

Air Cargo Analysis.

Project 2 | **Gradable** ⓘ

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

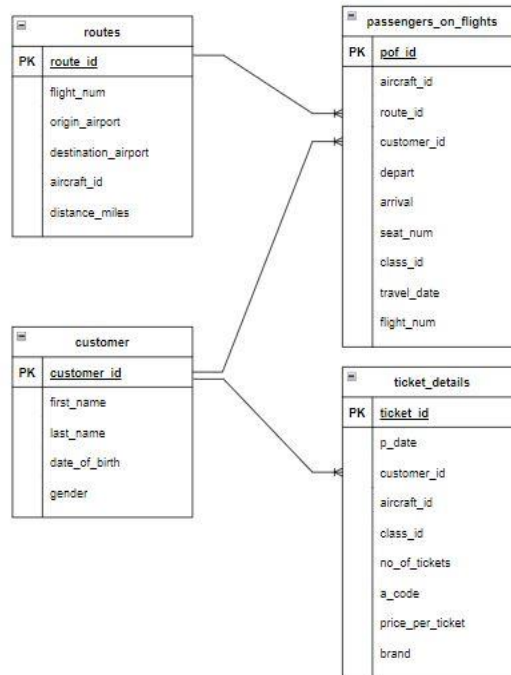
- p_date – Ticket purchase date
- customer_id – ID of the customer
- aircraft_id – ID of each aircraft in a brand
- class_id – ID of travel class
- no_of_tickets – Number of tickets purchased
- a_code – Code of each airport
- price_per_ticket – Price of a ticket
- brand – Aviation service provider for 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: SQL Code and Output Screenshots

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: Query 1 x ScienceQtechEmployeePerform ...

Limit to 1000 rows

```
1 • create database aircargo;
2 • show databases;
```

Result Grid

Database
aircargo
employee
information_schema
mysql
performance_schema
project
sys

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

Filter objects

- employee
 - Tables
 - emp_record_table
 - Views
 - Stored Procedures
 - Functions
- project
 - Tables
 - data_science_team
 - proj_table
 - Views
 - Stored Procedures
 - Functions
- sys

Query 1 x ScienceQtechEmployeePerform...

Limit to 1000 rows

```

3 • use aircargo;
4
5 • create table if not exists customer(
6     customer_id int not null auto_increment primary key,
7     first_name varchar(20) not null,
8     last_name varchar(20) not null,
9     date_of_birth date not null,
10    gender char(1) not null
11 );
12
13 • describe customer;
14

```

Result Grid

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	auto_increment
first_name	varchar(20)	NO		NULL	
last_name	varchar(20)	NO		NULL	
date_of_birth	date	NO		NULL	
gender	char(1)	NO		NULL	

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Schemas

Filter objects

- aircargo
 - Tables
 - customer
 - Views
 - Stored Procedures
 - Functions
- employee
- project
- sys

Query 1 x ScienceQtechEmployeePerform...

Limit to 1000 rows

```

15 • load data local infile 'C:\Program Files\MySQL\MySQL Workbench 8.0\data\datasets\customer.csv'
16 into table customer
17 fields terminated by ',' enclosed by '"' lines terminated by '\n' ignore 1 rows;
18
19 • select * from customer;

```

Result Grid

customer_id	first_name	last_name	date_of_birth	gender
1	Julie	Sam	1989-01-12	F
2	Steve	Ryan	1983-04-03	M
3	Morris	Lois	1993-12-09	M
4	Cathenna	Emily	1977-09-14	F
5	Aaron	Kim	1991-02-18	M
6	Alexander	Scot	1985-02-12	M
7	Anderson	Stewart	1992-01-11	M
8	Floyd	Ted	1993-02-21	M
9	Leo	Travis	1994-03-22	M
10	Melvin	Tracy	1995-04-23	M
11	Roger	Walson	1996-05-24	M
12	Shirley	Wally	1997-06-25	F
13	Solomon	Walter	1998-07-26	M
14	Carol	Vernon	1999-08-27	F
15	Linda	William	1986-09-28	F
16	Christine	Willis	1987-10-06	F
17	Catherine	Shad	1988-11-09	F
18	Gloria	Richie	1989-12-04	F
19	Joyce	Paul	1990-06-02	F
20	Sara	Oliver	1991-01-01	F
21	Chirsty	Josh	2004-01-10	M
22	Pheny	Eri	1999-01-29	M
23	Erwin	Tosh	1994-02-03	M
24	Calvin	Willis	1994-02-15	M
25	Moss	Morris	2011-02-18	M
26	Bryan	Collin	2011-02-28	M
27	Cherly	Vernon	1992-03-19	F
28	Du plesis	Chris	1994-04-17	M

customer 3 x

Output

Administration Schemas

Information

Table: customer

Columns:

- customer_id int AI PK
- first_name varchar(20)
- last_name varchar(20)
- date_of_birth date
- gender char(1)

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

SCHEMAS

- aircargo
 - Tables
 - customer
 - Views
 - Stored Procedures
 - Functions
 - employee
 - project
 - sys

Query 1 x ScienceQtechEmployeePerform...

```

21 • create table if not exists routes(
22     route_id int not null unique primary key,
23     flight_num int constraint chk_1 check (flight_num is not null),
24     origin_airport char(3) not null,
25     destination_airport char(3) not null,
26     aircraft_id varchar(10) not null,
27     distance_miles int not null constraint check_2 check (distance_miles > 0)
28 );
29
30 • describe routes;
  
```

Result Grid

Field	Type	Null	Key	Default	Extra
route_id	int	NO	PRI	NULL	
flight_num	int	YES		NULL	
origin_airport	char(3)	NO		NULL	
destination_airport	char(3)	NO		NULL	
aircraft_id	varchar(10)	NO		NULL	
distance_miles	int	NO		NULL	

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

SCHEMAS

- aircargo
 - Tables
 - customer
 - routes
 - Views
 - Stored Procedures
 - Functions
 - employee
 - project
 - sys

Query 1 x ScienceQtechEmployeePerform...

```

31
32 • load data local infile 'C:\Program Files\MySQL\MySQL Workbench 8.0\data\datasets\routes.csv'
33 into table routes
34 fields terminated by ',' enclosed by '"' lines terminated by '\n' ignore 1 rows;
35
36 • select * from routes;
  
```

Result Grid

	route_id	flight_num	origin_airport	destination_airport	aircraft_id	distance_miles
1	1111	EWB	HNL	767-301ER	4962	
2	1112	HNL	EWB	767-301ER	4962	
3	1113	EWB	LHR	A321	3466	
4	1114	JFK	LAX	767-301ER	2475	
5	1115	LAX	JFK	767-301ER	2475	
6	1116	HNL	LAX	767-301ER	2556	
7	1117	LAX	ORD	A321	1745	
8	1118	ORD	EWB	A321	719	
9	1119	DEN	LAX	ERJ142	862	
10	1120	HNL	DEN	A321	3365	
12	1122	ABI	ADK	767-301ER	4300	
13	1123	ADK	BQN	A321	2232	
14	1124	BQN	CAK	A321	2445	
15	1125	CAK	ANI	767-301ER	2000	
16	1126	ALB	APN	A321	1700	
17	1127	APN	BLV	767-301ER	1900	
18	1128	ANI	BGR	ERJ142	2450	
19	1129	ATW	AVL	A321	2222	
20	1130	AVL	BOI	767-301ER	3134	
21	1131	BFL	BET	A321	2425	
22	1132	BGR	BJI	ERJ142	1242	
23	1133	BLV	BFL	767-301ER	2354	
24	1134	BJI	BQN	A321	1575	
25	1135	RDM	BJI	A321	2425	
26	1136	BET	BTM	ERJ142	1311	
27	1137	BOI	CLD	A321	578	

Table: routes

Columns:

- route_id: int PK
- flight_num: int
- origin_airport: char(3)
- destination_airport: char(3)
- aircraft_id: varchar(10)
- distance_miles: int

routes 5 x

Output

Action Output

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS

Filter objects

aircargo

Tables

customer

routes

Views

Stored Procedures

Functions

employee

project

sys

Query 1 ScienceQtechEmployeePerform...

Limit to 1000 rows

```

37
38 • create table if not exists pof(
39     pof_id int auto_increment primary key,
40     customer_id int not null,
41     aircraft_id varchar(10) not null,
42     route_id int not null,
43     depart char(3) not null,
44     arrival char(3) not null,
45     seat_num char(4) not null,
46     class_id varchar(15) not null,
47     travel_date date not null,
48     flight_num int not null,
49     constraint fk_pof foreign key (customer_id) references customer(customer_id)
50 );
51
52 • describe pof;

```

Result Grid

Field	Type	Null	Key	Default	Extra
pof_id	int	NO	PRI		auto_increment
customer_id	int	NO	MUL		
aircraft_id	varchar(10)	NO			
route_id	int	NO			
depart	char(3)	NO			
arrival	char(3)	NO			
seat_num	char(4)	NO			
class_id	varchar(15)	NO			
travel_date	date	NO			
flight_num	int	NO			

Administration Schemas

Information

Table: routes

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: SCHEMAS

Filter objects

aircargo

Tables

customer

pof

routes

Views

Stored Procedures

Functions

employee

project

sys

Query 1 ScienceQtechEmployeePerform...

Limit to 1000 rows

```

54 • load data local infile 'C:\Program Files\MySQL\MySQL Workbench 8.0\data\datasets\passengers_on_flights.csv'
55 into table routes
56 fields terminated by ',' enclosed by '"' lines terminated by '\n' ignore 1 rows;
57
58 • select * from pof;

```

Result Grid

	pof_id	customer_id	aircraft_id	route_id	depart	arrival	seat_num	class_id	travel_date	flight_num
1	2	A321	34	CRW	COD	01B	Business	2019-01-26	1117	
2	2	767-301ER	4	JFK	LAX	01E	Economy	2018-09-02	1114	
3	1	ERJ142	9	DEN	LAX	01EP	Economy Plus	2019-12-26	1119	
4	1	CRJ900	30	BUR	STT	01FC	First Class	2018-11-04	1140	
5	5	767-301ER	12	ABI	ADK	02B	Business	2018-07-02	1122	
6	5	ERJ142	18	ANI	BGR	02E	Economy	2020-05-06	1128	
7	8	A321	38	CST	DAL	02EP	Economy Plus	2020-08-09	1148	
8	4	767-301ER	5	LAX	JFK	02FC	First Class	2020-04-06	1115	
9	7	767-301ER	20	AVL	BOI	03B	Business	2020-07-08	1130	
10	5	ERJ142	22	BGR	BJI	03E	Economy	2020-05-31	1132	
11	11	ERJ142	31	BTM	CHA	03EP	Economy Plus	2018-08-02	1141	
12	4	767-301ER	4	JFK	LAX	03FC	First Class	2020-04-30	1114	
13	11	767-301ER	5	LAX	JFK	04B	Business	2020-11-12	1115	
14	8	A321	43	CBM	BOI	04E	Economy	2018-05-02	1153	
15	17	A321	13	ABI	ADK	04EP	Economy Plus	2019-06-03	1123	
16	9	767-301ER	15	CAK	ANI	04FC	First Class	2020-09-10	1125	
17	11	767-301ER	4	JFK	LAX	05B	Business	2020-11-09	1114	
18	10	A321	10	HNL	DEN	05E	Economy	2020-10-11	1120	
19	19	CRJ900	47	DAL	LAX	05EP	Economy Plus	2021-01-13	1157	
20	9	CRJ900	33	CDC	CST	05FC	First Class	2018-02-01	1143	
21	15	A321	14	BQN	CAK	06B	Business	2018-11-02	1124	
22	14	ERJ142	35	STT	CDB	06E	Economy	2019-04-02	1145	
23	19	CRJ900	30	BUR	STT	06EP	Economy Plus	2020-12-17	1140	
24	13	A321	13	ADK	BQN	06FC	First Class	2019-01-05	1123	
25	21	CRJ900	45	CCR	EWR	07B	Business	2020-03-07	1155	
26	14	767-301ER	42	CSG	BOS	07E	Economy	2020-01-25	1152	
27	22	ERJ142	22	BGR	BJI	07EP	Economy Plus	2020-02-09	1132	
28	16	CRJ900	39	COD	SCC	07FC	First Class	2019-05-04	1149	

Table: pof

Columns:

- pof_id int AI PK
- customer_id int
- aircraft_id varchar(10)
- route_id int
- depart char(3)
- arrival char(3)
- seat_num char(4)
- class_id varchar(15)
- travel_date date
- flight_num int

Output

Action Output

#	Time	Action
108	13:49:11	SHOW DATABASES

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

aircargo

Tables

customer

pof

routes

Views

Stored Procedures

Functions

employee

project

sys

Query 1 ScienceQtechEmployeePerform...

Limit to 1000 rows

```

59
60 • create table if not exists ticket_details(
61     tkt_id int auto_increment primary key,
62     p_date date not null,
63     customer_id int not null,
64     aircraft_id varchar(10) not null,
65     class_id varchar(15) not null,
66     no_of_tkts int not null,
67     a_code char(3) not null,
68     price_per_tkt decimal(5,2) not null,
69     brand varchar(30) not null,
70     constraint fk_tkt_dts foreign key (customer_id) references customer(customer_id)
71 );
72
73 • describe ticket_details;

```

Result Grid

Field	Type	Null	Key	Default	Extra
tkt_id	int	NO	PRI	NULL	auto_increment
p_date	date	NO		NULL	
customer_id	int	NO	MUL	NULL	
aircraft_id	varchar(10)	NO		NULL	
class_id	varchar(15)	NO		NULL	
no_of_tkts	int	NO		NULL	
a_code	char(3)	NO		NULL	
price_per_tkt	decimal(5,2)	NO		NULL	
brand	varchar(30)	NO		NULL	

Administration Schemas

Information

Table: pof

Columns:

pof_id int AI PK

customer_id int

aircraft_id varchar(10)

route_id int

depart char(3)

arrival char(3)

seat_num char(4)

class_id varchar(15)

travel_date date

flight_num int

Result 8

Output

Action Output

#	Time	Action
110	13:49:12	SHOW COLUMNS FROM 'aircargo'.pof
111	13:49:17	PREPARE stmt FROM 'INSERT INTO 'aircargo'.pof ('customer_id','aircraft_id','route_id','depart','arrival','seat_num','class_id','travel_date','flight_num...'

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

aircargo

Tables

customer

pof

routes

ticket_details

Views

Stored Procedures

Functions

employee

project

sys

Query 1 ScienceQtechEmployeePerform...

Limit to 1000 rows

```

75 • load data local infile 'C:\Program Files\MySQL\MySQL Workbench 8.0\data\datasets\ticket_details.csv'
76 into table ticket_details
77 fields terminated by ',' enclosed by '"' lines terminated by '\n' ignore 1 rows;
78
79 • select * from ticket_details;

```

Result Grid

tkt_id	p_date	customer_id	aircraft_id	class_id	no_of_tkts	a_code	price_per_tkt	brand
1	2018-12-26	27	767-301ER	Economy	1	DAL	130.00	Emirates
2	2020-02-02	22	ERJ142	Economy Plus	1	AGB	220.00	Jet Airways
3	2020-03-03	21	CRJ900	Business	1	BOH	490.00	British Airways
4	2020-04-04	4	767-301ER	First Class	1	AGB	390.00	Emirates
5	2020-05-05	5	ERJ142	Economy	1	CTM	120.00	Jet Airways
6	2020-07-07	7	767-301ER	Business	1	BFS	430.00	Emirates
7	2020-08-08	8	A321	Economy Plus	1	DAL	275.00	Qatar Airways
8	2020-09-09	9	767-301ER	First Class	1	BOH	380.00	Emirates
9	2020-10-10	10	A321	Economy	1	MCO	135.00	Qatar Airways
10	2020-11-11	11	767-301ER	Business	1	AGB	465.00	Emirates
11	2020-12-12	19	CRJ900	Economy Plus	1	DEN	225.00	British Airways
12	2019-01-01	13	A321	First Class	1	YVR	395.00	Qatar Airways
13	2019-02-02	14	ERJ142	Economy	1	CTM	120.00	Jet Airways
14	2019-03-03	25	767-301ER	Business	1	BHX	499.00	Emirates
15	2019-04-04	16	CRJ900	First Class	1	YVR	395.00	British Airways
16	2019-05-05	17	A321	Economy Plus	1	BFS	250.00	Qatar Airways
17	2019-06-06	18	767-301ER	Economy	1	YVR	190.00	Emirates
18	2019-07-07	24	A321	Business	1	CTM	480.00	Qatar Airways
19	2019-08-08	20	CRJ900	First Class	1	MCO	365.00	British Airways
20	2019-09-21	25	767-301ER	Economy	1	BOH	150.00	Emirates
21	2019-10-22	29	A321	Business	1	PEK	410.00	Qatar Airways
22	2019-11-23	1	ERJ142	Economy Plus	1	BFS	250.00	Jet Airways
23	2019-12-24	14	767-301ER	Economy	1	BHX	170.00	Emirates
24	2019-01-25	2	A321	Business	1	YVR	505.00	Qatar Airways
25	2018-01-01	9	CRJ900	First Class	1	AGB	390.00	British Airways
26	2018-02-01	19	767-301ER	Economy	1	AGB	100.00	Emirates
27	2018-03-01	18	767-301ER	First Class	1	BFS	375.00	Emirates
28	2018-04-01	29	FR 1142	Business	1	FMM	510.00	Jet Airways

ticket_details 9

Output

Action Output

#	Time	Action
118	13:56:01	SHOW SESSION VARIABLES LIKE 'lower_case_table_names'

Administration Schemas

Information

Table: ticket_details

Columns:

tkt_id int AI PK

p_date date

customer_id int

aircraft_id varchar(10)

class_id varchar(15)

no_of_tkts int

a_code char(3)

price_per_tkt decimal(5,2)

brand varchar(30)

- 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.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'aircargo' database schema with tables 'customer', 'pof', 'routes', and 'ticket_details'. The main editor shows a query window with the following SQL code:

```

77 fields terminated by ',' enclosed by '"' lines terminated by '\n' ignore 1 rows;
78
79 select * from ticket_details;
80
81 select * from customer where customer_id in (select distinct customer_id from pof where route_id between 1 and 25) order by customer_id;

```

The 'Result Grid' displays the following data:

customer_id	first_name	last_name	date_of_birth	gender
1	Julie	Sam	1989-01-12	F
2	Steve	Ryan	1983-04-03	M
4	Catherina	Emily	1977-09-14	F
5	Aaron	Kim	1991-02-18	M
7	Anderson	Stewart	1992-01-11	M
9	Leo	Travis	1994-03-22	M
10	Melvin	Tracy	1995-04-23	M
11	Roger	Walson	1996-05-24	M
13	Solomon	Walter	1998-07-26	M
15	Linda	William	1986-09-28	F
17	Catherine	Shad	1988-11-09	F
18	Gloria	Richie	1989-12-04	F
22	Pheny	Eri	1999-01-29	M
24	Calvin	Willis	1994-02-15	M
25	Moss	Morris	2011-02-18	M
29	Watson	Ronald	1991-01-11	M
31	James	Robert	1994-04-12	M
44	Bily	Brian	2002-10-26	M
46	Louis	Douglas	1997-09-22	M
49	Russell	Peter	1996-06-01	M
50	Rose	Arthur	1996-05-23	F
NULL	NULL	NULL	NULL	NULL

- Write a query to identify the number of passengers and total revenue in business class from the ticket_details table.

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

```

79 select * from ticket_details;
80
81 select * from customer where customer_id in (select distinct customer_id from pof where route_id between 1 and 25) order by customer_id;
82
83 select count(distinct customer_id) as num_passengers, sum(no_of_tkts * price_per_tkt) as total_revenue from ticket_details where class_id = 'Business';

```

The 'Result Grid' displays the following data:

num_passengers	total_revenue
11	6034.00

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

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

```

81 select * from customer where customer_id in (select distinct customer_id from pof where route_id between 1 and 25) order by customer_id;
82
83 select count(distinct customer_id) as num_passengers, sum(no_of_tkts * price_per_tkt) as total_revenue from ticket_details where class_id = 'Business';
84
85 select concat(first_name, ' ', last_name) as full_name from customer;

```

The 'Result Grid' displays the following data:

full_name
Julie Sam
Steve Ryan
Morris Lee
Catherina Emily
Aaron Kim
Alexander Scott
Anderson Ste...
Floyd Ted
Leo Travis
Melvin Tracy
Roger Walson
Shirley Wally
Solomon Walter
Carol Vernon
Linda William
Christine Willis
Catherine Shad
Gloria Richie
Joyce Paul
Sara Oliver
Christy Josh
Pheny Eri
Erwin Tosh
Calvin Willis
Moss Morris
Bryan Collin
Cherly Vernon
Du plessis Chris

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

MySQL Workbench interface showing a query in the Query Editor:

```

83 • select count(distinct customer_id) as num_passengers, sum(no_of_tkts * price_per_tkt) as total_revenue from ticket_details where class_id = 'Business';
84
85 • select concat(first_name, " ", last_name) as full_name from customer;
86
87 • select first_name, last_name from customer where customer_id in (select distinct b.customer_id from customer a, ticket_details b);

```

The Result Grid shows the following data:

first_name	last_name
Julie	Sam
Steve	Ryan
Cathenna	Emily
Aaron	Kim
Anderson	Stewart
Floyd	Ted
Leo	Travis
Nelvin	Tracy
Roger	Walson
Solomon	Walter
Carol	Vernon
Linda	William
Christine	Willis
Catherine	Shad
Gloria	Richie
Joyce	Paul
Sara	Oliver
Christy	Josh
Pheny	Eri
Calvin	Willis
Moss	Morris
Cherly	Vernon
Du plessis	Chris
Watson	Ronald
James	Robert
Christopher	Sean
Mark	Ethan
Kyle	Mark

7. 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.

MySQL Workbench interface showing a query in the Query Editor:

```

86
87 • select first_name, last_name from customer where customer_id in (select distinct b.customer_id from customer a, ticket_details b);
88
89 • select first_name, last_name from customer where customer_id in (select distinct customer_id from ticket_details where brand = 'Emirates');
90

```

The Result Grid shows the following data:

first_name	last_name
Cherly	Vernon
Cathenna	Emily
Anderson	Stewart
Leo	Travis
Roger	Walson
Moss	Morris
Gloria	Richie
Carol	Vernon
Joyce	Paul
Aaron	Kim
Steve	Ryan
James	Robert
Russell	Peter
Bily	Brian

8. 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.

MySQL Workbench interface showing a query in the Query Editor:

```

90
91 • select class_id, count(distinct customer_id) as num_passengers from pof group by class_id having class_id = 'Economy Plus';
92
93 • select * from customer a
94   inner join (select distinct customer_id from pof where class_id = 'Economy Plus') b
95   on a.customer_id = b.customer_id;
96

```

The Result Grid shows the following data:

customer_id	first_name	last_name	date_of_birth	gender	customer_id
1	Julie	Sam	1989-01-12	F	1
8	Floyd	Ted	1993-02-21	M	8
11	Roger	Walson	1996-05-24	M	11
17	Catherine	Shad	1988-11-09	F	17
19	Joyce	Paul	1990-06-02	F	19
22	Pheny	Eri	1999-01-29	M	22
32	Christopher	Sean	1993-06-21	M	32
47	Sophia	Carl	1999-08-11	F	47
50	Rose	Arthur	1996-05-23	F	50

9. Write a query to identify whether the revenue has crossed 10000 using the IF clause on the ticket_details table.

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```

94 inner join (select distinct customer_id from pof where class_id = 'Economy Plus') b
95 on a.customer_id = b.customer_id;
96
97 • select if((select sum(no_of_tkts * price_per_tkt) as total_revenue from ticket_details) > 10000, 'Crossed 10K', 'Not Crossed 10K') as revenue_check;
98

```

The result grid shows the output of the query:

revenue_check
Crossed 10K

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

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```

99 • create user if not exists 'gayuram'@'127.0.0.1' identified by 'password123';
100 • grant all privileges on aircargo to gayuram@127.0.0.1;
101

```

The output window shows the execution results:

#	Time	Action	Message	Duration / Fetch
178	15:40:44	create user if not exists 'gayuram'@'127.0.0.1' identified by 'password123'	0 row(s) affected	0.047 sec
179	15:41:28	grant all privileges on aircargo to gayuram@127.0.0.1	0 row(s) affected	0.000 sec

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

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```

99 • create user if not exists 'gayuram'@'127.0.0.1' identified by 'password123';
100 • grant all privileges on aircargo to gayuram@127.0.0.1;
101
102 • select class_id, max(price_per_tkt) from ticket_details group by class_id;
103 • select distinct class_id, max(price_per_tkt) over (partition by class_id) as max_price from ticket_details order by max_price;

```

The result grid shows the output of the query:

class_id	max_price
Economy	190.00
Economy Plus	295.00
First Class	395.00
Business	510.00

12. For the route ID 4, write a query to view the execution plan of the passengers_on_flights table.

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```

102 • select class_id, max(price_per_tkt) from ticket_details group by class_id;
103 • select distinct class_id, max(price_per_tkt) over (partition by class_id) as max_price from ticket_details order by max_price;
104
105 • explain select * from pof where route_id = 4;

```

The result grid shows the execution plan for the query:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	pof	HULL	ALL	HULL	HULL			50	10.00	Using where

13. Write a query to extract the passengers whose route ID is 4 by improving the speed and performance of the passengers_on_flights table.

The screenshot shows MySQL Workbench with a query editor containing the following SQL statements:

```
103 • select distinct class_id, max(price_per_tkt) over (partition by class_id) as max_price from ticket_details order by max_price;
104
105 • explain select * from pof where route_id = 4;
106 • create index idx_rid on pof (route_id);
107 • explain select * from pof where route_id = 4;
```

The 'Result Grid' shows the execution plan for the final query (Query 107):

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	pof		ref	idx_rid	idx_rid	4	const	3	100.00	

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

The screenshot shows MySQL Workbench with a query editor containing the following SQL statements:

```
105 • explain select * from pof where route_id = 4;
106 • create index idx_rid on pof (route_id);
107 • explain select * from pof where route_id = 4;
108
109 • select customer_id, aircraft_id, sum(price_per_tkt * no_of_tkts) as total_price from ticket_details group by customer_id, aircraft_id order by customer_id, aircraft_id;
```

The 'Result Grid' shows the execution plan for the final query (Query 109):

customer_id	aircraft_id	total_price
1	CRJ900	320.00
1	ERJ142	250.00
2	767-30 IER	130.00
2	A321	505.00
4	767-30 IER	780.00
5	767-30 IER	430.00
5	ERJ142	240.00
7	767-30 IER	430.00
8	A321	465.00
9	767-30 IER	380.00
9	CRJ900	390.00
10	A321	135.00
11	767-30 IER	930.00
11	ERJ142	295.00
13	A321	395.00
14	767-30 IER	170.00
14	ERJ142	120.00
15	A321	430.00
16	CRJ900	395.00
17	A321	250.00
18	767-30 IER	565.00
19	767-30 IER	100.00
19	CRJ900	450.00
20	CRJ900	680.00
21	CRJ900	490.00
22	ERJ142	220.00
24	A321	480.00
25	767-30 IER	640.00
27	767-30 IER	130.00
28	ERJ142	170.00
29	A321	410.00
29	ERJ142	510.00
31	767-30 IER	130.00
32	ERJ142	770.00

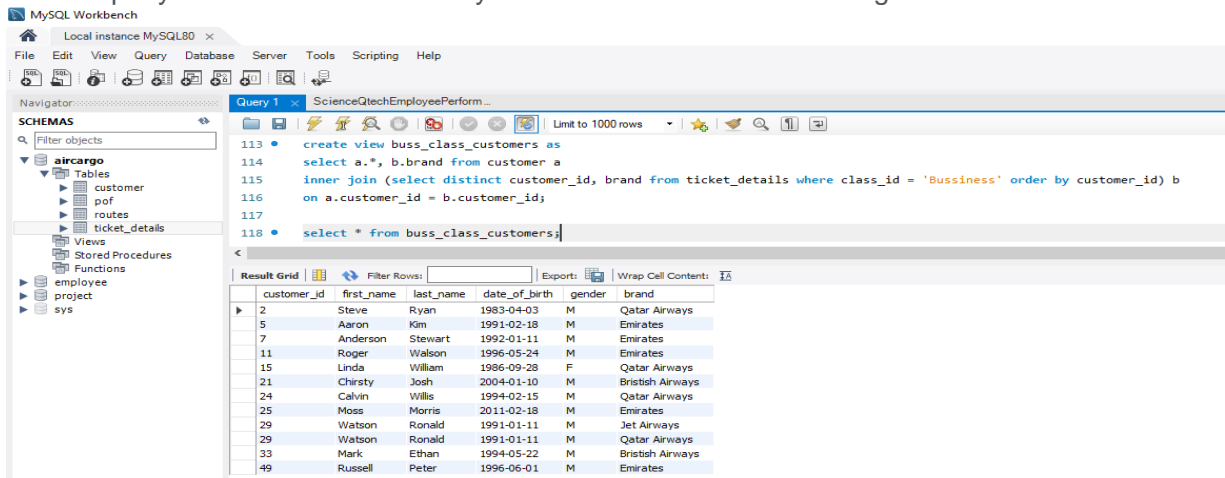
The screenshot shows MySQL Workbench with a query editor containing the following SQL statements:

```
107 • explain select * from pof where route_id = 4;
108
109 • select customer_id, aircraft_id, sum(price_per_tkt * no_of_tkts) as total_price from ticket_details group by customer_id, aircraft_id order by customer_id, aircraft_id;
110
111 • select customer_id, aircraft_id, sum(price_per_tkt * no_of_tkts) as total_price from ticket_details group by customer_id, aircraft_id with rollup order by customer_id, aircraft_id;
```

The 'Result Grid' shows the execution plan for the final query (Query 111):

customer_id	aircraft_id	total_price
1	15369.00	
1	CRJ900	320.00
1	ERJ142	250.00
2	635.00	
2	767-30 IER	130.00
2	A321	505.00
4	780.00	
4	767-30 IER	780.00
5	670.00	
5	767-30 IER	430.00
5	ERJ142	240.00
7	430.00	
7	767-30 IER	430.00
8	465.00	
8	A321	465.00
9	770.00	
9	767-30 IER	380.00
9	CRJ900	390.00
10	135.00	
10	A321	135.00
11	1225.00	
11	767-30 IER	930.00
11	ERJ142	295.00
13	395.00	
13	A321	395.00
14	250.00	
14	767-30 IER	170.00
14	ERJ142	120.00
15	430.00	
15	A321	430.00
16	395.00	
16	CRJ900	395.00
19	440.00	

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



MySQL Workbench interface showing the creation of a view named `buss_class_customers`. The query editor contains the following SQL:

```

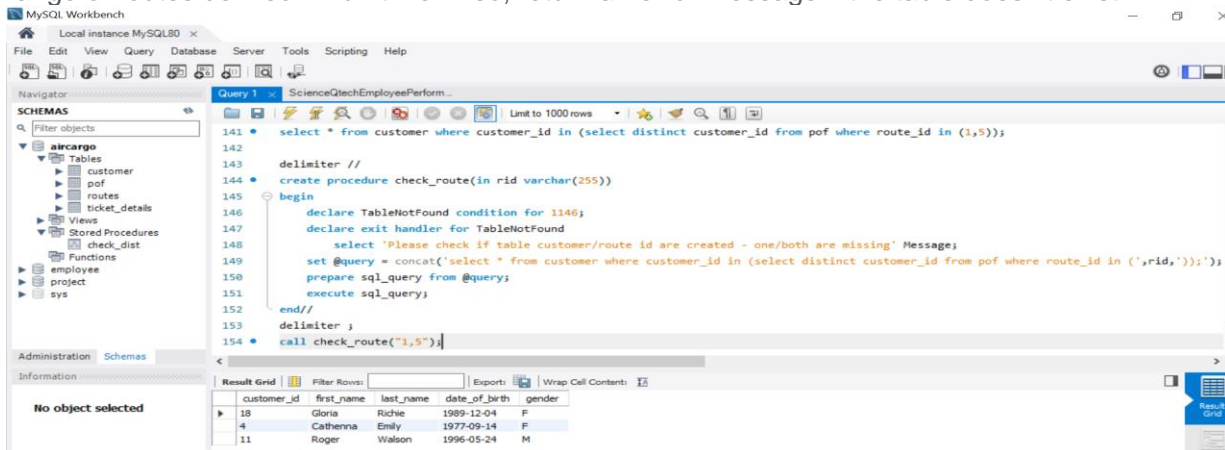
113 • create view buss_class_customers as
114 • select a.*, b.brand from customer a
115 • inner join (select distinct customer_id, brand from ticket_details where class_id = 'Business' order by customer_id) b
116 • on a.customer_id = b.customer_id;
117 •
118 • select * from buss_class_customers;

```

The result grid displays the following data:

customer_id	first_name	last_name	date_of_birth	gender	brand
2	Steve	Ryan	1983-04-03	M	Qatar Airways
5	Aaron	Km	1991-02-18	M	Emirates
7	Anderson	Stewart	1992-01-11	M	Emirates
11	Roger	Walson	1996-05-24	M	Emirates
15	Linda	William	1986-09-28	F	Qatar Airways
21	Christy	Josh	2004-01-10	M	British Airways
24	Calvin	Willis	1994-02-15	M	Qatar Airways
25	Moss	Morris	2011-02-18	M	Emirates
29	Watson	Ronald	1991-01-11	M	Jet Airways
29	Watson	Ronald	1991-01-11	M	Qatar Airways
33	Mark	Ethan	1994-05-22	M	British Airways
49	Russell	Peter	1996-06-01	M	Emirates

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.



MySQL Workbench interface showing the creation of a stored procedure named `check_route`. The query editor contains the following SQL:

```

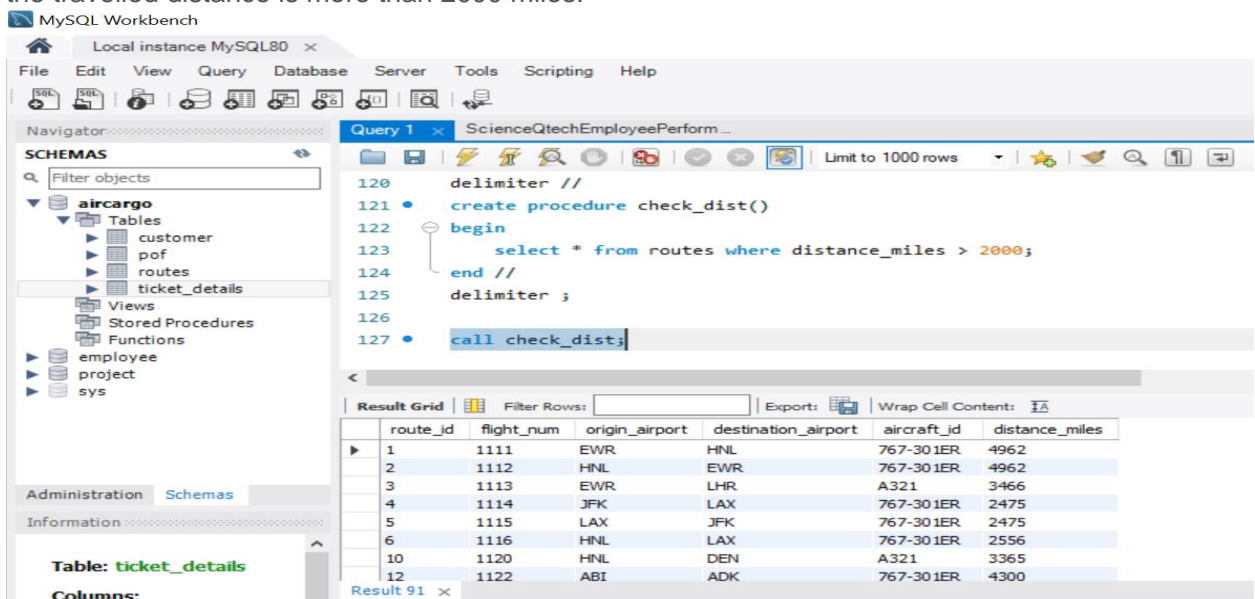
141 • select * from customer where customer_id in (select distinct customer_id from pof where route_id in (1,5));
142 •
143 •
144 • create procedure check_route(in rid varchar(255))
145 • begin
146 • declare TableNotFound condition for 1146;
147 • declare exit handler for TableNotFound
148 • select 'Please check if table customer/route id are created - one/both are missing' Message;
149 • set @query = concat('select * from customer where customer_id in (select distinct customer_id from pof where route_id in (',rid,');');
150 • prepare sql_query from @query;
151 • execute sql_query;
152 • end//
153 • delimiter ;
154 • call check_route("1,5");

```

The result grid displays the following data:

customer_id	first_name	last_name	date_of_birth	gender
18	Gloria	Richie	1989-12-04	F
4	Cathenna	Emily	1977-09-14	F
11	Roger	Walson	1996-05-24	M

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.



MySQL Workbench interface showing the creation of a stored procedure named `check_dist`. The query editor contains the following SQL:

```

120 • delimiter //
121 • create procedure check_dist()
122 • begin
123 • select * from routes where distance_miles > 2000;
124 • end //
125 • delimiter ;
126 •
127 • call check_dist;

```

The result grid displays the following data:

route_id	flight_num	origin_airport	destination_airport	aircraft_id	distance_miles
1	1111	EWR	HNL	767-301ER	4962
2	1112	HNL	EWR	767-301ER	4962
3	1113	EWR	LHR	A321	3466
4	1114	JFK	LAX	767-301ER	2475
5	1115	LAX	JFK	767-301ER	2475
6	1116	HNL	LAX	767-301ER	2556
10	1120	HNL	DEN	A321	3365
12	1122	ABI	ADK	767-301ER	4300

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 ≥ 0 AND ≤ 2000 miles, intermediate distance travel (IDT) for >2000 AND ≤ 6500 , and long-distance travel (LDT) for >6500 .

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: aircargo* x ScienceQtechEmployeePerform...

Limit to 1000 rows

144 select flight_num, distance_miles, case
145 when distance_miles between 0 and 2000 then "SDT"
146 when distance_miles between 2001 and 6500 then "IDT"
147 else "LDT"
148 end distance_category from routes;

Result Grid

flight_num	distance_miles	distance_category
1111	4962	IDT
1112	4962	IDT
1113	3466	IDT
1114	2475	IDT
1115	2475	IDT
1116	2556	IDT
1117	1745	SDT
1118	719	SDT
1119	862	SDT
1120	3365	IDT
1122	4300	IDT
1123	2232	IDT
1124	2445	IDT
1125	2000	SDT
1126	1700	SDT
1127	1900	SDT
1128	2450	IDT
1129	2222	IDT
1130	3134	IDT
1131	2425	IDT

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: aircargo* x ScienceQtechEmployeePerform...

Limit to 1000 rows

138 prepare sql_query from @query;
139 execute sql_query;
140 end//
141 delimiter ;
142 call check_route("1,5");
143
144 select flight_num, distance_miles, case
145 when distance_miles between 0 and 2000 then "SDT"
146 when distance_miles between 2001 and 6500 then "IDT"
147 else "LDT"
148 end distance_category from routes;
149
150 delimiter //
151 create function group_dist(dist int)
152 returns varchar(10)
153 deterministic
154 begin
155 declare dist_cat char(3);
156 if dist between 0 and 2000 then
157 set dist_cat = 'SDT';
158 elseif dist between 2001 and 6500 then
159 set dist_cat = 'IDT';
160 elseif dist > 6500 then
161 set dist_cat = 'LDT';
162 end if;
163 return(dist_cat);
164 end //
165
166 create procedure group_dist_proc()
167 begin
168 select flight_num, distance_miles, group_dist(distance_miles) as distance_category from routes;
169 end //
170 delimiter ;
171
172 call group_dist_proc();

Output

Action Output

#	Time	Action	Message
261	19:40:08	create function group_dist(dist int) returns varchar(10) deterministic begin declare dist_cat char(3); if dist between 0 and 2000 then set dist_cat = 'SDT'; ...	0 row(s) affected
262	19:40:08	create procedure group_dist_proc() begin select flight_num, distance_miles, group_dist(distance_miles) as distance_category from routes; end	0 row(s) affected

Object Info Session

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

aircargo ScienceQtechEmployeePerform...

Limit to 1000 rows

```

165
166 • create procedure group_dist_proc()
167 begin
168     select flight_num, distance_miles, group_dist(distance_miles) as distance_category from routes;
169 end //
170 delimiter ;
171
172 • call group_dist_proc();

```

Result Grid

flight_num	distance_miles	distance_category
1111	4962	IDT
1112	4962	IDT
1113	3466	IDT
1114	2475	IDT
1115	2475	IDT
1116	2556	IDT
1117	1745	SDT
1118	719	SDT
1119	862	SDT
1120	3365	IDT
1122	4300	IDT
1123	2232	IDT
1124	2445	IDT
1125	2000	SDT
1126	1700	SDT
1127	1900	SDT
1128	2450	IDT
1129	2222	IDT
1130	3134	IDT
1131	2425	IDT
1132	1242	SDT
1133	2354	IDT
1134	1575	SDT
1135	2425	IDT
1136	1311	SDT
1137	578	SDT
1138	246	SDT
1139	909	SDT
1140	780	SDT
1141	660	SDT

Result 112

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

File Edit View Query Database Server Tools Scripting Help

Navigator

aircargo ScienceQtechEmployeePerform...

Limit to 1000 rows

```

173
174 • select p_date, customer_id, class_id, case
175     when class_id in ('Business', 'Economy Plus') then "Yes"
176     else "No"
177     end as complimentary_service from ticket_details;

```

Result Grid

p_date	customer_id	class_id	complimentary_service
2018-12-26	27	Economy	No
2020-02-02	22	Economy Plus	Yes
2020-03-03	21	Business	Yes
2020-04-04	4	First Class	No
2020-05-05	5	Economy	No
2020-07-07	7	Business	Yes
2020-08-08	8	Economy Plus	Yes
2020-09-09	9	First Class	No
2020-10-10	10	Economy	No
2020-11-11	11	Business	Yes
2020-12-12	19	Economy Plus	Yes
2019-01-01	13	First Class	No
2019-02-02	14	Economy	No
2019-03-03	25	Business	Yes
2019-04-04	16	First Class	No
2019-05-03	17	Economy Plus	Yes
2019-06-06	18	Economy	No
2019-07-07	24	Business	Yes
2019-08-09	20	First Class	No
2019-09-21	25	Economy	No
2019-10-22	29	Business	Yes
2019-11-23	1	Economy Plus	Yes
2019-12-24	14	Economy	No
2019-01-25	2	Business	Yes
2018-01-01	9	First Class	No
2018-02-01	19	Economy	No
2018-03-01	18	First Class	No
2018-04-01	29	Business	Yes
2018-05-01	8	Economy	No
2018-06-01	20	First Class	No
2018-07-01	5	Business	Yes
2018-08-01	11	Economy Plus	Yes
2018-09-01	2	Economy	No

Result 115

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: aircargo ScienceQtechEmployeePerform...

SCHEMAS

Filter objects

aircargo

- Tables
 - customer
 - pof
 - routes
 - ticket_details
- Views
- Stored Procedures
 - check_dist
 - check_route
- Functions
 - employee
 - project
 - sys

Administration Schemas

Information

No object selected

```

179 delimiter //
180 • create function check_comp_serv(cls varchar(15))
181 returns char(3)
182 deterministic
183 begin
184     declare comp_ser char(3);
185     if cls in ('Business', 'Economy Plus') then
186         set comp_ser = 'Yes';
187     else
188         set comp_ser = 'No';
189     end if;
190     return(comp_ser);
191 end //
192
193 • create procedure check_comp_serv_proc()
194 begin
195     select p_date, customer_id, class_id, check_comp_serv(class_id) as complimentary_service from ticket_details;
196 end //
197 delimiter ;
198
199 • call check_comp_serv_proc();

```

Result Grid

Filter Rows: Export: Wrap Cell Content:

p_date	customer_id	class_id	complimentary_service
2018-12-26	27	Economy	No
2020-02-02	22	Economy Plus	Yes
2020-03-03	21	Business	Yes
2020-04-04	4	First Class	No
2020-05-05	5	Economy	No
2020-07-07	7	Business	Yes
2020-08-08	8	Economy Plus	Yes
2020-09-09	9	First Class	No
2020-10-10	10	Economy	No
2020-11-11	11	Business	Yes
2020-12-12	19	Economy Plus	Yes
2019-01-01	13	First Class	No
2019-02-02	14	Economy	No
2019-03-03	25	Business	Yes

Result 116 x

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 x

File Edit View Query Database Server Tools Scripting Help

Navigator: aircargo ScienceQtechEmployeePerform...

SCHEMAS

Filter objects

aircargo

- Tables
 - customer
 - pof
 - routes
 - ticket_details
- Views
- Stored Procedures
 - check_comp_serv_proc
 - check_dist
 - check_route
 - group_dist_proc
- Functions
 - f() check_comp_serv
 - f() group_dist
- employee
- project
- sys

Administration Schemas

Information

No object selected

```

197 delimiter ;
198
199 • call check_comp_serv_proc();
200
201 • select * from customer where last_name = 'Scott' limit 1;
202

```

Result Grid

Filter Rows: Edit: Export/Import: Wrap Cell Content: Fetch rows:

customer_id	first_name	last_name	date_of_birth	gender
37	Samuel	Scott	2000-01-28	M
NULL	NULL	NULL	NULL	NULL

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

aircargo ScienceQtechEmployeePerform...

Limit to 1000 rows

197
198
199 • call check_comp_serv_proc();
200
201 • select * from customer where last_name = 'Scott' limit 1;
202
203
204 • delimiter //
205
206 • create procedure cust_iname_scott()
207
208 begin
209 declare c_id int;
210 declare f_name varchar(20);
211 declare l_name varchar(20);
212 declare dob date;
213 declare gen char(1);
214
215 declare cust_rec cursor
216 for
217 select * from customer where last_name = 'Scott';
218
219 create table if not exists cursor_table(
220 c_id int ,
221 f_name varchar(20),
222 l_name varchar(20),
223 dob date,
224 gen char(1)
225);
226
227 open cust_rec;
228 fetch cust_rec into c_id, f_name, l_name, dob, gen ;
229 insert into cursor_table(c_id, f_name, l_name, dob, gen) values (c_id, f_name, l_name, dob, gen);
230 close cust_rec;
231
232 select * from cursor_table;
233
234 end//
235
236 delimiter ;
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

Output

Action Output

Time Action Message

287 20.21.35 select * from customer where last_name = 'Scott' limit 1 1 row(s) returned

288 20.22.51 create procedure cust_iname_scott()begin declare c_id int; declare f_name varchar(20); declare l_name varchar(20); declare dob date; decla... 0 row(s) affected

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

aircargo ScienceQtechEmployeePerform...

Limit to 1000 rows

206
207 declare c_id int;
208 declare f_name varchar(20);
209 declare l_name varchar(20);
210 declare dob date;
211 declare gen char(1);
212
213 declare cust_rec cursor
214 for
215 select * from customer where last_name = 'Scott';
216
217 create table if not exists cursor_table(
218 c_id int ,
219 f_name varchar(20),
220 l_name varchar(20),
221 dob date,
222 gen char(1)
223);
224
225 open cust_rec;
226 fetch cust_rec into c_id, f_name, l_name, dob, gen ;
227 insert into cursor_table(c_id, f_name, l_name, dob, gen) values (c_id, f_name, l_name, dob, gen);
228 close cust_rec;
229
230 select * from cursor_table;
231
232 end//
233
234 delimiter ;
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

Result Grid

Filter Rows: Exports Wrap Cell Contents

c_id	f_name	l_name	dob	gen
37	Samuel	Scott	2000-01-28	M

Result 123

Output

Action Output

Time Action Message

269 19.52.47 call check_comp_serv_proc() 50 row(s) returned

Done By,
Gayathri Ramesh