4-3.1 Hotel Exercises

Suppose we have the following four relations:

HOTEL (hotel_no, hotel_name, hotel_city)

ROOM (room_no, hotel_no, room_type, room_price)

BOOKING (hotel_no, guest_no, bdate_from, bdate_to, room_no)

GUEST (guest_no, guest_name, guest_address)

Write the relational algebra for the following queries (your answer **must** show an understanding of query efficiency i.e. you must not make use of unnecessary joins, nor carry attributes and tuples up through the query which are not necessary):

project: π

select: σ

1. List the number and name for all hotels

<aside> 💡 PROJECT hotel_no, hotel_name(HOTEL)

</aside>

1. List all single rooms with a price below $50

<aside> 💡 (SELECT room_type=”single”(SELECT room_price <$50(ROOM)))

</aside>

1. List the numbers and names of all hotels in Melbourne

<aside> 💡 In relational algebra, the order of operations is typically from the inside out, following the order of operations similar to mathematics. Here's the order in which the steps would come for the query "List the numbers and names of all hotels in Melbourne":

1. Selection (σ): Select rows where `hotel_city` is equal to "Melbourne".
2. Projection (π): Project the `hotel_no` and `hotel_name` attributes from the result of the selection operation.

R3 = PROJECT hotel_no, hotel_name(SELECT hotel_city = “Melbourne” (HOTEL))

OR

R3a = SELECT hotel_city = “Melbourne” (HOTEL)

R3 = PROJECT hotel_no , hotel_name(R3a)

</aside>

1. List all numbers and names of hotels which have a presidential suite room

HOTEL (hotel_no, hotel_name, hotel_city)

ROOM (room_no, hotel_no, room_type, room_price)

BOOKING (hotel_no, guest_no, bdate_from, bdate_to, room_no)

GUEST (guest_no, guest_name, guest_address)

<aside> 💡 A = PROJECT π_hotel_no(SELECT σ_room_type = “Presidential suite”(ROOM))

Select the room types of presidential suite and then only project the hotel numbers that have the room type=PSDSUITE

B = PROJECT π_hotel_no,hotel_name(HOTEL)

Grab all the hotel number and hotel names

C = A ⋈ B

Now, we join the 2 relations. In natural join, it looks at COMMON/same columns with same name and sees what values in the columns are the SAME. So, eventually we get all hotel numbers that have roomtype=PSDSUITE but also the name of the hotels.

</aside>

1. List the price and type of all rooms at the Grosvenor Hotel

HOTEL (hotel_no, hotel_name, hotel_city)

ROOM (room_no, hotel_no, room_type, room_price)

BOOKING (hotel_no, guest_no, bdate_from, bdate_to, room_no)

GUEST (guest_no, guest_name, guest_address)

<aside> 💡 SELECT hotel_name = “Grosvenor Hotel”

A = PROJECT hotel_no (SELECT hotel_name = “Grosvenor Hotel”(HOTEL))

This gives me the hotel numbers of all hotels with name = Grosvenor

REASON WHY WE WANT TO PROJECT HOTEL_NO IS THAT HOTEL_NO ALSO APPEARS AS A FOREGIN KEY IN THE ROOM RELATION. THUS, WE CAN USE JOIN TO LOOK FORE COMMON VALUES BETWEEN COMMON COLUMNS

Then we get:

PROJECT room_type, room_price(A ⋈ (PROJECT hotel_no,room_type,room_price(ROOM)))