Query design (DDD)

A travel agency uses a relational database to store details on a booking system.

It stores details of Scottish holiday resorts, hotels in each resort, customers and their bookings. These details are stored in four separate entities.

The attributes stored in each entity are shown below.

Resort	Hotel	Customer	Booking
resortID	hotelRef	customerNo	hotelRef*
resortName	hotelName	firstname	customerNo*
resortType	resortID *	surname	startDate
	starRating	address	numberOfNights
	seasonStartDate	town	numberInParty
	mealPlan	postcode	
	checkInTime		
	pricePersonNight		

The design of an SQL query should indicate:

- the fields and/or calculations required
- the table(s) or query(-ies) needed to provide the details required
- any search criteria to be applied
- what grouping is needed (if appropriate)
- the criteria to be applied to the grouping (if appropriate)
- the field(s) used to sort the data and the type(s) of sort required

Encourage candidates to plan — this helps to reduce the amount of frustration they may otherwise encounter when working with the SQL code.

Candidates can use a simple table template to indicate the planned design of the SQL query, see the following examples.

Example 1: HAVING with GROUPING and row COUNT

Display the resort name and number of hotels in any resort that has at least two hotels.

Field(s)/ calculation(s)	resortName, Number of Hotels = COUNT(*)
Table(s) query(-ies)	Resort, Hotel
Search criteria	
Grouping	resortName
Having	COUNT(*) >= 2
Sort order	

Example 2: HAVING with GROUPING and sort

Display the full name and the total cost of all bookings for each customer. The query should only list details of customers whose total cost exceeds £2000 and should list the details of the biggest spending customer first.

Field(s)/ calculation(s)	firstName, surname, Total cost of all Bookings = SUM(pricePersonNight * numberNights * numberInParty)
Table(s) query(-ies)	Customer, Booking, Hotel
Search criteria	
Grouping	firstName, surname
Having	SUM(pricePersonNight * numberNights * numberInParty) >= 2000
Sort order	SUM(pricePersonNight * numberNights * numberInParty) DESC

Example 3: HAVING with conditional statement

Display the average price per person, per night for each holiday resort. Display only those resorts with an average price per person, per night that exceeds £100.

Field(s)/ calculation(s)	resortName, Average Price = AVG(pricePersonNight)
Table(s) query(-ies)	Resort, Hotel
Search criteria	
Grouping	resortName
Having	AVG(pricePersonNight) > 100

Sort order	

Example 4: NOT operator

Display the name and type of non-coastal resort, together with the name and meal plan for each hotel that meets these criteria.

Field(s)/ calculation(s)	resortName, resortType, hotelName, mealPlan
Table(s) query(-ies)	Resort, Hotel
Search criteria	resortType NOT "coastal"
Grouping	
Having	
Sort order	

Example 5: BETWEEN operator with numeric values

Display the full name and total number of bookings made by each customer who has made between two and four bookings.

Field(s)/ calculation(s)	firstName, surname, Total Bookings = COUNT(*)
Table(s) query(-ies)	Customer, Booking
Search criteria	
Grouping	surname, firstName
Having	COUNT(*) BETWEEN 2 and 4;
Sort order	

Example 6: BETWEEN operator with text

Display the surname, postcode, and town of customers who live in towns that begin with the letters ‘E’ through to ‘M’. The query should list customers in alphabetical order of town.

Field(s)/ calculation(s)	surname, postcode, town
Table(s) query(-ies)	Customer
Search criteria	town BETWEEN "E" and "M"
Grouping	
Having	

Sort order	town ASC

Example 7: IN operator

Display the hotel name and meal plan for hotels that offer room only, half board or full board.

Field(s)/ calculation(s)	hotelName, mealPlan
Table(s) query(-ies)	Hotel
Search criteria	mealPlan IN the list ("Room Only", "Half Board", "Full Board")
Grouping	
Having	
Sort order	

Example 8: NOT with the IN operator

Display the name and type of resorts that are neither city nor country resorts.

Field(s)/ calculation(s)	resortName, resortType
Table(s) query(-ies)	Resort
Search criteria	resortType NOT IN the list ("city", "country");
Grouping	
Having	
Sort order	

Example 9: subquery in the where clause

Display the hotel name, star rating, and price per person for the most expensive hotel.

<table>
<tr><td colspan="2">Field(s)/ calculation(s)</td><td colspan="3">hotelName, starRating, pricePersonNight</td></tr>
<tr><td colspan="2">Table(s) query(-ies)</td><td colspan="3">Hotel</td></tr>
<tr><td rowspan="2">Search criteria</td><td rowspan="2">pricePersonNight =</td><td rowspan="2">Inner query</td><td>Field(s)/ calculation(s)</td><td>MAX(pricePersonNight)</td></tr>
<tr><td>Table(s)</td><td>Hotel</td></tr>
</table>

			Search criteria	
Grouping				
Having				
Sort order				

Example 10: subquery in the where clause

Display the resort name, hotel name, and star rating of all hotels that have a below-average star rating.

Field(s)/ calculation(s)	resortName, hotelName, starRating			
Table(s) query(-ies)	Resort, Hotel			
Search criteria	starRating <	Inner query	Field(s)/ calculation(s)	AVG(starRating)
			Table(s)	Hotel
			Search criteria	
Grouping				
Having				
Sort order				

Example 11: subquery using the NOT operator

Display the full name and postcode of the customer who booked the same hotel as the customer with ID 111.

Field(s)/ calculation(s)	firstName, surname, postcode			
Table(s) query(-ies)	Customer, Booking			
Search criteria	customerNo NOT 111			
	AND hotelRef =	Inner query	Field(s)/ calculation(s)	hotelRef
			Table(s)	Booking
			Search criteria	customerNo = 111
Grouping				
Having				

Sort order	

Example 12: subquery using the IN operator

Display the hotel name and star rating of all hotels booked by the customer with ID 315.

Field(s)/ calculation(s)	hotelName, starRating			
Table(s) query(-ies)	Hotel			
Search criteria	hotelName IN	Inner query	Field(s)/ calculation(s)	hotelName
			Table(s)	Hotel, Booking
			Search criteria	customerNo = 315
Grouping				
Having				
Sort order				

Example 13: subquery using the NOT and IN operators

Display the names and types of resort **not** booked by the customer with ID 315.

Field(s)/ calculation(s)	resortName, resortType			
Table(s) query(-ies)	Resort			
Search criteria	resortName NOT IN	Inner query	Field(s)/ calculation(s)	resortName
			Table(s)	Resort, Hotel, Booking
			Search criteria	customerNo = 315
Grouping				
Having				
Sort order				

Example 14: subquery using the ANY operator

Display the customer number, hotel reference, and booking cost for any booking that costs more than any bookings made by customers with surnames Lowden, Shawfair or Sheriffhall.

Field(s)/ calculation(s)	customerNo, hotelRef, Booking Cost = pricePersonNight * numberNights * numberInParty			
Table(s) query(-ies)	Booking, Hotel			
Search criteria	pricePersonNight * numberNights * numberInParty > ANY	Inner query	Field(s)/ calculation(s)	pricePersonNight * numberNights * numberInParty
			Table(s)	Booking, Hotel, Customer
			Search criteria	surname in ("Sheriffhall", "Lowden", "Shawfair")
Grouping				
Having				
Sort order				

Example 15: subquery using the EXISTS operator

Display the details (hotel name, star rating, meal plan and resort name) of all 3-star hotel bookings. The query should list the hotels in alphabetical order of meal plan.

Field(s)/ calculation(s)	hotelname, mealPlan, starRating, resortName			
Table(s) query(-ies)	Hotel, Resort			
Search criteria	starRating = 3			
	AND EXISTS	Inner query	Field(s)/ calculation(s)	*
			Table(s)	Booking
			Search criteria	
Grouping				
Having				
Sort order	mealPlan ASC			

Example 16: subquery using the NOT and EXISTS operators

Display the full name and address of customers who have never made a booking.

<table>
<tr><td colspan="2">Field(s)/ calculation(s)</td><td colspan="4">firstName, surname, address</td></tr>
<tr><td colspan="2">Table(s) query(-ies)</td><td colspan="4">Customer</td></tr>
<tr><td rowspan="3">Search criteria</td><td rowspan="3">NOT EXISTS</td><td rowspan="3">Inner query</td><td>Field(s)/ calculation(s)</td><td>*</td></tr>
<tr><td>Table(s)</td><td>Booking</td></tr>
<tr><td>Search criteria</td><td></td></tr>
<tr><td>Grouping</td><td colspan="4"></td></tr>
<tr><td>Having</td><td colspan="4"></td></tr>
<tr><td>Sort order</td><td colspan="4"></td></tr>
</table>

Example 17: query requiring two subqueries

Display the name, star rating, and total number of customer nights booked for hotels that have:

- a total number of customer nights booked that is more than the total number of nights booked by the customer with ID 290 (number of nights booked multiplied by number in party)

and

- a star rating which is less than that of the hotel with the highest star rating

The query should list the hotels from lowest star rating to the highest.

<table>
<tr><td colspan="2">Field(s)/ calculation(s)</td><td colspan="3">hotelName, starRating, Nights x Number in Party = SUM(numberNights*numberInParty)</td></tr>
<tr><td colspan="2">Table(s) query(-ies)</td><td colspan="3">Hotel, Booking</td></tr>
<tr><td rowspan="6">Search criteria</td><td rowspan="3">numberNights * numberInParty ></td><td rowspan="3">Inner query</td><td>Field(s)/ calculation(s)</td><td>SUM(numberNights * numberInParty)</td></tr>
<tr><td>Table(s)</td><td>Booking</td></tr>
<tr><td>Search criteria</td><td>customerNo = 290</td></tr>
<tr><td rowspan="3">AND starRating <</td><td rowspan="3">Inner query</td><td>Field(s)/ calculation(s)</td><td>MAX(starRating)</td></tr>
<tr><td>Table(s)</td><td>Hotel</td></tr>
<tr><td>Search criteria</td><td></td></tr>
<tr><td>Grouping</td><td colspan="3">hotelName, starRating</td></tr>
</table>

Sort order	starRating ASC