

# Air Cargo Analysis

## Course-end Project 2

### Description

Air Cargo is an aviation company that provides air transportation services for passengers and freight. Air Cargo uses its aircraft to provide different services with the help of partnerships or alliances with other airlines. The company wants to prepare reports on regular passengers, busiest routes, ticket sales details, and other scenarios to improve the ease of travel and booking for customers.

#### Project Objective:

You, as a DBA expert, need to focus on identifying the regular customers to provide offers, analyze the busiest route which helps to increase the number of aircraft required and prepare an analysis to determine the ticket sales details. This will ensure that the company improves its operability and becomes more customer-centric and a favorable choice for air travel.

**Note:** You must download the dataset from the course resource section in the LMS and create the tables to perform the above objective.

#### Dataset description:

**Customer:** Contains the information of customers

- customer\_id – ID of the customer
- first\_name – First name of the customer
- last\_name – Last name of the customer
- date\_of\_birth – Date of birth of the customer
- gender – Gender of the customer

**passengers\_on\_flights:** Contains information about the travel details

- aircraft\_id – ID of each aircraft in a brand
- route\_id – Route ID of from and to location
- customer\_id – ID of the customer
- depart – Departure place from the airport
- arrival – Arrival place in the airport
- seat\_num – Unique seat number for each passenger
- class\_id – ID of travel class
- travel\_date – Travel date of each passenger
- flight\_num – Specific flight number for each route

**ticket\_details:** Contains information about the ticket details

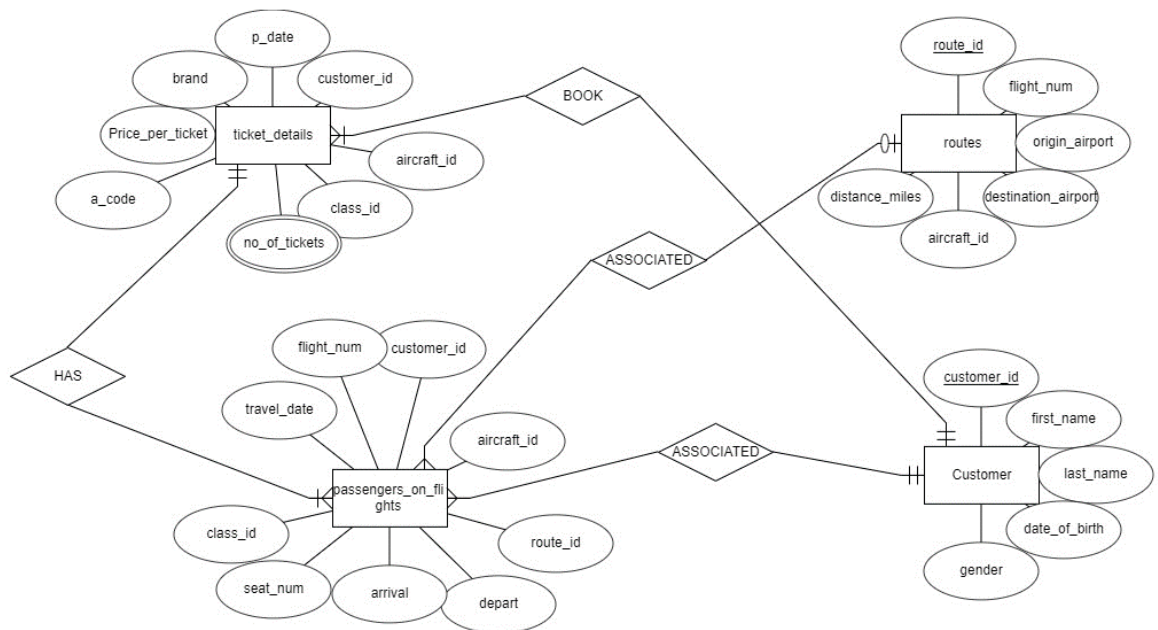
- p\_date – Ticket purchase date
- customer\_id – ID of the customer
- aircraft\_id – ID of each aircraft in a brand
- class\_id – ID of travel class
- no\_of\_tickets – Number of tickets purchased
- a\_code – Code of each airport
- price\_per\_ticket – Price of a ticket
- brand – Aviation service provider for each aircraft

**routes:** Contains information about the route details

- Route\_id – Route ID of from and to location
- Flight\_num – Specific flight number for each route
- Origin\_airport – Departure location
- Destination\_airport – Arrival location
- Aircraft\_id – ID of each aircraft in a brand
- Distance\_miles – Distance between departure and arrival location

**Following operations should be performed:**

1. Create an ER diagram for the given airlines database.



2. Write a query to create route\_details table using suitable data types for the fields, such as route\_id, flight\_num, origin\_airport, destination\_airport, aircraft\_id, and distance\_miles. Implement the check constraint for the flight number and unique constraint for the route\_id fields. Also, make sure that the distance miles field is greater than 0.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

Columns

Indexes

Foreign Keys

Triggers

ticket\_details

Columns

Indexes

Foreign Keys

Triggers

Views

businessclasscustomers

Stored Procedures

Functions

corporate

hospital

navair

Administration Schemas

Information

Table: routes

Columns:

Route\_id int PK

Flight\_num varchar(30)

Origin\_airport varchar(30)

Destination\_airport varchar(30)

Aircraft\_id varchar(30)

Distance\_miles decimal(10,2)

28 no\_of\_tickets int not null,

29 a\_code varchar(30) not null,

30 price\_per\_ticket float not null,

31 brand varchar(30) not null

32 );

33

34 create table routes(

35 Route\_id int not null primary key,

36 Flight\_num varchar(30) check ( Flight\_num REGEXP '^([0-9]{4})\$'),

37 Origin\_airport varchar(30) not null,

38 Destination\_airport varchar(30) not null,

39 Aircraft\_id varchar(30) not null,

40 Distance\_miles decimal(10,2) check (Distance\_miles > 0)

41 );

42

43

44 select \* from passengers\_on\_flights

45 where route\_id <=25;

46

47 Select count(\*) as no\_of\_passengers, sum(Price\_per\_ticket) as total\_revenue from ticket\_details

Output

Action Output

#	Time	Action	Message
24	19:04:52	SHOW DATABASES	OK
25	19:04:53	SHOW SESSION VARIABLES LI...	OK
26	19:04:53	SHOW COLUMNS FROM 'aircar...	OK
27	19:04:57	TRUNCATE TABLE 'aircargo'.no...	OK

3. Write a query to display all the passengers (customers) who have travelled in routes 01 to 25. Take data from the passengers\_on\_flights table.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

Columns

Indexes

Foreign Keys

Triggers

ticket\_details

Columns

Indexes

Foreign Keys

Triggers

Views

businessclasscustomers

Stored Procedures

Functions

corporate

hospital

navair

Administration Schemas

Information

Schema: aircargo

33

34 create table routes(

35 Route\_id int not null primary key,

36 Flight\_num varchar(30) check ( Flight\_num = '^([0-9]{4})\$'),

37 Origin\_airport varchar(30) not null,

38 Destination\_airport varchar(30) not null,

39 Aircraft\_id varchar(30) not null,

40 Distance\_miles decimal(10,2) check (Distance\_miles > 0)

41 );

42

43 select \* from passengers\_on\_flights

44 where route\_id <=25;

Result Grid

aircraft_id	route_id	customer_id	depart	arrival	seat_num	class_id	travel_date	flight_num
767-301ER	4	2	JFK	LAX	01E	Economy	2018-09-02	1114
ERJ142	9	1	DEN	LAX	01EP	Economy Plus	2019-12-26	1119
767-301ER	12	5	ABT	ADK	02B	Business	2018-07-02	1122
ERJ142	18	5	ANI	BGR	02E	Economy	2020-05-06	1128
767-301ER	5	4	LAX	JFK	02FC	First Class	2020-04-06	1115
767-301ER	20	7	AVL	BOI	03B	Business	2020-07-08	1130
ERJ142	22	5	BGR	BII	03E	Economy	2020-05-31	1132
767-301ER	4	4	JFK	LAX	03FC	First Class	2020-04-30	1114
767-301ER	5	11	LAX	JFK	04B	Business	2020-11-12	1115
A321	13	17	ABT	ADK	04EP	Economy Plus	2019-06-03	1123
767-301ER	15	9	CAK	ANI	04FC	First Class	2020-09-10	1125
767-301ER	4	11	JFK	LAX	05B	Business	2020-11-09	1114
A321	10	10	ANI	DEN	05E	Economy	2020-10-11	1120

passengers\_on\_flights 2 x

Output

4. Write a query to identify the number of passengers and total revenue in business class from the ticket\_details table.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

Filter objects

SCHEMAS

- aircargo
  - Tables
    - customer
    - passengers\_on\_flights
    - routes
    - ticket\_details
      - Columns
      - Indexes
      - Foreign Keys
      - Triggers
  - Views
  - Stored Procedures
  - Functions
- corporate
- hospital
- payroll
- sample
- school
- scitech\_emp
  - Tables

Administration Schemas

Information

Table: ticket\_details

Columns:

p_date	date
customer_id	int
aircraft_id	varchar(30)
class_id	varchar(30)
no_of_tickets	int
a_code	varchar(30)
price_per_ticket	float
brand	varchar(30)

codes School Ranking Analysis code Patient Diagnosis Report code\_1 Employee Data Analysis code Payroll Calculation code

Limit to 1000 rows

```

41  });
42
43  • select * from passengers_on_flights
44    where route_id <=25;
45
46  • Select count(*) as no_of_passengers, sum(Price_per_ticket) as total_revenue from ticket_details
47    where class_id = 'bussiness';
48
49

```

Result Grid

	no_of_passengers	total_revenue
▶ 13		6034

Export: | Wrap Cell Content: |

Result 3 x

Output

Action Output

#	Time	Action	Message
❌ 1	21:39:07	Select distinct(customer_id) as no_of_passengers, sum(Price_per_ticket) as total_revenue from ticket_details w...	Error Code: 1140. In aggre
✅ 2	21:43:14	Select count(*) as no_of_passengers, sum(Price_per_ticket) as total_revenue from ticket_details where class_j...	1 row(s) returned

- Write a query to display the full name of the customer by extracting the first name and last name from the customer table.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

- aircargo
  - Tables
    - customer
    - passengers\_on\_flights
    - routes
    - ticket\_details
  - Views
  - Stored Procedures
  - Functions
- corporate
- hospital
- payroll
- sample
- school
- sqltech\_emp
  - Tables
  - Views
  - Stored Procedures
  - Functions
- sql\_basics
- sys

Information: Table: customer

Columns:

- customer\_id int PK
- first\_name varchar(30)
- last\_name varchar(30)
- date\_of\_birth date
- gender char(1)

Object Info Session

Query:

```

41 );
42
43 • select * from passengers_on_flights
44   where route_id <=25;
45
46 • Select count(*) as no_of_passengers, sum(Price_per_ticket) as total_revenue from ticket_details
47   where class_id = 'bussiness';
48
49 • select concat(first_name, ' ',last_name) as full_name from customer;
  
```

Result Grid

full_name
Julie Sam
Steve Ryan
Morris Lois
Cathenna Emily
Aaron Kim
Alexander Scot
Anderson Stewart
Floyd Ted
Leo Travis

Output

#	Time	Action	Message
5	21:51:42	SHOW SESSION VARIABLES LIKE 'lower_case_table_names'	OK
6	21:51:42	SHOW COLUMNS FROM 'aircargo'.customer	OK
7	21:51:48	TRUNCATE TABLE 'aircargo'.customer	OK
8	21:51:48	PREPARE stmt FROM 'INSERT INTO 'aircargo'.customer ('customer_id','first_name','last_name','date_of_...	OK
9	21:51:48	DEALLOCATE PREPARE stmt	OK
10	21:55:27	select concat(first_name,' 'last_name) as full_name from customer LIMIT 0, 1000	50 row(s) returned

6. Write a query to extract the customers who have registered and booked a ticket. Use data from the customer and ticket\_details tables.

The result tables has 17 null values in vlookup function.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

- aircargo
  - Tables
    - customer
    - passengers\_on\_flights
    - routes
    - ticket\_details
  - Views
  - Stored Procedures
  - Functions
- corporate
- hospital
- payroll
- sample
- school
- sqltech\_emp
  - Tables
  - Views
  - Stored Procedures
  - Functions
- sql\_basics
- sys

Information: Schema: aircargo

Query:

```

45
46 • Select count(*) as no_of_passengers, sum(Price_per_ticket) as total_revenue from ticket_details
47   where class_id = 'bussiness';
48
49 • select concat(first_name, ' ',last_name) as full_name from customer;
50
51 • select distinct c.customer_id, c.first_name, c.last_name, t.customer_id_to_lookup from customer as c
52   LEFT JOIN ticket_details as t on
53     c.customer_id = t.customer_id;
  
```

Result Grid

customer_id	first_name	last_name	to_lookup
1	Julie	Sam	1
2	Steve	Ryan	2
3	Morris	Lois	3
4	Cathenna	Emily	4
5	Aaron	Kim	5
6	Alexander	Scot	6
7	Anderson	Stewart	7
8	Floyd	Ted	8
9	Leo	Travis	9

Output

#	Time	Action	Message
1	03:47:19	select distinct c.customer_id, c.first_name, c.last_name, t.customer_id_to_lookup from customer as c LEFT JOIN ticket_details as t on c.customer_id = t.customer_id	50 row(s) returned

- Write a query to identify the customer's first name and last name based on their customer ID and brand (Emirates) from the ticket\_details table.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'aircargo' database schema with tables like 'customer', 'passengers\_on\_flights', 'routes', and 'ticket\_details'. The main editor contains the following SQL query:

```

51 select distinct c.customer_id, c.first_name, c.last_name, t.customer_id to_lookup from customer as c
52 LEFT JOIN ticket_details as t on
53 c.customer_id = t.customer_id;
54
55 select c.first_name, c.last_name from customer as c
56 inner join ticket_details as t on
57 c.customer_id = t.customer_id
58 where t.brand = 'Emirates';
59

```

The 'Result Grid' shows the output of the second query, displaying the first and last names of customers who traveled with Emirates:

first_name	last_name
Cherly	Vernon
Cathenna	Emily
Anderson	Stewart
Leo	Travis
Roger	Walson
Moss	Morris
Gloria	Richie
Moss	Morris
Carol	Vernon

The 'Action Output' pane at the bottom shows the execution results:

#	Time	Action	Message
1	03:47:19	select distinct c.customer_id, c.first_name, c.last_name, t.customer_id to_lookup from customer as c LEFT JOIN ticket_details as t on c.customer_id = t.customer_id;	50 row(s) returned
2	03:59:44	select c.first_name, c.last_name from customer as c inner join ticket_details as t on c.customer_id = t.customer_id where t.brand = 'Emirates';	18 row(s) returned

- Write a query to identify the customers who have travelled by *Economy Plus* class using Group By and Having clause on the passengers\_on\_flights table.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'aircargo' database schema. The main editor contains the following SQL query:

```

55 select c.first_name, c.last_name from customer as c
56 inner join ticket_details as t on
57 c.customer_id = t.customer_id
58 where t.brand = 'Emirates';
59
60 select customer_id from passengers_on_flights
61 where class_id = 'Economy Plus'
62 group by customer_id having count(*)>0;
63

```

The 'Result Grid' shows the output of the second query, displaying the customer IDs of passengers who traveled in the 'Economy Plus' class:

customer_id
1
8
11
17
19
22
32
47
50

The 'Action Output' pane at the bottom shows the execution results:

#	Time	Action	Message
1	14:20:53	select customer_id from passengers_on_flights where class_id = 'Economy Plus' group by customer_id having count(*)>0;	9 row(s) returned

- Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket\_details table.



MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

aircargo

- customer
- passengers\_on\_flights
- routes
- ticket\_details

Views

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

sciQtech\_emp

sql\_basics

sys

Administration Schemas

Information

No object selected

Query Editor

```

60 • select customer_id from passengers_on_flights
61 • where class_id = 'Economy Plus'
62 • group by customer_id having count(*)>0;
63
64 • select
65 • if (sum(Price_per_ticket) >10000, 'yes','no')as revenue_crossed_10000
66 • from ticket_details;
67
68

```

Result Grid

revenue_crossed_10000
yes

Result 3 x

Output

Action Output

#	Time	Action	Message
1	14:20:53	select customer_id from passengers_on_flights where class_id = 'Economy Plus' group by customer_id having c...	9 row(s) returned
2	14:27:55	select if (sum(Price_per_ticket) >10000, 'yes','no')as revenue_crossed_10000 from ticket_details LIMIT 0, 1000	1 row(s) returned

10. Write a query to create and grant access to a new user to perform operations on a database.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

aircargo

- customer
- passengers\_on\_flights
- routes
- ticket\_details

Views

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

sciQtech\_emp

sql\_basics

sys

Administration Schemas

Information

Table: customer

Columns:

Column	Type	PK
customer_id	int	PK
first_name	varchar(30)	
last_name	varchar(30)	
date_of_birth	date	
gender	char(1)	

Query Editor

```

183
184
185
186
187 • CREATE USER 'saina'@'localhost' IDENTIFIED BY 'AIR_1990';
188 • GRANT ALL PRIVILEGES ON aircargo TO 'saina'@'localhost';
189 • FLUSH PRIVILEGES;
190
191
192
193
194
195
196

```

Output

Action Output

#	Time	Action	Message
1	20:59:34	CREATE USER 'saina'@'localhost' IDENTIFIED BY 'AIR_1990'	0 row(s) affected
2	20:59:34	GRANT ALL PRIVILEGES ON aircargo TO 'saina'@'localhost'	0 row(s) affected
3	20:59:34	FLUSH PRIVILEGES	0 row(s) affected

11. Write a query to find the maximum ticket price for each class using window functions on the ticket\_details table.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

ticket\_details

Views

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

sciotech\_emp

sql\_basics

sys

Administration Schemas

Information

Table: ticket\_details

Columns:

p\_date date

customer\_id int

aircraft\_id varchar(30)

class\_id varchar(30)

no\_of\_tickets int

a\_code varchar(30)

price\_per\_ticket float

brand varchar(30)

```

63
64 • select
65 if (sum(Priceschool_per_ticket) > 10000, 'yes', 'no') as revenue_crossed_10000
66 from ticket_details;
67
68
69 • select customer_id, aircraft_id, class_id, brand, MAX(Price_per_ticket) over (partition by class_id)
70 max_ticket_price from ticket_details;
71

```

Result Grid

	customer_id	aircraft_id	class_id	brand	max_ticket_price
25	767-301ER	Business	Emirates	\$10	
49	767-301ER	Business	Emirates	\$10	
21	CRJ900	Business	British Airways	\$10	
33	CRJ900	Business	British Airways	\$10	
29	ERJ142	Business	Jet Airways	\$10	
7	767-301ER	Business	Emirates	\$10	
24	A321	Business	Qatar Airways	\$10	
15	A321	Business	Qatar Airways	\$10	
2	A321	Business	Qatar Airways	\$10	

Result 1 x

Output

Action Output

#	Time	Action	Message
1	10:28:53	select customer_id, aircraft_id, class_id, brand, MAX(Price_per_ticket) over (partition by class_id) max_ticket_p...	50 row(s) returned

12. Write a query to extract the passengers whose route ID is 4 by improving the speed and performance of the passengers\_on\_flights table.

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

ticket\_details

Views

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

sciotech\_emp

sql\_basics

sys

Administration Schemas

Information

Table: passengers\_on\_flights

Columns:

aircraft\_id varchar(30)

route\_id int

customer\_id varchar(30)

depart varchar(30)

arrival varchar(30)

seat\_num varchar(30)

class\_id varchar(30)

travel\_date date

flight\_num int

```

69 • select customer_id, aircraft_id, class_id, brand, MAX(Price_per_ticket) over (partition by class_id)
70 max_ticket_price from ticket_details;
71
72 • CREATE INDEX idx_route_id ON passengers_on_flights (route_id);
73
74 • SELECT *
75 FROM passengers_on_flights
76 WHERE route_id = 4;
77

```

Result Grid

	aircraft_id	route_id	customer_id	depart	arrival	seat_num	class_id	travel_date	flight_num
767-301ER	4	2	JFK	LAX	01E	Economy	2018-09-02	1114	
767-301ER	4	4	JFK	LAX	09FC	First Class	2020-04-30	1114	
767-301ER	4	11	JFK	LAX	05B	Business	2020-11-09	1114	

passengers\_on\_flights 2 x

Output

Action Output

#	Time	Action	Message
1	10:28:53	select customer_id, aircraft_id, class_id, brand, MAX(Price_per_ticket) over (partition by class_id) max_ticket_p...	50 row(s) returned
2	10:47:01	CREATE INDEX idx_route_id ON passengers_on_flights (route_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
3	10:47:21	SELECT * FROM passengers_on_flights WHERE route_id = 4 LIMIT 0, 1000	3 row(s) returned

13. For the route ID 4, write a query to view the execution plan of the passengers\_on\_flights table.



MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

ticket\_details

Views

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

scitech\_emp

sql\_basics

sys

Administration Schemas

Information

Table: passengers\_on\_flights

Columns:

aircraft\_id varchar(30)

route\_id int

customer\_id int

depart varchar(30)

arrival varchar(30)

seat\_num varchar(30)

class\_id varchar(30)

travel\_date date

flight\_num int

76 WHERE route\_id = 4;

77

78 EXPLAIN SELECT \*

79 FROM passengers\_on\_flights

80 WHERE route\_id = 4;

81

82

83

84

Result Grid

Filter Rows:

Export: | Wrap Cell Contents: |

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	passengers_on_flights	NULL	ref	idx_route_id	idx_route_id	4	const	3	100.00	NULL

Result 3

Output

Action Output

#	Time	Action	Message
1	10:28:53	select customer_id, aircraft_id, class_id, brand, MAX(Price_per_ticket) over (partition by class_id) max_ticket_p...	50 row(s) returned
2	10:47:01	CREATE INDEX idx_route_id ON passengers_on_flights (route_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
3	10:47:21	SELECT * FROM passengers_on_flights WHERE route_id = 4 LIMIT 0, 1000	3 row(s) returned
4	11:03:02	EXPLAIN SELECT * FROM passengers_on_flights WHERE route_id = 4	1 row(s) returned

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

ticket\_details

Views

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

scitech\_emp

sql\_basics

sys

Administration Schemas

Information

Table: passengers\_on\_flights

Columns:

aircraft\_id varchar(30)

route\_id int

customer\_id int

depart varchar(30)

arrival varchar(30)

seat\_num varchar(30)

class\_id varchar(30)

travel\_date date

flight\_num int

77

78 EXPLAIN SELECT \*

79 FROM passengers\_on\_flights

80 WHERE route\_id = 4;

81

82 EXPLAIN ANALYZE SELECT \*

83 FROM passengers\_on\_flights

84 WHERE route\_id = 4;

85

Result Grid

Filter Rows:

Export: | Wrap Cell Contents: |

EXPLAIN

-> Index lookup on passengers\_on\_flights using idx\_route\_id (route\_id=4) (cost=1.05 rows=3) (actual time=0.0434..0.0661 rows=3 loops=1)

Result 4

Output

Action Output

#	Time	Action	Message
1	10:28:53	select customer_id, aircraft_id, class_id, brand, MAX(Price_per_ticket) over (partition by class_id) max_ticket_p...	50 row(s) returned
2	10:47:01	CREATE INDEX idx_route_id ON passengers_on_flights (route_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
3	10:47:21	SELECT * FROM passengers_on_flights WHERE route_id = 4 LIMIT 0, 1000	3 row(s) returned
4	11:03:02	EXPLAIN SELECT * FROM passengers_on_flights WHERE route_id = 4	1 row(s) returned
5	11:04:15	EXPLAIN ANALYZE SELECT * FROM passengers_on_flights WHERE route_id = 4	1 row(s) returned

14. Write a query to calculate the total price of all tickets booked by a customer across different aircraft IDs using rollup function.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator School Ranking Analysis code Patient Diagnosis Report code\_1 Employee Data Analysis code Payroll Calculation code sciQtech\_emp code"

SCHEMAS

Filter objects

aircargo

- customer
- passengers\_on\_flights
- routes
- ticket\_details

Views

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

sciQtech\_emp

sql\_basics

sys

Administration Schemas

Information

Table: ticket\_details

Columns:

Column Name	Data Type
p_date	date
customer_id	int
aircraft_id	varchar(30)
class_id	varchar(30)
no_of_tickets	int
a_code	varchar(30)
price_per_ticket	float
brand	varchar(30)

```

84 WHERE route_id = 4;
85
86 • select customer_id, aircraft_id, class_id, brand, sum (price_per_ticket) as total_price from ticket_details
87 group by rollup (aircraft_id);
88
89 • SELECT Customer_ID, Aircraft_ID, SUM(price_per_ticket) AS TotalPrice
90 FROM Ticket_details
91 GROUP BY Customer_ID, Aircraft_ID with rollup;
92

```

Result Grid

Customer_ID	Aircraft_ID	TotalPrice
1	CRJ900	320
1	ERJ142	250
1	767-301ER	570
2	767-301ER	130
2	A321	505
2	767-301ER	635
4	767-301ER	780
4	767-301ER	780
5	767-301ER	430

Result 6 x

Output

Action Output

#	Time	Action	Message
1	13:35:10	SELECT Customer_ID, Aircraft_ID...	75 row(s) returned

15. Write a query to create a view with only business class customers along with the brand of airlines.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator School Ranking Analysis code Patient Diagnosis Report code\_1 Employee Data Analysis code Payroll Calculation

SCHEMAS

Filter objects

aircargo

- customer
- passengers\_on\_flights
- routes
- ticket\_details

Columns

Indexes

Foreign Keys

Triggers

Views

- businessclasscustomers

Stored Procedures

Functions

corporate

hospital

payroll

sample

school

sciQtech\_emp

sql\_basics

sys

Administration Schemas

Information

Schema: aircargo

```

87 group by rollup (aircraft_id);
88
89 • SELECT Customer_ID, Aircraft_ID, SUM(price_per_ticket) AS TotalPrice
90 FROM Ticket_details
91 GROUP BY Customer_ID, Aircraft_ID with rollup;
92
93 • create view BusinessClassCustomers as
94 SELECT Customer_ID, Brand
95 from ticket_details
96 WHERE Class_id = 'Bussiness';
97
98
99
100
101
102
103
104
105
106

```

Output

Action Output

#	Time	Action	Message
1	13:38:50	create view BusinessClassCustom...	0 row(s) affected

16. Write a query to create a stored procedure to get the details of all passengers flying between a range of routes defined in run time. Also, return an error message if the table doesn't exist.

The screenshot shows the MySQL Workbench interface with a stored procedure named `GetPassengersByRouteRange` being created. The procedure takes two input parameters: `start_route INT` and `end_route INT`. It begins by declaring a variable `table_exists INT` and selecting the count of rows in the `passenger_table` into `table_exists`. It then checks if `table_exists > 0`. If true, it constructs a dynamic SQL query to select all columns from `passenger_table` where the route is between `start_route` and `end_route`. The query is prepared, executed, and the results are returned. If the table does not exist, it returns an error message: "Error: The passenger\_table does not exist."

```

220
221 DELIMITER $$
222
223 CREATE PROCEDURE GetPassengersByRouteRange(
224     IN start_route INT,
225     IN end_route INT
226 )
227 BEGIN
228     DECLARE table_exists INT;
229     SELECT COUNT(*)
230     INTO table_exists
231     FROM information_schema.tables
232     WHERE table_name = 'passenger_on_flight';
233
234     IF table_exists > 0 THEN
235         SET @sql_query = CONCAT(
236             'SELECT * FROM passenger_table WHERE route BETWEEN ',
237             start_route, ' AND ', end_route
238         );
239
240         PREPARE dynamic_statement FROM @sql_query;
241         EXECUTE dynamic_statement;
242         DEALLOCATE PREPARE dynamic_statement;
243     ELSE
244         SELECT 'Error: The passenger_table does not exist.' AS ErrorMessage;
245     END IF;
246 END;
247 $$
248
249 DELIMITER ;
250
251

```

17. Write a query to create a stored procedure that extracts all the details from the routes table where the travelled distance is more than 2000 miles.

The screenshot shows the MySQL Workbench interface with a stored procedure named `GetRoutesWithDistanceAbove2000` being created. The procedure simply selects all rows from the `routes` table where the `Distance_miles` is greater than 2000. Below the code editor, the results of the procedure are displayed in a table grid.

```

100
101 CREATE PROCEDURE GetRoutesWithDistanceAbove2000()
102 BEGIN
103     SELECT * FROM routes
104     WHERE Distance_miles > 2000;
105 END //
106
107 DELIMITER ;
108 CALL GetRoutesWithDistanceAbove2000();

```

Route_id	Flight_num	Origin_airport	Destination_airport	Aircraft_id	Distance_miles
1	1111	EWB	HNL	767-301ER	4962.00
2	1112	HNL	EWB	767-301ER	4962.00
3	1113	EWB	LHR	A321	3466.00
4	1114	JFK	LAX	767-301ER	2475.00
5	1115	LAX	JFK	767-301ER	2475.00
6	1116	HNL	LAX	767-301ER	2556.00
10	1120	HNL	DEN	A321	3365.00
12	1122	ABT	ADK	767-301ER	4300.00
13	1123	ADK	BQN	A321	2232.00

18. Write a query to create a stored procedure that groups the distance travelled by each flight into three categories. The categories are, short distance travel (SDT) for  $\geq 0$  AND  $\leq 2000$  miles, intermediate distance travel (IDT) for  $>2000$  AND  $\leq 6500$ , and long-distance travel (LDT) for  $>6500$ .

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays a tree view of the database structure, including tables like 'passengers\_on\_flights', 'routes', 'ticket\_details', and 'businessclasscustomers'. The 'Information' pane shows details for the 'routes' table, including columns: 'Route\_id' (int PK), 'Flight\_num' (varchar(30)), 'Origin\_airport' (varchar(30)), 'Destination\_airport' (varchar(30)), 'Aircraft\_id' (varchar(30)), and 'Distance\_miles' (decimal(10,2)).

The main editor displays a SQL script for creating a stored procedure named 'distance\_travelled\_groups()'. The script uses a CASE statement to categorize flights based on distance in miles:

```

110 Delimiter //
111 Create procedure distance_travelled_groups()
112 begin
113     select FLIGHT_NUM, Distance_miles,
114         CASE
115             WHEN Distance_miles >= 0 AND Distance_miles <=2000 THEN 'SHORT_DISTANCE_TRAVELLED(SOT)'
116             WHEN Distance_Miles > 2000 AND Distance_Miles <= 6500 THEN 'Intermediate_Distance_Travel(IDT)'
117             WHEN Distance_Miles > 6500 THEN 'Long-Distance_Travel(LDT)'
118         END as Distance_category
119     from
120     routes;
121 end //
122
123 DELIMITER ;
124
125 CALL DISTANCE_TRAVELLED_GROUPS();

```

The 'Result Grid' shows the output of the procedure call, displaying columns 'FLIGHT\_NUM', 'Distance\_miles', and 'Distance\_category'. The results are as follows:

FLIGHT_NUM	Distance_miles	Distance_category
1111	4962.00	Intermediate_Distance_Tra
1112	4962.00	Intermediate_Distance_Travel(IDT)
1113	3466.00	Intermediate_Distance_Travel(IDT)
1114	2475.00	Intermediate_Distance_Travel(IDT)
1115	2475.00	Intermediate_Distance_Travel(IDT)
1116	2556.00	Intermediate_Distance_Travel(IDT)

The 'Output' pane shows the execution log, indicating that the procedure was created successfully and returned 49 rows.

19. Write a query to extract ticket purchase date, customer ID, class ID and specify if the complimentary services are provided for the specific class using a stored function in stored procedure on the ticket\_details table.

Condition:

- If the class is *Business* and *Economy Plus*, then complimentary services are given as Yes, else it is No

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

ticket\_details

Views

businessclasscustomers

Stored Procedures

distance\_travelled\_groups

extractticketinfo

Functions

getcomplimentaryservices

corporate

hospital

payroll

sample

school

scitech\_emp

sql\_basics

Information

No object selected

Result Grid

Filter Rows:

Exports

Wrap Cell Contents

127 DELIMITER //

128 CREATE FUNCTION getcomplimentaryservices(class\_ID Varchar(30)) RETURNS VARCHAR(3)

129 DETERMINISTIC

130 READS SQL DATA

131 BEGIN

132 DECLARE complimentary VARCHAR(3);

133

134 IF class\_ID IN (1, 2) THEN

135 SET complimentary = 'Yes'; -- Business or Economy Plus

136 ELSE

137 SET complimentary = 'No';

138 END IF;

139

140 RETURN complimentary;

141 END //

142 DELIMITER ;

p_date	customer_id	class_id	complimentary_services
2018-12-26	27	Economy	No
2020-02-02	22	Economy Plus	No
2020-03-03	21	Business	No
2020-04-04	4	First Class	No
2020-05-05	5	Economy	No
2020-07-07	7	Business	No

Result 1 x

Output

Action Output

#	Time	Action	Message
1	14:17:18	CREATE FUNCTION getcomplimentaryservices(class_ID Varchar(30)) RETURNS VARCHAR(3) DETERM...	0 row(s) affected
2	14:17:18	create procedure extractticketinfo() Begin select p_date, customer_id, class_id, getcomplimentaryservices(...	0 row(s) affected
3	14:17:22	call extractticketinfo()	50 row(s) returned

Object Info

Session

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

aircargo

Tables

customer

passengers\_on\_flights

routes

ticket\_details

Views

businessclasscustomers

Stored Procedures

distance\_travelled\_groups

extractticketinfo

Functions

getcomplimentaryservices

corporate

hospital

payroll

sample

school

scitech\_emp

sql\_basics

Information

No object selected

Result Grid

Filter Rows:

Exports

Wrap Cell Contents

143

144 delimiter //

145 create procedure extractticketinfo()

146 Begin

147 select p\_date, customer\_id, class\_id, getcomplimentaryservices(class\_id) as complimentary\_services from ticket\_details;

148 End //

149

150 DELIMITER ;

151

152 call extractticketinfo();

153

154

155

156

157

158

p_date	customer_id	class_id	complimentary_services
2018-12-26	27	Economy	No
2020-02-02	22	Economy Plus	No
2020-03-03	21	Business	No
2020-04-04	4	First Class	No
2020-05-05	5	Economy	No
2020-07-07	7	Business	No

Result 1 x

Output

Action Output

#	Time	Action	Message
1	14:17:18	CREATE FUNCTION getcomplimentaryservices(class_ID Varchar(30)) RETURNS VARCHAR(3) DETERM...	0 row(s) affected
2	14:17:18	create procedure extractticketinfo() Begin select p_date, customer_id, class_id, getcomplimentaryservices(...	0 row(s) affected
3	14:17:22	call extractticketinfo()	50 row(s) returned

Object Info

Session

20. Write a query to extract the first record of the customer whose last name ends with Scott using a cursor from the customer table.



MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

Filter objects

aircargo

- Tables
  - customer
  - passengers\_on\_flights
  - routes
  - ticket\_details
- Views
  - businessclasscustomers
- Stored Procedures
  - distance\_travelled\_groups
  - extractticketinfo
  - GetCustomerWithLastName
  - GetPassengersByRouteRange
- Functions
  - getcomplimentaryservices
- corporate
- hospital
- payroll
- sample
- rebrand

Administration Schemas

Information

Table: customer

Columns:

customer_id	int PK
first_name	varchar(30)
last_name	varchar(30)
date_of_birth	date
gender	char(1)

Object Info Session

```
198 DELIMITER $$
199
200 CREATE PROCEDURE GetCustomerWithLastNameScott()
201 BEGIN
202 DECLARE customer_id INT;
203 DECLARE first_name VARCHAR(50);
204 DECLARE last_name VARCHAR(50);
205 DECLARE done INT DEFAULT 0;
206 DECLARE customer_cursor CURSOR FOR
207 SELECT customer_id, first_name, last_name
208 FROM customer
209 WHERE last_name LIKE 'Scott'
210 LIMIT 1;
211 DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
212 OPEN customer_cursor;
213 SET done = 0;
214 FETCH NEXT FROM customer_cursor INTO customer_id, first_name, last_name;
215 IF done = 0 THEN
216 SELECT CONCAT('Customer ID: ', customer_id, ', First Name: ', first_name, ', Last Name: ', last_name) AS Result;
217 ELSE
218 SELECT 'No customers with a last name ending with "Scott" found.' AS Result;
219 END IF;
220 CLOSE customer_cursor;
221 END $$
222
223 DELIMITER ;
224
```

Output

#	Time	Action	Message
1	22:57:24	CREATE PROCEDURE GetCustomerWithLastNameScott() BEGIN - Declare variables for cursor usage DEC...	0 row(s) affected

Auto disab man curre tog

Context He