SELECT<br>total_occup<br>ants_per_sta<br>y AS<br>numeric_col<br>umn_name,<br>COUNT(*)<br>as<br>frequency<br>FROM<br>totalguests<br>GROUP BY<br>total_occup<br>ants_per_sta<br>y<br>ORDER BY<br>frequency<br>DESC<br>LIMIT 1; | SELECT<br>DISTINCT "25th<br>Percentile" AS<br>"Percentile",<br>first_value(total_o<br>ccupants_per_stay<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.25 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>total_occupants_p<br>er_stay,<br>percent_rank()<br>OVER (ORDER<br>BY<br>total_occupants_p<br>er_stay) p FROM<br>totalguests) t | SELECT<br>DISTINCT "50th<br>Percentile" AS<br>"Percentile",<br>first_value(total_o<br>ccupants_per_stay<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.50 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>total_occupants_p<br>er_stay,<br>percent_rank()<br>OVER (ORDER<br>BY<br>total_occupants_p<br>er_stay) p FROM<br>totalguests) t | SELECT<br>DISTINCT "75th<br>Percentile" AS<br>"Percentile",<br>first_value(total_o<br>ccupants_per_stay<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.75 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>total_occupants_p<br>er_stay,<br>percent_rank()<br>OVER (ORDER<br>BY<br>total_occupants_p<br>er_stay) p FROM<br>totalguests) t | SELECT<br>DISTINCT "95th<br>Percentile" AS<br>"Percentile",<br>first_value(total_o<br>ccupants_per_stay<br>)<br>OVER (ORDER<br>BY CASE WHEN<br>p <= 0.95 THEN<br>p END DESC) AS<br>value<br>FROM (SELECT<br>total_occupants_p<br>er_stay,<br>percent_rank()<br>OVER (ORDER<br>BY<br>total_occupants_p<br>er_stay) p FROM<br>totalguests) t | SELECT<br>AVG(total_occ<br>upants_per_sta<br>y) AS median<br>FROM (<br>SELECT<br>total_occupants<br>_per_stay,<br>ROW_NUMBE<br>R() OVER<br>(ORDER BY<br>total_occupants<br>_per_stay) AS<br>row_num,<br>COUNT(*)<br>OVER () AS<br>total_rows<br>FROM<br>totalguests<br>) AS subquery<br>WHERE<br>row_num IN<br>(FLOOR((total<br>_rows + 1) / 2),<br>CEIL((total_ro<br>ws + 1) / 2)); |

**2.2(f) room table**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| room_id | SELECT room_id AS numeric_column_name, COUNT(*) as frequency FROM room GROUP BY room_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM room) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM room) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM room) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM room) t | SELECT AVG(room_id) AS median FROM ( SELECT room_id, ROW_NUMBER() OVER (ORDER BY room_id) AS row_num, COUNT(*) OVER () AS total_rows FROM room ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

**2.2(f)Revenue Table**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| revenue_id | SELECT revenue_id AS numeric_column_name, COUNT(*) as frequency FROM revenue GROUP BY revenue_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(revenue_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT revenue_id, percent_rank() OVER (ORDER BY revenue_id) p FROM revenue) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(revenue_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT revenue_id, percent_rank() OVER (ORDER BY revenue_id) p FROM revenue) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(revenue_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT revenue_id, percent_rank() OVER (ORDER BY revenue_id) p FROM revenue) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(revenue_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT revenue_id, percent_rank() OVER (ORDER BY revenue_id) p FROM revenue) t | SELECT AVG(revenue_id) AS median FROM ( SELECT revenue_id, ROW_NUMBER() OVER (ORDER BY revenue_id) AS row_num, COUNT(*) OVER () AS total_rows FROM revenue ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| adr | SELECT adr AS numeric_column_name, COUNT(*) as frequency FROM revenue GROUP BY adr ORDER BY | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(adr) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(adr) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(adr) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(adr) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT | AVG(adr) AS median FROM ( SELECT adr, ROW_NUMBER() OVER (ORDER BY adr) AS row_num, COUNT(*) |

| | | | | | | |
|---|---|---|---|---|---|---|
| | frequency DESC LIMIT 1; | adr, percent_rank() OVER (ORDER BY adr) p FROM revenue) t | adr, percent_rank() OVER (ORDER BY adr) p FROM revenue) t | adr, percent_rank() OVER (ORDER BY adr) p FROM revenue) t | adr, percent_rank() OVER (ORDER BY adr) p FROM revenue) t | OVER () AS total_rows FROM revenue ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| est_tot_rev_stay | SELECT est_tot_rev_stay AS numeric_column_name, COUNT(*) as frequency FROM revenue GROUP BY est_tot_rev_stay ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(est_tot_rev_stay) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT est_tot_rev_stay, percent_rank() OVER (ORDER BY est_tot_rev_stay) p FROM revenue) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(est_tot_rev_stay) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT est_tot_rev_stay, percent_rank() OVER (ORDER BY est_tot_rev_stay) p FROM revenue) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(est_tot_rev_stay) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT est_tot_rev_stay, percent_rank() OVER (ORDER BY est_tot_rev_stay) p FROM revenue) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(est_tot_rev_stay) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT est_tot_rev_stay, percent_rank() OVER (ORDER BY est_tot_rev_stay) p FROM revenue) t | SELECT AVG(est_tot_rev_stay) AS median FROM ( SELECT est_tot_rev_stay, ROW_NUMBER() OVER (ORDER BY est_tot_rev_stay) AS row_num, COUNT(*) OVER () AS total_rows FROM revenue ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| revenue_per_occupant | SELECT revenue_per_occupant AS numeric_column_name, COUNT(*) as frequency FROM revenue GROUP BY revenue_per_occupant ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(revenue_per_occupant) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT revenue_per_occupant, percent_rank() OVER (ORDER BY revenue_per_occupant) p FROM revenue) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(revenue_per_occupant) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT revenue_per_occupant, percent_rank() OVER (ORDER BY revenue_per_occupant) p FROM revenue) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(revenue_per_occupant) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT revenue_per_occupant, percent_rank() OVER (ORDER BY revenue_per_occupant) p FROM revenue) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(revenue_per_occupant) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT revenue_per_occupant, percent_rank() OVER (ORDER BY revenue_per_occupant) p FROM revenue) t | SELECT AVG(revenue_per_occupant) AS median FROM ( SELECT revenue_per_occupant, ROW_NUMBER() OVER (ORDER BY revenue_per_occupant) AS row_num, COUNT(*) OVER () AS total_rows FROM revenue ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| reservation_id | SELECT reservation_id AS numeric_col | SELECT DISTINCT "25th Percentile" AS "Percentile", | SELECT DISTINCT "50th Percentile" AS "Percentile", | SELECT DISTINCT "75th Percentile" AS "Percentile", | SELECT DISTINCT "95th Percentile" AS "Percentile", | AVG(reservation_id) AS median FROM ( |

| | | | | | | |
|---|---|---|---|---|---|---|
| | umn_name, COUNT(*) as frequency FROM revenue GROUP BY reservation_id ORDER BY frequency DESC LIMIT 1; | first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p FROM revenue) t | first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p FROM revenue) t | first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p FROM revenue) t | first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p FROM revenue) t | SELECT reservation_id, ROW_NUMBER() OVER (ORDER BY reservation_id) AS row_num, COUNT(*) OVER () AS total_rows FROM revenue ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| customer_id | SELECT customer_id AS numeric_column_name, COUNT(*) as frequency FROM revenue GROUP BY customer_id ORDER BY frequency DESC | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM revenue) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM revenue) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM revenue) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM revenue) t | SELECT AVG(customer_id) AS median FROM ( SELECT customer_id, ROW_NUMBER() OVER (ORDER BY customer_id) AS row_num, COUNT(*) OVER () AS total_rows FROM revenue ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

**2.2(f) Reservationtable**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| reservation_id | SELECT reservation_id AS numeric_column_name, COUNT(*) as frequency FROM reservation GROUP BY reservation_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(reservation_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT reservation_id, percent_rank() OVER (ORDER BY reservation_id) p | SELECT AVG(reservation_id) AS median FROM ( SELECT reservation_id, ROW_NUMBER() OVER (ORDER BY reservation_id) AS row_num, COUNT(*) OVER () AS total_rows FROM reservation |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | FROM reservation) t | FROM reservation) t | FROM reservation) t | FROM reservation) t | ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| hotel_id | SELECT hotel_id AS numeric_column_name, COUNT(*) as frequency FROM reservation GROUP BY hotel_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM reservation) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM reservation) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM reservation) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM reservation) t | SELECT AVG(hotel_id) AS median FROM ( SELECT hotel_id, ROW_NUMBER() OVER (ORDER BY hotel_id) AS row_num, COUNT(*) OVER () AS total_rows FROM reservation ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| room_id | SELECT room_id AS numeric_column_name, COUNT(*) as frequency FROM reservation GROUP BY room_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM reservation) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM reservation) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM reservation) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(room_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT room_id, percent_rank() OVER (ORDER BY room_id) p FROM reservation) t | SELECT AVG(room_id) AS median FROM ( SELECT room_id, ROW_NUMBER() OVER (ORDER BY room_id) AS row_num, COUNT(*) OVER () AS total_rows FROM reservation ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| reservation_status_date | SELECT reservation_status_date AS numeric_column_name, COUNT(*) as frequency FROM reservation GROUP BY reservation_status_date ORDER BY frequency | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(reservation_status_date) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT reservation_status_date, percent_rank() | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(reservation_status_date) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT reservation_status_date, percent_rank() | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(reservation_status_date) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT reservation_status_date, percent_rank() | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(reservation_status_date) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT reservation_status_date, percent_rank() | SELECT AVG(reservation_status_date) AS median FROM ( SELECT reservation_status_date, ROW_NUMBER() OVER (ORDER BY reservation_status_date) AS row_num, COUNT(*) |

| | | | | | | |
|---|---|---|---|---|---|---|
| | DESC LIMIT 1; | OVER (ORDER BY reservation_status_date) p FROM reservation) t | OVER (ORDER BY reservation_status_date) p FROM reservation) t | OVER (ORDER BY reservation_status_date) p FROM reservation) t | OVER (ORDER BY reservation_status_date) p FROM reservation) t | OVER () AS total_rows FROM reservation ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| booking_id | SELECT booking_id AS numeric_column_name, COUNT(*) as frequency FROM reservation GROUP BY booking_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM reservation) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM reservation) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM reservation) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM reservation) t | SELECT AVG(booking_id) AS median FROM ( SELECT booking_id, ROW_NUMBER() OVER (ORDER BY booking_id) AS row_num, COUNT(*) OVER () AS total_rows FROM reservation ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

### 2.2 (f) Hoteltable

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| hotel_id | SELECT hotel_id AS numeric_column_name, COUNT(*) as frequency FROM hotel GROUP BY hotel_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM hotel) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM hotel) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM hotel) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(hotel_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT hotel_id, percent_rank() OVER (ORDER BY hotel_id) p FROM hotel) t | SELECT AVG(hotel_id) AS median FROM ( SELECT hotel_id, ROW_NUMBER() OVER (ORDER BY hotel_id) AS row_num, COUNT(*) OVER () AS total_rows FROM hotel ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

**2.2(f) customer table**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| customer_id | SELECT customer_id AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY customer_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM customer) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM customer) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM customer) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(customer_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT customer_id, percent_rank() OVER (ORDER BY customer_id) p FROM customer) t | SELECT AVG(customer_id) AS median FROM ( SELECT customer_id, ROW_NUMBER() OVER (ORDER BY customer_id) AS row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| adults | SELECT adults AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY adults ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM customer) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM customer) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM customer) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(adults) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT adults, percent_rank() OVER (ORDER BY adults) p FROM customer) t | SELECT AVG(adults) AS median FROM ( SELECT adults, ROW_NUMBER() OVER (ORDER BY adults) AS row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| children | SELECT children AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY children ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT children, percent_rank() OVER (ORDER BY children) p | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT children, percent_rank() OVER (ORDER BY children) p | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT children, percent_rank() OVER (ORDER BY children) p | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(children) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT children, percent_rank() OVER (ORDER BY children) p | SELECT AVG(children) AS median FROM ( SELECT children, ROW_NUMBER() OVER (ORDER BY children) AS row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | FROM customer) t | FROM customer) t | FROM customer) t | FROM customer) t | WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| babies | SELECT babies AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY babies ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(babies) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT babies, percent_rank() OVER (ORDER BY babies) p FROM customer) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(babies) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT babies, percent_rank() OVER (ORDER BY babies) p FROM customer) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(babies) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT babies, percent_rank() OVER (ORDER BY babies) p FROM customer) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(babies) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT babies, percent_rank() OVER (ORDER BY babies) p FROM customer) t | SELECT AVG(babies) AS median FROM ( SELECT babies, ROW_NUMBER() OVER (ORDER BY babies) AS row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| is_repeated_guest | SELECT is_repeated_guest AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY is_repeated_guest ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(is_repeated_guest) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT is_repeated_guest, percent_rank() OVER (ORDER BY is_repeated_guest) p FROM customer) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(is_repeated_guest) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT is_repeated_guest, percent_rank() OVER (ORDER BY is_repeated_guest) p FROM customer) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(is_repeated_guest) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT is_repeated_guest, percent_rank() OVER (ORDER BY is_repeated_guest) p FROM customer) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(is_repeated_guest) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT is_repeated_guest, percent_rank() OVER (ORDER BY is_repeated_guest) p FROM customer) t | SELECT AVG(is_repeated_guest) AS median FROM ( SELECT is_repeated_guest, ROW_NUMBER() OVER (ORDER BY is_repeated_guest) AS row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| previous_cancellations | SELECT previous_cancellations AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY previous_cancellations | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(previous_cancellations) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT previous_cancella | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(previous_cancellations) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT previous_cancella | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(previous_cancellations) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT previous_cancella | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(previous_cancellations) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT previous_cancella | SELECT AVG(previous_cancellations) AS median FROM ( SELECT previous_cancellations, ROW_NUMBER() OVER (ORDER BY previous_cancellations) AS |

| | | | | | | |
|---|---|---|---|---|---|---|
| | ORDER BY frequency DESC LIMIT 1; | tions, percent_rank() OVER (ORDER BY previous_cancellations) p FROM customer) t | tions, percent_rank() OVER (ORDER BY previous_cancellations) p FROM customer) t | tions, percent_rank() OVER (ORDER BY previous_cancellations) p FROM customer) t | tions, percent_rank() OVER (ORDER BY previous_cancellations) p FROM customer) t | row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| previous_bookings_not_canceled | SELECT previous_bookings_not_canceled AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY previous_bookings_not_canceled ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(previous_bookings_not_canceled) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT previous_bookings_not_canceled, percent_rank() OVER (ORDER BY previous_bookings_not_canceled) p FROM customer) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(previous_bookings_not_canceled) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT previous_bookings_not_canceled, percent_rank() OVER (ORDER BY previous_bookings_not_canceled) p FROM customer) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(previous_bookings_not_canceled) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT previous_bookings_not_canceled, percent_rank() OVER (ORDER BY previous_bookings_not_canceled) p FROM customer) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(previous_bookings_not_canceled) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT previous_bookings_not_canceled, percent_rank() OVER (ORDER BY previous_bookings_not_canceled) p FROM customer) t | SELECT AVG(previous_bookings_not_canceled) AS median FROM ( SELECT previous_bookings_not_canceled, ROW_NUMBER() OVER (ORDER BY previous_bookings_not_canceled) AS row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| booking_id | SELECT booking_id AS numeric_column_name, COUNT(*) as frequency FROM customer GROUP BY booking_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM customer) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM customer) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM customer) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM customer) t | SELECT AVG(booking_id) AS median FROM ( SELECT booking_id, ROW_NUMBER() OVER (ORDER BY booking_id) AS row_num, COUNT(*) OVER () AS total_rows FROM customer ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

**2.2f iscanceled table**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| is_canceled | SELECT is_canceled AS numeric_column_name, COUNT(*) as frequency FROM cancellationstatus GROUP BY is_canceled ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(is_canceled) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT is_canceled, percent_rank() OVER (ORDER BY is_canceled) p FROM cancellationstatus ) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(is_canceled) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT is_canceled, percent_rank() OVER (ORDER BY is_canceled) p FROM cancellationstatus ) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(is_canceled) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT is_canceled, percent_rank() OVER (ORDER BY is_canceled) p FROM cancellationstatus ) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(is_canceled) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT is_canceled, percent_rank() OVER (ORDER BY is_canceled) p FROM cancellationstatus ) t | SELECT AVG(is_canceled) AS median FROM ( SELECT is_canceled, ROW_NUMBER() OVER (ORDER BY is_canceled) AS row_num, COUNT(*) OVER () AS total_rows FROM cancellationstatus ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

**2.2(f) bookingdate table**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| lead_time | SELECT lead_time AS numeric_column_name, COUNT(*) as frequency FROM Bookingdate GROUP BY lead_time ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time) p FROM bookingdate) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time) p FROM bookingdate) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time) p FROM bookingdate) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time) p FROM bookingdate) t | SELECT AVG(lead_time) AS median FROM ( SELECT lead_time, ROW_NUMBER() OVER (ORDER BY lead_time) AS row_num, COUNT(*) OVER () AS total_rows FROM bookingdate ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| arrival_date | SELECT arrival_date AS numeric_column_name, COUNT(*) as frequency FROM | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(arrival_date) | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(arrival_date) OVER (ORDER | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(arrival_date) OVER (ORDER | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(arrival_date) OVER (ORDER | SELECT AVG(arrival_date) AS median FROM ( SELECT arrival_date, |

|  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|
|  | Bookingdate GROUP BY arrival_date ORDER BY frequency DESC LIMIT 1; | OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM bookingdate) t | BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM bookingdate) t | BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM bookingdate) t | BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM bookingdate) t | ROW_NUMBER() OVER (ORDER BY arrival_date) AS row_num, COUNT(*) OVER () AS total_rows FROM bookingdate ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| booking_date | SELECT booking_date AS numeric_column_name, COUNT(*) as frequency FROM Bookingdate GROUP BY booking_date ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(booking_date) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT booking_date, percent_rank() OVER (ORDER BY booking_date) p FROM bookingdate) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(booking_date) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT booking_date, percent_rank() OVER (ORDER BY booking_date) p FROM bookingdate) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(booking_date) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT booking_date, percent_rank() OVER (ORDER BY booking_date) p FROM bookingda | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(booking_date) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT booking_date, percent_rank() OVER (ORDER BY booking_date) p FROM bookingdate | SELECT AVG(booking_date) AS median FROM ( SELECT booking_date, ROW_NUMBER() OVER (ORDER BY booking_date) AS row_num, COUNT(*) OVER () AS total_rows FROM bookingdate ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |

**2.2(f) booking table**

| Numeric Column Name | Mode | 25% | 50% | 75% | 95% | Median |
|---|---|---|---|---|---|---|
| booking_id | SELECTbookibg_id, COUNT(*) as frequency FROM Booking GROUP BY booking_id ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM booking) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM booking) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM booking) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(booking_id) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT booking_id, percent_rank() OVER (ORDER BY booking_id) p FROM booking) t | SELECT AVG(booking_id) AS median FROM ( SELECT booking_id, ROW_NUMBER() OVER (ORDER BY booking_id) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| lead_time | SELECT lead_time AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY lead_time ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time)  p FROM booking) t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time)  p FROM booking) t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time)  p FROM booking) t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(lead_time) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT lead_time, percent_rank() OVER (ORDER BY lead_time)  p FROM booking) t | SELECT AVG(lead_time) AS median FROM ( SELECT lead_time, ROW_NUMBER() OVER (ORDER BY lead_time)  AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| arrival_date | SELECT arrival_date AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY arrival_date ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM booking) t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM booking) t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM booking) t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(arrival_date) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT arrival_date, percent_rank() OVER (ORDER BY arrival_date) p FROM booking) t | SELECT AVG(arrival_date) AS median FROM ( SELECT arrival_date, ROW_NUMBER() OVER (ORDER BY arrival_date) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| agent | SELECT agent AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY agent ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(agent) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT agent, percent_rank() OVER (ORDER BY agent) p FROM booking) t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(agent) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT agent, percent_rank() OVER (ORDER BY agent) p FROM booking) t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(agent) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT agent, percent_rank() OVER (ORDER BY agent) p FROM booking) t | "SELECT DISTINCT ""95th Percentile""  AS ""Percentile"", first_value(agent) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT agent, percent_rank() OVER (ORDER BY agent) p FROM booking) t | SELECT AVG(agent) AS median FROM ( SELECT agent, ROW_NUMBER() OVER (ORDER BY agent) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| days_in_waiting_list | SELECT days_in_waiting_list AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY days_in_waiting_list ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(days_in_waiting_list) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT days_in_waiting_list, percent_rank() OVER (ORDER BY days_in_waiting_list) p FROM booking) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(days_in_waiting_list) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT days_in_waiting_list, percent_rank() OVER (ORDER BY days_in_waiting_list) p FROM booking) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(days_in_waiting_list) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT days_in_waiting_list, percent_rank() OVER (ORDER BY days_in_waiting_list) p FROM booking) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(days_in_waiting_list) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT days_in_waiting_list, percent_rank() OVER (ORDER BY days_in_waiting_list) p FROM booking) t | SELECT AVG(days_in_waiting_list) AS median FROM ( SELECT days_in_waiting_list, ROW_NUMBER() OVER (ORDER BY days_in_waiting_list) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| required_car_parking_spaces | SELECT required_car_parking_spaces AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY required_car_parking_spaces ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(required_car_parking_spaces) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT required_car_parking_spaces, percent_rank() OVER (ORDER BY required_car_parking_spaces) p FROM booking) t | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(required_car_parking_spaces) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT required_car_parking_spaces, percent_rank() OVER (ORDER BY required_car_parking_spaces) p FROM booking) t | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(required_car_parking_spaces) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT required_car_parking_spaces, percent_rank() OVER (ORDER BY required_car_parking_spaces) p FROM booking) t | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(required_car_parking_spaces) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT required_car_parking_spaces, percent_rank() OVER (ORDER BY required_car_parking_spaces) p FROM booking) t | SELECT AVG(required_car_parking_spaces) AS median FROM ( SELECT required_car_parking_spaces, ROW_NUMBER() OVER (ORDER BY required_car_parking_spaces) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| total_of_special_requests | SELECT total_of_special_requests AS numeric_column_name, COUNT(*) as frequency FROM | SELECT DISTINCT "25th Percentile" AS "Percentile", first_value(total_of_special_requests ) OVER (ORDER BY CASE | SELECT DISTINCT "50th Percentile" AS "Percentile", first_value(total_of_special_requests ) OVER (ORDER BY CASE | SELECT DISTINCT "75th Percentile" AS "Percentile", first_value(total_of_special_requests ) OVER (ORDER BY CASE | SELECT DISTINCT "95th Percentile" AS "Percentile", first_value(total_of_special_requests ) OVER (ORDER BY CASE | SELECT AVG(total_of_special_requests) AS median FROM ( SELECT total_of_special_requests, ROW_NUMBE |

| | | | | | | |
|---|---|---|---|---|---|---|
| | Booking GROUP BY total_of_special_requests ORDER BY frequency DESC LIMIT 1; | WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT total_of_special_requests, percent_rank() OVER (ORDER BY total_of_special_requests) p FROM booking) t | WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT total_of_special_requests, percent_rank() OVER (ORDER BY total_of_special_requests) p FROM booking) t | WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT total_of_special_requests, percent_rank() OVER (ORDER BY total_of_special_requests) p FROM booking) t | WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT total_of_special_requests, percent_rank() OVER (ORDER BY total_of_special_requests) p FROM booking) t | R() OVER (ORDER BY total_of_special_requests) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| stays_in_weekend_nights | SELECT stays_in_weekend_nights AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY stays_in_weekend_nights ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM booking) t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM booking) t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM booking) t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(stays_in_weekend_nights) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT stays_in_weekend_nights, percent_rank() OVER (ORDER BY stays_in_weekend_nights) p FROM booking) t | SELECT AVG(stays_in_weekend_nights) AS median FROM ( SELECT stays_in_weekend_nights, ROW_NUMBER() OVER (ORDER BY stays_in_weekend_nights) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| stays_in_week_nights | SELECT stays_in_week_nights AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY stays_in_week_nights ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM booking) t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM booking) t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM booking) t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(stays_in_week_nights) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT stays_in_week_nights, percent_rank() OVER (ORDER BY stays_in_week_nights) p FROM booking) t | SELECT AVG(stays_in_week_nights) AS median FROM ( SELECT stays_in_week_nights, ROW_NUMBER() OVER (ORDER BY stays_in_week_nights) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | CEIL((total_rows + 1) / 2)); |
| total_occupants_per_stay | SELECT total_occupants_per_stay AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY total_occupants_per_stay ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(total_occupants_per_stay) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT total_occupants_per_stay, percent_rank() OVER (ORDER BY total_occupants_per_stay) p FROM booking) t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(total_occupants_per_stay) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT total_occupants_per_stay, percent_rank() OVER (ORDER BY total_occupants_per_stay) p FROM booking) t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(total_occupants_per_stay) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT total_occupants_per_stay, percent_rank() OVER (ORDER BY total_occupants_per_stay) p FROM booking) t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(total_occupants_per_stay) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT total_occupants_per_stay, percent_rank() OVER (ORDER BY total_occupants_per_stay) p FROM booking) t | SELECT AVG(total_occupants_per_stay) AS median FROM ( SELECT total_occupants_per_stay, ROW_NUMBER() OVER (ORDER BY total_occupants_per_stay) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| booking_changes | SELECT booking_changes AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY booking_changes ORDER BY frequency DESC LIMIT 1; | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(booking_changes) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT booking_changes, percent_rank() OVER (ORDER BY booking_changes) p FROM booking) t | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(booking_changes) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT booking_changes, percent_rank() OVER (ORDER BY booking_changes) p FROM booking) t | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(booking_changes) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT booking_changes, percent_rank() OVER (ORDER BY booking_changes) p FROM booking) t | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(booking_changes) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT booking_changes, percent_rank() OVER (ORDER BY booking_changes) p FROM booking) t | SELECT AVG(booking_changes) AS median FROM ( SELECT booking_changes, ROW_NUMBER() OVER (ORDER BY booking_changes) AS row_num, COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total_rows + 1) / 2), CEIL((total_rows + 1) / 2)); |
| arrival_date_week_number | SELECT arrival_date_week_number AS numeric_column_name, COUNT(*) as frequency FROM Booking GROUP BY arrival_date_week_number | SELECT DISTINCT "25th Percentile"  AS "Percentile", first_value(arrival_date_week_number) OVER (ORDER BY CASE WHEN p <= 0.25 THEN p END DESC) AS value FROM (SELECT | SELECT DISTINCT "50th Percentile"  AS "Percentile", first_value(arrival_date_week_number) OVER (ORDER BY CASE WHEN p <= 0.50 THEN p END DESC) AS value FROM (SELECT arrival_date_week | SELECT DISTINCT "75th Percentile"  AS "Percentile", first_value(arrival_date_week_number) OVER (ORDER BY CASE WHEN p <= 0.75 THEN p END DESC) AS value FROM (SELECT arrival_date_week | SELECT DISTINCT "95th Percentile"  AS "Percentile", first_value(arrival_date_week_number) OVER (ORDER BY CASE WHEN p <= 0.95 THEN p END DESC) AS value FROM (SELECT arrival_date_week | SELECT AVG(arrival_date_week_number) AS median FROM ( SELECT arrival_date_week_number, ROW_NUMBER() OVER (ORDER BY arrival_date_week_number) AS row_num, |

| | | | | | | |
|---|---|---|---|---|---|---|
| | ORDER BY frequency DESC LIMIT 1; | arrival_date_week _number, percent_rank() OVER (ORDER BY arrival_date_week _number) p FROM booking) t | _number, percent_rank() OVER (ORDER BY arrival_date_week _number) p FROM booking) t | _number, percent_rank() OVER (ORDER BY arrival_date_week _number) p FROM booking) t | _number, percent_rank() OVER (ORDER BY arrival_date_week _number) p FROM booking) t | COUNT(*) OVER () AS total_rows FROM booking ) AS subquery WHERE row_num IN (FLOOR((total _rows + 1) / 2), CEIL((total_ro ws + 1) / 2)); |

**2.2 g arrivaldate tabel**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| arrival_date | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(arrival_date) AS non_null_count,<br>(COUNT(*) - COUNT(arrival_date)) AS missing_quantity,<br>((COUNT(*) - COUNT(arrival_date)) / COUNT(*)) AS missing_rate<br>FROM arrivaldate; |
| arrival_date_day_of_month | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(arrival_date_day_of_month) AS non_null_count,<br>(COUNT(*) - COUNT(arrival_date_day_of_month)) AS missing_quantity,<br>((COUNT(*) - COUNT(arrival_date_day_of_month)) / COUNT(*)) AS missing_rate<br>FROM arrivaldate; |
| arrival_date_year | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(arrival_date_year) AS non_null_count,<br>(COUNT(*) - COUNT(arrival_date_year)) AS missing_quantity,<br>((COUNT(*) - COUNT(arrival_date_year)) / COUNT(*)) AS missing_rate<br>FROM arrivaldate; |
| arrival_date_month | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(arrival_date_month) AS non_null_count,<br>(COUNT(*) - COUNT(arrival_date_month)) AS missing_quantity,<br>((COUNT(*) - COUNT(arrival_date_month)) / COUNT(*)) AS missing_rate<br>FROM arrivaldate; |

**2.2g totalgueststable**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| adults | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(adults) AS non_null_count,<br>(COUNT(*) - COUNT(adults)) AS missing_quantity,<br>((COUNT(*) - COUNT(adults)) / COUNT(*)) AS missing_rate<br>FROM totalguests; |
| children | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(children) AS non_null_count,<br>(COUNT(*) - COUNT(children)) AS missing_quantity,<br>((COUNT(*) - COUNT(children)) / COUNT(*)) AS missing_rate<br>FROM totalguests; |
| babies | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(babies) AS non_null_count,<br>(COUNT(*) - COUNT(babies)) AS missing_quantity, |

| | |
|---|---|
| | ((COUNT(*) - COUNT(babies)) / COUNT(*)) AS missing_rate<br>FROM totalguests; |
| total_occupants_per_stay | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(total_occupants_per_stay) AS non_null_count,<br>(COUNT(*) - COUNT(total_occupants_per_stay)) AS missing_quantity,<br>((COUNT(*) - COUNT(total_occupants_per_stay)) / COUNT(*)) AS missing_rate<br>FROM totalguests; |

**2.2 g roomtable**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| room_id | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(room_id) AS non_null_count,<br>(COUNT(*) - COUNT(room_id)) AS missing_quantity,<br>((COUNT(*) - COUNT(room_id)) / COUNT(*)) AS missing_rate<br>FROM room; |

**2.2 g revenue table**

| | |
|---|---|
| Numeric Column Name | Missing Quantity and Rate |
| revenue_id | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(revenue_id) AS non_null_count,<br>(COUNT(*) - COUNT(revenue_id)) AS missing_quantity,<br>((COUNT(*) - COUNT(revenue_id)) / COUNT(*)) AS missing_rate<br>FROM revenue; |
| adr | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(adr) AS non_null_count,<br>(COUNT(*) - COUNT(adr)) AS missing_quantity,<br>((COUNT(*) - COUNT(adr)) / COUNT(*)) AS missing_rate<br>FROM revenue; |
| est_tot_rev_stay | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(est_tot_rev_stay) AS non_null_count,<br>(COUNT(*) - COUNT(est_tot_rev_stay)) AS missing_quantity,<br>((COUNT(*) - COUNT(est_tot_rev_stay)) / COUNT(*)) AS missing_rate<br>FROM revenue; |
| revenue_per_occupant | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(revenue_per_occupant) AS non_null_count,<br>(COUNT(*) - COUNT(revenue_per_occupant)) AS missing_quantity,<br>((COUNT(*) - COUNT(revenue_per_occupant)) / COUNT(*)) AS missing_rate<br>FROM revenue; |
| reservation_id | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(reservation_id) AS non_null_count,<br>(COUNT(*) - COUNT(reservation_id)) AS missing_quantity,<br>((COUNT(*) - COUNT(reservation_id)) / COUNT(*)) AS missing_rate<br>FROM revenue; |
| customer_id | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(customer_id) AS non_null_count,<br>(COUNT(*) - COUNT(customer_id)) AS missing_quantity,<br>((COUNT(*) - COUNT(customer_id)) / COUNT(*)) AS missing_rate<br>FROM revenue; |

**2.2g reservation table**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| reservation_id | SELECT COUNT(*) AS total_rows, COUNT(reservation_id) AS non_null_count, (COUNT(*) - COUNT(reservation_id)) AS missing_quantity, ((COUNT(*) - COUNT(reservation_id)) / COUNT(*)) AS missing_rate FROM reservation; |
| hotel_id | SELECT COUNT(*) AS total_rows, COUNT(hotel_id) AS non_null_count, (COUNT(*) - COUNT(hotel_id)) AS missing_quantity, ((COUNT(*) - COUNT(hotel_id)) / COUNT(*)) AS missing_rate FROM reservation; |
| room_id | SELECT COUNT(*) AS total_rows, COUNT(room_id) AS non_null_count, (COUNT(*) - COUNT(room_id)) AS missing_quantity, ((COUNT(*) - COUNT(room_id)) / COUNT(*)) AS missing_rate FROM reservation; |
| reservation_status_date | SELECT COUNT(*) AS total_rows, COUNT(reservation_status_date) AS non_null_count, (COUNT(*) - COUNT(reservation_status_date)) AS missing_quantity, ((COUNT(*) - COUNT(reservation_status_date)) / COUNT(*)) AS missing_rate FROM reservation; |
| booking_id | SELECT COUNT(*) AS total_rows, COUNT(booking_id) AS non_null_count, (COUNT(*) - COUNT(booking_id)) AS missing_quantity, ((COUNT(*) - COUNT(booking_id)) / COUNT(*)) AS missing_rate FROM reservation; |

**2.2g hotel table**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| hotel_id | SELECT COUNT(*) AS total_rows, COUNT(hotel_id) AS non_null_count, (COUNT(*) - COUNT(hotel_id)) AS missing_quantity, ((COUNT(*) - COUNT(hotel_id)) / COUNT(*)) AS missing_rate FROM hotel; |

**2.2 g customer table**

| | |
|---|---|
| Numeric Column Name | Missing Quantity and Rate |
| customer_id | SELECT COUNT(*) AS total_rows, COUNT(customer_id) AS non_null_count, (COUNT(*) - COUNT(customer_id)) AS missing_quantity, ((COUNT(*) - COUNT(customer_id)) / COUNT(*)) AS missing_rate FROM customer; |
| adults | select COUNT(*) AS total_rows, COUNT(adults) AS non_null_count, (COUNT(*) - COUNT(adults)) AS missing_quantity, |

| | |
|---|---|
| | ((COUNT(*) - COUNT(adults)) / COUNT(*)) AS missing_rate<br>FROM customer; |
| children | select<br>COUNT(*) AS total_rows,<br>COUNT(children) AS non_null_count,<br>(COUNT(*) - COUNT(children)) AS missing_quantity,<br>((COUNT(*) - COUNT(children)) / COUNT(*)) AS missing_rate<br>FROM customer; |
| babies | select<br>COUNT(*) AS total_rows,<br>COUNT(babies) AS non_null_count,<br>(COUNT(*) - COUNT(babies)) AS missing_quantity,<br>((COUNT(*) - COUNT(babies)) / COUNT(*)) AS missing_rate<br>FROM customer; |
| is_repeated_guest | select<br>COUNT(*) AS total_rows,<br>COUNT(is_repeated_guest) AS non_null_count,<br>(COUNT(*) - COUNT(is_repeated_guest)) AS missing_quantity,<br>((COUNT(*) - COUNT(is_repeated_guest)) / COUNT(*)) AS missing_rate<br>FROM customer; |
| previous_cancellations | select<br>COUNT(*) AS total_rows,<br>COUNT(previous_cancellations) AS non_null_count,<br>(COUNT(*) - COUNT(previous_cancellations)) AS missing_quantity,<br>((COUNT(*) - COUNT(previous_cancellations)) / COUNT(*)) AS missing_rate<br>FROM customer; |
| previous_bookings_not_canceled | select<br>COUNT(*) AS total_rows,<br>COUNT(previous_bookings_not_canceled) AS non_null_count,<br>(COUNT(*) - COUNT(previous_bookings_not_canceled)) AS missing_quantity,<br>((COUNT(*) - COUNT(previous_bookings_not_canceled)) / COUNT(*)) AS missing_rate<br>FROM customer; |
| booking_id | select<br>COUNT(*) AS total_rows,<br>COUNT(booking_id) AS non_null_count,<br>(COUNT(*) - COUNT(booking_id)) AS missing_quantity,<br>((COUNT(*) - COUNT(booking_id)) / COUNT(*)) AS missing_rate<br>FROM customer; |

**2.2 g is cancelled table**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| is_canceled | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(is_canceled) AS non_null_count,<br>(COUNT(*) - COUNT(is_canceled)) AS missing_quantity,<br>((COUNT(*) - COUNT(is_canceled)) / COUNT(*)) AS missing_rate<br>FROM cancellationstatus; |

**2.2 g bookingdate table**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| lead_time | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(lead_time) AS non_null_count,<br>(COUNT(*) - COUNT(lead_time)) AS missing_quantity,<br>((COUNT(*) - COUNT(lead_time)) / COUNT(*)) AS missing_rate<br>FROM bookingdate; |
| arrival_date | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(arrival_date) AS non_null_count,<br>(COUNT(*) - COUNT(arrival_date)) AS missing_quantity, |

| | |
|---|---|
| | ((COUNT(*) - COUNT(arrival_date)) / COUNT(*)) AS missing_rate FROM bookingdate; |
| booking_date | SELECT COUNT(*) AS total_rows, COUNT(booking_date) AS non_null_count, (COUNT(*) - COUNT(booking_date)) AS missing_quantity, ((COUNT(*) - COUNT(booking_date)) / COUNT(*)) AS missing_rate FROM bookingdate; |

**2.2g bookingtable**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| booking_id | SELECT COUNT(*) AS total_rows, COUNT(booking_id) AS non_null_count, (COUNT(*) - COUNT(booking_id)) AS missing_quantity, ((COUNT(*) - COUNT(booking_id)) / COUNT(*)) AS missing_rate FROM Booking; |
| lead_time | "SELECT COUNT(*) AS total_rows, COUNT(lead_time) AS non_null_count, (COUNT(*) - COUNT(lead_time)) AS missing_quantity, ((COUNT(*) - COUNT(lead_time)) / COUNT(*)) AS missing_rate FROM Booking; " |
| arrival_date | SELECT COUNT(*) AS total_rows, COUNT(arrival_date) AS non_null_count, (COUNT(*) - COUNT(arrival_date)) AS missing_quantity, ((COUNT(*) - COUNT(arrival_date)) / COUNT(*)) AS missing_rate FROM Booking; |
| agent | SELECT COUNT(*) AS total_rows, COUNT(agent) AS non_null_count, (COUNT(*) - COUNT(agent)) AS missing_quantity, ((COUNT(*) - COUNT(agent)) / COUNT(*)) AS missing_rate FROM Booking; |
| days_in_waiting_list | SELECT COUNT(*) AS total_rows, COUNT(days_in_waiting_list) AS non_null_count, (COUNT(*) - COUNT(days_in_waiting_list)) AS missing_quantity, ((COUNT(*) - COUNT(days_in_waiting_list)) / COUNT(*)) AS missing_rate FROM Booking; |
| required_car_parking_spaces | SELECT COUNT(*) AS total_rows, COUNT(required_car_parking_spaces) AS non_null_count, (COUNT(*) - COUNT(required_car_parking_spaces)) AS missing_quantity, ((COUNT(*) - COUNT(required_car_parking_spaces)) / COUNT(*)) AS missing_rate FROM Booking; |
| total_of_special_requests | SELECT COUNT(*) AS total_rows, COUNT(total_of_special_requests) AS non_null_count, (COUNT(*) - COUNT(total_of_special_requests)) AS missing_quantity, ((COUNT(*) - COUNT(total_of_special_requests)) / COUNT(*)) AS missing_rate FROM Booking; |
| stays_in_weekend_nights | SELECT COUNT(*) AS total_rows, COUNT(stays_in_weekend_nights) AS non_null_count, (COUNT(*) - COUNT(stays_in_weekend_nights)) AS missing_quantity, ((COUNT(*) - COUNT(stays_in_weekend_nights)) / COUNT(*)) AS missing_rate FROM Booking; |
| stays_in_week_nights | SELECT COUNT(*) AS total_rows, COUNT(stays_in_week_nights) AS non_null_count, |

| | |
|---|---|
| | (COUNT(*) - COUNT(stays_in_week_nights)) AS missing_quantity,<br>((COUNT(*) - COUNT(stays_in_week_nights)) / COUNT(*)) AS missing_rate<br>FROM Booking; |
| total_occupants_per_stay | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(total_occupants_per_stay) AS non_null_count,<br>(COUNT(*) - COUNT(total_occupants_per_stay)) AS missing_quantity,<br>((COUNT(*) - COUNT(total_occupants_per_stay)) / COUNT(*)) AS missing_rate<br>FROM Booking; |
| booking_changes | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(booking_changes) AS non_null_count,<br>(COUNT(*) - COUNT(booking_changes)) AS missing_quantity,<br>((COUNT(*) - COUNT(booking_changes)) / COUNT(*)) AS missing_rate<br>FROM Booking; |
| arrival_date_week_number | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(arrival_date_week_number) AS non_null_count,<br>(COUNT(*) - COUNT(arrival_date_week_number)) AS missing_quantity,<br>((COUNT(*) - COUNT(arrival_date_week_number)) / COUNT(*)) AS missing_rate<br>FROM Booking; |

**2.2g total nights table**

| Numeric Column Name | Missing Quantity and Rate |
|---|---|
| stays_in_weekend_nights | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(stays_in_weekend_nights) AS non_null_count,<br>(COUNT(*) - COUNT(stays_in_weekend_nights)) AS missing_quantity,<br>((COUNT(*) - COUNT(stays_in_weekend_nights)) / COUNT(*)) AS missing_rate<br>FROM totalnights; |
| stays_in_week_nights | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(stays_in_week_nights) AS non_null_count,<br>(COUNT(*) - COUNT(stays_in_week_nights)) AS missing_quantity,<br>((COUNT(*) - COUNT(stays_in_week_nights)) / COUNT(*)) AS missing_rate<br>FROM totalnights; |
| stays_in_total_nights | SELECT<br>COUNT(*) AS total_rows,<br>COUNT(Stay_in_total_nights) AS non_null_count,<br>(COUNT(*) - COUNT(Stay_in_total_nights)) AS missing_quantity,<br>((COUNT(*) - COUNT(Stay_in_total_nights)) / COUNT(*)) AS missing_rate<br>FROM totalnights; |

## 2.3 SQL Commands (a, b, c, d)

HotelTable

| Column | A | B | C |
|---|---|---|---|
| Hotel | select hotel,frequency,frequency/(select count(*) from buan6320.reservation) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.reservation) as cumulative_percent_frequency | | |

| | | | |
|---|---|---|---|
| | from ( select hotel, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select hotel, count, row_number() over (order by count) as nrow from( select hotel,count(hotel) as count from (select hotel from buan6320.reservation join buan6320.hotel on buan6320.reservation.hotel_id = buan6320.hotel.hotel_id )one group by hotel order by count asc ) t ) tt) ttt; | | |

Reservation table

| Column | A | B | C |
|---|---|---|---|
| reservation_status | select reservation_status,frequency,frequency/(select count(*) from buan6320.reservation) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.reservation) as | /* max */ select * from ( select reservation_status,count(reservation_status) as count from buan6320.reservation group by reservation_status order by count | select count(reservation_status) from (select ifnull(reservation_status,'Missing') as reservation_status from buan6320.reservation)t where |

| | | | |
|---|---|---|---|
| | cumulative_percent_frequency from ( select reservation_status, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select reservation_status, count, row_number() over (order by count) as nrow from( select reservation_status,count(reservation_status) as count from buan6320.reservation group by reservation_status order by count asc ) t ) tt) ttt; | asc)t where t.count = (select max(count) from (select reservation_status,count(reservation_status) as count from buan6320.reservation group by reservation_status order by count asc)tt); /* min */ select * from ( select reservation_status,count(reservation_status) as count from buan6320.reservation group by reservation_status order by count asc)t where t.count = (select min(count) from (select reservation_status,count(reservation_status) as count from buan6320.reservation group by reservation_status order by count asc)tt); | reservation_status = 'missing'; |
| assigned_room_type | select assigned_room_type,frequency,frequency/(select count(*) from buan6320.reservation) as frequency_percentile,cumulative_total,cumulative_total/(select | /* max */ select * from ( select assigned_room_type,count(assigned_room_type) as count from buan6320.reservation group by | select count(assigned_room_type) from (select ifnull(assigned_room_type,'Missing') as assigned_room_type from |

| | | | |
|---|---|---|---|
| | count(*) from buan6320.reservation) as cumulative_percent_frequency from ( select assigned_room_type, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select assigned_room_type, count, row_number() over (order by count) as nrow from( select assigned_room_type,count(assigned_room_type) as count from buan6320.reservation group by assigned_room_type order by count asc ) t ) tt) ttt; | assigned_room_type order by count asc)t where t.count = (select max(count) from (select assigned_room_type,count(assigned_room_type) as count from buan6320.reservation group by assigned_room_type order by count asc)tt); select * from ( select assigned_room_type,count(assigned_room_type) as count from buan6320.reservation group by assigned_room_type order by count asc)t where t.count = (select min(count) from (select assigned_room_type,count(assigned_room_type) as count from buan6320.reservation group by assigned_room_type order by count asc)tt); | buan6320.reservation)t where assigned_room_type = 'missing'; |
| market_segment | select market_segment, frequency,frequency/(select count(*) from buan6320.reservation) as frequency_percentile,cumulative_total,cumulative_total/(select | /* max */ select * from ( select market_segment, count(market_segment) as count from buan6320.reservation group by market_segment | select count(market_segment) from (select ifnull(market_segment,'Missing') as market_segment from buan6320.reserv |

| | | | |
|---|---|---|---|
| | count(*) from buan6320.reservation) as cumulative_percent_frequency from ( select market_segment, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select market_segment, count, row_number() over (order by count) as nrow from( select market_segment, count(market_segment) as count from buan6320.reservation group by market_segment order by count asc ) t ) tt) ttt; | order by count asc)t where t.count = (select max(count) from (select market_segment, count(market_segment) as count from buan6320.reservation group by market_segment order by count asc)tt); /* min */ select * from ( select market_segment, count(market_segment) as count from buan6320.reservation group by market_segment order by count asc)t where t.count = (select min(count) from (select market_segment, count(market_segment) as count from buan6320.reservation group by market_segment order by count asc)tt); | ation)t where market_segment = 'missing'; |
| distribution_channel | select distribution_channel,frequency,frequency/(select count(*) from buan6320.reservation) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.reservation) as cumulative_percent_frequency from ( select | /* max */ select * from ( select distribution_channel,count(distribution_channel) as count from buan6320.reservation group by distribution_channel order by count asc)t where t.count = (select max(count) from | /* 3c distribution_channel */ select count(distribution_channel) from (select ifnull(distribution_channel,'Missing') as distribution_channel from buan6320.reservation)t where |

|  |  |  |  |
|---|---|---|---|
|  | distribution_channel, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select distribution_channel, count, row_number() over (order by count) as nrow from( select distribution_channel,count(distribution_channel) as count from buan6320.reservation group by distribution_channel order by count asc ) t ) tt) ttt; | (select distribution_channel,count(distribution_channel) as count from buan6320.reservation group by distribution_channel order by count asc)tt); /* min */ select * from ( select distribution_channel,count(distribution_channel) as count from buan6320.reservation group by distribution_channel order by count asc)t where t.count = (select min(count) from (select distribution_channel,count(distribution_channel) as count from buan6320.reservation group by distribution_channel order by count asc)tt); | distribution_channel = 'missing'; |

Customer table

| Column | A | B | C |
|---|---|---|---|
| customer_type | select customer_type,frequency,frequency/(select count(*) from buan6320.customer) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.customer) as | /* max */ select * from ( select customer_type,count(customer_type) as count from buan6320.customer group by customer_type order by count asc)t where t.count = (select | select count(customer_type) from (select ifnull(customer_type,'Missing') as customer_type from buan6320.customer)t where customer_type = 'missing'; |

| | | | |
|---|---|---|---|
| | cumulative_percent_frequency from ( select customer_type, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select customer_type, count, row_number() over (order by count) as nrow from( select customer_type,count(customer_type) as count from buan6320.customer group by customer_type order by count asc ) t ) tt) ttt; | max(count) from (select customer_type,count(customer_type) as count from buan6320.customer group by customer_type order by count asc)tt); /* min */ select * from ( select customer_type,count(customer_type) as count from buan6320.customer group by customer_type order by count asc)t where t.count = (select min(count) from (select customer_type,count(customer_type) as count from buan6320.customer group by customer_type order by count asc)tt); | |
| country | select country,frequency,frequency/(select count(*) from buan6320.customer) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.customer) as cumulative_percent_frequency from ( select country, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select country, count, row_number() over (order by | /* max */ select * from ( select country,count(country) as count from (select ifnull(country,'Missing') as country from buan6320.customer) one group by country order by count asc)t where t.count = (select max(count) from (select country,count(country) as count from (select ifnull(country,'Missing') as country from buan6320.customer) one group by country | select count(country) from (select ifnull(country,'Missing') as country from buan6320.customer)t where country = 'missing'; |

| | | | |
|---|---|---|---|
| | count) as nrow<br>from(<br><br>select<br>country,count(countr<br>y) as count<br>from (select<br>ifnull(country,'Missin<br>g') as country<br>from<br>buan6320.customer)<br>one<br>group by country<br>order by count asc<br><br>) t ) tt) ttt<br>order by<br>cumulative_total<br>desc; | order by count<br>asc)tt);<br><br>/* min */<br>select * from (<br>select<br>country,count(countr<br>y) as count<br>from (select<br>ifnull(country,'Missin<br>g') as country<br>from<br>buan6320.customer)<br>one<br>group by country<br>order by count asc)t<br>where t.count =<br>(select min(count)<br>from (select<br>country,count(countr<br>y) as count<br>from (select<br>ifnull(country,'Missin<br>g') as country<br>from<br>buan6320.customer)<br>one<br>group by country<br>order by count<br>asc)tt); | |
| is_repeated_guest | select<br>is_repeated_guest,fr<br>equency,frequency/(<br>select count(*) from<br>buan6320.customer)<br>as<br>frequency_percentil<br>e,cumulative_total,c<br>umulative_total/(sele<br>ct count(*) from<br>buan6320.customer)<br>as<br>cumulative_percent_<br>frequency<br>from (<br>select<br>is_repeated_guest,<br>count as<br>frequency,nrow,<br>sum(count) over<br>(order by nrow) as<br>cumulative_total<br>from (<br><br>select<br>is_repeated_guest, | | |

| | | | |
|---|---|---|---|
| | count, row_number() over (order by count) as nrow from( select is_repeated_guest,count(is_repeated_guest) as count from buan6320.customer group by is_repeated_guest order by count asc ) t ) tt) ttt; | | |

Booking Table

| Column | A | B | C |
|---|---|---|---|
| meal | select meal,frequency,frequency/(select count(*) from buan6320.booking) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.booking) as cumulative_percent_frequency from ( select meal, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select meal, count, row_number() over (order by count) as nrow from( | /* max */ select * from ( select meal,count(meal) as count from buan6320.booking group by meal order by count asc)t where t.count = (select max(count) from (select meal,count(meal) as count from buan6320.booking group by meal order by count asc)tt); /* min */ select * from ( select meal,count(meal) as count from buan6320.booking | select count(meal) from (select ifnull(meal,'Missing') as meal from buan6320.booking)t where meal = 'missing'; |

| | | | |
|---|---|---|---|
| | select meal,count(meal) as count from buan6320.booking group by meal order by count asc ) t ) tt) ttt; | group by meal order by count asc)t where t.count = (select min(count) from (select meal,count(meal) as count from buan6320.booking group by meal order by count asc)tt); | |
| Arrival_Date | select arrival_date_month,frequency,frequency/(select count(*) from buan6320.booking) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.booking) as cumulative_percent_frequency from ( select arrival_date_month, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select arrival_date_month, count, row_number() over (order by count) as nrow from( select arrival_date_month,count(arrival_date_month) as count from (select arrival_date_month from buan6320.booking join buan6320.arrivaldate on buan6320.booking.arrival_date = buan6320.arrivaldate.arrival_date )one group by arrival_date_month order by count asc ) t ) tt) ttt | select * from ( select arrival_date_month,count(arrival_date_month) as count from (select arrival_date_month from buan6320.booking join buan6320.arrivaldate on buan6320.booking.arrival_date = buan6320.arrivaldate.arrival_date )one group by arrival_date_month order by count asc)t where t.count = (select max(count) from (select arrival_date_month,count(arrival_date_month) as count from (select arrival_date_month from buan6320.booking join buan6320.arrivaldate on buan6320.booking.arrival_date = buan6320.arrivaldate.arrival_date )one group by arrival_date_month order by count asc)tt); select * from ( select arrival_date_month,count(arrival_date_month) as count from (select arrival_date_month from buan6320.booking | |

| | | | |
|---|---|---|---|
| | | join buan6320.arrivaldate on buan6320.booking.arrival_date = buan6320.arrivaldate.arrival_date )one group by arrival_date_month order by count asc)t where t.count = (select min(count) from (select arrival_date_month,count(arrival_date_month) as count from (select arrival_date_month from buan6320.booking join buan6320.arrivaldate on buan6320.booking.arrival_date = buan6320.arrivaldate.arrival_date )one group by arrival_date_month order by count asc)tt); | |
| Agent | select agent,frequency,frequency/(select count(*) from buan6320.booking) as frequency_percentile,cumulative_total,cumulative_total/(select count(*) from buan6320.booking) as cumulative_percent_frequency from ( select agent, count as frequency,nrow, sum(count) over (order by nrow) as cumulative_total from ( select agent, count, row_number() over (order by count) as nrow from( select agent,count(agent) as count from (select ifnull(agent,'Missing') | /* max */ select * from ( select agent,count(agent) as count from (select ifnull(agent,'Missing') as agent from buan6320.booking)one group by agent order by count asc)t where t.count = (select max(count) from (select agent,count(agent) as count from (select ifnull(agent,'Missing') as agent from buan6320.booking)one group by agent order by count asc)tt); /* min */ select * from ( select | select count(agent) from (select ifnull(agent,'Missing') as agent from buan6320.booking)t where agent = 'missing'; |

| | | | |
|---|---|---|---|
| | as agent<br>from<br>buan6320.booking)on<br>e<br>group by agent<br>order by count asc<br><br>) t ) tt) ttt<br>order by<br>cumulative_total desc; | agent,count(agent) as<br>count<br>from (select<br>ifnull(agent,'Missing')<br>as agent<br>from<br>buan6320.booking)on<br>e<br>group by agent<br>order by count asc)t<br>where t.count = (select<br>min(count) from<br>(select<br>agent,count(agent) as<br>count<br>from (select<br>ifnull(agent,'Missing')<br>as agent<br>from<br>buan6320.booking)on<br>e<br>group by agent<br>order by count asc)tt); | |
| deposit_type | select<br>deposit_type,frequenc<br>y,frequency/(select<br>count(*) from<br>buan6320.booking) as<br>frequency_percentile,c<br>umulative_total,cumul<br>ative_total/(select<br>count(*) from<br>buan6320.booking) as<br>cumulative_percent_fr<br>equency<br>from (<br>select deposit_type,<br>count as<br>frequency,nrow,<br>sum(count) over (order<br>by nrow) as<br>cumulative_total<br>from (<br><br>select deposit_type,<br>count, row_number()<br>over (order by count)<br>as nrow<br>from(<br><br>select<br>deposit_type,count(de<br>posit_type) as count<br>from<br>buan6320.booking<br>group by deposit_type<br>order by count asc<br><br>) t ) tt) ttt; | /* max */<br>select * from (<br>select<br>deposit_type,count(de<br>posit_type) as count<br>from<br>buan6320.booking<br>group by deposit_type<br>order by count asc)t<br>where t.count = (select<br>max(count) from<br>(select<br>deposit_type,count(de<br>posit_type) as count<br>from<br>buan6320.booking<br>group by deposit_type<br>order by count asc)tt);<br><br>/*min */<br><br>select * from (<br>select<br>deposit_type,count(de<br>posit_type) as count<br>from<br>buan6320.booking<br>group by deposit_type<br>order by count asc)t<br>where t.count = (select<br>min(count) from<br>(select<br>deposit_type,count(de<br>posit_type) as count<br>from<br>buan6320.booking<br>group by deposit_type<br>order by count asc)tt); | select<br>count(deposit_type)<br>from<br>(select<br>ifnull(deposit_type,'Mis<br>sing') as deposit_type<br>from<br>buan6320.booking)t<br>where deposit_type =<br>'missing'; |

## 2.4 SQL Commands (a, b, c, d, e)

Reservation table step 2.4 a

| Primary Key Column Name | SQL Command |
|---|---|
| reservation_id | INSERT INTO project2.reservation<br>VALUES (1119390,1,1,'Check-Out','2017-9-07','A', 'Online TA','TA/TO',10119490) |

Reservation table step 2.4 b

| Numeric Value Column Name | SQL Command |
|---|---|
| room_id | UPDATE project2.reservation<br>SET room_id = 'THREE'<br>WHERE reservation_id = 1000001 |

Reservation table step 2.4 c

| Character Value Column Name | SQL Command |
|---|---|
| distribution_channel | UPDATE project2.reservation<br>SET distribution_channel = 15<br>WHERE reservation_id = 1000001 |

Reservation table step 2.4 d

| Foreign Key Column Name | SQL Command |
|---|---|
| booking_id | INSERT INTO project2.reservation<br>VALUES (1000000,2,3,'Check-Out','2015-07-01','C', 'Direct','Direct',10000100)<br><br>Delete from project2.reservation<br>where booking_id = 10000101<br><br>INSERT INTO project2.reservation<br>VALUES (1000001,2,3,'Check-Out','2015-07-01','C', 'Direct','Direct',10000101)<br><br>UPDATE project2.reservation<br>SET booking_id = 15<br>WHERE reservation_id = 1000001 |

| | |
|---|---|
| | UPDATE project2.reservation<br>SET booking_id = 10000102<br>WHERE reservation_id = 1000001 |

Reservation table step 2.4 e

| Query to re-assemble the original dataset | SQL Command |
|---|---|
| | SELECT<br>    r.reservation_id,<br>    r.hotel_id,<br>    h.hotel,<br>    r.room_id,<br>    ro.reserved_room_type,<br>    b.booking_id,<br>    b.lead_time,<br>    b.meal,<br>    b.arrival_date,<br>    b.agent,<br>    b.days_in_waiting_list,<br>    b.required_car_parking_spaces,<br>    b.total_of_special_requests,<br>    b.deposit_type,<br>    b.stays_in_weekend_nights,<br>    b.stays_in_week_nights,<br>    b.total_occupants_per_stay,<br>    b.booking_changes,<br>    b.arrival_date_week_number,<br>    ad.arrival_date_day_of_month,<br>    ad.arrival_date_year,<br>    ad.arrival_date_month,<br>    tn.Stay_in_total_nights,<br>    bd.booking_date,<br>    rev.revenue_id,<br>    rev.adr,<br>    rev.est_tot_rev_stay,<br>    rev.revenue_per_occupant,<br>    c.customer_id,<br>    c.adults,<br>    c.children,<br>    c.babies,<br>    c.customer_type,<br>    c.is_repeated_guest,<br>    c.previous_cancellations,<br>    c.previous_bookings_not_canceled,<br>    c.country,<br>    cs.is_canceled,<br><br>    r.reservation_status,<br>    r.reservation_status_date,<br>    r.assigned_room_type,<br>    r.market_segment,<br>    r.distribution_channel<br>FROM Reservation r<br>LEFT JOIN Hotel h ON r.hotel_id = h.hotel_id<br>LEFT JOIN Room ro ON r.room_id = ro.room_id<br>LEFT JOIN Booking b ON r.booking_id = b.booking_id<br>LEFT JOIN ArrivalDate ad ON b.arrival_date = ad.arrival_date<br>LEFT JOIN TotalNights tn ON b.stays_in_weekend_nights = tn.stays_in_weekend_nights<br>                    AND b.stays_in_week_nights = tn.stays_in_week_nights<br>LEFT JOIN BookingDate bd ON b.lead_time = bd.lead_time<br>                      AND b.arrival_date = bd.arrival_date |

| | |
|---|---|
| | LEFT JOIN Revenue rev ON r.reservation_id = rev.reservation_id LEFT JOIN Customer c ON rev.customer_id = c.customer_id LEFT JOIN CancellationStatus cs ON r.reservation_status = cs.reservation_status; |