

## OUTPUT OF ONLINE-TICKET-SYSTEM

### Task 1: DATABASE DESIGN

```
1 • SELECT * FROM ticketbookingsystem.booking;
```

Result Grid						
Filter Rows:						
Edit: Export/Import						
booking_id	customer_id	event_id	num_tickets	total_cost	booking_date	
502	202	102	3	2550	2025-01-02	
503	203	103	4	20000	2025-01-26	
504	204	104	2	1600	2025-02-05	
505	205	105	6	12000	2025-05-02	
506	206	106	1	2000	2025-12-02	
507	207	107	5	3000	2025-06-08	
508	208	108	4	4000	2025-09-08	
509	209	109	2	2000	2025-11-02	
510	210	110	3	6000	2025-03-06	
913	201	101	5	7500	2025-03-18	
NULL	NULL	NULL	NULL	NULL	NULL	

Result Grid					
Filter Rows:					
Edit: Export/Import					
customer_id	customer_name	email	phone_number	booking_id	
201	Rohan	rohan@gmail.com	8632157942	913	
202	Mruganka	mruganka@gmail.com	9784563211	502	
203	Om	om@gmail.com	987654321	503	
204	Rutuja	rutuja@gmail.com	9531247866	504	
205	Sunny	sunny@gmail.com	8745963215	505	
206	Vaishnavi	vaishnavi@gmail.com	8796321455	506	
207	Atharva	atharva@gmail.com	9863214789	507	
208	Isha	isha@gmail.com	9874563215	508	
209	Yash	yash@gmail.com	8963214578	509	
210	Shravani	shravani@gmail.com	9863257412	510	
211	Rahul	rahul@gmail.com	8632157000	NULL	
NULL	NULL	NULL	NULL	NULL	

customer 1 x

Result Grid										
Filter Rows:										
Edit: Export/Import										
Wrap Cell Content: I A										
event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	booking_id	event_type	
101	Coldplay: Music of the Spheres Tour	2025-04-18	18:00:00	2	1000	250	500	913	Concert	
102	Bengaluru Open	2025-06-06	10:00:00	5	2000	1100	850	502	Sports	
103	Ed Sheeran: Mathematics Tour	2025-12-25	17:00:00	1	5000	1000	5000	503	Concert	
104	YJHD: Premiere	2025-09-26	18:00:00	6	300	50	800	504	Movie	
105	Khelo India Winter Games	2025-11-02	10:00:00	10	1000	200	2000	505	Sports	
106	Guns N Roses	2025-05-26	16:00:00	8	3000	1000	2000	506	Concert	
107	3 Idiots	2025-06-20	17:00:00	7	300	50	600	507	Movie	
108	ISSF Junior World	2025-03-31	11:00:00	4	1000	200	1000	508	Sports	
109	Lord of the Rings	2025-08-25	19:00:00	3	500	50	1000	509	Movie	
110	21 Savage	2025-10-13	17:00:00	3	4000	1000	2000	510	Concert	
111	World Cup	2025-12-25	17:00:00	1	5000	1000	5000	503	Sports	
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	

	venue_id	venue_name	address
▶	1	Whispering Grove	Pune
	2	Serene Vista	Mumbai
	3	Emerald Pavillion	Delhi
	4	Sunstone Terrace	Chennai
	5	Moonlight Garden	Bangalore
	6	Azure Canopy	Chandigarh
	7	Crystal Stream	Hyderabad
	8	Starlight Studio	Kolkata
	9	Curated Space	Jaipur
	10	Regal Hall	Ahmedabad
•	NULL	NULL	NULL

## Task 2: Select, Where, Between, AND, LIKE:

```

89 • select * from event;
90 • select event_name from event;
91

```

event_name
▶ Coldplay: Music of the Spheres Tour
Bengaluru Open Bengaluru Open
Ed Sheeran: Mathematics Tour
YJHD: Premiere
Khelo India Winter Games
Guns N Roses
3 Idiots
ISSF Junior World
Lord of the Rings
21 Savage
World Cup

```

92 /*3. Write a SQL query to select events with available tickets. */
93 • select event_name, event_id, available_seats from event where available_seats > 0;
94

```

event_name	event_id	available_seats
▶ Coldplay: Music of the Spheres Tour	101	250
Bengaluru Open	102	1100
Ed Sheeran: Mathematics Tour	103	1000
YJHD: Premiere	104	50
Khelo India Winter Games	105	200
Guns N Roses	106	1000
3 Idiots	107	50
ISSF Junior World	108	200
Lord of the Rings	109	50
21 Savage	110	1000
World Cup	111	1000
• NULL	NULL	NULL

```

95      /*4. Write a SQL query to select events name partial match with 'cup'. */
96 •   insert into event values(111, 'World Cup', '2025-12-25', '17:00:00', 001, 5000, 1000, 5000, 503, 'Sports');
97 •   select * from event where event_name like '%cup%';
98

```

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	booking_id	event_type
111	World Cup	2025-12-25	17:00:00	1	5000	1000	5000	503	Sports
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```

99      /*5. Write a SQL query to select events with ticket price range is between 1000 to 2500.*/
100 •   select * from event where ticket_price >= 1000 and ticket_price <= 2500;
101

```

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	booking_id	event_type
105	Khelo India Winter Games	2025-11-02	10:00:00	10	1000	200	2000	505	Sports
106	Guns N Roses	2025-05-26	16:00:00	8	3000	1000	2000	506	Concert
108	ISSF Junior World	2025-03-31	11:00:00	4	1000	200	1000	508	Sports
109	Lord of the Rings	2025-08-25	19:00:00	3	500	50	1000	509	Movie
110	21 Savage	2025-10-13	17:00:00	3	4000	1000	2000	510	Concert
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```

102      /*6. Write a SQL query to retrieve events with dates falling within a specific range.*/
103 •   select * from event where event_date >= '2025-04-01' and event_date <= '2025-06-20';
104

```

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	booking_id	event_type
101	Coldplay: Music of the Spheres Tour	2025-04-18	18:00:00	2	1000	250	500	913	Concert
102	Bengaluru Open	2025-06-06	10:00:00	5	2000	1100	850	502	Sports
106	Guns N Roses	2025-05-26	16:00:00	8	3000	1000	2000	506	Concert
107	3 Idiots	2025-06-20	17:00:00	7	300	50	600	507	Movie
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```

105      /*7. Write a SQL query to retrieve events with available tickets that also have "Concert" in their name. */
106 •   select * from event where available_seats > 0 and event_type like '%Concert%';
107

```

event_id	event_name	event_date	event_time	venue_id	total_seats	available_seats	ticket_price	booking_id	event_type
101	Coldplay: Music of the Spheres Tour	2025-04-18	18:00:00	2	1000	250	500	913	Concert
103	Ed Sheeran: Mathematics Tour	2025-12-25	17:00:00	1	5000	1000	5000	503	Concert
106	Guns N Roses	2025-05-26	16:00:00	8	3000	1000	2000	506	Concert
110	21 Savage	2025-10-13	17:00:00	3	4000	1000	2000	510	Concert
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```

108      /*8. Write a SQL query to retrieve users in batches of 5, starting from the 6th user.*/
109 •   select * from customer limit 5 offset 5;
110

```

customer_id	customer_name	email	phone_number	booking_id
206	Vaishnavi	vaishnavi@gmail.com	8796321455	506
207	Atharva	atharva@gmail.com	9863214789	507
208	Isha	isha@gmail.com	9874563215	508
209	Yash	yash@gmail.com	8963214578	509
210	Shravani	shravani@gmail.com	9863257412	510
NULL	NULL	NULL	NULL	NULL

booking_id	customer_id	event_id	num_tickets	total_cost	booking_date
505	205	105	6	12000	2025-05-02
507	207	107	5	3000	2025-06-08
913	201	101	5	7500	2025-03-18
NULL	NULL	NULL	NULL	NULL	NULL

Result Grid		Filter Rows:	Edit:	Export/Import:	Wrap Cell Content:
	customer_id	customer_name	email	phone_number	booking_id
▶	211	Rahul	rahul@gmail.com	8632157000	NULL
✱	NULL	NULL	NULL	NULL	NULL

[illegible][illegible]



### Task 3: Aggregate functions, Having, Order By, GroupBy and Joins:

```
125 1. Write a SQL query to List Events and Their Average Ticket Prices. */
126 • select e.event_id, e.event_name, avg(ticket_price) as average_ticket_price from event e join booking b on e.event_id = b.event_id
127 group by e.event_id, e.event_name order by average_ticket_price desc;
128
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

event_id	event_name	average_ticket_price
103	Ed Sheeran: Mathematics Tour	5000
105	Khelo India Winter Games	2000
106	Guns N Roses	2000
110	21 Savage	2000
108	ISSF Junior World	1000
109	Lord of the Rings	1000
102	Bengaluru Open	850
104	YJHD: Premiere	800
107	3 Idiots	600
101	Coldplay: Music of the Spheres Tour	500

```
128
129 /*2. Write a SQL query to Calculate the Total Revenue Generated by Events. */
130 • select sum(total_cost) as total_revenue from booking;
131
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

total_revenue
60650

```
132 /*3. Write a SQL query to find the event with the highest ticket sales. */
133 • select e.event_id, e.event_name, max(num_tickets) as total_tickets_sold from booking b join event e on e.event_id = b.event_id
134 group by e.event_id, e.event_name order by total_tickets_sold desc limit 1;
135
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows: |

event_id	event_name	total_tickets_sold
105	Khelo India Winter Games	6

```
136 /*4. Write a SQL query to Calculate the Total Number of Tickets Sold for Each Event. */
137 • select e.event_id, e.event_name, sum(num_tickets) as total_tickets_sold from booking b join event e on e.event_id = b.event_id group by
138 e.event_id, e.event_name order by total_tickets_sold desc;
139
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

event_id	event_name	total_tickets_sold
105	Khelo India Winter Games	6
107	3 Idiots	5
101	Coldplay: Music of the Spheres Tour	5
103	Ed Sheeran: Mathematics Tour	4
108	ISSF Junior World	4
102	Bengaluru Open	3
110	21 Savage	3
104	YJHD: Premiere	2
109	Lord of the Rings	2
106	Guns N Roses	1

```
140 /*5. Write a SQL query to Find Events with No Ticket Sales. */
141 • select e.event_id, e.event_name from event e left join booking b on e.event_id = b.event_id where b.num_tickets is null;
142
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

event_id	event_name
111	World Cup

```

143  /*6. Write a SQL query to Find the User Who Has Booked the Most Tickets. */
144  • select c.customer_id, c.customer_name, sum(num_tickets) as total_tickets from booking b join customer c on c.customer_id = b.customer_id
145  group by customer_id order by total_tickets desc limit 1;
146

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
customer_id	customer_name	total_tickets		
205	Sunny	6		

```

147  /*7. Write a SQL query to calculate the average Ticket Price for Events in Each Venue. */
148  • select v.venue_id, v.venue_name, avg(e.ticket_price) as average_ticket_price from venue v join event e on v.venue_id = e.venue_id
149  join booking b on e.event_id = b.event_id group by v.venue_id, v.venue_name order by average_ticket_price desc;
150

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
venue_id	venue_name	average_ticket_price	
1	Whispering Grove	5000	
10	Regal Hall	2000	
8	Starlight Studio	2000	
3	Emerald Pavilion	1500	
4	Sunstone Terrace	1000	
5	Moonlight Garden	850	
6	Azure Canopy	800	
7	Crystal Stream	600	
2	Serene Vista	500	

```

151  /*8. Write a SQL query to calculate the total Number of Tickets Sold for Each Event Type.*/
152  • select e.event_type, sum(b.num_tickets) as total_tickets_sold from event e join booking b on e.event_id = b.event_id
153  group by e.event_type order by total_tickets_sold desc;
154

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
event_type	total_tickets_sold		
Sports	13		
Concert	13		
Movie	9		

```

155  /*9. Write a SQL query to calculate the total Revenue Generated by Events in Each Year. */
156  • select year(e.event_date) as event_year, sum(b.num_tickets * e.ticket_price) as total_revenue from event e
157  join booking b on e.event_id = b.event_id group by event_year order by event_year desc;
158

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
event_year	total_revenue		
2025	55650		

```

159  /*10. Write a SQL query to list users who have booked tickets for multiple events.*/
160  • select customer_id, count(distinct event_id) as event_count from booking group by customer_id having count(distinct event_id) > 1
161  order by event_count desc;
162

```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:
customer_id	event_count		

```

162
163  /* 11. Write a SQL query to calculate the Total Revenue Generated by Events for Each User. */
164 • select customer_id, sum(total_cost) as total_revenue from booking group by customer_id order by total_revenue desc;
165

```

Result Grid

Filter Rows:

Export:




Wrap Cell Content:

	customer_id	total_revenue
▶	203	20000
	205	12000
	201	7500
	210	6000
	208	4000
	207	3000
	202	2550
	206	2000
	209	2000
	204	1600

```

166  /*12. Write a SQL query to calculate the Average Ticket Price for Events in Each Category and Venue.*/
167 • select e.event_type, v.venue_name, avg(e.ticket_price) as average_ticket_price from event e join venue v on e.venue_id = v.venue_id
168 join booking b on e.event_id = b.event_id group by e.event_type, v.venue_name order by e.event_type, average_ticket_price desc;
169

```




Result Grid |  Filter Rows:  | Export:  | Wrap Cell Content: 

	event_type	venue_name	average_ticket_price
▶	Movie	Emerald Pavillion	1000
	Movie	Azure Canopy	800
	Movie	Crystal Stream	600
	Sports	Regal Hall	2000
	Sports	Sunstone Terrace	1000
	Sports	Moonlight Garden	850
	Concert	Whispering Grove	5000
	Concert	Starlight Studio	2000
	Concert	Emerald Pavillion	2000
	Concert	Serene Vista	500

```

170  /*13. Write a SQL query to list Users and the Total Number of Tickets They've Purchased in the Last 30
171  Days*/
172 • select c.customer_name, b.customer_id, sum(b.num_tickets) as total_tickets_purchased from booking b left join customer c on
173 c.customer_id = b.customer_id where b.booking_date >= curdate() - interval 30 day group by b.customer_id, c.customer_name
174 order by total_tickets_purchased desc;
175

```

Result Grid   Filter Rows:  | Export:  | Wrap Cell Content: [IA](#)

	customer_name	customer_id	total_tickets_purchased
▶	Sunny	205	6
	Atharva	207	5
	Rohan	201	5
	Isha	208	4
	Shravani	210	3
	Yash	209	2
	Vaishnavi	206	1

#### Task 4: Subquery and its types

```
177 1. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery*/
178 • select v.venue_id, v.venue_name,
179 (select avg(e.ticket_price) from booking b join event e on e.event_id = b.event_id where e.venue_id = v.venue_id)
180 as average_ticket_price from venue v order by average_ticket_price desc;
181
```

Result Grid Filter Rows: Export: Wrap Cell Content:

	venue_id	venue_name	average_ticket_price
▶	1	Whispering Grove	5000
	8	Starlight Studio	2000
	10	Regal Hall	2000
	3	Emerald Pavillion	1500
	4	Sunstone Terrace	1000
	5	Moonlight Garden	850
	6	Azure Canopy	800
	7	Crystal Stream	600
	2	Serene Vista	500
	9	Curated Space	NULL

```
182 /*2. Find Events with More Than 50% of Tickets Sold using subquery.*/
183 • select e.event_id, e.event_name from event e where(select sum(e.available_seats) from event e) > (0.5 * e.totat_seats);
184
```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:

	event_id	event_name
▶	101	Coldplay: Music of the Spheres Tour
	102	Bengaluru Open
	103	Ed Sheeran: Mathematics Tour
	104	YJHD: Premiere
	105	Khelo India Winter Games
	106	Guns N Roses
	107	3 Idiots
	108	ISSF Junior World
	109	Lord of the Rings
	110	21 Savage
	111	World Cup
*	NULL	NULL

```
185 /*3. Calculate the Total Number of Tickets Sold for Each Event. */
186 • select e.event_id, e.event_name, (select sum(b.num_tickets) from booking b where b.event_id = e.event_id)
187 as total_tickets_sold from event e order by total_tickets_sold desc;
188
```

Result Grid Filter Rows: Export: Wrap Cell Content:

	event_id	event_name	total_tickets_sold
▶	105	Khelo India Winter Games	6
	101	Coldplay: Music of the Spheres Tour	5
	107	3 Idiots	5
	103	Ed Sheeran: Mathematics Tour	4
	108	ISSF Junior World	4
	102	Bengaluru Open	3
	110	21 Savage	3
	104	YJHD: Premiere	2
	109	Lord of the Rings	2
	106	Guns N Roses	1
	111	World Cup	NULL

```
189 /*4. Find Users Who Have Not Booked Any Tickets Using a NOT EXISTS Subquery.*/
190 • select c.customer_id, c.customer_name from customer c where not exists(select 1 from booking b where b.customer_id = c.customer_id);
191
```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content:

	customer_id	customer_name
▶	211	Rahul
*	NULL	NULL



```

191
192 /*5. List Events with No Ticket Sales Using a NOT IN Subquery*/
193 • select e.event_id, e.event_name from event e where e.event_id not in (select b.event_id from booking b);
194

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

event_id	event_name
111	World Cup
NULL	NULL

```

195 /*6. Calculate the Total Number of Tickets Sold for Each Event Type Using a Subquery in the FROM Clause. */
196 • select e.event_type, sum(e.available_seats) as total_tickets_sold from event e
197 join (select b.event_id, sum(b.num_tickets) as total_tickets_sold from booking b group by b.event_id) as ticket_data on
198 e.event_id = ticket_data.event_id group by e.event_type order by total_tickets_sold desc;
199

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

event_type	total_tickets_sold
Concert	3250
Sports	1500
Movie	150

```

200 /*7. Find Events with Ticket Prices Higher Than the Average Ticket Price Using a Subquery in the WHERE Clause.*/
201 • select e.event_id, e.event_name, e.ticket_price from event e join booking b on e.event_id = b.event_id
202 where e.ticket_price > (select avg(e.ticket_price) from event e) order by e.ticket_price desc;
203

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

event_id	event_name	ticket_price
103	Ed Sheeran: Mathematics Tour	5000
105	Khelo India Winter Games	2000
106	Guns N Roses	2000
110	21 Savage	2000

```

204
209 /*9. List Users Who Have Booked Tickets for Events in a Given Venue Using a Subquery in the WHERE Clause.*/
210 • select c.customer_id, c.customer_name from customer c where c.customer_id in(select distinct b.customer_id from booking b
211 where b.event_id in(select e.event_id from event e where e.venue_id = (select v.venue_id from venue v where
212 v.venue_name = 'Serene Vista')));
213

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

customer_id	customer_name
201	Rohan
NULL	NULL

```

214 /*10. Calculate the Total Number of Tickets Sold for Each Event Category Using a Subquery with GROUP BY*/
215 • select e.event_type, (select sum(b.num_tickets) from booking b where b.event_id in(select e.event_id from event e
216 where e.event_id = b.event_id)) as total_tickets_sold from event e group by e.event_type order by total_tickets_sold desc;
217

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

event_type	total_tickets_sold
Concert	35
Sports	35
Movie	35

```

218  /*11. Find Users Who Have Booked Tickets for Events in each Month Using a Subquery with DATE_FORMAT.*/
219  • select distinct c.customer_id, c.customer_name, (select date_format(b.booking_date, '%Y-%m') from booking b where
220    b.customer_id = c.customer_id order by b.booking_date limit 1) as booking_month from customer c where c.customer_id in
221    (select distinct b.customer_id from booking b where b.booking_date is not null) order by booking_month, c.customer_id;
222

```

Result Grid			
Filter Rows:			
Export:   Wrap Cell Content: <a href="#">IA</a>			
customer_id	customer_name	booking_month	
202	Mruganka	2025-01	
203	Om	2025-01	
204	Rutuja	2025-02	
201	Rohan	2025-03	
210	Shravani	2025-03	
205	Sunny	2025-05	
207	Atharva	2025-06	
208	Isha	2025-09	
209	Yash	2025-11	
206	Vaishnavi	2025-12	

```

223  /*12. Calculate the Average Ticket Price for Events in Each Venue Using a Subquery */
224  • select v.venue_id, v.venue_name, (select avg(e.ticket_price) from event e where e.event_id in
225    (select e.event_id from event e where e.venue_id = v.venue_id)) as avg_ticket_price from venue v order by avg_ticket_price desc;
226

```

Result Grid			
Filter Rows:			
Export:   Wrap Cell Content: <a href="#">IA</a>			
venue_id	venue_name	avg_ticket_price	
1	Whispering Grove	5000	
8	Starlight Studio	2000	
10	Regal Hall	2000	
3	Emerald Pavilion	1500	
4	Sunstone Terrace	1000	
5	Moonlight Garden	850	
6	Azure Canopy	800	
7	Crystal Stream	600	
2	Serene Vista	500	
9	Curated Space	1000	